Mark 6 usage examples (Release 1.0)
14 November 2013
Note: only selected VSI-H command responses shown

Example 1: Prepare a new 1-module RAID group for recording

Insert a new module in slot 1, connect data cables and turn key to enable power

mstat?1; *Get info on module in slot 1*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
-	1	-	8	0	-	16000	unknown	unknown	unk

The modules in slot 1 is newly assembled and has never been initialized.

mod_init = 1 : 8 : ABC00781 : raid : new ; *Initialize the module in slot#1 and register with 8 disks for VLBI configuration as a RAID0 configuration*

mstat? 1; *Get status of module in slot 1*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
-	1	ABC00781/16/4/8	8	8	16000	16000	initialized	unprotected	raid

group = new : 1 ; *Create new 1-module group*

mstat? 1 ; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
1	1	ABC00781/16/4/8	8	8	16000	16000	closed	unprotected	raid

group = open : 1 ; *Create new 1-module group*

mstat? 1 ; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
1	1	ABC00781/16/4/8	8	8	16000	16000	open	ready	raid

The module is now ready for recording

Example 2: Prepare a new 2-module Scatter/Gather group for recording

Insert a new module in slot 1 and 2, connect data cables and turn key to enable power

mstat?1; *Get info on module in slot 1*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
-	1	-	8	0	-	16000	unknown	unknown	unk
-	2	-	8	0	-	16000	unknown	unknown	unk

The modules in slot1 and 2 are newly assembled and has never been initialized.

mod_init = 1 : 8 : ABC00781 : sg : new ; *Initialize the module in slot#1 and register with 8 disks for VLBI configuration as a sg configuration*

mod_init = 2 : 8 : DEF00788 : sg : new ; *Initialize the module in slot#2 and register with 8 disks for VLBI configuration as sg configuration*

mstat? 12; *Get status of module in slot 1 & 2*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
-	1	ABC00781/16/4/8	8	8	16000	16000	initialized	unprotected	sg
-	1	DEF00788/16/4/8	8	8	16000	16000	initialized	unprotected	sg

group = new : 12 ; *Create new 2-module group from modules in slots 1 & 2*

mstat? 12 ; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
12	1	ABC00781/16/4/8	8	8	16000	16000	closed	unprotected	sg
12	2	DEF00788/16/4/8	8	8	16000	16000	closed	unprotected	sg

group = open : 12 ; *Open the group for recording*

mstat? 12 ; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
12	1	ABC00781/16/4/8	8	8	16000	16000	open	ready	sg
12	2	DEF00788/16/4/8	8	8	16000	16000	open	ready	sg

The module is now ready for recording

Example 3: To physically remove a new 1-module raid group for shipping

mstat?1; *Get info on module in slot 1*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
1	1	ABC00781/16/4/8	8	8	16000	16000	open	ready	raid

The module in slot1 is being used, to remove a disk you will have to close the group and unmount it before removing power and physically removing the device from the system

group = close : 1; *Removes the module from recording mode*

mstat? 1; *Get status of module in slot 1*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
1	1	ABC00781/16/4/8	8	8	16000	16000	closed	unprotected	raid

group = unmount : 1 ; *Unmount the disk from the mount point, it is now safe to physically remove power and extract it from the bay.*

mstat? 1 ; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
-	-	-	-	-	-	-	-	-	-

Example 4: Prepare two 2-module groups for recording

Mount four modules and connect data cables

mstat?all; *Get info on all mounted modules*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
12	1	QRS00450/16/WD	8	8	89	16000	closed	protected	sg
12	2	QRS00451/16/WD	8	8	91	16000	closed	protected	sg
3-	3	ABC00150/8/SS	8	8	126	8000	incomplete	protected	sg
3-	-	ABC00151/8/SS	8	8	126	8000	unmounted	-	sg
-	4	-	8	-	-	8000	unknown	-	unk

Modules in slot#s 1&2 are an existing complete group - full, closed, protected; module in slot#3 is one module of a 2-module group being dissolved (it's unmounted companion module is shown); we wish to pair the module in slot #3 with the module in slot #4, which is newly assembled and has never been initialized.

Erase group 12

group = unprotect : 12 ; *Must unprotect before erase*
group = erase : 12; *Erase; group 12 is now empty and available*

Create a new 2-module group from modules in slot#s 3 & 4

group = unprotect : 30 ; *Must unprotect slot#3 module before initialization (unmounted companion module will have to be initialized separately; might possibly not even exist)*

mod_init = 3 : 8 ; *Initialize module in slot#3 and register with 8 disks for VLBI configuration (disks can be configured for sg)*

mod_init = 4 : 8 : ABC00781 :sg : new ; *Initialize new module in slot#4; must specify new MSN*

mstat? all; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
12	1	QRS00450/8/WD	8	8	16000	16000	closed	unprotected	sg
12	2	QRS00451/8/WD	8	8	16000	16000	closed	unprotected	sg
34	3	ABC00150/8/SS	8	8	16000	16000	initialized	null	sg
34	4	ABC00781/8/SS	8	8	16000	16000	initialized	null	sg

group = 34 : new ; *Create new 2-module group*

mstat? all; *Get status of all modules/groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
12	1	QRS00450/8/WD	8	8	16000	16000	closed	unprotected	sg
12	2	QRS00451/8/WD	8	8	16000	16000	closed	unprotected	sg
34	3	ABC00150/8/SS	8	8	16000	16000	closed	unprotected	sg
34	4	ABC00781/8/SS	8	8	16000	16000	closed	unprotected	sg

Example 5: Define input streams, open a group from Example 1, record a scan, and do scan checks

input_stream = add : RDBE1 : vdif : 8224 : 42: 36 : eth0: 192.162.1.38 ;

Define 1st input data stream, source, data format, packet characteristics and source-IP address

input_stream = add : RDBE2 : vdif : 8224 : 42: 36 : eth1 : 192.162.1.40 ;

Define 2nd data-input stream

input_stream = commit ; *Commit the data-input stream changes*

group = open : 12 ; *Open group 12 for recording*

mstat?all; *Get info on all groups*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2	Type
12	1	QRS00450/8/WD	8	8	16000	16000	open	ready	sg
12	2	QRS00451/8/WD	8	8	16000	16000	open	ready	sg
34	3	ABC00450/8/SS	8	8	16000	16000	closed	unprotected	sg
34	4	ABC00781/8/SS	8	8	16000	16000	closed	unprotected	sg

record = on : : :076-1233 : exp123 : wf ; *Start recording on open group*

mstat?; *Get info on the open group 12*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
12	1	QRS00450/8/WD	8	8	16000	16000	open	recording
12	2	QRS00451/8/WD	8	8	16000	16000	open	recording

record = off ; *Stop recording*

Get group 12 status immediately….

mstat? ; *Buffers flushing to disks*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
12	1	QRS00450/8/WD	8	8	15801	16000	open	flushing
12	2	QRS00451/8/WD	8	8	15801	16000	open	flushing

scan_info?; *Will be rejected until data buffers are flushed to disks*

!scan_info? 0 : 0 : 12 : 1 : 076-1233_exp123_wf : complete : 2011h076d12h33m00s : 80 : 2 :0 ;

scan_check?; *Do quick data sanity check*

!scan_check? 0 : 0 : 12 : 1 : 076-1233_exp123_wf : 2 :
RDBE1 : OK : vdif : 2011h076d12h33m01s : 79.9 : 40.0 : 4.0 :
RDBE2 : OK : vdif : 2011h076d12h33m01s : 79.9 : 40.0 : 4.0;

Example 6: Attempt to start recording with insufficient space left on open group.

mstat? ; *Get all-group status*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
12	1	QRS00450/8/WD	8	8	19	16000	open	ready
12	2	QRS00451/8/WD	8	8	18	16000	open	ready
34	3	ABC00450/8/SS	8	8	16000	16000	closed	unprotected
34	4	ABC00781/8/SS	8	8	16000	16000	closed	unprotected

Only ~37GB space remaining on group 12; want to do 60-sec recording

rtime? 8192 ; *Get remaining recording time at 8Gbps*

!rtime = 0 : 0 : 12 : 8192 : 35 : 37 : 32000 ; *Only 35 seconds of recording time available (37GB); not enough*

group = protect : 12; P*rotect group 12 (automatically closes group)*

group = unmount : 12 ; *Prepare group 12 for physical removal*

mstat? ; *Get all-group status*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
12	1	QRS00450/8/WD	8	8	19	16000	unmounted	-
12	2	QRS00451/8/WD	8	8	18	16000	unmounted	-
34	3	ABC00450/8/SS	8	8	16000	16000	closed	unprotected
34	4	ABC00781/8/SS	8	8	16000	16000	closed	unprotected

group = open : 34 ; *Open group 34 for recording*

record = 12y076d13h30m : 60 : :076-1330 : exp123 : wf ; *Record for 60 seconds on group 34*

mstat? ; *Get all-group status (group 12 modules have been removed)*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
34	3	ABC00450/8/SS	8	8	15970	16000	open	ready
34	4	ABC00781/8/SS	8	8	15970	16000	open	ready

A new 2-module group may be mounted in slots 1&2 to takeover recording when group 34 becomes full.

Example 7: Identify module and S/N of undiscovered disk in a 2-module group

mstat?; *Get current all-group status*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
12	1	QRS00650/8/WD	7	8	14000	16000	incomplete	-
12	2	QRS00651/8/WD	8	8	16000	16000	incomplete	-

Only 7 of 8 'registered disks' (at initialization time) in slot #1 module have been discovered.
Identify serial number of undiscovered disk (dead or physically missing) in slot#1.

disk_info? serial : 1 ; *Get registered S/Ns for disks in slot #1*

!disk_info? 0:0: serial : 1 : QRS00650_8_WD : 7 : 8 : SN1:SN2:-SN3:SN4:SN5:SN6:SN7:SN8;
SN3 is undiscovered ('-' preceding S/N)

Example 8: Check disks in module for uniformity of usage; find slow disk

mstat?; *Get current group status*

Group	Slot#	eMSN	#disks found	#disks registered	GB (remaining)	GB (total)	Status1	Status2
12	1	QRS00850/8/WD	8	8	8240	16000	open	ready
12	2	QRS00851/8/WD	8	8	8778	16000	open	ready

Module in slot #2 has significantly more remaining space, which may indicate one or more slow disks in module in slot #2.

Get slot #1 disk-by-disk usage (GB)

disk_info? Usage : 1 ;
!disk_info? 0:0: usage : 2 : QRS00851_8_WD : 8 : 8 : 971 : 970 : 970 : 969 : 971 : 970 : 970 : 969;
Slot 1 module disk usage is very uniform

Get slot #2 disk-by-disk usage (GB)

disk_info? Usage : 2 ;
!disk_info? 0:0: usage : 2 : QRS00851_8_WD : 8 : 8 : 970: 970: 969: 970: 432 : 970: 970: 971;
5th disk in list has much less data than other 7 disks, so is likely slow.

All 'disk_info?' queries are returned in same physical-disk order as S/N query, so S/N query will identify the culprit.

disk_info? Serial : 2 ; *Get slot 2 disk S/Ns*
!disk_info? 0:0: serial : 2 : QRS00851_8_WD : 8 : SN1:SN2:SN3:SN4:SN5:SN6:SN7:SN8;

SN5 is serial# of slow disk, which can be positively identified and dealt with as appropriate.