FISHER-ROSEMOUNT

RS3™

Configuration Quick Reference Guide

Performance Series 1, Release 4.0

August 1999
Manual PN: 1984-2812-0808

Fisher-Rosemount Systems, Inc.
12000 Portland Avenue South
Burnsville, Minnesota 55337 U.S.A.

Telephone: (612) 895-2000
TWX/Telex: 192177
FAX: (612) 895-2044

FISHER-ROSEMOUNT

RS3™

Configuration Quick Reference Guide

About This Manual

This document is intended as a quick reference guide to some of the commonly referenced information about RS3 Performance Series™ Release 1. This document is only part of the RS3 documentation and is not intended to replace other RS3 manuals.

Changes for This Release

-

Revision Level for This Manual

Revision Table for 1984-2812-08xx

This Version	1984-2812-0809—August, 1999
Supercedes	1984-2812-0808—May, 1996
	1984-2812-0807—August, 1994
	1984-2812-0806—January, 1993
	1984-2812-0805—February, 1991
	1984-2812-0804—August, 1990
	1984-2812-0803—March, 1990
	1984-2812-0802—September, 1989
	1984-2812-0801—March, 1989

References to Other Manuals

References to other RS3 user manuals list the manual, chapter, and sometimes the section as shown below.

Example Entries:

For ..., see CC: 3.

↑ ↑

Manual Title Chapter

For ..., see CC: 1-1.

↑ ↑

Manual Title Chapter-Section

Abbreviations of Manual Titles

AL = Alarm Messages

BA = ABC Batch

CB = ControlBlock Configuration

CC = Console Configuration

DT = Disk and Tape Functions

IO = I/O Block Configuration

OP = Operator's Guide

OV = System Overview and Glossary

PW = PeerWay Interfaces

RB = Rosemount Basic Language

RI = RNI Release Notes and Installation Guide

RP = RNI Programmer's Reference Manual

SP = Site Preparation and Installation

SV = Service

Reference Documents

Prerequisite Documents

You should be familiar with the information in the following documents before using this manual:

System Overview Manual and Glossary	1984-2640-21x0
Software Release Notes, Performance Series 1	1984-2818-0110

Related Documents

You may find the following documents helpful when using this manual:

ABC Batch Software Manual	1984-2654-21x0
Alarm Messages Manual	1984-2657-19x1
ABC Batch Quick Reference Guide	1984-2818-1103
Console Configuration Manual	1984-2643-21x0
ControlBlock Configuration Manual	1984-2646-21x0
I/O Block Configuration Manual	1984-2645-21x0
Operator's Guide	1984-2647-19x1
PeerWay Interfaces Manual	1984-2650-21x0
Rosemount Basic Language Manual	1984-2653-21x0
RNI Programmer's Reference Manual	1984-3356-02x1
RNI Release Notes and Installation Guide	1984-3357-02x1
Service Manual, Volume 1	1984-2648-21x0
Service Manual, Volume 2	1984-2648-31x0
Service Quick Reference Guide	1984-2816-0904
Site Preparation and Installation Manual	1984-2642-21x0
Software Discrepancies for Performance Series 1	1984-2818-0311
User Manual Master Index	1984-2641-21x0

Contents

Section 1:
Screens/Operator Interface

Alarm Areas

- An alarm area is a configured group of plant unit numbers.
- Plant units can belong to more than one alarm area.
- Area names are used throughout the system.
- Each alarm area has its own alarm lists (active and cleared).
- For more information about alarm areas, see CC: 6.

Area Name Configuration Screen (ANC)

- The **Graphic** field specifies a process graphic from the **local** hard disk that is associated with the area. The operator can cursor to the field and press [SELECT] to call up the graphic.
- [EXCH] switches between displaying **Unit Ranges** and **Descriptors**.
- [SELECT] an area name to call up that alarm–area list.
- [SELECT] **Unit Ranges** to call up the Plant Unit Config screen.
- If you make changes to the **Graphic** or **Unit Ranges** fields of an area, the changes affect only the local configuration—the changes are not made to other consoles that have the same area configured. To make these changes at other consoles, you must go to each console that contains the area name and **in node number order**, delete the area name, and reenter it.

Disabling an Alarm Area

- When an alarm area is disabled:
  - The alarms from that area are not shown in alarm type lists
  - The alarm area lists for that area are empty
- You specify which key classes can disable and enable alarm areas on the Alarm Configuration screen.

❑ **To disable an alarm area:**

- From the Area Name Configuration screen, cursor to the **Status** field for the area. Press [NEXT OPTION] until Disabled appears. Press [ENTER].
- From the command line, type **AD :** (*area name*) [ENTER]

Enabling an Alarm Area

❑ **To enable an alarm area:**

- From the Area Name Configuration screen, cursor to the **Status** field for the area. Press [NEXT OPTION] until Enabled appears. Press [ENTER].
- From the command line, type **AE :** (*area name*) [ENTER]

Alarm and Event Lists

- Alarm lists are configured on the Alarm List Configuration screen.
- Event lists are configured on the Event List Configuration screen.
- Events are configured on ControlBlock logic steps and various I/O block actions.
- For more information about alarm and event lists, see CC: 6.

Alarm List Configuration and Event List Configuration Screens

- The sum of all alarm and event lists (the sum of all **Size** fields on the Alarm List Configuration and Event List Configuration screens) cannot exceed 3850 lines.
- To make changes to either list effective, the lists must be initialized with the **Initialize Lists** field on the Alarm List Configuration screen. **This will erase all alarm and event lists**.
- **Size** specifies the number of entries that are saved in a list. Once the **Size** value is reached, each new entry replaces the oldest entry.
- **Trigger** specifies the number of entries that cause the list to be printed. For example, if Trigger⇒24, then the list is printed after every 24 new entries.
- **Trigger** should be ≤ **Size** in order to print all list entries.
- If **either** the Active or Cleared Alarm list for an alarm type reaches the **Trigger** value, **both** lists are printed.

Alarm List Configuration Screen Only (ALC)

- The **Alarm Print** field specifies format characteristics. **Normal** indicates that alarm lists are printed with a header and with a form feed command that prints the list on a new page. **Single Line** indicates that lists are printed without a form feed command or a header.
- If **Alarm Print⇒Single Line** and **Trigger⇒1**, then alarms are printed one at a time without a form feed command or a header.
- **Backup Printer Node** must specify an existing printer node.
- **Restore Primary Printer Nodes** flashes when the backup printer has been used to print alarm lists. Printing will continue at the backup node unless the primary node is restored by cursoring to this field and pressing [ENTER].

Event List Configuration Screen Only (ELC)

- **Event Ranges** specify the events that make up each event list. Events can be entered individually (⇒9) or in a range (⇒12,17).
- Event types 1 through 255 cannot be entered in more than one list. Event types 241 to 255 are special events that initiate printouts of the following:

  241—Hardware Alarm Lists (Active and Cleared)
  242—Process Alarm Lists (Active and Cleared)
  243—System Status Lists (Active and Cleared)
  244—Disk Event List
  245—Event List 1
  246—Event List 2
  247—Event List 3
  248—Event List 4
  249—Event List 5
  250—Event List 6
  251—Event List 7
  252—Event List 8
  253—Event List 9
  254—Event List 10
  255—Operator Change Log

Alarm Broadcast Configuration

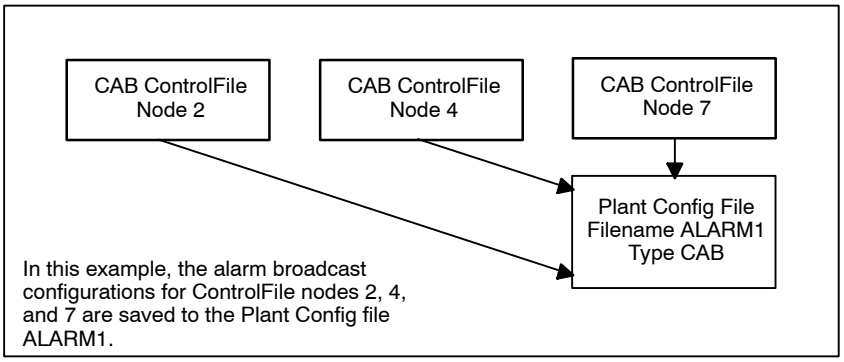
- An alarm broadcast configuration specifies which nodes are **sent alarms** from a ControlFile or console.
- Each ControlFile and console has its own alarm broadcast configuration.
- This is useful if multiple PeerWays are connected or if you want to limit PeerWay traffic.
- For more information about alarm broadcast configuration, see CC: 6.

Configure Alarm Broadcast Screen (CAB)

- Nodes that are sent alarms are **backlit**.
- You can **select** or **deselect** a node by cursoring to the number and pressing [ENTER].
- You can select or deselect a **range** of nodes with the **Set Range** and **Clear Range** fields.

Saving and Loading the Alarm Broadcast Configuration

- The alarm broadcast configuration for one or more ControlFiles can be saved to a CAB-type plant configuration file. Alarm broadcast configurations for a Console or SRU are saved in the Alarm/Event Configuration portion of the console configuration file.
- If you read the contents of a CAB-type plant configuration file, the CAB Config Contents screen appears. The highlighted numbers on this screen show which nodes have alarm broadcast configurations saved in the file. It **does not** show any alarm broadcast configurations.
- CAB files contain only alarm broadcast configurations. The figure below shows an example of saving alarm broadcast configurations.
- For more information about saving and loading a CAB file, see DT: 2.



Alarm Configuration Screen

- Alarm management parameters are defined on the Alarm Configuration and Automatic Alarm Deletion screens.
- For more information about configuring alarm management parameters, see CC: 6.

Alarm Configuration Screen Fields (AC)

- **Actual Log Size** displays maximum size of alarm log, in bytes.
- **Alarm Log Size** specifies the maximum number of entries in the alarm log. If you change the log size, the alarm log is cleared.
- **Alarm Log Types** specifies which alarm types are included in the alarm log. To enter, type the first letter(s) of the alarm type(s). For example, **phb** specifies process, hardware and batch alarms; ***** specifies all alarm types. For event lists, specify numbers; for example, **1-5** specifies event lists 1 through 5. **1-0** specifies all event lists.
- **Alarm Summary** specifies how the alarm line is displayed at the console. **Enabled** specifies the alarm summary mode; **Disabled** specifies the alarm banner mode.
- **Alarm View Toggle** specifies the alarm animation mode. **Full** mode displays all animation including suppressed, disabled, and inhibited alarms; **Focused** masks out animation of suppressed, disabled, and inhibited alarms.
- **Default Priority** specifies the default alarm priorities for I/O blocks and ControlBlocks. The default priorities are used when blocks are created at the console.
- **Hardware Alarms Have More Priority Than** specifies the relative priority of hardware alarms. The entry specifies which alarm type follows hardware alarms in the system priority scheme.
- **Read Console Lists** specifies which consoles are searched for alarm messages during alarm regeneration.
- **Suppressed Type** specifies which alarm types can be suppressed. To enter, type the first letter(s) of the alarm type(s). For example, **sb** specifies system status and batch alarms; ***** specifies that all possible alarm types will be suppressed.
- **Trend: Backup and File Full Alarms** specifies whether or not trend file backup and overflow messages are included in the alarm log.
- **Unacked, Cleared Alarm** specifies how unacknowledged, cleared alarms are displayed.
- **ISA** blink, backlighted in color of the most critical, unacknowledged alarm condition
- **RMT** blink, backlighted in the color of the current condition
- **Color** backlighted in a configured alarm color

Automatic Alarm Deletion Screen (AC [PAGE AHEAD])

- **Delete Alarms/Events** specifies whether the alarm deletion process is enabled or disabled. The alarm deletion process automatically deletes entries from the alarm lists and event lists if the entries:
  - are cleared
  - are of the types configured in the **Types** field
  - are older than the time configured in the **Time** field
  - meet the delete criteria specified in the **Acknowledged**, **Logged**, **Printed**, and **Priorities** fields for each alarm type.
- **Time** specifies a time interval for the alarm delete process. All alarms older than the specified time are to be deleted. Format is dd/hh:mm.
- **Types** specifies the alarm types that can be deleted in the alarm deletion process. To enter, type the first letter(s) of the alarm type(s); for example, **ph** specifies process and hardware alarms; ***** specifies all alarm types. For event lists, specify numbers; for example, **1-5** specifies event lists 1 through 5. **1-0** specifies all event lists.
- Pressing [ENTER] on the **Initialize Deletion Criteria for all types** field clears the Deletion Criteria table.

Alarm Log

- Alarm log stores alarm and event messages in a console disk file.
- You use the Alarm Configuration screen to configure which alarm and event types are stored in the alarm log.
- For more information about alarm logs, see CC: 6.

Log Display Configuration Screen (LDC)

- The **Log Display Configuration** determines the format of the Alarm Log Display.
- Log display configurations are part of the console configuration file.
- Up to 30 log display configurations can be configured at a console.
- The **Sort** fields specify how the alarm log information is sorted. **Sort Crit** specifies how to sort the alarm log. **SORT Status** specifies whether or not the log can be sorted from the Alarm Log Display screen.
- **Find** fields specify which information is **included** in the log display.
- **Filter** fields specify which information is **excluded** from log display.
- Table 1–1 shows examples of log display configurations.

❑ **To move from the Log Display Configuration screen to the Alarm Log Display:**

— Cursor to “LOG”, and press [SELECT].

Alarm Log Display Screen (ALD)

- Displays information from the alarm log file according to a log display configuration.
- You can temporarily modify the log display configuration by cursoring to the **Find** and **Filter** fields at the top of the screen and pressing [SELECT]. Changes are in effect until you leave screen.
- To **sort** the log, the sort **SORT Status** field of the log the display configuration field must be ON. Then, you must cursor to the **Sort** field on the Alarm Log Display screen and press [ENTER].
- **Top Entry** is the number of log entry shown at the top of the display. **Entry Count** is the number of displayed entries. (1500 is the highest value shown, even if more are displayed.)

❑ **To move from the Alarm Log Display to the Log Configuration Screen:**

— Cursor to “CNFG”, and press [SELECT].

Printing the Alarm Log

❑ **To print alarms on the current screen in 132-column format:**

1. Cursor to “PPRT“ on the Alarm Log Display screen.
2. Press [SELECT].

❑ **To print all alarms in 132-column format:**

1. Cursor to “LPRT“ on the Alarm Log Display screen.
2. Press [SELECT].

Enabling/Disabling the Alarm Log

- You can enable/disable the alarm log from the Alarm Configuration screen with the **Alarm Log** field.

❑ **To enable the alarm log from the command line, type**

ELA [ENTER].

❑ **To disable the alarm log from the command line, type**

DLA [ENTER].

Sample Portions of Log Display Configurations

Configuration	Description
Find Crit>unit=1-4	Displays all alarms from plant units 1-4
Find Crit>blk=boiler1, boiler2	Displays all alarms from blocks boiler1 and boiler2
Find Crit>blk=boiler1 & type=crit	Displays all critical alarms from block boiler1; the ‘**&**’ represents a logical ***and*** operation
Find Crit>blk=boiler1 | unit=1-5	Displays all alarms from block boiler1 and all alarms from plant units 1-5; the ‘**|**’ represents a logical ***or*** operation
Find Crit>blk=boiler1 | >unit=1-5	Displays all alarms from block boiler1 and all alarms from plant units 1-5
Find Crit>unit=1-4 Filt Crit>state=clear	Displays all alarms from plant units 1-4, except for cleared alarms

Alarm Inhibit

- You can prevent alarms from being generated by a ControlBlock or a Controller Processor.
- When a ControlBlock or Controller Processor is inhibited:
  - A clear message is sent on the PeerWay for each uncleared alarm from the ControlBlock or Controller Processor.
  - All alarms from the ControlBlock or Controller Processor are inhibited from generating and are not sent on the PeerWay.
- For more information about inhibiting alarms, see CC: 6.

Inhibiting Alarms from a ControlBlock

- You can use the **inhibit** function in ControlBlock logic to prevent alarms from being generated.

NOTE: The inhibit function does not work on MPCI controllers.

- Block alarms are inhibited only for those evaluation cycles in which the inhibit function is executed.

Inhibiting Alarms from a Controller Processor

- You can use the **inha** function in ControlBlock logic to prevent alarms from being generated by a Controller Processor.
- You can also use the **Alarm Inhib** field on the ControlFile Status screen to inhibit alarms from a Controller Processor.
- Once alarms are inhibited, they stay inhibited until the opposite function is executed—that is, until the **enba** function is used in ControlBlock logic, or until the **Alarm Inhib** field is changed on the ControlFile Status screen.

Alarm Suppression

- You can suppress active block alarms at a console.
- You specify which alarm types can be suppressed on the Alarm Configuration screen.
- You specify which key classes can suppress alarms on the Alarm Configuration screen.

What Happens When Alarms Are Suppressed

- When you suppress an active alarm from a block, all active alarms from that block that are of the specified alarm type(s) are suppressed.
- Suppressed alarms do not appear on alarm-type or alarm-area lists.
- Suppressed alarms appear in active and cleared suppressed alarm lists.
- If the console regenerates alarms, the suppressed alarm lists are not rebuilt. Instead, regenerated alarms that were in the suppressed alarm lists appear in the alarm type lists.

Suppressing an Alarm

❑ **To suppress an active alarm:**

1. Call up an alarm list that contains the alarm that you want to suppress.
2. Cursor to the alarm, and type **S**.

Unsuppressing an Alarm

❑ **To unsuppress an alarm:**

1. Call up a suppressed alarm list (active or cleared) that contains the alarm that you want to unsuppress.
2. Cursor to the alarm, and type **E**.

Callup and Display Buttons

- **Callup** buttons are configured on the Alarm Annunciation screen. Callup buttons and assigned commands can be viewed on the Callup Buttons screen. For more information, see CC: 2.
- Console **display** buttons are configured on the Display Buttons screen.

Alarm Annunciation Screen (ACC)

- The Alarm Annunciation screen is used to configure commands and alarm annunciation information for callup buttons.
- **No.** specifies the command number. The command number is an arbitrary designation used for assigning commands. The command number must be at the top of the screen to configure the associated command.
- To get a command number to the top of the screen:
  - Type the desired command number in the No. field, and press [ENTER], or
  - Cursor to the desired command number and press [SELECT].
- **Command** contains a set of keystrokes that are executed from the command line when the callup button is pressed. For example, the command **G:1*** executes the keystrokes **G : 1** [ENTER] and calls up the display for group 1.

  Certain groups of characters can be configured to represent special functions. These are listed in the table on the next page.
- **Push Button** specifies to which push button the command is assigned. The Push Button entry consists of the callup bank number, a period, and the button number. For example, button number 12 on the second bank of callup buttons on a Multitube Command Console is configured as **Push Button⇒2.12**.

  If the entry is **none**, then no push button is associated with the command. However, if plant units, areas, or block addresses are configured in the **Associated Unit**/**Area**/**Block Addresses** field, then the command is executed if one of the blocks is in alarm and the [ACTIVE ALARM] button is pressed.
- **LAMP** or **LED** indicates whether or not the push button lights when a block in the **Associated Unit**/**Area**/**Block Addresses** field is in alarm. If the alarm is unacknowledged, the light also blinks.
- **Associated Unit**/**Area**/**Block Addresses** is used to associate units, areas, or blocks with a callup button. If a block in this field is in alarm and **LAMP** or **LED** is **yes**, then the callup button lights. If a plant, area, or block in this field is in alarm and the **[ACTIVE ALARM]** button is pressed, then the command in the **Command** field is executed.

Display Buttons Screen (CDB)

- The Display Buttons screen is used to configure commands for Multitube Command Console display buttons.
- The left part of the Display Buttons screen shows a representation of the Display Buttons keyboard. The right part of the display lists the commands that are assigned to the buttons. These commands can be executed by cursoring to the button number on the left side of the display and pressing [ENTER].
- The **Command** field contains a set of keystrokes that is executed from the command line when the display button is pressed. For example, the command **G:1*** executes the keystrokes **G : 1** [ENTER] and calls up the display for group 1.

❑ **To select a command so that it can be configured:**

- Type the number in the Button field and press [ENTER], or
- Cursor to the button number on the left side of the display, and press [SELECT].

Certain groups of characters can be configured to represent special functions. These are listed in the table below.

Special Command Characters

Characters	Description
*	Represents the [ENTER] button.
# (block input)	Changes the status of a discrete input.
~	Used at the beginning of a command to prevent the keystrokes from being entered on the command line.
#∧C	Performs a console restart.
#∧H	Moves the cursor to the left.
#∧J	Moves the cursor down.
#∧K	Moves the cursor up.
#∧L	Moves the cursor to the right.

Loading the Plant Program Using a Script File

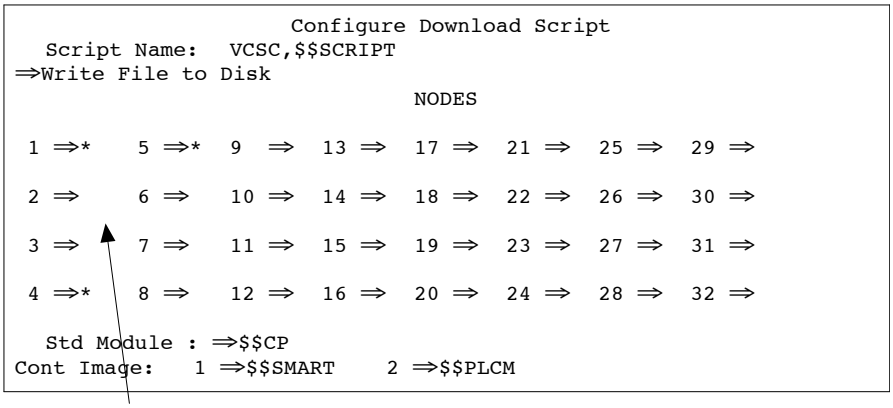
- Script files can be configured to simplify the loading of files. You specify a $$CPxxxx file, additional images, and the ControlFile nodes to which these files are to be loaded.
- A script file can be generated by typing **CDS ,** [*filename*]. If the filename is omitted, the default script file $$SCRIPT is displayed.
- The Configure Download Script screen contains five pages for each script file. Each page lists a $$CPxxxx file and additional images that are to be loaded to specified ControlFiles.
- Script files are saved on disk and are part of the Plant Program folder. Script files are downloaded to ControlFiles using the “Disk Load Program or Script” disk operation.
- A typical installation would have one script file. Each combination of a $$CPxxxx file and additional images that are required for one or more ControlFiles is defined on one page of the script file. If there are more than five combinations, additional script files are used. When the script file is downloaded to all nodes on the PeerWay, all ControlFiles are loaded with the appropriate Plant Program files.

Configure Download Script Screen

- **Script Name** specifies the user-assigned name of this script file. The script file name can contain up to nine alphanumeric characters, and the file can contain up to five pages of scripts.
- **Nodes** specify the nodes that can be downloaded with the files specified on this page of the script file. All nodes with an asterisk can be loaded with the specified $$CPxxxx file and additional images if the nodes are included in the **Node Range** field on the Disk Load Program or Script disk activity screen. A node can be included on only one page of an individual script file.

Configure Download Script Screen (continued)

- **Standard Module** specifies the name of the $$CPxxxx file that is associated with this page of the script file. Press [NEXT OPTION] for the file choices.
- **Write File To Disk** saves the script file to disk when you press [ENTER] with the cursor on this field. You do not need to perform this operation to write a file to disk; any time you leave the Configure Download Script screen after changing a script, the file is automatically saved.
- **Cont(roller) Images** specifies the additional Controller Processor images that are associated with this page of the script file. The first image listed is additional image 1, the second is additional image 2, and so on. Use the [NEXT OPTION] button to access the image choices.
- For more information about the Configure Download Script screen, see DT: 2.



Nodes 1, 4, and 5 have an asterisk. The script file will attempt to download the files $$CP, $$SMART, and $$PLCM to nodes 1, 4, and 5.

One Page of a Configure Download Script Screen

ControlFile Status Screen (CFS)

The ControlFile Status screen is shown below.

```
                    CONTROL FILE STATUS                01-Nov-92

Node Address>2  File Status->Norm  Batch CP  Batch Config NVM 56K
Left CP: Boot 4.02  Prgm 15.03  Avail Links 40  Idle Time 62.%
Right CP: Boot: 4.02
Left Program NVM Free 46. %   BRAM V1.14        Soft Count 0
Additional Images:  1. MPC2+
Config NVM Free 156K
```

	A	B	C	D	E
	ALARM				
Control Type	MPC	PLC	MPC-P		
Boot Rev	5.17	5.17	5.20		
Prgm Rev	12.48	12.48	12.48		
Idle Time	40. %	52. %	44. %		
Free Space	26. %	38. %	42. %		
Avail Links	36	36	38		
Avl Trnd Spc	1240	1264	1228		
Primary Status	→Norm	→Norm	→Norm		
Alarm Inhib	→no	→no	→no		
LOCAL Inhib	→no	→ no	→no		
Start Cal					
Jumper Code					
SC Timeout	None	None	None		
Scan Time	.5 S	1. S	.5 S		
NVMem Used	56K	48K	48K		

ControlFile Status Screen

The top part of the screen shows Coordinator Processor (ControlFile) information.

- **File Status** indicates the normal/standby status of the Controller Processors. Any changes to this field affect all Controller Processors. Also, if one or more Controller Processors are in stand-by, this field shows “Standby”.
- The field immediately to the right of **File Status** shows the type of Coordinator Processor card in the ControlFile.
- **Batch Config NVM** shows the amount of nonvolatile memory allocated to batch processing.
- **Left CP: Boot** and **Right CP: Boot** show the software revision level of the CP boot ROMs. The CPs can be switched by cursoring to the **CP: Boot** field of the currently inactive CP and pressing [ENTER].

ControlFile Status Screen (continued)

- **Prgm**, **Avail Links**, and **Idle Time** appear next to the currently active CP. **Primary** shows the CP software revision level.
- The first field on the next line shows which CP is currently the primary (Left or Right). **Program NVM Free** shows the space available in nonvolatile memory. The next field shows which type of nonvolatile memory is in the ControlFile.
- **Soft Count** shows counting errors that occurred during background diagnostics.
- **Additional Images** indicate other images that are loaded into CP.
- **Config NVM Free** shows the amount of unused nonvolatile memory.

The bottom part of the screen shows Controller Processor information.

- **Alarm** indicates that there is an active alarm that would cause a controller switch.
- **Control Type** shows the software image that is running on the Controller Processor.
- **Boot Rev** shows the software revision level of the Controller Processor boot ROM.
- **Prgm Rev** shows the Controller Processor software revision level.
- **Primary Status** shows the normal/standby status of the Controller Processor.
- **Alarm Inhib** determines whether or not alarms are prevented from being sent by the Controller Processor.
- **LOCAL Inhib** determines whether or not blocks in the Controller Processor are prevented from being in LOCAL mode.
- **Start Cal** is used to calibrate A/D converters on Controller Processor.
- **Jumper Code** shows jumper settings, which determine MPC functionality and image selection. For more jumper information, see SV: 4.
- **SC Timeout** indicates a timeout value for SCI communications with blocks on the Controller Processor.
- **Scan Time** indicates the scan time for the Controller Processor.
- **NVMem Used** shows the amount of memory presently allocated to the Controller Processor.
- For more information about ControlFile Status screen, see CB:3.

Color Configuration

- Console colors can be configured using the Master Color Selection and Color Configuration screens.

Master Color Selection Screen (CCP)

- The Master Color Selection screen is used to configure the background/foreground color combinations used by the console.
- 16 color combinations can be configured for a console.
- **Background** colors must be chosen from the configured foreground colors.
- Each foreground color has an individual background intensity. The background intensity of a color matches the foreground intensity of that color.
- Press [PAGE AHEAD] to get to the Color Configuration screens.

Color Configuration Screens (CCU)

- The Color Configuration screens are used to assign color combinations to system display components.
- The color palette from the Master Color Selection screen is shown on each Color Configuration screen.
- For more information about configuring colors, see CC: 2.

Group Displays

- The table below shows how to **add a faceplate** to a group display.

Adding a Faceplate to a Group Display

Entry	Faceplate Displayed
tag or address [ENTER]	Standard faceplate: **continuous** faceplate for **continuous** function ControlBlocks, **discrete** faceplate for **discrete** function ControlBlocks
#tag or #address [ENTER]	Nonstandard faceplate: **discrete** faceplate for **continuous** function ControlBlocks, **continuous** faceplate for **discrete** function ControlBlocks
+tag or +address [ENTER]	Continuous faceplate
-tag or -address [ENTER]	Discrete faceplate

❑ **To delete a faceplate from a group display:**

1. Cursor to the faceplate.
2. Type **none** and press **[ENTER]**.

❑ **To switch between displaying the faceplate descriptors and tags or addresses for a group:**

— Press **[EXCH]** when the cursor is at the home position.

❑ **To switch between displaying the faceplate descriptor and tag or address for a faceplate:**

— Press **[EXCH]** when a single faceplate is highlighted.

- For more information about configuring group displays, see CC: 2.

Node Ownership and Plant Unit Ownership

- Node ownership and plant unit ownership work together to determine if a console can configure blocks or batch tasks and to determine which alarms are stored in console RAM. The table below describes how node ownership and plant unit ownership work together.
- Node ownership is configured on the Plant Status screen. For more information, see the Plant Status screen heading in this section or CC: 2.
- Plant unit ownership is configured on the Plant Unit Configuration screen. For more information, see CC: 2.

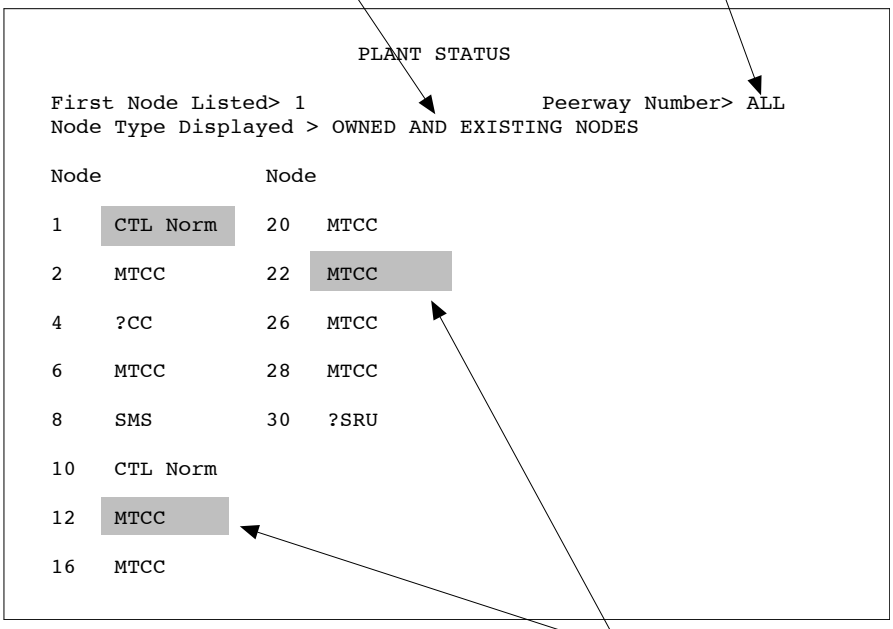
Node Ownership and Plant Unit Ownership Matrix

If:	then:	only if the console:
a block or batch task has a **non-zero** plant unit number,	the console can configure and control the block or batch task, and the console stores alarms from the block or batch task	owns the **plant unit** number.
a block or batch task has a plant unit number of **zero,**	the console can configure and control the block or batch task, and the console stores alarms from the block or batch task	owns the source **node**.
the source is not a block or batch task,	the console can configure and control the source, and the console stores alarms from the source	owns the source **node**.

Plant Status Screen

The types of nodes displayed. Use the [NEXT OPTION] button to step through the choices.

PeerWay number of displayed nodes.



Types of nodes that can be displayed:

CC	-- Pedestal Command Console
CTL	-- ControlFile
DI	-- Diogenes Interface
HIA	-- Highway Interface Adaptor
MC	-- MiniConsole
MTCC	-- Multitube or Hardened Command Console
RFI	-- Rosemount Factory Interface
RNI	-- RS3 Network Interface
SCI	-- Supervisory Computer Interface
SMS	-- System Manager Station
SRU	-- System Resource Unit
VAX	-- MicroVax to Peerway Interface

An owned node is backlighted. Cursor to the field and press [ENTER] to toggle between the owned and not-owned state.

Note: A node type preceded by a "?" means that the node type is no longer supported.

Process Graphics

- A process graphic display on an MTCC is a pictorial representation of plant conditions.

❑ **To call up the viewing mode of a graphic on an MTCC, type**

PG [*volume*] , [*filename*].

❑ **To call up the configuration mode of a graphic on an MTCC, type**

PGG [*volume*] , [*filename*].

- A process graphic is configured by creating various types of objects on the graphic. A list of object types available with vector graphic displays is shown on the next page.
- A **symbol** is a combination of one or more objects that represents a commonly used graphic feature. Symbols reside in the Process Symbols folder on the console disk. A standard symbol library exists in the Process Symbols folder. The user can also create symbols.
- For more information on configuring graphic displays, see CC: 3.

Graphics Object Types

Shapes

Arc — Displays an arc.
Box — Displays a box.
Triangle — Displays a triangle.
Circle — Displays a circle.
Line — Displays one or more lines.

Block information

Blkdes — Displays a block descriptor.
Mode — Displays the mode of a block.
Tag — Displays a block tag.

Block variable information

Disc — Displays one line of a discrete faceplate.
Hdev — Displays a bar with horizontal travel that shows the difference between two block variables.
Link — Displays the value of a block variable.
Ticbar — Displays a horizontal line that travels along a vertical axis. The line shows the value of a block variable.
Units — Displays the units of a block variable.
Vdev — Displays a bar with vertical travel that shows the difference between two block variables.

Permanent faceplates

Group — Displays a continuous or discrete faceplate.
Unit — Displays a faceplate from a unit display.
Overview — Displays a faceplate from an overview display.

Text

Text — Displays user-configured text. Can also be used to execute a command.

Trend information

Trend — Displays graphic trend information for a block variable from a trend file.

Batch faceplate

Bface — Displays a batch faceplate for use on the graphic display.

Display manipulation

Anchor — Provides a pointer for Pagefrwd, Pageback, and Text objects.
Icon — Hides a group of objects from view until selected.
Pageback — Executes a command when [PAGE BACK] is pressed.
Pagefrwd — Executes a command when [PAGE AHEAD] is pressed.
Target — Provides a target for quick cursor positioning.

Alarm Information

Alarm — Provides access to alarms from a process graphic.

Message Pairs

- Message pairs are used throughout the system to represent discrete states, such as in discrete ControlBlock faceplates and logic steps.
- Standard and user messages are arranged in pairs with one message for the ON (1) state and one message for the OFF (0) state.
- Each message can be up to **eight** characters long with each space counted as one character.
- User message pairs are numbered 1–155. Standard message pairs are numbered 1–100 and are identified with an asterisk (*).
- For more information about message pairs, see CC: 2.

Standard Message Pairs (MPS)

- Standard message pairs are preconfigured and are displayed on the STD Message Pairs screen.
- Standard message pairs can be changed by the user. However, standard message pairs are used throughout the system and care should be taken when configuring standard message pairs so that unwanted changes do not occur.
- Use [PAGE AHEAD] and [PAGE BACK] to navigate through additional standard message–pair screens.

User Message Pairs (MPU)

- User message pairs are configured by the user on the User Message Pairs screen.
- Use [PAGE AHEAD] and [PAGE BACK] to navigate through additional user message-pair screens.

Message Pair Screen Operations

- True and False correspond to the ON/OFF states of a discrete action.
- To add or change a message, cursor to the True or False field, type in the desired message, and press [ENTER].
- Attributes C1 through C6 determine the color (text and background) of the message. Attributes are assigned on the Color Configuration screen.

Standard Message Pairs 1-50

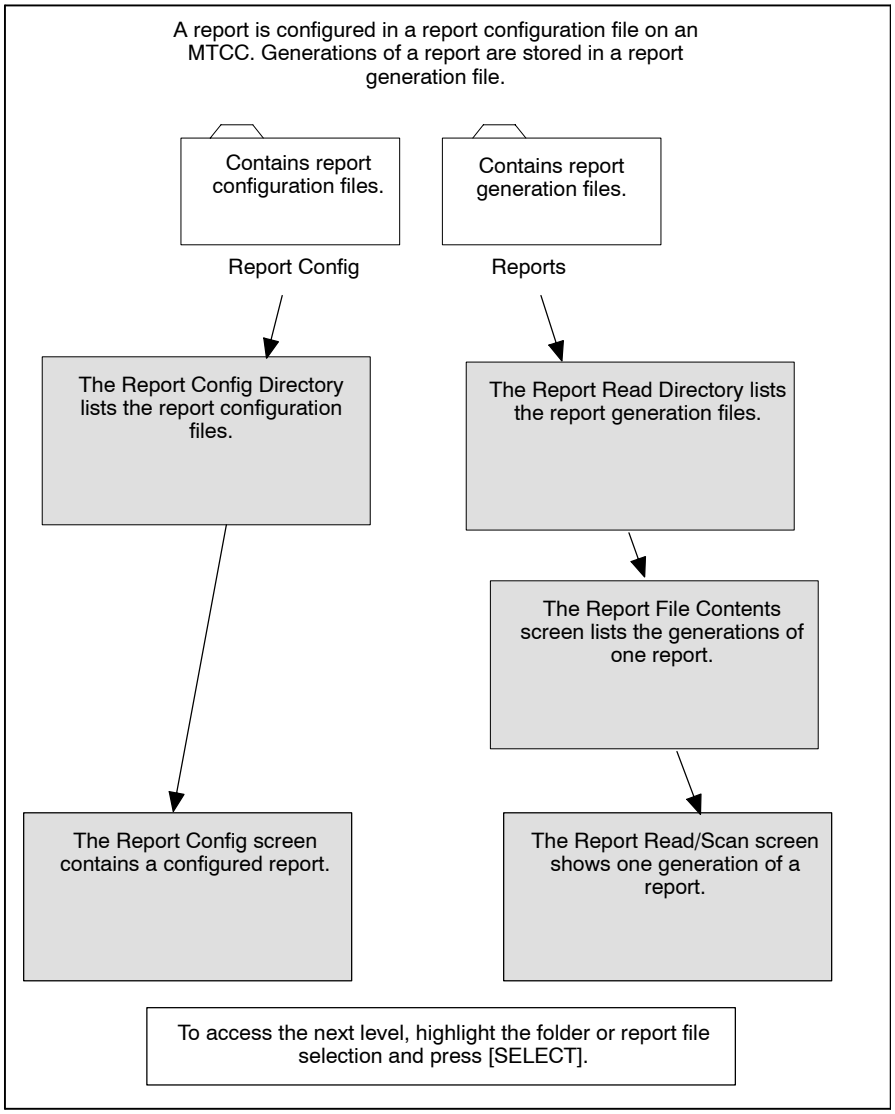
No.	True (on)	False (off)	No.	True (on)	False (off)
*1	ON	OFF	*26	REV CFRM	rev cfrm
*2	START	start	*27	OFF CFRM	off cfrm
*3	LOW	low	*28	OPN CFRM	opn cfrm
*4	HIGH	high	*29	CLS CFRM	cls cfrm
*5	FORWARD	forward	*30	RUN CFRM	run cfrm
*6	REVERSE	reverse	*31	SHUTDOWN	shutdown
*7	STOP	stop	*32	INTERLOK	interlok
*8	OPEN	open	*33	RESET	reset
*9	CLOSE	close	*34	POSN ‘A’	posn ‘A’
*10	RUN	run	*35	POSN ‘B’	posn ‘B’
*11	AUTO	OPERATOR	*36	‘A’ CFRM	‘A’ cfrm
*12	ENABLE	enable	*37	‘B’ CFRM	‘B’ cfrm
*13	PUMP	pump	*38	OPERATOR	operator
*14	AGITATE	agitate	*39	--AUTO--	--auto--
*15	TRANSFER	transfer	*40	STATUS	status
*16	DRAIN	drain	*41	FAILED	
*17	WASH	wash	*42	ON FAIL	on fail
*18	PROCEED	proceed	*43	LO FAIL	lo fail
*19	DISABLE	disable	*44	HI FAIL	hi fail
*20	DISABLED	disabled	*45	FWD FAIL	fwd fail
*21	CONFIRM	confirm	*46	REV FAIL	rev fail
*22	ON CFRM	on cfrm	*47	OFF FAIL	off fail
*23	LO CFRM	lo cfrm	*48	OPN FAIL	opn fail
*24	HI CFRM	hi cfrm	*49	CLS FAIL	cls fail
*25	FWD CFRM	fwd cfrm	*50	RUN FAIL	run fail

Standard Message Pairs 51-100

No.	True (on)	False (off)	No.	True (on)	False (off)
*51	IGNORE	ignore	*76	PMP CFRM	pmp cfrm
*52	FAILED	failed	*77	DRN CFRM	drn cfrm
*53	SW FAIL	sw fail	*78	WSH CFRM	wsh cfrm
*54	TRIPPED	tripped	*79	SETPOINT	setpoint
*55	LOCKOUT	lockout	*80	NEXT	next
*56	MCC OFF	mcc off	*81	ON	off
*57	TRAVEL	travel	*82	ON	off
*58	ABORT	abort	*83	STOP	stop
*59	ABORTED		*84	START	stop
*60	NOT USED	not used	*85	RUN	stop
*61	CLS TRP	cls trp	*86	RUN	stop
*62	RUN TRP	run trp	*87	OPEN	close
*63	RETRY LM	retry lm	*88	-OUTPUT-	-output-
*64	MCC ALRM	mcc alrm	*89	ALARM	ok
*65	V TRAVEL	v travel	*90	ALARM	ok
*66	FORWARD	forward	*91	ON	OFF
*67	REVERSE	reverse	*92	ON	OFF
*68	HOLD	hold	*93	START	STOP
*69	HOLD		*94	START	STOP
*70	STATUS	status	*95	RUN	STOP
*71	PROCEED		*96	RUN	STOP
*72	RUN		*97	OPEN	CLOSE
*73	ACTIVE	active	*98	OPEN	CLOSE
*74	INACTIVE	inactive	*99	ALARM	OK
*75	STP CFRM	stp cfrm	*100	ALARM	

Reports

- For more information about reports, see CC: 4.



Report File Structure

Report Object Types

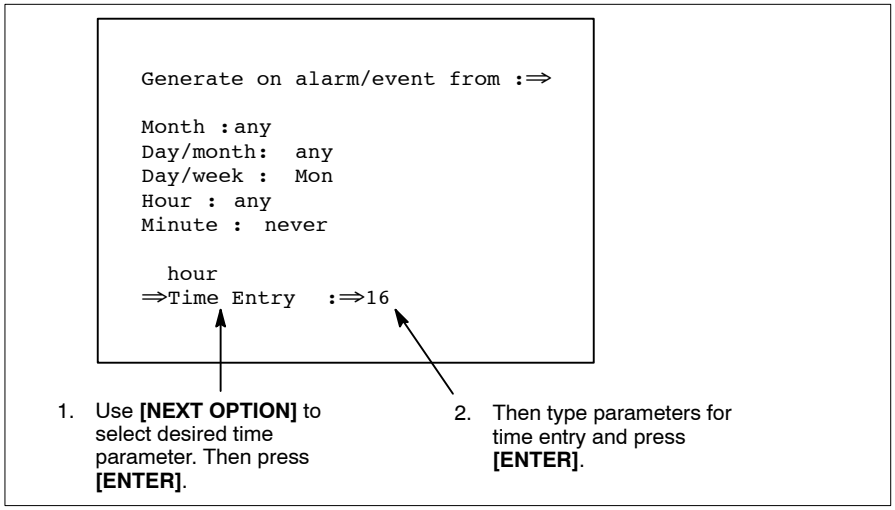
Description
Text
text — allows up to 40 characters of text to be typed directly into the report. ban1 — displays a large-type banner or heading. Each character will be 10 lines high and 8 columns wide. Maximum of 10 characters. ban2 — displays a large-type banner or heading. Each character will be 10 lines high and 16 columns wide. Maximum of 5 characters.
Operator comments
comment — allows the operator to enter up to 70 characters in a report.
Style of report
print — permits entry of a hexadecimal code to control printer operation. newpage — commands the printer to execute a form feed.
Block information
tag — displays the tag of the specified block. addr — displays the address of the specified block. descriptor — displays the descriptor of the specified block. func — displays the block function configured for the specified block. mode — displays the current mode of the specified block. units — displays the engineering units associated with a variable. value — displays the value of the specified continuous or discrete variable. accumulate — displays a result calculated from several value objects.
Trend file information
trend log — creates a table of periodic readings of one or more variables over a specified time span.
Time and date
time — inserts the current time when the report is generated. date — inserts the current date when the report is generated.
Alarm and event lists
list — inserts the current contents of an alarm or event list into the report.
Generated report
generate — allows another generated report to be included in the report being configured.
SQC
There are many report objects to support SQC. For more information, see SR: 4.

Report Configuration Screen (RG *filename*)

❑ **To configure a report to generate on time and date:**

1. From the Report Configuration screen, cursor to the **Time Entry** field; and use **[NEXT OPTION]** to select the desired parameter.
2. Cursor to the arrow immediately to the right, and enter the desired month, day, hour, and so forth.

- The example below shows how to configure a report to generate at 4 P.M. each Monday.



❑ **To configure a report to generate on alarms and events:**

1. From the Report Configuration screen, cursor to the arrow after the **Generate on alarm/event from** field.
2. Enter block addresses for alarm and event information. Examples are shown below:
   - —ControlBlock continuous input (alarm) =2C-12/A
   - —ControlBlock continuous output (alarm) =2C-12
   - —ControlBlock discrete output (alarm and event) =2C-12/b
   - —Input/output block (alarm and event) =2CA101

❑ **To generate a report by command:**

- Cursor to the Command field. Press [NEXT OPTION] until **Generate this report** appears. Press [ENTER].

Trending

- Trending can be enabled and disabled on an MTCC from the command line or from the Trend File Setup screen.
- For more information about trending, see CC: 5.

Enabling and Disabling Trending

❑ **To enable trending from the command line on an MTCC, type ET [ENTER].**

❑ **To disable trending from the command line on an MTCC, type DT [ENTER].**

Trend File Setup Screen (TFS)

- The Trend File Setup screen shows information about the trend files at a console.

Trend File Configuration Screen (TFC)

- The Trend File Configuration screen is used to configure each trend file on the MTCC.
- Trend file 0 stores real-time data only. No history data is available.
- Trend file 0 is a trend file in RAM only. It is not saved on disk. With trend file 0,there is no control over time duration; frequency (60 data bits) controls duration. For example, if the frequency is every minute, the duration is one hour. Your sample rate is 60.
- **Variables Max** specifies how many variables can be configured for the trend file. This field must have a non-zero number in it in order to enter any variables on this screen.

Trend Group Configuration Screen (TGC)

- The Trend Group Configuration screen is used to configure a trend group. The trend group can then be graphically displayed on a Trend Group screen. A trend group can contain up to four block variables that have been configured in trend files on a console.
- **Node** determines the PeerWay node where the desired trend file exists. An asterisk (*****) indicates the default node configured on the Disk Directory PeerWay screen.
- The **Vertical Graph Scaling** fields specify the default scaling for the trend group graph. Whenever the Trend Group screen is called up for the trend group, the default scaling is displayed.

Trend Group Menu Screen (TGM: *or* TGD:)

- The Trend Group Menu lists all configured MTCC trend groups and the variables in them. A trend group can be called up by cursoring to the group and pressing [SELECT].

Trend Group Screen (TG:#)

- The Trend Group screen displays graphic trend information about the block variables in the trend group.
- If the lower right corner of the graph shows **Current**, then the most recent 60 samples are shown. The bottom of the screen shows the current values.
- To show data from a previous time, cursor to **Start Time** and enter the desired time. The lower right corner of the graph shows **History**. To get back to **Current** data, cursor to **Start Time** and press [ENTER].
- The right hand side of the graph shows the **percent engineering units scaling**. The default scaling is configured on the Trend Group Configuration screen. The left side of the display shows the graph scaling of one of the trend group variables (the variable shown in the field immediately below the scaling). This variable can be changed by cursoring to the variable and pressing [NEXT OPTION], and then [ENTER].
- The graph can be configured to show 60, 120, 240, or 480 samples of data.
- The cursor can be moved into the graph and used as a **slidewire**. The variable values at the slidewire location are shown at the bottom of the screen. To get the most recent values to appear again, move the slidewire to the far right of the graph.

Configuring Trend Files on an MTCC

- An MTCC has one disk and, therefore, one set of trend files.
- An MTCC can contain up to 10 trend files.
- Up to to 80 block variables can be configured in one trend file.
- Up to to 200 block variables can be configured per MTCC.

Reloading Archived Trend Files

- Care must be taken when reloading archived trend files to avoid duplicate file names in the Trend Data folder.

  Trend file names should be changed when saving the file to tape or when the file is restored to disk. The first four characters of the file name must remain the same; therefore, to change the file name, simply add one or more characters to the existing file name. For example, *trf1* could be renamed *trf17/88*.

Tuning Display Screen (TD:)

- The Tuning Display screen combines faceplates, loop variables, and trending capabilities on one screen to help tune a control loop.
- Up to 10 tuning displays can be configured for a console.
- The ControlBlocks do not have to be configured in a trend file to be trended on the Tuning Display.
- Faceplates on the Tuning Display have the variable values at the bottom of the faceplate.
- The Tuning Display screen contains a trend display window that can be used to trend up to four faceplate variables. Up to 300 data samples are stored for each trended variable. Up to 50 data samples can be shown on the trend window for a variable.
- Faceplate variables are trended and stored only while the Tuning Display screen is displayed on the console.

Tuning Display Directory (TDD)

- The Tuning Display Directory screen is used to assign titles to the Tuning Display screens.
- For more information about Tuning Displays, see CB: 2.

Tuning Display Operations

❑ **To configure a ControlBlock for the display:**

- Cursor to the > symbol at the top of the screen and type the tag of the ControlBlock. A ControlBlock must be configured in this field to be tuned or trended.

❑ **To trend a ControlBlock variable:**

- Cursor to the desired variable value in the faceplate and press **[SELECT]**.

❑ **To remove a ControlBlock variable from the trend display window but continue trending the variable (this method can be used to reduce clutter in the trend window):**

- Cursor to the desired variable value in the faceplate and press **[SELECT]** or **[ENTER]**.

❑ **To discontinue trending a ControlBlock variable:**

1. Cursor to the desired variable value in the faceplate.
2. Press [NEXT OPTION] until the value is white.
3. Move the cursor away from the value.

❑ **To change the color of a ControlBlock variable:**

1. Cursor to the desired variable value in the faceplate.
2. Press [NEXT OPTION] until the desired color appears (white discontinues trending of the variable).

❑ **To change the trend sample frequency:**

- Cursor to the **Sample Freq** field and enter the desired value. If the **Sample Freq** field is changed, the trend display and data storage are cleared.

Tuning Display Operations (continued)

❑ **To use the slidewire:**

- Move the cursor into the trend window. The trend display is now frozen. The time of day and ControlBlock variable values associated with the cursor location are displayed below the trend window. To view trend data for other times, move the slidewire to the left or right edge of the trend display and continue rolling the trackball. To end the slidewire, move the cursor up or down out of the trend window.

❑ **To change the ControlBlock tuning variables:**

- Cursor to the tuning variable and make the desired change.

❑ **To change the ControlBlock that has its tuning variables displayed:**

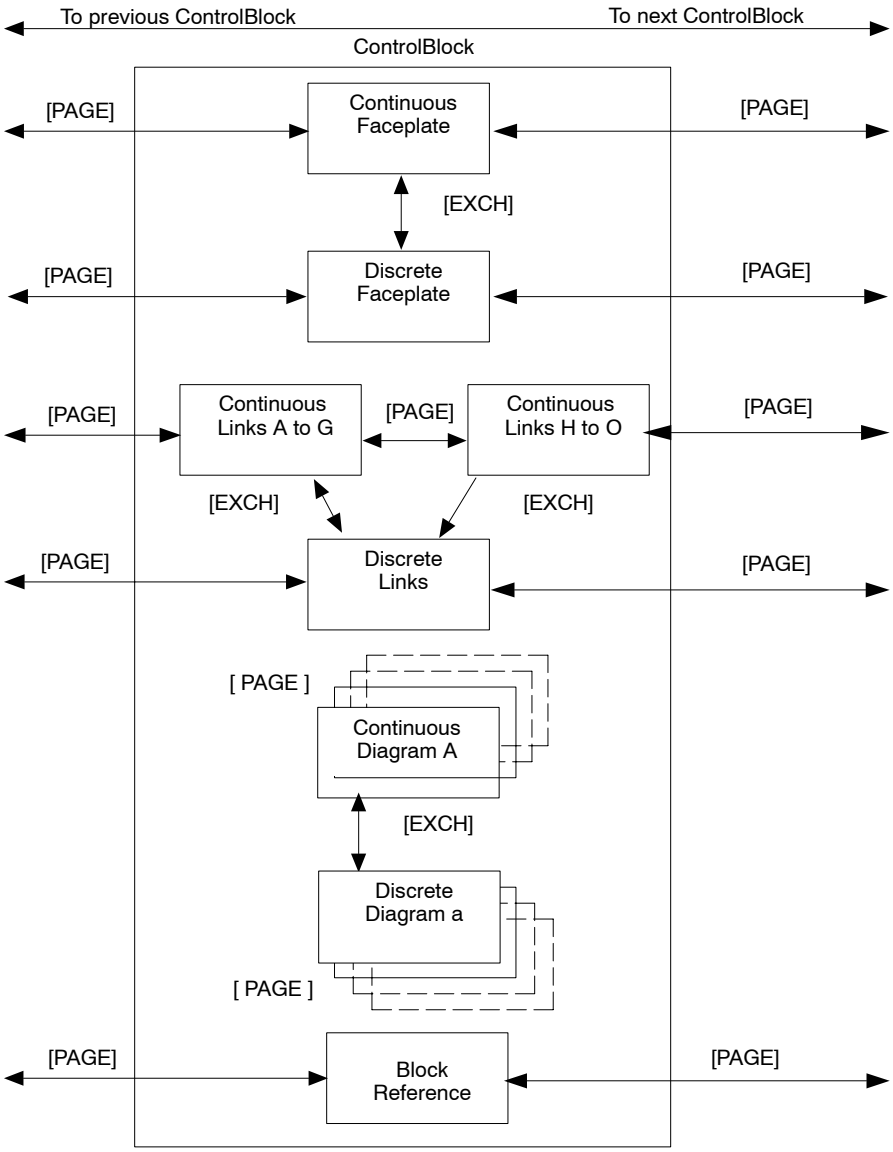
- Cursor to the **Tag**/**Addr** field and press **[SELECT]**.

❑ **To change the faceplate and trend window scaling:**

1. Cursor to the scaling variables at the **left** of the faceplate. The values represent the percentage of the ControlBlock engineering units range.
2. Make any desired changes. The changes will be reflected in the faceplate bars and the trend window.

Section 2: ControlBlocks

ControlBlock Screen Relationships



[PAGE]=[PAGE AHEAD] and [PAGE BACK] buttons

Continuous Faceplate Screen

The following figures describe the configuration of a Continuous Faceplate.

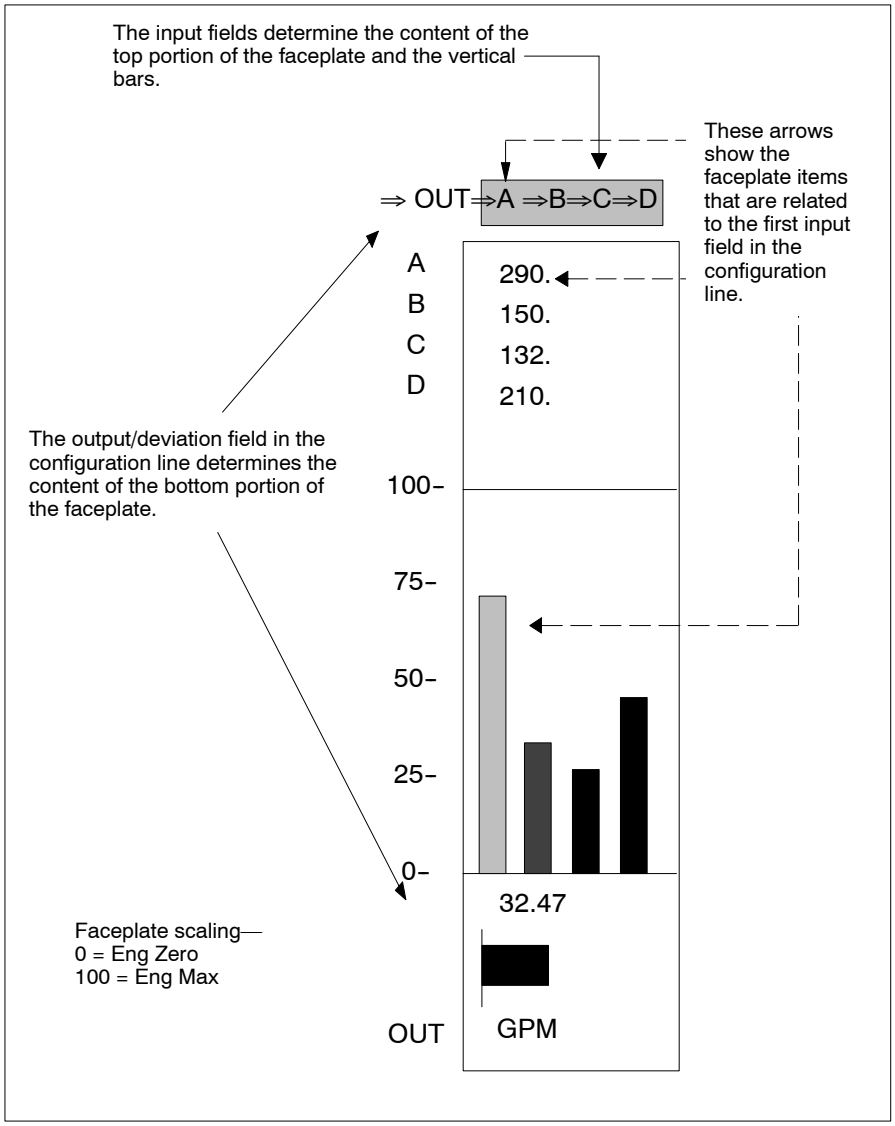


Figure 2.1. Continuous Faceplate Configuration

Table 2.1. Continuous Faceplate Configuration Line Fields

Field	Access Level	Description	Allow-able Entries
Output/ Deviation field	Conf	Determines the content of the bottom of the faceplate including value, horizontal bar, and units. OUT = continuous output All other entries display a zero-centered bar that represents the deviation of the first input field from the second input field. The entries display different percentages of the engineering units scale on the horizontal bar. The engineering units scale is configured on the Continuous Links screen. DEV = Max length of bar is 100% of scale. DV2 = Max length of bar is 50% of scale. DV3 = Max length of bar is 20% of scale. DV4 = Max length of bar is 10% of scale DV5 = Maximum length of bar is 5% of scale. DV6 = Maximum length of bar is 2% of scale. EXAMPLE: if an input register has an engineering scale of 0 to 200 units and an entry of DV4 (10% of scale), then the output/deviation bar reaches its maximum length when the deviation is 10% of scale, or 20 units (10% of 200).	NONE, **OUT,** DEV, DV2, DV3, DV4, DV5, DV6
First input field	Conf	Determines the first input register for the faceplate. The first value shown in the top part of the faceplate is the value of the first register; first bar from the left corresponds to this register.	NONE, **A** to O
Second input field	Conf	Determines the second input register for the faceplate. The second value shown in the top part of the faceplate is the value of the second register; second bar from the left corresponds to this register. This field defaults to being selected and is highlighted.	NONE, A to O **Default= B**
Third input field	Conf	Determines the third input register for the faceplate. The third value shown in the top part of the faceplate is the value of the third register; third bar from the left corresponds to this register.	NONE, A to O **Default= C**
Fourth input field	Conf	Determines the fourth input register for the faceplate. The fourth value shown in the top part of the faceplate is the value of the fourth register; fourth bar from the left corresponds to this register.	NONE, A to O **Default= D**

NOTE: Bold text indicates default selections.

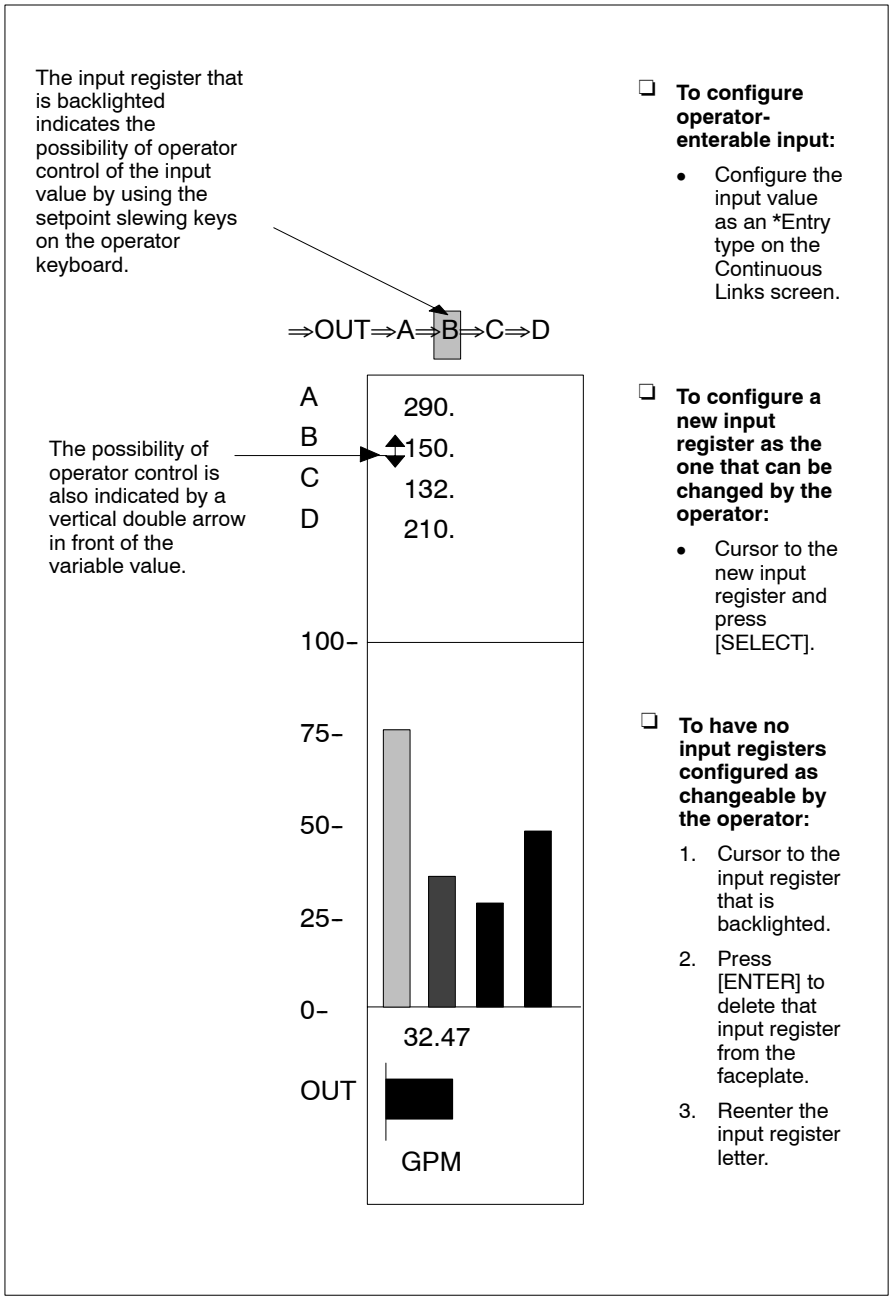


Figure 2.11. Configuring Operator-Enterable Input

Continuous Links Screen

- The **Source** field is used to enter the input. The input types are listed below.

Continuous Input Types

To configure this input type:	Enter this in the Source field:
Input block, such as AIB, MIB	address or tag (=2AA101)
ControlBlock output (Q)	address or tag (=3C-12)
ControlBlock input register (A through O)	address/input or tag/input (CALC-11/E)
Input value that can be changed by an operator or configuror	*ENTRY
Input value that can be changed by a supervisor or configuror	*SUPVSR
Input value that can be changed by a configuror or ControlBlock logic	*VALUE, *TIMER, *COUNTER, *NONE

- **Hold** specifies whether or not the block output is held at its last value if a bad signal is received from the input source.
- The **Units** field specifies the user-defined, engineering units associated with the input.
- **Scaling**: ControlBlock continuous inputs are scaled according to **Eng Zero** and **Eng Max**. The **Eng Zero** and **Eng Max** values do not limit the value of the continuous input.

  ControlBlock inputs can have scaling values assigned to them by other sources. An input with an asterisk (*) in front of the **Eng Zero** value has the same scaling values as the configured input source. An input with a register letter in front of the **Eng Zero** value has the same scaling values as that register.
- **Faceplate representation**: The **Eng Zero** value is represented as the 0 height on the ControlBlock faceplate. The **Eng Max** value is represented as the 100 height on the ControlBlock faceplate.

  Values below the **Eng Zero** value are represented by a bar below the 0 height. Values above the **Eng Max** value are represented by a bar above the 100 height. Values outside of the **Eng Zero**-**Eng Max** range are nonlinearly represented on the faceplate.
- For more information, see CB: 3.

Continuous Diagram Screen

- The Continuous Diagram screens are used primarily to configure alarm limits for each continuous input and the continuous output.
- **Sample Time**, **Alarm Priority**, **Plant Unit**, **Auto Lock**, **OPR Alm Ent**, **Dev Crit**, and **Dev Adv** apply to the entire ControlBlock.
- **Sample Time** specifies how often the ControlBlock sends an output. The Controller Processor **Scan Time** on the ControlFile Status screen determines how often the ControlBlock scans inputs and performs calculations. **Sample Time** cannot be shorter than **Scan Time**. If they are the same, **Sample Time** is backlighted.
- **Auto Lock** indicates whether or not the ControlBlock is locked in AUTO mode.
- **OPR Alm Ent** fields determine whether or not the operator can modify critical and advisory alarms (**Crit**) or advisory alarms (**Adv**) for the ControlBlock.
- **Dev Crit** and **Dev Adv** apply to the PV-SP deviation.
- For more information, see CB: 4.

Discrete Faceplate Screen

Discrete Faceplate Screen

To make this appear on a faceplate:	Type this in the Display field:	Ex
Discrete input state	A discrete input (@ a through @o)	**@a**
Discrete output state	A discrete output (a through p)	**d**
Continuous variable value	A continuous variable (A through O, Q)	**A**
Continuous variable value, operator enterable	# and the continuous variable (#A through #O, #Q); the position (line) above must be empty.	**#B**
Continuous variable, engineering units	% and the continuous variable (%A through %O, %Q)	**%Q**
Continuous variable, engineering units, with message pair color	%, the continuous variable (%A through %O, %Q), a dash, and the message pair color (1 through 6)	**%A-2**
Std. Msg. Pair-true condition	The standard message pair number (*1 through *100) and +	***78+**
Std. Msg. Pair-false condition	The standard message pair number (*1 through *100) and -	***32-**
User Msg. Pair-true condition	The user message pair number (1 through 155) and +	**78+**
User Msg. Pair-false condition	The user message pair number (1 through 155) and -	**32-**
Symbols	A single symbol (such as *, -, =, +) displays a line filled with that character.	**-----**

- **Unit** fields 1, 2, 3 specify three lines of the discrete faceplate that appear on a Unit faceplate.
- **Overview** field specifies one line of the discrete faceplate that appears on an Overview faceplate.
- For more information about ControlBlock discrete faceplates, see CB: 5.

Discrete Links Screen

- The Discrete Links screen identifies the discrete inputs of a ControlBlock.
- **Source** field is used to enter one of the input types listed below:
  - a CIB, COB, DIB, or DOB tag or address (=3AA101)
  - a ControlBlock discrete output (=6F–14/e)
  - a manually operated discrete input from the table below.

Discrete Input Sources

Type of Input	"Source" Field Entry	Function or Example
Discrete or Contact Input Block	Address or Tag	Example: =2AA101
Output of another ControlBlock (a through p)	Address/input or Tag/input	Example: =3D–15/c
Input value that can be changed from the console keyboard or by a Batch program.	*M ON	Remains OFF until turned ON. Remains ON for one evaluation cycle.
	*M OFF	Remains ON until turned OFF. Remains OFF for one evaluation cycle.
	*TOGGLE	Changes state each time that the input is actuated with [ENTER].
	*SELECT	Denotes input as part of a group. When one input in a group is ON, all others in that group are OFF. Use [ENTER] to turn on one input.
	*VALUE	Denotes a value that can be changed from another source (such as Batch). Cannot be changed from the faceplate.
Fixed state input value that must be reconfigured to be changed.	*ON	Remains ON at all times. Cannot be changed from the faceplate
	*OFF	Remains OFF at all times. Cannot be changed from the faceplate.

- **State** shows the current state of the discrete input.
- **Msg** indicates the message pair assigned to the discrete input. The message pair for an input can be changed for use within the ControlBlock.
- For more information, see CB: 3.

ControlBlock Types

Control Block Types

Abbrev.	Function
ATPID	Autotuning
D	Derivative-only control
DT	Dead Time
I	Integral-only control (step invariant form)
IB	Integral-only control (bilinear transformation form)
ID	Integral+Derivative control
LL	Lead /Lag
MAN	Manual
MATH	User-defined
P	Proportional-only control
PD	Proportional+Derivative control
PI	Proportional+Integral control
PID	Proportional+Integral+Derivative control
PLI	Piecewise Linear Interpolator
POLY	7th Order Polynomial
RB	Ratio /Bias
SS	Signal Selector
TOT	Stack Totalizer
TOTSP	Setpoint Totalizer
VLIM	Velocity Limiter
DASMC	Auto Sequence Motor Control
DASVC	Auto Sequence Valve Control
DDDMC	Dual Direction Motor Control
DDSMC	Dual Speed Motor Control
DISC	Discrete (User-defined)
DMC	Motor Control
DMVC	Motorized Valve Control
DVC	Valve Control
RBL	*Rosemount Basic Language Control

* Requires an RBL image.

ControlBlock Modes

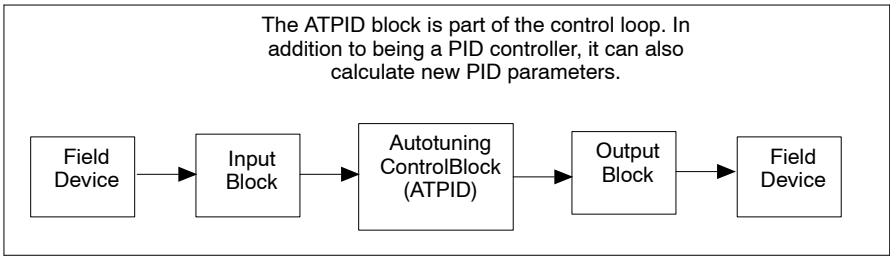
ControlBlock Modes and Applications

Mode	ControlBlock Applications	
	PID Family	**All Other Continuous Functions**
Auto	• Q is controller function driven. • Q can be overridden by logic. • Controller function responds to local setpoint (LS).	• Q is function driven. • Q can be overridden by logic.
Remote	• Q is controller function driven. • Q can be overridden by logic. • Controller function responds to remote setpoint (RS). • This is a normal mode for cascade secondary.	NA
Manual *or* Operator	• Operator directly manipulates Q from the console. • Q can be overridden by logic.	• Operator directly manipulates Q from the console. • Q can be overridden by logic.
Local	• Operator directly manipulates Q from the console. • Q **cannot** be overridden by logic.	• Operator directly manipulates Q from the console. • Q **cannot** be overridden by logic.
Comp SP	• Q is controller function driven. • Q can be overridden by logic. • Controller function responds to local setpoint (LS) which is determined by supervisory computer.*	NA
DDC	• Q is determined by supervisory computer.*	• Q is determined by supervisory computer.*

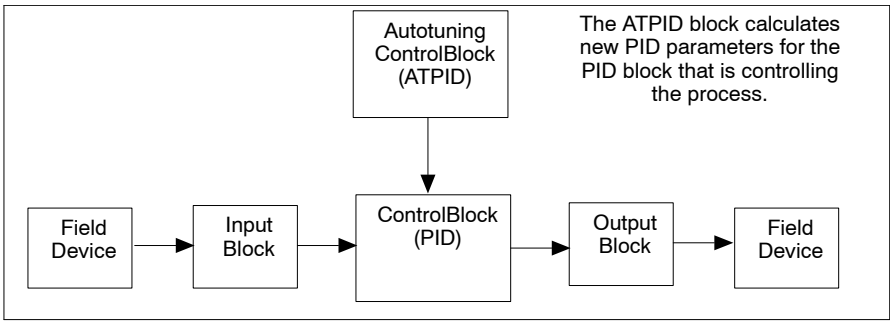
* For more information, see PW:1.

Autotuning (ATPID) ControlBlock

- An Autotuning ControlBlock (ATPID) is a PID controller with autotuning capabilities.
- The ATPID is designed to emulate the feedback procedure that a process control engineer uses to tune a control loop.
- The ControlBlock screens for an Autotuning ControlBlock are similar to the screens for configuring a PID block.
- An Autotuning ControlBlock uses two additional screens:
  - The ATC Configuration screen allows you to configure special autotuning parameters.
  - The ATC Diagnostics screen allows Rosemount personnel to evaluate the operation of the Autotuning ControlBlock.
- You can use an ATPID to control a loop directly or to provide PID parameters to other ControlBlocks only when autotuning is needed. (See figures below.)



ATPID ControlBlock as Part of a Control Loop



ATPID ControlBlock Providing PID Parameters for Another Block

Operating an Autotuning ControlBlock

The following is a typical operating sequence to tune a PID loop with the Autotuning ControlBlock:

1. Set up autotuning session.
2. Enable autotuning.
3. Create a setpoint disturbance.
4. Wait for the evaluation to finish.
5. Enter new PID values into the block.
6. Repeat disturbances.

   a. If ATPID block is part of control loop,
   enter final PID values into nonvolatile memory.

   or

   b. If ATPID block is being used to tune another block,
   disconnect ATPID block and resume normal operations.

Setting up an Autotuning Session

Before you enable autotuning, complete these steps on the ATC Configuration screen:

1. Enter the appropriate option in the **Open Loop Stable** field, which determines which algorithm to use: **yes**=open loop stable; **no**=open loop unstable.
2. Ensure that the PV—SP error is less than 1/2 of the smaller of the **Set Point Trig Mag** and **Load Trig Mag** field values. If the defaults for these fields are used, then the PV—SP error must be less than 2.5% of the PV.
3. Specify **Deriv Act→no** for the first evaluation. If you want derivative control for a second-order or higher process, first tune the loop with derivative control disabled (PI control); then enable derivative control, and tune the loop again.
4. Specify **ATune Opt→SP** for at least the first evaluation. The ATPID block calculates a deadtime only on setpoint triggers. If a deadtime exists for the process, the initial evaluation should be triggered by a setpoint change. Otherwise, enter an accurate value in the **Process Deadtime** field.

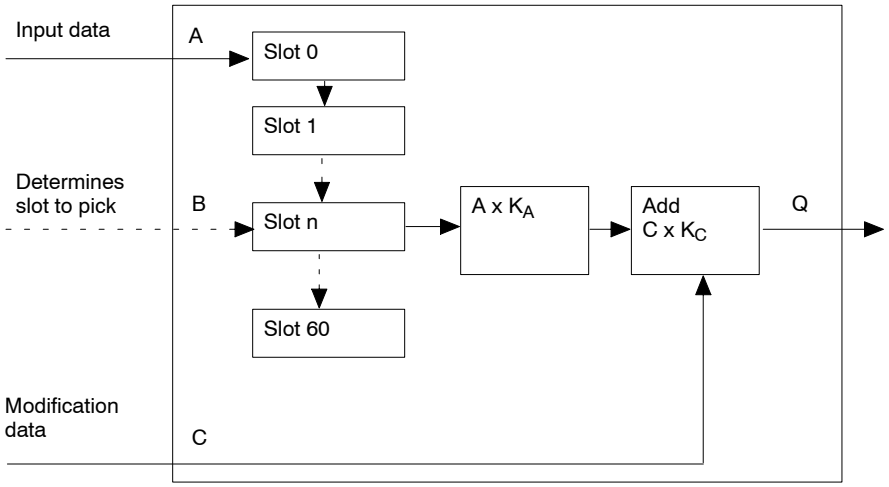
   ATune Opt determines the type of disturbance that triggers an autotuning evaluation: **SP**=setpoint changes only; **Load**=load changes only; **SP+Load**=both setpoint and load changes.
5. Set up a method to monitor the process using trending or tuning displays.
6. In the **Loop Damping** field, enter .25 for quarter-wave damping and then enter an appropriate value in the **Minimum Period** field.

If you are using an ATPID block to tune another block, complete these additional steps before enabling autotuning:

1. Link the PV and LS of the PID block to be tuned to the ATPID.
2. Record present PID values for future reference. Enter the values into the ATPID block for the first evaluation.
3. Observe the action of the block to be tuned (either direct or reverse). Use the same action for the ATPID.
4. Ensure both ATPID and PID blocks are in AUTO mode.
5. On the block to be tuned, ensure that the local setpoint is operator enterable. If another link is configured, remove the link temporarily while using autotuning. Record the link and reconnect it after the tuning session is complete.

- For more information about the Autotuning ControlBlock, see CB:8.

Dead Time (DT) Function ControlBlock

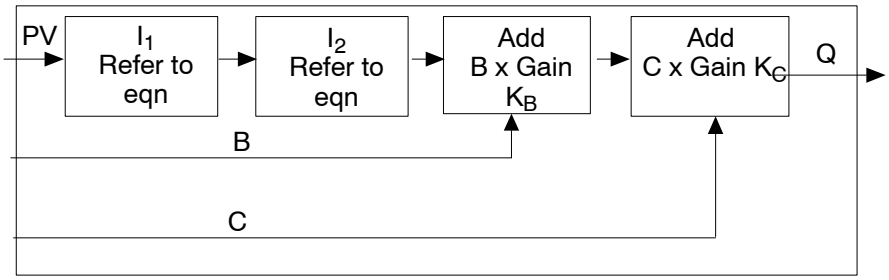


- The Dead Time (DT) function provides a configurable delay period.
- **Block Output**=(A x K_A) + (C x K_C)
- **Dead Time** =sample time x B x scaling factor

NOTE: A scaling factor of 60 changes the B input to seconds.

- Continuous Faceplate screen notes:
  - The K_A field defaults to 0. Change to desired value.
  - If the Dead Time block should be configured to delay only and not to change value, then configure **$K_A \rightarrow 1$** and **$K_C \rightarrow 0$**.
  - The block output high and low limits should be configured to allow all desirable output values.
- Two special logic step functions exist for the Dead Time ControlBlock. The **vfill** and **fill** functions are used in the ACTIONS part of a logic step. The vfill function fills all of the dead time slots with a value and is configured **vfill xx**, where xx is the value. The fill function fills all of the dead time slots with the hold forward flag value and is configured **fill**.
- [PAGE AHEAD] to call up Dead Time **input A plots**.
- For more information about the Dead Time function ControlBlock, see CB: 2.

Lead/Lag (LL) Function ControlBlock



- The Lead/Lag (LL) function provides one lead term and two lag terms for modeling process dynamics.
- **Lead/Lag equations:**
  $I_1=I_1' +(T_S/(T_S+2T_2))\ (K_1(M+M')-2I_1')-\ (1/(T_S+2T_2))(2T_1K_1(M-M'))$
  $I_2=I_2'+(T_S/(T_S+2T_3))\ (I_1+I_1'-2I_2')$
  $Q= (I_2) + (B \text{ x Gain } K_B) + (C \text{ x Gain } K_C)$
  where: $T_S=$ sample time
  - $T_1=$ lead time
  - $T_2=$ lag time 1
  - $T_3=$ lag time 2
  - $M =$ current P
  - $M'=$ PV from previous evaluation
  - $K_1=$ gain on input A
  - $I_1 =$ current output of first stage lead lag
  - $I_1' =$ previous output of first stage lead lag
  - $I_2 =$ current output of second stage lag
  - $I_2' =$ previous output of second stage lag
- **Continuous Faceplate screen notes:**
  - If the lead term is configured, one or both lag terms must also be configured.
  - The block output high and low limits should be configured to allow all desirable output values.
- For more information about the Lead/Lag function ControlBlock, see CB: 2.

Manual (MAN) Function ControlBlock

- The Manual (MAN) function ControlBlock has the block output **Q** determined by the operator.
- The Manual function ControlBlock is intended for two primary purposes:
  - As a manual loading station for operator control of valves and other devices.
  - As a place for the configuror to access logic steps when unconcerned with the major function of the ControlBlock.
- The operator is not required to make any entries for the block to be functional.
- The operator can control the output only if the block is in Local or Manual mode.
- For more information about the Manual function ControlBlock, see CB: 2.

User-Defined MATH Function ControlBlock

- The MATH function ControlBlock allows you to define a function containing equations that determine the value of the block output or any continuous input.
- Multiple equations can be written on an equation line if they are separated by semicolons.
- Equations can have up to **40** characters.
- Equations that compute the **continuous output Q** can be configured in two ways:

  Q=expression

  In this method, the letter Q is used. This type of equation is performed when the block is in Manual or Auto mode. For example, Q = A + B. This method overrides tracking and operator entry.

  expression

  In this method, the letter Q is not used. This type of equation is performed only when the MATH block is in Auto mode. For example, A + B.

User-Defined MATH Function ControlBlock (continued)

NOTE: The last equation in the list on the continuous faceplate determines the block output; however, the first type of equation (Q=expression) overrides any equations of the second type (A+B).

- Equations that compute the value of a continuous input can be configured in one way:
  - **input register=expression**

    In this method, the value of the expression is placed in the specified continuous input register. For example: A=B+C.
- Continuous Faceplate screen notes:
  - **Back Calc** is an optional user-defined tracking calculation to generate the tracking value that is sent back on the link specified by the Track Input entry.
  - **Max Gain** determines the rate of response of the actual tracking value sent along the **Track Input** entry to the tracking value; this value is either generated by **Back Calc** or received from another block through a forward link.

    The larger the Max Gain entry, the slower the response of the tracking value, as shown in the following equation:

$$(\Delta \text{ tracking value}) = \frac{(\text{calc value} - \text{actual value})}{(\text{Max Gain})}$$

- For more information about the MATH function ControlBlock, see CB: 2.

PID Function ControlBlock

- The PID function ControlBlock performs some combination of proportional, integral, and derivative control. Any combination of PID control can be configured (P, PI, PID, I, IB, PD, ID,D).
- **Block Output**=P+I+D+K_f(FF)

  where P = (cont gain or prop band) (SP—PV)

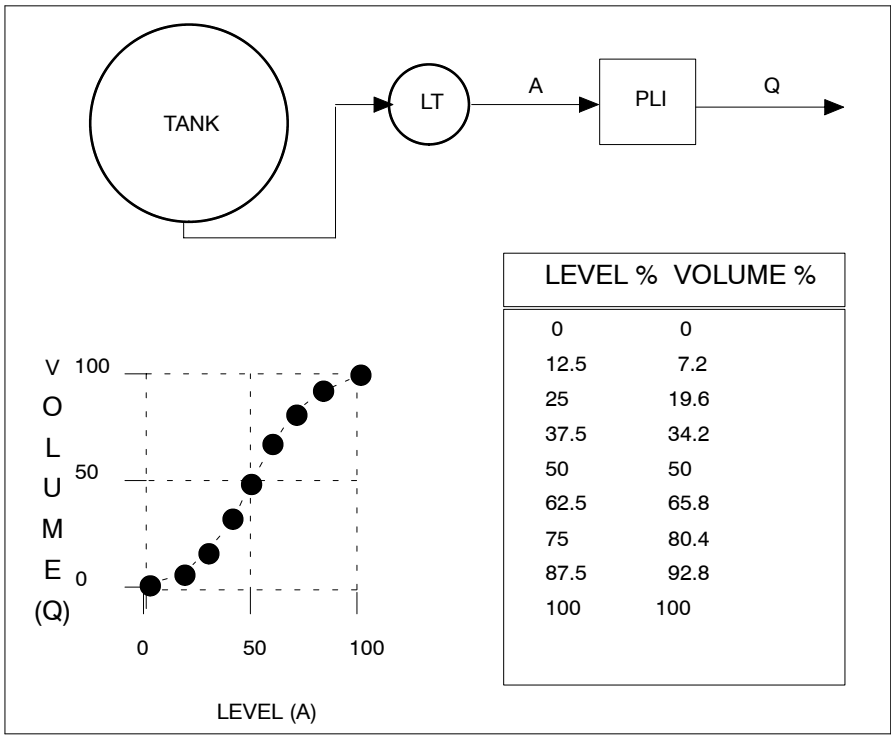
  I = I+(cont gain or prop band)(sample time/integral time)

  (SP—PV)

  D = (cont gain or prop band) (derivative time/sample time) (ΔPV)

  K_f = feedforward gain
- **Continuous Faceplate screen notes:**
  - **LS-PV Track**—if yes, LS tracks PV when the block mode is Manual or Local.
  - **Track Input** selects input through which backtracking signals are sent.
  - **Ratio** option—operator changes ratio, configuror changes bias.
  - **Bias** option—operator changes bias, configuror changes ratio.
  - **Gap** option modifies PID algorithm so that Q does not change until the required change exceeds the **Gap** value.
  - **Deadband** option modifies the PID algorithm so that Q does not change until |SP-PV| ≥ **Err DdBand**.
  - **Error Squared** option produces a nonlinear output and modifies the PID algorithm so that the SP—PV term is replaced by |SP—PV| x (SP—PV).
- For more information about the PID function ControlBlock, see CB: 2.

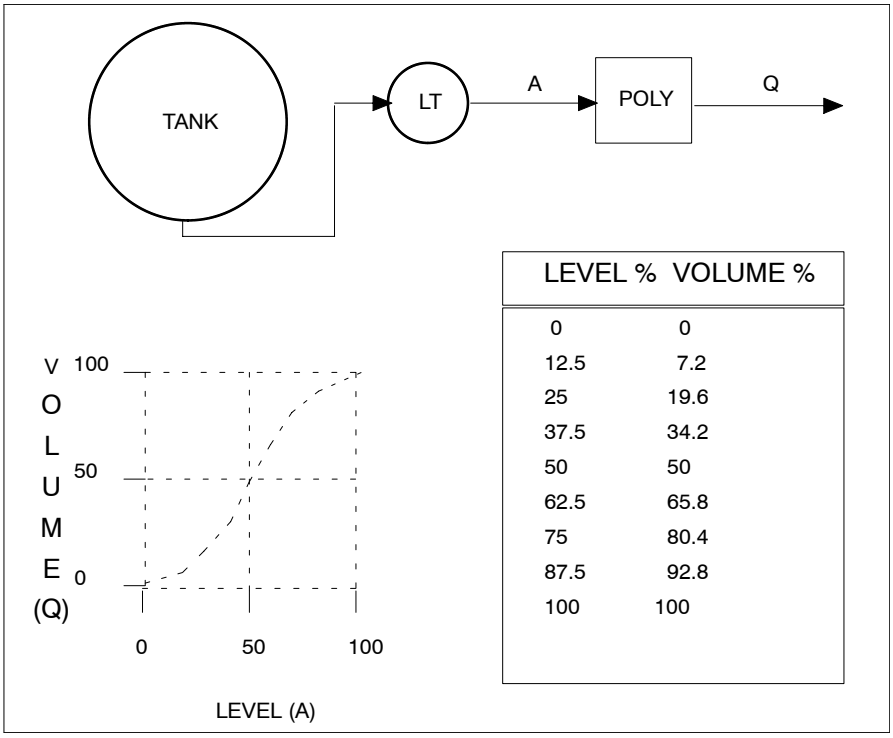
Piecewise Linear Interpolator (PLI) Function ControlBlock



- The Piecewise Linear Interpolator (PLI) function approximates a curve with a series of straight line segments.
- Continuous Faceplate screen notes:
  - 96 pairs of input/output values can be configured.
  - When the last PV entry on a page has been configured, **[PAGE AHEAD]** to the next 12 data point entries.
  - Nonzero input/output pairs must be configured in ascending order of input values.

    $$(\Delta \text{ Track Input}) = \frac{(\Delta \text{ received tracking signal})}{(\text{Max Gain})}$$
- For more information about the Piecewise Linear Interpolator function ControlBlock, see CB: 2.

Polynomial (POLY) Function ControlBlock



- The Polynomial (POLY) function approximates a curve with a polynomial equation.
- Polynomial equation:
  $Q=K_0C+K_1(A+B)+K_2(A+B)^2+K_3(A+B)^3+K_4(A+B)^4+K_5(A+B)^5+K_6(A+B)^6+K_7(A+B)^7$
- Continuous Faceplate screen notes:

  $$(\Delta \text{ Track Input}) = \frac{(\Delta \text{ received tracking signal})}{(\text{Max Gain})}$$

- For more information about the Polynomial function ControlBlock, see CB: 2.

Ratio/Bias (RB) Function ControlBlock

- The Ratio/Bias (RB) function ControlBlock provides adjustable ratio and bias capabilities.
- Q=[(PV — Pre-Bias) x Ratio x Ratio Gain] + Bias]
- **Ratio**
  - The **B** input is the ratio value.
  - To configure a **ratio** value of **0** to **n**:

    Configure the ratio term (B) **Eng Zero**⇒0.00, **Eng Max**⇒n, and **Ratio Gain**⇒n, where **n** is the maximum ratio value.
- **Bias**
  - The **C** input is the bias value and should have the same scaling as the output **Q**.
  - The **Bias Bar Scaling** fields allow the bias term bar on the continuous faceplate to be scaled differently than the bias value. The **Bias Bar Scaling: Zero** value is represented as 0 on the faceplate bias bar. The **Bias Bar Scaling: Full Scale** is represented as 100 on the faceplate bias bar.
  - The differences between Bias and Pre-Bias are listed in the table below:

Bias Differences

Bias Term	Pre-Bias Term
Added to output after ratio applied	Subtracted from PV before ratio applied
Operator entry	Configuror entry
Can be put on faceplate	Cannot be put on faceplate

- For more information about the Ratio/Bias function ControlBlock, see CB: 2.

Signal Selector (SS) Function ControlBlock

- The Signal Selector (SS) function ControlBlock is a general purpose selector that can function in the input or the output side of a loop. The selector serves as a one-of-N switch.
- The selected input value is not altered by the Signal Selector function. However, the output is affected by the output high and low limits.
- The selected input value is placed in the Signal Selector block output.
- If the Signal Selector function detects a Hold Forward signal from an input and the **Hold** field value on the Continuous Links screen for that input is **yes**, then the input is not used by the selector.

  The Signal Selector function does not pass on a Hold Forward signal until all inputs send Hold Forward signals.

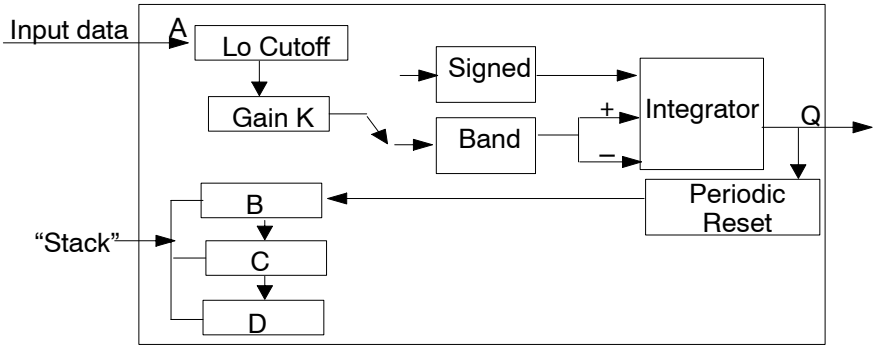
Signal Selector (SS) Function ControlBlock (continued)

- **Continuous Faceplate screen notes:**
  - **No. of Inputs** specifies the number of inputs to be used by the selector, starting with input A. For example, a value of 3 uses inputs A, B, and C.
  - **Select Number** specifies the nth-highest value to be selected. A value of 1 selects the highest input, a value of 2 selects the second-highest input, and so forth. A value of 0 selects the lowest input.
  - When the function is selecting ControlBlock outputs, the unselected ControlBlocks are open-looped. The **Trackband** and **Track Input** fields are used to prevent windup of any or all unselected ControlBlocks.

    All inputs specified in the **Track Input** field track the selected ControlBlock. The **Trackband** field specifies, in percent, how close the unselected ControlBlocks track the selected ControlBlock.

    For example, if **Track Input⇒ABC** and **Trackband⇒5.00** are specified and Input B is selected, then inputs A and C will be no less than 5.00% lower than B.
- For more information about the Signal Selector function ControlBlock, see CB: 2.

Stack Totalizer (TOT) Function ControlBlock

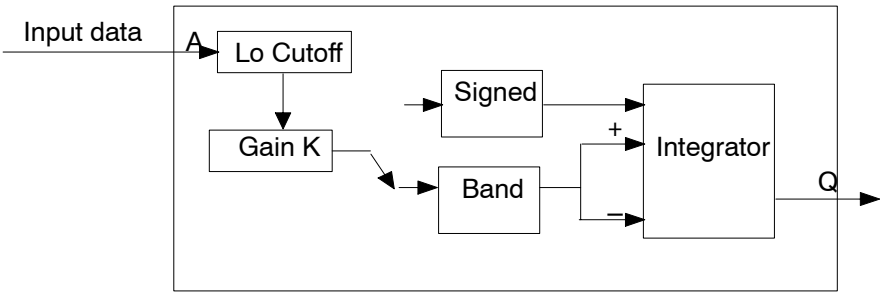


- The Stack Totalizer (TOT) function is an integrator with an output that is stored and then reset to zero when the output reaches a high limit or receives a reset command. Each time that the output is reset, its value (before reset) is placed at the top of a “stack,” and each value in the stack is shifted one place; that is, C→D, B→C, and Q→B.
- Totalizing equation: new total = previous total + $\frac{\text{(input A *gain)}}{\text{integration time}}$
- A positive input increases the total;
  a negative input decreases the total.
- **A** input contains the current PV value.
  Q output contains the current accumulated total.
  B input (T1) contains previous Q value when high limit or reset occurred.
  C input (T2) contains previous B value when high limit or reset occurred.
  D input (T3) contains previous C value when high limit or reset occurred.

Stack Totalizer (TOT) Function ControlBlock (continued)

- **Continuous Faceplate screen notes**:
  - Integ Time is the time that it takes for the output to go from 0% to 100% when the input is at 100%.
  - **Cutoff Type** specifies how to handle positive and negative values: **Band** totalizes both positive and negative values; **Signed** totalizes only positive values.
- A special logic step function exists for the Stack Totalizer block. The **treset** function is used to reset the stack totalizer.
- For information about clearing logic steps o and p, see “Setpoint Totalizer Function ControlBlock” (next page). For more information about the Stack Totalizer function ControlBlock, see CB: 2.

Setpoint Totalizer (TOTSP) Function ControlBlock



- The Setpoint Totalizer (TOTSP) function is an integrator with an output that resets to zero when the output reaches a high limit.
- Totalizing equation:

$$\text{new total} = \text{previous total} + \frac{(\text{input A} * \text{gain})}{\text{integration time}}$$

- A positive input value increases the total; a negative input value decreases the total.
- Logic steps o and p can be set and must be cleared by user logic.
  - **Logic step o** is set when overflow occurs—that is, when the value of the total exceeds the high output limit. When logic step o is set, the high limit value is subtracted from the total value.
  - **Logic step p** is set when underflow occurs—that is, when the value of the total goes below the low limit.

NOTE: This information about logic steps o and p also applies to the Stack Totalizer Function ControlBlock described on the previous page.

- Continuous Faceplate screen notes:
  - Integ Time is the time that it takes for the output to go from 0% to 100% when the input is at 100%.
  - **Cutoff Type** specifies how to handle positive and negative values: **Band** totalizes both positive and negative values; **Signed** totalizes only positive values.
- For more information about the Setpoint Totalizer function ControlBlock, see CB: 2.

Velocity Limiter (VLIM) Function ControlBlock

- The Velocity Limiter (VLIM) function ControlBlock provides an output that is a rate-limited function of the input.
- Rate limits are expressed in engineering units of the input per second. Rate limits are accurate regardless of sample time.
- If the difference between the output and the input is less than the rate limit, then the output and input will be the same value.
- **Continuous Faceplate screen notes:**
  - **Rise Limit** is the normal rate limit in the increasing direction.
  - When (PV–Q) > **Dev Trigger (Rise)**, then the **Rise Hi Lim** rate limit is used.
  - When (Q–PV) > **Dev Trigger (Fall)**, then the **Fall Hi Lim** rate limit is used.
- For more information about the Velocity Limiter function ControlBlock, see CB: 2.

Discrete (DISC) Function ControlBlock

- The Discrete (DISC) function ControlBlock has a block output **Q** that is determined by the operator or by an equation that is written by the configuror.
- The Discrete function ControlBlock is intended for two primary purposes:
  - To use the discrete faceplate for display purposes.
  - As a place where the configuror can get access to logic steps and is not concerned with the major function of the ControlBlock.
- The operator is not required to make any entries for the block to be functional.
- The operator can control the output only if the block is in Local or Manual mode.
- For more information about the Discrete function ControlBlock, see CB: 2.

Motor Controller Discrete Function ControlBlocks

- Motor Controller ControlBlocks include DMC, DASMC, DDSMC, and DDDMC function ControlBlocks. The table below describes the common motor-controller fields on the Continuous Faceplate screen.
- For more information about the discrete motor controllers, see CB: 2.

Motor Controller CB Continuous Faceplate Screen Fields

Field	Function
Interlock	If **yes**, then **@k** must be true for the motor to start or to remain running. (Interlock⇒yes required for DASMC)
Ignore Interlock	If **yes**, motor can start or run even though **@k** is false. Permits temporary bypass of @k. (Field present only if Interlock⇒yes)
Retry	(DMC, DASMC only) If **yes**, the number of unsuccessful starting attempts is limited to a preset value (input **I)**, after which a time period (input **J)** must elapse before further attempts are allowed.
Intermediate Stop	(DDSMC, DDDMC only) If **yes**, a change in speed or direction is not permitted unless Confirm Off (**@h**) is true. Any change attempted while the motor is running switches the block to stop.
Confirm Off	If **yes**, **@h** is expected to be true within time period **G** following a Stop command (**b**); otherwise an alarm (**h**) occurs. If **no**, the false state of **@g** (Confirm On) constitutes a Confirm Off.
Ignore Confirm Off	If **yes**, no alarm occurs if Confirm Off is not received when expected. Permits temporary bypass of **@h**. (Field present only if Confirm Off⇒yes)
Security Lockup	If **yes**, recovery from a Tripped state requires **@l** (reset) to be momentarily true after the fault has cleared. If **no**, a Stop command releases the block from the Tripped state if the fault has cleared.
MCC Alarm	If **yes**, step **l** can be used to generate an alarm indicating motor control center power failure. (Requires Confirm Off⇒yes)
Ignore Confirm On	If **yes**, no alarm occurs even though Confirm On, Forward, Reverse, High, or Low is not received when expected. Permits temporary bypass of **@g** and/or **@i**.
Trip Delay	If **yes**, block transfer to the Tripped state is delayed by a time **F** to allow the motor to ride through momentary power failures. If **no**, the block transfers immediately to the Tripped state whenever a run confirm signal is lost without a Stop command.

DMC Function ControlBlock

- The DMC (Motor Controller) function is a preconfigured ControlBlock that is used with a single-speed electric motor.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **K**, **L**, **M**, **N**, and **O**; the following are predefined:
  - **F** Start timer counts time between Start command (step a) and Confirm On (**@g**).
  - **G** Stop timer counts time between Stop command (step b) and Confirm Off (**@h**).
  - **H** Preconfigured as *Timer, but not used by DMC function; therefore, input H should not be used.
  - **I** Retry counter used with the Retry option.
  - **J** Lockout timer used with the Retry option.
- DMC discrete input functions when the inputs are true:

@**a**	manual start	**@i**	user-configurable
@b	manual stop	**@j**	stop (regardless of block mode)
@c	user-configurable	**@k**	OK to run
@d	automatic start	**@l**	releases security lockup
@e	automatic stop	**@m**	user-configurable
@f	user-configurable	**@n**	user-configurable
@g	confirm on	**@o**	user-configurable
@h	confirm off		

DMC Function ControlBlock (continued)

- DMC discrete output conditions when discrete outputs are true:
  - **a** start
  - **b** stop
  - **c** user-configurable
  - **d** ignore option(s) selected
  - **e** one of **f**, **g**, **h**, **j**, or **k** outputs is true, indicating failure condition
  - **f** more than one confirm or command input is true
  - **g** failed to start within time limit
  - **h** failed to stop within time limit
  - **i** user-configurable
  - **j** confirm changed without command, indicating that motor has tripped
  - **k** locked off, waiting for **@l** (reset)=true
  - **l** If “MCC Alarm” field is configured to “yes,” output is true for motor control center power loss and both confirms are off. For all other configuration options, the step is user–configurable.
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about DMC function ControlBlock, see CB: 2.

DASMC Function ControlBlock

- The DASMC (Auto Sequence Motor Controller) function is a preconfigured ControlBlock for use with a single-speed electric motor that is one of several motors that must be automatically started in a predetermined sequence.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **K**, **L**, **M**, **N**, and **O**; the following are predefined:
  - **F** Start timer counts time between Start (step a) and Confirm On(@g).
  - **G** Stop timer counts time between Stop (step b) and Confirm Off(@h).
  - **H** Delay On timer used with logic that guarantees a delay between Start request (**@d** & **@k** true) and output of Start command (step a).
  - **I** Retry counter used with the Retry option.
  - **J** Lockout timer used with the Retry option.
- DASMC discrete input functions when the inputs are true:

@a	manual start	**@i**	user-configurable
@b	manual stop	**@j**	stop (regardless of block mode)
@c	user–configurable	**@k**	OK to start after delay **H**
@d	OK to start after delay **H** if **@k** true	**@l**	if **@d** true releases security lockup
@e	automatic stop	**@m**	user-configurable
@f	user–configurable	**@n**	user-configurable
@g	confirm on	**@o**	user-configurable
@h	confirm off		

DASMC Function ControlBlock (continued)

- DASMC discrete output conditions when the outputs are true:
  - **a** start
  - **b** stop
  - **c** user-configurable
  - **d** ignore option(s) selected
  - **e** one of **f**, **g**, **h**, **j**, or **k** outputs is true, indicating failure condition
  - **f** more than one confirm or command input is true
  - **g** failed to start within time limit
  - **h** failed to stop within time limit
  - **i** user-configurable
  - **j** confirm changed without command, indicating motor has tripped
  - **k** locked off, waiting for **@l** (reset)=true
  - **l** If “MCC Alarm” field is configured to “yes,” output is true for motor control center power loss and both confirms are off. For all other configuration options, the step is user–configurable.
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about DASMC function ControlBlock, see CB:2.

DDSMC Function ControlBlock

- The DDSMC (Dual Speed Motor Controller) function is a preconfigured ControlBlock used with a two-speed electric motor.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **K**, **L**, **M**, **N**, and **O**; the following are predefined:
  - **F** Start timer counts time between High command (step a) and Confirm High (@g), or Low command (step c) and Confirm Low(@i).
  - **G** Stop timer counts time between Stop command (step b) and Confirm Off (**@h**).
  - **H** Start timer for low speed counts between step c and Confirm (**@i**)
  - **I** Retry counter used with the Retry option.
  - **J** Lockout timer used with the Retry option.
- DDSMC discrete input functions when the inputs are true:

@a	manual start high	**@i**	confirm low speed
@b	manual stop	**@j**	stop (regardless of block mode)
@c	manual start low	**@k**	OK to run
@d	automatic start high	**@l**	releases security lockup
@e	automatic stop	**@m**	user-configurable
@f	automatic start low	**@n**	user-configurable
@g	confirm high speed	**@o**	user-configurable
@h	confirm off		

DDSMC Function ControlBlock (continued)

- DDSMC discrete output conditions when the outputs are true:
  - **a** start high speed
  - **b** stop
  - **c** start low speed
  - **d** ignore option(s) selected
  - **e** one of **f** to **l** outputs is true, indicating failure condition
  - **f** more than one confirm or command input is true
  - **g** failed to start at high speed within time limit
  - **h** failed to stop within time limit
  - **i** failed to start at low speed within time limit
  - **j** confirm changed without command, indicating motor has tripped
  - **k** locked off, waiting for **@l** (reset)=true
  - **l** If “MCC Alarm” field is configured to “yes,” output is true for motor control center power loss and both confirms are off. For all other configuration options, the step is user-configurable.
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about the DDSMC function ControlBlock, see CB: 2.

DDDMC Function ControlBlock

- The DDDMC (Dual Direction Motor Controller) function is a preconfigured ControlBlock used with a reversible electric motor.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **K**, **L**, **M**, **N**, and **O**; the following are predefined:
  - **F** Forward timer counts time between Forward command (step a) and Confirm Forward (**@g**).
  - **G** Stop timer counts time between Stop command (step b) and Confirm Off (**@h**).
  - **H** Reverse timer counts time between Reverse command (step c) and Confirm Reverse (**@i**).
  - **I** Retry counter used with the Retry option.
  - **J** Lockout timer used with the Retry option.
- DDDMC discrete input functions when the inputs are true:

@a	manual start forward	**@i**	confirm on reverse
@b	manual stop	**@j**	stop (regardless of mode)
@c	manual start reverse	**@k**	OK to run
@d	automatic start forward	**@l**	releases security lockup
@e	automatic stop	**@m**	user-configurable
@f	automatic start reverse	**@n**	user-configurable
@g	confirm on forward	**@o**	user-configurable
@h	confirm off		

DDDMC Function ControlBlock (continued)

- DDDMC discrete output conditions when the outputs are true:
  - **a** start forward
  - **b** stop
  - **c** start reverse
  - **d** ignore option(s) selected
  - **e** one of **f** to **l** outputs is true, indicating failure condition
  - **f** more than one confirm or command input is true
  - **g** failed to start in forward direction within time limit
  - **h** failed to stop within time limit
  - **i** failed to start in reverse direction within time limit
  - **j** confirm changed without command, indicating motor has tripped
  - **k** locked off, waiting for **@l** (reset)=true
  - **l** If “MCC Alarm” field is configured to “yes,” output is true for motor control center power loss and both confirms are off. For all other configuration options, the step is user-configurable.
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about the DDDMC function ControlBlock, see CB: 2.

Valve Controller Discrete Function ControlBlocks

- Valve Controller ControlBlocks include DVC, DASVC, and DMVC functions. The table below describes the common valve-controller fields on the Continuous Faceplate screen.
- For more information about the discrete valve controllers, see CB: 2.

Valve Controller Discrete Function ControlBlocks (continued)

Valve Controller CB Continuous Faceplate Screen Fields

Field	Function
Interlock	If **yes**, then **@k** must be true for the valve to open or to remain open. (Interlock⇒yes required for DASVC)
Ignore Interlock	If **yes**, the valve can open or remain open even though **@k** is false. Permits temporary bypass of **@k**. (Field present only if Interlock⇒yes)
Confirm Close	If **yes**, **@h** is expected to be true within time period **G** following a Close command (**b**); otherwise an alarm (**h**) occurs. If **no**, the false state of **@g** (Confirm Open) constitutes a Confirm Close.
Ignore Confirm Close	If **yes**, no alarm occurs if Confirm Close is not received when expected. Permits temporary bypass of **@h**. (Field present only if Confirm Close⇒yes)
Security Lockup	If **yes**, recovery from a Failed state requires **@I** (reset) to be momentarily true after the fault has cleared. If **no**, a Close command releases the block from the Failed state if the fault has cleared.
Confirm Open	If **yes**, **@g** is expected to be true within time period **F** following an Open command (**a**), otherwise an alarm (**h**) occurs. If **no**, the false state of **@h** (Confirm Close) constitutes a Confirm Open.
Ignore Confirm Open	If **yes**, no alarm occurs even though confirm open is not received when expected. Permits temporary bypass of **@g**. (Field present only if Confirm Open⇒yes)

DVC Function ControlBlock

- The DVC (Valve Controller) function is a preconfigured ControlBlock that is used with any valve that requires a contact closure to maintain the open position, closed position, or both.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **I**, **J**, **K**, **L**, **M**, **N**, and **O**; the following are predefined:
  - **F** Open timer counts time between Open command (step a) and Confirm Open (**@g**).
  - **G** Close timer counts time between Close command (step b) and Confirm Close (**@h**).
  - **H** Preconfigured as ***Timer**, but not used by DVC function; therefore, input H should not be used.
- DVC discrete input functions when the inputs are true:

@a	manual open	**@i**	user–configurable
@b	manual close	**@j**	close (regardless of block mode)
@c	user configurable	**@k**	OK to open and remain open
@d	automatic open	**@l**	releases security lockup
@e	automatic close	**@m**	user-configurable
@f	user configurable	**@n**	user-configurable
@g	confirm open from valve	**@o**	user-configurable
@h	confirm closed from valve		

DVC Function ControlBlock (continued)

- DVC discrete output conditions when the outputs are true:
  - **a** open
  - **b** close
  - **c** user configurable
  - **d** ignore option(s) selected
  - **e** one of outputs **f**, **g**, **h**, or **k** is true, indicating failure condition
  - **f** more than one confirm or command is true, suggesting switch failure
  - **g** failed to open within time limit
  - **h** failed to close within time limit
  - **i** user configurable
  - **j** one of the confirm timers is running, indicating valve is traveling
  - **k** locked closed, waiting for **@l** (reset)=true
  - **l** user-configurable
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about the DVC function ControlBlock, see CB: 2.

DASVC Function ControlBlock

- The DASVC (Auto Sequence Valve Controller) function is a preconfigured ControlBlock for use with a valve that is one of several valves that must be automatically opened in a predetermined sequence.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **I**, **J**, **K**, **L**, **M**,**N**, and **O**; the following are predefined:

  F Open timer counts time between Open command (step a) and Confirm Open (**@g**).

  G Close timer counts time between output of a Close command (step b) and Confirm Close (**@h**).

  H Delay On timer used with logic that guarantees a delay between Open request (**@d**) and Open command (step a).

- DASVC discrete input functions when the inputs are true:

@a	manual open	**@i**	user-configurable
@b	manual close	**@j**	close(regardless of block mode)
@c	user configurable	**@k**	OK to open after delay and remain open if **@d** true
@d	OK to open after delay if **@k** true	**@l**	releases security lockup
@e	automatic close	**@m**	user-configurable
@f	user configurable	**@n**	user-configurable
@g	confirm open from valve	**@o**	user-configurable
@h	confirm closed from valve		

DASVC Function ControlBlock (continued)

- DASVC discrete output conditions when the outputs are true:
  - **a** open
  - **b** close
  - **c** user-configurable
  - **d** ignore option(s) selected
  - **e** one of outputs **f**, **g**, **h**, or **k** is true, indicating failure condition
  - **f** more than one confirm or command is true, suggesting switch failure
  - **g** failed to open within time limit
  - **h** failed to close within time limit
  - **i** user-configurable
  - **j** one of the confirm timers is running, indicating that valve is traveling
  - **k** locked closed, waiting for **@l** (reset)=true
  - **l** user-configurable
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about the DASVC function ControlBlock, see CB: 2.

DMVC Function ControlBlock

- The DMVC (Motorized Valve Controller) function is a pre-configured ControlBlock that is used with a motorized valve controller that requires a contact closure to change but not to maintain position.
- The configuror can use continuous inputs **A**, **B**, **C**, **D**, **E**, **I**, **J**, **K**, **L**, **M**, **N**, and **O**; the following are predefined:

  F Open timer counts time between Open command (step a) and Confirm Open (**@g**).

  G Close timer counts time between Close command (step b) and Confirm Open (**@h**).

  H Delay On timer used with logic that guarantees a delay between Open request (**@d**) and Open command (step a).
- DMVC discrete input functions when the inputs are true:

@a	manual open	**@i**	user-configurable
@b	manual close	**@j**	close (regardless of block mode)
@c	manual stop	**@k**	OK to open and remain open
@d	automatic open	**@l**	releases security lockup
@e	automatic close	**@m**	user-configurable
@f	automatic stop	**@n**	user-configurable
@g	confirm open from valve	**@o**	user-configurable
@h	confirm closed from valve		

DMVC Function ControlBlock (continued)

- DMVC discrete output conditions when the outputs are true:
  - **a** open
  - **b** close
  - **c** user-configurable
  - **d** ignore option(s) selected
  - **e** one of outputs **f**, **g**, **h**, or **k** is true, indicating failure condition
  - **f** more than one confirm or command is true, suggesting switch failure
  - **g** failed to open within time limit
  - **h** failed to close within time limit
  - **i** user-configurable
  - **j** one of the confirm timers is running, indicating the valve is traveling
  - **k** locked closed, waiting for **@I** (reset)=true
  - **l** user-configurable
  - **m** user-configurable
  - **n** user-configurable
  - **o** user-configurable
  - **p** user-configurable
- For more information about the DMVC function ControlBlock, see CB: 2.

RBL ControlBlock

- The RBL ControlBlock allows you to communicate with weigh scales, analyzers, and other devices by reading and writing through the serial communications lines.
- Communication with these external devices is controlled by a Rosemount Basic Language program that includes the communication instructions to read and write using the serial ports on the Controller Processor.
- RBL ControlBlocks use standard ControlBlock Links and Faceplate screens, but not the Continuous Diagram and Discrete Diagram screens.
- **Continuous Faceplate screen notes:**
  - Program Status displays one of the following conditions:
    RBLC Download: Downloading from disk.
    Idle: Not executing.
    Retry Download: Download failed, trying again.
    RBLC Running: (Program line)
  - **GOTO RBLC Monitor**: Press [ENTER] or [SELECT] on this field to call up a screen that allows you to monitor the execution of the RBL program.
  - **GOTO RBLC Script**: Press [ENTER] or [SELECT] on this field to call up the script for the program that is specified for this block.
  - Block Mode is not used; however, the block must be in “manual“ mode to be deleted.
- For more information about the RBL ControlBlock, see RB: 2.

Section 3:
Logic Steps

Logic Steps

- Logic steps are configured on the Discrete Diagram screens of ControlBlocks.
- The fields in the **Conditions** portion of the screen are used to determine the state of the logic step.
  - The first field contains the letter of the logic step (for example: **a**). When the statement in this field is true, the logic step is **on**. When the statement is false, the logic step is **off**.
  - When the statement in **Set** field is true, the logic step is **on**.
  - When the statement in **Clear** field is true, the logic step is **off**.
  - If both **Set** and **Clear** are true at the same time, the **Clear** condition prevails.
- The fields in the **Actions** portion of the screen are used to perform various functions, such as acting on the continuous portion of the block.
  - The **Rise** action is executed when the logic step goes from off to **on**.
  - The **On** action is executed each evaluation cycle that the logic step is **on**.
  - The **Fall** action is executed each time that the logic step goes from on to **off**.
  - The **Off** action is executed each evaluation cycle that the logic step is **off**.
- The **Report** field is used to configure an event or alarm to be associated with the logic step. When the **Report** field is configured, other event and alarm fields appear.
- For more information about logic steps and discrete inputs, see CB: 6.

Order of Precedence

The following two tables list mathematical and logical operators and functions. The list below is the order of precedence in which logic statement components are evaluated:

1. Parentheses
2. Single arithmetic components (abs, round, int)
3. Root and trigonometric functions (sqrt, sqrl, sin, cos, tan, and so forth.)
4. Binary arithmetic (**, *, /, %, max, min, +, –)
5. Logic (rise, fall, ~, &, |, ↑)
6. Relational (==, ~=, >=, >, <=, <)
7. if?then:else
8. Assignment (=)

Mathematical and Logical Operators

Mathematical and Logical Operators

Oper.	Description	Format	Explanation
( )	order evaluation—operations within parentheses are first	(x+y)*z	(2+10)*4 is 48
**	exponentiation	x**y	2**3 is 2^3 is 8
*	multiply	x*y	4*3 is 12
/	divide	x/y	12/4 is 3
%	divide modulo (remainder)	x%y	5%2 is 1
+	add	x+y	7+2 is 9
-	subtract	x-y	9-7 is 2
rise	true if variable (x) just became true.	risex	returns true for only one evaluation cycle after x changes, even if x remains true
fall	true if variable (x) just became false.	fallx	returns true for only one evaluation cycle after x changes, even if x remains false
~	unary logical negation	~x	x is negated
&	conditional and—true if and only if both expressions are true; else false	x&y	returns true (1) if x & y >0; else, returns false (0)
|	conditional inclusive or—true if and only if either one or both expressions are true; else false	x|y	returns true (1) if x>0 or y>0, or both; else false (0)
↑	conditional exclusive or—true if and only if either expression is true; else false	x↑y	returns true (1) if x>0 or y>0, but not both; else false (0)
==	relational equality—is one value equal to another	x==y	If x is equal to y, the expression is true
~	relational inequality	~=x	3~=6 is true
>=	relational greater than or equal to	x>=y	8>=6 is true
>	relational greater	x>y	12>4 is true
<=	relational less than or equal to	x<=y	2<=2 is true
<	relational less than	x<y	5<8 is true
x?y:z	if-else selection	v=x?y:z	If x is true, v equals y, if x is false, v equals z
=	assignment—sets one expression equal to value of another expression	x=y	x equals y

NOTE: x, y, and z may be single discrete variables (a through p, @a through @o), single continuous variables (A-O, Q), or expressions.

Mathematical Functions

Mathematical Functions

Oper.	Description	Format	Explanation
abs	absolute value function	abs x	abs 5 is 5, abs –5 is 5
acos	inverse cosine trigonometric function	acos x	acos .5 is 1.04720
asin	inverse sine trigonometric function	asin x	asin 1 is 1.57080
atan	inverse tangent trigonometric function	atan x	atan .5 is .463648
cos	cosine trigonometric function (radians)	cos x	cos 0 is 1.000
exp	exponent of e function	exp x	exp 1 is 2.72
exp2	exponent of 2 function	exp2 x	exp2 6 is 2^6 is 64
fract	fraction function—returns the fractional part of a number	fract x	fract 3.2 is .2
int	integer function—returns greatest integer less than or equal to a number	int x	int 4.73 is 4, int –4.2 is –5
ln	base e log function (natural log)	ln x	ln 15 is 2.71
log	base 10 log function	log x	log 52,400 is 4.72
log2	base 2 log function	log2 x	log2 600 is 9.23
max	maximum value function	x max y	10 max 1 is 10
min	minimum value function	x min y	10 min 1 is 1
round	rounding function—variable or expression is rounded to nearest integer	round x	round 2.2 is 2 , round 2.6 is 3
sign	signum (sign) function returns +1, 0, or –1	sign x	sign –4 is –1
sin	sine trigonometric function (radians)	sin x	sin 1 is .841471
sqrl	square root limited gain function—returns the square root of x if x>.01 of scale; otherwise returns 10*x	sqrl x	sqrl 16=4 sqrl .0001=.001
sqrt	square root function—returns the square root of a number	sqrt x	sqrt 16 is 4 sqrt .0001 is .01
tan	tangent trigonometric function (radians)	tan x	tan .90 is 1.26016

NOTE: x, y, and z may be single discrete variables (a through p, @a through @o), single continuous variables (A–O, Q), or expressions.

Logic Statement Functions

Logic Statement Functions

Function	Description
count	Counts the occurrences of a specified event and becomes true after a preset number of counts.
delay	Turns on a logic step after a condition has been true for an uninterrupted length of time, and then turns off the logic step after the condition has been false for the same uninterrupted length of time.
duty	Converts a controller output to a variable period pulse train.
period	Generates pulses at a preset interval as long as a specified condition remains true.
timer	Turns on a logic step after a condition has been true for a cumulative length of time.
wait	Turns on a logic step after a condition has been true for an uninterrupted length of time.
ramp	Increases or decreases a variable (Y) toward a target value (X) at a rate (Z).
time	Provides a true/false value that depends on the current clock time or date.
setuauto	Sets the logic step mode to Auto.
setuman	Sets the logic step mode to Manual.
setmode	Sets the operating mode of the ControlBlock being configured.
mode	Tests the mode of a ControlBlock.
sstand	Sets the Controller Processor card to Standby mode.
snorm	Sets the Controller Processor card to Normal mode.
ifstand	Tests if the Controller Processor is in Standby mode.
inhibit	Inhibits all alarms from a ControlBlock
inha	Inhibits all alarms and events on the Controller Processor.

* These functions are not described in detail and are listed here for reference only.

(continued on next page)

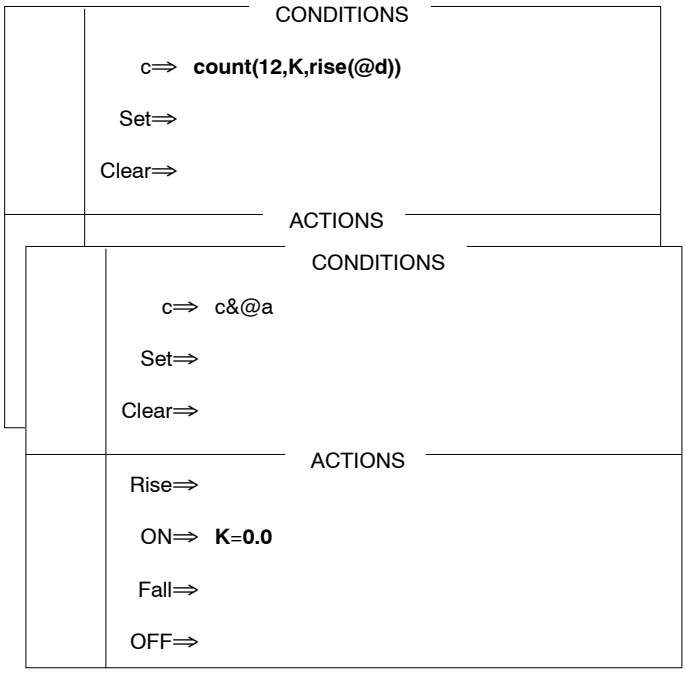
Logic Statement Functions (continued)

Function	Description
enba	Enables all alarms and events on the Controller Processor.
if?then:else	Specifies a condition, an action that occurs if the condition is true, and another action that occurs if the condition is false.
treset	Resets a stack totalizer to zero and shifts inputs B, C, and D.
fhf*	Forces hold forward action. Expressed as fhf.
track*	All blocks continue to operate if the output is not linked to anything. Expressed as track.
fill*	Invalidates any numbers currently in the Dead Time stack and sets the hold forward flag. The Dead Time stack can then be filled with numbers and can contain a combination of valid and invalid numbers. Expressed as fill. For information about using the fill function with a Dead Time function ControlBlock, see Control Block:6.
vfill x*	Fills the Dead Time stack with a user-specified value. The output goes to the user-specified value until new information is received. Expressed as vfill x. For information about using the vfill function with a Dead Time function ControlBlock, see ControlBlock:6.
fnow*	Forces an evaluation of the block function. When “fnow” is on, a new value is put in the dead time stack. When “fnow” is off, the stack and display freeze. Expressed as fnow.
norate*	Disables all operator entry rate limits. Expressed as norate.
notrack*	Forces a block to ignore received backtracking by clearing all four track action flags. Has no effect on Hold Forward. Expressed as notrack.
siterm x*	Sets the value of a PID integration term. The siterm function should be used carefully because the tracking system depends on being able to adjust to the integral term. Expressed as siterm x.
ssm*	ssm is the number of seconds since midnight. The ssm function allows you to schedule an event between midnight and 1 a.m.
sss*	ssm is the number of seconds since Sunday midnight. The ssm function allows you to schedule an event.
V*	V is the output tracking value in a tracking scheme. Using V in a logic statement allows you to test for the tracking value coming back to the primary block.

* These functions are not described in detail and are listed here for reference only.

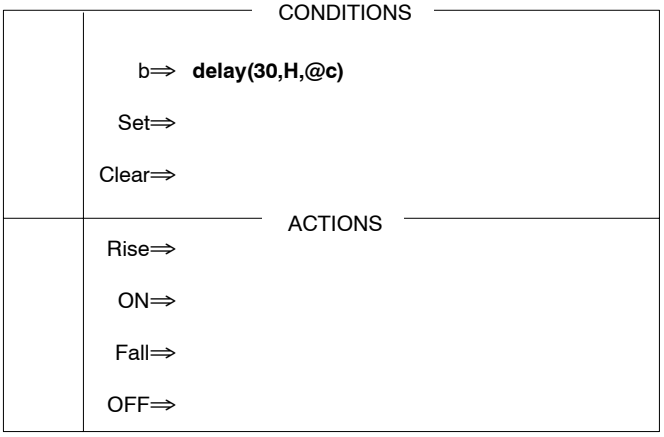
count Function

- The count function is an event counter in which Y is used to count the number of evaluation cycles that the condition Z is true. The logic step becomes true when the count Y reaches the value X. Y must be reset by a separate logic action statement.
- The count function is expressed as **count(X,Y,Z)**.
  - X is target count. When Y equals X, the logic step turns on.
  - Y is the counter register (*COUNTER).
  - Z is a condition.
- In the example below, the counter (input K) is incremented each time that the input @d turns on. When the counter reaches 12, logic step c is turned on. Logic step e is used to reset the counter.
- For more information about the count function, see CB: 6.



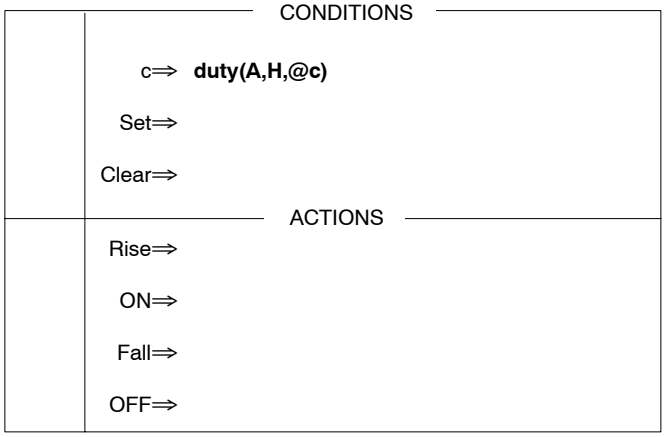
delay Function

- The delay function is a dual, sequential timer that turns on a logic step after a condition has been true for an uninterrupted length of time and then turns off the logic step after the condition has been false for the same uninterrupted length of time.
- The delay function is expressed as **delay(X,Y,Z)**.
  - X is the **uninterrupted** number of seconds that Z must be true to turn on the step, and also the uninterrupted number of seconds that Z must be true to turn off the step.
  - Y is the timer register (*TIMER).
  - Z is a condition.
- In the example below, the delay timer (input H) runs whenever @c is true. When the timer has been running for 30 uninterrupted seconds, logic step b is turned on. When the timer reaches 30 again, the logic step is turned off.
- For more information about the delay function, see CB: 6.



duty Function

- The duty function is a counter that can be used to convert a controller output to a variable period pulse train.
- The duty function is expressed as **duty(X,Y,Z)**.
  - X is a controller output. X must be from 0 to 1.
  - Y is the counter register (*COUNTER). For each evaluation cycle that Z is on, Y=Y+X. When Y≥1.0, Y=Y−1.0.
  - Z is a condition.
- In the example below, A is the output of a controller in the range of 0 to 1. Input H is the counter register. Each evaluation cycle that @c is true, the controller output value is added to the register H. When H≥1, logic step c turns on, and H=H−1.0.
- For more information about the duty function, see CB: 6.

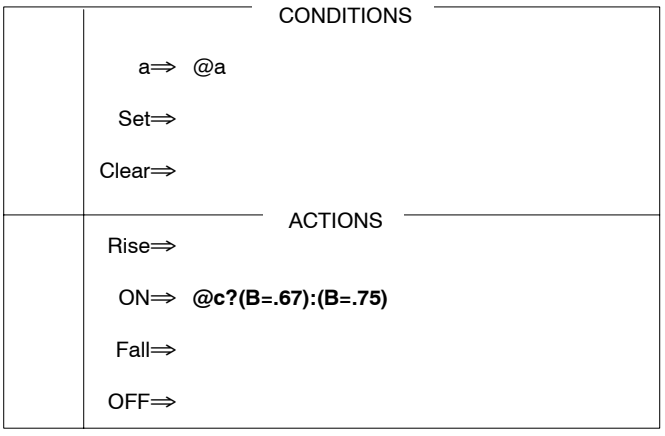


if?then:else Function

- The if?then:else function specifies a condition, an action that occurs if the condition is true, and another action that occurs if the condition is false.
- Example if?then:else function statements are listed below.

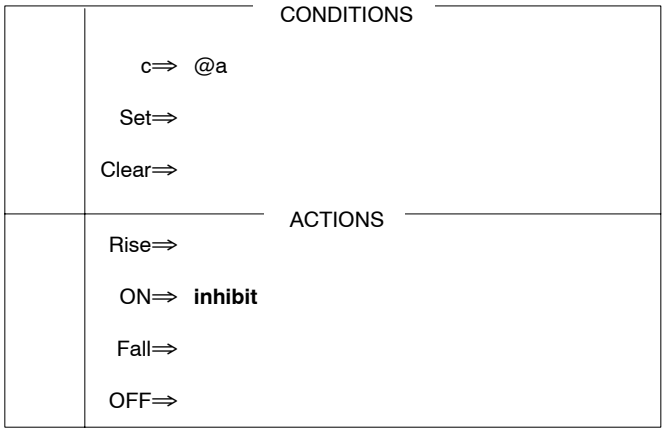
F=@a?E+.2:E–.2	If @a is true, then F=E+.2. If @a is false, then F=E–.2.
D=@a & @b?D+.1:D	If @a and @b are true, then D=D+.1. If @a or @b is false, then D remains unchanged.
A<.15?setmode 1:setmode 2	If A<15%, then block mode set to manual. If A >=15%, then block mode set to auto.

- The figure below also shows an example if?then:else function statement.
  - If @a and @c are true, then B is set to .67.
  - If @a is true and @c is false, then B is set to .75.
  - If @a is false, do nothing to B.
- For more information about the if?then:else function, see CB: 6.



inhibit Function

- The inhibit function inhibits all of the alarms from a ControlBlock from generating and being sent on the PeerWay.
- When the inhibit function is executed, the following tasks occur:
  - The block sends a clear message on the PeerWay for each uncleared alarm in the block.
  - All alarms from the block are inhibited from generating and are not sent on the PeerWay.
- The inhibit function is expressed as **inhibit** with no parameters.
- The alarms are inhibited only for those evaluation cycles in which the inhibit function is executed.
- You configure an inhibit expression in the Actions portion of a logic statement as shown in the example below.
- For more information about the inhibit function, see CB: 6.



mode Function

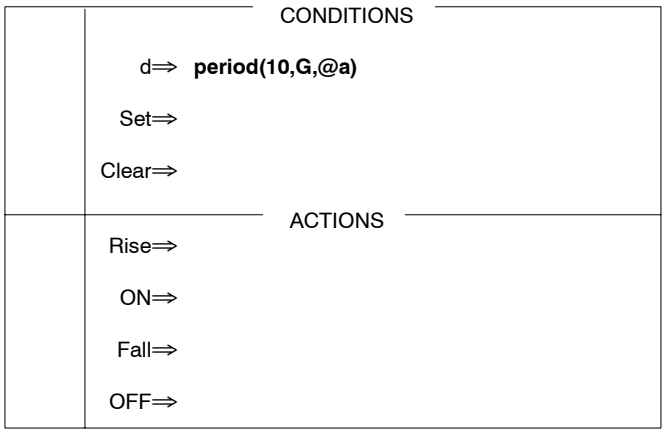
- The mode function is used to test the mode of a block.
- To test for the mode of the block currently being configured, use **mode==n**

  where n represents the block mode according to the following list:

  0=LOCAL
  1=MANUAL
  2=AUTO
  3=REMOTE
  4=DDC
  5=COMPUTER
- To test for the mode of another block, the ControlBlock must test for the mode block status bit. The other block must be configured as a continuous link into the current block. The block tag or address is followed by /MD. The test expression is then configured in a logic step. For more information, see the Block Status Bits heading in Section 4.
- For more information about the mode function, see CB: 6.

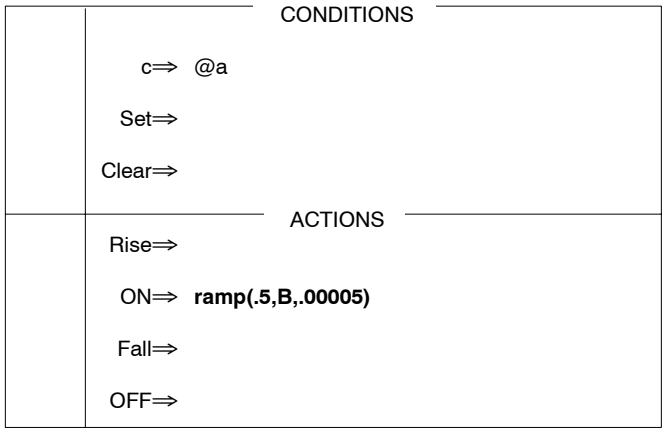
period Function

- The period function is a pulse generator that operates as long as a specified condition is true.
- The period function is expressed as **period(X,Y,Z)**.
  - X is the number of seconds **between pulses**. As long as the condition is true, the logic step is turned on for one evaluation cycle every X seconds.
  - Y is the timer register (*TIMER). Y is reset to zero whenever Y=X or when Z becomes false.
  - Z is a condition.
- In the example below, the period timer (input G) runs whenever @a is true. When the timer reaches 10 seconds, logic step d is turned on for one evaluation cycle.
- For more information about the period function, see CB: 6.



ramp Function

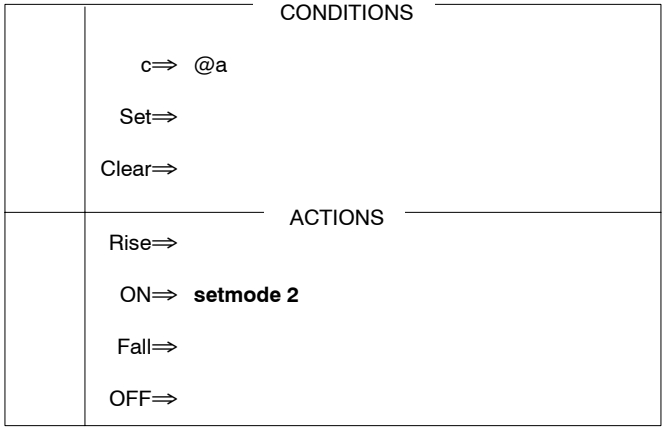
- The ramp function increases or decreases a variable (Y) toward a target value (X) at a rate (Z).
- The ramp function is expressed as **ramp(X,Y,Z)**
  - X is the target value. X must be from 0 to 1.
  - Y is the variable to be ramped (the output Q cannot be ramped).
  - Z is the ramp rate in units per second. Z can be positive (increasing) or negative (decreasing).
- In the example below, a ramp function is used to ramp the temperature of a vessel up to 500 °F at 3 °F/minute, or 0.05 °F/second. The range of the variable B is 0–1000.
  - The target value is 500. Since the range is 0–1000, the value of X is .5.
  - The rate is .05 °F/second. Since the range is 0–1000, the value of Z is .00005.
- For more information about the ramp function, see CB: 6.



setmode Function

- The setmode function is used to set the operating mode of the block being configured.
- To set the mode of the block, use the expression **setmode n**, where n represents the block mode according to the following list:

  0=LOCAL
  1=MANUAL
  2=AUTO
  3=REMOTE
  4=DDC
  5=COMPUTER
- In the example below, logic step c is used to set the block mode to AUTO whenever input @a is on.
- For more information about the setmode function, see CB: 6.



setuauto and setuman Functions

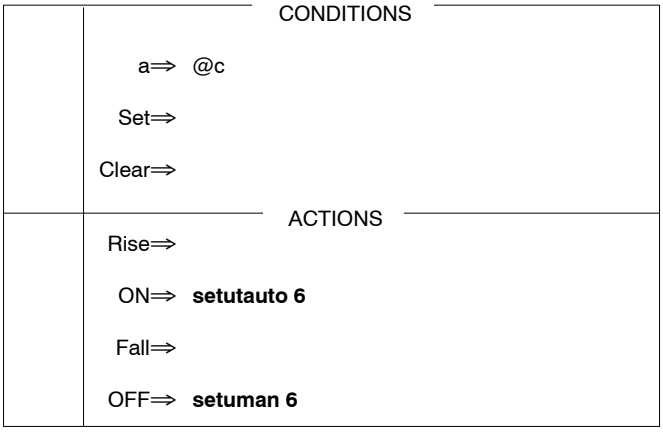
- The following functions set the logic step to Auto or Manual mode.
  - setuauto Sets the logic step to Auto mode.
  - setuman Sets the logic step to Manual mode.
- The setuauto function is expressed as **setuauto x**

  where x is the number corresponding to the particular logic step (for example, the number assigned to logic step a is 1, logic step p is 16).
- The setuman function is expressed as **setuman x**

  where x is the number corresponding to the particular logic step (for example, the number assigned to logic step a is 1, logic step p is 16).

The example below shows a logic statement that sets the mode of logic step f (6) to Auto when the step is on and sets the logic step mode to Manual when the step is off.

- For more information about the setuauto and setuman functions, see CB: 6.



time Function

- The time function provides a true/false value that depends on the current clock time or date. The possible time functions are listed below:

Time Functions

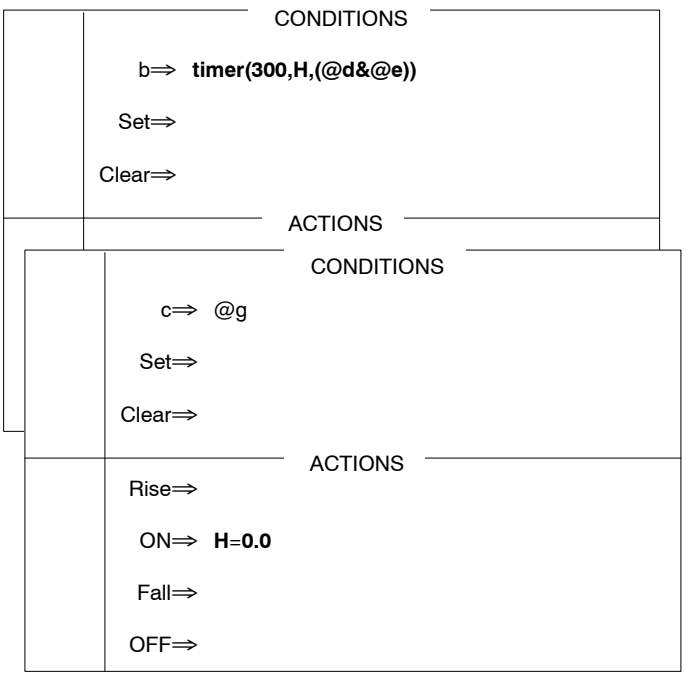
Func	Time or Date Parameter	Func	Time or Date Parameter
time 0	Number of minutes since January 1.	time 7	Day of the week (Sunday=1).
time 1	Current number of seconds on the system clock.	time 15	Number of days in the current month.
time 2	Current number of minutes on the system clock.	time 16	Number of days in the current year.
time 3	Current number of hours on the system clock.	time 21	Clock time in *hhmmss*.
time 4	Day of the month.	time 22	Clock time in *yymmdd*.
time 5	Month of the year (January=1).	time 23	Clock time in *mmddyy*.
time 6	Year AD (for example, 1989).	time 24	Clock time in *ddmmyy*.

- Some example time functions are listed below:
  - The expression “time 3<8” is true when the hour is less than 8.
  - The expression “time 7==1” is true all day Sunday.
  - The expression “time 2==30” is true at 30 minutes after the hour until 31 minutes after the hour.
- An expression that is to be true for a given span of clock time must be written with an **and** function.

  For example, to specify the time from 9:00 AM to 11:30 AM each day, the following expression must be used: (time 21>=90000) & (time 21<=113000)
- For more information about the time function, see CB: 6.

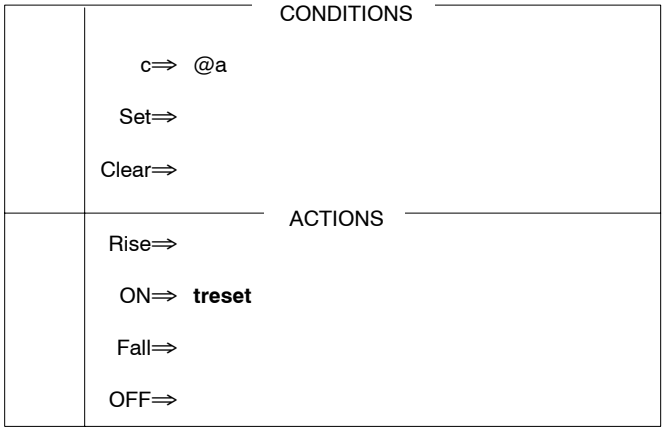
timer Function

- The timer function is a timer that turns on a logic step after a condition has been true for a **cumulative** length of time. The timer expression must be reset by a separate logic step action.
- The timer function is expressed as **timer(X,Y,Z)**.
  - X is the **cumulative** number of seconds that Z must be true to turn on the step.
  - Y is the timer register (*TIMER).
  - Z is a condition.
- In the example below, the timer (input H) runs whenever @d and @e are true. When the timer reaches 300 seconds, step b is turned on. The timer will be reset to zero whenever @g becomes true.
- For more information about the timer function, see CB: 6.



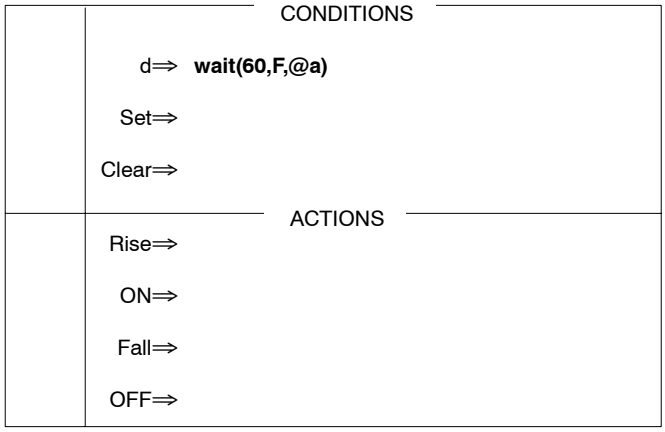
treset Function

- The treset function resets a stack totalizer, shifts inputs B, C, and D, and thus bypasses the “Periodic Reset” fields on the Continuous Faceplate screen for a TOT function ControlBlock.
- The treset function is expressed as **treset** with no parameters.
- The example below shows a sample logic step configured to reset a stack totalizer.
- For more information about the treset function, see CB: 6



wait Function

- The wait function is a timer that turns the logic step on after a condition has been true for an uninterrupted length of time.
- The wait function is expressed as **wait(X,Y,Z)**.
  - X is the **uninterrupted** number of seconds that Z must be true to turn on the step.
  - Y is the timer register (*TIMER).
  - Z is a condition.
- The figure below shows an example of a wait statement. In the example, the input @a must be true for at least 60 continuous seconds for the output step a to turn on.
- For more information about the wait function, see CB: 6.



sstand, snorm, ifstand, inha, enba Functions

- The **sstand**, **snorm, ifstand, inha**, and **enba** functions affect the entire Controller Processor. If you use one or more of these functions, they should be configured in a single ControlBlock with an appropriate tag and descriptor to provide a simple trace.
- The sstand, snorm, inha, and enba functions are configured in the Actions part of the logic step. The ifstand function is configured in the Conditions part of the logic step.
- The **sstand** function sets the Controller Processor card to Standby mode.
- The **snorm** function sets the Controller Processor to Normal mode.
- The **ifstand** function returns a value of 1 if the Controller Processor is in Standby mode. Otherwise, a value of 0 is returned.
- The **inha** function inhibits all alarms and events on the Controller Processor. When the inha function is executed, all active alarms for the Controller Processor are cleared and an alarm is generated.
- The **enba** function enables all alarms and events on the Controller Processor.

sstand, snorm, ifstand, inha, enba Functions (continued)

- In the example below, when input @a is on, the Controller Processor is set to Normal mode.
- For more information about these functions, see CB: 6.

	CONDITIONS
	b⇒ **@a**
	Set⇒
	Clear⇒
	ACTIONS
	Rise⇒
	ON⇒ **snorm**
	Fall⇒
	OFF⇒

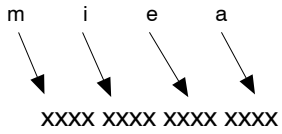
Section 4:
Flags

Flags

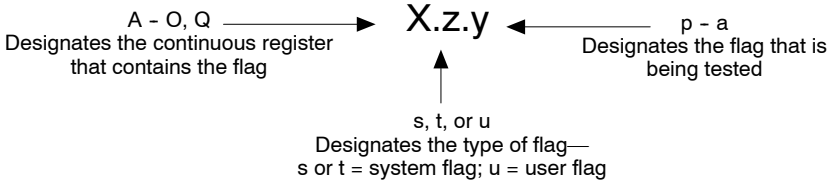
- When a block or console is linked to a ControlBlock, an information packet is continually passed to that block.
- The ControlBlock information packet contains 64-bit representations of block and status information consisting of system flags, user flags, and an analog value or variable.
- Format
  - #0000 0010 0000 0000 is a binary representation of flags.
  - $0200 is a hexadecimal representation of flags.
  - For example: A.u==#0000 0110 0010 1010 and A.u==$062A both test for the same condition.

Flag Notation

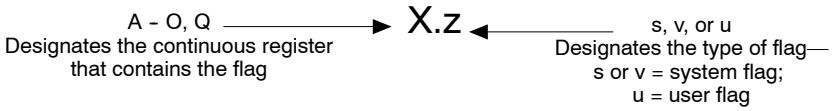
- The following shows how the letters p–a are used to refer to flag positions in the system and user flags.



- Flags can be tested individually as shown below.
  Examples:
  - A.s.a Tests system flag a of register A
  - C.u.p Tests user flag p of register C
  - Q.t.e Tests system flag e of the continuous output Q



- Flags can be tested all at once using the following notation:
  Examples:
  - A.s Tests the system flags of register A
  - C.u Tests the user flags of register C
  - Q.v Tests the system flags of the continuous output Q



User and System Flags

- User and system flags are true/false values that represent block and status information.
- Testing for I/O block flags in a ControlBlock:
  - When an I/O block is linked to a ControlBlock continuous input, the I/O block user and system flags are sent to the ControlBlock.
  - The ControlBlock continuous input contains the I/O block system flags, d (valid data) and g (hold forward), and the ControlBlock link system flags, a–c, e–f, and h–p.
  - The ControlBlock continuous input user flags contain alarm status information for the I/O block.
  - For example, if an I/O block is linked to input A of a ControlBlock, then:
    - A.u==$0 tests for all user flags to be equal to zero.
    - A.s.d==0 tests for system flag d to be equal to zero.
- Testing for ControlBlock flags from a continuous input:
  - When a ControlBlock is linked to another ControlBlock, the source continuous input ControlBlock user and system flags are sent to the destination ControlBlock.
  - System flags:
    - Z.s.x and Z.s (Z is the continuous input) access continuous input system flags.
  - User flags:
    - Z.u.x and Z.u (Z is the continuous input) access continuous input user flags.
  - For example, if a ControlBlock is linked to input E of another ControlBlock, then:
    - E.s.g==1 tests for a hold forward flag from the source ControlBlock.
    - E.u.b==0 tests for user flag b of the source ControlBlock to be 0, or off.

User and System Flags (continued)

- Testing for ControlBlock discrete outputs
  - When output Q is linked to another ControlBlock, the source block discrete outputs can be tested with the destination block Q user flags.
  - For example:
    - Q.u.b==0 tests for discrete output b of the source ControlBlock to be 0, or off.
- Testing for ControlBlock flags within a block:
  - System flags:
    - Q.v accesses alarm, tracking, and logic step flags of a block
    - Q.t.x accesses mode and rate limit flags of a block
  - For example:
    - Q.t.m==1 tests for a rate limit inhibit flag of that ControlBlock
- For more information about system and user flags, see CB: 7.

ControlBlock Input System Flags for Continuous Inputs

ControlBlock Block System Flags for Continuous Inputs A through O (Z.s and Z.s.x)

Letter	Bit Representation	Description
p	1xxx xxxx xxxx xxxx	Critical high alarm
o	x1xx xxxx xxxx xxxx	Critical low alarm
n	xx1x xxxx xxxx xxxx	Advisory high alarm
m	xxx1 xxxx xxxx xxxx	Advisory low alarm
l	xxxx 1xxx xxxx xxxx	Rate-of-change alarm
k	xxxx x1xx xxxx xxxx	Critical high message
j	xxxx xx1x xxxx xxxx	Critical low message
i	xxxx xxx1 xxxx xxxx	Advisory high message
h	xxxx xxxx 1xxx xxxx	Advisory low message
g(1)	xxxx xxxx x1xx xxxx	Hold forward
f	xxxx xxxx xx1x xxxx	Logic active
e	xxxx xxxx xxx1 xxxx	Tracking active
d(1)(2)	xxxx xxxx xxxx 1xxx	Valid data
c	xxxx xxxx xxxx x1xx	Rate of change message
	xxxx xxxx xxxx xx11	Unconfigured block

(1) These flags contain information about the source ControlBlock. All other flags are developed in the destination ControlBlock based on events, such as alarm or logic, that occur in the destination block.

(2) When redundant controllers switch, valid data flag resets for one scan.

ControlBlock System Flags for Continuous Output

ControlBlock System Flags for Continuous Output Q (Q.t.x)*

Letter	Bit Representation	Description
p	1xxx xxxx xxxx xxxx	Output high limit
o	x1xx xxxx xxxx xxxx	Output low limit
n	xx1x xxxx xxxx xxxx	Manual output change rate limit
m	xxx1 xxxx xxxx xxxx	Rate limit inhibit
l	xxxx 1xxx xxxx xxxx	Tracking flag VH (value is high lim for the output)
k	xxxx x1xx xxxx xxxx	Tracking flag VL (value is low lim for the output)
j	xxxx xx1x xxxx xxxx	Tracking flag QH (output must not go any higher)
i	xxxx xxx1 xxxx xxxx	Tracking flag QL (output must not go any lower)
h	xxxx xxxx 1xxx xxxx	Now—the block is evaluating this evaluation cycle
g	xxxx xxxx x1xx xxxx	Hold forward
f	xxxx xxxx xx1x xxxx	Logic active on the output
e	xxxx xxxx xxx1 xxxx	Tracking active
d	xxxx xxxx xxxx 1xxx	Not used
	xxxx xxxx xxxx x101	Comp SP mode
	xxxx xxxx xxxx x100	DDC mode
	xxxx xxxx xxxx x011	Remote mode
	xxxx xxxx xxxx x010	Auto mode
	xxxx xxxx xxxx x001	Manual mode
	xxxx xxxx xxxx x000	Local mode

* There is no Q.t function.

ControlBlock System Flags for Continuous Output Q (Q.v)*

Letter	16 Bit Code	Description
	1xxx xxxx xxxx xxxx	Critical high
	x1xx xxxx xxxx xxxx	Critical low
	xx1x xxxx xxxx xxxx	Alarm inhibit
	xxx1 xxxx xxxx xxxx	Advisory high
	xxxx 1xxx xxxx xxxx	Advisory low
	xxxx x1xx xxxx xxxx	Invalid function
	xxxx xx1x xxxx xxxx	Critical deviation
	xxxx xxx1 xxxx xxxx	Advisory deviation
	xxxx xxxx 1xxx xxxx	Insufficient inputs
	xxxx xxxx x1xx xxxx	Wrong input type
	xxxx xxxx xx1x xxxx	Logic problem
	xxxx xxxx xxx1 xxxx	Invalid track count
	xxxx xxxx xxxx 1xxx	Input configuration error
	xxxx xxxx xxxx x101	Comp SP mode requested
	xxxx xxxx xxxx x100	DDC mode requested
	xxxx xxxx xxxx x011	Remote mode requested
	xxxx xxxx xxxx x010	Auto mode requested
	xxxx xxxx xxxx x001	Manual mode requested
	xxxx xxxx xxxx x000	Local mode requested

* There is no Q.v.x function.

Console Flags

- When a console node is linked to a ControlBlock continuous input, the console user and system flags are sent to the ControlBlock.
- The ControlBlock continuous input user flags contain the console user flags.
- The ControlBlock continuous input system flags contain the console system flag d.
- When you link a console in the ControlBlock Continuous Links screen Source field, you enter the console address followed by A1. For example =28A1.
- For more information about console flags, see CB: 7.

Console System Flags

Letter	16 Bit Code	Description
d	xxxx xxxx xxxx 1xxx	Valid data (Always set to 1 in the link message reply from the console)*

* By testing this flag, the ControlBlock can determine whether a console is present.

Console User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Reserved
o	x1xx xxxx xxxx xxxx	Acknowledge button pressed
n	xx1x xxxx xxxx xxxx	Unacknowledged, cleared Batch alarm
m	xxx1 xxxx xxxx xxxx	Unacknowledged, active Batch alarm
l	xxxx 1xxx xxxx xxxx	Acknowledged, active hardware alarm
k	xxxx x1xx xxxx xxxx	Acknowledged, active critical alarm
j	xxxx xx1x xxxx xxxx	Acknowledged, active advisory alarm
i	xxxx xxx1 xxxx xxxx	Acknowledged, active system alarm
h	xxxx xxxx 1xxx xxxx	Unacknowledged, cleared hardware alarm
g	xxxx xxxx x1xx xxxx	Unacknowledged, cleared critical alarm
f	xxxx xxxx xx1x xxxx	Unacknowledged, cleared advisory alarm
e	xxxx xxxx xxx1 xxxx	Unacknowledged, cleared system alarm
d	xxxx xxxx xxxx 1xxx	Unacknowledged, active hardware alarm
c	xxxx xxxx xxxx x1xx	Unacknowledged, active critical alarm
b	xxxx xxxx xxxx xx1x	Unacknowledged, active advisory alarm
a	xxxx xxxx xxxx xxx1	Unacknowledged, active system alarm

Analog Input Block User Flags

AIB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Critical high alarm active
o	x1xx xxxx xxxx xxxx	Critical low alarm active
	xx1x xxxx xxxx xxxx	Not assigned
m	xxx1 xxxx xxxx xxxx	Advisory high alarm active
l	xxxx 1xxx xxxx xxxx	Advisory low alarm active
k	xxxx x1xx xxxx xxxx	Default calibration alarm active
j	xxxx xx1x xxxx xxxx	Hardware high alarm active
i	xxxx xxx1 xxxx xxxx	Hardware low alarm active
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm active
g	xxxx xxxx x1xx xxxx	Not usable
	xxxx xxxx xx00 xxxx	I/O Type FIC
	xxxx xxxx xx01 xxxx	I/O Type FIM
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

Analog Output Block User Flags

AOB User Flags

Letter	16 Bit Code	Description
	111x xxxx xxxx xxxx	Not assigned
m	xxx1 xxxx xxxx xxxx	Input configuration alarm
	xxxx 1xxx xxxx xxxx	Not assigned
k	xxxx x1xx xxxx xxxx	Default calibration alarm
j	xxxx xx1x xxxx xxxx	Invalid input alarm
	xxxx xxx1 xxxx xxxx	Not assigned
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm
g	xxxx xxxx x1xx xxxx	Not usable
	xxxx xxxx xx00 xxxx	I/O Type FIC
	xxxx xxxx xx01 xxxx	I/O Type FIM
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x000	Manual Mode
	xxxx xxxx xxxx x001	Auto Mode
	xxxx xxxx xxxx x010	Override mode

Contact Input Block & Discrete Input Block User Flags

CIB and DIB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Critical on alarm active
o	x1xx xxxx xxxx xxxx	Critical off alarm active
	xx1x xxxx xxxx xxxx	Not assigned
m	xxx1 xxxx xxxx xxxx	Advisory on alarm active
l	xxxx 1xxx xxxx xxxx	Advisory off alarm active
k	xxxx x1xx xxxx xxxx	Raw state
j	xxxx xx1x xxxx xxxx	Hardware on alarm active
i	xxxx xxx1 xxxx xxxx	Hardware off alarm active
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm active
g	xxxx xxxx x1xx xxxx	Not usable
f	xxxx xxxx xx1x xxxx	Output state
e	xxxx xxxx xxx1 xxxx	Filtered value
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x010	Override mode
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

Contact Output Block & Discrete Output Block User Flags

COB and DOB User Flags

Letter	16 Bit Code	Description
	111x xxxx xxxx xxxx	Not assigned
m	xxx1 xxxx xxxx xxxx	Input configuration alarm
	xxxx 11xx xxxx xxxx	Not assigned
j	xxxx xx1x xxxx xxxx	Invalid input link
i	xxxx xxx1 xxxx xxxx	Value from input link
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm
g	xxxx xxxx x1xx xxxx	Not usable
f	xxxx xxxx xx1x xxxx	Field value
e	xxxx xxxx xxx1 xxxx	Last field value
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x010	Override mode
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

Smart Transmitter Input Block User Flags

SIB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	FIC/Transmitter comm error
o	x1xx xxxx xxxx xxxx	Transmitter status failure
n	xx1x xxxx xxxx xxxx	Transmitter status warning
m	xxx1 xxxx xxxx xxxx	not assigned
	xxxx 1xxx xxxx xxxx	not assigned
k	xxxx x1xx xxxx xxxx	Default calibration alarm active
j	xxxx xx1x xxxx xxxx	Hardware high alarm active
i	xxxx xxx1 xxxx xxxx	Hardware low alarm active
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm active
g	xxxx xxxx x1xx xxxx	Not usable
	xxxx xxxx xx00 xxxx	I/O Type FIC
	xxxx xxxx xx01 xxxx	I/O Type FIM
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x010	Simulate mode
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

HART Output Block User Flags

HOB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Deviation alarm
o	x1xx xxxx xxxx xxxx	Field device status failure
n	xx1x xxxx xxxx xxxx	Field device status warning
m	xxx1 xxxx xxxx xxxx	Input configuration alarm
	xxxx 1xxx xxxx xxxx	Not assigned
k	xxxx x1xx xxxx xxxx	Default calibration alarm
j	xxxx xx1x xxxx xxxx	Invalid input alarm
	xxxx xxx1 xxxx xxxx	Not assigned
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm
g	xxxx xxxx x1xx xxxx	Hold forward
	xxxx xxxx xx00 xxxx	I/O Type FIC
	xxxx xxxx xx01 xxxx	I/O Type FIM
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

Multiplexer Input Block User Flags

MIB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Critical high alarm active
o	x1xx xxxx xxxx xxxx	Critical low alarm active
n	xx1x xxxx xxxx xxxx	Critical rate alarm active
m	xxx1 xxxx xxxx xxxx	Advisory high alarm active
l	xxxx 1xxx xxxx xxxx	Advisory low alarm active
k	xxxx x1xx xxxx xxxx	Advisory rate alarm active
j	xxxx xx1x xxxx xxxx	Hardware high alarm active
i	xxxx xxx1 xxxx xxxx	Hardware low alarm active
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm active
g	xxxx xxxx x1xx xxxx	Not usable
f	xxxx xxxx xx1x xxxx	Critical deviation alarm active
e	xxxx xxxx xxx1 xxxx	Advisory deviation alarm active
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

Pulse Input/Output Block User Flags

PIOB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Illegal PIO configuration
o	x1xx xxxx xxxx xxxx	Not assigned
n	xx1x xxxx xxxx xxxx	Not assigned
m	xxx1 xxxx xxxx xxxx	Input configuration alarm
l	xxxx 1xxx xxxx xxxx	Invalid input alarm
k	xxxx x1xx xxxx xxxx	Counter is at or above target
j	xxxx xx1x xxxx xxxx	Counter is disabled
i	xxxx xxx1 xxxx xxxx	Input cutoff active
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm active
g	xxxx xxxx x1xx xxxx	Not usable
f	xxxx xxxx xx1x xxxx	Copy of input state
e	xxxx xxxx xxx1 xxxx	Not assigned
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

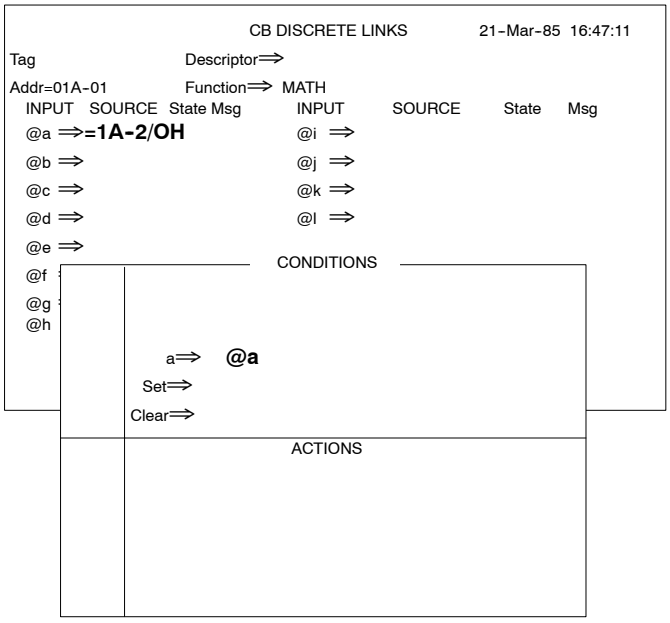
Temperature Input Block User Flags

TIB User Flags

Letter	16 Bit Code	Description
p	1xxx xxxx xxxx xxxx	Critical high alarm active
o	x1xx xxxx xxxx xxxx	Critical low alarm active
	xx1x xxxx xxxx xxxx	Input configuration alarm
m	xxx1 xxxx xxxx xxxx	Advisory high alarm active
l	xxxx 1xxx xxxx xxxx	Advisory low alarm active
k	xxxx x1xx xxxx xxxx	Default calibration active for at least 1 of the 8 calibration ranges
j	xxxx xx1x xxxx xxxx	Hardware high alarm active
i	xxxx xxx1 xxxx xxxx	Hardware low alarm active
h	xxxx xxxx 1xxx xxxx	Hardware misc alarm active
g	xxxx xxxx x1xx xxxx	Not usable
	xxxx xxxx xx11 xxxx	Not assigned
d	xxxx xxxx xxxx 1xxx	Not usable
	xxxx xxxx xxxx x001	Auto mode
	xxxx xxxx xxxx x000	Manual mode

Block Status Bits

- Block status bits contain some of the same information as the system flags and, therefore, provide an easy way to access some system flags.
- The status bits for I/O blocks contain alarm information and some of the system flags.
- The status bits for ControlBlocks contain information such as the ControlBlock mode, output limits, and some of the system flags.
- You use a code (two alphabetical letters or a question mark followed by a number) to test block status bits.
- The tables on the following pages show the codes that are used to link the block status bits.
- For example, in the figure below discrete input @a is used to represent the output high limit block status bit of ControlBlock =1A–2. Logic step a turns on when input @a is on—that is, when ControlBlock =1A–2 has reached its output high limit.
- For more information about block status bits, see CB:7.



Example Using Block Status Bits

I/O Block Status Bit Codes

I/O Block Status Bits (except PLCB)

Code	Description
?1	System flag b
?2	System flag c
?3	System flag d—valid data
?4	System flag e
?6	System flag f
?A	System flag k
?D	System flag n
AH	Advisory high alarm
AL	Advisory low alarm
CH	Critical high alarm
CL	Critical low alarm
HF	Hold forward (value is not good)
HH	Hardware high alarm
HL	Hardware low alarm
HM	Hardware miscellaneous alarm (faulty I/O hardware)

Programmable Logic Controller Block Status Bit Codes

PLCB Status Bits

Code	Description
?1	System flag b—Mode ON=auto, fail OFF=simulate
?2	System flag c—Mode ON=auto, fail OFF=simulate
?3	System flag d—Output valid
?4	System flag e—PLC read exception error
?6	System flag f—PLC address error
?A	System flag k—Trying to write to a read only address
?D	System flag n—First time block has been evaluated
AH	System flag m—Input configuration alarm
AL	System flag l—PLC write exception error
CH	Wrong PLC port in redundant mode
CL	System flag o—Too many PLCs configured
HF	System flag g—Hold forward
HH	System flag j—Invalid input alarm
HL	Not assigned
HM	System flag h—Hardware misc alarm active

ControlBlock Status Bit Codes

ControlBlock Status Bits

Code	Description
?2	System flag c—ControlBlock mode 0=LOCAL 1=DDC, Comp SP, Manual, Auto, or Remote
?3	System flag d—Valid data
?4	System flag e—Tracking active
?8	System flag i—Tracking flag QL (output must not go any lower)
?9	System flag j—Tracking flag QH (output must not go any higher)
?A	System flag k—Tracking flag VL (value is a low limit for the output)
?B	System flag l—Tracking flag VH (value is a high limit for the output)
?C	System flag m—Rate limit inhibit
HF	Hold forward
LA	Logic active on the output
MD	Block mode must be linked to a continuous input 0=LOCAL 1=MANUAL 2=AUTO 3=REMOTE 4=COMPUTER
NW	Now (this evaluation cycle)—the block is evaluating
OH	Output high limited
OL	Output low limited
RL	Manual output change rate limit

Section 5:
MTCC Commands

Using MTCC Commands

This section lists commands that can be entered from the command line at the top of the MTCC screen. You can access these commands by typing the indicated letters, or by repeatedly pressing [NEXT OPTION] and [PREV OPTION].

The first two tables list MTCC commands that call up screens. The first table is organized by primary function, while the second is an alphabetical list of all MTCC screen callup commands.

The third and fourth tables list the MTCC commands that perform operations, first by primary function and then alphabetically by command name.

NOTE: The availability of some options may depend on the system configuration.

MTCC Commands that Call Up Screens

MTCC Commands that Call Up Screens

Screen Called Up	Entry	Command Line Text
Alarm Areas		
Area Name Configuration	**ANC**	Area Name Config:
Graphic for specified area	**ANG**	Area Name Graphic:
Alarms and Events		
Alarm Configuration	**AC**	Alarm Configuration
Alarm List Configuration	**ALC**	Alarm List Configuration
Alarm Log Display (for remote console or archived files)	**ALD** *node#, filename*	Alm Log Disp:
Alarm Log Display (for local console only)	**ALS** *LDC filename*	Alm Log Search:
Configure Alarm Broadcast	**CAB**	Cnfg Alarm Broadcast
Event List Configuration	**ELC**	Event List Configuration
Log Display Configuration	**LDC**	Log Display Config:
Alarm and Event Lists		
Active Batch Alarms	**BAAA**	BATCH Active Alarms
Alarm Annunciation	**AAC**	Alarm Annunciation Cnf:
Active Hardware Alarms	**HA**	Hardware Active Alarms
Active Process Alarms	**AAL** *or* **PA**	Alarm Active List Process Active Alarms
Active Suppressed Alarms	**ASA**	Alarm Suppressed Actv
Active System Status	**SSA**	System Status Active
Advisory Active Alarms	**ADAA**	ADV Actv Alrms
Alarm and Event Lists (continued)		

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Advisory Cleared Alarms	**ADCA**	ADV Clrd Alrms
Cleared Batch Alarms	**BACA**	BATCH Cleared Alarms
Cleared Hardware Alarms	**HC**	Hardware Cleared Alarms
Cleared Process Alarms	**PC**	Process Cleared Alarms
Cleared Suppressed Alarms	**ASC**	Alarm Suppressed Clrd
Cleared System Status	**SSC**	System Status Cleared
Critical Active Alarms	**CRAA**	CRIT Actv Alarms
Critical Cleared Alarms	**CRCA**	CRIT Clrd Alarms
Disk Event List	**DEL**	Disk Event List
Event List	**EL:**	Event List:
Plant Area (ACTV)	**AN:**	Area Name:
Plant Area (ACTV)	**AAA**	Actv Area Alms:
Plant Area (CLRD)	**CAA**	Cleared Area Alms:
Batch & RBLC		
Batch Acquire Queues	**BAQ**	BATCH Queues:
Batch Configuration	**BAC:**	BATCH Config:
Batch Control Recipe Config	**BACR**	BATCH Control Rec:
Batch Formula Table	**BAFT:**	BATCH Formula Tbl:
Batch Input	**BAI**	BATCH Input:
Batch Log	**BAL:**	BATCH Log:
Batch Master Recipe Config	**BAMR**	BATCH Master Rec:
Batch Material Properties	**BAMP**	BATCH Mat Prop
Batch & RBLC (continued)		

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Batch Materials Table	**BAMT**	BATCH Material Tbl:
Batch Monitor	**BAM:**	BATCH Monitor:
Batch Operations Table	**BAOT**	BATCH Oper Tbl:
Batch Overview	**BAO:**	BATCH Overview:
Batch Plant Unit Status	**BAP**	BATCH Plant Unit
Batch Run	**BAR:**	BATCH Run:
Batch Script	**BAS:**	BATCH Script:
Batch Unit Set	**BAUS:**	BATCH Unit Set:
Batch Units Table	**BAUT**	Batch Unit Tbl:
Batch Working Recipe	**BAWR**	BATCH Working Rec:
RBL Directory (Batch)	**BAD**	BATCH Directory:
RBL Directory (RBL)	**RBLD**	RBL Directory:
RBL File Contents (Batch)	**BAF**	BATCH File Cont:
RBL File Contents (RBL)	**RBLF**	RBL File Contents:
RBL File Transfer	**FT**	File Transfer
RBLC Monitor	**RBLM**	RBLC Monitor
RBLC Script	**RBLS**	RBLC Script:
Block		
Block Directory	**CBD**:	Cont Block Directory:
Block Print Setup	**BP**	Block Print
Block References	**BR**	Block References:
Block Status	**CBS**	Cont Block Status:
Block (continued)		

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Continuous Diagram	**BD**	Block Diagram:
Continuous Faceplate	**BF**	Block Faceplate:
Continuous Links	**BL**	Block Links:
Group Display screen or Continuous Faceplate	**TA**	Tag/Address:
Callup Buttons		
Alarm Annunciation	**AAC**	Alm Annunciation Cnf:
Callup Buttons	**CCB**	Config Callup Buttons
Calls up screen or performs the operation that is assigned to a callup button.	**CUB**:	Call Up Button:
Display Buttons	**CDB**	Config Display Buttons:
Color		
Color Configuration	**CCU**	Config Color Usage
Master Color Selection	**CCP**	Config Color Palette
Console Configuration		
Console Configuration	**CCC**	Config Command Console
Console Menu		
Menu Command Console	**MCC**	Menu Command Console
ControlFile and I/O		
ControlFile Dump	**CFD** ***node#*** *or* ***CP#***	Cont File Dump:
ControlFile Dump or Node Dump	**ND**	Node Dump:
ControlFile Links	**CFL**	Cont File Links:
ControlFile and I/O (continued)		

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
ControlFile Status	**CFS**	Cont File Status:
Field I/O Status	**FS**	Field I/O Status:
Transmitter Maint Log	**TL**	Trans Log:
Diagnostics		
Background Diagnostics	**MCT**	Menu Confidence Tests
Memory View	**MV**	Memory View
Diogenes Interface		
Diogenes I/F Configuration	**CDI**	Configure Diogenes I/F:
Disk and File		
Calls up specified file	**DDF**	Disk Dir File:
Configure Download Script	**CDS**	Cfg Dnld Script:
Disk Activity (last screen)	**DA**	Disk Activity
Disk Directory	**DD:**	Disk Dir:
Disk Directory PeerWay	**DDP**	Disk Dir PeerWay
Disk Folder Configuration	**DC:**	Disk Configure:
Graphics		
Graphic Viewing	**PG:**	Pr Graph:
Link Editor	**LE**	Link Edit:
Process Graphics Generation	**PGG**	Pr Graph Gen:
Process Graphics Symbol generation screen	**PGS**	Pr Graph Sym:
HIA		
Configure HIA	**CH**	Config HIA:
Login		

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
User Name Detail	**UND**	User Name Detail
Plant Unit Access	**PUA**	Plant Unit Access
User Profile Directory	**UPD**	User Profile Directory
Login	**CUP**	Change User Password
Login	**LI**	Log In:
Log out	**LO**	Log Out:
Message Pairs		
STD Message Pairs	**MPS**	Message Pairs Std
User Message Pairs	**MPU**	Message Pairs User
Operating Displays		
Group	**G:**	Group:
Group Directory	**GD:**	Group Directory:
Overview	**O:**	Overview:
Overview Directory	**OD:**	Overview Directory:
Unit	**U:**	Unit:
Unit Directory	**UD:**	Unit Directory:
Operator Log		
Operator Change Log	**OL**	Operator Log
PeerWay and Plant		
PeerWay Node	**PN**	PeerWay Node:
PeerWay Overview	**PO**	PeerWay Overview:
PeerWay Performance	**PP**	PeerWay Performance

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
PeerWay and Plant (continued)		
Plant Status	**PS**	Plant Status:
Plant Unit Configuration	**PUC**	Plant Unit Configuration
Plant Configuration Backup		
Backup Plant Config	**BUP**	(??)
PLC		
Configure PLC	**CP**	Configure PLC:
Reports		
Report Configuration	**RG**	Reprt Gen:
Report Configuration Directory	**RCD**	Reprt Config Directory:
Report File Contents	**RFC**	Reprt File Contents:
Report Read Directory	**RRD:**	Reprt Read Directory:
Report Read/Scan	**RR:**	Reprt Read:
Report Status	**RS**	Reprt Status:
RFI		
Configure RFI	**CR:**	Configure RFI:
RNI		
Configure RNI	**CRN**	Configure RNI:
RNI Status	**CRN (EXCH)**	Configure RNI:
SCI		
Configure SCI	**CS**	Configure SCI:
Trend		
Trend File Configuration	**TFC**	Trend File Config:

(continued on next page)

MTCC Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Trend (continued)		
Trend File Setup	**TFS**	Trend File Setup
Trend Group	**TG:**	Trend Group:
Trend Group Configuration	**TGC**	Trend Group Config:
Trend Group Menu	**TGD**: *or* **TGM**	Trend Group Directory Trend Group Menu
Tuning Displays		
Tuning Display	**TD:**	Tuning Display
Tuning Display Directory	**TDD**	Tuning Display Directory
VAX		
Host Mode	**HM**	Hostmode:
VAX RPQNA Status Display	**VS**	VAX Status:

Alphabetical List of MTCC Screen-Callup Commands

Alphabetical List of MTCC Screen-Callup Commands

Entry	Command Line Text	Screen Called Up
AAA	Actv Area Alms:	Plant Area (ACTV)
AAC	Alm Annunciation Cnf:	Alarm Annunciation
AAL	Alarm Active List	Active Process Alarms
AC	Alarm Configuration	Alarm Configuration
ADAA	ADV Actv Alrms	Advisory Active Alarms
ADCA	ADV Clrd Alrms	Advisory Cleared Alarms
ALC	Alarm List Configuration	Alarm List Configuration
ALD *node#, filename*	Alm Log Disp:	Alarm Log Display (for remote console or archived files)
ALS *LDC filename*	Alm Log Search:	Alarm Log Display (for local console only)
AN:	Area Name:	Plant Area (ACTV)
ANC	Area Name Config:	Area Name Configuration
ANG	Area Name Graphic:	Graphic for specified area
APB	Abort Plant Config Backup	(??)
ASA	Alarm Suppressed Actv	Active Suppressed Alarms
ASC	Alarm Suppressed Clrd	Cleared Suppressed Alarms
BAAA	BATCH Active Alarms	Active Batch Alarms
BAC:	BATCH Config:	Batch Configuration
BACA	BATCH Cleared Alarms	Cleared Batch Alarms
BACR	BATCH Control Rec:	Batch Control Recipe Config
BAD	BATCH Directory:	RBL Directory (Batch)

(continued on next page)

Alphabetical List of MTCC Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
BAF	BATCH File Cont:	RBL File Contents (Batch)
BAFT	BATCH Formulas Tbl:	Batch Formulas Table
BAI	BATCH Input:	Batch Input
BAL:	BATCH Log:	Batch Log
BAM:	BATCH Monitor:	Batch Monitor
BAMP	BATCH Mat Prop	Batch Material Properties
BAMR	BATCH Master Rec:	Batch Master Recipe Config
BAMT	BATCH Material Tbl:	Batch Materials Table
BAO:	BATCH Overview:	Batch Overview
BAOT	BATCH Oper Tbl:	Batch Operations Table
BAP	BATCH Plant Unit	Batch Plant Unit Status
BAQ	BATCH Queues:	Batch Acquire Queues
BAR:	BATCH Run:	Batch Run
BAS:	BATCH Script:	Batch Script
BAUS	BATCH Unit Set	Batch Unit Set
BAUT	BATCH Unit Tbl:	Batch Units Table
BAWR	BATCH Working Rec:	Batch Working Recipe
BD	Block Diagram:	Continuous Diagram
BF	Block Faceplate:	Continuous Faceplate
BL	Block Links:	Continuous Links
BP	Block Print	Block Print Setup
BR	Block References:	Block References
BUP	(??)	BackUp Plant Config

(continued on next page)

Alphabetical List of MTCC Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
CAA	Cleared Area Alms:	Plant Area (CLRD)
CAB	Cnfg Alarm Broadcast	Configure Alarm Broadcast
CBD:	Cont Block Directory:	Block Directory
CBS	Cont Block Status:	Block Status
CCB	Config Callup Buttons	Callup Buttons
CCC	Config Command Console	Console Configuration
CCP	Config Color Palette	Master Color Selection
CCU	Config Color Usage	Color Configuration
CDB	Config Display Buttons:	Display Buttons
CDI	Configure Diogenes I/F:	Diogenes I/F Configuration
CDS	Cfg Dnld Script:	Configure Download Script
CFD ***node#*** *or* ***CP#***	Cont File Dump:	ControlFile Dump
CFL	Cont File Links:	ControlFile Links
CFS	Cont File Status:	ControlFile Status
CH	Config HIA:	Configure HIA
CP	Configure PLC:	Configure PLC
CR:	Configure RFI:	Configure RFI
CRAA	CRIT Actv Alarms	Critical Active Alarms
CRCA	CRIT Clrd Alarms	Critical Cleared Alarms
CRN	Configure RNI	Configure RNI
CRN [EXCH]	RNI Status	RNI Status
CS	Configure SCI:	Configure SCI

(continued on next page)

Alphabetical List of MTCC Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
CUB:	Call Up Button:	Calls up screen or performs the operation that is assigned to a callup button.
CUP	Change User Password	Login
DA	Disk Activity	Disk Activity (last screen)
DC:	Disk Configure:	Disk Folder Configuration
DCB	Disable Console Config Backup	(??)
DD:	Disk Dir:	Disk Directory
DDF	Disk Dir File:	Calls up specified file
DDP	Disk Dir PeerWay	Disk Directory PeerWay
DPB	Disable Plant Config Backup	(??)
DEL	Disk Event List	Disk Event List
ECB	Enable Console Config Backup	(??)
EL:	Event List:	Event List
ELC	Event List Configuration	Event List Configuration
EPB	Enable Plant Config Backup	(??)
FS	Field I/O Status:	Field I/O Status
FT	File Transfer	RBL File Transfer
G:	Group:	Group
GD:	Group Directory:	Group Directory
HA	Hardware Active Alarms	Active Hardware Alarms
HC	Hardware Cleared Alarms	Cleared Hardware Alarms
HM	Hostmode:	Host Mode
LDC	Log Display Config:	Log Display Configuration

(continued on next page)

Alphabetical List of MTCC Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
LE	Link Edit:	Link Editor
LI	Log in:	Login
LO	Log out:	Log out
MCC	Menu Command Console	Menu Command Console
MCT	Menu Confidence Tests	Background Diagnostics
MPS	Message Pairs Std	STD Message Pairs
MPU	Message Pairs User	User Message Pairs
MV	Memory View	Memory View
ND	Node Dump:	ControlFile Dump or Node Dump
O:	Overview:	Overview
OD:	Overview Directory:	Overview Directory
OL	Operator Log	Operator Change Log
PA	Process Active Alarms	Active Process Alarms
PC	Process Cleared Alarms	Cleared Process Alarms
PG:	Pr Graph:	Graphic Viewing
PGG	Pr Graph Gen:	Process Graphics Generation
PGS	Pr Graph Sym:	Process Graphics Symbol generation screen
PN	PeerWay Node:	PeerWay Node
PO	PeerWay Overview:	PeerWay Overview
PP	PeerWay Performance	PeerWay Performance
PS	Plant Status:	Plant Status
PUA	Plant Unit Access	Plant Unit Access

(continued on next page)

Alphabetical List of MTCC Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
PUC	Plant Unit Configuration	Plant Unit Configuration
RBLD	RBL Directory:	RBL Directory (RBL)
RBLF	RBL File Contents:	RBL File Contents (RBL)
RBLM	RBLC Monitor	RBLC Monitor
RBLS	RBLC Script:	RBLC Script
RCD	Reprt Config Directory:	Report Configuration Directory
RFC	Reprt File Contents:	Report File Contents
RG	Reprt Gen:	Report Configuration
RR:	Reprt Read:	Report Read/Scan
RRD:	Reprt Read Directory:	Report Read Directory
RS	Reprt Status:	Report Status
SSA	System Status Active	Active System Status
SSC	System Status Cleared	Cleared System Status
TA	Tag/Address:	Group Display screen or Continuous Faceplate
TD:	Tuning Display	Tuning Display
TDD	Tuning Display Directory	Tuning Display Directory
TFC	Trend File Config:	Trend File Configuration
TFS	Trend File Setup	Trend File Setup
TG:	Trend Group:	Trend Group
TGC	Trend Group Config:	Trend Group Configuration
TGD:	Trend Group Directory	Trend Group Menu
TGM	Trend Group Menu	Trend Group Menu
TL	Trans Log:	Transmitter Maint Log

(continued on next page)

Alphabetical List of MTCC Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
U:	Unit:	Unit
UD:	Unit Directory:	Unit Directory
UND	User Name Detail	User Name Detail
UPD	User Profile Directory	User Profile Directory
VS	VAX Status:	VAX RPQNA Status Display

MTCC Commands that Perform Operations

MTCC Commands That Perform Operations

Operation Performed	Entry	Command Line Text
Alarms and Events		
Ack all alarms on a screen	**SA**	Screen Acknowledge
Alarm area—disable	**AD:**	Area DIsable:
Alarm area—enable	**AE:**	Area Enable:
Alarm banner mode	**ASD**	Alarm Summary Disable
Alarm summary mode	**ASE**	Alarm Summary Enable:
Alarm logging—disable	**DLA**	Disable Logging Alarms:
Alarm logging—enable	**ELA**	Enable Logging Alarms:
Alarm regeneration—initiate	**AR**	Alarm Regen:
Toggles alarm view	**AVT**	Alarm View Toggle
Silence horn	**SH**	Silence Horn
Alarm and Event Lists		
Active Process Alarms List—print Cleared Process Alarms List—print	**P**[sp]**PA** *or* **P**[sp]**AA**	Pr Process Alarm List Pr Alarm Active List
Act Sys Status Alarms List—print Clrd Sys Status Alarms List—print	**P**[sp]**SSA**	Pr System Status Active
Active Hardware Alarm List —print Cleared Hardware Alarm List—print	**P**[sp]**HA**	Pr Hardware Alarm List
Active Batch Alarms List—print Cleared Batch Alarms List—print	**P**[sp]**BA**	Pr BATCH Alarm List
Disk Event List—print	**P**[sp]**DEL**	Pr Disk Event List
Event List—print	**P**[sp]**EL:**	Pr Event List:

(continued on next page)

MTCC Commands That Perform Operations (continued)

Operation Performed	Entry	Command Line Text
Batch		
Batch—disable	**DBAS**	Disable Batch System:
Batch—enable	**EBAS**	Enable Batch System:
Batch Bubble memory—initialize	**BAW**	BATCH Wipe:
Batch Log—disable	**DBAL**	Disable Batch Log:
Batch Log—enable	**EBAL**	Enable Batch Log:
Console		
Abort backup	**AB**	Abort Backup:
Abort Tape Restore	**ATR**	Abort Tape Restore:
Disable console config backup	**DCB**	Disable Console Cfg Backup
Enable console config backup	**ECB**	Enable Console Cfg Backup
Eject a floppy disk	**EF**	Eject Floppy:
Reboot the console	**RBC**	Re-Boot Console
Shut down operation of disk	**DS:**	Disk Shutdown
Console Screens		
Mark—remove	**RM**	Remove Mark:
Next page—call up	**PF**	Page Forward
Previous page—call up	**PB**	Page Backward
Recall the previous screen	**R:**	Recall:
ControlFile		
Kill operation of Controller Proc	**KC**	Kill Controller:
Nonvolatile memory—eliminate areas of nonvolatile memory reserved for non-existent blocks	**FB**	Free Bubble
Nonvolatile memory—initialize configuration information	**IC**	Init CF Nvmem Config:
Wipe bubble memory	**WB**	Wipe Bubble:

(continued on next page)

MTCC Commands That Perform Operations (continued)

Operation Performed	Entry	Command Line Text
Diagnostics		
Stop memory dump before it finishes	**KMD**	Kill Memory Dump
Captures RAM memory of a node in a file	**MD** *node#* **,** *filename*	Memory Dump:
Mask		
Toggle tag mask	**TTM:**	Toggle Tag Mask
Operator Log		
Operator Log—print	**P**[sp]**OL**	Pr Operator Log
Plant Configuration		
Abort plant config backup	**APB**	Abort Plant Cfg Backup
Disable plant config backup	**DPB**	Disable Console Cfg Backup
Enable plant config backup	**EPB**	Enable Console Cfg Backup
Reports		
Print a report	**P**[sp]**R** *or* **RP**	Pr Report: Reprt Prt:
Reports—disable	**DR**	Disable Reports
Reports—enable	**ER**	Enable Reports
Screen print		
Character print of screen	**SC**	Screen Char Print:
Graphic print of screen	**SG**	Screen Graph Print:
Configured mode print of screen	**SP**	Screen Print:
Transmitter Log		
Trans Maintenance Log—disable	**DML**	Disable Maint Log:
Trans Maintenance Log—enable	**EML**	Enable Maint Log:
Trending		
Trending—disable	**DT**	Disable Trending
Trending—enable	**ET**	Enable Trending

Alphabetical List of MTCC Operation Commands

Alphabetical List of MTCC Operation Commands

Entry	Command Line Text	Operation Performed
AB	Abort Backup	Abort backup
AD:	Area Disable:	Alarm area—disable
AE:	Area Enable:	Alarm area—enable
APB	Albort plant config backup	Abort Plant Cfg Backup
AR	Alarm Regen:	Alarm regeneration—initiate
ASD	Alarm Summary Disable	Alarm banner mode
ASE	Alarm Summary Enable:	Alarm summary mode
ATR	Abort Tape Restore	Abort tape restore
AVT	Alarm View Toggle	Toggles alarm view
BAW	BATCH Wipe:	Batch Bubble memory—initialize
DBAL	Disable Batch Log:	Batch Log—disable
DBAS	Disable Batch System:	Batch—disable
DCB	Disable console config backup	Disable Console Cfg Backup
DLA	Disable Logging Alarms:	Alarm logging—disable
DML	Disable Maint Log:	Trans Maintenance Log—disable
DPB	Disable plant config backup	Disable Plant Cfg Backup
DR	Disable Reports	Reports—disable
DS:	Disk Shutdown	Shut down operation of disk
DT	Disable Trending	Trending—disable
EBAL	Enable Batch Log:	Batch Log—enable

(continued on next page)

Alphabetical List of MTCC Operation Commands (continued)

Entry	Command Line Text	Operation Performed
EBAS	Enable Batch System:	Batch—enable
ECB	Enable console config backup	Ensable Console Cfg Backup
EF	Eject Floppy:	Eject a floppy disk
ELA	Enable Logging Alarms:	Alarm logging—enable
EML	Enable Maint Log:	Trans Maintenance Log—enable
EPB	Enable plant config backup	Enable Plant Cfg Backup
ER	Enable Reports	Reports—enable
ET	Enable Trending	Trending—enable
FB	Free Bubble	Nonvolatile memory—eliminate areas of nonvolatile memory reserved for non-existent blocks
IC	Init CF Nvmem Config:	Nonvolatile memory—initialize configuration information
KC	Kill Controller:	Kill operation of Controller Proc
KMD	Kill Memory Dump	Stop memory dump before it finishes
MD *node#*, *filename*	Memory Dump:	Captures RAM memory of a node in a file
PB	Page Backward	Previous page—call up
PF	Page Forward	Next page—call up
P[sp]**AA**	Pr Alarm Active List	Active Process Alarms List—print Cleared Process Alarms List—print
P[sp]**BA**	Pr BATCH Alarm List	Active Batch Alarms List—print Cleared Batch Alarms List—print
P[sp]**HA**	Pr Hardware Alarm List	Active Hardware Alarm List —print Cleared Hardware Alarm List—print
P[sp]**OL**	Pr Operator Log	Operator Log—print

(continued on next page)

Alphabetical List of MTCC Operation Commands (continued)

Entry	Command Line Text	Operation Performed
P[sp]**PA**	Pr Process Alarm List	Active Process Alarms List—print Cleared Process Alarms List—print
P[sp]**R**	Pr Report:	Print a report
P[sp]**DEL**	Pr Disk Event List	Disk Event List—Print
P[sp]**EL:**	Pr Event List:	Event List—Print
P[sp]**SSA**	Pr System Status Active	Act Sys Status Alarms List—print Clrd Sys Status Alarms List—print
R:	Recall:	Recall the previous screen
RBC	Re-Boot Console	Reboot the console
RM	Remove Mark:	Mark—remove
RP	Reprt Prt:	Print a report
SA	Screen Acknowledge	Ack all alarms on a screen
SC	Screen Char Print:	Character print of screen
SG	Screen Graph Print:	Graphic print of screen
SH	Silence Horn	Silence console horn
SP	Screen Print:	Configured mode print of screen
TTM:	Toggle Tag Mask	Toggle between masking and no masking
WB	Wipe Bubble:	Wipe bubble memory

Section 6:
SMS Commands

Using SMS Commands

This section contains tables listing commands that can be entered from the command line at the top of the System Manager Station (SMS) screen. You can access these commands by typing the indicated letters, or by repeatedly pressing [NEXT OPTION] and [PREV OPTION].

The first two tables list SMS commands that call up screens; the commands are organized by primary function in the first table, and alphabetically by command in the second table.

The third and fourth tables list the SMS commands that perform operations; the commands are organized by primary function in the first of these two tables, and alphabetically in the second.

SMS Commands that Call Up Screens

SMS Commands that Call Up Screens

Screen Called Up	Entry	Command Line Text
Alarm Areas		
Area Name Configuration	**ANC**	Area Name Config:
Alarms and Events		
Alarm Configuration	**AC**	Alarm Configuration
Alarm List Configuration	**ALC**	Alarm List Configuration
Alarm Log Display (for remote console or archived files)	**ALD** *node#, filename*	Alm Log Disp:
Alarm Log Display (for local console only)	**ALS** *LDC filename*	Alm Log Search:
Configure Alarm Broadcast	**CAB**	Cnfg Alarm Broadcast
Event List Configuration	**ELC**	Event List Configuration
Log Display Configuration	**LDC**	Log Display Config:
Alarm and Event Lists		
Active Batch Alarms	**BAAA**	BATCH Active Alarms
Alarm Annunciation	**AAC**	Alarm Annunciation Cnf:
Active Hardware Alarms	**HA**	Hardware Active Alarms
Active Process Alarms	**AAL** *or* **PA**	Alarm Active List Process Active Alarms
Active Suppressed Alarms	**ASA**	Alarm Suppressed Actv
Active System Status	**SSA**	System Status Active
Advisory Active Alarms	**ADAA**	ADV Actv Alrms
Advisory Cleared Alarms	**ADCA**	ADV Clrd Alrms
Alarm and Event Lists (continued)		

(continued on next page)

SMS Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Cleared Batch Alarms	**BACA**	BATCH Cleared Alarms
Cleared Hardware Alarms	**HC**	Hardware Cleared Alarms
Cleared Process Alarms	**PC**	Process Cleared Alarms
Cleared Suppressed Alarms	**ASC**	Alarm Suppressed Clrd
Cleared System Status	**SSC**	System Status Cleared
Critical Active Alarms	**CRAA**	CRIT Actv Alarms
Critical Cleared Alarms	**CRCA**	CRIT Clrd Alarms
Disk Event List	**DEL**	Disk Event List
Event List	**EL:**	Event List:
Plant Area (ACTV)	**AN:**	Area Name:
Plant Area (ACTV)	**AAA**	Actv Area Alms:
Plant Area (CLRD)	**CAA**	Cleared Area Alms:
RBLC		
RBL Directory (RBL)	**RBLD**	RBL Directory:
RBL File Contents (RBL)	**RBLF**	RBL File Contents:
RBL File Transfer	**FT**	File Transfer
RBLC Monitor	**RBLM**	RBLC Monitor
RBLC Script	**RBLS**	RBLC Script:
Block		
Block Directory	**CBD**:	Cont Block Directory:
Block Print Setup	**BP**	Block Print
Block References	**BR**	Block References:
Block (continued)		

(continued on next page)

SMS Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Block Status	**CBS**	Cont Block Status:
Continuous Diagram	**BD**	Block Diagram:
Continuous Faceplate	**BF**	Block Faceplate:
Continuous Links	**BL**	Block Links:
Group Display screen or Continuous Faceplate	**TA**	Tag/Address:
Callup Buttons		
Alarm Annunciation	**AAC**	Alm Annunciation Cnf:
Callup Buttons	**CCB**	Config Callup Buttons
Calls up screen or performs the operation that is assigned to a callup button.	**CUB**:	Call Up Button:
Display Buttons	**CDB**	Config Display Buttons:
Color		
Color Configuration	**CCU**	Config Color Usage
Master Color Selection	**CCP**	Config Color Palette
Console Configuration		
Console Configuration	**CCC**	Config Command Console
Console Menu		
Menu Command Console	**MCC**	Menu Command Console
ControlFile and I/O		
ControlFile Dump	**CFD** ***node#*** *or* ***CP#***	Cont File Dump:
ControlFile Dump or Node Dump	**ND**	Node Dump:
ControlFile and I/O (continued)		

(continued on next page)

SMS Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
ControlFile Links	**CFL**	Cont File Links:
ControlFile Status	**CFS**	Cont File Status:
Field I/O Status	**FS**	Field I/O Status:
Transmitter Maint Log	**TL**	Trans Log:
Diagnostics		
Background Diagnostics	**MCT**	Menu Confidence Tests
Memory View	**MV**	Memory View
Diogenes Interface		
Diogenes I/F Configuration	**CDI**	Configure Diogenes I/F:
Disk and File		
Calls up specified file	**DDF**	Disk Dir File:
Configure Download Script	**CDS**	Cfg Dnld Script:
Disk Activity (last screen)	**DA**	Disk Activity
Disk Directory	**DD:**	Disk Dir:
Disk Directory PeerWay	**DDP**	Disk Dir PeerWay
Disk Folder Configuration	**DC:**	Disk Configure:
HIA		
Configure HIA	**CH**	Config HIA:
Login		
User Name Detail	**UND**	User Name Detail
Plant Unit Access	**PUA**	Plant Unit Access
User Profile Directory	**UPD**	User Profile Directory
Login	**CUP**	Change User Password

(continued on next page)

SMS Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
Login (continued)		
Login	**LI**	Log In:
Log out	**LO**	Log Out:
Message Pairs		
STD Message Pairs	**MPS**	Message Pairs Std
User Message Pairs	**MPU**	Message Pairs User
Operating Displays		
Group	**G:**	Group:
Group Directory	**GD:**	Group Directory:
Overview	**O:**	Overview:
Overview Directory	**OD:**	Overview Directory:
Unit	**U:**	Unit:
Unit Directory	**UD:**	Unit Directory:
Operator Log		
Operator Change Log	**OL**	Operator Log
PeerWay and Plant		
PeerWay Node	**PN**	PeerWay Node:
PeerWay Overview	**PO**	PeerWay Overview:
PeerWay Performance	**PP**	PeerWay Performance
Plant Status	**PS**	Plant Status:
Plant Unit Configuration	**PUC**	Plant Unit Configuration
Plant Configuration Backup		
Backup Plant Config	**BUP**	BackUp Plant Config

(continued on next page)

SMS Commands that Call Up Screens (continued)

Screen Called Up	Entry	Command Line Text
PLC		
Configure PLC	**CP**	Configure PLC:
RFI		
Configure RFI	**CR:**	Configure RFI:
RNI		
Configure RNI	**CRN**	Configure RNI:
RNI Status	**CRN (EXCH)**	Configure RNI:
SCI		
Configure SCI	**CS**	Configure SCI:
Tuning Displays		
Tuning Display	**TD:**	Tuning Display
Tuning Display Directory	**TDD**	Tuning Display Directory
VAX		
Host Mode	**HM**	Hostmode:
VAX RPQNA Status Display	**VS**	VAX Status:

Alphabetical List of SMS Screen-Callup Commands

Alphabetical List of SMS Screen-Callup Commands

Entry	Command Line Text	Screen Called Up
AAA	Actv Area Alms:	Plant Area (ACTV)
AAC	Alarm Annunciation Cnf:	Alarm Annunciation
AAC	Alm Annunciation Cnf:	Alarm Annunciation
AAL	Alarm Active List	Active Process Alarms
AC	Alarm Configuration	Alarm Configuration
ADAA	ADV Actv Alrms	Advisory Active Alarms
ADCA	ADV Clrd Alrms	Advisory Cleared Alarms
ALC	Alarm List Configuration	Alarm List Configuration
ALD *node#, filename*	Alm Log Disp:	Alarm Log Display (for remote console or archived files)
ALS *LDC filename*	Alm Log Search:	Alarm Log Display (for local console only)
AN:	Area Name:	Plant Area (ACTV)
ANC	Area Name Config:	Area Name Configuration
ANG	Area Name Graphic:	Graphic for specified area
APB	Abort Plant Config Backup	(??)
ASA	Alarm Suppressed Actv	Active Suppressed Alarms
ASC	Alarm Suppressed Clrd	Cleared Suppressed Alarms
BAAA	BATCH Active Alarms	Active Batch Alarms
BACA	BATCH Cleared Alarms	Cleared Batch Alarms
BD	Block Diagram:	Continuous Diagram
BF	Block Faceplate:	Continuous Faceplate

(continued on next page)

Alphabetical List of SMS Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
BL	Block Links:	Continuous Links
BP	Block Print	Block Print Setup
BR	Block References:	Block References
CAA	Cleared Area Alms:	Plant Area (CLRD)
CAB	Cnfg Alarm Broadcast	Configure Alarm Broadcast
CBD:	Cont Block Directory:	Block Directory
CBS	Cont Block Status:	Block Status
CCB	Config Callup Buttons	Callup Buttons
CCC	Config Command Console	Console Configuration
CCP	Config Color Palette	Master Color Selection
CCU	Config Color Usage	Color Configuration
CDB	Config Display Buttons:	Display Buttons
CDI	Configure Diogenes I/F:	Diogenes I/F Configuration
CDS	Cfg Dnld Script:	Configure Download Script
CFD ***node#*** *or* ***CP#***	Cont File Dump:	ControlFile Dump
CFL	Cont File Links:	ControlFile Links
CFS	Cont File Status:	ControlFile Status
CH	Config HIA:	Configure HIA
CP	Configure PLC:	Configure PLC
CR:	Configure RFI:	Configure RFI
CRAA	CRIT Actv Alarms	Critical Active Alarms
CRCA	CRIT Clrd Alarms	Critical Cleared Alarms
CRN	Configure RNI	Configure RNI

(continued on next page)

Alphabetical List of SMS Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
CRN [EXCH]	RNI Status	RNI Status
CS	Configure SCI:	Configure SCI
CUB:	Call Up Button:	Calls up screen or performs the operation that is assigned to a callup button.
CUP	Change User Password	Login
DA	Disk Activity	Disk Activity (last screen)
DC:	Disk Configure:	Disk Folder Configuration
DCB	Disable Console Config Backup	(??)
DD:	Disk Dir:	Disk Directory
DDF	Disk Dir File:	Calls up specified file
DDP	Disk Dir PeerWay	Disk Directory PeerWay
DEL	Disk Event List	Disk Event List
DPB	Disable Plant Config Backup	(??)
ECB	Enable Console Config Backup	(??)
EL:	Event List:	Event List
ELC	Event List Configuration	Event List Configuration
EPB	Enable Plant Config Backup	(??)
FS	Field I/O Status:	Field I/O Status
FT	File Transfer	RBL File Transfer
G:	Group:	Group
GD:	Group Directory:	Group Directory
HA	Hardware Active Alarms	Active Hardware Alarms
HC	Hardware Cleared Alarms	Cleared Hardware Alarms

(continued on next page)

Alphabetical List of SMS Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
HM	Hostmode:	Host Mode
LDC	Log Display Config:	Log Display Configuration
LI	Log in:	Login
LO	Log out:	Log out
MCC	Menu Command Console	Menu Command Console
MCT	Menu Confidence Tests	Background Diagnostics
MHC	Material History Config:	Material History Configure
MPS	Message Pairs Std	STD Message Pairs
MPU	Message Pairs User	User Message Pairs
MV	Memory View	Memory View
ND	Node Dump:	ControlFile Dump or Node Dump
O:	Overview:	Overview
OD:	Overview Directory:	Overview Directory
OL	Operator Log	Operator Change Log
PA	Process Active Alarms	Active Process Alarms
PC	Process Cleared Alarms	Cleared Process Alarms
PN	PeerWay Node:	PeerWay Node
PO	PeerWay Overview:	PeerWay Overview
PP	PeerWay Performance	PeerWay Performance
PS	Plant Status:	Plant Status
PUA	Plant Unit Access	Plant Unit Access
PUC	Plant Unit Configuration	Plant Unit Configuration
RBLD	RBL Directory:	RBL Directory (RBL)

(continued on next page)

Alphabetical List of SMS Screen-Callup Commands (continued)

Entry	Command Line Text	Screen Called Up
RBLF	RBL File Contents:	RBL File Contents (RBL)
RBLM	RBLC Monitor	RBLC Monitor
RBLS	RBLC Script:	RBLC Script
SSA	System Status Active	Active System Status
SSC	System Status Cleared	Cleared System Status
TA	Tag/Address:	Group Display screen or Continuous Faceplate
TD:	Tuning Display	Tuning Display
TDD	Tuning Display Directory	Tuning Display Directory
TL	Trans Log:	Transmitter Maint Log
U:	Unit:	Unit
UD:	Unit Directory:	Unit Directory
UND	User Name Detail	User Name Detail
UPD	User Profile Directory	User Profile Directory
VS	VAX Status:	VAX RPQNA Status Display

SMS Commands that Perform Operations

SMS Commands That Perform Operations

Operation Performed	Entry	Command Line Text
Alarms and Events		
Ack all alarms on a screen	**SA**	Screen Acknowledge
Alarm area—disable	**AD:**	Area DIsable:
Alarm area—enable	**AE:**	Area Enable:
Alarm banner mode	**ASD**	Alarm Summary Disable
Alarm summary mode	**ASE**	Alarm Summary Enable:
Alarm logging—disable	**DLA**	Disable Logging Alarms:
Alarm logging—enable	**ELA**	Enable Logging Alarms:
Alarm regeneration—initiate	**AR**	Alarm Regen:
Silence console horn	**SH**	Silence Horn
Toggles alarm view	**AVT**	Alarm View Toggle
Alarm and Event Lists		
Active Process Alarms List—print Cleared Process Alarms List—print	**P**[sp]**PA** *or* **P**[sp]**AA**	Pr Process Alarm List Pr Alarm Active List
Act Sys Status Alarms List—print Clrd Sys Status Alarms List—print	**P**[sp]**SSA**	Pr System Status Active
Active Hardware Alarm List —print Cleared Hardware Alarm List—print	**P**[sp]**HA**	Pr Hardware Alarm List
Active Batch Alarms List—print Cleared Batch Alarms List—print	**P**[sp]**BA**	Pr BATCH Alarm List
Disk Event List—print	**P**[sp]**DEL**	Pr Disk Event List
Event List—print	**P**[sp]**EL:**	Pr Event List:
Console		
Abort backup	**AB**	Abort Backup:
Abort tape restore	**ATR**	Abort Tape Restore:

(continued on next page)

SMS Commands That Perform Operations (continued)

Operation Performed	Entry	Command Line Text
Disable console config backup	**DCB**	Disable Console Cfg Backup
Enable console config backup	**ECB**	Enable Console Cfg Backup
Eject a floppy disk	**EF**	Eject Floppy:
Reboot the console	**RBC**	Re-Boot Console
Shut down operation of disk	**DS:**	Disk Shutdown
Console Screens		
Mark—remove	**RM**	Remove Mark:
Next page—call up	**PF**	Page Forward
Previous page—call up	**PB**	Page Backward
Recall the previous screen	**R:**	Recall:
ControlFile		
Kill operation of Controller Proc	**KC**	Kill Controller:
Nonvolatile memory—eliminate areas of nonvolatile memory reserved for non-existent blocks	**FB**	Free Bubble
Nonvolatile memory—initialize configuration information	**IC**	Init CF Nvmem Config:
Wipe bubble memory	**WB**	Wipe Bubble:
Diagnostics		
Stop memory dump before it finishes	**KMD**	Kill Memory Dump
Captures RAM memory of a node in a file	**MD** *node#* , *filename*	Memory Dump:
Mask		
Toggle between masking and no masking	**TTM**:	Toggle Tag Mask:
Operator Log		
Operator Log—print	**P**[sp]**OL**	Pr Operator Log

(continued on next page)

SMS Commands That Perform Operations (continued)

Operation Performed	Entry	Command Line Text
Plant Configuration		
Abort plant config backup	**DPB**	Abort Plant Cfg Backup
Disable console config backup	**DPB**	Disable Plant Cfg Backup
Enable console config backup	**EPB**	Enable Plant Cfg Backup
Screen print		
Character print of screen	**SC**	Screen Char Print:
Configured mode print of screen	**SP**	Screen Print:
Graphic print of screen	**SG**	Screen Graph Print:
Transmitter Log		
Trans Maintenance Log—disable	**DML**	Disable Maint Log:
Trans Maintenance Log—enable	**EML**	Enable Maint Log:

Alphabetical List of SMS Operation Commands

Alphabetical List of SMS Operation Commands

Entry	Command Line Text	Operation Performed
AB	Abort Backup	
AD:	Area Disable:	Alarm area—disable
AE:	Area Enable:	Alarm area—enable
APB	Abort Plant Config Backup	Abort plant configuration backup
AR	Alarm Regen:	Alarm regeneration—initiate
ASD	Alarm Summary Disable	Alarm banner mode
ASE	Alarm Summary Enable:	Alarm summary mode
ATR	Abort Tape Restore	
AVT	Alarm View Toggle	Toggles alarm view
DCB	Disable Console Config Backup	Disable console configuration backup
DLA	Disable Logging Alarms:	Alarm logging—disable
DML	Disable Maint Log:	Trans Maintenance Log—disable
DPB	Disable Plant Config Backup	Disable plant configuration backup
DS:	Disk Shutdown	Shut down operation of disk
ECB	Enable Console Config Backup	Enable console configuration backup
EF	Eject Floppy:	Eject a floppy disk
ELA	Enable Logging Alarms:	Alarm logging—enable
EML	Enable Maint Log:	Trans Maintenance Log—enable
EPB	Enable Plant Config Backup	Enable plant configuration backup

(continued on next page)

Alphabetical List of SMS Operation Commands (continued)

Entry	Command Line Text	Operation Performed
FB	Free Bubble	Nonvolatile memory—eliminate areas of nonvolatile memory reserved for non-existent blocks
IC	Init CF Nvmem Config:	Nonvolatile memory—initialize configuration information
KC	Kill Controller:	Kill operation of Controller Proc
KMD	Kill Memory Dump	Stop memory dump before it finishes
MD *node#*, *filename*	Memory Dump:	Captures RAM memory of a node in a file
PB	Page Backward	Previous page—call up
PF	Page Forward	Next page—call up
P[sp]**AA**	Pr Alarm Active List	Active Process Alarms List—print Cleared Process Alarms List—print
P[sp]**BA**	Pr BATCH Alarm List	Active Batch Alarms List—print Cleared Batch Alarms List—print
P[sp]**HA**	Pr Hardware Alarm List	Active Hardware Alarm List —print Cleared Hardware Alarm List—print
P[sp]**OL**	Pr Operator Log	Operator Log—print
P[sp]**PA**	Pr Process Alarm List	Active Process Alarms List—print Cleared Process Alarms List—print
P[sp]**DEL**	Pr Disk Event List	Disk Event List—Print
P[sp]**EL:**	Pr Event List:	Event List—Print
P[sp]**SSA**	Pr System Status Active	Act Sys Status Alarms List—print Clrd Sys Status Alarms List—print
R:	Recall:	Recall the previous screen
RBC	Re-Boot Console	Reboot the console
RM	Remove Mark:	Mark—remove
SA	Screen Acknowledge	Ack all alarms on a screen

(continued on next page)

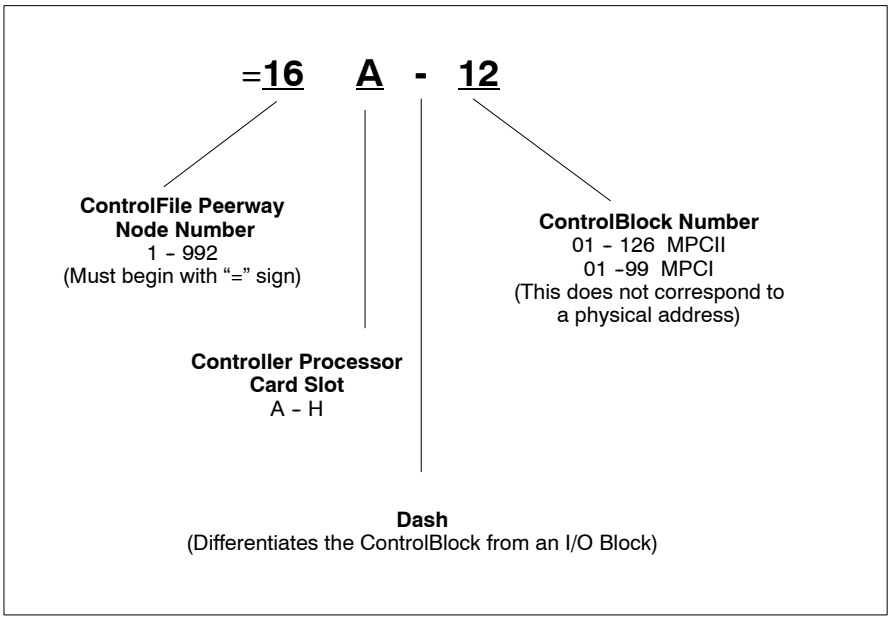
Alphabetical List of SMS Operation Commands (continued)

Entry	Command Line Text	Operation Performed
SC	Screen Char Print:	Character print of screen
SG	Screen Graph Print:	Graphic print of screen
SH	Silence Horn	Silence console horn
SP	Screen Print:	Configured mode print of screen
TTM	Toggle Tag Mask	
WB	Wipe Bubble:	Wipe bubble memory

Section 7:
Addressing

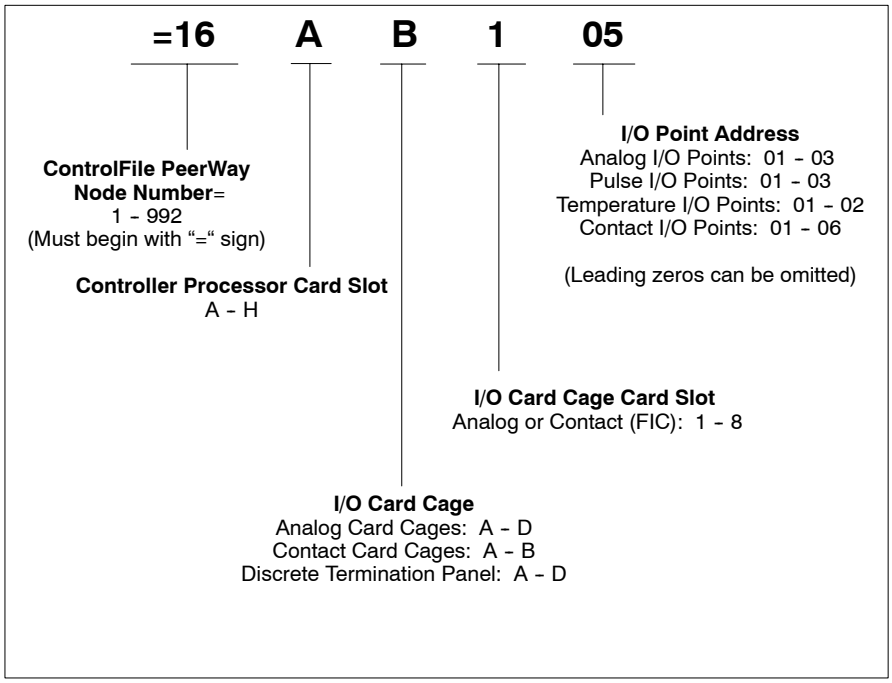
This section describes **RS3** addressing. For more information about addressing, see OV.

ControlBlock Addressing



ControlBlock Addressing Structure

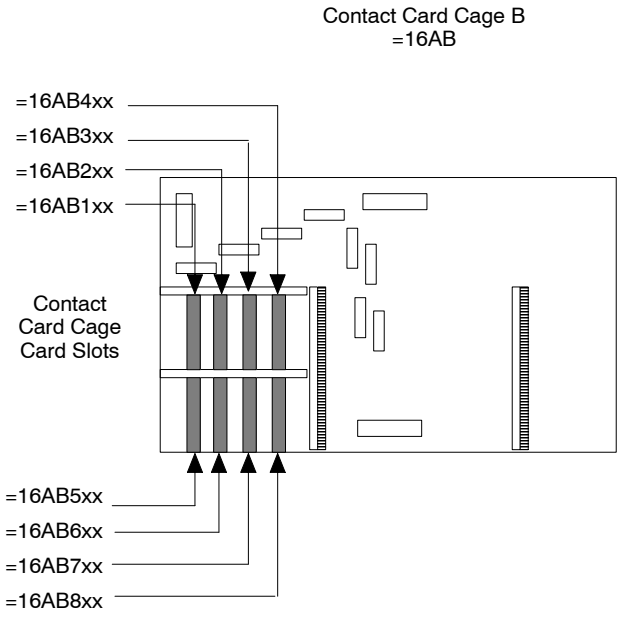
Analog and Contact I/O Addressing in a Card Cage



FlexTerm I/O Point Addressing Structure

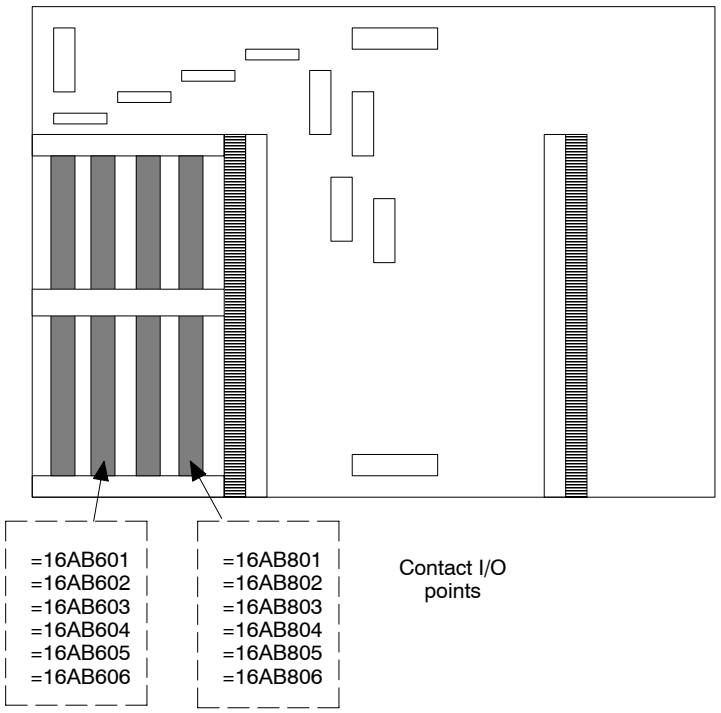
Pages 7–4 through 7–8 show examples of analog and contact I/O addressing in a Card Cage.

Contact Card Cage Slot Addressing

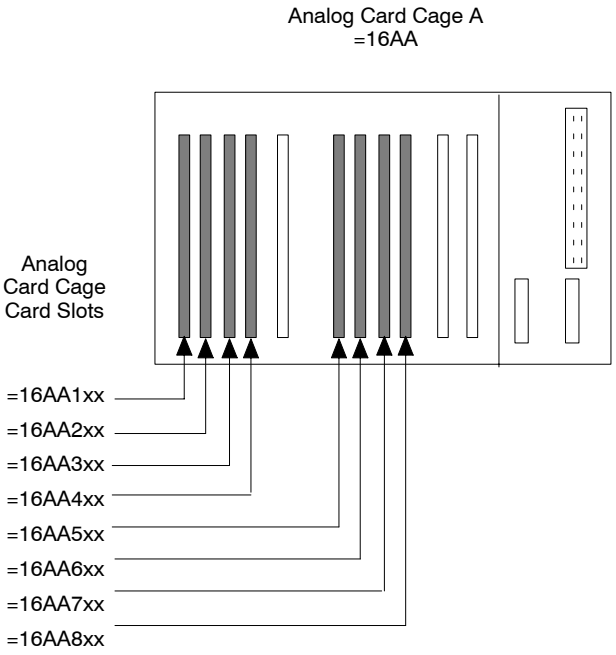


Contact FIC I/O Point Addressing

Contact Field Interface Card (FIC) Slots
=16AB1 through =16AB8

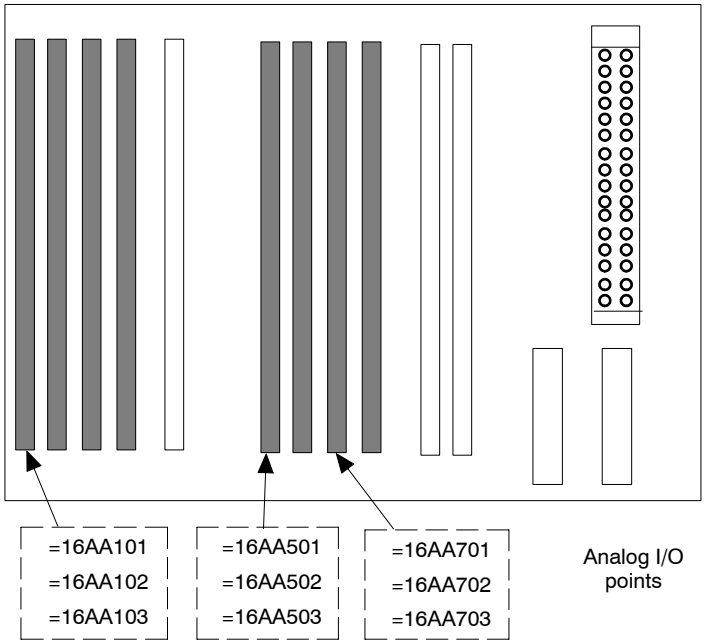


Analog Card Cage Slot Addressing



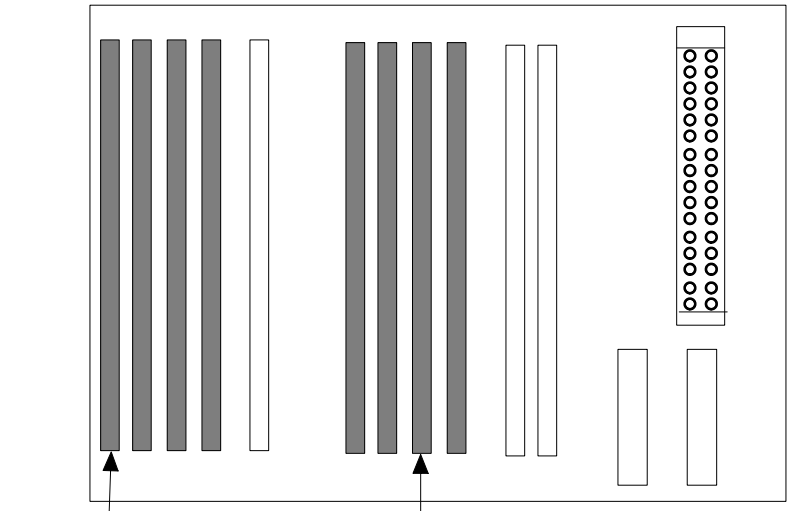
Analog FIC I/O Point Addressing

Analog Field Interface (FIC) Card Slots
=16AA1 through =16AA8



Pulse FIC I/O Point Addressing

Pulse Field Interface Card (FIC) Slots
=16AA1 through =16AA8

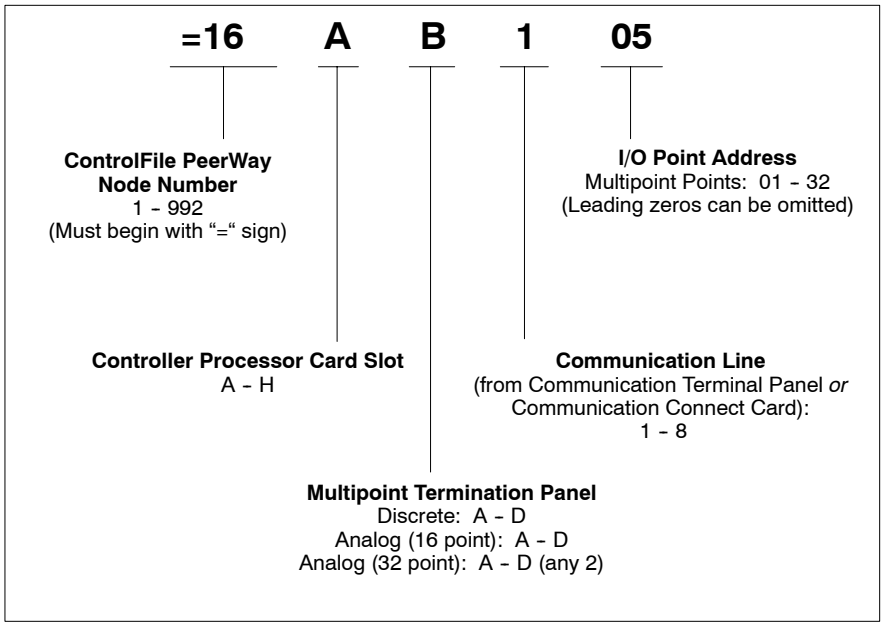


Pulse I/O points

=16AA101 Pulse input or output
=16AA102 Pulse input or output
=16AA103 Pulse mA output

=16AA701 Pulse input or output
=16AA702 Pulse input or output
=16AA703 Pulse mA output

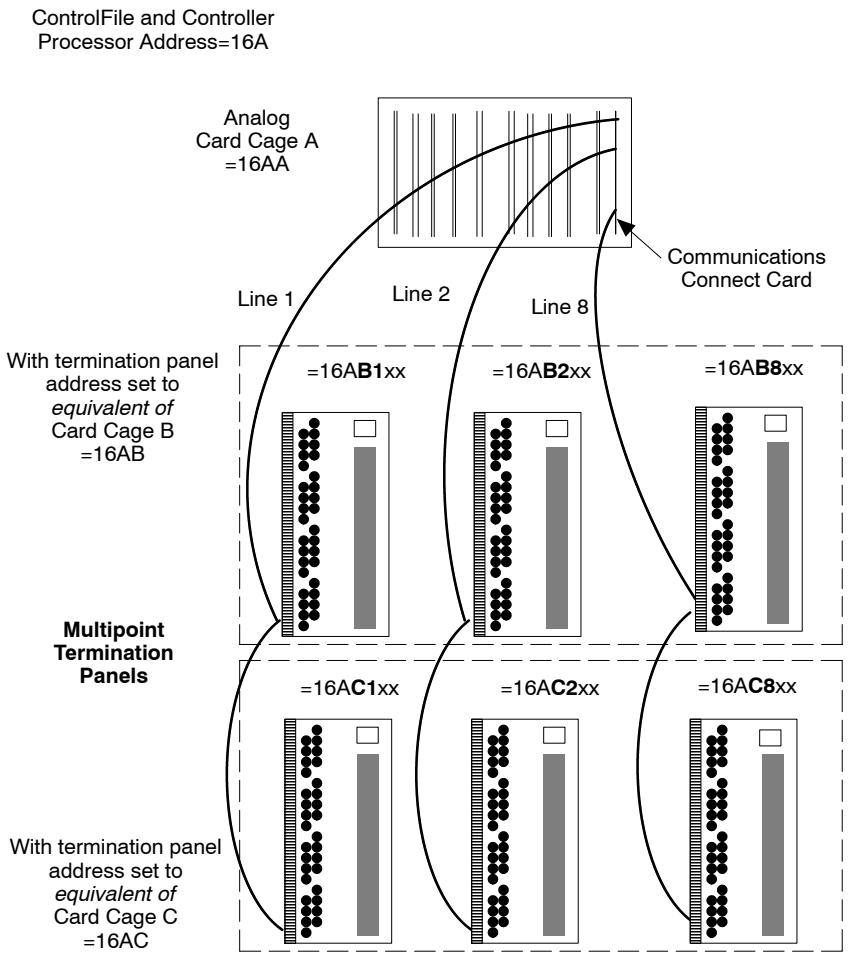
Multipoint I/O Addressing



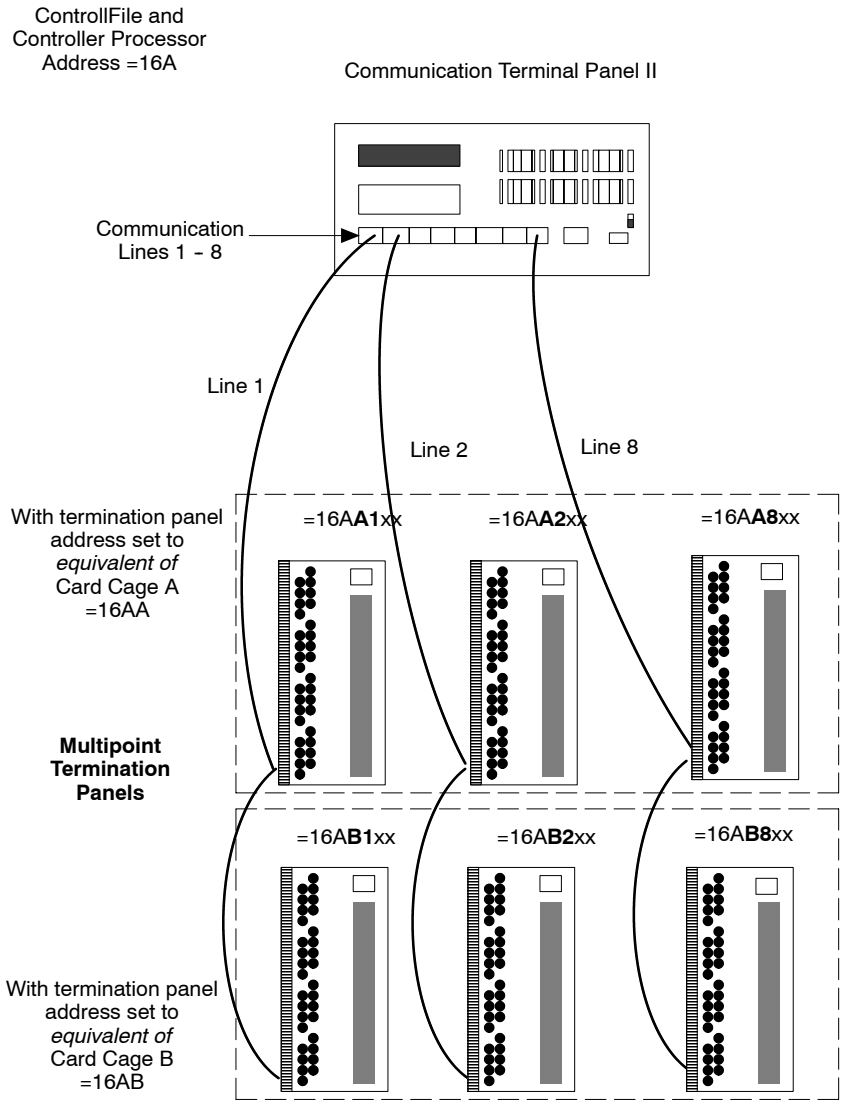
Multipoint I/O Addressing

Pages 7–10 thought 7–16 illustrate the configuration possibilities.

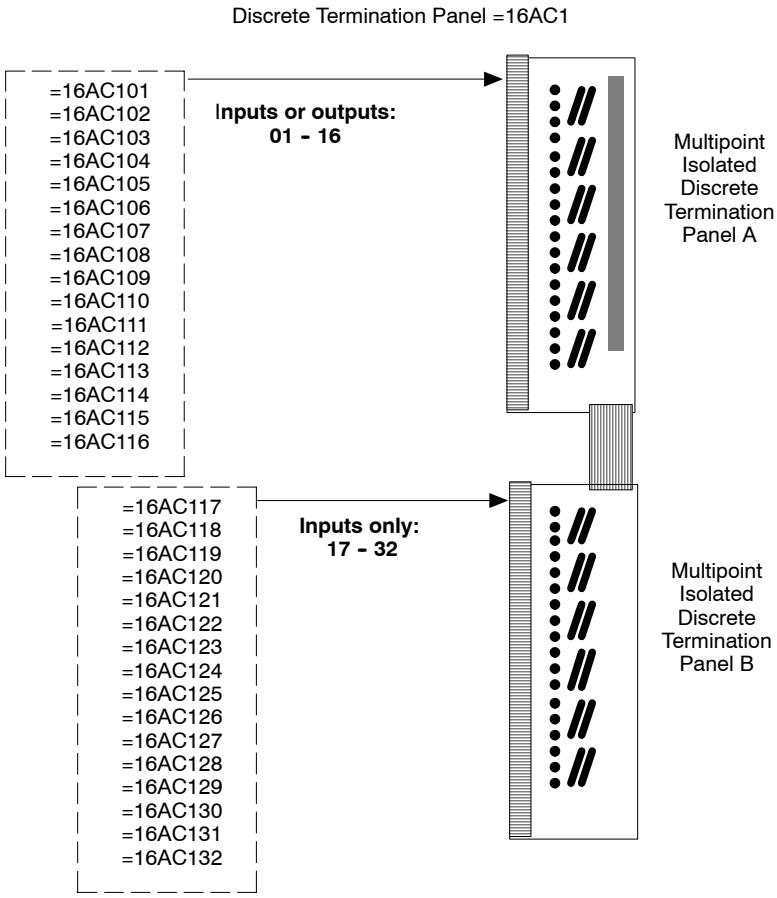
Communication Connect Card Addressing



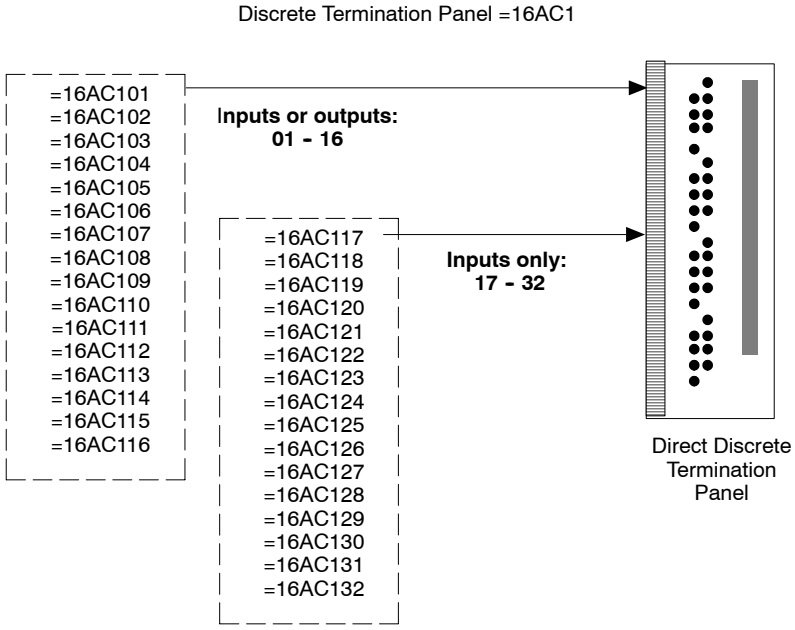
Communication Terminal Panel II Addressing



Multipoint Isolated Discrete I/O Point Addressing

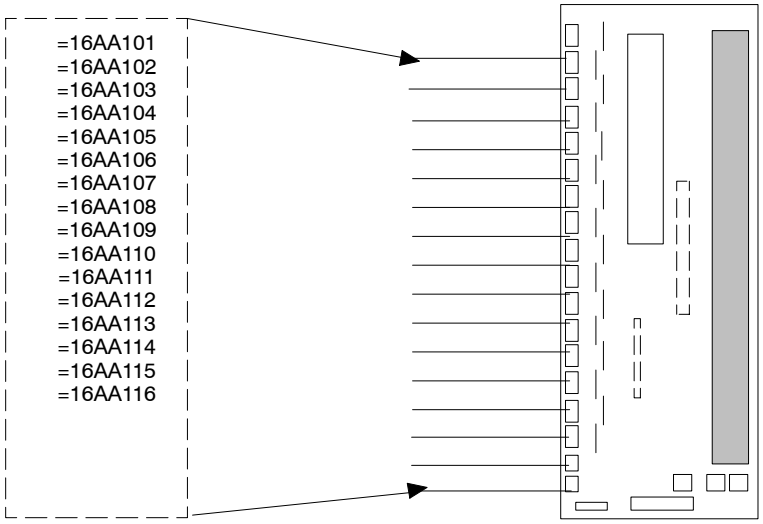


Multipoint Direct Discrete I/O Point Addressing



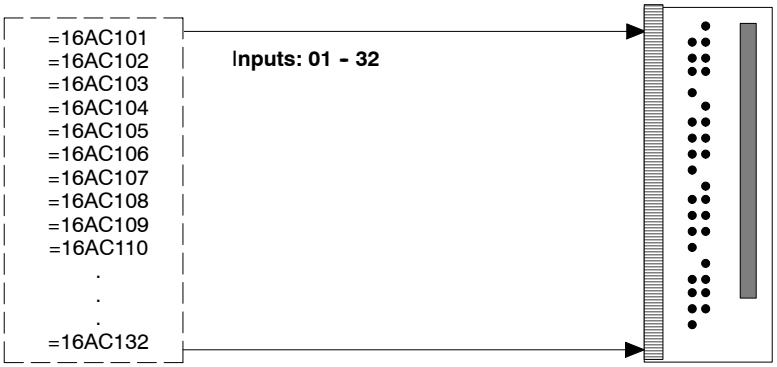
Multipoint Analog I/O Point Addressing (16 point)

The points on 16–point analog Field Interface Modules (FIMs) are either all inputs or all outputs, depending on the FIM type.

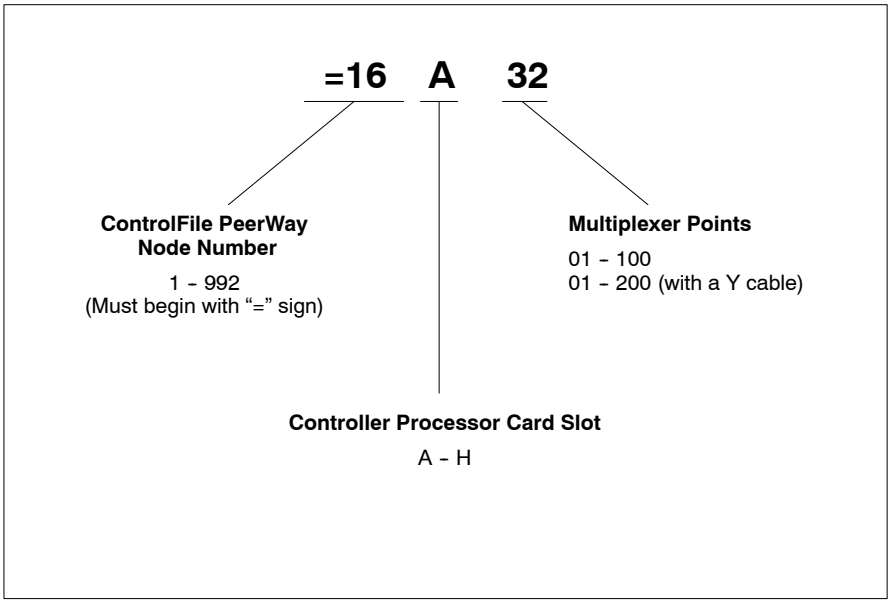


Multipoint Analog Input Point Addressing (32 point)

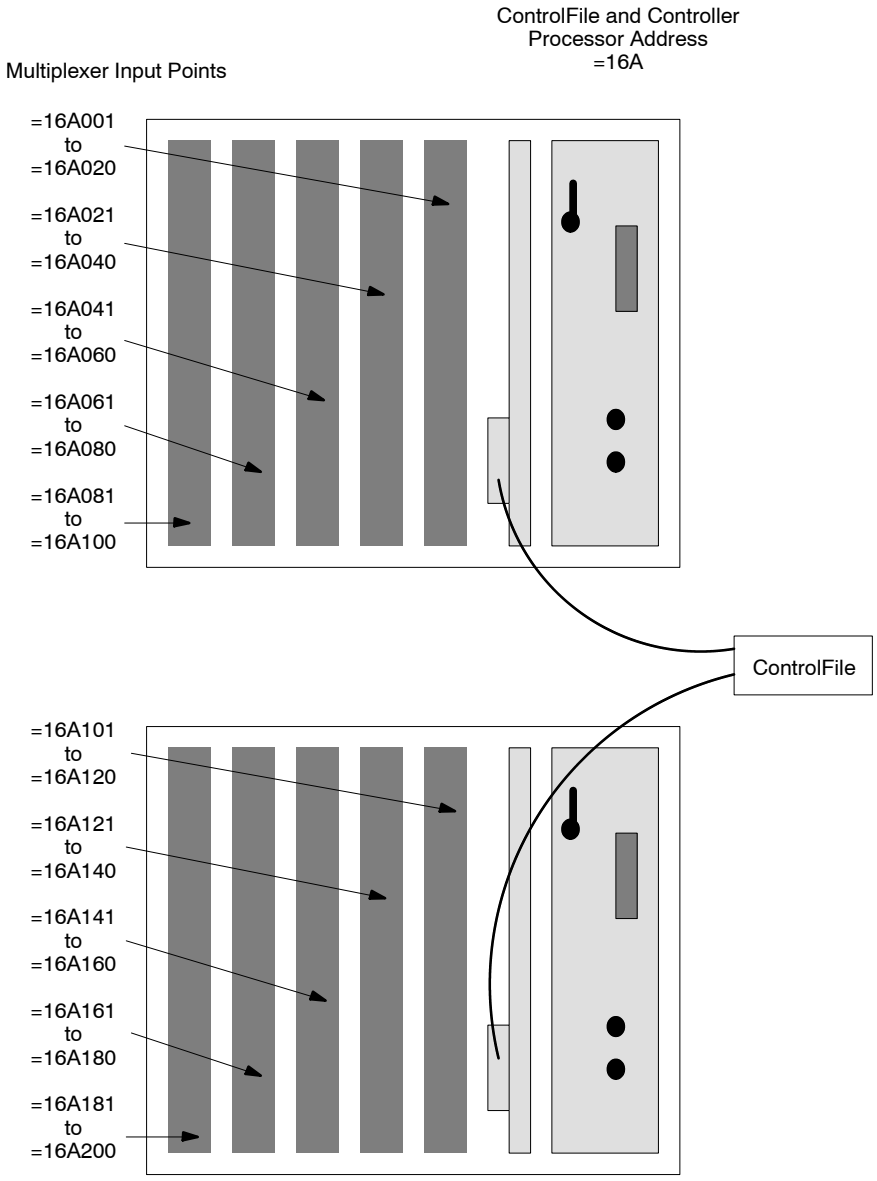
32–point multipoint analog Field Interface Modules (FIMs) have all input points.



Multiplexer FEM I/O Point Addressing

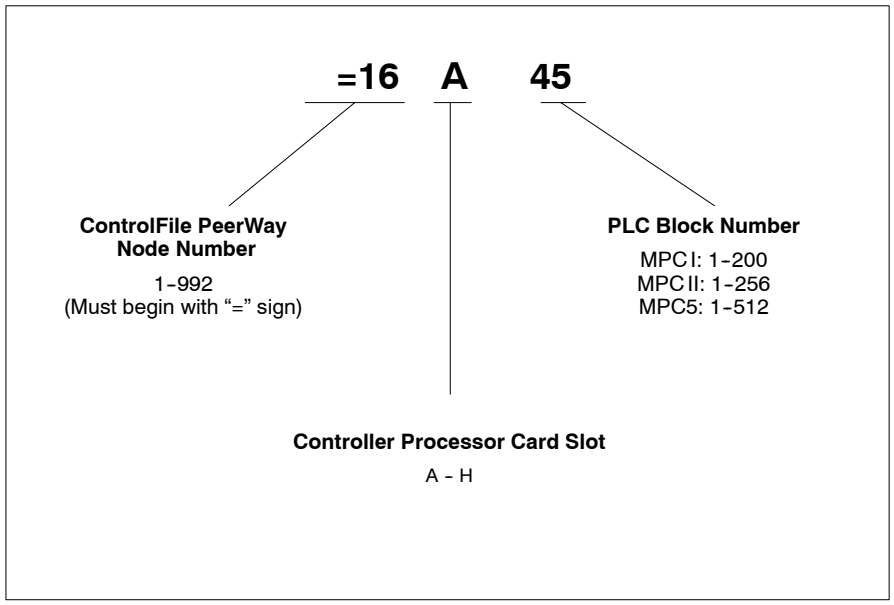


Multiplexer Addressing


ControlFile and Controller
Processor Address
=16A
Multiplexer Input Points
=16A001
to
=16A020
=16A021
to
=16A040
=16A041
to
=16A060
=16A061
to
=16A080
=16A081
to
=16A100
ControlFile
=16A101
to
=16A120
=16A121
to
=16A140
=16A141
to
=16A160
=16A161
to
=16A180
=16A181
to
=16A200

PLC FlexTerm Block Addressing

PLC blocks exist only logically within the Controller Processor. No physical relationship exists.



PLC FlexTerm Addressing

Section 8:
Password Security

User Profile Directory

The User Profile Directory screen allows a system manager to view all users that have been defined in the system and to move to the User Name Detail screen or Plant Unit Access screen for a selected user. This screen can be viewed only by a system manager who is logged in.

❑ **To access the User Profile Directory:**

1. On the command line, type

   UPD [ENTER]

2. To view additional users in the User Profile Directory, press [PAGE FORWARD] or [PAGE BACKWARD].

❑ **To move to the User Name Detail screen from the User Profile Directory:**

1. Locate the user whose User Name Detail screen you want to view, and position the cursor anywhere between the user number and the access level fields.
2. Press [SELECT].

❑ **To move to the Plant Unit Access screen from the User Profile Directory:**

1. Locate the user whose plant unit access screen you want to view, and position the cursor anywhere on the PUA field.
2. Press [SELECT].

Plant Unit Access Screen

The Plant Unit Access screen allows a system manager to create the files that define which plant units a user can operate. The plant-unit access (PUA) information is stored in the $$PASSWD file. The same filename can be assigned to multiple users; however, no more than 991 files (one for each user) can be defined.

NOTE:

- A PUA filename must have been configured on the Plant Unit Access screen before you can enter it on the User Name Detail screen. Create the necessary PUA filenames before you add users to the password system.
- Plant unit ownership must be configured on both the Plant Unit Configuration and Plant Unit Access screens.

❑ **To define a new plant-unit access file:**

1. With system manager access, call up the Plant Unit Access screen by typing

   PUA [ENTER]

2. Cursor to the ‘PUA filename’ field, enter a valid filename, and press [ENTER].
3. Configure the plant units that users with this PUA filename can access by doing one of the following:
   - Cursor to each plant unit that this user should be able to access and press [ENTER].
   - Cursor to the ‘Set range’ field, enter two numbers separated by a comma, and then press ENTER].
   - Clear an enabled field by pressing [ENTER] again on a plant unit number or by entering values in the ‘Clear range’ field.
4. To save the $$PASSWD file after you enter appropriate values in all necessary fields, type

   [Ctrl] **W**

Plant Unit Access Screen (continued)

Plant unit access definitions cannot be deleted, but the filename and plant unit assignments can be changed.

❑ **To change information in an existing Plant Unit Access file:**

1. With system manager access, call up the Plant Unit Access screen and do one of the following:
   - On the command line, type

     PUA ***filename*** [ENTER]
   - In the User Profile Directory, cursor to the name of the file you want to change and press [SELECT].
2. To clear any previously enabled plant units, do one of the following:
   - Cursor to a plant unit marked with an asterisk that you do not want this user to access and press [ENTER].
   - Cursor to the “Clear range” field, enter a single number or two numbers separated by a comma, and then press ENTER].
3. To enable the appropriate fields, do one of the following:
   - Cursor to each plant unit that this user should be able to access and press [ENTER].
   - Cursor to the “Set range” field, enter a single number or two numbers separated by a comma, and then press ENTER].
4. To cancel changes and recall the original plant unit access configuration, type

   [Ctrl] **R**
5. To save the $$PASSWD file after you enter appropriate values in all necessary fields, type

   [Ctrl] **W**

User Name Detail Screen

The User Name Detail screen allows a system manager to add users to the password management system. This screen can be edited only when a system manager is logged in, but it can be viewed by a configuror. The information on this screen is stored in the $$PASSWD file in the Console Configuration folder.

❑ **To add a new user to the system:**

1. With system manager access, call up the User Name Detail screen by typing

   UND [ENTER]

   The User Name Detail screen and the next available user number is automatically assigned.
2. Cursor to the “User Name” field and enter up to 32 characters.
3. Cursor to the “Login Name” field and enter between 4 and 16 characters that this user can use to log in.
4. Cursor to the “Access” field, press [NEXT OPTION] until the appropriate access level appears, and then press [ENTER].
5. Cursor to the “PUA Filename” field, and enter up to 9 characters that define the name of a file containing this user’s plant-unit access privileges.
6. Cursor to other fields that you want to define and enter appropriate values as listed in the table on the following page.
7. To save the $$PASSWD file after you enter appropriate values in all necessary fields, type

   [Ctrl] **W**

User Name Detail Screen (continued)

❑ **To change information on the User Detail screen:**

1. With system manager access, call up the User Name Detail screen and do one of the following:
   - On the command line, type

     UND ***login_name*** [ENTER]
   - In the User Profile Directory, cursor to the name of the user whose profile you want to change and press [SELECT].
2. Cursor to the fields in the table and enter the appropriate values for fields that you want to modify.
3. To remove changes and recall the original user profile, type

   [Ctrl] **R**
4. To save the $$PASSWD file after you enter appropriate values in all necessary fields, type

   [Ctrl] **W**

User Name Detail Screen (continued)

User Name Detail Screen

Field	Allowable Entries *
Number	1 to 991
User Name	Up to 32 alphanumeric characters (spaces are allowed)
Login Name	4 to 16 alphanumeric characters (spaces allowed)
Access	NONE, OPER, SUPER, RCPM, CONF, SYSMGR
PUA Filename	Up to nine alphanumeric characters that represent an existing file.
Login/Logout Times	
Days	Mon, Tue, Wed, Thu, Fri, Sat, Sun, **any**
Start/End Time	00:00 to 23:59
Inactive User Logout	00:01 to 23:59
Password	Canceled, Cleared
Last Change	Display only
Next Change	Display only
GLOBAL SYSTEM PARAMETERS	
Password	
Change Required Every	**0** to 365 days
Change not allowed for	**0** to 365 days
Inactive User Logout	0:01 to 23:59
NEW SYSMGR	See “Changing Configuror to System Manager” on page 8-8 for additional information.

* Bold indicates the default value.

Changing a Configuror to a System Manager

If an existing system manager is unavailable, you can change a configuror to a system manager under the following conditions:

- The configuror is currently logged in.
- Another configuror or supervisor is available to approve the change.

Such a change is only valid while the configuror is logged in, unless the configuror changes the access level on his or her user name detail screen and saves that configuration to the $$PASSWD file to make the change permanent.

❑ **To change a configuror to a system manager:**

1. As a configuror, log into the system and call up the User Name Detail screen by typing

   UND ***(your_login_ name)*** [ENTER]

   NOTE: You can raise yourself to a system manager from any User Name Detail screen.

2. With another configuror or supervisor immediately available, cursor to the “Raise to SYSMGR: (press ENTER)” field and press [ENTER].
3. Cursor to the “Additional User” field and allow the other configuror or supervisor to enter his or her name.
4. Cursor to the “Password” field and allow the other configuror or supervisor to enter his or her password.

   The configuror who was originally logged in now has the access level and privileges of a system manager until he or she logs off.

Changing a Configuror to a System Manager (continued)

❑ **To make the system manager access level permanent:**

1. Cursor to the “Access Level” field on the User Name Detail screen for the new system manager.
2. Press [NEXT OPTION] until SYSMGR appears and then press [ENTER].
3. Cursor to any other fields that need updating and make the necessary changes.
4. To save the $$PASSWD file after updating the appropriate fields, type

   [Ctrl] **W**

   The configuror who was originally logged in now has permanent system manager status.

NOTE: If you move to the User Name Detail screen for the configuror who has been raised to a SYSMGR, the access level is highlighted in yellow. The access level for such a configuror is also highlighted in yellow on the User Profile Directory.

FISHER-ROSEMOUNT

RS3™
Configuration Quick Reference Guide

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W