

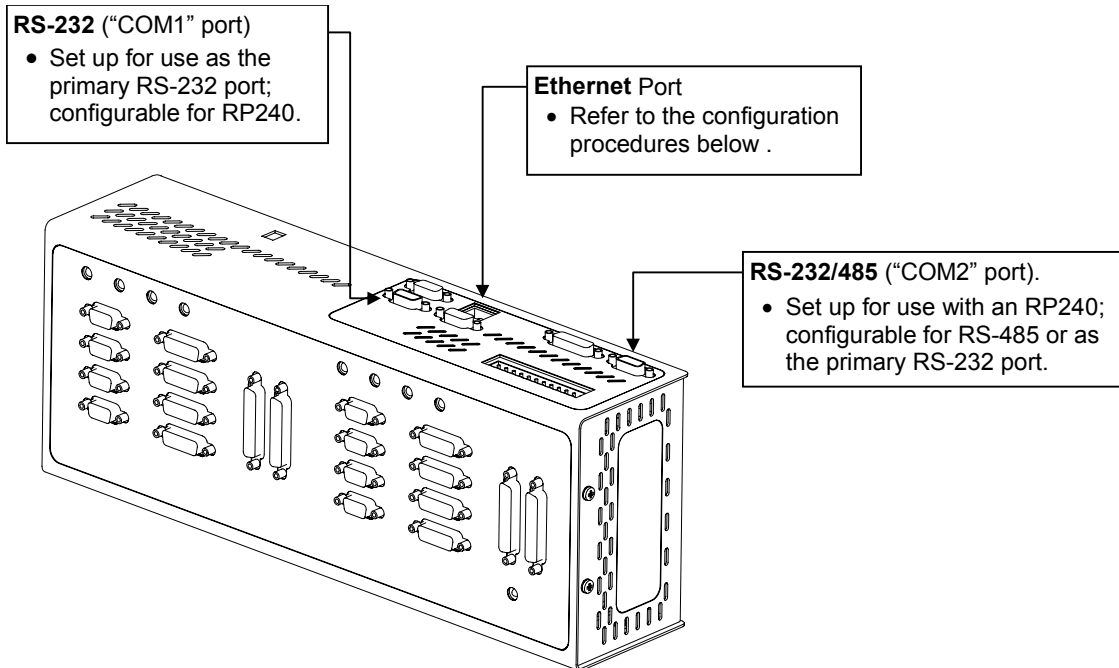
CHAPTER TWO

Communication

IN THIS CHAPTER

- Motion Planner™ communication features 36
- Serial Communication:
 - Controlling multiple serial ports 37
 - RS-232C daisy-chaining 38
 - RS-485 multi-drop 41

Communication Options



USING MULTIPLE PORTS

You can communicate to either the Ethernet port or the RS-232 port (COM1) at any give time; the port that you communicate to first is the only one that is recognized until you cycle power.

You can communicate to the Ethernet port or the RS-232 port (COM1) while the 6K is also communicating with an RP240 via the RS-232/485 port (COM2).

Motion Planner Communication Features

Motion Planner provides easy direct communication links to the product:

- Communicate directly from any Motion Planner utility (Editor, Terminal, and Panels).
- Communication setup parameters (Ethernet and serial communication).
- Download the 6K product's soft operating system. The operating system is loaded at the factory, but can use this feature to download upgrades.
- Download motion programs to the controller, and upload motion programs from the controller.

Communications Server: Also available on the Motion Planner CD is a 32-bit OLE automation server for adding 6K communication capability to your custom applications created with programming languages such as Visual Basic or Visual C++. For details, see *Com6srvr User's Guide for Gemini & 6K Series Products*.

Serial Communication

In this section:

- Controlling Multiple Serial Ports
- RS-232 Daisy Chaining
- RS-485 Multi-Drop

Controlling Multiple Serial Ports

Every 6K Series product has two serial ports. The “RS-232” connector is referenced as the “COM1” serial port, and the “RS-232/485” connector is referenced as the “COM2” serial port.

XON/XOFF

The XON/XOFF command was created to enable or disable XON/XOFF ASCII handshaking. (XON/XOFF1 enables XON/XOFF, XON/XOFF0 disables XON/XOFF) Defaults: XON/XOFF1 for the COM1 port, XON/XOFF0 for the COM2 port.

Controllers on a multi-drop do not support XON/XOFF; to ensure that XON/XOFF is disabled for COM2, send the PORT2 command followed by the XON/XOFF0 command.

Configuring the COM Port

To control the applicable port for setting up serial communication and transmitting ASCII text strings, use the PORT command. PORT1 selects COM1 and PORT2 selects COM2.

- Serial communication setup commands (see list below) affect the COM port selected with the last PORT command. For example, to configure the COM2 port for 6K language commands only (e.g., to communicate to the 6K product over an RS-485 interface), execute the PORT2 command, then execute the DRPCHK0 command.

- DRPCHK..... RP240 Check
- E..... Enable Serial Communication
- ECHO..... Enable Communication Echo
- BOT..... Beginning of Transmission Characters
- BAUD..... Serial Communication Baud Rate
- EOT..... End of Transmission Characters
- EOL..... End of Line Terminating Characters
- ERRBAD..... Error Prompt
- ERRDEF..... Program Definition Prompt
- ERRLVL..... Error Detection Level
- ERRORK..... Good Prompt
- XON/XOFF..... Enable or disable XON/XOFF

- The PORT command also selects the COM port through which the WRITE and READ commands transmit ASCII text strings. If an RP240 is connected, the DWRITE command (and all other RP240 commands) will affect the RP240 regardless of the PORT command setting. If no RP240 is detected, the commands are sent to the COM2 port. DWRITE text strings are always terminated with a carriage return.

Setup for 6K Language or RP240

To configure the COM ports for use with 6K language commands or an RP240, use the DRPCHK command. The DRPCHK command affects the COM port selected with the last PORT command. The default for COM1 is DRPCHK0; the default for COM2 is DRPCHK3. The DRPCHK setting is automatically saved in non-volatile memory. **NOTE:** Only one COM port can be set to DRPCHK2 or DRPCHK3 at any given time.

- DRPCHKØ..... Use the COM port for 6K language commands only. This is the default setting for COM1, and if using RS-485 half duplex on COM2. Power-up messages appear on all ports set to DRPCHKØ.
- DRPCHK1..... Check for the presence of an RP240 at power-up/reset. If an RP240 is present, initialize the RP240. If an RP240 is not present, use the port only for 6K language commands. **NOTE:** RP240 commands will be sent at power-up and reset.
- DRPCHK2..... Check for the presence of an RP240 every 5-6 seconds. If an RP240 is plugged in, initialize the RP240.
- DRPCHK3..... Check for the presence of an RP240 at power-up/reset. If an RP240 is present, the initialize the RP240. If an RP240 is not present, use the COM port for DWRITE commands only, and ignore received characters. This is the default setting for COM2, unless you are using RS-485 multi-drop communication (in that case the default changes to DRPCHKØ).

RS-485 compatible products: If you are using RS-485 communication in a multi-drop (requires you to change an internal jumper to select half duplex), the default setting for COM2 is DRPCHKØ. If the internal jumper setting is left at full duplex, the default setting for COM2 is DRPCHK3.

Selecting a Destination Port for Transmitting from the Controller

To define the port (COM port) through which the 6K product sends its responses, you have 3 options:

- Do nothing different. The response will be sent to the COM port through which the request was made. If the command is in a stored program, the report will be sent to the COM port selected by the most recent PORT command.
- Prefix the command with [. This causes the response to be sent to both COM ports. (e.g., the [TFS command response will be sent through both COM ports)
- Prefix the command with]. This causes the response to be sent to the alternative COM port. For example, if a report back (e.g.,]TAS) is requested from COM1, the response is sent through COM2. If the command is in a stored program, the report will be sent out the alternate port from the one selected by the most recent PORT command.

RS-232C Daisy-Chaining

Up to ninety-nine stand-alone 6K Series products can be daisy-chained. There are two methods of daisy-chaining: one uses a computer or terminal as the controller in the chain; the other uses one 6K product as the master controller. Refer to your product's *Installation Guide* for daisy-chain connections.

Follow these steps to implement daisy-chaining:

Step 1

To enable and disable communications on a particular controller unit in the chain, you must use the Daisy-Chain Address (ADDR) command to establish a unique device address for each the unit. The ADDR command automatically configures unit addresses for daisy chaining. This command allows up to 99 units on a daisy chain to be uniquely addressed.

Sending ADDR_{*i*} to the first unit in the daisy chain sets its address to be (*i*). The first unit in turn transmits ADDR(*i* + 1) to the next unit to set its address to (*i* + 1). This continues down the daisy chain until the last unit of (*n*) daisy-chained units has its address set to (*i* + *n*).

Note that a controller with the default device address of zero (0) will send an initial power-up start message similar to the following:

```
*PARKER 6K MOTION CONTROLLER
*NO REMOTE PANEL
```

Step 2

Connect the daisy-chain with a terminal as the master (see diagram in the product's *Installation Guide*).

It is necessary to have the error level set to 1 for all units on the daisy-chain (ERRLVL1). When the error level is not set to 1, the controller sends ERROK or ERBAD prompts after each command, which makes daisy-chaining impossible. Send the ERRLVL1 command to each unit in the chain. (**NOTE:** To send a the ERRLVL1 command to one specific unit on the chain, prefix the command with the appropriate unit's device address and an underline.)

Commands

```
1_ ERRLVL1      ; Set error level to 1 for unit 1
2_ ERRLVL1      ; Set error level to 1 for unit 2
3_ ERRLVL1      ; Set error level to 1 for unit 3
```

After this has been accomplished, a carriage return sent from the terminal will not cause any controller to send a prompt. Verify this. Instructions below (step 3) show how to set the error level to 1 automatically on power-up by using the controller's power-up start program (highly recommended).

After the error level for all units has been set to ERRLVL1, send a 6K series command to all units on the daisy-chain by entering that command from the master terminal.

Commands

```
OUT1111        ; Turn on onboard outputs 1-4 on all units
A50,50          ; Set accel to 50 (all units, axes 1 & 2)
```

To send a 6K series command to one particular unit on the chain, prefix the command with the appropriate unit's device address and an underline:

Commands

```
2_ OUT0         ; Turn off onboard output 1 on unit 2
4_ OUT0         ; Turn off onboard output 1 on unit 4
```

To receive data from a particular controller on the chain, you **must** prefix the command with the appropriate unit's device address and an underline:

Commands

```
1_ A           ; Request acceleration information from unit 1
*A50,50        ; Response from unit 1
```

Use the E command to enable/disable RS-232C communications for an individual unit. If all 6K controller units on the daisy chain are enabled, commands without a device address identifier will be executed by all units. Because of the daisy-chain's serial nature, the commands will be executed approximately 1 ms per character later on each successive unit in the chain (assuming 9600 baud).

Units with the RS-232C disabled (EØ) will not respond to any commands, except E1; however, characters are still echoed to the next device in the daisy chain.

Commands

```
3_ E0          ; Disable RS-232C on unit 3
VAR1=1         ; Set variable 1 to 1 on all other units
3_ E1          ; Enable RS-232C on unit 3
3_ VAR1=5      ; Set variable 1 to 5 on unit 3
```

Verify communication to all units by using the techniques described above.

Step 3

Now that communication is established, programming of the units can begin (alternatively, units can be programmed individually by connecting the master terminal to one unit at a time). To allow daisy-chaining between multiple controllers, the ERRLVL1 command must be used to prevent units from sending error messages and command prompts. In every daisy-chained unit, the ERRLVL1 command should be placed in the program that is defined as the STARTP program:

```

Program
DEF chain          ; Begin definition of program chain
ERRLVL1           ; Set error level to 1
GOTO main         ; Go to program main
END               ; End definition of program chain
STARTP chain      ; Designates program chain as the power-up program

```

To define program main for unit 0:

```

Program
0_DEF main        ; Begin definition of program main on unit 0
0_GO              ; Start motion
0_END             ; End definition of program main on unit 0

```

Step 4

After all programming is completed, program execution can be controlled by either a master terminal, or by a 6K Series controller used as a master.

Daisy-Chaining from a Computer or Terminal

Controlling the daisy-chain from a master computer or terminal follows the examples above:

```

Commands
0_RUN main        ; Run program main on unit 0
1_RUN main        ; Run program main on unit 1
2_GO1             ; Start motion on unit 2 axis 1
3_2A              ; Get A command response from unit 3 axis 2

```

Daisy-Chaining from a Master 6K Controller

Controlling the daisy-chain from a master 6K controller (the first unit on the daisy-chain) requires the programs stored in the master controller to control program and command execution on the slave controllers. The example below demonstrates the use of the WRITE command to send commands to other units on the daisy chain.

NOTE

The last unit on the daisy-chain must have RS-232C echo disabled (ECHOØ command).

Master controller's main program:

```

Program
DEF main          ; Program main
L                 ; Indefinite loop
  WHILE (IN.1 = b0) ; Wait for input 1 to go active
  NWHILE
  GOL              ; Initiate linear interpolated move
  WHILE (IN.1 = b1) ; Wait for input 1 to go inactive
  NWHILE
  WRITE"2_D2000,4000" ; Send message "2_D2000,4000" down daisy chain
  WRITE"2_ACK"       ; Send message "2_ACK" down the daisy chain
LN                 ; End of loop
END               ; End of program main

```

Controller unit 2 ack program:

```

Program
DEF ack          ; Program ack
GO11             ; Start motion on both axes
END              ; End of program ack

```

Daisy-Chaining and RP240s

RP240s cannot be placed in the controller daisy chain; RP240s can only be connected to the designated RP240 port on a controller. It is possible to use only one RP240 with a controller daisy-chain to input data for multiple units on the chain. The example below (for the controller master with an RP240 connected) reads data from the RP240 into variables 1 (*data1*) & 2 (*data2*), then sends the messages 3_Ddata1, data2<CR> and 3_GO<CR>.

Sample portion of code:

```
L ; Indefinite loop
VAR1=DREAD ; Read RP240 data into variable 1
VAR2=DREAD ; Read RP240 data into variable 2
EOT0,0,0,0 ; Turn off <CR>
WRITE"3_D" ; Send message "3_D" down the daisy chain
WRVAR1 ; Send variable 1 data down the daisy chain
WRITE", " ; Send message ", " down the daisy chain
EOT13,0,0,0 ; Turn on <CR>
WRVAR2 ; Send variable 2 data down the daisy chain
WRITE"3_GO" ; Send message "3_GO" down the daisy chain
LN ; End of loop
```

RS-485 Multi-Drop

Up to 99 6K Series products can be multi-dropped. Refer to your product's *Installation Guide* for multi-drop connections.

To establish device addresses, using the ADDR command:

The ADDR command allows you to establish up to 99 unique addresses. To use the ADDR command, you must address each unit individually before it is connected on the multi drop. For example, given that each product is shipped configured with address zero, you could set up a 4-unit multi-drop with the commands below, and then connect them in a multi drop:

1. Connect the unit that is to be unit 1 and transmit the Ø_ADDR1 command to it.
2. Connect the unit that is to be unit 2 and transmit the Ø_ADDR2 command to it.
3. Connect the unit that is to be unit 3 and transmit the Ø_ADDR3 command to it.
4. Connect the unit that is to be unit 4 and transmit the Ø_ADDR4 command to it.

If you need to replace a unit in the multi drop, send the Ø_ADDRi command to it, where "i" is the address you wish the new unit to have.

To send a 6K command from the master unit to a specific unit in the multi-drop, prefix the command with the unit address and an underscore (e.g., 3_OUTØ turns off output 1 on unit 3). The master unit (if it is not a 6K product) can receive data from a multi-drop unit.

The ECHO command was enhanced with options 2 and 3. The purpose is to accommodate an RS-485 multi-drop configuration in which a host computer communicates to the "master" 6K controller over RS-232 (COM1 port) and the master 6K controller communicates over RS-485 (COM2 port) to the rest of the units on the multi-drop. For this configuration, the echo setup should be configured by sending to the master the following commands executed in the order shown. In this example, it is assumed that the master's device address is set to 1. Hence, each command is prefixed with "1_" to address only the master unit.

```
1_PORT2 ..... Subsequent command affects COM2, the RS-485 port
1_ECHO2 ..... Echo characters back through the other port, COM1
1_PORT1 ..... Subsequent command affects COM1, the RS-232 port
1_ECHO3 ..... Echo characters back through both ports, COM1 and COM2
```

NOTE

Controllers on a multi-drop do not support XON/XOFF. To ensure that XON/XOFF is disabled for COM2, send the PORT2 command followed by the XONOFFØ command.