

(Archived) Handling KBT/Solar357 Observations (ScheduleKBT.py Method)

Running The Solar Observations

Kaira Background Task (KBT) is the scheduling software used to observe the Sun in mode 357 with I-LOFAR. It is currently run through a python script that handles all telescope side operations and currently has work-in-progress support to launch a beamformed data recording process on `rcu357_1beam_datastream` to simplify Solar observations.

Currently, all scripts related to KAIRA are stored in the `~/Scripts/kbt` folder on the `LSC`, while the scheduling script is stored in `~/Scripts/Python/obs_schedule`. In almost all cases, the only reason to enter the KBT folder will be to modify the RCU configuration in the case of downed LBA antenna or hardware failures (N.B., the `rcu357` and `rcu357_1beam_datastream` are not automatically synced, you will need to manually transfer these files to the `rcu357` prior to taking observations for the changes to be made).

Once you are in the `~/Scripts/Python/obs_schedule` folder, you will need to run the Python 2 script `ScheduleSolarKBT_DMCK_wip.py`. Most of the flags are not needed for standard observations, and in most cases the command can be run with only the `-s` and `-e` parameters to have the station observe while the Sun is above 10 degrees elevation on the current day, starting either with the Sun reaching 10 degrees, or immediately if the Sun is already passed that altitude.

Unless you have used the `-u` flag, the script will give you a prompt to confirm the start and stop times of the observation when you initially run the command. The observation will not start until the prompt has been accepted.

Such a command would look like this:

```
$ python ScheduleSolarKBT_DMCK_wip.py -e rcu357_1beam_datastream -r
```

In the case you wish to **schedule an observation**, you will need to add the `-d` flag, and optionally start (`-t`) and stop (`-p`) times.

```
$ python ScheduleSolarKBT_DMCK_wip.py -e rcu357_1beam_datastream -r -d YYYY-MM-DD -t HH:MM -p HH:MM
$ python ScheduleSolarKBT_DMCK_wip.py -e rcu357_1beam_datastream -r -d 2022-02-22 -t 09:30 -p 14:55
```

Additionally, if you have to split up the observations, the `-u` flag may be of use, as it will allow the script to run without any input to confirm the start/end times. For example, if there are intermediate observations at 11:00 - 11:30 and 14:15 - 14:45, a `schedule_kbt.sh` script could be written like this

```
python ScheduleSolarKBT_DMCK_wip.py -u -e rcu357_1beam_datastream -r -d 2022-02-22 -p 10:55
python ScheduleSolarKBT_DMCK_wip.py -u -e rcu357_1beam_datastream -r -d 2022-02-22 -t 11:31 -p 14:10
python ScheduleSolarKBT_DMCK_wip.py -u -e rcu357_1beam_datastream -r -d 2022-02-22 -t 14:46
```

Processing the Solar Observations

When run with the `-r` flag, the `ScheduleKBT` script will launch a recording process on `ucc1` to record the raw voltage correlations from the station. Once the observations for local mode have been completed, you will need to process these voltages in order to convert them into Stokes parameters and down-sample them in order to save on space.

Do not start processing solar observation until after the station has been handed back to ASTRON. The process is entirely disk limited, and running the processing script while other data is still being recorded will cause packet loss.

This is performed by using software within a Docker container on `ucc1`. Scripts are copied to each Solar recording directory to simplify the process. To start, you will need to connect to `ucc1` and then go to your solar observations.

```
obs@ucc1~$ cd /mnt/ucc1_recording/data
obs@ucc1:/mnt/ucc1_recording/data$ cd sun/YYYYMMDD_sun357 # Enter the solar data folder for a given observation
```

Afterwards, you will want to enter a `tmux` shell and launch the Docker container to get access to the software you will need, your current directory will be stored in `$LAST_DIR` to make getting back to it easier.

```
obs@ucc1:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357$ tmux new -s solarProcessing
```

```
< enter tmux shell >
```

```
obs@ucc1:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 $ bash enter_docker.sh
```

```
< Docker container with processing software will load >
```

```
root@aaaaaaa:/home/user $ cd $LAST_DIR # Your previous directory was saved in $LAST_DIR, go back to to
root@aaaaaaa:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 #
```

You will then be able to run the processing script, which will hand processing the data, then compressing the outputs. The results can then be transferred wherever they will need to be sent to and the container and shell can be closed.

```
root@aaaaaaa:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 # bash process_data.sh # Run the processing
script
```

```
< allow processing to complete, it may take a few seconds for the initial output to appear in ./output_files/ >
```

```
< exit the container, tmux shells, notify someone to transfer the data to DIAS >
```

```
root@aaaaaaa:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 # exit
```

```
obs@ucc1:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 $ exit
```

```
obs@ucc1 ~ $ exit
```

Command Flow Summary

```
llofar@LGC~ $ cd ~/Scripts/Python/obs_schedule
```

```
llofar@LGC ~/Scripts/Python/obs_schedule $ tmux new -s KBT
```

```
< enter tmux shell >
```

```
llofar@LGC ~/Scripts/Python/obs_schedule $ python ScheduleSolarKBT_DMCK_wip.py
```

```
[TTT]-d YYYY-MM-DD \ # Date for scheduled observation
```

```
-r \ # -r will enable beamforming on ucc1
```

```
-e rcu357_1beam_datastream \ # Name of experiment (no need to change)
```

```
-t HH:MM \ # Start Time (not always needed)
```

```
-p HH:MM \ # End Time (not always needed)
```

```
# Optional
```

```
-u \ # Unattended mode, does not require confirmation of times to start script
```

< Control+b, d to exit tmux shell once things are running as intended >

< Allow all observations to complete, return later, exit the tmux shell with `exit` >

```
llofar@LGC~ $ ucc1
```

< ucc1 is an alias to connect to ucc1 >

```
obs@ucc1~ $ cdr2 # cd to the recording drive
```

```
obs@ucc1:/mnt/ucc1_recording/data $ cd sun/YYYYMMDD_sun357 # Enter the solar data folder for a given observation
```

```
obs@ucc1:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 $ tmux new -s solarProcessing
```

< enter tmux shell >

```
obs@ucc1:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 $ bash enter_docker.sh
```

< Docker container with processing software will load >

```
root@aaaaaaa:/home/user $ cd $LAST_DIR # Your previous directory was saved in $LAST_DIR, go back to to
```

```
root@aaaaaaa:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 # bash process_data.sh # Run the processing script
```

< allow processing to complete, it may take a few seconds for the initial output to appear in ./output_files/ >

< exit the container, tmux shells, notify someone to transfer the data to DIAS >

```
root@aaaaaaa:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 # exit
```

```
obs@ucc1:/mnt/ucc1_recording/data/sun/YYYYMMDD_sun357 $ exit
```

```
obs@ucc1 ~ $ exit
```

Revision #7

Created 21 February 2022 11:44:38 by David

Updated 30 May 2023 15:01:14 by David