This manual describes the DCL level commands provided by the MDS-Plus system.

Revision/Update Information: This is a new manual.
Operating System and Version: VMS Version 5.0 or higher
Software Version: MDSplus Version 1.0 Field Test
February 2, 1993

The information in this document is subject to change without notice and should not be construed as a commitment by the Massachusetts Institute of Technology, Los Alamos National Laboratory or Istituto Gas Ionizzati. The authors assume no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No guarantee is made that the software is adequately or completely described in, or behaves in accordance with this document.

Copyright © 1989 by Massachusetts Institute of Technology

All Rights Reserved. Printed in U.S.A.

The following are trademarks of Digital Equipment Corporation:

<table>
<thead>
<tr>
<th>Product</th>
<th>DEC/MMS</th>
<th>VAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC</td>
<td>DECnet</td>
<td>VMS</td>
</tr>
<tr>
<td>DECwindows</td>
<td>MicroVAX</td>
<td></td>
</tr>
<tr>
<td>DEC/CMS</td>
<td>Rdb/VMS</td>
<td></td>
</tr>
</tbody>
</table>
This part contains reference pages for all of the MDS-Plus commands available at DCL. Detailed descriptions of the commands can be found in the individual manuals for the utilities described.
ACTIONS

Program to look at and/or modify the actions in a tree.

FORMAT

MCR ACTIONS

-TREE treename
-SHOT shot-number

Command options

-TREE treename
-SHOT shot-number

Defaults

-TREE cmod
-SHOT -1

restrictions

None.

OPTIONS

-TREE treename

Specifies the name of the TREE whose actions you wish to manipulate.

-SHOT

Specifies the shot number of the TREE whose actions you wish to manipulate. This will almost always be -1 which is the model.

DESCRIPTION

The actions program is a convenient way to modify the actions in an experiment. When it is invoked, a window with all of the actions in the tree will appear on the screen sorted in the order they will be executed in. The details of any of the actions (sequence-number, phase, or what to do) can be modified by clicking on the name of the action in the list. A Refresh must be done for the ordering of the list to reflect changes to individual actions.

EXAMPLES

1. $ MCR ACTIONS -TREE CMOD -SHOT -1

   This command invokes the actions program on the model of the CMOD experiment. See Figure 1 for a typical display.

2. Clicking the mouse on one of the actions

   After clicking the mouse on one of the actions a dialog as in Figure 2 will appear. The details of the action can then be modified by typing in the boxes an clicking on either Ok or Apply.
**Figure 1** Actions display for CMOD experiment.
Figure 2  Action detail dialog.
ACTMON

Program to monitor the execution of actions in an experiment shot cycle.

FORMAT

MCR ACTMON -MONITOR monitor-name

Command options Defaults
-MONITOR monitor-name -MONITOR cmod_monitor

restrictions

None.

OPTIONS

-MONITOR monitor-name

Specifies the name action monitor for this display to communicate. This name must match the /MONITOR qualifiers on the dispatch commands being monitored.

DESCRIPTION

ACTMON is a program for monitoring the progress of the actions in the shot cycle. You must specify the name of the monitor ACTMON should communicate with. This is the same as the monitor specified in the dispatch commands. For CMOD the default ‘CMOD_MONITOR’ will display information about the CMOD shot cycle.

When invoked it displays a paned window on the screen. The top half shows all of the actions sorted by phase with the current status of each action. The bottom half lists all of the action / dispatching events as they occur. See Figure 3 for a sample display.
Figure 3  Action Monitor Display

ACTMON

File

On

INITIALIZATION

\ENGINEERING\PS_CAMAC\EF3:DECODER:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH1:DECODER:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH2:DECODER:INIT_ACTION
\ENGINEERING\PS_CAMAC\TMX:DECODER:INIT_ACTION
\ENGINEERING\PS_CAMAC\TMX\J221_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH2\J221_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH1\J221_2:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH1\J221_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\EF3\J221_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH2\L0212_2:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH2\L0212_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\EF3\L0212_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH1\L0212_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH1\L0212_2:INIT_ACTION
\ENGINEERING\PS_CAMAC\BUS\L0212_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\BUS\A12_2:INIT_ACTION
\ENGINEERING\PS_CAMAC\TMX\L0212_2:INIT_ACTION
\ENGINEERING\PS_CAMAC\BUS\A12_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\TMX\L0212_1:INIT_ACTION
\ENGINEERING\PS_CAMAC\TMX\L0212_3:INIT_ACTION
\ENGINEERING\PS_CAMAC\OH2\L0212_3:INIT_ACTION

15:18:01  800 NEW SHOT ******************* *************** CHOD
15:18:40  800 PHASE ******************* *************** INITIALIZATION
15:18:40  800 DISPATCH DAQ3_CAMAC \ENGINEERING:
15:18:54  800 DISPATCH DAQ3_CAMAC \ENGINEERING:
15:18:54  800 DISPATCH DAQ3_CAMAC \ENGINEERING:
15:18:54  800 DISPATCH DAQ3_CAMAC \ENGINEERING:
15:18:54  800 DOING DAQ3_CAMAC \ENGINEERING:
CCL

Invokes the Camac Command Language utility. See the CAMAC Command Language Reference Manual for more information on this program.

**FORMAT**

```
CCL  [command-string]
```

<table>
<thead>
<tr>
<th>Command Qualifiers</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>/[NO]COMMAND[=file-spec]</td>
<td>/COMMAND=CCLINIT</td>
</tr>
<tr>
<td>/[NO]LIBRARY[=file-spec]</td>
<td>/NOLIBRARY</td>
</tr>
</tbody>
</table>

**restrictions**

None.

**prompts**

None.

**PARAMETERS**

`command-string`

Specifies a CCL command s trying to be executed. If you use the `command-string` parameter, CCL will return to DCL after completion of the specified command. Any valid CCL command can be used, including the invocation of a command macro or command procedure file.

**DESCRIPTION**

The CCL command invokes the Camac Command Language utility. This program allows the user to perform CAMAC I/O interactively. See the Camac Command Language Reference Manual for detailed information on the various commands available.

When the CCL utility is invoked, the command interpreter looks for the existence of auxiliary macro library files named CCL$LIBRARY, CCL$LIBRARY_1, CCL$LIBRARY_2, . . . . If these libraries exist, they are opened for read access and macros previously stored in them are available for use. See the MDS Command Language Interpreter Reference Manual for detailed information on the various commands available.

**COMMAND QUALIFIERS**

`/COMMAND[=filespec]`

Specifies a CCL command procedure file to be executed then CCL is invoked. The default directory for the file specification is SYS$LOGIN and the default file type is .CCL. If this qualifier is omitted or no file specification is is given, the utility will attempt to execute a file called CCLINIT in your SYS$LOGIN directory. If a logical name CCLINIT is defined, the file pointed to by this logical name will be executed.

The `/COMMAND` qualifier is used to define a startup file for commands which you always want to invoke when you enter CCL.
/LIBRARY=filespec
/NOLIBRARY
Specifies the name of a VMS text library to be used as the primary macro library. If a macro library is specified, the library will be opened for read and write access. You can invoke macros from this library by typing their names to the CCL prompt. New macros, or modifications to existing macros, can be saved to this library using the SAVE command.

EXAMPLES

1 $ CCL
   CCL> PIO/FUNC=8 DIGITIZER
   ...
   CCL> EXIT

   The CCL command above invokes the CCL utility. The sample CCL command sends an F=8 command to the CAMAC module called DIGITIZER.

2 $ CCL/LIBRARY=MY_CCL_LIB
   CCL> MY_MACRO
   ...
   CCL> EXIT

   The CCL command above invokes the CCL utility with the macros in the library MY_CCL_LIB added to the command set. The sample CCL command invokes the macro called MY_MACRO.
DWPAD

Program to provide a set of push buttons to hold scope panel definitions.

FORMAT

MCR DWPAD  [-DEFAULT default-file]

Command options                  Defaults
-DEFAULT default-file             -DEFAULT SYS$LOGIN:PAD_DEFAULTS.DAT

OPTIONS

-DEFAULT default-file-name

Specifies the name of a file which contains a description of the buttons for
the pad. This file can be created using the Save Current Settings or Save
Current Settings As menu options in the DWPAD program.

The directory specified in the default-file-name option becomes the default
directory for the Use Saved Settings From menu choice. This is true even
if the default-file specified can not be opened.

DESCRIPTION

DWPAD is an interactive DECWINDOWS application which provides a set
of push buttons containing scope panel definitions. These panels can then
be pasted in DWSCOPE panels.

Users can set up DWPADS which contain all, or most of, the traces they
may be interested in. They can then paste the particular traces into
scopes as required. See DWSCOPE command description in this manual
for a brief description of the DWSCOPE utility.

A typical pad display can be seen in Figure 4.

EXAMPLES

1    $ MCR DWPAD -DEF CMOD$MODELS:[ENGINEERING.SCOPES]ENG_PAD.DAT

This command will bring up DWPAD using the setup stored in
CMOD$MODELS:[ENGINEERING.SCOPES]ENG_PAD.DAT. Clicking
on one of the buttons will copy it to the clip-board for pasting into a
DWSCOPE. The display from this example can be seen in Figure 4.
Figure 4  Typical DWSPAD Display
DWSCOPE

Program to display experimental data on X-based devices.

<table>
<thead>
<tr>
<th>FORMAT</th>
<th>MCR DWSCOPE</th>
<th>-DEFAULT default-file</th>
</tr>
</thead>
</table>

**Command options**

-DEFAULT default-file

**Defaults**

-DEFAULT SYS$LOGIN:SCOPE_DEFAULTS.DAT

**OPTIONS**

-DEFAULT default-file-name

Specifies the name of a file which contains a description of the panels of the scope. This file can be created using the Save Current Settings or Save Current Settings As menu options in the scope program.

The directory specified in the default-file-name option becomes the default directory for the Use Saved Settings From menu choice. This is true even if the default-file specified can not be opened.

**DESCRIPTION**

DWSCOPE is an interactive DECWINDOWS application for viewing MDS-Plus data items. It displays the data in stacks of panels with one trace per panel.

Zooming, panning, and digitizing are controlled using the mouse. See the DWSCOPE documentation for details of its use.

Panel descriptions can be cut and pasted from panel to panel or to and from DWPAD buttons. See DWPAD command description in this manual for a brief description of the DWPAD utility.

A typical scope display can be seen in Figure 5.

**EXAMPLES**

```bash
$ MCR DWSCOPE -DEF CMOD$MODELS:[ENGINEERING.SCOPES]*.DAT
```

This command will bring up DWSCOPE with no default file and the Use Saved Settings From pointing at the [ENGINEERING.SCOPES] directory of the CMOD$MODELS device. After choosing Use Saved Settings From the screen will look something like Figure 6. After choosing one of the displays it will look like Figure 5.
Figure 5  Typical DWSCOPE Display
Figure 6  DWSCOPE Use Saved Settings From

Filter

cmod$models:[engineering,scopes]*.DAT

Directories

g.scopes[ING]

Files

[S]BUS.DAT;3
[S]COMB.DAT;1
[S]EF1_SCOPE.DAT;2
[S]EF2_SCOPE.DAT;1
[S]EF3_SCOPE.DAT;1
[S]EF4_SCOPE.DAT;3
[S]EFC_SCOPE.DAT;1
[S]ENG_OLD_PAD.DAT;1

Select Configuration File:

cmod$models:[engineering,scopes]

OK  Filter  Cancel  Help

(0,0)
LOGBOOK

LOGBOOK is a character cell terminal based interface to the experiment wide electronic Logbook.

FORMAT

@SYS$MANAGER:LOGBOOK
LOGBOOK

DESCRIPTION

LOGBOOK is a character cell terminal based interface to the experiment wide electronic Logbook. LOGBOOK is based on the TPU editor. It provides read and write access to the logbook.

The command procedure SYS$MANAGER:LOGBOOK.COM defines the symbol logbook. It only needs to be done once for each terminal.

Detailed documentation on the logbook can be found in Logbook Reference Manual.

EXAMPLES

1
$ @SYS$MANAGER:LOGBOOK.COM
$ LOGBOOK

DO + LOGBOOK

After typing the logbook command at the $ prompt a standard TPU editor buffer will appear in your terminal. Type DO LOGBOOK and Figure 7 will appear.
Figure 7  LOGBOOK initial screen

```
DECTerm 1

File  Edit  Commands  Options  Print

Max logbook shot until now is 920410008 23-JUL-1992 12:25:55.32

Topics: HYBRID  MDS-PLUS  MHD  POWER_SYSTEMS  SESSION  LEADER  SOFTWARE
       TCI

Keywords:

Buffer: LOGBOOK_INFO
LOGBOOK settings for (PF1 E) MAKE Entry and (PF1 V) SHOW View
(PF1 R) SET Run       ALL       "<Undefined>"
(PF1 S) SET Shot      ALL       "<Undefined>"
(PF1 T) SET Topic     "<Undefined>"
No SScript.

Buffer: LOGBOOK_SETTINGS

[EOB]

Buffer: LOGBOOK_SCRATCH

Thank you
```
OLD_TRAVERSER

The OLD_TRAVERSER is a DECWINDOWS based programs for viewing and manipulating MDS-Plus trees. This utility has been replaced by the MOTIF based traverser.

FORMAT

MCR OLD_TRAVERSER  [command options]

Command Options             Defaults
-TREE tree-name
-SHOT shot-number
-EDIT

restrictions

None.

OPTIONS

-TREE tree-name
The -TREE option is used to make the OLD_TRAVERSER open a particular tree when it comes up.

-SHOT shot-number
The -SHOT option is used in conjunction with the -TREE option to specify which shot the traverser should open. Shot -1 is the model, shot 0 is the current shot, and any other number is some specific shot.

-EDIT
The -EDIT option causes the traverser to open the specified tree for edit. This allows the addition, deletion, and renaming of nodes as well as node tag manipulation. Do not open trees for EDIT unless it is necessary since you need exclusive access to the tree when in edit mode.

DESCRIPTION

The OLD_TRAVERSER is a decwindows based MDS-Plus tree manipulator. It can be used view and modify the contents of nodes in a tree as well as perform tree editing operations.

This utility has been replaced by the MOTIF based traverser.

The buttons in the menus in the menu bar at the top of the window operate on the ‘selected’ node(s). To select a node click left on the nodes name in the traverser window. To select more than one node drag left around the nodes.

There is a menu which pops up using the middle mouse button. This menu will operate on the node which was under the mouse when the button was pressed.

To traverse down in the tree double click left on the node you wish to descend into.
To traverse up in the tree double click right on any node currently displayed.

**EXAMPLES**

```
$ MCR OLD_TRAVERSER -TREE cmod -SHOT -1
```

This command opens the model of the CMOD tree. Figure 8 shows the display created by this command.

Figure 8  Old Traverser display
STATE

Program to monitor/control the CMOD state machine.

FORMAT

MCR STATE -TREE treename [-manual | -camac] [color-options]

<table>
<thead>
<tr>
<th>Command options</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>-TREE treename</td>
<td>-TREE cmod</td>
</tr>
<tr>
<td>-MANUAL</td>
<td>monitoring</td>
</tr>
<tr>
<td>-CAMAC</td>
<td>monitoring</td>
</tr>
<tr>
<td>-STARTINGCOLOR</td>
<td>-STARTINGCOLOR yellow</td>
</tr>
<tr>
<td>-CURRENTCOLOR</td>
<td>-CURRENTCOLOR green</td>
</tr>
</tbody>
</table>

restrictions

Only one user can run state in MANUAL or CAMAC mode for a given experiment. Most users will not specify -manual or -camac on the command line and so will run the state program in monitoring mode, monitoring the state of the one controlling copy of state.

OPTIONS

-TREE treename
Specifies the name of the TREE whose state you wish to monitor or control.

This tree is used to get the current shot number for display in the state dialog box on the screen.

-MANUAL
Makes the push buttons in the state dialog define the state for this experiment.

Note that using this option makes this the ONE AND ONLY copy of state which controls the state for this experiment.

-CAMAC
Causes state to communicate with the PLC via the IGOR modules to get the current state information.

Note that using this option makes this the ONE AND ONLY copy of state which controls the state for this experiment.

-STARTINGCOLOR color-name
On color/Grayscale systems only: Specifies what color to use to denote that the dispatcher has started working on a phase.

To the the monochrome behavior use -STARTINGCOLOR yellow -CURRENTCOLOR yellow.

For grayscale systems -STARTINGCOLOR gray25 -CURRENTCOLOR gray50 are fairly good choices.
STATE

-CURRENTCOLOR color-name
On color/Grayscale systems only: Specifies what color to use to denote that the dispatcher has completed working on a phase.
To the the monochrome behavior use -STARTINGCOLOR yellow -CURRENTCOLOR yellow.
For grayscale systems -STARTINGCOLOR gray25 -CURRENTCOLOR gray50 are fairly good choices.

DESCRIPTION
The state program is used to control and monitor a state machine for the phases of the experiment. Most users will invoke STATE in a mode which monitors the controlling copy of state. The lead engineer or shot leader will create the controlling copy.
If neither -MANUAL or -CAMAC is used on the command line state will communicate with the ONE copy of state which is running the shot cycle.

EXAMPLES
1
$ MCR STATE -TREE CMOD
This command invokes the state program in the monitoring mode and displays the current state and shot number for the CMOD experiment.

2
$ MCR STATE -TREE CMOD -STARTING gray25 -CURRENT gray50
This command invokes the state program in the monitoring mode and displays the current state and shot number for the CMOD experiment. Use this for grayscale systems. See Figure 9 for sample display in state monitor mode.

Figure 9 State Display in state monitor mode

Shot Cycle Control/Monitor

Experiment: CMOD Remaining: 0:00:00
Shot: 805 In State: 00:00:22
Standby CoolDown Test Recool Init Check Pulse Abort
Figure 10  State Display in camac mode

This command invokes the controlling copy of state for the CMOD experiment. It should only be invoked by the lead engineer or the shot leader. See Figure 10 for sample display in camac mode.

Figure 11  State Display in manual mode

This command invokes the controlling copy of state for the CMOD experiment. It should only be invoked by the lead engineer or the shot leader. See Figure 11 for sample display in manual mode.
SETEVENT

the SETEVENT command generates an MDS event.

<table>
<thead>
<tr>
<th>FORMAT</th>
<th>SETEVENT event-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Qualifiers</td>
<td>Defaults</td>
</tr>
</tbody>
</table>

restrictions

Depending on how your system was installed, MDS events may only be visible to processes in the same group as the issuing process. MDS events are visible to processes running on CPUs in the same CLUSTER as the issuing process. They are not transmitted over DECNET.

prompts

EVENT event-name

PARAMETERS event-name

Specifies the name of the event to generate. Names are converted to uppercase. The event name can be any alphanumeric string from 1 to 25 characters long.

DESCRIPTION

The SETEVENT command is used to generate MDS events for interprocess synchronization. Other processes in the cluster can wait for, or perform asynchronous operations on, events set with the SETEVENT command.

EXAMPLES

1 $ SETEVENT DATA_READY

This command generates the event called DATA_READY. Any processes waiting for this event would continue after this command is invoked.
**TCL**

Interpreter for viewing and manipulating MDS-Plus trees.

### FORMAT

**TCL**  
\[command-string\]

<table>
<thead>
<tr>
<th>Command Qualifiers</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>[[NO]]COMMAND[=file-spec]</td>
<td>/COMMAND=TCLINIT</td>
</tr>
<tr>
<td>[[NO]]LIBRARY[=file-spec]</td>
<td>/NOLIBRARY</td>
</tr>
</tbody>
</table>

### restrictions

None.

### prompts

None.

### PARAMETERS

**command-string**

Specifies a TCL command to be executed. If you use the command-string parameter, TCL will return to DCL after completion of the specified command. Any valid TCL command can be used, including the invocation of a command macro or command procedure file.

### DESCRIPTION

The TCL command invokes the Tree Command Language interpreter. This is an interactive program for looking at, modifying, and editing MDS-Plus trees. See the Tree Command Language Reference Manual or the HELP command from within TCL for detailed information about this utility.

### COMMAND QUALIFIERS

\[COMMAND[=filespec]\]

\[NOCOMMAND\]

Specifies a TCL command procedure file to be executed when TCL is invoked. The default directory for the file specification is SYS$LOGIN and the default file type is .TCL. If this qualifier is omitted or no file specification is given, the utility will attempt to execute a file called TCLINIT in your SYS$LOGIN directory. If a logical name TCLINIT is defined, the file pointed to by this logical name will be executed.

The \[COMMAND\] qualifier is used to define a startup file for commands which you always want to invoke when you enter TCL.

\[LIBRARY[=filespec]\]

\[NOLIBRARY\]

Specifies the name of a VMS text library to be used as the primary macro library. If a macro library is specified, the library will be opened for read and write access. You can invoke macros from this library by typing their names to the TCL prompt. New macros, or modifications to existing macros, can be saved to this library using the SAVE command.
The TRAVERSER is a DECWINDOWS based program for viewing and manipulating MDS-Plus trees.

**FORMAT**

MCR TRAVERSER  

[command options]

<table>
<thead>
<tr>
<th>Command Options</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>-TREE tree-name</td>
<td></td>
</tr>
<tr>
<td>-SHOT shot-number</td>
<td></td>
</tr>
<tr>
<td>-EDIT</td>
<td></td>
</tr>
<tr>
<td>-READONLY</td>
<td></td>
</tr>
</tbody>
</table>

**restrictions**

None.

**OPTIONS**

**-TREE tree-name**  
The -TREE option is used to make TRAVERSER open a particular tree when it comes up.

**-SHOT shot-number**  
The -SHOT option is used in conjunction with the -TREE option to specify which shot TRAVERSER should open. Shot -1 is the model, shot 0 is the current shot, and any other number is some specific shot.

**-EDIT**  
The -EDIT option causes TRAVERSER to open the specified tree for edit. This allows the addition, deletion, and renaming of nodes as well as node tag manipulation. Do not open trees for EDIT unless it is necessary since you need exclusive access to the tree when in edit mode.

**-READONLY**  
The -READONLY option causes TRAVERSER to open the specified tree in read only mode. This can be used to traverse a tree in a safe mode.

**DESCRIPTION**

The TRAVERSER is a decwindows based MDS-Plus tree manipulator. It can be used view and modify the contents of a tree as well as perform tree editing operations. It is a motif based replacement for the old traverser which can be invoked with the OLD_TRAVERSER command.

The TRAVERSER can display a tree in either an outline format or one of three drawn tree formats. This feature is controled with the Customize menu.

The buttons in the menus in the menu bar at the top of the window operate on the 'selected' node(s). To select a node click left on the nodes name in the TRAVERSER window. To select more than one node drag left around the nodes.
There is a menu which pops up using the middle mouse button. This menu will operate on the node which was under the mouse when the button was pressed.

Nodes which have descendents have a . . . in their label on the display. To open a node, that is see its descendents, double click left on the node you wish to open. To close a node which is open double click left on the node.

The command window at the bottom of the TRAVERSER display accepts any TCL commands. Output is displayed in a separate dialog box.

EXAMPLES

1. $ MCR TRAVERSER -TREE hybrid -SHOT -1

   This command opens the model of the HYBRID tree. Figure 12 shows the display created by this command.

   After opening a few of the nodes and switching to TREE display mode the display would look like Figure 13.
Figure 12  Traverser display
Figure 13  Traverser tree display
WFEVENT

The WFEVENT command waits for one or more MDS Events.

**FORMAT**

```
WFEVENT  event-name [, . . . ]
```

**Command Qualifiers**

<table>
<thead>
<tr>
<th>Qualifier</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>/[NO]LOG</td>
<td>/NOLOG</td>
</tr>
<tr>
<td>/[NO]OR</td>
<td>/NOOR</td>
</tr>
</tbody>
</table>

**restrictions**

Depending on how your system was installed, MDS events may only be visible to processes in the same group as the issuing process.

MDS events are visible to processes running on CPUs in the same CLUSTER as the issuing process. They are not transmitted over DECNET.

**prompts**

```
EVENT  event-name
```

**PARAMETERS**

```
event-name [, . . . ]
```

Specifies the name of the event(s) to wait for. Names are converted to uppercase. The event name can be any alphanumeric string from 1 to 25 characters long.

**DESCRIPTION**

The WFEVENT command will wait until the specified event occurs before continuing. If a list of events is supplied and the /OR qualifier is present then any of the events in the list will cause the command to complete. Otherwise WFEVENT will not complete until all of the events specified on the command line have occurred.

**COMMAND QUALIFIERS**

```
/LOG
/NOLOG
```

Controls whether an informational message is displayed when the event(s) occur.

```
/OR
/NOOR
```

Controls whether WFEVENT waits for one or all of the events listed on the command line. If /OR is specified, control returns when any of the events listed occurs. If omitted, all the events listed must occur before control returns to the calling process.
### EXAMPLES

1. `$ WFEVENT DATA_READY

This command waits for the event DATA_READY. When it occurs WFEVENT returns and the operating system prompts for the next command.

2. `$ WFEVENT DATA READY /LOG

%MDS$-I-EVENT_FULL, 21-JUL-1992 14:19:27.06, Event DATA READY has just occurred

Username: JAS
Image: FREDJR$DKA200:[MDS$ROOT.][SYSEXE]SETEVENT.EXE;12
Process: JAS
Data:

This command waits for the event DATA READY. When it occurs WFEVENT prints out the information above and returns then the operating system prompts for the next command.
WLOGBOOK is a Widget IDL based electronic logbook based on RDB.

**FORMAT**  
@SYS$MANAGER:WLOGBOOK

**DESCRIPTION**
WLOGBOOK is a widget based interface to the electronic logbook. It was implemented using the IDL widget set.

Using WLOGBOOK you can view and create entries in the experiment wide electronic logbook. See the WLOGBOOK Reference Manual for detailed directions on its use.

**EXAMPLES**

```idl
$ @sys$manager:wlogbook
```

IDL. Version 2.2.2 (vms vax).
All rights reserved. Unauthorized reproduction prohibited.
Site: 573.
Licensed for use by: MIT Plasma Fusion Center

```
% Compiled module: MDSPLUS.
% Compiled module: SQL.
% Compiled module: WLOGBOOK.
% Compiled module: XREGISTERED.
% Compiled module: XPMENU.
% Compiled module: EDIT_MENU.
% Compiled module: XMENU.
% Compiled module: LOGSETTABLE.
% Compiled module: XMANAGER.
```

The example above demonstrates how to invoke WLOGBOOK. Figure 14 is what WLOGBOOK will look like when it is first invoked.
Figure 14  WLOGBOOK Display
Index

A

ACTIONS • 3
ACTMON • 6

C

CCL • 8
Commands
   Actions • 3
   Actmon • 6
   Ccl • 8
   Dwpad • 10
   Dwscope • 12
   Logbook • 15
   Old Traverser • 17
   Setevent • 22
   State • 19
   Tcl • 23
   Traverser • 24
   Wfevent • 28
   Wlogbook • 30

O

OLD_TRAVERSER • 17

S

SETEVENT • 22
STATE • 19

T

TCL • 23
TRAVERSER • 24

W

WFEVENT • 28
WLOGBOOK • 30

D

DWPAD • 10
DWSCOPE • 12

E

Events
   Generating Events • 22
   Waiting For Events • 28

L

LOGBOOK • 15