ASE3 BOP Procedures

$WAIS/note/xped/ASE3/notebook/BOP/

Table of Contents

I. GENERAL NOTES....3
Procedure Notes....3
Best Practices....3
Python external module dependencies....3
External program dependencies....3
Supl Datasets....3
II. BOP SETUP....4
III. DOWNLOAD....5
Download RAD/serial....5
Download CAM....5
Download POS....5
IV. SPLIT PSTs....6
Breakout ELSA as a single continuous flight....6
Use pstedit.py to create or edit modify_pst.rc....6
V. BREAKOUT/QUALITY/PRINT....8
i. AQN (ELSA)....8
Check Output....8
ii. VHF Radar....8
iii. GPS....9
iv. CAM....10
v. Flight QC....11
VI. PRINT....11
AQN....11
VHF....11
GPS....11
Flight QC....12
VII. Quality Control (QC)....13
Magnetics....13
RADAR....15
VIII. SCAN....16
Flight Notes....16
QC Sheets....16
Flight plans....16
IX. ARCHIVE....17
One-time Computer Setup....17
Per Tape/Disk Setup....17
Per Flight Setup....18

Write Archive Data....18
Check the archive (tape-based archive):....19
Check the archive (filesystem-based archive)....20
Generate an archive listing....20
Typical usage (tape-based archive):....20
Typical usage (filesystem-based archive):....20
Extract Files from a Tape Archive....20
Extract Files from a Filesystem-based Archive....21
Useful tape commands:....21
X. CLEAN....22
APPENDIX A. glnk nested scripts....23
APPENDIX B: grad nested scripts....24
APPENDIX N: Other....24

I. GENERAL NOTES

Procedure Notes

- [] brackets are used to denote optional arguments (i.e. not required for the code to run). [1]
- {} are used for variables that are required and will depend on the specific season, flight etc.
- ASE3 has been put into the xped field within these notes, but a different season can be used.
- F## is the flight number (i.e. F02, F12)
- J### is for the Julian day (i.e. J055, J132).
  - Ground test's flight name is the Julian day of the start of the test.
  - Convention has flight name first character (F,J) is capitalized.
- Any qualitative notes about what you find in this process go in: $WAIS/note/xped/ASE3/notebook/BOPnotes

Best Practices

- Start by reading the flight notes. This will inform you of any unusual aspects of the data that may have affected the data collection process.
- Avoid overloading the QNAP by being mindful of the I/O and bandwidth requirements of your task.
- Mark your initials on the media envelopes when you finish downloading or deleting.
- Do not delete any original data unless it has been archived and verified.
- Archiving tapes (copies and full tapes) should be distributed to different locations to avoid single point failures.
- Always unmount disks before removing.
- Include successful examples of qc from a relevant past season on the QNAP.

Python external module dependencies

- gdal
- gpxpy
- matplotlib
- numpy
- scipy
- pyproj
- PyQt4
- seaborn

External program dependencies

- enscript
- ps2pdf
- gmt (in progress to remove)
- timeconvert
- teqc
- pdftk

Supl Datasets

The following datasets are required in $WAIS/targ/supl

1 GNG – I changed these to match the GNU documentation conventions. In previous versions, {} denoted optional, and <> denoted required.

II. BOP SETUP

With the Ethernet switch as a central node, connect the switch to the router, laptop, and QNAP (port 1). The LTO drive is connected via USB3 directly to the laptop. The LTO usually mounts to /dev/nst0.

Make a symlink so $WAIS will work with whichever QNAP is used. First, remove the old symlink.

```
$ sudo rm -f {path to media}/ICE
```

Second, make new symlink pointing to the QNAP being used using an absolute (not relative) path.

```
$ sudo ln -s {path to media}/{media or folder name} {path to media}/ICE
```

Example:

```
$ sudo ln -s /media/ICE09 /media/ICE
```

To check the symlink, verify that you can 'cd' into $WAIS.

Set environmental variables within the ~/.bashrc file. The $WAIS and $ICE alias are set to the same location. The symlink set above (/media/ICE) points to the QNAP. Philosophically, there needs to be only one of these aliases, but practically, both have been used in the past so $ICE alias is maintained, but it is actively being removed from ASE3 field code. Note that historically $QXPED was $QPROJ.

```
QXPED=ASE3
```

Set additional environmental variables in $WAIS/code/xped/ASE3/development/setup_env.sh

```
QYEAR=2019
QPLAT=PBA
```

Open a new terminal window and confirm the environmental variables are displayed correctly. If the environmental variables are not defined or are not defined correctly, verify the paths specified in the ~/.bashrc file and open a new terminal window.

Run the code that makes all the appropriate files in code/xped/ASE3/development.

```
$ cd $WAIS/code/xped/ASE3/development
$ ./make.sh
```

Make a season specific RAD params file in $WAIS/code/xped/ASE3/quality/xlob/RAD/params by copying a file from the most recent season that used the same system and rename the file to HERA-ASE3.conf

Update xped_settings.py

- mapproject (only needs to be edited if the season is not in Antarctica.)

III. DOWNLOAD

After data comes from the acquisition systems it needs to be downloaded to the QNAP.

Download RAD/serial

```
$ cd $WAIS/code/xped/ASE3/download
$ ./dljkb2 --radar HERA -i /media/qc-kpr/RAD##/j###  -f {F##} -n
```

The -n flag displays a list of files that will be copied. Verify that the paths are as expected and then run the command again, excluding the -n flag. Verify data is copied and in correct location ($WAIS/orig/xped/ASE3/acqn/HERA/F##) and then mark the media envelope with your initials.

Download CAM

```
$ cd $WAIS/code/xped/ASE3/download
$ ./dlcam -i /media/qc-kpr/CAM## -f {F##} -n
```

The -n flag displays a list of files that will be copied. Verify that the paths are as expected and then run the command again, excluding the -n flag. Verify data is copied and in correct location ($WAIS/orig/xped/ASE3/acqn/CAM/F##/DCIM) and then mark the media envelope with your initials.

Download POS

```
$ cd $WAIS/code/xped/ASE3/download
$ ./dl_pos --flight {F##} --instr TPC --rxname PBA13  -i {path to files}
```

Verify data is copied and in correct location ($WAIS/orig/xped/ASE3/acqn/TPC/F##/PBA13) and then mark the media envelope with your initials.

```
$ cd $WAIS/code/xped/ASE3/download
$ ./dl_pos --flight {F##} --instr TRM --rxname THW11  -i {path to files}
```

Verify data is copied and in correct location ($WAIS/orig/xped/ASE3/acqn/TRM/F##/THW11) and then mark the media envelope with your initials.

IV. SPLIT PSTs

When ELSA data is recorded as a single transect, breakout is a three step process:

1. Breakout ELSA data as a single continuous flight to extract basic time and position information.
2. Use the *pstedit.py* GUI to generate a *modify_pst.rc* file from the extracted information and define PSTs.
3. Re-run breakout (**V. BREAKOUT/QUALITY/PRINT**).

Breakout ELSA as a single continuous flight

Run $WAIS/code/xped/ASE3/all/glnk with the --full argument. Put 'breakout' at the end of the command. This will only run the breakout portion of the glnk code. This saves time, since we don't need to generate quality control plots at this point. See full glnk syntax in **i. AQN (ELSA)** on page 8.

```
$ cd $WAIS/code/xped/ASE3/all
$ glog ./glnk  --flight {F##} --full breakout
```

Use pstedit.py to create or edit modify_pst.rc

```
usage: pstedit.py [-h] --flight FLIGHT [--radar {HERA,MARFA,HiCARS2}]
                  [--rcfile RCFILE] [--mapproj MAPPROJ] [--mapscale {m,km}]
                  [--dt DT] [-v] [--waypoints] [--WAIS WAIS] [--xped XPED]

optional arguments:
  -h, --help            show this help message and exit
  --flight FLIGHT       Flight name
  --radar {HERA,MARFA,HiCARS2}
                        Radar data type
  --rcfile RCFILE       Read and write this modpst file instead of the default
  --mapproj MAPPROJ     Map projection to use (default: epsg:3031)
  --mapscale {m,km}     Map Scale (default=km)
  --dt DT               Interval, in seconds, to label flight time on map.
                        (default: no labels)
  -v, --verbose
  --waypoints           plots waypoint names on map. Requires a flight plan
                        CSV data file in the note/Flight_Plan directory.
  --WAIS WAIS           Root directory. Default: $WAIS
  --xped XPED           xped name. Default: $QXPED
```

The --radar flag is typically omitted, since radar has not yet been broken out. If updates to PSTs are needed after radar breakout the --radar flag can be included and radar profiles will be displayed. The --

waypoints and --dt flags can display additional waypoint and time information to assist in adjusting PSTs.

Here is a typical way to run pstedit.py to create and edit a *modify_pst.rc* file.

```
$ cd $WAIS/code/xped/ASE3/all
$ ./pstedit.py --flight {F##}
```

The gui will pop up with GPS elevation and map of track. Line 1 in the dialog box has the name of the flight and the first and last time in GPSnc1 ct counter time for the entire flight. Note that other streams may start slightly before or end slightly after. *modify_pst.rc* is saved to $WAIS/orig/xped/ASE3/acqn/ELSA/F##.

modify_pst.rc has a row for each PST with three columns:

```
ASE3/PBA0a/F##X##x 0s:500000s P/S/T
ASE3/PBA0a/F##X##x 500000s:10000000s P/S/T
```

1. xped/set/flight
2. counter start:counter end (in 10 us counts)
3. project/set/transect

Set the min and max defined counter in the *modify_pst.rc* file to zero and some arbitrarily high number. This will ensure all streams are included. Ensure that what you enter in Column 1 matches the flight name displayed in Line 1. Remember to include the ‘s’ after the counters.

Next, use *pstedit.py* to add PST counter breaks. Ensure that continuity is maintained between PSTs (e.g. pst1 ends at 1000 and pst2 starts at 1000). Use the flight notes and the header sheet to define individual PSTs.

PST naming conventions

- F##T##{a,b,etc} : Transits, flight time between ‘official’ survey lines
- F##G##{a,b,etc} : Times when the helicopter is on the ground, or nearly (initial take off / refueling / final landing)
- IByydoy{a,b,etc} : flights along operation ice bridge lines the signifier (yydoy) is the year and doy of the OIB flight path. This date was retrieved from the OIB flight lines files .kmls (e.g. Browse_YYYY_Antarctica_{OIB_platform}.kml).
- X lines are roughly flow parallel
- Y lines are roughly flow perpendicular
- X and Y line signifier are capitalized
- {a,b,c,etc} : signify multiple legs on the same PST line and count sequentially.
- Code requires that each component of the PST be 8 characters or less.

V. BREAKOUT/QUALITY/PRINT

Scripts in $WAIS/code/xped/ASE3/all breakout data AND generate files for printing.

i. AQN (ELSA)

```
usage: glnk [-h] --flight FLIGHT [--full] [-v] [--papersize {A4,letter}]
            [--WAIS WAIS] [--xped XPED] [--platform PLATFORM] [--nomodpst]
            [tasks [tasks ...]]

positional arguments:
  tasks                 ELSA tasks to perform. Select from breakout, quality,
                        print. Default: perform all

optional arguments:
  -h, --help            show this help message and exit
  --flight FLIGHT       Flight name to process (e.g., F01)
  --full                Breakout flight as full single transect
  -v, --verbose         Verbose script output
  --papersize {A4, letter}
                        Paper size
  --WAIS WAIS           WAIS root path variable. Default: $WAIS
  --xped XPED           Expedition name. Default: $QXPED
  --platform PLATFORM   Airborne platform name. Default: $QPLAT
  --nomodpst            Disable parsing of modify_pst.rc file in orig data to
                        modify pst, when performing breakout
```

Run glnk to breakout ELSA data and generate files for printing.

```
$ cd $WAIS/code/xped/ASE3/all
$ glog ./glnk --flight {F##}
```

Check Output

After you run glnk, check that all expected PSTs are there using ncdu or du (du: disk usage)

```
$ du -hcs $WAIS/targ/xped/ASE3/breakout/ELSA/F##/*/*/*
```

OR

```
$ ncdu $WAIS/targ/xped/ASE3/breakout/ELSA/F##
```

ii. VHF Radar

```
usage: grad [-h] [--WAIS WAIS] [--xped XPED] --flight FLIGHT -radar RADAR
            [-nmax NMAX] [--overwrite] [-v] [--full] [--platform PLATFORM]
            [--nomodpst]
```

```
            [tasks [tasks ...]]

positional arguments:
  tasks                 ELSA tasks to perform. Select from breakout, quality,
                        print. Default: perform all

optional arguments:
  -h, --help            show this help message and exit
  --WAIS WAIS           WAIS root path variable
  --xped XPED           xped name
  --flight FLIGHT       Flight name
  --radar RADAR         Radar type
  --nmax NMAX           Max number of radar files to process, then quit
  --overwrite           Overwrite output data
  -v, --verbose         Verbose script output
  --full                Process entire flight as a full, single transect
  --platform PLATFORM   platform (e.g. GCX,JKB,IBH)
  --nomodpst            Disable parsing of modify_pst.rc file in orig data to
                        modify pst, when performing breakout
```

Run grad to breakout radar data and generate files for printing.

```
$ cd $WAIS/code/xped/ASE3/all
$ glog ./grad --flight {F##} --radar HERA
```

iii. GPS

Run ggps to breakout GPS data and generate files for printing. This should be run after AQN/ELSA breakout.

For this script to run one of these sets of conditions must be met
1. There is GPSnc1 data and --jday is not set
2. --jday and --year are set (--year defaults to current year)
3. --year is set and the flight is a ground test with a J### (jday) name

GGPS options can be summarized:

```
usage: ggps.py [-h] --flight FLIGHT [--jday JDAY] [--year YEAR] [-v]
               [--WAIS WAIS] [--xped XPED] [--platform PLATFORM]

optional arguments:
  -h, --help            show this help message and exit
  --flight FLIGHT       Flight name
  --jday JDAY           day of year, of data collection, e.g., J365. Default:
                        auto-detect day of year from ELSA data
  --year YEAR           Year of data collection. Default: current year
  -v, --verbose         Verbose script output
  --WAIS WAIS           WAIS root path variable. Default: $WAIS
  --xped XPED           Expedition name. Default: $QXPED
  --platform PLATFORM   Airborne platform name. Default: $QPLAT
```

There are three typical ways to run ggps:

Example 1:
If ELSA GPSnc1 data is available for the flight, then ggps.py can use this to get the required GPS day, week, and year information, so use the simplest form:

```
$ cd $WAIS/code/xped/ASE3/all
$ ./glog ./ggps.py --flight F01
```

Example 2:
If ELSA GPSnc1 data isn't available, or the current year isn't the same as when data was collected, you may also need to set the –jday and/or –year options. This example manually specifies the 314th day of year in the year 2018.

```
$ cd $WAIS/code/xped/ASE3/all
$ ./glog ./ggps.py --flight F01 --jday J314 --year 2018
```

Example 3:
Note that sometimes flight names are the same as the Julian day, for example for ground test data. In this case, you don't need to specify the jday, but you may need to specify the year.

```
$ cd $WAIS/code/xped/ASE3/all
$ ./glog ./ggps --flight J314 --year 2018
```

iv. CAM

```
usage: gcam [-h] [--WAIS WAIS] [--xped XPED] --flight FLIGHT [-v] [--platform
PLATFORM] [--nmax NMAX] [tasks [tasks...]]

positional arguments:
  tasks                 CAM tasks to perform, select from breakout or quality.
                        Otherwise, perform all

optional arguments:
  -h, --help            show this help message and exit
  --WAIS WAIS           WAIS root path variable
  --xped XPED           Expedition name
  --flight FLIGHT       Flight name
  -v, --verbose         Verbose script output
  --platform PLATFORM   platform (e.g. GCX,JKB,IBH)
  --radar RADAR         Radar type
  --nmax NMAX           Max number of images to process before quitting
```

Run the gcam code for breakout and quality (make movie).

```
$ cd $WAIS/code/xped/ASE3/all
$ glog ./gcam --flight {F##}
```

v. Flight QC

```
usage: gqc.py [-h] [--WAIS WAIS] [--xped XPED] --flight FLIGHT
              [--papersize {letter,A4}] --radar {HiCARS2,MARFA,HERA}
              [--verbose]
              [tasks [tasks…]]

positional arguments:
  tasks                 Flight QC Tasks to Perform

optional arguments:
  -h, --help            show this help message and exit
  --WAIS                Root variable
  --xped                xped name
  --flight              flight name
  --papersize {A4, letter}
                        QC paper size
  --radar {HiCARS2,MARFA,HERA}
                        Radar name
  --verbose, --v
```

Run the gqc.py code to produce the Flight QC, Geology QC, Ice QC, and Review pdfs.

```
$ cd $WAIS/code/xped/ASE3/all
$ glog ./gqc.py --flight {F##} --radar HERA all
```

VI. PRINT

AQN

- The glnk script at the breakout step should have generated a folder $WAIS/targ/xped/ASE3/print/F##/F##_pcor

VHF

- The grad script at the breakout step should generate a folder $WAIS/targ/xped/ASE3/print/F##/F##_HERA

GPS

- The ggps script at the breakout step should have generated a folder on your desktop
- /$WAIS/targ/xped/ASE3/print/F##/F##_GPS

Flight QC

- The plot_flights.py script creates a folder $WAIS/targ/xped/ASE3/print/F##/F##_flightQC
- This folder contains the QC review sheet

VII. Quality Control (QC)

On the flight based QC sheet, highlight quality of each sub-section of transects using the following color code:
1 = blue; All's Well
2 = yellow; EDS or FOP issue
3 = orange; GAN issue
pink = missing data or sections where the instrument could not be expected to perform normally

Note any error messages, synchronization problems, or times missing in each dataset

FLIGHT NOTES
Review the flight notes to check any PST renaming, FOP passes to BOP, and events that could explain data singularities

POS/CG (eGPSpp1 from TPC and GPSnp[1,4] from IGM)
Each receiver has at least 7 satellites, with few I's (indicates loss of lock). See quality/flight/F##_GPS Ideally we want observations per cycle slip (o/slps) to be >1000. On the Flight QC page. The black histograms in the GPS streams represent cycle slips and dips in the red lines represent loss of lock. $WAIS/targ/xped/ASE3/print/F##/. GPS quality documents can also be found in the quality directory.

The teqc guide: $WAIS/note/xped/ASE3/notbook/teqc_symbols.txt

Blue: At least one POS receiver for full section of transect (Note if both do not meet this test)
Yellow: Cycle slips correlated with strong mag event, or aircraft maneuvers
Orange: No antennas recorded or loss of L1, L2 and/or C1/P1

POS/BASE (BASE GPS; GPStp3):
Blue: At least one base receiver for full section of transect
Yellow: Cycle slips correlated with strong mag event
Orange: No antennas recorded, or loss of L1, L2 and/or C1/P1

Magnetics

SCALAR MAGNETIC FIELD (MAG/FIELD): qc of MAG-823A cesium magnetometer

Blue: Flight based magnetic anomaly (magnitude of the range of the magnetic field strength) is less than ± 1000 nT; correlation with geology; transect based magnetic field strength is between 55,000 and 65,000 nT
Yellow: Flight based magnetic anomaly (magnitude of the range of the magnetic field strength) is

greater than ± 1000 nT, transect based magnetic field strength has aircraft noise (ripples, steps)
Orange: transect based magnetic field strength is below 55,000 nT; abrupt tears present; signal voltages are low in breakout data

GEOMAGNETIC FIELD (MAG/GEO): qc of base magnetics and geomagnetic activity
Blue: Rate of change is less than 3nT/min
Yellow: Rate of change is greater than 3nT/min
Orange: No functioning base mag was recorded

ELSA TIME SYNCHRONIZATION (ACQN/TCG): qc of PXI-6683 gps card

Exam the pcor plots for ct stats.

Blue: No seq-skipped (dropped) packets in GPSnc1 sample stats, regular timing
Yellow: dropped packets correlated with times of magnetic events
Orange: dropped packets or irregular timing on both systems

ELSA DATA ACQUISITION (ACQN/REC): qc of PXI data acquisition
Blue: No skipped sequence order (dropped packet) in sample stats, regular timing on all streams, no tears in radar data
Orange: dropped packets, irregular timing or tears

DOWNWARD LOOKING CAMERA (ALT/CAM): qc of Canon camera (mplayer $ICE/targ/xped/$QPROJ/quality/xlob/CAM/{F##}/CAM_F{##}.mp4)
Blue: Movie in focus, window/lens is clean, record is complete
Yellow: Movie out of focus, window/lens is dirty, and/or record is incomplete; excessive compression artifacts in original images
Orange: Shutter failure, evidence of optical damage, no GPS time stamp

SPAN attitude (ALT/SPAN): qc of NOVATEL SPAN/iMAR FSAS IMU and flight dynamics
Blue: Pitch is realistic (± a few degrees), centered on zero, and roll is always <10˚
Yellow: roll exceeds 10˚, and/or biased from zero
Orange: roll and pitch decorrelated from flight plan, rapid changes, does not correlate with IMU

LASER RANGE (ALT/LAS): qc of Riegl LD90 laser

Exam pcor and flight based QC sheets
Blue: surface detected and correlates with VHF radar
Yellow: surface dropouts due to height or cloud (compare to inflight checklists, flight notes and/or CAM)
Orange: loss of surface signal

RADAR

QC HF by comparing HF and VHF on gqc sheet:
SURFACE (Low gain, channel 1):
Blue: Surface traceable without interruption across entire section
Yellow: One or more locations where the surface is present but poorly defined or indistinct
Orange: One or more location where the surface is not present in the radargram
Pink: No data expected

Note: Traces should not saturate at the surface
LAYERS (Low/High gain):
Blue: Layers visible to a depth of ~8 μs above the bed over >80% of the section
Yellow: Layers visible to a depth of ~8 μs above the bed over 40-80% of the section
Orange: Layers visible to a depth of ~8 μs above the bed over <40% of the section
Pink: No data expected

BEDROCK (High gain, channel 2):
Blue: Bed is clear and traceable across full section of transect
Yellow: Bed is faint or partially obscured by noise at one or more points along the section
Orange: Bed is not present or completely obscured by noise at one or more points along the section
Pink: No data expected

VIII. SCAN

Flight Notes

Flight Notes may include:
1. Flight header sheet
2. Flight plan (map, profile, waypoints)
3. Flight note page(s)
4. in-flight check list
5. start up procedures
6. shut down procedures
7. other

Scan the flight notes into a pdf format and copy them to:
$WAIS/note/xped/ASE3/acqn/Flight_Logs/F##

Flight notes hard copy is stored in the Flight Notes Binder

QC Sheets

QC sheets may include
1. QC review page
2. BOP checklist

Scan the flight based QC sheet into a pdf format and copy them to:
$WAIS/note/xped/ASE3/acqn/Flight_Logs/F##

QC hard copy is stored in the Flight Notes Binder

Flight plans

Digital flight plans may include
1. src file
2. planning packet pdf
3. kml
4. csv
5. gpx
6. pumpkin drop data file

Digital copies of the flight plan and all associated files, including .src file, are copied into $WAIS/note/acqn/Flight_Plans/F##

IX. ARCHIVE

Archiving is the process of generating file that contains a copy that consists of a collection of data acquired on a flight. This copy is created on a LTO tape (called tape-based archiving) or on a disk (filesystem-based archiving).

One-time Computer Setup

Check whether your user is a member of the tape group:

```
$ groups
users adm tape sudo www-data
```

If the group “tape” is not listed, then allow permission to use the tape drive by adding your user to the tape group:

```
$ sudo usermod -a -G tape {your_username}
```

You may need to logout and login for membership to take effect.

Although **not preferred**, you may alternatively set world writable permission to write to the LTO drive. This may need to be done each time you insert the tape drive.

```
$ sudo chmod a+wr /dev/nst0
```

In ASE3 we are making two copies of orig and one copy of targ.

Orig: ELSA, RAD, TRM, TPC, CAM, SOB, notes
targ: ELSA, RAD, TRM, TPC, CAM, notes

Per Tape/Disk Setup

For each archive tape or archive disk, A new tag needs to be added or overwritten, using add_tag.py.

To use, cd into code/xped/ASE3/archive, and run

```
usage: add_tag.py [-h] --tag TAG --copy COPY [-y] [-v] [-o OUTPUT]
                  [--WAIS WAIS] [--xped XPED]

Adds tag to archive tape or directory

optional arguments:
  -h, --help            show this help message and exit
  --tag TAG
  --copy COPY
  -y, --yes             Answer yes to overwrite prompts
```

```
  -v, --verbose         Verbose script output.
  -o OUTPUT, --output OUTPUT
                        Output device or directory (default /dev/nst0)
  --WAIS WAIS           Root directory. Default: $WAIS
  --xped XPED           xped name. Default: $QXPED
```

Typical Usage:

```
$ glog ./add_tag.py --tag orig --copy {#}
```

The –tag option is typically orig or targ, and –copy is copy number of the tape (1 or 2).

Per Flight Setup

Archiving is a two-step process. find_data.py generates an Index (or bill of materials) of files to be included in the archive. Then arc_syst.py performs the commands to manage storage media (the tape or disk) and write the files in that bill of materials to the storage media.

run **find_data.py** to create a list of the flight specific paths of data to be archived. This code will generate a list of directories to be included in the archive. This list is saved to $WAIS/note/xped/ASE3/archive/{tag}/Index/F##.{orig,targ}

```
usage: find_data.py [-h] --flight FLIGHT --tag {orig,targ, sunday}
                    [--jday JDAY]
                    [-o OUTPUT] [--overwrite] [--WAIS WAIS] [--xped XPED] [-v]

optional arguments:
  -h, --help            show this help message and exit
  --flight FLIGHT       Flight to be archived (e.g., F01)
  --tag {orig,targ}     Archive tag name
  --jday JDAY           First jday to archive from EXTERNAL data (typically
                        jday of previous flight)
  -o OUTPUT, --output OUTPUT
                        Write index to specified output file
  --overwrite           Overwrite existing index file
  --WAIS WAIS           Root directory (default: $WAIS)
  --xped XPED           xped name (default: $QXPED)
  -v, --verbose         Verbose output
```

Example:

```
$ glog ./find_data.py --flight F01 --tag orig
```

Write Archive Data

Run **arc_syst.py** to issue the commands to create an archive of data.

```
usage: arc_syst.py [-h] --flight FLIGHT --tag TAG --copy COPY
                   [--dereference-links] [-n] [-v] [-q] [-i INPUT] [-o OUTPUT]
                   [--WAIS WAIS] [--xped XPED]

Archive data from a list created with find_data.py

optional arguments:
  -h, --help            show this help message and exit
  --flight FLIGHT       Flight to archive (e.g., F01)
  --tag TAG             Tag to archive
  --copy COPY           Copy number of the tape (e.g., 1,2...)
  --dereference-links   Dereference symbolic links when creating archives.
  -n, --dryrun          Dry run: Perform checks but do not actually write
                        archive.
  -v, --verbose         Verbose script output.
  -q, --quiet           quiet script output.
  -i INPUT, --input INPUT
                        Input index file to use (optional)
  -o OUTPUT, --output OUTPUT
                        Output directory for disk-based archiving. Default:
                        Use tape
  --WAIS WAIS           Root directory. Default: $WAIS
  --xped XPED           xped name. Default: $QXPED
```

Example:

To archive data to a tape-based archive, use arc_syst with no extra flags as shown. Here, we create the archive for flight F01, with the tag orig, copy 1.

```
$ ./glog ./arc_syst.py --flight F01 --tag orig  --copy 1
```

To archive data to a filesystem-based archive, use arc_syst with the -o option, pointing to the root of the hard drive. Here, the hard drive is mounted at /media/qc-kpr/HDTAPE01

```
$ ./glog ./arc_syst.py --flight F)1 --tag orig  --copy 1
         -o /media/qc-kpr/HDTAPE01
```

Check the archive (tape-based archive):

Using terminal commands check the status of the tape drive. This will tell you its current position

```
$ mt -f /dev/nst0 status
```

The output will have a line that says file number. File number 0 (zero) is the tag added by add_tag.py. File number 1 will be the first archived flight, file number 2 will be the second archived flight, etc. Use the step forward command to advance the tape to the desired position. The final numeric argument should represent the number of step you wish to move forward. If the numeric argument is omitted the

default value is 1. If the last file is reached, use `mt -f /dev/nst0 rewind` (tape does not necessary physically rewind)to reset file number to zero. Then use `fsf` to advance position.

```
$ mt -f /dev/nst0 fsf #
```

Once the status output file number matches the desired position for the archive to be verified, read the list of files that are archived.

```
$ tar -tvf /dev/nst0
```

This command will output a list of files and their size as well as any errors. If errors are found, re-archiving will be necessary. Check the file list and the file size for any discrepancies with expectations. If anything unusual is discovered, re-archive.

Check the archive (filesystem-based archive)

For filesystem-based archives, cd to the tape archive directory, and check the archive using the tar command.

```
$ tar -tvzf /mnt/sdb1/00#.tar.gz
```

Compared to the tape-based archive, we give the -z flag because the tar archive was gzip compressed.

Generate an archive listing

You can use the script read_tape_contents.py to generate a listing of all files in all archives on a tape to files. This has the effect of verifying a tape without

Typical usage (tape-based archive):

```
$ ./read_tape_contents.py -o /tmp/contents_orig1/
```

This reads a tape and outputs a file listing to /tmp/contents_orig1.

Typical usage (filesystem-based archive):

```
$ ./read_tape_contents.py -d /mnt/sdb1 -o /tmp/contents_orig1/
```

This reads the filesystem-based archive in /mnt/sdb1 and outputs a file listing to /tmp/contents_orig1.

Extract Files from a Tape Archive

To extract files from a tape, you must position the tape head at the correct file position, change directories to the root WAIS directory, and begin extracting the file. In the example below, we seek to the 2nd file on the tape (the first file after the tag).

```
$ mt -f /dev/nst0 asf 2
$ cd $WAIS
$ tar -xvf /dev/nst0
```

Extract Files from a Filesystem-based Archive

To extract files from a filesystem-based archive, change directories to the root WAIS directory, and begin extracting the file. In the example below, we seek to the 2nd file on the archive (the first file after the tag).

```
$ cd $WAIS
$ tar -zxvf /mnt/sdb1/00#_F##.tar.gz
```

Useful tape commands:

see tape status: size, number of files, etc	mt -f /dev/nst0 status
eject tape (physically rewinds):	mt -f /dev/nst0 offline
List files from current position:	tar -tvf /dev/nst0
read tape from current position	tar -xvf /dev/nst0 -C /target/directory (omit -C then ./)
move 'tape position' to beginning	mt -f /dev/nst0 rewind
move forward files	mt -f /dev/nst0 fsf # (# is number pf files forward, if number is omitted, defaults to 1)

X. CLEAN

We keep, at a minimum, data from the two last flights on media.

Check that all the data on the media is present in the correct location on the QNAP.

The linux diff command can be helpful.

Once the data existence is verified,

use the rm command within the command line to remove the media files from the media. DO NOT drag into the trash from the windows. This does not remove the files from the media.

APPENDIX A. glnk nested scripts

Code called from $WAIS/code/xped/ASE3

- ./all/glnk
  - ./breakout/ELSA/bo_elsa
    - $WAIS/syst/linux/src/breakout/ELSA/breakout.py
      - $WAIS/syst/linux/src/breakout/ELSA/rewrite.py
      - $WAIS/syst/linux/src/breakout/ELSA/deelsa.py
      - $WAIS/syst/linux/src/breakout/ELSA/dbprep.py
    - ./breakout/ELSA/streams/AQNnr3/bo_bxds
    - ./breakout/ELSA/streams/AVNap1/bo_bxds
    - ./breakout/ELSA/streams/AVNnp1/bo_bxds
    - ./breakout/ELSA/streams/GPSnc1/bo_bxds
    - ./breakout/ELSA/streams/LASsz1/bo_bxds
  - ./quality/pcor/ELSA/qc_elsa.py
    - ./quality/pcor/ELSA/gen_map_tracks.py
    - ./quality/pcor/ELSA/ct_stats/check_elsa.py
      - ./quality/pcor/ELSA/ct_stats/utils/sequence_check.c
      - ./quality/pcor/ELSA/ct_stats/time_stats.py
      - ./quality/pcor/ELSA/ct_stats/radar_stats.py
        - gmt
    - ./quality/pcor/ELSA/LASsz1/run_quality.py
    - ./quality/pcor/ELSA/plot_streams.py
    - ./quality/pcor/ELSA/plot_map.py
    - ./quality/pcor/ELSA/plot_stream_rates.py
  - ./print/ELSA/print_elsa.py

APPENDIX B: grad nested scripts

Code called from $WAIS/code/xped/ASE3

- ./all/grad
  - ./breakout/RAD/bo_rad
  - ./quality/xlob/RAD/qc_rad
    - ./quality/xlob/RAD/run_rp
    - ./quality/xlob/RAD/checkstats.sh
    - ./quality/xlob/RAD/calc_chirp_stats.py
    - ./quality/xlob/RAD/plot_chirp_stats.py
    - ./quality/xlob/RAD/plot_radar.py
  - ./print/print_rad.py
    - ./development/cmn/elsa_psts.py

APPENDIX N: Other

wireless BOP connection:

- check router for IP address
- log into router through internet browser and check settings for wireless name/ connect ability/ and password
- ssh into qc@192.168.#.### (look for label on router)