ICS ELECTRONICS
division of Systems West Inc.

IEEE 488 APPLICATION BULLETIN

CONTROLLING SERIAL TEMPERATURE CONTROLLERS FROM THE GPIB BUS

INTRODUCTION

This application note describes how to interface and control industrial temperature controllers with serial interfaces from the GPIB bus. Typical temperature controllers are the Tenny Versatrenn or the Watlow 942, 988 and F4 series controllers.

BACKGROUND

Temperature controllers used in temperature and environmental chambers often have the ability to be remotely controlled over a RS-232 and RS-422 or RS-485 serial link. In certain manufacturing applications, it is convenient to control the chambers from the same GPIB bus that is being used to control other devices and collect test data.

Two of the major temperature controller manufacturers are Tenny (Lunaire) and Watlow Controls. Their controllers employ unique serial command protocols such as X-on/X-off, ANSI X3.28 (ACK-NAK), and the Modbus RTU Packet Protocol. Each protocol requires special programming considerations if it is going to be handled in the test program. This application note describes the important features of each protocol and solutions to simplify the programming effort.

ICS manufactures several GPIB-to-Serial Interfaces that can be used to convert GPIB commands to serial strings and serial data back into GPIB responses. The Model 4894A is a small, stand-alone product in its own metal case. The Model 4804 and the 4814 are board products. All three transparently convert GPIB commands to serial strings and serial strings back into GPIB responses. The Model 4814 has a firmware option that automatically handles the ANSI X3.28 ACK-NAK protocol and simplifies the user's software. Other ICS GPIB-to-Modbus Controllers such as the Models 4899, 4809 and 4819, convert simple GPIB commands into the Modbus RTU Packet Protocol and make it easy to control Modbus slave devices from the GPIB bus.

THE X-ON/X-OFF PROTOCOL

The X-on/X-off protocol is for point-to-point communication and requires a carriage return character (<CR>) at the end of each command. The controller responds with a an X-off and then sends a X-on character when ready for the next command. With this protocol, the user should set the ICS GPIB-to-Serial interfaces to operate with the X-on/X-off protocol and to add a line feed (<LF>) character to the temperature controller responses before passing them back to the GPIB controller. ICS GPIB-to-Serial interfaces will automatically strip the X-on and X-off characters from the temperature controller responses so they do not go back to the GPIB controller. The added linefeed character terminates the response string and makes it easier for many GPIB controller drivers to input he response string.

The user should start the dialog by sending the temperature controller a single X-on character. The problem with the X-on/X-off protocol is that the user does not know when the temperature controller is ready for another command. To avoid overrunning the temperature controller with multiple commands, insert a nominal delay of 100 milliseconds between commands. After the program runs successfully, the delay time can be shortened until the controller hangs up or displays a protocol error. Then increase the delay time by 20 milliseconds for a safe value.

Typical X-ON/X-OFF dialog is shown below. *M i*ndicates Master, *T* indicates the temperature controller.

Send Command
M: =<sp>SP1<sp>500<CR> Set temp to 50

Query
M: ?<sp>C1<CR> Request sensor #1 reading
T: 500<CR><LF> Value = 50.0

ANSI X3.28 (ACK-NAK) PROTOCOL

The ANSI X3.28 or ACK-NAK Protocol is a little harder for the user to use but it has the advantage of providing a response to every command. This lets the user know when the temperature controller is ready for the next command. The protocol can also be used for multiple temperature controllers on the same serial link.

The protocol rules are:

1. Initiate communication by addressing a specific temperature controller.
2. Start every message with a <STX> character and end the message with a <ETX> character.
3. Use the <EOT> character to give the temperature controller permission to respond. Acknowledge the response message with an acknowledge (<ACK>) character.
3. Restart communication if the temperature controller does not respond within a reasonable time.

This protocol, while not hard to generate in a PC, it is somewhat difficult to use with most GPIB card drivers. Because the controller responses do not end with a linefeed, the user has to anticipate the number of characters expected or create a subroutine that reads individual characters from the GPIB interface until the serial poll status indicates the buffer is empty. ICS's 4894A and 4804 Interfaces can be set to generate an EOI when the last character is read out of the buffer. This lets the user read the response strings with a normal GPIB input or enter command.

Figure 1 shows an example of the ANSI X3.28 protocol for a temperature controller at address 0. *M* indicates Master, *T* indicates the temperature controller.

Open Link	
M: 0<ENQ>	Open link with temp ctlr #0
T: 0<ACK>	Acknowledge link open
Send Command	
M: <STX>=<sp>SP1<sp>100<ETX>	Set temp setpoint
T: <ACK>	Acknowledge command
Query	
M: <STX>?<sp>C1<ETX>	Request sensor #1 reading
T: <ACK>	Acknowledge query
M: <EOT>	Okay to send response
T: <STX>500<CR><ETX>	Value = 50.0
M: <ACK>	Acknowledge
T: <EOT>	End of query
Disconnect	
M: <DLE><EOT>	End link, no response

If the temperature controller returns an NAK character, the master is to resend the last transmission.

Figure 1 ANSI X3.28 Protocol Sequence

4814's ANSI X3.28 OPTION

ICS's Model 4814's GPIB to Serial Interface has an internal firmware option that automatically handles the ANSI X3.28 ACK-NAK protocol and simplifies the GPIB communication by eliminating the ENQ, STX and ETX characters. The user's program only has to send the 4814 the temperature controller commands or query strings and wait for GPIB responses that are terminated with linefeeds. The 4814 accepts GPIB messages that are terminated with a line feed and prepares responses that are terminated with a carriage return-line feed sequence, <CR><LF>. This option program is enabled by turning on the rocker switch 7 on the 4814's Address Switch.

The 4814 with the ANSI X3.28 ACK-NAK option enabled, reduces the example in Figure 1 to the simplier printable character strings shown in Figure 2

Open Link	
M: 0<LF>	Open link with temp ctlr
T: <ACK><CR><LF>	Acknowledge link open
Send Command	
M: =<sp>SP1<sp>100<LF>	Set temp setpoint
T: <ACK><CR><LF>	Acknowledge
Query	
M: ?<sp>C1<LF>	Request sensor #1 rdg
T: 500<CR><LF>	Value = 50.0
Disconnect	
M: <DLE><LF>	End link, no response.

Note: The disconnect command is not required if there is a separate GPIB controller for each temperature controller.

Figure 2 4814-Controller Sequence

CONNECTING TO THE CONTROLLER

The Model 4814 GPIB-to-Serial Interface has three connectors: a GPIB connector, a RS-422 connector and a RS-232 Connector. Figure 3 shows the connector and switch locations.

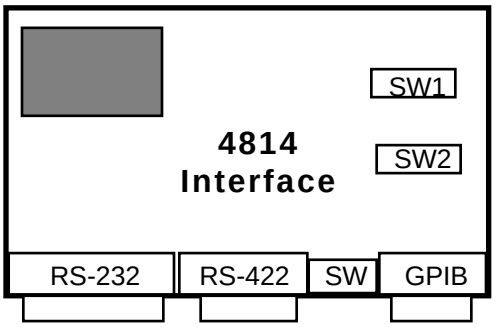

Figure 3 4814 Connector Layout

RS-232 connections are good for short distances up to 50 feet and in electrically quiet environments. They are fairly simple and require just three wires. The transmitter of one device is connected to the receiver of the other device. The user can always determine the RS-232 transmitter by measuring the output voltage on each line and selecting the line with the most negative reading (typically -6 to -25 Vdc).

RS-422 signals have better common mode noise rejection and can be used over longer distances than can RS-232 signals. RS-422 connections are harder to establish as there are two pairs of wires to connect. RS-422 signal names are confusing as they vary from full names like SendData to abbreviations like Tx and Rx. They often end in (A), (B) or + and -. (A signals are positive true and normally mate to + signals.) Figure 3 shows how the 4814 is connected to a Watlow 942 Controller with RS-422 signals. The dotted ground wire is optional and is not needed for the serial signals. Use a cable with two twisted pairs of wires. Use one pair of wires for each pair of signals. For long runs, use a shielded cable and connect the shield to ground at one end of the cable.

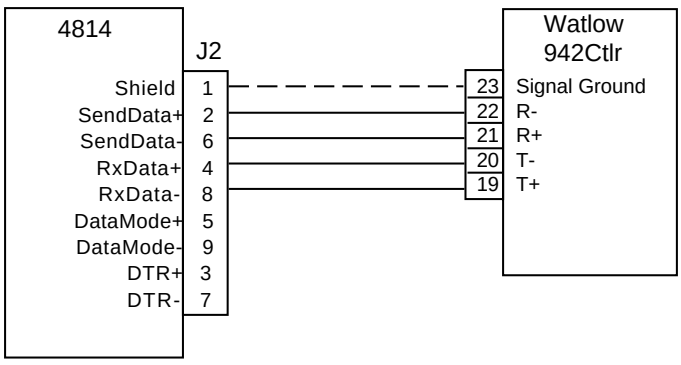

Figure 4 4814 to Watlow 942 Connections

Set the 4814's Address Switch to the desired GPIB address. (The address switch is located between the GPIB and RS-422 connectors. A logic 1 setting is up (away from the PC board). Refer to page 3-4 in the 4814 manual for the rocker assignments and locations.) Set the Address Switch rockers 6 and 7 on to enable the RS-422 signals and the ANSI X3.28 ACK-NAK option firmware. Set the Character Format Switch (SW1) to 8 data bits, no parity and X-on off. Set the Baud rate switch to 9600 baud and CR-LF on. If the 4814 is to be used with the X-on/X-off protocol, set the X-on rocker on and turn the OPT rocker on the Address Switch off.

To debug the connections, verify that the temperature controller is really set to the same serial characteristics as the GPIB-to-Serial Interface. Check the signal type selection setting, controller address setting, baud rate, number of data bits, parity and protocol. Make any necessary changes to the controller settings and internal switches.

Then use a live keyboard program to send commands over the GPIB bus from the PC's keyboard and to read back the responses from the temperature controller. Two such programs are ICS's PC2KYBD for DOS or GPIBKYBD for Windows. Follow the example sequence in Figure 2. Start the dialog by sending the temperature controller its address. Be sure to read the ACK returned by the temperature controller after it is sent its address. Next query the controller by asking its model number "? <space> mdl " or by asking it its temperature "? <space>C1"

QUICK BASIC 4814 ACK-NAK PROGRAM

The example program in Figure 5 shows how to use a 4814 to communicate with a temperature controller using the ANSI X3.28 ACK-NAK protocol. A typical application has the 4814 or similar GPIB-to-Serial Interface as one device on the GPIB bus along with other measuring instruments. All communication to the temperature controller is done by addressing the 4814 at its GPIB address. In this example, the 4814 has the ACK-NAK option enabled.

The user initiates the communication with the temperature controller in the initialization section of the program by sending the controller its address. The temperature setting, temperature reading and other chamber control instructions and queries are put in the main section of the program. The example Quick Basic program shown below uses ICS's GPIB command set and a 488-PC2 card. The ICS GPIB commands can be easily exchanged for those for another GPIB controller card and/or the program can be converted to another language such as C, Visual Basic or HP Basic.

The program uses three variables. TempCtlr% is the 4814's GPIB address. You need a separate address variable for each

GPIB device. CmdStr$ is the second variable. It is a string that holds the current instruction that the program sends through the 4814 to the temperature controller. Resp$ is the third variable and it is used to read responses from the temperature controller. You can always create multiple response or command string variables if they make sense in your program.

```
INITIALIZATION
TempCtlr% = 4                              '4814 GPIB address
CmdStr$ = "1"                              'temp ctlr address
CALL ieOutput(TempCtlr%, CmdStr$)          'sends temp ctlr ad
                                                dress
msDelay 100                                'wait 100 ms
Resp$ = String$(80," ")                    'clear input string
CALL ieEnter(TempCtlr%, CmdStr$)           'read ACK

MAIN SECTION
CmdStr$ = "= SP1 500"                      'set temperature
CALL ieOutput(TempCtlr%, CmdStr$)          'sends string
Resp$ = String$(80," ")                    'clear input string
CALL ieEnter(TempCtlr%, CmdStr$)           'read ACK
msDelay 100                                '100 ms delay
CmdStr$ = "? c1"                           'temp query
CALL ieOutput(TempCtlr%, CmdStr$)          'sends string
Resp$ = String$(80," ")                    'clear input string
CALL ieEnter(TempCtlr%, CmdStr$)           'read temperature
```

Figure 5 Example Visual Basic Program

If you need to send the temperature controller multiple messages or query multiple parameters such as temperature, setpoint, error status, it is necessary to put a delay of 100 ms between each query sequence. The delay can be reduced once the program has been debugged to speed up the program.

If you are using a 4894A or 4804 Interface, set the interface to generate a EOI when talking out the last character from the Rx buffer. Use the sequence shown in the ANSI X3.28 ACK-NAK protocol paragraph in your program to communicate with the temperature controller. The EOT, STX and ETX are non-printable characters. Their character values are shown in the ASCII chart in the appendix of the Interface's instruction manual. For Basic programs use the CHR$() function to convert the character's decimal value into a string characters. For C, use '\n' type formats to convert the character's octal value into a string characters (STX = '\2'). Provide a short delay after each command so that the GPIB Interface has time to receive the temperature controller's serial response before reading the response string. Else, the interface will generate an EOI while outputting only a partial string.

MODBUS PROTOCOL

The Modbus RTU Protocol consists of message packets made up of 8-bit binary bytes. Each packet has a device address byte, a command byte, register data bytes and a two byte CRC checksum. The slave device responds to each command packet with an ACK, error or response packet. It is a very robust protocol that was designed for industrial control applications. The Modbus Protocol is harder to handle over the GPIB bus than the ANSI X3.28 ACK-NAK protocol because of the lack of defined terminators but it is not impossible. It requires a more sophisticated PC program because of the binary data conversions and CRC calculations. The program must also be able to handle the response packets which the Modbus slave device sends back after every command. For more information about the Modbus Protocol and sample programs refer to ICS's Application Notes AB48-24 and AB48-25 or to your temperature controller's instruction manual.

GPIB to MODBUS CONTROLLER SOLUTION

ICS has three GPIB-to-Modbus Controller products that simplify the Modbus temperature control program by generating the Modbus RTU Packets and automatically handling the response packets. These products are the Model 4899 GPIB-to-Modbus Controller which is a small box product, and two board products, the Model 4809 and 4819, which are designed to be mounted inside the temperature chamber. Modbus slave devices use addressable registers as a way to accept commands and to pass back responses. All of the above products let the user use simple commands to set registers in the Modbus slave device and read back register contents. For more information about the 4899, 4809 and 4819 GPIB-to-Modbus Controllers refer to ICS Application Note AB48-25 and to the product data sheets.

SUMMARY

This application note has described three serial protocols for communicating with Tenny and Watlow Temperature Controllers. The X-ON /X-OFF Protocol is straight forward and can be implemented with virtually any ICS GPIB-to-Serial Interface. Its drawback is that it does not provide a response to every command. The ANSI X3.28 ACK-NAK Protocol is more difficult to use but it has the advantage of providing a response after every command to verify that the command was received. The Models 4894A and 4804 terminate controller response strings with an EOI which make it easier to read back the responses with any GPIB driver software. ICS's Model 4814 GPIB-to-Serial Interface includes a firmware option that simplifies the ACK-NAK protocol and makes it easier for the user to implement. The Modbus Protocol is a packet oriented protocol that uses binary characters and ends each packet with a CRC sequence. The Modbus protocol is the most difficult protocol to use as it requires a significantly more complex program to generate the packets and to handle the response packets. ICS Models 4899, 4809 and 4819 GIB-to-Modbus Controllers automatically handle the packet protocol and let the user control Modbus devices with simple commands over the GPIB bus.