Prerequisits

- Go through the system setup procedure for the DBBC3 and verify that the system is in a working condition.

System initialization

Additional information can be found also on the EHT-wiki

Module initialization

ONLY IF REQUIRED: Initialise the modules using the following command. It has to be repeated separately for each Mark6 that has modules that need initialising (example below references the Mark6 using hostname '`recorder1`').

This command will erase all existing data on the modules.

If you are unsure whether to initialise a set of modules, request guidance from AOC.

```
backendctl mark6 recorder1 modules 1,2,3,4 init-fresh
```

if the modules are still in "open" state they must be unmounted before

```
backendctl mark6 recorder1 group unmount
```

Repeat for all recorders

Initialize, configure & validate the DBBC3

Load the OCT_D firmware

On the DBBC3 desktop

- close any other running control software programs
- close the DBBC3 client program
- double click the icon labeled "`DBBC3 Control OCT_D_v120.exe`"
- answer first question with "y" in order to do a full reload of the firmware.
- wait until the control software has fully loaded and responds with "`Waiting for connection on port 4000`"

Setup the system

Verify that the setup for using 2GHz filters is activated:

- Inspect c:\DBBC_CONF\OCT_D_120\dbbc3_config_file_oct_D_120.txt
- Check that the 2GHz version of the fila10g files is being referenced, e.g. `oct_D_2GHz_core3H_1.fila10g`. If you find a reference to e.g. 1GHz setups you need to change the setup by following the instructions in the README file located in the c:\DBBC_CONF\OCT_D_120\ folder. In case

Note: the target setup for the DBBC3 is defined in `/etc/backend.conf`.

The default setup is valid for 230 and 86 GHz. For switching between 230 and 345 GHz the following changes need to be made to the setup.

- make sure the DBBC3 client is not running
- configure the DBBC3 using `backendctl`(on the EHT Control Computer cc-pico)*:*

```
backendctl dbbc3 dbbc3 configure
```

- check for any errors

Validate the system

- make sure the DBBC3 client is not running
- validate the DBBC3 using `backendctl`(on the EHT Control Computer cc-pico)*:*

```
backendctl dbbc3 dbbc3 check
```

- check for any errors

Check time synchronisation

Time syncronisation can be checked with the tick command via the serial interface.

Follow these steps below exactly. Omitting any step will lead to mal-functioning and will require to completely reload the firmware.

On the DBBC3 desktop:

- double-click the `putty` icon
- in putty open connection e.g. to DBBC3 Board A
- in the window hit enter to get to the command prompt and execute:
- tick
- compare the timestamps to a radio-controlled clock
- **when done hit enter to stop the tick command**
- close the putty window

Validate the VLBI System (Except DBBC3)

on the EHT control computer run:

```
backendctl whole check
```

This will check the setup of the control computer and the recorders. The check of the DBBC3 is not yet included in this procedure (see above).

Adjust power levels (DBBC3)

Basically low/high power levels should have been reported by setup script (see above).

In DBBC3 client e.g. on windows desktop or

on the control computer:

```
/home/oper/rottmann/dbbc3/utilities/dbbc3client.py dbbc3
```

check attenuators, e.g. for board A:

```
dbbcifa
```

attenuator settings should be within 20-40, agc should be on

if reported attenuator level is out of range 20-40 the IF power must be decreased/increased.

Do test recording

```
backendctl mark6 all run test-recording 20 30
```

Recording starts with a delay of 20 seconds. Visually check if all recorders are actually recording.

Record & plot

log into the recorder e.g. recorder1

```
ssh -Y recorder1
```

execute:

```
plotdbbc3_m6.sh
```

This will do a short test recording and plot the resulting spectrum in both polarizations

Load and execute the schedule

Scedules are located under `/srv/vexstore`

load the schedule that has been triggered by the AOC:

```
backendctl mark6 all schedule load trigger
```

Follow the schedule:

```
backendctl whole schedule follow trigger
```

Start the Mark6 monitoring client

copy the vex file (e.g. from /srv/vexstore/trigger) to /home/oper/shared/schedules

```
vex2xml.py -f {vexfile} -s Pv
```

check the contents of the generated {schedule}.xml if it contains scans

```
m6schedulemon.py recorder1 {schedule}.xml  &
```

repeat for all recorders you want to monitor