SOFIA Command Language (SCL) User's Manual

Document Number: SOF-DA-MAN-OP02-2181

Date: October 8, 2014
Revision: -

AFRC
Armstrong Flight Research Center
Edwards, CA 93523

ARC
Ames Research Center
Moffett Field, CA 94035

German Space Agency, DLR
Deutsches Zentrum für Luft und
Raumfahrt

SOFIA Command Language (SCL) User Manual

AUTHOR:

Jeff Homan — 12/9/14

ARC / Jeff Homan, Systems Engineer — Date

CONCURRENCE:

Steve D Culp — 12/18/14

SETI Institute / Steve Culp, Software Systems Engineer — Date

B Wohler — 2014-12-19

SETI Institute / Bill Wohler, Software Engineer — Date

for Nancy McKown — 1/16/15

USRA / Nancy McKown, Mission Operations Manager — Date

May Windrem — 1/6/2015

NASA / May Windrem, Observing/Ground System Technical Manager — Date

Daniel — 12/19/14

NASA / Dan Kozarsky, SE&I Lead — Date

APPROVAL:

Lisa Bjarke — 1/6/2015

NASA / Lisa Bjarke, SOFIA Observatory Systems Director — Date

SOFIA Command Language (SCL) User's Manual

1. Introduction

The SOFIA Command Language (SCL) is the user interface for conducting science observations. With a human readable syntax, SCL was created to facilitate the monitoring and control of all observing on the Stratospheric Observatory for Infrared Astronomy (SOFIA). This document provides a high-level overview description, geared primarily for Science Instrument (SI) developers, of the functional aspects of SCL, but should be useful to all users.

The remainder of this section provides the overall purpose and scope for this document, information on related documents, and the relevant change history. The following sections then provide an overview of the Observatory system, a description of SCL, and then discussions of using it.

1.1. Purpose/Scope

SCL provides the mechanisms for communication and data exchange between the MCCS and its users in order to control observing on SOFIA and to access other MCCS software functions, controls, and services from on-board computers and observatory workstations. SCL commands are entered directly over a network connection, or indirectly via the workstation graphical user interface (GUI) or by activating previously composed scripts (XML files). These SCL commands are sent to the MCCS, which validates and translates them into the appropriate direct commands, and routes them to the telescope or other subsystem.

The purpose of this document is to familiarize users with the functional and higher level aspects of SCL, and how to use it to observe on SOFIA. Details on SCL protocols and XML file format can be found in MCCS_SI_04 as noted below in Section 1.2.

1.2. Applicable Documents

The following documents, including Interface Control Documents (ICD), are referenced herein.

Document Number	Title
SOF-DA-ICD-SE03-052	*MCCS_SI_04, SOFIA MCCS to SI Functional ICD*
SCI-US-ICD-SE03-2023	*DCS_SI_01, Data Cycle System (DCS) ICD*
SOF-DF-ICD-SE03-047	*TA_MCCS_F, TA to MCCS Functional ICD*
APP-AR-SPE-SW02-2012	*MCCS Coordinate Transform Software Architectural Design: coord.convert*

1.3. Change History

Revision	Date	Description	Approval
Rev. -	Oct 2014	Original Release	**TBD**

2. Observatory System Overview

This section provides an overview to the Observatory systems, starting with an overview of a science flight and a basic introductory SCL session. Additional subsections discuss the key concepts of AOIs and tracking, summarize the Observatory components, and briefly discuss the in-flight data products.

2.1. Anatomy of a Science Flight

The objective of a typical science mission is the successful execution of the science flight plan and the associated observing activities. This execution is accomplished by the on-board mission team -- Mission Director (MD), Telescope Operator(s) (TO), Science Flight Planner (SFP), SI Team, and flight deck personnel (e.g., pilots) -- who work together to execute the science flight plan using pre-planned products and the on-board observatory systems to capture sufficient data of all planned targets following the science flight plan.

While flight deck personnel fly the airplane, the Mission Director enforces flight leg start/stop times and controls in-flight access to the instrument itself when hardware adjustments are needed. The Science Flight Planner monitors the flight plan and works with the MD and others to coordinate in-flight replanning as needed. The Telescope Operator sets up the Telescope Assembly (TA) and acquires targets before verbally handing over TA control to the SI Team, who then operate the instrument to carry out the observations.

The typical flight profile begins with a departure after sunset and concludes with a landing before sunrise the following morning. The science flight consists of a sequence of planned observing legs, each focused on one or more astronomical targets. As the aircraft burns fuel and concludes an observing leg, it typically climbs in altitude to reduce the amount of water vapor in the atmosphere above the aircraft and in the line-of-sight of the telescope. Observing legs can vary in extent, from about ten minutes to four hours in duration. The number of observing legs will vary in accordance with the observing plans to be executed during the flight. The number of astronomical objects that can be observed in one flight typically ranges between two and sixteen, with an average of about seven. Generally, at least one of these targeted objects during the flight is designated as a calibrator, which is a well-characterized object whose measurement is used to calibrate/correct the data from the other targets; without such calibration the observational data would yield only relative results, not absolute ones, which significantly reduce their overall scientific value. The aircraft heading (i.e., flight direction) is changed as necessary to allow target visibility to the TA, which views out the port/left side of the aircraft; the TA has a limited range of motion with approximately +/-3 deg in azimuth and 20-60 deg in elevation. These constraints, when coupled with the observing targets, generally yields a science flight plan with a zig-zagging pattern.

A typical flight lasts ten hours, with about 8-9 hours available for observing time, assuming the required systems are all working satisfactorily. The full mission timeline is for a 24-hour period and includes pre-, in-, and post-flight activities. The pre-flight activities include clearing of any required observatory system discrepancies, or “squawks”, performance of routine maintenance items, and execution of the day-of-flight setup for the on-board mission systems. The in-flight activities begin for the flight/mission crew with the with a Crew Briefing conducted immediately prior to boarding the aircraft. The flight sequence then proceeds as the Observatory first takes off and ascends, arrives at the appropriate altitude to open the cavity door, acquires, tracks, and observes the first object, and then similarly executes the subsequent flight legs and the transitions between them. At the conclusion of all observing legs, the crew implement instrument and telescope shutdown procedures, close the cavity door, descend, land, and complete the remaining flight procedure steps. Once on the ground the post-flight activities are conducted, which includes performing aircraft and mission system follow-up checks, and off-loading and processing the collected science and engineering data.

2.2. Basic SCL Session

Interacting with the MCCS is conducted through an SCL session, which begins with a `login`, ends with a `logout`, and includes lots of commands in between to carry out observations and monitor HK data. To quickly introduce the user to SCL, a very basic example session is shown below in which a user named *wmiller* logs in to SCL with the role of a TO, checks the installed software version, looks at a bit of HK, and then logs out; throughout the system provides various messages about what it is doing in response to the user's commands. See Section 3 for full details on using SCL commands and HK.

```
1 login user=wmiller role=to
1 A
1 :
2 revision
2 A
2 : message="Compiled  2013-05-20 12:36:53 -0500 (Mon, 20 May 2013)  by
HOUGHA "
3 set verbose=normal
3 A
3 : verbose="normal"
4 revision
4 A
4 I message="MCS 38, tagged mccs_20130520_rev38, current release level
CAE_IOC"
4 : message="Compiled  2013-05-20 12:36:53 -0500 (Mon, 20 May 2013)  by
HOUGHA "
5 set verbose=none
5 A
5 : verbose="none"
6 get list=[das.ic1080_10hz.baro_alt ta_tsc.tsc_mcs_hk.tsc_status
ta_state.tsc_status] showlabels=yes
6 A
6 : das.ic1080_10hz.baro_alt=2587.000000
ta_tsc.tsc_mcs_hk.tsc_status=67108864
ta_state.tsc_status="STAB_INERTIAL_ONGOING"
7 logout
7 A
7 :
```

2.3. AOIs and Tracking

At the beginning of a flight the MCCS uses aircraft and telescope attitude and position data inputs to estimate the relationship (transformation) between the gyro-based Inertial Reference Frame (IRF) and sky coordinates (e.g., Equatorial Reference Frame (ERF)). This coordinate transformation is refined after the first pointing of the telescope in flight using known sources, and is updated and maintained throughout the flight.

For target tracking the TO defines one or more Areas of Interest (AOI) around track-stars that are then used to improve TA pointing stabilization at low frequencies (~1 Hz). High frequency pointing stability is handled through the gyro servo control loop (in the TASCU), but slow gyro drift can cause the gradual increase of pointing error if not corrected with external optical reference points on the sky – stars in either the FPI, FFI or the WFI – by the Tracker system.

For position stability the Tracker system currently supports two types of tracking, both based on AOI track-star centroids:

- 2-axis – Keeps a known gyro-offset held constant between a track-star and the tracking position, which is usually tied to the SI boresight.
- 3-axis – This includes the 2-axis above plus a Rotation of Field (ROF) stability enhancement that senses the motion of two rotation-stars simultaneously in one imager (usually in the WFI or FFI) to insure that the orientation of the line connecting them remains constant.

AOIs may be defined in any of the imagers (FPI, FFI or WFI). The Tracker uses AOIs to identify and designate image frame objects as tracking sources, and performs centroid calculations on each. The system can currently support only eight (8) concurrent AOI definitions, all of which can be used simultaneously in the various imagers, although only six (6) in any one imager at a given time; the Tracker stores the active AOIs with numerical names trc_aoi1 to trc_aoi8. AOIs are typically defined by the TO, associated with a sky position, and are displayed on the associated workstation GUI imager window for reference purposes. Once an AOI (or any position) is defined, the MCCS establishes an associated housekeeping (HK) data table that contains the following information that is accessible to the SI and users: AOI number, imager camera, calculated centroid (RA/Dec and pixels), row/column size and location (pixels), peak counts, and centroid quality indicator. When an AOI is activated it initiates centroid calculations, which the TA then uses to apply relative pointing corrections to maintain the calculated centroid at the center of the AOI.

Tracking parameters can be specified using either the ta_pos.track command or the coord.position command (see Section 3). In this way a defined position can be pre-configured for tracking, and after issuing the move commands (e.g., `ta_pos.goto pos=…`) the Tracker will be configured accordingly with tracking initiated. Positions that include tracking keywords can be defined before the AOIs have been defined or activated. An error will only be returned if a user asks the TA to track using a position's tracking information, and one of the required AOIs has yet to be defined or activated.

Figure 2-1 shows a screen capture of a section of the FPI with overlays indicating positions, nod and chop throws, and AOIs (the labels explain the color-coding). The Tracker performs the calculations to try to find the centroid of the target within the AOI. A centroid quality indication is provided by the blue bar along the left edge of the AOI box and is also recorded within an HK table. If a centroid, with quality indication greater than a quality threshold defined internal to the Tracker, is present within an AOI, the Tracker will adjust the center of the AOI to be coincident with the calculated centroid. If no centroid (of sufficient quality) is measured, the Tracker maintains the position of the AOI fixed with respect to IRF. For valid centroids, the Tracker provides the centroid location in an HK table and is reported with respect to a number of reference frames, such as imager pixels, IRF, or the Telescope Assembly Reference Frame (TARF). The MCCS will keep track of the AOI parameters and dynamically translate the relevant information into the proper format for use as AOIs within the Tracker.

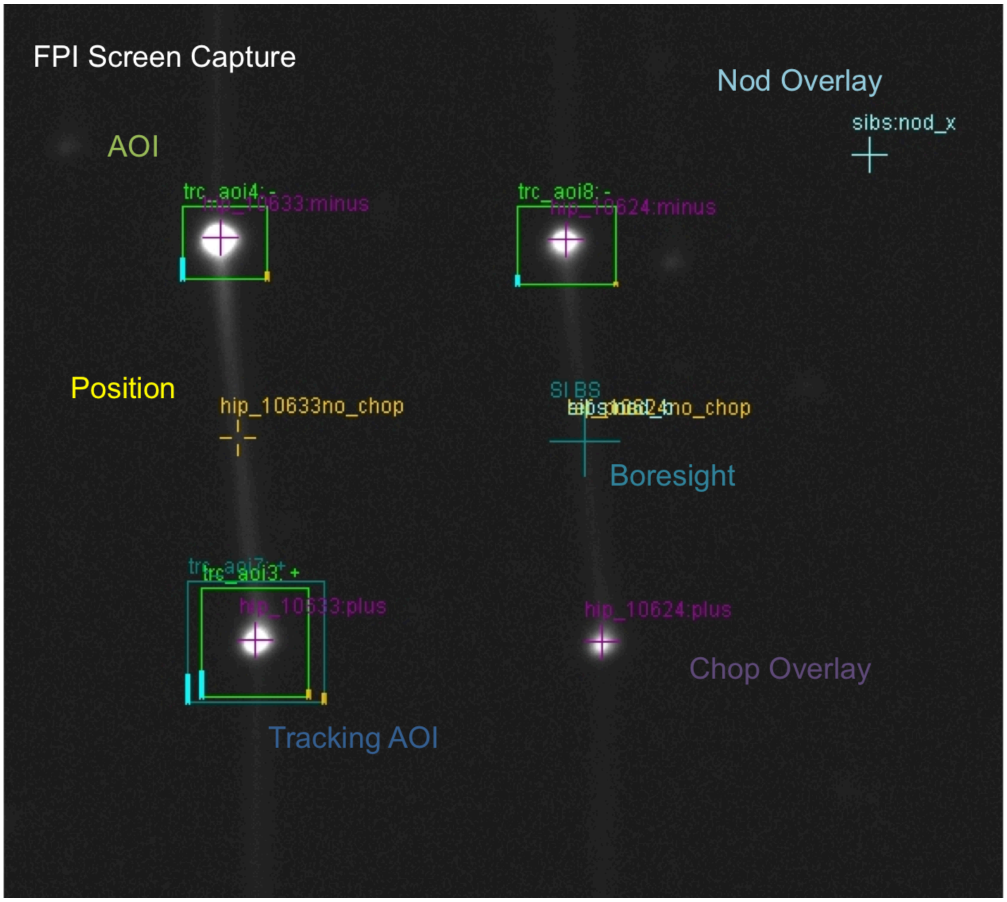

Figure 2-1: FPI screen image showing Areas of Interest (AOI).

2.3.1. Tracking Keywords and Attributes

Tracking keywords are associated with the tracking command `ta_pos.track` and are as follows: *centroid*, *rof1*, *rof2*, *limb1*, *limb2*, *inertial*, *boresight*, and *track_mode*. These same quantities (except for track_mode) appear as attributes of the position command as well. Thus a position can be pre-configured for tracking, and after issuing the commands `ta_pos.goto pos="position_name"` or `nod.goto pos=(a,b,or x)` the `ta_pos.track` keywords listed below (excluding track_mode) will be assigned the values given by the position attributes in the `ta_pos.goto` or `nod.goto` commands.

These quantities have the same meaning in both the keyword and attribute form, that is:

- centroid – Identity of the AOI around the track-star for positional corrections (when centroid tracking).
- rof1 – Identity of the AOI around the first rotation-star for rotation corrections (when centroid tracking with inertial=yes).
- rof2 – Identity of the AOI around the second rotation-star for rotation corrections (when centroid tracking with inertial=yes).
- inertial – Indicate (yes/no). When centroid tracking, indicates whether or not to correct for gyro drift. When rof tracking, require this flag to be set to “yes.”
- track_mode - Indicates which tracking mode to use (none/centroid/rof). Its default value is calculated to be the value consistent with activated AOIs that have been assigned to centroid or rof keywords.

The tracking keyword, track_mode, can be used to turn off tracking if set to none, or to select a subset of the tracking initially specified. If track_mode is set to centroid, only the centroid AOI will be used for centroid tracking. If track_mode is set to rof, then the centroid, rof1, and rof2 AOIs will be used for centroid tracking with rotation stabilization.

Centroid positional tracking can only be initiated if there is a valid AOI for the centroid keyword/attribute. While centroid tracking, the flags for boresight and inertial will be set to those values given for the keywords.

ROF tracking can only be initiated if there are valid AOIs for the centroid, rof1, and rof2 keywords/attributes AND the `inertial=yes` flag is set. If the keyword/attribute inertial was given a no value (either due to a default or a newly assigned value), then an error will be returned after requesting rof tracking, and the command will be aborted. This is an artifact of the Tracker software design. However, it is expected that the effects of drift in LOS will be very small compared with the other two axes, so rof tracking is not required for pointing stability.

NOTE: The above tracking attributes for position can be defined for a position even before the AOIs have been fully defined or activated. An error will only be returned if a user asks the TA to track using a position’s tracking information, and one of the required AOIs has yet to be defined or activated.

2.3.2. Tracking Position Conventions

In order to communicate the AOIs used for tracking to the SI, the TO may define one to three positions with the fixed names of *nodtracka*, *nodtrackb*, and *nodtrackx*. These contain the centroid and ROF AOIs to be used in beams a, b, and x respectively during nod activities (see Section 5.3). If only nodtracka is defined, that information is used for all three nod positions. When moving to a reference location, a fourth position *reftrack* may contain the centroid AOI to be used there. The same ROF stars used for on-source tracking are always used at the reference position. The record of which AOI is in use for what position can be obtained by subscribing to the associated HK, as shown in the following example; subscribing to HK is covered in Section 3.2.

```
subscribe list=[nodtracka.centroid nodtracka.rof1 nodtracka.rof2
nodtrackb.centroid nodtrackb.rof1 nodtrackb.rof2
nodtrackx.centroid nodtrackx.rof1 nodtrackx.rof2
reftrack.centroid reftrack.rof1 reftrack.rof2]
```

2.4. Observatory Components

The SOFIA observatory is a Boeing 747SP aircraft containing a 2.5m TA with a mounted SI, all controlled via the MCCS. The SI is comprised of the actual detector instrumentation along with one or more computer control systems and associated equipment and mounting racks, and is maintained by either the Principal Investigator (PI) as a PI Science Instrument (PSI), or by the Observatory team as a Facility Science Instrument (FSI). The SI communicates with the MCCS via the PI patch panel (see *MCCS_SI_05*), along with an optional direct link to the TA's secondary mirror as noted below. There is a phased development for SIs, and the current list of available 1st- and 2nd-generation SIs are shown in Table 2-1.

Table 2-1: SOFIA Science Instruments, including classification, type, and origin.

Instrument Name	Acronym	Class	Generation	Home Institution	Country	Type of Instrument
High Speed Imaging Photometer for Occultations	HIPO	PSI	1st	Lowell Observatory	U.S.	Optical High-speed Imager
Faint Object InfraRed CAmera for the SOFIA Telescope	FORCAST	FSI	1st	Cornell	U.S.	Mid-IR Camera and Grism Spectrometer
German Receiver for Astronomy at Terahertz Frequencies	GREAT	PSI	1st	MPIfR, KOSMA, DLR-WS	Germany	Heterodyne Spectrometer
High-resolution Airborne Wideband Camera	HAWC	FSI	1st	University of Chicago	U.S.	Far-IR Bolometer Camera
Field-Imaging Far-Infrared Line Spectrometer	FIFI-LS	FSI	1st	MPE, Garching	Germany	Imaging Grating Spectrometer
First-Light Infrared Test Experiment Camera	FLITECAM	FSI	1st	UCLA	U.S.	Near-IR Camera and Grism Spectrometer
Echelon-cross-Echelle Spectrograph	EXES	PSI	1st	UC Davis	U.S.	Echelon Spectrometer
High-resolution Airborne Wideband Camera Polarization	HAWC-Pol	FSI	2nd	Jet Propulsion Lab (JPL)	U.S.	Far-IR Polarimeter
High-resolution Airborne Wideband Camera, Plus, Plus	HAWC++	FSI	2nd	Johns Hopkins University	U.S.	Adds a large-format detector array

The TA is comprised of four major subsystems:

- TA Master Computer Processor (TAMCP): This computer is a communications hub that connects to the MCCS and to the internal TA subsystems. It passes commands from the MCCS to the other parts of the TA, and it collects housekeeping data from the TA subsystems and forwards that data to the MCCS.
- TA Servo Control Unit (TASCU): This computer is the heart of the telescope's pointing and control capability for the two drive motors: it supervises the Coarse Drive and uses fiber optic gyroscopes to control the inertial attitude of the Fine Drive via electromagnetic torque motors.
- Tracker: The Tracker computer makes use of the three visible light charge-coupled device (CCD) imagers to monitor TA pointing on the sky, and to correct the residual gyroscope drift via handshaking with the TASCU. The three CCD imagers are as follows:
  - Focal Plane Imager (FPI): The FPI is a 1024x1024 pixel CCD camera with a 9' x 9' field-of-view (FOV) that shares the telescope's focal plane with the SI via the telescope's second tertiary mirror. Since it views nearly the same field of view as the SI, it is generally used for set-up, pointing, and tracking. The FPI mechanism includes a back-focus adjustment to make this

imager parfocal with the SI.

- Fine Field Imager (FFI): The FFI consists of a headring-mounted separate telescope and camera with a 1024x1024 pixel CCD and 67' x 67' FOV. The FFI can be used for pointing setup and tracking when using the fully reflective telescope tertiary mirror (note that this mirror is not yet available).
- Wide Field Imager (WFI): The WFI consists of a headring-mounted separate telescope and camera with a 1024x1024 pixel CCD and 6^{o} x 6^{o} FOV. The WFI is primarily used for sky-field recognition and monitoring sky rotation stability.

- Secondary Mirror Control Unit (SMCU): This computer controls the Focus Centering Mechanism (FCM) for TA collimation and focus, and also the Tilt/Chopping Mechanism (TCM) of the Secondary Mirror Assembly (SMA). The observer can control the SMCU through the MCCS via SCL. The SI can also control the secondary mirror tilt/tip position via analog voltages presented to the PI Patch Panel (see *TA _SI_04*). The chopper frequency is generally controlled via a synchronizing transistor-transistor-logic (TTL) signal from the SI. The secondary mirror itself is typically a dichroic, which splits the incoming light beam into two paths, one to the FPI and the other to the SI.

The MCCS is comprised of a suite of hardware and associated software that provides a wide variety of functions and services to support observing including the following: power distribution, data acquisition, data archiving, network services and timing, audio communication, video, and the Observatory crew workstations with associated graphical user interface. It also provides operator supervisory control and monitoring of the Observatory, especially the TA, via SCL to allow operators/users to send commands and monitor HK.

2.5. Flight Data Products

The flight data products include a select set of pre-flight planning files along with the raw science and engineering data collected during the flight. The in-flight data are recorded on the MCCS archiver for post-flight transfer to the permanent archive for follow-on processing and analysis. Table 2-2 provides a summary of the major flight data products.

Table 2-2: Major flight data products, including data source, format/file extension, and relevant descriptive notes.

Data Product	Source	Format/Extension	Descriptive Notes
science data	SI	fits (standard binary)	SI-produced science data in standard FITS format; the file format and associated FITS keywords are defined in ICD DCS-SI-01.
HK data	MCCS	ark (SOFIA-specific binary)	HK engineering data recorded for all MCCS subsystems and external systems (e.g., aircraft avionics).
non-HK data	MCCS	ark (SOFIA-specific binary)	High-rate data MCCS data, typically from the DAS, that is most useful for post-flight analysis.
audio	MADS	mp3 (standard binary)	Recording of up to 10 audio communication channels used by the mission crew.
video	DVDS/VPARS	mp4 (standard binary)	Streaming video data from multiple sources, including cabin and cavity cameras as well as mission crew workstations.
raw TAIPS images	DVDS/TAIPS	ark (SOFIA-specific binary)	Raw TAIPS imager (WFI/FFI/FPI) images that are stored with the associated HK engineering data.
logged TAIPS images	DVDS/TAIPS	fits (standard binary)	Raw TAIPS images that are saved/logged on command (via ta_{wfi,ffi,fpi}.log) and written in fits format.
command-and-final-response log	MCCS	log (ASCII text)	File containing all MCCS commands and their final responses, which is extremely useful for following in-flight progress and performing post-flight data analysis.
GUI log	MCCS	log (ASCII text)	Log file produced for a workstation running the MCCS GUI; each flight typically produces such log files for the MD, TO, SFP, and EPO workstations.
comment log	MCCS	txt (ASCII text)	Log file produced for each MCCS user who use the "comment" command.
ephemeris	SFP	eph (ASCII text)	Ephemeris planning file to be used for tracking a non-inertial moving target (e.g., comet).
science flight plan	FMI	fp (SOFIA-specific binary)	The science flight plan defines the targets to be observed on each leg during the flight, and is used to track the actual vs. planned aircraft path. A standard set of associated human-readable files are also produced for reference purposes, including a visual map (png) and various text reports (e.g., mis).

3. SCL Description

The SOFIA Command Language allows scientists and other operations personnel to monitor and control certain parts of the Observatory subsystems (e.g., telescope) and to access other MCCS software functions, controls, and services from on-board computers and workstations. The following subsections provide an overview of various aspects of SCL. Section 3.1 introduces SCL commands, their associated structure, and the highlights those most useful to users. HK types and their usage in monitoring are discussed in Section 3.2. A brief overview of the supported reference frames (RF) is provided in Section 3.3, while some additional key SCL concepts are summarized in Section 3.4.

3.1. Commands

SCL commands provide users with the means of control in performing all observing activities. Commands are organized into so-called *commandable* blocks based on related functional activities that cover the expected user needs for conducting science observations, and with command privileges associated with various user roles (see Section 3.4.4). Users support and perform SOFIA science activities by choosing, building, and executing the relevant SCL command sequences to conduct the basic SOFIA observing-related activities -- connect and login, setup and initialization, control and monitoring, and logout and disconnect -- as described in detail in Section 4. The commands listed here are referred to as non-pass-through commands, which are high-level, user-friendly commands. A large set of lower-level pass-through commands are also available (see TA_MCCS_F) but are generally not recommended for any but the most advanced users.

3.1.1. Choosing a Command

A user chooses a command from one of the 18 commandables, which are functional groupings of related commands. Table 3-1 shows each commandable, along with its functional description and list of most popular/useful SCL commands. Note that the general nomenclature for an SCL command is `commandable.command` (e.g., `nod.define`).

Table 3-1: Available commandables, functional descriptions, and most useful commands.

Commandable	Functional Description	Popular Commands
<commandable>	This pseudo-command group provides global functions that can be applied to any of the other command groups below by substituting its name before the command (e.g., `session.cancel` or `coord.alive`).	• alive • cancel, • help
[session]	General infrastructure commands required for a user to log on/off, initiate an MCCS session, set up for observing, subscribe to HK, and log data and comments. Note: Because session commands are all uniquely named, they can be invoked *without* the commandable prefix (e.g., `session.get` and `get` are equivalent).	• alias • get • login • logout • subscribe • unalias
ars	Commands to control the Archiver, and not for general use.	N/A
cdds	Commands to control the CDDS, and not for general use.	• set_aperture • track_aperture
coord	Coordinate-related commands to initiate an SI, define and list positions for observing and tracking, conduct coordinate transforms, initiate and maintain the inertial telescope-to-sky pointing calibration, and define the active boresight.	• aoi_* • correct • delete • init • instrument • list • position
das	Commands to control the DAS, and not for general use.	N/A
dither	Commands that define multi-position dither sets, and execute moves between those dither positions including their related tracking information. The associated HK keeps track of dither position use and sequence, as well as which dither position contains the active boresight. See Section 5.4 for information on dither mode.	• define • goto • next
nod	Commands that define two- or three-beam nod sets, and execute moves between those beam positions including their related tracking information. The associated HK keeps track of nod beam position use and sequence, as well as which beam position contains the active boresight. See Section 5.3 for information on nod mode.	• define • goto • next
ntp	Commands to control the NTP server, and not for general use.	N/A

rien (i.e., fictitious SI)	Instrument-related commands that inform the MCCS of the SI configuration. Use of these commands is predicated on first initiating an SI via the coord.instrument command, which directs the MCCS to read-in the named SI's pre-defined configuration information from its specific XML file (e.g., `great_data.xml` -- see Section 6 below). Once initiated, the commands in this group can be used to set the SI's operating mode, as well as adjust its alignment and boresight. Note that once an instrument is initiated, the fictitious rien command group takes on the name of that actual instrument (e.g., `great.mode` sets the operating mode for the SI called GREAT).	• mode • si_align • si_boresight • sibs_change
sma	Commands to initialize the SMA and to control all FCM and TCM operations (e.g., focus, centering, offset, and chop). There are commands to support calibrating the chop parameters on the sky at the beginning of each flight. And the sma.chop command defines two- and three-point chop positions on the sky, and assigns boresights to each for use in telescope pointing. The associated HK keeps track of the chop positions and sequence in use, as well as which position contains the active boresight. See Section 5.2 for information on chop mode.	• align_* • chop • focus • init
ta	Commands for general TA control, and not for general use.	N/A
ta_ffi	Commands for obtaining images from the FFI.	• log
ta_fpi	Commands for obtaining images from the FPI.	• log
ta_pos	Commands for moving the telescope in an absolute or relative sense within a number of different reference frames, performing tracking, and managing LOS rewinds. The associated HK contains information about where the active boresight is pointed on the sky.	• fpi_focus • goto • rewind • track • tweak
ta_state	Commands to control the operational states of the TA (e.g., in "servo" mode for inertial pointing). Access to the majority of these commands is restricted to the TO role.	• camera
ta_wfi	Commands for obtaining images from the WFI.	• log
wvm_if	Commands for control of the WVM, and not for general use.	N/A

A complete list of all available high-level commands, and their associated required and/or optional keywords, can be found in MCCS_SI_04, Appendix A.

3.1.2. Building a Command Sequence

Building a command is performed either directly, via network connection (terminal window), or indirectly within the MCCS GUI (command tool or hard-coded command-to-button assignment) or from a previously defined XML-based script file; this document focuses mainly on the direct build method.

In building a command sequence the user chooses the following:

- command identifier (cid): A user-defined arbitrary integer to identify the command with its associated command response(s).
- command: The functional command to perform, from one of the commandable groups in Table 3-1.
- keyword(s): Zero or more arguments to the specified/selected command. Although not all commands require keywords, for those that do some are required while others are optional. In addition, the order of keywords in the command sequence is not important, and some keywords are mutually exclusive (e.g., no mixed coordinates).
- value(s): Zero or more settings for a keyword, depending on its function, where values can be of various types:
    - number: an integer, real, or sexigesimal number (e.g., 12, 3.15, 12h13m14.5s)
    - word: a position name or enumerated list option selection (e.g., TRUE)
    - string: any sequence of printable ASCII characters separated by double quotes (e.g., "this is a string"); other characters can be used via escape sequences (e.g., "this is a string with an escaped double-quote (\")")
    - list: one or more values in brackets separated by whitespace, which can be either a simple list (e.g., [23.5 16.5]) or a compound list [[23.5 16.5] [15.3 19.6]]

- attribute: A characteristic modifier, identified in parentheses, for a value. A value can have zero or more attributes, and an attribute can include one or more nested keyword=value pairs (see Section 3.4.2.4 on inline position definitions). There are three main types of attributes that are useful to users:
    - time – define which timestamp is of interest; additional details are provided in Section 3.4.1
    - units – define specific units to use, instead of defaults; additional details are provided in Section 3.4.5
    - position-related – define various properties associated with a position; additional details are provided in Section 3.4.2

In addition to command-specific keywords the MCCS supports various global keywords that apply to any individual command; these can also be set individually to apply to all commands (e.g., via `set attr=all`). The major global keywords are the following; the initial default value is in *italics*:

- attr: Global time attribute: all, mcstime, *none*, pkt_timestamp, sampletime
- showlabels: Whether or not item labels are displayed in responses: false, *true*
- showunits: Whether or not item units are displayed in responses: *false*, true

- verbose: Verbosity level for responses: all, *none*, normal

For more details on these and the other global keywords see Section 4.1.1.1 of MCCS_SI_04.

The MCCS supports a Global Default structure that provides values for keywords, attributes, and units that are not explicitly listed in a command. Not all keywords/attributes have defaults, some being optional and others requiring specification each time they are used. Those that do have defaults are of three types:

- fixed: A value that is preset in MCCS and never changes.
- calculated: A value that is automatically calculated by the MCCS.
- previous (same-as-last-time): A value that stays the same until changed by a subsequent command, and therefore need not be specified until a change is warranted. A value may or may not be assigned at system boot; in the latter case a default takes affect the first time it is set/used in a command. ***NOTE**: A "previous" default is the same across all MCCS sessions so care must be taken to make sure only one session is controlling the observing, or that each operator must view the displays to see the current default values before issuing a command or modifying an entity.*

Default types and values are listed with the associated keywords in Appendix A of MCCS_SI_04.

The following are some examples of valid command sequences, with color-coding to show the cid (grey), command (red), keyword(s) (blue), value(s) (green), and attribute(s) (purple).

EXAMPLE #1: A simple command, with no keywords, values, or attributes.
25 logout

EXAMPLE #2: A typical command with multiple keywords and values.

26 coord.set comment="test comment" file="mytestfile.txt"

EXAMPLE #3: A complicated command with multiple keywords, values, and attributes; in this example the attribute includes an inline position definition (see Section 3.4.2.4).

27 ta_pos.goto pos=great_target(ra=193.87125(units=degrees) dec=-2.970417(units=degrees) centroid=trc_aoi3 inertial=yes chop=no_chop) track_mode=centroid

3.1.3. Executing a Command Sequence

A command sequence built directly in a terminal window can be executed by ending it with a line-termination character (<CR>, <LF>, or <RETURN>), which causes the sequence to be sent to the Platform Interface Subsystem (PIS) for processing. The PIS translates the sequence into the appropriate distribution protocol and format, and then forwards it to the appropriate subsystem (or the PIS itself), where final execution occurs. Depending on the command function, its execution may be virtually instantaneous or take some noticable time to complete; in the case of certain open-ended commands (e.g., `subscribe`), they run continuously until specifically cancelled by the issuing user. Command execution can be monitored via command response HK (see Section 3.2), and note that user commands can also spawn internal commands, which are assigned negative cid numbers by the PIS. See *SOFIA Command Language Protocol* in MCCS_SI_04 Section 3.3.1 for more details on command sequence formatting.

3.2. Housekeeping

Housekeeping provides the means for users to monitor observing activities and system performance. The HK data items are named and identified using a prefix and a value name as shown in of MCCS_SI_04, Appendix B. The prefix is associated with a major MCCS subsystem (e.g., ars, das, or wvm_if) or a major command group (e.g., dither, nod, or sma), while the value name identifies the actual item identifier; the full path name of the item is a concatenation of the prefix with the value name, separated by a period ("."). For example, HK items related to a position can be found with the prefix `coord.pos.ANY` where ANY is a placeholder for the name of a defined position. The row and column pixel location for that position in each TA imager is recorded in addition to the position in each of the various reference frames. For example, given a defined position named m82 (M82 galaxy), the IRF elevation (u), cross-elevation (v), and line-of-sight (w) axis values for this position are stored in the following HK parameters:

- coord.pos.m82.irf_u
- coord.pos.m82.irf_v
- coord.pos.m82.irf_w

The following subsections describe the types and organization of HK items, as well as how to access and monitor them via SCL during a flight. Additional HK details are available in MCCS_SI_04. For HK specific to pass-through commands see TA_MCCS_F.

3.2.1. HK Types and Organization

The MCCS HK stream is comprised of system and user-defined parameters that fall into five categories:

1. Engineering – Parameters that provide operational and status engineering information (e.g., mirror temperatures, power voltage/current levels, target position parameters).

2. Commands and their responses – Commands sent by users (as well as internally-generated commands), along with their response messages.
3. Comments – User comments that facilitate in-flight communications.
4. Alerts and alarms – Messages that reflect out-of-limit conditions for engineering parameters, as well as other out-of-character subsystem conditions (e.g., faults).
5. System-generated – Special items related to user sessions and system connections.

The HK items are organized into so-called *dataGroups*, which are packets of one or more items associated with a common timestamp (there are many single-item dataGroups). For example:

- sma.oper_state -- This is a single HK data item published by the sma commandable, and belongs to its own dataGroup called *sma.oper_state*.
- coord.pos.sibs.ra, coord.pos.sibs.dec -- These two HK data items are published by the coord commandable, and belong to a common dataGroup called *coord.pos.sibs* (note: "sibs" position name is used in lieu of "ANY" placeholder noted above).
- das.cabin_env.md_console_temp_1, das.cabin_env.to_console_temp_1, das.cabin_env.mission_area_humidity -- These three HK data items are published by the *das* commandable, and belong to the common dataGroup called *das.cabin_env*.

Each dataGroup has its own associated timestamp from the PIS, known as *mcstime* (see Section 3.4.1).

3.2.2. Monitoring HK Items

During a flight any HK is accessed for monitoring via one of the following SCL command requests:

- subscribe: An open-ended request for one or more items that continues until cancelled by the requestor (via `cancel` command). The command can request updates based on various sampling, time interval, and/or trigger criteria.
- get: A one-time request to see the latest value(s) of the requested item(s). Although a `get` is a single-readout `subscribe`, it also provides access to a small set of data (e.g., the current time) that are not available via subscription.

In the above command requests one or more HK items are specified by their full path name or by the shortest unique path which includes the item name. In the latter case the MCCS searches the available parameters and returns an error if there is more than one item that ends with the provided name. After commanding such a request the user will receive one (get) or more (subscribe) response messages, each including the associated cid, a response status code, and possibly an associated message string. The typical response status codes are shown in the Table 3-2 below.

Table 3-2: SCL command response status codes.

Code	Meaning	Example
A	Acknowledgement response	100 A
I	Intermediate response	101 I 26.312500 28.687500 28.750000
:	Final successful response	102 : message="DONE"
E or S or F	Various types of error responses	103 F message="Timed out trying to deliver command from source ...
#	Abort response	104 # message="Command Canceled"

As noted above, any negative cid numbers (e.g., "-15") in the response indicate internal commands, which allows the user to trace command movement through the relevant system(s).

3.2.2.1. An Example of Monitoring Engineering HK

For example, the following are valid unique housekeeping data item identifiers:

- coord.sibs.ra OR sibs.ra
- coord.sibs.dec OR sibs.dec
- ta_tsc.responses.tsc_fd_pos_abs OR responses.tsc_fd_pos_abs OR tsc_fd_pos_abs

The current values for these parameters may be obtained using the following SCL commands, which will generate the appended responses:

```
320 get list=[sibs.ra sibs.dec]
320 A
320 : sibs.ra=5h46m53.931s sibs.dec=-9d23m43.063s

321 get list=[ta_tsc.responses.tsc_fd_pos_abs responses.tsc_fd_pos_abs
tsc_fd_pos_abs]
321 A
321 : ta_tsc.responses.tsc_fd_pos_abs="" responses.tsc_fd_pos_abs=""
tsc_fd_pos_abs=""
```

Note that the `sibs.ra` and `sibs.dec` items may *not* be shortened to `ra` and `dec` since there is more than one data item each ending with the names *ra* and *dec*. However, `tsc_fd_pos_abs` may omit the `ta_tsc.responses` prefix since there is only one item with this name in MCCS. In general, we currently suggest that users fully qualify the HK items in order to prevent errors.

As another example, a subscription to the aircraft latitude, longitude, and heading information, along with its initial intermediate (I) response, would look like the following; the final cancel command would cancel the original subscription (cid=322):

```
322 subscribe list=[das.ins_1_12hz.hybrid_lat das.ins_1_12hz.hybrid_lon]
showLabels=no
322 A
322 I 34.6135523790000 -118.0754739010000
398 cancel cmdid=322
```

3.2.2.2. An Example of Monitoring Command and Response HK

Upon sending a command sequence the user will generally receive one or more response messages from the PIS, and in some cases additional lower-level response messages from additional subsystems. In some cases specific HK items are also updated that can be monitored. In general, two or more responses will be provided indicating the receipt, progress, and success or failure of command execution, as shown in the following examples.

```
26 coord.set comment="test comment" file="mytestfile.txt"
26 A
26 : message="DONE"

27 ta_pos.goto pos=great_target(ra=193.87125(units=degrees)
dec=-2.970417(units=degrees) centroid=trc_aoi3 inertial=yes chop=no_chop)
track_mode=centroid
27 A
27 : message="All sub-commands completed successfully -14
ta_tsc.tsc_fd_pos_abs fd_des_pos_abs_quat= [-0.07002198581814356
-0.8973148948292972 0.3932217938083664 -0.1878816699247206] -14 W
message=\"SEQ_COUNT_CHG\" -15 ta_trc.trc_mode_tracking_centroid_offset
trc_des_inertial_enable= 1 trc_des_boresight_enable= 0 trc_des_aoi_trc= 3
trc_des_rel_irf_quat=[-8.21436734235714e-05 -4.099191233130213e-05
1.101513763834654e-05 0.9999999957253733] -15 W message=\"SEQ_COUNT_CHG\"
```

3.2.2.3. An Example of Monitoring Alert and Alarm HK

On-going monitoring of alert and alarm HK requires slightly different subscription command options and format. For example, a subscription to FMS alarm warnings, along with its initial intermediate response, would look something like the following; the cancel command would cancel the original subscription (cid=24).

```
24 session.subscribe list=ta_tsc_alarm_error
24 I ta_tsc_alarm_error="The value
mcs.ta_proxy.ta_tsc.tsc_mcs_hk.atc_rate_el (-0.0233967) is below the error
limit at -0.0174"

399 cancel cmdid=24
```

3.3. Reference Frames and Coordinate Systems

This section introduces reference frames and the coordinate systems used within each reference frame. Complete descriptions of all of the SOFIA reference frames, including their relationship to each other is found in *MCCS Coordinate Transform Software Architectural Design: coord.convert*.

3.3.1. Supported Reference Frames

The MCCS supports the reference frames shown in Table 3-1; for each the table shows the following:

- Acronym – reference frame keyword for use in commands (case insensitive)
- Name – name of reference frame
- Type – type of reference frame: sky-,observatory-, or SI-based in spherical, cartesian, or special coordinates
- Origin – origin for reference frame
- Center – center for reference frame
- Coordinate Labels – labels for associated coordinates used in the coord.pos.ANY housekeeping

The Equatorial (ERF) and Ecliptic (EcRF) reference frames require an equinox, and J2000 is the default and only one currently supported. Some of these reference frames are only used internally by the MCCS, and are thereby unavailable to users.

Table 3-1: Reference Frames supported by the MCCS.

Acronym	Name	Type	Origin	Center	Coordinate Labels
ERF	Equatorial (precessed to the equinox)	Sky Spherical[1]	Vernal Equinox	Geocentric	ra, dec, rof
EcRF	Ecliptic	Sky Spherical[1]	Vernal Equinox	Geocentric	lambda, beta, psi
GalRF	Galactic	Sky Spherical[1]	Galactic Center	Geocentric	lii, bii, rof
TARF	Telescope Assembly	Observatory Special[2]	TA boresight	TA bearing center	u(tarf), v(tarf), w(tarf)
SMARF	Secondary Mirror Assembly	Observatory Cartesian	SM Vertex	SMA center	u(smarf), v(smarf), w(smarf)
IRF	Inertial (Gyros)	Observatory Special[2]	TA boresight	TA bearing center	u(irf), v(irf), w(irf)
WFIRF	Wide Field Imager	Observatory Special[2]	Upper left of WFI	TA boresight	x(wfi), y(wfi), -
FFIRF	Fine Field Imager	Observatory Special[2]	Upper left of FFI	TA boresight	x(ffi), y(ffi), -
FPIRF	Focal Plane Imager	Observatory Special[2]	Upper left of FPI	TA boresight	x(fpi), y(fpi), -
SIRF	Science Instrument	SI Special[2]	Upper left of SI	TA boresight	x(si), y(si), -

Notes to table:

[1] All spherical coordinate systems also include a position angle/orientation. They are defined by subsequent rotations about three axes. While subsequent rotations must be orthogonal, the first and third may be along the same axis.

[2] These coordinate systems are defined by three simultaneous rotations about orthogonal axes. For the case of imagers and the SIs, units are pixels, others are in degrees. Note that since the imagers do not rotate with respect to the telescope, their rotation angle is always zero.

3.3.2. EL, XEL, and LOS in an Airborne Observatory

Limits on the telescope range of motion in TARF affect how the TA and aircraft must be controlled during science observations. Therefore it is important for users to understand this reference frame and the implications of its specific motion limits.

As shown in Figure 3-1, the axes fixed to TARF provide a relative coordinate system (u, v, w) that defines the following relative rotations: Elevation (EL, rotation about u, centerline of the Nasmyth tube), Line-of-Sight (LOS, rotation about w), and Cross-Elevation (XEL, rotation about v = w X u).

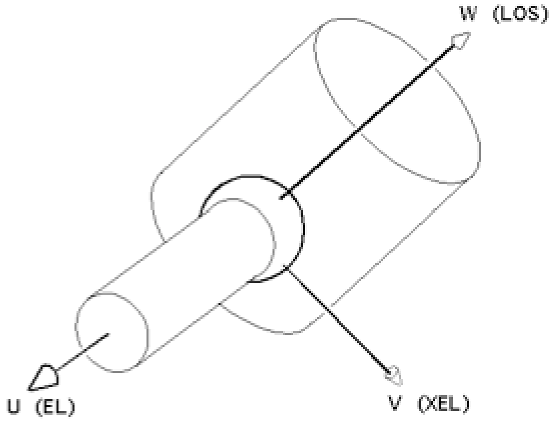

Figure 3-1: TARF axes and associated rotations.

In TARF, EL is set mainly by the telescope drive (adjustable to 23^{o}-58^{o}, unvignetted), XEL by aircraft heading, and LOS by the sky rotation. The motions of the latter two are coupled and limited according to the following:

$(XEL)^2 + (LOS)^2 \quad (2.5^{o})^2$

Note that all three rotations and their limits are also affected to some degree by aircraft attitude changes, so the actual operational ranges are somewhat smaller. When inertially pointed to a position on the sky, the values of the gyro coordinates are constant, but the translated values of EL, XEL, and LOS slowly change as the aircraft moves and the Earth rotates. XEL is kept in range via aircraft heading adjustments, but at some point LOS will reach a limit. When such a limit is reached the MCCS will have to reset/rewind the telescope LOS with respect to TARF.

3.3.3. Rotation of Field

The *z-coordinate* for sky coordinates is defined as the *Rotation of Field* (ROF), or rotation angle, at the position specified on the sky. The ROF with respect to each imager is defined as the angle between the *up* direction in the imager's focal plane and celestial north. This definition is shown in Figure 3-2, along with the corresponding sign convention for this rotation angle. ROF is the airborne observatory equivalent to the Vertical Position Angle (VPA) on ground-based observatories – where the up direction in the ROF definition is replaced by *local zenith*.

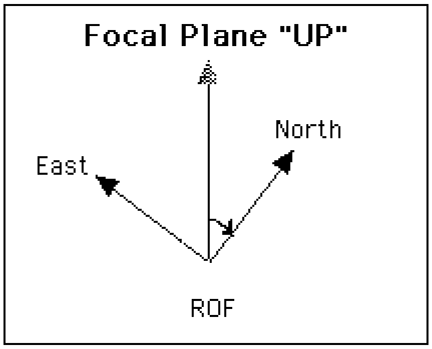

Figure 3-2: Depiction of ROF and associated sign convention.

Because of the limited range of the telescope's LOS, at any given time the allowable ROF values for a given position on the sky will be restricted to within ±2.5 degrees of the ROF value when LOS=0 (see Section 3.3.2). Once the telescope is inertially pointing, the ROF will stay constant while LOS slowly changes as the celestial object moves across the night sky, until LOS has reached a limit. This can take between a few minutes to hours, depending on where the telescope is pointed on the sky; note that for some fields near the celestial poles, ROF changes considerably over the FOV of the WFI. As LOS approaches a limit, the MCCS will issue a warning and then the observer or TO must issue a rewind command, which will move the LOS position back within its normal range (and thus change ROF by up to ~5^{o} minus any specified LOS margin, which must also include the effects of aircraft attitude motions that eat into the telescope motion limits). Upon completion of the LOS rewind the telescope is ready for the continuation of observations.

To translate between the SOFIA Observatory ROF and the SI reference frame, the MCCS publishes a VPA HK parameter. As shown in Figure 3-3 below, the VPA is expressed as degrees east of north in the TA imagers.

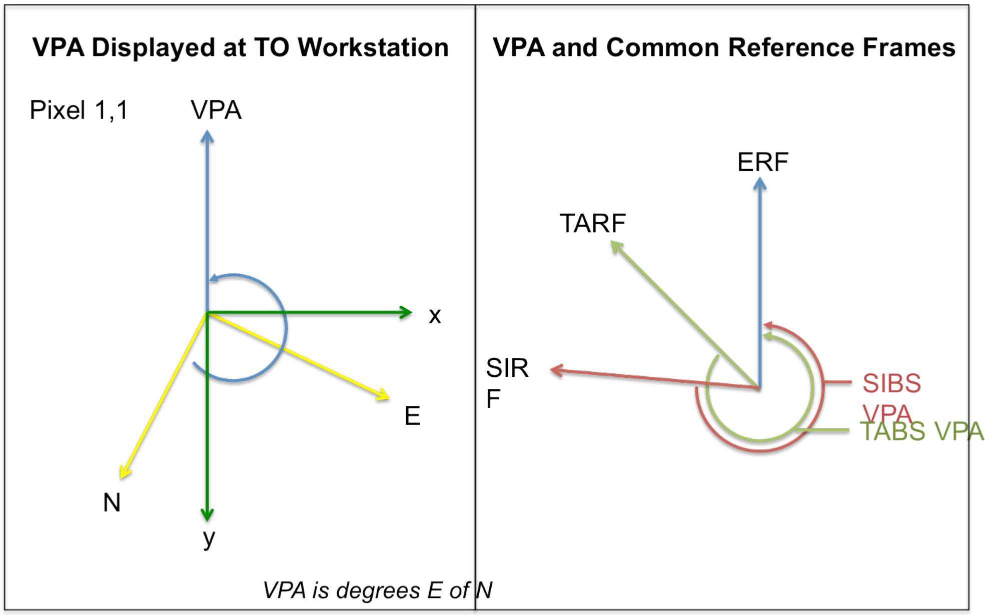

Figure 3-3: SOFIA Observatory Vertical Position Angle (VPA).

Using the published VPA value, the SI may convert between ERF, TARF, and the SI reference frame. For most SIs there will be a fixed VPA angle defined by the mechanical interface between the TA mounting flange and the SI optics. However, some SIs may incorporate a K-mirror into their optical path, requiring a more complex conversion between reference frames.

3.3.4. Conversion of Equinoxes and Reference Frames

The `coord.convert` command can be used to convert coordinates between reference frames, as shown in the following examples. The first shows an in-line-defined position being converted from ERF to GalRF, while the second converts a similar position from J2000, the default equinox, to B1950.

```
1 coord.convert position=great_target_temp(ra=10.369500(units=degrees)
dec=11.117832) to=GalRF
1 A
1 A message=""
1 : message="All sub-commands completed successfully  Changing the position
to
  RA = 0.6913, declination = 11.1178   1 : (lii=118.989923798520840
bii=-51.677666487358010
  rof=3.574985148684945)    "
```

Please note that the `coord.convert` command currently returns incorrect results when commanded to convert Ecliptic coordinates with equinoxes other than *J2000*. This is a known discrepancy and will be corrected in a future revision of the MCCS.

```
1 coord.position name=great_target_temp(ra=9.694100644132037(units=degrees)
  dec=10.84369336556711(units=degrees) equinox=B1950)
1 A
1 : message="DONE"
2  get list=[great_target_temp.ra(format="%lf")
great_target_temp.dec(format="%lf")]
2 A
2 : great_target_temp.ra=0.689601 great_target_temp.dec=11.117854
```

3.4. Key SCL Concepts

This section discusses some additional key SCL concepts including timestamps, positions, aliases, and users and their roles.

3.4.1. Timestamps

An important characteristic and attribute of observatory HK data is the time when it was produced, also known as its timestamp. All MCCS times are Universal Time (UT) as provided by the MCCS IRIG/NTP subsystem, which maintains time synchronization with GPS satellites and distributes that time to all onboard systems, via the NTP protocol over MCCS networks, and directly via IRIG-B protocol to selected locations such as the TA and SI patch panels. NTP and IRIG protocols typically provide accuracy of a millisecond or better.

When MCCS acquires HK data items for a given dataGroup – which includes one or more associated HK data items – it also records the related acquisition time. For some dataGroups there may be multiple timestamps representing different points during the data acquisition cycle. The following are the three main types of timestamp attribute:

- mcstime – default timestamp noting data reception by and in the PIS
- pkt_timestamp – close-to-the-source timestamp noting data reception, typically from DAS or TA sources
- sampletime – closest-to-the-source timestamp noting data reception, typically from DAS sources, before forwarding to the PIS

As with all other HK timestamp data is accessed via SCL `get` and `subscribe` commands, either using specific timestamp requests or using the global `attr` (attribute) keyword, which can take on the appropriate values: *none* (default), *mcstime*, *pkt_timestamp*, or *sampletime*. A number of relevant examples are shown below.

```
1 set attr=all
1 A
1 : attr="all"
2 get list=das.ic1080_10hz.pitch
2 : pitch=10.323(mcstime=2014-10-07T17:21:07.989Z
pkt_timestamp=2999-12-31T23:59:59.999Z)
```

Note how all the user requested was pitch, but they received both its value and its attribute values in the response. If the user just wishes to see just the time when the PIS acquired each data item (mcstime) or just the time that the data was acquired, they can use the following commands.

```
3 set attr=mcstime
3 A
3 : attr="mcstime"
4 get list=das.ic1080_10hz.pitch
4 : pitch=10.323(mcstime=2012-07-28T03:38:07.989Z)
5 set attr=sampletime
5 A
5 : attr="sampletime"
6 get list=das.ic1080_10hz.pitch
6 : pitch=10.323(sampletime=2012-07-28T03:38:35.854Z)
```

Note that any attribute name can be used as the value for the `attr` keyword.

To disable this behavior, use set `attr` to *none*.

```
8 set attr=none
8 A
8 : attr="none"
```

In each of the above examples, the value of attr is set globally for the user’s session, and will affect all subsequent get and subscribe commands until the user changes the value of attr or exits the session. However, the attr keyword can also be used as a keyword on those commands, which will cause the associated responses to follow the specific settings. Finally, time attribute values may also be obtained explicitly using a ".mcstime" notation attached to an HK data item.

```
9 get list=das.ins_1_50hz.pitch_sampletime
9 A
9 : das.ins_1_50hz.pitch_sampletime=2012-07-28T03:38:38.326Z
```

```
10 get list=ta_wfi.ccd_image_data.focus_position (attr=mcstime)
10 :
ta_wfi.ccd_image_data.focus_position=326.423(mcstime=2012-07-28T03:41:32.7
81Z)
```

In the above case, ccd_image_data is the name of a dataGroup with many individual data items, but with a single mcstime attribute for the whole data block.

Note that in all the above examples, `subscribe` could have been used instead of `get` to continuously obtain value updates.

3.4.2. Positions

A position provides the means to define and move to a specific location on the sky. Users typically define positions as absolute locations in a given reference frame, which are then used to move the telescope to those locations via various commands. The following subsections describe position attributes and their usage, a number of pre-defined positions, and the two methods to create user-defined positions.

3.4.2.1. Position-Related Attributes

Each defined position can contain a large number of attributes that describe the following:

- sky location -- coordinate information related to a specific RF
- data recording -- defines coordinate system for HK reporting and recording
- tracking -- defines AOI(s) to use for tracking (if desired)
- chopping -- defines an associated chop beam (if desired)

The full list is shown in Table 3-2. Note that not all must be used in a position definition; for example, tracking or chop information may not be relevant. In addition, many of the sky location coordinate attributes are mutually exclusive and dependent on the underlying coordinate system being used. For example, if the first coordinate attribute in a position definition is RA, then all other coordinate labels except Dec will result in an error message. The MCCS GUI shows the relevant attributes for a given reference frame, but a user writing a command or XML script to define a position must be aware of which coordinate labels are grouped together. The following subsections provide examples of defining and using positions with associated attributes from Table 3-2. The SI observing examples in Section 5 provide further insight on position definitions and usage of the various attributes.

Table 3-2: Postion-Related Attributes

Attributes	Reference Frame	Default Units	Type	Description
default			STRING	Name of position to get attribute values from. **Can only use this attribute for new position definitions**. For new positions defaults to *here*. Other position attribute values below, override attribute values given by this default attribute.

Location on Sky Attributes				
equinox			STRING	Equinox of the coordinate system
epoch			STRING	When used with planetary (or other solar system) coordinates, the epoch is the date at which the orbital elements were valid. When used with stellar positions, this is the date at which the position was valid (using the coordinate system specified by the equinox). The software will then apply (current - epoch) years of proper motion to figure out where the star is now.
ra	ERF	HHhMMmSS.Sss (sid_hr)	Sexagesimal Or FLOAT8	Position along the RA axis. Defaults to sexagesimal format (e.g.,12h13m14.53s but can be real format (e.g.,12.2207hours)
dec	ERF	DddMMmSS.Ss (degree)	Sexagesimal Or FLOAT8	Position along the declination axis. Defaults to sexagesimal format (e.g.,12d13m14.5s but can be real format (e.g.,12.2207 degrees)
ra_year	ERF	arcsec/year	FLOAT8	Annual proper motion along the RA axis. The Right Ascension of the target position will be the supplied RA + ra_year * (current-equinox) where the difference between current and equinox is in Julian years.
dec_year	ERF	arcsec/year	FLOAT8	Annual proper motion along the declination axis. The declination of the target position will be the supplied arcseconds per year + dec_year * (current-equinox) where the difference between current and equinox is in Julian years.
el	TARF	degrees	FLOAT8	Position about the elevation axis
xel	TARF	degrees	FLOAT8	Position about the cross-elevation axis
los	TARF	degrees	FLOAT8	Position about the Line of Sight axis
xwfi	WFIRF	pixels	FLOAT8	Position along the X axis of the WFI
ywfi	WFIRF	pixels	FLOAT8	Position along the Y axis of WFI
xffi	FFIRF	pixels	FLOAT8	Position along the X axis of WFI
yffi	FFIRF	pixels	FLOAT8	Position along the Y axis of FFI
xfpi	FPIRF	pixels	FLOAT8	Position along the X axis of FFI
yfpi	FPIRF	pixels	FLOAT8	Position along the Y axis of FPI
xsi	SIRF	pixels	FLOAT8	Position along the X axis of SI
ysi	SIRF	pixels	FLOAT8	Position along the Y axis of SI
lambda	EcRF	degrees	FLOAT8	Position along the ecliptic longitude axis
beta	EcRF	degrees	FLOAT8	Position along the ecliptic latitude axis
lii	GalRF	degrees	FLOAT8	Position along the galactic longitude axis
bii	GalRF	degrees	FLOAT8	Position along the galactic latitude axis
lsg	SGRF	degrees	FLOAT8	Position along the super-galactic longitude axis
bsg	SGRF	degrees	FLOAT8	Position along the super-galactic latitude axis
ephemeris		-	STRING	Name of the ephemeris file if position is non-sidereal

Recording Information Attributes				
coord_sys	-		STRING	Coordinate system for MCCS to report/record the Position. Defaults to coordinate system used to define the position.
Tracking Attributes				
centroid	-		STRING	AOI (set) to be used to position track
rof1	-		STRING	AOI (set) to track rotation (with the other supplied by rof2)
rof2	-		STRING	Second AOI (set) to track rotation (with the other supplied by rof1)
inertial	-		BOOL4	Is source being tracked inertial, and should the tracker correct for gyro drift? yes/no
boresight	-		BOOL4	Should the Tracking Position be coincident with the designated boresight? yes/no
Chopped Image Identification Attribute				
chop	-		STRING	The is an indicator to enable the observer to inform the MCCS which chopped image of the position he/she wishes to instruct the MCCS to observe. This is only relevant if chopping. Possible values are “plus”, “minus”, “zero”, “no_chop”.

3.4.2.2. Useful Pre-Defined Positions

In addition to user-defined positions, there are a number of pre-defined ones that can be used, each associated with particular attributes that cannot be changed by the user. The pre-defined positions are listed in Table 3-3, and are categorized as follows:

- general -- positions of general use
- boresight -- positions associated with the TA, SI, and imager boresights
- AOI/tracking -- positions used for tracking
- nod -- positions used for nodding
- pattern -- positions used for moving in complicated patterns

Table 3-3: Pre-defined positions supported in SCL.

Position Name	Position Description
General Positions	
actual	The current position seen by the designated boresight.
desired	The position of the designated boresight that will result from the execution of the current command.
cmded	Instantaneous position along the trajectory of the telescope generated to move from *actual* to *desired*.
reference	A generic user *reference* position saved for future use.
Boresight Positions	
tabs	TA boresight (this is the same as the origin of TARF – see Section 3.3)
sibs	SI boresight, as specified by the SI team or user in the *<si>_data.xml* file or as modified through <*si*>.sibs_change command.
wfi_fov	Position of the center of the WFI FOV.
ffi_fov	Position of the center of the FFI FOV.
fpi_fov	Position of the center of the FPI FOV.

AOI/Track-Point Positions	
trc_aoi1	Position defined as AOI #1.
trc_aoi2	Position defined as AOI #2.
trc_aoi3	Position defined as AOI #3.
trc_aoi4	Position defined as AOI #4.
trc_aoi5	Position defined as AOI #5.
trc_aoi6	Position defined as AOI #6.
trc_aoi7	Position defined as AOI #7.
trc_aoi8	Position defined as AOI #8.
trk_pos	Tracking position (valid while tracking). The tracking position is not the Area Of Interest where the tracking star is located; rather it is the position (by default the location of sky at which the SI boresight, sibs, is pointed) that will be kept fixed as a result of the tracking action at the AOI.
Nod Positions	
a	Position for nod-beam A.
b	Position for nod-beam B.
x	Position for nod-beam X.
Pattern Positions	
user_pat_pt_1	Pattern position #1.
user_pat_pt_2	Pattern position #2.

3.4.2.3. Creating and Using Positions

A position is most generally defined using the coord.position command. The name argument defines the position name, and all of its related attributes are then provided in parentheses. The following example defines a position named "star1" with attributes listed in parentheses to set coordinates RA=12h13m14.5s and Dec=–12d13m14s as well as a 1950 equinox.

```
1 coord.position name=star1(ra=12h13m14.5s dec=–12d13m14s equinox=b1950)
```

As shown in the example, the position name is associated with the "name=" command keyword, and the attributes are then listed together within parentheses in attN=valueN format and separated by whitespace, where attN corresponds to any option in Table 3-2. Once defined, a position is characterized by its assigned attributes, so that this position can be used in subsequent commands, as shown in the example below.

```
2 ta_pos.goto pos=star1
```

The system supports up to fifty (50) simultaneous user-defined positions. Unwanted positions can be deleted (via `coord.delete`) and new ones defined at will.

3.4.2.4. Inline Position Definition

In addition to the general method of position definiton via `coord.position` noted above, there is also a shorthand method known as inline position definition, or inlining. The inlining method allows one to define both a position name and its associated attributes together and then immediately use that position in a command that accepts a position argument. The example below shows how the two separate actions shown above -- (1) defining a position named "star1" with specific attributes, and then (2) moving to that position -- can be combined in a single SCL expression by use of inlining.

```
3 ta_pos.goto pos=star1(ra=12h13m14.5s dec=–12d13m14s equinox=b1950)
```

Because the inlining method simplifies commanding, it is often the preferred approach to position definition. A number of examples of inlining will be shown throughout the remainder of this document.

3.4.3. Aliases

All or part of an SCL command can be defined in an *alias*. The `alias` command is used to define an alias, while the `unalias` command is used to delete an existing alias. The name of the alias must begin with a letter, and may include any number of alphanumeric characters and underscores after the initial letter, and its value is a string. To use a defined alias, precede the alias with a "$" sign. Only one substitution pass is performed so aliases contained within the substituted string are not expanded. A few examples are shown below.

```
341 alias temps="get list=[das.gps_1_10hz.gps_lat das.gps_1_10hz.gps_lon
das.gps_1_10hz.gps_msl_alt]"
341 A
341 :
342 alias showAll="showUnits=yes showLabels=yes"
342 A
342 :
343 $temps
343 A
343 : das.gps_1_10hz.gps_lat=34.6135523790000
das.gps_1_10hz.gps_lon=-118.0754739010000
    das.gps_1_10hz.gps_msl_alt=2587.000000
344 $temps $showAll
344 A
344 : das.gps_1_10hz.gps_lat=34.6135523790000(units=deg)
    das.gps_1_10hz.gps_lon=-118.0754739010000(units=deg)
    das.gps_1_10hz.gps_msl_alt=2587.000000(units=ft)
```

The session environment includes predefined aliases that MOPS maintains in a configuration file (*global_alias.xml*). To see a list of aliases for your session, enter `get list=alias`, as shown in the example below; this is necessary, for example, to create a mapping from the alias defined in the *DCS-SI-01* and the actual housekeeping item.

```
345 get list=alias
345 : alias="define=tasim.mark,data=get
list=data_list,altitude=das.ins_1_25hz.hybrid_alt,gross_weight=das.ac_fuel
.gross_weight,

ground_speed=das.ins_1_25hz.hybrid_ground_speed,heading=das.ins_1_25hz.hyb
rid_true_heading,latitude=das.ins_1_12hz.hybrid_lat,

longitude=das.ins_1_12hz.hybrid_lon,pitch=das.ins_1_50hz.pitch,pressure_al
titude=das.adc_1_15hz.press_alt,roll=das.ins_1_50hz.roll,

static_air_temp=das.adc_1_2hz.static_air_temp,static_outside_air_pressure=
das.adc_1_2hz.static_outside_air_press,

track_angle=das.ins_1_25hz.hybrid_true_track_angle,true_airspeed=das.adc_1
_8hz.true_airspeed,wind_angle=das.ins_1_12hz.wind_angle,
      wind_speed=das.ins_1_12hz.wind_speed"
```

Most of the predefined aliases are HK data items associated with the aircraft avionics. There are several sources for most avionics items, and the preferred source may change from flight to flight, and may differ from the values in this document. Because of this, it is recommended that these items be referred to by their alias. For instance, in the example above the alias `$altitude` returns the value of *das.ins_1_25hz.hybrid_alt* and `$static_air_temp` returns the value of *das.adc_1_2hz.static_air_temp*. However, the specific HK parameter (source, sampling rate, etc.) may change based on a variety of considerations for any given flight.

When an alias name is used to access HK data (e.g. in a `get` or `subscribe` command) it is important to understand that the actual HK item name will appear on responses (if `showLabels=True` is used) and not the alias name. The user may use the position of the values in the response string to identify which value is associated with which requested item. Alternatively, the output of `get list=alias` may be used to create a mapping between alias and housekeeping value.

3.4.4. Users and Roles

All SCL sessions require a valid user and associated role. Each approved SCL user is assigned a unique user ID and one or more applicable roles with which to login to the MCCS. Each role has permissions to execute certain commands, and all roles can execute all the general commands (e.g., `login`, `subscribe`). The most relevant roles and their associated responsibilities are the following:

- ifd: The In-Flight/Mission Director is responsible for coordinating all observing activities in the main cabin.
- to: The Telescope Operator is responsible for controlling the TA and interacting with the science users during observations.
- fp: The Science Flight Planner is responsible for tracking the flight execution, and adjusting the plan as required.
- gi: A Guest Investigator is a visiting science user.
- epo: An Education and Public Outreach user is a flight participant as part of the mission's EPO efforts.
- instrument: This is the Science Instrument that is conducting observations. For example, the GREAT SI would log into the system as user *great_si* with role *instrument*.

3.4.5. Units

As noted previously, units are an attribute associated with some keywords and typically have fixed defaults that can be changed for a given command via the *units* attribute (e.g., `units=arcsec`). Table 3-4 lists the most common units useful to users; for others see Table 13 in Section 4.1.1 of MCCS_SI_04.

Table 3-4: Common units supported in SCL.

Full Unit Name	Accepted Abbreviation	I	Full Unit Name	Accepted Abbreviation
arcminutes	arcmin	I	micrometers	micron
arcminutes_per_second	arcmin/sec	I	millimeter_per_second	mm/sec
arcseconds	arcsec	I	millimeters	mm
arcseconds_per_second	arcsec/sec	I	milliseconds	msec
arcseconds_per_year	arcsec/yr	I	minutes	min
astromical_units	au	I	pixels	pix
degrees	deg	I	pixels_per_second	pix/sec
degrees_per_second	deg/sec	I	seconds	sec
hertz	hz	I	sidereal_hours	sid_hr
hours	hr	I	sidereal_minutes	sid_min
julian_years	yr	I	sidereal_seconds	sid_sec

3.4.6. Observing Metrics and Flags

Some other key concepts that all users should be aware of include the high-level observatory metrics, and the in-flight communications and science data collection flags that, among other things, contribute to those metrics.

There are three main observatory metrics that are generated from flight data to measure overall observatory performance:

- Research Hours (RH) -- Measures how many flight hours are spent at altitude and on target heading, with all systems up and ready to support science observing.
- Science Flight Hours (SFH) -- Measures how many flight hours are spent on science target legs and available for science data collection.
- Data Collection Time (DCT) Measures the actual time science data is collected.

The ratios of these metrics are efficiency measurements: DCT/SFH for how well the instrument makes use of the available science time to collect data; and SFH/RH for how well the observatory makes available time for collecting science data.

While in flight the mission team makes use of the following three HK status flags to help coordinate in-flight communications between the MD, TO, and SI personnel:

- `coord.obsstat_md` -- Used by the MD to indicate arrival on target heading or turning off it.
- `coord.obsstat_to` -- Used by the TO to indicate completion of TA setup and handover to the SI for observing, or for retaking TA control for reconfiguration activities.
- `coord.obsstat_si` -- Used by the SI to indicate when actively observing and not.

During the flight the MD, TO, and SI each set their respective flags to the appropriate values at the proper times in order to indicate changes in overall observing status, which are then made available for all users for situational awareness.

In addition, the MD and TO status flags govern the accummulator "clocks" for the RH and SFH observatory metrics, respectively. The DCT metric, however, is not governed by the SI status flag but by the following science data collection flags:

- `coord.si.integ` -- Used to indicate when actively integrating on a target to collect science data.
- `coord.si.exptime` -- Used at the end of an integration to record the associated integration time in seconds.
- `coord.si.utcstart` -- Used at the start of an integration to record the UTC start time of the integration.
- `coord.si.utcend` -- Used at the end of an integration to record the UTC end time of the integration.

The SI indicates the start of data collection by setting the integration flag *on*, and then ends the integration by setting the flag *off* while also setting the exposure time and associated UTC start and end times. This information is then used to calculate overall DCT.

For additional details on the observatory metrics, and the status and science data collection flags, see MCCS_SI_04 Section 3.3.6 and Appendix B.

4. Using SCL

The basic MCCS SCL observing session for SOFIA flows as follows:

1. Log in to the MCCS and create an SCL session.
2. Set the SI and its associated mode.
3. Define any required aliases.
4. Subscribe to HK.
5. Set the SI boresight.
6. Begin logging commands and responses.
7. Observe one or more targets with the SI.
8. Perform various inter-observation tasks.
9. Log out and destroy the SCL session.

The subsections below describe each of these steps, except the actual observing (#7), which is covered in Section 5.

4.1. Create a Session and Log In

To use SCL a user first must establish one or more socket connections with the MCCS for sending commands and accessing HK; SI users typically set up a separate connection for each. Each connection establishes an MCCS session, which can be done via the MCCS GUI or, more generally, via direct telnet by a user through a terminal window.

The text-based telnet protocol is fairly human-friendly, and is shown in the example below where an SI user connects to the standard IP address and port. Once the connection is acknowledged the user must log in, with a specific role, at which point general commanding can commence.

```
$ telnet 10.4.1.17 6555
Trying 10.4.1.17...
Connected to 10.4.1.17.
Escape character is '^]'.
1 login user=hipo_si role=instrument
1 A
1 :
```

4.2. Set the Instrument and Mode

Once logged in, the SI is defined for observing via the `coord.instrument` command, which prompts the MCCS to read the SI's XML configuration file and make its data available for use within the session. This XML file includes SI obsering mode and other definitions, which can then be accessed via the general SI SCL commands, which are prefixed with the SI name (see Section 3.3.5, and the fictitious SI named *rien* in Appendix A, of MCCS_SI_04). The SI observing mode is then defined via the SI's mode command.

This process is displayed in the following example, using the instrument GREAT. Command ID #22 is sent to verify that the SI was defined correctly. In practice, this command is sent first to avoid sending the `coord.instrument` command if the instrument has already been set correctly.

```
20 coord.instrument name=great
20 : message="DONE"
21 get list=coord.current_instrument
21 : coord.current_instrument="great"
22 great.mode current=great_standard
22 : message="DONE"
```

4.3. Define Aliases

If the user wants to define any aliases for the session, this presents a good time to do so. SI users may wish to coordinate with MOPS to make use of the list of the pre-defined aliases, which can be useful in creating a mapping between the aliases defined in the DCS_SI_01 FITS keyword dictionary and the corresponding HK. Details on defining aliases are described in Section 3.4.3, and Appendix B of MCCS_SI_04 lists all the available HK items.

4.4. Subscribe to HK

The next step in the process is to subscribe to relevant HK. Whereas the `get` command provides a one-time readout of the most recent value(s) for the requested HK item(s), the `subscribe` command provides on-going updates to the associated HK items so that changes can be seen as they occur. Users can also subscribe to alerts and alarms, which provide timely warnings and failure notices for the SOFIA subsystems.

The example below shows how the GREAT SI obtains the HK needed to populate the FITS headers of its science data. The responses for command ID #40 and #41 are each all on one line, but are broken up here for readability. An additional good subscription would be to the nodtrack_{a,b,x} and reftrack position HK items.

```
40 get list=alias
40 : alias="define=tasim.mark,data=get
list=data_list,altitude=das.ins_1_25hz.hybrid_alt,

gross_weight=das.ac_fuel.gross_weight,ground_speed=das.ins_1_25hz.hybrid_g
round_speed,

heading=das.ins_1_25hz.hybrid_true_heading,latitude=das.ins_1_12hz.hybrid_
lat,
  longitude=das.ins_1_12hz.hybrid_lon,pitch=das.ins_1_50hz.pitch,
  pressure_altitude=das.adc_1_15hz.press_alt,roll=das.ins_1_50hz.roll,

static_air_temp=das.adc_1_2hz.static_air_temp,track_angle=das.ins_1_25hz.h
ybrid_true_track_angle,

true_airspeed=das.adc_1_8hz.true_airspeed,wind_angle=das.ins_1_12hz.wind_a
ngle,
  wind_speed=das.ins_1_12hz.wind_speed,
  static_outside_air_pressure=das.adc_1_2hz.static_outside_air_press"

41 subscribe list=[mission_id great.si_config.current_mode coord.obsstat_md
coord.obsstat_si
  coord.obsstat_to coord.pos.sibs.alt coord.pos.sibs.azim
coord.pos.sibs.dec coord.pos.sibs.el
  coord.pos.sibs.equinox coord.pos.sibs.los coord.pos.sibs.ra
coord.pos.sibs.vpa coord.pos.sibs.xel
  coord.sky_los_rate das.adc_1_2hz.static_air_temp
das.adc_1_2hz.static_outside_air_press
  das.adc_1_8hz.true_airspeed das.ic1080_10hz.baro_alt
das.ic1080_15hz.press_alt
  das.ins_1_12hz.hybrid_lat das.ins_1_12hz.hybrid_lon
das.ins_1_25hz.hybrid_alt
  das.ins_1_25hz.hybrid_ground_speed das.ins_1_25hz.hybrid_true_heading
  das.ins_1_25hz.hybrid_true_track_angle fltexec.leg_data.leg_seq
nod.amplitude nod.coord_sys
  nod.current nod.pos_angle sma.chop.frequency sma.chop.phase
sma.chop.profile sma.chop_symmetry
  sma.sky_amp2 sma.sky_amplitude sma.sky_angle sma.sky_coord_sys
sma.sky_tilt sma.sky_tip
  ta_mcp.mcp_hk_pms.pms_temp_1 ta_mcp.mcp_hk_pms.pms_temp_2
ta_mcp.mcp_hk_pms.pms_temp_3
  ta_mcp.mcp_hk_pms.sm_temp_1 ta_pos.time_of_last_rewind ta_pos.track.state
ta_scs.fcm_status.fcm_act_t
  ta_scs.scs_status.scs_state ta_state.tsc_status
ta_trc.trc_status_table.main_op_mode_id
  ta_tsc.tsc_mcs_hk.fbc_status ta_tsc.tsc_mcs_hk.fd_status
wvm_if.wvmdata.water_vapor]
```

4.5. Set the SI Boresight

The last step required before beginning observations is to ensure alignment of the SI boresight (SIBS) with that of the TA. Although an estimated alignment is provided in the SI's XML configuration file, minor tweaks are generally required due to slight differences each time the SI is installed on the Observatory. The boresight alignment is usually accomplished, for SIs with enough pixels to determine the centroid of the target, by staring at a bright target. For single-pixel and similar SIs, a cross scan is performed across such a target and the boresight moved appropriately until the SI signal is maximized. Once the alignment is determined, the information is commanded into the MCCS for general use.

The example below shows an alignment example using the GREAT SI. The first sibs_change command moves the SI boresight to the desired position in the focal plane; the second one is only needed if it is desirable to redefine the origin of SIRF at (0,0) after the first move. The `si_boresight` command translates the SIBS position in the focal plane to the TARF coordinate system, which is necessary to define the axis of rotation for LOS rewinds. Since LOS rewinds are always performed around the SIBS position, the TA needs to know the position to rotate around if SIBS is redefined. The `si_boresight` command defines this up-to-date relationship between SIBS and TARF after the `sibs_change` command. Finally, the active boresight is set to be on the SI (sibs) for pointing.

```
50 great.sibs_change xsi=2.3619 ysi=2.3619 move=yes
51 great.sibs_change xsi=0. ysi=0. move=no
52 great.si_boresight pos=great_target_temp(xsi=2.3619 ysi=2.3619
chop=no_chop)
53 ta_pos.track centroid=trc_aoi3 rof1=trc_aoi1 rof2=trc_aoi2 inertial=yes
54 coord.set boresight=sibs
```

4.6. Begin Logging Commands and Responses

Although the MCCS saves all its commands, and their final responses (only), to a log file (/archive/mission_data/cmdlog_<timestamp>.txt), this file is often difficult to monitor in flight (but is an excellent information source for seeing what happened and when). However, it is very helpful for each user to communicate via SCL with other users by sending pertinent comments -- commands issued, progress made, problems encountered, etc. -- so that all users remain aware of all activities and can confirm proper telescope operation and respond quickly to any MCCS error conditions.

Such comments can be published into HK via the `coord.set` command, for subscription by interested users. Comments for all users are also optionally logged in text files in their respective MCCS work areas. We suggest that the user name such log files using the Mission ID, such as <mission_id>.txt or, when multiple SIs are flying concurrently, as <mission_id>-<si>.txt. If the Mission ID is not available, which may be the case in a simulation lab, use the current date in ISO-8601 (yyyy-mm-dd) format.

In the example below, the user *great_si* logs in, selects a log file named 2014-05-21_GR_F175.txt, and sends comments on those commands to the log file in

/archive/mission_data/mcs_users/great_si/comment_files/2014-05-21_GR_F175.txt

the contents of which are shown below based on the example.

```
1 login user=hipo_si role=instrument
1 A
2 coord.set comment="1 login user=hipo_si role=instrument"
2014-05-21_GR_F175.txt
2 A
1 :
3 coord.set comment="1 :" 2014-05-21_GR_F175.txt
3 A
2 : message="DONE"
3 : message="DONE"
```

```
$ cat
/archive/mission_data/mcs_users/great_si/comment_files/2014-05-21_GR_F175.
txt
2014-05-21T21:28:44.051Z  1 login user=hipo_si role=instrument
2014-05-21T21:28:48.265Z  1 :
```

4.7. Perform Inter-Observation Tasks

Between and during observations there are generally a variety of tasks that need to be performed, including the following: performing LOS rewinds, updating SMA focus, and updating observing flags. These are described in the following subsections.

4.7.1. Perform LOS Rewinds

While observing, the SI Team members can visually monitor the TA status via the MCCS GUI's Telescope Status Display (TSD). During such observations, the TA rotates to follow the field-of-view up to +/-2.5 degrees, at which point it reaches the line of sight axis rotation limit and has to be rewound. Depending on where the target is located in the sky, it is considered either a fast, medium, or slow rotator, which will dictate when the telescope needs to be rewound and how far it should be rotated; this information will be provided to the observer by the TOs. The current rotator convention is shown in the table below, and is governed by the following HK items:

- ta_state.fd_deg_los -- Shows how close the TA is to the LOS limit.
- coord.sky_los_rate -- Sign shows the positive/negative direction of the rotator (although the value may not be accurate).

Rotator Speed	Trigger At	Rewind To
Slow (<0.1 degree/minute)	+/- 1	+/- 1
Medium	+/- 1.5	+/- 1.5
Fast (> 2.5 degree/minute)	+/- 1.5	+/- 2

For example, when a fast positive rotator (per sign of `coord.sky_los_rate`) reaches the +1.5 degree trigger value, then rewind the TA to -2.0 degrees via the `ta_pos.rewind` command. This command is typically issued between observations. This is shown in the example below while not chopping.

```
1 ta_pos.rewind beam=0 los=-2.0
1 : message="DONE"
```

4.7.2. Update SMA Focus

Another action taken between observations is focusing the TA via the SMA, which is done periodically to account for temperature fluctuations. Although this should be done only when required, continuous updates should be avoided to preserve the longevity of the SMA mechanisms.

The threshold at which a focus adjustment is required -- when the current focus exceeds the calculated focus by X microns -- is SI dependent, and may be as small as 10 microns for short-wavelength SIs, but as much as 25+ microns for long-wavelength ones. The relevant HK for monitoring SMA focus is as follows:

- ta_scs.fcm_status.fcm_act_t -- Current/actual SMA focus value.
- sma.focus_fcm_t_calc -- Calculated SMA best-focus based on current temperature.

In the following example a focus change was commanded via `sma.focus`, telling the SMA to set the current/actual value to the calculated best-focus one.

```
2 sma.focus temp_focus=use
2 : message="DONE"
```

4.7.3. Update Observing Flags

The final major between-observation activity is that of updating the appropriate observing flags, which are described in Section 3.4.6. First, the proper coordinated setting of the observing flags (coord.obsstat_{md,to,si}) by the MD, TO, and SI are critical for keeping all on-board users informed of the overall observing status. Second, proper use of detector integration flags by the SI are crucial for recording data collection time periods. Collectively, these flags are then used post-flight in the generation of the observatory metrics -- RH, SFH, and DCT -- for the flight.The following example shows a simple typical flag sequence, where the MD communicates "on heading" (`obsstat_md=3`), the TO communicates "ready" (`obsstat_to=3`) to handover to the SI, the SI communicates "observing" (`obsstat_si=3`) and then conducts and integration. Once all integrations are complete the SI, TO, and MD will reset their flags (obsstat_{md,to,si}=2).

```
10 coord.set obsstat_md=3
10 : message="DONE"
20 coord.set obsstat_to=3
20 : message="DONE"
30 coord.set obsstat_si=3
30 : message="DONE"
31 coord.set si.integ=1
31 : message="DONE"
32 coord.set si.integ=0 si.exptime=30.0 si.utcstart=1382692015.284
si.utcend=1382692050.422
32 : message="DONE"
33 coord.set obsstat_si=2
33 : message="DONE"
21 coord.set obsstat_to=2
21 : message="DONE"
11 coord.set obsstat_md=2
11 : message="DONE"
```

4.8. Log Out and Destroy a Session

When done using SCL the user logs out, which destroys the session by closing the socket to MCCS. Before logging out we recommend that the user cancel all outstanding subscriptions. The following example shows such a subscription cancellation and subsequent logout.

```
70 cancel cmdid=41
70 : message="Cancellations complete."
71 logout
71 A
71 :
```

5. SI Observing

This section describes the observing modes that the SI uses to perform infra-red astronomy. A number of SCL examples are also provided; in them long single lines have been wrapped for readability, and SCL responses have been suppressed.

5.1. Stare Mode

In stare mode the Observatory uses the designated SI boresight location within the focal plane to point the telescope at a given RA and Dec on the sky. The chopper remains centered and static so that there is only a single image of the sky in the focal plane, which is tracked throughout the observation. An example image from the FPI is shown in Figure 5-1 below.

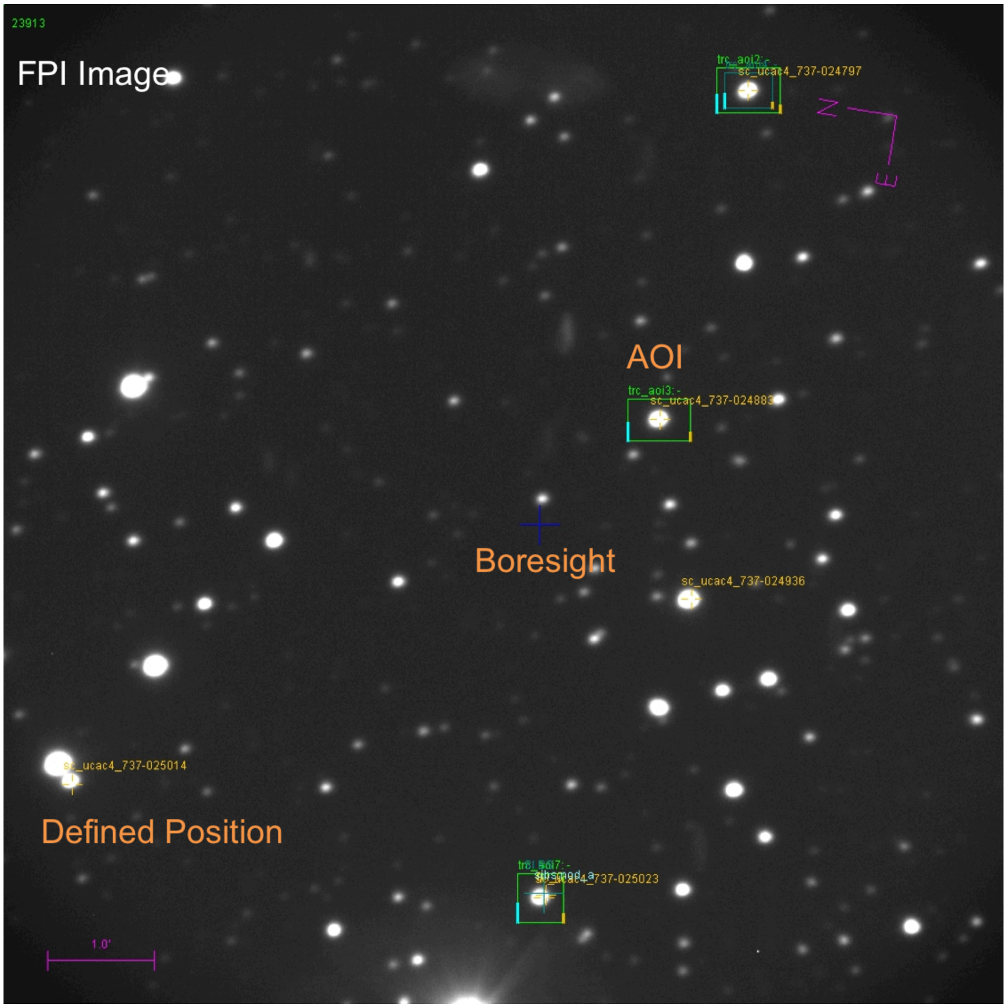

Figure 5-1: FPI screen image showing star field with boresight, position, and AOI overlays.

The example below from GREAT shows a stare mode observation with absolute moves in ERF; the following describes each command:

1. Define the absolute position `great_target` inline for the reference position, then move to it and begin tracking on AOI5 (as defined by the TO).
2. SI integrates to gather data.
3. Integration is complete.
4. Redefine and update the absolute position `great_target` inline for the target position, then move to it and begin tracking on AOI3.
5. SI integrates to gather data.
6. Integration is complete.

```
1 ta_pos.goto pos=great_target(ra=193.87125(units=degrees)
dec=-2.970417(units=degrees)
   centroid=trc_aoi5 inertial=yes chop=no_chop ) track_mode=centroid
2 coord.set si.integ=1
3 coord.set si.integ=0 si.exptime=10 si.utcstart=1402630574.0
si.utcend=1402630584.0
4 ta_pos.goto pos=great_target(ra=193.85039(units=degrees)
dec=-2.970417(units=degrees)
   centroid=trc_aoi3 inertial=yes chop=no_chop) track_mode=centroid
5 coord.set si.integ=1
6 coord.set si.integ=0 si.exptime=40 si.utcstart=1402630600.0
si.utcend=1402630640.0
```

5.2. Chop Mode

Chop mode is an IR observing technique for facilitating subtraction of sky background radiation, in which the SMA chopper is under a control loop tip/tilt oscillation. SCL supports two- and three-point chop positions with the TTL sychronization signal being provided either internally by the SMA or externally by the SI. This point/synchronization combination results in four basic chop setups: two-point internal, two-point external, three-point internal, and three-point external. There is also a fifth setup, an external analog one, which is completely driven by the SI electronics through special connectors. Because the SMA oscillates at relatively small angles (e.g., 30" to 5') and high frequency (e.g., 1-5 Hz), the user will see multiple target images in the SI and FPI FOVs (if using the dichroic secondary mirror), one for each chop point. The chop boresight positions are internally pre-defined in SCL for both two-point (plus/minus/no_chop) and three-point (plus/zero/minus/no_chop) chops. Typically, the plus position is used on-source and the minus one off-source. The major chop setup parameters are the number of chop points, chop frequency, position angle relative to the noted reference frame, amplitude(s) between the various chop positions, SMA tip/tilt settings, and synchronization source. Figure 5-2 shows an example of a two-point chop with the source in the plus position on the SI boresight; note the two offset images of the FOV, one for each chop position. This phenomenon can be seen in the FPI image in Figure 5-3, which shows duplicate stars visible during a two-point chop. Chop mode can be combined with other modes (e.g., nod) as discussed in Section 5.6.

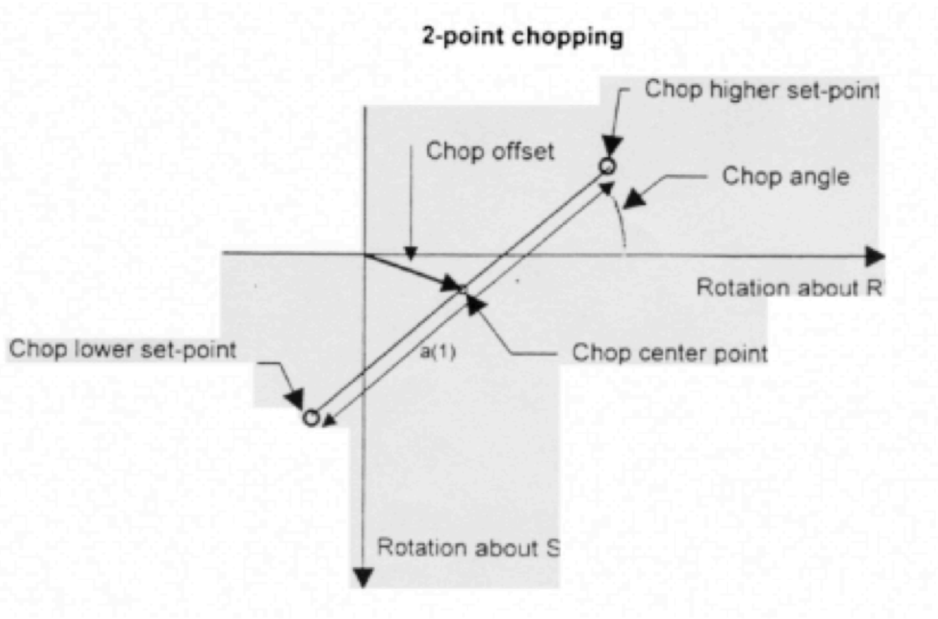

Figure 5-2: Example 2-point chop with IR source on the SI boresight.

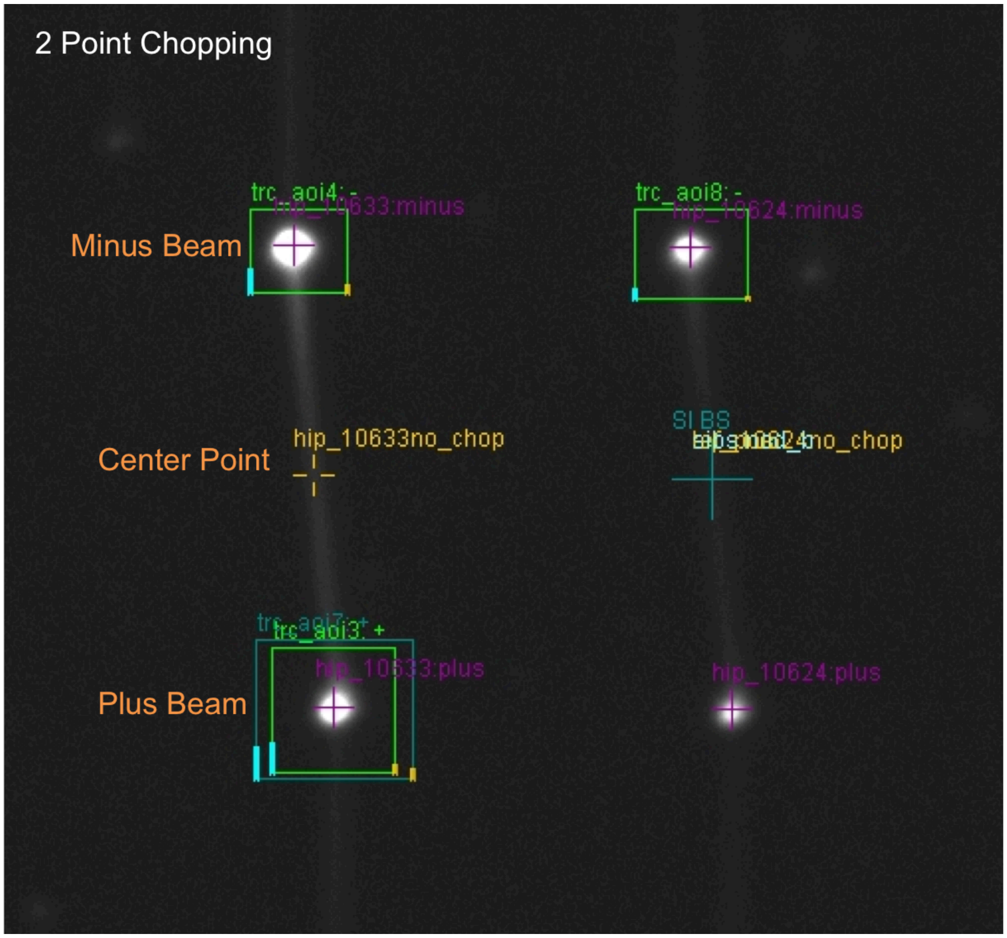

Figure 5-3: FPI screen image showing 2-point chop with plus and minus beams.

5.3. Nod Mode

Nod mode, or beam-switching, is an IR observing technique for facilitating subtraction of thermal radiation from the telescope itself. In nod mode the telescope is moved back and forth between a number of fixed positions, or *beams*, on the sky. The chopper is centered and static so there is only a single image of the sky in the focal plane. SCL allows the user to easily set up a nod sequence, which is an ordered series of two (A-B) or three (A-B-X) nod-beams on the sky, with associated tracking information; typically one beam will be on-source and the other(s) off-source. The nod sequence definition includes the number of beams, the position angle of the beams relative to the noted reference frame, and the amplitude(s) between and associated track position(s) for each beam. Nod mode can be combined with other modes (e.g., chop) as discussed in Section 5.6. Figure 5-4 shows and example of a two-beam nod, with the IR source on the SI boresight in the nod-A beam. The FPI image in Figure 5-5 illustrates a visible star in both the nod-a and nod-b positions.

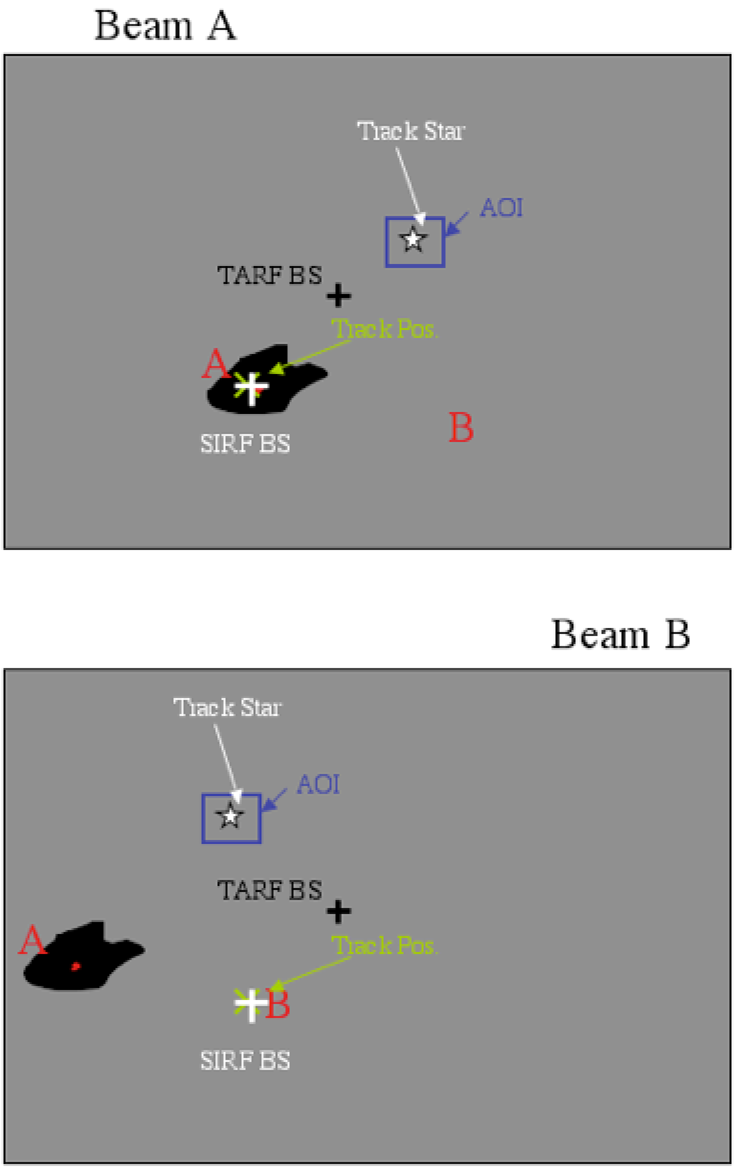

Figure 5-4: Example 2-beam nod, with IR source in nod-A on SI boresight.

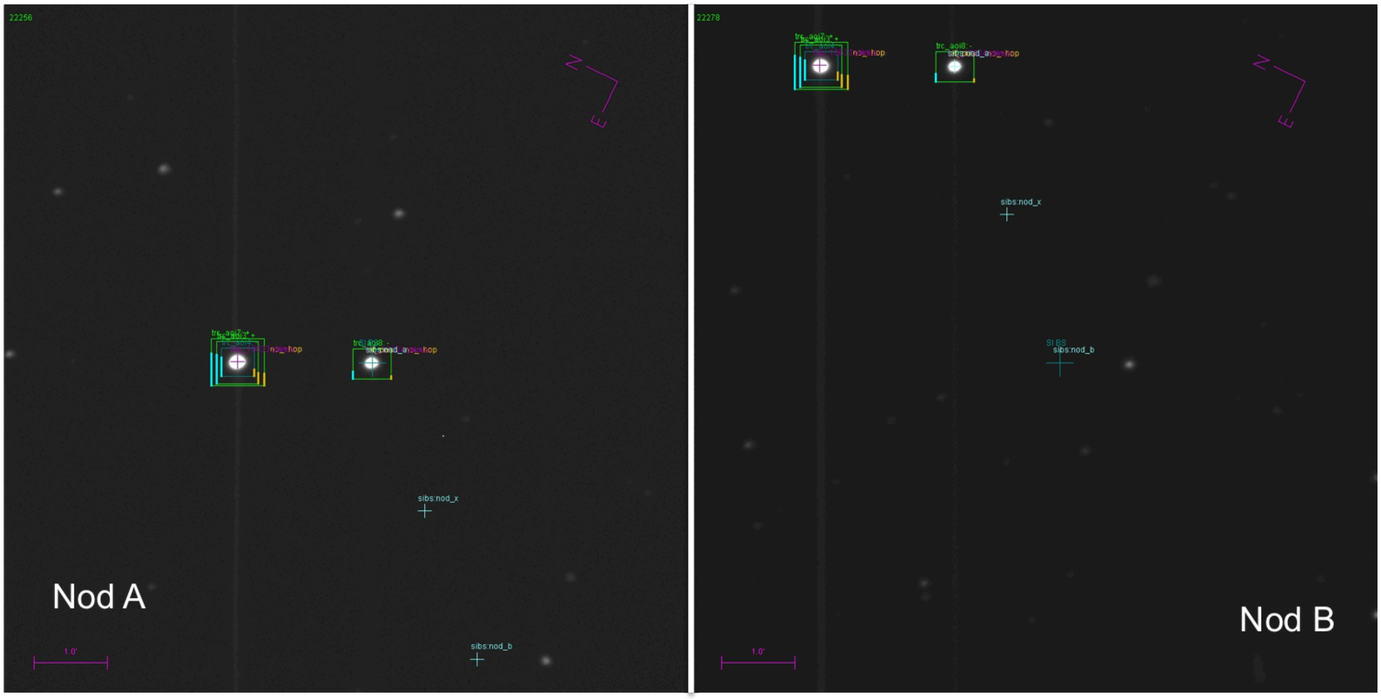

Figure 5-5: Pair of FPI screen images showing the nod-a and nod-b positions.

5.4. Dither Mode

In dither mode the telescope is moved to each of an arbitrary number of discrete sky positions that in the end return to the starting point (e.g., to sample regions of an extended source). The user defines the desired positions and then links them together into a dither sequence using the `dither.define` command. The TA is then moved to each of the dither positions using the `dither.goto` command. An example of a nine point dither is shown in Figure 5-6. Dither mode can be combined with other modes (e.g., chop and nod) as discussed in Section 5.6.

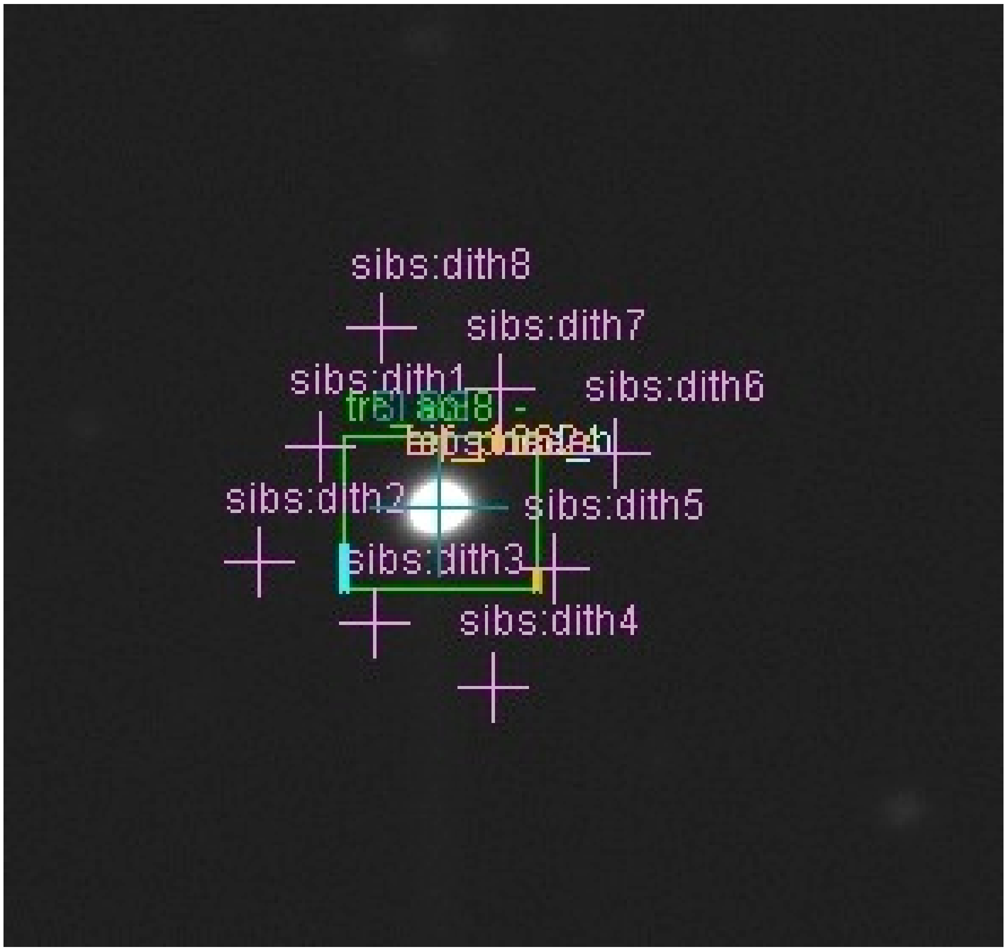

Figure 5-6: Example nine-point dither pattern shown in the FPI with GUI-generated overlays.

5.5. Scan Mode

In addition to discrete pointings on the sky, the TA can also be used in scan mode to execute smooth, controlled motions across the sky in order to quickly build up an image of an extended source with minimum duty cycle. To define a scan pattern in SCL the user provides the start/end points/positions, scan rate, reference frame, start time, and chop position. When executed, the TA will then perform the associated great-circle, constant velocity scan. Scan mode can be combined with other modes (e.g., chop) as discussed in Section 5.6 and shown in Figure 5-7.

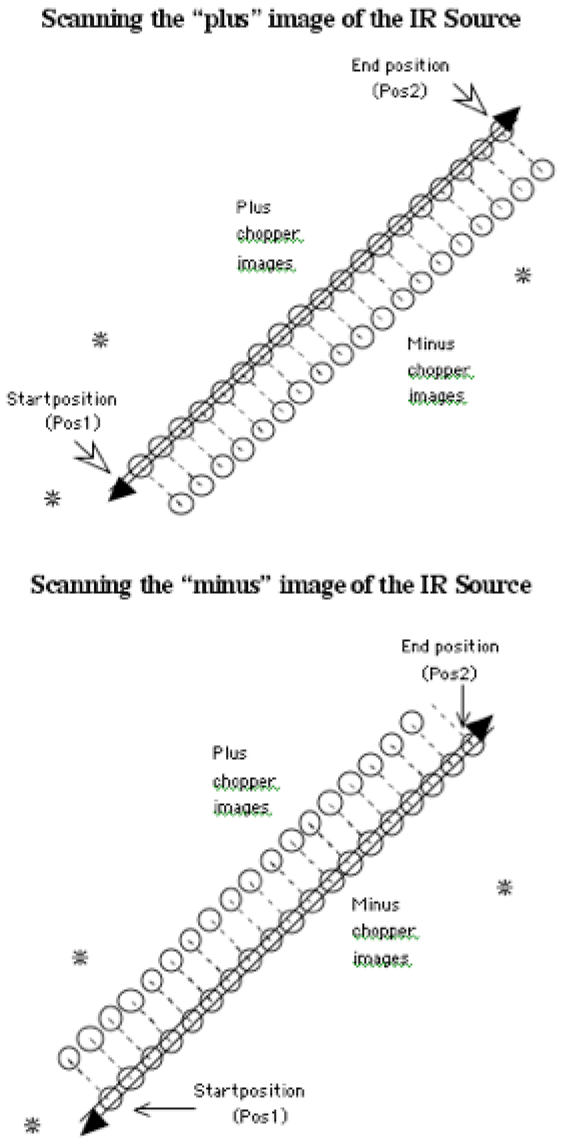

Figure 5-7: Example scan with chop: initial scan in chop-plus position (left) and return scan in chop-minus (right).

The example below from GREAT shows the process for observing in scan mode. The target was Saturn, and scans along the x- and y-axes were conducted in SI coordinates as part of verifying the GREAT spectrometer's boresight. The following describes each command in detail:

1. Define the absolute position `great_target` inline, then move to it and begin tracking on AOI3 (as defined by the TO).
2. Define and move to the relative position `great_target_relative` in SI coordinates and begin tracking on AOI3 (as defined by TO).
3. Turn off tracking as required by the `ta_pos.pattern` command.
4. Define and execute a smooth x-axis scan in SI coordinates.
5. SI integrates to gather data during the scan.
6. Integration is complete.
7. Return to starting absolute position of `great_target` and resume tracking.
8. Redefine and move to the relative position `great_target_relative` in SI coordinates and begin tracking on AOI3 (as defined by TO).
9. Turn off tracking as required by the `ta_pos.pattern` command.
10. Define and execute a smooth y-axis scan in SI coordinates.
11. SI integrates to gather data during the scan.
12. Integration is complete.

```
1 ta_pos.goto pos=great_target(ra=193.87125(units=degrees)
dec=-2.970417(units=degrees)
   centroid=trc_aoi3 inertial=yes chop=no_chop) track_mode=centroid
2 ta_pos.goto pos=great_target_relative(xsi=-16.740971428571427 ysi=1.711
   centroid=trc_aoi3 inertial=yes chop=no_chop) track_mode=centroid
3 ta_pos.track track_mode=off
4 ta_pos.pattern coord_sys=sirf points=[[4.6876 1.711] [41.83045714285714
1.711]]
   chop=no_chop rate=6.0 start_time=09:06:32.472 type=smooth return=false
repeat=1
5 coord.set si.integ=1
6 coord.set si.integ=0 si.exptime=21 si.utcstart=1402630800.01
si.utcend=1402630821.01
7 ta_pos.goto pos=great_target(ra=193.87125(units=degrees)
dec=-2.970417(units=degrees)
   centroid=trc_aoi3 inertial=yes chop=no_chop) track_mode=centroid
8 ta_pos.goto pos=great_target_relative(xsi=4.6876 ysi=-16.14614285714286
   centroid=trc_aoi3 inertial=yes  chop=no_chop) track_mode=centroid
9 ta_pos.track track_mode=off
10 ta_pos.pattern coord_sys=sirf points=[[4.6876 1.711] [4.6876
38.853857142857144]]
    chop=no_chop rate=6.0 start_time=09:07:18.043 type=smooth return=false
repeat=1
11 coord.set si.integ=1
12 coord.set si.integ=0 si.exptime=21 si.utcstart=1402630842.30
si.utcend=1402630863.30
```

5.6. Combined Modes

Using SCL the basic functional building blocks described in the previous sections – stare, chop, nod, dither, and scan – can be combined in many ways to conduct more complicated science observations. The following are some typical examples based on observing with the FORCAST and GREAT instruments:

- Stare while slewing: Useful for conducting *sky dips* to measure the sky background variation vs. EL (e.g., with a spectrometer).
- Stare and scan: Useful for conducting a *peak-up* to refine the SI boresight location (e.g., on a spectrometer).
- Chopped scan: Useful for scanning a region while also gathering sky background information; an example was shown in Figure 5-7.
- Chop/nod: Useful for ensuring proper sky and instrument background subtraction from the source. The chop/nod combinations are often defined as either *matched*, with coincident chop/nod beams (A-plus and B-minus) in the same direction, or *orthogonal*, where nod and chop directions are perpendicular. Figure 5-8 shows an example of a matched two-point chop and two-beam nod sequence, while a similar matched two point chop and two-beam nod is shown in the FPI image in Figure 5-9.
- Chop/nod/dither: An extension of chop/nod to move the overall pattern over an extended area of sky.

The possible combinations are numerous based on the imagination of the user.

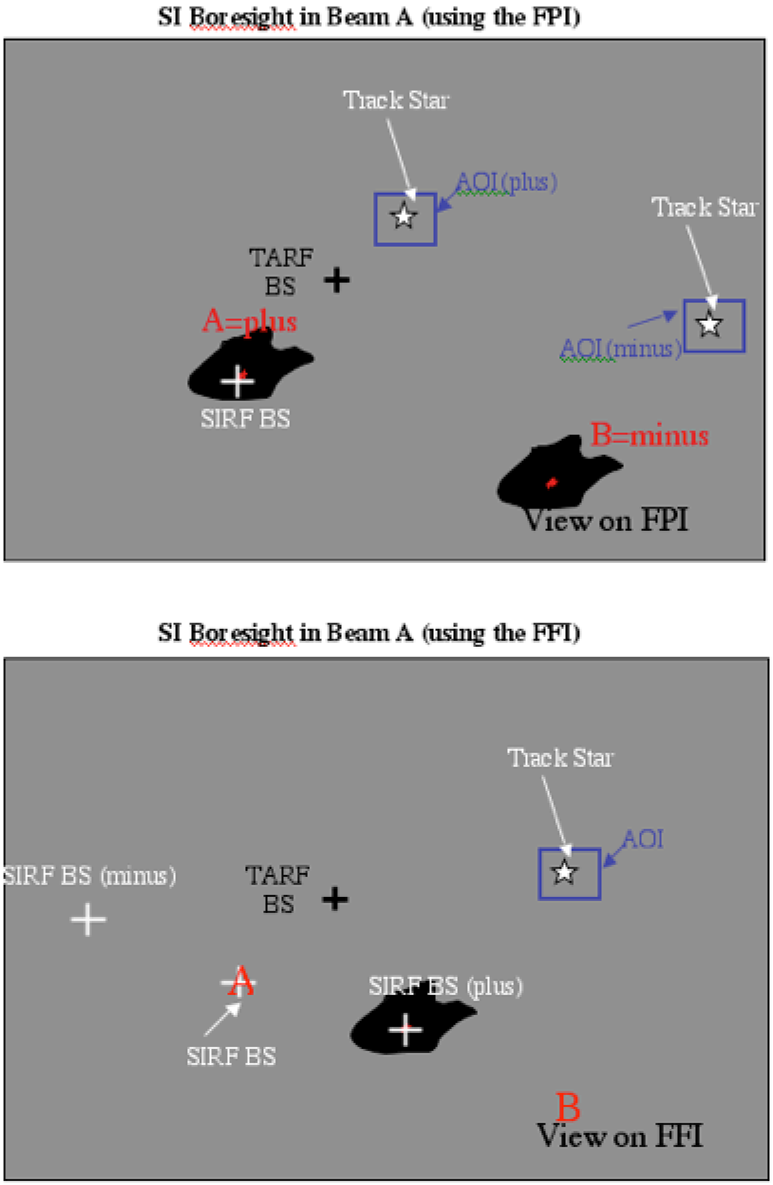

Figure 5-8: Example of matched two-point chop and two-beam nod (A-plus, B-minus).

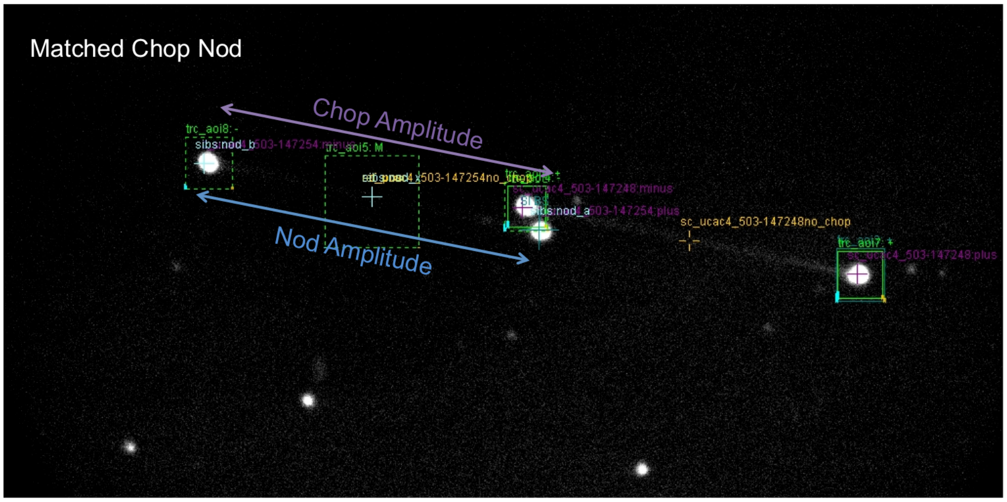

Figure 5-9: FPI screen image of a matched chop nod observation.

This example from GREAT shows the process for observing in combined chop/nod mode, in particular a matched parallel three-beam nod with two-point chop; the following steps describe the associated commands:

1. Set up and initialize a two-point chop using external/SI sync.
2. Define the nod position `great_target_plus` at the chop-plus location and associate it with tracking via AOI5 (as defined by the TO).
3. Define the nod position `great_target_minus` at the chop-minus location and associate it with tracking via AOI5.
4. Define the nod position `great_target_no_chop` at the no-chop location and associate it with tracking via AOI5.
5. Move to the nod-A beam position on the sky and begin tracking.
6. Define the 3-beam nod sequence for observing from the nod-A beam position.
7. SI integrates to gather data.
8. Integration is complete.
9. Move to the nod-B beam position on the sky.
10. SI integrates to gather data.
11. Integration is complete.

```
1 sma.chop settling_time=5. pos_ang=270.0 coord_sys=ERF sync_src=external
  amplitude=90.0 profile=2 tip=0. tilt=0.
2 coord.position action=ignore name=great_target_plus(chop=plus
ra=193.52227(units=degrees)
  dec=-2.8222175(units=degrees) centroid=trc_aoi5 inertial=yes)
3 coord.position action=ignore name=great_target_minus(chop=minus
ra=193.52227(units=degrees)
  dec=-2.8222175(units=degrees) centroid=trc_aoi5 inertial=yes)
4 coord.position action=ignore name=great_target_no_chop(chop=no_chop
ra=193.52227(units=degrees)
  dec=-2.8222175(units=degrees) centroid=trc_aoi5 inertial=yes)
5 ta_pos.goto pos="great_target_plus"(chop=plus) track_mode=centroid
6 nod.define profile=3 amplitude=90.0 amp2=90.0 pos_angle=270.0
  track_pos_a=great_target_plus track_pos_b=great_target_minus
  track_pos_x=great_target_no_chop coord_sys=ERF
7 coord.set si.integ=1
8 coord.set si.integ=0 si.exptime=40 si.utcstart=1402630800.01
si.utcend=1402630840.01
9 nod.goto pos=b
10 coord.set si.integ=1
11 coord.set si.integ=0 si.exptime=40 si.utcstart=1402630900.01
si.utcend=1402630940.01
```

Another example below, this time from FORCAST, shows the process for observing in combined dither/nod/chop mode, in particular a three-position dither with a three-beam nod and orthogonal two-point chop; the following steps describe the associated commands:

1. Define a base dither position at SI pixel position (0,0) with associated tracking and chop position information.
2. Define a `dith1` dither position to be applied relative to the base, 6” in increasing-x direction and 10” in increasing-y, with associated tracking and chop position information.
3. Define a `dith2` dither position to be applied relative to the base, 10” in increasing-x direction and 6” in decreasing-y, with associated tracking and chop position information.
4. Define the dither set with all offsets relative to those of base.
5. Define a `nodtracka` position specifying tracking-enabled using the TO-defined AOI3 for the nod-A position (as defined below).
6. Define a `nodtrackb` position specifying tracking-enabled using AOI3 for the nod-B position (as defined below).
7. Get the current SI boresight location on the sky.
8. Define a target position using the result of the previous command called `81_0025_2` for easy recovery.
9. Set up the chop parameters and begin chopping.
10. Verify the chopper status and paramters in HK.
11. Define a 3-beam nod with A and B positions for observing, and X for convenient control.
12. Once defined, the current position is nod-A, so move to nod-X, half-way to nod-B.
13. Apply the same amplitudes with opposing angle to redefine the nod set symmetric about the target position.
14. Nod to nod-A redundantly to turn on tracking.
15. Move to base dither position.
16. Nod to nod-A redundantly to enable tracking.
17. SI integrates to gather data.

18. Integration is complete.
19. Nod to nod-B.
20. SI integrates to gather data.
21. Integration is complete.
22. Move to `dith1` dither position.
23. Nod to nod-A.
24. SI integrates to gather data.
25. Integration is complete.
26. Nod to nod-B.
27. SI integrates to gather data.
28. Integration is complete.
29. Move to `dith2` dither position and continue the cycle as desired.

```
1 coord.position action=ignore name=base(chop=no_chop coord_sys=SIRF
guide_star=no
  inertial=yes xsi=0.00(units=arcseconds) ysi=0.00(units=arcseconds))
2 coord.position action=ignore name=dith1(chop=no_chop coord_sys=SIRF
guide_star=no
  inertial=yes xsi=6.00(units=arcseconds) ysi=10.00(units=arcseconds))
3 coord.position action=ignore name=dith2(chop=no_chop coord_sys=SIRF
guide_star=no
  inertial=yes xsi=10.00(units=arcseconds) ysi=-6.00(units=arcseconds))
4 dither.define pos=[base dith1 dith2]
5 coord.position action=ignore name=nodtracka(chop=no_chop coord_sys=SIRF
guide_star=no
  inertial=yes ra_year=0.0 dec_year=0.0 epoch=2000.0
  xsi=0(units=arcseconds) ysi=0(units=arcseconds) centroid=trc_aoi3)
6 coord.position action=ignore name=nodtrackb(chop=no_chop coord_sys=SIRF
guide_star=no
  inertial=yes ra_year=0.0 dec_year=0.0 epoch=2000.0
  xsi=0(units=arcseconds) ysi=0(units=arcseconds) centroid=trc_aoi3)
7 get list=[coord.pos.sibs.ra coord.pos.sibs.dec] showlabels=false
8 coord.position name=81_0025_2(coord_sys=erf ra=2.5303944 dec=89.2642222
chop=no_chop) action=ignore
9 sma.chop amplitude=30.00 coord_sys=SIRF phase=0 pos_ang=-70.00 profile=2
  settling_time=5.0 sync_src=external tip=0.00 tilt=0.00
10 get list=[sma.sky_amplitude ta_sma.tcm_chopper_status.tcm_amplitude_2
sma.sky_angle] showlabels=false
11 nod.define coord_sys=SIRF profile=3 pos_angle=20.0000 amplitude=30.00
amp2=30.00
12 nod.goto pos=x
13 nod.define coord_sys=SIRF profile=3 pos_angle=-160.0000 amplitude=30.00
amp2=30.00
   track_pos_a=nodtracka track_pos_b=nodtrackb
14 nod.goto pos=a
15 dither.goto pos=base
16 nod.goto pos=a
17 coord.set si.integ=1
18 coord.set si.integ=0 si.exptime=40 si.utcstart=1402630400.00
si.utcend=1402630440.00
19 nod.goto pos=b
20 coord.set si.integ=1
21  coord.set si.integ=0 si.exptime=40 si.utcstart=1402630500.00
si.utcend=1402630540.00
22 dither.goto pos=dith1
23 nod.goto pos=a
24 coord.set si.integ=1
25 coord.set si.integ=0 si.exptime=40 si.utcstart=1402630600.00
si.utcend=1402630640.00
26 nod.goto pos=b
27 coord.set si.integ=1
28 coord.set si.integ=0 si.exptime=40 si.utcstart=1402630700.00
si.utcend=1402630740.00
29 dither.goto pos=dith2
```

6. SI Testing and Integration

Before any new SI is allowed to observe on SOFIA it must pass through a number of testing and integration activities to ensure that it is ready for such observations. First, a so-called *SI Ops Team* is formed consisting of SOFIA Mission and Science Operations personnel -- MD, TO, SFP, Instrument Scientist, and Instrument Operator -- who are dedicated to the new SI. The new SI's developers then interface with the SI Ops Team, mainly through the associated MD and TO.

The second major step is to generate the <si>_data.xml configuration file that establishes the SI's interface with the MCCS, which includes the following SI information:

- SCL command and response items
- Alerts and alarms
- HK values
- Defaults: mode, focus, scaling, boresight pixels, etc.

With the XML configuration file established, the team the proceeds to the third step, which is to conduct Tier Testing to establish SCL functionality. This testing verifies the interface and configuration definitions between the MCCS and SI, and proceeds through four incrementally increasing levels of complexity:

- Tier 1, Basic Connectivity: Create a TCP/IP connection to MCCS session and issue successful login and logout commands.
- Tier 2, HK Data Acquisition: Check XML configuration file, alerts/alarms, HK acquisition (SCL subscribe command).
- Tier 3, Basic SCL Command Handling: Verify error handling, and integrated SCL command formatting, submission, response (coord, nod, chop, ta_pos, etc.).
- Tier 4, Observing Scenarios: Exercise observing scripts in the same manner as to be used in flight.

Once the SI has successfully completed tier testing, it is ready for operational training and testing in the simulator labs and on the aircraft.

7. Acronyms

The following list defines the acronyms used in this document.

Acronym	Definition
AOI	Area of Interest
AOR	Astronomical Observation Request
ASCII	American Standard Code for Information Interchange
CCD	Charge-Coupled Device
CDDS	Cavity Door Drive System
cid	Command Identifier
CR	Carriage Return
DAS	Data Acquisition System
DCS	Data Cycle System
DCT	Data Collection Time
Dec/dec	Declination
DVDS	Digital Video Distribution Subsystem
EcRF	Ecliptic Reference Frame
EL	Elevation (u) Axis
EPO	Education and Public Outreach
ERF	Equatorial Reference Frame
FCM	Focus Control Mechanism
FFI	Fine Field Imager
FFIRF	Fine Field Imager RF

Acronym	Definition
FITS	Flexible Image Transport System
FLITECAM	First-Light Infrared Test Experiment Camera
FMS	Flight Management System
FORCAST	Faint Object Infra-Red Camera for the SOFIA Telescope (SI)
FOV	Field of View
FPI	Focal Plane Imager
FPIRF	Focal Plane Imager Reference Frame
GalRF	Galactic Reference Frame
GREAT	German Receiver for Astronomy at Terrahertz Frequencies (SI)
GUI	Graphical User Interface
HIPO	High-speed Imaging Photometer for Occultations (SI)
HK	Housekeeping
ICD	Interface Control Document
IR	Infra-Red
IRF	Inertial RF
IRIG	Inter-Range Instrumentation Group
JPL	Jet Propulsion Lab
LF	Line Feed
LOS	Line of Sight (w Axis)
MADS	Mission Audio Data Subsystem
MCCS	Mission Communications and Control System
MD	Mission Director
MOPS	Mission Operations
NASA	National Aeronautics and Space Administration
NTP	Network Time Protocol
PI	Principal Investigator
PIS	Platform Interface Subsystem
RA/ra	Right Ascension
RF	Reference Frame
RH	Research Hours
ROF	Rotation of Field
SCL	SOFIA Command Language
SFH	Science Flight Hours
SFP	Science Flight Planner

SI	Science Instrument
SIBS	Science Instrument Boresight
SIRF	Science Instrument RF
SMA	Secondary Mirror Assembly
SMARF	Secondary Mirror Assembly RF
SMCU	Secondary Mirror Control Unit
SOFIA	Stratospheric Observatory for Infra-Red Astronomy
TA	Telescope Assembly
TAIPS	Telescope Assembly Image Processing Subsystem
TAMCP	TA Master Computer Processor
TARF	Telescope Assembly RF
TASCU	Telescope Assembly Servo Control Unit
TCM	Tilt/Chopping Mechanism
TO	Telescope Operator
TSD	Telescope Status Display
TTL	Transistor-Transistor Logic
UCLA	University of California, Los Angeles
UT	Universal Time (Greenwich Mean Time)
VPA	Vertical Position Angle
VPARS	Video Processing and Recording Subsystem
WFI	Wide Field Imager
WFIRF	Wide Field Imager RF
WVM	Water Vapor Monitor
XEL	Cross-Elevation (v) Axis
XML	Extensible Markup Language

8. Errata

The following are known issues with this current version of the user's guide. These will be addressed in future revisions.

Table 8-1: Errata items.

Date Discovered	Errata Description	Section Impacted
10/16/14	Once the MCCS-SI-04 units table is updated with milliarcseconds, Table 3-4 will have to updated accordingly.	3.4.5
10/17/14	Per DR SOF-1042, `coord.convert` does not properly convert Ecliptic coordinates with equinoxes different from J2000. This is has been documented in section 3.3.4, and the issue should be removed from the document when the issue is resolved.	3.3.4