

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

INSTRUCTION MANUAL

Sensoray Model 621/720

Digital I/O Board
(Rev F)
November 23, 2016

For Technical Support contact Sensoray Co., Inc.
7313 SW Tech Center Dr., Tigard, Oregon 97223, USA
Tel:(503) 684-8005 Fax:(503) 684-8164
e-mail: techsupport@sensoray.com
www.sensoray.com

TABLE OF CONTENTS

LIMITED WARRANTY **3**

SPECIAL HANDLING INSTRUCTIONS **4**

1. INTRODUCTION **5**

2. INSTALLATION **6**

3. RESET **6**

4. DEVICE AND VENDOR IDENTIFICATION **6**

5. DIGITAL I/O INTERFACE **6**

5.1 Overview 6
5.2 Registers 8
5.2.1 I/O Ports *9*
Example: Setting I/O direction 9
5.2.2 Latch/Output enable mode (channels 64-71) *9*
Example: Setting/Clearing Latch/Output enable 9
5.2.3 Edge type select register (channels 0-63) *10*
Example: Edge select register 10
5.2.4 Arming & Disarming Capture (channels 0-63) *10*
Example: Setting up to capture edges 11
5.2.5 Tabled Arming & Disarming Example *11*
5.2.6 Edge capture flag register (channels 0-63) *12*
Example: Reading Edge Capture Flag Register 12
5.2.7 Clearing Captured Edges *12*
Example: Clearing captured Edges 12

6. INTERRUPTS **12**

6.1 Overview 12
6.2 Individual interrupt enable/disable 12
6.3 Clearing an interrupt flag 13
6.4 Global Interrupt Enable/Disable 13
Example: GIE (Global Interrupt Enable) 13
6.5 Interrupt Flags 13

7. SX20 VERSION 0.6 DLL **14**

7.1 Overview 14
7.2 DLL Functions 14
X20_InitSystem *14*
X20_CloseSystem *14*
X20_WritePort *14*
X20_ReadPort *15*
X20_Reset *15*
X20_GetHDIO *15*
X20_SetBit *15*
X20_Clearbit *16*
X20_ReadBit *16*
X20_InterruptOpen *16*
X20_InterruptClose *16*
X20_InterruptEnable *17*
X20_InterruptDisable *17*
7.3 Types used in the SX20 DLL 17

APPENDIX A: SPECIFICATIONS.....18

APPENDIX B: ACCESSORIES.....18

APPENDIX C: CONNECTOR PINOUT.....19

APPENDIX D: REVISION INFORMATION.....20

APPENDIX E: TECHNICAL SUPPORT.....20

Limited Warranty

Sensoray Company, Incorporated (Sensoray) warrants the Model 621/720 hardware to be free from defects in material and workmanship and perform to applicable published Sensoray specifications for two years from the date of shipment to purchaser. Sensoray will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

The warranty provided herein does not cover equipment subjected to abuse, misuse, accident, alteration, neglect, or unauthorized repair or installation. Sensoray shall have the right of final determination as to the existence and cause of defect.

As for items repaired or replaced under warranty, the warranty shall continue in effect for the remainder of the original warranty period, or for ninety days following date of shipment by Sensoray of the repaired or replaced part, whichever period is longer.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. Sensoray will pay the shipping costs of returning to the owner parts that are covered by warranty.

Sensoray believes that the information in this manual is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, Sensoray reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult Sensoray if errors are suspected. In no event shall Sensoray be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, SENSORAY MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF SENSORAY SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. SENSORAY WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEROF.

Special Handling Instructions

The Model 621/720 circuit boards contains CMOS circuitry that is sensitive to Electrostatic Discharge (ESD).

Special care should be taken in handling, transporting, and installing the board to prevent ESD damage to the board. In particular:

- Do not remove the circuit board from its protective anti-static bag until you are ready to configure the board for installation.
- Handle the circuit board only at grounded, ESD protected stations.
- Remove power from the PCI system before installing or removing the circuit board.

1. Introduction

The Sensoray models 621/720 are 72 channel digital I/O cards. The features include:

- 72 digital I/O channels.
- 64 channels have edge detection and interrupt capability.
- 8 channels have input latch and output enable capability.
- 64 I/O channels are grouped in eight blocks of eight channels. 8 I/O channels are grouped in two blocks of four channels. Each block may be programmed as inputs or outputs.

The 620 is a PCI card whereas the 721 is a Compact PCI card. Except for where otherwise specified, this document applies to both models.

The board is powered from the 5V supply of the PCI bus. No auxiliary power supplies are required. Low power circuitry is used to minimize system power consumption and enhance reliability.

A high-density connector on the card's front panel is provided for connecting the I/O channels to external circuitry. A solid state relay board (see Sensoray model 720RB) will often be used to facilitate connections to real world circuits.

Block Diagram

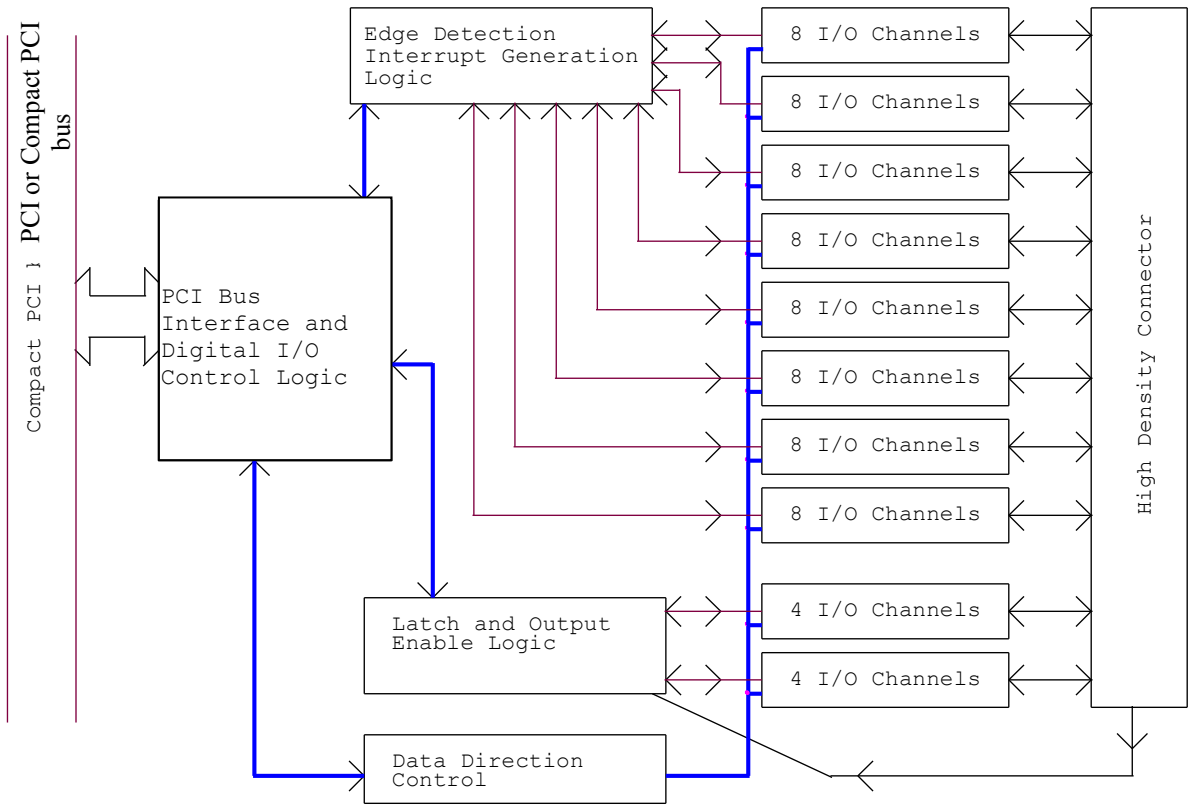


Figure 1

2. Installation

The installation of the board into a PCI or Compact PCI system is straight forward as no special adjustment (jumpers) are required.

3. Reset

After a system reset or at power up the board is in its default state

- Global interrupt enable (GIE) is disabled
- Channel interrupt enables are cleared (disabled)
- Event capture enables are indeterminate
- Edge selections are indeterminate
- All I/O channels are all set to input mode
- Output registers are indeterminate
- Channel 64 through 71 are set to simple I/O mode (no latching/output enable)
- Light emitting diode 'D1' will be on (controlled by LOE bit which is clear)

4. Device and Vendor Identification

This is the information that identifies a board in the system.

Device ID:	9050	Sub Device ID:	6000
Vendor ID:	10B5	Sub Vendor ID:	0001

5. Digital I/O Interface

Note: Any examples referred to in this manual are written in Visual Basic format. Look at the C & Visual Basic samples included with the board for more examples.

5.1 Overview

The board provides 72 digital I/O channels. 64 channels have edge detection capability. 8 channels have input latch and output enable capability.

The 64 edge detection capable channels (channels 0-63) are divided into groups of eight. Each group can be set as inputs or outputs by setting its corresponding data direction register (DDR).

If edge capture is desired, the arm capture register can be enabled for any particular input channel. In addition, either capture on a positive or negative going edge can be selected.

A capture event can generate an interrupt if a channel's corresponding interrupt enable bit is set. If an interrupt occurs, a channel interrupt flag will be set such that the user can determine the source.

The eight remaining I/O channels (channels 64-71) are divided into two groups of four. Each group can be set as inputs or outputs by setting its corresponding data direction register (DDR).

A latch/output enable (LOE) mode is available for channels 64 through 71. Setting the latch enable register (LOE) enables this mode. When enabled this feature is applied to all eight channels.

The LOE mode allows a user to latch the data inputs (channels 64 through 71) by asserting the LATCH input line. The data can be then read at the user's convenience. In addition, the outputs in this mode are tri-stated unless the user asserts the OUTEN input line.

5.2 Registers

I/O bank 0 I/O port **Offset: 00 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I/O CHANNELS 0-31																															

I/O bank 1 I/O port **Offset: 04 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I/O CHANNELS 32-63																															

I/O bank 2 I/O port **Offset: 08 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	**I/O CHANNELS 64-71**							

Bank 0 edge capture flags **Offset: 0C hex – Read only**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPTURE FLAGS CHANNELS 0-31																															

Bank 1 edge capture flags **Offset: 10 hex – Read only**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPTURE FLAGS CHANNELS 32-63																															

Bank 0 interrupt flags **Offset: 14 hex – Read only**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERRUPT FLAGS CHANNELS 0-31																															

Bank 1 interrupt flags **Offset: 18 hex – Read only**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERRUPT FLAGS CHANNELS 32-63																															

Bank 0 edge select register **Offset: 1C hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDGE SELECT REGISTERS CHANNELS 0-31																															

Bank 1 edge select register **Offset: 20 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDGE SELECT REGISTERS CHANNELS 32-63																															

Bank 0 arm capture register **Offset: 24 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARM CONTROL REGISTERS CHANNELS 0-31																															

Bank 1 arm capture register **Offset: 28 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARM CONTROL REGISTERS CHANNELS 32-63																															

Bank 0 interrupt enable register **Offset: 2C hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERRUPT ENABLE REGISTERS CHANNELS 0-31																															

Bank 1 interrupt enable register **Offset: 30 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERRUPT ENABLE REGISTERS CHANNELS 32-63																															

Data direction/Misc./Status register **Offset: 34 hex – Read/Write**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GIRQ*	X	X	X	IRQ3*	IRQ2*	IRQ1*	IRQ0*	CAP3*	CAP2*	CAP1*	CAP0*	X	X	**LOE**	GIE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARM	X	X	X	X	X	**DDR9** 68-71	**DDR8** 64-67	**DDR7** 56-63	**DDR6** 48-55	**DDR5** 40-47	**DDR4** 32-39	**DDR3** 24-31	**DDR2** 16-23	**DDR1** 8-15	**DDR0** 0-7

Bits marked with an * are read only.

5.2.1 I/O Ports

Reading an I/O port returns either the input logic level (input mode) or the state of the output register (output mode).

The DDR registers are used to set a particular I/O group to function as either inputs or outputs. The DDR bits affect the I/O channels as shown in the table below.

DDR9	DDR8	DDR7	DDR6	DDR5	DDR4	DDR3	DDR2	DDR1	DDR0
CHANNELS 68-71	CHANNELS 64-67	CHANNELS 56-63	CHANNELS 48-55	CHANNELS 40-47	CHANNELS 32-39	CHANNELS 24-31	CHANNELS 16-23	CHANNELS 8-15	CHANNELS 0-7

If a DDR bit is set to zero the associated channels will be inputs. When a DDR bit is set to one the associated channels will be outputs.

If a channel is changed to an output the setting of the output register will be output. To avoid unwanted output levels when making a channel group outputs, preset the output registers (write to the associated channels) with desired values.

Note: The values read and written to the I/O ports are true (i.e. not inverted).

Example: Setting I/O direction

```
Call WritePort(Handle, &H34, &H3FF, &H3FF)    'This will make all the channels outputs

Call WritePort(Handle, &H34, &H0, &H3FF)      'This will make all the channels inputs
```

5.2.2 Latch/Output enable mode (channels 64-71)

Latching and Output Enable is only available on channels 64-71.

When the LOE bit (bit 17) of the "Data direction/Misc./Status" register is set to one, LOE mode is in effect.

In this mode data present on input channels 64-71 is latched into a holding register as the LATCH strobe input falls from 5V to 0V. The host any time thereafter can read the latched data.

Additionally, any channels that have been set as outputs are inactive (tri-state) unless OUTEN strobe input is brought low.

Note: If this mode is disabled the channels act as straight I/O ports.

The light emitting diode 'D1' is linked to the LOE bit of the "Data direction/Misc./Status" register. When this bit is 0 the LED is on and when it is 1 the LED if off.

Example: Setting/Clearing Latch/Output enable

```
Call SetBit(Handle,&H34,17)                   'Sets the LOE bit of the DDR/Misc register

Call ClearBit(Handle,&H34,17)                 'Clears the LOE bit of the DDR/Misc register
```

5.2.3 Edge type select register (channels 0-63)

Input channels may be configured to detect either rising or falling edge transitions.

Set the bits in the “edge select” to “0” to detect rising edges for the associated relay channels, or to “1” to detect falling edges.

Example: Edge select register.

```
Call WritePort(Handle, &H1C, &H0, &HFFFFFFFF)        'Setup Channel 0-31 to capture leading edges

Call WritePort(Handle, &H20, &HFFFFFFFF, &HFFFFFFFF) 'Setup Channel 32-63 to capture trailing
                                                      'edges
```

5.2.4 Arming & Disarming Capture (channels 0-63)

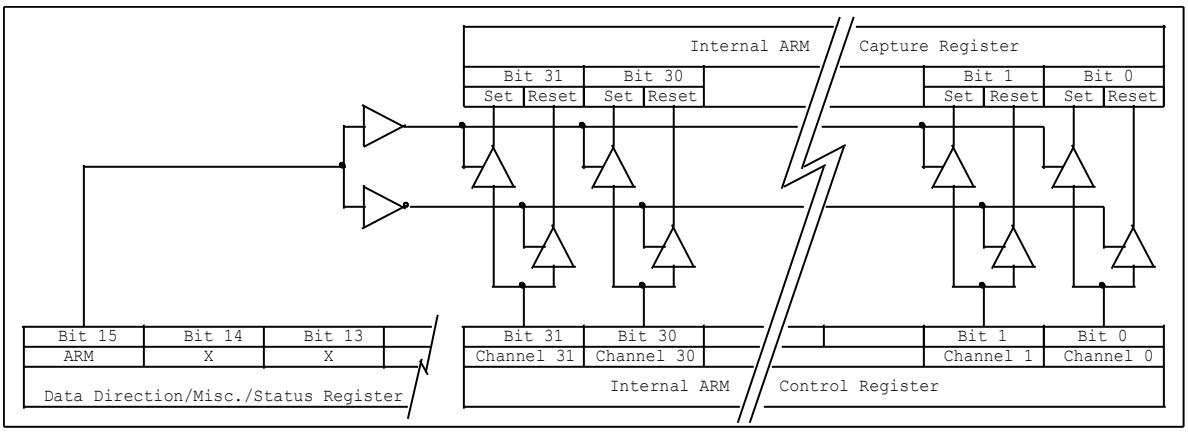


Figure 2

A relay channel must be "armed" before it will capture a detected edge.

An internal ARM Capture register is used to arm or disarm edge capturing. A “1” in the corresponding channel position will allow edge capturing while a “0” will disable edge capturing on that channel. The host system does not have direct access to this register. It is written to using the ARM Control register and the “ARM” bit of the “Data direction/Misc./Status” register.

If ARM bit = “1”: All channels whose corresponding bit in the ARM Control register is a “1” are armed. Those channels with a “0” in the ARM Control register remain unchanged.

If ARM bit = “0”: All channels whose corresponding bit in the ARM Control register is a “1” are disarmed. Those channels with a “0” in the ARM Control register remain unchanged.

See Figure 2 for a diagram of the process.

Reading the “ARM Control” register accesses the internal ARM Capture register directly and will give the current status of the register. A “0” means that channel is disabled while a “1” shows it is armed.

If a channel is armed for edge capture and the same channel is set for output, a change in output level could result in a captured event. Unless you want a “loop-back” event, a channel set for output should have capturing disabled.

Example: Setting up to capture edges

```
Call ClearBit(Handle, &H34, 15)                          'Clear the arm bit in the DDR
                                                         'register
Call WritePort(Handle, &H24, &HFFFFFFFF, &HFFFFFFFF)     'Clears all previous captured events
                                                         '& associated interrupts
Call WritePort(Handle, &H28, &HFFFFFFFF, &HFFFFFFFF)     'Clears all previous captured events
                                                         '& associated interrupts
Call WritePort(Handle, &H1C, &HFFFFFFFF, &HFFFFFFFF)     'Setup to capture trailing (1-0)
                                                         'edges
Call WritePort(Handle, &H1C, &H0, &HFFFFFFFF)            'Setup to capture leading (0-1) edges
Call SetBit(Handle, &H34, 15)                            'Set the arm bit in the DDR register
Call WritePort(Handle, &H24, &HFFFFFFFF, &HFFFFFFFF)     'Enables Edge capture on all channels
Call WritePort(Handle, &H28, &HFFFFFFFF, &HFFFFFFFF)     'Enables Edge capture on all channels
Call WritePort(Handle, &H2C, &HFFFFFFFF, &HFFFFFFFF)     'Enables all interrupts on edge
                                                         'capture
Call WritePort(Handle, &H30, &HFFFFFFFF, &HFFFFFFFF)     'Enables all interrupts on edge
                                                         'capture
Call InterruptEnable(Handle)                             'Enable the interrupts
DoEvents
```

5.2.5 Tabled Arming & Disarming Example

Operation	ARM bit of DDR/Misc. (bit 15)	ARM Capture Register (Chan 31 ..... Chan 0)
Starting state of Internal ARM Capture Register: (Presumed for the example)		0.....00000000
To program channel 0, 2, 3 to capture an edge, write a “1” to the ARM bit of the DDR/Misc. register and write to ARM Control Register with a “1” in the corresponding bits for channels 0, 2 & 3.	1	0.....00001101
Result stored in the Internal ARM Capture Register: (Channels 0,2 & 3 will be captured)		0.....00001101
To also capture channel 6, write a “1” to the ARM bit of the DDR/Misc. register and write a “1” to bit 6 of the ARM Control Register.	0	0.....01000000
Result stored in the Internal ARM Capture Register: (Channels 0,2, 3 & 6 will be captured)		0.....01001101
To Disarm channel 3 & 0, write a “0” to the ARM bit of the DDR/Misc. register and write a “1” to bits 3 & 0 of the ARM Control Register.	0	0.....00001001
Result stored in the Internal ARM Capture Register: (Channels 0 & 3 will be disarmed) (Channels 2 & 6 will be captured)		0.....01000100

Figure 3

5.2.6 Edge capture flag register (channels 0-63)

Each channel is associated with a one-bit capture register. The capture register is responsible for logging the occurrence of a detected edge. When a selected edge occurs while capture is enabled, the associated capture flag is set. All the capture flags, for a given I/O bank, can be read simultaneously through the edge capture flag port.

Note that before the capture register can log a detected edge, the channels capture register must be armed. After capturing an edge, the capture register remains set until explicitly reset (cleared) by the host.

Example: Reading Edge Capture Flag Register

```
EdgeCap1=ReadPort(Handle,&H0C)              'Reads Edge Capture Flags for channels 0-31
EdgeCap2=ReadPort(Handle,&H10)              'Reads Edge Capture Flags for channels 32-63
```

5.2.7 Clearing Captured Edges

Sometime after a captured edge is detected, it is necessary to reset the associated capture register. This is accomplished by disarming the associated “arm capture” register.

Example: Clearing captured Edges

```
Call ClearBit(Handle, &H34, 15)                          'Clear the arm bit in the DDR
                                                         'register
Call WritePort(Handle, &H24, &HFFFFFFFF, &HFFFFFFFF)     'Clears all previous captured events
                                                         '& associated interrupts (0-31)
Call WritePort(Handle, &H28, &HFFFFFFFF, &HFFFFFFFF)     'Clears all previous captured events
                                                         '& associated interrupts (31-63)
```

6. Interrupts

6.1 Overview

I/O channels 0-63 may be individually programmed to request interrupt service upon capture of a selected edge. Obviously if a channel has not been enabled to capture edges it cannot generate an interrupt. When an interrupt occurs the interrupt routine needs to check all the interrupt flags as there could be more that one interrupt. The speed at which interrupts can be received will depend greatly on the language used, speed of the processor and the programming style of the interrupt routine.

6.2 Individual interrupt enable/disable

The channels may be individually programmed to request interrupt service upon capture of a detected edge. Enable or disable a channels interrupt by writing a ‘1’ to the corresponding interrupt enable register. Set the bits in the Interrupt Enable register to enable interrupts for the associated I/O channels, or clear the bits to disable interrupts.

6.3 Clearing an interrupt flag

An interrupt flag is cleared by clearing its associated capture register. This is accomplished clearing the associated Arm Capture register. See the section Clearing Captured Edges.

6.4 Global Interrupt Enable/Disable

Clearing this bit in the DDR/Misc. register will disable all interrupts simultaneously. Setting the bit will allow enabled interrupts to generate a system interrupt.
When GIE is disabled, interrupts are prevented from reaching the PCI bus host, regardless of other interrupt enable states (i.e. I/O channel interrupt enables). If interrupts are to be recognized, GIE must be enabled.
This bit can be manipulated in two ways:

Example: GIE (Global Interrupt Enable)

```
       Call InterruptEnable(Handle)                'Sets bit 16 of DDR/Misc. register
       Call InterruptDisable(Handle)
Or
       Call SetBit(Handle,&H34,16)                 'Sets bit 16 of DDR/Misc. register
       Call ClearBit(Handle,&H34,16)               'Sets bit 16 of DDR/Misc. register
```

6.5 Interrupt Flags

A ‘1’ in the Interrupt Flag register signifies that an interrupt has occurred on the corresponding channel input since the last time the interrupts were enabled.

7. SX20 Version 0.6 DLL

7.1 Overview

The DLL needs to reside in the windows system directory.
During development the SX20.h & SX20f.h must be in the same directory as the users source code.
Note that the users interrupt callback function must use the __stdcall calling convention.
When using Visual Basic it was found that a DoEvents command needs to be issued after calling functions related to the interrupt routine. This hands control back to the operating system so that it can execute the function.
Only one instance of the SX20 DLL needs to be running to be able to handle up to a maximum of eight cards.

7.2 DLL Functions

X20_InitSystem

Must be run before any of the functions that follow can be called.

Prototype

```
ECODE X20_InitSystem (PCI * pci)
```

Parameters

pci - information of cards installed.

pci.boards - number of boards installed. (Supplied by DLL)
pci.PCIslot[i] - slot in which card 'i' is placed. (Supplied by DLL)

Return Value

Returns an error code if initialization fails. Otherwise zero is returned.

X20_CloseSystem

Must be called when finished using the driver.

Prototype

```
void X20_CloseSystem (void)
```

X20_WritePort

This function is used to write to any port on the selected card.

Prototype

```
void X20_WritePort(HDIO han, DWORD offset, DWORD value, DWORD mask)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.
offset - Offset address of register on card
value - 32-bit value to write
mask - 32-bit mask value for writing data to selected port. Only bits set to '1' will be affected by write function. If mask is set to 0xffffffff, all the bits of the register will be affected.

X20_ReadPort

This function is used to read any port on the selected card.

Prototype

```
DWORD X20_ReadPort(HDIO han, DWORD offset)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.
offset - Offset address of register on card.

Return Value

Returns 32-bit value from selected port.

X20_Reset

Generates a software reset on the selected card.

Prototype

```
void X20_Reset(HDIO han)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.

X20_GetHDIO

Knowing the slot number in which a card is installed, this function allows the user to determine its handle number.

Prototype

```
ECODE X20_GetHDIO(HDIO *han, DWORD slot)
```

Parameters

*han - Pointer to card handle
slot - Slot number of card installed

X20_SetBit

Used to set any bit at any register offset, on the selected card, to ‘1’.

Prototype

```
void X20_SetBit(HDIO han, DWORD offset, DWORD bit)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.
offset- Offset of the register containing the bit to be set.
bit- Corresponding number of the bit to be set (0-31).

X20_Clearbit

Used to clear any bit at any register offset, on the selected card, to '0'.

Prototype

```
void X20_ClearBit(HDIO han, DWORD offset, DWORD bit)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.
offset- Offset of the register containing the bit to be cleared.
bit- Corresponding number of the bit to be cleared (0-31).

X20_ReadBit

Used to read any bit at any register offset, on the selected card.

Prototype

```
void X20_ReadBit(HDIO han, DWORD offset, DWORD bit)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.
offset- Offset of the register containing the bit to be read.
bit- Corresponding number of the bit to be read (0-31).

Return Value

Returns '1' is the bit is set, '0' if it is clear.

X20_InterruptOpen

Opens an interrupt thread for the selected card. This thread should be closed before the DLL is closed. Giving the interrupt thread too high a priority can cause other components of the windows operating system to slow down especially while many interrupts are being handled.

Prototype

```
ECODE X20_InterruptOpen(HDIO han, FPTR handleFunction, DWORD priority)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.
handleFunction- Address of the users interrupt callback routine. It must use the __stdcall calling convention.
priority- A setting used by windows to determine the interrupt thread's priority.
THREAD_PRIORITY_LOWEST = -2
THREAD_PRIORITY_BELOW_NORMAL = -1
THREAD_PRIORITY_NORMAL = 0
HREAD_PRIORITY_ABOVE_NORMAL = 1
THREAD_PRIORITY_HIGHEST = 2
THREAD_PRIORITY_TIME_CRITICAL = 15

X20_InterruptClose

Closes an open interrupt thread for the selected card.

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.

X20_InterruptEnable

Enables the interrupts by setting the GIE bit of the DDR/Misc register on the selected card.

Prototype

```
void X20_InterruptEnable(HDIO han)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.

X20_InterruptDisable

Disables the interrupts by clearing the GIE bit of the DDR/Misc register on the selected card.

Prototype

```
void X20_InterruptDisable(HDIO han)
```

Parameters

han - Card handle. If there is only one card installed this value will be zero. If there is more than one card in the system, this value will correspond to the card number.

7.3 Types used in the SX20 DLL

```
typedef DWORD HDIO;                     //dio board handle;
typedef DWORD ECODE;                    //error code;

typedef struct
{                                       //system structure;
    DWORD boards;                       //number of compatible PCI boards found;
    DWORD PCIslot[SYS_DIOS];            //PCI slot numbers for boards found;
} PCI;

typedef struct
{
    HDIO han;
    FPTR  func;                         //pointer to users function;
    DWORD priority;                     //interrupt thread priority;
    DWORD total;                        //number of interrupts occurred;
    DWORD lost;                         //number of interrupts lost;
} INT_DATA;
```

Appendix A: Specifications

General

Interface	Parameter	Description
Bus	Type	PCI/ Compact PCI, 32-bit, 33MHz
	Address requirements	I/O registers: 56-bytes memory address block, PCI configuration registers require an additional amount.
I/O	Input characteristics	TTL/CMOS compatible.

Limits

	Parameter	Typical	Min	Max	Units
I/O	Output sink current			20	mA
	Output source current			20	mA
	Input current			±10	uA
	Total output current (all outputs)			500	mA
	LATCH strobe pulse width	125			ns
	OUTEN strobe pulse width	125			ns
	Captured pulse width		100		ns
Power	Operating range		+4.75	+5.25	V
	Quiescent current	200			mA
Temperature	Operating range		0	70	C

Appendix B: Accessories

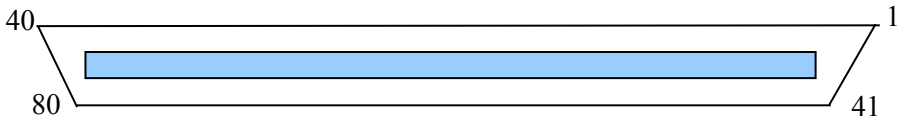
There are custom cables and breakout boards available for this product. Contact sales@sensoray.com or visit www.sensoray.com for further details.

The mating connector for the board's I/O connector is made by Fujitsu:

FCN-247R080-G/E	-connector
FCN-240C080-A/S	-connector cover

There are several other manufacturers that make similar connectors that will mate with the board's connector.

Appendix C: Connector Pinout



PIN	FUNCTION	PIN	FUNCTION
1	GND	41	I/O 37
2	I/O 0	42	I/O 38
3	I/O 1	43	I/O 39
4	I/O 2	44	I/O 40
5	I/O 3	45	I/O 41
6	I/O 4	46	I/O 42
7	I/O 5	47	I/O 43
8	I/O 6	48	I/O 44
9	I/O 7	49	I/O 45
10	I/O 8	50	I/O 46
11	I/O 9	51	I/O 47
12	I/O 10	52	GND
13	I/O 11	53	I/O 48
14	I/O 12	54	I/O 49
15	I/O 13	55	I/O 50
16	I/O 14	56	I/O 51
17	I/O 15	57	I/O 52
18	GND	58	I/O 53
19	I/O 16	59	I/O 54
20	I/O 17	60	I/O 55
21	I/O 18	61	I/O 56
22	I/O 19	62	I/O 57
23	I/O 20	63	I/O 58
24	I/O 21	64	I/O 59
25	I/O 22	65	I/O 60
26	I/O 23	66	I/O 61
27	I/O 24	67	I/O 62
28	I/O 25	68	I/O 63
29	I/O 26	69	GND
30	I/O 27	70	LATCH
31	I/O 28	71	I/O 64
32	I/O 29	72	I/O 65
33	I/O 30	73	I/O 66
34	I/O 31	74	I/O 67
35	GND	75	I/O 68
36	I/O 32	76	I/O 69
37	I/O 33	77	I/O 70
38	I/O 34	78	I/O 71
39	I/O 35	79	OUTEN
40	I/O 36	80	GND

Figure 4

APPENDIX D: Revision information

Date	Rev.	Revised by	Description of changes
10/07/99	E	Dennis Frost	Added sub device & vendor ID's.
11/23/16	F	Colette Roth	Added Model 720 text

APPENDIX E: Technical Support

For technical support contact Sensoray Company Inc.

Tel: (503) 684-8073
Fax: (503) 684-8164
e-mail: support@sensoray.com
WWW: www.sensoray.com

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP