This page is deprecated (only valid for 2021 EHT observations).

Prerequisits

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

System initialization

More detailed 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_v110.exe`" (Version number can be different)
- 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`"

Load the filters and validate the system

make sure the DBBC3 client is not running

```
cd /home/oper/rottmann/dbbc3/utilities
./setupDBBC3_OCT_D.py dbbc3 -n 4
```

After carrying out a number of tests and verifications the script will ask whether you want to load the filters. Answer with 'y'.

Note: any checks prior to this question will fail if the filters have already been loaded.

All checks should report "OK".

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:

- open the dbbc3 client program and issue:

```
disableloop
```

- 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
- in the dbbc3 client program issue:

```
enableloop
```

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:

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