Quick-Start Guide to Accessing SOFIA Engineering Data last updated 10/05/23 by D.Hoffman

1) Prerequisites
   a. linux-like computer system (e.g., Mac, Windows PC with Cygwin, or a linux-like system you have an account on)
   b. network access to SOFIA Engineering Data at IRSA
   c. Python3
   d. Java Development Kit 11+ (for running taipsExtractor)
   e. adequate free disk space on your computer:
      ~300 Mbytes is needed for installation of quick-start files as described in step 2 below. And space for the engineering data you expect to download – this can easily take many Gbytes depending on how much you download. These can be two separate spaces.

2) Install quick-start files on your computer
   a. connect to IRSA SOFIA housekeeping data server (https://irsa.ipac.caltech.edu/data/SOFIA/HOUSEKEEPING/tools/)
   b. download 'quick-start.tgz' to your computer – this includes scripts for processing engineering data and sample input files to check that the scripts work on your computer
   c. choose where to install the quick-start files on your computer and install in that location (the installation will need read-write-execute permissions and ~300 Mbytes)
```
cd <install path>
tar xvf <directory containing download>/quick-start.tgz
```
   d. set environment variables to provide access to the scripts (the commands below are for bash; update as appropriate for your preferred shell)
```
export DHT_PATH=<install path>
export PATH=$DHT_PATH/quick-start/bin:$PATH
export DHWG_LIB=$DHT_PATH/quick-start/lib
```
   e. add the 3 above export commands to your login setup file (e.g. ~/.bash_profile) – this is so that you don't have to re-run these commands every time you want to use the scripts
   f. do the steps in the "Quick-Start Tutorial for using Data Handling Tools Scripts" section below -- this will familiarize you with the Data Handling Tools scripts included in the quick-start files and ensure they work on your computer

   The above steps only need to be done once to setup your computer and familiarize you with the scripts included in the quick-start files. Subsequently, you can go straight to step 3 below and use the scripts to process SOFIA engineering data.

3) Retrieve desired data from IRSA
   a. use table below to determine location of desired data on IRSA server
   b. retrieve data from IRSA server (e.g. with wget or curl)
      – see "Retrieving engineering data from IRSA" below
   c. use DHT scripts to process and examine data,
      or alternatively, examine text data directly or process with your own tools

Note that the quick-start files only include the few scripts that most users would need to process and examine SOFIA engineering data. The tools/DHT directory on the IRSA server has the latest full release of all DHT scripts (DHT_2.1.0) and other relevant files. See tools/README for a description of the content of the tools directory and see tools/DHT_2.1.0/README for a quick description of the full release and of all the scripts it includes. (Most users will probably not need to use the additional scripts, but they are included in the event they are needed.)

Data Type	Location	Common Uses
Flight Summary	tools/doc/flight_summaries/	review PDFs by observing cycle with flight paths, altitude profiles, target information, and special notes for each observing leg; tables used to make the plots are in the time_data subdirectory
FPI, FFI, WFI images	fltdata/<mid>/ark/	create TA image FITS files using taipsExtractor
housekeeping (HK)	products/<mid>/hk/	examine desired HK with sample.py
command logs	products/<mid>/cmdlogs/	review commands and final responses
reports generated from HK	products/<mid>/reports/	summary of significant events such as door open, inertially stable TA, altitude changes
GUI logs	fltdata/<mid>/mission_data/	examine all commands from a GUI and all responses from MCCS
metrics data	products/<mid>/metrics/	examine observatory metrics data
products data tarballs	products/<mid>/tarballs/	obtain all products data of a particular type for a SOFIA operation; contains gzipped tarballs for most types and a text file showing content; tarballs vary in size up to a couple Gbytes
pre-2013 ark files	archive/ark/<date>/	examine HK data from before 2013 using xt.py to extract the data and sample.py to examine it
pre-2013 ancillary files	ancillary/<date>/	examine GUI logs and other assorted files from SOFIA operations before 2013

Retrieving engineering data from IRSA

When you don't already have needed engineering data files on your computer, you'll need to retrieve them from the IRSA server first.

1) determine the Mission ID (MID) of the SOFIA operation of interest
2) determine the location of the desired data from the table above
3) use wget or curl to retrieve the desired folders or files

   Note: Many engineering data directories contain large amounts of data – for example, the fltdata/<mid>/ark directory from a typical flight contains several hundred Gbytes, and the products/<mid>/hk directory contains 20 to 50 Gbytes. So before requesting transfer of large directories or large numbers of files, you should confirm that you have adequate space on your system and that your connection to the IRSA server is fast enough to retrieve it in reasonable time.

   The products/<mid>/tarballs directory contains compressed tarballs for various products data. It may be preferable to transfer the tarball(s) for the needed products data types rather than trying to select and retrieve many individual product files. These tarballs vary in size up to several

Gbytes – so if you only need a few individual HK files, it may be faster to just grab those, but if you want many HK files it may be faster to grab the corresponding tarball.

Downloading files with wget and curl

```
wget <URL of file>                              # download a single file
wget -i <file containing URLs of files>         # download multiple files
wget http://example.com/images/{1..50}.jpg      # download numbered files
```

By default, wget sends retrieved data to same file name as was found at the URL. It's probably best with SOFIA engineering data to retain original file names – since some programs, such as taipsExtractor assume that the file name follows SOFIA naming conventions.

```
curl <URL of file> -o <file name>                       # download a single file
curl "http://{one,two}.example.com" -o "file_#1.txt"    # download matching files
curl "http://{site,host}.host[1-5].com" -o "#1_#2"      # download matching files
```

By default, curl sends retrieved data to stdout, and the '-o' option reroutes it to the named file. The last two curl examples illustrate use of '#1' & '#2' to set the name of output file using the matching fields.

Steps for retrieving imager *.ark files

There is an added step for retrieving the archive (*.ark) files needed for getting FPI, FFI, or WFI images. There are many imager files for most flights, and their size generally precludes downloading all of them – therefore you will need to identify which ark files to retrieve. This can be done as follows:

1. identify the Mission ID (MID), which imager(s), and start/end time periods you want
2. navigate to fltdata/<mid>/ark
3. use date/times on imager ark file names to identify ark files that contain desired image data (Notes: FPI, FFI, WFI images are in *ta_fpi*, *ta_ffi*, and *ta_wfi* ark files respectively. ark file names include time that file was started – for example, 'archiver.ta_fpi.200909060432.ark' contains FPI images after 2020-09-09T06:04:32 UTC.)
4. use wget or curl to copy them to your computer

Quick-Start Tutorial for using Data Handling Tools Scripts

This section is to familiarize you with the provided DHT scripts and ensure they work on your computer.

The examples below assume that the quick-start files have been installed on your computer, and that PATH and DHWG_LIB environment variables have been set as described above.

1. Extract TA images from imager *.ark files

```
IN=$DHT_PATH/quick-start/sample_data  # adjust to where quick-start files was installed
OUT=$HOME/tmp/fits_out                # directory to write output files
mkdir -p $OUT

# Familiarize yourself with taipsExtractor capabilities with the help option (-h).
# See quick-start/doc/README.taipsExtractor for more info.
taipsExtractor -h
```

```
# Create fits files for FPI image data from 2020-09-09T06:04:40 to 10 seconds later.
# Note: if needed, you can check the flight MD flash report for leg start/end times.
taipsExtractor -a $IN -f $OUT -s 200909060440 -e 200909060450 --fpi --verbose

# Check out the created fits files (there should be 7 new ~2 Mbyte FITS files).
# Note that each file name includes the exposure start time for that image
# (e.g. FPI.20200909T060445.219Z.fits for exposure started 2020-09-09T06:04:45.218 UTC)
ls -l $OUT/
# Could use a fits viewer (e.g. ds9) to examine extracted images.
```

2. Convert between SOFIA timestamp formats

```
IN=$DHT_PATH/quick-start/sample_data  # adjust to where quick-start files was installed


# Many SOFIA engineering data files are chronological text files with a UTC
# timestamp in column 1, where the timestamp is either in human readable form
# such as '2020-09-09T06:04:40.261Z', or a Unix epoch time such as '1599631480.261'.

# Become familiar with timeconv.py options with help option (-h).
timeconv.py -h

# For example, given the following

head -4 $IN/hk/das.ic1080_2_2hz.lat_fms_1.out
#mcstime        das.ic1080_2_2hz.lat_fms_1 (FLOAT4, units=degrees)…
1599613430.057151       34.6069
1599613430.557074       34.6069
1599613431.057100       34.6069

# timeconv.py can convert the Unix UTC times in column 1 to human-readable UTC times.

head -4 $IN/hk/das.ic1080_2_2hz.lat_fms_1.out | timeconv.py
#mcstime        das.ic1080_2_2hz.lat_fms_1 (FLOAT4, units=degrees)…
2020-09-09T01:03:50.057151Z     34.6069
2020-09-09T01:03:50.557074Z     34.6069
2020-09-09T01:03:51.057100Z     34.6069

# Note the use of a Unix pipe above to send output from head into timeconv.py.
# timeconv.py always takes its input from stdin which make it easy to convert
# output from other programs.

# It can also convert human-readable UTC time in column 1 to Unix time.

head -4 $IN/cmdlog_20200908T223906Z.txt
2020-09-08T22:39:06.394Z archiver h ***** Integrated command log file created - …
2020-09-08T22:39:06.402Z archiver S SessionManager started on archiver
2020-09-08T22:39:07.383Z archiver.session_1 + 10.4.1.15:1025 connected to archiver:6555
2020-09-08T22:39:07.383Z archiver.session_1:2 c 1 login user=wvm role=wvm_system

head -4 $IN/cmdlog_20200908T223906Z.txt | timeconv.py -t0
1599604746.394 archiver h ***** Integrated command log file created - …
1599604746.402 archiver S SessionManager started on archiver
1599604747.383 archiver.session_1 + 10.4.1.15:1025 connected to archiver:6555
1599604747.383 archiver.session_1:2 c 1 login user=wvm role=wvm_system
```

3. Get HK values for a selected period

```
# Become familiar with sample.py options with help option (-h)
sample.py -h


# Get RA, Dec, and elevation above horizon during leg 8 (NGC6946) on flight F683

# find HK names if not known
#  --> check MCCS_SI_04 appendix B or TA_MCCS_F (for TA-specific HK)
#  --> coord.pos.actual.ra, coord.pos.actual.dec, coord.pos.actual.alt

# find time period of leg 8 if not known
#  --> check MD flash report for F683
#  --> 2020-09-09T07:14 to 2020-09-09T07:58

# acquire HK values for selected time period with sample.py

OUT=$HOME/tmp         # can be any directory where user has write permission
mkdir -p $OUT

sample.py $IN/hk -o $OUT/out1.tsv -s 2020-09-09T07:14 -e 2020-09-09T07:58 -d
  coord.pos.actual.ra$ -d coord.pos.actual.dec$ -d coord.pos.actual.alt$

# Note ending HK names with '$' avoids matching other HK that may start with
$ same name.  These are regular expressions – so '$' means match end of string.

# examine a few lines of tab-separated-value file created by above sample.py command
head $OUT/out1.tsv
```

4. View significant events and logs chronologically

```
# Become familiar with merge.py options with help option (-h)
merge.py -h

IN=$DHT_PATH/quick-start/sample_data  # adjust to where quick-start files was installed
OUT=$HOME/tmp         # can be any directory where user has write permission

# Get all ta_pos commands issued
grep ' c .* ta_pos' $IN/cmdlog*.txt > $OUT/ta_pos_cmds

# Merge with significant events report
merge.py -k2 -s 2020-09-09T05:30 $OUT/ta_pos_cmds $IN/generateKeyEventReport.txt | more
```

5. Extract HK from archive files

```
# Become familiar with xt.py options with help option (-h)
# (the most used options are -d, -o, -p, -v, and -s)
xt.py -h

IN=$DHT_PATH/quick-start/sample_data  # adjust to where quick-start files was installed
OUT=$HOME/tmp         # can be any directory where user has write permission


# Note that most HK is automatically extracted after each SOFIA operation
# and accessible in products/<mid>/hk/.  But some data such as pre-2013 ark files,
# DAS archive files (in raw/das/das_archive/), and CECS archive files (in
# raw/cecs/cecs_arcive) are not extracted during post-operation processing
# and may need to be downloaded so that desired data may be extracted using
# xt.py.

# Extract all HK from a DAS (Data Acquisition Subsystem) ark file.
# The command below creates a *.out file for every HK value in the ark file.
# Each *.out file is named after the HK value it contains.  It is a text file
# with Unix time in column 1 and HK value in column 2.

rm -rf $OUT/hk_das
xt.py -d. -o $OUT/hk_das $IN/das.das.200908223206.ark

# examine resulting data

ls -l $OUT/hk_das

head $OUT/hk_das/* | more
```

Additional Help

While IRSA does not maintain these tools, and makes no guarantee they will work on your system, you can request limited guidance and assistance by sending an email to: irsasupport (at sign) ipac.caltech.edu