ADR101

ANALOG/DIGITAL/RS232
INTERFACE

USER MANUAL

V 3.0

Caution: The **ADR101** 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)**

Warranty: This ADR101 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.

TABLE OF CONTENTS

READ ME FIRST

Thank you for purchasing this ADR101, serial data acquisition and control interface. There are three steps to using the ADR101.

1.Connecting a computer or terminal to the ADR101.

2.Providing power to the ADR101.

3.Sending commands to the ADR101.

This manual will provide guidance for completing these steps along with BASIC and TURBO C programming tips. An applications section is also provided to describe how to interface various electronic transducers and other devices to the ADR101. Additional applications and programming examples are available on our web page at http://www.ontrak.net/

FEATURES

-2, 8-bit analog inputs (0 -5 VDC)
-8 digital I/O lines individually programmable as input or output
-High current digital I/O lines (sink 20mA/source 20mA)
-Three wire RS232 interface
-Low power requirements (5 volts at 15mA)
-Power-up via standard wall adapter (optional)
-Simple yet versatile commands
-Easy to use with Visual BASIC or TURBO C programs
-Compatible with all ADR series interfaces

1a)THE ADR101 RS232 INTERFACE

The ADR101 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 ADR101 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;

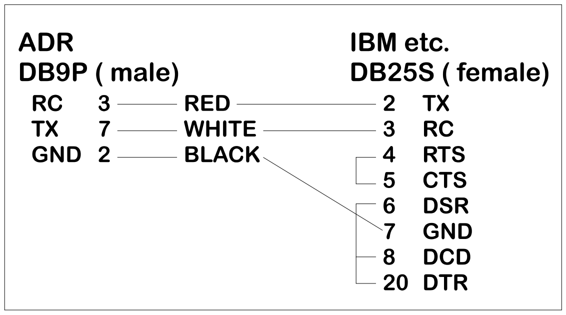


Figure 1: Supplied Cable Wiring Diagram

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 ADR101 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;

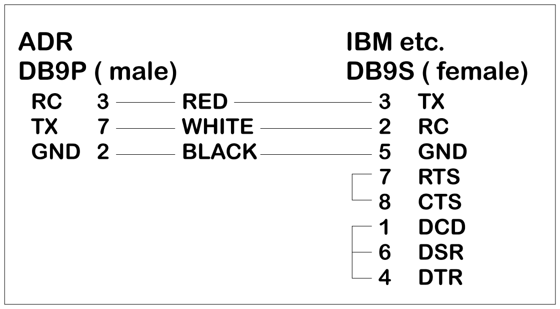


Figure 2 : Modified Wiring Diagram For 9-PIN SERIAL PORTS

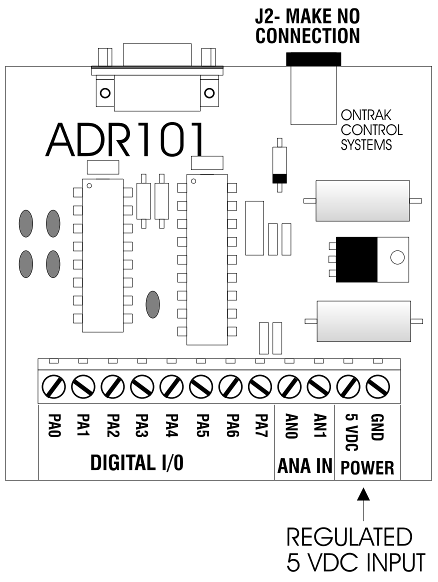
If the host computer has a female DB25 connector, a male-to-male adapter is required to use the supplied cable. This may be purchased at most computer dealers. Apple Macintosh computers may be connected to the ADR101 using MAC to DB25 DTE conversion cable.
Once connected to the RS232 based host computer or terminal, the RS232 port should be configured to the following specifications to allow communication with the ADR101.
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.

2.PROVIDING POWER TO THE ADR101

The ADR101 may be powered using a regulated 5 volt power supply **or** a suitable wall adaptor.

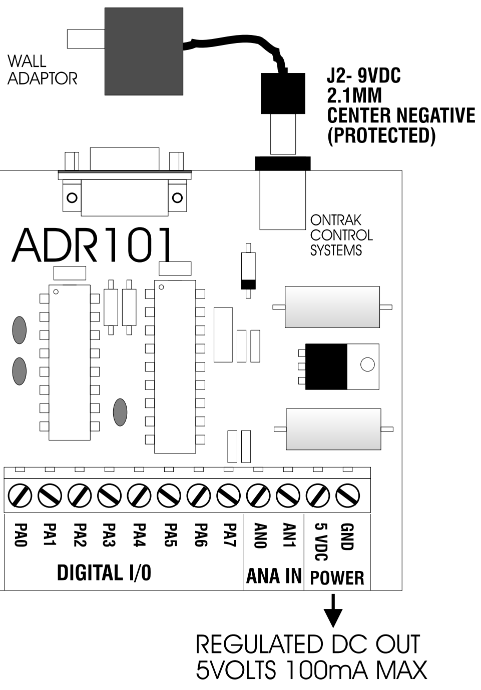
POWER-UP USING A 5 VOLT REGULATED SUPPLY

If the ADR101 is to be powered using a regulated 5 volt power supply, the 5VDC and GND connections are to be made to the ADR101 via the main terminal block TB1. The supply must be able to provide a minimum of 30 mA and up to 130mA if the ADR101 is to source current from the digital outputs. **Care must be taken to avoid improper power supply connection as permanent damage to the ADR101 may result if connected improperly.** No connection to J2 is to be made if the ADR101 is powered by a regulated 5 volt supply.
For safe operation the total current sourced by digital I/O should not exceed 100mA.



POWER-UP USING A WALL ADAPTOR

The ADR101 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 200-500mA .(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 ADR101. When the ADR101 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 amount of current sourced by the digital ports. **For safe operation the total current sourced by digital I/O and the power terminals should not exceed 100mA.**



ANALOG INPUT COMMAND SUMMARY

RAn	Returns status of analog input in % full scale (0 - 5VDC Ref.) (n=0 or 1)
RDn	Returns status of analog input in decimal format (0 - 5VDC Ref.) (n=0 or 1)

DIGITAL COMMAND SUMMARY

CPAxxxxxxxx	Configures PORT A. (x=1 for input, x=0 for output)
SPAxxxxxxxx	Output binary data to PORT A. (x=1 or 0)
RPA	Returns status of all I/O lines in PORT A in binary format.
RPAn	Returns status of I/O line specified by n. (n= 0 to 7)
MAddd	Outputs decimal data (ddd) to PORT A. (ddd= 0 to 255)
PA	Returns status of PORT A in decimal format.
RESPAn	Resets I/O line specified by n in PORT A. (n= 0 to 7)
SETPAn	Sets I/O line specified by n in PORT A. (n= 0 to 7)

3. ADR101 COMMANDS

a) ANALOG INPUT COMMANDS

There are 2 analog inputs, with a resolution of 8-bits, on the ADR101 labeled AN0 to AN1. The analog input range is 0 to 5 VDC. The commands used to read analog inputs allow data to be retrieved in two formats.

RAn Returns status of analog port specified by n in % full scale format. (n = 0 or 1) (Input voltage range used for conversion 0 to 5VDC)

example;

```
RA0<CR>
36.5
```

(Input AN0 is at 36.5% of full-scale or .365 X 5.00 = 1.825 V)

RDn Returns status of analog port specified by n in decimal format. (n = 0 or 1) Returns integer value from 000 to 255.(Input voltage range used for conversion is 0 to 5VDC)

example;

```
RD1<CR>
202
```

(To convert to voltage; voltage =(reading/255) X 5)
(Input AN1 is (202/255) X 5 =3.96V

c) DIGITAL PORT COMMANDS

There is one, eight bit digital port on the ADR101 labeled PORT A. The individual I/O lines are labeled PA0-PA7. The following commands allow the user to;

-configure individual bits an input or output
-SET or RESET individual bits
-read individual bits
-read entire port in binary or decimal format
-write to entire port in binary or decimal format.

The digital port commands are;

CPAxxxxxxxx Configures each bit of PORT A . All eight bits must be specified. Order is MSB-LSB (x=1 for input, x=0 for output)

example;

```
CPA11110000<CR>
```

(PA7 ,PA6, PA5, PA4 are configured as inputs and PA3, PA2, PA1, PA0 are configured as outputs)

SPAxxxxxxxx Outputs binary data to PORT A. All eight bits must be specified.

Order is MSB-LSB. Individual bits configured as input are not effected by this command. (x=1 or 0)

example; SPA10101000<CR>

(PA7, PA5, PA3 are set, PA6, PA4, PA2, PA1, PA0 are reset)

RPA Returns status of all I/O lines in PORT A in binary format. Order is MSB-LSB. Individual lines configured as output will return last data set on the port.

example; RPA<CR>
0 1 1 1 0 0 1 0
(PA7, PA3, PA2, PA0 are low, PA6, PA5 ,PA4, PA1 are high)

RPAn Returns status of I/O line in PORT A specified by n.(n=0 to 7)

example; RPA4<CR>
1
(PA4 is high)

MAddd Outputs decimal data (ddd) to PORT A. Individual lines configured as input are not effected by this command. (ddd= 000 to 255)

example; MA255<CR>

(All lines of PORT A are set)

PA Returns status of PORT A in decimal format (000-255). Individual lines configured as output will return last data set on PORT A.

example; PA<CR>
128
(PA7 is high, PA6 thru PA0 are low)

RESPAn Resets I/O line specified by n in PORT A. This command has no effect on I/O lines configured as input. (n=0 to 7)

example; RESPA4<CR>
(PA4 is reset)

SETPAn Sets I/O line specified by n in PORT A. This command has no effect on I/O lines configured as input. (n=0 to 7)

example; SETPA3<CR>
(PA3 is set)

4.SENDING COMMANDS IN BASIC TO THE ADR101

OPENING A SERIAL FILE

Commands may be sent to the ADR101 using a terminal emulation program such as Hyperterminal by simply typing commands and pressing the enter key. With BASIC, the ADR101 is connected to the computer via a serial cable and BASIC treats the ADR101 as a serial file. Before commands can be sent to the ADR101 this serial file must be opened and initialized. This should be done at the start of any program that is to access the ADR101. 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 ADR101 using PRINT#1 and INPUT#1 commands.

SENDING COMMANDS

Sending commands in BASIC to the ADR101 can be done using PRINT#1 commands. For example, sending an RD0 command could be done as shown below;

```
20 PRINT#1, "RD0"
```

Extra spaces inside the quotes are ignored by the ADR101. Avoid sending commands on consecutive lines because a <CR> is not sent after the first command resulting in an unrecognized command. This problem arises with the configuring of a digital port and then trying to access the port immediately after it is configured. A REM statement should be inserted between consecutive PRINT#1 commands as shown below;

```
20 PRINT#1, "CPA00000000"
30 REM FORCES <CR>
40 PRINT#1, "SETPA0"
```

Variable names may also be used with PRINT#1 commands. One example of this is shown below. This program configures PORT A as output and then increments it from 0 to 255.

```
10 OPEN "COM1:9600,n,8,1,CS,DS,RS" AS#1
20 PRINT#1, "CPA00000000"
30 FOR X = 0 to 255
40 PRINT#1, "MA",X
50 NEXT X
60 END
```

RECEIVING DATA

When reading analog inputs or the digital port, data is sent from the ADR101 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 ADR101. 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 RPA command then eight variable names must be used with the INPUT#1 command. Examples of both cases are shown below;

```
20 PRINT#1, "RD0"
30 INPUT#1, ANADAT
40 PRINT#1, "RPA"
50 INPUT#1, PA7,PA6,PA5,PA4,PA3,PA2,PA1,PA0
```

The variable names used in the INPUT#1 commands now contain the data sent by the ADR101 The data can now be scaled, printed, displayed, saved or whatever is required by the application.

A BASIC PROGRAM EXAMPLE

A complete BASIC program which reads analog port 0 and sets PA0 if the analog port is above decimal value 128 (2.5 volts) is shown below;

```
10 OPEN "COM1:9600,n,8,1,CS,DS,RS" AS#1          ;opens and configures serial file
20 PRINT#1, "CPA11111110"                        ;configures PA0 as output
30 REM FORCES <CR>
40 PRINT#1, "RESPA0"                             ;resets PA0
50 REM FORCES <CR>
50 PRINT#1, "RD0"                                ;sends RD0 command
60 INPUT#1, AN0                                  ;receives data into variable AN0
70 IF AN0>128 THEN PRINT#1, "SETPA0": GOTO 50    ;sends SETPA0 command if AN0>128
                                                 and returns to line 50
80 PRINT#1, "RESPA0" : GOTO 50                   ;resets PA0 and returns to 50
```

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 ADR101

This section will demonstrate how to send and receive data from the ADR101 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 ADR101 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 ADR101 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 (see programming examples). The bioscom command needs only to be executed once before the ADR101 is accessed.

SENDING COMMANDS TO THE ADR101

To send commands to the ADR101 the "fprintf" command is used. For example, the following command sends an RD0 (read analog port 0) command to the ADR101;

```
fprintf (stdaux,"RD0 \xD");
```

The \xD suffix sends a carriage return after the command which is needed by the ADR101 to recognize a command. Integer variables may also be used in the command line. For example, the following command sends a MAddd (make port A=ddd) command, where DOUT is a previously defined integer value of 0 to 255.

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

RECEIVING DATA FROM THE ADR101

If a command sent to the ADR101 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 an RD0 command and stores the retrieved data in an integer variable named AN0;

```
fprintf (stdaux,"RD0 \xD");
fscanf (stdaux,"%D",&an0);
rewind (stdaux);
```

In this example, the command PA (read port A)is sent to the ADR101 and the retrieved data is stored in an integer variable named PORTA;

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

The following test programs outline the proper syntax for using the commands in simple applications. The first program retrieves the status of analog port 0 and displays the data on the video screen. The second program configures PORT A as output, sets the port to decimal 255, reads back the port status and displays the data on the video screen.

/* PROGRAM EXAMPLE ONE - ANALOG PORT TEST PROGRAM */

```
#include <stdio.h>
#include <bios.h>
#define   com1  0
#define  settings  (0xE3)

main( )
{           /* declare an0 as an integer  number */
int an0 ;
            /* configure com1 9600 baud, 8 bit words, no parity */
bioscom (0,settings,com1);
            /* send RD0 command to ADR101 on com1 */
fprintf(stdaux,"RD0 \xD");
            /* read data from com1 and store it at address of an0 */
fscanf (stdaux,"%d",&an0);
            /* initialize com1 buffer */
rewind (stdaux);
            /* print data on screen */
printf ("ANALOG PORT 0= %d  \n",an0);
}
```

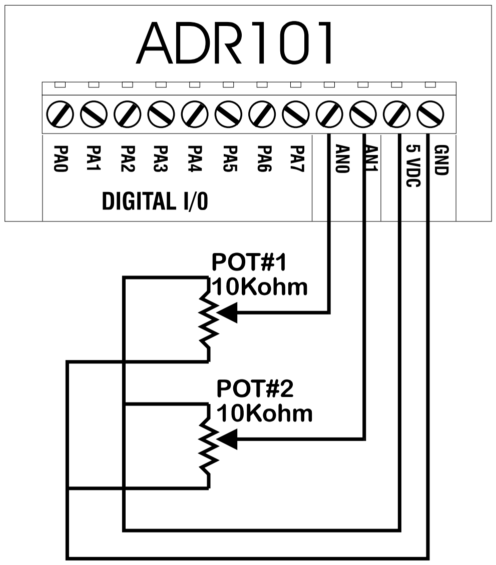
```
/* PROGRAM EXAMPLE TWO - DIGITAL PORT TEST PROGRAM */

#include <stdio.h>
#include <bios.h>
#define   com1  0
#define  settings  (0xE3)

main ( )
{           /* declare PORTA and DOUT as integer numbers */
int PORTA,DOUT ;
            /* set DOUT to integer 255 */
DOUT=255;
            /* configure com1 9600 baud, 8 bit words, no parity */
bioscom (0,settings,com1);
            /* send CPA00000000 command to ADR101 on com1 */
fprintf (stdaux,"CPA00000000 \xD");
            /* send MAddd (ddd=DOUT) command to ADR101 on com1 */
fprintf (stdaux,”MA %d \xD",DOUT );
            /* send PA command to ADR101 on com1 */
fprintf (stdaux,"PA \xD");
            /* read data from com1 and store at address of PORTA */
fscanf (stdaux,"%d",&PORTA );
            /* initialize com1 buffer */
rewind (stdaux);
            /* print data on screen */
printf ("PORT A is %d DECIMAL \n",PORTA);
}
```

7. Interfacing to the ADR101 (Basic Examples)

The following, show basic examples of interfacing various devices to the ADR101. Sample programs are written in BASIC and demonstrate proper command syntax.

A) Reading Potentiometer Position

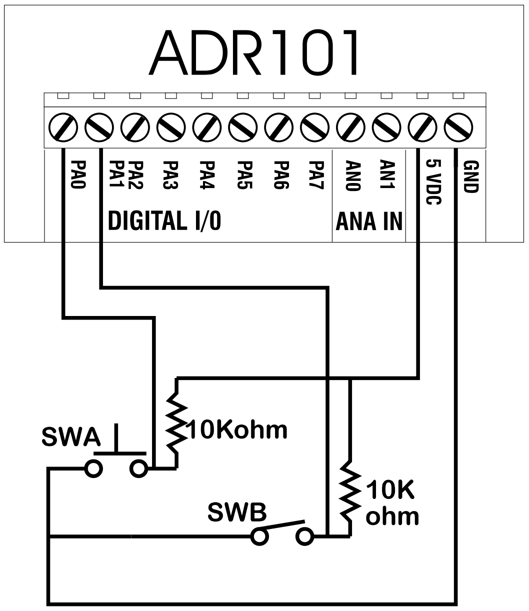
To monitor potentiometer position, the potentiometer must be biased with 5VDC. The wiper of the pot is then connected to one of the analog inputs. The sample BASIC program reads the potentiometer position using the RD0 command which responds with a decimal value between 000 and 255. The value is then converted to a percent and displayed 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, "RD0"                               ;send RD0 command to ADR101
50 INPUT#1, POT                                 ;retrieve data from ADR101
60 POT=(POT/255)*100                     ;convert data to percent
70 PRINT "Potentiometer Position is", POT       ;display it
80 GOTO 30                                      ;repeat procedure
```

B) Connecting Switches to Digital Ports

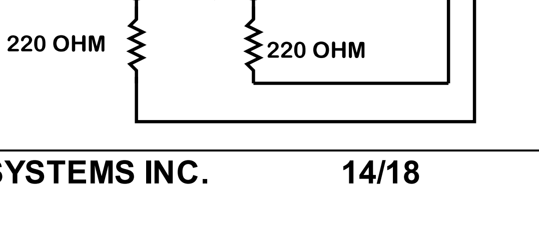
To connect switches to digital I/O lines only one additional component is required. Each digital input line used to read a switch must be tied to +5V via a 10Kohm resistor. This is to avoid leaving the digital port floating when the switch is in the open position. The switch is then connected between the digital port and ground. The sample BASIC program first configures the digital I/O lines as input and then reads the switches and displays their status on the video screen.



```
10 OPEN "COM1:9600,N,8,1,CS,DS,RS" AS#1                          ;opens com port
20 CLS                                                 ;clears screen
30 LOCATE 1,1                                          ;locates cursor
40 PRINT#1, "CPA11111111"                                        ;configures port as input
50 REM                                                           ;forces <CR>
60 PRINT#1, "RPA0"                                               ;reads PA0 ( SW1 )
70 INPUT#1, SW1                                                  ;saves status in variable SW1
80 PRINT#1, "RPA1"                                               ;reads PA1 ( SW2 )
90 INPUT#1,SW2                                                   ;saves status in variable SW2
100 T1$="CLOSED" IF SW1=1 THEN T1$="OPEN        "                ;define T1$
110 T2$="CLOSED" IF SW2=1 THEN T2$="OPEN        "                ;define T2$
120 PRINT "SW1 is " T1$                                          ;print SW1 status
130 PRINT "SW2 is " T2$                                          ;print SW2 status
140 GOTO 60                                                      ;repeat procedure
```

C) Connecting LED's to Digital Ports

LED's may be controlled using the digital I/O lines on the ADR101. Only one additional component is needed to drive LED's. A current limit resistor is required for each LED with a value of around 220 Ohms. LED's may be controlled by sourcing (LED1) or sinking (LED2) drive current. LED1 is turned on by setting PA0 to a logic one or turned off by resetting PA0 to a logic zero. The sample BASIC program demonstrates how to turn the LED1 on and off.

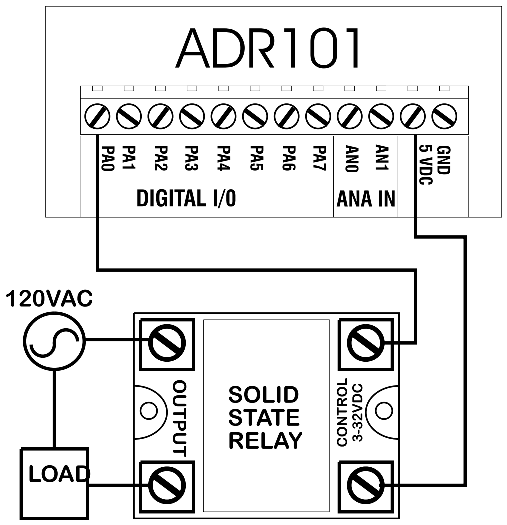


```
10 OPEN "COM1:9600,N,8,1,CS,DS,RS" AS#1          ;opens com port
20 CLS                                  ;clears screen
30 PRINT#1, "RESPA0"                    ;resets PA0*
40 REM                                           ;forces <cr>
50 PRINT#1, "CPA11111110"                        ;configures PA0 as output
60 REM Turn on LED                               ;forces <cr>
70 PRINT#1, "SETPA0"                    ;turns on LED
80 REM Turn off LED                              ;forces <cr>
90 PRINT#1, "RESPA0"                    ;turns off LED
100 END
```

* PA0 remains in high impedance state until the CPA command is used to configure the port as output.

D) Driving Solid State Relays

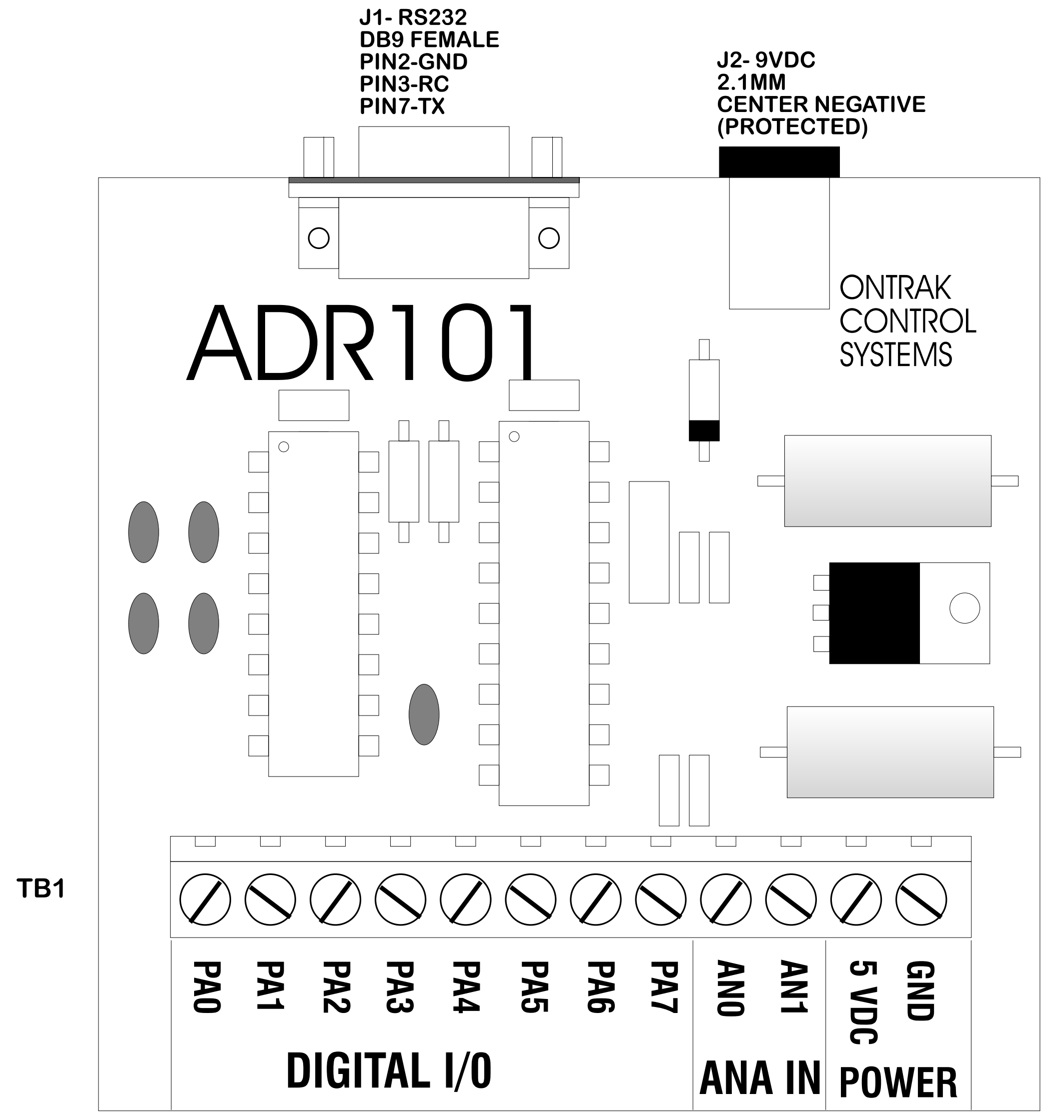
Solid-State relays that require a DC voltage to operate may be driven by ADR101 digital I/O lines directly if the current input specification for the relay is 20mA or less. The relay must be rated for the proper voltage and current required by the load. Each relay requires one digital I/O line to operate and requires no other external components. The sample BASIC program demonstrates how the relay is turned on. In this configuration the digital I/O line sinks the relay drive current. Note that the I/O line is SET before the CPA command is used to configure the port as output to avoid the relay turning on unexpectedly when the port is configured.



```
10 OPEN "COM1:9600,N,8,1,CS,DS,RS" AS#1          ;opens com port
20 CLS                                  ;clears screen
30 PRINT#1, "SETPA0"                    ;sets PA0
40 REM                                           ;forces <cr>
50 PRINT#1, "CPA11111110"                        ;configures PA0 as output
60 REM Turn on relay                             ;forces <cr>
70 PRINT#1, "RESPA0"                    ;turns relay on
80 REM Turn off relay                            ;forces <cr>
90 PRINT#1, "SETPA0"                    ;turns relay of
```

APPENDIX A CONNECTION DIAGRAM

.



APPENDIX B ELECTRICAL SPECIFICATIONS

ADR101

Supply Voltage	5VDC+/- 10% or 7-9VDC via wall adaptor
Supply Current*	15mA Typical, 30mA Maximum
Operating Temperature	0-50C

* All digital outputs unloaded. Supply current maximum increases depending on current drawn from power supply terminals and current sourced by digital I/O.

Analog Inputs (8)

Resolution	8 bits
Accuracy	+/- 0.5%
Range	0-5VDC
Input Impedance	1Mohm

Digital I/O (8)

Sink Current	20mA Max
Source Current	20mA Max
Vout High	4.00V Min
Vout Low	0.8V Max
Vin High	2.2V min
Vin Low	0.8V max

Note: Maximum total source or sink current from all I/O lines in PortA = 100mA

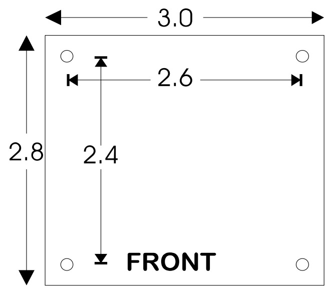
Communication Interface

RS232
9600 baud, 8 bit words, no parity, 1 start bit

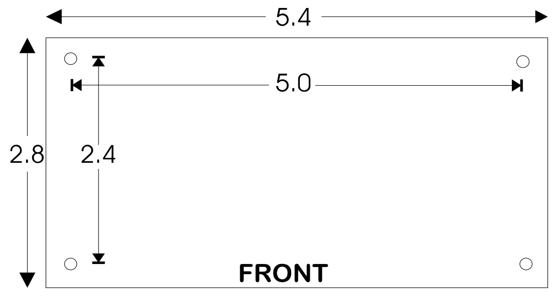
Visit our web site at http://www.ontrak.net/ for additional applications and programming examples.

APPENDIX C MOUNTING DIMENSIONS

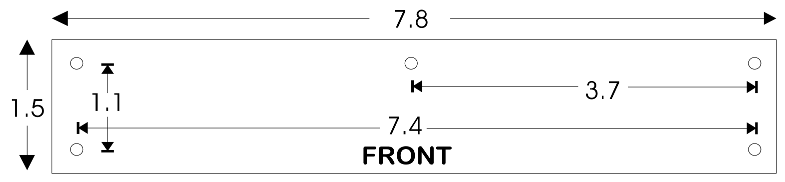
ADR Interface Mounting Dimensions (in inches)



ADR101,ADR112
Ext. Dim 3.0 X 2.8
Hole Pat. 2.6 X 2.4
Hole Dia 0.125



ADR2000A,ADR2000B
ADR2100,ADR2010
ADR2200
Ext. Dim 5.4 X 2.8
Hole Pat. 5.0 X 2.4
Hole Dia 0.125



ADRTERM
Ext. Dim 7.8 X 1.5
Hole Pat. 7.4 X 1.1
Hole Dia 0.125

.