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.

Find the *Advantech PCL-731* at our website: ***Click HERE***

# PCL-731
## 48-bit Digital I/O Card

**Copyright**
This documentation is copyrighted 1994 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice.
No part of this manual may be reproduced, copied, translated or transmitted in any form or by any means without the prior written permission ofAdvantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties which may result from its use.

**Ackn owledgments**
PC-LabCard is a trademark ofAdvantech Co., Ltd. IBM and PC are trademarks of International Business Machines *Corporation.* OPTO-22 is a trademark of OPTO-22 Corporation.

Part No. 2003731010 2nd Edition
Printed in Taiwan January 1995

# Contents

**Chapter 1 General information ..... 1**
Introduction ..... 2
Features ..... 2
Applications ..... 2
Specifications ..... 3
Connector pin assignments ..... 4
Block diagram ..... 5

**Chapter 2 Installation ..... 7**
Initial inspection ..... 8
Switch and jumper settings ..... 8

**Chapter 3 Operation ..... 13**
Configuration ..... 14
Interrupt handling ..... 17

**Chapter 4 Programming ..... 19**
Demo programs ..... 21

# Chapter 1

# General information

# Introduction

The PCL-731 is a 48-bit DIO card. It provides you with 48 bits of parallel digital input/output and emulates mode 0 of the 8255 PPI chip. Buffered inputs and outputs offer a high driving capacity.

The card's 48 bits are divided into six 8-bit I/O ports:
A0, B0, C0, Al, B1 and C1.You can configure each port as either input or output via software. The card also offers two hardware interrupt lines to the PC.

The card offers two 50-pin OPTO-22 compatible connectors which can connect to a PCLD-7216 SSR I/O module carrier board, PCLD-885 power relay output board, PCLD-785B and PCLD-7225 24/16-channel relay output boards or a PCLD-782B 24/16 channel opto-isolated D/I board.

# Features

- 48 TTL digital I/O lines.
- Emulates mode O of 8255 PPI
- Interrupt handling
- OPTO-22 compatible 50-pin connectors
- Output status readback

# Applications

- Industrial AC/DC I/O module monitoring and control
- Relay and switch monitoring and control
- Parallel data transfer
- Sensing the signals of TTL, DTL, CMOS logic
- Driving indicator LEDs

# Specifications

**I/O address assignments**

| Address | … | Function |
|---|---|---|
| BASE+0 | | PORT A0 |
| BASE+1 | | PORT B0 |
| BASE+2 | | PORT C0 |
| BASE+3 | | CFG REG |
| BASE+4 | | PORT A1 |
| BASE+5 | | PORT B1 |
| BASE+6 | | PORT C1 |
| BASE+7 | | CFG REG |

**Input signal**

- **Logic** high voltage: 2.0 to 5.25 V
- **Logic** low voltage: 0.0 to 0.80 V
- High level input current: 20 uA
- **Low** level input current: -0.2 mA

**Output signal**

- Logic high voltage: 2.4 V Min.
- **Logic** low voltage: 0.4 V Max.
- High level output current: -15 mA Max. (source)
- **Low** level output current: 24 mA Max. (sink)
- Driving capacity: 15 LS **TTL**

**Transfer rate**

- Typical: 300 Kbytes/sec
- Maximum: 500 Kbytes/sec

**Power consumption**

- **Typical:** 0.5 A @ 5 Vdc (+/-5%)
- **Connector:** Two OPTO-22 compatible 50-pin connectors (CN1 and CN2)

## Connector pin assignments

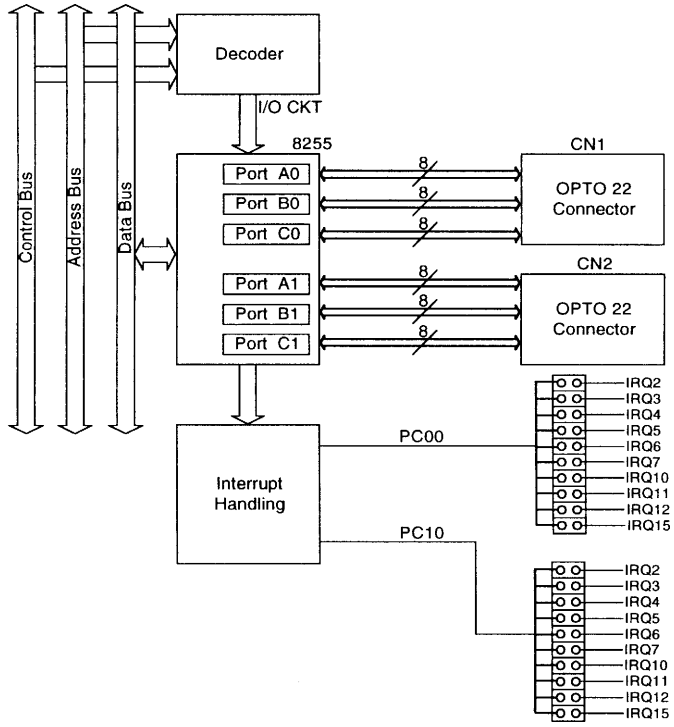
**CN1**

| | | | |
|---|---|---|---|
| PC 07 | 1 | 2 | GND |
| PC 06 | 3 | 4 | GND |
| PC 05 | 5 | 6 | GND |
| PC 04 | 7 | 8 | GND |
| PC 03 | 9 | 10 | GND |
| PC 02 | 11 | 12 | GND |
| PC 01 | 13 | 14 | GND |
| PC 00 | 15 | 16 | GND |
| PB 07 | 17 | 18 | GND |
| PB 06 | 19 | 20 | GND |
| PB 05 | 21 | 22 | GND |
| PB 04 | 23 | 24 | GND |
| PB 03 | 25 | 26 | GND |
| PB 02 | 27 | 28 | GND |
| PB 01 | 29 | 30 | GND |
| PB 00 | 31 | 32 | GND |
| PA 07 | 33 | 34 | GND |
| PA 06 | 35 | 36 | GND |
| PA 05 | 37 | 38 | GND |
| PA 04 | 39 | 40 | GND |
| PA 03 | 41 | 42 | GND |
| PA 02 | 43 | 44 | GND |
| PA 01 | 45 | 46 | GND |
| PA 00 | 47 | 48 | GND |
| +5 V | 49 | 50 | GND |

**CN2**

| | | | |
|---|---|---|---|
| PC 17 | 1 | 2 | GND |
| PC 16 | 3 | 4 | GND |
| PC 15 | 5 | 6 | GND |
| PC 14 | 7 | 8 | GND |
| PC 13 | 9 | 10 | GND |
| PC 12 | 11 | 12 | GND |
| PC 11 | 13 | 14 | GND |
| PC 10 | 15 | 16 | GND |
| PB 17 | 17 | 18 | GND |
| PB 16 | 19 | 20 | GND |
| PB 15 | 21 | 22 | GND |
| PB 14 | 23 | 24 | GND |
| PB 13 | 25 | 26 | GND |
| PB 12 | 27 | 28 | GND |
| PB 11 | 29 | 30 | GND |
| PB 10 | 31 | 32 | GND |
| PA 17 | 33 | 34 | GND |
| PA 16 | 35 | 36 | GND |
| PA 15 | 37 | 38 | GND |
| PA 14 | 39 | 40 | GND |
| PA 13 | 41 | 42 | GND |
| PA 12 | 43 | 44 | GND |
| PA 11 | 45 | 46 | GND |
| PA 10 | 47 | 48 | GND |
| +5 V | 49 | 50 | GND |

# Black diagram




**PCL-731 Block Diagram**

# Chapter 2

# Installation

## Initial inspection

Before you install the PCL-731, first check to make sure it was not damaged during shipping. If you find any damage or the card fails to meet specifications, please contact your local dealer or sales representative. Also contact the shipper and save the shipping materials for the shipper's inspection.

## Switch and jumper settings

We designed the PCL-731 with ease-of-use in mind. It has one function switch and six jumper settings. The following sections tell how to configure the card. You may want to refer to the figure below for help in identifying card components.

***Warning:*** *Do not install or remove the PCL-731 I/O card while the power is ON, as this may damage the card.*

The following diagram shows the location of switches and jumpers:

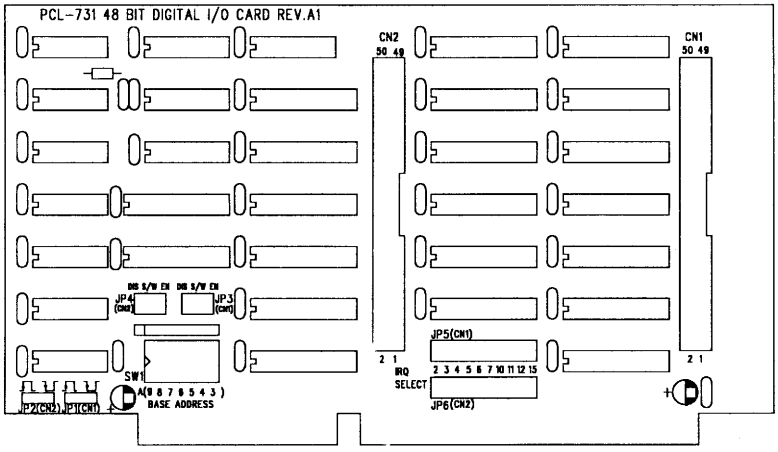



**Figure 2-1: Jumper and switch locations**

## Base address selection (SW1)

You control the PCL-731's operation by reading or writing data to the PC's I/O (input/output) port addresses. The card requires 8 consecutive address locations. Switch SW1 sets the card's base (beginning) address. Valid base addresses range from Hex 000 to Hex 3F8. Other devices in your system may, however, be using some of these addresses.

We set the PCL-73 1 for a base address of Hex 300 at the factory. If you need to adjust it to some other address range, set switch SW1 as shown below:

### Card I/O addresses (SW1)

| Range (hex) | Switch position | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 000 - 007 | On | On | On | On | On | On | On |
| 008 - 00F | On | On | On | On | On | On | Off |
| 200 - 207 | Off | On | On | On | On | On | On |
| 208 - 20F | Off | On | On | On | On | On | Off |
| *300 - 307 | Off | Off | On | On | On | On | On |
| 3F8 - 3FF | Off | Off | Off | Off | Off | Off | Off |

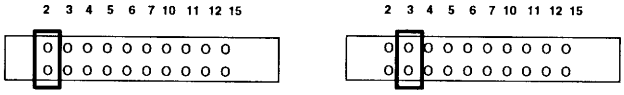
* = default

Switches 1-7 control the PC bus address lines as follows:

| **Switch** | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| **Line** | A9 | A8 | A7 | A6 | A5 | A4 | A3 |

## Interrupt settings (JP5, JP6)

Jumpers JP5 and JP6 control the IRQ levels, as shown below:



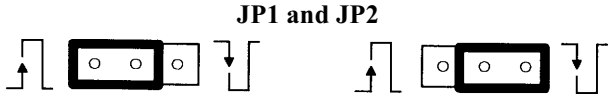

JP5 (PCOO) (Default)

JP6 (PC10) (Default)

***Note:*** *You must set each I/O line to a different interrupt level.*

## Interrupt level (JPI1, JP2)

Jumpers JPI and JP2 select the trigger edge (rising or falling) for I/O lines PC00 and PC10, respectively. Jumpers settings appear below:




Rising edge interrupt triggering (Default)

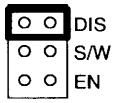
Falling edge interrupt triggering

If you set the PCL-73 1 to trap on a rising edge, the card will generate an interrupt if the I/O line (PC00 or PC10) changes from TTL LOW to TTL HIGH. Conversely, if you set the card to trap on a falling edge change, then it will generate an interrupt when the I/O line changes from TTL HIGH to TTL LOW.

## Interrupt mode (JP3, JP4)

The PCL-731 provides two I/O lines (PCOO and PCIO) which you can use to generate hardware interrupts to the PC. Jumper JP3 controls interrupt line PCOO, and jumper JP4 controls interrupt line PC10. The DIS setting for each jumper (shown in the figure below) disables the corresponding line's interrupt capability. The EN setting enables the line's interrupt capability.

JP3 and JP4




(Default)

The S/W setting allows you to enable and disable control by external or programmable interrupt. I/O line PC04 controls the interrupt on line PC00, and line PC14 controls the interrupt on line PC10. Bring line PC04 to TTL LOW to enable the interrupt capability on PC00. Send line PC04 to TTL HIGH to disable the interrupt capability on PC00. I/O line PC14 controls PC10 in the same way.

# Chapter 3

# Operation

The PCL-731 emulates MODE 0 of an Intel 8255 programmable peripheral interface (PPI) chip. The card is pin compatible with most industrial solid state I/O racks and modules, such as those manufactured by OPTO-22, Potter Brumfield, Gordos, etc.

The PCL-731's two 50-pin male IDC connectors interface with directly to OPTO-22 racks. The PCL-731 offers two I/O lines (PC00 and PC 10) to generate hardware interrupts, as described on page 10.

# Configuration

Mode 0 of the 8255 provides simple input/output functions. No handshaking is required since you read or write data directly to or from a specified port.

## 8255 MODE 0 function definitions

- Six 8-bit ports (Port A0, B0, C0, Al, B1 and C1)
- Any port can be used for input or output
- Outputs are latched, whereas inputs are not latched

The PCL-731 requires ten I/O ports, identified below:

I/O port assignments

| Location | Write | Read |
|---|---|---|
| BASE+0 | 8255 Port A0 | 8255 Port A0 |
| BASE+1 | 8255 Port B0 | 8255 Port B0 |
| BASE+2 | 8255 Port C0 | 8255 Port C0 |
| BASE+3 | 8255 Mode Register for Ports A0, B0 and C0 | N/A |
| BASE+4 | 8255 Port A1 | 8255 Port A1 |
| BASE+5 | 8255 Port B1 | 8255 Port B1 |
| BASE+6 | 8255 Port C1 | 8255 Port C1 |
| BASE+7 | 8255 mode Register for Ports A1, B1 and C1 | N/A |

## 8255 data registers

The PCL-731's I/O ports (BASE+0 to 2 and BASE+4 to 6) directly map to the 8255 ports. Bit assignments for each I/O port appear below:

BASE+0 8255 Port A0 (read/write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | PA07 | PA06 | PA05 | PA04 | PA03 | PA02 | PA01 | PA00 |

BASE+1 8255 Port B0 (read/write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | PB07 | PB06 | PB05 | PB04 | PB03 | PB02 | PB01 | PB00 |

BASE+2 8255 Port C0 (read/write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | PC07 | PC06 | PC05 | PC04 | PC03 | PC02 | PC01 | PC00 |

BASE+4 8255 Port A1 (read/write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | PA17 | PA16 | PA15 | PA14 | PA13 | PA12 | PA11 | PA10 |

BASE+5 8255 Port B1 (read/write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | PB17 | PB16 | PB15 | PB14 | PB13 | PB12 | PBI1 | PB10 |

BASE+6 8255 Port C1 (read/write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | PC17 | PC16 | PC15 | PC14 | PC13 | PC12 | PC11 | PG10 |

## 8255 made registers

BASE+3 8255 Mode Register, A0, B0, C0 (write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | 1 | 0 | 0 | PA0 | PC0H | 0 | PB0 | PC0L |

Where :

PA0: 0= Port A0 as output
1= Port A0 as input

PB0: 0 = Port B0 as output
1= Port B0 as input

PC0L: 0 = Port C0 low-nibble as output (PC03-PC00)
1= Port C0 low-nibble as input

PC0H: 0 = Port C0 high-nibble as output (PC07-PC04)
1 = Port C0 high-nibble as input

BASE + 7 8255 Mode Register, Al, B1, C1 (write)

| **Bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Value** | 1 | 0 | 0 | PA1 | PC1H | 0 | PB1 | PC1L |

Where :

PA1: 0 = Port Al as output
1= Port Al as input

PB1: 0 = Port Bl as output
1= Port Bl as input

PC1L:0 = Port C1 low-nibble as output (PC13-PCIO)
1= Port C1 low-nibble as input

PCIH:0 = Port C1 high-nibble as output (PC17-PC14)
1 = Port C1 high-nibble as input

*Note: After power-on or reset, all ports (Port A0, B0, C0, A1, B1 and C1) are set to input mode.*

## Interrupt handling

The PCL-731 offers two I/O lines, PC00 and PC10, which you can use to generate a hardware interrupt to the CPU. Interrupts are edge-triggered. Please refer to page 10 for jumper settings and description.

***Note:*** *Since the PCL-731's digital input data are not latched, the card provides no "first event" trapping to determine which input was active first.*

Although interrupts are normally triggered by external signals, the PCL-731 can send output data to emulate an interrupt signal. See the example programs in the following section.

# Chapter 4

# Programming

The following programming examples show how to use the card's readback function to monitor the output status, how to use the interrupt function (rising and falling edge) and how to set the initial value for the output port. The example programs run under Turbo C version 2.0 or later.

## Programming notes

You can program the PCL-731's ports in software for input, output or tri-state. When you power-on or reboot your system, however, all of the ports will be reset to input. When you configure one of the ports for output for the first time and sent data to it, it will not output until you have set the output buffer. This prevents external devices from being damaged before they are initialized. When a port is set for output, a read action on the port will return the data to be output.

## Demo programs

```
/*****************************************************
* This demo program shows how to use the PCL-731's *
*  readback function to monitor the output status. *
*                                                   *
*  Hardware setting:                                 *
*      1. Base address set at 0x300                  *
*****************************************************/
#include <stdio.h>
#include <conio.h>
#include <process.h>
#include <dos.h>

main()
{
int base = 0x300; /* set base address to 300 (hex)  */
int portA;       /" save readback value of port_A1 */
int portB;       /* save readback value of port_B1 */
int portC;       /* save readback value of port_C1 */

/* handle screen */
clrscr();
gotoxy(30,3);
textattr(0x70);
cputs("PCL-731 DEMO PROGRAM");
gotoxy(ll,6);
printf("PortA0 output value -> ");
gotoxy(ll,8);
printf("PortB0 output value -> ");
gotoxy(ll,l0);
printf("PortC0 output value -> ");
gotoxy(43,6);
printf("PortAl Readback -> ");
gotoxy(43,8);
printf("portB1 Readback -> ");
gotoxy(43,10);
printf("portC1 Readback -> ");
```

```
/* initialization */
outportb (base+3,0x80); /* set 8255 port0 all as
                           output */
for ( j=0;j<0x100;j++ )
{
  outportb (base,j); /* out j to port AO */
  gotoxy (34,6);
  printf("%2x",j);
  portA = inportb (base);
  gotoxy(63,6);
  printf("%2x",portA);
  if ( portA != j )
  {
       printf ("\7");  /* beep */
       gotoxy (30,13);
       textattr(0x09);
       cprintf("PortAl readback error!");
       getch();
       exit(l);  /* quit to dos */
  }
  outportb (base+l,j); /* out j to port BO */
  gotoxy (34,8);
  printf("%2x",j);
  portB = inportb (base+l);
  gotoxy(63,8);
  printf("%2x",portB);
  if ( portB != j )
  {
       printf ("\7");  /* beep */
       gotoxy (30,13);
       textattr(0x09);
       cprintf("PortB1 readback error!");
       getch();
       exit(l);  /* quit to dos */
  }
```

```
outportb (base+2,j); /* out j to port CO */
  gotoxy (34,10);
  printf("%2x",j);
  portC = inportb (base+2);
  gotoxy(63,1O);
  printf("%2x",portC);
  if ( portC != j )
  {
       printf ("\7");  /* beep */
       gotoxy (30,13);
       textattr(0xO9);
       cprintf("PortC1 readback error!");
       getch();
       exit(l);  /* quit to dos */
  }
) /* end of for */
) /* end of main()  */
```

```
/*****************************************************
*  This demo program shows how to use the interrupt *
*  function (rising edge) of the PCL-731.           *
*  Hardware settings:                               *
*     1. Base address set at OX300                  *
*     2. JP5 set at IRQ 2                           *
*     3. JP3 set at EN or S/W                       *
*     4. JP1 set at RISING trigger                  *
*****************************************************/
#include <dos.h>
#include <stdio.h>
#include <conio.h>

/* new INT OAH ISR */
void interrupt alarm (void)
{
  char Port61_Old_Status;
  char Port61_New_Status;
  int DelayTime = 0x300;
  int Count = 0;
  int i;

  /"*  get original port 61H status and save it  **/
  Port61_New_Status=Port6l_0ld_Status=inportb(0x61);

  /**  set port 61h bitl to O  **/
  Port61_New_Status &= Oxfd; /* 1111 1101 in binary */

  for (Count=O;Count<0x300;Count++,DelayTime--
  {
       Port61_New_Status ^= 0x02;  /* ON/OFF bitl */
       outportb (0x61, Port61_New_Status);
       for (i=O ; i<DelayTime ; i++);
  }
 outportb (0x61, Port6l_0ld_Status);
 outportb (0x20, 0x20);  /* send EOI to 8259 */
) /* end of ISR */
```

```
void main()
{
int IMR,base=0x300;
void interrupt (*Int_A_Old_Vector)();

Int_A_Old_Vector = getvect(0x0a);/*get old int Oah ISR*/
setvect (0x0a,alarm);  /* set new int Oah ISR */
IMR = inportb(Ox21); /*get 8259 interrupt mask register*/

/* initialization */
outportb (base+3,Ox80); /* set Port AO, BO, CO to */
                  /*  output mode */
clrscr();
printf("Press any key to generate a rising edge"
     "interrupt");
getch();
outportb(0x21,0xfb & IMR); /* set IRQ2 nonmasked */
/* PC04=O-->interrupt enable, set PCOO=O and then set
PCOO = 1 to generate a rising edge signal */
outportb (base+2,0xO);
printf("\n\nPort CO = %x",inportb(0x2c2));

/* generate a rising edge signal PCOO */
outportb (base+2,0xO1); /" set PC04 = O -> interrupt
enable, set PCOO = 1 */
printf("\n\nPort CO = %x",inportb~0x2c2));
printf("\n\nPress any key to quit...");
getch();

/* restore old INT OAH ISR "/
setvect (0x0a,Int_A_Old_Vector);
outportb (Ox21,IMR);  /* restore 8259 interrupt mask
"/

/* set ports as INPUT to release IRQ2 line */
outportb (base+3,Ox9b);
)  /* end of main()  */
```

**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