User Manual

Tektronix

VX4469A
ARINC 629 Communication Module

070-9147-01

This document supports firmware version 1.00
and above.

CE

Printed in the U.S.A.

Tektronix, Inc., P.O. Box 1000, Wilsonville, OR 97070–1000

WARRANTY

Tektronix warrants that this product will be free from defects in materials and workmanship for a period of three (3) years from the date of shipment. If any such product proves defective during this warranty period, Tektronix, at its option, either will repair the defective product without charge for parts and labor, or will provide a replacement in exchange for the defective product.

In order to obtain service under this warranty, Customer must notify Tektronix of the defect before the expiration of the warranty period and make suitable arrangements for the performance of service. Customer shall be responsible for packaging and shipping the defective product to the service center designated by Tektronix, with shipping charges prepaid. Tektronix shall pay for the return of the product to Customer if the shipment is to a location within the country in which the Tektronix service center is located. Customer shall be responsible for paying all shipping charges, duties, taxes, and any other charges for products returned to any other locations.

This warranty shall not apply to any defect, failure or damage caused by improper use or improper or inadequate maintenance and care. Tektronix shall not be obligated to furnish service under this warranty a) to repair damage resulting from attempts by personnel other than Tektronix representatives to install, repair or service the product; b) to repair damage resulting from improper use or connection to incompatible equipment; or c) to service a product that has been modified or integrated with other products when the effect of such modification or integration increases the time or difficulty of servicing the product.

THIS WARRANTY IS GIVEN BY TEKTRONIX WITH RESPECT TO THIS PRODUCT IN LIEU OF ANY OTHER WARRANTIES, EXPRESSED OR IMPLIED. TEKTRONIX AND ITS VENDORS DISCLAIM ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. TEKTRONIX' RESPONSIBILITY TO REPAIR OR REPLACE DEFECTIVE PRODUCTS IS THE SOLE AND EXCLUSIVE REMEDY PROVIDED TO THE CUSTOMER FOR BREACH OF THIS WARRANTY. TEKTRONIX AND ITS VENDORS WILL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IRRESPECTIVE OF WHETHER TEKTRONIX OR THE VENDOR HAS ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

EC Declaration of Conformity

We

Tektronix Holland N.V.
Marktweg 73A
8444 AB Heerenveen
The Netherlands

declare under sole responsibility that the

VX4469A and all options

meets the intent of Directive 89/336/EEC for Electromagnetic Compatibility. Compliance was demonstrated to the following specifications as listed in the Official Journal of the European Communities:

EN 55011 Class A Radiated and Conducted Emissions

EN 50081-1 Emissions:

EN 55022 Class B Radiated and Conducted Emissions

EN 50082-1 Immunity:

IEC 801-2 Electrostatic Discharge Immunity
IEC 801-3 RF Electromagnetic Field Immunity
IEC 801-4 Electrical Fast Transient/Burst Immunity
IEC 801-5 Power Line Surge Immunity

To ensure compliance with EMC requirements this module must be installed in a mainframe which has backplane shields installed which comply with Rule B.7.45 of the VXIbus Specification. Only high quality shielded cables having a reliable, continuous outer shield (braid & foil) which has low impedance connections to shielded connector housings at both ends should be connected to this product. In addition, each twisted pair in the cable connected to the S2 port should be individually shielded.

Table of Contents

Appendices

List of Figures

List of Tables

General Safety Summary

Review the following safety precautions to avoid injury and prevent damage to this product or any products connected to it.

Only qualified personnel should perform service procedures.

While using this product, you may need to access other parts of the system. Read the *General Safety Summary* in other system manuals for warnings and cautions related to operating the system.

Injury Precautions

Avoid Electric Overload

To avoid electric shock or fire hazard, do not apply a voltage to a terminal that is outside the range specified for that terminal.

Ground the Product

This product is indirectly grounded through the grounding conductor of the power cord. To avoid electric shock, the grounding conductor must be connected to earth ground. Before making connections to the input or output terminals of the product, ensure that the product is properly grounded.

Do Not Operate Without Covers

To avoid electric shock or fire hazard, do not operate this product with covers or panels removed.

Use Proper Fuse

To avoid fire hazard, use only the fuse type and rating specified for this product.

Do Not Operate in Wet/Damp Conditions

To avoid electric shock, do not operate this product in wet or damp conditions.

Do Not Operate in Explosive Atmosphere

To avoid injury or fire hazard, do not operate this product in an explosive atmosphere.

Product Damage Precautions

Use Proper Fuse To avoid fire hazard, use only the fuse type and rating specified for this product

Use Proper Power Source Do not operate this product from a power source that applies more than the voltage specified.

Provide Proper Ventilation To prevent product overheating, provide proper ventilation.

Do Not Operate With Suspected Failures If you suspect there is damage to this product, have it inspected by qualified service personnel.

Safety Terms and Symbols

Terms in This Manual These terms may appear in this manual:

WARNING. *Warning statements identify conditions or practices that could result in injury or loss of life.*

CAUTION. *Caution statements identify conditions or practices that could result in damage to this product or other property.*

Terms on the Product These terms may appear on the product:

DANGER indicates an injury hazard immediately accessible as you read the marking.

WARNING indicates an injury hazard not immediately accessible as you read the marking.

CAUTION indicates a hazard to property including the product.

Symbols on the Product The following symbols may appear on the product:

DANGER
High Voltage

Protective Ground
(Earth) Terminal

ATTENTION
Refer to
Manual

Double
Insulated

Service Safety Summary

Only qualified personnel should perform service procedures. Read this *Service Safety Summary* and the *General Safety Summary* before performing any service procedures.

Do Not Service Alone

Do not perform internal service or adjustments of this product unless another person capable of rendering first aid and resuscitation is present.

Disconnect Power

To avoid electric shock, disconnect the main power by means of the power cord or, if provided, the power switch.

Use Care When Servicing With Power On

Dangerous voltages or currents may exist in this product. Disconnect power, remove battery (if applicable), and disconnect test leads before removing protective panels, soldering, or replacing components.

To avoid electric shock, do not touch exposed connections.

Preface

This is the user manual for the VX4469A ARINC 629 Communication Module.

Please read and follow all instructions for installation and configuration. Use the Installation Checklist to insure proper installation, and as a record of initial settings.

This manual assumes you are familiar with VXIbus instruments and operation, and with the purpose and function of this instrument. The *Operating Basics* section gives a summary of VXIbus operation, and presents an overview of this instrument's operation.

The *Syntax and Commands* section has a summary of all the commands, and detailed descriptions of each command. You may also wish to make a copy of the *Quick Reference Guide*, located in *Appendix I*, to keep by the instrument.

Conventions

The names of all switches, controls, and indicators appear in this manual exactly as they appear on the instrument.

Specific conventions for programming are given in the sections *Syntax and Commands* and in *Programming Examples*.

Getting Started

Getting Started

This section begins with a brief description of the VX4469A, and then explains how to configure and install the module in a VXIbus mainframe. Then you can choose to perform the quick functional check, also included in this section, to gain confidence that the instrument operates properly.

Product Description

The VX4469A ARINC 629 Communication Module supports from one to three ARINC 629 terminals. The standard board has one terminal. One or two additional terminals are available as options. The VX4469A is designed to transmit and receive data on ARINC 629 buses through a current (transformer) coupling device that interfaces to the bus itself. Data to be transmitted is stored in system memory shared by the terminal IC and an on-board 80186 processor. The transmit schedule and system memory data locations are stored in a Transmit Personality PROM (XPP). Data that is received is stored in the same system memory. A Receive Personality PROM (RPP) and a Multiple Personality PROM (MPP) contain information on which data to receive and where to store it in the shared system memory.

Note that the VX4469A actually uses RAM instead of PROM for storing XPP, RPP, and MPP information. All references to personality PROMs in this manual are actually references to RAM locations.

Each ARINC 629 terminal on the VX4469A Module consists of:

- System memory shared by the VX4469A 80186 controller and each terminal. It is used to store received data and data to be transmitted.
- A personality RAM that contains the information on what data to transmit and receive and the data's location in shared memory.
- A terminal protocol IC which interprets the personality PROM and translates data to/from 16-bit shared memory words and Manchester bit serial encoding.
- A Serial Interface Module (SIM) that modulates/demodulates the Manchester coding and is intended to drive a current coupler on a twisted pair bus.
- An external trigger interface through the front panel and a VXI TTL trigger interface through the backplane.

***NOTE**. To help insure optimum ARINC 629 compatibility, each terminal uses the VLSI Terminal IC and Serial Interface Module (SIM) technology developed by Boeing.*

A pseudo bus module that replaces the SIM and provides a voltage bus without SIMs or current couplers is also available. *Appendix G* gives additional information on the pseudo bus module.

The VX4469A allows you to program the Personality PROMs (RAMs) and to read and write data from/to the shared memory.

The VX4469A also has two test modes. The test modes allow transmitting data with Manchester, parity and timing errors as well as causing collisions on the ARINC 629 bus.

Terminal Programmability

Each terminal is software programmable for transmit interval, terminal gap, sync gap, block/independent mode, and alternate mode. Each terminal may be disabled individually. The three terminals may all be on the same bus or they may be on two or three different busses. Terminals may also be configured to be receive only.

The 32 MHz clock for the processor and terminal ICs may be provided externally to test external systems' sensitivity to clock frequency. There are time stamp clock and time stamp clock reset inputs and outputs to allow synchronizing time on several systems.

ARINC 629

ARINC 629 is a bit-serial, time-multiplexed, multiple-transmitter, data bus developed for use primarily on certain commercial aircraft. All data transmitted on the ARINC 629 bus consists of at least a label, followed by up to 256 16-bit data words.

Data to be transmitted or received is placed in a terminal's shared memory by an on-board 80186 processor. The terminal IC uses a sophisticated built-in data management scheme to read the personality PROM and manage transmission and reception.

The basic unit of data transmission is a wordstring. A wordstring consists of a label and between 0 and 256 data words. A given terminal is allowed to transmit up to 31 wordstrings per transmission.

Protocol Timers

Each terminal determines when to transmit by using three timers and by monitoring bus activity. The first timer, Transmit Interval, determines the minimum amount of time a terminal waits between transmissions.

The second timer, Sync Gap, is used to insure that all terminals can transmit before any terminal re-transmits, and insures that there is a bus quiet time at least Sync Gap long after each terminal has transmitted.

The third timer, Terminal Gap, determines which terminal transmits first if two or more terminals' Transmit Interval timers have elapsed and the bus is busy.

Modes

The VX4469A supports both Periodic and Aperiodic modes. In Periodic mode, the transmit interval time is longer than the time required for all terminals to complete one transmission. In Aperiodic mode, after all terminals have transmitted, none will transmit again until there has been a bus quiet time of Sync Gap. Thereafter the terminals will tend to transmit in Terminal Gap order, shortest to longest.

A bus that is not fully loaded runs in Periodic mode, while a bus with terminals transmitting more data than will fit in the transmit interval will automatically run in Aperiodic mode. A bus will normally switch between Periodic and Aperiodic modes as terminals have more or less data to transmit.

Transmit Schedules

The order in which a terminal transmits data is determined by the contents of the Transmit Personality PROM. The Transmit Personality PROM is a 31 row by 31 column arrangement in RAM and is fully user-programmable.

There are three modes of scheduling: Block, Independent and Alternate. In Block mode, the wordstrings described in a single row are transmitted. Independent mode transmits one wordstring from each column of the array. Alternate mode can only be entered from Block mode. It is similar to Block mode with its first row number defined in the first control cell. The terminal IC can be switched back and forth between Block mode and Alternate mode while the bus is in operation.

Fuses

The VX4469A Module has 5 VDC and ±24 VDC fuses. The fuses protect the module in case of an accidental shorting of the power bus or any other situation where excessive current might be drawn.

If the +5 V fuse opens, the VXIbus Resource Manager will be unable to assert SYSFAIL INHIBIT on this module to disable SYSFAIL*.

If any fuse opens, the fault must be removed before replacing the fuse. Refer to a qualified service person for assistance.

BITE (Built-In Test Equipment)

Built in Test Equipment (BITE) uses regular data transmissions to check SIM and coupler functionality, so as not to corrupt operation of the bus. The SIM constantly monitors the wraparound path, including itself and the selected channel. In addition, a special test function is available which selects and monitors the spare coupler channel, and then reports the results. Visual BITE is

provided for each terminal through a series of LEDs that indicate terminal active, string active, receive error, transmit error, bus busy, and transmit enable. The processor also has an LED indicating that an error has occurred, and another LED is toggled on and off while the processor is idling, giving an indication of how busy the processor is.

Binary Transfer

Refer to *Appendix J: Binary Transfer* for information relating to National Instruments GPIB-VXI/C Slot 0 modules.

Accessories

Table 1–1 lists the standard accessories included with the VX4469A.

Table 1–1: Standard Accessories

Accessory	Part Number
VX4469A User Manual	070-9147-XX

Table 1–2 lists the optional accessories for the VX4469A.

Table 1–2: Optional Accessories

Accessory	Part Number
Adds one additional terminal	VX4469A-01
Adds two additional terminals	VX4469A-02
Pseudo-bus SIM	See Appendix G for information

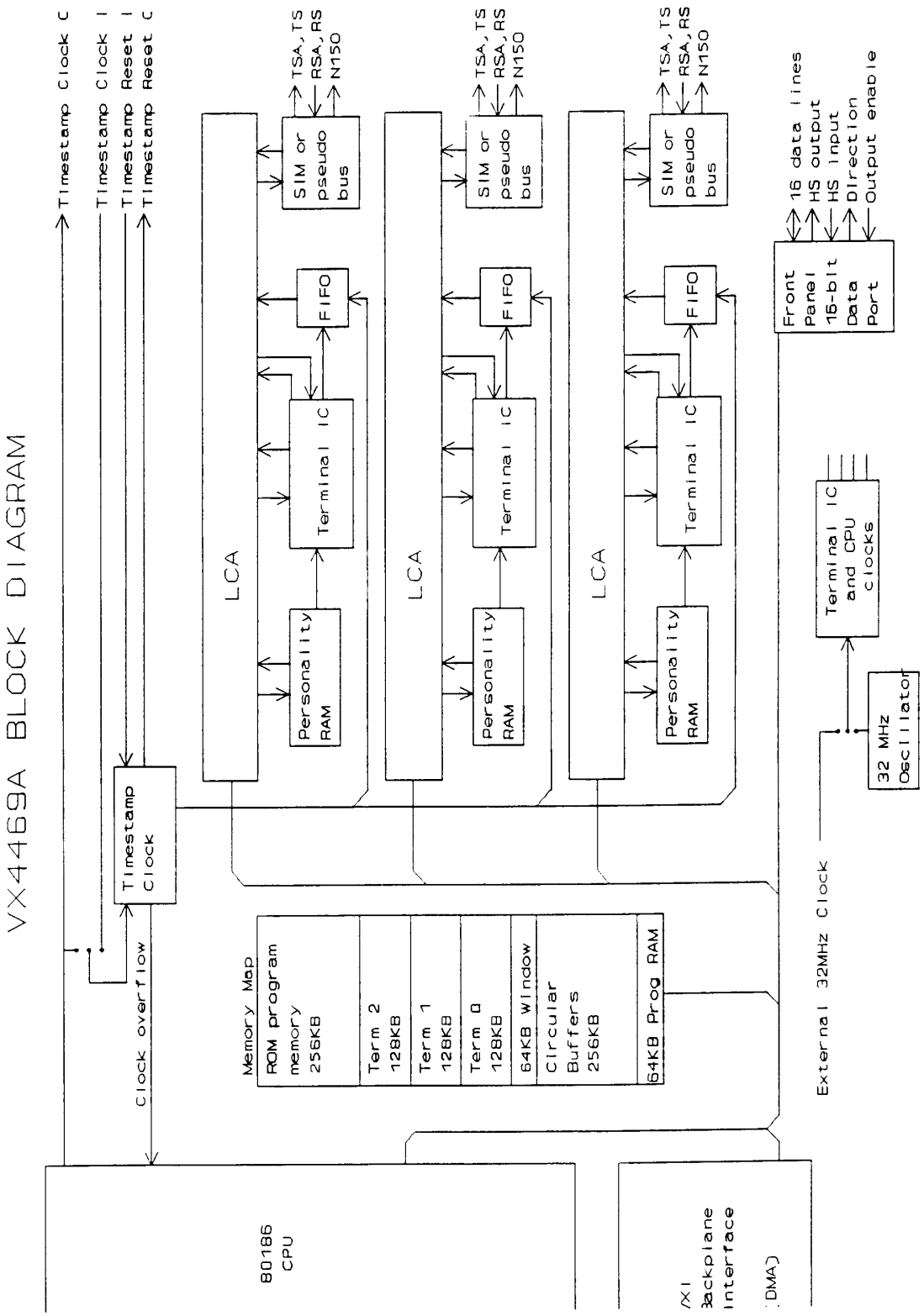


Figure 1–1: System Configuration

Controls And Indicators

The following controls and indicators are provided to select and display the functions of the VX4469A Module's operating environment. See Figures 1–2 and 1–3 for their physical locations.

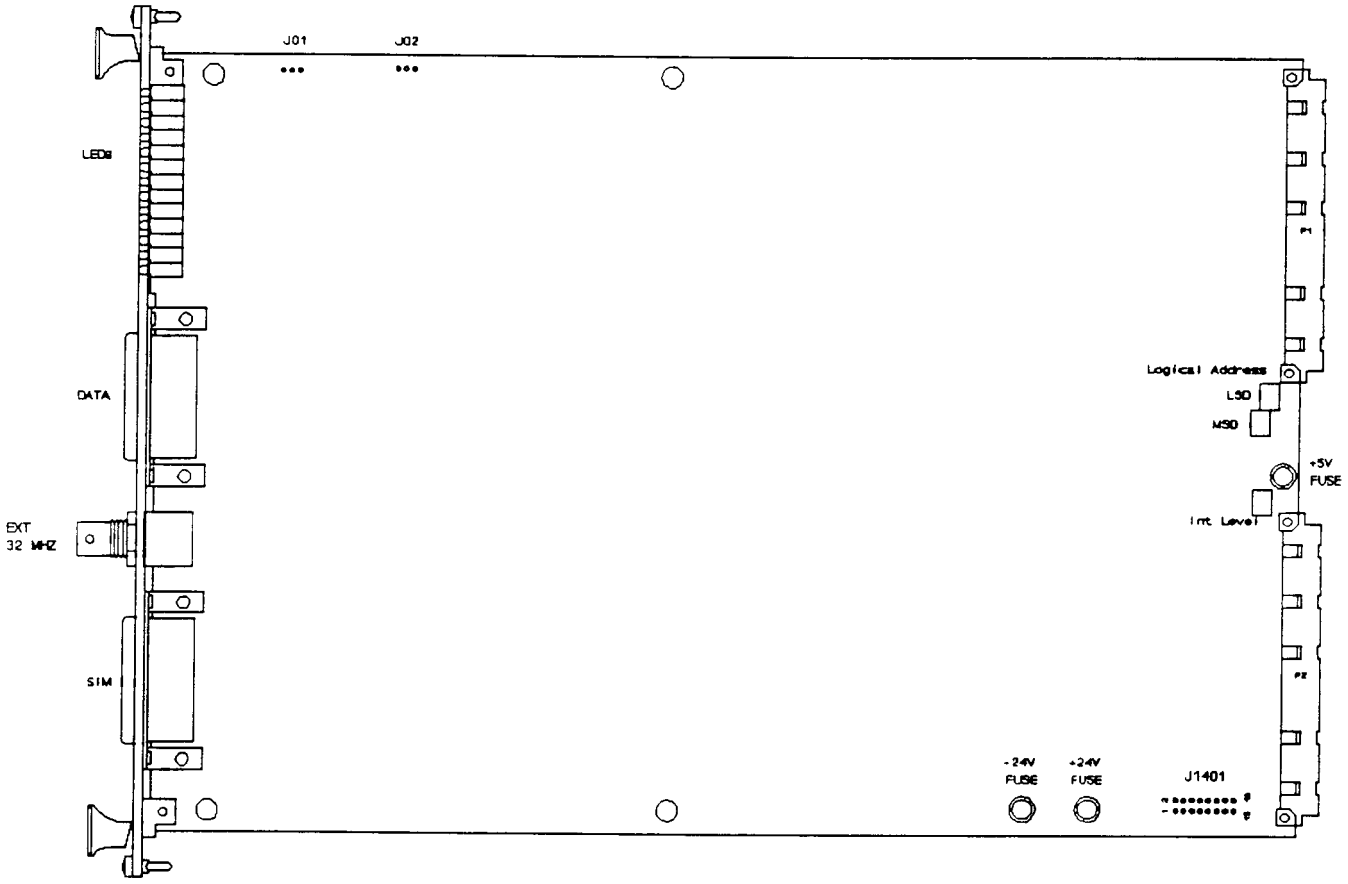


Figure 1–2: VX4469A Controls and Indicators

Switches

The following switches must be correctly set to insure proper operation. See *Configuration* for details of how to set the switches.

Logical Address Switches
VMEbus Interrupt Level Select Switch
Jumpers

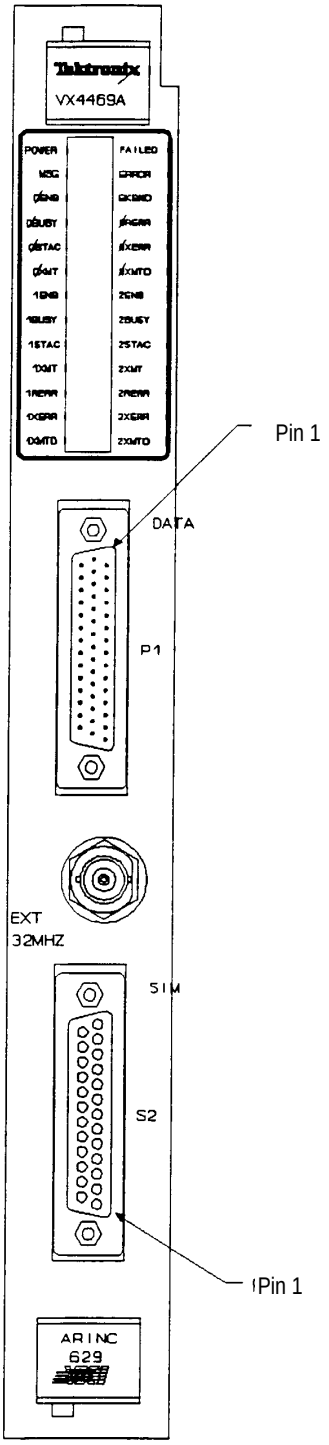


Figure 1–3: VX4469A Front Panel

LEDs

The following LEDs are visible at the top of the VX4469A Module's front panel to indicate the status of the module's operation. See *Status and Events* for a description of each LED's meaning.

Power LED
Failed LED
MSG LED
Error LED
Background LED

Configuration

The following switches must be correctly set to insure proper operation. Refer to Figure 2 for their physical locations.

Logical Address Switches

Each function module in a VXIbus System must be assigned a unique logical address, from 1 to 255 decimal. The base VMEbus address of the VX4469A is set to a value between 1 and FFh (255d) by two **hexadecimal** rotary switches. Align the desired switch position with the arrow on the module shield.

The actual physical address of the VX4469A Module is on a 64 byte boundary. If the switch representing the most significant digit (MSD) of the logical address is set to position X and the switch representing the least significant digit (LSD) of the logical address is set to position Y, then the base physical address of the VX4469A will be [(64d * XYh) + 49152d]. For example:

L.A.	MSD	LSD	Base Physical Address (d)
Ah	0	A	(64 * 10) + 49152 = 49792d
15h	1	5	(64 * 21) + 49152 = 50496d

where:

L.A. = Logical Address
MSD = Most Significant Digit
LSD = Least Significant Digit

IEEE-488 Address

Using the VX4469A Module in an IEEE-488 environment requires knowing the module's IEEE-488 address in order to program it. Different manufacturers of IEEE-488 interface devices may have different algorithms for equating a logical address with an IEEE-488 address. Consult the operating manual of the Resource Manager/IEEE-488 Interface Module being used for additional information.

If the VX4469A is being used in a MATE system, VXIbus logical addresses are converted to IEEE-488 addresses using the algorithm specified in the MATE IAC standard (MATE-STD-IAC). This algorithm is described in detail in the 73A-156 Operating Manual.

VMEbus Interrupt Level Select Switch

Each function module in a VXIbus System can generate an interrupt on the VMEbus to request service from the interrupt handler located on its commander. When using the VX4469A with a Tektronix/CDS commander module, set the interrupt level to the same level as the interrupt handler on that commander. The VMEbus interrupt level on which the VX4469A Module generates interrupts is set by a BCD rotary switch. Align the desired switch position with the arrow on the module shield.

Valid Interrupt Level Select switch settings are 1 through 7, with setting 1 equivalent to level 1, etc. The level chosen should be the same as the level set on the VX4469A's interrupt handler, typically the module's commander. Setting the switch to an invalid interrupt level (0, 8, or 9) will disable the module's interrupts.

Interrupts are used by the module to return VXIbus Protocol Events to the module's commander. Refer to *Operating Basics* for information on interrupts. The VXIbus Protocol Events supported by the module are listed in the *Specifications*.

Jumpers

J01 32 MHz Clock

1	2	3	(factory)	selects internal 32 MHz clock for terminal ICs and processor.
1	2	3		selects external (front panel BNC connector) for 32 MHz clock.

J02 Time Stamp Clock

1	2	3	(factory)	selects internally generated clock for time stamp.
1	2	3		selects external clock (front panel connector S2 pin 4) for time stamp.

Configure VXI TTL Trigger Lines

The VXI TTL trigger lines are not configured at the factory. You may need to jumper the trigger lines on J1401 to the VX4469A trigger inputs and outputs. Refer to Figure 1–2 for the location of J1401.

The eight header pins marked VXI TTL Trigger Line 0 – 7 connect to the VXI backplane VXI TTL Trigger Lines. The other eight header pins connect to the terminal and software trigger inputs and outputs. Terminal and software trigger lines may be connected to any of the backplane VXI TTL trigger lines. Refer to Figure 1–4.

Refer to *Appendix C: Trigger Lines* for additional information regarding VXI TTL Trigger lines.

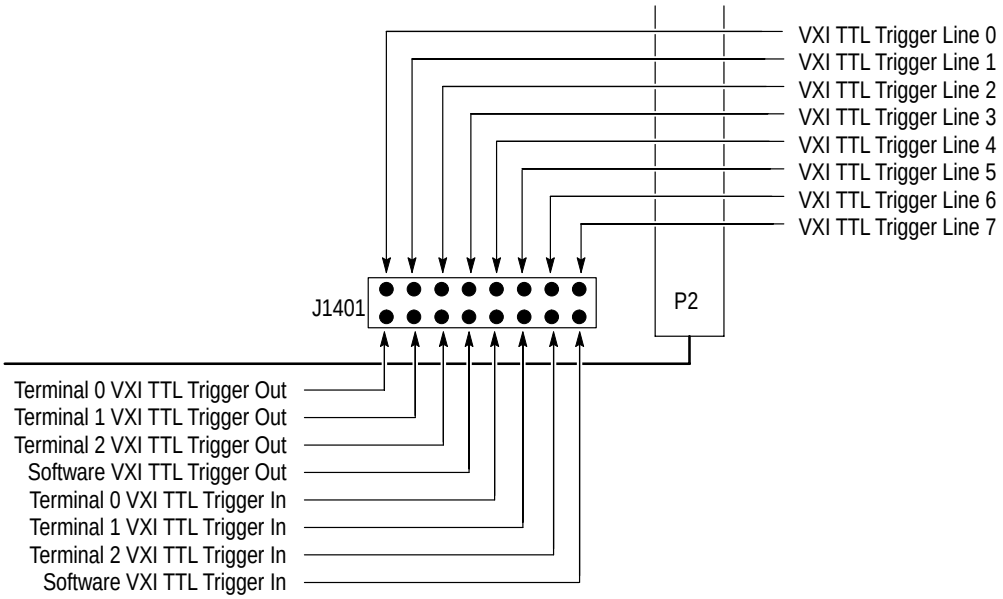


Figure 1–4: Trigger Lines

Installation

This section describes how to install the VX4469A.

Requirements And Cautions

The VX4469A Module is a C size VXIbus instrument module and therefore may be installed in any C or D size VXIbus mainframe slot other than slot 0. If the module is being installed in a D size mainframe, consult the operating manual for the mainframe to determine how to install the module in that particular mainframe. Setting the module's Logical Address switch defines the module's programming address. Refer to *Configuration* for information on selecting and setting the module's logical address.

Tools Required

The following tools are required for proper installation:

Slotted screwdriver set.

CAUTION. *Note that there are two printed ejector handles on the module. To avoid installing the card incorrectly, make sure the ejector marked "VX4469A" is at the top.*

In order to maintain proper mainframe cooling, unused mainframe slots must be covered with the blank front panels supplied with the mainframe.

CAUTION. *Verify that the mainframe is able to provide adequate cooling and power with this module installed. Refer to the mainframe Operating Manual for instructions.*

If the VX4469A is used in a Tektronix/CDS VXIbus Mainframe, all VX4469A cooling requirements will be met.

CAUTION. *If the VX4469A Module is inserted in a slot with any empty slots to the left of the module, the VME daisy-chain jumpers must be installed on the backplane in order for the VX4469A Module to operate properly. Check the manual of the mainframe being used for jumpering instructions.*

Installation Procedure

CAUTION. *The VX4469A Module is a piece of electronic equipment and therefore has some susceptibility to electrostatic damage (ESD). ESD precautions must be taken whenever the module is handled.*

1. Record the revision level, serial number (located on the label on the top shield of the VX4469A), and switch settings on the Installation Checklist.
2. Verify that the switches are switched to the correct values.
3. Make sure power is off in the mainframe.
4. The module can now be inserted into one of the instrument slots of the mainframe.
5. Cable Installation: Use a suitable cable to interface between the module I/O connector and the Unit Under Test (UUT).

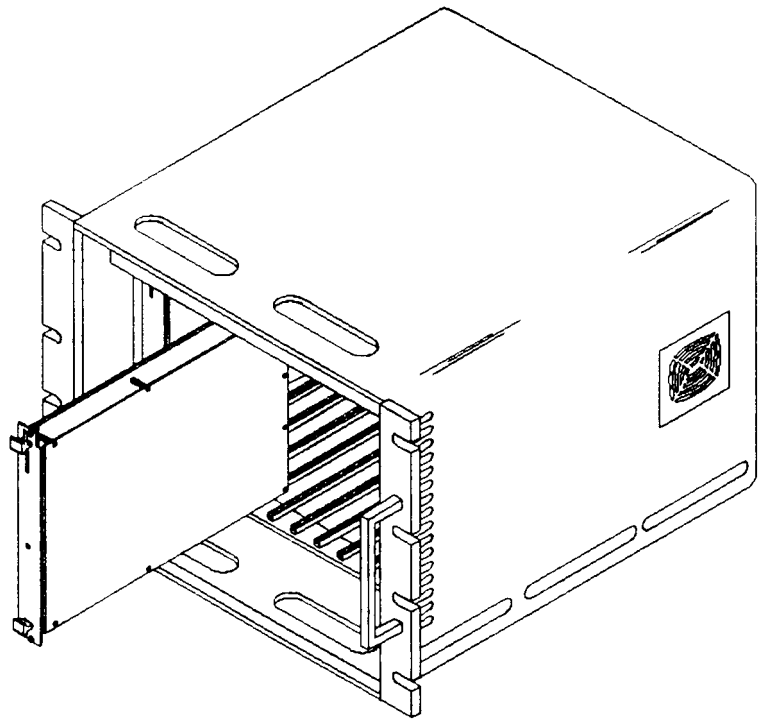


Figure 1–5: Module Installation

Installation Checklist

Installation parameters will vary depending on the mainframe being used. Be sure to consult the mainframe Operating Manual before installing and operating the module.

Revision Level: _____

Serial No.: _____

Mainframe Slot Number: _____

Switch Settings:

VXIbus Logical Address Switch: _____
(FFh enables dynamic configuration.)

Interrupt Level Select Switch: Dynamically programmed by the Resource Manager.

Cables Installed: (if any)

Performed by: _____ Date: _____

Operating Basics

Operating Basics

Functional Check

The VX4469A Module will execute a self test at power-on, or upon direction of a VXIbus hard or soft reset condition, or upon command. A VXIbus hard reset occurs when another device, such as the VXIbus Resource Manager, asserts the backplane line SYSRST*. A VXIbus soft reset occurs when another device, such as the VX4469A's commander, sets the Reset bit in the VX4469A's Control register.

At power-on, as well as during self test, all module outputs remain isolated from the module's front panel connector.

During a power-on, or hard or soft reset, the following actions take place:

1. The SYSFAIL* (VME system-failure) line is set active, indicating that the module is executing a self test, and the Failed LED is lit. In the case of a soft reset, SYSFAIL* is set. However, all Tektronix/CDS commanders will simultaneously set SYSFAIL INHIBIT. This is done to prevent the resource manager from prematurely reporting the failure of a card.

2. If the self test completes successfully, the SYSFAIL* line is released, and the module enters the VXIbus PASSED state (ready for normal operation). SYSFAIL* will be released within three seconds in normal operation.

   If the self test fails, the SYSFAIL* line remains active (or is set active, in the case of a commanded self test or soft reset), and the module makes an internal record of what failure(s) occurred. It then enters the VXIbus FAILED state, which allows an error message to be returned to the module's commander.

The default power-on setup and data is as follows:

current terminal is 0
all terminals are in independent mode (not block mode)
all terminals are in normal mode (not alternate mode)
all terminals are disabled
all terminals' bus requests are enabled
all terminals' transmit intervals (TI) are 0
all terminals' terminal gaps (TG) are 0
all terminals' sync gaps (SG) are 10
all terminals' channel IDs are 0
all terminals' serial interface module receive thresholds are 1000_{10}mv
the data radix is hex
the command parameter radix is decimal
the timestamp clock period (tick) is 10_{10} microseconds

the overload timer is 500_{10} milliseconds
the system interrupt on error is enabled
the system interrupt on vector instruction b, c, or f is enabled
all circular buffer definitions are erased
all instruction blocks are initialized to 'no instructions' or 'done'
all vector indexes are set to instruction block 0
all terminal shared memory is initialized to 0s
all terminal personality PROM is initialized to 1s (ff hex)
all error messages are cleared from the error message queue
the error message format is set to normal
the VXI FHS active bit is disabled
the interrupt register is cleared

SYSFAIL* Operation

SYSFAIL* becomes active during power-up, hard or soft reset, self test, or if the module loses any of its power voltages. When the mainframe Resource Manager detects SYSFAIL* set, it will attempt to inhibit the line. This will cause the VX4469A Module to deactivate SYSFAIL* in all cases except when +5 volt power is lost.

Functional Overview

The VX4469A ARINC 629 Communication Module is programmed by ASCII characters issued from the system controller to the VX4469A Module via the module's VXIbus commander and the VXIbus mainframe backplane. The module is a VXIbus Message Based Device and communicates using the VXIbus Word Serial Protocol. Refer to the manual for the VXIbus device that will be the VX4469A Module's commander for details on the operation of that device.

If the module's commander is a Tektronix/CDS Resource Manager/IEEE-488 Interface Module, refer to that Operating Manual and the programming examples in this manual for information on how the system controller communicates with the commander being used.

The VX4469A ARINC 629 Communication Module supports from one to three ARINC 629 terminals. The VX4469A is designed to transmit and receive data on ARINC 629 buses through a current (transformer) coupling device that interfaces to bus itself. Data to be transmitted is stored in system memory shared by the Terminal IC and an on-board 80186 processor. The transmit schedule and system memory data locations are stored in a Transmit Personality PROM (XPP). Data that is received is stored in the same system memory. A Receive Personality PROM (RPP) and a Multiple Personality PROM (MPP) contain information on which data to receive and where to store it in the shared system memory.

Note that the VX4469A actually uses RAM instead of PROM for storing XPP, RPP, and MPP information. All references to personality PROMs in this manual are actually references to RAM locations.

If the VX4469A is read without first giving it a command that would return information, it will return its default message of:

VX4469A TMx<cr><lf>

where x is the current default terminal. If the VX4469A has any errors in its error queue, the default message will be:

VX4469A TMx ERRORS<cr><lf>

Power-on

The VX4469A Module will complete its self test and be ready for programming five seconds after power-on. The VXIbus Resource Manager may add an additional one or two second delay. The MSG LED will blink during the power-up sequence as the VXIbus Resource Manager addresses all modules in the mainframe. The default condition of the module after power-on is described in *Functional Check*.

VXIbus Basics

The VX4469A Module is a C size single slot VXIbus Message-Based Word Serial instrument. It uses the A16, D16 VME interface available on the backplane P1 connector and does not require any A24 or A32 address space. The module is a D16 interrupter.

The VX4469A Module is neither a VXIbus commander or VMEbus master, and therefore it does not have a VXIbus Signal register. The VX4469A is a VXIbus message based servant.

The module supports the Normal Transfer Mode of the VXIbus, using the Write Ready, Read Ready, Data In Ready (DIR), and Data Out Ready (DOR) bits of the module's Response register.

A Normal Transfer Mode read of the VX4469A Module proceeds as follows:

1. The commander reads the VX4469A's Response register and checks if the Write Ready and DOR bits are true. IF they are, the commander proceeds to the next step. If not, the commander continues to poll these bits until they become true.

2. The commander writes the Byte Request command (0DEFFh) to the VX4469's Data Low register.

3. The commander reads the VX4469A's Response register and checks if the Read Ready and DOR bits are true. If they are, the commander proceeds to the next step. If not, the commander continues to poll these bits until they become true.

4. The commander reads the VX4469A's Data Low register.

A Normal Transfer Mode Write to the VX4469A Module proceeds as follows:

1. The commander reads the VX4469A's Response register and checks if the Write Ready and DIR bits are true. If they are, the commander proceeds to the next step. If not, the commander continues to poll the Write Ready and DIR bits until they are true.

2. The commander writes the Byte Available command which contains the data (0BCXX or 0BDXX, depending on the End bit) to the VX4469A's Data Low register.

The VX4469A Module has a register beyond those defined for VXIbus message based devices. This register may be used for 16 bit data transfers between the VXI backplane and the terminal shared memory. Any attempt by another module to read or write to any undefined location of the VX4469A's address space may cause incorrect operation of the module.

As with all VXIbus devices, the VX4469A Module has registers located within a 64 byte block in the A16 address space.

The base address of the VX4469A device's registers is determined by the device's unique logical address and can be calculated as follows:

Base Address = V * 40H + C000H

where V is the device's logical address as set by the Logical Address switches.

VX4469A VXI Registers

Below is a list of the VX4469A VXI registers with a complete description of each. In this list, RO = Read Only, WO = Write Only, R = Read, and W = Write. The offset is relative to the module's base address.

Table 2–1: Register Definitions

Register	Address	Type	Value (Bits 15-0)
ID Register	0000H	RO	1011 1111 1111 1100 (BFFCh)
Device Type	0002H	RO	See Device Type definition below
Status	0004H	R	Defined by state of interface
Control	0004H	W	Defined by state of interface
Offset	0006H	WO	Not used

Table 2–1: Register Definitions (Cont.)

Register	Address	Type	Value (Bits 15-0)
Protocol	0008H	RO	1111 0111 1111 1111 (F7FFh)
Response	000AH	RO	Defined by state of the interface
Data High	000CH		Not used
Data Low	000EH	W	See Data Low definition below
Data Low	000EH	R	See Data Low definition below
Data Transfer	0020H	RW	See GRD and GWD commands.

Word Serial Commands

A write to the Data Low register causes this module to execute some action based on the data written. The device-specific Word Serial command this module responds to and the result of this command is:

Command	Response
Read Protocol	FE6Bh

VX4469A Interrupts

The VX4469A will interrupt its commander with the following “event” either if the error interrupt is enabled (SSEE command) and an error is added to the error queue, or if vector interrupts are enabled (SSVE command) and a vector function b, c or f occurs.

Request True:

```
15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
 1  1  1  1  1  1  0  1  <--Logical Address--->
```

ARINC 629

Each ARINC 629 terminal on the VX4469A Module consists of the following:

- System memory shared by the VX4469A 80186 controller and each terminal. It is used to store data to be transmitted and received.
- A personality PROM (RAM) that contains the information on what data to transmit and receive and the data's location in shared memory.
- A terminal protocol IC which interprets the personality PROM (RAM) and translates data to/from 16-bit shared memory words and Manchester bit serial encoding.
- A Serial Interface Module (SIM) that modulates/demodulates the Manchester coding and is intended to drive a current coupler on a twisted pair bus.

All data transmitted on the ARINC 629 bus consists of at least a label. Zero to 256 16-bit data words follow a label.

Data to be transmitted is placed in a terminal's shared memory by the 80186 processor. The terminal IC reads the transmit portion of the Personality PROM (RAM) to determine what labels to transmit and where the data, if any, to be transmitted with each label is located in shared memory.

Data to be received and where to place it in shared memory is determined by the terminal IC reading the receive and multiple personality portions of the Personality PROM (RAM). The 80186 can then read the received data from shared memory.

Protocol Timers

Each terminal determines when to transmit by using three timers and monitoring bus activity. The first timer is the Transmit Interval and should be the same for all terminals on the bus. It determines the minimum amount of time a terminal waits between transmissions. This timer is reset as soon as the terminal begins to transmit and counts to completion independent of what happens on the bus.

The second timer is the Sync Gap. The sync gap time is the same for each terminal on the bus. The sync gap is used to insure that all terminals have a turn to transmit before any terminal re-transmits. The Sync Gap timer is reset when the terminal begins to transmit. Until it counts to completion, any bus activity will reset it again. Once it has counted to completion, it will not reset again until the terminal transmits again. This insures that there is a bus quiet time at least Sync Gap long after each terminal has transmitted before that terminal transmits again.

The third timer is the Terminal Gap. The terminal gap time is different for each terminal on the bus, and determines which terminal transmits first if two or more terminals' Transmit Interval timers have elapsed and the bus is busy. The Terminal Gap timer is reset with any bus activity and will count only after the Sync Gap timer has completed. The terminal will transmit after the Transmit Interval timer and Terminal Gap timer have counted to completion. If two or more terminals' Transmit Interval and Sync Gap timers have completed, the terminal with the shortest Terminal Gap will transmit next. All terminals' Terminal Gap times are shorter than the common Sync Gap time.

Modes

Periodic Mode

A bus that is not fully loaded runs in Periodic mode. In this mode, the transmit interval time is longer than the time required for all terminals to complete one transmission. After a terminal transmits, the Transmit Interval timer begins to count. Before the Transmit Interval timer completes, there will be a quiet time on the bus at least Sync Gap long so the Sync Gap timer will have counted to completion. If another terminal is not transmitting when the transmit interval completes, the terminal gap timer will have completed also and the terminal will transmit as soon as its transmit interval has completed. What normally happens is that all the terminals tend to creep up behind the terminal with the slowest clock. In this condition, the terminals are not transmitting in any particular order but each one is separated from the previously transmitting terminal by its terminal gap. The first one transmitting will be the one with the slowest clock.

Aperiodic Mode

A bus with terminals transmitting more data than will fit in the transmit interval will automatically run in Aperiodic mode. In this mode, after all terminals have transmitted, none will transmit again until there has been a bus quiet time of Sync Gap. Thereafter the terminals will tend to transmit in Terminal Gap order, shortest to longest.

A bus may normally switch between Periodic and Aperiodic modes as terminals have more or less data to transmit.

Instrument I/O

Transmit Schedules

The basic unit of data transmission is a wordstring. A wordstring consists of a label and between 0 and 256 data words. A terminal is allowed to transmit up to 31 wordstrings per transmission. The order in which a terminal transmits data is determined by the transmit portion of its personality PROM. The transmit personality PROM is divided into a 31 by 31 array of cells each eight bytes long.

Each cell describes a wordstring to transmit, the label, data address in shared memory and data length.

There are three modes of scheduling: Block, Independent and Alternate. These modes are controlled by a VX4469A Control register driving input pins on the terminal IC.

In Block mode, the wordstrings described in a single row are transmitted. A row counter is incremented with each transmission. The row counter is reset when it becomes larger than maximum row value (y modulo) stored in the first control cell located in the 32nd row of the array.

Independent mode transmits one wordstring from each column of the array. Each column may have a different number of wordstrings defined. The number for each column (y modulo) is defined in the control cell in the 32nd row of that column. There is a separate row counter for each column. When the y modulo row of each column is reached, its row counter is reset to 0.

Alternate mode can only be entered from Block mode. It is intended that the terminal IC can be switched back and forth between Block mode and Alternate mode while the bus is in operation. The schedule cells for Alternate mode are located in the last rows of the 31 by 31 array. It is similar to Block mode with its first row number defined in the first control cell. Alternate schedule is transmitted one row of wordstrings at a time until the row defined in the 31st row of the array. This last row is repeated until the terminal IC is returned to Block mode.

Hardware CRC

The VX4469A also has hardware capability for generating CRCs on transmitted wordstrings. The CRC hardware reads the label and data at the same time the terminal IC does. The CRC hardware calculates the CRC and supplies the CRC word to the terminal IC at the time the terminal IC is reading the last word of the wordstring. The CRC hardware works with the terminal IC and does not require any bus time from the 80186. The VX4469A also has hardware CRC verification on receive data.

Receive Interrupt Vectors and Label Extension

When an ARINC 629 terminal transmits a wordstring, the first word of that wordstring is its label. The high four bits of this label word is called the label extension. The label extension is the channel ID of the terminal transmitting the wordstring.

When a terminal receives a wordstring, it can optionally generate an interrupt vector. You can program the VX4469A to use this interrupt vector number for a variety of functions, including storing in FIFO memory the interrupt vector number and a timestamp.

The terminal IC does not use the label extension when it generates an interrupt vector. Therefore the interrupt vector generated when a label is received is independent of the channel ID of the transmitter.

The VX4469A, under program control, can substitute the label extension for the low four bits of the late interrupt vector. This allows you to easily differentiate between terminals transmitting the same label with different channel IDs. The VX4469A also can return to the user the label extension with timestamp data.

Transmit Channel ID

The VX4469A can enable a terminal to use a Channel ID stored in the high four bits of an Xpp label field, instead of the channel ID set with the SC command. This allows a single terminal to simulate multiple terminals with different channel IDs.

Memory Switching

The terminal IC reads and writes data from and to shared memory. The VX4469A uses memory switching to insure that data in shared memory is not partially overwritten by the terminal IC or user while the other is reading it.

Each terminal has its own 64 Kwords of shared memory. When the shared memory for a terminal is normal, addresses or locations in the shared memory are the same for both the terminal IC and commands such as RD and WD. When memory is switched, the high order addressing bit is inverted for only the terminal IC. This causes a terminal IC which is normally programmed to read and write in the lower 32 Kwords of its shared memory to now read and write in the upper 32 Kwords.

Thus you can examine and modify data in half the shared memory while the terminal IC is operating out of the other half. You can then request a memory switch and examine and modify new data. Switching automatically happens between wordstrings.

A second memory switching mode switches the memory address for the terminal IC in the low half of shared memory to the high half. Address in the high half of memory are not switched. This makes it convenient to use circular buffers with data always found in the high half and also to examine/change other data using memory switching.

Syntax and Commands

Command Syntax

Command protocol and syntax for the VX4469A Module are as follows:

1. Each command is terminated by a semicolon or a line feed.
2. White space characters (including space, tab, and carriage return) are ignored.
3. Non-printing characters are indicated by the following:

   <cr> carriage return.

   <lf> line feed.

   <tm> terminator, either a linefeed or semicolon.
4. Characters may be sent as either upper or lower case.
5. Comments may be added to commands and will be ignored by the VX4469A. Begin the comment with an ! and end the command with a terminator. The ! must not be in the middle of a command, but may be placed after a line feed or semi-colon.
6. In the command descriptions, the following conventions have been used:

   Brackets [ ] are used to show optional parts of commands.

   Parts of commands enclosed in parenthesis ( ) contain two or more choices, one of which must be used.

   Lower case letters are used to represent numeric values. The descriptions following the commands describe the use and range of these numbers.

Functional Command Groups

This section lists the VX4469A commands by functional command group.

System Commands

These low-level commands are typically sent by the module's commander, transparent to the user of the module. An exception is the Read Status command, which is sent whenever a Serial Poll on an IEEE-488 system is performed. Most commanders or Slot 0 devices have specific ASCII commands which will cause them to send one of these low-level commands to a specified instrument. Refer to the Operating Manual of the commander or Slot 0 device for information on these commands.

Command	Effect
Clear	The module clears its VXIbus interface and any pending commands. Current module operations are unaffected.
Begin Normal Operation	The module will begin operation per VXI Specification.
Read Protocol	The module will return its protocol to its commander.
Read Status	The module will return its VXI status byte to its commander.
Set Lock	Set the Lock bit of the Response register.
Clear Lock	Clears the Lock bit of the Response register.
Read Interrupters	Returns the value FFF9, indicating there is one interrupter on this module.
Read Interrupt Line	Returns the interrupt line per VXI Specification.
Asynchronous Mode Control	Returns information that events are being sent as interrupts per VXI Specification.
Abort Normal Operation	Causes this device to cease normal operation per VXI Specification.
End Normal Operation	Causes this device to cease normal operation per VXI Specification.
Control Event	Used by a commander to selectively enable the generation of events by a servant.
Read Protocol Error	Returns the module's most recent error code, which includes multiple query errors, unsupported commands, and DOR violations.
Byte Available	Transfers module commands to this module.
Byte Request	Requests data be returned form the module.

Command	Effect
Control Response	Returns information indicating response interrupts are not supported.
Trigger	This module will accept the Trigger command, although no part of this instrument will be affected by it.

Module Commands

A summary of the VX4469A Module's commands is listed below. This is followed by detailed descriptions of each of the commands. A sample BASIC program using these commands is shown in the *Programming Examples* section.

Command Summary

Detailed descriptions of each command (in alphabetical order) are given following the summary. An overview of the commands is as follows:

Command	Description
BR	binary read: BRC circular buffer — BRG registers BRD data — BRM multiple PP BRR receive PP — BRX transmit PP
BW	binary write: BWC circular data — BWD data BWM multiple PP — BWR receive PP BWX transmit PP
CC	clear circular buffer data.
CCA	clear all circular buffers of data.
CF	clear a specified terminal's hardware FIFO.
DC	define circular buffer.
F	fill the Personality PROM (RAM) for a specified terminal with all ones (FF hex), or fills shared memory with zeros.
GRD	supply data to the 16-bit register on the VXI backplane.
GWD	accept data via the 16-bit register on the VXI backplane.
HN	switch this terminal's memory addressing to normal at the beginning of the next wordstring transmitted or received by this terminal.
HPN	switch this terminal's memory addressing to normal at the beginning of the next wordstring transmitted by this terminal whose transmit cell in the Xpp has the Switch bit true.

Command	Description
HPS	switch this terminal's memory addressing to switched at the beginning of the next wordstring transmitted by this terminal whose transmit cell in the Xpp has the Switch bit true.
HR	reset this terminal's memory addressing to normal immediately. If it is already in normal, this command has no effect.
HS	switch this terminal's memory addressing to switched at the beginning of the next wordstring transmitted or received by this terminal.
IA	initialize module to power-up, except power-up ROM.
IC	initialize circular buffer definitions.
IM	reset module to power-up state and then program it to record in circular buffer 0 all labels with timestamps that are being transmitted on a bus.
IN	initialize and monitor all terminals.
IVI	initialize vector instructions.
IVX	initialize vector index table.
LC	list circular buffer status.
LCB	list circular buffer status in binary format.
LE	set up the VX4469A to return any error messages in its error queue.
LG	set up the VX4469A to return the current value of the high 4 bits of terminals currently in Test Mode.
LH	list memory switch status.
LP	set up the VX4469A to return the contents of the power-up PROM.
LR	list revision.
LS	set up the VX4469A to send back information about its setup.
LVI	set up the VX4469A to return the instructions in a particular instruction block.
LVX	set up the VX4469A to return vector index information.
NRD	front panel data port read data.
NWD	front panel data port write data.
RC	read circular buffer.
RCC	read circular buffer, calculating and appending a CRC, using the circular buffer cell size as the 'number of data words' in the wordstring.
RCV	read circular buffer variable, calculating and appending a CRC, using the first data word in the circular buffer cell or the cell size, whichever is less, as the 'number of data words' in the wordstring.
RD	read data from terminal's shared memory.
RDC	read data from shared memory, calculating and appending a CRC.

Command	Description
RDV	read data variable from shared memory, calculating and appending a CRC, using the word at 'addr' in shared memory as the 'number of words' in the wordstring.
RG	read terminal IC's status registers.
RI	read interrupt status.
RM	read Multiple Personality PROM (RAM).
RR	read Receive Personality PROM (RAM).
RS	read Serial Interface Module status.
RX	read Transmit Personality PROM (RAM).
SBD	disable the terminal IC from accessing the shared memory.
SBE	enable the terminal IC to access the shared memory.
SC	set the terminal ID for transmit data and receive data channeling.
SCL	set the Channel ID to use or not use the Xpp label field for the channel ID.
SD	disable the specified terminal(s).
SE	enable the specified terminal(s).
SFB	set error message format to brief.
SFN	set error message format to normal.
SH	set the memory switch mode.
SI	set the ARINC 629 parameters TI (transmit internal), TG (terminal gap), and SG (sync gap).
SKD	disables VXI TTL trigger, External trigger or label enables on a terminal in test mode 2.
SKI	causes a terminal in test mode 2 to transmit immediately.
SKL	causes a terminal in test mode 2 to transmit after it receives a particular label.
SKV	causes a terminal in test mode 2 to transmit after it receives a VXI TTL trigger.
SKX	causes a terminal in test mode 2 to transmit after it receives an external trigger.
SMA	set the alternate mode pin on the terminal IC true.
SMB	set the protocol transmit mode to block.
SMI	set the protocol transmit mode to independent.
SMN	set the alternate mode pin on the terminal IC false. This command allows switching the terminal IC from alternate mode to block mode.
SO	set overload timer value.

Command	Description
SQVE	enable/disable setting the terminal VXI trigger on a communication error.
SQVG	enable/disable setting the terminal VXI trigger on beginning to transmit.
SQVI	enable/disable setting the terminal VXI trigger on interrupt vector bit 13.
SQVR	enable/disable setting the terminal VXI trigger on interrupt vector bit 13 when this terminal has just received a wordstring that does not have a valid CRC.
SQXE	enable/disable setting the terminal VXI trigger on a communication error.
SQXG	enable/disable setting the terminal VXI trigger on beginning to transmit.
SQXI	enable/disable setting the terminal VXI trigger on interrupt vector bit 13.
SQXR	enable/disable setting the terminal VXI trigger on interrupt vector bit 13 when this terminal has just received a wordstring that does not have a valid CRC.
SR	set the radix of numeric data or command parameters.
SSE	set system error interrupt.
SST	set receive threshold.
SSV	set system vector interrupt.
ST	sets the time-stamp clock period.
SVI	set up a list of commands to be executed whenever a particular vector or vectors is/are generated.
SVX	set which instruction block is to be used by each vector.
SW	set the terminal to timestamp the end of a wordstring.
SXD	set data transfer to disable VXI fast handshake protocol.
SXE	set data transfer to enable VXI fast handshake protocol.
TQV	test a terminal's VXI TTL trigger.
TQX	test a terminal's external trigger.
TQSV	test the VX4469A's software VXI TTL trigger.
TQSX	test the VX4469A's software external trigger.
TS	test SIM.
URM	a user friendly way of reading data from the Multiple Personality PROM.
URR	a user friendly way of reading data from the Receive Personality PROM.
URX	a user friendly way of reading data from the Transmit Personality PROM.
UWM	a user friendly way of writing data to the Multiple Personality PROM.

Command	Description
UWR,f	a user friendly way of writing data to the Receive Personality PROM.
UWX,c,f	a user friendly way of writing data to the Transmit Personality PROM.
WC	write data to a cell in a circular buffer.
WCC	write data to a circular buffer, calculating, and appending a CRC.
WD	write data to a terminal's shared memory.
WDC	write data, calculating and appending a CRC, to shared memory starting at 'addr', including 'label' in the CRC calculation.
WM	writes Multiple Personality PROM.
WR	writes Receive Personality PROM.
WX	writes Transmit Personality PROM.

All commands must end with a terminator <tm>, which may be a line feed <LF> or semi-colon. White space characters are ignored.

The optional [U] (user friendly) and [B] (binary) parameters are mutually exclusive. A command can not use both of these parameters at the same time.

The following summary shows the commands grouped according to function, and may be useful as a quick reference guide.

(t)	terminal number prefix valid for all commands
(G)	use 16-bit register
(U)	user friendly prefix valid for some commands, not for use with (B)
(B)	binary prefix valid for some commands, not for use with (U)!!
(N)	front panel prefix, valid for some commands.
CT	current terminal

Read

(B)RC	read circular buffer
RCC	read circular buffer CRC
RCV	read circular buffer CRC variable
(G)RD	read data to 16-bit register
(t)(B)RG	read CT's registers
(t)(N)(B)RD	read CT's data
(t)RDC	read CT's data CRC
(t)RDV	read CT's data CRC variable
(t)(U,B)RR	read CT's receive PP
(t)(U,B)RX	read CT's transmit PP
(t)(U,B)RM	read CT's multiple PP
(t)RI	read interrupt status

Write

(B)WC	write circular buffer
WCC	write circular buffer CRC
(G)WD	accept data via 16-bit register
(t)(B)(N)WD	write CT's data
(t)WDC	write CT's data with CRC
(t)(U,B)WR	write CT's receive PP
(t)(U,B)WX	write CT's transmit PP
(t)(U,B)WM	write CT's multiple PP

List

LC	list circular buffer status
LE	list error queue
LG	list high 4 bits of terminals currently in Test Mode
LH	list memory switch status
LS	list module's setup
LVI	list vector instructions
LVX	list vector index
LR	list revision

Set

(t)SBD	set CT's instrument bus disable
(t)SBE	set CT's instrument bus enable
(t)SC	set CT's channel ID
(t)SCL	use or not use Xpp label field
(t)SD	set CT disable
SDA	set all terminals disable
(t)SE	set CT enable
SEA	set all terminals enable
SFB	set error message format to brief
SFN	set error message format to normal
SHO	set memory switch to OR mode
SHS	set memory switch to invert mode
(t)SI	set CT's intervals (ti,tg,sg)
(t)SKD	set kollision disable
(t)SKI	set kollision immediate
(t)SKL	set kollision label
(t)SKV	set kollision VXI TTL
(t)SKX	set kollision external
(t)SMA	set CT's mode to alternate
(t)SMB	set CT's mode to block
(t)SMI	set CT's mode to independent
(t)SMN	set CT's mode to not alternate
SO	set overload timer value
SRDD	set data radix to decimal
SRDH	set data radix hexadecimal
SRCD	set command radix to decimal
SRCH	set command radix to hexadecimal
SSE	set system error interrupt
SST	set receive threshold
SSV	set system vector interrupt
ST	set timestamp period
SVI	set vector instructions
SVX	set vector index
SXD	set data transfer to disable VXI fast handshake protocol.
SXE	set data transfer to enable VXI fast handshake protocol.

Fill

(t)FR	fill CT's receive PP with 1s
(t)FX	fill CT's transmit PP with 1s
(t)FM	fill CT's multiple PP with 1s
(t)FD	fill CT's shared memory with 0s
(t)FP	fill CT's PP with 1s
(t)FA	fill CT's PP with 1s and shared memory with 0s

Clear

CC	clear circular buffer
CCA	clear all circular buffers
(t)CF	clear CT's hardware FIFO
CFA	clear all terminals' hardware FIFO

Define

DC	define circular buffer

Initialize

IA	initialize all
IC	initialize circular buffer definitions
IM	initialize and record timestamped transmitting labels
IVI	initialize vector instruction cells
IVX	initialize vector index table

Test

(t)TS	test SIM

Memory Switch

(t)HN	switch memory addressing to normal
(t)HS	switch memory addressing to switched
(t)HR	reset memory addressing to normal
(t)HPN	switch memory addressing to normal
(t)HPS	switch memory addressing to switched

Command Descriptions

Detailed descriptions of the VX4469A Module's commands, in alphabetical order, are listed on the following pages.

NOTE. *All numbers in these descriptions are hexadecimal unless otherwise indicated.*

The radix for data and command parameters both to and from the module are determined by the set radix commands, SRxx. The radix for data or parameters sent to the module may be modified for a particular number by preceding that number with a % character for decimal or # character for hexadecimal. For instance, if the write data to shared PROM is used, and the current radix for both data and command parameters is hex, decimal numbers may be included as follows:

WD,100,7,8,9,%10,B,C,%13,E

Or, if the data and command parameters are decimal, hex numbers may be included as follows:

WD,#100,7,8,9,10,#B,12,#D,13

Some of the commands, such as RD (read data) or SE (set enable), are terminal dependent. A terminal is selected by specifying a terminal number. If the board is a 3-terminal board, the number may be 0, 1, or 2. Any command may be prefixed with a terminal number, or a terminal may be selected by the terminal number followed by a terminator character. Until a different terminal number is specified, all terminal-specific commands will refer to the previously selected terminal. The currently selected terminal is indicated in the default readback message.

BRC

Syntax `BRC,circular buffer number,number of words<tm>`

Purpose Read circular buffer, binary. Sets the VX4469A to return data from a circular buffer in binary data format.

Description circular buffer number is 0 to 0f

number of words is 0 to 07fff (32,767 decimal).

If the number of words requested is greater than the number in one cell, other cells will be read and returned until the number is satisfied. If the number of words requested is zero, the VX4469A will return data forever. This command may be canceled before completion by sending another command.

Once the data is completely read from a cell, that cell is erased.

The data is returned as two 8-bit binary characters per word, with the least significant bits of each word first. Binary mode uses no delimiters between data words or circular buffer cells.

Example BRC,3,23<tm>

sets up the VX4469A to return data from circular buffer 3 until 23 words have been read. If circular buffer 3 has no full cells, the system controller will be held off until data is available. If the remainder of 23 divided by circular buffer 3's cell size is not zero, the last partial cell read will not be deleted from the circular buffer.

BRD

Syntax `[t]BRD,start address,number of words<tm>`

Purpose The Binary Read Data command reads data from the current terminal's shared memory.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the shared memory space is by 16-bit word and not by byte.
number of words	is 0 to 07fff (32,767 decimal).

The data is returned as two 8-bit binary characters per word, with the least significant eight bits of each word first. Binary mode uses no delimiters between data words.

Example 2BRD,100,5<tm>

reads five words of data from terminal 2, beginning at shared memory address 100.

BRG

Syntax `[t]BRG<tm>`

Purpose The Binary Read Registers command sets up the VX4469A to read and return the contents of the terminal IC's internal registers in binary format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

The three registers are read at the time data is requested from the VX4469A. They are read one at a time, so it is possible that the registers will change while being read.

Six binary bytes are returned in the order shown, with the least significant byte of each word first:

aaaa,bbbb,cccc<cr><lf>

where aaaa is the Error register, bbbb is the last word monitor, and cccc is the Interrupt Vector register.

The Error register bits are as follows:

```
f  e  d  c  b  a  9  8  7  6  5  4  3  2  1  0
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |_impersonate
|  |  |  |  |  |  |  |  |  |  |  |  |  |__|__op mode
|  |  |  |  |  |  |  |  |  |  |  |  |__ short string
|  |  |  |  |  |  |  |  |  |  |  |__ parity error
|  |  |  |  |  |  |  |  |  |  |__ receive error flag
|  |  |  |  |  |  |  |  |  |__ no busa, write
|  |  |  |  |  |  |  |  |__ no busa, read
|  |  |  |  |  |  |  |__ pam error - tg,sg
|  |  |  |  |  |  |__ pam error - ti
|  |  |  |__|__|__ transmit monitor errors (encoded)
|  |  |__ transmit error flag
|  |__ transmitter enable
|__ string error
```

The last word monitor register contains the last label or data word sent for a transmit operation and the last label word received for a receive operation.

The interrupt vector register (IVR) contains the 15-bit interrupt vector for the current received or transmitted wordstring. The most significant bit of the IVR contains the string error bit. The IVR makes available the value of the current

interrupt vector at any time whether or not the Personality PROMS have been programmed to provide interrupt vector strobes.

BRM

Syntax `[t]BRM,start address,number of bytes<tm>`

Purpose The Binary Read Multiple command reads the Multiple Personality PROM in binary format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. The terminal must be disabled to read or write its personality PROM.
number of bytes	from 1 to 10,000.

The data is returned as one 8-bit binary character per byte. Binary mode uses no delimiters between data bytes.

Example 0BRM,0,100<tm>

sets up to read back 100 bytes of terminal 0 Multiple Personality PROM in binary format, starting at address 0.

BRR

Syntax `[t]BRR,start address,number of bytes<tm>`

Purpose The Binary Read Receive command reads the Receive Personality PROM in binary format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. The terminal must be disabled to read or write its personality PROM.
number of bytes	from 1 to 10,000.

The data is returned as one 8-bit binary character per byte. Binary mode uses no delimiters between data bytes.

Example 0BRR,200,10<tm>

sets up to read back 10 bytes of terminal 0 Receive Personality PROM in binary format, starting at address 200.

BRT

Syntax `[t]BRT,segment,start address,number of bytes<tm>`

Purpose The Binary Read Test Ram command sets up the VX4469A to return the contents of the current terminal's Test Ram in binary format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
segment	Test Ram segment.
start address	the byte address within the Test Ram segment.

Note that the addressing in the Test Ram space is by 8-bit byte. The terminal must be in Test Mode and disabled to read or write its Test Ram.

The data is returned as one 8-bit binary character per byte. Binary mode uses no delimiters between data bytes. If the user continues to read after "number of bytes" bytes, the VX4469A will return space,<cr>,<lf> characters.

Example 0BRT,2,0,10<tm>

sets up to read back 10 bytes of terminal 0 Test Ram starting at segment 2 address 0 in binary format.

BRX

Syntax `[t]BRX,start address,number of bytes<tm>`

Purpose The Binary Read Xmit command reads the Transmit Personality PROM in binary format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. The terminal must be disabled to read or write its personality PROM.
number of bytes	from 1 to 10,000.

The data is returned as one 8-bit binary character per byte. Binary mode uses no delimiters between data bytes.

Example 0BRX,0,100<tm>

sets up to read back 100 bytes of terminal 0 Transmit Personality PROM in binary format, starting at address 0.

BWC

Syntax `BWC,circular buffer number,number of words<tm>data`

Purpose Binary Write Circular buffer. This command writes data to a cell in a circular buffer using binary format.

Description

circular buffer number	The circular buffer that the data is to be written to.
number of words	0 to 07fff (32,767 decimal)

If the number of words requested is greater than the number in one cell, other cells will be written into until the number is satisfied. If the number of words requested is zero, all data transmitted to the VX4469A will be written into the circular buffer cells until the command is canceled. This command may be canceled before completion by reading from the module.

Once the data to a cell is completely written, that cell is released to be read by a vector instruction.

The data is written as two 8-bit binary characters per word, the least significant with bits of each word first. Binary mode uses no delimiters between data words or circular buffer cells.

Example BWC,4,10<tm>data

Writes 10 words of data (20 bytes) following the <tm> in circular buffer 4.

Refer to *Fault Management and Internal Test Functions* in *Appendix G* for further information.

BWD

Syntax `[t]BWD,start address,number of words<tm>data`

Purpose The Binary Write Data command writes data to shared memory in binary format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the data space is by 16-bit word and not by byte.
number of words	0 to 0ffff.

The data is written as two 8-bit binary characters per word, the least significant byte of each word first. Binary mode uses no delimiters between data words or circular buffer cells.

Example BWD,240,43<tm>data

Writes 43 words of data (86 bytes) following the <tm> in the currently selected terminal's shared memory.

BWM

Syntax `[t]BWM,start address,number of bytes<tm>data`

Purpose The Binary Write Multiple command writes Multiple Personality PROM in binary format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. A terminal must be disabled to read or write its Personality PROM.
number of bytes	from 1 to 10,000.

The data is written as one 8-bit character per byte.

Example 1BWM,100,400<tm>data

writes 400 bytes of data following the <tm> in terminal 1's Multiple Personality PROM starting at address 100.

BWR

Syntax `[t]BWR,start address,number of bytes<tm>data`

Purpose The Binary Write Receive command writes Receive Personality PROM in binary format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. A terminal must be disabled to read or write its Personality PROM.
number of bytes	from 1 to 8000.

The data is returned as single 8-bit binary characters. Binary mode uses no delimiters between data bytes.

Example 1BWR,0,1000<tm>data

writes 1000 bytes of data following the <tm> in the terminal 1's Receive Personality PROM.

BWT

Syntax `[t]BWT,segment,start address,number of bytes<tm>data`

Purpose The Binary Write Test Ram command writes the Test Ram in decimal binary format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
segment	the Test Ram segment that data is to be written to.
start address	the offset into the segment you wish to start writing data.
	Note: *The addressing in the Test Ram space is by 8-bit byte. A terminal must be in Test Mode and disabled to read or write its Test Ram.*
data	data bytes are expected in numeric ASCII characters with comma delimiters. The numbers have the following ranges: The data is written as one 8-bit character per byte.

Example 0BWT,2,0,127<tm>data

writes 127 bytes of data following the <tm> in terminal 0's Test Ram starting at segment 2 address 0.

BWX

Syntax `[t]BWX,start address,number of bytes<tm>data`

Purpose The Binary Write Xmit command writes Transmit Personality PROM in binary format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. A terminal must be disabled to read or write its Personality PROM.
number of bytes	from 1 to 10,000.

The data is written as one 8-bit character per byte.

Example 0BWX,200,500<tm>data

writes 500 bytes of data following the <tm> in terminal 0's Transmit Personality PROM starting at address 200.

CC

Syntax `CC,n<tm>`

Purpose This command clears a circular buffer of any data.

Description n is a circular buffer number, 0 through 15.

Example CC,3<tm>

clears all data out of circular buffer 3.

CCA

Syntax `CCA<tm>`

Purpose This command clears all circular buffers of any data.

Description clears all circular buffers of any data.

Example CCA<tm>

clears all data out of all circular buffers.

CF

Syntax `[t]CF[A]<tm>`

Purpose This command clears a specified terminal's hardware FIFO.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
[A]	If the letter 'A' is added to this command, all three terminals' FIFOs are emptied.

Example 1CF<tm>

empties terminal 1's hardware vector FIFO.

DC

Syntax `DC,circular buffer number,cell size,number of cells<tm>`

Purpose Define Circular buffer. This command allows you to define up to 16 circular buffers for use in transferring data on or off the module and between terminals. All terminals have access to all the circular buffers.

Description

circular buffer number	a number from 0 to 0f.
cell size	the number of words per cell
number of cells	the number of cells in the circular buffer

A circular buffer is a 80186 memory buffer that stores user-defined blocks of data. The number of words per block or cell and the number of cells in a circular buffer are defined. The total number of words that a circular buffer may contain is 7fff (32,767 decimal).

All data is written to one end of the circular buffer and read from the other end. This means that data is read out in the same order it was written in.

A typical use for a circular buffer is synchronizing new wordstring data with a terminal transmitting. The user commands the VX4469A to check for data in one of the circular buffers after each transmission of a wordstring. If there is data in the circular buffer, the VX4469A will move a block of data from the circular buffer to shared memory for the next transmission of the wordstring. If there is no data available, the old data is transmitted. Data is put in the circular buffer by writing to the circular buffer with the WC command. Data won't be transferred to shared memory until a complete block of data is in the circular buffer.

Example DC,1,5,100<tm>

define circular buffer 1 to have 100 5 word cells.

F (FR, FX, FM, FP, FD, FA)

Syntax `[t]F[ram]<tm>`

Purpose This command fills the personality PROM for a specified terminal with all ones (FF hex), or fills shared memory with zeros.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
[ram]	a single letter, which must be one of the following: R the receive portion of the Personality PROM (RAM). X the transmit portion of the Personality PROM (RAM). M the multiple portion of the Personality PROM (RAM). P all of the Personality PROM (RAM). D shared memory. A all Personality and data PROM (RAM).

Example 1FX<tm>

fills terminal 1's Transmit Personality PROM (RAM) with ones.

GRD

Syntax `[t]GRD,start address, number of words<tm> data.`

Purpose Register read data. Sets up the VX4469A to supply data to its 16-bit register on the VXI backplane.

Description

[t]	an optional terminal number, which must be either 0, 1 or 2. If omitted, the last value specified is used.
start address	the current terminal's shared memory address where the first data supplied is stored.
number of words	0 to 07FFF

After this command is received by the VX4469A, 16-bit data read from the VX4469A base address + 20 hex will be from sequential locations in the current terminal's shared memory. Data after the number of words specified will be undefined.

Example 1GWD,0,100<tm>

Sets up the VX4469A to transmit 100 words of data to terminal 1 shared memory starting at address 0.

GWD

Syntax `[t]GWD,start address, number of words<tm> data.`

Purpose Register write data. Sets up the VX4469A to accept data via its 16-bit register on the VXI backplane.

Description

[t]	an optional terminal number, which must be either 0, 1 or 2. If omitted, the last value specified is used.
start address	the current terminal's shared memory address where the first data received will be stored.
number of words	0 to 07FFF

After this command is received by the VX4469A, 16-bit data written to the base address + 20 hex will be put in sequential locations in the current terminal shared memory. Data written after the number of words specified will be lost.

Example 1GWD,0,100<tm>

Sets up the VX4469A to receive 100 words of data to terminal 1 shared memory starting at address 0.

HN

Syntax `[t]HN<tm>`

Purpose Switch this terminal's memory addressing to normal at the beginning of the next wordstring transmitted or received by this terminal.

Description

[t]	an optional terminal number, which must be 0, 1, or 2. If omitted, the last value specified is used.

See *Appendix G* for a general description of memory switching.

See also commands HPN, HS, HPS, HR, LH, SHO and SHS.

Example 1HN<tm>

This command will switch memory addressing to normal.

HPN

Syntax [t]HPN<tm>

Purpose Switch this terminal's memory addressing to normal at the beginning of the next wordstring transmitted by this terminal whose transmit cell in the XPP has the switch bit true.

Description

[t]	an optional terminal number, which must be 0, 1, or 2. If omitted, the last value specified is used.

See *Appendix G* for a general description of memory switching.

See also commands HS, HPS, HN, HR, LH, SHO and SHS.

Example 1HPN<tm>

This command will switch memory addressing to normal.

HPS

Syntax [t]HPS<tm>

Purpose Switch this terminal's memory addressing to switched at the beginning of the next wordstring transmitted by this terminal whose transmit cell in the XPP has the switch bit true.

Description

[t]	an optional terminal number, which must be 0, 1, or 2. If omitted, the last value specified is used.

See *Appendix G* for a general description of memory switching.

See also commands HPN, HS, HN, HR, LH, SHO and SHS.

Example 1HPS<tm>

This command will switch memory addressing to normal.

HR

Syntax [t]HR<tm>

Purpose Reset this terminal's memory addressing to normal immediately.

Description

[t]	an optional terminal number, which must be 0, 1, or 2. If omitted, the last value specified is used.

If the terminal's memory addressing is already in normal, this command has no effect.

See *Appendix G* for a general description of memory switching.

See also commands HPN, HS, HPS, HN, LH, SHO and SHS.

Example 1HR<tm>

This command will reset memory addressing to normal.

HS

Syntax [t]HS<tm>

Purpose Switch this terminal's memory addressing to switched at the beginning of the next wordstring transmitted or received by this terminal.

Description

[t]	an optional terminal number, which must be 0, 1, or 2. If omitted, the last value specified is used.

See *Appendix G* for a general description of memory switching.

See also commands HPN, HN, HPS, HR, LH, SHO and SHS.

Example 1HS<tm>

This command will switch memory addressing to switched.

IA

Syntax IA<tm>

Purpose Initialize All.

Description Reset the module to its power-up status.

Example IA<tm>

The module is reset to its power-up status.

IC

Syntax `IC<tm>`

Purpose Initialize circular buffer definitions.

Description This command allows redefining the circular buffer definitions by undoing all the definitions created by the DC command. Individual circular buffer definitions can’t be changed.

Example IC<tm>

No circular buffer definitions remain after this command.

IM

Syntax [t]IM<tm>

Purpose Initialize and monitor.

Description

[t]	an optional terminal number, which must be 0, 1, or 2. If omitted, the last value specified is used.

Reset the module to its power-up status. Then program terminal [t] to monitor a bus.

Example 2IM<tm>

The module is reset to its power-up status. The module is then programmed as follows:

The Receive Personality PROM for terminal 2 is programmed to generate early interrupt vectors equal to the label for all possible labels.

The vector index table indexes for all those interrupt vectors are set up to instruction block 1.

Instruction block 1 is set up to write the interrupt vector (label) and timestamp to circular buffer 0. (function 1)

Circular buffer 0 is set up with 2AAA cells of three words each.

Terminal 2 is enabled.

All labels on the bus that terminal 2 is connected to will be recorded in circular buffer 0 with time stamps.

IN

Syntax `IN<tm>`

Purpose Initialize and monitor all terminals.

Description Reset the module to its power-up status. Then program the VX4469A monitor all terminals.

Example IN<tm>

The module is reset to its power-up status. The module is then programmed as follows:

The Receive Personality PROMs for all terminals are programmed to generate late interrupt vectors equal to the label for all possible labels. Each terminal is also programmed to timestamp the end of each wordstring.

The vector index table indexes for all those interrupt vectors are set up to instruction block 1.

Instruction block 1 is set up to write the interrupt vector (label), 40 bits of timestamp, the CID or label extension, and terminal number into circular buffer 0 (interrupt vector instruction 7).

Circular buffer 0 is set up with 1fff cells of four words each.

All terminals are enabled.

All labels with CID on the bus connected to each terminal will be recorded in circular buffer 0, identified by the receiving terminal, with time stamps of the label and the end of its wordstring.

IVI

Syntax `IVI<tm>`

Purpose Initialize vector instructions.

Description Changes all vector instruction blocks to start with the instruction “done”.

See *Appendix G* for more information on the use of this command.

Example IVI<tm>

All vector instruction blocks now start with the instruction “done”.

IVX

Syntax `IVX<tm>`

Purpose Initialize vector index table.

Description Changes all vector indexes to instruction block 0.

See *Appendix G* for more information on the use of this command.

Example IVX<tm>

All vector indexes are changed to instruction block 0.

LC

Syntax LC[,n]<tm>

Purpose List circular buffer status.

Description Sets the VX4469A up to return the currently defined circular buffers and the number of cells in each that have data.

n	buffer number

If n is specified, the VX4469A will return information beginning at the specified buffer.

Example LC<tm>

Following this command, a typical readback would be:

```
BUFFER  CELL SIZE  NUM CELLS  CELLS USED<cr><lf>0
000       0010       0010        0000  <cr><lf>
0001      0005       0010        0000  <cr><lf>
0002      0003       1000        0000  <cr><lf>
007B KWORDS BUFFER AREA LEFT<cr><lf>
<cr><lf>
```

LC,2<tm>

Following this command, the readback would begin with buffer 2.

See also command LCB.

LCB

Syntax LCB[,n]<tm>

Purpose List circular buffer status in binary format.

Description Sets up the VX4469A to return the number of cells that have data in each circular buffer in binary format.

n	buffer number

If n is specified, the VX4469A will return information beginning at the specified buffer.

The number of cells in use for each terminal is returned as two bytes per circular buffer, the least significant byte first. Circular buffers which are not defined return a value of <0>, <0>. The values are returned in order, starting with circular buffer 0 or circular buffer n if specified. The user need read only up to the values of the circular buffer they are interested in. If the user keeps reading, the values all circular buffers up to circular buffer f are returned, followed by <space>, <cr>, <lf>. Continually reading the VX4469A will repeat the requested data updated to the status at the time the data was read.

Example LCB<tm>

Following this command, a typical readback would be:

```
<7><0><0><0><3f><7><0><0><0><0><0><0><0><0><0><0><0><0><0><0><0><0>
<0><0><0><0><7><0><0><0><0><0><0><0><space><cr><lf>
```

LCB,c<tm>

If we assume that circular buffer c is being emptied by an interrupt vector instruction, following this command, reading back three times might return:

```
<1><0><0><0><0><0><0><0><space><cr><lf><
1><0><0><0><0><0><0><0><space><cr><lf>
<0><0><0><0><0><0><0><0><space><cr><lf>
```

See also command LC.

LE

Syntax LE<tm>

Purpose The List Errors command sets up the VX4469A to return any error messages in its error queue.

Description After all errors are returned, the VX4469A returns a space, carriage return, and line feed. At this time it also resets the communication error flipflops. Only one communication error per terminal will be in the error queue at one time.

Error messages are stored in the error queue in two formats. In normal format, the command string up to the point the error was detected is stored along with a descriptive error message followed by a carriage return and line feed. In brief format, only a single character representing an error message followed by a carriage return and line feed is stored in the error queue. The format is controlled by the SFB and SFN commands.

See *Appendix F* for a list and description of error messages.

Example LE<tm>

This command sets up to return anything in the error queue. If there were no messages in the error queue, the following would be returned:

```
<sp><cr><lf>
```

If the format was set to normal at the time of the error and an error was in the queue, the following is an example of what might be returned:

```
UWZ SYNTAX ERROR<cr><lf><
sp><cr><lf>
```

If the format was set to brief and an error was in the queue, the following would be returned for the same error:

```
9<cr><lf><
sp><cr><lf>
```

LG

Syntax LG<tm>

Purpose List Program Counter of terminals in Test Mode.

Description Sets the VX4469A up to return the current value of the high 4 bits of terminals currently in Test Mode. The VX4469A will continue to return updated values until a new command is sent. This command is useful for verifying that a terminal in Test Mode has changed segments after a SG command.

Example LG<tm>

Following this command, a typical readback would be:

```
0-03 2-0E<cr><lf>0
-03 2-00<cr><lf>
```

This shows that terminals 0 and 2 are in test mode. Terminal 0's Program Counter is in segment 3. Terminal 2's program counter changed from segment e to segment 0 between readings.

If no terminals are in Test Mode, the read back will consist of only a <cr><lf>.

See *Appendix G* for general information on test modes.

LH

Syntax LH<tm>

Purpose List memory switch status. This command sets up the VX4469A to return memory switch status.

Description The VX4469A will return the memory switch status each time it is read until another command is given. The returned string's length is determined by the number of terminals installed on the module. If three terminals are installed, the string would have the following format:

0c 1c 2c<cr><lf>

Where c is either N for normal or S for switched in invert mode or O for switched in OR mode.

Example LH<tm>

The returned string might be:

```
ON 1N 2S
```

indicating the first and second are normal, and the third is switched.

See also *Appendix G* for a general discussion of memory switching.

See also commands HPN, HN, HPS, HR, HS, LH, SHO and SHS.

LR

Syntax LR<tm>

Purpose This command returns the VX4469A software revision level.

Example LR<tm>

Following this command, a typical readback would be:

```
REV 1.1<cr><lf>
```

LS

Syntax LS<tm>

Purpose The List Setup command sets up the VX4469A to send back information about its setup.

Example LS<tm>

sets up the module to return its current setup as follows:

```
TERMINL 0  TERMINL 1  TERMINL 2<cr><lf>
  INDEP      INDEP      INDEP
    or         or         or  <cr><lf>
  BLOCK      BLOCK      BLOCK
  ALTMODE    ALTMODE    ALTMODE
    or         or         or    <cr><lf>
  NOTALTM    NOTALTM    NOTALTM
  ENABLE     ENABLE     ENABLE
    or         or         or    <cr><lf>
  DISABLE    DISABLE    DISABLE
  BREQENB    BREQENB    BREQENB
    or         or         or    <cr><lf>
  BREQDIS    BREQDIS    BREQDIS
TI   ti0        ti1        ti2<cr><lf>
TG   tg0        tg1        tg2<cr><lf>
SG   sg0        sg1        sg2<cr><lf>
CID cid0       cid1       cid2<cr><lf>
THD 1000 MV    1000 MV    1000 MV
    or         or         or   <cr><lf>
THD  700 MV     700 MV     700 MV
XPP xpp0       xpp1       xpp2<cr><lf>
VXITRG igre     igre       igre<cr><lf>
EXTTRG igre     igre       igre<cr><lf>
  ENABLE      ENABLE     ENABLE
ETS  or          or         or    <cr><lf>
  DISABLE     DISABLE    DISABLE
DATA RADIX HEX
                 or    <cr><lf>
DATA RADIX DECIMAL
COMMAND RADIX HEX
                 or    <cr><lf>
COMMAND RADIX DECIMAL
TIME STAMP CLOCK PERIOD xxxx<cr><lf>
OVERLOAD TIMER yyyy<cr><lf>
```

```
ERROR INTERRUPT ENABLED
                or  <cr><lf>
ERROR INTERRUPT DISABLED
VECTOR INTERRUPTS ENABLED
                or  <cr><lf>
VECTOR INTERRUPTS DISABLED
ERROR FORMAT NORMAL
                or  <cr><lf>
ERROR FORMAT BRIEF
<sp><cr><lf>
```

where:

INDEP	independent mode. See SMI command.
BLOCK	block mode. See SMB command.
ALTMODE	alternate mode. See SMA command.
NOTALTM	not alternate mode. See SMN command.
ENABLE	terminal IC is enabled. See SE command.
DISABLE	terminal IC is disabled. See SD command.
BREQENB	bus request enabled. See SBE command.
BREQDIS	bus request disabled. See SBD command.
ti	the transmit interval, 0 to 07f (127_{10}).
tg	the terminal gap, 0 to 07f (127_{10}).
sg	the synch gap, 10, 20, 40, or 07f (127_{10}).
cid	the channel ID, 0 to 0f, or ‘L’ if the high order four bits of the label field of the transmit personality PROM cell is to be used for the channel ID.
THD	SIM receive threshold voltage. See SSTH, SSTL.
xpp	Xpp segment currently in use. See SP
igre	the currently enabled VXI and external triggers. See SQVxx and SQXxx commands.
ETS	status of enabling time stamping end of wordstrings. See the SW command.
DATA RADIX	see SDRH, SDRD commands.
COMMAND RADIX	see SCRH, SCRD commands.

TIME STAMP CLOCK PERIOD	see ST command.
OVERLOAD TIMER	see SO command.
ERROR INTERRUPT	see SSE command.
VECTOR INTERRUPTS	see SSV command.
ERROR FORMAT	see SFB, SFN commands.
FHS	see SXD, SXE commands.

NOTE*. Only data for the number of terminals on the module is returned.*

If a terminal is in Test Mode 1, the INDEP/BLOCK field will be replaced with TSTMODE1 and the following fields will be blank:

ALTMODE/NOTALTM
BREQENB/BREQDIS
CID
XPP

If a terminal is in Test Mode 2, the INDEP/BLOCK field will be replace with TSTMODE2 and the following fields will be blank:

ALTMODE/NOTALTM
BREQENB/BREQDIS
TI
TG
SG
CID
XPP

LVI

Syntax `LVI,instruction block number<tm>`

Purpose List vector instructions. Sets up the VX4469A to return the instructions in a particular instruction block.

Description instruction block number is 0 to 0ff (255_{10}).

See *Appendix G* for more information on the use of this command.

Example LVI,56<tm>

Sets up the VX4469A to return the instructions in instruction block 56. The instructions are returned in the following format:

```
BLOCK   FUNC  PRM1  PRM2  PRM3  PRM4  PRM5  PRM6  PRM7<cr><lf>
56      02    00    02  1200                        <cr><lf>
56      01    03    02  1200                        <cr><lf>
56      00                                          <cr><lf>
<sp><cr><lf>
```

The functions and their parameters (PRMx) are defined in *Appendix G*.

LVX

Syntax `LVX,[interrupt vector]<tm>`

Purpose List vector index table. Sets up the VX4469A to return the interrupt vector index table.

Description interrupt vector number is 0 to 1fff (8191_{10}).

See *Appendix G* for more information on the use of this command.

Example LVX,16<tm>

Sets up the VX4469A to return the interrupt vector index table starting at interrupt vector number 16. The first number is the vector interrupt number. The second number is the instruction block number that will be executed if this interrupt vector occurs. Consecutive table values are returned until a new command is given. They are returned in the following format:

```
0016 15
0017 15
0018 13
0019 00
0020 00
```

NRD

Syntax `[t]NRD,start address, number of words<tm> data.`

Purpose Front panel data port read data. Sets up the VX4469A to supply data to its 16-bit register on the front panel.

Description

[t]	an optional terminal number, which must be either 0, 1 or 2. If omitted, the last value specified is used.
start address	the current terminal's shared memory address where the first data supplied is stored.
number of words	0 to 07FFF

After this command is received by the VX4469A, 16-bit data read from the VX4469A front panel will be read from sequential locations in the current terminal's shared memory. Attempted reads after the number of words specified will not be responded to.

Example 1NRD,0,100<tm>

Sets up the VX4469A to transmit 100 words of data to terminal 1 shared memory starting at address 0. See *Appendix E* for a description of the front panel data port handshaking and data signals.

NWD

Syntax `[t]NWD,start address, number of words<tm> data.`

Purpose Front panel data port write data. Sets up the VX4469A to accept data via its 16-bit register on the front panel.

Description

[t]	an optional terminal number, which must be either 0, 1 or 2. If omitted, the last value specified is used.
start address	the current terminal's shared memory address where the first data received will be stored.
number of words	0 to 07FFF

After this command is received by the VX4469A, 16-bit data written to the front panel connector will be put in sequential locations in the current terminal shared memory. Data written after the number of words specified will not be responded to.

Example 1NWD,0,100<tm>

Sets up the VX4469A to receive 100 words of data to terminal 1 shared memory starting at address 0. See *Appendix E* for a description of the front panel data port handshaking and data signals.

RC

Syntax `RC,circular buffer number<tm>`

Purpose Read circular buffer. Sets the VX4469A to return data from a circular buffer.

Description circular buffer number is 0 to 0f.

Once the data from a cell is completely read, that cell is available for new data.

***NOTE**. Data will be discontinuous if the circular buffer is read at a rate less than the terminal IC is writing new data and the circular buffer is full. If timestamp data is included, the high bit of the time indicates data was lost See Appendix G, Vector Instruction Block Functions, function 1.*

If the data radix is decimal, the data will be returned as a signed decimal number for each 16_{10}-bit word. The numbers will be in groups of eight, with the numbers separated by commas, and groups separated by <cr><lf> characters. Cells are separated by <cr><lf> characters.

If the data radix is hexadecimal, the data will be returned as for decimal, only in unsigned hexadecimal numbers.

Example RC,7<tm>

returns all data from circular buffer 7.

RCC

Syntax `RCC[T],cbnum,label<tm>`

Purpose Read circular buffer calculating CRC. The Read Circular buffer CRC command sets up the VX4469A to return data from a circular buffer. A CRC is calculated and is returned after the data from each cell.

Description

[T]	optional, indicates the first three words of each cell are timestamp data to be skipped when calculating the CRC.
cbnum	the circular buffer number.
label	a 16-bit number that includes the channel ID in the high four bits.

The VX4469A is set up to return data from circular buffer ‘cbnum’. For each cell returned, a CRC is calculated on ‘label’ and the words in the cell. The circular buffer words are returned with the calculated CRC. The CRC is not written; it is only appended to the end of the returned data. If the optional T is added to the command, the first three words of each cell (usually timestamp data) are ignored when calculating the CRC. See *Appendix G* for more information about the VX4469A and CRC.

Example RCC,4,3fd7<tm>

Reads data from circular buffer 4. Using 3 as the channel ID and fd7 as the label, a CRC is calculated and returned after the last data word. Assuming the circular buffer cell size is 4 and the data words were 1, 2, 3, and 258, the following would be returned from the VX4469A if it were read:

`0001,0002,0003,0258 0000<cr><lf>`

RCCT,5,3fd7<tm>

Reads data from circular buffer 5. Using 3 as the channel ID and fd7 as the label, a CRC is calculated and returned after the last data word. Assuming the circular buffer cell size is 7 and contains timestamp information and the data words were 1, 2, 3, and 258, the following would be returned from the VX4469A if it were read:

`0fd7,34D7,0036,0001,0002,0003,0258 0000<cr><lf>`

RCV

Syntax `RCV[T],cbnum,label<tm>`

Purpose Read circular buffer with variable wordstring length, calculating CRC. The Read Circular buffer CRC command sets up the VX4469A to return data from a circular buffer. A CRC is calculated and is returned after the data from each cell.

Description

[T]	optional, indicates the first three words of each cell are timestamp data to be skipped when calculating the CRC.
cbnum	the circular buffer number.
label	a 16-bit number that includes the channel ID in the high four bits.

The VX4469A is set up to return variable length data from circular buffer ‘cbnum’. The number of words returned is determined by the first data word in the cell. For each cell returned, a CRC is calculated on ‘label’ and the words in the cell. The circular buffer words are returned with the calculated CRC. The CRC is not written; it is only appended to the end of the returned data. If the optional T is added to the command, the first three words of each cell (usually timestamp data) are ignored for the number of data words and when calculating the CRC. See *Appendix G* for more information about the VX4469A and CRC.

Example RCV,2,314<tm>

Sets up the VX4469A to return data from circular buffer 2. Using 0 as the channel ID and 314 as the label, a CRC is calculated and returned after the last data word. Assuming the circular buffer cell size is 7 and the data words were 4, 2, 3, 9182, 5, 6, and 7, the following would be returned from the VX4469A if it were read:

`0004,0001,0002,9182 0000<cr><lf>`

RCVT,3,314<tm>

Sets up the VX4469A to return data from circular buffer 3. Using 0 as the channel ID and 314 as the label, a CRC is calculated and returned after the last data word. Assuming the circular buffer cell size is A and the timestamp and data words were 314, 44D6, 778,4, 2, 3, 9182, 5, 6, and 7, the following would be returned from the VX4469A if it were read:

`0314,44D6,0778,0004,0001,0002,9182 0000<cr><lf>`

RD

Syntax `[t]RD,start address<tm>`

Purpose The Read command reads data.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the data space is by 16_{10}-bit word and not by byte.
D	Data. If the data radix (see Set Radix command) is decimal, the data will be returned as a signed decimal number for each 16_{10}-bit word. The numbers will be in groups of eight, with the numbers separated by commas, and groups separated by <cr><lf> characters. Each group is preceded with the address of the first word in that group. If the data radix (see Set Radix command) is hexadecimal, the data will be returned as for decimal, only in unsigned hexadecimal numbers. Each group is preceded with the address of the first word in that group.

Example RD,350<tm>

sets up the module to return data from the current terminal starting from address 350. Data will continue to be returned until the end of shared memory is reached or a new command is sent.

```
0350 0000,0000,0000,0000,0000,0000,0000,0000<cr><lf>
0358 0000,0000,0000,0000,0000,0000,0000,0000<cr><lf>
0360 0000,0000,0000,0000,0000,0000,0000,0000<cr><lf>
0368 0000,0000,0000,0000,0000,0000,0000,0000<cr><lf>
0370 0000,0000,0000,0000,0000,0000,0000,0000<cr><lf>
```

The first number of each line (0350, 358, 360, etc.) is the address in shared memory of the first data word on that line.

RDC

Syntax `RDC,addr,label,count<tm>`

Purpose The Read Data CRC from shared memory command sets up the VX4469A to return data from the current terminal's shared memory. A CRC is calculated and appended to the returned data.

Description

addr	the starting address in shared memory.
label	a 16-bit number that includes the channel ID in the high 4 bits.
count	number of shared memory data words to return.

'Label' and 'count' shared memory words are used to calculate the CRC. 'Count' shared memory words are returned, followed by the calculated CRC. The CRC is not written to shared memory; it is only appended to the end of the returned data. See *Appendix G* for more information about CRC.

Example 2RDC,400,1ff0,6<tm>

Reads data from terminal 2's shared memory starting at address 400. Using 1 as the channel ID and ff0 as the label, a CRC is calculated and returned after the last data word. Assuming the length of the data is 6 and the data words were 1,2,3,4,5 and 7B0F, the following would be returned from the VX4469A if it were read:

```
0400 0001,0002,0003,0004,0005,7B0F 0000<cr><lf>
```

The first number, 0400, is the address in shared memory of the first data word on that line.

RDV

Syntax `RDV,addr,label<tm>`

Purpose The Read Data Variable from shared memory command calculates and appends a CRC, using the word at ‘addr’ in shared memory as the ‘number of data words’ in the wordstring.

Description

addr	the starting address in shared memory.
label	a 16-bit number that includes the channel ID in the high 4 bits.

‘Label’ and ‘number of data words’ shared memory words are used to calculate the CRC. ‘Number of data words’ shared memory words are returned with the calculated CRC. The CRC is not written to shared memory, it is only appended to the end of the returned data.

Example 0RDV,300,ff7<tm>

Reads data from terminal 0’s shared memory starting at location 300. Using 0 as the channel ID and ff7 as the label, a CRC is calculated and returned after the last data word. The word at 300 in shared memory is used as the number of words in the wordstring. Assuming the data words starting at 300 were 3, 2, and EA83, the following would be returned from the VX4469A if it were read:

```
0300 0003,0002,EA83 0000<cr><lf>
```

The first number, 0300, is the address in shared memory of the first data word, 0003, on that line.

RG

Syntax `[t]RG<tm>`

Purpose The Read Registers command sets up the VX4469A to read and return the contents of the terminal's IC three internal registers, Error, Last word and Interrupt vector.

Description

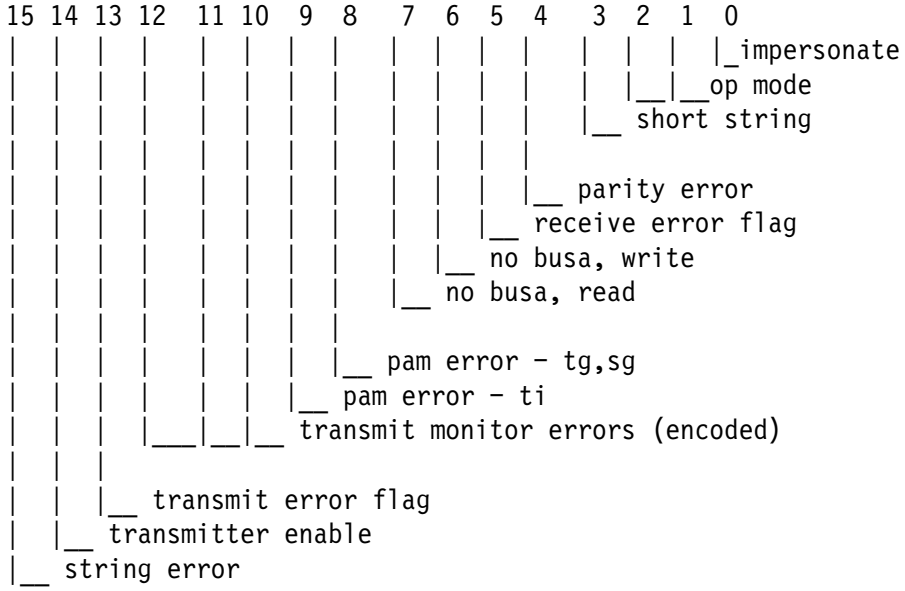
[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

The three registers are read at the time data is requested from the VX4469A. They are read one at a time, so it is possible that the registers will change while being read. The format of the data returned if the data radix is decimal or hex is as follows:

aaaa,bbbb,cccc<cr><lf>

where aaaa is the Error register, bbbb is the last word monitor, and cccc is the Interrupt Vector register.

The Error register bits are as follows:



The last word monitor register contains the last label or data word sent for a transmit operation and the last label word received for a receive operation.

The interrupt vector register (IVR) contains the 15-bit interrupt vector for the current received or transmitted word-string. The most significant bit of the IVR

contains the string error bit. The IVR makes available the value of the current interrupt vector at any time whether or not the personality PROMS have been programmed to provide interrupt vector strobes.

A more detailed description of the Error register is provided in *Appendix F*.

RI

Syntax `RI<tm>`

Purpose Read system Interrupt register. The interrupt register contains bits corresponding to different interrupt conditions.

Description Sending the RI command sets up the VX4469A for returning an 8-bit character. The high order bit, bit 7, is always 0, bit 6 is always a 1, and bits 5 through 0 are defined as follows:

Bit 7	Always 0
Bit 6	Always 1
Bit 5	New error(s) in error queue.
Bit 4	CRC error - vector interrupt function d
Bit 3	Vector interrupt function b, c, or f, 8
Bit 2	Vector interrupt function b, c, or f, 4
Bit 1	Vector interrupt function b, c, or f, 2
Bit 0	Vector interrupt function b, c, or f, 1

Reading the register clears the register. The bits in the register are always set, independent of whether the interrupts are enabled or disabled. Refer to *Appendix G* for a description of vector interrupt functions.

Example RI<tm>

This sets up the VX4469A to return a character that represents the state of the Interrupt status register. The VX4469A will continue sending this status until another command is sent. Examples of what might be returned are:

b<cr><lf>	An error was placed in the error queue and at least 1 interrupt vector function b, c, or f,2 was executed since the status was last read.
@<cr><lf>	No errors or interrupt vector functions b, c, or f have occurred since the status was last read.
A<cr><lf>	At least 1 interrupt vector function b, c, or f,1 was executed.
D<cr><lf>	At least 1 interrupt vector function b, c, or f,4 was executed.

The following table shows the characters returned for all the possible bit combinations:

Table 3–1: Interrupt Register Coding

char	hex	error	CRC	intrp8	intrp4	intrp2	intrp1
@	40						
A	41						x
B	42					x	
C	43					x	x
D	44				x		
E	45				x		x
F	46				x	x	
G	47				x	x	x
H	48			x			
I	49			x			x
J	4a			x		x	
K	4b			x		x	x
L	4c			x	x		
M	4d			x	x		x
N	4e			x	x	x	
O	4f			x	x	x	x
P	50		x				
Q	51		x				x
R	52		x			x	
S	53		x			x	x
T	54		x		x		
U	55		x		x		x
V	56		x		x	x	
W	57		x		x	x	x
X	58		x	x			
Y	59		x	x			x
Z	5a		x	x		x	
[	5b		x	x		x	x
\	5c		x	x	x		
]	5d		x	x	x		x
^	5e		x	x	x	x	
_	5f		x	x	x	x	x

Table 3–1: Interrupt Register Coding (Cont.)

char	hex	error	CRC	intrp8	intrp4	intrp2	intrp1
‘	60	x					
a	61	x					x
b	62	x				x	
c	63	x				x	x
d	64	x			x		
e	65	x			x		x
f	66	x			x	x	
g	67	x			x	x	x
h	68	x		x			
i	69	x		x			x
j	6a	x		x		x	
k	6b	x		x		x	x
l	6c	x		x	x		
m	6d	x		x	x		x
n	6e	x		x	x	x	
o	6f	x		x	x	x	x
p	70	x	x				
q	71	x	x				x
r	72	x	x			x	
s	73	x	x			x	x
t	74	x	x		x		
u	75	x	x		x		x
v	76	x	x		x	x	
w	77	x	x		x	x	x
x	78	x	x	x			
y	79	x	x	x			x
z	7a	x	x	x		x	
{	7b	x	x	x		x	x
|	7c	x	x	x	x		
}	7d	x	x	x	x		x
~	7e	x	x	x	x	x	
DEL	7f	x	x	x	x	x	x

RM

Syntax `[t]RM,start address<tm>`

Purpose The Read Multiple command reads the current terminal's multiple personality PROM.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the personality PROM space is by 8-bit byte. The terminal must be disabled to read or write its personality PROM.

If the data radix (see Set Radix command) is decimal, the data will be returned as an unsigned decimal number for each 8-bit byte. The numbers will be in groups of eight. The numbers are separated by commas and groups are separated by <cr><lf> characters. Each group will be preceded with the address of the first byte of that block.

If the data radix (see Set Radix command) is hexadecimal, the data will be returned as for decimal, only in hexadecimal numbers.

Example 0RM,40<tm>

sets up to read back terminal 0's multiple personality PROM starting at address 40.

```
0040 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0048 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0050 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
```

RR

Syntax `[t]RR,start address<tm>`

Purpose The Read Receive command reads the current terminal's receive personality PROM.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the personality PROM space is by 8-bit byte. The terminal must be disabled to read or write its personality PROM.

If the data radix (see Set Radix command) is decimal, the data will be returned as an unsigned decimal number for each 8-bit byte. The numbers will be in groups of eight. The numbers are separated by commas and groups are separated by <cr><lf> characters. Each group will be preceded with the address of the first byte of that block.

If the data radix (see Set Radix command) is hexadecimal, the data will be returned as for decimal, only in hexadecimal numbers.

Example 0RR,60<tm>

sets up to read back terminal 0's receive personality PROM starting at address 60.

```
0060 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0068 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0070 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
```

Data will continue to be returned until another command is sent to the module.

RS

Syntax `[t]RS<tm>`

Purpose Read the current terminal's Serial Interface Module (SIM) status register.

Description

[t]	an optional terminal number, which must be 0, 1 or 2. If omitted, the last value specified is used.

This command sets up the VX4469A to return the contents of the SIM status register. The TS (Test SIM) command is normally performed prior to using this command.

Example RS<tm>

The following list shows possible responses returned if the VX4469A is read following this command:

0<cr><lf> coupler fault.

1<cr><lf> 1 good coupler channel.

2<cr><lf> 2 good coupler channels.

3<cr><lf> SIM fault or no test.

***NOTE**. This data is undefined if a Pseudo Bus Module is used in place of a SIM.*

See *Pseudo Bus* in *Appendix G* for more information about SIMs.

RT

Syntax `[t]RT,segment,start address<tm>`

Purpose The Read Test Ram command sets up the VX4469A to return the contents of the current terminal's Test Ram.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
segment	Test Ram segment.
start address	the byte address within the Test Ram segment.

Note that the addressing in the Test Ram space is by 8-bit byte. The terminal must be in Test Mode and disabled to read or write its Test Ram.

If the data radix (see Set Radix command) is decimal, the data will be returned as an unsigned decimal number for each 8-bit byte. The numbers will be in groups of eight. The numbers are separated by commas and groups are separated by <cr><lf> characters. Each group will be preceded with the address of the first byte of that block.

If the data radix (see Set Radix command) is hexadecimal, the data will be returned as for decimal, only in hexadecimal numbers.

Example 0RT,0,7<tm>

sets up to read back terminal 0's Test Ram segment 0 starting at address 7.

```
0007 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
000E FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0017 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
```

RX

Syntax `[t]RX,start address<tm>`

Purpose The Read Xmit command reads the current terminal's Transmit Personality PROM.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the personality PROM space is by 8-bit byte. The terminal must be disabled to read or write its Personality PROM.

If the data radix (see Set Radix command) is decimal, the data will be returned as an unsigned decimal number for each 8-bit byte. The numbers will be in groups of eight. The numbers are separated by commas and groups are separated by <cr><lf> characters. Each group will be preceded with the address of the first byte of that block.

If the data radix (see Set Radix command) is hexadecimal, the data will be returned as for decimal, only in hexadecimal numbers.

Example 0RX,0<tm>

sets up to read back terminal 0's Transmit Personality PROM starting at address 0.

```
0000 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0008 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
0010 FF,FF,FF,FF,FF,FF,FF,FF<cr><lf>
```

SBD

Syntax `[t]SBD<tm>`

Purpose The Set Bus Disable command disables the terminal IC from accessing the shared memory.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Using this command results in the terminal IC transmitting data with a parity error.

Example 2SBD<tm>

disables the terminal IC for terminal 2 from accessing shared memory. The last word read from shared memory will be transmitted with a parity error in place of all data the terminal IC would normally have read from shared memory.

SBE

Syntax [t]SBE<tm>

Purpose The Set Bus Enable command enables the terminal IC to access the shared memory.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

This is the normal operating mode. See the SBD command for further discussion.

Example 2SBE<tm>

enables the terminal IC for terminal 2 to access shared memory.

SC

Syntax `[t]SC,c<tm>`

Purpose The Set Channel ID command sets the channel ID inputs to the terminal IC.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
c	specifies the channel ID, with a valid range of 0 to 15_{10}. The channel ID is placed in the four high bits of all labels transmitted from a terminal. The channel ID is also used to index into the multiple personality PROM for an address offset for received data.

See also the SCL command.

Example 1SC,9<tm>

sets terminal 1's Channel ID to 9.

SCL

Syntax `[t]SCLn<tm>`

Purpose Set the Channel ID to use or not use the XPP label field for the channel ID.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	must be either E or D: E use the XPP label field D do not use the XPP label field

If SCLE is used, the channel ID is set from the high four bits of the label field in each cell of the Transmit Personality PROM. This allows you to specify the channel ID for each word string that is transmitted.

The channel ID is set at the end of the previously transmitted string. If the terminal is receiving data also, you will need to program the multiple personality PROM to allow for the changing channel IDs.

SCLD disables this feature and the channel ID reverts to its programmed value before the SCLE command.

Example 1SCLE<tm>

Sets terminal 1’s Channel ID to be the high four bits of the XPP label field.

SD

Syntax `[t]SD[A]<tm>`

Purpose The Set Disable command disables the specified terminal(s).

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
[A]	if the letter 'A' is added to the command, all terminals are disabled.

Example

2SD<tm> Disables terminal 2.

SDA<tm> All terminals are disabled.

SE

Syntax `[t]SE[A]<tm>`

Purpose The Set Enable command enables the specified terminal(s).

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
[A]	if the letter 'A' is added to the command, all terminals are enabled simultaneously.

Example

1SE<tm> Enables terminal 1.

SEA<tm> All terminals are enabled.

SF

Syntax SFn<tm>

Purpose The Set error Format command determines the way error messages are stored in the error queue.

Description

m	must be either N or B: N Descriptive messages are stored in the error queue. B brief – A single character is stored in the error queue for each message.

Example

SFN<tm> Descriptive messages are store in the error queue.

SFB<tm> Single character messages are stored in the error queue.

See the LE command and *Memory Switching* in *Appendix G* for related information.

SG

Syntax `[t]SG,m<tm>`

Purpose The Set seGment is a Test Mode command that writes a Test Ram segment number to a terminal's segment register.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
m	Test Ram segment number in the range of 0 to f.

Example 1SG,5<tm>

Sets terminal 1's Test Ram segment register to 5. Terminal 1 must be in Test Mode to use this command.

SH

Syntax SHm<tm>

Purpose The Set Memory Switching Mode command is used to select between invert and OR mode.

Description

m	must be either O or S:	
	O	OR mode; only the low half of shared memory is switched to high. The high half of memory remains high.
	S	invert mode; the low half of shared memory is switched high and the high half of memory is switched low.

Example SHO<tm>

When memory is switched, the high bit of the shared memory for the switched terminal is always one for the terminal IC.

See *Appendix G* for a general discussion of memory switching.

See also commands HPN, HN, HPS, HR, HS, and LH.

SI

Syntax `[t]SI,ti,tg,sg<tm>`

Purpose The Set Interval command sets the ARINC 629 parameters TI (transmit internal), TG (terminal gap), and SG (sync gap).

Description This command sets the output of a hardware register connected to input pins on the terminal IC.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
ti	the Transmit Interval (TI) value. TI has a range of 0 to 07f (127_{10}) representing a time of (500_{10}* (ti + 1)) + $.5625_{10}$ microseconds.
tg	the Terminal Gap (TG) value. It has a range of 2 to 126_{10} representing a time of tg + 1.6875_{10} microseconds.
sg	the Sync Gap (SG) value. It has possible values of 16_{10}, 32_{10}, 64_{10} or 127_{10} representing sg + 1.6875_{10} microseconds.

The terminal IC will compare these inputs with values in the receive personality PROM. If different, the terminal will only receive, not transmit. See the UWR command.

The transmit function of a terminal may be turned off and on by using this command to make the values different or the same as the values in the receive personality PROMs.

Example 0SI,9,4,10<tm>

sets terminal 0's timers to

TI = 5000.5625_{10} μsec
TG = 5.6875_{10} μsec
SG = 17.6875_{10} μsec

SKD

Syntax `[t]SKD<tm>`

Purpose The Set Kollision Disable command disables VXI TTL trigger, External trigger or label enables on a terminal in test mode 2.

Description This command disables VXI TTL trigger, External trigger or label enables on a terminal in test mode 2 if they were previously enabled.

This command is valid only in test mode 2.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example 0SKD<tm>

SKI

Syntax `[t]SKI<tm>`

Purpose The Set Kollision Immediate command causes a terminal in test mode 2 to transmit immediately.

Description If a terminal is in test mode 2, it is programmed with data to transmit and it is set enabled (SE command), this command will cause the terminal to transmit the next set of data in it's test ram.

This command is valid only in test mode 2.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example 0SKI<tm>

SKL

Syntax `[t]SKL,b,c<tm>`

Purpose The Set Kollision Label command causes a terminal in test mode 2 to transmit after it receives a particular label.

Description If a terminal is in test mode 2, it is programmed with data to transmit and it is set enabled (SE command), this command will cause the terminal to transmit the next set of data in it's test ram each time the terminal receives a particular label.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
b	is the label including CID that is to trigger this terminal to transmit test data.
c	is a bit time count to wait after detecting the label until the test data is transmitted. The terminal will transmit the test data start 2 + c bit times after parity bit of the label b.

This command is valid only in test mode 2.

Example 0SKL,5034,%30<tm>

SKV

Syntax `[t]SKV,c<tm>`

Purpose The Set Kollision VXI TTL command causes a terminal in test mode 2 to transmit after it receives a VXI TTL trigger.

Description If a terminal is in test mode 2, it is programmed with data to transmit and it is set enabled (SE command), this command will cause the terminal to transmit the next set of data in its test RAM each time the terminal receives a VXI TTL trigger.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
c	is a bit time count to wait after receiving the VXI TTL trigger until the test data is transmitted. The terminal will transmit the test data start 1 + c bit times after the negative going edge of the VXI TTL trigger. Triggers received while the terminal is transmitting will be ignored.

This command is valid only in test mode 2.

Example 0SKV,%30<tm>

SKX

Syntax `[t]SKX,c<tm>`

Purpose The Set Kollision eXternal command causes a terminal in test mode 2 to transmit after it receives an external trigger.

Description If a terminal is in test mode 2, it is programmed with data to transmit and it is set enabled (SE command), this command will cause the terminal to transmit the next set of data in its test RAM each time the terminal receives an external.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
c	is a bit time count to wait after receiving the external trigger until the test data is transmitted. The terminal will transmit the test data start 1 + c bit times after the negative going edge of the external trigger. Triggers received while the terminal is transmitting will be ignored.

This command is valid only in test mode 2.

Example 0SKV,%30<tm>

SL

Syntax `[t]SL,m<tm>`

Purpose The Set Logic command sets a terminal to Test mode or Terminal IC mode.

Description This command changes the programming of a terminal LCA for operation in either Terminal IC mode or Test Mode.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
m	0 for Terminal IC mode, or 1 for Test mode 1, or 2 for Test mode 2.

Example 0SL,1<tm>

sets terminal 0's logic mode to Terminal IC mode.

SMA

Syntax [t]SMA<tm>

Purpose The Set Mode Alternate sets the alternate mode pin on the terminal IC true.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

This command allows switching the terminal IC from Block mode to Alternate mode.

Example 0SM<tm>

sets the alternate bit on terminal 0's terminal IC.

SMB

Syntax `[t]SMB<tm>`

Purpose The Set protocol transmit Mode to Block command sets the protocol transmit mode to Block.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example 1SMB<tm>

sets the protocol transmit mode for terminal 1 to Block.

SMI

Syntax `[t]SMI<tm>`

Purpose The Set protocol transmit Mode to Independent command sets the protocol transmit mode to Independent.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example 1SMI<tm>

sets the protocol transmit mode for terminal 2 to Independent.

SMN

Syntax [t]SMN<tm>

Purpose The Set Mode Normal sets the alternate mode pin on the terminal IC false. This command allows switching the terminal IC from alternate mode to block mode.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example 0SMN<tm>

sets the alternate bit on terminal 0's terminal IC.

SO

Syntax SO,m<tm>

Purpose The Set Overload command sets an overload timer on the 80186 processor.

Description

m	time in 5 millisecond increments.

If the 80186 is constantly processing interrupt vector instructions for longer than the overload timer command, an overload error is generated. If this timer is set for the shortest TI time of the terminals on the board, this error will occur if the board is overloaded with interrupt vector instructions.

This timer can only be set in 5 millisecond increments (for example, 5, 10, 15, 20). Numbers in between will be rounded off to multiples of 5. See also the LS command.

Example SO,5<tm>

sets the overload timer to 5 milliseconds.

SP

Syntax `[t]SP,m<tm>`

Purpose The Set Xpp segment command selects which of 4 Xpp segments to use for programming, reading or execution.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
m	Xpp segment 0, 1, 2 or 3.

If the terminal is disabled, this command selects which Xpp segment to use for writes or reads and which segment will be used when the terminal is enabled.

If the terminal is enabled, this command will change the Xpp segment the the terminal IC is using. The first CLRX pulse from the terminal IC after this command is executed will cause the switch to take place. The CLRX pulse occurs each time the terminal IC finishes transmitting a message.

Example SP,2<tm>

selects Xpp segment 2.

SQVE

Syntax `[t]SQVEn<tm>`

Purpose Enable/disable the terminal pulsing its VXI TTL trigger line on a communication error.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a VXI TTL trigger capability. This command allows enabling or disabling pulsing this VXI TTL trigger if a communication error occurs for this terminal. If this trigger condition is enabled and occurs, it will lock out all other trigger conditions for this terminal until the communication error is cleared by the LE command.

Example SQVEE<tm>

This command enables this terminal to trigger on a communication error.

SQVG

Syntax `[t]SQVGn<tm>`

Purpose Enable/disable the terminal pulsing its VXI TTL trigger on beginning to transmit.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a VXI TTL trigger capability. This command enables or disables this terminal pulsing its VXI TTL trigger whenever the terminal begins to transmit.

Example SQVGE<tm>

This command enables this terminal to trigger whenever it begins to transmit.

SQVI

Syntax `[t]SQVIn<tm>`

Purpose Enable/disable the terminal pulsing its VXI TTL trigger on interrupt vector bit 13.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a VXI TTL trigger capability. This command allows enabling or disabling pulsing this VXI TTL trigger if an interrupt vector occurs for this terminal with bit 13 set.

Example SQVIE<tm>

This command enables this terminal to trigger on an interrupt vector that has bit 13 set.

SQVR

Syntax `[t]SQVRn<tm>`

Purpose Enable/disable the terminal to pulse its VXI TTL trigger if its interrupt vector bit 13 is on and it has just received a wordstring that does not have a valid CRC.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a VXI TTL trigger capability. This command allows enabling or disabling pulsing this VXI TTL trigger if an interrupt vector occurs for this terminal with bit 13 set and the terminal has just received a word string that does not have a valid CRC.

Example SQVRE<tm>

This command enables this terminal to trigger on an interrupt vector that has bit 13 set and it has just received a wordstring without a valid CRC.

SQXE

Syntax `[t]SQXEn<tm>`

Purpose Enable/disable the terminal to pulse its external trigger on a communication error.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a external trigger capability. This command allows enabling or disabling pulsing this external trigger if a communication error occurs for this terminal. If this trigger condition is enabled and occurs, it will lock out all other trigger conditions for this terminal until the communication error is cleared by the LE command.

Example SQXEE<tm>

This command enables this terminal to trigger on a communication error.

SQXG

Syntax `[t]SQXGn<tm>`

Purpose Enable/disable the terminal to pulse its external trigger on beginning to transmit.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a external trigger capability. This command allows enabling or disabling pulsing this external trigger whenever this terminal begins to transmit.

Example SQXGE<tm>

This command enables this terminal to trigger whenever it begins to transmit.

SQXI

Syntax `[t]SQXIn<tm>`

Purpose Enable/disable the terminal to pulse its external trigger on interrupt vector bit 13.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a external trigger capability. This command allows enabling or disabling pulsing this external trigger if an interrupt vector occurs for this terminal with bit 13 set.

Example SQXIE<tm>

This command enables this terminal to trigger on an interrupt vector that has bit 13 set.

SQXR

Syntax `[t]SQXRn<tm>`

Purpose Enable/disable the terminal to pulse its external trigger on interrupt vector bit 13 when this terminal has just received a wordstring that does not have a valid CRC.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

Each terminal on a VX4469A has a external trigger capability. This command allows enabling or disabling pulsing this external trigger if an interrupt vector occurs for this terminal with bit 13 set and the terminal has just received a word string that does not have a valid CRC.

Example SQXRE<tm>

This command enables this terminal to trigger on an interrupt vector that has bit 13 set and it has just received a wordstring without a valid CRC.

SR (SRDD, SRDH, SRCD, SRCH)

Purpose The Set Radix command sets the radix of numeric data or command parameters. This determines the format of numeric data or command parameters.

Description Set radix of numeric data:

SRDD<tm> – Set data radix to decimal.

SRDH<tm> – Set data radix to hexadecimal. (default)

Set radix of command parameters:

SRCD<tm> – Set command radix to decimal.

SRCH<tm> – Set command radix to hexadecimal. (default)

The radix of a particular number may also be specified by preceding the number with # for hexadecimal or % for decimal.

Example

SRDD<tm>	Sets the data radix to decimal.
SRCH<tm>	Sets the command radix to hexadecimal.

SSEn

Syntax `SSEn<tm>`

Purpose Set system errors interrupt enable/disable.

Description

n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

The VX4469A can interrupt the system controller on any condition that causes a message to be placed in the error queue and lights the error LED.

Example SSED<tm>

This command disables interrupts from errors.

SSTn

Syntax `[t]SSTn<tm>`

Purpose Set the SIM threshold either high or low.

Description

[t]	an optional terminal number, which must be 0, 1 or 2. If omitted, the last value specified is used.
n	a single letter which must be either H or L: H set threshold high (1000_{10} mv) L set threshold low (700_{10} mv)

***NOTE**. This command has no effect if a Pseudo Bus Module is used in place of a SIM.*

Example SSTH<tm>

This command sets the current terminal SIM's receive threshold to 1000_{10} mV.

SSVn

Syntax `SSVn<tm>`

Purpose Enable or disable the system's vector interrupts.

Description

n	a single letter, which must be either E or D: E enable interrupts D disable interrupts

The VX4469A can interrupt the system controller when function f is executed in an instruction block.

Example SSVD<tm>

This command disables the system's vector interrupts. See *Vector Instruction Block Functions* in *Appendix G*, function f.

ST

Syntax ST,n<tm>

Purpose The Set Time stamp command sets the time-stamp clock period.

Description The range for the value of n is 1 to 07fff (32767_{10}) microseconds, in one microsecond steps. The clock is also set to 0 and enabled when this command is issued.

Example ST,%10<tm>

This command sets the time stamp clock period to 10 μs.

SVI

Syntax

```
SVI,instruction block number,function,arguments[,function,
arguments]<tm>
```

Purpose

Set vector instructions. This command sets up a list of commands to be executed whenever a particular vector or vectors is/are generated.

Description

instruction block number	0 to 0ff (255_{10}). Instruction blocks may be defined or modified at any time but may cause undefined results if this is done while a vector is using the block.
function	0 to 1f.
arguments	The arguments vary with the instructions.

See *Appendix G* for more information on the use of this command.

Example

svi,56,2,0,0,1000,2,0,1,1000<tm>

This command sets up a list of instructions to be contained in instruction block 56.

SVX

Syntax `SVX,vector number,instruction block number<tm>`

Purpose Set vector index. This command specifies which instruction block is to be used by each vector.

Description

vector number	0 to 1fff (8191_{10}).
instruction block number	0 to 0ff (255_{10}).

Example SVX,43,17<tm>

Sets VX4469A so that when vector 43 occurs, the instructions in instruction block 17 are executed.

SW

Syntax `[t]SWn<tm>`

Purpose Set the terminal to timestamp the end of a wordstring.

Description

n	a single letter, which must be either E or D: E enable end of wordstring timestamp. D disable end of wordstring timestamp.

If the Rpp interrupt vector has bit 13 on for a label, a timestamp will be generated with that interrupt vector number about 2.5 bit times after the end of the wordstring.

This command is useful for determining the length of a received wordstring. If the length of a wordstring is unknown, the Rpp is set up to receive the label with no data. In this case an early interrupt vector will occur 3 bit times after the label, a late interrupt vector will occur 3.5 bit times after the label whether there is data in the wordstring or not. The SWE command will enable generating an interrupt vector 2.5 bit times after the end of the word string whether it is just a label or a label with data. Thus the difference in time between a late interrupt vector and an end of wordstring interrupt vector divided by 10 microseconds per word is the number of data words in a wordstring.

Example 2SWE<tm>

Sets VX4469A so that when terminal 2 receives a wordstring with a label Rpp interrupt vector with bit 13 on, an interrupt vector is generated.

TQSV

Syntax `TQSV<tm>`

Purpose Test the VX4469A's software VXI TTL trigger.

Description This command causes a negative pulse on the VX4469A's software VXI TTL trigger line.

TQSX

Syntax `TQSX<tm>`

Purpose Test the VX4469A's software external trigger.

Description This command causes a negative pulse on the VX4469A's software external trigger line.

TQV

Syntax `[t]TQV<tm>`

Purpose Test a terminal's VXI TTL trigger.

Description

[t]	an optional terminal number, which must be 0, 1 or 2. If omitted, the last value specified is used.

This command causes a negative pulse on a terminal's VXI TTL trigger line.

TQX

Syntax `[t]TQX<tm>`

Purpose Test a terminal's external trigger.

Description

[t]	an optional terminal number, which must be 0, 1 or 2. If omitted, the last value specified is used.

This command causes a negative pulse on a terminal's external trigger line.

TS

Syntax `[t]TS<tm>`

Purpose Test SIM. This command initiates a self test in the Serial Interface Module of the current terminal. See *Pseudo Bus* in *Appendix G* and the RS command for more information.

Description

[t]	an optional terminal number, which must be 0, 1 or 2. If omitted, the last value specified is used.

***NOTE**. This data in undefined if a Pseudo Bus Module is used in place of a SIM.*

This command will probably cause a communication error from the terminal I.C. When the TS command is executed, the Serial Interface Module switches the Current Coupler to test its other set of electronics. The current coupler will probably be transmitting correctly, but the monitored echo coming back from the Current Coupler is usually missing a few bits.

URM

Syntax `[t]URM,p<tm>`

Purpose A user friendly way of reading data from the multiple personality PROM. This command sets the VX4469A to return the contents in an easy to read format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
p	the offset pointer value of 0 to 7f (127_{10}).

Example The following is an example of the format returned by VX4469A. The data radix is hexadecimal. In this example, receive cells with an offset pointer of 1 will store data in shared memory for this terminal at the address stored in the receive personality PROM cell plus the value in the table below. Assume the Personality PROM data address is 100. Data will only be stored if this terminal has a channel ID of 0. Other values of Channel ID will find values of FFFF hex in the table which when added to 100 will be greater than FFFF hex. If the received label has an extension of 3 (the transmitter's CID is three) then the number F hex will be added to 100 (10F hex) for the address to store the data.

URM,1<tm>

```
   EXT   0/8   1/9   2/A   3/B   4/C   5/D   6/E   7/F<cr><lf>
CID00  0000  0005  000A  000F  0014  0019  001E  0023<cr><lf>
  8-F  0028  002D  0032  0037  003C  0041  0046  004B<cr><lf>
CID01  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID02  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID03  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID04  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID05  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID06  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID07  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID08  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CID09  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
```

```
CIDOA  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CIDOB  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CIDOC  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CIDOD  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CIDOE  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
CIDFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
  8-F  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF<cr><lf>
```

URR

Syntax `[t]URR<tm>`

Purpose A user friendly way of reading data from the receive personality PROM. This command sets the VX4469A to return the contents in an easy to read format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example The following is an example of the format returned by VX4469A. The data radix is hexadecimal. The following commands were sent to the VX4469A prior to the URR command.

```
uwr,L,5,2,10<tm>
uwx,0,0,b0,d100,n5,e0,mz<tm>
uwx,0,y0<tm>
uwx,1,0,b10,d120,n6,e10,mz<tm>
uwx,1,1,b11,d140,n2,e11,mz<tm>
uwx,1,y1<tm>
uwx,2,0,b20,e20<tm>
uwx,2,1,b21,d180,n20,e21<tm>
uwx,2,2,b22,d1a0,n10,e22<tm>
uwx,2,y2<tm>
uwr,100,d1000,n5,L100,p1<tm>
uwr,101,d2000,n5,L101,p1<tm>
uwr,102,d3000,n5,L102,p1<tm>
uwm,1,0,i5<tm>
urr<tm>
```

The following was read back from the VX4469A:

```
LABL RPADR BADDR NDATA OSPTR OSADR R/M E VECTOR L, MSC
0000, 0000, FFFF, 0005, 007F, FE00,  R,   7FFF   , 1F
0010, 0080, FFFF, 0006, 007F, FE00,  R,   7FFF   , 1F
0011, 0088, FFFF, 0002, 007F, FE00,  R,   7FFF   , 1F
0020, 0100, FFFF,     , 007F, FE00,  R,   7FFF   , 1F
0021, 0108, FFFF, 0020, 007F, FE00,  R,   7FFF   , 1F
0022, 0110, FFFF, 0010, 007F, FE00,  R,   7FFF   , 1F
0100, 0800, 1000, 0005, 0001, 0200,  M,   0100  L, 00
  EXT   0/8   1/9   2/A   3/B   4/C   5/D   6/E   7/F
CID00  1000  1005  100A  100F  1014  1019  101E  1023
  8-F  1028  102D  1032  1037  103C  1041  1046  104B
LABL RPADR BADDR NDATA OSPTR OSADR R/M E VECTOR L, MSC
0101, 0808, 2000, 0005, 0001, 0200,  M,   0101  L, 00
```

```
  EXT   0/8   1/9   2/A   3/B   4/C   5/D   6/E   7/F
CID00  2000  2005  200A  200F  2014  2019  201E  2023
  8-F  2028  202D  2032  2037  203C  2041  2046  204B
LABL RPADR BADDR NDATA OSPTR OSADR R/M E VECTOR L, MSC
0102, 0810, 3000, 0005, 0001, 0200,  M,   0102  L, 00
  EXT   0/8   1/9   2/A   3/B   4/C   5/D   6/E   7/F
CID00  3000  3005  300A  300F  3014  3019  301E  3023
  8-F  3028  302D  3032  3037  303C  3041  3046  304B
       TI  TG  SG    TI  TG  SG    TI  TG  SG    TI  TG  SG
CID00  05  02  10    05  02  10    05  02  10    05  02  10
CID04  05  02  10    05  02  10    05  02  10    05  02  10
CID08  05  02  10    05  02  10    05  02  10    05  02  10
CID0C  05  02  10    05  02  10    05  02  10    05  02  10
```

LABL — the label number implied by the address of the cell.

RPADR — the address of the cell in the receive personality PROM.

BADDR — the base address of data to write to shared memory. The offset address in the multiple personality PROM will be added to this address to determine the actual address to write data at in shared memory.

NDATA — the number of data words expected with this label. If NDATA is less than the number received, only NDATA will be stored.

OSPTR — the offset pointer to a table in the multiple personality PROM.

OSADR — the address of the offset table OSPTR in the multiple personality PROM.

R/M — indicates whether the terminal IC will look in the receive personality PROM for the offset values or in the multiple personality PROM. The way that the terminal IC is wired to the Personality PROM on the VX4469A will almost always require that the multiple personality PROM be used. It is automatically selected by the UWR command.

E — will show an E if the bit enabling an early vector is set.

VECTOR — the number that will be associated with this cell being transmitted if E and/or L is set.

L — will show an L if the bit enabling a late vector is set.

MSC — a value used by the transmit monitoring function of the terminal IC. If it is non-0 then that label is enabled to be transmitted. If the label is the first to be transmitted in a

message (column 0), then MSC is the maximum number of word strings that will be transmitted in that message.

The table at the end of the read back is the values of ti, tg, and sg for each possible channel ID that is compared to the input pins for ti, tg, and sg on the terminal IC.

URX

Syntax `[t]URX<tm>`

Purpose A user friendly way of reading data from the Transmit Personality PROM. This command sets the VX4469A to return the contents in an easy to read format.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.

Example The following is an example of the format returned by VX4469A when it is in independent mode. The data radix is hexadecimal. The following commands were sent to the module:

```
uwx,0,0,b0,d100,n5,e0,mz<tm>
uwx,0,y0<tm>
uwx,1,0,b10,d120,n6,e10,mz<tm>
uwx,1,1,b11,d140,n2,e11,mz<tm>
uwx,1,y1<tm>
uwx,2,0,b20,e20<tm>
uwx,2,1,b21,d180,n20,e21<tm>
uwx,2,2,b22,d1a0,n10,e22<tm>
uwx,2,y2<tm>
urx<tm>
```

Reading from the module then returned the following data:

```
COL ROW XPADR CID   LBL DADDR EOM NDATA ISHR E VECTOR L MSC MDATA  INDPNDNT
 00  00  0000  00  0000  0100      0005      E  0000    1F  0005
Y MODULO (00F8) 00    Y SYNC (00F9) FF
COL ROW XPADR CID   LBL DADDR EOM NDATA ISHR E VECTOR L MSC MDATA  INDPNDNT
 01  00  0100  00  0010  0120      0006      E  0010    1F  0006
 01  01  0108  00  0011  0140      0002      E  0011    1F  0002
Y MODULO (01F8) 01    Y SYNC (01F9) FF
COL ROW XPADR CID   LBL DADDR EOM NDATA ISHR E VECTOR L MSC MDATA  INDPNDNT
 02  00  0200  00  0020  FFFF   M            E  0020    1F
 02  01  0208  00  0021  0180   M  0020      E  0021    1F  0020
 02  02  0210  00  0022  01A0   M  0010      E  0022    1F  0010
Y MODULO (02F8) 02    Y SYNC (02F9) FF
```

COL the X index of the array of transmit cells.

ROW the Y index of the array of transmit cells.

XPADR the absolute address of the cell.

CID	the channel ID to be transmitted with this cell if the SCL command has been issued.
LBL	the label to be transmitted.
DADDR	the address in the terminal's shared memory where data to be transmitted with this label is located.
EOM	will show an M if this is the last cell in the message (row) to be transmitted.
NDATA	the number of data words to be transmitted with the label. If blank, no data is to be transmitted.
ISHR	will show an I if this cell has its label Channel ID bit on. will show an S if this cell has its sync bit on. will show a H if this cell has its switch memory bit on. will show a R if this cell has its hardware CRC bit on.
E	will show an E if the bit enabling an early vector is set.
VECTOR	the number associated with this cell being transmitted if E and/or L is set.
L	will show an L if the bit enabling a late vector is set.
MSC	the maximum string count. This value is actually in the receive personality PROM cell for label LBL. This number must be non-zero for the terminal IC to transmit this cell. The number associated with the first column of cells needs to be at least as big as the number of cells to be transmitted in this message (until a cell with the EOM bit set is transmitted). This is used by the transmit monitoring portion of the terminal IC.
MDATA	the maximum number of words to be transmitted by this cell. This number is also located in the receive personality PROM cell for label LBL. This is used by the transmit monitoring portion of the terminal IC.
INDPNDNT	the current state of the Independent/Block mode pin and tells how the data in the Transmit Personality PROM was interpreted. If the mode had been BLOCK, then COL and ROW would have been reversed and the order of the data displayed would be by row rather than by column.

If the VX4469A terminal had been in Block mode, the data would look like this:

```
COL ROW XPADR CID   LBL DADDR EOM  DATA ISHR E VECTOR L MSC MDATA  BLOCK
 00  00  0000  01  0000  0100      0005      E  0000    1F  0005
 01  00  0100  02  0010  0120      0006      E  0010    1F  0006
 02  00  0200  04  0020  FFFF   M           E  0020    1F
Y MODULO (F8) 00    Y SYNC (F9) FF   Y ALTMODE (FA) FF
```

UWM

Syntax `[t]UWM,p,r,t,n[,n...]<tm>`

Purpose A user friendly way of writing data to the multiple personality PROM.

Description The multiple personality PROM is 128_{10} arrays that are referred to by the offset pointer in the receive personality PROM cells. Each array is a 16_{10} by 16_{10} matrix of values to add to the data address in the receive personality PROM cell. For each of 16_{10} possible channel ID's the terminal may have, there are 16_{10} values for each of the possible 16_{10} channel ID's the terminal may receive.

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
p	offset pointer value. 0 to 07f (127_{10}).
r	receiver channel ID. r may optionally be Z, which will fill the 16_{10} by 16_{10} array with all ones. r may optionally be Ii, in which case the array will be filled starting with 0 in the first cell and then increasing by i in each cell through the entire array.
t	transmitter channel ID. t may optionally be Z, which will put all ones in the 16_{10} array cells for receiver channel ID r. t may also optionally be Ii, in which case the 16_{10} array cells for receiver channel ID r will be filled starting with 0 and then increasing by i.
n	If the r and t numeric values are specified, then n is put in that cell. Additional values for following cells may be added, separated by a comma.

Example UWM,5,i6<tm>

Fills offset pointer array 5 with offset values starting with 0 and incrementing by 6 in each adjacent cell.

UWM,3,0,i4<tm>

Fills offset pointer array 3, receiver channel ID 0, transmit channel ID 0 to 0f with values starting with 0 and increasing by 4.

UWR,f

Syntax `[t]UWR,f<tm>`

Purpose A user friendly way of writing data to the receive personality PROM.

Description If f is I or L then this command will be writing the Personality PROM transmit interval, terminal gap and sync gap values. The receive personality PROM has a table of ti, tg, and sg values for each of the 16 possible channel ID's this terminal may have. The following syntax applies:

Syntax:
`[t]UWR,(Ii,L),ti,tg,sg<tm>`

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
Ii	ID where i is this terminal's Channel ID. Only channel i's slot in the table will be set to ti, tg, and sg.
L	All channel slots in the table will be set to ti, tg, and sg.

ti, tg, and sg. See the SI command for ranges for these values.

If f specifies a label number, then the (Ii,L), ti, tg, and sg parameters must be omitted. If the value of f is a number between 0 and 4095 then f is a label number and the following syntax applies:

Syntax:
`[t]UWR,b[,Dd][,V[Z]][,N(Z,n)[,E(Z,e)],[,L(Z,l)][,Pp][,Z]<tm>`

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
b	label number. 0 to 0ffb (4091_{10}).
Dd	d = data address. 0 to 0ffff ($65{,}565_{10}$).
V	variable length data. If Z is included, fixed length.
Nn	n = length of data. Z indicates no data.
Ee	e = early vector. Z turns off early vector.
Ll	l = late vector (if both E and L are given, the vector will have the value of the second one) Z turns off late vector.
Pp	p = Offset pointer. 0 to 7f (127_{10}).
Z	fills cell with ones except for transmit monitor function.

Example UWR,i4,2,6,10<tm>

Sets the ti, tg, and sg values in the table for channel ID 4.

UWR,L,2,6,10<tm>

Sets the ti, tg, and sg values in the table for all channel IDs.

UWT

Syntax `[t]UWT,s,m,f...<tm>`

Purpose A user friendly way of writing data to a terminal's Test RAM.

Please refer to *Test Mode* in *Appendix G* for more details of how to use this command and examples.

Description

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used. The terminal must be in Test Mode.
s	Test RAM segment number in the range of 0 to f.
m	B is data is to written at the beginning of segment s or A if data is to be appended to data already written to this segment by the UWT command.
f	is any combination of the following functions separated by commas: Ln[Mm][S][P] — write a label word of n which has a range of 0 to ffff. Mm, if used, will make bit m a missing bit. S, if used, will invert the sync pattern. P, if used, will invert the parity bit. Dn[Mm][S][P] — write a data word of n which has a range of 0 to ffff. Mm, if used, will make bit m a missing bit. S, if used, will invert the sync pattern. P, if used, will invert the parity bit. C[Mm][S][P] — write a data word which is the CRC of the preceding label and data words. Mm, if used, will make bit m a missing bit. S, if used, will invert the sync pattern. P, if used, will invert the parity bit.
Bc[c...]	write bits of value c. c may be 0, 1, H or L. 0 is a normal bit of value 0, 1 is a normal bit of value 1, H is a high level with no transitions and L is a low level with no transitions.
Nn	write a byte to the Test RAM with a value of n. This allows the user to program transition times within a bit time with a resolution of 1/8th of a bit time.
En	terminates a message. The number of Test RAM bytes in the message if put at the beginning of the message and a control byte of value n is place at the end of the message. n = 0 means the next message continues after this one. n = 1 means loop, start with the beginning of the segment for the next message. n = 2 means start a new segment if a SG,x command has been issued, otherwise continue. n = 4 means disable this terminal.

UWX,c,f

Syntax `[t]UWX,c[f]<tm>`

Purpose A user friendly way of writing data to the Transmit Personality PROM.

Description If f is L, S, or Y, then this command will be writing the Personality PROM transmit control cell for column c. The following syntax applies:

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
c	column number.
[f]	two letters, which must be one of the following pairs: Ll l = alternate mode starting row. Ss s = sync mode row. Yy y = y modulo value.

Example UWX,0,y3<tm>

Sets the y modulo for column 0 to 3. (If in block mode, this y modulo is the only one used.) See the Transmit Schedules portion of the *Operation* section near the beginning of the manual for more discussion of y modulo.

If the value of f is a number between 0 and 1e (30_{10}), then f is a row number and the following syntax applies:

[t]UWX,c,r[,Bb][,Dd][,V[Z]][,N(Z,n)[,E(Z,e)][,L(Z,l)][,Cc]

[,M[Z]][,I[Z]][,S[Z]][,H[Z]][,R[Z]][,Z]<tm>

[t]	an optional terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
c	column number.
r	row.
Bb	b = label number. 0 to 0fff plus [(0 to 0f) * 1000] for CID.
Dd	d = data address. 0 to 0ffff (65565_{10}).
V	variable length data. If Z is included, fixed length.
Nn	n = length of data. Z indicates no data.
Ee	e = early vector. Z turns off early vector.

Ll	l = late vector (if both E and L are given, the vector will have the value of the second one) Z turns off late vector.
Cc	c = maximum string length.
M	End of message bit. Z turns it off.
I	Label channel ID bit. Z turns it off.
S	Sync bit. Z turns it off.
H	Switch memory bit. Z turns it off.
R	Hardware CRC bit. Z turns it off.
Z	fills cell with ones except for transmit monitor function.

WC

Syntax `WC,circular buffer number,data<tm>`

Purpose Write circular buffer. Write data to a cell in a circular buffer.

Description

circular buffer number	the circular buffer number the data is to be written to.
data	either in decimal or hexadecimal ASCII characters with comma delimiters.

The full number of words in a cell need not be written. If a terminator <tm> is encountered before the number of words in the cell are received, the cell will contain undefined data in the words not written.

Example WC,4,1,2,3,4,5

Writes the words 1, 2, 3, 4, and 5 to a cell in circular buffer 4.

WCC

Syntax `WCC,cbnum,label,data,...,data<tm>`

Purpose Write data to a circular buffer, calculating and appending a CRC.

Description

cbnum	the circular buffer number, which includes ‘label’ in the calculation.
label	a 16-bit number that includes the channel ID in the high four bits.
data	data words are expected in ASCII characters with comma delimiters.

Example WCC,3,6BC,1,2,3,4

The data 1, 2, 3, and 4 is written to circular buffer 3. A CRC is calculated on 6BC, 1, 2, 3, and 4, and written to circular buffer 3 after the ‘4’.

WD

Syntax `[t]WD,start address,data<tm>`

Purpose The Write Data command writes data in decimal or hexadecimal format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the data space is by 16_{10}-bit word and not by byte.
data	data words are expected in ASCII characters with comma delimiters.

Example WD,120,1,2,3,4,5,6<tm>

writes the word values 1, 2, 3, 4, 5, and 6 to consecutive locations in the current terminal in shared memory, starting at address 120.

WDC

Syntax `WDC,addr,label,data,...,data<tm>`

Purpose The Write Data command calculates and appends a CRC to shared memory, starting at ‘addr’, and including ‘label’ in the CRC calculation.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
addr	the starting address in shared memory.
label	a 16-bit number that includes the channel ID in the high 4 bits.
data	data words are expected in ASCII characters with comma delimiters.

Example 2WDC,101,4F6,1,2,3,4,5<tm>

writes the numbers 1, 2, 3, 4, and 5 to terminal 2 in shared memory, starting at address 101. A CRC is calculated on 4F6, 1, 2, 3, 4, and 5, and is written following the ‘5’ at location 106.

WM

Syntax `[t]WM,start address,data<tm>`

Purpose The Write Multiple command writes the multiple personality PROM in decimal or hexadecimal format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. A terminal must be disabled to read or write its Personality PROM.
data	data bytes are expected in numeric ASCII characters with comma delimiters. The numbers have the following ranges:

	Range	
	Decimal	Hexadecimal
Data	–128 to –255	0 to FF
Addr multiple personality	0 to 65535	0 to FFFF

Example WM,100,0,0,0,1,0,2<tm>

writes six bytes of the multiple personality PROM starting at address 100 with the data 0, 0, 0, 1, 0, and 2.

WR

Syntax `[t]WR,start address,data<tm>`

Purpose The Write Receive command writes the receive personality PROM in decimal or hexadecimal format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. A terminal must be disabled to read or write its personality PROM.
data	data bytes are expected in numeric ASCII characters with comma delimiters. The numbers have the following ranges:

	Range	
	Decimal	Hexadecimal
Data	–128 to –255	0 to FF
Addr receive personality	0 to 32767	0 to 7FFFF

Example WR,30,0,1,2,3,4,5<tm>

writes the first six bytes of the receive personality PROM with the data 0, 1, 2, 3, 4, and 5.

WT

Syntax `[t]WT,segment,start address,data<tm>`

Purpose The Write Test RAM command writes the Test RAM in decimal or hexadecimal format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
segment	the Test RAM segment that data is to be written to.
start address	Note that the addressing in the Test RAM space is by 8-bit byte. A terminal must be in Test Mode and disabled to read or write its Test RAM.
data	data bytes are expected in numeric ASCII characters with comma delimiters. The numbers have the following ranges:

	Range	
	Decimal	Hexadecimal
Data	–128 to –255	0 to FF
Test RAM	0 to 8192	0 to 1FFFF

Example WT,3,0,5,4,3,2,1,0<tm>

writes the first six bytes of the Test RAM segment 3 with the data 5, 4, 3, 2, 1, and 0.

WX

Syntax `[t]WX,start address,data<tm>`

Purpose The Write Xmit command writes the Transmit Personality PROM in decimal or hexadecimal format.

Description

[t]	an optional literal terminal number, which must be either 0, 1, or 2. If omitted, the last value specified is used.
start address	Note that the addressing in the Personality PROM space is by 8-bit byte. A terminal must be disabled to read or write its Personality PROM.
data	Data bytes are expected in numeric ASCII characters with comma delimiters. The numbers have the following ranges:

	Range	
	Decimal	Hexadecimal
Data	–128 to –255	0 to FF
Addr Transmit Personality	0 to 8192	0 to 1FFFF

Example WX,0,5,4,3,2,1,0<tm>

writes the first six bytes of the Transmit Personality PROM with the data 5, 4, 3, 2, 1, and 0.

Programming Examples

The following commands set up the VX4469A to monitor all activity on a ARINC 629 bus connected to terminal 0. Before using this example, you must be sure that the VX4469A is properly connected to an active ARINC 629 bus.

Send these command strings to the VX4469A Module being exercised, using whatever programming language and syntax is appropriate for your system. Data returned from the VX4469A module is shown in monospaced type.

As is the case for the first occurrence of the “RC,0” command below, if there is no data captured in the circular buffer of the VX4469A card, it will return a <sp><cr><lf>.

0IM;	Command to the VX4469A to set up to monitor terminal 0 bus activity.
RC,0;	Command to set up the VX4469A to return data from circular buffer 0.
`<sp><cr><lf>`	Read from the VX4469A, then print the returned data.
RC,0;	Command to set up the VX4469A to return data from circular buffer 0.
`4100,0213,0000`	Read from the VX4469A, then print the returned data.
`4200,0316,0000`	
`<sp><cr><lf>`	
RC,0;	Command to set up the VX4469A to return data from circular buffer 0.
.	

.

Status and Events

Status and Events

Status

The module's current operating status may be determined from the LEDs on the front panel, and by issuing the IA or IP command.

Power LED This green LED is normally lit and is extinguished if the +5 V power supply fails or if the +5 V fuse blows.

Failed LED This normally off red LED is lit whenever SYSFAIL* is asserted, indicating a module failure. Module failures include failure to correctly complete a self test, loss of a power rail, or failure of the module's central processor.

If the module loses any of its power voltages, the Failed LED will be lit and SYSFAIL* asserted. A module power failure is indicated when the module's Power LED is extinguished.

MSG LED This green LED is normally off. When lit, it indicates that the module is processing a VMEbus cycle. The LED is controlled by circuitry that appears to stretch the length of the VMEbus cycle. For example, a five microsecond cycle will light the LED for approximately 0.2 seconds. The LED will remain lit if the module is being constantly addressed.

ERROR When lit, indicates an error has occurred and a message is in the error queue.

BACKGROUND Switches on and off at a rate inversely proportional to processor loading.

Each terminal has the following LEDs on the front panel of the module.

LED	When lit, indicates:
ENABLE	the terminal is enabled.
BUS BUSY	the terminal detects activity on the ARINC 629 bus.
STRING ACTIVE	the terminal IC is receiving or transmitting.
TRANSMITTING	the terminal IC is transmitting.
RECEIVE ERROR	the terminal IC receive error flag is/was true.

TRANSMIT ERROR	the terminal IC transmit error flag is/was true.
TRANSMIT DISABLE	the terminal IC has disabled the SIM for transmitting.

Events

The VX4469A stores error messages in an error queue. The VX4469A will return these error messages with a LE command.

The error messages are stored in two formats, normal or brief. The default, power-on mode is normal. In normal mode, the messages are stored in the error queue as character strings that are a description of the error. In brief mode, the messages are stored in the error queue as a single code character. The SFN command causes error messages to be stored in normal mode and the SFB command causes error messages to be stored in brief mode.

In either mode, errors are returned with a carriage return and a line feed character appended to the message. In either mode, a empty error queue is signaled by returning the three character message: space, carriage return and line feed.

The following is a list of the error messages and the single character in hex that represents that error message.

Hex Character	Error Message
21	ATTEMPT TO READ PERSONALITY RAM OF ENABLED TERMINAL
22	ATTEMPT TO WRITE PERSONALITY RAM OF ENABLED TERMINAL
23	B, G, N OR U NOT DEFINED FOR COMMAND
24	BUF ALREADY DEFINED
25	BUFFER CELL FULL
26	BUFFER FULL
27	BUFFER NOT DEFINED
28	BUFFER TOO LARGE
29	CHAIN TO SAME BLOCK
2a	CID TOO LARGE
2b	COM ERROR TM0
2c	COM ERROR TM1
2d	COM ERROR TM2
2e	EXT TOO LARGE
2f	INSTRUCTION BLOCK FULL
30	INTERRUPT MUST BE 1,2,4 OR 8

Hex Character	Error Message
31	INVALID NUMBER
32	LABEL >= 0FF8H WHEN SETTING DATA LENGTH
33	LABEL >= 0FF8H WHEN SETTING MSC
34	LABEL TOO LARGE
35	MEMORY BOUNDARY VIOLATION
36	NOT ENOUGH BUFFER SPACE
37	OFFSET TOO LARGE
38	OVERLOAD
39	SYNTAX ERROR
3a	U NOT DEFINED FOR COMMAND
3b	UNDEFINED FUNCTION
3c	X TOO LARGE
3d	Y TOO LARGE
3e	LCA PROGRAM FAILURE
3f	ATTEMPT TO FILL TEST RAM OF ENABLED TERMINAL
40	ATTEMPT TO READ TEST RAM OF ENABLED TERMINAL
41	ATTEMPT TO WRITE TEST RAM OF ENABLED TERMINAL
42	ATTEMPT TO CHANGE LOGIC MODE OF ENABLED TERMINAL
43	NOT VALID IN TEST MODE
44	NOT VALID IN THIS MODE
45	ONLY VALID IN THIS MODE
46	TEST MEMORY WRAP ERROR

Appendices

Appendix A: Specifications

This appendix contains the VX4469A specifications. All specifications are warranted unless they are designated *typical*. Typical characteristics describe typical or average performance and provide useful reference information.

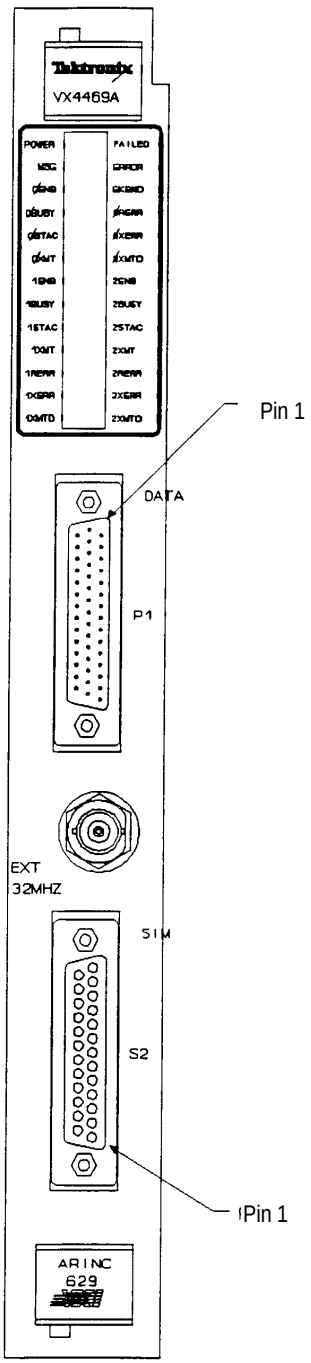
Table A–1: Specifications

Characteristic	Description
Number of Terminals	One standard; expandable to three.
Terminal IC and SIMs	Each terminal uses Boeing-designed VLSI terminal IC and serial interface modules.
Data Bus Coupling	Each terminal transmits and receives through a standard user-supplied Current Mode Coupler. External power to the CMC is not required for operation.
Operating Mode	Periodic and aperiodic.
Transmit Schedule Modes	Block, Independent, Alternate. Determined by contents of the XPP.
Multiple Terminal Emulation	Multiple terminal emulation using a single terminal and programming individual CID for each wordstring. Multiple terminal emulation using separate terminals and terminal gap timers.
Timers	Programmable. Time Interval, Terminal Gap, and Synch Gap.
Memory Allocation	Supports maximum addressable memory space for the RPP, MPP, XPP, and shared memory.
Buffer and Program Memory	256 Kbytes.
Power Requirements	All required DC power is provided by the internal power supply in the mainframe.
Voltage	+5 Volt supply: +4.75 VDC to +5.25 VDC. +24 Volt supply: +23.5 VDC to +24.5 VDC. –24 Volt supply: –23.5 VDC to –24.5 VDC.
Replacement Fuses	+5 V: Littlefuse P/N 273004. ± 24 V: Littlefuse P/N 273001.
Cooling	Provided by the fan in the VXIbus mainframe. Less than 10° C temperature rise with 1.2 liters/sec of air at a pressure drop of 0.03 mm of H_2O.
Temperature	0° C to +50° C, operating. –40° C to +85° C, storage.

Table A–1: Specifications (Cont.)

Characteristic	Description
Humidity	Less than 95% R.H. non-condensing, 0° C to +30° C. Less than 75% R.H. non-condensing, +31° C to +40° C. Less than 45% R.H. non-condensing, +41° C to +50° C.
Radiated Emissions	Complies with VXIbus Specification.
Conducted Emissions	Complies with VXIbus Specification.
Module Envelope Dimensions	VXI C size. 262 mm × 353 mm × 30.5 mm (10.3 in × 13.9 in × 1.2 in)
Dimensions, Shipping	When ordered alone, the module's shipping dimensions are: 406 mm × 305 mm × 102 mm. (16 in × 12 in × 4 in).
Weight	1.57 kg. (3.5 lbs.)
Weight, Shipping	When ordered alone, the module's shipping weight is: 2.02 kg. (4.5 lbs.)
Mounting Position	Any orientation.
Mounting Location	Installs in an instrument module slot (slots 1–12) of a C or D size VXIbus mainframe. (Refer to D size mainframe manual for information on required adapters.)
Front Panel Signal Connectors	DB25S (SIM I/O). DB44P (Data Port). Refer to Appendix B for pinouts.
Recommended Cable or Connector	VX1782P for the SIM I/O connector. VX1785S for the data port connector.
Equipment Supplied	One VX4469A ARINC 629 Module. One User Manual.
Options	VX4469A–01: Adds one additional terminal. VX4469A–02: Adds two additional terminals.
Software Revision	V1.0

Appendix B: Input/Output Connections



Data	Pin #	SIMS
D15	1	0 RXCX
GND	2	GND
D10	3	TIME RESET IN *
D8	4	TIME IN *
D5	5	2 N150
D3	6	2 RSA
GND	7	2 TSA
OUT ENABLE	8	1 N150
	9	1 RSA
	10	1 TSA
	11	0 N150
XTRIGIN T0	12	0 RSA
XTRIGIN T1	13	0 TSA
XTRIGIN T2	14	GND
	15	TIME RESET OUT *
DIRECTION	16	TIME OUT *
D13	17	GND
D11	18	2 RSB
GND	19	2 TSB
D6	20	GND
D4	21	1 RSB
D1	22	1 TSB
HS INPUT	23	GND
GND	24	0 RSB
	25	0 TSB
	26	
GND	27	
GND	28	
GND	29	
	30	
D14	31	
D12	32	
D9	33	
D7	34	
GND	35	
D2	36	
D0	37	
HS OUTPUT	38	
	39	
	40	
XTRIGOUT T0	41	
XTRIGOUT T1	42	
XTRIGOUT T2	43	
SOFTTRIG	44	

Figure B–1: VX4469A Front Panel

Appendix C: Trigger Lines

The VX4469A provides trigger interfaces that include external input/output, VXI trigger input/output terminals, and an external general software terminal. Trigger outputs can be programmed to signal other instruments regarding events detected by the VX4469A. The trigger inputs from other instruments can be used by the VX4469A to initiate VX4469A functions.

The external trigger interfaces are through the front panel DATA connector. Refer to *Appendix B: Input/Output Connections* for information on the front panel Data connector. The VXI TTL trigger lines interface through the backplane. Refer to Jumpers in the *Getting Started* section of this manual for information regarding the TTL trigger lines.

The following table lists commands that are described in the *Commands Description* section of this manual. Refer to *Vector Instruction Block Functions* in *Appendix F: Advanced Technical Support* section. Tasks 10 through 13 support the VX4469A trigger functions.

Table C–1: VX4469A Trigger Commands

External Trigger Commands	VXI TTL Trigger Commands
SKD	SKD
SKX	SKV
SQXE	SQVE
SQXG	SQVG
SQXI	SQVI
SQXR	SQVR
TQSX	TQSV
TQX	TQV

Appendix D: Performance Verification

Please consult the factory for performance verification information.

Appendix E: Front Panel Data Port

The VX4469A has a bi-directional TTL 16-bit data port on its front panel. This allows 16-bit data transfers between the VX4469A and an external device. The transfers are controlled by two handshake lines, one input and one output, both active low.

The handshaking processes is as follows:

1. The transmitting device indicates it has data available by making its handshake output go low and waits for the receiving device to acknowledge.
2. The receiving device acknowledges by making its handshake output go low and waits for the transmitting device to remove its data available signal.
3. The transmitting device makes its handshake output go high and waits for the receiving device to remove its acknowledge.
4. The receiving device makes its handshake output go high and ends the data transfer.

The VX4469A as a transmitter clocks the 16-bit data out at the same time it indicates data available (step 1).

The VX4469A as a receiving device latches the data when the transmitting device makes its output go high (step 3).

The VX4469A also provides two more signal lines:

- Pin 1 is a VX4469A output that indicates data direction. A high on pin 1 indicates the VX4469A is a transmitter. A low on pin 1 indicates the VX4469A is a receiver.
- Pin 12 is a VX4469A input that allows the interfaced device to control the VX4469A's data outputs when the VX4469A is a transmitter. A high on pin 12 will cause the data VX4469A data lines to go to a high impedance state. A low on pin 12 will cause the VX4469A data lines to go to a low impedance state. If the VX4469A is set up for receiving (NWD command), the VX4469A data lines will always be in a high impedance state.

See the NRD and NWD commands for setting up the VX4469A to transfer data through its front panel data port. See *Appendix B* for the pin number definitions.

Appendix F: Error Register

The types of errors and diagnostic information contained in the terminal IC Error register are listed below. The contents of the Error register can be returned by the RG command. (See *Status and Events* for a listing of error codes and messages.)

String Error (Bit 15)

This bit indicates whether a string error has taken place in the current wordstring. A string error immediately flags the occurrence of any of the errors appearing in the error register except for the operating mode bits 2 and 1. The string error is reset after subsystem activity is completed for each wordstring, or in the case of no bus acknowledge, before subsystem activity for the next data word begins.

The string error bit can be used to determine whether errors read from the error register are current, or whether they were made in a prior string. For example, if the string error bit is reset, any other errors latched in the error register must be from a prior string. The string error bit also appears as the most significant bit of the interrupt vectors.

TXE: Transmitter Enable (Bit 14)

This signal, when active, indicates that the transmitter is permanently shut down until power is recycled. TXE is activated after the occurrence of seven consecutive transmit errors. In addition to being in the error register, TXE is an external DATAC output pin which lights the terminal transmit disable LED when true.

XERF: Transmit Error Flag (Bit 13)

When active, this signal indicates that a transmit error has occurred and the monitor has inhibited the transmission. The wordstring in which the error was detected may be truncated. In addition to being in the error register, XERF is an external DATAC output pin. This pin triggers a flip-flop which causes the terminal transmit error LED to light and a ‘COM ERROR’ message to be placed in the error queue. The type of error that has occurred is encoded in bits 12 through 10 of the error register and is described below.

TX Monitor Error (Bits 12, 11, 10)

The binary encoding of these three bits is as follows:

0 no error

1 EOM = End Of Message error: The monitor encounters more wordstrings than expected. This error could be caused by the value of the message string count (MSC) in the RPP being too small.

2 data NVALDWD = Invalid Data Word error: This error is due to the detection of improper Manchester bi-phase, parity, or synchronization of a transmitted data word. This error is most likely caused by noise on the bus.

3 EOS or NDATA IXT = End of String error: The monitor detects that the transmitter is still transmitting, but the monitor had expected the string to end. This error condition can occur when the monitor expects either a multi-word or label only string and receives more data than anticipated. The error could be caused by a word count value in the RPP that is less than the word count value in the XPP. In the case of an error during a label-only string, a discrepancy may exist between data bits in the RPP and XPP.

4 BUSQ IXT = Bus Quiet error: The monitor expected another word to be transmitted but the bus is quiet following a data sync pattern. This error indicates a hardware failure characterized by an untimely quiet bus. Potentially, this error could be the result of the serial bus inputs being disconnected from RXI and RXN, a failed transmit monitor, or a short in the bus stub, etc.

5 label NVALDWD = Invalid Label Word error: This error is due to detection of improper Manchester bi-phase, parity or synchronization of a transmitted label. The error is most likely due to noise on the bus.

6 NXOK = Not Okay to Transmit error: The transmitted label's extension (EXT) is not equal to the terminal's channel ID (CID) or the message string count (MSC) in the RPP monitor cell is equal to IF. The first condition may result in the terminal "impersonating" another DATAC terminal on the bus, since its erroneous EXT may match the CID of another terminal. The second condition could be caused by an improperly programmed or addressed RPP.

7 NANWD or BUSQ = Not Another Word or Bus Quiet error: The monitor expected data but detects a quiet bus. This error indicates a hardware failure as in '4' above.

PAM Errors (Bits 9, 8)

A PAM error occurs when there has been an error in the Protocol Access Module (PAM) within the DATAC device. Specifically, there was disagreement between a protocol value programmed in the RPP and the protocol value on the straps (i.e., DATAC pins). This error could be caused by a strap failure or incorrect programming or addressing of the RPP. There are two PAM error bits in the error register. The first, TI, indicates that the disagreement was with the transmit interval values, and the second, SG/TG, indicates that the disagreement was with the Sync Gap or Terminal Gap values. Without agreement between the two TI, SG and TG values, the DATAC chip will not transmit. If the VX4469A is set to use the high four bits of the transmit cell label for the channel ID, then the receive personality RAM must have the correct TI, TG, and SG programmed into all the channel IDs being used.

No Bus Acknowledge, Read (Bit 7)

This bit indicates that DATAC did not receive a BUSA during an attempted shared memory read. Therefore, the data word transmitted by the device after the attempted read was invalid. Other DATAC terminals on the bus will not accept the invalid data word because the affected DATAC chip transmits incorrect parity under these conditions. In addition, XERF is set to indicate to the subsystem that a transmit error has occurred. DATAC's internal error counter is not incremented, however, since the condition of no bus error does not constitute a functional DATAC error. This error will occur if the SBD command is used.

No Bus Acknowledge, Write (Bit 6)

This bit indicates that DATAC did not receive a BUSA during an attempted subsystem write. Therefore, the data word just received by DATAC was not written into shared memory. This error causes the receive error flag (RERF) to be set to zero. This error will occur if the SBD command is used.

RERF: Receive Error Flag (Bit 5)

This signal indicates that a receive error has occurred and reception of the wordstring was truncated. A receive error can be caused by 1) the detection of improper Manchester bi-phase, parity or synchronization of a received label or data word, 2) a received wordstring that was shorter than expected, or 3) no bus acknowledge during an attempted shared memory write. This pin triggers a flip-flop which causes the terminal transmit error LED to light and a 'COM ERROR' message to be placed in the error queue.

Parity Error (Bit 4)

A parity error occurs in conjunction with a label or data NVALDWD error and a receive error (RERF) and is detected when a received label or data word has had bad parity. Since it is unlikely that noise or a clash would cause bad parity by itself, if the parity error bit is set, it may indicate that the bit was loaded deliberately by the transmitting terminal. The transmitter sends incorrect parity to indicate that it did not receive data from its subsystem. (See No bus acknowledge, read.)

Short String Error (Bit 3)

This bit occurs in conjunction with a receive error (RERF) and indicates that the wordstring received was shorter than expected. This condition can alert the subsystem that data received may not be reliable.

DATAC Operating Mode (Bits 2, 1)

There are four input test pins on the DATAC chip (ENCO, ENC1, ENC2, ENC4) used to command the DATAC chip into a test mode. In general, these test modes are only used for diagnostic testing during chip manufacture, and should not be enabled. Because the device is not guaranteed to function properly when in a test mode, the test logic was designed so the any one test pin failure (i.e., any one test input being inadvertently grounded), would not be sufficient to enable the DATAC into a test mode. In addition, failure of two test pins will cause DATAC to cease any transmitting operation.

Bits 1 and 2 of the error register indicate whether the DATAC is operating near to or in any one of these test modes.

Bit 2	Bit 1	Operation
0	0	DATAC is operating in normal mode. The chip is two test pin failures away from any test mode.
0	1	DATAC is operating in an optional mode. The chip is one test pin failure away from a test mode.
1	0	DATAC is operating in an optional mode. The chip is one test pin failure away from a test mode.
1	1	DATAC is in a test mode. Operation will not be normal. At least two of the test pins have failed.

Impersonation Error

An impersonation error occurs when the received label's extension (EX) is equal to the receiving terminal's channel ID and the message string count in the RPP is not equal to 1F. A MSC not equal to 1F indicates that the receiving terminal does in fact transmit the label in question. This condition may be the result of the receiving terminal's or the transmitting terminal's CID being set incorrectly. Note that the user friendly commands complement the MSC value, so a 1F would be 0.

Last Word Monitor

The last word monitor register (LWM) contains the last label or data word sent for a transmit operation and the last label word received for a receive operation. For a transmit operation, the label and data words recorded in the LWM have been transmitted out on the DATAC bus and received by the DATAC transmit monitor. This design facilitates subsystem wrap around verification. For receive operations, the LWM holds the label of the current or most recently received wordstring until it is overwritten by the next incoming label.

Interrupt Vector Register

The interrupt vector register (IVR) contains the fifteen bit interrupt vector for the current received or transmitted wordstring. The most significant bit of the IVR contains the string error bit. The IVR makes available to a subsystem the value of the current interrupt vector at any time whether or not the personality PROMS have been programmed to provide interrupt vector strobes.

Appendix G: Advanced Technical Support

Application Note

The Interrupt Vector, Vector Index, Instruction Block, and Circular Buffer features of the VX4469A are very powerful. This appendix gives one example of how to use them. All numbers in this example will be in HEX.

This example assumes that the VX4469A is connected to an active ARINC 629 bus. Assume you want to save all data that a terminal IC receives with a particular label. Assume also that the label is 55, that the label will have 6 words of data with it, and you decide to have the terminal IC 1 store the data starting at address 100. The following command will set up the Receive Personality RAM for this situation:

1uwm,0,i0	select terminal 1 set up offset table 0 for all zeros.
uwr,55,n6,d100,p0,L155	set up receiving label 55, words data at base address 100, offset pointer table 0, generate interrupt vector 155 late or immediately after the data is received.

Set up a circular buffer to store this data in by setting up an instruction block as follows:

dc,0,6,100	define circular buffer 0, 6 words per cell, 100 cells

Program an instruction block to copy data from address 100 to the circular buffer as follows:

svi,1,2,0,1,100	instruction block 1, function 2 (copy data to circular buffer), circular buffer 0, terminal 1, shared memory address 100

The number of words transferred each time the instruction block is executed is the number of words per cell as defined for the circular buffer. The VX4469A also needs to know how to connect interrupt vector 155 with instruction block 1. This is done by setting a vector index as follows:

svx,155,1	set index for vector 155 to instruction block 1.

Now when terminal 1 is enabled and data is received with a label 55, the terminal IC will generate an interrupt vector 155. The VX4469A processor will look up vector 155 in the vector index table and find that it is to execute vector instruction block 1. Vector instruction block 1 is executed, which will copy six words of data from terminal 1's shared memory address 100 to circular buffer 0.

You may then start reading data from circular buffer 0 with one of the two following commands:

rc,0	Read from circular buffer 0. Each read from the board will now transfer 6 words to the system controller from one cell and then free the cell. If no words are available, a <sp><cr><lf> will be returned and the command terminated.
brc,0,6000	Read from circular buffer 0 in binary mode. 6000 words or c000 bytes will be returned as they are available. The VX4469A will hold the transfer until data is available. Requesting 0 words will return data from circular buffer 0 until another command is sent to VX4469A.

Vector Instruction Block Functions

When a vector instruction block is executed, the VX4469A will start reading at the start of the block and perform all functions until it finds the function 0. The default initialized condition of the instruction blocks is the function 0 at its beginning.

The instruction blocks are programmed using the SVI command. The currently defined functions and their parameters are listed below. The examples at the end of this section show the correct syntax.

Note that instruction block 0 is the default instruction block for the vector indexes. It is recommended that instruction block 0 not be changed from its default state of just function 0.

Funct.	Parameters	Description
0	none	Terminates instruction block. This function is automatically added as the last function. It uses one word in an instruction block.
1	circular buffer number terminal number data address	Copies vector, timestamp, and data from terminal shared memory to a circular buffer. The number of words copied is the defined size of a cell in the circular buffer. If the cell size is one, only the vector word is put in the circular buffer. If the cell size is 2, the vector word and low 16 bits of the timestamp are copied. If the cell size is 3, the vector word, low word of the timestamp and the high word of the timestamp are copied to the circular buffer. The amount that the cell size is greater than 3 is the number of consecutive words that are copied from the terminal's shared memory to the circular buffer. If the circular buffer is full, no data is transferred. But the next time it can transfer data, the high order bit of the timestamp high word will be set to indicate overflow. This function uses two words in an instruction block. It takes 127 ms + 1.6 ms per word in the circular buffer cell.
2	circular buffer number terminal number terminal data address	This function is the same as function 1 except that only data, not vector nor timestamp information, is transferred. This function uses two words in an instruction block.
3	circular buffer number terminal number terminal data address	This function is the same as function 1, except that vector and 3 words of timestamp information are transferred. The second word of timestamp is a full 16 bits of timestamp. The third word of timestamp contains 11 more bits of timestamp (bits 0 to 10). Bit 11 is the overflow bit, and bits 12 to 15 are the extension bits of the label if this is a receive interrupt vector. This function uses two words in an instruction block.
4	circular buffer number terminal number terminal data address	This function copies data from the circular buffer to the terminal data address. The number of words transferred is the cell size of the circular buffer. Each time data is copied, that cell is removed from the circular buffer. This function uses two words in an instruction block.
5	circular buffer number terminal number terminal data address	This function copies data from the circular buffer to the terminal address. The number of words transferred is the cell size of the circular buffer. Each time data is copied, a pointer is bumped to the next cell but the cell is not deleted. For instance, you could write 100 sets of data to a circular buffer and then have the VX4469A transmit the 100 sets one at a time, repeating all 100 until stopped. This function uses two words in an instruction block.

Funct.	Parameters	Description
7	circular buffer number terminal number terminal data address	This function is the same as function 3, except that the third word of timestamp information contains 8 bits (0 to 7) of high order timestamp, and two bits which are the terminal (0, 1 or 2) that created the interrupt vector. Bit 11 is the overflow bit, and bits 12 to 15 are the extension bits of the label if this is a receive interrupt vector. Including the terminal number in the timestamp is useful if more than one terminal is placing information in a circular buffer. This function uses two words in an instruction block.
8	circular buffer number terminal number terminal data address	This function copies data from the circular buffer to the terminal address. The number of words transferred is the cell size of the circular buffer. The first word of data will be written to shared memory without disturbing the high 4 bits of the word in shared memory. This allows updating a wordstring with a freshness counter in the high 4 bits of the first word. Each time data is copied, that cell is removed from the circular buffer (marked unused). This function uses two words in an instruction block.
9	data monitor address mask word 1 mask word 2 ID word circular buffer number terminal number terminal data address	Each time this interrupt vector instruction runs, the two words of data at shared memory address “data monitor address” are checked. They are checked by first masking them with the mask words 1 and 2 (logical and) and then comparing the results with the results of the previous check. If either of them has changed, this instruction copies vector, timestamp, ID and data from terminal shared memory to a circular buffer. The ID is a user defined number and has no effect other than to identify for the user the data a particular vector instruction places in a circular buffer. The number of words copied is the defined size of a cell in the circular buffer. If the cell size is one, only the vector word is put in the circular buffer. If the cell size is 2, the vector word and low 16 bits of the timestamp are copied. If the cell size is 3, the vector word, low word of the timestamp and the high word of the timestamp are copied to the circular buffer. If the cell size is 4, the vector word, low word of the timestamp, the high word of the timestamp and the ID word are copied to the circular buffer. The amount that the cell size is greater than 4 is the number of consecutive words that are copied from the terminal’s shared memory starting at “terminal data address” to the circular buffer. If the circular buffer is full, no data is transferred. But the next time it can transfer data, the high order bit of the timestamp high word will be set to indicate overflow. This function uses eight words in an instruction block.
a	data monitor address mask word 1 mask word 2 ID word circular buffer number terminal number terminal data address	This function is the same as function 9 except that only ID and data, not vector or timestamp information, is transferred. This function uses eight words in an instruction block.
b	circular buffer number interrupt number interrupt count	Interrupt on circular buffer nearing empty. This function sets an ‘interrupt number’ bit in the system interrupt status register if a circular buffer has less than ‘interrupt count’ cells used when this instruction is executed. If the system vector interrupts are enabled, this function will also cause a system interrupt. The ‘interrupt number’ may be 1, 2, 4, or 8. See the RI command for further information. This function uses two words in an instruction block.

Funct.	Parameters	Description
c	circular buffer number interrupt number interrupt count	Interrupt on circular buffer nearing full. This function sets an 'interrupt number' bit in the system interrupt status register if a circular buffer has more than or equal to 'interrupt count' cells used when this instruction is executed. If the system vector interrupts are enabled, this function will also cause a system interrupt. The 'interrupt number' may be 1, 2, 4, or 8. See the RI command for further information. This function uses two words in an instruction block.
d	countmode countlabel terminal address	Interrupt on CRC error. 'd' is the function code. 'countmode' determines whether the word in shared memory 'address' is the 'number of data words' ('countmode' = 1) or if 'count' is the 'number of data words' ('countmode' = 0). 'label' is included as part of the data for calculating the CRC along with 'number of data words' in 'terminal's shared memory starting at 'address'. No CRC is written, but if the result is not &0000 hex, then a system interrupt occurs and bit 4 in the interrupt register is set. See the RI command for more information on interrupts. This function uses 5 words in an instruction block.
e	countmode count label terminal address	Calculate and append a CRC. 'e' is the function code. 'countmode' determines whether the word in shared memory 'address' is the 'number of data words' (variable) or if 'count' is the 'number of data words'. 'label' is included as part of the data for calculating the CRC along with 'number of data words' – 1 in 'terminal"s shared memory starting at 'address'. The CRC is written in shared memory at location ['address' + 'number of data words' – 1]. This function in 'countmode' = 0 takes 26 microseconds plus 40 microseconds per 'count'. In 'countmode' = 1, this function takes 30 microseconds plus 42 microseconds per 'number of data words'. This function uses 5 words in an instruction block.
f	interrupt number	System interrupt. This function sets a bit, 'interrupt number', in the system interrupt status register. If the system vector interrupts are enabled, this function will also cause a system interrupt. The interrupt number may be 1, 2, 4 or 8. See the RI command for further information.
10	delay between steps number of steps step size reset value terminal number shared memory address	Ramp. This function changes the data word at 'terminal number' / 'shared memory address'. The first time this function in an instruction block is executed, it sets the data word to 'reset value'. Then after being executed 'delay between steps' times, it adds 'step size' to data word. This is repeated 'number of steps' times. After the data word is stepped 'number of step' times, the data word is not changed. This function uses eight words in an instruction block.
11	delay between steps number of steps step size reset value terminal number shared memory address	Sawtooth. This function changes the data word at 'terminal number' / 'shared memory address'. The first time this function in an instruction block is executed, it sets the data word to 'reset value'. Then after being executed 'delay between steps' times, it adds 'step size' to data word. This is repeated 'number of steps' times. After the data word is stepped 'number of steps' times, the data word is then reset to 'reset value' and the process is repeated continuously. This function uses eight words in an instruction block.
12	delay between steps number of steps step size reset value terminal value shared memory address	Triangle. This function changes the data word at 'terminal number' / 'shared memory address'. The first time this function in an instruction block is executed, it sets the data word to 'reset value'. Then after being executed 'delay between steps' times, it adds 'step size' to data word. This is repeated 'number of steps' times. After the data word is stepped 'number of step' times, the data word is then stepped backwards 'number of steps' – 1 times. The process is repeated continuously. This function uses eight words in an instruction block.

Funct.	Parameters	Description
13	delay between steps step size terminal number shared memory address	Add. This function changes the data word at ‘terminal number’ / ‘shared memory address’. After being executed ‘delay between steps’ times, it adds ‘step size’ to data word. This is repeated continuously. This function uses five words in an instruction block.
14	delay between steps terminal number shared memory address	Freshness Counter. This function keeps a 4-bit freshness counter at ‘terminal number’ / ‘shared memory address’. The first time this function executes, it reads the upper four bits of the data word at ‘terminal number’ / ‘shared memory address’, saves the four bits internal to the instruction block, and writes the four bits back to shared memory. Each time it executes thereafter, the four bits stored in the instruction block are written to the high four bits of ‘terminal number’/ ‘shared memory address’. After executing ‘delay between steps’ times, the four bits internal to the instruction block are incremented before they are written to shared memory. A value of 0 ‘delay between steps’ will increment the four bits every time. A value of 1 will increment the four bits every other time, etc. This function allows updating data in a wordstring from a circular buffer while maintaining a freshness counter anywhere in the string. It also handles freshness counter updates less frequently than every TI. This function uses five words in an instruction block.
1d	variable number of parameters.	Test –> Task. This function enables executing a task conditional upon the results of a test. See the section labeled Test –> Task Function later in this Appendix. This function uses a variable number of words in an instruction block depending upon the particular test and task used.
1f	instruction block	Chain. Instruction blocks contain 16 words each. If the number of functions needed for a particular vector is more than 16 words including the terminating function 0, Instruction blocks may be chained together with this command. When executed, this command switches execution to ‘instruction block’. As many instruction blocks may be chained together as desired. This function uses one word in an instruction block.

Test –> Task Functions

Interrupt vector instruction 1d interprets a variety of tasks that a executed only after a successful result of a variety of tests. The tests currently defined are:

Test	Parameters	Description
0	circular buffer number	Test for a circular buffer's in and out pointers being equal. They will be equal if the circular buffer is empty or full. This test uses two words in an instruction block.
1	circular buffer number	Test for a circular buffer's out and rotate pointers being equal. The rotate pointer is used by interrupt instruction 5 to keep track of the next cell to move to shared memory. When the two pointers are equal, the next cell to be moved is the first one written to the circular buffer. This test uses two words in an instruction block.
10	terminal number A shared memory addr A terminal number B shared memory addr B	Test for two shared memory words being equal. The two words may be in the memory space of two different or the same terminal. This test uses four words in an instruction block.
11	terminal number A shared memory addr A terminal number B shared memory addr B	Test for two shared memory words being not equal. The two words may be in the memory space of two different or the same terminal. This test uses four words in an instruction block.
ff	none	The following task is done unconditionally. This test uses one word in an instruction block.
The following tasks are currently defined for the Test –> Task vector instruction 1d.		
0	interrupt vector instruction block	Change an interrupt vector instruction block. This task is the same as an svx command. If the condition is true in the test portion of this function, the vector index table entry for "interrupt vector" will be changed to "instruction block". This is the same as svx,"interrupt vector","instruction block". This task does not terminate the execution of its instruction block. Any function(s) defined after this one will also be executed whether or not this task was executed. This task uses two words in an instruction block.
1	instruction block	Chain to another interrupt vector instruction block. If the condition is true in the test portion of this function, the remainder of the current instruction block is not executed and control will be passed to the beginning of "instruction block". If the result of the test is not true, execution will continue in the current block. This task uses one word in an instruction block.
8	terminal number	Enable "terminal number". If the condition is true in the test portion of this function, "terminal number" will be enabled. This is the same as the SE command. Whether or not this task is executed, execution will continue in the current block. This task uses one word in an instruction block.
9	terminal number	Disable "terminal number". If the condition is true in the test portion of this function, "terminal number" will be disabled. This is the same as the SD command. Whether or not this task is executed, execution will continue in the current block. This task uses one word in an instruction block.
a	terminal number xpp segment	Switch "terminal number" terminal's xpp segment to "xpp segment". If the condition is true in the test portion of this function, "terminal number" terminal will be switched to "xpp segment". This is the same as the SP command. Whether or not this task is executed, execution will continue in the current block. This task uses one word in an instruction block.

Test	Parameters	Description
10	terminal number	VXI Trigger. If the condition is true in the test portion of this function, “terminal number” will output a negative pulse on its VXI trigger output. This is the same as a TQV command. Whether or not this task is executed, execution will continue in the current block.
11	terminal number	External Trigger. If the condition is true in the test portion of this function, “terminal number” will output a negative pulse on its external trigger output. This is the same as a TQX command. Whether or not this task is executed, execution will continue in the current block.
12	none	VXI Software Trigger. If the condition is true in the test portion of this function, “terminal number” will output a negative pulse on the VX4469A software VXI TTL trigger output. This is the same as a TQSV command. Whether or not this task is executed, execution will continue in the current block.
13	none	External Software Trigger. If the condition is true in the test portion of this function, “terminal number” will output a negative pulse on the VX4469A software external trigger output. This is the same as a TQSX command. Whether or not this task is executed, execution will continue in the current block.

Examples

1. SVI,1,4,0,0,1000

instruction block 1,
function 4,
circular buffer 0,
terminal 0,
data address 1000

In this example, every time this instruction block 1 is executed, the VX4469A will move a cell of data from circular buffer 0 to terminal 0's data address 1000.

2. SVI,2,2,0,0,2000,1,1,0,2000

instruction block 2,
function 2,
circular buffer 0,
terminal 0,
data address 2000
function 1,
circular buffer 1,
terminal 0,
data address 2000

In this example, when instruction block 2 is executed, the VX4469A will copy one cell of data from terminal 0's data address 2000 to circular buffer 0. It will also copy one cell of vector, timestamp and terminal 0's data at address 2000 to circular buffer 1. In this case, circular buffer 0 is used to transfer received data to be transmitted again. Circular buffer 1 is used to store vector, timestamp, and data received for retrieval.

3. SVI,1,8,0,0,1000

instruction block 1,
function 8,
circular buffer 0,
terminal 0,
data address 1000

In this example, every time this instruction block 1 is executed, the VX4469A will move a cell of data from circular buffer 0 to terminal 0's data address 1000. The first word of data will be written to shared memory without disturbing the high 4 bits of the word in shared memory. This allows updating a wordstring with a freshness counter in the high 4 bits of the first word.

4. SVI,2,9,106,f800,0,34,3,2,102

instruction block 2,
function 9,

data monitor address 106,
maskword1 f800,
maskword2 0000,
ID 34,
circular buffer 3,
terminal 2,
data address 102

When this instruction block is executed, the word at terminal 2 shared memory address 106 will be logically ANDed with f800 hex and shared memory address 107 will be ANDed with 0000. The results of these two operations will be compared with the results of the previous time this instruction block was executed. If either of the results have changed and there is room, the interrupt vector number, 2 words of timestamp, ID and data starting at shared memory address 108 will be written to circular buffer 3. This has the effect of writing data to circular buffer 3 only when any of the high 5 bits of terminal 2 shared memory address 106 change. When the VX4469A parses the SVI command for this instruction, the “previous result” words are initialized with 0.

5. SVI,1,5,4,0,104

instruction block 1,
function 5,
circular buffer 4,
terminal 0,
data address 104

SVI,2,5,4,0,104,1d,1,4,0,233,3

instruction block 2,
function 5,
circular buffer 4,
terminal 0,
data address 104
function 1d,
test 1,
circular buffer 4,
task 0,
vector number 233,
instruction block 3

SVI,3,5,3,0,104

instruction block 3,
function 5,
circular buffer 3,
terminal 0,
data address 104

SVI,4,5,3,0,104,1d,1,3,0,233,1

instruction block 4,
function 5,
circular buffer 3,
terminal 0,
data address 104
function 1d,
test 1,
circular buffer 3,
task 0,
vector number 233,
instruction block 1

The above four instruction blocks would be useful for switching between two circular buffers at the first entry in the circular buffers.

Assume that circular buffer 4 contains four 1-word cells with the numbers 1, 2, 3, and 4, and circular buffer number 3 also has four 1-word cells, but with the numbers 1, 3, 5, and 7. Initially, a svx,233,1 would cause instruction block 1 to be executed with vector number 233. The sequence 1,2,3,4,1,2,3,4,1,2,3,4,1,2,3.... will be copied to terminal 0 address 104. If the command svx,233,2 is given, the sequence will continue from circular buffer 4 until the number 4 is copied to address 104. At this point, test 1 will be true and instruction block 3 will be written at vector index 233. Thus instruction block 3 will be used and the sequence will now be 1,3,5,7,1,3,5,7,1,3... The transition will be automatically made at the end of the sequence in circular buffer 4. The command svx,233,4 would change back to the sequence in circular buffer 4 at the end of the sequence in circular buffer 3. The lvx,233 command would return the current circular buffer being used.

Pseudo Bus

The VX4469A may be operated without Serial Interface Modules (SIMs) and current couplers with a voltage mode pseudo bus (Tektronix P/N 950-7909-00). Pseudo bus modules are plugged into the SIM sockets of all units that are to share a bus. All that is required externally is a pair of wires connecting all the TSA, TSB outputs that would normally go to a current coupler.

This bus is similar to and compatible with one developed at Boeing which is described in National Semiconductor Corporation's preliminary data sheet for the XD15U9ADJ ARINC 629 (DATAC Terminal Device), DATAC Pseudo Bus.

The VX4469A Pseudo Bus Modules use a 74LS240 bus driver instead of 7406 invertor, providing more reliable results.

The VX4469A Pseudo Bus Modules may be suitable for users who wish to do some portion of their development/testing without SIMs and current couplers. The Pseudo Bus Modules do not support the SIM BITE features.

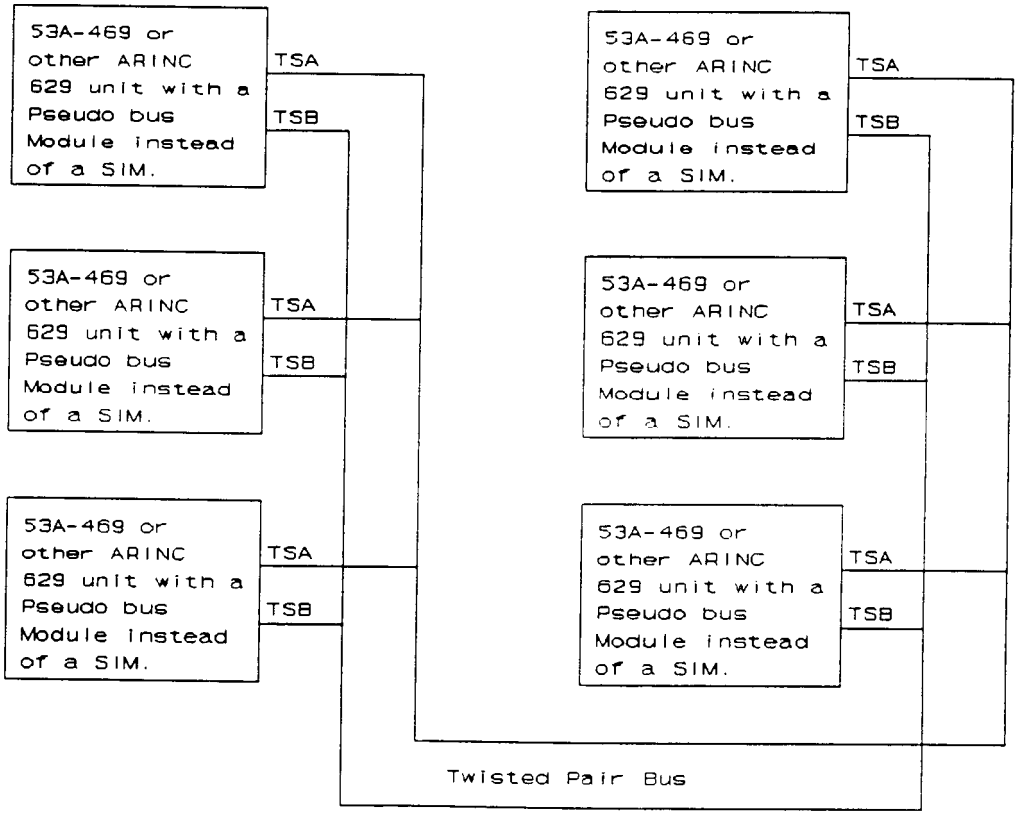


Figure G–1: Pseudo Bus Configuration

Fault Management And Internal Test Functions

The SIM constantly monitors its signal quality and reports any problems to the ARINC 629 terminal. This is done by monitoring the amplitude and symmetry on both the transmit and receive stub interfaces. If a problem is detected, the SIM signals this to the ARINC 629 terminal by preventing the next two transitions in the Manchester regenerating logic (a single deletion might not be noticed immediately in the receive mode). The missing transitions will be detected by the checking logic in the terminal, which can truncate any transmission in progress to prevent a faulty message from being sent, or reject any message being received.

The SIM is designed to work with a current mode coupler which may contain two separate channels for reliability. The SIM selects which channel is used in the coupler by controlling the polarity of the doublet sent on the transmit stub. If an error is detected by wraparound checking in the SIM during transmit, then at the beginning of the next wordstring the SIM will order the coupler to switch to the opposite channel.

If the SIM is unable to find a functioning coupler channel, it will enter a safety mode called “coupler fault” (COUF) mode. The SIM and coupler transmitters are disabled until the TXE signal is cycled.

The SIM constantly monitors the wraparound path including itself and the selected coupler channel. In addition, a test function is available which selects and monitors the spare coupler channel, and then reports the status of both channels. The test sequence is requested by a TS command and the results are returned by an RS command.

Fault Management in the SIM

ARINC 629 uses several levels of error detection, which is generally combined with multiple levels of redundancy to insure system reliability. The SIM constantly monitors signal quality and reports any irregularities to the ARINC 629 terminal, and may take remedial action as well. This is done through several functions:

1. TRANSMIT STUB MONITOR When transmitting, every doublet the SIM drives onto the transmit stub is monitored for amplitude and symmetry. If a problem is detected, the SIM signals this to the ARINC 629 terminal by preventing the next two transitions in the Manchester regenerating logic. The missing transitions will be detected by the wraparound checking in the terminal, which can truncate any transmission to prevent a faulty message from being sent. The terminal may take other action as well.

   The transmit stub monitor is also used to help locate whether a problem is in the SIM or the coupler.

2. RECEIVE INTEGRITY MONITOR Similarly, the receiver checks incoming doublets for amplitude and symmetry. Any problem is reported to the

terminal by deleting two transitions in the Manchester regenerating logic (a single deletion might not be noticed in the receive mode). During transmit mode (TXHB=0), when the bus signal quality is being monitored by wraparound checking, a second pair of input comparators is also activated. These wraparound comparators have higher input thresholds of ±1400 millivolts, to insure that the signal being transmitted onto the bus has sufficient amplitude to operate receivers at the far end of the bus. The receiver data stream is still derived from the lower-threshold comparators, however, This is to allow detection of collisions on the bus, less delay, etc.

3. COUPLER CHANNEL SELECTION The SIM is designed to work with a current mode coupler which contains two separate channels for reliability. The SIM selects which channel is used in the coupler by controlling the polarity of the doublet sent on the transmit stub. If an error is detected by wraparound checking in the SIM during transmit, then at the beginning of the next wordstring the SIM will order the coupler to switch to the opposite channel.

The SIM never changes doublet polarity while transmission of a wordstring is in progress. The decision to change coupler channels is made internally in the SIM after the end of a wordstring, as denoted by the TXHB going high followed by a bus quiet condition. However, this will not be apparent externally until the beginning of the next wordstring, when the SIM emits doublets of inverted polarity. (The term ‘inverted polarity’ is relative, as there is no dominant or absolute polarity. It is not necessary to keep track of true/complement polarity in the stub cable conductors, etc.)

Note that the coupler must receive a complete doublet before it detects a command to change channels. Thus the channel change in the coupler occurs between the first and second doublets of a wordstring, which are part of the sync pattern. If the channel change is due to a failure of one coupler channel, the first doublet of the new wordstring may be lost due to the finite switch-over time in the coupler. Because of this, the error counters in the SIM are locked to their current state during the first wordstring of inverted polarity — more on this in #4 below.

The criteria for determining whether to change channels are as follows:

The number of valid doublers emitted onto the transmit stub during a wordstring is counted, and the number of valid doublets detected from the receive stub is also counted. If the two numbers do not match when the wordstring is finished (wraparound delay can be as much as three bit times), the wordstring is declared to be bad, and the channel select flipflop is toggled. This scheme will not change channels if the SIM transmitter fails, but there are other possible SIM-related failures where a channel change may occur, even though the fault location circuits (see #5) are able to discern that the failure occurred in the SIM.

4. COUPLER FAULT HANDLING If the SIM detects an error in the received data from its own transmissions, it will change coupler channels (invert doublet polarity) at the beginning of the next wordstring. As mentioned in #3, the next wordstring may be corrupted during switch-over, but this will not cause an extra channel change.

   An internal error counter in the SIM is incremented whenever a channel change occurs, and is cleared whenever a wordstring is received with no errors (provided it is not the first wordstring of a channel change, when the error counter ignores all inputs). If three channel changes occur with no intervening ‘good’ wordstrings, a coupler fault (COUF) condition, 0, is entered, and can only be cleared by the TS command. During COUF, the coupler can be placed in receive only mode or powered down completely, but cannot enter transceiver mode. The SIM transmitter is also disabled. Note that COUF status is available with an RS command, regardless of whether a TS command has been issued.

5. FAULT LOCATION Considerable design effort has been expended to locate any errors with regard to whether they occurred in the SIM or in the coupler. In addition to the transmit stub monitor, proprietary circuits check the SIM receiver to detect the component failure. If an error appears to have occurred elsewhere than the coupler, a channel change may still occur, but the error counter will not be incremented and thus a coupler fault condition will not be entered. If non-coupler errors occur too frequently, the ARINC 629 terminal will eventually disable the SIM and coupler transmitters to prevent corruption of the bus.

Coupler Testing by the SIM

ARINC 629 uses several levels of error detection, which is generally combined with multiple levels of redundancy to insure system reliability. Each current mode coupler contains two separate channels, either of which is capable of driving the 629 bus. The SIM selects one of the channels while the other channel remains dormant. In order to check all channels for functionality, there must be some provision for the SIM to switch coupler channels upon reception of a system command.

The SIM constantly monitors the wraparound path including itself and the selected channel. In addition, a test function is available which selects nd monitors the spare coupler channel, and then reports the results via the status. The “BITE” (Built-In Test) mode uses regular data transmissions to check SIM and coupler functionality, to avoid corrupting the operation of the bus.

The test sequence begins with the TS command, which causes the next complete wordstring to be transmitted with the inverted doublet polarity. (If transmission of a wordstring is in progress when the BTC signal falls, nothing happens until the next wordstring.) At the end of the inverted wordstring, both channels of the coupler have been tested, because the SIM constantly monitors wraparound signal quality and thus has checked both ‘inverted’ and ‘non-inverted’ wor-

dstrings. When a bus quiet condition occurs after the inverted wordstring, the status codes become available with the RS command. If the coupler channel selected by ‘inverted’ polarity functions properly, the polarity remains ‘inverted’; if not, the SIM switches back to the original polarity.

The BITE sequence is requested by a TS command, and the sequence is actually entered when the following conditions are met:

1. transmission of a wordstring is not already in progress, and

2. monitoring of the last wordstring transmitted did not detect any errors.

The status register is cleared when the TS command is given.

The coupler test sequence may take a terminal interval or more to execute, which can be tens of milliseconds, and until the test is complete, the “no test” code is issued. The specific time required for the execution of a test sequence is as follows:

1. If transmission of a wordstring is in progress, the test will be delayed until the wordstring finishes.

2. The spare channel is tested during the next wordstring transmitted. This may begin almost immediately or as much as an aperiodic epoch, which can be several tens of milliseconds.

3. Test results become available after the end of the wordstring plus a delay of 0.75 microseconds, to allow data to wrap around through cabling delays and be processed.

A ‘no test’ status code, 3, can mean that the test sequence is still in progress (just described), that the SIM is defective (as determined by the transmit monitor), or that the SIM transmitter is not operating due to some input signal or a power fault trip disabling the SIM transmitter.

A ‘2 good’ code, 2, means that the SIM and both coupler channels are working properly.

A ‘1 good’ code, 1, means that the SIM and one coupler channel are working properly, but the other coupler channel either is defective or could not be tested. ARINC 629 allows a means for the SIM to determine whether the coupler has changed channels or not, based on the presence of an inversion of doublet polarity in the coupler wraparound path.

A coupler fault code, 0, on the status pins means that the SIM has been unable to find a functioning coupler channel, and has ceased all transmissions to prevent corruption of the 629 bus. The 0 code appears whether a BIT sequence has been requested or not, and cannot be cleared except by the TS command.

Data Flow Between Backplane And Bus

The four examples in this section show the various paths of data flow across the VX4469A. In all cases the Terminal IC, as directed by its personality RAM, moves data from the shared memory to the bus, or from the bus to shared memory. The important differences discussed here involve ways of moving data between shared memory and the backplane or system controller.

Examples A and B (Figures G–2 and G–3) show data being transferred directly in and out of shared memory. If the data is transferred as ASCII hex or decimal characters, the data is first translated between ASCII and binary with routines running in the 80186 before being transferred. If the data is transferred in 8-bit binary bytes, two bytes per 16-bit word, then the transfer goes directly via DMA between the backplane and shared memory. A binary transfer is much faster than ASCII.

If the data is being translated between ASCII and binary, the binary data is moved in or out of shared memory one 16-bit word at a time. If the data is being moved in binary/DMA mode, the data is moved in or out of shared memory in 8-bit bytes.

A result of this direct mode of transferring data is that it is not synchronized with activity on the bus. If data is in the process of being updated in shared memory when it is time for the terminal IC to transmit that data, the data transmitted could be partially new and partially old data. A similar situation exists for received data.

Examples C and D (Figures G–4 and G–5) show data being transferred via circular buffers. When the terminal IC is receiving or transmitting a label or label with data, the terminal IC can generate an ‘interrupt vector’. The VX4469A can use this interrupt vector to do various things, including moving data between circular buffers and shared memory. This way, the data movement is synchronized with the bus activity.

Set the VX4469A to use vector interrupts in the following manner. In the personality RAMs, all the transmit and receive cells can be programmed with an interrupt vector. The cells also contain two bits that enable strobing out the interrupt vector near the beginning of the label/data transfer with the bus or after the label/data transfer. The two bits may be enabled or disabled in any combination.

When an interrupt vector is strobed, hardware on the VX4469A stores the interrupt vector and timestamp and interrupts the 80186. The 80186 reads the interrupt vector and uses the low 13 bits of it as an index into a 4096-byte vector index table. Each location in the vector index table contains a number representing one of 256 instruction blocks. The instruction blocks contain functions that instruct the 80186 to do a variety of tasks.

The default value in all locations of the vector index table is instruction block 0. The default function in all of the instruction blocks, including block 0, is function 0. Function 0 is the terminating function that tells the 80186 that it is finished with this interrupt vector. Use the SVX and SVI commands to put other values in these areas as part of programming the VX4469A. See *Vector Instruction Block Functions* for a description of the functions that may be used in the instruction blocks.

The circular buffers also allow buffering a significant amount of data on the VX4469A. This allows time for the system controller to do other things besides read and/or write data, without losing any data.

Example A: Reading Data from Shared Memory Directly

This command programs the receive personality RAM:

0uwr,120,d100,n5,p6

0 selects terminal 0.

user friendly **w**rite **r**eceive personality RAM

ARINC 629 label **120**

store **d**ata at shared memory address **100**

expect a **n**umber of **5** words of data. If more than five words are received, no error will be declared, but only five will be stored. If fewer than five are received, a short string error will be generated in the terminal IC.

use offset **p**ointer table **6** in the multiple personality RAM.

This command programs the multiple personality RAM:

uwm,6,i0

user friendly **w**rite **m**ultiple personality RAM

offset **p**ointer table **6**

starting with an offset of 0, fill the entire 256 offset values in table 6 with offset values **i**ncrementing each value by **0**. In other words, fill the offset table full of zeros.

This command enables terminal 0:

se

set **e**nable

Now, whenever label 120 is transmitted on the bus, terminal 0 will copy the first five words of data after the label into shared memory for terminal 0 at address 100.

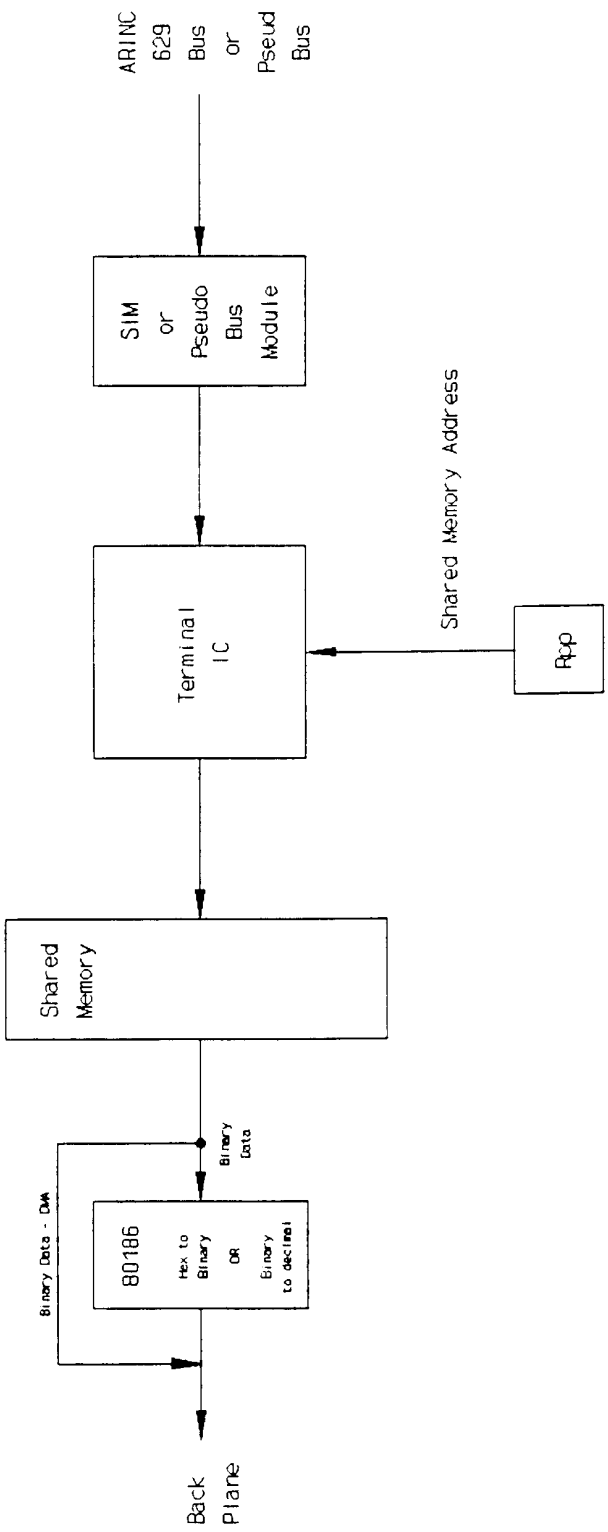


Figure G–2: Read Data from Shared Memory Directly

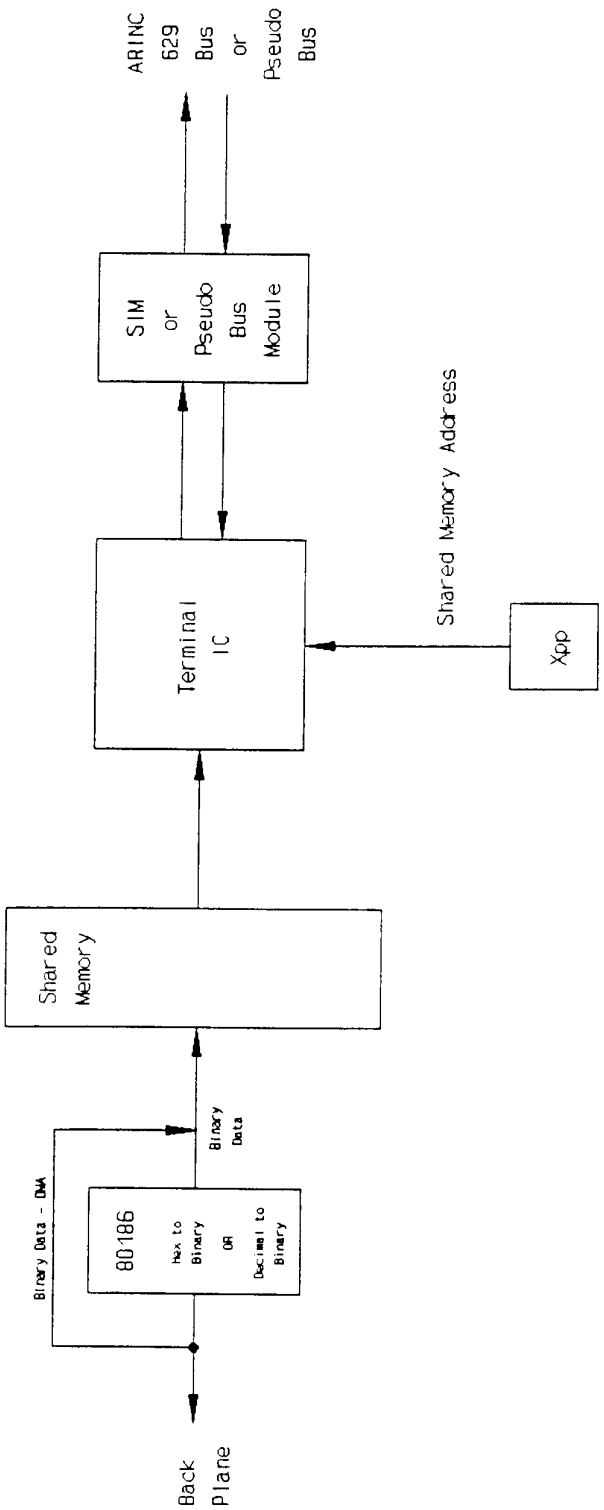


Figure G–3: Writing Data to Bus via Writes to Shared Memory

Use a read command for the system controller to read directly from the shared memory. The following command sets up the VX4469A to return the data:

rd,100

read **d**ata

starting from address **100**

Each read of the module after this command will return the address of the first word of the line plus a line of eight words separated by commas. This will continue, incrementing the address by eight each time, until a new command is sent to the VX4469A. A typical next command might be rd,100 again, to repeat sending the data starting at address 100.

Example B: Writing Data to Shared Memory Directly

This command programs a cell in the transmit personality RAM. It also programs the receive personality RAM to monitor transmitting this label.

uwx,0,0,b220,n7,d200

user friendly **w**rite **x**mit personality RAM.

cell in column **0**.

cell in row **0**.

la**b**el **220**.

number of data words is **7**.

get **d**ata to transmit from shared memory address **200**.

This command sets the y modulo for column 0:

uwx,0,y0

user friendly **w**rite **x**mit personality RAM.

column **0**.

y modulo is **0**.

This command programs the interval timer monitor values in the receive personality RAM:

uwr,L,9,2,10

user friendly **w**rite **r**eceive personality RAM.

a**ll** interval timer monitor cells (1 for each CID)

a transmit interval of **9**.

a terminal gap of **2**.

a sync gap of **10**.

This command programs the input pins of the terminal IC for TI, TG and SG:

si,9,2,10

set **i**nterval

a transmit interval of **9**.

a terminal gap of **2**.

a sync gap of **10**.

This command writes data directly into the shared memory:

wd,200,1,2,3,4,5,6,7

write **d**ata

starting at shared memory address **200**

data for word at 200 – **1**

data for word at 201 – **2**

data for word at 202 – **3**

data for word at 203 – **4**

data for word at 204 – **5**

data for word at 205 – **6**

data for word at 206 – **7**

This command enables the terminal IC:

se

set **e**nable

The wd command may be repeated as the terminal IC is transmitting. However, there is no synchronization with the data being transmitted, and a partially changed set of data might be transmitted.

Example C: Reading Data from Shared Memory Via a Circular Buffer

This command programs the receive personality RAM. For this command, the same commands as described in Example A will be used, with some additions to use a circular buffer.

0uwr,120,d100,n5,p6,L0d

0 selects terminal 0.

user friendly **w**rite **r**eceive personality RAM

ARINC 629 label **120**

store **d**ata at shared memory address **100**

expect a **n**umber of **5** words of data. If more than five words are received, no error will be declared, but only five will be stored. If fewer than five are received, a short string error will be generated in the terminal IC.

use offset **p**ointer table **6** in the multiple personality RAM.

generate a vector interrupt of **d** after (**l**ate) label 120 and five words of data have been received.

This command sets an instruction block number in the vector index table:

svx,d,5

set **v**ector inde**x**

for interrupt vector number **d**

to instruction block number **5**

This command sets the functions in instruction block 5:

svi,5,2,1,0,100

set **v**ector **i**nstruction

instruction block **5**

function **2**

circular buffer **1**

terminal **0** shared memory

shared memory address **100**

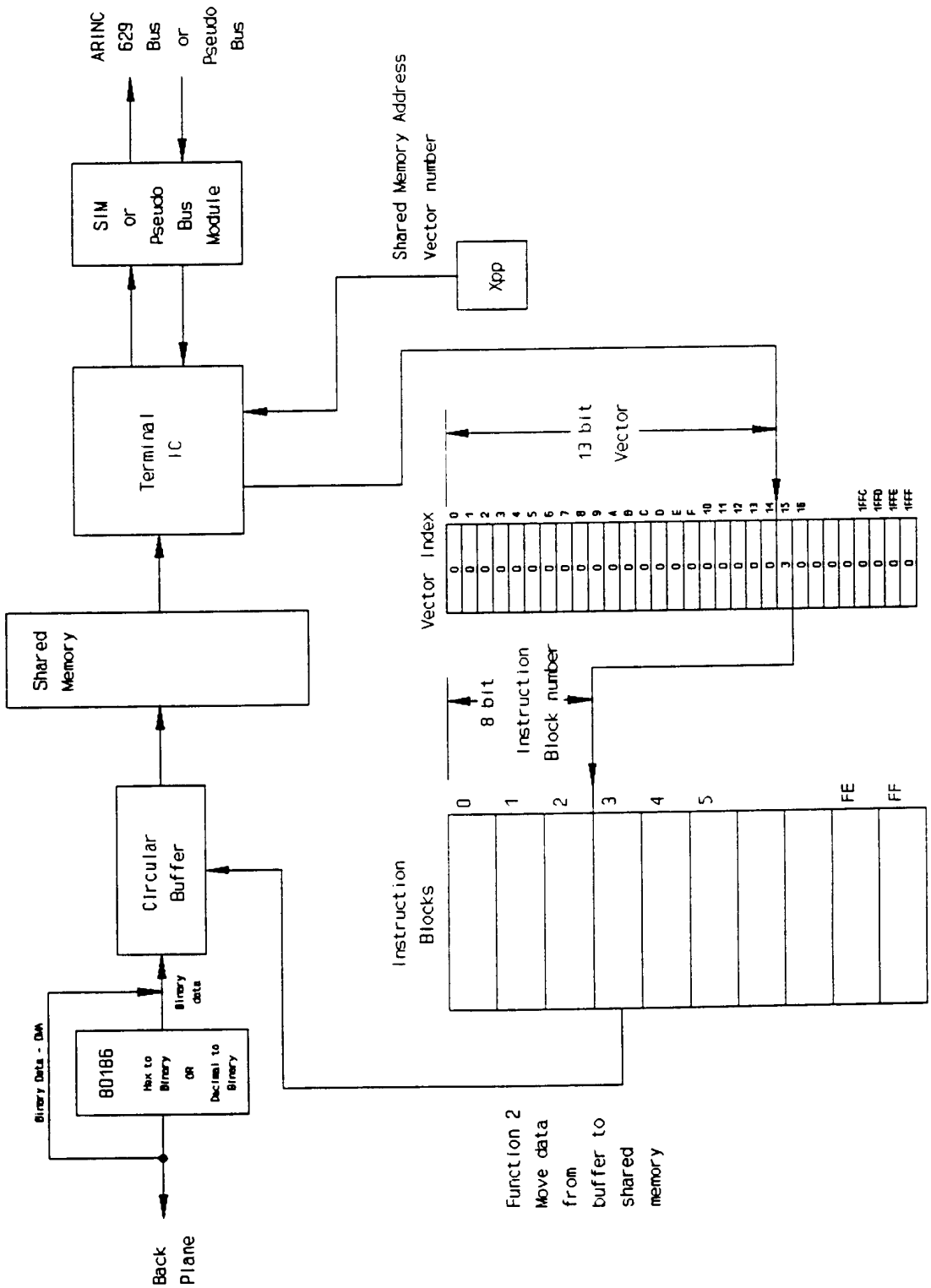


Figure G–4: Receiving Data from Bus via Circular Buffer

This command defines circular buffer 1:

dc,1,5,300

define **c**ircular buffer

circular buffer number **1**

5 words per cell

300 cells in circular buffer.

This command programs the multiple personality RAM:

uwm,6,i0

user friendly **w**rite **m**ultiple personality RAM

offset **p**ointer table **6**

starting with an offset of 0, fill the entire 256 offset values in table 6 with offset values, **i**ncrementing each value by **0**. In other words, fill the offset table full of zeros.

This command enables terminal 0:

se

set **e**nable

Now, whenever label 120 is transmitted on the bus, terminal 0 will copy the first five words of data after the label into shared memory for terminal 0 at address 100. Terminal 0 will then strobe a vector interrupt of 0d into the VX4469A hardware. The 80186 will be interrupted and read the interrupt vector of 0d. The 80186 will use the 0d as an index into the vector index table and find instruction block number 5. The 80186 will execute the functions in instruction block 5. The first function is ‘2’ which tells it to transfer data to a circular buffer. The number of words it transfers is five, from the circular buffer definition. The following command sets up to read the data from the circular buffer:

rc,1

read **c**ircular buffer

number **1**

Each read will return five words and delete them from the circular buffer. If there is no data in the circular buffer, a space, carriage return and line feed characters will be returned, and a new read command must be sent to read more data.

If the circular buffer is full, no data will be transferred to the circular buffer until at least one cell of the circular buffer has been read.

The system controller may also read directly from the shared memory, using the read command. The data will still also be stored in circular buffer 1. The following command sets up the VX4469A to return the data directly from shared memory:

rd,100

read **d**ata

starting from address **100**

Each read of the module after this instruction will return the address of the first word of the line plus a line of eight words separated by commas. This will continue, incrementing the address by eight each time, until a new command is sent to the VX4469A. A typical next command might be ‘rd,100’ again, to repeat sending the data starting at address 100.

Example D: Writing Data to Shared Memory via a Circular Buffer

This command programs a cell in the transmit personality RAM. It also programs the receive personality RAM to monitor transmitting this label. In Example D, Example B is expanded to use a circular buffer.

uwx,0,0,b220,n7,d200,L15

user friendly **w**rite **x**mit personality RAM.

cell in column **0**.

cell in row **0**.

la**b**el **220**.

number of data words is **7**.

get **d**ata to transmit from shared memory address **200**.

generate an interrupt vector **15** after (**l**ate) label 220 and 7 data words have been transmitted.

This command sets an instruction block number in the vector index table:

svx,15,3

set **v**ector inde**x**

for interrupt vector number **15**

to instruction block number **3**

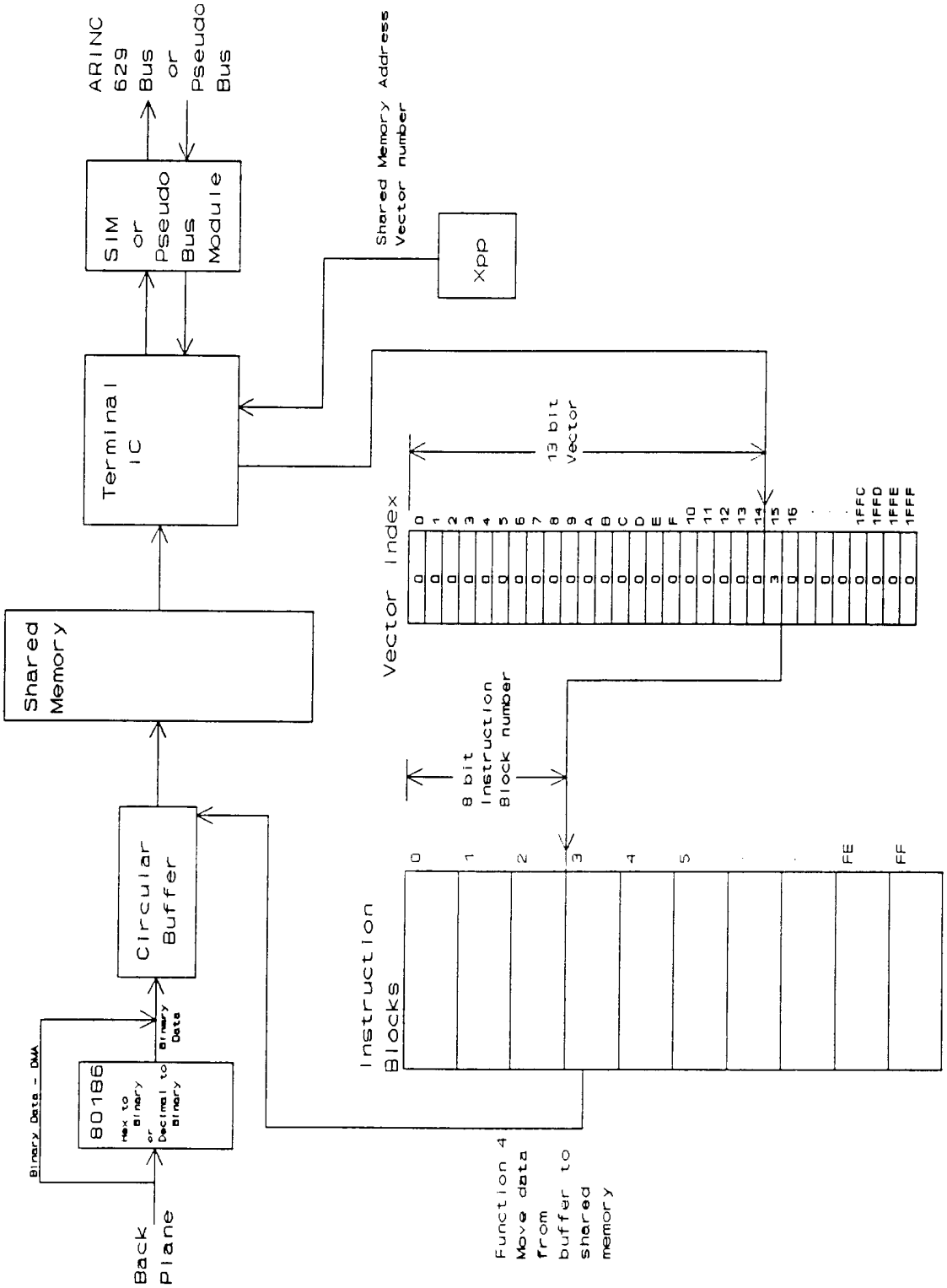


Figure G–5: Writing Data to Bus via a Circular Buffer

This command sets the functions in instruction block 3:

svi,3,4,9,0,200

set **v**ector **i**nstruction

instruction block **3**

function **4**

circular buffer **9**

terminal **0** shared memory

shared memory address **200**

This command defines circular buffer 9:

dc,9,7,800

define **c**ircular buffer

circular buffer number **9**

7 words per cell

800 cells in circular buffer.

This command sets the y modulo for column 0:

uwx,0,y0

user friendly **w**rite **x**mit personality RAM.

column **0**.

y modulo is **0**.

This command programs the interval timer monitor values in the receive personality RAM:

uwr,L,9,2,10

user friendly **w**rite **r**eceive personality RAM.

a**l**l interval timer monitor cells (1 for each CID)

a transmit interval of **9**.

a terminal gap of **2**.

a sync gap of **10**.

This command programs the input pins of the terminal IC for CID:

si,9,2,10

set **i**nterval

a transmit interval of **9**.

a terminal gap of **2**.

a sync gap of **10**.

This command writes data directly into the shared memory for the first time data is transmitted:

wd,200,1,2,3,4,5,6,7

write **d**ata

starting at shared memory address **200**

data for word at 200 – **1**

data for word at 201 – **2**

data for word at 202 – **3**

data for word at 203 – **4**

data for word at 204 – **5**

data for word at 205 – **6**

data for word at 206 – **7**

This command writes data into circular buffer 9. Each set of seven numbers is stored as a cell in the circular buffer.

wc,9,11,12,13,14,15,16,17

write **c**ircular buffer data

circular buffer number **9**

data for cell **11,12,13,14,15,16,17**

wc,9,21,22,23,24,25,26,27

write **c**ircular buffer data

circular buffer number **9**

data for cell **21,22,23,24,25,26,27**

wc,9,31,32,33,34,35,36,37

write **c**ircular buffer data

circular buffer number **9**

data for cell **31,32,33,34,35,36,37**

This command enables the terminal IC:

se

set **e**nable

The wc command may be repeated as the terminal IC is transmitting. There is now synchronization with the data being transmitted, and a partially changed set of data should not be transmitted.

If the terminal IC transmits data and the circular buffer is empty, the old data will be transmitted until data is available in the circular buffer.

The terminal IC may, of course, be programmed to transmit and receive a very complex set of labels and data using a variety of circular buffers and vector interrupt instruction blocks.

CRC Support

Cyclic redundancy check (CRC) is a powerful means of error detection. The originator of a wordstring calculates a CRC word from the contents of the wordstring and appends this word to the end of the wordstring. The receiver again calculates the CRC to verify that the wordstring was received correctly.

To calculate a CRC, all the bits of a wordstring are strung together and then divided, modulo 2, by a number. The remainder of this division is the CRC which is appended to the end of the wordstring.

Error detecting with a CRC is described in general in the January 1961 issue of the Proceedings of the IRE (now IEEE) in a paper titled “Cyclic Codes for Error Detection”. The method used by the VX4469A is a 16-bit CRC described in the ARINC SPECIFICATION 629 and with one difference, more specifically in the ARINC SPECIFICATION 429-12, Appendix 7.

In this method, the divisor is 10001000000100001 binary or the polynomial $x^{16} + x^{12} + x^5 + 1$, the first 16 bits of the wordstring are 1’s complemented or the CRC accumulator is initialized to all 1’s, and the remainder (the CRC word) is appended to the wordstring. If the receiver calculates the CRC on a received wordstring including the originator’s CRC word, the result should always be 0000 hex. For calculating the CRC, the wordstring is processed as transmitted, starting with the most significant bit of the label. This method is the same as the

algorithm used in ARINC 429, except that the CRC is not complemented before being transmitted.

In ARINC 629, the label is included in the CRC calculation. As an example, if the label is 0 and the data string is the 5 words 1, 2, 3, 4, and 5, the CRC appended would be 6D8E hex.

0000	label
0001	data word
0002	data word
0003	data word
0004	data word
0005	data word
6D8E	originator's CRC

If the receiver calculates a CRC on these 7 words or any wordstring using this CRC algorithm, the result would be 0000 hex.

0000	label
0001	data word
0002	data word
0003	data word
0004	data word
0005	data word
6D8E	originator's CRC
0000	receiver's CRC

For the following descriptions, 'label' is a 16-bit number that includes the channel ID in the high 4 bits.

The VX4469A supports CRC with the following commands:

WDC,addr,label,data,...,data<tm>	Write Data, calculating and appending a CRC, to shared memory starting at 'addr', including 'label' in the CRC calculation.
WCC,cbnum,label,data,...,data<tm>	Write data to a Circular buffer, calculating and appending a CRC. Circular buffer number is 'cbnum' and include 'label' in the calculation.
RDC,addr,label,count<tm>	Read Data from shared memory, calculating and appending a CRC. 'Label' and 'count' shared memory words are used to calculate the CRC. 'Count' shared memory words are returned followed by the calculated CRC. The CRC is not written to shared memory; it is only appended to the end of the returned data.

RDV,addr,label<tm>	Read Data Variable from shared memory, calculating and appending a CRC, using the word at 'addr' in shared memory as the 'number of data words' in the wordstring. 'Label' and 'number of data words' shared memory words are used to calculate the CRC. 'Number of data words' shared memory words are returned with the calculated CRC. The CRC is not written to shared memory, it is only appended to the end of the returned data.
RCC,cbnum,label<tm>	Read Circular buffer, calculating and appending a CRC, using the circular buffer cell size as the 'number of data words' in the wordstring. 'Label' and 'number of data words' circular buffer words are used to calculate the CRC. 'Number of data words' circular buffer words are returned with the calculated CRC. The CRC is not written, it is only appended to the end of the returned data.
RCV,cbnum,label<tm>	Read Circular buffer Variable, calculating and appending a CRC, using the first data word in the circular buffer cell or the cell size, whichever is less, as the 'number of data words' in the wordstring. 'Label' and 'number of data words' circular buffer words are used to calculate the CRC. 'Number of data words' circular buffer words are returned with the calculated CRC. The CRC is not written, it is only appended to the end of the returned data.

The VX4469A also supports CRCs with the following interrupt vector functions:

d,countmode,count,label,terminal, address	Interrupt on CRC error. 'd' is the function code. 'countmode' determines whether 'count' is a maximum number of data words and the word in shared memory 'address' is the 'number of data words' (variable) or if 'count' is the 'number of data words'. 'label' is included as part of the data for calculating the CRC along with 'number of data words' in 'terminal's shared memory starting at 'address'. No CRC is written, but if the result is not 0000 hex, then a system interrupt occurs and bit 4 in the interrupt register is set.
e,countmode,count,label,terminal, address	Calculate and append a CRC. 'e' is the function code. 'countmode' determines whether 'count' is a maximum number of data words and the word in shared memory 'address' is the 'number of data words' (variable) or if 'count' is the 'number of data words'. 'label' is included as part of the data for calculating the CRC along with 'number of data words' – 1 in 'terminal''s shared memory starting at 'address'. The CRC is written in shared memory at location ['address' + 'number of data words' – 1].

Hardware CRC

The VX4469A also has hardware capability for generating CRCs on transmitted wordstrings. The CRC hardware reads the label and data at the same time the terminal IC does. The CRC hardware calculates the CRC and supplies the CRC word to the terminal IC at the time the terminal IC is reading the last word of the wordstring. The CRC hardware works with the terminal IC and does not require any bus time from the 80186.

All that is required to use the CRC hardware is to set true (low) bit 0 in the fifth word of the XPP cell for each wordstring that requires a CRC. Use the UWX command or WX command to do this.

The CRC is calculated on the label and n-1 data words. The nth word transmitted will be the CRC. The contents of shared memory are not modified. For fixed length wordstrings, n is the number of words stored in the XPP transmit cell. If the wordstring is variable length, the number of words is the first data word of the wordstring. N must be at least 2 for the CRC hardware to work.

The CRC hardware will use the proper channel ID even if the label CID functions are used.

Memory Switching

Each terminal installed on a VX4469A has a data or shared memory address space of 64 Kwords. The VX4469A has two modes which can switch the addressing of the terminal ICs.

In the invert mode, the VX4469A can switch the upper and lower 32 Kwords of this memory for the terminal IC. This allows you to examine or update one half of the shared memory while the terminal IC is reading or writing the other half.

In the second mode, OR mode, the VX4469A can switch the terminal ICs addressing the lower 32 Kwords to the upper 32 Kwords. Terminal IC addressing in the upper 32 Kwords remains the same. This mode is useful if it is desired to use circular buffers and memory switching.

The memory addressing for the commands such as RD and WD does not change. The memory addressing for the terminal IC does change.

When memory is normal, addresses for the terminal IC are the same as the address for the RD and WD commands. When memory is switched, addresses for the terminal IC have the high order address bit inverted in invert mode or logically ORed with 1 in OR mode. If the terminal IC was programmed in its RPP and MPP to store data at location 100, it will store data at location 100 when memory is normal. If memory is switched, it will store data at location 8100. If the terminal IC was programmed to store data at C000, it will store data at C000 when memory is normal or switched in OR mode. It will store data at 4000 if switched in invert mode. The same applies to data for transmitting.

There are eight commands to support this feature.

The LH command reads the current status of the addressing on as many terminals as are installed on the VX4469A. If there are three terminals installed, the LH command will set up the VX4469A to return a string in this format:

0c 1c 2c<cr><lf>

where c is either N for normal or S for switched in invert mode, or O for switched in OR mode.

The HR command will reset the memory switch logic for the current terminal to normal addressing. It will also cancel any requested switches. This is done asynchronously to any data being transferred.

The HS command will set up the VX4469A to change the memory addressing for the current terminal to the switched state. The change will occur at the beginning of the next transmitted or received wordstring. If a wordstring is being transmitted or received at the time this command is given, the change will not occur until the beginning of the next wordstring.

The HN command will set up the VX4469A to change the memory addressing for the current terminal to the normal state. The change will occur at the beginning of the next transmitted or received wordstring. If a wordstring is being transmitted or received at the time this command is given, the change will not occur until the beginning of the next wordstring.

The HPS command will set up the VX4469A to change the memory addressing for the current terminal to the switched state. The change will occur at the beginning of this terminal's next transmitted wordstring whose Xpp cell has the switch bit set. The switch bit is bit 1 of the 5th byte in the XPP cell. This bit may be set with the UWX command or the WX command.

The HPN command will set up the VX4469A to change the memory addressing for the current terminal to the normal state. The change will occur at the beginning of this terminal's next transmitted wordstring whose XPP cell has the switch bit set. The switch bit is bit 1 of the 5th byte in the XPP cell. This bit may be set with the UWX command or the WX command.

The SHO command will set up all terminals to use OR mode if any of them are switched.

The SHS command will set up all terminals to use invert mode if any of them are switched.

Interrupt Vectors and Timestamp

The ARINC 629 protocol terminal IC provides for the generation of “interrupt vectors”. Interrupt vectors are user-defined numbers that the terminal IC outputs at the time of transmitting specific wordstrings or receiving specific labels. You can program these interrupt vector numbers into the receive and transmit personality PROMs.

Each terminal on the VX4469A includes a 36-bit hardware FIFO (First In, First Out buffer) that will store these interrupt vector numbers along with the current timestamp time. The VX4469A processor is interrupted whenever something is in one of the FIFOs. It then reads the FIFOs, and can perform a variety of functions based on the interrupt vector number and programming by the user. See Appendix F, and commands SVX, SVI, DC, WC, and RC.

In the transmit personality PROM, each of the cells in the 31-by-31 array defining a wordstring to transmit can be programmed with an interrupt vector number. Each cell also includes two bits to enable generating this interrupt vector at the beginning (early) and/or end (late) of the wordstring. Each of the 4092 cells in the Rpp also contains space for an interrupt vector. See commands UWX and UWR for programming the interrupt vector numbers and enabling bits.

Figure G–6 shows how the VX4469A is connected to the terminal ICs to take advantage of the interrupt vectors. When the VX4469A is programmed to put an interrupt vector in a circular buffer, the source of the bits in the circular buffer are as follows:

Bits 4 through 12 are always the same as bits 4 through 12 in the personality PROM.

Bit 13 is the same as the string error bit in the terminal IC’s error register. If this bit is set, an error was detected in this wordstring.

Bit 14 is the terminal IC’s string active pin (STAC). If this bit is 1, this is an early interrupt vector. If it is 0, this is a late interrupt vector.

Bit 15 is a status bit from the FIFO itself indicating that the FIFO is almost full. If this bit is set, you should assume that the VX4469A processor is falling behind in processing the interrupt vectors, and some interrupt vectors may have been lost.

When receiving a label, a standard terminal IC will output the same interrupt vector no matter what the label extension (transmitter’s CID); although the terminal IC can be programmed to put the data at different memory locations depending upon the label extension. The VX4469A provides extra hardware to substitute the label extension for the low four bits (bits 0 through 3) of the interrupt vector.

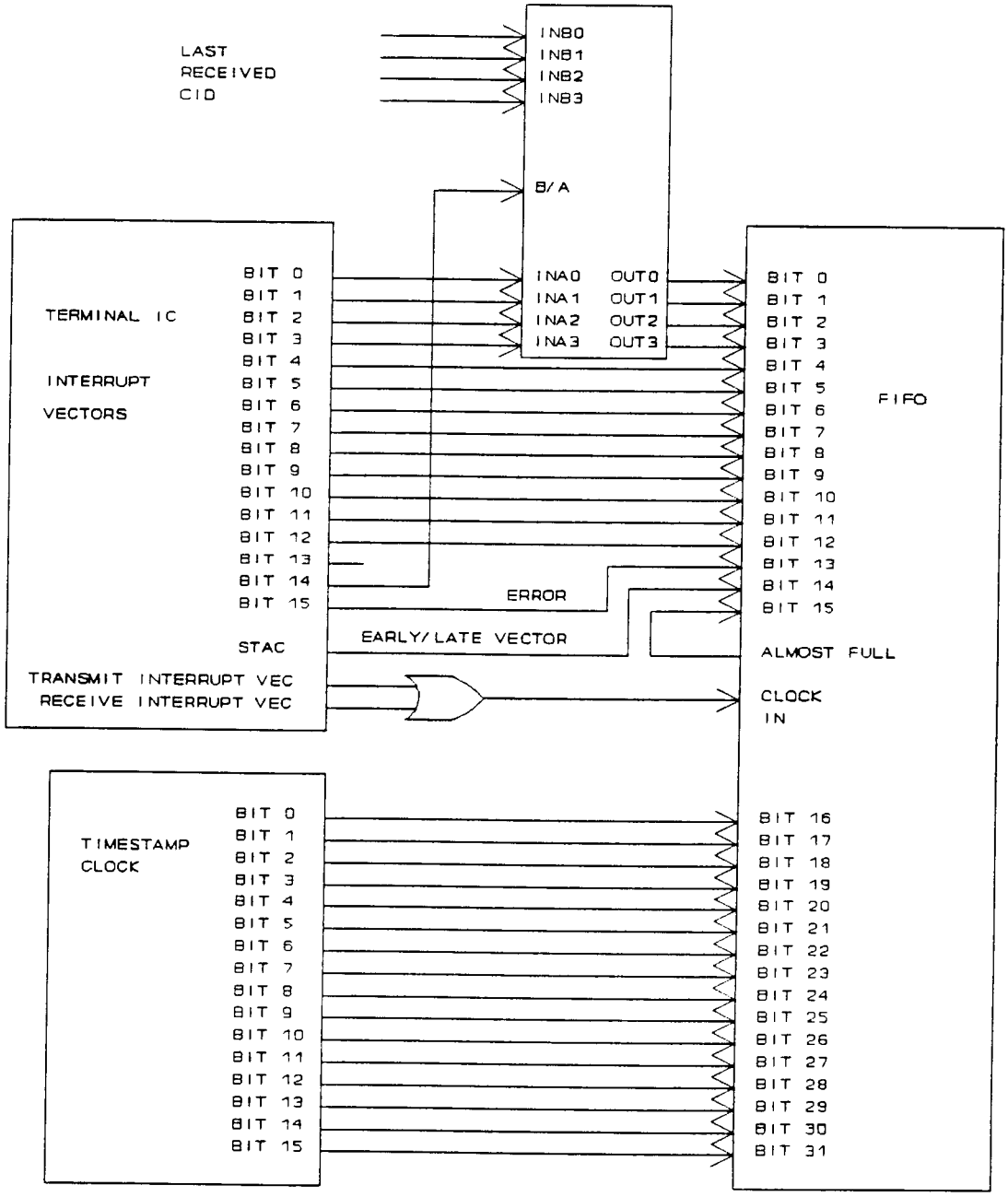


Figure G–6: Interrupt Vectors

Bits 0 through 3 depend on how bit 14 of the interrupt vector in the personality PROMs is programmed. If bit 14 is 0, then bits 0 through 3 are the same as bits 0 through 3 of the interrupt vector in the Personality PROMs. If bit 14 is 1, then bits 0 through 3 are the label extension of the last received label. This feature works properly only for late interrupt vectors, not early.

For example, suppose you want to monitor the data transmitted by the left, center, and right Framistaff LRUs. The Framistaff data has a label of 100 hex or 256 decimal and two data words. The left, center, and right LRUs have channel IDs of 0, 1 and 2. The following commands would set up the VX4469A terminal 0 to save LRU ID, time, and data for each of these LRUs' transmissions in circular buffer 0. All the commands are shown using hex numbers.

```
ia                          ; initialize VX4469
uwr,100,d200,n2,p0,L4100    ; receive personality prom – label 100
                            ; write data at base address 200
                            ; write 2 words of data
                            ; use Mpp offset table 0
                            ; generate a late interrupt vector of 100
                            ;  with bit 14 set so the low 4 bits will
                            ;  be the transmitters CID
uwm,0,0,i2                  ; set up the Mpp table to write data in
                            ;  shared memory at address
                            ;  200 + (2 * label extension)
svx,100,1                   ; if vector interrupt is 100, execute
                            ;  instruction block 1
svx,101,2                   ; if vector interrupt is 101, execute
                            ;  instruction block 2
svx,102,3                   ; if vector interrupt is 102, execute
                            ;  instruction block 3
svi,1,1,0,0,200             ; instruction block 1
                            ;  function 1 – vector,timestamp,data
                            ;  circular buffer 0
                            ;  terminal 0, data address 200
svi,2,1,0,0,202             ; instruction block 2
                            ;  function 1 – vector,timestamp,data
                            ;  circular buffer 0
                            ;  terminal 0, data address 202
svi,2,1,0,0,204             ; instruction block 3
                            ;  function 1 – vector,timestamp,data
                            ;  circular buffer 0
                            ;  terminal 0, data address 204
dc,0,4,1000                 ; define circular buffer 0
                            ;  4 words per cell
                            ;  1000 cells
se                          ; enable terminal 0
rc,0                        ; read from circular buffer 0
```

The following data is an example of what might be read back from the VX4469A.

```
0100,E88B,0000,0003,0101      ; interrupt vector 100, (label is 100, CID ;  is 0)
                              ; low order 16 bits timestamp is E88B
                              ; high order timestamp is 0000
                              ; first Framistaff data word – 0003
                              ; second Framistaff data word – 0101
0101,E88E,0000,0003,00FE      ; interrupt vector 101, (label is 100, CID ;  is 1)
                              etc...
0102,E891,0000,0003,0104
0100,F443,0000,0003,0101
0101,F446,0000,0003,00FE
0102,F449,0000,0003,0104
0100,FFFB,0000,0003,0101
0101,FFFE,0000,0003,00FE
0102,0001,0001,0003,0104
0100,0BB3,0001,0003,0101
0101,0BB6,0001,0003,00FE
0102,0BB9,0001,0003,0104
0100,176B,0001,0003,0101
0101,176E,0001,0003,00FE
0102,1771,0001,0003,0104
0100,2323,0001,0003,0101
0101,2326,0001,0003,00FE
0102,2329,0001,0003,0104
0100,2EDB,0001,0003,0101
0101,2EDE,0001,0003,00FE
```

Test Modes

Each terminal of a VX4469A can be operated in three modes. The default mode (DATAC mode) uses the DATAC or terminal IC to control the Serial Interface Module (SIM) and its communication with the ARINC 629 bus it is connected to. When set to Test Mode 1 or 2 (command SL,1 or SL,2) the terminal's LCA is reprogrammed to directly control the SIM and the data transmitted to the 629 bus.

The Test Modes allow the user to transmit word strings with many types of errors to test the response of the UUT to those errors. In test mode, the terminal does not monitor its own transmission and will not terminate on any errors including collisions.

The Test Mode 1 uses TI, TG, and SG timers the same as DATAC mode. The bus is monitored and the terminal will begin transmitting at the time determined by its timers. The terminal's personality RAM becomes Test RAM and the user loads the Test RAM with the data to be transmitted.

Test Mode 2 does not have the TI, TG and SG timers. In Test Mode 2, transmission is initiated by an external event. These events can be a command from the user, a VXI TTL trigger input, an external trigger input, or a wordstring label detected on the bus.

The Test RAM is 128 Kbytes and is segmented into 16 equal 8K byte segments. Each byte of Test RAM corresponds to one bit time on the 629 bus. A normal 629 label or word consists of a three bit time sync pattern: 16 bit times for 16 data bits (high order bit first), and one bit time for a parity bit. A normal 629 word is 20 bit times and requires 20 bytes of Test RAM.

Data is written by the user into Test RAM as a series of messages. One message is transmitted each TI or as bus traffic permits. A message is stored in Test RAM with a 2 byte bit time count, that many bytes of bit description, and a control byte. The control byte determines looping, segment change, and terminal disabling.

The Test RAM may not be written to while its terminal is enabled, but the user may switch randomly between segments while the terminal is enabled.

Data to transmit may be written across segment boundaries. The user is responsible for determining the length of data in each segment and whether data crosses a segment boundary.

ARINC 629 Data Format

Data transmitted from the terminal IC or the LCA to the SIM is by logic level as a stream of bits with Manchester coding. The value of a bit depends on the direction of the logic level transition in the middle of a bit time. A bit value of one is a transition from high to low and a bit value of zero is a transition from low to high. If two bits in a row have the same value, a transition between bit times is necessary.

To synchronize the receiving terminal to the transmitting terminal, each data or label word starts with a sync pattern. A sync pattern consists of three bit times. The first and third bit times contain no transitions. The second bit time contains a bit value of one (a high to low transition) for a label word. For a data word, the second bit time contains a bit value of zero (a low to high transition).

The final bit in a label or data word is a parity bit. ARINC 629 uses odd parity. The value of the parity bit is made such that the sum of all the bits in a label or data word is odd. The value of the bit in the sync pattern (a one for a label, a zero for a data word) is included in this calculation.

A message, the data transmitted by a terminal when it is its turn to transmit, consists of one or more word strings. A word string consists of a label word and optional data words. Word strings in one message are separated by 4 bit times of no transitions.

VX4469A Test Mode Programming

The VX4469A supports Test Mode with a number of commands:

nSL,n	puts terminal n into Test Mode 1 or 2.
nSI,ti,tg,sg	programs the timers for a terminal in Test Mode 1.
nSKI	immediately transmit next portion of test ram. Test Mode 2.
nSKL,b,c	enables transmitting from the test ram whenever label b is received. The VX4469A pauses c bit times after label b is received and then transmits the next portion of the test ram. Test Mode 2.
nSKV,c	enables transmitting from the test ram whenever this terminal detects a VXI TTL trigger input. The VX4469A pauses c bit times after the trigger is received and then transmits the next portion of the test ram. Test Mode 2.
nSKX,c	enables transmitting from the test ram whenever this terminal detects an external trigger input. The VX4469A pauses c bit times after the trigger is received and then transmits the next portion of the test ram. Test Mode 2.
nSKD	disables transmitting from the test ram. Test Mode 2.
nSE	enables terminal n
nSD	disables terminal n
nSG,n	writes n to the Segment Register. The Segment Register is written to the high four bits of the Test Mode Program Counter when the LCA is enabled, a loop occurs, or the LCA switches to a new segment.
nLG	sets up the VX4469A to return the current value of the high four bits of the Test Mode Program Counter. When the LCA starts a segment, the high four bits of its Test Mode Program Counter are loaded from the Segment Register.
nRT,s,a	sets up to return the contents of Test RAM starting at segment s, offset a.
nBRT,s,a,n	sets up to return the contents of Test RAM in binary mode starting at segment s, offset a, n bytes.
nWT,s,a,data	writes data to Test RAM starting at segment s, offset a.
nBWT,s,a,n;data	writes binary data to Test RAM starting at segment s, offset a, n bytes.
nUWT,s,B,..	user friendly write to Test RAM starting at the beginning of segment s.
nUWT,s,A,..	user friendly write to Test RAM appending to the data already written to segment s with the UWT command.

The UWT command has the following arguments:

Ln[Mm][P][S]	write a 20 bit label word of n. optional Mm will make bit m a missing bit. optional P will invert the parity bit. optional S will invert the sync pattern.
Dn[Mm][P][S]	write a 20 bit data word of n. optional Mm will make bit m a missing bit. optional P will invert the parity bit. optional S will invert the sync pattern.
C[Mm][P][S]	write a 20 bit data word which is the CRC of the preceding label and data words. optional Mm will make bit m a missing bit. optional P will invert the parity bit. optional S will invert the sync pattern.
Bc[c...]	write bits of value c. c may be 0, 1, H, or L. 0 is a normal bit of value 0, 1 is value 1, H is a high level with no transitions for one bit time, and L is a low level with no transitions for one bit time.
Nn	write a byte to Test RAM with the value of n. This allows the user to program transition times within a bit time with a resolution of 1/8th of a bit time. The data in a Test RAM byte is shifted out to the SIM at 16 MHz, high order bit first. n=F0 would be a normal bit of value 1.
En	E terminates a message. The number of Test RAM bytes in the message is put at the beginning of the message and a control byte of value n is placed at the end of the message. The terminal will stop transmitting until the next TI or when bus traffic allows in Test Mode 1 or the next enabled trigger in Test Mode 2. n = 0 means continue. The LCA will read the next two bytes from Test RAM as the count for the next message and continue from there when it is time to transmit again. n = 1 means loop. The LCA will read the two bytes at the beginning of the segment last written with the SG,n command. If no new SG,n command has been issued, it will loop to the beginning of its current segment. n = 2 means start a new segment if an SG,n command has been issued, otherwise continue on in this segment when it is time to transmit again. This allows switching to a new segment before the end of the current segment. n = 4 means disable this terminal. This is useful for transmitting a message or set of messages only once. Each message series must end with a control byte value of 1 or 4.

Example: UWT,3,b,b1,L123,d234,d345,d4321,e0,

ARINC 629 Multi-Transmitter Data Bus

ARINC 629 is a digital serial data bus developed by Boeing Aircraft for transferring data and control information within airplanes. A 629 bus may be shared by up to 120 transmitters without a separate controller. The protocol avoids collisions between transmitters, insures that all transmitters have equal access to the bus, and allows the bus to operate at full bandwidth. The bus would be suitable for a wide range of other applications. The following describes the basic protocol of ARINC 629.

Aeronautical Radio, Inc. (ARINC) Specification 629 describes a multi-transmitter serial data bus for use in commercial aircraft. Each device interfaced to the bus may take turns broadcasting data on, as well as receiving data from, the bus. A device may have more than one terminal to interface it to more than one bus.

Each device is interfaced to the bus with a terminal controller, a serial interface module, and a current coupler. The bus itself is a twisted pair current loop with 130 ohm termination at each end. The terminal controller is programmed with a "personality" ROM. A typical device using this interface would share some memory with the terminal controller. (See Figure G–7.)

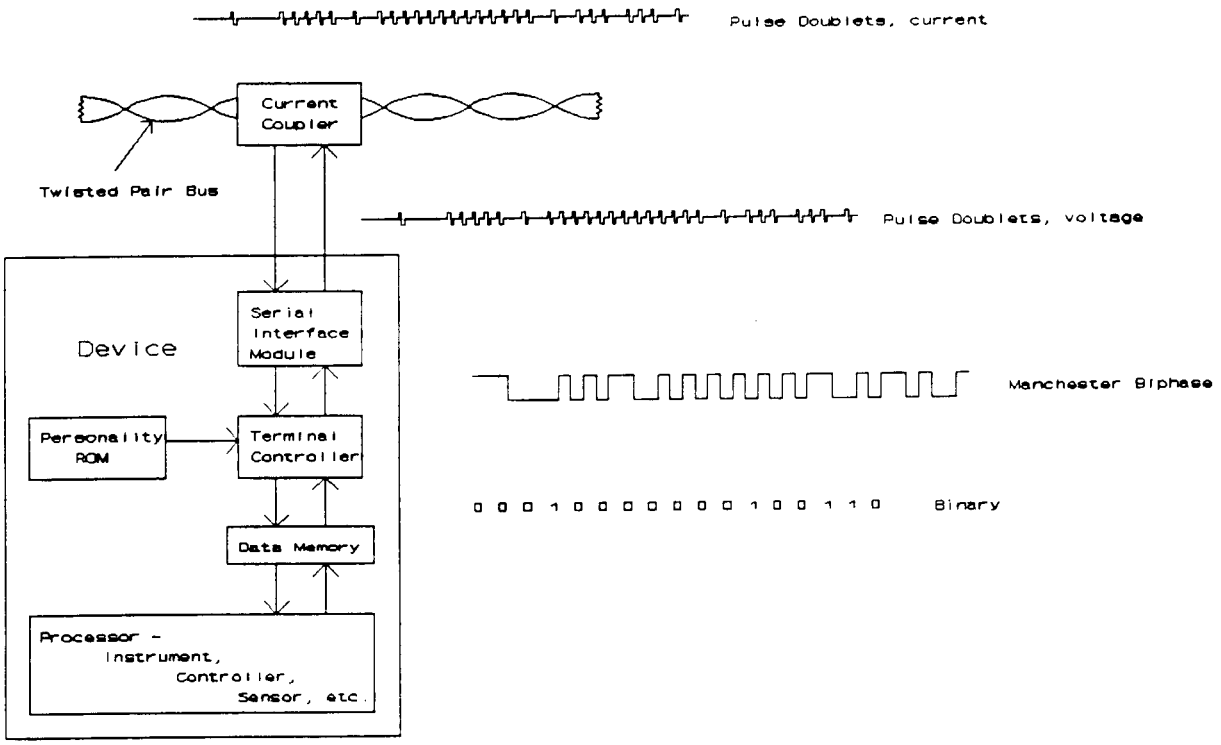


Figure G–7: ARINC 629 Terminal

Each terminal monitors bus activity and maintains three timers that avoid collisions, insure equal access to all terminals, and allow full bandwidth without a common controller.

Data Format

A terminal transmits one message at a time. A message consists of one to 31 wordstrings. A wordstring is made of one to 257 words. Each word has three bits of sync, 16 bits of data, and one bit of parity.

The first word of a wordstring is an identifying label. Twelve least significant bits of the label identify the data following in the wordstring and four most significant bits identify a channel ID number. Redundant devices on the same bus would transmit the same label and data with a different channel ID.

There may be from zero to 256 16-bit data words following the label in a wordstring. The number of data words associated with a particular label may be constant or variable.

Words in a wordstring are strung together with no time in between.

There is a bus quiet time of four bit-times between wordstrings in a single message.

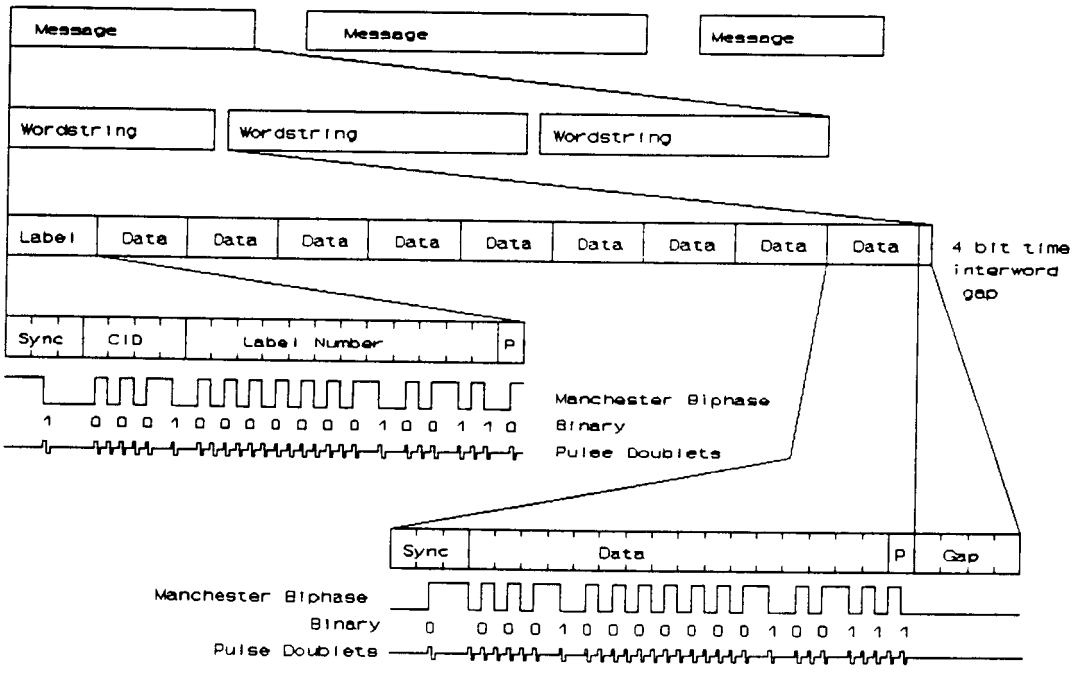


Figure G–8: ARINC 629 Data Format

The data is Manchester encoded when transferred between the terminal controller and the serial interface module. The serial interface module sends and receives pulse doublets to and from the current coupler. The current coupler magnetically couples the pulse doublets with the twisted pair. There is one pulse doublet for each transition of the Manchester signal. (See Figure G–8.)

Bus Access Control

Each terminal controller monitors the bus and uses three internal timers to determine its own time to begin transmitting a message.

There is no external controller.

The three timers are the Transmit Interval (TI), Terminal Gap (TG) and Synchronization Gap (SG).

All the terminals on a bus use the same value for their Terminal Interval (TI) timer. If a bus is not overloaded, each terminal will transmit once within a TI time interval. The TI time is set for the fastest data update rate required by the devices on the bus. The TI timer in the terminal controller may be set to 1000 to 128,000 bit-times in 1000 bit-time increments.

Each terminal on a bus will have a different value for its Terminal Gap (TG) timer. The TG timer avoids collisions on the bus (two terminals starting to transmit at the same time). A terminal partly determines it is its turn to transmit when there has been a silence on the bus of its TG timer value. TG timer values vary from seven to 255 bit times in two bit-time increments.

The third timer, the Synchronization Gap (SG) timer, insures that each terminal has a turn to transmit. All the terminals on a bus use the same SG time. The SG time needs to be longer than the longest TG used on the bus. After a terminal has transmitted, it will not be able to transmit again until after there has been a bus quiet time SG long. This allows all terminals to complete their TG times and transmit before a terminal transmits a second time. The SG timer may be set to 35, 67, 131 or 257 bit times.

A terminal will transmit when all three of its timers have completed.

The Transmit Interval timer is reset when the terminal begins to transmit. It will then count to completion.

The Synchronization Gap timer is reset when a terminal begins to transmit. Until the SG timer has counted to completion, it will be reset whenever the bus is busy. After it has counted to completion it will not reset, even if the bus is busy, until the terminal transmits again.

The Terminal Gap timer is reset whenever the bus is busy. It is inhibited from counting until the SG timer has completed. (See Figure G–9.)

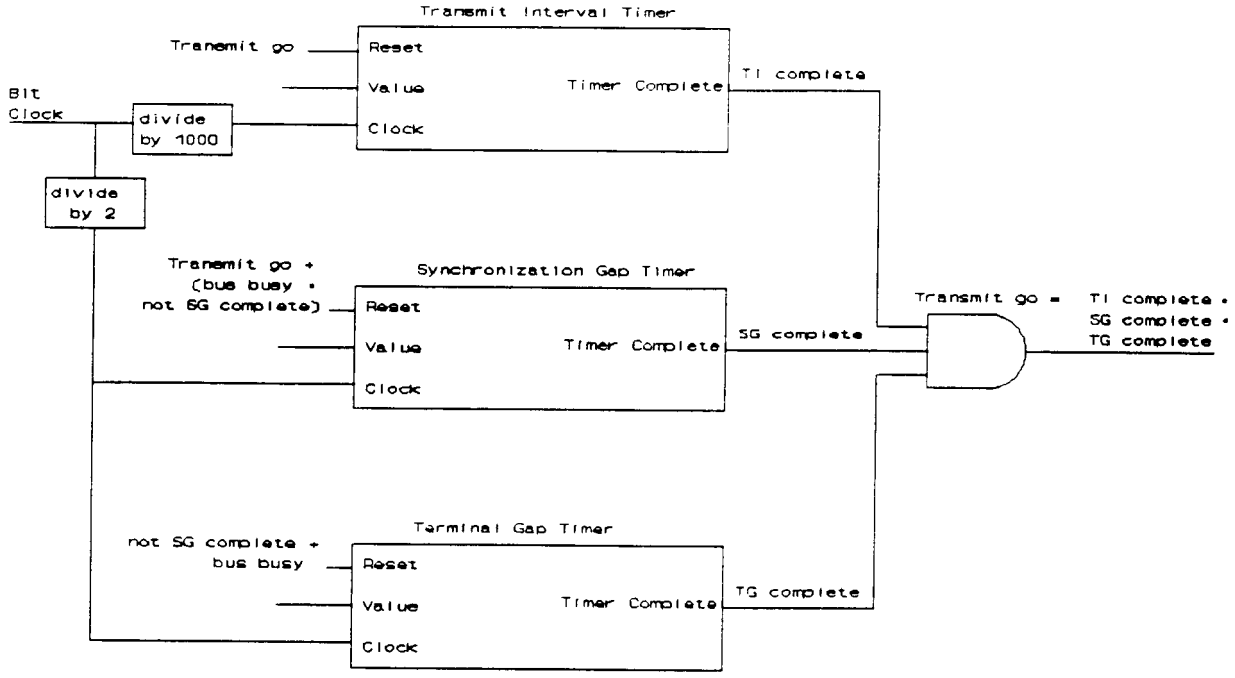


Figure G–9: Terminal Gap Timer

On a bus that is not fully loaded, each terminal's SG and TG timers usually have counted to completion before its TI timer completes. Thus each terminal transmits exactly once every TI interval. (See Figure G–10.) In the real world, one terminal will have the slowest clock and all the other terminals will back up behind it, each terminal's TG time separating it from the earlier terminal. (See Figure G–11.)

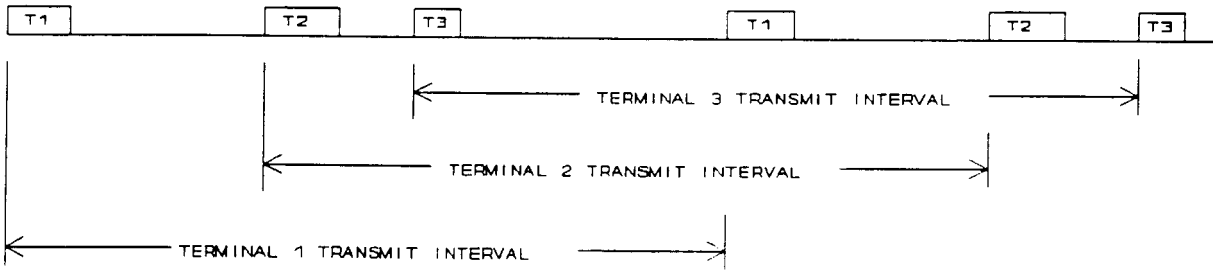


Figure G–10: Power-on Periodic Mode

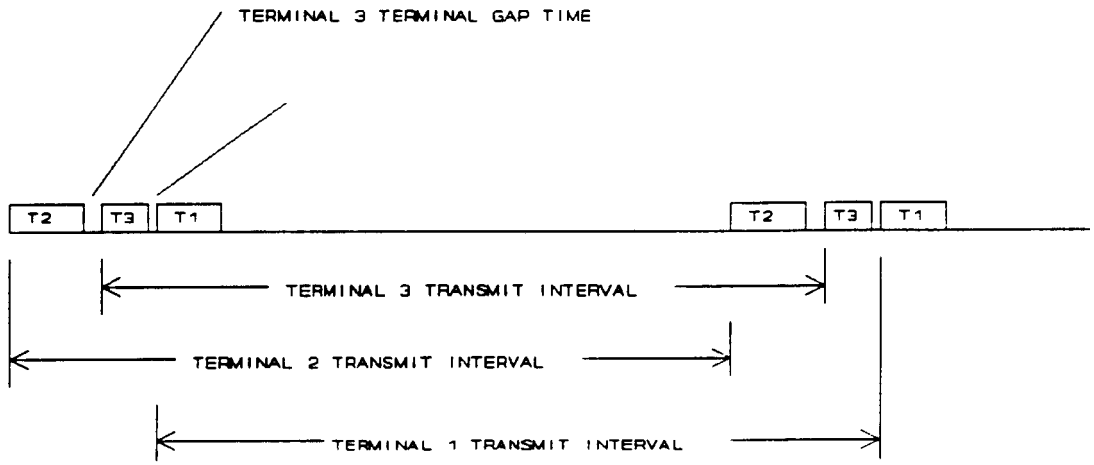


Figure G–11: Periodic Mode

On a bus that is overloaded (more data is transmitted by the terminals than will fit in a TI), the terminals will transmit in order from the shortest TG to the longest TG followed by a SG time before they repeat. Each terminal will transmit less than once per TI but each will get a chance to transmit. (See Figure G–12.)

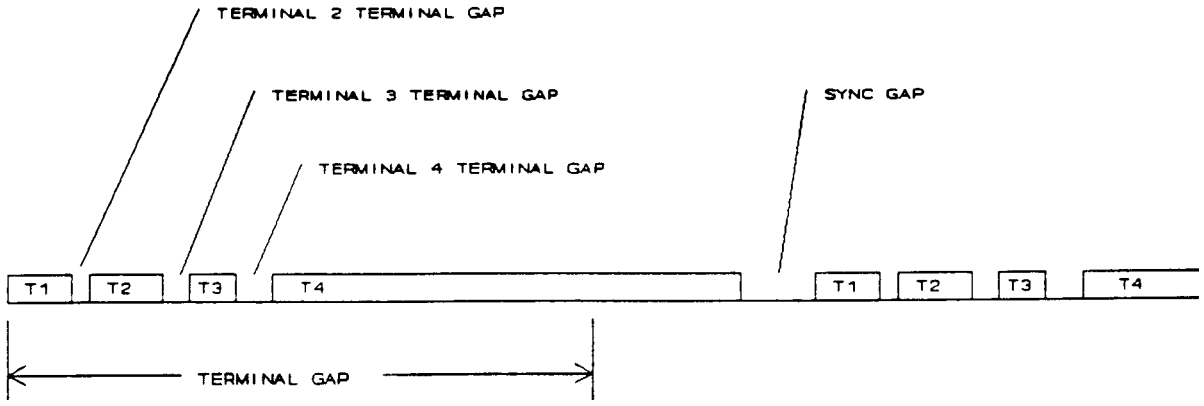


Figure G–12: Aperiodic Mode

Terminal Transmit Programming

The data a terminal transmits is controlled by programming a schedule in the transmit portion of the personality ROM (Xpp). The Xpp is organized as a 31 × 31 array of cells. Each cell describes a wordstring to transmit. There is a 32nd row of cells in the array for control.

A transmit cell contains the following information:

- a 12-bit label to transmit
- a 16-bit address of where the data to transmit with the label is stored in the memory it shares with the device
- an 8-bit number of words to transmit (word count)
- a 15-bit number that may be output to the device when this cell is transmitted (an “interrupt vector”)
- a bit to enable strobing the interrupt vector at the beginning of transmitting the wordstring
- a bit to enable strobing the interrupt vector at the end of transmitting the wordstring
- a bit indicating this is the last wordstring of a message (end of message bit)
- a bit to enable transmitting data with this label
- a bit to control using the first data word as word count rather than the Xpp word count (variable wordstring length)
- a sync bit used to enable switching to alternate mode

The terminal controller will interpret the schedule in Block or Independent mode via an input pin.

In Block mode, the terminal controller will transmit as a message the wordstrings defined in a single row of cells until it transmits the wordstring whose cell has the end of message bit true. The terminal controller uses a row pointer to keep track of the row to transmit. After transmitting, the row pointer is compared to a “y modulo” value in the first control cell. If the row pointer is less than the y modulo, the row pointer is incremented, otherwise the row pointer is reset. Thus, in Block mode, the terminal controller can repeatedly transmit up to 31 messages of up to 31 wordstrings.

Alternate mode is a way of switching to a different transmit schedule. It may be used only in block mode. The top rows of the Xpp are programmed with one schedule and the bottom rows are programmed with another schedule. The device switches between the two schedules via an input pin to the terminal controller.

Independent mode allows a much more complicated transmit schedule. In independent mode each column of cells in the Xpp has its own row pointer and y

modulo. The terminal controller transmits as a message the wordstrings defined by the cells pointed to by the row pointers in each row until it transmits a wordstring whose cell has the end of message bit set. The row pointer for each column is compared to its y modulo value and the row pointer is incremented or reset.

Independent mode is particular useful for scheduling different update rates for different types of data. Data that should be updated every TI would be defined in a column by itself. Data that should be updated every other TI would be defined in a column with one other cell.

Terminal Receive Programming

The data a terminal saves in memory is controlled by programming the receive portion of the personality ROM, the Rpp and Mpp. The Rpp is an I × 4092 array of cells, one for each of the possible label numbers. Each cell contains the following information:

a 16-bit base address of where the data received is to be stored in the memory it shares with the device.

an 8-bit number of words to receive (word count)

a 15-bit number that may be output to the device when this label is received (an “interrupt vector”)

a 7-bit number that selects a table of address offset values in the Mpp (offset table pointer)

a bit to enable strobing the interrupt vector at the beginning of receiving this wordstring

a bit to enable strobing the interrupt vector at the end of receiving the wordstring

a bit to enable receiving data with this label

a bit to control using the first data word as word count rather than the Rpp word count (variable wordstring length)

The address in memory where data is to be stored is the sum of the base address in the Rpp cell plus an address offset value from the Mpp portion of the personality ROM. The Mpp contains 128 tables of offset values. The table for a particular cell is the offset table pointer. The tables are a 16 × 16 array of 16-bit values. The value the terminal controller uses is indexed with the channel ID of the receiving terminal controller and the channel ID of the transmitting terminal controller.

Appendix H: Options

Option 01 Options 01 adds one additional channel.

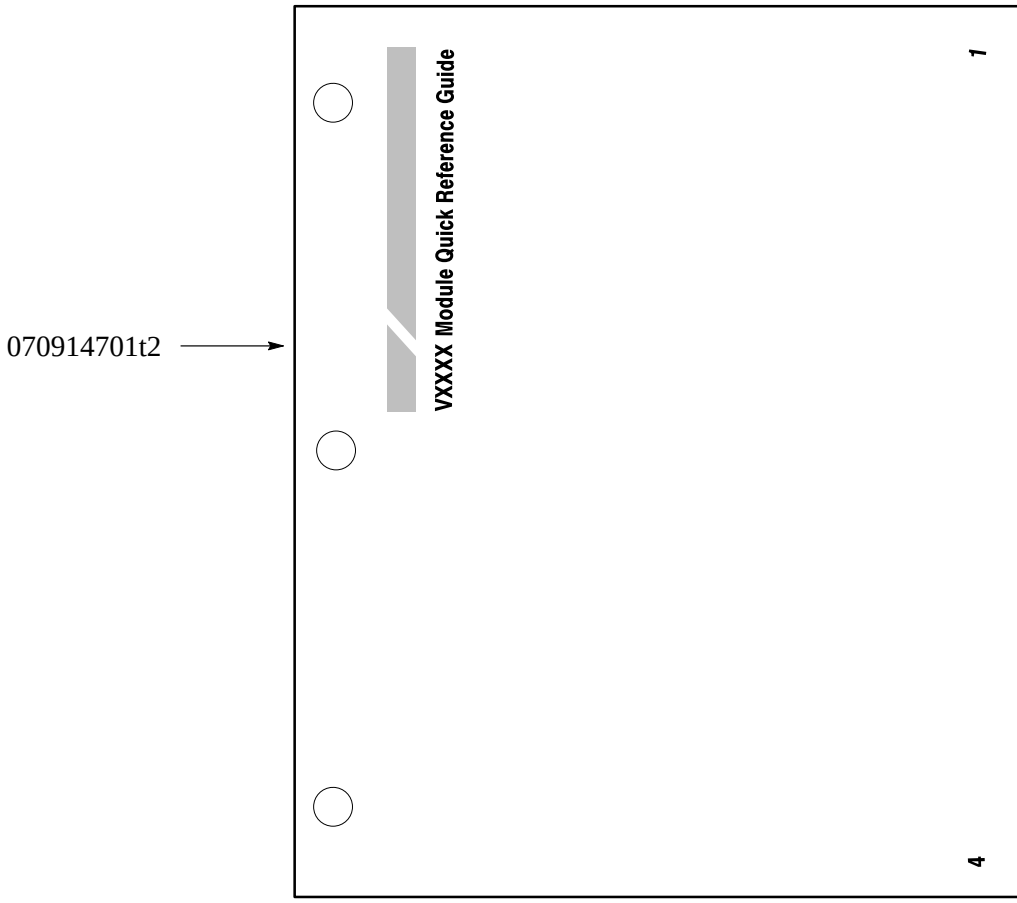
Option 02 Option 02 adds two additional channels.

Appendix I: VX4469A Module Quick Reference Guide

The following page is a quick reference for the VX4469A ARINC 629 Communication Module. You may wish to make a copy of it to keep by the instrument.

REPLACE THIS PAGE

with file 070914701t2 as shown below.



Appendix J: Binary Transfer

If you are using a National Instruments GPIB-VXI/C Slot 0 module and are planning on using the binary transfer capabilities of the modules above, you will need to load a CI (Code Instrument) into the GPIB-VXI/C Slot 0.

NOTE. *The GPIB-VXI/C Slot0 has an internal buffer that holds the data to be read out. The buffer will automatically take a reading from the module upon a GPIB read. The buffer will read the module until it receives an END BIT (bit 8 set in the response to a byte request command). The Tektronix products above do not set bit 8 on readback, thus the GPIB-VXI/C Slot0 will fill its buffer with data (approximately 450 Kbytes). If you only request 1 Kbytes of data over GPIB there still will be 449 Kbytes of data in the buffer. This data will remain in the buffer until read out. If you should request data from another module the data that you will receive back will be from the data that is left over in the buffer (449 Kbytes).*

National Instruments has developed a code instrument that will read the exact number of bytes that was requested over the GPIB bus from the module. The code instrument will not read more data then requested and will have no leftover data in the buffer. Refer to the National Instruments GPIB-VXI/C manual for information on code instruments.

If you need any assistance call 1-800-TEK-WIDE or contact your local Tektronix representative.

Appendix K: User Service

This appendix contains service-related information that covers the following topics:

- Preventive maintenance
- User-replaceable Parts

Preventive Maintenance

You should perform inspection and cleaning as preventive maintenance. Preventive maintenance, when done regularly, may prevent malfunction and enhance reliability. inspect and clean the module as often as conditions require by following these steps:

1. Turn off power and remove the module from the VXIbus mainframe.
2. Remove loose dust on the outside of the instrument with a lint-free cloth.
3. Remove any remaining dirt with lint-free cloth dampened in a general purpose detergent-and-water solution. Do not use abrasive cleaners.

User-Replaceable Parts

Replacement parts are available through your local Tektronix field office or representative.

Changes to Tektronix instruments are sometimes made to accommodate improved components as they become available. Therefore, when ordering parts, it is important to include the following information in your order.

- Part number
- Instrument type or model number
- Instrument serial number
- Instrument modification number, if applicable.

Table K–1: User-Replaceable Parts

Part Description	Part Number
User Manual	070-9147-XX
Label, Tek CDS	950-0942-00
Label, VXI	950-1022-00
Fuse, Micro 1 Amp 125 V Fast	159-0116-00
Fuse, Micro 5 Amp 125 V Fast	159-0207-00
Collar Screw, Metric 2.5 × 11 Slotted	950-0952-00
Shield, Front	950-1572-00
Screw, Phillips Metric 2.5 × 4 FLHD SS	211-0867-00

Glossary

Glossary

The terms in this glossary are defined as used in the VXIbus System. Although some of these terms may have different meanings in other systems, it is important to use these definitions in VXIbus applications. Terms which apply only to a particular instrument module are noted. Not all terms appear in every manual.

The standard VX4469A Module includes one ARINC 629 terminal. As options, one or two additional channels may be added. In the following definitions, the phrase “terminal specific” means that the description applies to each terminal or that there is one for each terminal. The phrase “module specific” implies that the description applies to the module or that there is one per module.

Accessed Indicator
An amber LED indicator that lights when the module identity is selected by the Resource Manager module, and flashes during any I/O operation for the module.

ACFAIL*
A VMEbus backplane line that is asserted under these conditions: 1) by the mainframe Power Supply when a power failure has occurred (either ac line source or power supply malfunction), or 2) by the front panel ON/STAND-BY switch when switched to STANDBY.

Alternate Mode
If a terminal is in Block mode it may be switched on command between Alternate and Normal modes. This allows a terminal to switch between two different transmit schedules without having to stop and reprogram the Personality PROMs. The transmit schedule for Alternate mode is programmed into the lower portion of the Transmit Personality PROM. Terminal specific.

ARINC 629
Specification defining a standard for digital communication between avionics system elements. It is available from Aeronautical Radio, Inc., 2551 Riva Road, Annapolis, MD 21401, (301) 266–4110. The VX4469A is designed for use with equipment following this specification.

Backplane
The printed circuit board that is mounted in a VXIbus mainframe to provide the interface between VXIbus modules and between those modules and the external system.

Block Mode
Block mode is one of two ways the terminal IC can interpret the schedule in the Transmit Personality PROM. The Transmit Personality PROM is

organized in rows and columns of cells. In Block mode, during each TI (Transmit Interval – see definition below), terminal IC will transmit the labels from the cells in a single row. Each TI, a different row is transmitted. After the number of rows indicated by the y modulo is transmitted, the terminal IC will transmit beginning with the first row again. The terminal IC can be programmed to run in Block mode or Independent mode. Terminal specific.

Bus Arbitration

In the VMEbus interface, a system for resolving contention for service among VMEbus Master devices on the VMEbus.

Circular Buffer

Circular buffers are variable-sized data buffers in the 80186's PROM space. They are not specific to any terminal. Circular buffers are used to buffer data between the system controller and a terminal's shared memory because the VX4469A can move data very quickly between circular buffers and shared memory.

CLK10

A 10 MHz, ±100 ppm, individually buffered (to each module slot), differential ECL system clock that is sourced from Slot 0 and distributed to Slots 1–12 on P2. It is distributed to each module slot as a single source, single destination signal with a matched delay of under 8 ns.

CLK100

A 100 MHz, ±100 ppm, individually buffered (to each module slot), differential ECL system clock that is sourced from Slot 0 and distributed to Slots 1–12 on P3. It is distributed to each module slot in synchronous with CLK10 as a single source, single destination signal with a maximum system timing skew of 2 ns, and a maximum total delay of 8 ns.

Commander

In the VXIbus interface, a device that controls another device (a servant). A commander may be a servant of another commander.

Communication Registers

In word serial protocol, a set of device registers that are accessible to the commander of the device. Such registers are used for inter-device communications, and are required on all VXIbus message-based devices.

Configuration Registers

A set of registers that allow the system to identify a (module) device type, model, manufacturer, address space, and memory requirements. In order to support automatic system and memory configuration, the VXIbus standard specifies that all VXIbus devices have a set of such registers, all accessible from P1 on the VMEbus.

CRC
Cyclic redundancy check. A method of error detection.

C-Size Card
A VXIbus instrument module that is 340.0 by 233.4 mm by 30.48 mm (13.4 by 9.2 in by 1.2 in).

DC Supplies Indicator
A red LED indicator that illuminates when a DC power fault is detected on the backplane.

DUT
Device Under Test. Also UUT – Unit Under Test.

ECLTRG
Six single-ended ECL trigger lines (two on P2 and four on P3) that function as inter-module timing resources, and that are bussed across the VXIbus subsystem backplane. Any module, including the Slot 0 module, may drive and receive information from these lines. These lines have an impedance of 50 ohms; the asserted state is logical High.

External System Controller
The host computer or other external controller that exerts overall control over VXIbus operations.

FAILED Indicator
A red LED indicator that lights when a device on the VXIbus has detected an internal fault. This might result in the assertion of the SYSFAIL* line.

FIFO
(First In – First Out) A hardware device that stores data. Data strobed into it is available to be strobed out again on a first in, first out basis. The VX4469A uses these devices to temporarily store vector interrupt and timestamp data. Terminal specific.

Independent Mode
Independent mode is one of two ways the terminal IC can interpret the schedule in the Transmit Personality PROM. The Transmit Personality PROM is organized in rows and columns of cells. In Independent mode a terminal IC will transmit a cell from each column each TI. Each column has a y modulo that indicates when the beginning of that column should be started again.

Instrument Module
A plug-in printed circuit board, with associated components and shields, that may be installed in a VXIbus mainframe. An instrument module may contain more than one device. Also, one device may require more than one instrument module.

Interrupter
A device capable of asserting VMEbus interrupts and performing the interrupt acknowledge sequence.

Interrupt Handler
A functional module that detects interrupt requests generated by Interrupters and responds to those requests by requesting status and identity information.

IRQ
The Interrupt ReQuest signal, which is the VMEbus interrupt line that is asserted by an Interrupter to signify to the controller that a device on the bus requires service by the controller.

Local Bus
A daisy-chained bus that connects adjacent VXIbus slots.

Logical Address
The smallest functional unit recognized by a VXIbus system. It is often used to identify a particular module.

Mainframe
Card Cage. For example, the Tektronix VX1400 Mainframe, an operable housing that includes 13 C-size VXIbus instrument module slots.

Message Based Device
A VXIbus device that supports VXI configuration and communication registers. Such devices support the word serial protocol, and possibly other message-based protocols.

MODID Lines
Module/system identity lines.

MPP
Multiple Personality PROM.

Multiple Personality PROM
The portion of the Personality PROM used in conjunction with the Receive Personality PROM for determining the actual location in shared memory that received data will be stored. Terminal specific.

Personality PROMS
PROMS that are used to program the terminal IC. The VX4469A uses a single static RAM instead of ROM to allow easy programming. Terminal specific.

Personality RAM
Static RAM used by the VX4469A in place of personality PROMS to program the terminal IC. Terminal specific.

Power Monitor
A device that monitors backplane power and reports fault conditions.

Pseudo Bus Module

A Pseudo Bus Module is not defined in the ARINC 629 Specification. It is used to replace a SIM and current coupler. It fits into the socket normally used by a SIM and allows direct 2-wire connection to other terminal ICs that also have Pseudo Bus Modules. In other words, it is a cheap substitute for SIMs and current couplers that may be used for testing or development until SIMs and current couplers are available, or even later to avoid the cost of purchasing SIMs and current couplers. Terminal specific.

P1

The top-most backplane connector for a given module slot in a vertical mainframe such as the Tektronix VX1400. The left-most backplane connector for a given slot in a horizontal mainframe.

P2

The bottom backplane connector for a given module slot in a vertical C-size mainframe such as the VX1400; or the middle backplane connector for a given module slot in a vertical D-size mainframe such as the VX1500.

Query

A form of command that requires a response.

READY Indicator

A green LED indicator that lights when the power-on diagnostic routines have been completed successfully. An internal failure or failure of +5 volt power will extinguish this indicator.

Receive Personality PROM

The portion of the Personality PROM used to program the terminal IC for receiving data and monitoring transmitted data. Terminal specific.

Register Based Device

A VXIbus device that supports VXI register maps, but not high level VXIbus communication protocols; includes devices that are register-based servant elements.

Resource Manager

A VXIbus device that provides configuration management services such as address map configuration, determining system hierarchy, allocating shared system resources, performing system self test diagnostics, and initializing system commanders.

RPP

Receive Personality PROM.

Self Calibration

A routine that verifies the basic calibration of the instrument module circuits, and adjusts this calibration to compensate for short- and long-term variables.

Self Test
A set of routines that determine if the instrument module circuits will perform according to a given set of standards. A self test routine is performed upon power-on.

Serial Interface Module
Serial Interface Module (SIM). This is a hybrid integrated circuit that interfaces between a terminal IC and a current coupler. Terminal specific.

Servant
A VXIbus message-based device that is controlled by a commander.

SG
See Sync Gap.

Shared Memory
Each terminal shares a unique portion of the module's system memory with the module's controller, an 80186 processor. This memory is where the terminal IC reads and writes data to be transmitted and received on the ARINC 629 bus. Terminal specific.

SIM
See Serial Interface Module.

Slot 0 Controller
See Slot 0 Module. Also see Resource Manager.

Slot 0 Module
A VXIbus device that provides the minimum VXIbus slot 0 services to slots 1 through 12 (CLK10 and the module identity lines), but that may provide other services such as CLK100, SYNC100, STARBUS, and trigger control.

Sync Gap
Sync Gap (SG) is one of three programmable timers within the terminal IC. When the terminals on a bus are operating in Aperiodic mode it insures that each terminal transmits at an equal rate. It is equal for all devices on a bus and greater than the terminal gap of any device on the bus. Terminal specific.

SYSFAIL*
A signal line on the VMEbus that is used to indicate a failure by a device. The device that fails asserts this line.

System Controller
The system controller is a device outside of the VX4469A that communicates with the VX4469A to control it and retrieve or provide data.

System Hierarchy
The tree structure of the commander/servant relationships of all devices in the system at a given time. In the VXIbus structure, each servant has a commander. A commander may also have a commander.

Terminal Gap
Terminal Gap (TG) is one of three programmable timers within the terminal IC. It is used to prevent transmit conflicts between terminals. It is different for every terminal on a bus.

Terminal IC
The terminal IC is a large integrated circuit (ASIC) developed to function as a terminal controller on an ARINC 629 bus. It interfaces with a Personality PROM, shared memory, and a SIM or Pseudo Bus Module. Terminal specific.

Test Program
A program, executed on the system controller, that controls the execution of tests within the test system.

Test System
A collection of hardware and software modules that operate in concert to test a target DUT.

TG
See Terminal Gap.

TI
See Transmit Interval.

Transmit Interval
Transmit Interval (TI) is one of three programmable timers within the terminal IC. It is the same for all terminals on a bus. On a bus that is not fully loaded, each terminal transmits once during each transmit interval.

Transmit Personality PROM
The portion of the Personality PROM used to program the terminal IC for transmitting data. Terminal specific.

TTLTRG
Open collector TTL lines used for inter-module timing and communication.

Vector Interrupt
The terminal IC may be programmed, using the Personality PROMs, to output a 15-bit number near the beginning or end of received or transmitted data. On a VX4469A Module, 13 bits of this number are strobed into the terminal FIFO along with error and timestamp information. This number may be used to identify data, to cause the VX4469A to manipulate data, and to cause an interrupt to the system controller. Terminal specific.

VXIbus Subsystem
One mainframe with modules installed. The installed modules include one module that performs slot 0 functions and a given complement of instrument modules. The subsystem may also include a Resource Manager.

Word Serial Communications
Inter-device communications using the Word Serial Protocol.

Word Serial Protocol
A VXIbus word oriented, bi-directional, serial protocol for communications between message-based devices (that is, devices that include communication registers in addition to configuration registers).

WSP
See Word Serial Protocol.

XPP
Xmit Personality PROM.

y modulo
The transmit scheduler uses a y modulo value to know how far down columns to go. When it transmits a row number that is the same as the y modulo value, the pointer for that column is set to 0.

10-MHz Clock
A 10 MHz, ±100 ppm timing reference. Also see CLK10.

100-MHz Clock
A 100 MHz, ±100 ppm clock synchronized with CLK10. Also see CLK100.

488-To-VXIbus Interface
A message based device that provides for communication between the IEEE-488 bus and VXIbus instrument modules.

VX4469A Module Quick Reference Guide

Numbers in parentheses refer to the page(s) in the Operating Manual.

SETUP

Be sure all switches are correctly set. (See page 1–6.)
Follow Installation guidelines. (See page 1–11.)

The default condition of the VX4469A Module after the completion of power-up self test is described in the *SYSFAIL, Self Test, and Initialization* section, p. 2–1.

LEDs

When lighted, the LEDs indicate the following:

Power	power supplies functioning
Failed	module failure
Error	an error has been found in self test or programming
MSG	module is processing a VMEbus cycle
BACKGROUND	switches on and off at a rate inversely proportional to processor loading.

Each terminal has the following LEDs on the front edge of the module:

ENABLE	the terminal is enabled.
BUS BUSY	the terminal detects activity on the ARINC 629 bus.
STRING ACTIVE	the terminal IC is receiving or transmitting.
RECEIVE ERROR	the terminal IC receive error flag is/was true.
TRANSMIT ERROR	the terminal IC transmit error flag is/was true.
TRANSMIT DISABLE	the terminal IC has disabled the SIM for transmitting.

SYSTEM COMMANDS

These non-data commands are initiated by the VX4469A's commander. The following VXIbus Instrument Protocol commands will affect the VX469A:

ABORT NORMAL OPERATION	END NORMAL OPERATION
ASYNCHRONOUS MODE CONTROL	BYTE REQUEST
BEGIN NORMAL OPERATION	CONTROL RESPONSE
BYTE AVAILABLE	READ PROTOCOL
CLEAR	READ STATUS
CONTROL EVENT	TRIGGER
SET LOCK	CLEAR LOCK
READ INTERRUPTERS	READ INTERRUPT LINE
READ PROTOCOL ERROR	

COMMAND SYNTAX

Command protocol and syntax for the VX4469A Module is as follows: (3–1)

1. Each command is terminated by a semicolon or a linefeed.
2. White space characters (including space, tab, and carriage return) are ignored.
3. Characters may be sent as either upper or lower case.
4. Non-printing characters are indicated by the following:
   <cr> carriage return. <lf> line feed.
   <tm> terminator, either a line feed or semicolon.

5. Comments added to commands will be ignored by the VX4469A. Begin the comment with an ! and end the command with a terminator. The ! must not be in the middle of a command, but may be placed after a line feed or semi-colon.
6. Brackets [ ] are used to show optional parts of commands.

   Parts of commands enclosed in parentheses ( ) contain two or more choices, one of which must be used.

   Lower case letters are used to represent numeric values.

MODULE COMMANDS

All commands must end with a terminator <tm>, which may be a line feed <LF> or semi-colon. White space characters are ignored.

BR	binary read: BRC circular buffer (3–12) BRD data (3–13) BRR receive PP (3–17) BRG registers (3–14) BRM multiple PP (3–16) BRT Test RAM (3–18) BRX transmit PP (3–19)
BW	binary write: BWC circular data (3–20) BWM multiple PP (3–22) BWX transmit PP (3–25) BWD data (3–21) BWR receive PP (3–23)
CC	clear circular buffer data. (3–26)
CCA	clear all circular buffers of data. (3–27)
CF	clear a specified terminal's hardware FIFO. (3–28)
DC	define circular buffer. (3–29)
F	fill the Personality PROM for a specified terminal with all ones (FF hex), or fill shared memory with zeros. (3–30)
GRD	supply data to the 16-bit register on the VXI backplane. (3–31)
GWD	accept data via the 16-bit register on the VXI backplane. (3–32)
HN	switch this terminal's memory addressing to normal at the beginning of the next wordstring transmitted or received by this terminal. (3–33)
HPN	switch this terminal's memory addressing to normal at the beginning of the next wordstring transmitted by this terminal whose transmit cell in the XPP has the switch bit true. (3–34)
HPS	switch this terminal's memory addressing to switched at the beginning of the next wordstring transmitted by this terminal whose transmit cell in the XPP has the switch bit true. (3–35)
HR	reset this terminal's memory addressing to normal immediately. (3–36)
HS	switch this terminal's memory addressing to switched at the beginning of the next wordstring transmitted or received by this terminal. (3–37)
IA	initialize module to power-up, except power-up ROM. (3–38)
IC	initialize circular buffer definitions. (3–39)
IM	initialize and monitor. (3–40)
IN	initialize and monitor all terminals. (3–41)
IVI	initialize vector instructions. (3–42)
IVX	initialize vector index table. (3–43)
LC	list circular buffer status. (3–44)
LCB	list circular buffer status in binary format. (3–45)
LE	set up the VX4469A to return any error messages in its error queue. (3–46)

LG	set up the VX4469A to return the current value of the high 4 bits of terminals currently in Test Mode. (3–47)
LH	list memory switch status. (3–48)
LR	list revision. (3–49)
LS	return information about VX4469A setup. (3–50)
LVI	return the instructions in a particular instruction block. (3–53)
LVX	return vector index information. (3–54)
NRD	front panel data port read data. (3–55)
NWD	front panel data port write data. (3–56)
RC	read circular buffer. (3–57)
RCC	read circular buffer, calculating and appending a CRC, using the circular buffer cell size as the 'number of data words' in the wordstring. (3–58)
RCV	read circular buffer variable, calculating and appending a CRC, using the first data word in the circular buffer cell or the cell size, whichever is less, as the 'number of data words' in the wordstring. (3–59)
RD	read data from terminal's shared memory. (3–60)
RDC	read data from shared memory, calculating and appending a CRC. (3–61)
RDV	read data variable from shared memory, calculate and append a CRC, using the word at addr in shared memory as the number of words in the wordstring. (3–62)
RG	read terminal IC's status registers. (3–63)
RI	read interrupt status. (3–65)
RM	read multiple personality PROM. (3–68)
RR	read receive personality PROM. (3–69)
RS	read Serial Interface Module status. (3–70)
RX	read Transmit Personality PROM. (3–72)
SBD	disable the terminal IC from accessing the shared memory. (3–73)
SBE	enable the terminal IC to access the shared memory. (3–74)
SC	set the terminal ID for transmit data and receive data channeling. (3–75)
SCL	set the Channel ID to use or not use the XPP label field for the channel ID. (3–76)
SD	disable the specified terminal(s). (3–77)
SE	enable the specified terminal(s). (3–78)
SF	set error message format to brief or normal. (3–79)
SG	writes a Test RAM segment number to a terminal's segment register. (3–80)
SH	set memory switch mode. (3–81)
SI	set the ARINC 629 parameters TI, TG, and SG. (3–82)
SKD	disables VXI TTL trigger, External trigger or label enables on a terminal in test mode 2. (3–83)
SKI	causes a terminal in test mode 2 to transmit immediately. (3–84)
SKL	causes a terminal in test mode 2 to transmit after it receives a particular label. (3–85)
SKV	causes a terminal in test mode 2 to transmit after it receives a VXI TTL trigger. (3–86)
SKX	causes a terminal in test mode 2 to transmit after it receives an external trigger. (3–87)
SMA	set the alternate mode pin on the terminal IC true. (3–89)
SMB	set the protocol transmit mode to block. (3–90)

SMI	set the protocol transmit mode to independent. (3–91)
SMN	set the alternate mode pin on the terminal IC false. (3–92)
SO	set overload timer value. (3–93)
SQVE	enable/disable setting the terminal VXI trigger on a communication error. (3–95)
SQVG	enable/disable setting the terminal VXI trigger on beginning to transmit. (3–96)
SQVI	enable/disable setting the terminal VXI trigger on interrupt vector bit 13. (3–97)
SQVR	enable/disable setting the terminal VXI trigger on interrupt vector bit 13 when this terminal has just received a wordstring that does not have a valid CRC. (3–98)
SQXE	enable/disable setting the terminal VXI trigger on a communication error. (3–99)
SQXG	enable/disable setting the terminal VXI trigger on beginning to transmit. (3–100)
SQXI	enable/disable setting the terminal VXI trigger on interrupt vector bit 13. (3–101)
SQXR	enable/disable setting the terminal VXI trigger on interrupt vector bit 13 when this terminal has just received a wordstring that does not have a valid CRC. (3–102)
SR	set the radix of numeric data or command parameters. (3–103)
SSE	set system error interrupt. (3–104)
SST	set receive threshold. (3–105)
SSV	set system vector interrupt. (3–106)
ST	set the time-stamp clock period. (3–107)
SVI	set up a list of commands to be executed whenever a particular vector or vectors is/are generated. (3–108)
SVX	set which instruction block is to be used by each vector. (3–109)
SW	set the terminal to timestamp the end of a wordstring. (3–110)
TQSV	test the VX4469A's software VXI TTL trigger. (3–111)
TQSX	test the VX4469A's software external trigger. (3–112)
TQV	test a terminal's VXI TTL trigger. (3–113)
TQX	test a terminal's external trigger. (3–114)
TS	test SIM. (3–115)
URM	a user friendly way of reading data from the multiple personality PROM. (3–116)
URR	a user friendly way of reading data from the receive personality PROM. (3–118)
URX	a user friendly way of reading data from the Transmit Personality PROM. (3–121)
UWM	a user friendly way of writing data to the multiple personality PROM. (3–124)
UWR	a user friendly way of writing data to the receive personality PROM. (3–125)
UWX	a user friendly way of writing data to the Transmit Personality PROM. (3–128)
WC	write data to a cell in a circular buffer. (3–130)
WCC	write data to a circular buffer, calculating and appending a CRC. (3–131)
WD	write data to a terminals shared memory. (3–132)
WDC	write Data, calculating and appending a CRC, to shared memory starting at 'addr', including 'label' in the CRC calculation. (3–133)
WM	write multiple personality PROM. (3–134)
WR	write receive personality PROM. (3–135)
WX	write Transmit Personality PROM. (3–137)