

# VGOS Operators Notes

2018-09-26

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Pre-setup</b>	<b>2</b>
A	DRUDG experiment files	2
<b>3</b>	<b>Experiment setup</b>	<b>2</b>
A	Verify NTP is sync'd	2
B	Start FS log file	2
C	Check RDBE Status	3
D	Check Mark 6 Status	3
E	Check MCI Status	3
F	Mount Mark 6 Modules	3
G	Verify RDBE time, offsets, and VDIF epochs	4
H	Initialize pointing	5
I	Set mode and attenuators	5
J	Check RDBEs	6
K	Check pointing	6
L	Make test recording	7
<b>4</b>	<b>Start experiment</b>	<b>8</b>
A	Start non-FS multi-cast logging	8
B	Send "Ready" message	8
C	Start schedule	8
D	Send "Start" message	8
<b>5</b>	<b>Monitor experiment</b>	<b>8</b>
A	Monitor scan_check	8
B	Check RDBE Monitor	9
<b>6</b>	<b>Post experiment</b>	<b>9</b>
A	Stop the schedule	9
B	Stop multicast logging	9
C	Check pointing and SEFDs	9
D	Send "End" message	10
E	Send test scan data files	10
F	Transfer log file	10
G	Remove the module for shipping	11
H	E-transfer module (KPGO)	11
<b>7</b>	<b>Appendix</b>	<b>12</b>
A	Setting Up Password-less SSH	12
B	Setting Up Password-less Log Transfers	13
C	Manually uploading log files	13
D	Manually processing schedules	13
E	Schedule rotation	13
F	Module conditioning	14
G	Setup Field System PC from Cold start	14
H	Setup of RDBEs from a cold start	15
I	Set-up of Mark 6 server from a cold start	16
J	Setup MCI server	17
K	Connecting RDBE IF inputs	17

## 1. Introduction

These notes cover basic procedures for running VGOS experiment operations with the Field System (FS). They are described in a logical order for the tasks that need to be completed. If another order of operations make more sense for some locale that is fine of course.

A basic assumption is that all the equipment is already up and running. The procedures check this and refer to the appendix for set-up instructions for a device if it isn't already running. In addition to these setup procedures, the appendix also contains instructions for some other useful operations.

As written here, the procedures are mostly station independent. However, some station specifics are noted. Some of the station specific content may be eventually removed by some additional scripts that hid the dependencies. This is also a good idea because the scripts can reduce the complexity of the operations.

## 2. Pre-setup

### A. DRUDG experiment files

This step is first for two reasons. First, the schedule file is typically available before the day of the experiment. Secondly, the number of modules in a recording group may eventually depend on the experiment. When that is the case, the number needed will conveyed through the listing that is made in this step.

To create the station specific SNAP and procedure files from the session schedule, fetch the schedule from IVS and DRUDG it by entering, in a FS PC shell, the command

```
fesh -d <sched>
```

where <sched> is the schedule name, eg v16033. If you have manually received the schedule, follow the instructions in the [Manually processing schedules](#) section of the appendix.

Along with the .snp and .prc files for the FS, this will create the listings file <sched><stn id>.lst in /usr2/sched which shows a human readable summary of the schedule. Here <stn id> is the two letter station code, eg "gs" for GGAO. To view this on the FS PC, open in a text viewer; eg, in a FS shell

```
less /usr2/sched/<sched><stn id>.lst
```

Review the information in this listings file to ensure you have sufficient storage for the observation.

## 3. Experiment setup

### A. Verify NTP is sync'd

In the FS enter:

```
check_ntp
```

The output is of the form

remote	refid	st	t	when	poll	reach	delay	offset	jitter
*192.168.1.20	18.26.4.105	2	u	444	1024	377	0.208	-0.336	0.396
+192.168.1.21	18.26.4.105	2	u	167	1024	377	0.215	-0.822	0.153

The offsets should be small and there must be a server with an asterisk \* in the first column. It may take a few minutes to get an \*.

### B. Start FS log file

In the FS, open the experiment log so the set-up will recorded in that log:

```
log=<schedule><stn id> # eg v16033gs
```

### C. Check RDBE Status

From the Field System, check the RDBEs

```
rdbe_status
```

The response values should be 0x0f41. If the response is not correct for any RDBEs, refer to [Setup of RDBEs from a cold start](#) in the appendix.

### D. Check Mark 6 Status

From the Field System, check the Mark6s

```
mk6=dts_id?
```

You should receive a sensible response similar to

```
!dts_id?0:Mark6-4605:1.0.24-1:1.2;
```

If the response is not correct for any Mark6s, refer to [Set-up of Mark 6 server from a cold start](#) in the appendix.

### E. Check MCI Status

You can test whether this is needed by using the FS SNAP procedure:

```
dewar
```

If it is working, you will see the readouts for the vacuum pressure and 20K and 70K stages. If not, see the [Setup MCI server](#) section in the appendix.

### F. Mount Mark 6 Modules

This step is included just after initial verification that all the devices are working, but before the more detailed pre-experiment checks. This allows any issues with getting the modules mounted resolved as soon as possible. However, these steps can be done as part of the making the test recording if that is preferred. All commands are entered into the FS unless otherwise noted.

1. This is to help with debugging, display and clear the Mark 6 message queue:

```
mk6=msg?
```

If an unexplained error happens during the following procedure, please use this command again to get more information.

2. Initialize module; create, mount, and open module

Check status:

```
mk6=mstat?all
```

After the two fields: return code and cplane status (hopefully mstat?0:0) there are 10 fields per group:

```
group:slot:emsn:#disks found:#disks nominal:free space:total space:status1:status2:type
```

It may be easier to read if individual groups are queried; eg. for group 1:

```
mk6=mstat?1
```

If the module has already been initialized, i.e. status1 is initialized, and the data is no longer needed (**be certain first**), erase it:

```
mk6=group=unprotect:<group>  
mk6=group=erase:<group>
```

If the module has not been initialized (status1 is "unknown" and no emsn?), initialize it:

```
mk6=mod_init=<slot#>:<#disks>:<emsn>:<type>:<new>
```

For example

```
mk6=mod_init=1:8:HAY%0001:sg:new
```

**Note:** due to a current incompatibility, the FS–Mark 6 connection will timeout during long running commands such as this.

Until this is fixed, you can optionally run this command directly on the Mark 6 from a shell prompt with

```
ssh root@mark6a
da_client
mod_init=<slot#>:<#disks>:<MSN>:<type>:<new>;
```

note the final semicolon is necessary in da\_client but is automatically added by the FS.

Create, open and mount the group:

```
mk6=group=new:<slots>
mk6=group=mount:<slots>
mk6=group=open:<slots>
```

(Slots is a list of slot numbers included in the group, without any separators, eg <slots>=12)

To query if the group is created properly:

```
mk6=group?
```

Print out should have the group number at the end. If it is -, something has gone wrong.

## G. Verify RDBE time, offsets, and VDIF epochs

In the FS, check RDBE time, offsets, and VDIF epochs:

```
time
```

This will display the pps\_offset, dot, and gps\_offset:

```
2016.320.18:11:39.28/rdbec/!dbe_pps_offset?0:-1.953124995e-08;
2016.320.18:11:39.28/rdbec/!dbe_pps_offset?0:-1.953124995e-08;
2016.320.18:11:39.29/rdbea/!dbe_pps_offset?0:-1.953124995e-08;
2016.320.18:11:39.29/rdbec/!dbe_pps_offset?0:-1.953124995e-08;
2016.320.18:11:39.29/rdbec/!dbe_dot?0:2016-320-18-11-39.296s:-0.007s:33;
2016.320.18:11:39.29/rdbec/!dbe_dot?0:2016-320-18-11-39.296s:-0.018s:33;
2016.320.18:11:39.29/rdbec/!dbe_dot?0:2016-320-18-11-39.296s:-0.024s:33;
2016.320.18:11:39.29/rdbec/!dbe_dot?0:2016-320-18-11-39.296s:0.000s:33;
2016.320.18:11:39.29/rdbec/!dbe_gps_offset?0:-3.758203125e-05;
2016.320.18:11:39.29/rdbec/!dbe_gps_offset?0:-3.758203125e-05;
2016.320.18:11:39.29/rdbec/!dbe_gps_offset?0:-3.758203125e-05;
2016.320.18:11:39.29/rdbec/!dbe_gps_offset?0:-3.758203125e-05;
```

The offsets should be small (GPS typically  $\pm$  a few tens of  $\mu$ s, PPS less than  $\pm 0.1 \mu$ s) and the DOT times should be the same the FS log timestamps, within about 0.1 seconds. The third field of 'dot' output is the VDIF epoch. All RDBEs must have the same value, (33 in this case). The VDIF epochs of all RDBEs are also shown in the RDBE status window.

**If any of the DOT times are not correct**, the ones that are wrong must be set with 'fmset'. To do this from the FS console, press <Control><Shift>T to start fmset, or type

```
fmset
```

in an FS PC shell.

You can select the RDBE to set by letter: a, b, c, or d.

With that RDBE's time being displayed, verify that the time is correct by comparing it to the FS/Computer time. If it is off by a lot, use '.' to get it close – within a few seconds. Once it is close, you can use '+' and/or '-' to increment and/or decrement the RDBE by a second at a time until it agrees with the FS time.

**If the PPS offset is greater in magnitude than  $\pm 1\text{e-}7$  ( $\pm 0.1\text{ }\mu\text{s}$ )** for an RDBE, it must be resync'd. You can try using the 's' command in 'fmset' for each RDBE that has too large an offset. This command will take approximately 45 seconds per RDBE to complete. If as a result, the PPS offset comes within the tolerance, please check that all the RDBEs are sending data with

```
mk6in
```

If an interface is not showing receipt of data, reinitialize the corresponding RDBE with:

```
rdbe_init<id>
```

where <id>=a, b, c, or d for the interface that is not receiving data, eth2, eth3, eth4, or eth5 respectively.

If this does not solve the data transmission problem, or the 's' did not make the offset small enough, restart the RDBE, see [Setup of RDBEs from a cold start](#) in the appendix.

**If the displayed VDIF epochs are not the same**, use the ';' command for each RDBE to set the epoch to the nominal one.

**If this is the first experiment since December 31 and June 30**, and the RDBEs have not had their epochs reset since that date, they should be reset. Use the ';' command in 'fmset' for each RDBE to set it to the nominal VDIF epoch. The third field in the 'dot' output above must be the same for all the RDBEs. The VDIF epochs of all RDBEs are also shown in the RDBE status window.

If any changes were necessary due to the above considerations, check the values again with the FS command:

```
time
```

to verify they are correct. If not, follow the above instructions again and/or seek assistance.

## H. Initialize pointing

In the FS, initialize pointing configuration and send antenna to a test source:

```
proc=point
initp
casa
```

The following sources are the most reliable for these small antennas:

Source	Approximate L.S.T. of transit
Taurus A	05:30
Virgo A	12:30
Cygnus A	20:00
Cas A	23:30

### KPGO:

verify Az and El for source are acceptable

```
antenna=operate
```

If not, try a different source first.

Local apparent sidereal time (L.A.S.T) is displayed in the antenna monitor window (monan) at GGAO and KPGO. Cas A is always up at GGAO and Westford, but another source may be more appropriate at times.

## I. Set mode and attenuators

While waiting for the antenna to move to the test source, setup the experiment mode and adjust attenuators. In the FS,

```
proc=<schedule><stn id> # eg. 'v16033gs'
setupbb
ifdbb
```

```
mk6bb
auto          # sets the attenuators
proc=point
```

## J. Check RDBEs

Locate the RDBE Monitor window (monit6) or start it by pressing <Control><Shift>6. Noting that for some fields the display switches between IF0 and IF1 every second, check for each RDBE that:

1. DOT ticking and correct time
2. All RDBEs have the same epoch
3. DOT2GPS value small ( $\pm$ a few  $\mu$ s) and stable (varies by 0.1  $\mu$ s or less)
4. DOT2PPS value small ( $\pm$ 0.1  $\mu$ s) and stable (varies by 0.004  $\mu$ s or less).
5. RMS values close to 20. They may be higher if the antenna has reached the source since the "auto" command above. They may be higher or lower if the elevation has changed significantly since the "auto" or there is variable RFI.
6. Tsys IF0 and IF1 about 50-100, may be jumping a bit
7. Phase-cal amplitude about 10-100, phase stable to within a few degrees

Leave the window open for later monitoring.

Check RDBE data connectons

```
rdbe=data_connect?
```

and verify that band a,b,c,and d equal 0,1,2, and 3, respectively.

## K. Check pointing

Check that the antenna is now on the source we selected earlier with the FS command

```
onsource
```

The result should be TRACKING. If the antenna status is still SLEWING wait until you see an onsource message in the FS log window or onsource indicates tracking.

Once the antenna is onsource, start the pointing check with

```
fivept
```

This will take a few minutes. Once complete fivept will give you output in the form:

```
xoffset      Az      El      xEl_offs  El_offs
0.00452      99.4469  30.8190  0.01417  -0.00806  0.00801  1 1 01d0 virgoa
```

The xEl\_offs and El\_off values (ie. the 3rd and 4th columns) are the beam spaces offsets of the pointing fit. The absolute value of these should be less than  $\sim 0.02$  degrees in each coordinate. There should also be the flags '1 1' in the 3rd and 4th columns from the end.

Next, measure the SEFDs on test source

```
onoff
```

This will also take a few minutes. Once complete onoff will give you output in the form:

source	Az	El	De	I	P	Center	Comp	Tsys	SEFD	Tcal(j)	Tcal(r)
VAL virgoa	170.9	63.0	15a0	1	l	3016.40	0.9943	52.03	2895.6	55.657	1.67
VAL virgoa	170.9	63.0	15a1	2	r	3016.40	1.0088	47.93	2549.8	53.201	1.60
VAL virgoa	170.9	63.0	15b0	3	l	5256.40	0.9946	49.58	2742.3	55.306	1.66
VAL virgoa	170.9	63.0	15b1	4	r	5256.40	1.0148	41.57	2549.6	61.331	1.84
VAL virgoa	170.9	63.0	15c0	5	l	6376.40	0.9831	42.57	2294.2	53.891	1.62
VAL virgoa	170.9	63.0	15c1	6	r	6376.40	0.9862	44.09	2248.1	50.992	1.53
VAL virgoa	170.9	63.0	15d0	7	l	10216.40	1.0121	51.91	3009.5	57.979	1.74
VAL virgoa	170.9	63.0	15d1	8	r	10216.40	0.9870	53.64	3084.2	57.496	1.72
source	Az	El	De	I	P	Center	Comp	Tsys	SEFD	Tcal(j)	Tcal(r)

Verify SEFDs for eight bands are reasonable. They should be in the range ~2000-3000.

Finally, zero the offsets:

```
azeloff=0d,0d
```

NOTE: there is no space between arguments.

## L. Make test recording

1. Check Mark 6 inputs with

```
mk6in
```

which will show the Gb/s by interface in the FS log. For example, a rate of 2 Gb/s should look like

```
#popen#mk6in/eth2 2.078 eth3 2.079 eth4 2.079 eth5 2.079 Gb/s
```

If one or more interfaces are not showing the approximate nominal data rate (initially 2 Gb/s per interface), it is likely that the corresponding RDBEs need to be reconfigured.

To see more details of the Ethernet ports state for the Mk6, use

```
mk6=input_stream?
```

Sample output:

```
mk6a/!input_stream?0:0:rdbeB:vdif:8224:42:66:eth3:127.0.0.1:12000:0:rdbeC:vdif:8224:42:66:
eth4:127.0.0.1:12000:0:rdbeA:vdif:8224:42:66:eth2:127.0.0.1:12000:0:rdbeD:vdif
:8224:42:66:eth5:127.0.0.1:12000:0;
```

which shows rdbeA going to eth2, rdbeB going to eth3, rdbeC going to eth4, and rdbeD going to eth5.

2. The modules and groups should have been set-up already. If not refer to [Mount Mark 6 Modules](#) above.
3. In FS, record some test data:

```
mk6=record=on:30:30
```

verify that lights on the mk6 flash appropriately. You can check recording status with:

```
mk6=record?
```

It should progress starting as “recording”, then transitioning to “off”.

**If the status stays “pending”**, it may be that not all the RDBEs are sending data. You can check this by using the FS SNAP procedure

```
mk6in
```

as shown in step #1 above.

If all interfaces are receiving data at the correct rate, it may be that the VDIF epochs of all the RDBEs don't agree. See [Verify RDBE time, offsets, and VDIF epochs](#) for detail on how to verify/set.

You can also check if the disk is full with

```
mk6=rtime?
```

4. Once recording ends, check quality:

```
mk6=scan_check?
```

Results should show vdif, the time when recording was started, 30 seconds of data, 30 GB of data with an 8 Gbps data rate.

(Data rate will eventually go to 16 and 32 Gbps.)

## 4. Start experiment

### A. Start non-FS multi-cast logging

From the FS enter:

```
start_mlog
```

If there are no errors reported and the “Done” message is printed the logging has been started.

**NOTE:** this will eventually be replaced by the Monitoring and Archiving System (MAS).

### B. Send “Ready” message

From FS console window enter Control-Shift-G.

**EH:** Jason will eventually move `vgos-msg-gui.py` to the FS machines so we can have better functionality. Maybe the placement of the window should be controlled locally by `.Xresources`.

At this point a GUI window should pop up. Enter the session name, station code (lower case) and select the type of message from the drop down list.

- Click the update values button. This collects the information in real time and the SEFDs from the pointing check in the log file.
- Complete the maser offset value by looking at the maser counter in the maser room.
- In the “to” email address field, send it to `ivs-vgos-ops@lists.nasa.gov`
- Enter a brief comment, include weather information.
- Click the send message button when finished.

### C. Start schedule

In a Linux shell (xterm), look at the list file `<schedule><stn id>.lst` created in the DRUDG step (eg. `v16033gs.lst`). Find the first observation and note line number ‘nnn’ after scan name at start of line.

Now, in the FS, start schedule:

```
schedule=<session><stn id>,#<nnn>
```

**Note:** the pound sign (#) is required and there should be no spaces in the command

### D. Send “Start” message

Send “Start” message using the same procedure as in [Send “Ready” message](#).

## 5. Monitor experiment

### A. Monitor scan\_check

To display `scan_check` results as they come in (and the old ones so far) open the ‘scnch’ window (Control<Shift>K).

Results should show `vdif`, reasonable record start time, about equal seconds and GBs of data (typically 30+), and 8 Gbps data rate. Be aware `scan_checks` *occasionally* fails and the data is okay. An occasional ‘failure’ in scan check typically will not have the data rate at the end, have zeroes for many fields, or very small nonsensical numbers. This is not an indication of anything other than the flakiness of the `scan_check` program itself. Many ‘failures’ in a row, however, should be worrisome to an operator.

Position and size window for convenient viewing, new output will follow any changed size. You can stop this with Control-C

NOTE: The placement and size of the window can be controlled by the `.Xresources` file.



## B. Check RDBE Monitor

Check the display for reasonable values periodically, note some fields alternate between IF0 and IF1 every second:

1. DOT ticking and correct time
2. VDIF epoches for all RDBEs agree, if there is any disagreement some will be in inverse video
3. DOT2GPS value small ( $\pm$ a few  $\mu$ s) and stable (varies by 0.1  $\mu$ s or less).
4. DOT2PPS value small ( $\pm$ 0.1  $\mu$ s) and stable (varies by 0.004  $\mu$ s or less).
5. While *recording* RMS values are close to 20 for the new server, sometime RFI can cause the value to be off, but it should always be between 10 and 40 during recording. If values are outside the nominal range they will be shown in inverse video.
6. Tsys IF0 and IF1 about 50-100 (may lower at Wf due to use of a preliminary cal value), may be jumping a bit
7. Phase-cal amplitude about 10-100, phase stable to within a few degrees.

## 6. Post experiment

### A. Stop the schedule

In the FS, run

```
schedule=
```

### B. Stop multicast logging

In the FS, run

```
stop_mlog
```

### C. Check pointing and SEFDs

Select procedure 'point' library

```
proc=point
```

If the FS has been restarted since the initial check, you will need to re-initialize the pointing set-up

```
initp
```

#### KPGO:

antenna=off (to allow verification of Az and El for source before antenna moves)

Try Cas-A as a source:

```
casa
```

#### KPGO:

If necessary, try other sources from table in [Initialize pointing](#) until one with a good position is found, then:

antenna=operate (restart antenna)

If you re-ran 'initp' above, you will need to restore the experiment set-up:

```
proc=<schedule><stn id> # eg. 'v16033gs'  
setupbb  
ifdbb  
mk6bb  
auto  
proc=point
```

Wait until the antenna is on source. You can either watch the log or check with

```
onsource
```

The result should be “tracking”.

As in the [Check pointing](#) section in pre-experiment, run a pointing check

```
fivept
```

and check the “xoffset” offset values are small. Then check the SEFDs

```
onoff
```

and verify SEFDs for eight bands are reasonable, ~2000-3000.

Finally, zero the offsets

```
azeloff=0d,0d
```

#### KPGO:

```
source=stow (wait until stow is reached) antenna=off
```

### D. Send “End” message

Send “End” message using the same procedure as in [Send “Ready” message](#). Include details such as the stop time and the current weather conditions on-site.

### E. Send test scan data files

- For KPGO, this step is skipped as e-transfers are done using a separate dedicated Mark 6.

In a terminal, log in to the Mark 6

**EH:** this is the part I know the least about and I suspect it different for different stations, maybe using something besides “gather”, have tried it at GGAO?

```
ssh mark6a
gator <group> <filename>.vdif ~/
dqa -d <filename>.vdif
scp <filename>_*.vdif evlbi1.haystack.mit.edu:/data-st12/vgos/<exp>/
```

**EH:** Maybe put into a script, or something, to minimize typing? As it is, it is definitely too much typing  
**AB:** I agree, though easy for me to say as I know nothing about bash. The only variables should be scan name, group#, and server at Haystack

### F. Transfer log file

In FS, close experiment log:

```
log=station
```

In a terminal, copy the log to CDDIS and Haystack with plog. If you are transferring the most recent log you can use

```
plog -l
```

Otherwise use

```
plog <session> # v16033
```

Or if the log file does not conform to the standard naming convention

```
plog <log file path> # eg: plog /usr2/log/v16033gs.log
```

If this is not successful, see [Manually uploading log files](#) in the appendix

## G. Remove the module for shipping

Close and unmount disk module(s) and prepare for e-transferring a scan or experiment.

In the FS

```
mk6=group=close:<slots>
mk6=group=unmount:<slots>
```

Before removing, check the modules are unmounted with

```
mk6=mstat?all
```

Once all show unmounted, turn keys off, remove module(s).

To clear module info in the Field System, again run:

```
mk6=mstat?all
```

## H. E-transfer module (KPGO)

Insert Mark6 modules into the e-transfer Mark6

From the da-client mount the modules and verify all disks are seen:

```
da-client
group=mount:<slots>;
mstat?all;
```

(if you get "6:0:1" restart cplane)

```
group=open:<slots> list?
```

From another xterm window gather the scan(s) to your RAID disk, and de-thread if necessary:

For test scan that needs to be de-threaded:

```
gator <slots> <scan name>.vdif /mnt/raid
dqa -d <scan name>.vdif
```

(this will create 4 files with thread ID on scan name)

For scans where you intend to transfer the entire experiment use gather464:

```
gator -t <slots> "scan name".vdif /mnt/raid
```

Start tsunami server specifying the scans of the session to transfer

```
tsunamid <scan_name>_*.vdif
```

You will see the available scans to be pulled

At another xterm window (in "oper", not "root")

Ssh to Haystack storage nodes:

```
ssh evlbi1.haystack.mit.edu  #(password is oper password)
cd /data-st12/vgos
```

Run tsunami, setting the transfer rate, error free, and connecting back to your machine

```
tsunami set rate 100M set error 0 connect 146.88.148.18
```

Make sure needed files are there to be pulled

```
dir
```

Pull files

```
get *
```

Once transfer is complete exit tsunami client to get prompt back

```
exit  
ls <scan_name>* (verify all scans were copied)
```

After last scan has copied logout

```
logout  
<Ctrl-C> (to quit server)
```

From da-client unmount the disk and prepare for shipping

```
group=umount:<slots>;
```

turn keys off, remove modules

```
mk6=mstat?all
```

(clears module info and checks the modules are unmounted)

## 7. Appendix

### A. Setting Up Password-less SSH

It is convenient to setup password-less login for local devices from the Field System PC. You can do this with SSH using public-key cryptography. To generate public/private key pair with SSH (if you don't already have one), run

```
ssh-keygen
```

Accept the defaults and enter a blank password when prompted.

For each computer you want to enable password-less login, append your public key to `.ssh/authorized_keys` on the remote host. On a recent versions of the Field System OS (i.e., FSL9 based on Debian Wheezy) use the command (use the target host node name or IP address in place of `$host`):

```
ssh-copy-id $host
```

If this is not available use

```
cat ~/.ssh/id_rsa.pub | ssh $host 'cat >> ~/.ssh/authorized_keys'
```

If you do not wish to have *completely* password-less login, an alternative is to encrypt your ssh key with a password and use ssh-agent to unlock it for your session. The upshot is you still have the convenience of password-less login, you just have to enter your password **once** after you login to the FS computer. However this is not recommended for devices that FS procedures use ssh to connect to, including, RDBEs, Mark 6s, MCI, and backend-PCs, which will need to have no prompt for smooth operations.

This is also more secure since the ssh key is encrypted on disk and if anyone ever takes your key, they can not gain access to your systems.

This is a good idea for remote terminals, although is slightly more cumbersome for local access.

To encrypt your private key, enter a password when you generate it. To encrypt an old key, or change its password, use

```
ssh-keygen -p -f ~/.ssh/id_rsa
```

The process for adding your public key to a login to a remote host is the same as above.

Now, when you want to use your ssh key, add it to your ssh-agent with

```
ssh-add ~/.ssh/id_rsa
```

This will decrypt your private key and allow any ssh clients the current login session to use it without a password.

## B. Setting Up Password-less Log Transfers

To store your password for log files transfer to CDDIS, add the following line to the ~/.netrc file:

```
machine urs.earthdata.nasa.gov login <username> password <password>
```

replacing the appropriate fields with your username and password.

This is required for plog to operate correctly.

## C. Manually uploading log files

In general, plog should take care of this for you. However, the following manual procedures are here for completeness:

### CDDIS

log into CDDIS:

```
curl -c $HOME/.urs_cookies -n -L https://depot.cddis.eosdis.nasa.gov/CDDIS_FileUpload/login
```

Upload the file:

```
curl -X POST -b $HOME/.urs_cookies -F "fileType=VLBI" -F "file[]=@<log file path>" https://depot.cddis.eosdis.nasa.gov/CDDIS_FileUpload/upload/
```

### Haystack:

```
scp <log file path> evlbi1.haystack.mit.edu:/data-st12/vgos/logs
```

## D. Manually processing schedules

1. Put schedule in /usr2/sched on FS PC
2. Run drudg, from FS PC shell:

```
cd /usr2/sched
drudg <schedule>.skd
```

Now, in drudg, give the following commands (ignoring text after # and filling in the variables)

```
<stn id>          # the two-letter station id (eg. 'gs' at GGAO)
3                # make .snp
12               # make .prc
9                # change printer output destination
<schedule><stn id>.lst # destination file, eg 'v16033gs.lst'
                  # three more <Return>s
5                # print summary
0                # exit DRUDG
```

■ **AB:** Should use /exper/ directory?

## E. Schedule rotation

1. Start DRUDG with original schedule
2. Pick option 10 in DRUDG.
3. Specify the full file name for the output file, i.e., include the .skd. I suggest you call it hYYDDD.skd. The "h" is to avoid confusing it with *real* schedules.
4. Pick the start time. YYYY MM DD HH MM SS
5. Pick the duration — usually 24 hours
6. End DRUDG
7. Restart DRUDG with the new file to make the normal output (or skip step #6 reselect schedule, option #8)

## F. Module conditioning

1. Load modules and enter da-client

```
ssh mark6a da-client
```

2. In da-client, initialize the modules with the same mod\_init command used for experiment set up:

```
mod_init=<slot#>:<#disks>:<MSN>:<type>:<new>;
```

3. Create a new group with the modules you want to condition. If more than 1 module is being conditioned, group them together.

```
group=new:WXYZ;  
group=mount:WXYZ;  
group=open:WXYZ;
```

WXYZ=slot 1, 2, 3, or 4. Only enter slots with modules in them.

4. Check to the status. It should say "open:ready" for the modules included in the group.

```
mstat?all;
```

5. Leave da-client and navigate to logs.

```
<Control>+C  
cd /home/oper/logs
```

6. Run the hammer script. After, all 8 lights on each module should be lit.

```
nohup hammer.sh &
```

7. Analyze the hammer log file that is generated by the hammer script. The file will generally follow the format-Mark6 serial number, module serial number, and timestamp.log. You need to turn the .log file into a .png file so you can view it graphically, and determine if the module is healthy. Use the hammerplot.sh tool on the log file, eg:

```
hammerplot.sh Mark6-5002-CAR%0005-hammer-20180920191245.log
```

This produces a .png file with a similar name. The important information is at the top, a ratio labelled worst write/required. This should be around 97% or better. So larger than 0.97xx. If this is not the case the module may be beginning to fail, or at least may not be fast enough to record a session without errors.

8. To break the group, do the mod\_init command on each module and reassign groups. If recording on all the modules simultaneously, no further action is needed before an observation besides a test recording.

## G. Setup Field System PC from Cold start

1. Start FS computer
2. Login as user oper
3. Check NTP by running

```
ntpq -np
```

The output is of the form

remote	refid	st	t	when	poll	reach	delay	offset	jitter
*192.168.1.20	18.26.4.105	2	u	444	1024	377	0.208	-0.336	0.396
+192.168.1.21	18.26.4.105	2	u	167	1024	377	0.215	-0.822	0.153

The offsets should be small and there must be a server with an asterisk \* in the first column. It may take a few minutes to get an \*.

4. Other local devices that use NTP (antenna, RDBE, Mark6, etc) can now be started
5. Start the Field System in the login shell

```
fs
```

## H. Setup of RDBEs from a cold start

### Power up RDBEs

Use the power button to start the system. If a reboot is needed, login to the system from a new shell:

```
ssh root@rdbe<id>  
reboot
```

The system will take some time to reboot, and you will notice the front panel will stop updating, though it will not go dark. When it is fully booted the PPS and GPS lights will be flashing, and the front panel will once again increment the VDIF second.

Check the status of all RDBEs with the FS command

```
rdbe_status
```

When the RDBEs respond with a status value 0x0f41, skip to [Check/Set RDBE times](#)

### Start RDBE server

From FS PC shell prompt, login to each RDBE:

```
ssh root@rdbe<id>
```

use <id>=a, b, c, or d to log into RDBE-<id>.

Then, on each RDBE run

```
rbin  
nohup ./rdbe_server 5000 6 &  
exit
```

Repeat this for each RDBE that has been restarted. You can verify when all the RDBEs have started from the FS with:

```
rdbe_status
```

There will be an error for each RDBE that is not ready. When all RDBEs respond with a status value 0x0941, proceed to the next step.

### Load Firmware

To load the firmware on all RDBEs, use the FS command:

```
rdbe_fpga
```

If you want to load the firmware individually for RDBE-<id>, you can use the FS command

```
rdbe_fpga<id>
```

Again, use <id>=a, b, c, or d as necessary.

There will be an error for each RDBE, since it will not respond right away and will time-out. You verify when this is finished from the FS with again using

```
rdbe_status
```

This time, the RDBEs should respond with status 0x0f41.

When all status values reach the correct value, proceed to next step

## Configure RDBEs

To initialize the configuration on all RDBEs, use the FS command:

```
rdbe_init
```

You should get four “success” messages.

If you need initialize an RDBE individually use, the FS command

```
rdbe_init<id>
```

You should get a “success” message.

## Check/Set RDBE times

It is **necessary** to check/set the time with `fmset` for an RDBE **every time** it is restarted. The time only needs to be set if it is not correct already.

To do this from the FS console, press <Control><Shift>T to start `fmset`, or type

```
fmset
```

in an FS PC shell.

You can select the RDBE to set by letter: a, b, c, or d.

With that RDBE's time being displayed, verify that the time is correct at the one second level ( $\pm$  a few tenths) by comparing it to the FS/Computer time. If it is off by a lot, use “.” to get it close, within a few seconds. Once it is close, you can use + and/or - to increment and/or decrement the RDBE by a second at time until it agrees with the FS time.

Be sure to exit with <Escape>.

## I. Set-up of Mark 6 server from a cold start

Use the power switch to start the Mark6. If you need to restart the system, log in as oper using a FS PC shell and reboot.

```
ssh oper@mark6a
sudo reboot
```

If possible, before rebooting, cleanly unmount any modules using the `group=` commands explained earlier in this manual. If this is not possible, ensure the key is switched off before issuing the reboot command.

## Check Mark 6 connection

From the Field System, check the Mark 6 connection

```
mk6=dts_id?
```

You should receive a sensible response similar to

```
!dts_id?0:Mark6-4605:1.0.24-1:1.2;
```

If so, you can skip the rest of this section.

## Starting Mark 6 servers

If you receive an error, check that the Mark 6 servers are running. The programs `cplane` and `dplane` need to be running on the Mark 6. These should startup after boot.

To check check if they are running perform

```
ssh root@mark6a
ps aux | grep plane
```

If they are not, start them

```
/etc/init.d/dplane start
/etc/init.d/cplane start
```



## J. Setup MCI server

You can test whether this is needed by using the FS SNAP procedure:

```
dewar
```

If it is working, you will see the readouts for the vacuum, 20K, and 70K stages. If not, start the server from the FS with:

```
startmci
```

**For Westford** to restart the server:

Log in to the MCI node PC from a new window on the FS PC

```
ssh 192.52.63.139
```

Confirm the server is running (called 'fenode\_server')

```
ps aux |grep server
```

If it is not running, start as follows

```
cd node-software/V0/  
./mci_fenodesrvr 192.52.63.139
```

At this point you should see data points all the way through DI345 scroll once per minute. If you close this window, the server will quit.

## K. Connecting RDBE IF inputs

To avoid possibly overloading, and damaging, the RDBE analog-to-digital (A2D) converters used to sample the signal, it is recommended to make sure that the attenuators are at the maximum setting (31.5)c when connecting cables to the IF inputs. Use the command:

```
rdbe_attenX=<if>,31.5
```

where:

- 'X' is the RDBE 'a', 'b', 'c', 'd' or null for all.
- '<if>' is the IF channel '0', '1', or 'both' (or null) for both

for example to set IF1 attenuator for RDBE-B to the maximum use:

```
rdbe_attenb=1,31.5
```

to set both IF attenuators for RDBE-C to the maximum use:

```
rdbe_attenc=,31.5
```

or to set all IF attenuators for all RDBEs to the maximum, use:

```
rdbe_atten=,31.5
```