

ADR2200

RS232/RS485 RELAY I/O
INTERFACE

USER MANUAL

V 3.0

Caution: The **ADR2200** is a static sensitive device. Observe proper procedures for handling static sensitive devices.

ONTRAK CONTROL SYSTEMS INC.

764 Notre Dame Avenue

Unit # 1

Sudbury Ontario

CANADA P3A 2T2

(705) 671-2652 (VOICE)

(705) 671-6127 (FAX)

www.ontrak.net (WEB)

Ontrak Control Systems Inc. reserves the right to change product specifications to improve the product.

Although every attempt has been made to insure accuracy of information contained in this manual, **Ontrak Control Systems Inc.** assumes no liability for inadvertent errors.

Warranty: This ADR2200 is warranted from defects in workmanship and materials for a period of 90 days. Liability for defects is limited to the purchase price of the product. This warranty shall not apply to defects resulting from improper modifications or use outside published specifications.

Hyperterminal and Windows are trademarks of Microsoft Corporation.
APPLE , MACINTOSH and MAC are trademarks of Apple Computer Inc.
PC, XT, AT, PS/2 are trademarks of International Business Machines Inc.
COPYRIGHT 1999 ONTRAK CONTROL SYSTEMS INC.

TABLE OF CONTENTS

| | |
|--|----|
| READ ME FIRST | 3 |
| 1. Communication options. | |
| a) The ADR2200 RS232 Interface. | 4 |
| b) The ADR2200 RS485 Interface | 5 |
| 2. Powering the ADR2200 | 5 |
| 3. ADR2200 Command Set | 6 |
| a) Relay Output Commands | 7 |
| b) Digital Input Commands | 8 |
| c) Event Counter Commands | 8 |
| c) Interrupt Commands | 9 |
| e) ID Command | 9 |
| 4. Using BASIC with ADR Products | 10 |
| 5. Using TURBO C with ADR Products | 11 |
| 6. Daisy Chain Options for the ADR2200 | 13 |
| 7. Relay Output Connections | 14 |
| 8. Digital Input Connections | 15 |
| 9. Event Counter Connections | 15 |
| 10. Using Interrupt Functions | 16 |

APPENDIX

| | |
|-----------------------------|----|
| A-CONNECTION DIAGRAM | 18 |
| B-ELECTRICAL SPECIFICATIONS | 19 |
| C-MOUNTING DIMENSIONS | 20 |

READ ME FIRST

Thank you for purchasing this ADR2200 Relay I/O Interface. There are three steps to using the ADR2200.

- 1.Connecting your computer or terminal to the ADR2200.
- 2.Providing power to the ADR2200.
- 3.Sending commands to the ADR2200.

This manual will provide guidance for completing these steps along with BASIC and TURBO C programming tips. Additional applications and programming examples are available on our web page at <http://www.ontrak.net/>

FEATURES

- 8, N.O. contact outputs
- 4 digital inputs
- 16-bit contact or TTL input event counter
- interrupt capabilities on all inputs to eliminate polling
- on-board RS232 to RS485 converter
- daisy-chainable up to 10 boards
- daisy-chainable power supply
- power-up via standard wall adapter (optional)
- simple yet versatile commands
- easy to use with Visual BASIC or TURBO C programs
- compatible with all ADR2000 series interfaces

1a)THE ADR2200 RS232 INTERFACE

The ADR2200 communicates via a standard RS232 port utilizing a simple three-wire interface. The only signals used are received data (RC), transmitted data (TX) and ground (GND). Most RS232 ports use hardware handshaking (i.e. DTR, DSR, CTS, RTS) signals to control the flow of data on the port. For this reason the cable required to connect to the ADR2200 must have jumpers on the DB25 end to satisfy these handshaking requirements. IBM or compatible computers may be used as a host computer with the supplied cable. The supplied cable has the following connections;

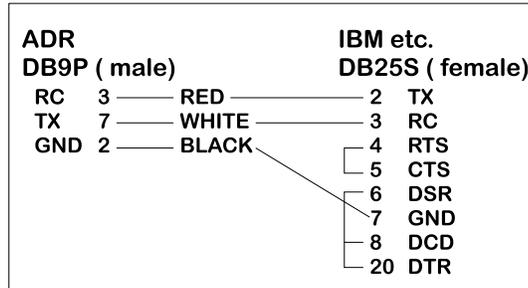


Figure 1: Supplied Cable Wiring Diagram

If the host computer has a 9-pin serial port connector, a 9-pin to 25-pin adapter cable will be required to connect to the ADR2200 cable. This adaptor is available at most computer dealers. If desired, the DB25 connector on the supplied cable may be removed and a female DB9S connector can be soldered in its place using the following wiring diagram;

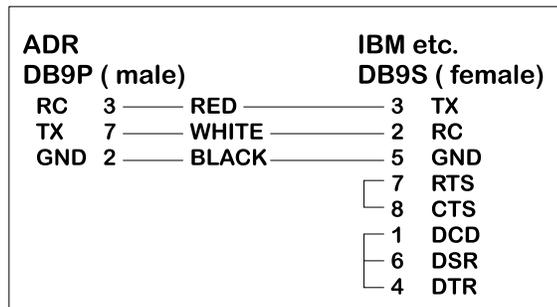


Figure 2: Modified Wiring Diagram For 9-PIN SERIAL PORTS

If the host computer has a female DB25 connector, a male-to-male adapter is required to use the supplied cable. This may be purchased at most computer dealers. Apple Macintosh computers may be connected to the ADR2200 using MAC to DB25 DTE conversion cable. Once connected to the RS232 based host computer or terminal, the RS232 port should be configured to the following specifications to allow communication with the ADR2200.

9600 baud - 8 bit words - 1 stop bit - no parity

If using BASIC or C consult the appropriate section in this manual for details on how to configure your serial port. If a terminal or terminal emulation program is used, configure your terminal to the above specifications using the operations manual for your terminal equipment or terminal emulation program.

1b)THE ADR2200 RS485 INTERFACE

The ADR2200 RS485 interface is a two-wire connection meeting all the standards of the EIA RS485 interface specifications. The supplied cable is **NOT** an RS485 cable. To communicate via RS485 the host computer must have an RS485 port and be connected directly with two wires (TR+ and TR-). A typical connection diagram is shown in figure 2.

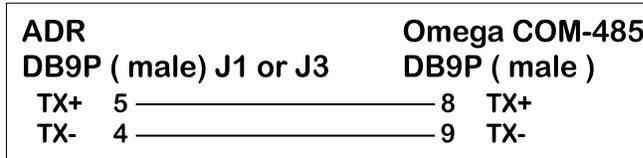


Figure 2 : Typical RS485 Connection

Note that both J1 and J3 are RS485 compatible ports. Connection from the host to the ADR2200 should be made using J1 and then J3 is used to enable daisy chaining additional ADR2000 series products.

The host RS485 port should be configured with the following specifications to enable communications to the ADR2200, 9600 Baud - 8 bit words - 1 stop bit - no parity. Line feeds should **NOT** be sent after commands as they may collide with data being returned from the ADR2200.

2.PROVIDING POWER TO THE ADR2200

The ADR2200 may be powered using a regulated 5 volt power supply or a suitable wall adaptor. Power to daisy chained ADR2200 may also be supplied via the daisy chain cable. See the Daisy chaining section of this manual for further information.

POWER-UP USING A 5 VOLT REGULATED SUPPLY

If the ADR2200 is to be powered using a regulated 5 volt power supply, the 5VDC and GND connections are to be made to the ADR2200 via the main terminal block TB1. The supply must be able to provide a minimum of 300mA. **Care must be taken to avoid improper power supply connection as permanent damage to the ADR2200 may result if connected improperly.** No connection to J2 is to be made if the ADR2200 is powered by a regulated 5 volt supply.

POWER-UP USING A WALL ADAPTOR

The ADR2200 has an on-board 5 volt regulator allowing the use of a 9-volt wall adaptor to power the internal circuits. The regulator should be able to provide from 300-500mA .(MODE 68-950-1) The regulator must have a standard 2.1mm, center negative, coaxial connector. The connector can then be inserted into J2 on the ADR2200. When the ADR2200 is powered by a wall adaptor, the on-board regulator also may provide a regulated 5 volts DC out to provide power to external circuits. This 5 volt supply is available on TB1. The amount of current available depends on the number of relays to be energized at one time.. **For safe operation no more than 50mA should be drawn from the power terminals to power external circuits.**

RELAY OUTPUT COMMAND SUMMARY

| | |
|------------|---|
| SKn | Sets (closes contact) Relay specified by n (n = 0 to 7) |
| RKn | Resets (opens contact) Relay specified by n (n = 0 to 7) |
| SPKxxxxxxx | Output binary data to Relay PORT K. (x=1 or 0) |
| MKddd | Outputs decimal data (ddd) to Relay PORT K. (ddd= 0 to 255) |
| RPK | Returns status of all Relays in PORT K in binary format. |
| RPKn | Returns status of Relay specified by n in PORT K (n = 0 to 7) |
| PK | Returns status of Relay PORT K in decimal format. |

DIGITAL INPUT COMMAND SUMMARY

| | |
|-------|---|
| RPA | Returns status of all I/O lines in PORT A in binary format. |
| RPA n | Returns status of I/O line specified by n. (n= 0 to 3) |
| PA | Returns status of PORT A in decimal format. |

EVENT COUNTER COMMAND SUMMARY

| | |
|-----|--|
| CE | Clear event counter. |
| RE | Returns present count of event counter. |
| REC | Returns present count of event counter and clears event counter. |

INTERRUPT COMMAND SUMMARY

| | |
|---------|--|
| IE | Enable Interrupts. |
| ID | Disable Interrupts |
| IS | Returns Interrupt Status (1 if enabled, 0 if disabled) |
| TLnnnnn | Loads Event Counter Trigger (nnnnn=0 to 65535) |
| TS | Returns Event Counter Trigger Value |

ID COMMAND

| | |
|-------|--|
| *IDN? | Returns 4 digit product identifier code. |
|-------|--|

3. ADR2200 COMMANDS

a) Relay Output Commands

There are 8 relays outputs on the ADR2200 labeled K0-K7. They can be controlled individually or as a port (PORT K). The following commands allow relays to be individually set or reset, or the whole port to be configured with a single command. Three commands also allow the reading of the relay status and are provided for applications where confirmation of commands is required.

SKn Sets (closes contact) of relay specified by n in port K. (n = 0 - 7)

example; SK3<cr>

(Relay K3 is energized and its contacts close)

RKn Resets (opens contact) of relay specified by n in port K. (n = 0 - 7)

example; RK0<cr>

(Relay K0 is de-energized and its contacts open)

SPKxxxxxxxx Outputs binary data to PORT K. All eight bits must be specified.
Order is MSB-LSB. (x=1 to set relay or 0 to reset relay)

example; SPA10101000<CR>

(Relays K7,K5 and K3 are energized (contacts close) and relays K6,K4,K2,K1 and K0 are de-energized (contacts open))

Mkddd Outputs decimal data (ddd) to PORT K. Relays are set or reset depending on binary equivalent of ddd (ddd= 000 to 255)

example; MK255<CR>

(All Relays are energized (contacts close))

NOTE: The following three commands allow reading of the status of the relays and are provided for use if verification of commands is desired.

RPK Returns status of all Relays in binary format. Order is MSB-LSB.

example; RPK<CR>

0 1 1 1 0 0 1 0

(K7, K3, K2, K0 are de-energized, K6, K5 ,K4, K1 are energized)

RPKn Returns status of relay specified by n.(n=0 to 7)

example; RPK4<CR>
1
(K4 is energized (contact closed))

PK Returns status of all relays in decimal format.

example; PK<CR>
128
(K7 is energized, PK6 thru PK0 are de-energized)

b) Digital Input Commands

There are four digital inputs on the ADR2200 labeled PA0 to PA3. (PORT A) There are three commands provided to read the status of the inputs in various formats.

RPA Returns status of all input lines in PORT A in binary format. Order is MSB-LSB.

example; RPA<CR>
0 1 1 1

(PA2, PA1, PA0, are high, PA3 is low)

RPA_n Returns status of input line in PORT A specified by n.(n=0 to 3)

example; RPA2<CR>
1
(PA2 is high)

PA Returns status of PORT A in decimal format.

example; PA<CR>
15
(All inputs are high)

c) Event Counter Commands

The ADR2200 is equipped with a 16-bit event counter that accepts TTL or contact input. There are three commands available to read, and clear the event counter. If the maximum count of 65535 is reached the counter will rollover to 00000 .

RE Returns decimal value of event counter

example; RE<CR>
00456
(Present count is 456.)

CE Clears event counter

example; CE<CR>
 (Event counter is cleared to 00000)

REC Reads and clears event counter

example; REC<CR>
 12034
 (Count is returned (12034) and counter is reset to 00000.)

e) Interrupt Commands

The ADR2200 has four digital input lines (PA0,PA1,PA2,PA3) that can be used to provide an interrupt to the host when an input is pulled low. Additionally, a sixteen bit trigger register can be loaded to provide an interrupt when the event counter reaches a specific count. All inputs have built in pull-up resistors tied to the 5 volt supply. When interrupts are enabled, bringing any line low or when the event counter reaches the trigger value, a two digit value is returned to the host. The first digit is the board address (0 - 9) and the second identifies the source of the interrupt (1 for PA0, 2 for PA1, 3 for PA2, 4 for PA3 and 5 for an event counter match. For example;

- an interrupt on PA0 on board 0 returns
01
- an event counter match on board 3 returns
35

Interrupts generated at the same instant will be returned with highest priority given to PA0, followed by PA1,PA2,PA3 and lastly, an event counter match. All interrupts are disabled on power up. See the applications section titled Using Interrupt Functions for further details.

The Interrupt commands are;

IE Enables all interrupts
ID Disables all interrupts.
IS Returns status of interrupts (0 if disabled, 1 if enabled)

TInnnnn Loads event counter trigger value (nnnnn = 0 to 65536)

example; TL10500<CR>
 (Event counter trigger is set to 10500)

TS Returns event counter trigger value.

Example TS<CR>
 10500
 (Trigger is currently set at 10500)

e) ID Command

*IDN? _____ Returns ID code (2200)
 _____ * may be omitted

4. Sending Commands in BASIC to the ADR2200

OPENING A SERIAL FILE

Commands may be sent to the ADR2200 using a terminal emulation program such as Hyperterminal by simply typing commands and pressing <cr>. With BASIC, the ADR2200 is connected to the computer via a serial cable and BASIC treats the ADR2200 as a serial file. Before commands can be sent to the ADR2200 this serial file must be opened and initialized. This should be done at the start of any program that is to access the ADR2200. The command to open a serial file is shown below;

```
10 OPEN "COM1:9600,n,8,1,CS,DS,RS" AS#1
```

This line opens a serial file and labels it as serial file #1. This allows access to the ADR2200 using PRINT#1 and INPUT#1 commands.

SENDING COMMANDS

Sending commands in BASIC to the ADR2200 can be done using PRINT#1 commands. For example, sending an RK0 command could be done as shown below;

```
20 PRINT#1, "RK0"
```

Extra spaces inside the quotes are ignored by the ADR2200. Avoid sending commands on consecutive lines because a <CR> is not sent after the first command resulting in an unrecognized command. A REM statement should be inserted between consecutive PRINT#1 commands as shown below;

```
20 PRINT#1, "SK0"  
30 REM FORCES <CR>  
40 PRINT#1, "RK6"
```

Variable names may also be used with PRINT#1 commands. One example of this shown below. This program turns on each relay in sequence with a slight delay determined by LINE 40.

```
10 OPEN "COM1:9600,n,8,1,CS,DS,RS" AS#1  
20 FOR X = 0 to 7  
30 PRINT#1, "SK",X  
40 FOR D= 1 to 100000 : NEXT D  
50 NEXT X  
60 END
```

RECEIVING DATA

When reading the digital port, event counter or relay status, data is sent from the ADR2200 to the host computers serial buffer. This data can be retrieved using INPUT#1 commands. The INPUT#1 command should be used following PRINT#1 commands if data is expected to be sent by the ADR2200. If a single piece of data is expected then one variable name should be used with the INPUT#1 command. If eight pieces of data are to be received as with the RPK command then eight variable names must be used with the INPUT#1 command. Examples of both cases are shown below;

```
20 PRINT#1, "RPA0"  
30 INPUT#1, PA0DAT  
40 PRINT#1, "RPK"  
50 INPUT#1, K7STAT,K6STAT,K5STAT,K4STAT,K3STAT,K2STAT,K1STAT,K0STAT
```

The variable names used in the INPUT#1 commands now contain the data sent by the ADR2200
The data can now be printed, displayed, saved or whatever is required by the application.

Visit our web page at www.ontrak.net for additional programming examples in BASIC, Visual Basic and C.

5) Sending Commands in TURBO C to the ADR2200

This section will demonstrate how to send and receive data from the ADR2200 using TURBO C. It outlines the commands used to, configure the serial port (bioscom), send data out through the serial port (fprintf), and receive data through the serial port (fscanf).

Commands used in TURBO C to access the ADR2200 require the following include files to be declared at the start of TURBO C programs;

```
#include <stdio.h>  
#include <bios.h>
```

CONFIGURING THE SERIAL PORT

The first step in accessing the ADR2200 via the serial port is configuring the serial port to the proper communication parameters which are, 9600 baud, 8 bit words, no parity. This is done using the "bioscom" command. The syntax for this command is;

```
bioscom (0,settings,com1);
```

where settings is previously defined as HEX E3 and com1 is defined as 0. Defining "settings" and "com1" should be done using;

```
#define com1 0  
#define settings (0xE3)
```

These statements should be placed immediately following your include files. The bioscom command needs only to be executed once before the ADR2200 is accessed.

SENDING COMMANDS TO THE ADR2200

To send commands to the ADR2200 the "fprintf" command is used. For example, the following command sends an SK0 (Set relay K0) command to the ADR2200;

```
fprintf (stdaux,"SK0 \xD");
```

The \xD suffix sends a carriage return after the command which is needed by the ADR2200 to recognize a command. Integer variables may also be used in the command line. For example, the following command sends a MKddd (make port K=ddd) command, where DOUT is a previously defined integer value of 0 to 255.

```
fprintf (stdaux,"MK %d \xD",DOUT);
```

RECEIVING DATA FROM THE ADR2200

If a command sent to the ADR2200 is a responsive command, that is, one that results in data being sent back to the host, the data is retrieved using the "fscanf" command. After this command is used the serial buffer must be re-initialized using the "rewind" command. The syntax for this command is;

```
rewind (stdaux);
```

This command is executed after data is retrieved using the "fscanf" command. For example, the following commands send a RPA0 command and stores the retrieved data in an integer variable named PA0;

```
fprintf (stdaux,"RPA0 \xD");  
fscanf (stdaux,"%D",&PA0);  
rewind (stdaux);
```

In this example, the command PA (read port A)is sent to the ADR2200 and the retrieved data is stored in an integer variable named PORTA;

```
fprintf (stdaux,"PA \xD");  
fscanf (stdaux,"%D",&PORTA);  
rewind (stdaux);
```

Visit our web page at www.ontrak.net for additional programming examples in BASIC, Visual Basic, C, Labview and TestPoint.

6. Daisy Chain Options for the ADR2200

Daisy chaining ADR2200 series boards involves three steps.

- A. Setting Address Jumpers
- B. Physically Connecting Boards
- C. Sending commands

A. Setting Address Jumpers

The ADR2200 can be daisy-chained, regardless of the type of serial interface provided by the host computer. Each board on the chain must be assigned an address via the BCD address jumper block on the ADR2200. Up to ten boards may be daisy-chained. The following table shows how to jumper the address jumper block to select a board address.

| Position 8 | Position 4 | Position 2 | Position 1 | Address |
|------------|------------|------------|------------|---------|
| | | | | |
| OPEN | OPEN | OPEN | OPEN | 0 |
| OPEN | OPEN | OPEN | JUMP | 1 |
| OPEN | OPEN | JUMP | OPEN | 2 |
| OPEN | OPEN | JUMP | JUMP | 3 |
| OPEN | JUMP | OPEN | OPEN | 4 |
| OPEN | JUMP | OPEN | JUMP | 5 |
| OPEN | JUMP | JUMP | OPEN | 6 |
| OPEN | JUMP | JUMP | JUMP | 7 |
| JUMP | OPEN | OPEN | OPEN | 8 |
| JUMP | OPEN | OPEN | JUMP | 9 |

Table 1. Address Jumper Settings.

B. Physically Connecting Boards

The ADR2200 series interface boards have two DB9 connectors that allow daisy chaining. The data format used in daisy chaining is RS485 regardless of the host communication type. To connect boards on a chain, a daisy chain cable must be constructed. The cable must provide two connections for the RS485 signals. A typical daisy-chain cable is shown in Figure 5a)

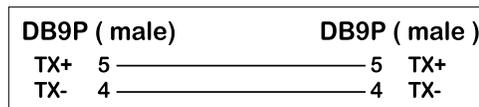


Figure 5a) Daisy-chain cable

Power may be shared in daisy-chained ADR2200 series interfaces if two extra conductors are added to the daisy-chain cable. Care should be taken that the output current limitation on the power supply is not exceeded. The connections for a powered daisy-chain cable are shown in Figure 5B) **NOTE: Power sharing is available only if power is applied via J2 (7-15VDC)**.

| DB9P (male) | | DB9P (male) | |
|--------------|---|---------------|-------|
| TX+ | 5 | 5 | TX+ |
| TX- | 4 | 4 | TX- |
| +9VDC | 6 | 6 | +9VDC |
| GND | 1 | 1 | GND |

Figure 5b) Powered Daisy-Chain Cable

The Daisy-chain cable can be connected from J3 to either J1 or J3 on additional ADR2200 interfaces. Both J1 and J3 have identical pinouts for RS485 and power signals used for daisy-chain applications. Figure 5c) shows a typical daisy-chain application. If a Powered daisy-chain cable is used, power need only be connected to J2 on any one ADR2200 in the chain.

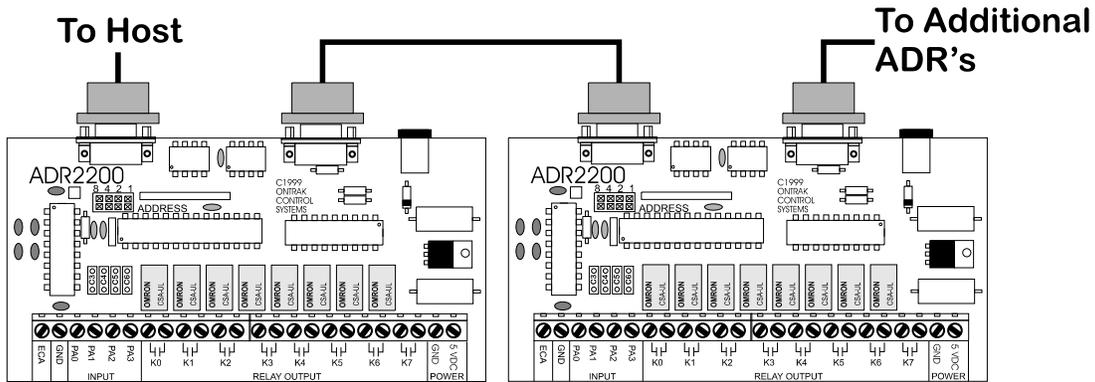


Figure 5c) Typical Daisy-Chain Application

C Sending Commands

Once a board is jumpered, it will respond only to commands preceded by its address as a single digit integer number. For example to read PA0 on board 3 the command "3RPA0" is sent. To set K4 on board 7 the command "7SK4" is sent. Spaces sent between the board address and commands are ignored. Board zero will respond to both commands with no preceding address and commands preceded with a zero for reasons of continuity. Never connect two boards with the same address on the same chain. This will result in both boards responding at the same time and will cause contention on the network with possible damage to the ADR boards.

7. Relay Output Connections

There are eight N.O. relay outputs on the ADR2200 labeled K0 thru K7. The relays are capable of switching up to 1Amp@30VDC or 0.5Amp @ 120VAC. Connections of AC or DC loads are made to TB1.

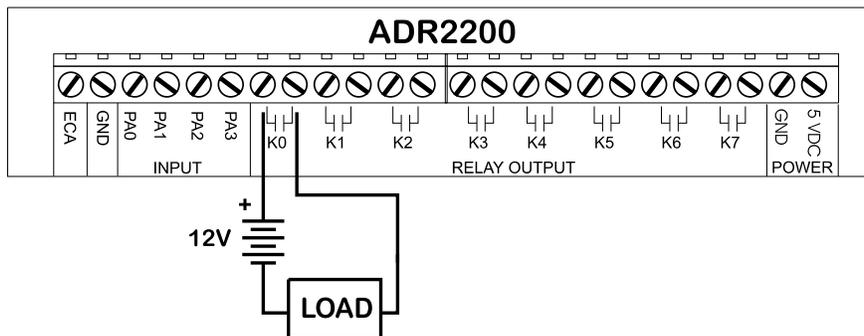


Figure 7 : Typical Relay Contact Connection

8. Digital Input Connections

The four digital inputs on the ADR2200 have weak pull-up resistors built-in allowing direct connection of dry contact inputs from any source including switches and relays. TTL signal sources including TTL or NPN type proximity sensors can also be directly connected to PA0 thru PA3.

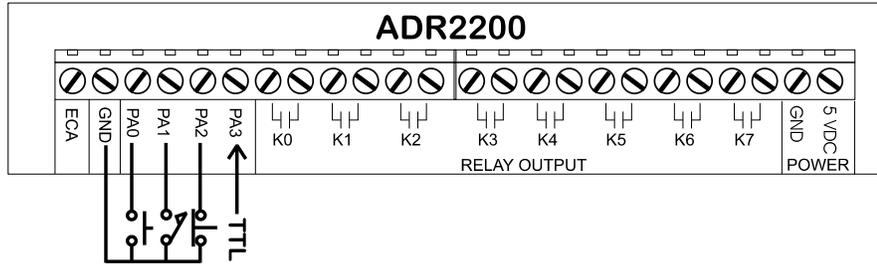


Figure 8 : Typical Digital Input Connections

9. Event Counter Connections

The event counter can be configured to accept TTL or dry contact inputs. TTL type inputs may be connected directly to the ECA input while dry contacts must be connected between GND and the ECA input. ECA is connected to the 5VDC supply via a 15K ohm pull-up resistor. **NOTE : C7 is a 0.1uF debounce capacitor for use with dry contact inputs. This capacitor must be removed if the input signal is a TTL type.** Events are counted on the rising edge of the input which is when the TTL signal switches from low to high or the dry contact opens.

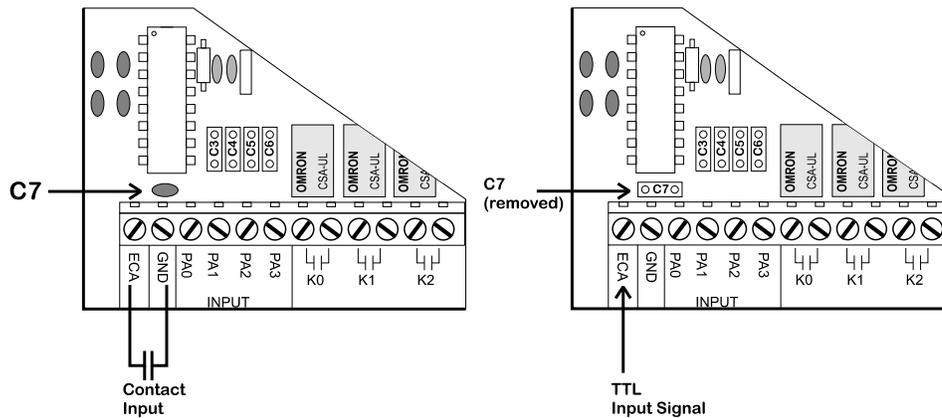


Figure 9 : Typical Event Counter Connections.

Note: The ADR2200 is shipped with C7 installed.

The following sample BASIC program reads the count input and displays the count result continuously on the video screen

```
10 OPEN"COM1:9600,N,8,1,CS,DS,RS" AS#1           ;open com port
20 CLS                                           ;clear screen
30 LOCATE 1,1                                    ;locate cursor
40 PRINT#1, "CE"                                ;clears event counter
50 PRINT "PRESENT COUNT IS",COUNT              ;displays count data
60 PRINT#1, "RE"                                ;sends RE command to ADR2200
70 INPUT#1, COUNT                               ;retrieves count data
80 GOTO 50                                       ;repeat procedure
```

10.Using Interrupt Functions

The ADR2200 has four digital input lines (PA0,PA1,PA2,PA3) that can be used to provide an interrupt to the host when an input is pulled low. Additionally, a sixteen bit trigger register can be loaded to provide an interrupt when the event counter reaches a specific count. All inputs have built in pull-up resistors tied to the 5 volt supply. When interrupts are enabled, bringing any line low or when the event counter reaches the trigger value, a two digit value is returned to the host. The first digit is the board address (0 - 9) and the second identifies the source of the interrupt (1 for PA0, 2 for PA1, 3 for PA2, 4 for PA3 and 5 for an event counter match. For example;

- an interrupt on PA0 on board 0 returns
01
- an event counter match on board 3 returns
35

Interrupts generated at the same instant will be returned with highest priority given to PA0, followed by PA1,PA2,PA3 and lastly, an event counter match. All interrupts are disabled on power up.

The Interrupt commands are;

| | |
|----------------|--|
| IE | Enables all interrupts |
| ID | Disables all interrupts. |
| IS | Returns status of interrupts (0 if disabled, 1 if enabled) |
| TLnnnnn | Loads event counter trigger value (nnnnn = 0 to 65536) |
| TS | Returns event counter trigger value. |

Notes To Operation.

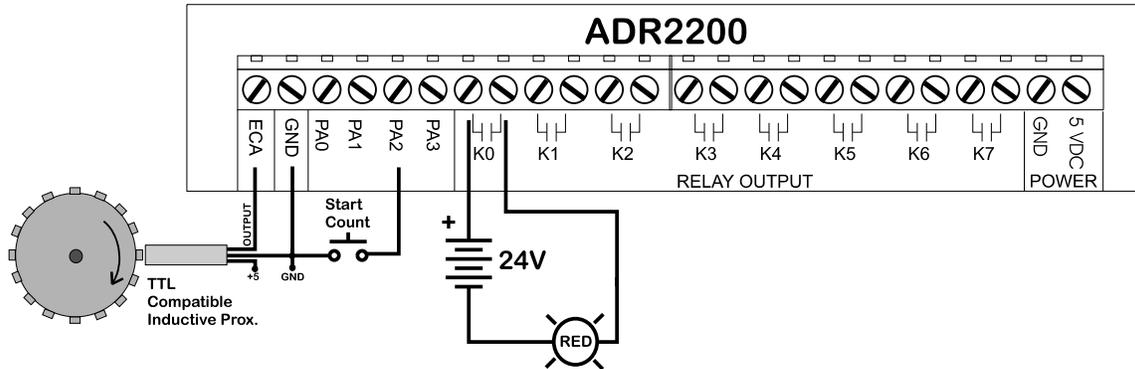
1. The **IE** command is used to enable all interrupts, however, the event counter match interrupt is not enabled unless a trigger value other than 0 is loaded into the trigger register using the **TLnnnnn** command. The trigger register is loaded with 0 on power-up. To disable the event counter match interrupt at any time, load the trigger register with 0 using the **TLnnnnn** command.
2. The **IS** (interrupt status) command should be used following an **ID** (interrupt disable) command to verify interrupts have been disabled. This may be required in cases where there is a possibility of an interrupt being generated when the **ID** command is issued. The primary communication used by ADR2000 series interfaces is Half-Duplex RS485 and interrupt data may collide with the **ID** command resulting in the ID command not being received by the ADR2200.
3. Once an interrupt is generated and data is sent to the host, no further interrupts will be generated by that particular input unless the IE command is sent. When interrupt data is sent to the host, that input is masked and the issuing the command IE is the only method

to un-mask the input.

The following example shows use of the interrupts using Visual BASIC.

The application is a counter application which is initiated when an operator activates the " Start Count " button which is connected to PA2. At this point the event counter counts the pulses from the TTL compatible proximity detector and energizes K0 when a count of 160 is reached.

Private Sub Start_Click()
With MSComm1



Rem Ensure Relay K0 is off.

.Output = "RK0" + Chr (13)

Rem Enable interrupts.

.Output = "IE" + Chr (13)

Rem Wait for interrupt from switch.

Do

Dummy = DoEvents()

Loop Until MSComm.InBufferCount >= 3

Rem Display interrupt in text window.

Text1.Text = .input

Rem Clear event counter

.Output = "CE" + Chr (13)

Rem Load trigger register with 160 (enables counter interrupt)

.Output = "TL160" + Chr (13)

Rem Wait for interrupt from event counter.

Do

Dummy = DoEvents()

Loop Until MSComm.InBufferCount >= 3

Rem Display interrupt in text window.

Text1.Text = .input

Rem Energize relay K0 (Turns on RED lamp.)

.Output = "SK0" + Chr (13)

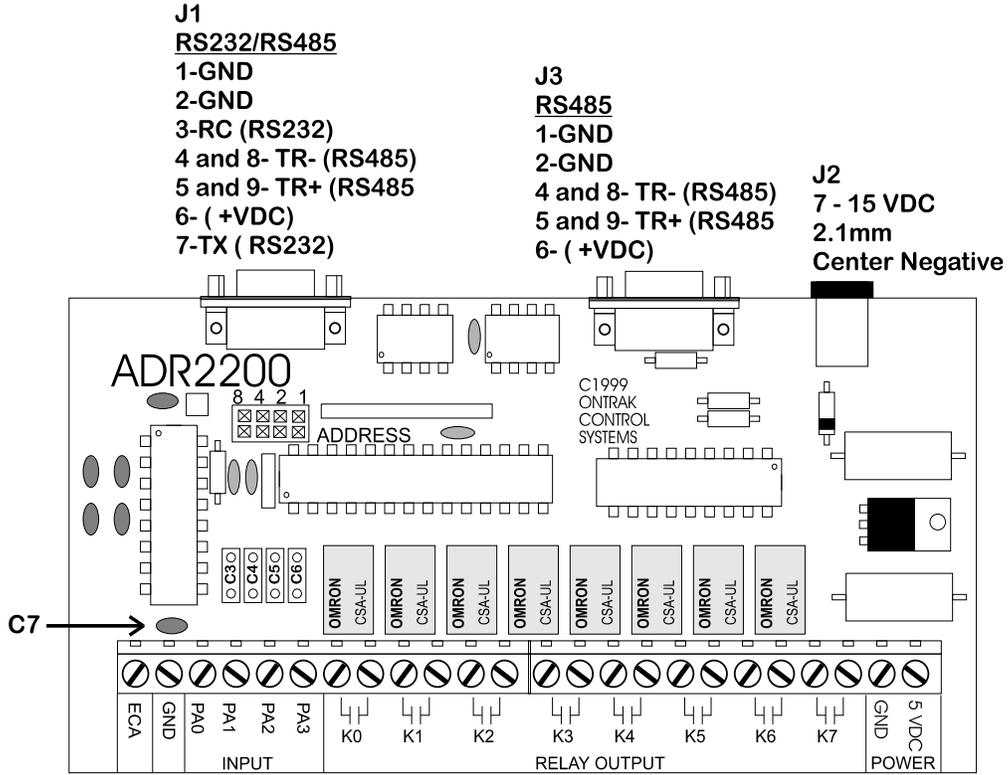
End With

End Sub

Notes on operation of program.

1. Waiting for interrupts requires a loop until 3 characters (Interrupt Data + CR) are received by the serial input buffer.
2. Interrupt data is retrieved to empty the input buffer and displayed in a text window for troubleshooting purposes.
3. Interrupt data can be read as a variable and tested to determine source of interrupt. In this example it was not required as only one interrupt was possible at one time. The counter interrupt (05) was not enabled until the switch interrupt (03) was received.

APPENDIX A CONNECTION DIAGRAM



APPENDIX B ELECTRICAL SPECIFICATIONS

ADR2200

| | |
|--|------------------------------|
| Supply Voltage | 5VDC+/- 10% or 7-15VDC |
| Supply Current (Relays De-energized) | 40mA Typical, 50mA Maximum |
| Supply Current (All Relays Energized) | 250mA Typical, 300mA Maximum |
| Operating Temperature | 0-50C |

Contact Outputs (8)

| | |
|-----------------|---|
| Type | SPST (N.O.) |
| AC Rating | 0.5 Amp @ 120 VAC Max. |
| DC Rating | 1.0Amp@30VDC Max. |
| Approvals | UL,CSA |
| Mechanical Life | 5,000,000 Operation Minimum |
| Electrical | 100,000 Operations Minimum at full-load |

Digital Inputs (4)

| | |
|--------------------|------------------------------------|
| Type | TTL or Contact (weak pull-up) |
| Input Voltage High | 4.00V minimum |
| Input Voltage Low | 0.8V maximum |

Event Counter (1)

| | |
|------------|----------------|
| Type | TTL or Contact |
| Resolution | 16 bits |

Communication Interface

RS232 and RS485
9600 baud, 8 bit words, no parity, 1 start bit
Daisy-chain via RS485

Visit our web site at <http://www.ontrak.net/> for additional applications and programming examples.

APPENDIX C MOUNTING DIMENSIONS

