Programming Guide for Giga-tronics ASCOR 3000 Series Switching Modules

All technical data and specifications in this publication are subject to change without prior notice and do not represent a commitment on the part of Giga-tronics, Incorporated.

Warranty

Giga-tronics Series 3000 Switching Modules are warranted against defective materials and workmanship for three years from date of shipment, or as detailed in the warranty section of this manual. Giga-tronics will, at its option, repair or replace products that are proven defective during the warranty period. This warranty DOES NOT cover damage resulting from improper use, nor workmanship other than Giga-tronics service. There is no implied warranty of fitness for a particular purpose, nor is Giga-tronics liable for any consequential damages. Specification and price change privileges are reserved by Giga-tronics.

CONTACT INFORMATION

Giga-tronics, Incorporated
4650 Norris Canyon Road
San Ramon, California 94583
Telephone: 800.726.4442 (only within the United States)
925.328.4650
Fax: 925.328.4700
On the Internet: www.gigatronics.com

Regulatory compliance information

This product complies with the essential requirements of the following applicable European Directives, and carries the CE mark accordingly.

89/336/EEC and 73/23/EEC	EMC Directive and Low Voltage Directive
EN61010-1 (1993)	Electrical Safety
EN61326-1 (1997)	EMC – Emissions and Immunity
Manufacturer's Name:	**Manufacturer's Address**
Giga-tronics, Incorporated	4650 Norris Canyon Road San Ramon, California 94583 U.S.A.
Type of Equipment:	**Model Series Number**
Switching Module	ALL

Declaration of Conformity on file. Contact Giga-tronics at the following;

Giga-tronics, Incorporated
4650 Norris Canyon Road
San Ramon, California 94583
Telephone: 800.726.4442 (only within the United States)
925.328.4650
Fax: 925.328.4700

Record of Changes to This Manual

Use the table below to maintain a permanent record of changes to this document. Corrected replacement pages are issued as Technical Publication Change Instructions (TPCI). When you are issued a TPCI, do the following:

1. Insert the TPCI at the front of the manual binder.
2. Remove the pages from the manual binder that are noted in the TPCI.
3. Replace the page(s) removed in the previous step with the corrected page(s).
4. Record the changes in the table below.

TPCI Number	TPCI Issue Date	Date Entered	Comments

Revision History			
Revision	**Description of Change**	**Chg Order #**	**Approved By**
A	Initial Release 4/97		
B	Updated 11/98		
C	Updated 8/00		
D	Reformatted 3/12		RCW

Contents

Programming

1.1 VXI Register Based Modules

The ASCOR Switching Module General Purpose Switch VXI Module is a register based device and supports VXIbus register maps. All controls to the Switching Module is done through registers. All registers can be accessed with the use of slot 0 computers, host computers with VXI-MXI, or host computers with GPIB and GPIB-VXI slot 0 controllers. The Switching Module is not a message based device and does not support VXIbus communication protocols.

1.2 ASCOR VXI Module Type

The Switching Module has operational registers in the A16 and A24 address spaces. The registers located in the A16 address space are VXI Device Registers. They are accessed as a 16-bit word. The registers located in the A24 address space are ASCOR Module Custom registers, and they can be accessed as a 16-bit word. Additionally, since the Switching Module is equipped with VXIMAXTM 16/32, these Custom registers can also be accessed as a 32-bit word.

1.3 VXI Device Registers

VXI Device Registers are located in the A16 address space, which can be accessed by A16 address mode. VXI Device Registers are separated into Configuration, Device Class Dependent, and Device Dependent registers. The Switching Module provides all 4 Configuration Registers and one Device Dependent register. The rest of the A16 register space is not populated. Programming examples are shown in Sections 4.1 - 4.3 using several popular interface libraries.

1.4 ASCOR Module Custom Registers

The Switching Module Custom registers are located in the A24 address space and they can only be accessed with A24 address mode. The method of accessing these Registers in the A24 address space is different from accessing the VXI Device Registers in the A16 address space. Therefore, care must be taken whenever accessing registers that are located in different address spaces. Programming examples are shown in Section 4.4 using several popular interface libraries.

1.5 Static and Dynamic Configurations

The Switching Module supports both Static Configuration and Dynamic Configuration of Logical Address. In Static Configuration mode the Logical Address of the module is set and cannot be changed by the resource manager. In Dynamic Configuration mode the Logical Address is determined by the resource manager based on other devices in the system. Procedures for changing the Logical Addresses for the Switching Module are discussed in ***(Page 22) Section 5.3: How to change the Switching Module Module's Logical Address.***

Chapter 2 VXI Device Register Description

The ASCOR Switching Module General Purpose Switch has VXI Device Registers located in the A16 address space. The VXI Specification defines 32 VXI Device Registers and they are all 16 bits wide. The first 4 registers are VXI Configuration Registers. The next 12 registers are VXI Device Class Dependent Registers. The last 16 registers are VXI Device Dependent Registers.

The Switching Module supports 5 of the 32 VXI Device Registers, four in VXI Configuration Registers and one in VXI Device Dependent Register. All other registers are not supported.

VXI Device Registers forASCOR Switching Module **VXI Device Dependent Registers**	
3Eh	ASCOR Control Register
3Ch	Register Not Used
3Ah	Register Not Used
38h	Register Not Used
36h	Register Not Used
34h	Register Not Used
32h	Register Not Used
30h	Register Not Used
2Eh	Register Not Used
2Ch	Register Not Used
2Ah	Register Not Used
28h	Register Not Used
26h	Register Not Used
24h	Register Not Used
22h	Register Not Used
20h	Register Not Used

VXI Device Class Dependent Registers	
1Eh	Register Not Used
1Ch	Register Not Used
1Ah	Register Not Used
18h	Register Not Used
16h	Register Not Used
14h	Register Not Used
12h	Register Not Used
10h	Register Not Used
0Eh	Register Not Used
0Ch	Register Not Used
0Ah	Register Not Used
08h	Register Not Used

VXI Configuration Registers	
06h	Offset Register
04h	Status / Control Register
02h	Device Type Register
00h	ID / Logical Address Register

2.1 VXI Configuration Registers

The first four registers in the A16 address space are Configuration Registers. Each register is 16-bits wide. They are explained below.

Offset	Description
00h	**ID Register (read) / Logical Address Register (write)** A read of this 16-bit register provides information about the Switching Module Module's configuration.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Device Class		Address Space		Manufacturer ID											

(Bits 15-14) Device Class: This field indicates the classification of the VXIbus device.

00b = Memory

01b = Extended

10b = Message Based

xxx = Register Based (ASCOR VXI Module)

(Bits 13-12) Address Space: This field indicates the addressing mode(s) of the device's operational registers.

00b = A16/A24

01b = A16/A32

10b = RESERVED

11b = A16 Only

(Bits 11-0) Manufacturer ID: This field uniquely identifies the manufacturer of the device.

FB5h = ASCOR

For A16 Modules, the register should read back a value of FFB5h.
For A16/A24 Modules, the register should read back a value of CFB5h.
For A16/A32 Modules, the register should read back a value of DFB5h.

A write to this 16-bit register is provided for Dynamic Configuration protocol. This register should only be written to by a resource manager. ***Do not write to this register***.

Offset	Description
02h	**Device Type (read/write)** A read of this 16-bit register provides information about the Switching Module Module's Device Type. This register indicates how much VMEbus memory is required by the VXI module, as well as the manufacture's unique model code.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Required Memory				Model Code											

(Bits 15-12) Required Memory: This field contains the value used for determining the A24 or A32 memory space resident on the device

7h = 64K bytes in A24 Address Space

Fh = 64K bytes in A32 Address Space

(Bits 11-0) Model Code: This field contains the manufacturer's unique module identifier.

F2Bh = The ASCOR Model Code for the Switching Module. This number is different from the ASCOR Model number. For the Switching Module, the register should read back a value of 7F2Ah. A write to this 16-bit register is provided for VXIbus definition. ***Do not write to this register***.

Offset	Description
04h	**Status Register (read) / Control Register (write)** A read of this 16-bit register provides information about the Switching Module status.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A24/A32	MODID	Device Dependent (Not used)										Rea-dy	Pas-sed	Dev. Dep.	

(Bit 15) A24/A32 Active: This bit indicates the accessibility of A24 or A32 registers.

1b = A24 or A32 Address Space Active

0b = A24 or A32 Address Space not Active (for A16 only devices)

(Bit 14) MODID: This bit indicates if the device is selected via the P2 MODID line.

1b = Device is not selected via the P2 MODID line. Used by the resource manager during Dynamic configuration.

(Bits 13-4) Device Dependent (Not used)

(Bit 3) Ready: This bit indicates if the device is ready to accept operational commands.

1b = Device is ready after power-on initialization sequence.

(Bit 2) Passed: This bit indicates if the power-on self test has successfully completed.

1b = Device does not support power-on self test (always pass)

(Bits 1-0) Device Dependent

00b = State of the corresponding bits of the Control register.

For the Switching Module Module, the register should read back a value of FFFCh.

A write to this 16-bit register causes specific actions to be executed.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A24/A32	Device Dependent(Not used)													Sys-fail	Dev.Rst

(Bit 15) A24/A32 Enable: This bit enables or disables A24 / A32 VMEbus registers.

1b = Enables A24 or A32 VMEbus registers. This bit must always remain a one after being set to one by the resource manager.

0b = Disables A24 or A32 VMEbus registers. This bit must always remain a zero after being cleared to zero by the resource manager. (for A16 only devices)

(Bits 14-2) Device Dependent (Not used) (Bit 1) Sysfail Inhibit: This bit controls the device's ability to drive the SYSFAIL line.

0b = Always set to zero (Sysfail not inhibited)

(Bit 0) Device Reset: This bit controls the state of the device

1b = Reset the device to power-on state.

0b = Normal operational mode

06h Offset Register (read/write)

A read of the 16-bit register provides information for calculating the base address of the Switching Module A24 operational registers.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset Value								Not used							

(Bits 15-8) Offset Value: This field is used for calculating the A24 Base Address.

(Bits 7-0) Don't Care bits (Not used)

To obtain the A24 base address for the Switching Module, take the 8 most significant bits of the Offset register and map them to the 8 most significant bits of the A24 Base Address. All other bits in the A24 Base Address are set to zeroes. For more detail refer to Q&A Section***: Q: How do I calculate the Switching Module Module's A24 Base Address?.***

A write to this 16-bit register is provided for Dynamic Configuration protocol. This register should only be written to by a resource manager. ***Do not write to this register.***

2.2 VXI Device Class Dependent Registers

The ASCOR Switching Module General Purpose Switch VXI Module does not use nor provide any of the 12 Device Class Dependent Registers.

Offset	Description
08h	Not Used
0Ah	Not Used
0Ch	Not Used
0Eh	Not Used
10h	Not Used
12h	Not Used
14h	Not Used
16h	Not Used
18h	Not Used
1Ah	Not Used
1Ch	Not Used
1Eh	Not Used

2.3 VXI Device Dependent Registers for the Switching Modules

The VXI Specification defines 16 Device Dependent Registers in the A16 address space following the Device Class Dependent Register space. Each register is 16 bits wide. The first 15 registers are not used nor provided by the ASCOR Switching Module General Purpose Switch, only the last register is used.

Offset	Description
20h	Not Used
22h	Not Used
24h	Not Used
26h	Not Used
28h	Not Used
2Ah	Not Used
2Ch	Not Used
2Eh	Not Used
30h	Not Used
32h	Not Used
34h	Not Used
36h	Not Used
38h	Not Used
3Ah	Not Used
3Ch	Not Used

3Eh Control Register (read/write)

A read of the 16-bit register provides the module control status.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										IRQ Level Select			Rvd	Reg Bit	Coil Ena

(Bits 15-6) Reserved

0h = Should always readback zeroes

(Bits 5-3) IRQ Level Select: These bits reflect the module's Interrupt Request Level
0h = No IRQ Level Selected, module Interrupt disabled.
1h = IRQ Level 1 Selected, module Interrupt enabled.
2h = IRQ Level 2 Selected, module Interrupt enabled.
3h = IRQ Level 3 Selected, module Interrupt enabled.
4h = IRQ Level 4 Selected, module Interrupt enabled.
5h = IRQ Level 5 Selected, module Interrupt enabled.
6h = IRQ Level 6 Selected, module Interrupt enabled.
7h = IRQ Level 7 Selected, module Interrupt enabled.

(Bit 2) Reserved
0h = Should always readback zero

(Bit 1) Reg Bit: This bit indicates the device's readback mode.
0b = Relay coil state readback is enabled

(Bit 0) Coil Enable: This bit indicates the device's coil driver state.
0b = Relay coil driver is enabled
1b = Relay coil driver is disabled

A write to this 16-bit register is sets the module control.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										IRQ LevelSelect			Rvd	RegBit	CoilEna

(Bits 15-6) Reserved
0h = Should always be set to zeroes

(Bits 5-3) IRQ Level Select: These bits select the module's Interrupt Request Level
0h = No IRQ Level Selected, module Interrupt disabled.
1h = IRQ Level 1 Selected, module Interrupt enabled.
2h = IRQ Level 2 Selected, module Interrupt enabled.
3h = IRQ Level 3 Selected, module Interrupt enabled.
4h = IRQ Level 4 Selected, module Interrupt enabled.
5h = IRQ Level 5 Selected, module Interrupt enabled.
6h = IRQ Level 6 Selected, module Interrupt enabled.
7h = IRQ Level 7 Selected, module Interrupt enabled.

(Bit 2) Reserved
0h = Should always be set to zero

(Bit 1) Reg Bit: This bit controls the device's readback mode.
0b = Enable relay coil state readback
1b = Enable data register state readback

(Bit 0) Coil Enable: This bit controls the device's relay coil driver.
0b = Enable relay coil driver
1b = Disable relay coil driver

2.4 VXI Device Dependent Registers for non-Switching Modules

The VXI Specification defines 16 Device Dependent Registers in the A16 address space following the Device Class Dependent Register space. Each register is 16 bits wide. The first 15 registers are not used nor provided by the ASCOR Switching Module General Purpose Switch, only the last register is used.

Offset	Description
20h	Custom Use
22h	Custom Use
24h	Custom Use
26h	Custom Use
28h	Custom Use
2Ah	Custom Use
2Ch	Custom Use
2Eh	Custom Use
30h	Custom Use
32h	Custom Use
34h	Custom Use
36h	Custom Use
38h	Custom Use
3Ah	Custom Use
3Ch	Custom Use

Chapter 3 Register Descriptions

3.1 ASCOR Custom Registers for the ASCOR Switching Module

The ASCOR Switching Module Custom Registers are located in the A24 Address Space. The custom registers start at offset 8000h within the A24 address space assigned by the resource manager. All Switching Module custom registers can be accessed as a 16-bit word. Additionally, since the ASCOR Switching Module General Purpose Switch features VXIMAX™, the custom registers can also be accessed as 32-bit words.

Offset Address in 16-Bit Mode

The Switching Module Custom Registers can be accessed in 16-bit mode. Address offsets for the custom registers increment by two (2). Sample offsets for the ASCOR Custom Registers in 16-bit word mode:

Offset	Description
8000h	First Custom Register
8002h	Second Custom Register

Offset address in 32-Bit Mode

The Switching Module Custom Registers can also be accessed in 32-bit mode. Address offsets for the custom registers increment by four (4). Sample offsets for the ASCOR Custom Registers in 32-bit word mode:

Offset	Description
8000h	First Custom Register
8004h	Second Custom Register
8008h	Third Custom Register

3.2 ASCOR Relay Registers

A subset of the ASCOR Switching Module Custom Registers are relay registers. See connector assignments for pin and channels assignments with associated relays.

Switching Module Relay Registers	
8000h	Relays K1-16
8002h	Relays K17-32
8004h	Relays K33-48

These registers, which are also located in the A24 Address Space, have a few unique properties:

Read / Write ASCOR Relay Registers are read / write registers. When a relay register is read the states of the coils associated to that register are returned. Normally, the states of the coils should match the values which were written. They may not match when error conditions occur or when relay coil drivers are disabled. On some rare occasions, the design of the module may not permit matching results.

Chapter 4 Programming with Registers

4.1 Programming VXI Device Registers

The Switching Module VXI Device Registers are read / write registers, but some device registers must be written only by the resource manager. See Section 2: VXI Device Register Description for registers reserved for the resource manager access. Since all 16 bits are programmed with a single write operation, care must be taken when values are written to the device registers in order to prevent unintended function enabling or disabling. In order to preserve the states of the functions that you do not want to alter, perform the following sequence of operations: 1. Read the device register first, 2. Modify only the bits you intend to program using the copy of the device register, 3. Write the new value back to the device register.Here are some example codes for reading the Switching Module ID Register.

Example using National Instruments NI-VXI calls with the Logical Address of 5

```
/*  C code segment for reading the ID Register using VXIinReg call.*/
        int16    ret;
        uint16   la        = 5;      /* Logical Address */
        uint16   reg       = 0;      /* ID Register offset */
        uint16    value16;

        /* Read the ID Register */
        ret = VXIinReg (la, reg, &value16);

        /* Check for read error */
        if (ret << 0)
                        /*  Error occurred during read.  */;

/*  C code segment for reading the ID Register using VXIin call.  */

        int16    ret;
        uint16   accessparms     = 1;       /* A16, Nonprivileged data access, Motorola Byte Order */
        uint32   address         = 0xC140;        /* LA * 0x40  0xC000  ID Register offset  */
        uint16   width           = 2;              /* 16-bit word */
        uint16    value16;

        /* Read the ID Register */
        ret = VXIin (accessparms, address, width, &value16);

        /* Check for read error */
        if (ret << 0)
                /*  Error occurred during read.  */;
```

Example using VXIplug&play VISA calls

/* C code segment for reading the ID Register using ViIn16 call. */

```
ViStatus            as3xxx_status;
ViSession    vi;            /* vi from previous call to as3xxx_init */
ViUInt16            space   = VI_A16_SPACE;
ViBusAddress offset  = 0x00; /* Offset of the ID Register */
ViUInt16            val16;

/* Read the ID Register */
as3xxx_status = viIn16 (vi, space, offset, &val16);

/* Check for read error */
if (as3xxx_status << VI_SUCCESS)
        /* Error occurred during read. */;
```

4.2 Resetting ASCOR VXI Module

The ASCOR Switching Module General Purpose Switch can be reset to a power up state by setting the Device Reset bit of the Status / Control (04h) register in the VXI Configuration Registers. Care must be taken when writing to this register since all bits other than the Device Reset bit must not be changed. In order to preserve the states of all other bits, perform the following sequence of operations:

1. Read the Status / Control register,
2. Set only the Device Reset bit,
3. Write the modified word to the Status / Control register.

After the reset operation, the module must be brought back to a normal operational mode in order for the relays to close. The Switching Module can be set back to the normal operation mode by clearing the Device Reset bit without modifying any other bits. Here are some example codes for resetting the Switching Module.

Example using National Instruments NI-VXI calls with the Logical Address of 5

/* C code segment for resetting the ASCOR VXI module using VXIinReg and VXIoutReg calls */

```
int16   ret;    uint16  la      = 5;    /* Logical Address */
uint16  reg     = 4;    /* Status / Control register offset */
uint16  value16;

/* Read the Status / Control Register */
ret = VXIinReg (la, reg, &value16);
```

```
        /* Check for read error */
        if (ret << 0)
                /*  Error occurred during read.  */;

        /* Set the Device Reset bit in the copy of the Status / Control Register */
        value16 |= 0x0001;

        /* Write to the Status / Control Register */
        ret = VXIoutReg (la, reg, value16);

        /* Check for write error */
        if (ret << 0)
                /*  Error occurred during write.  */;


/* Bring the module back to the normal operation by
        clearing the Device Reset bit in the copy of the Status / Control Register */
        value16 &= 0xFFFE;

        /* Write to the Status / Control Register */
        ret = VXIoutReg (la, reg, value16);

        /* Check for write error */
        if (ret << 0)
                /*  Error occurred during write.  */;

        /*  C code segment for resetting the ASCOR VXI module using VXIin and VXIout calls  */
        int16     ret;
        uint16   accessparms     = 1;              /* A16, Nonprivileged data access, Motorola Byte Order
        */
        uint32   address         = 0xC144;         /* LA * 0x40  0xC000  Control / Status Register */
        uint16   width           = 2;               /* Word */
        uint16   value16;
        uint32   value32;

        /* Read the Status / Control Register */
        ret = VXIin (accessparms, address, width, &value16);

        /* Check for read error */
        if (ret << 0)
                /*  Error occurred during read.  */;

        /* Set the Device Reset bit in the copy of the Status / Control Register */
        value32 = value16;
        value32 |= 0x0001;

        /* Write to the Status / Control Register */
```

```
ret = VXIout (accessparms, address, width, value32);

/* Check for write error */
 if (ret << 0)
        /*  Error occurred during write.  */;

/* Bring the module back to the normal operation by
clearing the Device Reset bit in the copy of the Status / Control Register */
value32 &= 0xFFFE;

/* Write to the Status / Control Register */
ret = VXIout (accessparms, address, width, value32);

/* Check for write error */
if (ret << 0)
        /*  Error occurred during write.  */;
```

Example using VXIplug&play VISA calls

```
/*  C code segment for resetting the ASCOR VXI Module  */

ViStatus        as3xxx_status;
ViSession       vi;             /*  vi from previous call to as3xxx_init  */
ViUInt16        space   = VI_A16_SPACE;
ViBusAddress    offset  = 0x04; /*  Offset of the Status / Control Register  */
ViUInt16        value16;

/* Read the Status / Control Register */
as3xxx_status = viIn16 (vi, space, offset, &value16);

/* Check for read error */
if (as3xxx_status << VI_SUCCESS)
        /*  Error occurred during read.  */;

/* Set the Device Reset bit in the copy of the Status / Control Register */
value16 |= 0x0001;

/* Write to the Status / Control Register */
as3xxx_status = viOut16 (vi, space, offset, value16);

/* Check for write error */
if (as3xxx_status << VI_SUCCESS)
        /*  Error occurred during write.  */;

/* Bring the module back to the normal operation by   clearing the Device Reset bit in the copy
of the Status / Control Register */
```

```
value16 &= 0xFFFE;

/* Write to the Status / Control Register */
as3xxx_status = viOut16 (vi, space, offset, value16);

/* Check for write error */
if (as3xxx_status << VI_SUCCESS)
        /*  Error occurred during write.  */;
```

4.3 Programming the ASCOR Switching Module Custom Registers

The ASCOR Switching Module Custom Registers can be accessed through the registers in the A24 address space. Since all 16 or 32 bits are programmed with a single write operation, care must be taken when values are written to these registers in order to prevent unintended side effects. To preserve the configuration that you do not want to program, perform the following sequence of operations:

1. Read the register first,
2. Modify only the bits you intend to program using the copy of the register,
3. Write the new value back to the register.

Refer to Section 1 for the definition of the custom registers found in the Switching Module.All Switching Module Custom Registers are located in the A24 address space. A unique A24 base address is assigned by the resource manager to the A24 module in the system. The assignment of the base address is performed every time when the resource manager is executed. The Switching Module Custom Registers start at an offset from the module's assigned A24 base address. The sum of the two values, A24 base address and the custom register offset, gives the unique custom register address. Some interface library calls require the A24 custom register address. VXIplug&play library calls require only the offset of the register from the base address. The A24 base address is added to the offset internally. Here are some example codes for writing to the Switching Module custom registers.

Example using National Instruments NI-VXI calls

```
/*  C code segment for writing the value 0x1000 to the first custom register,    assume A24 Base Address of 200000h */
        int16    ret;
        uint16   accessparms     = 2;      /* A24, Nonprivileged data access, Motorola Byte Order */
        uint32   address;
        uint16   width           = 2;      /* Word */
        uint32   value32;

        address = 0x208000;      /*  A24 Base Address  offset of the first custom register  */
        value32 = 0x1000;        /*  Value to write to the first custom register  */

        /* Write to the first custom register */
        ret = VXIout (accessparms, address, width, value32);

        /* Check for write error */
        if (ret << 0)
                /*  Error occurred during write.  */;
```

Example using VXIplug&play VISA calls

/* C code segment for writing the value 0x1000 to the first custom register */

```
ViStatus        as3xxx_status;
ViSession       vi;
 ViUInt16       space    = VI_A24_SPACE;
ViBusAddress    offset    = 0x8000;       /*  Offset of the first custom register  */
ViUInt16         value16;

value16= 0x1000;        /*  Value to write to the first custom register  */

/* Write to the first custom register */
as3xxx_status = viOut16 (vi, space, offset, value16);

/* Check for write error */
if (as3xxx_status << VI_SUCCESS)
        /*  Error occurred during write.  */;
```

4.4 Programming ASCOR Custom Registers for A 16/A24 Module

ASCOR Custom Registers can be accessed through the registers in the A24 address space. Since all 16 or 32 bits are programmed with a single write operation, care must be taken when values are written to these registers in order to prevent unintended side effects. To preserve the configuration that you do not want to program, perform the following sequence of operations:

- Read the register first,
- Modify only the bits you intend to program using the copy of the register,
- Write the new value back to the register,

The definition of the custom registers can be found in ASCOR VXI Modules' Technical Manual. All ASCOR Custom Registers are located in the A24 address space. A unique A24 base address is assigned by the resource manager to the A24 module in the system. The assignment of the base address is performed every time when the resource manager is executed. ASCOR Custom Registers start at an offset from the module's assigned A24 base address. The sum of the two values, A24 base address and the custom register offset, gives the unique custom register address. Some interface library calls require the A24 custom register address, VXIplug&play library calls require only the offset of the register from the base address. The A24 base address is added to the offset internally. Here are some example codes for writing to the ASCOR custom registers.

Example using National Instruments NI-VXI calls

```
/*  C code segment for writing the value 0x1000 to the first custom register,    assume A24 Base Address of 200000h */
        int16    ret;
        uint16   accessparms     = 2;      /* A24, Nonprivileged data access, Motorola Byte Order */
        uint32   address;
        uint16   width           = 2;      /* Word */
        uint32   value32;

        address = 0x208000;      /*  A24 Base Address  offset of the first custom register  */
        value32 = 0x1000;        /*  Value to write to the first custom register  */

        /* Write to the first custom register */
        ret = VXIout (accessparms, address, width, value32);

        /* Check for write error */
        if (ret << 0)
                /*  Error occurred during write.  */;
```

Example using VXIplug&play VISA calls

/* C code segment for writing the value 0x1000 to the first custom register */

```
ViStatus        as3xxx_status;
ViSession       vi;
 ViUInt16       space    = VI_A24_SPACE;
ViBusAddress    offset    = 0x8000;       /*  Offset of the first custom register  */
ViUInt16         value16;

value16= 0x1000;        /*  Value to write to the first custom register  */

/* Write to the first custom register */
as3xxx_status = viOut16 (vi, space, offset, value16);

/* Check for write error */
if (as3xxx_status << VI_SUCCESS)
        /*  Error occurred during write.  */;
```

4.5 Programming ASCOR Custom Registers for A16/A32 Module

ASCOR Custom Registers can be accessed through the registers in the A32 address space. Since all 32 bits are programmed with a single write operation, care must be taken when values are Written to these registers in order to prevent unintended side effects, To preserve the configuration that you do not want to program, perform the following sequence of operations:

1. Read the register first,
2. Modify only the bits you intend to program using the copy of the register.
3. Write the new value back to the register.

The definition of the custom registers can be found in ASCOR VXI Modules' Technical Manual. All ASCOR Custom Registers are located in the Al2 address space. A unique A32 base address is assigned by the resource manager to the A32 module in the system. The assignment of the base address is performed every time when the resource manager is executed. ASCOR Custom Registers start at an offset from the module's assigned A32 base address. The sum of the two values, A32 base address and the custom register offset. gives the unique custom register address, Some interface library calls require the A32 custom register address. VXJplug&pfay library calls require only the offset of the register from the base address. The A32 base address is added to the offset internally.

Here are some example codes for writing to the ASCOR custom registers.

Example using National Instruments NI-VXI calls

```
/*  C code segment for writing the value 0x1000 to the first custom register,    assume A32 Base Address of 200000h */
        int16    ret;
        uint16   accessparms     = 3;      /* A32, Nonprivileged data access, Motorola Byte Order */
        uint32   address;
        uint16   width           = 4;      /* LongWord */
        uint32   value32;

        address = 0x208000;      /*  A32 Base Address  offset of the first custom register  */
        value32 = 0x1000;        /*  Value to write to the first custom register  */

        /* Write to the first custom register */
        ret = VXIout (accessparms, address, width, value32);

        /* Check for write error */
        if (ret << 0)
                /*  Error occurred during write.  */;
```

Example using VXIplug&play VISA calls

/* C code segment for writing the value 0x1000 to the first custom register */

```
ViStatus        as3xxx_status;
ViSession       vi;
 ViUInt16       space    = VI_A32_SPACE;
ViBusAddress    offset    = 0x8000;       /*  Offset of the first custom register  */
ViUInt16         value32;

value16= 0x1000;        /*  Value to write to the first custom register  */

/* Write to the first custom register */
as3xxx_status = viOut32 (vi, space, offset, value16);

/* Check for write error */
if (as3xxx_status << VI_SUCCESS)
        /*  Error occurred during write.  */;
```

Chapter 5 Miscellaneous Questions and Answers

5.1 Q: How do I calculate the Switching Module Module's A16 Base Address?

A: The A16 Base Address of the Switching Module is derived from the Logical Address.

The formula for calculating the A16 Base Address is as follows:

A16 Base Address = C000h + LA x 40h where LA is the Logical Address of a module

Logical Address	A16 Base Address
1	C040h
2	C080h
3	C0C0h
4	C100h
5	C140h

And so on . . .

If the module's Logical Address is 5 then A16 Base Address is C140h and Device Register addresses are as follows:

Address	Device Registers
C140h	ID Register / Logical Address Register
C142h	Device Type Register
C144h	Status / Control Register
C146h	Offset Register
C17Eh	Relay Control Register

If the module's Logical Address is 8 then A16 Base Address is C200h and Device Register addresses are as follows:

Address	Device Registers
C200h	ID Register / Logical Address Register
C202h	Device Type Register
C204h	Status / Control Register
C206h	Offset Register
C23Eh	Relay Control Register

5.2 Q: How do I calculate the Switching Module Module's A24 Base Address?

A: The A24 Base Address of the Switching Module can be derived from the value stored in the Offset Register (04h). To obtain the A24 base address, take the 8 most significant bits of the Offset register and map them to the 8 most significant bits of the A24 Base Address. All other bits in the A24 Base Address are set to zeroes.

Offset Register (06h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	0	X	X	X	X	X	X	X	X

A24 Base Address

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Following are some examples of the Offset Register Values and the corresponding A24 Base Addresses.

Offset Register Values	Derived A24 Base Addresses
20XXh	200000h
30XXh	300000h
70XXh	700000h

Alternatively, A24 Base Address of a device can be obtained by issuing a library call.

Example using National Instruments NI-VXI calls with the Logical Address of 5

```
/*  C code segment for obtaining the device's A24 Base Address  */
        int16    ret;
        uint16   la         = 5;      /* Logical Address */
        uint16   field      = 12;     /* Base of A24/A32 address space */
        uint32   baseA24;

        /* Get the A24 Base Address */
        ret = GetDevInfo (la, field, &baseA24);

        /* Check for function error */
        if (ret << 0)
                /*  Error occurred during GetDevInfo.  */;
```

5.3 Q: How do I calculate the Switching Module Module's A32 Base Address?

A: The A32 Base Address of the Switching Module can be derived from the value stored in the Offset Register (06h). To obtain the A32 base address, take the 16 most significant bits of the Offset register and map them to the 16 most significant bits of the A32 Base Address. All other bits in the A32 Base Address are set to zeroes. This conversion works for the modules whose Required Memory in the Device Type Register is set to Fh (15)

Offset Register (06h)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

A32 Base Address

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Following are some examples of the Offset Register Values and the corresponding A24 Base Addresses.

Offset Register Values	**Derived A24 Base Addresses**
20XXh	200000h
30XXh	300000h
70XXh	700000h

Alternatively, A24 Base Address of a device can be obtained by issuing a library call.

Example using National Instruments NI-VXI calls with the Logical Address of 5

```
/*  C code segment for obtaining the device's A24 Base Address  */
        int16    ret;
        uint16   la         = 5;      /* Logical Address */
        uint16   field      = 12;     /* Base of A24/A32 address space */
        uint32   baseA24;

        /* Get the A24 Base Address */
        ret = GetDevInfo (la, field, &baseA24);

        /* Check for function error */
        if (ret << 0)
                /*  Error occurred during GetDevInfo.  */;
```

5.4 Q: How do I Change the Switching Module Module's Logical Address?

A: The Logical Address of ASCOR Switching Module Module can be changed manually using the two rotary switches located in the back of the module. These rotary switches represent the high and low hex digits of an eight bit Logical Address. Orient the Switching Module module as shown in the illustration below. The rotary switch on the bottom (D1) represents the high hex digit and the rotary switch on the top (D2) represents the low hex digit. Each rotary switch can be turned clockwise or counter-clockwise. Turn each rotary switch until the desired hex digit is aligned with the small white dot on the left side of the rotary switch casing. Valid Logical Addresses for Static Configuration are between 01h (1) and FEh (254). The Logical Address of 00h (0) is reserved for Slot 0 computer. Do not set the Logical Address of ASCOR VXI Modules to 0. The example below shows the rotary switch settings for a Switching Module Logical Address of 53.

ASCOR VXI Modules also supports Dynamic Configuration methods of assigning Logical Addresses. In order for Dynamic Configuration to work properly the resource manager software must also support Dynamic Configuration. Set the rotary switches on the modules to FFh (255) so that the resource manager software can dynamically assign Logical Addresses to ASCOR VXI Modules.