

---

---

# RSH Ethernet API

24 September, 2004

(PRELIMINARY)

---

---

## Background Information

This document describes the operation and syntax for the Remote Shell (rsh) command set supported by the WSD/HD and APR/ClipStore product lines, including the WSD/HD, WSD/HDX, and DDR. Support for rsh commands is a standard feature for the WSD/HD and APR/ClipStore product lines, and will be provided to new customers as well as existing WSD/HD customers through a software update.

### NOTE

These rsh commands are based upon the rsh command set utilized by the WSD/2Xtreme product. If an identical command exists between the 2Xtreme and WSD/HD or APR/ClipStore, then the same syntax and characteristics are to be used.

## Main Hardware Features

There is no special hardware required for the rsh command feature.

## Summary of Main Operational Features

The main operational features include the following:

- rsh commands can be issued from the WSD/HD or APR/ClipStore itself, or from any remote computer.
- rsh commands can be abbreviated, where applicable.
- A “help” command provides clues to operational syntax.

**THIS MANUAL IS PRELIMINARY, AND IS NOT COMPLETE. ERRORS ARE KNOWN TO EXIST. INTENDED FOR SAMPLE PURPOSES ONLY. WHEREVER “Not Yet Documented” IS SHOWN INDICATES AN OBVIOUS LACK OF INFORMATION.**

## List of rsh Commands

The following list provides a summary of the rsh commands. The outlined letters in each command (i.e. the letters “a S” in the command “addSeg”) indicate the short-cut version of the command — type only the outlined letters (in lower case) to execute the given command. **Optional** variables and arguments are shown in brackets, while **required** variables and arguments are not in brackets. Variables shown in *italics* are not literal, but instead represent either a frame number, a timecode value, a decimal number, and/or text.

The page numbers references the location in this document where you will find the complete description for each command.

▪ addSeg	.....Pg. 4	▪ move	..... Pg. 36
▪ audioIn	.....Pg. 5	▪ mute	..... Pg. 37
▪ audioOut	.....Pg. 6	▪ newClipName	..... Pg. 37
▪ bypass	.....Pg. 7	▪ output	..... Pg. 37
▪ clear	.....Pg. 7	▪ openSpace	..... Pg. 38
▪ clearSeg	.....Pg. 8	▪ pingpong	..... Pg. 39
▪ clipCopy	.....Pg. 9	▪ play	..... Pg. 40
▪ clipDelete	.....Pg. 9	▪ playmode	..... Pg. 40
▪ clipInfo	.....Pg. 12	▪ playseg	..... Pg. 41
▪ clipLoad	.....Pg. 10	▪ poster	..... Pg. 42
▪ clipMove	.....Pg. 12	▪ quit	..... Pg. 42
▪ clipPath	.....Pg. 13	▪ recDur	..... Pg. 43
▪ clipRelUID	.....Pg. 14	▪ recMode	..... Pg. 43
▪ clipRename	.....Pg. 15	▪ record	..... Pg. 43
▪ cueIn	.....Pg. 15	▪ recRel	..... Pg. 44
▪ cueOut	.....Pg. 16	▪ recTracks	..... Pg. 44
▪ defSeg	.....Pg. 17	▪ refAuto	..... Pg. 44
▪ disable	.....Pg. 17	▪ reference	..... Pg. 44
▪ edit	.....Pg. 18	▪ remoteEnable	..... Pg. 45
▪ enable	.....Pg. 25	▪ segPlaymode	..... Pg. 45
▪ entry	.....Pg. 26	▪ status	..... Pg. 45
▪ exit	.....Pg. 26	▪ stop	..... Pg. 45
▪ freeze	.....Pg. 27	▪ tcClipPlay	..... Pg. 45
▪ getClipList	.....Pg. 27	▪ tcgen	..... Pg. 45
▪ goSeg	.....Pg. 28	▪ tcgenSource	..... Pg. 46
▪ goto	.....Pg. 28	▪ timeCodeOffset	..... Pg. 46
▪ help	.....Pg. 29	▪ timeCodeSource	..... Pg. 46
▪ hostname	.....Pg. 29	▪ timeCodeType	..... Pg. 46
▪ input	.....Pg. 29	▪ unfreeze	..... Pg. 46
▪ jog	.....Pg. 30	▪ version	..... Pg. 47
▪ length	.....Pg. 30	▪ videoFormat	..... Pg. 47
▪ listDirectory	.....Pg. 31	▪ where	..... Pg. 47
▪ logout	.....Pg. 31	▪ wmeEncoder	..... Pg. 47
▪ loop	.....Pg. 32		
▪ loopSeg	.....Pg. 33		
▪ markIn	.....Pg. 34		
▪ markOut	.....Pg. 34		
▪ mode	.....Pg. 35		

## Functional Description and Operation of rsh Commands

This document contains the list of rsh commands supported by the Accom WSD/HD and APR/ClipStore digital disk recording products (both of which are referred to as “DDR” in this document). These commands can be used via the Ethernet connection if the WSD/HD or APR/ClipStore is set up properly as a Target device with your Workstation. Remember the following important points:

- A command must *always* be typed in lower case letters, whether using the abbreviated command or the command in its entirety.
- Typing “rsh hei di hel p” will list all available rsh commands on your computer terminal display.
- When some commands are typed without any variables, the status for the given command will be reported back to the controlling console. For example: simply typing “pm” (PlayMode) will report if DDR is in normal or segment list play mode. To change the play mode, type: “pm n” (sets play mode to “normal”).
- Any command that contains an outlined letter or a series of outlined letters indicates that the particular command may be *abbreviated* by typing only the outlined letters — in *lower case* text. For example, the command “[p]i ng[p]ong” can be entered either as “pp” when abbreviated, or as “pi ngpong” when typed in its entirety. Upper-case letters are *never* used when issuing rsh commands.
- Variables and arguments listed between brackets “[ ]” are optional.
- Variables and arguments separated by a vertical bar “|” indicate a choice, as in the “Bypass” command which can take the variable “on” or “off”.
- In the examples below, the DDR prompt “wsd%” is given by the interactive shell, during an active remote shell (rsh) session.
- Commands are accessible remotely from your workstation with the remote shell (rsh) command via Ethernet connection.
- The UNIX “rcp” command is used to perform remote file transfers outside the interactive rsh session. Therefore, do not use the “rcp” command during an active rsh session.
- **Optional** variables and arguments are shown in brackets “[ ]” and are not required in all cases.
- **Required** variables and arguments are not in brackets.
- Variables shown in *italics* (i.e. [ *speed* ]) are not literal, but instead represent a whole frame number (i.e. 234), a timecode value in H:M:S:F format (i.e. 2. 13. 23. 07), a decimal number (i.e. 2. 34), and/or entered text (i.e. a clip name). If a timecode value is expressed that includes “0” hours and “0” minutes, these numerals can be eliminated. For example, if the timecode to be expressed is 14 seconds, 23 frames — it may be expressed either as: “00. 00. 14. 23” or as “14. 23”.
- The examples shown below assume the controlling console is given the command prompt “wsd%” and the target DDR has been given the name “hei di”, and is shown as such in all the example rsh commands (i.e. wsd% rsh hei di <command>).

## Remote Shell (rsh) Ethernet Commands

The complete list and description of remote shell commands for the Accom DDR follow. The commands are listed in alphabetical order. For find a particular command, refer to the list of commands on page 2.

■ `addseg [in] [out] [speed]`

**Description:** Adds the currently loaded clip to the end of the currently active segment play list.

**Arguments:** None.

**Variables:** If no variables are specified, the entire clip is added to the play list with a speed of 1 times normal play speed.

The `[in]` variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) specifies the IN point of the clip material that is to be included in the added segment. All material *at and after* the specified IN point is **included** in the defined segment.

The `[out]` variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) specifies the OUT point of the clip material that is to be included in the added segment. All material *at and after* the specified OUT point is **excluded** from the defined segment.

The `[speed]` variable is expressed as a decimal number and specifies a play speed for the clip segment to be used during play list playback. If the `[speed]` variable is undefined when either the `[in]` variable or `[out]` variable is defined will result in a clip segment with 1.00 times normal play speed. The valid range of values for the `[speed]` variable is “-99.999” through “+99.999”.

**Replies:** None.

**EXAMPLES:**

Define a play list segment with an IN point at frame 100, OUT point at frame 200, and a speed of 1 times normal play speed:

```
wsd% rsh hei di addseg 100 200
```

Define a play list segment with an IN point at one second, OUT point at twelve seconds, and one-half normal play speed:

```
wsd% rsh hei di as 1.00 12.00 0.5
```

Define a play list segment that includes the entire clip, with a speed of 1 times normal play speed:

```
wsd% rsh hei di addseg
```

## ■ `audi o i n` aes | mute | tone

**Description:** Selects the digital audio input source. The audio source that is selected will be the audio that gets recorded during a subsequent clip record or clip edit operation.

**Arguments:** Specifying “aes” selects the AES/EBU digital audio input.

Specifying “mute” selects the silent source.

Specifying “tone” selects the **input** tone generator.

**Variables:** None.

**Replies:** If no argument is specified, the current status of the audio input source is returned to the controlling console. One of the following status lines will be returned to the controlling console:

- Audi o I nput Source = AES/EBU
- Audi o I nput Source = MUTE
- Audi o I nput Source = TONE

### EXAMPLES :

Select AES/EBU as the digital audio input source:

```
wsd% rsh hei di audi o i n aes
```

Select input tone generator as the digital audio input source:

```
wsd% rsh hei di ai tone
```

Find out the current source for the digital audio input:

```
wsd% rsh hei di ai
```

Information similar to the following is then returned to the controlling console:

```
Audi o I nput Source = AES/EBU  
wsd%
```

**■ `audiout` aes|mute|tone**

**Description:** Selects the digital audio output source. The audio output source that is selected will be the audio that is heard at both the analog audio outputs and the digital audio outputs from the DDR.

**Arguments:** Specifying “aes” selects the AES/EBU digital audio output.

Specifying “mute” selects the silent source for the digital audio output.

Specifying “tone” selects the **output** tone generator.

**Variables:** None.

**Replies:** If no argument is specified, the current status of the audio output source is returned to the controlling console. One of the following status lines will be returned to the controlling console:

- Audi o I nput Source = AES/EBU
- Audi o I nput Source = MUTE
- Audi o I nput Source = TONE

**EXAMPLES :**

Select AES/EBU as the digital audio output:

```
wsd% rsh hei di audi out aes
```

Select output tone generator as the digital audio output:

```
wsd% rsh hei di ao tone
```

Find out the current source for the digital audio output:

```
wsd% rsh hei di ao
```

Information similar to the following is then returned to the controlling console:

```
Audi o I nput Source = MUTE  
wsd%
```

## ■ `bypass` [on|off]

**Description:** Turns the **Bypass** function on/off, or toggles the state of the Bypass function. In the DDR, the “Bypass” function is analogous to the “E-E” function in a VTR, where the input video and audio are passed directly to the video/audio outputs, bypassing the disk storage system.

**Arguments:** When issued without any argument, this command toggles the state of the Bypass function to the opposite state.

When issued with the [on] argument, the Bypass function is explicitly turned ON.

When issued with the [off] argument, the Bypass function is explicitly turned OFF.

**Variables:** None.

**Replies:** None.

### EXAMPLES :

Toggle the video/audio Bypass function ON:

```
wsd% rsh hei di b on
```

Toggle the video/audio Bypass function OFF:

```
wsd% rsh hei di bypass off
```

Toggle the video/audio Bypass function to the opposite state:

```
wsd% rsh hei di bypass
```

## ■ `clear` [in|out]

**Description:** Clears the previously marked IN point and/or the marked OUT point.

**Arguments:** When issued with no arguments, both the IN point and OUT points are cleared at the same time.

When issued with the [in] argument, only the marked IN point is cleared.

When issued with the [out] argument, only the marked OUT point is cleared.

**Variables:** None.

**Replies:** None.

### EXAMPLES :

Clear both the marked IN and OUT points:

```
wsd% rsh hei di clear
```

Clear the marked IN point only:

```
wsd% rsh hei di c in
```

Clear the marked OUT point only:

```
wsd% rsh hei di clear out
```

**■ clearseg** [*all* | *segment\_number*]

**Description:** Clears all segments in the currently loaded segment play list, or clears only the specified segment number from the currently loaded segment play list.

**Arguments:** If no arguments or variables are specified, no action is taken.

When issued with the [*all*] argument, all segments are cleared from the currently loaded segment play list.

**Variables:** If no arguments or variables are specified, no action is taken.

The [*segment\_number*] variable is expressed as a whole number (i.e. 3, 14, 27), and removes only the specified segment from the currently loaded segment play list.

**Replies:** Specifying a decimal number for the [*segment\_number*] variable returns an error message.

**EXAMPLES :**

Clear all segments in the play list:

```
wsd% rsh hei di clearseg all
```

Clear only segment number 5:

```
wsd% rsh hei di clears 5
```

- `clipcopy` [(*source\_clip.mov*)] [*source\_in*] [*source\_out*] (*dest\_clip.mov*) [*dest\_in*]  
[*dest\_out*] [[v][k][a][t]] [nowait]

Description: Not yet documented.

- `clipdelete` [(*clipname.mov*)]

Description: Deletes a single clip from the DDR list of stored clips from within the currently active directory of clips. If the [(*clipname.mov*)] variable is not specified, then the currently loaded clip is deleted. If no clip is currently loaded, and [(*clipname.mov*)] is not specified, then an error message is returned.

Arguments: None.

Variables: The [(*clipname.mov*)] must be surrounded by parentheses ( ) when the “clipdelete” command is issued.

When issuing the clipdelete command, you may also include a valid clip directory path within the [(*clipname.mov*)] variable.

Replies: If no variables are supplied or if the clip defined is not present, an error message is returned.

## EXAMPLES:

Delete the currently loaded clip:

```
wsd% rsh heidi clipdelete
```

Delete the stored clip (from the current directory) that's named *Mountain bike trip.mov*:

```
wsd% rsh heidi cdel (Mountain bike trip.mov)
```

Delete the stored clip named *Golden Gate Bridge.mov* from the directory named `\QTMovies\Stories\`:

```
wsd% rsh heidi cdel (\QTMovies\Stories\Golden Gate Bridge.mov)
```

■ **clipinfo** [(clipname.mov)]

**Description:** Provides frame rate, video format, clip duration, timecode offset value, output mode any other information pertaining to the clip, as derived from the .mov header file associated with the clip. When “clipname.mov” is provided, the information is returned for the defined clip (without loading the clip if it’s not loaded already), or for the currently loaded clip if “clipname.mov” is not specified.

**Arguments:** None.

**Variables:** The [(clipname.mov)] variable must be surrounded by parentheses ( ) when the “clipinfo” command is issued.

When issuing the clipinfo command, you may also include a valid clip directory path within the [(clipname.mov)] variable.

If the [(clipname.mov)] variable is *not* specified, then the name of the currently loaded clip is returned to the controlling console, along other information described above.

**Replies:** If the clip defined is not present, an error message is returned.

**EXAMPLES:**

Get clip information for the clip named *Car Trip.mov*:

```
wsd% rsh heidi clipinfo (Car Trip.mov)
```

*Information similar to the following is then returned to the controlling console:*

```
Clip Name       = (Car Trip.mov)
Duration        = 00.02.15.10
Frame Rate      = 24psF
Video Format     = HD 1920x1080
Output Mode     = Frame
CineTL Mode     = Normal
Tracks          = VKAT
Color Format     = YUVA 4:2:2:4
Timecode Offset = 01.00.00.00
```

Query the currently loaded clip found in the DDR transport:

```
wsd% rsh heidi cload
```

*Information similar to the following is then returned to the controlling console:*

```
Clip Name       = (New Bridge Story.mov)
Duration        = 00.04.30.00
Frame Rate      = 59.94i
Video Format     = HD 1920x1080
Output Mode     = Frame
CineTL Mode     = Normal
Tracks          = V-AT
Color Format     = YUV- 4:2:2
Timecode Offset = 00.00.00.00
```

■ **clipload** [(clipname.mov)] [reload]

**Description:** Loads the defined clip from the DDR storage array and into the DDR’s active clip transport. If the defined clip is already loaded in the transport, then no action is taken — even the position within the clip does not change.

**Arguments:** When the [reload] argument is specified *with* the [(clipname.mov)] variable, then the specified clip is loaded and cued to the first physical frame of the clip. If the [reload] argument is specified

without [ *(clipname.mov)* ] variable, the currently loaded clip is cued to the first physical frame of the clip.

**Variables:** The [ *(clipname.mov)* ] variable must be surrounded by parentheses ( ) when the “clipload” command is issued.

When issuing the clipload command, you may also include a valid clip directory path within the [ *(clipname.mov)* ] variable.

If the [ *(clipname.mov)* ] variable is *not* specified, then the name of the currently loaded clip is returned to the controlling console.

**Replies:** If the clip defined is not present, an error message is returned.

## EXAMPLES :

Load into the DDR transport the clip named *Car Trip.mov*:

```
wsd% rsh heidi clipload (Car Trip.mov)
```

Load into the DDR transport the clip named *New Bridge Story.mov* from the directory named *\QTMovies\Stories\*:

```
wsd% rsh heidi cload (\QTMovies\Stories\New Bridge Story.mov)
```

Query the currently loaded clip found in the DDR transport:

```
wsd% rsh heidi cload
```

*Information similar to the following is then returned to the controlling console:*

```
Loaded clip: (New Bridge Story.mov)
```

Reload the currently loaded clip. This will cue the clip to the first physical frame within the clip:

```
wsd% rsh heidi cload reload
```

■ **clipmove** [*(source\_clip.mov)*] (*dest\_clip.mov*)

**Description:** Moves a single clip [*(source\_clip.mov)*] from the list of clips stored on the DDR, to a new clip (*dest\_clip.mov*). A valid clip directory path may be included within the [*(source\_clip.mov)*] and (*dest\_clip.mov*) variables. If the two directory paths specified in these two variables are different from each other, then the clip is moved to a new directory (the source clip is automatically deleted after the move is finished). If the directory paths are the same, then the clip is simply renamed (no clip data are moved).

It's not possible to move only a portion of a clip — the entire clip must be moved.

**Arguments:** None.

**Variables:** If the [*(source\_clip.mov)*] variable is not specified, then the currently loaded clip is used as the source clip. If no clip is currently loaded, and [*(source\_clip.mov)*] is not specified, then an error message is returned.

The [*(source\_clip.mov)*] and (*dest\_clip.mov*) variables must be surrounded by parentheses ( ) when the clipmove command is issued. Refer to the Examples given below.

**Replies:** If no variables are defined, an error message is returned. The (*dest\_clip.mov*) variable must be specified, otherwise an error message is returned.

If the [*(source\_clip.mov)*] variable does not match a stored clip, then an error message is returned.

If no clip is currently loaded, and [*(source\_clip.mov)*] is not specified, then an error message is returned.

If there is insufficient disk space in the target directory to accommodate the moved clip as defined by the command, then an error message will be displayed on the controlling console. This error message will appear before any attempt to move clip data begins.

#### EXAMPLES :

Move the currently loaded clip to a new clip named **My new clip.mov**:

```
wsd% rsh hei di clipmove (My new clip.mov)
```

Move the stored clip **Horizon Vista.mov** into a new clip named **Horizon Vista Revealed.mov**:

```
wsd% rsh hei di cm (Horizon Vista.mov) (Horizon Vista Revealed.mov)
```

Move the stored clip named **Trip.mov** from the current working directory into a new directory having the name of **\QTMovies\Special\** and the into a clip named **World Travel.mov**:

```
wsd% rsh hei di cm (Trip.mov) 3.15.6.15 (\QTMovies\Special\World Travel.mov)
```

## ■ `clipath` [ *(directory\_path)* ]

**Description:** Defines the clip path directory in the DDR storage array for loading and recording clips.

**Arguments:** None.

**Variables:** The [ *(directory\_path)* ] variable must be surrounded by parentheses ( ) when the “`clipath`” command is issued.

If the specified [ *(directory\_path)* ] does not exist or is otherwise not valid, an error message is returned.

**Replies:** If the [ *(directory\_path)* ] variable is not specified, then the currently active path is returned.

### EXAMPLES :

Change the clip path directory in the DDR to *\QTMovies\Excursion Project*:

```
wsd% rsh heidi clipath (\QTMovies\Excursion Project)
```

Change the clip path directory in the DDR to *\QTMovies\New Project*:

```
wsd% rsh heidi cpath (\QTMovies\New Project)
```

Query the current clip path directory:

```
wsd% rsh heidi cpath
```

*Information similar to the following is then returned to the controlling console:*

```
Current clip path directory: \QTMovies\Santana
```

■ `cliprebuild [(clipname.mov)] [value] [validate]`

**Description:** Rebuilds the movie header file for the specified clip. This function can be used to change the frame rate of the clip during playback.

**Arguments:** When the `[validate]` argument is specified, a reply is given to the controlling console that indicates the current format for the specified clip, and a listing as to which new formats are valid for the specified clip. These formats can then be used for the `[value]` variable.

**Variables:** The `[(clipname.mov)]` must be surrounded by parentheses ( ) when the `cliprebuild` command is issued.

The choices for the `[value]` variable are as follows:

- |              |               |               |               |
|--------------|---------------|---------------|---------------|
| ▪ 525        | ▪ 720/60p     | ▪ 1080/25p    | ▪ 1080/50i    |
| ▪ 625        | ▪ 1080/23.98p | ▪ 1080/29.97p | ▪ 1080/59.97i |
| ▪ 720/59.94p | ▪ 1080/24p    | ▪ 1080/30p    | ▪ 1080/60i    |

If the `[value]` variable is undefined, then the clip's header file is rebuilt using the existing video format that's already stored in the clip header file.

If the specified `[value]` variable is not valid for the given clip, then an error message is returned.

When issuing the `cliprebuild` command, you may also include a valid clip directory path within the `[(clipname.mov)]` variable.

**Replies:** If the `[(clipname.mov)]` variable is not specified, then the currently loaded clip is used — if no clip is currently loaded in such a case, then an error message is returned.

If the specified `[value]` variable is not valid for the specified clip (for example, trying to rebuild a 525 standard definition clip to have a new high definition 1080/24p frame rate), then an error message is returned.

See also the definition for the `[validate]` argument, above.

#### EXAMPLES :

Rebuild the header file of the currently loaded clip to be **1080/25p**:

```
wsd% rsh heidi cliprebuild 1080/25p
```

Rebuild the header file of the stored clip named **Roaring rapids.mov** to be **720/59.94p**:

```
wsd% rsh heidi crb (Roaring rapids.mov) 720/59.94p
```

Rebuild the header file of the stored clip named **New Bridge Story.mov** from the directory named `\QTMovies\Stories\` to be **1080/24p**:

```
wsd% rsh heidi crb (\QTMovies\Stories\New Bridge Story.mov) 1080/24p
```

Find out which formats are valid for the currently loaded clip:

```
wsd% rsh heidi crb validate
```

*Information similar to the following is then returned to the controlling console:*

```
Current format of this clip: (1080/25p)
Valid formats for this clip: (625) (720/59.94p) (720/60p) (1080/23.98p)
(1080/24p) (1080/25p) (1080/29.97p) (1080/30p) (1080/50i) (1080/59.97i)
(1080/60i)
wsd%
```

■ **cliprename** [(clipname.mov)] (new\_clipname.mov)

Description: Renames the specified clip.

Arguments: None.

Variables: If the [(clipname.mov)] variable is not specified, then the currently loaded clip is used. The (new\_clipname.mov) variable defines the new name of the clip.

Both the [(clipname.mov)] and (new\_clipname.mov) variables must be surrounded by parentheses ( ) when the cliprename command is issued.

When issuing the cliprename command, you may also include a valid clip directory path within the [(clipname.mov)] variable.

It's not possible to specify the directory path within the (new\_clipname.mov) variable, since this path must be the same as the path specified within the [(clipname.mov)] variable. If you wish to move a clip to another directory (which also deletes the original clip from the original directory, use the clipmove command (found on page 12 above).

Replies: If the [(clipname.mov)] variable is not specified, then the currently loaded clip is used — if no clip is currently loaded in such a case, then an error message is returned.

If the (new\_clipname.mov) variable is undefined, an error message is returned.

**EXAMPLES :**

Rename the currently loaded clip to be **Boxcar on siding.mov**:

```
wsd% rsh heidi cliprename (Boxcar on siding.mov)
```

Rename the clip on the DDR that's currently named **Bottle rocket.mov** to be renamed **Skyrocket in flight.mov**:

```
wsd% rsh heidi crn (Bottle rocket.mov) (Skyrocket in flight.mov)
```

Rename the clip that's currently named **Train ride.mov** located in the directory named **\QTMovies\Stories\** to be renamed **Airplane ride.mov**:

```
wsd% rsh heidi crn (\QTMovies\Stories\Train ride.mov) (Airplane ride.mov)
```

■ **cuein**

Description: Cues the DDR video/audio output to the currently marked IN point within the currently loaded clip. For example, if the IN point is marked at 2:15, then the cuein command will seek to the 2:15 location within the clip.

Arguments: None.

Variables: None.

Replies: None.

**EXAMPLES :**

Cue to the IN point that's marked in the currently loaded clip:

```
wsd% rsh heidi cuein
```

Or...

```
wsd% rsh heidi ci
```

## ■ cueout

Description: Cues the DDR video/audio output to the currently marked OUT point within the currently loaded clip. For example, if the OUT point is marked at 2:15, then the cueout command will seek to the 2:15 location within the clip.

Arguments: None.

Variables: None.

Replies: None.

**EXAMPLES :**

Cue to the OUT point that's marked in the currently loaded clip:

```
wsd% rsh hei di cueout
```

*Or...*

```
wsd% rsh hei di co
```

**■ `defseg in out [speed]`**

**Description:** Adds the currently loaded clip to the end of the currently active segment play list.

**Arguments:** None.

**Variables:** If no variables are specified, then the entire clip is added to the play list.

The *in* variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) marks the IN point for the defined segment. All material at the IN point and after the IN point is included in the segment.

The *out* variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) marks the OUT point for the defined segment. All material *at and after* the OUT point is *excluded* from the added segment.

The optional [*speed*] variable is expressed as a decimal number, and defines a play speed for the clip segment, where 1.00 = normal play speed. This play speed is used for this segment during list playback. The valid range of values for the [*speed*] variable is “-99.999” through “+99.999”.

**Replies:** None.

**EXAMPLES :**

Define a play list segment with an IN point at frame 100, OUT point at frame 200, and a speed of 1 times normal speed:

```
wsd% rsh hei di defseg 100 200 1.00
```

Define a play list segment with an IN point at one second, OUT point at twelve seconds, and one-half normal play speed:

```
wsd% rsh hei di def 1.00 12.00 0.5
```

Define a play list segment that includes the entire clip:

```
wsd% rsh hei di defseg
```

**■ `disable`**

**Description:** Removes “master” control and locks out the NetPanel and ShuttlePRO USB user interfaces on the DDR. This command ensures the DDR can only be controlled via rsh and not from the NetPanel and hardware control panel connected to the DDR. Use the “`enable`” command (on page 25) to restore control to the NetPanel and ShuttlePRO user interfaces.

**Arguments:** None.

**Variables:** None.

**Replies:** None.

- `edit` [`insert|preview`] [`w`] `source` [`w`] `dest` `dur` [`pnn`] [`wpnn`] [`dnn`] [`onn`] [`-cvka|-cvka[n]`] [`mon[n]`] [`moff[n]`] [`nowai t`]

Description: The “`edit`” command performs a frame-accurate **auto edit** to or from the DDR using an external VTR that’s connected to the RS422 “**Slave**” port on the DDR. Either the DDR or the VTR may be the recorder for any given edit. You may send some of the edit commands individually or as a complete string. The “`edit`” command can be used to provide setup operations on the DDR and/or VTR, in addition to performing auto edit operations.

Arguments: As follows:

<code>[w]</code>	= DDR
<code>a</code>	= Audio channel (or track)
<code>c</code>	= Record channel (or track)
<code>d</code>	= Edit delay
<code>dest</code>	= Destination (recorder) IN point, expressed as frame or timecode value
<code>dur</code>	= Edit record duration, expressed as frame or timecode value
<code>k</code>	= Key channel (or track)
<code>mon</code>	= Manual edit ON. This is an adjustment value for an “edit on” command when the auto-edit is performed “manually.” This argument is used only in the case where the record VTR did not respond to the normal auto-edit command. This argument controls the auto-edit by rolling the record VTR and then sending manual “edit on” and “edit off” commands to perform the edit. If the manually controlled edit is improperly positioned, this argument can be added to the command string to adjust the “edit on” position.
<code>moff</code>	= Manual edit OFF. This is an adjustment value for an “edit off” command when the auto-edit is performed “manually.” This argument controls the auto-edit by rolling the record VTR and then sending manual “edit on” and “edit off” commands to perform the edit. If the manually controlled edit is improperly positioned, this argument can be added to the command string to adjust the “edit off” position.
<code>nowai t</code>	= In normal execution of rsh commands, the command prompt is not returned until after a given command has completed. This behavior serves as a pacing mechanism for the controlling script; however, in the case of an auto-edit to/from the VTR, it also locks out further control of & status reporting by the DDR until the edit is completed. The “ <code>nowai t</code> ” option allows the command prompt to be returned while allowing the primary auto-edit command to continue. While performing the auto-edit, the “ <code>nowai t</code> ” option allows the controlling program to poll the progress (using the rsh “ <code>where</code> ” command) or abort the edit (using the rsh “ <code>stop</code> ” command). The “ <code>where</code> ” command reports the position and state of the edit so that completion can be determined by the controlling program.
<code>[nn]</code>	= Numerical value, either for timecode or frames; or for audio tracks in the <code>cv[nn]</code> variable.
<code>o</code>	= Edit offset
<code>p</code>	= Preroll for VTR
<code>source</code>	= Source (player) IN point, expressed as frame or timecode value
<code>v</code>	= Video track
<code>wp</code>	= Preroll for DDR

## NOTE

The external VTR must support Sony protocol, and be actively connected to the RS422 “Slave” port on the DDR for this command to work.

## EXAMPLES :

```
Recorder = VTR
Source   = WSD
```

Record a 122-frame duration edit onto the VTR, starting at timecode 01.00.00.00 (one hour) on the VTR, with the source DDR starting at frame 35:

```
wsd% rsh heidi edit 35 01.00.00.00 122
```

```
Recorder = WSD
Source   = VTR
```

Record a 40-frame duration edit onto the DDR, starting at timecode 15.00 (fifteen seconds) on the DDR, with the source VTR starting at timecode 01.00.12.00 (one hour, twelve seconds):

```
wsd% rsh heidi edit 01.00.12.00 w15.00 40
```

```
Recorder = VTR
Source   = WSD
```

Preview a 75-frame duration edit onto the VTR, starting at timecode 02.45.00.00 (two hours, 45 minutes) on the VTR, with the source DDR starting at timecode 5.00 (five seconds):

```
wsd% rsh heidi edit preview w5.00 02.45.00.00 75
```

## Optional Variables and Arguments

The “`edit`” command features **optional variables and arguments** that provide control over additional features in the VTR and DDR, in order to help simplify auto-editing operations on the DDR from a remote workstation computer.

- `edit p $\overline{nn}$`

Description: Sets the **VTR preroll** time, where  $\overline{nn}$  is length of the preroll, expressed in frame or timecode value. This preoll argument and variable can be used alone as shown above, or may follow the “dur” variable in a single string.

- `edit wp $\overline{nn}$`

Description: Sets the **DDR preroll** time, where  $\overline{nn}$  is length of the preroll, expressed in frame or timecode value. This preoll argument and variable can be used alone as shown above, or may follow the “dur” variable in a single string.

- `edit d $\overline{nn}$`

Description: Sets the **DDR edit delay**, where  $\overline{nn}$  is length of the delay, expressed in frames value only (cannot specify H:M:S:F timecode). This argument and variable determines where the DDR begins to record the edit. The  $\overline{nn}$  offset may be specified as a negative value by preceding this frame numeral with a

minus sign (-). This edit delay argument and variable can be used alone as shown above, or may follow the “dur” variable in a single string.

- `edit on[nn]`

Description: Sets the **DDR edit offset**, where `[nn]` is length of the offset, expressed in frames value only (cannot specify H:M:S:F timecode). This argument and variable determines where the VTR video is positioned with respect to the DDR edit IN point. The `[nn]` offset may be specified as a negative value by preceding this frame numeral with a minus sign (-). This edit offset argument and variable can be used alone as shown above, or may follow the “dur” variable in a single string.

- `edit [-cvka | -cvka[nn]]`

Description: Sets the **DDR edit record tracks (channel)**, where “c” indicates the record channel command, “v” indicates the video track; “k” is the key track; “a” is all audio tracks; and “a[nn]” are the specified audio tracks, where `[nn]` may be a numeral from 1 through 8. If more than one audio track is to be included in the edit, you may specify the tracks separated by commas:

**EXAMPLE :**

```
Recorder = VTR
Source   = WSD
```

Define an auto edit with the video track (no key track) and audio tracks 1, 2, 5 and 6 included in the edit record:

```
wsd% rsh hei di edit -cva1, 2, 5, 6 35 01.00.00.00 122
```

- `edit mon[nn]`

Description: This is an adjustment value for an “edit on” command when the auto edit is performed “manually.” This argument and variable is only used in cases where the record VTR does not respond to the normal auto edit command. By using this argument and variable, the edit is controlled “manually” by rolling the record VTR and then sending separate “edit on” (via the “mon” argument) and “edit off” (via the “moff[nn]” argument and variable) to perform the edit. If the manually controlled edit is improperly positioned, the “mon[nn]” numerical offset can be added to command string to adjust the edit position, where `[nn]` is length of the offset, expressed in frames value only (cannot specify H:M:S:F timecode). This argument and variable determines where the VTR begins to record the edit. The `[nn]` offset may be specified as a negative value by preceding this frame numeral with a minus sign (-).

- `edit moff[nn]`

Description: This is an adjustment value for an “edit off” command when the auto edit is performed “manually.” This argument and variable is only used in cases where the record VTR does not respond to the normal auto edit command. By using this argument and variable, the edit is controlled “manually” by rolling the record VTR and then sending separate “edit on” (via the “mon” argument) and “edit off” (via the “moff[nn]” argument and variable) to perform the edit. If the manually controlled edit is improperly positioned, the “moff[nn]” numerical offset can be added to command string to adjust the edit position, where `[nn]` is length of the offset, expressed in frames value only (cannot specify H:M:S:F timecode). This argument and variable determines where the VTR ends recording the edit. The `[nn]` offset may be specified as a negative value by preceding this frame numeral with a minus sign (-).

**E X A M P L E :**

```
Recorder = VTR
Source   = WSD
```

Define an “manual” auto edit with an edit IN delayed by 4 frames and the edit OUT by 2 frames:

```
wsd% rsh hei di edi t mon4 moff2 35 01.00.00.00 122
```

- **edi t nowai t**

**Description:** In the normal execution of rsh commands, the command prompt is not returned until after a command has completed. This behavior serves as a pacing mechanism for the controlling scripts. However, in the case of an edit to or from a VTR, this pacing mechanism locks out further control of and status reporting by the DDR until the edit completes. The “nowai t” argument allows control to be returned to the controlling console while allowing the edit command to continue. While performing the edit, this return of the command prompt allows the controlling program to poll the progress of the edit (using rsh “edi t progress” command) or to abort the edit (using rsh “stop”). The progress command reports the position and state of the edit so that future completion can be determined.

### Subset of VTR Commands

The “edi t” rsh command also features the following **subset of VTR commands** that provide control over additional features in the VTR via the Sony protocol, in order to help simplify auto-editing operations with the VTR and the DDR from a remote workstation computer. These commands are issued by themselves, as indicated below.

- **edi t auto on|off**

**Description:** Enables or disables the “auto edit” function in the VTR. Normally, the auto edit function is always enabled in the VTR. If for some reason the auto edit commands don’t work, then this may need to be enabled.

- **edi t bypass on|off**

**Description:** Enables or disables the “EE” function in the VTR. When enabled, the VTR’s outputs will show the outputs from the DDR (assuming the DDR video/audio outputs are routed into the VTR inputs). This command behaves the same as the “edi t ee” command.

- **edi t cmd cmdstr $\overline{nn}$**

**Description:** Allows any valid hex command string to be sent to the VTR. This is used only in cases where a given Sony protocol command is not implemented in the DDR rsh command set, and you wish to activate a given function that is otherwise supported by the Sony protocol. The numerals  $\overline{nn}$  are the command string in hex format.

- **edi t cue  $\overline{nn}$ |i n|out [nowai t]**

**Description:** Cues the VTR to the specified location on its tape. The numeral  $\overline{nn}$  may be expressed as either a timecode or frame value. If the “i n” argument is used in place of the  $\overline{nn}$  value, then the VTR will cue to the defined edit IN point. Likewise, if the “out” argument is used in place of the  $\overline{nn}$  value,

then the VTR will cue to the defined edit OUT point. The “nowai t” argument allows control to be returned to the controlling console while allowing this edit command to continue.

This “`edit cue`” command behaves the same as “`edit goto`” and “`edit search`” commands.

#### EXAMPLES :

Cue the tape in the VTR to **01.15.07.00** timecode location:

```
wsd% rsh hei di edit cue 01. 15. 07. 00
```

Cue the VTR to the defined edit OUT point on its tape:

```
wsd% rsh hei di edit cue out
```

- `edit ee on|off`

Description: Enables or disables the “EE” function in the VTR. When enabled, the VTR’s outputs will show the outputs from the DDR (assuming the DDR video/audio outputs are routed into the VTR inputs). This command behaves the same as the “`edit bypass`” command.

- `edit fastfwd`

Description: Places the VTR in Fast Forward, 30 times normal speed forward.

- `edit goto [nn]|in|out [nowai t]`

Description: Cues the VTR to the specified location on its tape. The numeral `[nn]` may be expressed as either a timecode or frame value. If the “i n” argument is used in place of the `[nn]` value, then the VTR will cue to the defined edit IN point. Likewise, if the “out” argument is used in place of the `[nn]` value, then the VTR will cue to the defined edit OUT point. The “nowai t” argument allows control to be returned to the controlling console while allowing this edit command to continue.

This “`edit goto`” command behaves the same as “`edit cue`” and “`edit search`” commands.

#### EXAMPLES :

Cue the tape in the VTR to **01.15.07.00** timecode location:

```
wsd% rsh hei di edit goto 01. 15. 07. 00
```

Cue the VTR to the defined edit OUT point on its tape:

```
wsd% rsh hei di edit goto out
```

- `edit help [1|2]`

Description: Displays page 1 or 2 of a list of edit commands and associated syntax. If there is no argument page 1 is displayed.

- `edit in [nn]|query|reset|recall|prerol|cue|search] [nowai t]`

Description: Sets the edit IN point for performing an insert edit when subsequently using the `edit insert` command (refer to next command, below), where: `[nn]` is the timecode value of the edit IN point on the VTR’s videotape. The other variables: “q” or “query” will return the currently set edit IN point;

“reset” clears the existing edit IN point; “recall” returns the previously set edit IN point — for example, if an edit insert has been performed the edit IN point is automatically cleared, but the “recall” command will recall this previously set edit IN point; the “preroll”, “cue” and “search” variables all do the same thing: they roll and park the VTR’s videotape to the currently set edit IN point. The “nowait” argument allows control to be returned to the controlling console while allowing this edit command to continue.

- **edit insert** [[v] [a1] [a2] [tc]] [nowait]

**Description:** Performs an insert Auto Edit on the VTR that does not directly involve the DDR. For example, if you wish to edit between the VTR and another external audio/video source. Where: [v] enables the VTR’s video track for recording; [a1] enables the VTR’s audio track 1 for recording; [a2] enables the VTR’s audio track 2 for recording; and [tc] enables the VTR’s timecode track for recording. The “nowait” argument allows control to be returned to the controlling console while allowing this edit command to continue.

### NOTE

The “edit in” and “edit out” arguments must be set prior to performing an “edit insert” command.

- **edit out** [nn|query|reset|recall|preroll|cue|search] [nowait]

**Description:** Sets the edit OUT point for performing an insert edit when subsequently using the **edit insert** command (refer to previous command, above), where: [nn] is the timecode value of the edit OUT point on the VTR’s videotape. The other variables: “q” or “query” will return the currently set edit OUT point; “reset” clears the currently set edit OUT point; “recall” returns the previously set edit OUT point — for example, if an edit insert has been performed the edit OUT point is automatically cleared, but the “recall” command will recall this previously set edit OUT point; the “preroll”, “cue” and “search” variables all do the same thing: they roll and park the VTR’s videotape to the currently set edit OUT point. The “nowait” argument allows control to be returned to the controlling console while allowing this edit command to continue.

- **edit jog** [-|nn|+nn|-nn]

**Description:** Jogs the VTR the specified number of frames as follows:

- **edit jog**  
Jogs the VTR forward by one frame.
- **edit jog-**  
Jogs the VTR backward by one frame.
- **edit jog+nn**  
Jogs the VTR forward by the specified number of frames, where [nn] is expressed as a frame value.
- **edit jog-nn**  
Jogs the VTR backward by the specified number of frames, where [nn] is expressed as a frame value.

- **edit play**  
Description: Places the VTR in forward play.
- **edit preroll** [*nn*][*query*] [*nowait*]  
Description: With no variables specified, this command cues the VTR's videotape to the edit IN point, minus the preroll value. The preroll duration can be specified with the *nn* variable (specified as a timecode value). With the *q* or *query* variable specified, the currently set preroll duration is returned to the controlling console. The "nowait" argument allows control to be returned to the controlling console while allowing this edit command to continue.
- **edit preview** [*v*] [*a1*] [*a2*] [*nowait*]  
Description: Performs an **edit preview** operation to perform an edit preview between the DDR and the VTR, without performing any recording. Both the edit recorder and player will roll during the preview. This **edit preview** command can also precede "SOURCE" in the **edit** command, so that only the edit player is previewed. The "nowait" argument allows control to be returned to the controlling console while allowing this edit command to continue.
- **edit progress**  
Description: When an auto edit command has been issued with the "nowait" argument specified, this "**edit progress**" command can be issued and returns the following progress status of the auto edit:
  - line 1: edit state numeric in decimal
  - line 2: edit state text
  - line 3: blank
  - line 4: ddr position in TC
  - line 5: ddr position in FRAMES
  - line 6: ddr position in FIELDS
  - line 7: blank
  - line 8: vtr position in TC
- **edit record**  
Description: Places the VTR in record. The recording on the VTR will be a still frame from the DDR video output.
- **edit rewind**  
Description: Places the VTR in rewind mode, at 30 times reverse normal play speed.
- **edit review** [*nowait*]  
Description: Reviews the previously recorded auto edit. The "nowait" argument allows control to be returned to the controlling console while allowing this edit command to continue.
- **edit search** [*nn*][*in*][*out*] [*nowait*]  
Description: Cues the VTR to the specified location on its tape. The numeral *nn* may be expressed as either a timecode or frame value. If the "in" argument is used in place of the *nn* value, then the VTR will

cue to the defined edit IN point. Likewise, if the “out” argument is used in place of the `[nn]` value, then the VTR will cue to the defined edit OUT point. The “nowait” argument allows control to be returned to the controlling console while allowing this edit command to continue.

This “`edit search`” command behaves the same as “`edit cue`” and “`edit goto`” commands.

## EXAMPLES :

Cue the tape in the VTR to **01.15.07.00** timecode location:

```
wsd% rsh hei di edit search 01. 15. 07. 00
```

Cue the VTR to the defined edit OUT point on its tape:

```
wsd% rsh hei di edit search out
```

- `edit shuttle [nn] [-|nn|+nn|-nn]`

Description: Places the VTR in Shuttle. `[nn]` specifies the shuttle speed and can be in the range +/- 128.

- `edit status`

Description: Returns the VTR status in hex with the function decoded edit status.

- `edit still`

Description: Places the VTR in Shuttle mode, at 0 speed. VTR output will be a still frame.

- `edit stop`

Description: Stops the VTR.

- `edit type`

Description: Returns the Sony device type.

- `edit where`

Description: Returns the current VTR timecode position and specifies if it is in drop frame or non-drop frame.

## ■ `enable`

Description: Restores “master” control to the NetPanel and ShuttlePRO USB user interfaces after the “`enable`” command has been used. Refer to the “`disable`” command on page 17.

Arguments: None.

Variables: None.

Replies: None.

**■ `entry` [tc|frames]**

**Description:** Sets the data entry mode on the DDR to be either timecode (H:M:S:F) or whole frame values.

**Arguments:** Selecting the “tc” setting changes the data entry mode to timecode.

Selecting the “frames” setting changes the data entry mode to frames.

**Variables:** None.

**Replies:** When no arguments are given, the current setting for “entry” mode is returned.

**E X A M P L E S :**

Change entry mode to timecode and then issue a “goto” command:

```
wsd% rsh hei di entry tc
wsd% rsh hei di goto 1.00
```

Change entry mode to frames and then issue a “goto” command:

```
wsd% rsh hei di en frame
wsd% rsh hei di g 270
```

**N O T E**

Frame and timecode values entered remotely will be translated as needed into the entry format expected by the DDR.

**■ `exit`**

**Description:** Terminates the current interactive rsh login session. Serves the same function as the `logout` and `quit` commands.

**Arguments:** None.

**Variables:** None.

**Replies:** None.

## ■ freeze [i n|di sk]

**Description:** Toggles the freeze function in the DDR. Can also explicitly freeze the video input or freeze the off-disk image at the current position within the current clip. While Freeze is active, the disk can be moved *without* canceling the Freeze mode. Any disk transport function (i.e. **Play**, **Jog**, **GoTo**, etc.) can move the disk without canceling Freeze. The Freeze mode is canceled after a “**Record**” operation is completed.

**Arguments:** Without arguments, this command acts the same as the NetPanel’s Freeze function. When **Freeze** is OFF, issuing the “freeze” command without any argument turns Freeze ON. If Freeze is ON, issuing the command without any argument turns Freeze OFF.

If the “i n” argument is specified, video input is grabbed and frozen.

If the “di sk” argument is specified and the DDR has Bypass ON, the Bypass will turn OFF and the frame at the current disk position of the currently loaded clip is frozen.

Issuing the “freeze” command without an argument also cancels an active Freeze.

**Variables:** None.

**Replies:** None.

### EXAMPLES :

Select Bypass (to obtain video input), and then Freeze:

```
wsd% rsh hei di freeze i n
```

To freeze the frame at the current position within the currently loaded clip:

```
wsd% rsh hei di f di sk
```

Toggle the Freeze function to the opposite state:

```
wsd% rsh hei di f
```

## ■ getcliplist

**Description:** Returns to the controlling console the list of clips that are stored in the currently active directory. To get a listing of clips from another directory, first change to the other directory (use the “cli ppath” command found on page 13), and then issue the “getcliplist” command.

**Arguments:** None.

**Variables:** None.

**Replies:** Returns to the controlling console the list of clips that are stored in the currently active directory, sorted in alphabetical order.

■ **goseg** *seg\_number*

**Description:** Selects the specified segment and makes it the current play list segment, in preparation for further operations such as “**defseg**”. The **Segment List Play** mode is automatically selected by the “**goseg**” command, as needed.

**Arguments:** None.

**Variables:** The *seg\_number* is expressed as a whole number, and defines the segment ID number (in the list of play list segments) that the “goto” operation works upon.

**Replies:** None.

**EXAMPLE :**

Go to segment 4 in the currently loaded play list:

```
wsd% rsh hei di goseg 4
```

■ **goto** *timecode|field|frame*

**Description:** Seeks the disk to a given location within the currently loaded clip.

**Arguments:** None.

**Variables:** Specifying decimal numbers indicates a timecode value (H:M:S:F). Using a period “.” in the number specifies timecode format, field 1. Using colons “:” specifies timecode format, field 2.

Specify a whole number (without either of these symbols) to define a whole field or frame number. Appending a plus “+” sign to the end of the number seeks to field 2 of that frame.

**Replies:** None.

**EXAMPLES :**

Seek disk to frame #100, field 1:

```
wsd% rsh hei di goto 100
```

Seek to frame #100, field 2:

```
wsd% rsh hei di g 100+
```

Seek to timecode 1.00 second, field 1:

```
wsd% rsh hei di g 1.00
```

Seek to timecode 1:00 second, field 2:

```
wsd% rsh hei di g 1:00
```

Seek to timecode 1.05 second, field 1:

```
wsd% rsh hei di g 1.05
```

Seek to timecode 1.12 second, field 1:

```
wsd% rsh hei di g 1.12
```

**NOTE**

Frame and timecode values entered remotely will be translated as needed into the entry format expected by the DDR.

■ **help**

**Description:** Displays a list of the rsh commands for quick reference.

**Arguments:** Any valid command. For example, “help input” will provide specific help for the “input” command.

**Variables:** None.

**Replies:** When no arguments are given, displays a list of the rsh commands for quick reference. When a valid “command” argument is given, help for that specific command is provided.

■ **hostname**

**Description:** Returns the host computer workstation’s name to the controlling console.

**Arguments:** None.

**Variables:** None.

**Replies:** Returns the host computer workstation’s name to the controlling console.

■ **input on|off**

**Description:** This command is equivalent to the **bypass** rsh command. The **input** command was implemented for compatibility with the Abekas A60 digital disk recorder.

The **input** command turns the **Bypass** function on/off, or toggles the state of the Bypass function. In the DDR, the “Bypass” function is analogous to the “E-E” function in a VTR, where the input video and audio are passed directly to the video/audio outputs, bypassing the disk storage system.

**Arguments:** When issued without any argument, this command toggles the state of the Bypass function to the opposite state.

When issued with the [on] argument, the Bypass function is explicitly turned ON.

When issued with the [off] argument, the Bypass function is explicitly turned OFF.

**Variables:** None.

**Replies:** None.

**EXAMPLES:**

Toggle the video/audio Bypass function ON:

```
wsd% rsh hei di i nput on
```

Toggle the video/audio Bypass function OFF:

```
wsd% rsh hei di i off
```

Toggle the video/audio Bypass function to the opposite state:

```
wsd% rsh hei di i nput
```

### ■ `jog [offset]`

**Description:** Issued without arguments, this command jogs the currently loaded clip forward when used from an interactive rsh session. If the output mode is set to “**Field**” mode, the clip will jog forward by 1 field. If the output mode is set to “**Frame**” mode, the clip will jog forward by 1 frame.

**Arguments:** None.

**Variables:** The `[offset]` variable is expressed as a whole number to specify the number of fields/frames to jog forward or backward. The `[offset]` variable may be specified as a negative value by preceding this field/frame numeral with a minus sign (-).

If you're unsure of whether the output mode is set to field or frame, you may first use the `output` command prior to issuing the `jog` command.

**Replies:** None.

#### EXAMPLES :

Set the output mode to “frame” and then jog the clip forward by 1 frame:

```
wsd% rsh hei di output frm
wsd% rsh hei di jog
```

Set the output mode to “frame” and then jog the clip forward by 6 frames:

```
wsd% rsh hei di output frm
wsd% rsh hei di jog 6
```

Set the output mode to “field” and then jog the clip backward by 16 fields, then jog forward 7 fields:

```
wsd% rsh hei di output fld
wsd% rsh hei di jog -16
wsd% rsh hei di jog -7
```

### ■ `length`

**Description:** Returns to the controlling console the length of the *currently loaded clip*, expressed in H:M:S:F timecode and in frames. To find out the amount of *remaining open space* available on the disk arrays for recording new clips, use the “`openSpace`” command, found on page 38.

**Arguments:** None.

**Variables:** None.

**Replies:** Returns the length of the *currently loaded clip*, expressed in H:M:S:F timecode and in frames.

#### EXAMPLES :

```
wsd% length
```

*Information similar to the following is then returned to the controlling console:*

```
Current clip name : Golden Gate Bridge
Current clip length: 1.03.23.17 (114107)
wsd%
```

## ■ `l``i``s``t``d``i``r``e``c``t``r``y`

**Description:** Returns to the controlling console the list of clips that are stored in the current directory. To list the contents of another directory, you must first change to the desired directory, by using the “`cl i p path`” command (found on page 13).

Serves the same function as the `g e t c l i p l i s t` command (found on page 27).

**Arguments:** None.

**Variables:** None.

**Replies:** Returns to the controlling console the list of clips that are stored in the currently active directory, sorted in alphabetical order.

## ■ `l``o``g``o``u``t`

**Description:** Terminates the current interactive rsh login session. Serves the same function as the `e x i t` and `q u i t` commands.

**Arguments:** None.

**Variables:** None.

**Replies:** None.

■ `[loop [in] [out] [[speed] [count] [wait]] [on|off]`

**Description:** When issued with no arguments or variables, toggles the state of the “**Loop**” play repeat flag (between ON or OFF), without initiating playback. When the loop flag is enabled, if the clip is currently stopped *outside* the boundaries of the marked IN and OUT points, a subsequent `play` command will cause the clip to immediately seek to the marked IN point and play forward from the marked IN point. If the clip is currently stopped *inside* the boundaries of the marked IN and OUT points, then a `play` command will play from the current position within the clip. In either case, the clip then plays forward to the marked OUT point. When the marked OUT point is reached, clip play immediately (and seamlessly) seeks back to the marked IN point, and plays forward again. This play cycle is repeated indefinitely until a transport command (other than `play`) is issued (unless the `[count]` argument is defined).

**Arguments:** When issued with arguments, the arguments are acted upon, and clip playback is initiated. There is no need for a subsequent `play` command to begin clip playback.

When the `[in]` variable is specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), the material at and after the IN point is included in the loop play.

When the `[out]` variable is specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), all material prior to the OUT point is included in the loop playback (the frame at the OUT point is *excluded* from playback).

The `[speed]` variable specifies the play speed of the loop playback (expressed as a decimal number to specify a multiple of 1 times play speed). The valid range of values for the `[speed]` variable is “-99.999” through “+99.999”.

The `[count]` is expressed as a whole number (from “0” to “999”) and defines how many times the loop plays. If “0” is specified, the clip will play from the marked IN point to the marked OUT point and then stops, without looping back to the IN point. If “1” is specified for the count, the clip will play from the marked IN point to the marked OUT point, seek back to the marked IN point, and then stop. After the count is finished, the loop mode is turned OFF.

**Variables:** When issued with variables, the variables are acted upon, and clip playback is initiated. There is no need for a subsequent `play` command to begin clip playback.

Specifying the `[wait]` argument will hold off returning the command prompt to the controlling console until loop playback ends. This provides a simple method for interactive control, especially if the `[count]` variable is used.

Specifying either of the `[on|off]` arguments allows the loop function to be explicitly turned ON or OFF. This provides an unambiguous method of enabling or disabling the loop function, if the current status of the loop flag is unknown.

**Replies:** None.

**EXAMPLES :**

Set loop boundaries at timecode 1.00 and 2.00 in the currently loaded clip, and initiate clip playback at 2.25 times normal play speed:

```
wsd% rsh heidi loop 1.00 2.00 2.25
```

Set loop boundaries at frame 20 and 50 in the currently loaded clip, and initiate clip playback at 1.25 times normal play speed, for six loop cycles:

```
wsd% rsh heidi l 20 50 1.25 6
```

Set loop boundaries at IN=1.15 and OUT=6.10 using the `markin` and `markout` commands, and then explicitly turn the loop flag ON (without initiating clip playback):

```
wsd% rsh heidi markin 1.15
```

```
wsd% rsh hei di markout 6.10
wsd% rsh hei di l on
```

■ **l o o p s e g** [*in*] [*out*] [*speed*]

**Description:** Selects the **Segment List Play** mode and at the same time enables the Loop play repeat function. This command requires that a segment play list be created prior to issuing the **l o o p s e g** command. A subsequent **pl ay** command must be issued in order to begin playback (i.e. the **l o o p s e g** command itself does not start disk playback).

If the “normal” play mode is active at time the **l o o p s e g** command is issued, then the “segment” play mode is automatically selected prior to enabling of the segment play list loop flag that’s initiated by the **l o o p s e g** command.

**Arguments:** None.

**Variables:** If no variables are specified, this command includes in the subsequent playback all segments that have been previously defined in the segment play list, with a play speed of 1 times normal play.

With the [*in*] and [*out*] variables defined, only a section of the total segment play list is included in the segment list loop play. Whole numbers indicate the segment play list IDs that are include in the segment list loop play, while decimal numbers indicate timecode (in H:M:S:F) on the segment play list timeline. The segment play list timeline always begins at 00.00.00.00 timecode. When the [*in*] variable is specified, the material at and after the IN point is included in the list loop playback. When the [*out*] variable is specified, all material prior to the OUT point is included in the list loop playback.

The [*speed*] variable (expressed as a decimal number) defines the “global” speed at which the segment play list will play. Keep in mind that each of the segments in the segment play list may each be defined with their own play speed. The global segment play speed then multiplies the individual segment’s play speed. With only the [*speed*] variable defined, the DDR will play all of the segments defined in the play list, at the specified global play speed. The valid range of values for the [*speed*] variable is “-99.999” through “+99.999”.

**Replies:** None.

**E X A M P L E S :**

Select segment list play Loop mode, including all segments in the play list; then play this list of clips at normal play speed:

```
wsd% rsh hei di l o o p s e g
wsd% rsh hei di pl ay
```

Select segment list play Loop mode, with play list segment 2 through segment 6 included, and define a play speed of one-half normal play speed; then initiate list playback:

```
wsd% rsh hei di l o o p s 2 6 0.5
wsd% rsh hei di pl ay
```

Select segment list play Loop mode, starting at play list timeline timecode 2 seconds, 5 frames and ending at play list timeline timecode 1 minute, 12 seconds and 6 frames; then play this portion of the play list at normal play speed:

```
wsd% rsh hei di l o o p s 2.05 1.12.06
wsd% rsh hei di pl ay
```

■ **markin** [*frame|timecode*]

**Description:** Sets a new mark IN point for the currently loaded clip. The IN point can be used for subsequent loop and ping-pong playback, defining a play segment, or for use in an edit recording.

**Arguments:** None.

**Variables:** When the [*frame|timecode*] variable is specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), the value is used as the new IN point. The material at the marked IN point is always *included* in subsequent playback.

With no variable specified, the current position within the currently loaded clip is used as the new IN point.

**Replies:** None.

**EXAMPLES :**

Set 3 seconds, 22 frames, field 1 as the mark IN point:

```
wsd% rsh hei di marki n 3.22
```

Set frame 650, field 2 as the mark IN point:

```
wsd% rsh hei di marki n 650+
```

Set the current disk position as the mark IN point:

```
wsd% rsh hei di marki n
```

■ **markout** [*frame|timecode*]

**Description:** Sets a new mark OUT point for the currently loaded clip. The OUT point can be used for subsequent loop and ping-pong playback, defining a play segment, or for use in an edit recording.

**Arguments:** None.

**Variables:** When the [*frame|timecode*] variable is specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), the value is used as the new OUT point. The material *at and after* the marked OUT point is always *excluded* from the subsequent playback.

With no variable specified, the current position within the currently loaded clip is used as the new OUT point.

**Replies:** None.

**EXAMPLES :**

Set 5 seconds, 0 frames, field 2 as the mark OUT:

```
wsd% rsh hei di markout 5:00
```

Set frame 20, field 1 as the mark OUT point:

```
wsd% rsh hei di markout 20
```

Set current disk position as the mark OUT point:

```
wsd% rsh hei di markout
```

■ **mode** [`field` | `frame` | 3:2]

**Description:** Changes DDR's video output mode between the "field", "frame" and "cine 3:2" settings. This command has identical operation with the `output` command.

**Arguments:** The `field` (or `fld`) setting selects the "Field" output mode. The `frame` (or `frm`) setting selects the "Frame" output mode. The 3:2 setting selects the "3:2 cine" output mode.

**Variables:** None.

**Replies:** None.

**EXAMPLES :**

Force the video output mode to "frame":

```
wsd% rsh hei di mode frm
```

Force the video output mode to "field":

```
wsd% rsh hei di mode fld
```

■ **move** *source\_in* [*source\_out*] *dest\_in* [*v|k|a*]

**Description:** Copies material from within the currently loaded clip to a new location within the same clip.

**Arguments:** The [*v|k|a*] argument allows moves that include only the video track ([*v*]), the key track ([*k*]), or the group of eight digital audio tracks ([*a*]). If this variable is undefined, then all tracks that are contained in the currently loaded clip are included in the move operation. Furthermore, if a given track is defined in the move command and yet the currently loaded clip does not contain such a track, then this argument is ignored for that track.

**Variables:** The source segment in the clip starts at the *source\_in* point and ends at the [*source\_out*] point (the frame at the [*source\_out*] point is *excluded* from the copy).

This segment is copied to a new location inside the same clip, as defined by the *dest\_in* variable.

For moves containing only a single frame of video, specify only the *source\_in* and *dest\_in* variables.

All of these IN and OUT points may be expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode.

Adding a "+" to the *source\_in*, *source\_out*, and *dest\_in* frame value will start/end the move on a field 2 of that frame. Using a colon ":" in these same arguments when expressing a timecode value also starts/ends the move on field 2.

**Replies:** None.

#### EXAMPLES :

Move one second of all tracks in the clip, starting at timecode 11.00f1 in the currently loaded clip, and moving this material to timecode location 5.00f1 in the same clip:

```
wsd% rsh hei di move 11.00 12.00 5.00
```

This same command, but using "frame" values instead of timecode (in 525-line standard definition):

```
wsd% rsh hei di move 330 360 150
```

Move one field of the audio track only, starting at timecode 4.10f1 in the currently loaded clip, and moving this material to timecode location 8.00f1 in the same clip:

```
wsd% rsh hei di move 4.10 8.00 a
```

This same command, but using "frame" values instead of timecode (in 525-line standard definition):

```
wsd% rsh hei di move 130 240 a
```

Move one frame of the video track only, starting at timecode 0.00f1 in the currently loaded clip, and moving this material to timecode location 12.00f1 in the same clip:

```
wsd% rsh hei di move 0.00 0.01 12.00 v
```

This same command, but using "frame" values instead of timecode (in 525-line standard definition):

```
wsd% rsh hei di move 0 1 500 v
```

Move four frames plus one field of all tracks in the clip, starting at timecode 5.00f1 in the currently loaded clip, and moving this material to timecode location 20:00f2 in the same clip:

```
wsd% rsh hei di move 5.00 5:10 20.00
```

This same command, but using "frame" values instead of timecode (in 525-line standard definition):

```
wsd% rsh hei di move 150 160+ 600
```

■ **mute** [on|off]

**Description:** Enables and disables the audio program output mute function. Both the set of eight analog and set of eight digital audio outputs are affected by this mute function.

**Arguments:** Specify “off” to turn OFF the audio output mute function.  
Specify “on” to turn ON the audio output mute function.

**Variables:** None.

**Replies:** None.

■ **newclipname** [auto|TOD] (*clipname*)

**Description:** The [auto|TOD] switch for this command determines whether the New Clip record mode automatically increments the currently loaded clip name for new clip recordings (auto), or whether the current time of day (TOD) is used for the clip name for new clip recordings. If the (*clipname*) variable is specified, it can be used to preset the clip name for new clip recordings prior to issuing the New Clip “record” command — with this variable specified, the [auto|TOD] switch is ignored. The “new clip name” command is applicable only when the “record mode” is set to “New Clip” (*see the “Record Mode” command on page 43 below*). The “new clip name” command has no effect if the “record mode” command is set to either “Append” or “Overwrite” mode.

**Arguments:** None.

**Variables:** If the (*clipname*) variable is not specified, then the setting of the [auto|TOD] switch is utilized. If the [auto|TOD] switch is set to “auto” then the name of the currently loaded clip is used as a “base name” for the New Clip recording, with a three-digit number appended to the end of the clip name, starting at “-000”. If there is no clip loaded (and thus no clip name to append), then the clip name will default to the current video format of the clip to be recorded, and is appended with “-000”. For example, if the video format is “1080 at 59.94i” then the new clip name will be preset as “1080-5994-000”. the [auto|TOD] switch is set to “TOD” then the name of the current time of day is used as the clip name for the New Clip recording.

If the (*clipname*) variable is is specified, then the setting of the [auto|TOD] switch is ignored, and the specified (*clipname*) is used for the New Clip record.

**Replies:** If the (*clipname*) variable is specified, then the specified clip name is returned. If the (*clipname*) variable is not specified, then the setting of the [auto|TOD] switch is returned.

■ **output** [field|frame|3:2]

**Description:** Changes DDR’s video output mode between the “field”, “frame” and “cine 3:2” settings. This command has identical operation with the **mode** command.

**Arguments:** The **field** (or **fld**) setting selects the “Field” output mode. The **frame** (or **frm**) setting selects the “Frame” output mode. The 3:2 setting selects the “3:2 cine” output mode.

**Variables:** None.

**Replies:** None.

**EXAMPLES :**

Force the video output mode to “frame”:

```
wsd% rsh heidi output frm
```

Force the video output mode to “field”:

```
wsd% rsh heidi ofld
```

■ `openspace [v|k|a]`

**Description:** Returns to the controlling console the amount of open (free) space that is available on the media disk storage in the DDR, expressed in both timecode and frames — the frame value appears in parentheses. The value of open space is with respect to the currently active video format. See the `videofformat` command on page 47 below for the possible video format settings.

**Arguments:** When no argument is given, the reply provided includes the free space available on all installed media storage sub systems (video, key and audio — assuming the key channel and digital audio options are installed in the DDR). The `[v|k|a]` argument can be used to request the open space on any one or more of the media storage sub systems.

**Variables:** None.

**Replies:** Returns to the controlling console the amount of open (free) space that is available on the media disk storage in the DDR, expressed in both timecode and frames (the frame value appears in parentheses), as shown in the examples below.

**EXAMPLES:**

Request the open space available on the video disk array only:

```
wsd% rsh hei di os v
```

*Information similar to the following is then returned to the controlling console:*

```
Current video format: HD 1080/24p
VIDEO disk array open space = 2.12.05.20 (190220)
wsd%
```

Request the open space available on the all installed disk arrays:

```
wsd% rsh hei di openspace
```

*Information similar to the following is then returned to the controlling console:*

```
Current video format: HD 1080/59.94i
VIDEO disk array open space = 1.07.16.08 (121088)
KEY disk array open space   = 1.15.02.23 (135083)
AUDIO disk array open space = 1.28.07.15 (158625)
wsd%
```

■ **pingpong** [*in*] [*out*] [[*speed*] [*count*] [*wait*]] [on|off]

**Description:** When issued with no arguments or variables, toggles the state of the “**Ping-Pong**” play repeat flag (between ON or OFF), without initiating playback. When the ping-pong flag is enabled, if the clip is currently stopped *outside* the boundaries of the marked IN and OUT points, a subsequent **play** command will cause the clip to immediately seek to the marked IN point and play forward from the marked IN point. If the clip is currently stopped *inside* the boundaries of the marked IN and OUT points, then a **play** command will play from the current position within the clip. In either case, the clip then plays forward to the marked OUT point. Clip play reverses direction when the OUT point is reached, and then plays backward towards the marked IN point. At the IN point, clip play reverses direction again and then plays forward to the marked OUT point. This play cycle is repeated indefinitely until a transport command (other than **play**) is issued (unless the [*count*] argument is defined).

**Arguments:** Specifying the [*wait*] argument will hold off returning the command prompt to the controlling console until ping-pong playback ends. This provides a simply method for interactive control, especially if the [*count*] variable is used.

Issuing the **pingpong** command with either of the [on|off] arguments allows the ping-pong function to be explicitly turned ON or OFF. This provides an unambiguous method of enabling or disabling the ping-pong function, if the current status of the ping-pong flag is unknown.

**Variables:** When issued with variables, the variables are acted upon, and clip playback is initiated. There is no need for a subsequent **play** command to begin clip playback.

When the [*in*] variable is specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), the material at and after the IN point is included in the ping-pong playback.

When the [*out*] variable is specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), all material prior to the OUT point is included in the ping-pong playback (the frame at the OUT point is *excluded* from playback).

The [*speed*] variable specifies the play speed of the ping-pong playback (expressed as a decimal number to specify a multiple of 1 times play speed). The valid range of values for the [*speed*] variable is “-99.999” through “+99.999”.

The [*count*] argument defines how many times the ping-pong plays, expressed in whole numbers from “0” to “999.” If “0” is specified, the clip will play from the marked IN point to the marked OUT point and then stop, *without* reversing play to the IN point. If “1” is specified for the count, the clip will play from the marked IN point to the marked OUT point, reverse direction and play back to the marked IN point, and then stop. After the count is finished, the ping-pong mode is turned OFF.

**Replies:** None.

**EXAMPLES :**

Set ping-pong boundaries at timecode 1.00 and 2.00 in the currently loaded clip, and initiate clip playback at 2 times normal play speed:

```
wsd% rsh hei di pingpong 1.00 2.00 2
```

Set ping-pong boundaries at frame 20 and 50 in the currently loaded clip, and initiate clip playback at 1 times normal play speed, for three ping-pong cycles:

```
wsd% rsh hei di pp 20 50 1 3
```

Set ping-pong boundaries at IN=1.15 and OUT=6.10 using the **markin** and **markout** commands, and then explicitly turn the ping-pong flag ON (without initiating clip playback):

```
wsd% rsh hei di markin 1.15
```

```
wsd% rsh hei di markout 6.10
wsd% rsh hei di pp on
```

### ■ **play** [*speed*]

**Description:** Issued with no argument, this command plays the clip forward at 1 times normal play speed.

**Arguments:** None.

**Variables:** When the [*speed*] variable is defined (expressed as a decimal number), the clip plays at the specified play speed. The valid range of values for the [*speed*] variable is “-99.999” through “+99.999”.

**Replies:** None.

#### EXAMPLES :

Play the currently loaded clip forward at 1x normal play speed:

```
wsd% rsh hei di pl ay
```

Play the currently loaded clip forward at 1/4 normal play speed:

```
wsd% rsh hei di p 0.25
```

Play the currently loaded clip in reverse at 1/2 normal play speed:

```
wsd% rsh hei di p -0.5
```

### ■ **playmode** [*n*ormal | *S*egment]

**Description:** When issued with a valid argument, this command selects the defined play mode. No clip playback is initiated by this command.

**Arguments:** Specifying the *n*ormal (or *n*) argument selects the “normal” playback mode.

Specifying the *S*egment (or *S*) argument selects the “segment” list play mode.

**Variables:** None.

**Replies:** When issued without an argument, the status of the current play mode is returned to the console (reply will be either “normal” or “segment”).

#### EXAMPLES :

Select “segment” play mode (list play):

```
wsd% rsh hei di pm s
```

Select “normal” play mode:

```
wsd% rsh hei di pl aymode n
```

**■ `pl ayseg [ in] [ out] [ speed]`**

**Description:** When no variables are defined, this command places the segment pointer at the first play list segment, and then plays all segments in the play list from beginning to end, at 1 times normal play speed.

If the “normal” play mode is active at time the `pl ayseg` command is issued, then the “segment” play mode is automatically selected prior to segment list playback that’s initiated by the `pl ayseg` command.

**Arguments:** None.

**Variables:** With only the `[ speed]` variable defined (expressed as a decimal number), the DDR will play all of the segments defined in the play list, at the specified global play speed. Keep in mind that the segments in the play list may each be defined with their own play speed. The global segment play speed then multiplies the individual segment’s play speed. The valid range of values for the `[ speed]` variable is “-99. 999” through “+99. 999”.

With the `[ in]` and `[ out]` variables defined (expressed as a decimal number to specify timecode, or expressed as a whole number to specify a segment ID number), only a section of the total segment play list is included in the segment list playback. Whole numbers indicate the segment play list IDs that are include in the segment list playback, while decimal numbers indicate timecode (in H:M:S:F) on the segment play list timeline. The segment play list timeline always begins at 00.00.00.00 timecode. When the `[ in]` variable is specified, the material at and after the IN point is included in the list playback. When the `[ out]` variable is specified, all material *at and after* the OUT point is *excluded* from the list playback.

**Replies:** None.

**EXAMPLES :**

Play the segment play list at one-half normal play speed:

```
wsd% rsh hei di pl ayseg 0.5
```

Play the segment play list from segment ID 3 through segment ID 7 at two times normal play speed:

```
wsd% rsh hei di pl ayseg 3 7 2
```

Play the segment play list from timecode 15.00 to timecode 1.23.03 at normal play speed:

```
wsd% rsh hei di pl ayseg 15.00 1.23.03
```

Play the segment play list from segment ID 6 to timecode 2.15.06 at normal play speed:

```
wsd% rsh hei di pl ayseg 6 2.15.06
```

**■ poster** ["clipname.mov"] [in]

**Description:** With no variables specified, this command creates a thumbnail poster image of the currently loaded clip, using the image at the current disk position within the clip. The poster image appears in the NetPanel user interface when the clip is loaded using the NetPanel user interface.

**Arguments:** None.

**Variables:** With only the [in] argument specified (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode), the command creates a thumbnail poster image of the currently loaded clip, using the image located at the specified frame or timecode within the clip.

With the ["clipname.mov"] argument specified, a poster image is created in the specified clip which is stored in the DDR. You may include a path or directory in the ["clipname.mov"] argument. Otherwise, the clip is assumed to be in the current working directory of the DDR storage array. The clip is not loaded into the workspace when the ["clipname.mov"] argument is used.

**Replies:** None.

**EXAMPLES :**

Create a poster image in the currently loaded clip, from the image located at fifteen seconds, 3 frames inside the clip:

```
wsd% rsh hei di poster 15.03
```

Create a poster image in the currently loaded clip, from where the disk is currently parked within the clip:

```
wsd% rsh hei di poster
```

Create a poster image in the clip named **Bridge Walk.mov**, from the image located at 1 hour, 12 minutes, 15 seconds, and 22 frames inside the clip:

```
wsd% rsh hei di poster "Bridge Walk.mov" 1.12.15.22
```

**■ quit**

**Description:** Terminates the current interactive rsh login session. Serves the same function as the `exit` and `logout` commands.

**Arguments:** None.

**Variables:** None.

**Replies:** None.

■ **recdur** [*duration*]

**Description:** Sets the current record duration for a future record operation (as later initiated by the **record** command, issued with no arguments).

**Arguments:** None.

**Variables:** The [*duration*] variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) defines the recording duration.

**Replies:** If no variable is defined, the currently set record duration is returned to the controlling console, in both timecode and frames. The frame value appears in parenthesis.

**EXAMPLES :**

Set the record duration to 25 seconds:

```
wsd% rsh hei di recdur 25.00
```

Set the record duration to 16 frames:

```
wsd% rsh hei di recd 16
```

Find out what the current record duration is set to:

```
wsd% rsh hei di recd
```

Information similar to the following is then returned to the controlling console:

```
Current record duration = 1.00.00.00 (108000)
wsd%
```

■ **recmode** [*newclip|append|overwrite*]

**Description:** Not documented yet.

**Arguments:** None.

**Variables:** None.

**Replies:** None.

■ **record** [*source*] *in* [*out*]

**Description:** Performs a record operation, starting at the specified IN point within the currently loaded clip and up to the specified OUT point.

**Arguments:** The [*source*] argument may be specified in one of two ways:

- If no source is specified, input video is assumed as the source.
- Specifying a number (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) for the [*source*] variable seeks the disk to a given frame (or timecode) location on disk, freezes this frame, and then uses the frozen frame as the recording source.

**Variables:** The “*in*” variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) defines the starting frame (or timecode) within the currently loaded clip at which point the recording starts. The IN point is included in the recording.

The [*out*] variable (expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode) defines the ending frame (or timecode) within the currently loaded clip at which point the recording ends. The OUT point is *excluded* from the recording

If the “i n” variable is defined and the [*out*] variable is not defined, DDR records for the current duration (as previously set by the `recdur` command — see page 43 above).

Specifying a “+” sign after the IN point causes DDR to begin recording on field 2.

A recording with a duration of only one frame can be achieved by specifying an IN point and OUT point that are one frame apart, or by previously issuing a `recdur` command with 1 frame in duration.

Replies: None.

#### EXAMPLES :

Not yet documented:

```
wsd% rsh hei di
```

Not yet documented:

```
wsd% rsh hei di
```

- `recrel frame_count`

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

- `rectracks [v|k|a1|a2|a3|a4|a5|a6|a7|a8|aal | ]`

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

- `refauto on|off`

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

- `reference [freerun|vi deoi n|sdext |hdext ]`

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **remoteenable** on|off

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **segmentmode** [tc|timecode|id|segid]

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **status** clip|record|machine|all

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **stop**

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **tcdisplay** [frames|timecode]

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **tcgen** [timecode]

Description: Not yet documented.

Arguments: None.

Variables: None.

(expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode)

Replies: None.

■ **tcgen**source [ext|int]

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **timecode**offset [*timecode*]

Description: Not yet documented.

Arguments: None.

Variables: None.

(expressed as a whole number to specify a frame number, or expressed as a decimal number to specify timecode)

Replies: None.

■ **timecode**source [l tc|tcn]

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **timecode**type [drop|non-drop]

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **unfreeze**

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **version**

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **videoformat [value]**

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **where**

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

■ **wmencoder [prf1|prf2|prf3|prf4|prf5] [start|stop]**

Description: Not yet documented.

Arguments: None.

Variables: None.

Replies: None.

END of Document