Certification

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology (formerly National Bureau of Standards), to the extent allowed by that organization's calibration facility, and to the calibration facilities of other International Standards Organization members.

Warranty

This Hewlett-Packard product is warranted against defects in materials and workmanship for a period of three years from date of shipment. Duration and conditions of warranty for this product may be superseded when the product is integrated into (becomes a part of) other HP products. During the warranty period, Hewlett-Packard Company will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by Hewlett-Packard (HP). Buyer shall prepay shipping charges to HP and HP shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to HP from another country

HP warrants that its software and firmware designated by HP for use with a product will execute its programming instructions when properly installed on that product. HP does not warrant that the operation of the product, or software, or firmware will be uninterrupted or error free.

Limitation Of Warranty

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied products or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

The design and implementation of any circuit on this product is the sole responsibility of the Buyer. HP does not warrant the Buyer's circuitry or malfunctions of HP products that result from the Buyer's circuitry. In addition, HP does not warrant any damage that occurs as a result of the Buyer's circuit or any defects that result from Buyer-supplied products.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HP SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Exclusive Remedies

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HP SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

Notice

The information contained in this document is subject to change without notice. HEWLETT-PACKARD (HP) MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. HP shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material. This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced, or translated to another language without the prior written consent of Hewlett-Packard Company. HP assumes no responsibility for the use or reliability of its software on equipment that is not furnished by HP.

U.S. Government Restricted Rights

The Software and Documentation have been developed entirely at private expense. They are delivered and licensed as "commercial computer software" as defined in DFARS 252.227- 7013 (Oct 1988), DFARS 252.211-7015 (May 1991) or DFARS 252.227-7014 (Jun 1995), as a "commercial item" as defined in FAR 2.101(a), or as "Restricted computer software" as defined in FAR 52.227-19 (Jun 1987)(or any equivalent agency regulation or contract clause), whichever is applicable. You have only those rights provided for such Software and Documentation by the applicable FAR or DFARS clause or the HP standard software agreement for the product involved

HP E2261A Quad RS-232 Interface M-Module User's Manual
Edition 1

Documentation History

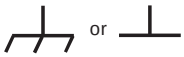
All Editions and Updates of this manual and their creation date are listed below. The first Edition of the manual is Edition 1. The Edition number increments by 1 whenever the manual is revised. Updates, which are issued between Editions, contain replacement pages to correct or add additional information to the current Edition of the manual. Whenever a new Edition is created, it will contain all of the Update information for the previous Edition. Each new Edition or Update also includes a revised copy of this documentation history page.

Edition 1 . . . . . September, 1997

Safety Symbols

Instruction manual symbol affixed to product. Indicates that the user must refer to the manual for specific WARNING or CAUTION information to avoid personal injury or damage to the product.

Indicates the field wiring terminal that must be connected to earth ground before operating the equipment — protects against electrical shock in case of fault.



Frame or chassis ground terminal—typically connects to the equipment's metal frame.

Alternating current (AC)

Direct current (DC).

Indicates hazardous voltages.

WARNING Calls attention to a procedure, practice, or condition that could cause bodily injury or death.

CAUTION Calls attention to a procedure, practice, or condition that could possibly cause damage to equipment or permanent loss of data.

WARNINGS

The following general safety precautions must be observed during all phases of operation, service, and repair of this product. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the product. Hewlett-Packard Company assumes no liability for the customer's failure to comply with these requirements.

Ground the equipment: For Safety Class 1 equipment (equipment having a protective earth terminal), an uninterruptible safety earth ground must be provided from the mains power source to the product input wiring terminals or supplied power cable.

DO NOT operate the product in an explosive atmosphere or in the presence of flammable gases or fumes.

For continued protection against fire, replace the line fuse(s) only with fuse(s) of the same voltage and current rating and type. DO NOT use repaired fuses or short-circuited fuse holders.

Keep away from live circuits: Operating personnel must not remove equipment covers or shields. Procedures involving the removal of covers or shields are for use by service-trained personnel only. Under certain conditions, dangerous voltages may exist even with the equipment switched off. To avoid dangerous electrical shock, DO NOT perform procedures involving cover or shield removal unless you are qualified to do so.

DO NOT operate damaged equipment: Whenever it is possible that the safety protection features built into this product have been impaired, either through physical damage, excessive moisture, or any other reason, REMOVE POWER and do not use the product until safe operation can be verified by service-trained personnel. If necessary, return the product to a Hewlett-Packard Sales and Service Office for service and repair to ensure that safety features are maintained.

DO NOT service or adjust alone: Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

DO NOT substitute parts or modify equipment: Because of the danger of introducing additional hazards, do not install substitute parts or perform any unauthorized modification to the product. Return the product to a Hewlett-Packard Sales and Service Office for service and repair to ensure that safety features are maintained.

Declaration of Conformity

according to ISO/IEC Guide 22 and EN 45014

Manufacturer's Name: Hewlett-Packard Company
Loveland Manufacturing Center

Manufacturer's Name: Hewlett-Packard Company
Loveland Manufacturing Center

declares, that the product:

Product Name: Quad RS-232 Interface M-Module

Model Number: HP E2261A

Product Options: All

conforms to the following Product Specifications:

Safety: IEC 1010-1 (1990) Incl. Amend 2 (1996)/EN61010-1 (1993)
CSA C22.2 #1010.1 (1992)
UL 3111-1 (1994)

EMC: CISPR 11:1990/EN55011 (1991): Group 1, Class A
EN61000-3-2:1995 Class A
EN61000-3-3:1995
EN50082-1:1992
IEC 1000-4-2:1995 : 4kV CD, 8kV AD
IEC 1000-4-3:1995: 3 V/m
IEC 1000-4-4:1995: 1kV Power Line, 0.5kV Signal Lines
ENV50141:1993/prEN50082-1 (1995): 3 Vrms
EN61000-4-8: 1993/prEN50082-1 (1995): 3 A/m

Supplementary Information: The product herewith complies with the requirements of the Low Voltage Directive 73/23/EEC and the EMC Directive 89/336/EEC and carries the "CE" mark accordingly.

Tested in a typical configuration in an HP C-Size VXI mainframe.

April 30, 1997

Jim White, QA Manager

European contact: Your local Hewlett-Packard Sales and Service Office or Hewlett-Packard GmbH, Department HQ-TRE, Herrenberger Straße 130, D-71034 Böblingen, Germany (FAX +49-7031-14-3143)

Notes:

Please fold and tape for mailing

Reader Comment Sheet

HP E2261A Quad RS-232 Interface M-Module User's Manual & SCPI Programming Guide
Edition 1

You can help us improve our manuals by sharing your comments and suggestions. **In appreciation of your time, we will enter you in a quarterly drawing for a Hewlett-Packard Palmtop Personal Computer** (U.S. government employees are not eligible for the drawing).

Your Name	City, State/Province
Company Name	Country
Job Title	Zip/Postal Code
Address	Telephone Number with Area Code

Please list the system controller, operating system, programming language, and plug-in modules you are using.

fold here

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 37 LOVELAND, CO

POSTAGE WILL BE PAID BY ADDRESSEE

HEWLETT-PACKARD COMPANY
Measurement Systems Division
Learning Products Department
P.O. Box 301
Loveland, CO 80539-9984

cut along this line

fold here

Please pencil-in one circle for each statement below:

	Disagree ◄				► Agree
• The documentation is well organized.	O	O	O	O	O
• Instructions are easy to understand.	O	O	O	O	O
• The documentation is clearly written.	O	O	O	O	O
• Examples are clear and useful.	O	O	O	O	O
• Illustrations are clear and helpful.	O	O	O	O	O
• The documentation meets my overall expectations.	O	O	O	O	O

Please write any comments or suggestions below–be specific.

Chapter 1
Getting Started

What is in this Manual?

This manual contains information about installing and using the HP E2261A Quad RS-232 Port M-Module. This information includes a description of this module and wiring/configuration information (Chapter 1), Standard Commands for Programmable Instruments (SCPI) programming information with example programs (Chapter 2), and register information with example programs (Chapter 3). Module specifications are provided in Appendix A of this manual.

The HP E2261A is intended to be installed and used on an M-Module Carrier (such as the HP E2251). For specific information about installing and using the HP E2261A M-Module in an HP carrier, refer to the carrier's manual(s).

HP E2261A Description

The HP E2261A provides four full-duplex (asynchronous) RS-232 Data Terminal Equipment (DTE) ports on one M-Module. Each port provides:

- Independently programmable baud rate: (13 fixed rates from 75 baud to 38400 baud)
- Independently programmable handshake mode: RTS/CTS and/or DTR/DSR hardware handshake, XON/XOFF software handshake, or NONE
- Independently programmable data format: 5 to 8 data bits plus parity; odd, even, or no parity; 1, 1.5, or 2 stop bits.
- Independent 2kByte Input and Output buffer for each port.

When installed in an HP carrier, such as the HP E2251 M-Module Carrier, the M-Module can be addressed and used like a standard VXIbus module. The carrier provides a logical address for the M-Module and VXI register configuration. When installed in an HP M-Module carrier the M-Module can be programmed by using Standard Commands for Programmable Instruments (SCPI) commands (see Chapter 2) or by the VXI*plug&play* driver (refer to the CD ROM supplied with your M-module). Alternately, you can program the M-Module at the register level (refer to Chapter 3 in this manual).

Installing the Module

If you are installing the HP E2261A in an HP M-Module carrier, such as the HP E2251, refer to the carrier's user manual for complete installation instructions.

Note The HP E2261A may not fit properly in either of the two interior slots (M4 or M5) of an HP E2251 Carrier if you use the optional HP E2261-61601 cable.

Note The procedures in this section assume the M-Module(s) have already been installed into an M-Module Carrier. Since installation is dependent on the carrier used, instructions for installing the M-Modules into the carrier are not included here. Refer to your M-Module Carrier documentation for installation instructions.

Wiring and Configuration

An optional cable (HP p.n. E2261-61601) splits the M-Module connector into four standard 9-pin (male) connectors. These 9-pin connectors have the same pin-out as a standard PC serial interface. If you need to make your own cable, Figure 1-1, Figure 1-2, and Table 1-1 shows the pin-out and wiring of the M-Module Connector and the 9-pin connectors[1]. Figure 1-3 shows how to assemble the 44-pin user connector hood if you are making your own cable.

WARNING **SHOCK HAZARD. Only qualified service-trained personnel who are aware of the hazards involved should install, remove, or configure modules. Before installing or removing any module or carrier, disconnect power from the mainframe and user wiring.**

Caution **STATIC ELECTRICITY. Static electricity is a major cause of component failure. To prevent damage to the electrical components on an M-Module or the carrier, observe anti-static techniques whenever installing, removing, or working on a carrier or M-Module.**

1. Signal names are from the perspective of the HP E2261A as the DTE device

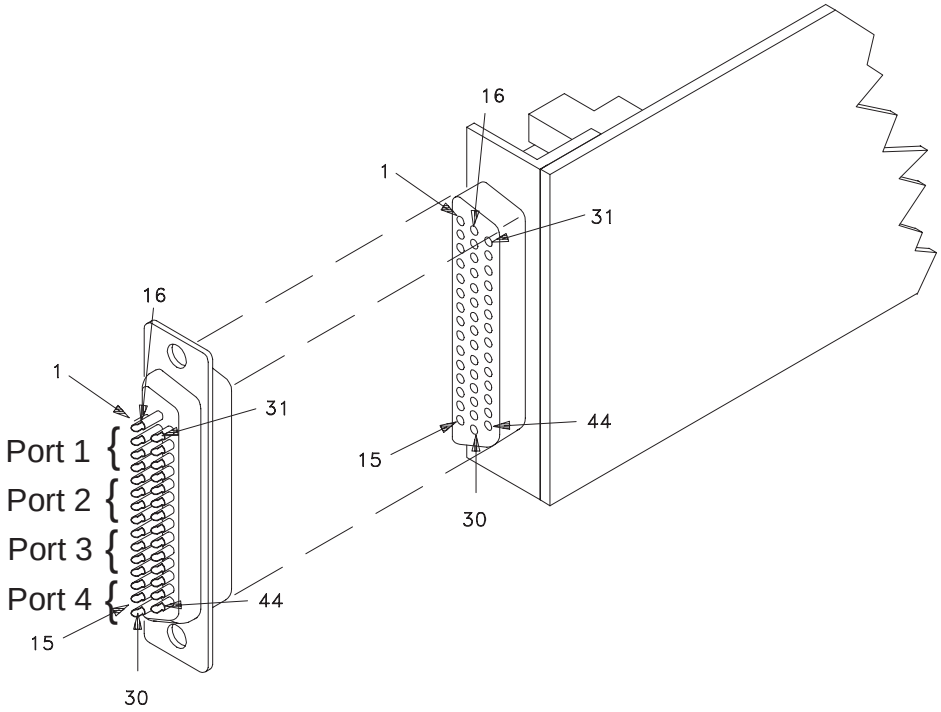


Figure 1-1. Ports on the HP E2261A Quad RS-232 Port M-Module

Table 1-1. HP E2261A Quad RS-232 Port M-Module Connector Pin Definitions

HP E2261A Quad RS-232 M-Module Pin Name and Function		Port 1 Pin #s	Port 2 Pin #s	Port 3 Pin #s	Port 4 Pin #s		9-Pin (Male) Connector Pin #
DTR	Data Terminal Ready	1	5	9	13		4
TD	Transmitted Data	2	6	10	14		3
RD	Received Data	3	7	11	15		2
RTS	Request to Send	16	20	24	28		7
CTS	Clear to Send	17	21	25	29		8
DSR	Data Set Ready	18	22	26	30		6
SG	Signal Ground	4, 8, 12, 19, 23, 27					5
	Chassis Connection[a]	31, 32, 33, . . . 43, 44					-

a. If you use a shielded cable, connect the shield to a chassis connector pin on the 44-pin connector. Do not connect the other end of the shield.

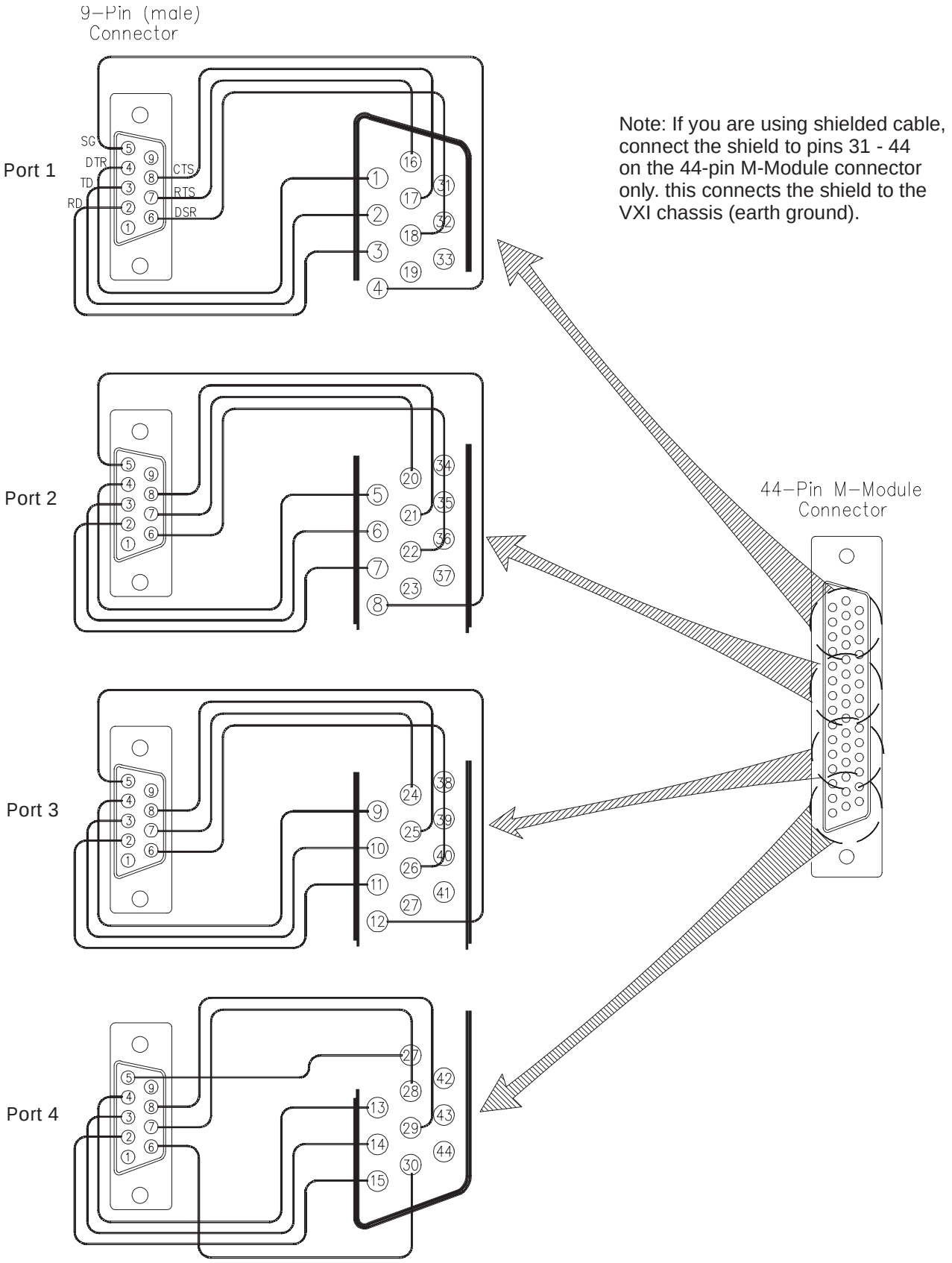


Figure 1-2. Optional Cable Wiring for HP E2261A

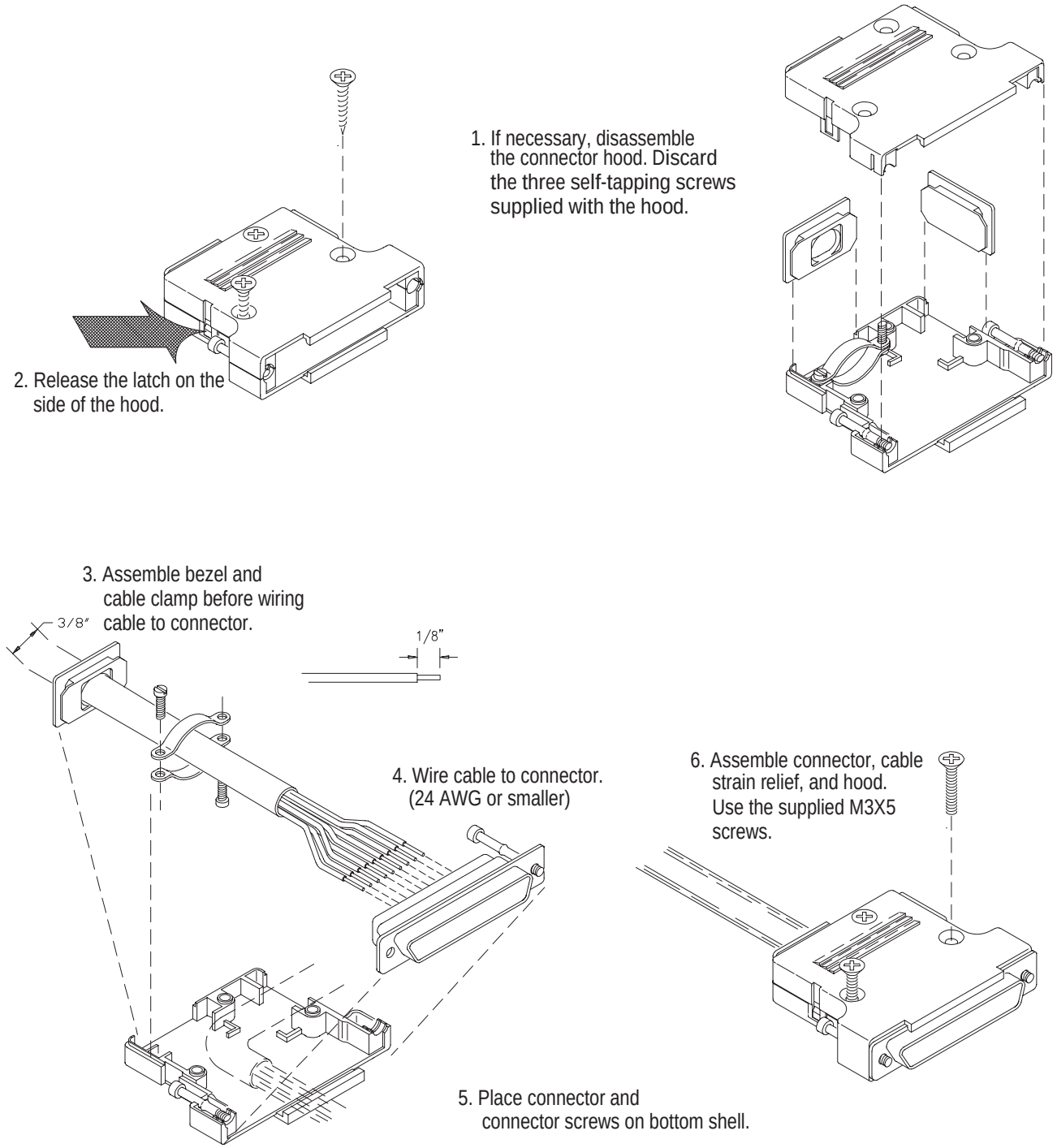


Figure 1-3. Assembling the 44-Pin User Connector Hood

Chapter 2
SCPI Programming

General Information

This chapter describes how to use and program the HP E2261A Quad RS-232 Port M-Module with the Standard Commands for Programmable Instruments (SCPI). This chapter includes example programs and a complete SCPI Command Reference.

Note Do not do register writes if you are controlling the module by the high level driver. This is because the driver will not know the module state and an interrupt may occur causing the driver and/or command module to fail.

The following example programs were developed with the ANSI C language using the HP VISA extensions. The program was written and tested in Microsoft® Visual C++ but should compile under any standard ANSI C compiler.

To run the programs you must have the HP SICL Library, the HP VISA extensions, and an HP 82340 or 82341 HP-IB module installed and properly configured in your PC. An HP E1406 Command Module is required.

Example 1: Reset, Self Test, Module ID

The following example reads the module ID string, performs module self-test, displays the results, closes channel 02 and queries the channel closure state. The result is returned to the computer and displayed ("1" = channel closed, "0" = channel open).

```
#include <visa.h>
#include <stdio.h>
#include <stdlib.h>

/* Interface address is 112, Module secondary address is 9*/
#define INSTR_ADDR "GPIB0::9::9::INSTR"

int main()
{
   ViStatus errStatus;  /*Status from each VISA call*/
   ViSession viRM;     /*Resource mgr. session */
   ViSession E2261A; /* Module session  */
   char id_string[256]; /*ID string*/
   char selftst_string[256]; /*self-test string*/

     /* Open the default resource manager  */
   errStatus = viOpenDefaultRM ( &viRM);
   if(VI_SUCCESS > errStatus){
      printf("ERROR: viOpenDefaultRM() returned 0x%x\n",errStatus);
```

```
    return errStatus;}


  /* Open the Module instrument session  */
errStatus = viOpen(viRM,INSTR_ADDR, VI_NULL,VI_NULL,&E2261A);
if(VI_SUCCESS > errStatus){
   printf("ERROR: viOpen() returned 0x%x\n",errStatus);
   return errStatus;}

  /* Set Timeout Value for *RST and Self Test */
viSetAttribute (E2261A,VI_ATTR_TMO_VALUE,10000);

  /* Reset the Module */
errStatus = viPrintf(E2261A, "*RST\n");
if(VI_SUCCESS > errStatus){
   printf("ERROR: viPrintf() returned 0x%x\n",errStatus);
   return errStatus;}

  /* Perform Module Self-Test */
errStatus = viQueryf(E2261A,"*TST?\n","%t",selftst_string);
if (VI_SUCCESS > errStatus) {
   printf("ERROR: viQueryf () returned 0x%x\n",errStatus);
   return errStatus;}
printf("Self Test Result is %s\n",selftst_string);

  /* Query the Module ID string */
errStatus = viQueryf(E2261A,"*IDN?\n","%t",id_string);
if (VI_SUCCESS > errStatus) {
   printf("ERROR: viQueryf () returned 0x%x\n",errStatus);
   return errStatus;}
printf("ID is %s\n",id_string);

  /* Close the Module Instrument Session */
errStatus = viClose (E2261A);
if (VI_SUCCESS > errStatus) {
   printf("ERROR: viClose() returned 0x%x\n",errStatus);
   return 0;}

  /* Close the Resource Manager Session */
errStatus = viClose (viRM);
if (VI_SUCCESS > errStatus) {
   printf("ERROR: viClose() returned 0x%x\n",errStatus);
   return 0;}

return VI_SUCCESS;
}
```

Example 2: Reading and Writing Data

The following program demonstrates how to write data to a standard (dumb) RS-232 terminal and how to read data from the terminal. The program sets the HP E2261A Port 1 to the following (default) parameters:

- 9600 Baud (both transmit and receive)
- 8 data bits
- No parity check
- 1 Stop Bit
- XON/XOFF pacing

Of course the external RS-232 terminal must be set to the same parameters.

```
#include <visa.h>
#include <stdio.h>
#include <stdlib.h>

    /* Interface address is 112, Module secondary address is 9*/
#define INSTR_ADDR "GPIB0::9::9::INSTR"

int main()
{
ViStatus errStatus;                 /*Status from each VISA call*/
ViSession viRM;                     /*Resource mgr. session */
ViSession E2261A;                   /* Module session  */
char Return_String[256];            /*String read from terminal*/

    /* Open the default resource manager  */
  errStatus = viOpenDefaultRM ( &viRM);
  if(VI_SUCCESS > errStatus){
    printf("ERROR: viOpenDefaultRM() returned 0x%x\n",errStatus);
    return errStatus;}

    /* Open the Module instrument session  */
  errStatus = viOpen(viRM,INSTR_ADDR, VI_NULL,VI_NULL,&E2261A);
  if(VI_SUCCESS > errStatus){
    printf("ERROR: viOpen() returned 0x%x\n",errStatus);
    return errStatus;}

    /* Set Timeout for 10 seconds */
  viSetAttribute (E2261A,VI_ATTR_TMO_VALUE,10000);

    /* Reset the Module */
  errStatus = viPrintf(E2261A, "*RST\n");
  if(VI_SUCCESS > errStatus){
    printf("ERROR: viviPrintf() returned 0x%x\n",errStatus);
    return errStatus;}

    /* Clear TX & RX Buffers */
  errStatus = viPrintf(E2261A,"DIAG:SER1:CLEAR TXRX\n");
  if (VI_SUCCESS > errStatus) {
    printf("ERROR: viPrintf() returned 0x%x\n",errStatus);
    return errStatus;}
```

```
    /* Send Text String, can be up to 256 characters */
   errStatus = viPrintf(E2261A,"SOUR:SER1:TEXT \"%s\"\n","This is a text
string.");
   if (VI_SUCCESS > errStatus) {
      printf("ERROR: viPrintf() returned 0x%x\n",errStatus);
      return errStatus;}

    /* Receive Text from Terminal */
   errStatus =
viQueryf(E2261A,"SENS:SER1:TEXT?\n","%t",Return_String);
   if (VI_SUCCESS > errStatus) {
      printf("ERROR: viPrintf() returned 0x%x\n",errStatus);
      return errStatus;}
   printf("Text Returned is %s\n",Return_String);

    /* Close the Module Instrument Session */
   errStatus = viClose (E2261A);
   if (VI_SUCCESS > errStatus) {
      printf("ERROR: viClose() returned 0x%x\n",errStatus);
      return 0;}

    /* Close the Resource Manager Session */
   errStatus = viClose (viRM);
   if (VI_SUCCESS > errStatus) {
      printf("ERROR: viClose() returned 0x%x\n",errStatus);
      return 0;}

   return VI_SUCCESS;
}
```

Chapter 3
SCPI Command Reference

General Information

This chapter describes how to use and program the HP E2261A Quad RS-232 Port M-Module with the Standard Commands for Programmable Instruments (SCPI). This chapter includes example programs and a complete SCPI Command Reference.

Note Do not do register writes if you are controlling the module by the high level driver. This is because the driver will not know the module state and an interrupt may occur causing the driver and/or command module to fail.

SCPI Command Reference

Command Types

Commands are separated into two types: IEEE 488.2 Common Commands and SCPI Commands.

Common Command Format

The IEEE 488.2 standard defines the common commands that perform functions such as reset, self-test, status byte query, and so on. Common commands are four or five characters in length, always begin with the asterisk character (*), and may include one or more parameters. The command keyword is separated from the first parameter by a space character. Some examples of common commands are shown below:

```
*RST          *ESR 32          *STB?
```

SCPI Command Format

The SCPI commands perform functions like closing switches, making measurements, and querying instrument states or retrieving data. A subsystem command structure is a hierarchical structure that usually consists of a top level (or root) command, one or more lower level commands, and their parameters. The following example shows part of a typical subsystem:

```
SYStem:
    [:COMMunicate]
        :SERial[1|2|3|4]
            :CONTrol
            :ERRormode
            :MODE
            [:RECeive]
                :BAUD
```

SYSTem is the root command, [:COMMunicate] is a second level command, :SERial is a third level command with options, and :CONTrol, :ERRmode, :MODE, and [:RECeive] are fourth level commands.

Command Separator

A colon (**:**) always separates one command from the next lower level command as shown below:

```
SYStem[:COMMunicate]:SERial[1|2|3|4]:CONTrol:CTS
```

Colons separate the root command from the second level command (SYStem:COMMunicate) and the second level from the third level (COMMunicate:SERial).

Abbreviated Commands

The command syntax shows most commands as a mixture of upper and lower case letters. The upper case letters indicate the abbreviated spelling for the command. For shorter program lines, send the abbreviated form. For better program readability, you may send the entire command. The instrument will accept either the abbreviated form or the entire command. For example, if the command syntax shows **COMMunicate**, then **COMM** and **COMMUNICATE** are both acceptable forms. Other forms of **COMMunicate**, such as **COMMUN** or **COM** will generate an error. You may use upper or lower case letters. Therefore, **COMMUNICATE**, **communicate**, and **CoMmUnIcAtE** are all acceptable.

Implied Commands

Implied commands are those which appear in square brackets ([]) in the command syntax. (Note that the brackets are not part of the command and are not sent to the instrument.) Examine the partial subsystem shown below:

```
SYStem:
    [:COMMunicate]
        :SERIAL[1|2|3|4]
            :CONTrol
            :ERRormode
            :MODE
            [:RECeive]
                :BAUD
```

Both [:COMMunicate] and [:RECeive] are an implied commands. Those commands are optional and may be ignored. The brackets around the 1|2|3|4 indicate an option, you should chose one of the options. If you don't chose an option, the default (in this example the default is 1) is used. To set the receiver baud rate for serial Port 1, either of the following commands will work:

```
    SYS:COMM:SER1:REC:BAUD
or
    SYS:SER:BAUD
or
    SYS:COMM:SER1:BAUD
or
    SYS:SER1:REC:BAUD
```

Command Parameters

The following table contains explanations and examples of parameter types you might see later in this chapter.

Parameter Type	Explanations and Examples
Numeric	Accepts all commonly used decimal representations of number including optional signs; decimal points; and scientific notation. 123; 123E2; -123; -1.23E2; .123; 1.23E-2; 1.23000E-01. Special cases include MINimum; MAXimum; and DEFault.
Boolean	Represents a single binary condition that is either true or false. ON; OFF; 1; 0
Discrete	Selects from a finite number of values. These parameters use mnemonics to represent each valid setting. An example is the SYSTem:SERial1:CONTrol:DTR <mode> where <mode> can be either ON, OFF, STAN, or IBF

Optional Parameters. Parameters shown within square brackets ([ ]) are optional parameters. (Note that the brackets are not part of the command and are not sent to the instrument.) If you do not specify a value for an optional parameter, the instrument chooses a default value. For example, consider the **`SERial[1|2|3|4]`** command. If you send the command without specifying a parameter, the default serial port (port 1) is used.

Linking Commands

Linking IEEE 488.2 Common Commands with SCPI Commands. Use a semicolon between the commands. For example:

```
*RST;SYSTem:ERR?
```

Linking Multiple SCPI Commands. Use both a semicolon and a colon between the commands. For example:

```
DIAG:LINE 1;:DIAG:LINE?
```

The DIAGnostic subsystem sets or queries the status of the HP E2261A. Refer to your VXIbus command module documentation for more information on the interrupt line. The port to be affected by some commands is specified by the port number after the :SERial command (i.e. :SERial1 is Port 1, :SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1).

Note The VXIbus Interrupt Line is controlled by the VXIbus M-Module Carrier **NOT** by the M-Module. DIAGnostic:LINE reprograms the HP E2251 M-Module Carrier. The commands will work properly only if the M-Module is installed in an HP E2251 M-Module Carrier.

Syntax

```
DIAGnostic
    :LINE <interrupt_line>
    :LINE?
    :SERial[1|2|3|4]
        :CLEARbuffer TX | RX | TXRX
        :RECeive
            :AVAilable?
            :CHANnel ENABle | DISAble
            CHANnel?
        :STATus?
        :TRANsmit
            :BUFFerstatus?
            :CHANnel ENABle | DISAble
            :CHANnel?
```

DIAGnostic:LINE <*interrupt_line*>

Sets the VXIbus interrupt line the HP E2261A M-Module will use and allows you to enable/disable interrupts. You should not need to modify the default value (line 1).

Note The VXIbus Interrupt Line is controlled by the VXIbus M-Module carrier, NOT by the M-Module. DIAGnostic:INTerrupt:LINE reprograms the HP M-Module carrier. It will work properly only if the M-Module is installed in an HP M-Module Carrier.

Parameter

Parameter Name	Parameter Type	Range of Values	Default Value
<interrupt_line>	numeric	0 - 7	1

Comments

- Only one value (0 - 7) can be set at one time. Specifying 0 disables M-Module interrupting.

Reset Condition *RST sets the value to the default line (line 1).

DIAGnostic:LINE?

Returns the value of the interrupt line of the HP E2261A as an integer between 0 and 7. The power-on, reset is line 1.

DIAGnostic:SERial[1|2|3|4]:CLEARbuffer TX | RX | TXRX

Clears the transmit buffers, receiver buffers, or both for the specified port. Specify TX to clear the transmit buffers, RX to clear the receiver buffers (both the FIFO and SRAM buffers), or TXRX to clear all buffers.

DIAGnostic:SERial[1|2|3|4]:RECeive:AVAilable?

Returns the number of data bytes available in the input buffer of the specified port.

DIAGnostic:SERial[1|2|3|4]:RECeive:CHANnel ENABle | DISAble

Enables or Disables the UART receivers for the specified ports. Note that all receivers are enabled by the driver at power-on. Any disabled receiver is automatically enabled by an attempt to read data from the port.

DIAGnostic:SERial[1|2|3|4]:RECeive:CHANnel?

Returns the current status of the specified port's receiver. The command returns the string "ENAB" for enabled, or the string "DISA" for disabled.

DIAGnostic:SERial[1|2|3|4]:STATus

Returns an integer value representing the status of the handshake lines for the specified port. The integer bit definitions are as follows:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = XOFF sent, receive buffer reached STOP point 1 = XON sent, data in RX buffer less than START value	Not used	0 = DTR OFF 1 = DTR ON	0 = RTS OFF 1 = RTS ON	0 = XOFF received transmit on hold 1 = XOFF not received, transmit allowed	Not used	0 = DSR OFF 1 = DSR ON	0 = CTS OFF 1 = CTS ON

DIAGnostic:SERial[1|2|3|4]:TRANsmit:BUFFerstatus

Returns the status of the output FIFO buffer for the specified port. If it is at least half full (greater than 1024 bytes), the command returns an integer 1. If the buffer is less than half full, the command returns an integer 0 indicating it is safe to send up to 1024 bytes of data to the buffer.

DIAGnostic:SERial[1|2|3|4]:TRANsmit:CHANnel ENABle | DISAble

Enables or disables the UART transmitter for the specified port. Note that all transmitters are enabled at power-on or reset. A disabled transmitter is automatically enabled when an attempt is made to transmit data.

DIAGnostic:SERial[1|2|3|4]:TRANsmit:CHANnel?

Returns the current status of the specified port's UART transmitter. The command returns the string "ENAB" for enabled, or the string "DISA" for disabled.

The SENse subsystem receives data from an external device and sends it to the host (system) computer. The port to be affected by the command is specified by the port number after the :SERial command (i.e. :SERial1 is Port 1, :SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1).

Syntax

```
SENSe
    :SERial[1|2|3|4]
                [:TEXT?] [n]
                :DATA? [n]
```

SENSe:SERIAL[1|2|3|4][:TEXT?] [*n*]

This command receives an unquoted text string from the external device. The command stores the ASCII characters in a buffer until a terminating condition is reached. When the terminating condition is reached, the text string is sent to the host (system) computer. The maximum string length is 2048 characters. If you need to receive more than 2048 characters, multiple :TEXT? command must be used.

There are three possible terminating conditions (see the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator commands).

- The first terminating condition is when a specified terminating character is received. This is typically a Carriage Return (CR) or Line Feed (LF) or both (CRLF). The terminating character must not be set to NONE. The :TEXT? command returns the data as soon as the terminator is received. Use the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:RECeive command to set the terminating character.

- The second terminating condition is a specified number of bytes are received from the external device. Use the parameter [*n*] to set the number of bytes. Default and maximum number of bytes is 2048.

- The third terminating condition is a time-out. The unquoted text string is returned to the host computer when a specified amount of time has elapsed. Set the timeout value with the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:TIMEout command. The warning message, “Read/write job timeout” is added to the error queue.

The terminator is normally not part of the returned ASCII string. You can use the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:PASSthrough command to make the terminator character(s) part of the returned data string.

SENSe:SERIAL[1|2|3|4]:DATA? [*n*]

This command receives either a binary block of information or ASCII text from the external device. The command stores the binary data in a buffer until a terminating condition is reached. When the terminating condition is reached, the binary data is sent to the host (system) computer. The maximum data length is 2048 characters. If you need to receive more than 2048 characters, multiple :DATA? command must be used.

There are three possible terminating conditions for data (see the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator commands).

- The first terminating condition is when a specified terminating character is received. This condition is NOT valid when receiving binary data; it is only valid when receiving ASCII text. The terminating character is typically a Carriage Return (CR) or Line Feed (LF) or both (CRLF). The terminating character must not be set to NONE. The :DATA? command returns the data as soon as the terminator is received. Use the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:RECeive command to set the terminating character. The terminator is normally not part of the returned ASCII string. You can use the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:PASSthrough command to make the terminator character(s) part of the returned data string.

- The second terminating condition is a specified number of bytes are received from the external device. Use the parameter [*n*] to set the number of bytes. Default and maximum number of bytes is 2048.

- The third terminating condition is a time-out. The data is returned to the host computer when a specified amount of time has elapsed. Set the timeout value with the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:TIMEout command. The warning message, “Read/write job timeout” is added to the error queue.

The SOURce subsystem transmits data from the specified HP E2261A port to an external device. The port to be affected by the command is specified by the port number after the :SERial command (i.e. :SERial1 is Port 1, :SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1).

Syntax

```
SOURce
        :SERial[1|2|3|4]
                [:TEXT] <"text string">
                :DATA <data>
```

SOURce:SERial[1|2|3|4][:TEXT] <"*text_string*">

This command sends a quoted ASCII text string from the host computer through the specified HP E2261A port to an external device. Only the text string is sent, the quotation marks are not sent (to send a quotation mark, use the SOURce:SERial[1|2|3|4]:DATA command). Maximum *text_string* size is 2046 characters (2048 including the quotation marks). Use multiple :TEXT commands to send more than 2046 characters. If a termination character is specified with the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator command, then the termination character is sent after each string.

The :TEXT command does not return control until the transmission is complete or a timeout occurs. If a timeout occurs, the string "Read/write job timeout" is added to the error queue.

SOURce:SERial[1|2|3|4]:DATA <*data*>

This command sends a definite or indefinite binary data block or ASCII data from the host computer through the specified HP E2261A port to an external device. The block header and/or trailer are not sent, only the block data. Maximum binary block size is 2048 characters.

If you are transmitting ASCII data, a termination character may be specified with the SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator command. The termination character is sent after the ASCII string. Do NOT use termination characters with binary data; set SYStem:COMMunicate:SERIAL[1|2|3|4]:TERMinator:TRANsmit to NONE.

The :DATA command does not return control until the transmission is complete or a timeout occurs. If a timeout occurs, the string "Read/write job timeout" is added to the error queue.

The STATus subsystem controls the SCPI-defined Operation and Questionable Status registers, Standard Event register, and the Status Byte register. Each is comprised of a condition register, an event register, an enable mask, and transition filters.

Note Transition filters are always set for positive edge transitions. When an event occurs, the condition is set and the event register bit is set true. If the event condition is cleared, the event status register remains set. The event status register is cleared upon reading that register.

Each status register works as follows: when a condition occurs, the appropriate bit in the condition register is set or cleared. The contents of the events register and the enable mask are logically ANDed bit-for-bit; if any bit of the result is set, the summary bit for that register is set in the status byte. The status byte summary bit for the Operation status register is bit 7; for the Questionable Signal status register, bit 3; and for the Standard Event registers is bit 5.

Syntax

```
STATus
        :OPERation
                    :CONDition?
                    :ENABle <mask>
                    :ENABle?
                    [:EVENt]?
        :PRESet
        :QUEStionable
                    :CONDition?
                    :ENABle <mask>
                    :ENABle?
                    [:EVENt]?
```

The STATus system contains five registers, two of which are under IEEE 488.2 control: the Event Status Register (*ESE?) and the Status Byte Register (*STB?). The Operational Status bit (OPR), Service Request bit (RQS), Event Summary bit (ESB), Message Available bit (MAV) and Questionable Data bit (QUE) in the Status Byte Register (bits 7, 6, 5, 4 and 3 respectively) can be queried with the *STB? command. Use the *ESE? command to query the *unmask* value for the Event Status Register (the bits you want logically "OR'd" into the Summary bit). The registers are queried using decimal weighted bit values. The decimal equivalents for bits 0 through 15 are included in Figure 3-1.

Note The Operation Status Condition, Event, and Enable registers exist for SCPI compliance only. No status bits are defined or reported in these registers.

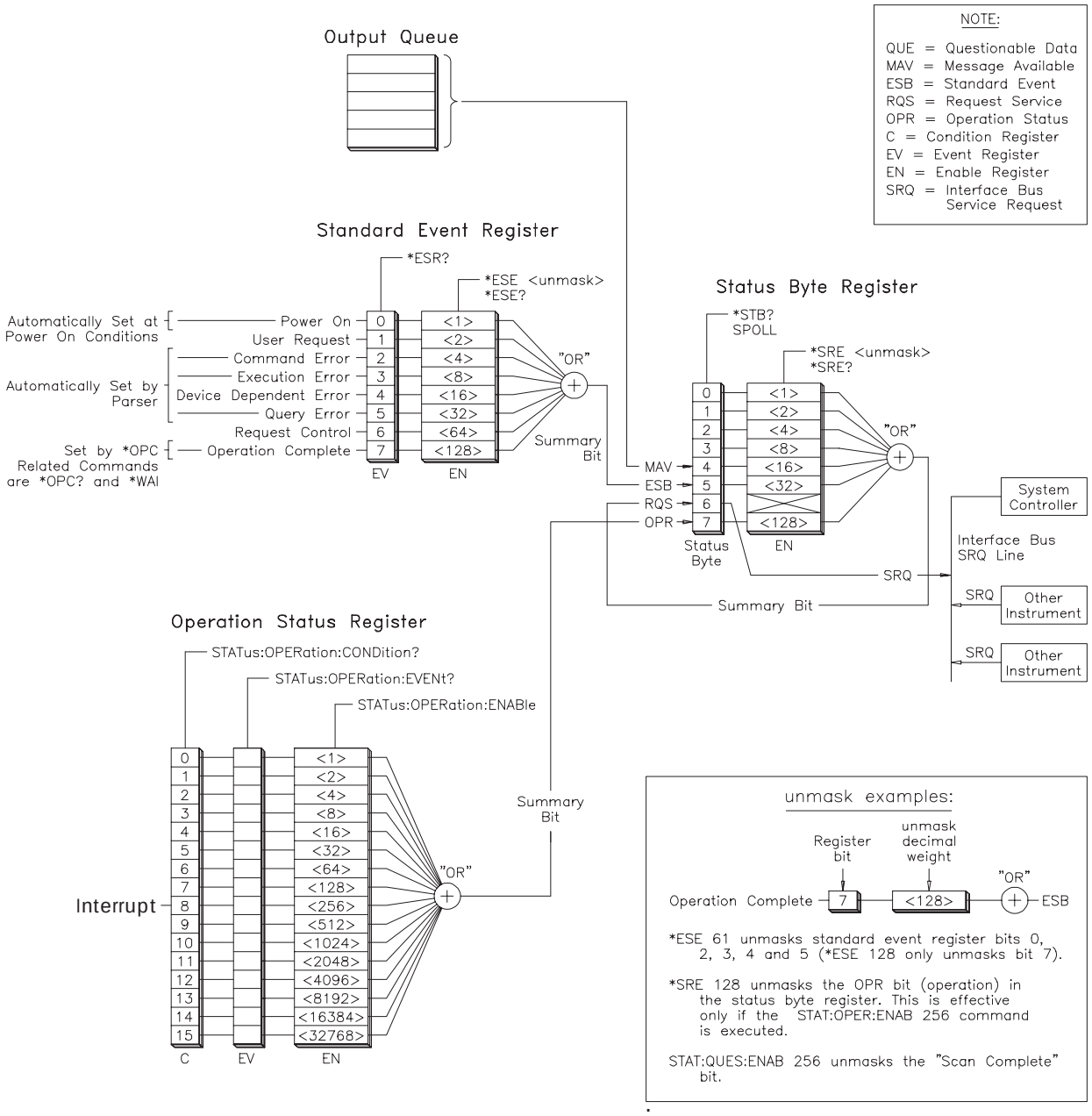


Figure 3-1. HP E2261A Status System Register Diagram

STATus:OPERation:CONDition?

Returns the value of the Operation Status Condition Register as a signed 16 bit integer.

Parameters None

Comments

- *RST clears all Status Operation Conditions.
- *CLS does not affect the contents of the of the Status Operation Conditions.
- The STATus:PRESet command does not affect the Status Operation Conditions.

STATus:OPERation:ENABle <*mask*>

Sets the value of the Operation Status Enable Register.

Parameters

Parameter Name	Parameter Type	Range of Values	Default
<*mask*>	numeric	0 to +32767 (0000_h to $FFFF_h$)	0

Comments

- <*mask*> determines which OPERation Status conditions are summed. See Figure 3-1. The events detected in the Port Summary Status Register are reported in bit 9 of the Operation Status Register which in turn is reported in bit 7 of the Status Byte Register.
- *RST and *CLS do not affect the value of the enable mask.
- STATus:PRESet sets the value of the enable mask to 0.

Example STAT:OPER:ENAB 0xFFFF *Enable all bits of the Operation Status Enable Register*

STATus:OPERation:ENABle?

Returns the value of the OPERation Status Enable Register as a signed 16 bit integer.

Parameters None

Comments

- The only defined bit is bit 9 which is the summary of the Data Available and Edge Status for Ports 0, 1, 2, and 3. See Figure 3-1.

STATus:OPERation[:EVENt]?

Returns the value of the Operation Status Event Register as a signed 16 bit integer and then clears the register to 0.

Parameters None

Comments

- *RST does not affect the contents of the Status Operation Event Register.
- *CLS clears the contents of the Status Operation Event Register.
- STAT:PRESet does not affect the contents of the Status Operation Event Register but does disable reporting the summary of this register in the Status Byte Register (STB?).

STATus:PRESet

Presets the Status system registers and conditions.

Parameters None

Comments

- Resets the following registers and conditions:

Register	Action		Register	Action
Status Byte	none		QUES Status enable	presets to 0
Standard Event event	none		OPER Status condition	none
Standard Event enable	presets to 0		OPER Status event	none
QUES Status Condition	none		OPER Status enable	presets to 0
QUES Status Event	none			

STATus:QUEStionable:CONDition?

Always returns a 0.

Note The Questionable Status Condition, Event, and Enable registers exist for SCPI compliance only. No status bits are defined or reported in these registers.

Parameters None

Comments
- No bits are defined.
- *RST clears all Status Questionable Conditions.
- *CLS does not affect the contents of the Status Questionable Conditions.
- The STAT:PRESet command does not affect the Status Questionable Conditions.

STATus:QUEStionable:ENABle <*mask*>

Sets the value of the Questionable Status Enable Register.

Note The Questionable Status Condition, Event, and Enable registers exist for SCPI compliance only. No status bits are defined or reported in these registers.

Parameters None

Comments
- No bits are defined.
- *RST and *CLS do not affect the value of the enable mask.
- The STAT:PRESet command sets the value of the enable mask to 0.

STATus:QUEStionable:ENABle?

Returns the value of the Questionable Status Enable Register as a signed 16 bit integer.

Note The Questionable Status Condition, Event, and Enable registers exist for SCPI compliance only. No status bits are defined or reported in these registers.

Parameters None

Comments
- No bits are defined.

STATus:QUEStionable[:EVENt]?

Returns the value of the Questionable Status Event Register as a signed 16 bit integer and then clears the register to 0.

Parameters None

Comments

- The following bits are defined:

Bit #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Use	Not Used			Port 1 Read Err.	Port 2 Read Err.	Port 3 Read Err.	Port 4 Read Err.	Not Used								

- This is a destructive read so that all register bits are cleared after the read is executed.
- *RST does not affect the contents of the Status Questionable Event Register.
- *CLS clears the contents of the Status Questionable Event Register.
- STAT:PRESet does not affect the contents of the Status Questionable Event Register but does disable reporting the summary of this register in the Status Byte register (STB?)

The SYSTEM command subsystem configures the RS-232 ports. The port to be affected by the command is specified by the port number after the :SERial command (i.e. :SERial1 is Port 1, :SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1).

Subsystem Syntax

```
SYSTem
    [:COMMunicate]
        :SERial[1|2|3|4]
            :CONTrol
                :CTS 0 | 1 | OFF | ON
                :CTS?
                :DSR 0 | 1 | OFF | ON
                :DSR?
                :DTR ON | OFF | STANdard | IBFull
                :DTR?
                :RTS ON | OFF | STANdard | IBFull
                :RTS?
            :ERRormode IGNOre|STOP
            :ERRormode?
            :MODE NORMal | ECHO | LLOOP | RLOOP
            :MODE?
            [:RECeive]
                :BAUD <baud_rate>
                :BAUD?
                :BITS  5 | 6 | 7 | 8
                :BITS?
                :BLOCKsize <blocksize>
                :BLOCKsize?
                :PACE
                    [:PROTocol] XON | NONE
                    [:PROTocol]?
                    :THReshold
                        :STARt <characters>
                        :STARt?
                        :STOP <characters>
                        :STOP?
                :PARity
                    :CHECk  1 | 0 | ON | OFF
                    :CHECk?
                    [:TYPE] EVEN | ODD | ZERO | ONE | NONE
                    [:TYPE]?
                :SBITs  1 | 1.5 | 2
                :SBITs?
            :TERMinator
                :TRANsmit CR|LF|CRLF|NONE|NR1
                :TRANsmit?
                :RECeive CR|LF|CRLF|NONE|NR1
                :RECeive?
                :PASSthrough 1 | 0 | ON | OFF
                :PASSthrough?
                :TIMEout <time>
                :TIMEout?
```

Subsystem Syntax (continued)

```
SYSTem
   [:COMMunicate]
      :SERial[1|2|3|4]
         :TRANsmit
            :AUTO  1 | 0 | ON | OFF
            :AUTO?
            :BAUD <baud_rate>
            :BAUD?
            :PACE
               [:PROTocol] XON | NONE
               [:PROTocol]?
   :ERRor?
   :VERsion?
```

The SYStem:COMMunicate:SERial[1|2|3|4]: ... commands set and/or modify the configuration of the serial ports on the HP E2261A. The port to be affected by the command is specified by the port number after the :SERial command (i.e. :SERial1 is Port 1, :SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1).

Comments

- Serial communication commands take effect after the end of the program message containing the command.
- Power-on / reset state for the HP E2261A Quad RS-232 Interface M-Module is:

  -- BAUD 9600
  -- BITS 8
  -- PARity NONE
  -- SBITs 1
  -- DTR OFF
  -- RTS OFF
  -- PACE NONE
  -- CTS 0 (Disabled)
  -- DSR 0 (Disabled)
  -- BLOCKsize 2048

SYSTem . . . :CONTrol:CTS 0 | 1 | OFF | ON

Executing SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:CTS 1 enables the Clear To Send (CTS) handshake mode for data transmission. Executing or SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:CTS 0 disables CTS handshaking. CTS is automatically enabled (:CTS 1) if either DTR or RTS is set to STANdard or IBUFull. Power-on/reset state is :CTS 0 (disabled) for all ports.

SYSTem . . . :CONTrol:CTS?

SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:CTS? returns the current state of the Clear to Send Handshake mode; it returns an integer 0 for disabled (default state) or 1 for enabled.

SYSTem . . . :CONTrol:DSR 0 | 1 | OFF | ON

Executing SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:DSR 1 enables the Data/Set Ready (DSR) handshake mode for data transmission. Executing or SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:DSR 0 disables DSR. DSR is automatically enabled (:DSR 1) if DTR is set to either STANdard or IBUFull. Power-on/reset state is DSR disabled (:DSR 0) for all ports.

SYSTem . . . :CONTrol:DSR?

SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:DSR? returns the current state of the Data/Set Ready Handshake mode; returns 0 for disabled (default state) or 1 for enabled.

SYSTem . . . :CONTrol:DTR ON | OFF | STANDard | IBFull

SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:DTR sets the hardware pacing scheme. DTR can be set to a static state (ON | OFF), can operate as a modem control line (STANDard), or can be used as a hardware handshake line (IBFull). The following table defines the DTR parameters:

Value	Definition
ON	DTR line is always asserted
OFF	DTR Line is always unasserted
STANdard	DTR will be asserted when the serial interface is ready to send output data. Data will be sent if the connected device asserts DSR and CTS. Refer to EIA Standard RS-232-C, section 4.4.
IBFull	While the input buffer is not yet at the :STOP threshold, DTR is asserted indicating it is ready to receive data. When the input buffer reaches the :STOP threshold, DTR becomes unasserted. When the number of bytes has reduced to the :STARt threshold, DTS becomes asserted again. The HP E2261A monitors DSR and CTS and stops transmission if either line becomes unasserted.

Power-on/reset state is :DTR OFF for all ports.

SYSTem . . . :CONTrol:DTR?

SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:DTR? returns the current setting for DTR line control; returns a string: “ON”, “OFF”, “STAN”, or “IBF”

SYSTem . . . :CONTrol:RTS ON | OFF | STANdard | IBFull

SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:RTS sets the hardware pacing scheme. RTS can be set to a static state (ON | OFF), can operate as a modem control line (STANDard), or can be used as a hardware handshake line (IBFull). The following table defines the RTS parameters:

Value	Definition
ON	RTS line is always asserted
OFF	RTS Line is always unasserted
STANdard	RTS will be asserted when the serial interface is ready to send output data. Data will be sent if the connected device asserts CTS and DSR. Refer to EIA Standard RS-232-C, section 4.4.
IBFull	While the input buffer is not yet at the :STOP threshold RTS is asserted indicating the HP E2261A is ready to receive. When the input buffer reaches the :STOP threshold, RTS becomes unasserted. When the number of bytes in the buffer drops to the :START threshold, RTS becomes asserted again. The HP E2261A also monitors CTS and stops transmission when CTS becomes unasserted.

Power-on/reset state is :RTS ON for all ports.

SYSTem . . . :CONTrol:RTS?

SYStem:COMMunicate:SERial[1|2|3|4]:CONTrol:RTS? returns the current setting for RTS line control; returns a string: “ON”, “OFF”, “STAN”, or “IBF”

SYSTem . . . :ERRormode IGNOre | STOP

SYStem:COMMunicate:SERial[1|2|3|4]:ERRormode specifies how the driver handles errors for the specified port. If the error handler is set to IGNore and an error occurs, the driver adds the error to the error queue; no further action is taken. If the error handler is set to STOP and an error occurs, the driver stops the current transmission, adds the error to the error queue. Data in the buffer is not cleared. An error on one port does not affect the other ports. Power-on/reset state is IGNore for all ports.

SYSTem . . . :ERRormode?

SYStem:COMMunicate:SERial[1|2|3|4]:ERRormode? returns a string indicating the error mode; it returns “IGNOre” or “STOP”.

SYSTem . . . :MODE NORMal | ECHO | LLOOP | RLOOP

Sets the specified port's operating mode. The NORMal mode is the power-on, reset default.

- ECHO sets the auto-echo mode where the UART receives data and echoes the data back.
- LLOOP mode is the Local Loop back mode. This mode is for diagnostic purposes only. When the UART has data in the transmit buffer, the data is echoed back to the receiver instead of sending it out the external device.
- RLOOP is the Remote Loop back mode. THis mode is for diagnostic purposes only. When the UART receives data from the external device, the data is sent back to the port's transmitter (echoed back to the external device). The data is not placed in the receive buffer.

SYSTem . . . :MODE?

Returns a string with the port's operating mode. The command returns either "ECHO", "LLOOP", or "RLOOP".

SYSTem . . . [:RECeive]:BAUD <*baud_rate*>

SYStem:COMMunicate:SERial[1|2|3|4][:RECeive]:BAUD sets the baud rate (receiver only) for the specified serial port. Valid baud rates are: 75, 110, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, and 38400. All other values result in an error. Power-on/reset baud rate is 9600 for all ports.

SYSTem . . . [:RECeive]:BAUD?

SYStem:COMMunicate:SERial[1|2|3|4][:RECeive]:BAUD? returns the current receiver baud rate for the specified port as an unsigned integer.

SYSTem . . . [:RECeive]:BITS 5 | 6 | 7 | 8

SYStem:COMMunicate:SERial[1|2|3|4][:RECeive]:BITS sets the number of data bits. Power-on/reset value is 8 bits for all ports.

SYSTem . . . [:RECeive]:BITS?

SYStem:COMMunicate:SERial[1|2|3|4][:RECeive]:BITS? returns the current number of data bits as an unsigned integer.

SYSTem . . . [:RECeive]:BLOCKsize <*block_size*>

SYStem:COMMunicate:SERial[1|2|3|4][:RECeive]:BLOCKsize sets the size of the receive data frame. Each port of the HP E2261A has a 2kByte FIFO and a 16kByte SRAM receive buffer. When the HP E2261A receives data from the external device, it passes the data from the UART to the SRAM receive buffer. When the FIFO is empty, and the amount of data in the receive buffer is equal to or greater than the <block_size>, the HP E2261A transfers the <block_size> of data from the receive buffer to the FIFO. In general, the larger the <block_size> the better through-put; however, performance may decrease if <block_size> is much smaller than the actual data size. The <block_size> may be set to any value from 1 to 2048 (bytes). Power-on/reset blocksize is 2048 bytes for all ports.

SYSTem . . . [:RECeive]:BLOCKsize?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:BLOCKsize? returns an integer indicating the current <block_size> value.

SYSTem . . . [:RECeive]:PACE[:PROTocol] XON | NONE

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PACE[:PROTocol] enables or disables receive pacing (XON/XOFF) protocol.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
protocol	discrete	XON | NONE	NONE

Comments

- While the protocol is XON, the serial interface will send XOFF when the buffer reaches the ...STOP threshold, and XON when the buffer reaches the ...STARt threshold.
- The XON character is control Q (ASCII 17), The XOFF character is control S (ASCII 19).
- Power-on Reset condition: NONE

SYSTem . . . [:RECeive]:PACE[:PROTocol]?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PACE[:PROTocol]? returns a string indicating the current receive pacing protocol. The returned string is either “XON” or “NONE”

SYSTem . . . [:RECeive]:PACE:THReshold:STARt <characters>

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PACE:THReshold:STARt configures the specified port's input buffer level. When the number of characters in the input buffer drops to or goes below the specified value, and pacing has been established, the port will indicate it is ready to receive more data.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
characters	numeric	1 to (value of STOP - 2)	8 kBytes

Comments

- ...STARt must be set to less than ...STOP.
- The ...STARt command interacts with the XON/XOFF, DTR, and RTS. Both DTR and RTS must be set to the IBFull mode. When the data in the input buffer drops to the STARt threshold:

  -- if XON/XOFF in enabled, then XON character (ctrl Q, ASCII 17) is sent.
  -- if SYS:CONT:DTR is set to IBFull mode, then DTR line is asserted.
  -- if SYS:CONT:RTS is set to IBFull mode, then RTS line is asserted.

  All three pacing forms can be enabled at the same time.

- Power-on/reset value is 8 kBytes.

SYSTem . . . [:RECeive]:PACE:THReshold:STARt?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PACE:THReshold:STARt? returns an integer as the current start threshold value.

SYSTem . . . [:RECeive]:PACE:THReshold:STOP <characters>

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PACE:THReshold:STOP configures the specified port's input buffer level. When the number of bytes in the input buffer goes to or above the specified value, and pacing has been established, the port will indicate that it is not ready to receive new data.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
characters	numeric	1 to (16 kBytes - 1)	10 kBytes

Comments

- ...STOP must be set to greater than ...STARt.
- The ...STOP command interacts with the XON/XOFF, DTR, and RTS. Both DTR and RTS must be set to the IBFull mode. When the data in the input buffer increases to the STOP threshold:

  -- if XON/XOFF in enabled, then XOFF character (ctrl S, ASCII 19) is sent.
  -- if SYS:CONT:DTR is set to IBFull mode, then DTR line is de-asserted.
  -- if SYS:CONT:RTS is set to IBFull mode, then RTS line is de-asserted.

  All three pacing forms can be enabled at the same time.

- Power-on/reset value is 10 kBytes

SYSTem . . . [:RECeive]:PACE:THReshold:STOP?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PACE:THReshold:STOP? returns an integer as the current stop threshold value.

SYSTem . . . [:RECeive]:PARity:CHECk 0 | 1 | OFF | ON

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PARity:CHECk controls whether or not the parity bit in received serial data frames will be considered significant.

When ...PARity:CHECk is set to 0 or OFF, received data is not checked for correct parity. Transmitted data still includes the type of parity configured with SYSTem . . . [:RECeive]:PARity[:TYPE].

When ...PARity:CHECk is set to 1 or ON, parity checking is performed in accordance with the selected PARity:TYPe

SYSTem . . . [:RECeive]:PARity:CHECk?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PARity:CHECk? returns an integer (0 or 1) indicating the current state of parity checking.

SYSTem . . . [:RECeive]:PARity[:TYPE] EVEN|ODD|ZERO|ONE|NONE

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PARity[:TYPE] configures the type of parity to be checked for received data, and generated for transmitted data.

Comments

- The following table defines each value of type:

Value	Definition
EVEN	If ...PARity:CHECK is ON the received parity bit must maintain even parity. The transmitted parity bit will maintain even parity.
ODD	If ...PARity:CHECK is ON the received parity bit must maintain odd parity. The transmitted parity bit will maintain odd parity.
ZERO	If ...PARity:CHECK is ON the received parity bit must be a zero. The transmitted parity bit will be a zero.
ONE	If ...PARity:CHECK is ON the received parity bit must be a logic one. The transmitted parity bit will be a logic one.
NONE	A parity bit must not be received in the serial data frame. No parity bit will be transmitted.

SYSTem . . . [:RECeive]:PARity[:TYPE]?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:PARity[:TYPE]? returns a string indicating the type of parity checked and generated. The string returned is one of the following: “EVEN”, “ODD”, “ZERO”, “ONE”, “NONE”.

SYSTem . . . [:RECeive]:SBITs 1 | 1.5 | 2

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:SBITs sets the number of stop bits to be used to transmit and receive data.

SYSTem . . . [:RECeive]:SBITs?

SYSTem:COMMunicate:SERial[1|2|3|4][:RECeive]:SBITs? returns the current stop bit setting.

SYSTem . . . :TERMinator:TRANsmit CR | LF | CRLF | NONE | <NRf>

This command defines a termination character for the specified port's transmit operations. The port to be affected by the command is specified by the port number after the :SERial command (i.e. SERial1 is Port 1, SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1). The termination character can be:

- Carriage Return (CR)
- Line Feed (LF)
- Both Carriage Return and Line Feed (CRLF)
- Termination character not sent (NONE)
- Any ASCII character (specify the ASCII value -2 to 255, -2 is CRLF, -1 is NONE)

If a terminator character is set, the driver sends the data after the terminator character is received.

Power-on/reset default is LF.

SYSTem . . . :TERMinator:TRANsmit?

This command returns the ASCII numeric equivalent of the current defined termination character:

- 13 for Carriage Return (CR)
- 10 for Line Feed (LF)
- -2 for both CR and LF
- -1 for NONE
- An ASCII integer value

SYSTem . . . :TERMinator:RECeive CR | LF | CRLF | NONE | <NRf>

This command defines a termination character for the specified port's receive operations. The port to be affected by the command is specified by the port number after the :SERial command (i.e. SERial1 is Port 1, SERial2 is Port 2, etc.). If a port number is not specified, Port 1 is defaulted (that is :SERial is the same as :SERial1). The termination character can be:

- Carriage Return (CR)
- Line Feed (LF)
- Both Carriage Return and Line Feed (CRLF)
- Termination character ignored (NONE)
- Any ASCII character (specify the ASCII value -2 to 255, -2 is CRLF, -1 is NONE)

If a terminator character is set, the driver sends the data after the terminator character is received.

Power-on/reset default is LF.

SYSTem . . . :TERMinator:RECeive?

This command returns the ASCII numeric equivalent of the current defined termination character:

- 13 for Carriage Return (CR)
- 10 for Line Feed (LF)
- -2 for both CR and LF
- -1 for NONE
- An ASCII integer value

SYSTem . . . :TERMinator:PASSthrough 1 | 0 | ON | OFF

When enabled (1 or ON), and a read operation is active, the termination character is passed-through as part of the data. Power-on/reset is disabled (OFF).

SYSTem . . . :TERMinator:PASSthrough?

Returns an integer, either 1 or 0, representing the passthrough status of the specified port. A 1 indicates terminator passthrough is enabled; a 0 (power-on default) means the terminator passthrough is disabled.

SYSTem . . . :TERMinator:TIMEout <*timeout*>

This command sets the termination timeout. If one SENSe:SERial[1|2|3|4]:TEXT? or SENSe:SERial[1|2|3|4]:DATA? command cannot be completed within the timeout period, the data that was received is sent to the host computer. If one SOURce:SERial[1|2|3|4]:TEXT? or SOURce:SERial[1|2|3|4]:DATA? command cannot be completed within the timeout period, the command is aborted and a transmit error is added to the error queue. <*time_out*> can be set from 0 to 65535 (in seconds). If <*time_out*> is 0, the timeout function is disabled. Power-on/reset default value is 1800 seconds (3 minutes).

SYSTem . . . :TERMinator:TIMEout?

This command returns the current timeout value (in seconds) as an unsigned integer.

SYSTem . . . :TRANsmit:AUTO 1 | 0 | ON | OFF

SYSTem:COMMunicate:SERial[1|2|3|4]:TRANsmit:AUTO when set to 1 or ON, sets the transmit baud rate to be the same as that set for receive. When OFF, the transmit baud rate may be set independently of the receive pacing mode. Setting the Transmit baud rate with the SYSTem . . . TRANSmit:BAUD <baud_rate> command automatically sets SYSTem . . . TRANSmit:AUTO to OFF. Power-on/reset default is ON.

SYSTem . . . :TRANsmit:AUTO?

SYSTem:COMMunicate:SERial[1|2|3|4]:TRANsmit:AUTO? returns an integer (0 or 1) indicating the current state of transmit/receive baud rate linkage.

SYSTem . . . :TRANSmit:BAUD <*baud_rate*>

This command sets the baud rate (transmitter only) for the specified serial port. Valid baud rates are: 75, 110, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, and 38400. All other values result in an error. Power-on/reset default baud rate is 9600.

SYSTem . . . :TRANSmit:BAUD?

This query returns the current transmitter baud rate for the specified port as an unsigned integer.

SYSTem . . . :TRANsmit:PACE[:PROTocol] XON | NONE

SYSTem:COMMunicate:SERial[1|2|3|4]:TRANsmit:PACE[:PROTocol] enables or disables the transmit pacing (XON/XOFF) protocol.

Comments

- Receipt of an XOFF character (control Q, ASCII 19) will hold off transmission of data until an XON character (control S, ASCII 17) is received.
- Power-on/reset default is NONE

SYSTem . . . :TRANsmit:PACE[:PROTocol]?

SYSTem:COMMunicate:SERial[1|2|3|4]:TRANsmit:PACE[:PROTocol]? returns a string indicating the current transmit pacing protocol. The string returned is either: “XON” or “NONE”

SYSTem:ERRor?

The SYSTem:ERRor? command requests the next entry from the HP E2261A’s error queue. This queue contains an integer (-32768 to +32767). Negative numbers are SCPI standard, positive numbers are instrument dependent. An error value of 0 indicates that no error has occurred.

SYSTem:VERSion?

The SYSTem:VERSion? returns the SCPI version to which the HP E2261A complies. The decimal value returned is in the form: YYYY.V where YYYY is the year and V is the version number within that year.

IEEE 488.2 Common Command Reference

The following table lists the IEEE 488.2 Common (*) Commands accepted by the HP M-Modules. For more information on Common Commands, refer to the HP 75000 Series C Mainframe (HP Model Number E1400/E1401) User's Manual or the ANSI/IEEE Standard 488.2-1987.

Command	Command Description
*CLS	Clears all status registers and clears the error queue.
*ESE *<register value>*	Enable Standard Event.
*ESE?	Enable Standard Event Query.
*ESR?	Standard Event Register Query.
*IDN?	Instrument ID Query; returns identification string of the module.
*OPC	Operation Complete.
*OPC?	Operation Complete Query.
*RCL *<numeric state>*	Recalls the instrument state saved by *SAV.
*RST	Resets the module to its power-on/reset state.
*SAV *<numeric state>*	Stores the instrument state.
*SRE *<register value>*	Service request enable, enables status register bits.
*SRE?	Service request enable query.
*STB?	Read status byte query.
*TRG	Triggers the module (Does not apply to HP E2261A).
*TST?	Self-test. Executes an internal self-test and returns only the first error encountered. Does not return multiple errors. After *TST? is executed, all data in buffers is lost, all module states are set to power-on/reset conditions. *TST? returns an integer; the lower 8-bits are defined as: Bit 0 - Port 1 SRAM Buffer test failed. Bit 1 - Port 2 SRAM Buffer test failed. Bit 2 - Port 3 SRAM Buffer test failed. Bit 3 - Port 4 SRAM Buffer test failed. Bit 4 - Port 1 UART failed. Bit 5 - Port 2 UART failed. Bit 6 - Port 3 UART failed. Bit 7 - Port 4 UART failed.
*WAI	Wait to Complete.

Note . . . These commands apply to VXI instruments as well as M-Modules. See the *HP 75000 Series C E1400/E1401 Mainframe User's Manual* or the ANSI/IEEE Standard 488.2-1987 for more information about these commands. Refer to the individual M-Module or VXI instrument user manual for details on IEEE 488.2 Common Command actions and results.

Command Subsystem	Description	See Also
DIAGnostic:LINE <interrupt_line>	Sets interrupt level between 0 and 7.	page 26
:LINE?	Returns interrupt level value.	page 27
:SERial[1|2|3|4]		
:CLEARbuffer TX | RX | TXRX	Clears the transmit buffer, receiver buffer, or both.	page 27
:RECeive		
ÄVAilable?	Queries the number of data bytes in the input buffer.	page 27
:CHANnel ENABle | DISAble	Enables/Disables the UART receivers.	page 27
:CHANnel?	Queries the status of the UART receivers.	page 27
:STATus?	Queries the handshake line status.	page 27
:TRANsmit		
:BUFFerstatus?	Queries the output FIFO, returns 1 if half full.	page 27
:CHANnel ENABle | DISAble	Enables/Disables the UART transmitters.	page 28
CHANnel?	Queries the status of the UART transmitters.	page 28
SENSe:SERial[1|2|3|4][:TEXT?] [*n*]	Receives unquoted string from external device.	page 29
:DATA? [*n*]	Receives binary block of data from external device.	page 30
SOURce:SERial[1|2|3|4][:TEXT] <"text string">	Sends a text string from host computer to external device.	page 31
:DATA <data>	Sends binary data block from host computer to external device.	page 31
STATus:OPERation:CONDition?	Returns value of Operation Status Condition Register.	page 34
:ENABle <mask>	Sets value of Operation Status Enable Register.	page 34
:ENABle?	Returns value of Operation Status Enable Register.	page 34
[:EVENt]?	Returns value of Operation Status Event Register.	page 35
:PRESet	Presets the Status system registers and conditions.	page 35
:QUEStionable:CONDition?	Always returns a 0.	page 36
:ENABle <mask>	Sets value of Questionable Status Enable Register.	page 36
:ENABle?	Returns value of Questionable Status Enable Register.	page 36
[:EVENt]?	Returns value of Questionable Event Register.	page 37
SYSTem commands are provided on next page.		

Command Subsystem	Description	See Also
SYSTem[:COMMunicate]:SERial[1|2|3|4]		
:CONTrol		
:CTS 0 | 1 | OFF | ON	Enables/Disables Clear to Send (CTS) handshake mode.	page 39
:CTS?	Returns current state of CTS handshake mode.	page 39
:DSR 0 | 1 | OFF | ON	Enables/Disables Data/Set Ready (DSRT) handshake mode.	page 40
:DSR?	Returns current state of DSR handshake mode.	page 40
:DTR ON | OFF | STANdard | IBFull	Sets Data Terminal Ready (DTR) output line.	page 40
:DTR?	Returns current setting of DTR output line.	page 40
:RTS ON | OFF | STANdard | IBFull	Sets Request to SEnd (RTS) output line.	page 41
:RTS?	Returns current setting of RTS output line.	page 41
:ERRormode IGNOre|STOP	Specifies how driver handles errors.	page 41
:ERRormode?	Returns string indicating how driver handles errors.	page 41
:MODE NORMal | ECHO | LLOOP | RLOOP	Sets port operation mode.	page 42
:MODE?	Queries port operation mode.	page 42
[:RECeive]		
:BAUD <*baud_rate*>	Sets receiver baud rate.	page 42
:BAUD?	Returns current baud rate.	page 42
:BITS 5 | 6 | 7 | 8	Sets number of data bits.	page 42
:BITS?	Returns current number of data bits.	page 42
:BLOCKsize <*blocksize*>	Sets size of receive data frame (block).	page 43
:BLOCKsize?	Returns current blocksize value.	page 43
:PACE		
[:PROTocol] XON | NONE	Enables/disables receive pacing (XON/XOFF).	page 43
[:PROTocol]?	Returns current receive pacing protocol.	page 43
:THReshold		
:STARt <*characters*>	Specifies number of characters before sending XON.	page 44
:STARt?	Returns current number of characters before sending XON.	page 44
:STOP <*characters*>	Specifies number of characters before sending XOFF.	page 45
:STOP?	Returns current number of characters before sending XOFF.	page 45
:PARity		
:CHECk 1 | 0 | ON | OFF	Specifies whether parity bit is significant.	page 46
:CHECk?	Returns current state of parity checking.	page 46
[:TYPE] EVEN|ODD|ZERO|ONE|NONE	Specifies type of parity checked for received data.	page 46
[:TYPE]?	Returns current type of parity checking.	page 46
:SBITs 1 | 1.5 | 2	Sets number of stop bits.	page 47
:SBITs?	Returns number of Stop bits.	page 47
:TERMinator		
:TRANsmit CR | LF | CRLF | NONE | numeric	Defines transmit termination character.	page 47
:TRANsmit?	Returns ASCII numeric equivalent of current termination character.	page 47
:RECeive CR | LF | CRLF | NONE | numeric	Defines receiver termination character.	page 48
:RECeive?	Returns ASCII numeric equivalent of current termination character.	page 48
:PASSthrough	Sets passthrough of termination character on receive.	page 48
:PASSthrough?	Returns passthrough status.	page 48
:TIMEout <time>	Sets termination timeout period.	page 49
:TIMEout?	Returns current termination timeout value.	page 49
:TRANsmit		
:AUTO 1 | 0 | ON | OFF	Sets transmit baud rate equal to receive baud rate.	page 49
:AUTO?	Returns state of transmit/receive baud rate linkage.	page 49
:BAUD <*baud_rate*>	Sets transmit baud rate.	page 49
:BAUD?	Returns Transmit baud rate.	page 49
:PACE		
[:PROTocol] XON | NONE	Enables/Disables transmit pacing (XON/OFF).	page 50
[:PROTocol]?	Returns current transmit pacing protocol.	page 50
SYSTem:ERRor?	Returns next error in queue.	page 50
SYSTem:VERSion?	Returns current SCPI version of driver.	page 50

Chapter 4
Register Programming Information

This chapter describes how to program the HP E2261A Quad RS-232 Interface M-Module at the register level in an HP E2251 Carrier installed in a VXIbus mainframe. Register programming is recommended only if you are unable to use the module's higher-level SCPI driver or VXI*plug&play* driver. The SCPI driver is documented in Chapter 2 of this manual. For information on the VXI*plug&play* driver, refer to the on-line help supplied on the CD ROM.

Note If you are using a high level driver, such as VXI*plug&play*, you do not need to know the register-based information in this chapter. Do not do register writes if you are also controlling the module by the high level driver. This is because the driver will not know the module state and an interrupt may occur causing the driver and/or command module to fail.

Module Description

Before considering the individual registers on the HP E2261A, you need to understand how the module's receiver and transmitter operate. This section also defines several terms used in this chapter. In this discussion, *receiver* is used to describe the HP E2261A receiving data from an external source and sending it to the host computer (VXI system Controller); transmitter describes the HP E2261A receiving data from the host computer and transmitting it to the external device. Figure 4-1 shows a simplified block diagram of the HP E2261A Quad RS-232 M-Module.

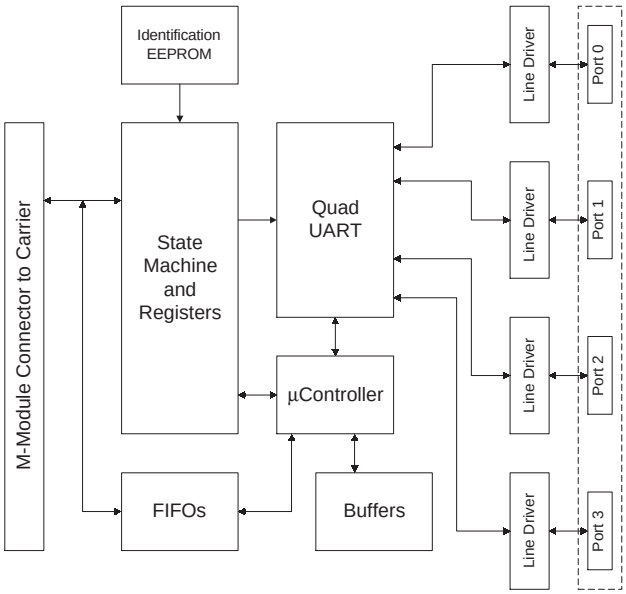


Figure 4-1. HP E2261A Simplified Block Diagram

At power-on or reset, the HP E2261A sets the default baud rate and data format for each port; both transmitter and receiver for each port is disabled. The host computer (VXI controller) must START/STOP specific port(s) to receive or transmit. The host computer can CLOSE a port without using the disable command. The difference between CLOSE and STOP is CLOSE clears the port's buffer.

Programming the Microcontroller

The host computer sends commands to and receives data from the microcontroller on the HP E2261A M-Module. Microcontroller register commands are described later in this chapter. The proper sequence to send a command to the HP E2261A microcontroller is:

1. Read the value of the CPRDY bit (bit 0) in the Command Status Register (at address 26_h) to verify the microcontroller is ready to receive a new command (bit value = 1). If bit value = 0 then the microcontroller is busy, do not send the command.

2. Write the parameter to the Parameter Register(s) 0/1 (addresses 22_h and 24_h respectively).

3. Write the command (an integer, listed in this chapter as a hexadecimal number) to the Command / Response Register (address 20_h). The command specifies the port. Writing to the Command Register signals the M-Module microcontroller to accept the command and parameter value.

4. Read the value of the CPRDY bit (bit 0) and the DONE bit (bit 7) in the Command Status Register (at address 26_h) to verify that the microcontroller has completed the command. Both bit values should be 1.

5. Read the CERR bit (bit 6) of the Command Status Register (address 26_h) to ensure an error did not occur. A bit value of 0 indicates an error did not occur.

6. Read the resulting value (if any) from the parameter registers.

Receiver Operation

The following discussion describes general receiver operation. If you are not using the supplied driver (either SCPI or VXI*plug&play*), then you should follow these general guidelines. Refer to Figure 4-2.

1. The host computer must open the port and enable the port to receive data. The host can send the Start Receiver Command ($2B_h$). If necessary, set the port's characteristics (baud rate, stop bits, handshake mode, etc.). After the receiver is enabled, the port begins waiting for a start bit from the device.

2. As data is received, the UART on the M-Module stores the data in the buffer and continues receiving data. Each port has a 16 kByte buffer.

3. When the amount of data in the buffer is greater than the defined BLOCK size, or if the amount of data in the buffer is less than a full BLOCK and the BLOCK timeout occurs, the M-Module microcontroller moves the data from the buffer to the Receiver FIFO memory — if the FIFO is empty. The microcontroller then sets the Receiver Ready Bit (bit 1) in the Port's Interrupt Status/Control Register (Register address 38_h, $3A_h$, $3C_h$, $3E_h$).

4. If the appropriate registers are set the microcontroller interrupts the host controller indicating that data is in the receiver FIFO memory. Note that each port has the same interrupt priority. If multiple ports are receiving data the host must determine which port interrupted by polling the individual Port Interrupt Status/Control Registers (Register address 38_h, $3A_h$, $3C_h$, $3E_h$).

5. When the host receives the interrupt it should read all of the data from the receiver FIFO. Read the data through the Port Transmit/Receive Register (Register Address 40_h, 42_h, 44_h, or 46_h). The host can determine the FIFO status in two ways; either read the exact number of bytes in the BLOCK or by monitoring the FIFO Status Register (register address 36_h).

6. After the data in the FIFO has been read and the FIFO is empty, the host computer must wait until an another receiver interrupt occurs.

7. When the host computer is finished receiving all the data, the host disables the port by either sending the Close Port Command (32_h) or the Stop Receiver Command ($2C_h$). These commands terminate receiver operation immediately. Note: the Close command clears the receiver buffer; characters in the FIFO are still available.

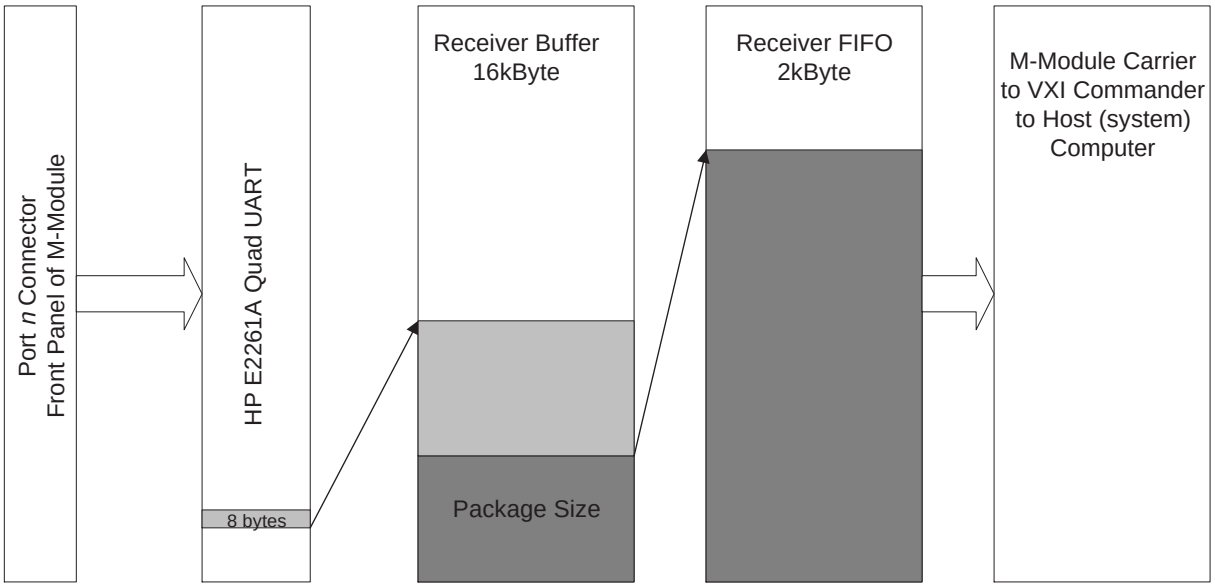


Figure 4-2. Receiver Data Flow

Transmitter Operation

The following discussion describes general transmitter operation. If you are not using the supplied driver (either SCPI or VXI*plug&play*), then you must follow these guidelines. Refer to Figure 4-3.

1. The host computer must open and enable the port to transmit data by sending the Open Port Command (31_h) and the Start Transmitter Command ($2D_h$).

2. The host then sends data to be transmitted to the Port Transmit/Receive Register (Register Address 40_h, 42_h, 44_h, or 46_h). This automatically sends the data to the Transmit FIFO (See Figure 4-3). If the Transmit FIFO changes from more-than-half-full to half-full, an interrupt occurs, the host (system) computer can then send up to 1kBytes of data. If the transmitter FIFO changes from non-empty to empty, an interrupt occurs, and the host can send up to 2kBytes of data.

   Note: the host computer can also monitor the FIFO Status Register to determine how full the Transmit FIFO is.

3. The host computer disables the transmitter by sending the Stop Transmitter Command ($2E_h$) or the Close Port Command (32_h). If the Close Port Command is executed, all of the data stored in the FIFO is lost. New data cannot be loaded in the FIFO if the port is disabled.

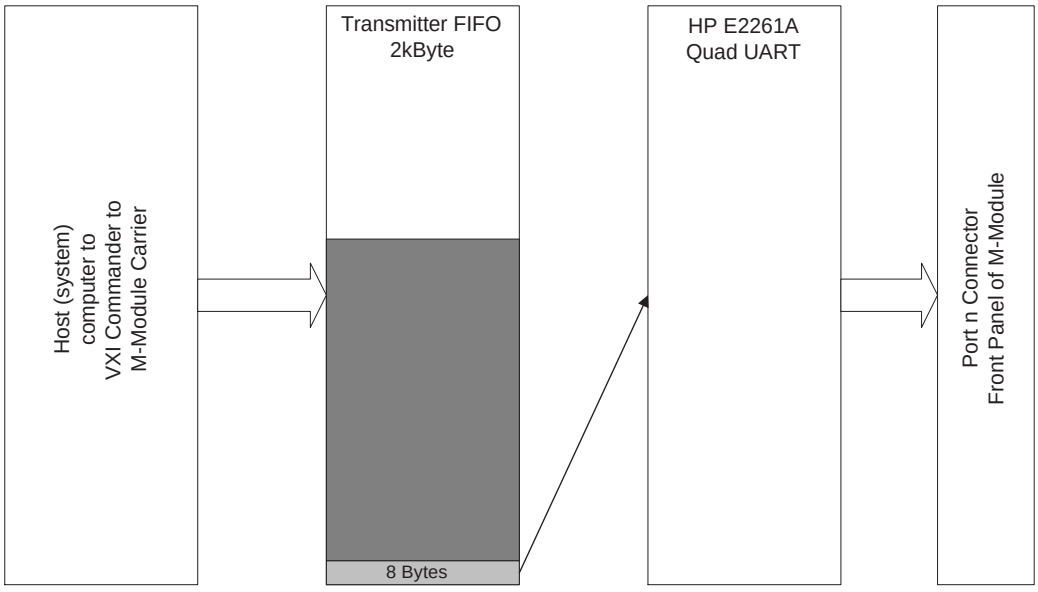


Figure 4-3. Transmitter Data Flow

Register Addressing in the VXIbus Environment

Logical address

Each module in a VXIbus system, whether VXI or M-Module, must have a unique logical address. The HP E2251 Carrier provides a logical address for each installed M-Module. Refer to the *HP E2251 Carrier Installation and Wiring Manual* and the *HP E2251 User's Manual* for details. If you are using a different carrier, refer to that carrier's documentation for register-based addressing information.

A16/A24 Memory Mapping

The VXI Specification allows for only 64 bytes of address space in A16 memory. However, the M-Module specification defines 256 bytes of address space. To resolve this conflict, the HP E2251 Carrier provides two memory segments for each installed M-Module. The first is in A16 memory space and contains the standard VXI registers. The second memory segment is in the A24 memory space and contains all other M-Module registers (these registers are described later in this chapter). Figure 4-4 shows the A16/A24 memory mapping for a typical M-Module.

Note

The M-Module's ID word (from the ID EEPROM) is mapped into the VXI Manufacturer ID register at address 00_h (A16 memory) and the M-Module's VXI device Type word is mapped into the VXI Device Type Register at address 02_h (A16 memory).

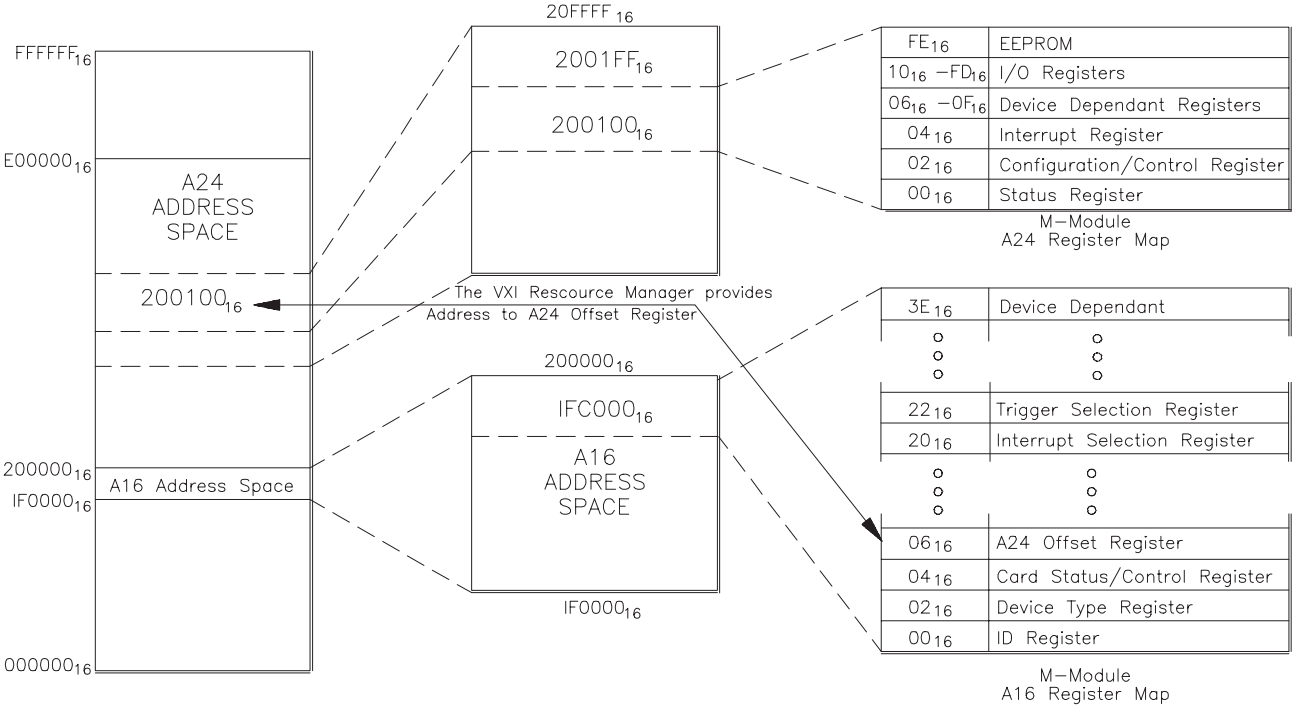


Figure 4-4. Typical M-Module A16/A24 Memory Mapping in an HP E2251 Carrier

Determining a Module's A16 Base Address

To access a register in A16 memory, you must specify a hexadecimal or decimal register address. The address consists of a base address plus a register offset. The A16 base address depends on whether or not you are using an HP E1406 Command Module.

When using an HP E1406 Command Module, the base address is computed as:

$1FC000 + (LADDR_h * 40_h)$
or (decimal)
2,080,768 + (LADDR * 64)

Where:
$1FC000_h$ (2,080,768) is the starting location of the VXI A16 addresses,
LADDR is the module's logical address
40_h (64) is the number of address bytes per register-based device.

For example, if the M-Module has a logical address of 78_h (120) the A16 base address becomes:

$1FC000h + (78_h * 40_h) = 1FC000_h + 1E00_h = 1FDE00_h$
or (decimal)
2,080,768 + (120 * 64) = 2,080,768 + 7680 = 2,088,448

When an HP Command Module is not a part of your VXIbus system (see Figure 4-4), the M-Module's base address is computed as:

$C000_h + (LADDR_h * 40_h)$
or (decimal)
49,152 + (LADDR * 64)

Where:
$C000_h$ (49,152) is the starting location of the register addresses,
LADDR is the module's logical address
40_h (64) is the number of address bytes per VXI device.

For example, if the M-Module has a logical address of 78_h (120) the A16 base address becomes:

$C000_h + (78_h * 40_h) = C000_h + 1E000_h = DE00_h$
or (decimal)
49,152 + (120 * 64) = 49,152 + 7680 = 56,832

Addressing A16 Register

As shown in Figure 4-4, VXI register for an M-Module (installed in an HP E2251 Carrier) are mapped into A16 address space. To access one of these registers, add the A16 base address to the register offset. For example, an M-Module's VXI Status/Control register has an offset of 04_h. To access this register (assuming the system does not have an HP E1406 Command Module), use the register address:

$1FDE00_h + 04_h = 1FDE04_h$
or (decimal)
2,088,488 + 4 = 2,088,452

Addressing A24 Registers

As shown in Figure 4-4, most of the registers for an M-Module are mapped into A24 address space. To access one of these registers:

1. Obtain the A24 base address by reading the A24 Offset register (06_h) in A16 memory.

2. Add the A24 base address to the A24 register offset.

For example, if the A24 base address is 200100_h, to access the Command/Response Register at address 20_h:

$200100_h + 20_h = 200120_h$
or (decimal)
2,097,408 + 32 = 2,097,440

A16 Memory Registers

When the HP E2261A Quad RS-232 Interface M-Module is used with an HP E2251 Carrier, the Carrier provides the following registers in A16 VXI memory:

- VXI ID Register at address 00_h
- VXI Device Type Register at address 02_h
- VXI Status Control Register at address 04_h
- VXI Offset Register at address 06_h

For additional information on these registers and a detailed description of the bit values, refer to the HP E2251 Carrier User's Manual.

VXI ID Register

The ID Register is a read only register at address 00_h (MSB) and 01_h (LSB). The default value of this register for the HP E2261A Quad RS-232 Interface M-Module is $CFFF_h$.

b+00$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	Device Class		Addr. Space		Manufacturer ID (4095 decimal for HP M-Modules)											

VXI Device Type Register

The Device Type Register is a read only register at address 02_h (MSB) and 03_h (LSB). The default value of this register for the HP E2261A Quad RS-232 Interface M-Module is $F25A_h$.

b+02$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	Required Memory				M-Module Model Code											

VXI Status/Control Register

The Status/Control Register is a read/write register (address 04_h and 05_h) that controls the module and indicates its status.

b+04$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write (Control)	A24 Enable	Reserved													Sysfail Inhibit	Reset
Read (Status)	A24 Active	MODID*	M-Module Device Dependent										Ready	Passed	Device Dependent	

- A24 Enable. Writing a 1 to this field enables access to the devices A24 registers.
- Sysfail Inhibit. Writing a 1 disables the M-Module from driving the SYSFAIL* line.
- Reset. Writing a 1 to this field forces the M-Module to reset.
- A24 Active. A 1 in this field indicates the M-Module's registers in A24 memory space can be accessed. Default = 1.
- MODID*. A 1 in this field indicates that the M-Module is not selected via the P2 MODID line. A 0 indicates the M-Modules is selected by a high state on the P2 MODID line.
- Ready. A 1 in this field indicates that the M-Module is ready to accept commands. A 0 indicates the M-Module is busy and not ready to accept commands.
- Passed. A 1 in this field indicates the M-Module passed its self test successfully. A zero indicates the M-Module is either executing or has failed its self test.

VXI Offset Register

The Offset Register (address 06_h and 07_h) contains the value of the base address for accessing registers in the A24 address space. There is no default value for this register.

b+06$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	A24 Space Base address for those M-Modules needing A24 memory															
Read	A24 Space Base address for M-Modules needing A24 memory															

Interrupt Selection Register

The Interrupt Selection Register (base + 20_h) specifies which VXI interrupt line the M-Module will use. M-Modules may generate interrupts to indicate that a SCPI command has completed. These interrupts are sent to and acknowledged by the HP Command Module or other system controller via one of seven VXI backplane interrupt lines. Different controllers treat the interrupt lines differently, and you should refer to your controller's documentation to determine how to set the interrupt level. HP Command Modules configured as VXI Resource Managers treat all interrupt lines as having equal priority. For interrupters using the same line, priority is determined by which mainframe slot they are installed in; lower-numbered slots have higher priority than higher-numbered slots. HP Command Modules service line 1 by default, so it is normally correct to leave the interrupt level set to the factory default of IRQ1.

b+20$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Reserved												INTC	VXI Interrupt Line		
Read (default value)	Reserved												INTC 1	VXI Interrupt Line 0	0	1

If your controller's documentation instructs you to change the interrupt level, you need to specify the level in the VXI Interrupt Selection Register. To cause the M-Module to interrupt on one of the VXI interrupt lines, write to the appropriate bits (refer to table below). To disable the module's interrupt, set the bits to 000. Selecting other than the default interrupt line 1 is not recommended. Reading the default value of this register returns the value CFF9$_h$.

Bits 2 - 0	Selected Interrupt Line
000	NONE (Interrupt Disabled)
001	IRQ1 (default)
010	IRQ2
011	IRQ3
100	IRQ4
101	IRQ5
110	IRQ6
111	IRQ7

M-Module specifications define three types of interrupts. The INTC bit (bit 3) determines which M-Module interrupt style is supported. If INTC is set to a 0 (default), the M-Module supports interrupt types A and B. If INTC is set to a 1, the M-Module supports interrupt type C (this is the default).

Type A Interrupts The interrupting M-Module removes the interrupt request upon a register access (software method) to the interrupting M-Module (such as reading the Status Register). DTACK* is not asserted during interrupt acknowledge.

Type B Interrupts The interrupting M-Module removes the interrupt request via a hardware method (on IACK* going low) but provides no vector information for the interrupt. This is the same as Type C interrupts except that no vector is supplied and DTACK* is not asserted.

Type C Interrupts The interrupting M-Module removes the interrupt request via a hardware method and provides an interrupt vector on the data bus and DTACK* is asserted during the interrupt acknowledge cycle. The M-Module removes the interrupt request by IACK* going low.

In VXI specifications however, only two types of interrupts are defined; RORA (Release on Register Access) and ROAK (Release on Acknowledge). The HP E2251 Carrier converts M-Module Type A interrupts to RORA and Types B and C interrupts to ROAK (default).

RORA Interrupts The interrupting device provides its logical address on the data bus (D0 - D7) during the interrupt acknowledge cycle that was initiated in response to its interrupt request.. It does not remove the interrupt request until its Status/Control register is accessed.

ROAK Interrupts The interrupting device removes the interrupt request upon the presence of a properly addressed interrupt acknowledge cycle and provides its logical address on the data bus (D0 - D7). A cause/status byte is also placed on the data bus (D15 - D8)

A24 Memory Registers

The HP E2261A has the following registers in A24 memory:

- Status Register at address 00_h
- Control Register at address 02_h
- Interrupt Vector Register at address 04_h
- Reserved addresses 06_h - $1E_h$
- Command / Response Register at address 20_h
- Parameter Register 0 at address 22_h
- Parameter Register 1 at address 24_h
- Command Status Register at address 26_h
- Interrupt Generator Register for Serial Port 1 at address 28_h
- Interrupt Generator Register for Serial Port 2 at address $2A_h$
- Interrupt Generator Register for Serial Port 3 at address $2C_h$
- Interrupt Generator Register for Serial Port 4 at address $2E_h$
- Reserved addresses 30_h - 34_h
- FIFO Status Register at address 36_h
- Interrupt Status/Control Register for Serial Port 1 at address 38_h
- Interrupt Status/Control Register for Serial Port 2 at address $3A_h$
- Interrupt Status/Control Register for Serial Port 3 at address $3C_h$
- Interrupt Status/Control Register for Serial Port 4 at address $3E_h$
- Transmit/Receive Register for Serial Port 1 at address 40_h
- Transmit/Receive Register for Serial Port 2 at address 42_h
- Transmit/Receive Register for Serial Port 3 at address 44_h
- Transmit/Receive Register for Serial Port 4 at address 46_h

Status Register

The Status Register is a read only register at address 00_h (MSB) and 01_h (LSB) in A24 memory. The default value of this register 01_h.

b+00_h	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	Not used											IREQ3	IREQ2	IREQ1	IREQ0	CPRDY

- **CPRDY:** Command/Parameter Ready. A “1” in this bit indicates that the Command and Parameter registers are empty and available for the next command from the host controller. A “0” indicates that the DONE bit in the Command Status Register (address 26_h) may not be valid.
- **IREQ*n*:** Interrupt Request Status Bit. A “1’ indicates the respective port (1 through 4) has an interrupt request. There are seven interrupt sources for each port: FIFO decreased below half full, one pre-defined BLOCK received, Receiver time-out, Transmit FIFO became empty, an error occurred, the termination character is received, or a self-generated interrupt occurs. For detailed information, refer to the Port Interrupt Status/Control Register. Any interrupt source sets IREQn if that source is enabled. If Port*n* and the global system interrupt is enabled, a “1” in an any of these bits will generate an interrupt to the host controller.
- At power-on or reset, all bits are “0” except CPRDY which has a value of “1”.

Control Register

The Control Register is a read/write register at address 002_h in A24 memory. The default value of this register is 00_h.

b+02_h	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	Not used										IEN3	IEN2	IEN1	IEN0	IENABLE	SRESET

- IEN*n*: Enable Interrupt of Port *n*. The bit must be set to a “1” for that port to generate an interrupt to the host controller.
- IENABLE: Global Interrupt Request Enable. Bit must be set to a “1” for interrupts to occur.
- SRESET: Soft Reset. Writing a “1” to this bit followed by writing a “0” causes the module to its soft reset state.
- At power-on or reset, all bits are “0”.

A24 Interrupt Vector Register

The A24 Interrupt Vector Register is a read-only register at address 04_h in A24 memory. During an Interrupt Acknowledge Cycle or Read Interrupt Vector Register Cycle, the value of this register is placed on the VXI data bus and then cleared. The default value of this register is 0_h.

b+04$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	Not used												ICH3	ICH2	ICH1	ICH0

- After the host controller receives the Interrupt Vector Register value, it should read the corresponding Interrupt Status Register to determine the actual source/cause of the interrupt. For more information, refer to the Interrupt Status/Control Register.

- At power-on or reset, all bits are “0”.

Command / Response Register

The Command / Response Register is a read/write register at address 20_h in A24 memory. This register is used to exchange commands and data between the host controller and the microcontroller on the HP E2261A M-Module. The default value (at power-on) of this register is 00_h.

b+20$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Not used								Command							
Read	Not used								Command Response							

- This register is used with the Parameter Registers (addresses 22_h and 24_h). The Parameter Registers must be set up before sending a command value to the Command / Response Register. In the following command descriptions, they are called PARM0 and PARM1.

- The Command Status Register (address 26_h) acts as a handshake signal for the Command / Response Register. Writing a value to the Command / Response register clears the CPRDY bit (bit 0) in the Command Status Register indicating that the Command / Parameter registers are not ready for the next data. Usually, a “0” on CPRDY interrupts the M-Module microcontroller. After the microcontroller execute the command, it writes a response to the Response Register and Parameter Registers. After sending the response, the microcontroller sets CPRDY indicating it is ready for the next command.

The following pages describe the HP E2261A microcontroller commands, parameters, and results. In the description PARM0 represents Parameter 0 register (22_h) and PARM1 represents Parameter 1 register (24_h). The commands are listed in order of their hexadecimal command code.

The following command (hexadecimal) code includes two *xx*’s followed by the actual command code. The *xx*’s (binary) represent the port number: 00 is Port 1, 01 is Port 2, 10 is Port 3, and 11 is Port 4. For example, to set the Port 2 (01_{binary}) transmitter baud rate ($xx21_h$ or $xx100001_{binary}$), use the command: 01100001_{binary} or 61_h.

Query Test Value Command, *xx*00$_h$

This command is used for test and debugging. Set test values in PARM0 and PARM1 with the *Set Command Test Value Command* (20_h) and then read them back with this command. The test values have no special meaning and are any values you want to use.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: *testvalue1*;PARM1: *testvalue0*
Default/Reset Value: PARM0:55_h PARM1:AA_h

Query Transmitter Baud Rate Command, *xx*01$_h$

This command returns a code representing the current port's transmitter baud rate. The baud rate is returned in the Result Parameters register. Set the transmitter baud rate with the *Set Transmitter Baud Rate Command* (21_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: baud rate code;PARM1: none
Default/Reset Value: PARM0: $0B_h$ PARM1: 00_h

Baud Rate Code	Exact Baud Rate
00_h	75
01_h	110
02_h	38400
03_h	150
04_h	300
05_h	600
06_h	1200
07_h	2000
08_h	2400
09_h	4800
$0A_h$	1800
$0B_h$	9600
$0C_h$	19200

Query Receiver Baud Rate Command, *xx*02$_h$

This command returns a code representing the current port's receiver baud rate. The baud rate is returned in the Result Parameters register. The table above lists the baud rate code and the exact baud rate. Set the receiver baud rate with the *Set Receiver Baud Rate Command* (22_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: baud code;PARM1: none
Default/Reset Value: PARM0: $0B_h$ PARM1: 00_h

Query Data Transfer Parity Mode Command, *xx*03_h

This command returns a code representing the data transfer parity mode. To set the parity mode, use the *Set Data Transfer Parity Mode Command* (23_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: parity mode code;PARM1: none
Default/Reset Value: PARM0: 04_h PARM1: 00_h

Parity Mode Code	Parity Type		Parity Mode Code	Parity Type
00_h	Even		03_h	Force 1
01_h	Odd		04_h	None
02_h	Force 0			

Query Character Length Command, *xx*04_h

This command returns a code representing the character data length. To set the character data length, use the *Set Character Length Command* (24_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: character length code;PARM1: none.
Default/Reset Value: PARM0: 03_h PARM1: 00_h

Character Length Code	Character Length		Character Length Code	Character Length
00_h	5		02_h	7
01_h	6		03_h	8

Query Stop Bit Length Command, *xx*05_h

This command returns a code representing the stop bit length. To set the Stop Bit length, use the *Set Stop Bit Length Command* (25_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: stop bit length code;PARM1: none
Default/Reset Value: PARM0: 07_h PARM1: 00_h

Stop Bit Length Code	Stop Bit Length		Stop Bit Length Code	Stop Bit Length
00_h	0.563		08_h	1.563
01_h	0.625		09_h	1.625
02_h	0.688		$0A_h$	1.688
03_h	0.750		$0B_h$	1.750
04_h	0.813		$0C_h$	1.813
05_h	0.875		$0D_h$	1.875
06_h	0.938		$0E_h$	1.838
07_h	1.000		$0F_h$	2.000

Query RTS/CTS Mode Command, *xx*06_h

This command returns a code representing the RTS/CTS Mode. To set the RTS/CTS mode, use the *Set RTS/CTS Mode Command* (26_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: RTS/CTS Mode code;PARM1: CTS Monitor.
Default/Reset Value: PARM0: 00_h PARM1: 00_h

PARM0 Code	RTS/CTS Mode		PARM1 Code	CTS Monitor
00_h	Idle		00_h	CTS Monitor OFF
01_h	On		01_h	CTS Monitor ON
02_h	Off			
03_h	Standard			
04_h	IBfull			

Query DTR/DSR Mode Command, *xx*07_h

This command returns a code representing the DTR/DSR Mode. To set the DTR/DSR Mode, use the *Set DTR/DSR Mode Command* (27_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: DTR/DSR Mode code;PARM1: DSR Monitor.
Default/Reset Value: PARM0: 00_h PARM1: 00_h

PARM0 Code	DTR/DSR Mode		PARM1 Code	DSR Monitor
00_h	Idle		00_h	DSR Monitor OFF
01_h	On		01_h	DSR Monitor ON
02_h	Off			
03_h	Standard			
04_h	IBfull			

Query Pace Mode Command, *xx*08_h

This command returns a code representing the Pace (Xon/Xoff) Mode. To set the Xon/Xoff mode, use the *Set Pace Mode Command* (28_h).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Xon/Xoff Mode code;PARM1: none
.Default/Reset Value: PARM0: 00_h PARM1: 00_h

Code	Pace Mode
00_h	Tx & Rx Pace Off
01_h	Tx Pace On
02_h	RX Pace On
03_h	Tx & Rx Pace On

Query BLOCK Size Command, *xx*09$_h$

This command returns a code representing the BLOCK size. The PARM0 and PARM1 registers are both 16-bit registers. The lower 12 bits of each register express the BLOCK length from 1 to 2048 bytes. The upper four bits are not used. To set the BLOCK size, use the *Set BLOCK Size Command* (29$_h$).

Set Parameters: PARM0: none; PARM1: none
Result ParametersPARM0: BLOCK low code;PARM1: BLOCK high code.
Default/Reset Value: PARM0: 00$_h$ PARM1: 08$_h$ (2048 bytes)

Query Port Mode Command, *xx*0A$_h$

This command returns a code representing the port mode. to set the port mode, use the *Set Port Mode Command* (2A$_h$).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Port Mode code;PARM1: none.
Default/Reset Value: PARM0: 00$_h$ PARM1: 00$_h$

Code	RTS/CTS Mode		Code	RTS/CTS Mode
00$_h$	Normal		02$_h$	Local Loop
01$_h$	Auto-Echo		03$_h$	Remote Loop

Query Line Status Command, *xx*0B$_h$

This command returns a code representing the status of the individual RS-232 handshake lines and pacing controller information.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Line Status Code;PARM1: none.
Default/Reset Value: PARM0: 001100xx$_b$PARM1: 00$_h$

Bit	7	6	5	4	3	2	1	0
Signal Line	TOFF	Not Used	DTR	RTS	ROFF	Not Used	DSR	CTS

where:

TOFF = 0 means XOFF has not been sent (default), TOFF = 1 means that XOFF has been sent.
DTR = 0 means DTR is ON, DTR = 1 means DTR is OFF (default).
RTS = 0 means RTS is ON, RTS = 1 means RTS is OFF (default).
ROFF = 0 means that XON has been received IF XOFF has been received previously, or it means that XOFF has not yet been received (def), ROFF = 1 means that XOFF has been received.
DSR = 0 means DSR is ON, DSR = 0 means DSR is OFF (default).
CTS = 0 means CTS is ON, CTS = 1 means CTS is OFF (default).

Query Filled Number in Receiver FIFO Command, *xx*0C$_h$

This command returns an unsigned integer representing the amount of data in the receiver FIFO when the host received an interrupt.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: FIFO Low Byte;PARM1: FIFO High Byte (lower 3 bits)
Default/Reset Value: PARM0: 00_h PARM1: 00_h

Query Error Code Command, 0D$_h$

This command returns a code representing one or more errors that have occurred. Reading this register clears the register.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Error Code;PARM1: none.
Default/Reset Value: PARM0: 00_h PARM1: 00_h

Bit	7	6	5	4	3	2	1	0
Signal Line	Not Used	Frame Error occurred during receive	Parity Error detected	Overflow Error detected	Not Used	Receiver Buffer is full	Not Used	Not Used

Query Filled Number in Receiver Buffer Command, *xx*0E$_h$

This command returns an unsigned integer indicating the number of bytes that has received by the port. The value does not include data in the receive FIFO if the receive FIFO is empty. If the receive FIFO is not empty, then the number will include the BLOCK data which has been moved into the receive FIFO (that generated an interrupt).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Low Byte;PARM1: High Byte
Default/Reset Value: PARM0: 00_h PARM1: 00_h

Query Error Mode Command, *xx*13$_h$

This command returns a value representing the established error mode. For actual error code refer to the *Query Error Code Command* (0D$_h$) command. To set the error mode, refer to the *Set Error Mode Command* (33$_h$) command.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Error Mode;PARM1: none
Default/Reset Value: PARM0: 00$_h$ PARM1: 00$_h$

Error Mode Code	Error Mode
00$_h$	Ignore Errors
01$_h$	Acknowledge Errors & Stop

Query Start Threshold Command, *xx*14$_h$

This command returns the start threshold value which is used to control the buffer space where DTE requests DCE restart line transfer activity. To set the Start Threshold anywhere in the 16kbyte buffer, use the *Set Start Threshold Command* (34$_h$).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Low Code;PARM1: High Code
Default/Reset Value: PARM0: 00$_h$ PARM1: 20$_h$ (8k buffer is full)

Query Stop Threshold Command, *xx*15$_h$

This command returns the stop threshold value which is used to control the buffer space where DTE requests DCE stop line transfer activity. To set the Stop Threshold anywhere in the 16kbyte buffer, use the *Set Stop Threshold Command* (35$_h$).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Low Code;PARM1: High Code
Default/Reset Value: PARM0: 00$_h$ PARM1: 28$_h$ (10k buffer full)

Query Parity Check Mode Command, *xx*1A$_h$

This command returns the current Parity Check Mode for the serial port. To set the parity check mode, use the *Set Parity Check Mode Command* (3A$_h$).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: Parity Check Mode;PARM1: none
Default/Reset Value: PARM0: 01$_h$ PARM1: 00$_h$

Parity Check Code	Parity Check Mode
00$_h$	Off
01$_h$	On (default)

Set Command Test Value Command, *xx*20$_h$

This command is used for test and debugging. Set test values in PARM0 and PARM1 with this command and then read them back with the *Query Command Test Value Command* (00_h). The test values have no special meaning and are any values you want to use.

PARM0: testvalue0;PARM1: testvalue1
Result: PARM0: testvalue1;PARM0: testvalue0

Set Transmitter Baud Rate Command, *xx*21$_h$

This command sets the current port's transmitter baud rate. To read the current transmitter baud rate value, use the *Query Transmitter Baud Rate Command* (01_h).

Set Parameters: PARM0: baud rate code;PARM1: none
Default/Reset Value: PARM0: $0B_h$ PARM1: 00_h

Baud Rate Code	Exact Baud Rate
00_h	75
01_h	110
02_h	38400
03_h	150
04_h	300
05_h	600
06_h	1200
07_h	2000
08_h	2400
09_h	4800
$0A_h$	1800
$0B_h$	9600 (default)
$0C_h$	19200

Set Receiver Baud Rate Command, *xx*22$_h$

This command sets the current port's receiver baud rate. To read the current baud rate value, use the *Query Receiver Baud Rate Command* (02_h). See the *Set Transmitter Baud Rate Command* (21_h) for a list of baud rates and codes.

Set Parameters: PARM0: baud rate code;PARM1: none
Default/Reset Value: PARM0: $0B_h$ PARM1: 00_h

Set Data Transfer Parity Mode Command, *xx*23$_h$

This command sets the data transfer parity mode. To read the current parity mode, use the *Query Data Transfer Parity Mode Command* (03_h).

Set Parameters: PARM0: parity mode code;PARM1: none
Default/Reset Value: PARM0: 04_h (none) PARM1: 00_h

Parity Mode Code	Parity Type		Parity Mode Code	Parity Type
00_h	Even		03_h	Force 1
01_h	Odd		04_h	None
02_h	Force 0			

Set Character Length Command, *xx*24$_h$

This command sets the character data length. To read the current character data length, use the *Query Character Length Command* (04_h).

Set Parameters: PARM0: character length code;PARM1: none.
Default/Reset Value: PARM0: 03_h PARM1: 00_h

Character Length Code	Character Length		Character Length Code	Character Length
00_h	5		02_h	7
01_h	6		03_h	8

Set Stop Bit Length Command, *xx*25$_h$

This command sets the stop bit length. To read the current stop bit length, use the *Query Stop Bit Length Command* (05_h).

Set Parameters: PARM0: stop bit length code;PARM1: none
Default/Reset Value: PARM0: 07_h PARM1: 00_h

Stop Bit Length Code	Stop Bit Length		Stop Bit Length Code	Stop Bit Length
00_h	0.563		08_h	1.563
01_h	0.625		09_h	1.625
02_h	0.688		$0A_h$	1.688
03_h	0.750		$0B_h$	1.750
04_h	0.813		$0C_h$	1.813
05_h	0.875		$0D_h$	1.875
06_h	0.938		$0E_h$	1.838
07_h	1.000		$0F_h$	2.000

Note: upper four bits are reserved and must be set to 00_h.

Set RTS/CTS Mode Command, *xx*26$_h$

This command sets the RTS/CTS Mode. To read the current RTS/CTS mode, use the *Query RTS/CTS Mode Command* (06$_h$).

Set Parameters: PARM0: RTS/CTS Mode code;PARM1: CTS Monitor.
Default/Reset Value: PARM0: 00$_h$ PARM1: 00$_h$

PARM0 Code	RTS/CTS Mode		PARM1 Code	CTS Monitor
00$_h$	No Change		00$_h$	CTS Monitor OFF
01$_h$	On		01$_h$	CTS Monitor ON
02$_h$	Off			
03$_h$	Standard			
04$_h$	IBfull			

Note: PARM1 is only used when PARM0 is 00$_h$, 01$_h$, or 02$_h$.

Set DTR/DSR Mode Command, *xx*27$_h$

This command sets the DTR/DSR Mode. To read the current DTR/DSR mode, use the *Query DTR/DSR Mode Command* (07$_h$).

Set Parameters: PARM0: DTR/DSR Mode code; PARM1: none.
Default/Reset Value: PARM0: 00$_h$ PARM1: 00$_h$

PARM0 Code	DTR/DSR Mode		PARM1 Code	DSR Monitor
00$_h$	No Change		00$_h$	DSR Monitor OFF
01$_h$	On		01$_h$	DSR Monitor ON
02$_h$	Off			
03$_h$	Standard			
04$_h$	IBfull			

Note: PARM1 is only used when PARM0 is less than 03$_h$.

Set Pace Mode Command, *xx*28$_h$

This command sets the Xon/Xoff Mode. To read the current Xon/Xoff mode, use the *Query Xon/Xoff Mode Command* (08$_h$).

Set Parameters: PARM0: Pace Mode code;PARM1: none.
Default/Reset Value: PARM0: 00$_h$ PARM1: 00$_h$

Code	Pace Mode		Code	Pace Mode
00$_h$	Tx & Rx Pace Off		02$_h$	RX Pace On
01$_h$	Tx Pace On		03$_h$	Tx & Rx Pace On

Set BLOCK Size Command, *xx*29$_h$

This command sets the BLOCK size. The PARM0 and PARM1 registers are both 16 bit registers. The lower 12 bits of each register express the BLOCK length from 1 to 2048 bytes. Block Size should have a value greater than 1. The upper four bits are not used. To read the current BLOCK size, use the *Query BLOCK Size Command* (09_h).

Set Parameters PARM0: BLOCK low code;PARM1: BLOCK high code.
Default/Reset Value: PARM0: 00_h PARM1: 08_h (2kBytes)

Set Port Mode Command, *xx*2A$_h$

This command sets the port mode. To read the current Port mode, use the *Query Port Mode Command* ($0A_h$).

Set Parameters: PARM0: Port Mode code; PARM1: watchdog timer
0 = off, 1 = on.
Default/Reset Value: PARM0: 00_h PARM1: 01_h

Code	Port Mode		Code	Port Mode
00_h	Normal		02_h	Local Loop
01_h	Auto-Echo		03_h	Remote Loop

Where:

Normal In this mode, the HP E2261A has the transmitter and receiver operating independently.

Auto-Echo This mode sets the port to retransmit data automatically. The following conditions are true:

1. Received data is clocked and retransmitted on the TX output.
2. Receiver clock is used for the transmitter.
3. Receiver must be enabled, but the transmitter does not need to be enabled.
4. Received parity is checked but not regenerated for transmission.

Local Loop In this diagnostic mode, the following are true:

1. The transmitter output is internally connected to the receiver input.
2. The transmitter clock is used for the receiver.
3. The TxD output is held high.
4. The RxD input is ignored.
5. Both transmitter and receiver must be enabled.
6. The host computer to the HP E2261A continues normally.

Remote Loopback In this mode, the following are true:

1. Received data is reclocked and transmitted on the TxD output.
2. The receiver clock is used for the transmitter.
3. The received data is not transferred to the host, and error status conditions are inactive.
4. The received parity is not checked and is not regenerated for transmission.
5. The receiver must be enabled, but the transmitter does not need to be enabled.
6. Character framing is not checked, and the stop bits are retransmitted as received.

Note: Be careful when switching between modes. The selected mode is activated or deactivated immediately upon mode selection, even if this occurs during a received or transmitted character. However, when switching out of Auto-Echo or Loop Back modes; if the mode change occurs immediately after a stop bit, and the transmitter is enabled, the transmitter will remain in Auto-Echo mode until the entire stop bit is retransmitted.

Start Receiver Command, *xx*2B$_h$

This command causes the port to start receiving data.

Set Parameters: PARM0: 0; PARM1: none

Stop Receiver Command, *xx*2C$_h$

This command stops the receiver immediately. Characters in the receiver FIFO and receiver buffer are available for transfer to the host computer.

Set Parameters: PARM0: 0; PARM1: none

Start Transmitter Command, *xx*2D$_h$

This command causes the transmitter to start transmitting data from its buffer.

Set Parameters: PARM0: 0;PARM1: none

Stop Transmitter Command, *xx*2E$_h$

This command causes the transmitter to stop transmitting.

Set Parameters: PARM0: 0;PARM1: none

Clear Receiver Buffer Command, *xx*2F$_h$

This command clears the receiver buffer but not the receiver FIFO memory.

Set Parameters: PARM0: 0;PARM1: none

Clear Transmitter FIFO Command, *xx*30$_h$

This command clears the transmitter FIFO memory. The host computer should stop sending data to the transmitter FIFO.

Set Parameters: PARM0: 0;PARM1: none

Open Port Command, *xx*31$_h$

This command sets the active to its default conditions. To close the current port, use the *Close Port Command* (32$_h$).

Set Parameters: PARM0: Open Code;PARM1: none
Default/Reset Value: PARM0: 00$_h$ PARM1: 00$_h$

Open Code	Meaning
00$_h$	Open single port
01$_h$	Open all four ports

The default conditions are:

Transmitter Characteristics	Setting		Receiver Characteristics	Setting
Transmitter	Disabled		Receiver	Disabled
Transmitter Baud Rate	9600		Receiver Baud Rate	9600
Character Length	8		Block Timeout	Enabled
Stop Bit Length	1		Error Mode	Ignore
Parity Mode	None		Start Threshold	8kBytes
Handshake	Off		Stop Threshold	10kBytes
RTS/CTS	Off		Parity Check	On
DTR/DSR	Off		Start Threshold	20$_h$
BLOCK Size	2048		Stop Threshold	28$_h$
Channel Mode	Normal			

Close Port Command, *xx*32$_h$

This command closes the port immediately. After the command executes, the selected port(s) are set to disabled transmitter, disabled receiver, receiver buffer and transmitter FIFO are cleared. To open a port and set it to its default conditions, use the *Open Port Command* (31_h).

Set Parameters: PARM0: Close Code; PARM1: none
Default/Reset Value: PARM0: 00_h PARM1: 00_h

Open Code	Meaning
00_h	Close single port
01_h	Close all four ports

Set Error Mode Command, *xx*33$_h$

This command sets the error mode. For actual error code refer to the *Query Error Code Command* ($0D_h$) command. To read the current error mode, use the *Query Error Mode Command* (13_h). In the Ignore mode, the HP E2261A continues activity when it detects an error; it will generate an interrupt to the host computer. In the STOP mode, an interrupt is generated and the HP E2261A stops receiver activity.

Set Parameters: PARM0: Error Mode;PARM1: none
Default/Reset Value: PARM0: 00_h PARM1: 00_h

Error Mode Code	Error Mode
00_h	Ignore
01_h	Acknowledge Errors & Stop

Set Start Threshold Command, *xx*34$_h$

This command sets Start Threshold value which controls the buffer space. When the number of bytes in the receiver buffer drops to or below the Start Threshold, and some form of pacing has been set, the HP E2261A indicates it is ready to receive. The Start threshold must be less than the Stop Threshold. To read the current Start Threshold, use the *Query Start Threshold Command* (14_h).

Set Parameters: PARM0: Start Low Code ;PARM1: Start High Code
Default/Reset Value: PARM0: 00_h PARM1: 20_h (8kbyte space avail.)

Set Stop Threshold Command, *xx*35$_h$

This command sets the Stop threshold value which controls the buffer space. When the number of characters in the receiver buffer goes above the Stop Threshold value, and some form of pacing has been set, the HP E2261A indicates it is not ready to receive data. You can set Stop Threshold anywhere in the 16kbyte buffer space, but it must never be equal to 0. The Start threshold must be less than the Stop Threshold. To read the current Stop Threshold, use the *Query Stop Threshold Command* (15_h).

Set Parameters: PARM0: Stop Low Code ;PARM1: Stop High Code
Default/Reset Value: PARM0: 00_h PARM1: 28_h (10kByte space avail.)

Set Parity Check Mode Command, *xx*3A$_h$

This command sets the parity check mode. To read the current parity check mode, use the *Query Parity Check Mode Command* ($1A_h$).

Set Parameters: PARM0: Parity Check Mode;PARM1: none
Default/Reset Value: PARM0: 01_h PARM1: 00_h

Parity Check Code	Parity Check Mode
00_h	Off
01_h	On

Query FIFO Depth Command, 40$_h$

This command returns the FIFO size used in the transmitter/receiver port.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: FIFO depth code;PARM1: none
Default/Reset Value: PARM0: 22_h PARM1: 00_h

Bit	7	6	5	4	3	2	1	0
Signal Line	Transmitter FIFO (in kBytes)				Receiver FIFO (in kBytes)			

Query Firmware Version Command, 80$_h$

This command returns the current firmware version number.

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: version number;PARM1: none

Query Self Test Result Command, $C0_h$

This command retruns an unsigned integer with the results of the self test. To initiate a self test, use the *Start Self Test Command* ($E0_h$).

Set Parameters: PARM0: none; PARM1: none
Result Parameters: PARM0: self test code;PARM1: none

Bit	7	6	5	4	3	2	1	0
Error	Port 4	Port 3	Port 2	Port 1	RAM 4	RAM 3	RAM2	RAM1

where: a “0” in Port *n* indicates that port’s receiver and transmitter work in auto-echo mode; a “1” indicates a failure. A “0” in RAM *n* indicates that port’s buffer is ok; a “1” indicates a failure.

Start Self Test Command, $E0_h$

This command initiates the internal self test. The test mode is set by the Parameter Register (a “1” means to test that port, a “0” means do not test that port). To read the results of the self test, use the *Query Self Test Result* Command ($C0_h$).

Set Parameters: PARM0: self test code;PARM1: none

Bit	7	6	5	4	3	2	1	0
Error	N/A	N/A	N/A	N/A	Port 4	Port 3	Port 2	Port 1

Parameter Register 0
Parameter Register 1

Parameter Registers 0 and 1 (addresses 22_h and 24_h respectively) are read/write registers. They are used to hold the command parameters (PARM0 and PARM1 respectively) transferred between the host controller and the M-Module microcontroller. The default value of this register is 00_h.

b+22h b+24h	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Not used								Command Parameter							
Read	Not used								Result of Command							

- The Parameter Registers must be set up before writing a command to the Command / Response Register (address 20_h).
- When an 8-bit parameter is required, use only Parameter Register 0. When a 16-bit parameter is required, Parameter Register 0 is used as bits 7 - 0; Parameter Register 1 is used as bits 15 - 8.
- Refer to the Command / Response Register for a list of commands and the appropriate parameters.
- At power-on / reset, all bits are set to 0.

Command Status Register

The Command Status Register is a read-only register at address 26_h. The register indicates the status of the HP E2261A M-Module microcontroller. The default value (at power-on) of this register is 01_h.

b+26h	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Not Used															
Read	Not used								DONE	CERR	Not Used	URDY	UPASS	Not Used	RRDY	CPRDY

- **CPRDY:** Command/Parameter Register Ready. A “1” indicates the Command and Parameter Registers are empty and available for the next command to be written to the M-Module microcontroller. A “0” indicates that the DONE bit is not valid. This bit is set by the microcontroller, and cleared in hardware by writing to the Command/ Response Register (20_h).
- **RRDY:** Response Ready. A “1” in this bit indicates that the Response Register is valid. This bit is set and cleared by the microcontroller.
- **UPASS:** Microcontroller Pass. A “1” in this bit indicates the M-Module microcontroller has successfully completed its built-in self test.
- **URDY:** Microcontroller Ready. A “1” in this bit indicates the HP E2261A microcontroller is ready to interact with the VXI backplane.
- **CERR:** Command/Parameter Error. A “1” in this bit indicates an error occurred while communicating with the M-Module.
- **DONE:** Done. A “1” in this bit indicates the microcontroller has completed processing and the other Command Status Register bits are valid. If the CPRDY bit is 0, this bit may not be valid.
- **RSV:** Reserved Bits
- At power-on, all bits are 0 except for CPRDY which is 1.

Interrupt Generation Registers

The four Interrupt Generation Registers are read/write registers at addresses 28_h for Port 1, $2A_h$ for Port 2, $2C_h$ for Port 3, and $2E_h$ for Port 4.

b+28h b+2Ah b+2Ch b+2Eh	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Not Used															Int
Read	Not used															Int

After the host computer writes to this register, a self-generated interrupt occurs. Refer to the Port Interrupt Status/Control Register, Bit 7.

FIFO Status Register

The FIFO Status Register is a read-only register at address 36_h. The register indicates the status of the four Port FIFO memories. The default value (at power-on) of this register is 00_h.

b+36h	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Not Used															
Read	Not used								Port 4 RCV	Port 4 XMIT	Port 3 RCV	Port 3 XMIT	Port 2 RCV	Port 2 XMIT	Port 1 RCV	Port 1 XMIT

- **Port *n* RCV:** A 0 in these bits indicates the port's receive FIFO buffer is empty.
- **Port *n* XMIT:** A 1 in these bit indicates the port's transmit buffer is half full.
- When the transmit buffer is not half full (a "0" in the Port n XMIT bits), the host controller can write up to 1kByte to the FIFO.
- The host controller can read data from a port only when the port's Port *n* RCV bit is a 1 indicating data is in the FIFO.

Port Interrupt Status / Control Registers

The Interrupt Status / Control registers are read/write registers at addresses 38_h for Port 1, $3A_h$ for Port 2, $3C_h$ for Port 3, or $3E_h$ for Port 4. If a port has an interrupt request, the host controller can read the respective register to determine the cause of the interrupt. There are seven possible causes of interrupts. The default value (at power-on) of these registers is 1_h.

b+38$_h$ b+3A$_h$ b+3C$_h$ b+3E$_h$	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Write	Not Used								EX1	Not Used	END	ERR	ETE	ERTO	ERF	EHF
Read	Not Used								EX1	Not Used	END	ERR	TE	RTO	RF	HF

Read Bits:

- **END:** A “1” in this bit indicates that the termination character has been received.
- **ERR:** Error Source. A “1” in this bit indicates an error has occurred with receiving data on the port.
- **EX1:** Self-Generation Interrupt. A “1” in this bit indicates that the host computer has generated a self-generation interrupt for the port. This interrupt is used to initialize a data transmit or receive by the driver with interrupt method. Refer to the Interrupt Generation Registers.
- **HF:** Transmit Half Full. A “1” in this bit indicates the transmit FIFO buffer has decreased to below half full. In this condition, the host controller can send up to 1kBytes to the transmit buffer.
- **RF:** Receive Filled BLOCK. A “1” in this bit indicates the receive buffer for this channel is filled with a pre-defined amount of data. The host controller can read the data from the FIFO buffer.
- **RTO:** Receive Time Out. A “1” in this bit indicates the microcontroller has not received data for a pre-defined time after the last BLOCK of data was sent to the FIFO. After receiving this interrupt request, the host controller should verify the actual data length or read the data via the FIFO Status Register.
- **TE:** Transmit Error. A “1” in this bit indicates that the port’s transmit FIFO has become empty. This means the host controller can send at least 2048 bytes of data to the transmit FIFO without the FIFO becoming full.
- **RSV:** Reserved.

Write Bits:

- EERR: When set to a “1”, this bit enables the Error Interrupt for the

port.

- ETE: When set to a “1”, this bit enables the Transmit FIFO Empty Interrupt for the port.
- ERT: When set to a “1”, this bit enables the Receive Time-out Interrupt.
- ERF: When set to a “1”, this bit enables the Receive FIFO Filled Interrupt for the port.
- ETF: When set to a “1”, this bit enables the Transmit FIFO Half Full Interrupt for the port.
- Ex1: Enable the self-Generation interrupt for the port.

Always read the Interrupt Vector Register (at address 04_h) to clear all sources of interrupts. After one interrupt source is disabled, a later interrupt request may be pending. That is, if the interrupt source is later enabled again, the interrupt request may immediately generate an interrupt unless Interrupt Status Register is read first.

Port Transmit / Receive Registers

The four port Transmit / Receive registers are read/write registers addresses 40_h for Port 1, 42_h for Port 2, 44_h for Port 3, and 46_h for Port 4.

b+40$_h$ b+42$_h$ b+44$_h$ b+46$_h$	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Write	Not Used								Transmit Data Byte							
Read	Not Used								Receive Data Byte							

- This register can be read only when the BHE*n* bit in the FIFO Status Register (at address 36_h) is a “1” *OR* the RF or RT0 or END bits of the Port Interrupt / Status Register for the port (addresses 38_h for Port 1, $3A_h$ for Port 2, $3C_h$ for Port 3, or $3E_h$ for Port 4) is set to a “1”. If the host controller is set for interrupts, it should read all of the data in the Receive FIFO of the port when an interrupt is generated by RF or RTO.

- Up to 1kBytes of data can be written to this register when BHF*n* bit in the FIFO Status Register (at address 36_h) is a “0” or up to 2kBytes can be written if the TE bit of the Port Interrupt / Status Register for the port (addresses 38_h for Port 1, $3A_h$ for Port 2, $3C_h$ for Port 3, or $3E_h$ for Port 4) is set to a “1”.

ID EEPROM Register

The ID EEPROM Register allows you to access the contents of the ID EEPROM. The ID EEPROM contains sixty-four 16-bit words of M-Module ID data and VXI M-Module data.

Note

This register is intended to be used by the higher-level software driver. The software driver must perform a series of many reads and writes to this register to perform the required functions within the EEPROM. **When register programming, it is much easier to read the module ID data from the VXI registers (A16 memory area) instead of reading the ID**

EEPROM Register. A16 addressing is discussed earlier in this chapter. Do NOT attempt to read the ID EEPROM. Do not attempt to read the EEPROM Registers.

b+80$_h$ - b+FE$_h$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Unused															
Read (default value)														ECS	ESK	EDIO

Bit Definitions

EDIO -- EEPROM Serial Data.

ESK -- EEPROM Shift Clock

ECS -- EEPROM Chip Select

Table 4-1. ID EEPROM Contents

Word Number	Description	ID EEPROM Contents
0	Sync Code	5346$_h$
1	M-Module Number (binary code)	067D$_h$
2	Revision Number (binary code)	0001$_h$
3	Module Characteristics	1868$_h$
4 - 7	Reserved	0000$_h$
8 - 15	User Defined	0000$_h$
16	VXI Sync Code	ACBA$_h$
17	VXI ID	CFFF$_h$ (Hewlett Packard)
18	VXI Device Type	F25A$_h$
19 - 31	Reserved	0000$_h$
32 - 63	Not Used	0000$_h$

Program Example

The following C language program demonstrates how to program at the register level. The program reads the ID, Device Type, Status, and A24 Offset Registers. This program was written and tested in Microsoft Visual C++ but should compile under any standard ANSI C compiler.

To run this program you must have the HP SICL library, the HP VISA library, an HP-IB interface module installed in your PC, and an HP E2406 Command Module.

```
#include <visa.h>
#include <stdio.h>
#include <stdlib.h>

ViSession viRM,m_mod;
int main()
{

unsigned short id_reg,dt_reg ;                  /* ID & Device Type Registers */
unsigned short stat_reg, a24_offset ;           /* Status & A24 Offset Regs*/
unsigned short cmd_stat_reg;                    /* Command Status Register */
unsigned short value=0, done=0;

ViStatus errStatus;                             /*Status from each VISA call*/

     /* Open the default resource manager  */
errStatus = viOpenDefaultRM ( &viRM);
if (VI_SUCCESS > errStatus){
   printf("ERROR: viOpenDefaultRM() returned 0x%x\n",errStatus);
   return errStatus;}

     /* Open the M-Module instrument session ; Logical Address = 8 */
errStatus = viOpen(viRM,"GPIB-VXI0::8",VI_NULL,VI_NULL,&m_mod);
if (VI_SUCCESS > errStatus){
   printf("ERROR: viOpen() returned 0x%x\n",errStatus);
   return errStatus;}

     /* read  and print the module's ID Register  */
errStatus = viIn16(m_mod,VI_A16_SPACE,0x00,&id_reg);
if (VI_SUCCESS > errStatus){
   printf("ERROR: viIn16() returned 0x%x\n",errStatus);
   return errStatus;}
printf("ID register = 0x%4X\n", id_reg);

     /* read  and print the module's Device Type Register   */
errStatus = viIn16(m_mod,VI_A16_SPACE,0x02,&dt_reg);
if (VI_SUCCESS > errStatus){
   printf("ERROR: viIn16() returned 0x%x\n",errStatus);
   return errStatus;}
printf("Device Type register = 0x%4X\n", dt_reg);
```

```
    /* read and print the module's Status Register  */
errStatus = viIn16(m_mod,VI_A16_SPACE,0x04,&stat_reg);
if (VI_SUCCESS > errStatus){
   printf("ERROR: viIn16() returned 0x%x\n",errStatus);
   return errStatus;}
printf("Status register = 0x%hx\n", stat_reg);

    /* read and print the module's A24 Offset Register  */
errStatus = viIn16(m_mod,VI_A16_SPACE,0x06,&a24_offset);
if (VI_SUCCESS > errStatus){
   printf("ERROR: viIn16() returned 0x%x\n",errStatus);
   return errStatus;}
printf("A24 Offset register value = 0x%hx\n", a24_offset);

    /*The following sequence sets Port 1 TX Baud to 1900 Baud*/
while (value = 0)  {                          /* is module ready? */
errStatus = viIn16(m_mod,VI_A16_SPACE,0x26,&cmd_stat_reg);
value = cmd_stat_reg & 0376;
}
    /* Send baud rate code to Parameter Register 0 */
errStatus = viOut16(m_mod,VI_A16_SPACE,0x22,0x0C);
    /* Send command to Command/Response Register */
errStatus = viOut16(m_mod,VI_A16_SPACE,0x20,0x21);
    /* Is module done? */
while (done = 0);  {
errStatus = viIn16(m_mod,VI_A16_SPACE,0x26,&cmd_stat_reg);
done = cmd_stat_reg & 0201;
}

    /* Close the M-Module Instrument Session */
errStatus = viClose (m_mod);
if (VI_SUCCESS > errStatus) {
   printf("ERROR: viClose() returned 0x%x\n",errStatus);
   return 0;}

    /* Close the Resource Manager Session */
errStatus = viClose (viRM);
   if (VI_SUCCESS > errStatus) {
   printf("ERROR: viClose() returned 0x%x\n",errStatus);
return 0;}

return VI_SUCCESS;
}
```

Appendix A
HP E2261A Specifications

M-Module Specification Compliance

The HP E2261A supports:

- A08, D16 accesses
- INTC interrupts
- IDENT capability

Note: Because of the HP E2261A's thickness, it is not guaranteed to fit in a non-HP M-Module carrier.

Capabilities

The following list describes the HP E2261A's serial communication capabilities:

- Four full-duplex (asynchronous) RS-232 Data Terminal Equipment (DTE) ports on one M-Module.
- 13 independently programmable Baud Rates for each port: 75, 110, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, 38400.
- Independently programmable handshake modes: RTS/CTS hardware handshake, XON/XOFF software handshake, or none.
- Independently programmable data format: 5 to 8 data bits plus polarity; odd, even, or no parity; 1, 1.5, or 2 stop bits.
- 2kByte input and output buffer for each port.

HP E2261A Specifications

Carrier Interface Specifications
The HP E2261A incorporates the standard 40-pin, 20x2 row connector interface to the carrier board for power and data/control, but does not have the 24-pin optional connector for carrying user-connections back onto the carrier board.

Power Requirements:

Cable Driving Capability: 50ft. with standard cable.

Standard M-Module Compliance: IDENT, A08, D16, INTC

M-Module Current: +5Vdc @ 0.1A
±12Vdc @ 0.01A

Appendix B
RS-232 Cables

This appendix lists several general purpose HP RS-232 cables and adapters. Consult your instrument's operating manual for information on which status lines are used for handshaking. Recommended cables and adapters are shown in **boldface** type; the others are listed because they may work better in some applications.

Note Use the HP E2261-61601 cable to separate the four RS-232 ports on the M-Module. The HP E2261A uses the DTR, TD, RD, RTS, CTS, DSR, and Ground lines only. Many of the cables listed in this appendix have status lines that will not be used.

Note The cables listed in this appendix will not fit within the confines of an M-Module Carrier.

Table B-4. HP 34401A, 33120A, 53131A to PC or Printer

Instrument Connector	Computer/Printer Connector	HP Cable Part Number	HP Adapter Part Number	Length
9-pin Male	25-pin Male	24542H 24542U **F1047-80002**[b]	none 5181-6641[a] **5181-6641**[b]	3m (9ft 10in) 3m (9ft 10in) 2.5m (8ft 2.5in)
9-pin Male	25-pin Female	24542G 24542U **F1047-80002**[b]	none 5181-6640[a] **5181-6640**[a]	3m (9ft 10in) 3m (9ft 10in) 2.5m (8ft 2.5in)
9-pin Male	9-pin Male	24542U 24542H & 24542M **F1047-80002**[b]	none none none	3m (9ft 10in) 6m (19ft 10in) 2.5m (8ft 2.5in)

a. One of four adapters in the HP 34399A RS-232 Adapter Kit. Kit includes four adapters to go from DB9 Female Cable (HP 34398A) to PC/Printer DB25 male or female, or to modem DB9 Female or DB25 Female.

b. Part of HP 34398A RS-232 Cable Kit. HP 34398A comes with RS-232, 9-pin female to 9-pin female Null modem/printer cable and one adapter 9-pin male to 25-pin female (HP p.n. 5181-6641). The adapter is also included in HP 34399A above.

Table B-5. HP 34401A, 33120A, 53131A to Modem

Instrument Connector	Computer/Printer Connector	HP Cable Part Number	HP Adapter Part Number	Length
9-pin Male	25-pin Female	24542M 24542U **F1047-80002**[b]	none 5181-6642[a] **5181-6642**[a]	3m (9ft 10in) 3m (9ft 10in) 2.5m (8ft 2.5in)
9-pin Male	9-pin Female	24542U **F1047-80002**[b]	5181-6639[a] **5181-6639**[a]	3m (9ft 10in) 2.5m (8ft 2.5in)

a. One of four adapters in the HP 34399A RS-232 Adapter Kit. Kit includes four adapters to go from DB9 Female Cable (HP 34398A) to PC/Printer DB25 male or female, or to modem DB9 Female or DB25 Female.

b. Part of HP 34398A RS-232 Cable Kit. HP 34398A comes with RS-232, 9-pin female to 9-pin female Null modem/printer cable and one adapter 9-pin male to 25-pin female (HP p.n. 5181-6641). The adapter is also included in HP 34399A above.

Table B-6. HP 54600 Series to Modem

Instrument Connector	Computer/Printer Connector	HP Cable Part Number	HP Adapter Part Number	Length
25-pin Female	25-pin Female	24542G	5181-6642[a]	3m (9ft 10in)
25-pin Female	9-pin Female	24542G **24542M**	5181-6639[a] none	3m (9ft 10in) 3m (9ft 10in)

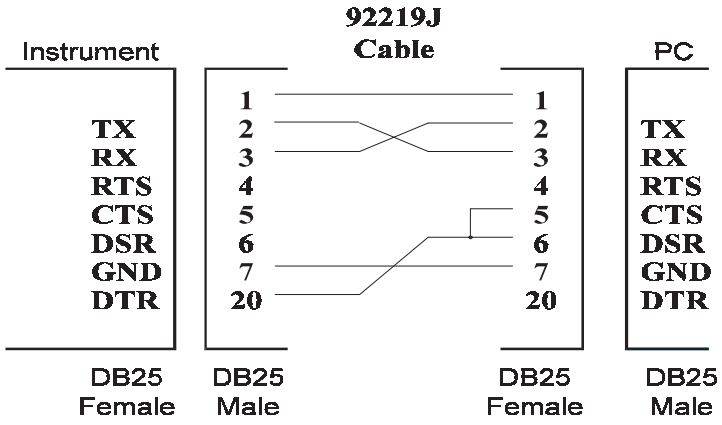
a. One of four adapters in the HP 34399A RS-232 Adapter Kit. Kit includes four adapters to go from DB9 Female Cable (HP 34398A) to PC/Printer DB25 male or female, or to modem DB9 Female or DB25 Female.

Table B-7. HP 54600 Series to PC or Printer

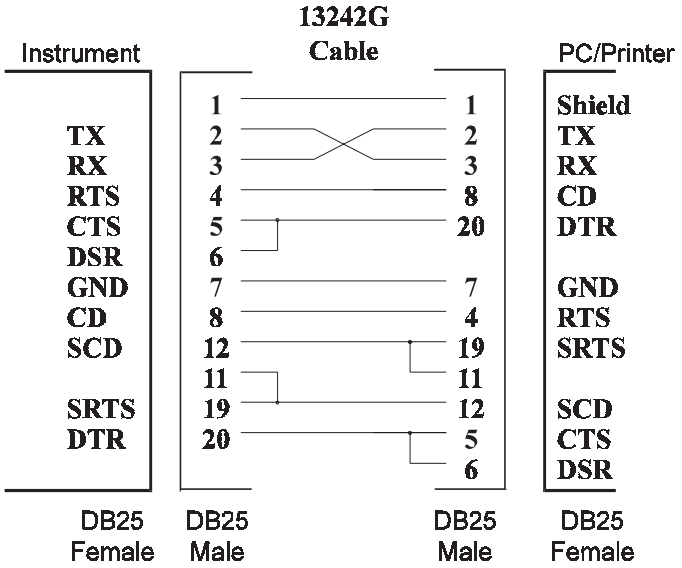
Instrument Connector	Computer/Printer Connector	HP Cable Part Number	HP Adapter Part Number	Length
25-pin Female	25-pin Male	17255D C2913A **24542G**	 **5181-6641**[a]	1.2m (3ft 11in) 1.2m (3ft 11in) 3m (9ft 10in)
25-pin Female	25-pin Female	13242G 17255M C2914A **24542G**	 **5181-6640**[a]	5m (16ft 8in) 1.5m (4ft 11in) 1.2m (3ft 11in) 3m (9ft 10in)
25-pin Female	9-pin Male	**24542G** 24542U F1047-80002[b]	none 5181-6640[a] 5181-6640[a]	3m (9ft 10in) 3m (9ft 10in) 2.5m (8ft 2.5in)

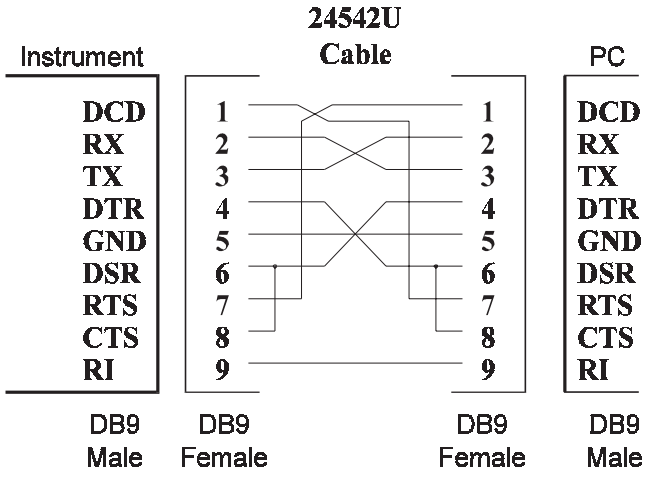
a. One of four adapters in the HP 34399A RS-232 Adapter Kit. Kit includes four adapters to go from DB9 Female Cable (HP 34398A) to PC/Printer DB25 male or female, or to modem DB9 Female or DB25 Female.

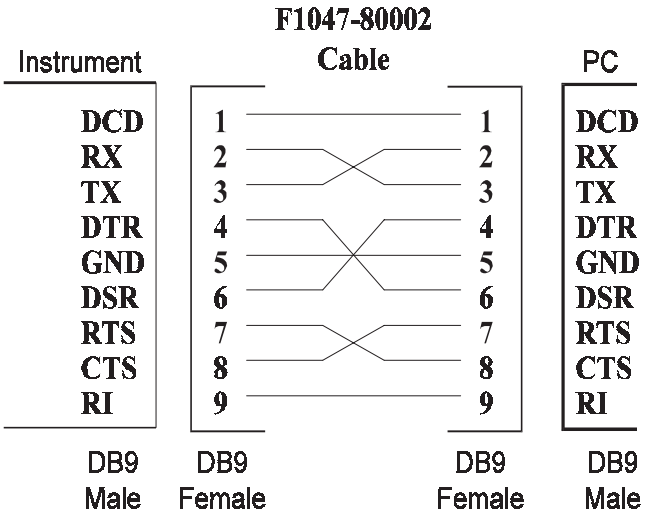
b. Part of HP 34398A RS-232 Cable Kit. HP 34398A comes with RS-232, 9-pin female to 9-pin female Null modem/printer cable and one adapter 9-pin male to 25-pin female (HP p.n. 5181-6641). The adapter is also included in HP 34399A above.

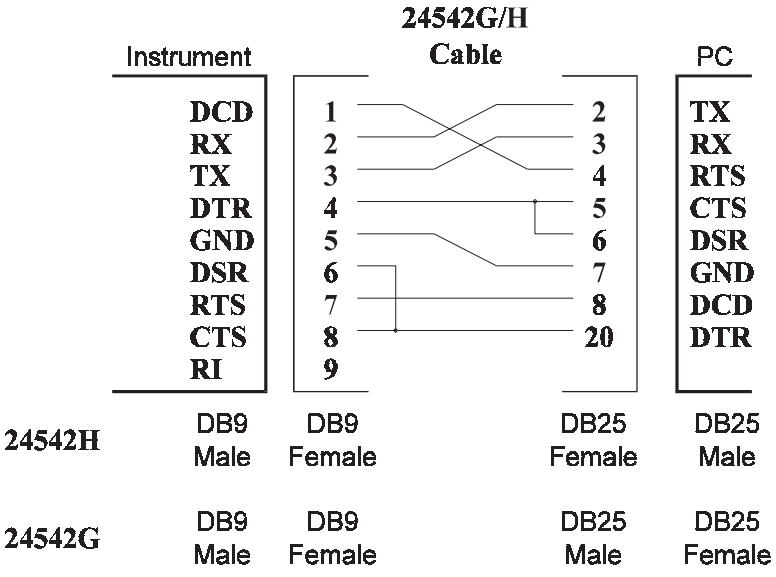


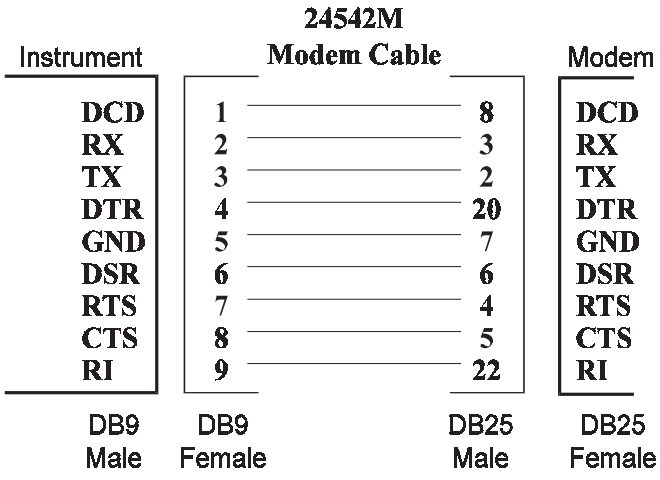
Note: The 92219J is directional. This cable may work differently when swapped end to end.

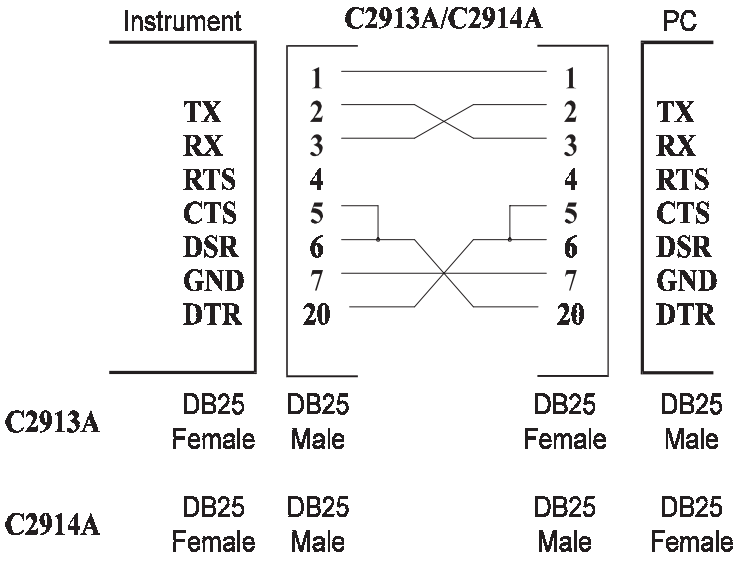


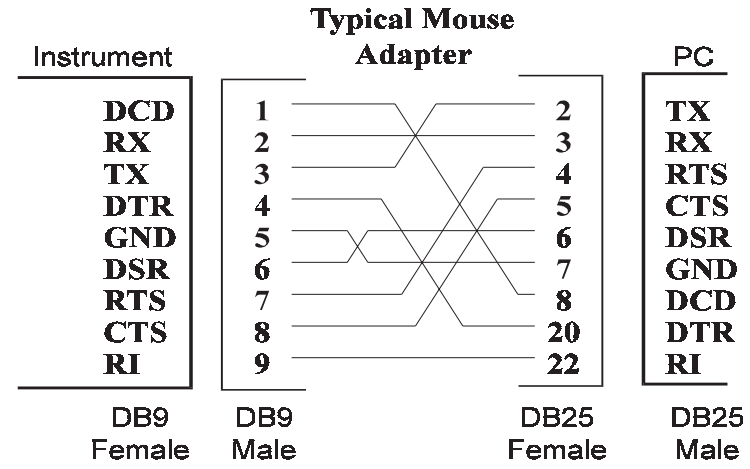



F1047-80002
Cable
Instrument
PC
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1
2
3
4
5
6
7
8
9
DB9
Male
DB9
Female
DB9
Female
DB9
Male

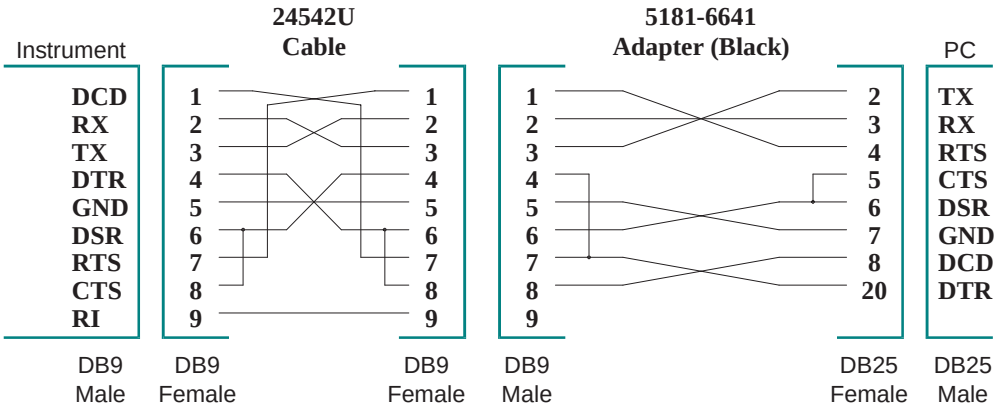

24542G/H
Cable
Instrument
PC
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1
2
3
4
5
6
7
8
9
2
3
4
5
6
7
8
20
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
24542H
DB9
Male
DB9
Female
DB25
Female
DB25
Male
24542G
DB9
Male
DB9
Female
DB25
Male
DB25
Female

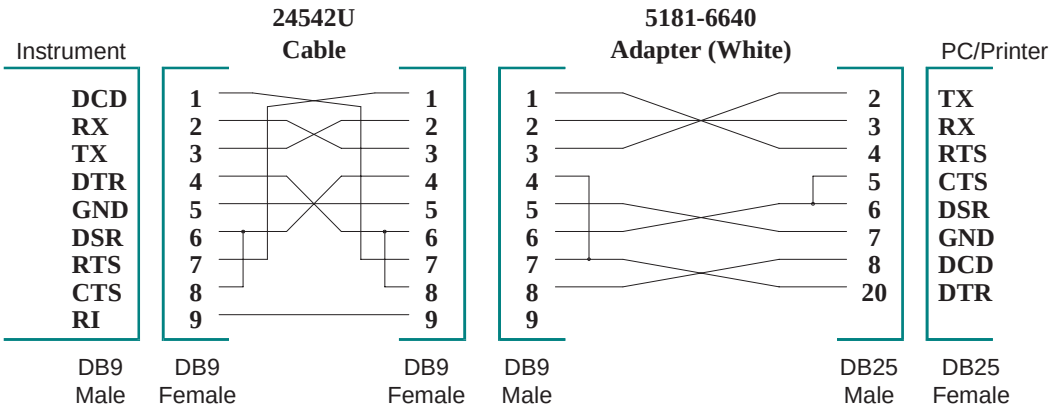

24542M
Modem Cable
Instrument
Modem
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1
2
3
4
5
6
7
8
9
8
3
2
20
7
6
4
5
22
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
DB9
Male
DB9
Female
DB25
Male
DB25
Female

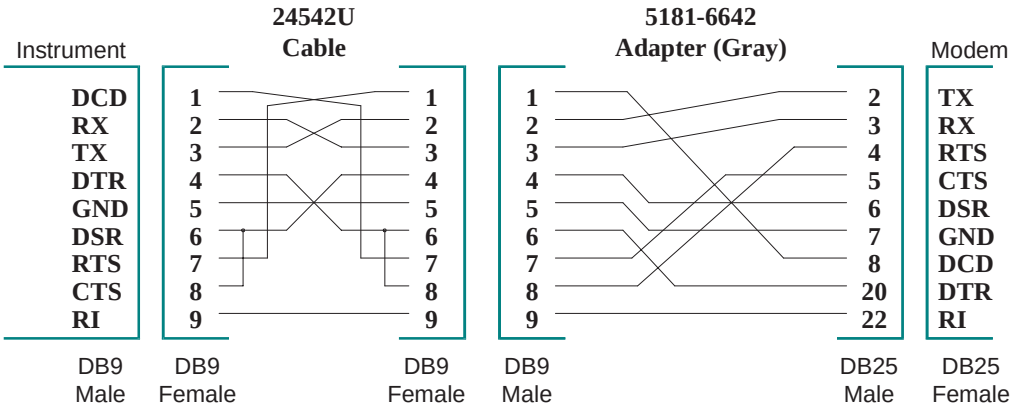


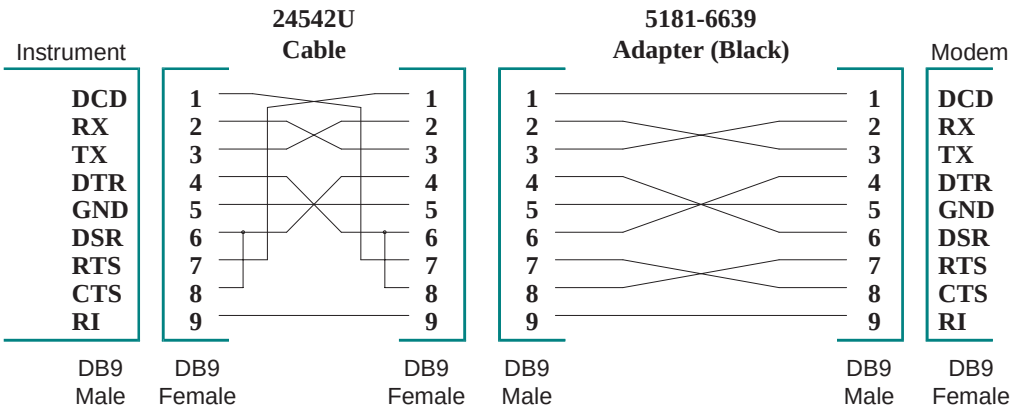


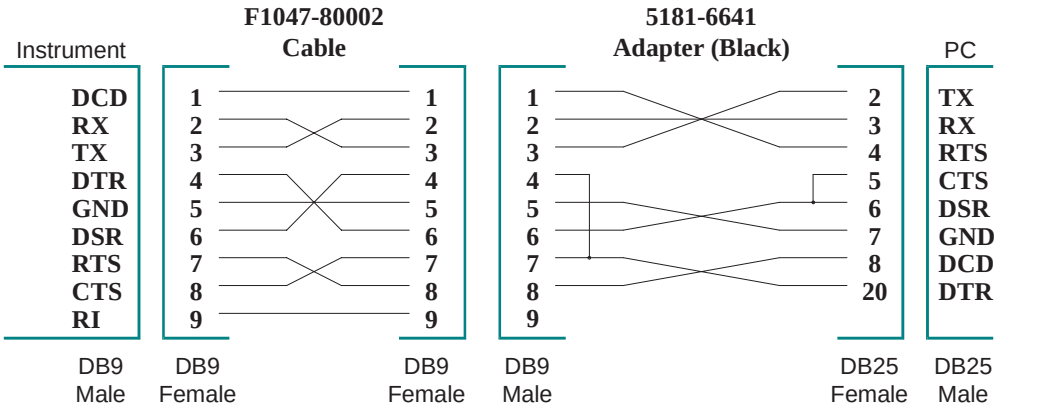
A mouse adapter works well as a 9 pin to 25 pin adapter with a PC.

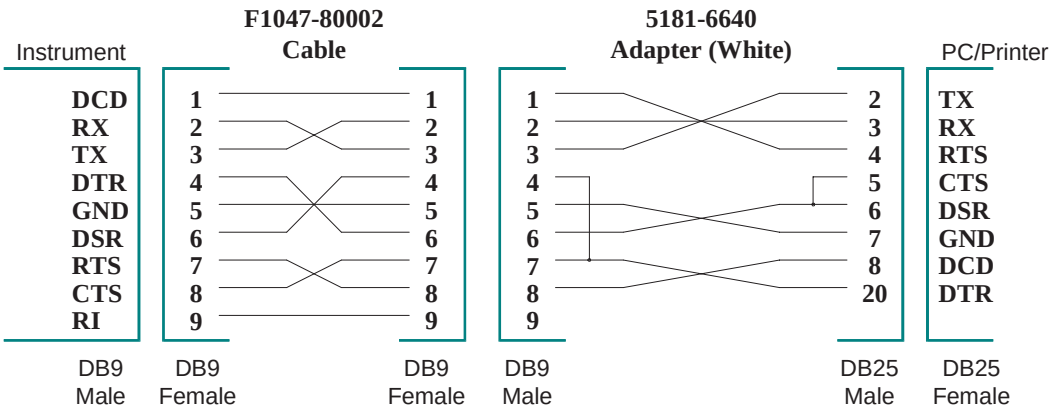

24542U
Cable
5181-6641
Adapter (Black)
Instrument
PC
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 20
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB25 Female
DB25 Male

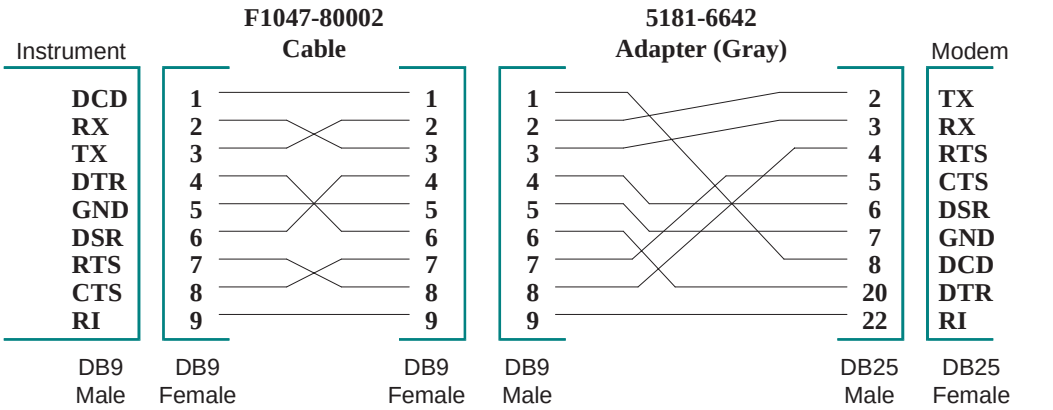

24542U
Cable
5181-6640
Adapter (White)
Instrument
PC/Printer
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 20
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB25 Male
DB25 Female

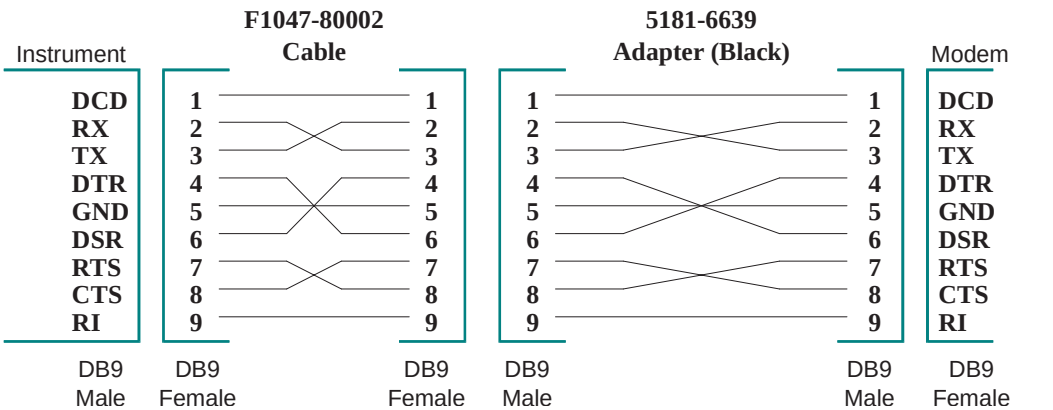

24542U
Cable
5181-6642
Adapter (Gray)
Instrument
Modem
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 20 22
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
RI
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB25 Male
DB25 Female


24542U
Cable
5181-6639
Adapter (Black)
Instrument
Modem
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB9 Male
DB9 Female


F1047-80002
Cable
5181-6641
Adapter (Black)
Instrument
PC
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 20
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB25 Female
DB25 Male


F1047-80002
Cable
5181-6640
Adapter (White)
Instrument
PC/Printer
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 20
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB25 Male
DB25 Female


F1047-80002
Cable
5181-6642
Adapter (Gray)
Instrument
Modem
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 20 22
TX
RX
RTS
CTS
DSR
GND
DCD
DTR
RI
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB25 Male
DB25 Female


F1047-80002
Cable
5181-6639
Adapter (Black)
Instrument
Modem
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
DCD
RX
TX
DTR
GND
DSR
RTS
CTS
RI
DB9 Male
DB9 Female
DB9 Female
DB9 Male
DB9 Male
DB9 Female

Symbols

A

B

C

D

E

T