

**UPDATE OF THE ENHANCED SINGLE-DISH CONTROL SYSTEM
ESCS 0.2 USER'S GUIDE**

S. Righini, A. Orlati

*INAF-IRA, Istituto Nazionale di Astrofisica
Istituto di Radioastronomia, Bologna, Italy*

IRA 442/11

v.1.6 19-mag-2011

SUMMARY

INTRODUCTION	3
WHAT'S NEW IN ESCS 0.2	4
Still pending issues.....	4
Updates - addition of new hardware devices.....	4
OBSERVER'S DESK	5
ALARM SYSTEM INDUCTION.....	5
ESCS 0.2 USER's GUIDE.....	5
Checklist.....	6
Details on the procedures.....	7
Possible problems and relative hints	17
APPENDIX A – Start from scratch.....	19
APPENDIX B – Monitors and commands.....	20
APPENDIX C – Schedules	25
Caveat: preScan and postScan in OTF LST-based schedules	34
Very important note on Tsys	35
List of useful commands for schedule procedures.....	37
APPENDIX D – FITS files	38

INTRODUCTION

The *Enhanced Single-dish Control System* (ESCS) is a “flavour” of the new control system produced for the Sardinia Radio Telescope which was specifically adapted to the Medicina 32 m dish and its devices. It aims at dramatically improving the single-dish potential of the antenna, providing tools and services not available in the VLBI control system (the so-called *Field System*, FS). ESCS was developed in the *Alma Common Software* (ACS) framework¹, which is optimised for the production of control software to handle complex instruments. Details on the system requirements and structure can be found in previous IRA Technical Reports². The first installation on the 32-m Medicina dish dates back to late 2008.

ESCS 0.1 did not include the frontend management, which was still performed via the FS. This first release was intensively tested and debugged thanks to its employment in scientific programs which required abundant data acquisition, exploiting the new observing modes and opportunities made available:

- fast data acquisition (sampling rate up to 1000 Hz);
- On-The-Fly scans in the main coordinate frames³;
- output files in almost-standard FITS format.

These improvements, coupled to the new hardware devices, allowed the observers to reach an unprecedented sensitivity level and to process their single-dish continuum data in a more straightforward way.

The present document illustrates the new release of the system, effective from March 2011, from the point of view of the typical user. This guide is meant to quickly show how to use the system, including details on schedules and data format, and how to solve the most common problems.

¹ <http://www.eso.org/~almamgr/AlmaAcs/>

² IRA-TechReport: 409-07 (Righini et al.), 423-08 (Orlati et al.), 424-08 (Orlati et al.).

³ IRA-TechReport: 425-08 (Righini).

WHAT'S NEW IN ESCS 0.2

Key updates in the new release are:

- 1) the **receiver** setup and management (Tsys measurements included) are performed by ESCS and do NOT require the use of the Field System;
- 2) **subreflector positioning** is carried out by ESCS;
- 3) **schedules** underwent major updates and allow more options. Old-style schedules will not run in ESCS 0.2;
- 4) the **FITS files** were enlarged in order to contain more information;
- 5) several actions which needed to be performed through the “Object Explorer” panel are available as simple **command lines**;
- 6) the **startup is quicker** and there are less operations to be performed to open all the terminals/monitors;
- 7) every project/group will use a **dedicated account** to access the system and the acquired data.

Still pending issues

ESCS 0.2 does NOT provide:

- subreflector monitoring and direct access to its actuators;
- MBFITS output;
- use of some of the station backends (as the polarimeter);
- derotator handling;
- realtime data display and pointing/focusing tools;
- automatic tools for antenna “wind park” and other safety checks;
- comprehensive GUI.

These tools are at present under development and will be integrated in future releases.

Updates - addition of new hardware devices

This document illustrates the setup for the available hardware devices as for March 2011. Please notice that new receivers and backends are under development and will be implemented in ESCS once they are installed on the antenna, while some active ones will be dismissed in a near future. In addition, **temporary restrictions or changes in the system configuration might hold: enquire the Medicina staff for updates.**

OBSERVER'S DESK

Single-dish observations exploiting ESCS 0.2 are carried out using a **dedicated machine**, located in the building next to the parabola. Remote usage of the ESCS system will be granted to expert users only, since some caveats and possible complications hold.

ALARM SYSTEM INDUCTION

All observers **must** receive a brief induction on the alarm system active in the Medicina station. Please contact s.mariotti@ira.inaf.it or s.righini@ira.inaf.it if you need to be given this training.

ESCS 0.2 USER'S GUIDE

With respect to the first release, ESCS 0.2 allows the user to perform a **faster setup**, as it requires less operations to be executed.

ESCS 0.2 does not need to be started and stopped for every observing session. Users generally find it already running on the observer's machine, thus they only have to activate tools like logfile, system monitors and data quicklook.

A **checklist** of the typical operations is given in the following page.

For instructions on the complete startup procedures (e.g. in case a reboot is necessary), see Appendix A.

Checklist

A. Login

→ use the login and password provided to your project

B. Logfile

→ activate a logfile > logFile *project_DOY.log*

→ open jlog to inspect it > jlog

C. Terminals and monitors

→ start the monitors > escsConsole

→ rearrange windows/monitors

D. Antenna/receiver/backend setup

→ overall setup > 2fs antenna=stow (in separate terminal, only if needed)

 > setupCCC (or other receiver code, in operatorInput)

→ specify Local Oscillator frequency and the bandwidth > setLO=*freq,freq*

 > bck=setSection=*ch,-1,bw,-1,-1,sr,-1*

→ computation of actual HPBW > device=*ch*

→ channel signal level equalization > bck=getTpi

 > bck=getAttenuation

 > bck=setAttenuation=*ch,att*

→ update of .bck file in schedule

→ Tsys measurement > antennaAzEl

 > preset=*Az,El*

 > tsys

E. FS-side monitoring

→ subreflector > 2fs scu

→ weather parameters > 2fs wx

F. Schedules

→ before > antennaTrack (if mount was previously put in *preset* mode)

→ launch > startSchedule=*username/schedulename.scd,N* [*@DOY-HH:MM:SS*]

→ stop > stopSchedule or > haltSchedule

G. FITS files: storage and quicklook

→ new terminal > cd /home/data

→ IDL > idl

 > .r quicklook

 > quicklook [*/mf*] [*xval=xval*]

Details on the procedures

A. Login

Start a session on the PC, log in with the username and password provided to your project/group.

B. Logfile

Open a terminal. Write this command:

```
> logFile project_DOY.log
```

Leave this logfile running, just reduce window to icon. When you need to change the logfile name, restore the terminal, interrupt with CTRL+C and repeat the command with the new filename.

In the same desktop, open a new terminal. Start the graphical logging client:

```
> jlog
```

and set the discard level to “Info” in the relative scroll menu. All system information, including warnings and errors, will be displayed in the bottom left panel. Clicking on a specific item will make details appear in the right panel.

Logfiles are written in the **/home/logs** folder.

C. Terminals and monitors

Open a new terminal, then:

```
> escsConsole
```

This command opens five panels at once: the **operatorInput** terminal and four monitors – **AntennaBoss**, **Observatory**, **Mount**, **TotalPower**. Rearrange them on desktop.

In case any of them does not automatically start, you can manually open them by means of command lines:

```
> operatorInput  
> antennaBossTui  
> observatoryTui  
> mountTui  
> genericBackendTui BACKENDS/TotalPower
```

You will perform all the antenna/receiver/backend setup from the **operatorInput** window, and it will also be used to start/stop the schedules. Monitors display a vast amount of information, see Appendix B for a complete description of their content and a list of all the commands available for the operatorInput.

Use the various virtual desktops according to your taste. It is advisable to display at least the AntennaBoss and Observatory monitors together with the data quicklook (see dedicated section) and the operatorInput, in order to monitor all the most meaningful parameters at a glance.

D. *Antenna/receiver/backend setup*

First of all, if the antenna is not stowed, you need to stow it from the Field System, as ESCS needs the antenna to be stowed before it can control the mount.

Open a terminal and use:

> 2fs antenna=stow

Then close this terminal or use it for the FS monitoring activities described in section E.

You don't need to wait for the antenna to reach the stow position: the command itself clears up the mount status and allows ESCS to take over.

When opening an ESCS observing session, you need to perform a startup which includes the antenna unstow and its configuration in tracking mode. This is done by means of a unique command, which is specific for the wanted receiver, to be used in the **operatorInput**:

> setupCCC or **setupCCCL** or **setupXXP** or **setupKKP** or **setupKKC**

for 5GHz, 8GHz, 22GHz SingleFeed, 22GHz MultiFeed receiver respectively

This command sets the antenna mount, the selected receiver and the total power continuum backend according to **default values**.

In particular, the Local Oscillator frequency and the bandwidth are set to:

Receiver	LO frequency	Frontend IF band (MHz)	Backend filter band (MHz)	Observed bandwidth (MHz)	Observed band (MHz)
CCCL	4600	100 – 250	50 – 250	150	4700 – 4850
CCC	4600	100 – 900	50 – 780	680	4700 – 5380
XXP	8080	100 – 900	50 – 780	680	8180 – 8860
KKP	21964	100 – 900	50 – 780	680	22064 – 22744
KKC	21964	100 – 2100	50 – 2400	2000	22064 – 24064

Notice that, depending on the devices in use, the sky frequency at the observed band starting point is given by the LO frequency plus an offset. For the present combinations of frontend and backend, which the above table refers to, this offset is 100 MHz. In general, the true observed band depends on the **intersection between the frontend IF band and the chosen backend filter**. The actual observed bandwidth and the band starting frequency are recorded in the output FITS files (see Appendix D).

To change the **LO frequency** or the bandwidth, use the following commands:

> setLO=freq1;freq2

notice the semicolon. Ideally, different values could be assigned to different IFs, thus tuning each channel to a different sub-band. For the present hardware, though, this is not possible: use a single value, e.g. **> setLO=4900;4900**

The XXP receiver is NOT tunable: only the default LO value is available!

Bandwidth and **sampling rate** can be changed like this (pay attention to the ‘-1’ values):

> bck=setSection=ch,-1,bw,-1,-1,sampleRate,-1

where *ch* is an integer specifying the channel number, *bw* is a double for the bandwidth (MHz) and *sampleRate* is also given in MHz.

bw is the backend filter bandwidth (MHz). It can be chosen from a restricted range of options:

- 300.0
- 730.0
- 1250.0 (for single feed receivers, the true bandwidth will be narrower)
- 2350.0 (for single feed receivers, the true bandwidth will be narrower)

These values do NOT correspond to the true observed bandwidth, for the reason discussed above. Notice in particular that only the 22 GHz Multi-Feed receiver can fully exploit a backend filter larger than 730 MHz, since all the other receivers have a narrower bandwidth.

If you are going to manually get the Tpi - in order to ascertain which attenuation values to use in your schedule - remember to set the LO frequency and the bandwidth as they will be employed in the schedule. After a setup command, in fact, they are set to defaults; instead if a schedule was previously run these values remain set as indicated in the last .bck and .cfg files reading.

Since the beamsize is a function of the observed frequency, changing the LO and bandwidth values implies its modification. It is necessary to explicitly **re-compute the HPBW** (Half Power Beam Width) by means of the command:

```
> device=ch
```

where *ch* is the number of the channel to be used as a reference (in our case, all channels will have the same LO frequency and bandwidth, so choose whichever you want). If this “fine tuning” of the HPBW is not performed, the system will consider its default value when estimating if the antenna is tracking correctly.

The initial setup command (setupCCC, etc...) also implies the upload of the correct pointing model, and the repositioning of the **subreflector**. To check whether the subreflector successfully completed its positioning, see the Field System operations in the following section.

It is now time to **check the signal levels** for the various channels and, if necessary, bring them to the proper range. To achieve average values, position the antenna at a fixed elevation of 45°.

Pointing the antenna to a **fixed Az-El position** is obtained by means of:

```
> antennaAzEl
```

which sets the mount mode to “fixed position”, then

```
> preset=Az,El
```

to specify the horizontal coordinates, like:

```
> preset=180d,45d (notice the “d” for degrees)
```

When the antenna has reached the destination, again in the **operatorInput**:

```
> bck=getTpi
```

It will return the signal levels in arbitrary counts, for all active channels (2 for the single feed receivers, 14 for the 22GHz Multi Feed). To ensure that the backend is working in its linear regime, these levels must stay in the 700-1300 counts range. It is advisable to equalize all the channels at ~1000 counts.

To achieve this, act on the attenuators: an attenuation from 0.0 dB to 15.0 dB can be applied to each channel. Default at startup is 9.0 dB. Instead, if a schedule was run and no setup command has been given again, each channel shows a different value.

So, first read the present attenuation values:

```
> bck=getAttenuation
```

It will return the list of values, for the active channels in increasing order (0,1,...,14).

As a reference, consider that decreasing the attenuation by 3.0 dB means doubling the signal level, increasing the attenuation by 3.0 dB means reducing the signal level to half the original value.

To **set a new attenuation** for a given channel:

```
> bck=setAttenuation=ch,att
```

Again, *ch* is the integer for the channel number and *att* is the attenuation in dB.

Iterate in reading the values with getTpi and changing the attenuations until all channels reach an acceptable signal level. Then, insert the final attenuations inside the .bck file of the schedule.

Each schedule, as shown in detail in Appendix C, is composed by 4 files, all of which must be present in the local folder `/home/schedules`. The .bck files contain the backend configuration, something like:

```
STD:BACKENDS/TotalPower {  
    integration=40  
    setSection=0,-1,300.0,-1,-1,0.000025,-1  
    setSection=1,-1,300.0,-1,-1,0.000025,-1  
    setAttenuation=0,5.0  
    setAttenuation=1,13.0  
    enable=1;1  
}
```

Attenuation values are the second ones after the setAttenuation commands (shown in blue in the above example, which is relative to single feed observations). Remember to update also the bandwidth and the sampling rate in the setSection lines, if you changed them during the previous setup phase.

Now, always at 45° of elevation (unless you have specific needs for a different position), take a reference T_{sys} to assess if the receiver is working correctly and estimate the observing conditions.

```
> tsys
```

It will return a T_{sys} value (K) for each channel.

Nominal values @El=45° for good weather conditions are:

@ 5 GHz → 29 K

@ 8 GHz → 40 K

@ 22 GHz Single Feed → > 150 K (as this old receiver is not optimally operating⁴)

@ 22 GHz Multi Feed → 70 K (in very good weather conditions it can be lower)

If T_{sys} is way above these values, try in another azimuth position⁵. If it is still so, the receiver might not be properly cooled: **ask for help!**

Pay attention: **never point the antenna to the Sun** or its proximities. Azimuth goes from 0° to 360°, counting from North and going Eastwards. Elevation goes from 0° (Horizon level) to 90° (Zenith).

NOTICE: the last measured T_{sys} value will be stored in the system and used to get a counts-to-Kelvin conversion factor, in turn applied to all the following acquisitions, until a new T_{sys} is measured. The FITS file (Appendix D) will contain the raw data (in counts) and also a table with the data stream calibrated (in K) using this counts-to-Kelvin factor.

CHANGING THE RECEIVER

To **change the receiver**, simply use a new setup command and follow again all the steps.

> **setupCCC** or **setupXXP**, **setupKKP**, **setupKKC**

E. *FS-side monitoring*

There are parameters which must still be checked using the old control system: the subreflector position and the weather parameters. Actually, the latter are visible also in ESCS, but it is quicker to read them using the Field System.

To read the **subreflector** encoders:

> **2fs** **scu** response is like scu/connected,fixed,primary,xxp,ok, (numbers)
 scu/connected,tracking,secondary,ccc,ok, (numbers)
 scu/connected,tracking,secondary,kkc,ok, (numbers)⁶

⁴ We still list it among the available receivers since, once the MF is transferred to SRT and while waiting for the new K-band dual-feed to be ready, it will be the only 22 GHz receiver available in Medicina.

⁵ RFI (man-made interferences) are azimuth-related and might affect your measurement.

⁶ It is important to be sure the subreflector is tracking correctly at high frequency: its position varies with the target elevation, so the numbers at the end of the 'scu' reply will change accordingly. If the subreflector does not track correctly, the targets will NOT be observed!

A commutation from primary to secondary focus – or viceversa – can take up to 8 minutes, during which the readouts are not available. Once the slewing is over, you might still get a “**nok**” (not ok) within the reply. Wait one more minute and try again. If the problem persists, or if the subreflector is “**disconnected**”, call for assistance.

During 8GHz (primary focus) observations, the subreflector is not involved in any action, so once it has reached the correct initial position you can ignore it.

Anytime you want to have a look at the **weather parameters**:

```
> 2fs wx
```

Response will be like:

```
2011.030.07:46:02.04/wx/5.4,1013.6,45.4,15.0
```

the numbers after /wx/ are Temperature (° C), Pressure (hPa), Relative Humidity (%) and Wind Speed (km/h), measured at ground level. While the first three weather parameters are stored into the output file, wind speed is not. So keep track of it on your logbook, and most of all check if its value stays below 50 km/h. Above this speed, the wind spoils the pointing accuracy and, if going above 65 km/h, it can damage the antenna → **stop the schedule and stow the antenna if you find too high a wind speed!**

E. *Schedules*

When the preliminary setup has been completed and the optimal configuration has been checked and updated in the .bck file, observations can start.

You can parse the schedule, in order to check whether it contains formal errors or inconsistencies, opening a new terminal and using:

```
> scheduleChecker username/schedulename.scd
```

If the antenna was in Az,El mode, restore it to tracking mode with the following command in the **operatorInput**:

```
> antennaTrack
```

To launch a schedule, then use the following command:

```
> startSchedule=username/schedulename.scd,N
```

“username” is the one you used to login to the machine, and it is unique for every project. Then follows the schedule name, in particular the name of the .scd file (see Appendix C). “N” is the schedule ID number from which ESCS must start reading it – it is particularly

useful in case of a sequential (i.e. not time-based) schedule.

You can specify **when to start** the schedule (UT):

> **startSchedule=***username/schedulename.scd,N@DOY-HH:MM:SS*

ESCS reads the configuration parameters from the schedule, which can be relative both to the receiver and the backend (see Appendix C), and accordingly sets these devices. This might take several seconds, especially when using the Multi Feed receiver.

While the setup takes place, several values change in the TotalPower monitor.

The last operation is the upload of the first pointing/scan read from the schedule, whose parameters will show up in the bottom section of the AntennaBoss monitor.

During the scans, all the three flags in the lower part of the AntennaBoss monitor must be a green “@”. Pay attention to the second one in particular (“Tracking”). It should turn to a red “o” only when the antenna is slewing between scans on the same source, or when slewing to/from a new source. **If tracking is not correct for some consecutive seconds during a scan, something is wrong.**

To abruptly interrupt the running schedule, truncating the ongoing acquisition:

> **stopSchedule**

Instead, to stop the schedule allowing the completion of the present file:

> **haltSchedule**

F. *FITS files: storage and quicklook*

Each subscan indicated in the schedule generates a FITS file, which is stored in **/home/data**. This is a folder directly pointing to the original location where data is being written, on a different machine. Files can be opened but not edited: they are in readonly mode. Users are responsible of copying their files locally (or store them in a backup place, even if a backup is also provided by the system) in case they need to edit them.

FITS filenames are built like:

Date - UT - LST - Project_label.fits

Example: 20110212-100934-163258-MyProject_Scan001.fits

The project and label parts come from the .scd file keywords.

Once a schedule is completed, it is advisable to create a new subfolder and move the relative files there.

The **data quicklook** is at present external to ESCS and does not allow interactive operations: it consists of a set of IDL procedures conceived in order to automatically display the acquire data together with some basic information and quantities.

To display your data, open a new terminal and go to folder **/home/data**.

Then start IDL:

```
> idl
```

Compile the program

```
> .r quicklook
```

Run the program

```
> quicklook [xval=xval]          for single-feed acquisitions
```

```
> quicklook, /mf [xval=xval]    for 22GHz multi-feed receiver
```

The program displays on screen the feed0 data of the second-last FITS file in the folder, which is for sure complete (the very last one can be incomplete, if it is being written at the time of the file copy). **Quicklook is automatically updated every 5 seconds.**

The x-axis can be optionally shown as:

sample number (xval='s', It is the default value)

elapsed time from the scan start (xval='t')

azimuth degrees (xval='a')

elevation degrees (xval='e')

declination degrees (xval='d')

right ascension hh.hhh ('xval='r').

Examples.

```
> quicklook, /mf, xval='a'  shows central feed data as counts vs azimuth
```

```
> quicklook, xval='t'      shows single feed data as counts vs elapsed time
```

If you want to **plot all the FITS files in a PS file for batch visual inspection**, compile another program:

```
> .r fitsplotter
```

Run the program

```
> fitsplotter [/mf] (for a basic on-line help launch > fitsplotter, help)
```

Open the output file generated by IDL

```
> $ gv fitsplots.ps
```

This file will show the content of all the channels, for every FITS file in the working folder. Each page shows the plots relative to two subscans, one for each row. *Rename the PS file if you want to keep it, otherwise it will be overwritten when fitsplotter runs again*

in that folder (unless you specifically give an output filename, see the above mentioned on-line help).

If you find an interesting file, you can zoom on it (or plot it against different variables), using the IDL program called *chN*, which opens and plots (or prints) only the Nth channel of a specified file, using the following syntax for the command line to be given inside IDL:

```
> chN, fin='filename.fits', N=chN, A=firstsample, B=lastsample, xtype='xtype', out='out'
```

It reads the given .fits file in the present path and plots the content of the data relative to channel N (for CHO use N=14). With N=15 and N=16 you get plots involving the scan coordinates only. If A and B are specified, the plot will zoom on the sample interval [A:B]. Default is to plot the entire file. The x-axis can be shown as (default is time):

- sample number (if xtype='s')
- elapsed time from the scan start (if xtype='t')
- azimuth degrees (if xtype='a')
- elevation degrees (if xtype='e')
- declination degrees (if xtype='d')
- right ascension hh.hhh (if 'xtype'='r').

Output can be on screen (out='s') or on a PS file (out='f'). Default is screen.

Several other basic procedures are available in order to plot the files. There are tools also to perform some preliminary operations like Gaussian fits, etc... They are stored in the main IDL library folder.

Possible problems and relative hints

I've launched a schedule but it was not uploaded by the system.

- Have you parsed the schedule using `scheduleChecker` to ascertain it did not contain errors or inconsistencies?
- Did you write the line number "N" at the end of the `startSchedule` command?
- Is the schedule name correct, including *username* and the `.scd` extension?
- Are all the schedule files in the correct folder `/home/schedules`?
- Maybe some file access/permission issues hold. Ask for assistance.

Some scans were skipped. What happened?

- Look at the jlog panel. Does it say "Late scan skipped"? If so, the scans were not performed because the antenna could not reach the start point in time.
- Was the target elevation too high? If so, the antenna cannot perform the scans because the azimuth rate would exceed the upper limit.
- Is the antenna in STOP mode? Some failures might have happened and the antenna is now blocked. If so, try to understand what's wrong and, as a first attempt, give an `> antennaSetup=Code` (CCC, XXP, KKC...) in the `operatorInput` terminal.

I've launched a schedule, it gets uploaded but then it stops automatically, and I see a "Schedule HALTED" or a "AZ rate too high" error in the jlog Logging Client.

- Have you parsed the schedule using `scheduleChecker` to ascertain it did not contain errors?
- Check the commanded coordinates in the top lines of the AntennaBoss monitor. Is the commanded elevation very high? In this case, the resulting scan azimuth rate is too high (it tends to infinite as the source approaches the zenith), as the Logging Client should also suggest, and maybe this caused a system failure (even if it should not). Stop the schedule and restart it.

I've launched a schedule, it gets uploaded and the antenna reaches the first commanded position, but then nothing happens.

- Is the schedule LST-based? If so, did you check the LST range of the schedule?
The LST times of the schedule are probably all elapsed for today, and the antenna is waiting for the LST of the first scan of the schedule (which will take place again...

tomorrow!).

The quicklook of the data is frozen on the same file.

- Is the schedule still running?

Look at the AntennaBoss monitor and see if the scans are being performed (the coordinates change very fast, if so). If the antenna is not moving, something is wrong with the schedule or the mount. Check if the Mount monitor shows a STOP status, in which case there is a failure: stop the schedule and give a

> **antennaSetup=Code** (CCC, XXP, KKC...) in the operatorInput terminal.

If, instead, the schedule is running, check if the files are being written in /home/data. If they are not updated, the FitsZilla component or the Storage Container are probably down and you need to restart the session.

I've launched a schedule, it runs correctly but I don't see any source in the scans.

- Is the subreflector in the correct position? Check it via the FS using the 'scu' command.
- Did you correctly set the LO frequency and the bandwidth? If the source is weak, maybe you cannot see it in a single scan (especially in the bandwidth is narrow).

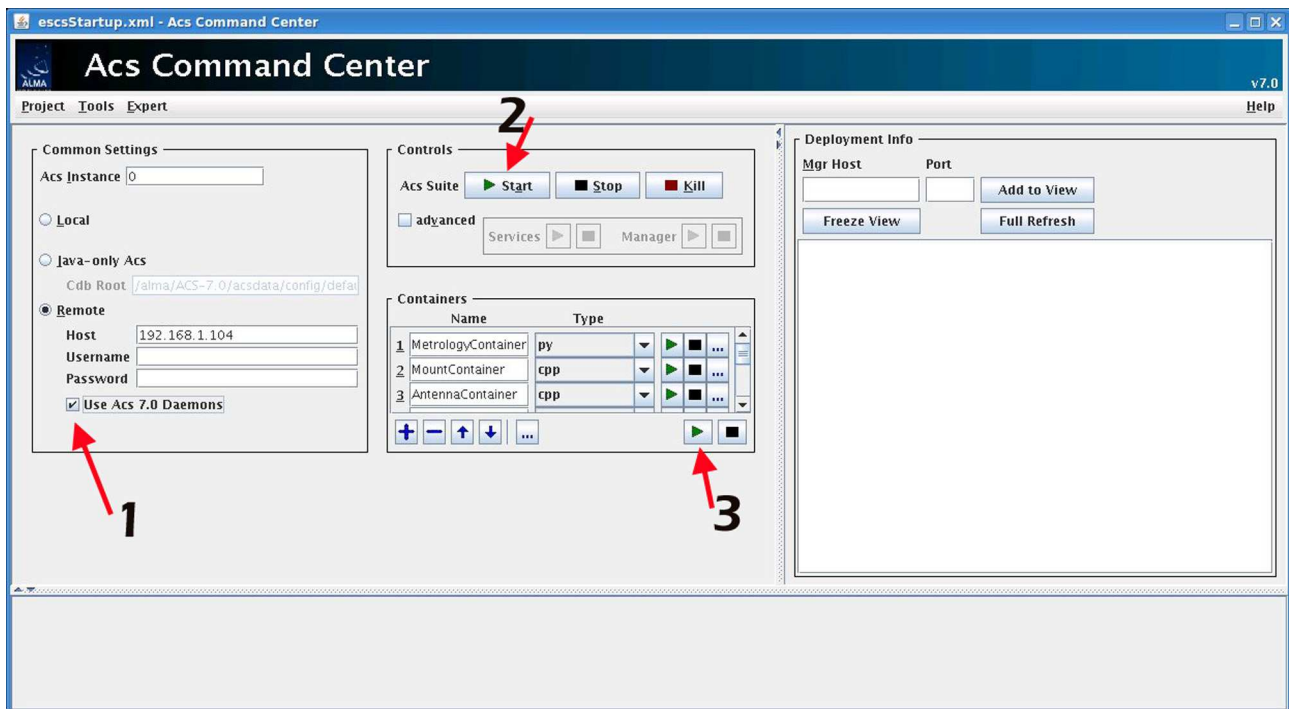
APPENDIX A – Start from scratch

Login as usual, with the name and password given to your project/group.
Open a terminal:

> **ssh -X manager@192.167.189.104** (ask system manager for password)

> **escs --start** (--help to see the options)

The ACScommandcenter panel will appear.



Select the “Use ACS Daemons” tick (number **1** in the above figure) in the bottom left-hand section (no need to insert a password).

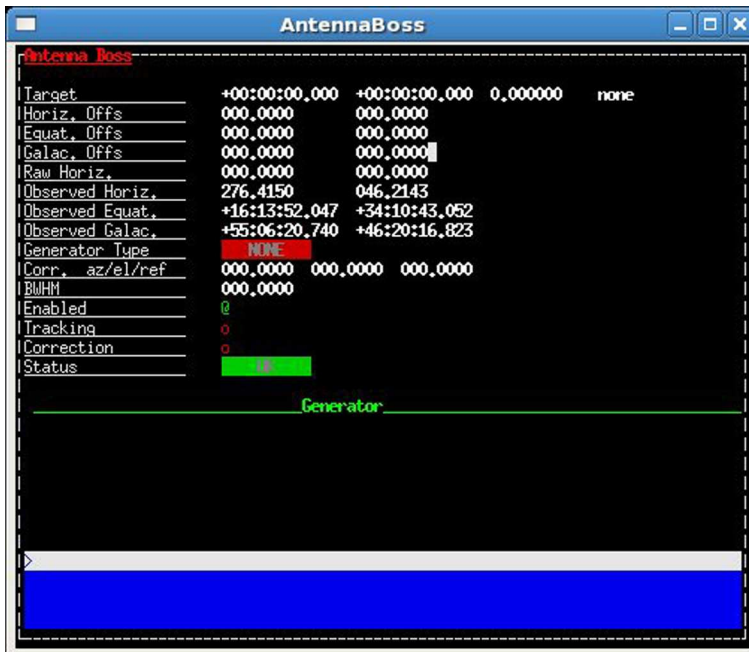
Click on the “Start” button in the top-centre section (**2**). Wait for the operations to complete; a series of items will appear in the right-hand part of the window.

In the bottom-centre section, click on the green arrow under the container list (**3**): this is the command to start them all.

Containers are 8: Metrology, Mount, Antenna, Receivers, TotalPower, NoiseGenerator, Management, FitsZilla. Their names will appear in the right panel as they get activated. Once they are all there, go on with the procedures as described in the main guide.

APPENDIX B – Monitors and commands

AntennaBoss



AntennaBoss				
Target	+00:00:00.000	+00:00:00.000	0.000000	none
Horiz. Offs	000.0000	000.0000		
Equat. Offs	000.0000	000.0000		
Galac. Offs	000.0000	000.0000		
Raw Horiz.	000.0000	000.0000		
Observed Horiz.	276.4150	046.2143		
Observed Equat.	+16:13:52.047	+34:10:43.052		
Observed Galac.	+55:06:20.740	+46:20:16.823		
Generator Type	NONE			
Corr. az/el/ref	000.0000	000.0000	000.0000	
BWHM	000.0000			
Enabled	@			
Tracking	o			
Correction	o			
Status	OK			
Generator				

This monitor shows the commanded and actual positions and gives a feedback on the pointing accuracy. The figure shows how it looks at startup.

Parameters list:

Target: coordinates of the target source (RAJ2000.0, DecJ2000.0, vlscr, target name), in case of an OTF scan they are relative to its central position.

Horiz. Offs: Horizontal (Az-El) offsets as read from schedule, degrees.

Equat. Offs: Equatorial (RA-Dec) offsets as read from schedule, degrees.

Galac. Offs: Galactic (l-b) offsets as read from schedule, degrees.

Raw Horizontal: commanded Az-El, including pointing model, refraction, etc...

Observed Horizontal: Az-El coordinates read from the mount encoders and cleaned from the pointing model and refraction contributions. Because of this, observed coordinates will differ from the raw ones.

Observed Equatorial: RA-Dec J2000.0 coordinates, converted from the observed horizontal

Observed Galactic: l-b Galactic coordinates, converted from the observed horizontal.

Generator Type: which component is in charge of the generation of the coordinates. It can be NONE (which is the condition at startup), OTF, MOON or SIDEREAL.

Corr. az/el/ref: azimuth and elevation corrections (degrees) applied by the pointing model, plus the refraction model contribution– which is an additional correction to elevation.

BWHM: Beam Width Half Maximum (corresponding to HPBW), degrees.

Enabled: a green "@" indicates that the antenna is correctly receiving commands; a red "o" means the communication is disabled.

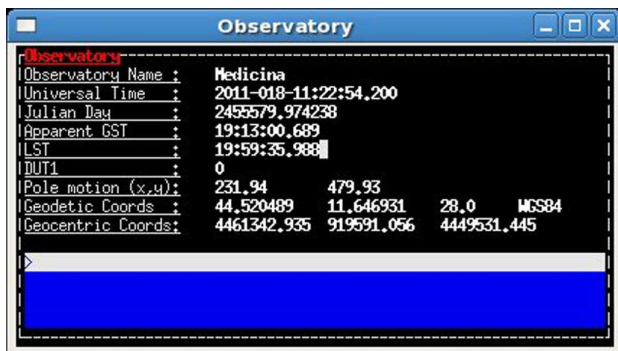
Tracking: it indicates whether the pointing error exceeds $0.1 \cdot \text{BWHM}$ or not. A green "@" corresponds to error $< 0.1 \cdot \text{BWHM}$. It should turn to a red "o" only when the antenna is slewing between scans on the same source, or when slewing to/from a new source.

Correction: application of the above horizontal coordinates corrections. If disabled (red circle), all corrections are zeroed.

Status: OK, WARNING or ERROR. "Warning" needs investigation but usually does not stop the ongoing activity (it also appears at startup, before the setup commands), "Error" generally appears if something stops the observations.

Generator: under this line all the scan setup parameters appear when it is commanded.

Observatory



It is devoted to the station coordinates and times.

Observatory Name: Medicina

Universal Time: YYYY-DOY-HH:MM:SS.SSS

Julian Day: d.ddd

Apparent GST: Greenwich Sidereal Time
HH:MM:SS.SSS

LST: Local Sidereal Time HH:MM:SS.SSS

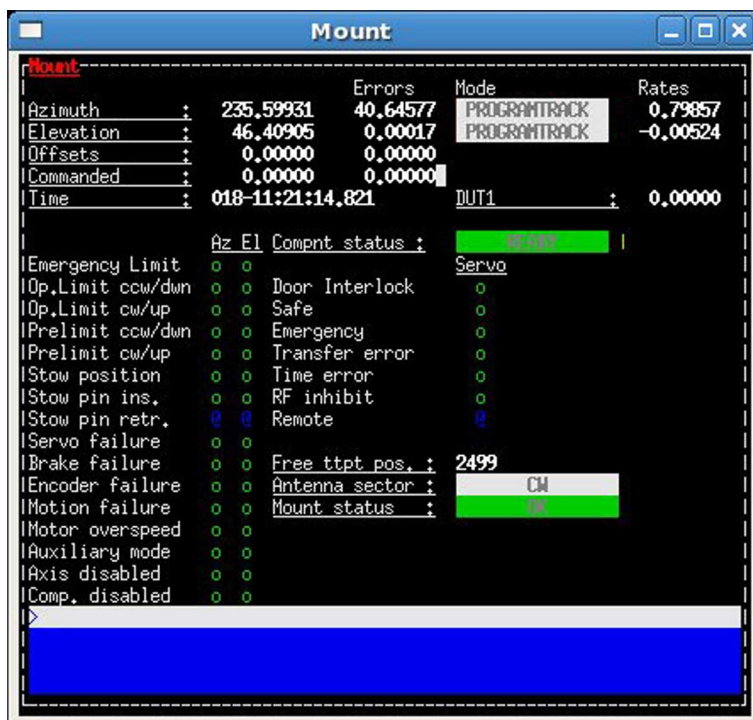
DUT1: difference between UT1 and UT (s), if applied.

Pole motion: celestial pole offset wrt a reference position (details are note provided here), measured in μ arcsec on a tangential projection.

Geodetic Coords: updated Latitude and Longitude (degrees) and Altitude (m) for the Medicina dish, plus the geodetic model code.

Geocentric Coords: geocentric cartesian coordinates (m) for the Medicina dish.

Mount



Top section shows the encoder **Az-El** coordinates and the relative position error (i.e. the difference between commanded and encoder coordinates), in degrees.

Offsets: Az-El offsets as defined by user.

Mode: it can be PRESET (fixed Az-El pointing), PROGRAMTRACK (for schedules), STOP (if axes brakes are on), UNKNOWN (usually indicating an error condition).

Rates: instant Az-El speeds in $^{\circ}/s$

This monitor shows many status flags, which would need many technical explanations to be understood. For average users, the only points to be taken into account are:

- in standard observing conditions, when a schedule runs, all flags should be green "o", with the exception of the **Remote** one, which should be a blue "@"
- immediate actions must be taken when a "failure" keyword turns steadily to red "o".

GenericBackend

GenericBackend: NOGENIS/TotalPower

Time : 2011-018-11:20:22.000 Integration : 0 Busy : 0 Time_Sync 0

Section : 14 Inputs : 14 Busy 0

	Freq	BW	Feed	S.R.	Pol	Bins	DBs	Sect	Tsus	Busy	Suspended	Sampling	CmdLine	DataLine
1900	0050.0	0300.0	02	2.50000e-05	LEFT	00001	109	11.0	00	0000.0	0	0		
1901	0050.0	0300.0	02	2.50000e-05	RIGHT	00001	101	11.0	01	0000.0	0	0		
1902	0050.0	2200.0	03	2.50000e-05	LEFT	00001	102	09.0	02	0000.0	0	0		
1903	0050.0	2200.0	03	2.50000e-05	RIGHT	00001	103	09.0	03	0000.0	0	0		
1904	0050.0	2200.0	04	2.50000e-05	LEFT	00001	104	09.0	04	0000.0	0	0		
1905	0050.0	2200.0	04	2.50000e-05	RIGHT	00001	105	09.0	05	0000.0	0	0		
1906	0050.0	2200.0	05	2.50000e-05	LEFT	00001	106	09.0	06	0000.0	0	0		
1907	0050.0	2200.0	05	2.50000e-05	RIGHT	00001	107	09.0	07	0000.0	0	0		
1908	0050.0	2200.0	06	2.50000e-05	LEFT	00001	108	09.0	08	0000.0	0	0		
1909	0050.0	2200.0	06	2.50000e-05	RIGHT	00001	109	09.0	09	0000.0	0	0		
1910	0050.0	2200.0	00	2.50000e-05	LEFT	00001	110	09.0	10	0000.0	0	0		
1911	0050.0	2200.0	00	2.50000e-05	RIGHT	00001	111	09.0	11	0000.0	0	0		
1912	0050.0	2200.0	01	2.50000e-05	LEFT	00001	112	09.0	12	0000.0	0	0		
1913	0050.0	2200.0	01	2.50000e-05	RIGHT	00001	113	09.0	13	0000.0	0	0		
1914							114							
1915							115							
1916							116							
1917							117							
1918							118							

The panel shows one row for every channel (here “section”).

Ignoring the specific numbers used in the figure above, the column meanings are:

Freq: value (MHz) to be added to the LO frequency in order to obtain the observed frequency at the beginning of the band

BW: bandwidth (MHz)

Feed: number of the receiver feed connected to this channel

S.R.: sampling rate (MHz)

Pol: polarization (Left or Right)

Bins: number of frequency bins (1 for total power)

DBs: attenuation (dB) applied to the channel

Sect: channel/section number

Tsys: the last measured Tsys (K)

Also some status flags are present, in the top right area. Pay attention only to

Time_Sync: if it frequently or steadily turns red call for assistance (the backend time is not synchronized)

Busy: when schedules are running, it must turn yellow. If it does not, the backend is not acquiring.

List of the available commands to be used in the operatorInput terminal (and in schedules)

NOTICE: all commands can be temporised adding the suffix "@DOY-HH:MM:SS", where DOY is the Day-Of-Year (1-366) and HH:MM:SS is the UT time.

They all can be used also in the init/pre-scan/post-scan procedures inside schedules. Observers are in charge of considering *if and when* the use of a certain command makes sense in their schedule, according to their specific needs and goals: this is something that no schedule parser can check.

- > **setupCCC** (or setupXXP or setupKKC *)
unstows the antenna, sets it to tracking mode, selects the pointing model, and configures the receiver and the backend using default parameters
- > **antennaSetup=Code** (CCC, XXP...)
unstows the antenna, sets it to tracking mode and configures the pointing model according to the specified receiver. It does NOT perform the receiver and backend setup
- > **antennaStop**
stops the antenna. Activities can start again only commanding a new setup
- > **antennaPark**
sends the antenna to stow position
- > **antennaAzEl**
sets the antenna to PRESET mode (fixed Az-El pointings). To enable tracking again, e.g. to run schedules, an antennaTrack or antennaSetup must follow
- > **preset=double'd',double'd'**
sends the antenna, if in PRESET mode, to the specified Az-El position.
Example: preset=180d,45d
- > **antennaTrack**
sets the antenna to PROGRAMTRACK mode. It does not change the pointing model or any receiver setup
- > **azelOffsets=double'd',double'd'**
sets the Az-El offsets (degrees). They are intended "on sky".
Example: azelOffsets=-0.05d,0.05d
- > **radecOffsets=double'd',double'd'**
sets the RA-Dec offsets (degrees). They are intended "on sky".
Example: radecOffsets=1.0d,0.0d
- > **lonlatOffsets=double'd',double'd'**
sets the b-l offsets (degrees). They are intended "on sky".
Example: lonlatOffsets=2.0d,-1.0d
- > **source=sourcename**
points the antenna, in sidereal tracking, to the specified source, which must be present in the local catalogue. If you need to insert frequently observed sources in this catalogue, contact the system manager
- > **moon**
points the antenna to the present coordinates of the center of the Moon

(*) in practice, it is a shortcut corresponding to this sequence:
antennaSetup=Code, receiversSetup=receiverCode,
bck=initialize=receiverCode, device=0, calOff

- > **receiversSetup=receiverCode** (KKC, XXP, CCC, ecc...)
 - configures the receiver using the default parameters. It does NOT act on the backend, pointing model or antenna mount mode
- > **setLO=freq;freq**
 - Local Oscillator frequency, in MHz (one per channel, usually the values are identical)
 - This LO frequency corresponds to: SkyFreq(@band start) – 100 MHz
- > **tsys**
 - measures the system temperature (K) in the position the antenna is pointing to. It returns a list of values, one for each channel in use (e.g. 14 values for the whole Multi Feed receiver). All the intermediate steps and calculations are stored in the active logfile
- > **calOn**
 - switches the calibration mark on
- > **calOff**
 - switches the calibration mark off
- > **bck=initialize=receiverCode** (KKC, XXP, CCC, ecc...)
 - configures the backend using the default parameters relative to the selected receiver. It does NOT act on the receiver, pointing model or antenna mount mode
- > **bck=getAttenuations**
 - reads the attenuation values (dB) currently configured for the active channels, and lists them according to increasing channel number
- > **bck=setAttenuation=ch,att**
 - sets to *att* (dB) the attenuator of channel *ch*
- > **bck=getTpi**
 - reads the signal intensity (counts) for the active channels, and lists them according to increasing channel number
- > **device=ch**
 - computes the beamsize, taking into account the present receiver and backend configurations relative to channel *ch*
- > **startSchedule=username/schedulename.scd,N**
 - runs schedule *schedulename.scd* (*username* is the login user), reading it from line *N*
- > **stopSchedule**
 - immediately stops the running schedule, truncating the acquisition
- > **haltSchedule**
 - completes the current scan and then stops the schedule

APPENDIX C – Schedules

Introduction: how scans work in ESCS

In ACS-ESCS we call “component” each system element which is in charge of a specific task. The component named **OTF** is responsible of computing the sequence of pointings which constitute a single on-the-fly subscan.

It gets the input values from a schedule, which is directly written by the user or can be produced by an ad hoc program.

In addition to the configuration constraints, OTF obtains other parameters from *AntennaBoss*, the component which manages the antenna pointing and calls OTF. The following table lists the input parameters needed to set a single subscan and passed as arguments of the method called *setSubScan*.

initAz	Azimuth (radians) where the antenna is located when OTF is called. This is automatically obtained from AntennaBoss.
initSector	CW or CCW relative to the initAz value. This is automatically obtained from AntennaBoss.
initEl	Elevation (radians) where the antenna is located when OTF is called. This is automatically obtained from AntennaBoss.
initTime	UT instant (100 ns since 1582-10-15 00:00:00) when the OTF is called. This is automatically obtained from AntennaBoss.
lon1	If the description is start-stop, longitude (radians) of the constant speed scan starting point. Otherwise it corresponds to the longitude (radians) of the subscan central point. Value is obtained from user input data.
lat1	If the description is start-stop, latitude (radians) of the constant speed scan starting point. Otherwise it corresponds to the latitude (radians) of the subscan central point. Value is obtained from user input data.
lon2	If the description is start-stop, longitude (radians) of the constant speed scan stopping point. Otherwise it corresponds to the full span in longitude (radians) of the subscan. Value is obtained from user input data.
lat2	If the description is start-stop, latitude (radians) of the constant speed scan stopping point. Otherwise it corresponds to the full span in latitude (radians) of the subscan. Value is obtained from user input data.
coordFrame	Reference frame for the input coordinates. It can be equatorial J2000.0, Galactic or horizontal. Value is user-defined.
geometry	Subscan geometry, it can be constant longitude, constant latitude or great circle. Value is user-defined.
subScanFrame	Reference frame for the execution of the subscan. It can be equatorial J2000.0, Galactic or horizontal. Value is user-defined.
description	Subscan definition. It can be start/stop or centre/span. Value is user-defined. Centre/span description does not apply to great circle scans.
direction	Increase or decrease, relative to the varying coordinate. Does not apply to great circle subscans.
startUT	UT instant (100 ns since 1582-10-15 00:00:00) at which the data acquisition is commanded to start. It coincides with the instant when the constant speed scanning starts. Value is user-defined.
subScanDuration	Duration of the constant speed scanning (s). Value is obtained from user input data.

The component also reads the following values, specific for the telescope in use, from the Configuration DataBase (CDB). Some of these parameters were determined for the Medicina dish by means of on-field tests:

- ⊕ Maximum azimuth slewing rate allowed by mechanical constraints ($^{\circ}/s$);
- ⊕ Maximum elevation slewing rate allowed by mechanical constraints ($^{\circ}/s$);
- ⊕ Maximum azimuth scan rate allowing correct pointing ($^{\circ}/s$);
- ⊕ Maximum elevation scan rate allowing correct pointing ($^{\circ}/s$);
- ⊕ Maximum azimuth acceleration ($^{\circ}/s^2$);
- ⊕ Maximum elevation acceleration ($^{\circ}/s^2$);
- ⊕ Acceleration "Scale Factor" – scaling the az-el acceleration in order to perform smooth acceleration/deceleration ramps before/after the constant speed scan;
- ⊕ Site information (longitude, latitude, elevation, etc...);
- ⊕ Observation information (HPBW, DUT1, etc...).

Once the initialisation and setup are complete, OTF controls if the user-defined parameters are correct. In particular it checks that:

- ⊕ values match with the available options (e.g. the coordFrame value cannot be "ecliptic" because this coordinate frame is not implemented);
- ⊕ values are consistent with each other. For example, a great circle geometry is not compatible with the centre/span description;
- ⊕ the azimuth and elevation rates stay below the allowed limits during the whole subscan. This is particularly critical for great circle scans, since they can imply very high values for the rates in restricted sections of their curved paths along the sky – for this reason, the whole path is simulated in advance. In case the az-el speeds exceed the instrumental limits the program raises an error and ends. If, instead, the speed is below the maximum values but exceeds the limits which have been tested to be reliable for a correct pointing, then a warning is raised.

The next step is the computation of the acceleration/deceleration ramps to be appended to the constant speed scan. The user-defined start coordinates and UT time are, in fact, relative to the beginning of the constant speed path. OTF automatically computes the ramps in order to match those constraints, using the acceleration and scale factor values written in the CDB to compute a uniformly accelerated route. The coordinates and UT time relative to the first point of the acceleration ramp are commanded to start the antenna motion.

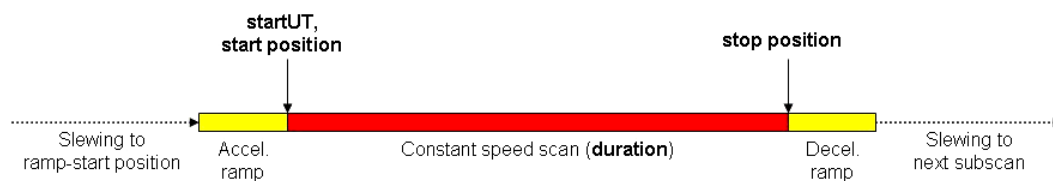


Fig. 1 Schematisation of the timeline/path of a subscan. The antenna system defines, elaborating the user-input information or reading the schedule, the subscan boundaries and duration (bold parameters), referring to the constant speed scan (red line), which are passed to the OTF component. This adds the acceleration/deceleration ramps (yellow lines), computing their start and stop positions and the relative timings. The initial ramp start position is commanded to the antenna, which slews to those coordinates, waits for the ramp start UT time, runs the accelerated path and begins the "constant speed path" at the correct time and position.

OTF then checks whether this ramp-start position can be reached in time: when called, it is passed the current UT instant and antenna position and computes if the necessary antenna slewing time is short enough to reach the ramp start coordinates within the commanded UT time. If the target is too far, a warning is raised and the general system (ESCS) cross-checks the impossibility to reach the commanded position; if the problem is confirmed, the subscan aborts and the system skips to the next one in the schedule (if any).

At this point, everything is set and the OTF component is asked to pre-compute the sequence of pointings composing the subscan, and also to check if the actual motion correctly follows the predicted positions.

Schedule structure

A schedule consists of 4 files, organised in order to **setup the frontend and backend** and to command the execution of a list of OTF subscans, which can be LST-based (temporised) or simply sequential.

All files must be present in the /archive/schedules folder on the 192.167.189.104 machine.

FileName.bck

This file lists the configuration parameters for the backend.

The content is organised as in the following example (for MultiFeed observations):

```
STD:BACKENDS/TotalPower {
    integration=40
    calSwitch=0
    setSection=0,-1,300.0,-1,-1,0.000025,-1
    setSection=1,-1,300.0,-1,-1,0.000025,-1
    setSection=2,-1,300.0,-1,-1,0.000025,-1
    setSection=3,-1,300.0,-1,-1,0.000025,-1
    setSection=4,-1,300.0,-1,-1,0.000025,-1
    setSection=5,-1,300.0,-1,-1,0.000025,-1
    setSection=6,-1,300.0,-1,-1,0.000025,-1
    setSection=7,-1,300.0,-1,-1,0.000025,-1
    setSection=8,-1,300.0,-1,-1,0.000025,-1
    setSection=9,-1,300.0,-1,-1,0.000025,-1
    setSection=10,-1,300.0,-1,-1,0.000025,-1
    setSection=11,-1,300.0,-1,-1,0.000025,-1
    setSection=12,-1,300.0,-1,-1,0.000025,-1
    setSection=13,-1,300.0,-1,-1,0.000025,-1
    setAttenuation=0,5.0
    setAttenuation=1,13.0
    setAttenuation=2,11.0
    setAttenuation=3,9.0
    setAttenuation=4,9.0
    setAttenuation=5,9.0
    setAttenuation=6,10.0
    setAttenuation=7,8.0
    setAttenuation=8,7.0
    setAttenuation=9,11.0
    setAttenuation=10,11.0
    setAttenuation=11,12.0
    setAttenuation=12,14.0
    setAttenuation=13,13.0
    enable=1;1;1;1;1;1;1;1;1;1;1;1;1;1;1
}
```

The first field is the procedure name and must be unique in the file, then follows the name of the backend instance the procedure refers to. The open bracket must lie on the same line. Between brackets the configuration commands for the backend are given. They must comply

with what the backend expects, so they may vary from case to case. The above example refer to the new continuum backend, which has 14 output channels which can be separately configured.

Meaning of the keywords:

integration = integration time for each sample which gets recorded in the output file
(ms, ≥ 1 , it can be greater than the sampling interval specified below)
calSwitch = activation of the switching calibration mark (0 = off, 1 = on), it is available only for the K-band receiver
setSection = list of setup parameters for each channel, which in order are:
section number (i.e. channel number, unique, no sorting necessary)
frequency (MHz) (*not applicable, i.e. '-1', for the total power backend*)
bandwidth filter (MHz)
feed number (*not applicable, i.e. '-1', for the total power backend*)
polarisation mode (L-R or Full Stokes, *not applicable, i.e. '-1', for the total power backend*)
sampling rate (MHz)
frequency bins number (*not applicable for the total power backend*)
setAttenuation = channel number and relative dB value
enable = flag for enabled channels (1= enabled, 0 = disabled); the number of flags must be equal to the number of channels (i.e. 2 for single feed receivers)

"-1" means that the specific value is not set here: it is either set to default, not applicable or read from other configuration files. In particular, notice how **the observed frequency must be set to "-1"** as it is specified in the .cfg file (see next section), and how the attenuation value is specified by a dedicated command line. In this framework, a typical **.bck file for single feed** receivers is:

```
STD:BACKENDS/TotalPower {
    integration=40
    setSection=0,-1,300.0,-1,-1,0.000025,-1
    setSection=1,-1,300.0,-1,-1,0.000025,-1
    setAttenuation=0,1.0
    setAttenuation=1,1.0
    enable=1,1
}
```

While a .bck file for the use of the **central feed only** of the new **22 GHz** multi-feed is:

```
STD:BACKENDS/TotalPower {
    integration=40
    calSwitch=0
    setSection=10,-1,2350.0,-1,-1,0.000025,-1
    setSection=11,-1,2350.0,-1,-1,0.000025,-1
    setAttenuation=10,9.0
    setAttenuation=11,9.0
    enable=0;0;0;0;0;0;0;0;0;0;0;0;1;1;0;0
}
```

NOTICE: the **bandwidth filter labels** have been corrected to actually match the nominal bandwidths (in MHz). The available values are: 300.0, 730.0, 1250.0, 2350.0 (frontend max. bandwidth, thus real observed bandwidth, is lower than these values!)

FileName.cfg

This file lists the configuration parameters for the frontend frequency and for the execution of some procedures that the users might want to run before or after a scan.

The first field is the name of the procedure which must be unique in the file. The open bracket must lie on the same line of the procedure name.

Between brackets the configuration commands must be provided one for each line.

The .cfg file can contain as many procedures as needed. Names are case sensitive.

An example of a possible content of the .cfg file:

```
INITALL {
    setLO=4900.0;4900.0
}

PRESCAN {
    tsys
    wait=2.000
}

POSTSCAN {
    wait=4.000
}

GOSTOW {
    antennaPark
}
```

There should be a procedure like **INITALL** (the name is user-defined), conceived to be called only once, when the schedule is loaded. One useful command to be inserted inside this call is the "**setLO**" one, to specify the **local oscillator** frequency value (MHz): this frequency setup can be manually performed during the overall system setup phase, but it also can be changed for each schedule inserting this command in the initialization procedure.

To know which LO frequency must be used, take the sky frequency you want to tune at the beginning of the observed band and subtract 100 MHz, for each IF (notice the semicolon between the two values).

For XXP receiver (8 GHz) only the default value is possible.

For example:

5 GHz → setLO=4900.0;4900.0 if observing band starts from 5.0 GHz
8 GHz → setLO=8080.0;8080.0
21 GHz → setLO=19900.0;19900.0 if observing in the 20-22GHz band (for the MF receiver the available bandwidth increases to 2 GHz)

The following two procedures are meant to be called before and after the execution of the actual scan, and can be named as the user prefers: remember that their name must be correspondingly called inside the **.scd** file (see next section).

In the above example, if the procedure **PRESCAN** is called before a certain scan, a Tsys will be measured before the scan begins. Please notice that these commands are NOT time-based, so a proper timing must be allowed if the schedule is LST-based, otherwise the Tsys measurements (implying that a calibration mark is switched on) could take place when the scan has already started. A schedule example given at the end of this document will make things clearer.

The above example of the **POSTSCAN** procedure, if called for a given scan, means that - after the completion of that scan - the system will wait for 4 seconds before commanding the

next scan to the antenna. This is usually done to let the antenna complete the deceleration ramp before slewing to another position. The "wait" command replaces the "gap" column which was present in the ESCS 0.1 **.scd** file (see later on). A lists of the main commands to be used inside the .cfg procedures is given at the end of this Appendix.

FileName.lis

This is the list of the subscan spatial and operational configurations. It specifies the setup parameters for a series of subscans labelled with a compulsory and unique ID number. Various fields are needed, depending on the specific type of scan required.

All values are TAB-separated.

ID = scan unique ID

type = at the moment this can be OTF, SIDEREAL, MOON. SiMPIE uses "OTF".

Then the scan **spatial configuration** are specified.

For **OTF scans**

lon1 = for descr=SS (later keyword), longitude (*) of the scan starting point.
for descr=CEN, longitude (*) of the subscan central point.

lat1 = for descr=SS (later keyword), latitude (!) of the scan starting point.
for descr=CEN, latitude (!) of the subscan central point.

lon2 = for descr=SS (later keyword), longitude (*) of the subscan ending point.
for descr=CEN, whole longitude span (*) of the subscan.

lat2 = for descr=SS (later keyword), latitude (!) of the subscan ending point.
for descr=CEN, whole latitude span (!) of the subscan.

frame = coordinate frame relative to the previously specified lon-lat coordinates:

EQ = Equatorial J2000.0 (longitude = RA, latitude = Dec)

HOR = Horizontal (longitude = azimuth, latitude = elevation)

GAL = Galactic (longitude = l, latitude = b)

sFrame = coordinate frame along which the scan is performed. It must be equal to "frame", apart from a single case: if "frame" is "EQ" and the scan description is "CEN" (see next keywords), then "sFrame" can also be "HOR", which means that Az-El scans will be performed across a sidereal position. This is usually exploited for pointing calibration campaigns.

geom = scan geometry:

LON = constant longitude

LAT = constant latitude

GC = great circle arc (only for start-stop description)

descr = scan description:

SS = start and stop positions

CEN = center position + scan span

dir = scan direction (ignored if geom=CG):

INC = the varying coordinate increases

DEC = the varying coordinate decreases

duration = scan actual duration in seconds (ramps excluded)

offFrame = frame for the user-defined offsets to be added to the lon-lat coordinates specified (the leading dash '-' is compulsory):

- EQOFFS = Equatorial
- HOROFFS = Horizontal
- GALOFFS = Galactic

At present **offFrame must be equal to sFrame**, which means that it is not possible to perform scans in an exotic way like scanning along the galactic longitude while applying equatorial offsets, ecc... but it is possible to apply Az,El offsets to Az,El scans across a sidereal (equatorial) position.

lonOff = longitude offset, in degrees, which can be specified as dd.dd'd' (decimal format, can be positive or negative, notice the 'd' suffix) or dd:mm:ss.s. It is meant to be "**on sky**", i.e. the actual span, in practice $\Delta\text{lon} \times \cos(\text{lat})$.

latOff = longitude offset, in decimal degrees.

For **SIDEREAL** (fixed position) scans – NOTICE: they have a **different column order!**

frame = coordinate frame relative to the following lon-lat coordinates:

- EQ = Equatorial (longitude = RA, latitude = Dec)
- HOR = Horizontal (longitude = azimuth, latitude = elevation)
- GAL = Galactic (longitude = l, latitude = b)

lon1 = longitude of the commanded position (*)

lat1 = latitude of the commanded position (!)

epoch = to be specified for Equatorial coordinates only (1950.0, 2000.0 or -1 i.e. "now")

offFrame = frame for the user-defined offsets to be added to the lon-lat coordinates specified (the leading '-' is compulsory):

- EQOFFS = Equatorial
- HOROFFS = Horizontal
- GALOFFS = Galactic

In the sidereal case **offFrame can be different from sFrame**.

lonOff = longitude offset, in degrees, which can be specified as dd.dd'd' (decimal format, can be positive or negative, notice the 'd' suffix) or dd:mm:ss.s. It is meant to be "**on sky**", i.e. the actual span, in practice $\Delta\text{lon} \times \cos(\text{lat})$.

latOff = longitude offset, in decimal degrees.

(*) longitudes are given in degrees and can be expressed according to two formats: ddd.ddd'd' (suffix 'd' char) or dd:mm:ss.sss. For Right Ascension only, also the format HH:MM:SS.SS'h' (suffix 'h' char) is allowed.

(!) latitudes are given in degrees and can be expressed according to two formats: +/-ddd.ddd'd' (suffix 'd' char) or dd:mm:ss.sss

Coordinates are range-checked, offsets are not checked since also negative values are allowed. For example a right ascension -12:16:24.45h is not legal but a right ascension offset -00:00:25h is ok.

Here follow some examples of legal scans inside the .lis file:

#ID	type	lon1	lat1	lon2	lat2	frame	sFrame	geom	descr	dir	duration	offFrame	lonOff	latOff
1	OTF	10.00d	15.00d	0.000d	3.000d	EQ	EQ	LON	CEN	INC	60.000			
2	OTF	10.00d	15.00d	3.000d	0.000d	EQ	EQ	LAT	CEN	INC	60.000			
3	OTF	15.50d	23.00d	18.00d	27.00d	EQ	EQ	GC	SS	INC	240.000	-EQOFFS	0.0d	-0.5d
4	OTF	0.00d	0.00d	30.00d	0.00d	GAL	GAL	LAT	SS	INC	120.000	-EQOFFS	0.0d	0.0d
5	OTF	45.00d	45.00d	20.00d	0.00d	HOR	HOR	LAT	CEN	DEC	90.000	-HOROFS	1.0d	0.0d
6	OTF	45.00d	45.00d	20.00d	0.00d	HOR	HOR	LAT	CEN	INC	90.000	-HOROFS	0.0d	0.0d

Subscans 1 and 2 make a cross scan performed in equatorial J2000.0 coordinates, where the source position (RA,Dec)=(10.0°,15.0°) is observed by means of a scan in Dec (first scan), and then a scan in RA (second scan), each of them being 3° wide and lasting 60 seconds – which means that the scanning speed is 3°/min. No offsets are given: the system will apply the last specified ones.

NOTICE: offsets, once set, are valid until a new value is specified.

They must be explicitly zeroed if they need to be unapplied!

If they are unwanted, it is wise to zero them in the INITALL procedure, see available commands.

ScheduleName.scd

This is the true schedule, i.e. the operating sequence of subscans. This file must begin with a header written as follows (keywords and values are **TAB-separated**):

```
PROJECT:  ProjectName
OBSERVER: ObserverName
SCANLIST: FileName.lis
PROCEDURELIST: FileName.cfg
BACKENDLIST: FileName.bck
MODE:      LST    1
SCANID:    1
INITPROC:  INITALL
```

The use of spaces or special symbols is not allowed in these user-defined strings.

“SCANLIST”, “PROCEDURELIST” and “BACKENDLIST” must report the correct names of the .lis, .cfg and .bck files to be employed. Names are case sensitive.

The “MODE” keyword refers to schedule timing. Its value can be **LST** (local sidereal time) or **SEQ** (sequential). If the LST timing is chosen, the user **must** specify after the label, with a tab-separated integer, the number of times the 24-h schedule is to be cyclically run (-1 means “ad libitum”, 1 means that the whole schedule will be run only for one 24-h slot).

For sequential schedules, an **LST** time can be optionally specified as follows, in order to start the schedule at that time and then proceed with the sequential observations:

MODE: SEQ HH:MM:SS.SS

The "SCANID" keyword is optional ; if specified, it sets the initial number for the enumeration of the output files. For example, setting this keyword to 100 means that a "100" value will be written in the corresponding keyword inside the FITS header for the first file, "101" for the second, etc...

"INITPROC" specifies which procedure listed in the .cfg file must be run at the schedule startup; if not present, it is considered NULL – which leads to no actions.

Then the actual schedule follows, again all the fields are **TAB-separated**.

An **LST-based schedule** is structured like this:

#ID	label	startLST	durat	scan	preScan	postScan	backend
1	dr21	10:12:04.000	60.0	1	PRESCAN@	POSTSCAN	STD:MANAGEMENT/FitsZilla
2	3c123	10:16:10.500	60.0	2	PRESCAN@	POSTSCAN	STD:MANAGEMENT/FitsZilla
3	n449	10:21:14.000	300.0	3	PRESCAN@	POSTSCAN	STD:MANAGEMENT/FitsZilla
4	nope	10:30:14.000	300.0	15	PRESCAN	NULL	STD:MANAGEMENT/FitsZilla
5	dummy	11:00:10.300	500.0	16	PRESCAN	NULL	STD:MANAGEMENT/FitsZilla
6	foofoo	11:42:00.000	300.0	22	PRESCAN	NULL	STD:MANAGEMENT/FitsZilla
7	w3oh	11:59:11.400	300.0	25	PRESCAN	NULL	NULL

Fields:

ID: progressive number that enumerates the scans, each of which will produce an output file. IDs must start from 1 and be incremental **and** continuous (no gaps allowed);

label: label to be included in the output file name. Could be the source name, or the ID label of the single scan within a map or a sequence, etc...;

startLST: exact Local Sidereal Time in the format HH:MM:SS.SSS when the data acquisition will begin. It must coincide with the beginning of the constant speed scan, since the acquisition will actually start only if the commanded start position has been reached. No time repetitions are allowed: LST instants must be incrementally ordered. 00:00:00.000 can be given after 23:59:59.999, but **the overall schedule cannot last more than 24 hours**;

duration: scan duration, in seconds. It must coincide with the one given in the .lis file for that specific scan. Data acquisition will stop at (startLST+duration), with an uncertainty of about 1 second due to backend characteristics – take this into account when performing short scans: it's better to set them 1 second longer than necessary.

scan: ID of the scan configuration to be retrieved from the .lis file;

preScan: name, case sensitive, of the procedure that is intended to be executed before data acquisition. It can be NULL if no operations are necessary, in which case nothing is done. It must be defined in the .cfg file. The default behaviour is that the scheduler will wait for the procedure to finish before doing anything else. If this is not the desired approach, a "@" must be appended, then the system will not wait for the procedure completion and will in any case start the acquisition at the startLST time;

postScan: name, case sensitive, of the procedure that is to be executed after the acquisition has stopped. It can be NULL if no operations are necessary, in which case nothing is done. It must be defined in the .cfg file. Synch/asynch behaviour is analogous to preScan;

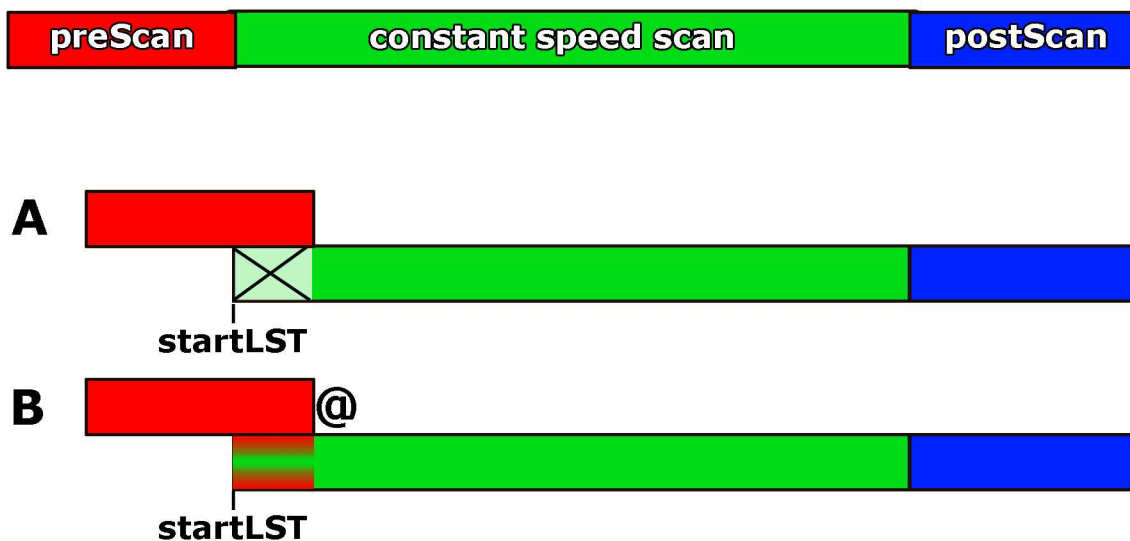
backend: it selects the definition (in the .bck file) to be used for the given scan. After the colon the scheduler expects to find the instance of the data collector/formatter to be used. At present, the only one available is STD:MANAGEMENT/FitsZilla, which writes the FITS files. NULL means that no data acquisition is performed.

As concerns **sequential schedules**, they are similar to the LST-based ones: they just don't have the startLST column, so the scans are performed sequentially in a "as soon as possible" mode. Of course, the visibility of the commanded source is checked: in case the source is not observable, the relative scan is skipped.

Caveat: preScan and postScan in OTF LST-based schedules

The use of the preScan and postScan procedures is still experimental, and will undergo major updates in a near future. For the time being, users are warned that their employment is quite unfriendly and treacherous.

preScan and postScan procedures must be employed in LST-based schedules *cum grano salis*, after the timings have been properly set and tested, as they could coincide with the acceleration ramps. It is also important to understand how exactly the Tsys measurements are performed and applied to the data samples. Otherwise, some unwanted effects might hold.



Ideally, the execution of the various parts of the scan should always be as shown by the top case in the above figure: the preScan procedure is executed first, then the scan is performed and - once the scan is over - the postScan operations takes place, no matter what the timings turn out to be.

In practice, this never happens with the present system.

In case of LST-based schedules, the key point is that the antenna motions will always be strictly consistent with the commanded scan positions, which are **fixed in LST** as specified in the SCD file.

So, if the timings in terms of startLST - which corresponds to the instant when the data

acquisition will nominally start (now happening **only if** the antenna has reached the commanded position) – have not been determined carefully taking into account the preScan and postScan needs, case A or case B shown above can take place.

A – preScan is used synchronously, so the system waits for the preScan procedure to be completed and only then starts the data acquisition. If startLST happens while the preScan is still operating, the antenna motion will take place as predicted (the scan starts) but the data acquisition will be delayed until preScan is done. In practice, one part of the scan is lost;

B – preScan is used asynchronously, so the system begins the data acquisition at startLST regardless of the preScan completion. This means that data could be acquired while other procedures are still active, e.g. with a calibration mark switched on to measure a Tsys.

As for OTF sequential scans: contrarily to what one might think, the logic behind sequential schedules is still time-based, even if in a hidden way: when a new scan is commanded, OTF estimates how much time is needed to get to the commanded position – knowing the current time and antenna position – and uses that information to compute a feasible startLST, on the basis of which the new scan path is computed by the usual “OTF machine”. This must be taken into account when considering if and how to use the pre/post-scan procedures.

Changes in this features might be coming, but are not imminent.

Very important note on Tsys

How and when the Tsys is measured is a key point. In ESCS 0.2, **the Tsys value is stored in the system every time it gets measured, and the most recent measurement is used for the conversion of the acquired samples (from counts to Kelvins).**

Inside the FITS file, the Tsys is no more a dummy value: a new table is added to the FITS file, which contains all the data samples converted into antenna temperature by means of a conversion factor, which is achieved from the last stored Tsys value (**raw data in counts are still stored as before**, though).

Thus, if the last Tsys was achieved in a different position, or even at a different frequency, it will be applied to the following scan if a new measurements is not performed. It is trivial to notice how this could easily become nonsense.

The best solution would be to **get a Tsys value right before the beginning** of each scan, or at least at the beginning of the first scan of a group taken on the same source.

To accomplish to this without cutting or polluting the scan (as instead it happens in the A-B cases above) the solution is to structure the schedule so that a fixed-position scan is performed right before the OTF scan, measuring the Tsys in the nearbies of the source, yet off-source, during its prescan procedure (see below).

Users must determine how often to update the Tsys values considering their own peculiar needs, taking into account how relevantly the elevation, weather conditions, etc... vary from scan to scan or from source to source. Ideally, it is possible to take a Tsys before every scan, but it is a fairly redundant choice.

In practice, let us suppose we want to update the **Tsys value every time we change our target**, so one time for every cross-scan group on a given source.

Here follows how our **mixed-type schedule** should be structured.

The **SCD** file (example for a sequential schedule):

```
PROJECT:      ProjectName
OBSERVER:    ObserverName
SCANLIST:     FileName.lis
PROCEDURELIST: FileName.cfg
BACKENDLIST:  FileName.bck
MODE: SEQ
SCANID:       1
INITPROC:     INITALL
```

#ID	label	durat	scan	preScan	postScan	backend
1	Tsys	1.0	1	PRESCAN	NULL	STD:MANAGEMENT/FitsZilla
2	s1	12.0	2	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
3	s1	12.0	3	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
4	s1	12.0	2	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
5	s1	12.0	3	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
6	Tsys	1.0	4	PRESCAN	NULL	STD:MANAGEMENT/FitsZilla
7	s2	12.0	5	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
8	s2	12.0	6	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
9	s2	12.0	5	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla
10	s2	12.0	6	NULL	POSTSCAN	STD:MANAGEMENT/FitsZilla

The relative **LIS** file (notice that SIDEREAL and OTF scans have different columns!):

OTF columns labels

#ID	type	lon1	lat1	lon2	lat2	frame	sFrame	geom	descr	dir	duration	offFrame	lonOff	latOff
-----	------	------	------	------	------	-------	--------	------	-------	-----	----------	----------	--------	--------

SIDEREAL columns labels

#ID	type	label	frame	lon1	lat1	epoch	offFrame	lonOff	latOff					
1	SIDEREAL	tsys1	EQ	10.00d	15.00d	2000.0	-EQOFFS	-1.6d	0.0d					
2	OTF	10.00d	15.00d	3.000d	0.000d	EQ	EQ	LAT	CEN	INC	12.000	-EQOFFS	0.0d	0.0d
3	OTF	10.00d	15.00d	0.000d	3.000d	EQ	EQ	LON	CEN	INC	12.000			
4	SIDEREAL	tsys2	EQ	50.00d	70.00d	2000.0	-EQOFFS	-1.6d	0.0d					
5	OTF	50.00d	70.00d	3.00d	0.00d	EQ	EQ	LAT	CEN	INC	12.000	-EQOFFS	0.0d	0.0d
6	OTF	50.00d	70.00d	0.00d	3.00d	EQ	EQ	LON	CEN	INC	12.000			

In practice, for each source we perform an initial SIDEREAL scan (= on a fixed sidereal position) in an offset position wrt the source, so that we are next to the ramp-start position of the following OTF scan.

The **CFG** file will contain the procedures (freq. is 5 GHz in this example):

```
INITALL {
    setLO=4900.0;4900.0
}

PRESCAN {
    waitOnsource
    tsys
    wait=2.000
}

POSTSCAN {
    wait=2.000
}
```

The “waitOnsource” command before the “tsys” one ensures that the measurement is performed when the dish is actually pointing to the commanded position.

List of useful commands for schedule procedures

- **tsys**
Performs the Tsys measurements. It can be used in any of the three procedures.
- **setLO**
To be used in the INITALL procedure. It sets the Local Oscillator frequency in order to observe, at the beginning of the band, the specified **sky frequency** (MHz).
Syntax: setLO=freq1; freq2; ...; freqN (N = number of IFs)
- **wait**
Sets a delay before the next operation is commanded.
Syntax: wait=s.s (seconds).
It can be used in any of the three procedures. Use it after a tsys measurements to allow the complete switch-off of the calibration mark (a 2 sec delay is enough).
- **waitOnsource**
It waits that the "Onsource" flag is raised before performing the following operations. "Onsource" means that the antenna has reached the commanded position.
Use this command only in the preScan of SIDEREAL scans.
- **antennaSetup**
It performs the antenna unstow and setup. Useful in the INITALL procedure, at the beginning of the schedules.
Syntax: antennaSetup=setupCode (i.e. SETUPCCC, SETUPXXP, etc... In **caps** chars.)
- **setupCCC (or setupXXP, setupKKC)**
It performs the antenna unstow and full receiver/backend default setup. Useful in the INITALL procedure, at the beginning of the schedules.
- **azelOffsets, radecOffsets, lonlatOffsets**
To set offsets. Since the offsets are considered valid until a new setting is commanded, even in the following schedule, to avoid using old or unwanted offsets it is advisable to zero them at the beginning of the schedule, in the INITALL procedure.
Syntax (for all of them): azelOffsets = 0.0d, 0.0d
- **antennaPark**
It sends the antenna to stow position. Useful in a proper procedure to be called after the last scan of a session.

New schedules can be launched in the post-scan procedure of a schedule, exploiting the same syntax used as a command line during the observations. Pay attention to this recursive usage, though.

Suggestion for the optimisation of the LIS target list

Generally speaking, the LIS file could be unique for a certain project, since it contains only the spatial configuration of the scans.

The schedule generator could create a LIS file out of the whole source list, each of which would thus be associated to a limited number of spatial configuration (say: one SIDEREAL for Tsys, two OTFs for the cross-scans).

This way, manually adding/editing scans to the SCD schedules – both sequential and LST-based – would be very easy as the scan numbers associated to a certain source would be always the same in the LIS file.

APPENDIX D – FITS files

The present version of the output file is an almost-standard FITS made out of the following 6 elements:

Index	Extension	Type	Dimension	View				
<input checked="" type="checkbox"/> 0	Primary	Image	0	Header	Image	Table		
<input checked="" type="checkbox"/> 1	SECTION TABLE	Binary	4 cols X 2 rows	Header	Hist	Plot	All	Select
<input checked="" type="checkbox"/> 2	RF INPUTS	Binary	9 cols X 2 rows	Header	Hist	Plot	All	Select
<input checked="" type="checkbox"/> 3	FEED TABLE	Binary	4 cols X 1 rows	Header	Hist	Plot	All	Select
<input checked="" type="checkbox"/> 4	DATA TABLE	Binary	12 cols X 431 rows	Header	Hist	Plot	All	Select
<input checked="" type="checkbox"/> 5	ANTENNA TEMP TABLE	Binary	2 cols X 431 rows	Header	Hist	Plot	All	Select

It opens and plots with any software reading regular FITS (FitsViewer, IDL routines, FITS I/O libraries, etc...).

0 – PRIMARY HEADER

Here the compulsory FITS header keywords are stored.

A list of observing site info and telescope setup details then follows.

1 – SECTION TABLE

It shows basic info about the sections (i.e. channels).

Column meaning and units are also described in its header, as for all the following tables.

Each row is dedicated to one channel:

	<input checked="" type="checkbox"/> id	<input checked="" type="checkbox"/> type	<input checked="" type="checkbox"/> sampleRate	<input checked="" type="checkbox"/> bins
Select	J	6A	D	J
<input checked="" type="checkbox"/> All			MHz	
<input type="button" value="Invert"/>	<input type="button" value="Modify"/>	<input type="button" value="Modify"/>	<input type="button" value="Modify"/>	<input type="button" value="Modify"/>
1	0	simple	2.5000000000000E-005	1
2	1	simple	2.5000000000000E-005	1

id = channel number

type = simple (total power) or full (Full Stokes, i.e. including polarimetry)

sampleRate = data sampling rate (MHz)

bins = number of frequency bins (1 for total power)

2 – RF INPUTS

Receiver general setup (ignore numbers written in table below).

Select	feed	ifChain	polarization	frequency	bandWidth	localOscillator	attenuation	calibrationMark	section
J	J	8A	D	D	D	D	D	D	J
All				MHz	MHz	MHz	db	K	
Invert	Modify	Modify	Modify	Modify	Modify	Modify	Modify	Modify	Modify
1	2	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	0
2	2	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	1
3	3	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	2
4	3	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	3
5	4	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	4
6	4	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	5
7	5	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	6
8	5	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	7
9	6	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	8
10	6	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	9
11	0	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	10
12	0	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	11
13	1	0		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	12
14	1	1		-9.119129139149E-03	-9.119129139149E-03	-9.119129139149E-03	9.000000000000E+00	-9.119129139149E-03	13

feed = feed number

ifChain = IF number

polarization = left or right

frequency = observed frequency at the beginning of the band (MHz)

bandWidth = actual observed bandwidth (MHz)

localOscillator = LO frequency (MHz)

attenuation = attenuation (dB) applied to the channel

calibrationMark = temperature of the calibration mark

section = number of section associated to this RF input

3 – FEED TABLE

Information on the feeds position (meaningful for Multi Feed receivers)

Select	id	xOffset	yOffset	relativePower
J	D	D	D	
All				
Invert	Modify	Modify	Modify	Modify
1	0	0.000000000000E+00	0.000000000000E+00	1.000000000000E+00

id = feed number

xOffset = x offset position (arcsec) w.r.t. the central feed, computed along azimuth axis: $x > 0$ for increasing azimuth, when the receiver is in its reference position (no rotation is applied to dewar)

yOffset = y offset position (arcsec) w.r.t. the central feed, computed along elevation axis: $y > 0$ for increasing elevation, when the receiver is in its reference position (no rotation is applied to dewar)

relativePower = nominal ratio between this feed gain and the central feed gain

4 – DATA TABLE

Large table containing all the raw data, one row for each sample.

Columns:

time = MJD (Modified Julian Day)

raJ2000 = J2000.0 Right Ascension (radians)

decJ2000 = J2000.0 Declination (radians)

az = azimuth (radians)

el = elevation (radians)

par_angle = parallactic angle (radians)

derot_angle = rotation angle of the dewar (radians), at present it still is a dummy value

flag_cal = calibration mark flag, 0=off, 1=on

flag_track = tracking flag: 1 = pointing error is $< 0.1 \times \text{HPBW}$

0 = pointing error is $> 0.1 \cdot \text{HPBW}$

weather = array of three values: temperature (°C), relative humidity (%) and atmospheric pressure (hPa), measured at ground level

Ch0,...,ChN = N columns, one for each channel, containing the signal intensity in arbitrary counts

5 – ANTENNA TEMP TABLE

It contains N columns (*Ch0*, ..., *ChN*) with the signal converted in antenna temperature (K). Conversion is performed using a *counts-to-K* factor retrieved from the last available Tsys measurement. This means that the conversion factor, if the Tsys value had been achieved in a distant time or position w.r.t. the data stream, could be obsolete and/or not applicable to the data! Pay much attention to the usage of this table.