IRIS/PASSCAL INSTRUMENT CENTER

GENERATING SEED
FROM RT125 DATA
USING ANTELOPE
FOR
STANDALONE STATIONS

PASSCAL Platforms: Mac OS10 & Linux
Version 2011.021

This document was written and produced by the
PASSCAL Data Group

Please send your feedback, questions, and comments to:
data_group@passcal.nmt.edu

APPENDICES

APPENDIX _DATA		APPENDIX_SOFTWARE	
Name	**Short description**	**Name**	**Short description**
Data Delivery Policy	Requirements on Data Delivery for PASSCAL	**PASSCAL & CONTRIBUTED SOFTWARE GUIDE**	Links to main software and brief detail on most used tools for data processing/archiving
SEED FORMAT	Brief description of seed format for archiving of passive source data	**FIXHDR HELP**	Detail instructions on using fixhdr
Batch file	Details on how to generate batch file in antelope	**Gui_DoFTP**	Detail Instructions on using gui_DoFTP to send data to PASSCAL
Q330 State of Health Channels (SOH)	Description of SOH in a Q330	**Antelope**	How to update and install Current Version Notes
RT130LOG & LOGPEEK	Rt130 log file and how to see it with logpeek		
Troubleshooting	Common problems when processing data for archiving	**Troubleshooting**	Common problems software related

GENERATING SEED FROM RT-125 DATA USING ANTELOPE

(2011-021)

Introduction

This detailed document serves to guide the data archiver through the process of data archiving, utilizing Linux or Mac OSX operating systems. This guide assumes the user has basic Linux/UNIX skills, which are essential for completing the archiving task. We begin this process with the data on a field or local computer and end with the submission of these data to the IRIS/PASSCAL Instrument Center (PIC). The submitted data undergo fundamental quality assurance checks prior to PIC submitting the data for archiving at the IRIS Data Management Center (DMC). Individuals utilizing the Solaris operating system will find this guide helpful though specific details, such as software installation, for example, may differ. The archiving of your data fulfills the principle investigator responsibilities defined in the PASSCAL Data Delivery Policy (Appendix A)

You will use tools developed by PASSCAL and Boulder Real Time Technologies (BRTT: Antelope) to create a valid dataless SEED volume (dataless) and mini-SEED (mseed) station-channel-day files for archiving purposes. Examples of command line usage, short scripts, and definitions of Antelope parameter files (pf) to generate a dataless and manipulate mseed may be found throughout this guide and its appendices. (same as for rt130 documentation)

Please take a moment to thoroughly review this guide before you start.

If you have any questions please contact: data_group@passcal.nmt.edu.

The steps described below for data processing and archiving (PASSCAL tools and ANTELOPE) work on most platforms. Please refer to Appendix B for specifics on platforms and/or limitations for Antelope version 4.10.

Notice that:

General scripts and commands are in **bold**.
Command-line usage is highlighted yellow
GUI options or menus are highlighted turquoise
Standard output is *italicized.*
URLs and email addresses are blue.
Important notes are **brown**.

List of Materials and Initial Steps

Prior to starting this submittal process you should contact the data group@passcal.nmt.edu and acquire, complete, and/or review:

1) Principal Investigator web page (new)
   http://www.passcal.nmt.edu/pihomepage
   Create an account and follow your experiment.
2) Mobilization and Network code request form, preferred done through PI Home Page:
3) Demobilization form, done through PI Home Page: To be completed by the end of the experiment)

4) PASSCAL Data Delivery Policy is found in Appendix_Data, or online at: http://www.passcal.nmt.edu/content/general-information/policy/data-delivery-policy

5) ANTELOPE – Notes on the current release are in Appendix_Software as well as the guide for updating and installing new versions.

6) Install the latest PASSCAL software package for your platform, which may be found at: http://www.passcal.nmt.edu/content/software-resources

7) Submit a bug report or problem to:
   http://www.passcal.nmt.edu/node/add/externalticket

PASSCAL field computers loaned to the principal investigator (PI) are shipped with PASSCAL software and the most current version of Antelope pre-installed. However, you may need to update the version of Antelope on the computer if it has been in field for more that one year. Additionally, BRTT releases patches throughout the year and it is recommended that you patch your version of Antelope by running **antelope_update** from the command line. For more information about Antelope visit http://www.brtt.com/ and/or read the man pages.

New versions of Antelope are usually released spring or summer each year. You should check for the most recent version at http://www.brtt.com/. If you have an old version of Antelope please fill out the proper form under http://www.iris.edu/manuals/antelope_irismember.htm**.** If your institution is not an IRIS member and you will process data from a PASSCAL experiment, please contact data_group@passcal.nmt.edu and we will process your license request.

NOTE: the PASSCAL Instrument Center will provide Antelope data processing for all PASSCAL experiments, as required by an agreement between IRIS and BRTT. Please direct your Antelope questions to: data_group@passcal.nmt.edu.

Directions

Steps in brief

Data Reduction and Timing Quality Control

1. Create an organized directory structure for your data
2. Back up the raw data from the RT125 datalogger
3. Data conversion from *TRD files to mseed
4. Evaluate time quality and time corrections using clockcorr
5. Modify headers using fixhdr
6. Change Endianess and Flag/Shift Timing
7. Modify default fields on your traces to MSEED format

Populate the Antelope Database

1. Create a Batch File
2. Build the Antelope Database
3. View your database
4. Creating mseed day volumes and adding them to your database
5. Assign calibration values from calibration table to wfdisc
6. Verify the Integrity of Your Database
7. Create the dataless SEED volume

Send Data to IRIS/PASSCAL

We recommend use of GUI_DoFTP to submit data to the PIC (current version 2008.038 or later version).

Steps in detail

Data Reduction and Timing Quality Control

a. Back up the raw data from the RT130 compact flash cards

We encourage PASSCAL and USArray/FlexArray users to backup the raw images, and simultaneously generate zip files of the raw images on a disk.

If you cannot find the programs listed above or have any trouble please let us know by writing to passcal@passcal.nmt.edu.

b. Create an organized directory structure for your data

You can generate your own directory structure and name it as you see fit. The following is PASSCAL's suggested structure. Let's call "EXPT" the directory where you have all the data. Create the following directories under EXPT and organize the files accordingly:

<my_cpu:EXPT > mkdir raw_data (TRD files)

<my_cpu:EXPT > mkdir ref_mseed– location for mseed files obtained after running trd2miniseed

<my_cpu:EXPT > mkdir day_volumes – where the day volumes will go after running miniseed2days

Note that a "response" directory will be automatically created by **dbbuild.**

IMPORTANT – Place mseed day volumes of waveforms and log files (after running miniseed2days and log2mseed) in a directory called, for example, "day_volumes". The name is optional, call it what you like, but be consistent throughout the process.

c. Convert Your TRD Files Into Mseed By Using trd2miniseed

You can view the help for this tool by typing **trd2mseed -h** in the command line. You should run something like

```
<my_cpu:EXPT >  trd2mseed -v -F file_list -2  -o mseed
```

```
where : -F read the list of trd files
          -2 Use Steim Compression 2
              -o create a directory called mseed
```

This should generate *.m files (miniseed files) in the format:

```
2008.069.04.00.00.11549.1.m    2008.069.18.00.00.11549.1.m
2008.069.05.00.00.11549.1.m    2008.069.19.00.00.11549.1.m
2008.069.06.00.00.11549.1.m    2008.069.20.00.00.11549.1.m
2008.069.07.00.00.11549.1.m    2008.069.21.00.00.11549.1.m
```

and a file called *passcal.pcf* that looks like:

```
#!PCF trd2mseed 2008.040
11549   2008:065:01:25:50:000   0.000000e+00   4.357416e-08   #Start
11549   2008:070:02:45:50:000   0.000000e+00   0.000000e+00   #End
```

d. Make time corrections

For this purpose you can use clockcor (please see help by typing in the command line **clockcor -h**). To apply the corrections you can do:

```
<my_cpu:EXPT clockcor -m -d TC passcal.pcf *.m
```

NOTE: If you want you can set flags on your DATALESS reporting the timing issues on your metadata.

e. Modify headers on traces

The PASSCAL software, **fixhdr,** ("fix-header") allows users to make changes to mseed fixed header values, change the endianess of the mseed headers, and apply bulk-timing shifts. It also has a batch mode (-b option) that can be run with template files created either by **fixhdr** or from scratch. Typing **fixhdr** on the command line launches the program.

Header fields will need to be modified following SEED format. Fields that you will be able to modify using fixhdr are: station name, channel, location code (optional, only if needed), and network code. From this step forward the steps are very much the same as for rt130 data processing.

➤ ***Change Endianess and Flag/Shift Timing***

fixhdr also provides a means for you to modify the endianess (byte-order) of your files from little to big (if required) and setting flags for questionable timing, or to apply time correction when needed. To launch **fixhdr** with a GUI (graphical user interface) you need to type on the command line:

```
<my_cpu:EXPT> fixhdr
```

Help is available within the program and may be viewed by choosing the help button in the GUI or by running **fixhdr** with the "–h" option. More information on **fixhdr** can be found on Appendix_Software

➤ *Modify default fields on your traces to MSEED format*

Header fields will need to be modified following SEED format (Appendix_Data). Fields to modify using **fixhdr** are: station name, channel, location code (optional, only if needed), and network code. Appendix_Data**,** shows several examples on proper channel naming. Please refer to the Standard for the Exchange of Earthquake Data, Reference Manual, SEED Format Version 2.4 (http://www.iris.edu/manuals/) for complete details on SEED format.
To build your batch file using **fixhdr** please refer to the help for **fixhdr.** Below is an example of a batch file built by saving the template with **fixhdr:**

```
# Header Changes
hdrlist{
# sta:chan:loc:net:sps          new sta:chan:loc:net
0965D:1C1::XX:100.0             PA01:EHZ::YW
0965D:1C2::XX:100.0             PA01:EHN::YW
0965D:1C3::XX:100.0             PA01:EHE::YW
09511:1C1::XX:100.0             PA02:EHZ::YW
09511:1C2::XX:100.0             PA02:EHN::YW
09511:1C3::XX:100.0             PA02:EHE::YW
0969D:1C1::XX:100.0             PA03:EHZ::YW
```

Note that in this example we are not using the location code. If you decide to use location code (let’s say you use 00) it should look something like:

```
# Header Changes
hdrlist{
# stat:chan:loc:net:sps         new sta:chan:loc:net
            0965D:1C1:00:XX:100.0    PA01:EHZ:00:YW
            }
```

Populate the Antelope Database

a. Create a Batch File

The next step is to create an Antelope database that defines your network and station configurations.

You will use the tool **dbbuild** in batch mode (**dbbuild –b**) to construct a CSS3.0 database. The batch file is an ascii file with specific keywords and details used to build the database with out the use of the GUI. It is an effective way to keep a history of your experiment and also allows you to reproduce most of your database from scratch, if necessary. Use the following template as an example and edit accordingly the fields in green. A batch file may have comments (denoted with #). The description for each field in the batch file (and how **dbbuild** works) may be found in the man pages dbbuild_batch and **dbbuild**.

```
#comment: This is a dbbuild batch file.
net CAFE active - Oregon, WA

# Three (3) Component station with TEXAN datalogger (rt125) and L28 geophone


sta K06L 47.95865 -122.9305 0.607826 K06L active
time  3/05/2008   0:00:00.000
datalogger rt125 2258 K06L # Reftek rt125 Datalogger
sensor l28 0.0 001  # Mark Products L28
axis Z 0 0 - 1 1
samplerate 100sps
channel Z HHZ
datalogger rt125 2257 K06L # Reftek 130 Datalogger
sensor l28 0.0 001  # Mark Products L28
axis N 0 90 - 2 1
samplerate 100sps
channel N HHN
datalogger rt125 2257 K06L # Reftek 130 Datalogger
sensor l28 0.0 001  # Mark Products L28
axis E 90 90 - 3 1
samplerate 100sps
channel E HHE
add

# One(1) Component station with TEXAN datalogger (rt125) and L28 geophone, notice changes in
dataloggers serial numbers after each deployment.

sta D07 47.95235 -122.92911 0.568 Blyn, WA USA
time 03/09/2008 21:06:00
datalogger rt125 2936
sensor l28 0 1
axis Z 0 180 - 1 32
samplerate 100sps
channel Z EHZ
add

sta D07 47.95235 -122.92911 0.568 Blyn, WA USA
time 05/09/2008 20:58:00
datalogger rt125 2434
sensor l28 0 1
axis Z 0 180 - 1 32
samplerate 100sps
channel Z EHZ
add

close K06L 12/31/2007 23:59:59
close D07 12/31/2007 23:59:59
```

Another example:

```
net ZI insert-project-name-here

sta 2180 43.82048 -107.35429 1.482 Big Horn Mountains
time 7/29/10 17:53:00
datalogger rt125 3999
sensor GS11V 0 71
axis Z 0 180 - 1 32
samplerate 100sps
channel Z ELZ
channel N off
channel E off
add

sta 2180 43.82048 -107.35429 1.482 Big Horn Mountains
time 8/2/10 17:10:00
datalogger rt125 131
sensor GS11V 0 71
axis Z 0 180 - 1 32
samplerate 100sps
channel Z ELZ
channel N off
channel E off
add

sta 2182 43.8296 -107.35974 1.479 Big Horn Mountains
time 7/29/10 17:50:00
datalogger rt125 2728
sensor GS11V 0 71
axis Z 0 180 - 1 32
samplerate 100sps
channel Z ELZ
channel N off
channel E off
add

sta 2182 43.8296 -107.35974 1.479 Big Horn Mountains
time 8/2/10 17:07:00
datalogger rt125 102
sensor GS11V 0 71
axis Z 0 180 - 1 32
samplerate 100sps
channel Z ELZ
channel N off
channel E off
add

close 2180 7/22/2010 23:59:59
close 2182 7/22/2010 23:59:59
```

If you have questions about dbbuild or the batch file, please refer to Appendixes_Data. The fields in green are inputs that you need to provide.

NOTE: in the example below we don't include location codes. If you prefer to use location codes you should have something like:

```
samplerate 200sps
channel Z EPZ  01
channel N EPN  01
channel E EPE  01
```

(Where 01 is the location code for data stream 1.)

We discourage the use of locations codes and suggest they be explicitly defined only when necessary to avoid ambiguity (such as when operating a dense network (stations within 1 km) or when recording multiple streams at sample rates sharing a common band code (first letter) within the channel code.
Your batch file is the history of all the changes, editions, removals, etc. done to the stations on your network, so it MUST include all of them covering the times frames from the very first sample rate on any channel, to the day the station is closed.
We suggest a slightly earlier time for start time (second line after station in the batch file) to assure all the traces are included on the metadata. This will prevent from further errors and problems during archiving. Please read this manual carefully and refer to the appendixes for detailed information on several steps in this guide.

b. Build the Antelope Database

Now that you have a batch file you can run **dbbuild** to create your Antelope database.

```
<my_cpu:EXPT> dbbuild -b your-database your-batch-file >& my_dbbuild.out
```

NOTE: The configuration for each station in your batch file agrees with the mseed headers.

The batch file filename should not end with a ".pf" suffix.
Before running dbbuild please make sure that your batch file is absolutely correct by checking station names, location codes (if you have used any), sensor orientation, start times, close statement, etc.
It is best to use a start time for each the station which is conservative (i.e. a little early rather than milli-seconds late).

Below is a subset of output from **dbbuild -b** (in the above example written to dbbuild.out).

```
: loading batch_file_bf
Added 20 records to calibration
Added 2 records to instrument
Added 1 record to network
Added 20 records to sensor
Added 1 records to site
Added 20 records to sitechan
Added 38 records to stage
```

By running **dbbuild** a series of tables and a new directory, "response" are created. These tables and directories are the constituents of the database.

```
<my_cpu:EXPT> ls
```

```
y_db.instrument
my_db.sensor
my_db.stage
my_db.lastid
my_db.site
my_db.network
my_db.sitechan
my_db.calibration     my_db.schanloc       my_db.snetsta                response
```

You may find more detailed descriptions of common errors/warnings when running **dbbuild** in Appendix_Data, Tables 1 & 2.

c. View your database

Using **dbe** you can visualize the current information in your database. At this point there are no waveforms.

```
<my cpu> dbe my_db
```

dbe is a general purpose tool for examining, exploring and editing Antelope relational CSS databases. For a detailed description on how to use **dbe** please read man dbe.

d. Create mseed day volumes and add them to your database

Now you have a database that describes your network (by running dbbuild), but you have not associated any waveforms with the meta-data.

To add your waveforms details to your database, use the command **miniseed2days** with the –d option as shown below**.** This will create the miniseed day volumes (from your header-corrected mseed files in the ref_mseed directory) and create an extra table for your database with the information regarding the waveforms called *my_db.wfdisc*:

```
<my_cpu:EXPT> miniseed2days –d my_db –w \
"day_volumes/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j" ref_mseed/ >& \
my_miniseed2days.out
```

Where:

- w specifies an alternate pattern for the output miniseed volumes. This pattern dictates the way the data records are allocated to files. PASSCAL requires the following format for quality control purposes:

```
“your_day_volumes_dir/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j"
```

-d db Run miniseed2db on the output files to create the database db

> Note: the mseed headers read by miniseed2db are the source of information used to populate the database's waveform table. You must ensure the mseed headers in the station-channel-day files produced by miniseed2days are correct. If the database (and its batch file) do not describe all of the data then errors will result when we check the consistency of the database.

e. Assign calibration values from calibration table to wfdisc

To assure that the calibration values are incorporated into the just created wfdisc table, please run dbfix_calib:

```
<my_cpu:EXPT> dbfix_calib my_db
```

f. Verify the Integrity of Your Database

Before you create a dataless you will want to be sure that your meta-data completely describes the waveform data, and that your database is error free. **Read** the man page on *dbversdwf* and *dbverify* for other tests you can run on the database. Examples of suggested tests are:

```
<my_cpu:EXPT> dbversdwf -dtu my_db >& my_dbversdwf_output
                         0 bad files

<my_cpu:EXPT > dbverify –tj my_db >& my_dbverify.out
```

Please refer to Appendix_Data, Table 3 for possible scenarios you may have when running these tests.

g. Create the dataless SEED volume

The dataless SEED volume, often referred to as a “dataless”, contains the meta-data describing the station and instrumentation of your experiment. To generate the dataless SEED volume, run ***mk_dataless_seed***, which builds the dataless from the contents of your experiment’s database. You will submit this file along with the waveforms to PASSCAL.

```
        <my_cpu:EXPT > mk_dataless_seed –v –o PI.04.my_db.20042082000.dataless
my_db
                        Using existing my_db.snetsta table
                        Finished building dataless wfdisc
                        PI.04.my_db.20042082000.dataless truncated to 24576 bytes
```

Using the option -o you can provide the dataless-seed filename to use. Please use the following naming convention:

NN.YY.dbname.YYYYJJJHHMM.dataless

Where:

NN is your network code
YY is the year of your data
YYYYJJJHHMM is the approximate current time – year-Julian day-hour-minute

To convert from calday to Julian day, for example March 1, 2007:

```
        <my_cpu> julday 03 01 2007
                Calendar Date 03 01 2007
```

To find the current julday:

```
        <my_cpu> julday
                Calendar Date 03 01 2007
```

To convert from Julian day to calendar day, for example day 150 of 2006:

```
                <my_cpu> calday 150 2006
                Calendar Date 05 30 2006
```

Verify the dataless

Now you may check the structure of the dataless with **seed2db.**

```
                <my_cpu:EXPT> seed2db –v my_db_dataless_seed
```

Please refer to Appendix_Data **,** for possible cases you may run into when running *seed2db*.

> Note: the dataless must describe the entire data set, including all service runs of data. The agreement, or lack thereof, between the dbbuild batch file, resulting database and dataless, and waveforms will be reflected in the availability of the data at the DMC.

Send Data to IRIS/PASSCAL

When ready to submit the data to PASSCAL, please make sure that you all your data are complete and the dataless fully describes the network. One simple way to verify all data for each station/Julian day is complete is by running the PASSCAL tool called StaDayVols.py , which is part of the PASSCAL software release.

What is a full miniseed day volume?

At PASSCAL a full miniseed day volume is compound by each data stream and state of health channel defined on the dataless per station and Julian day. Thus, if in a dataless there is description for a STS2/Quanterra - Q330 station with 2 data streams (e.g. 40sps and 1sps) recorded in 3 channels each (Z,N,E => BHZ,BHN,BHE & LHZ,LHN and LHE respectively) from day 001 2010 until day 365 2010, a full miniseed day volume expected for each Julian day will include:

Full miniseed day volume GS11V/RT125 recording 110sps = ELZ + ELN +ELE

What is the purpose of running StaDayVols.py before sending the data to PASSCAL?

StaDayVols.py - Determines what files in indir can create complete Station Day Volumes. The channels needed for completeness are taken from the css db "db" which must have sitechan, snetsta, and chanloc tables. If outdir is given; complete station day volumes will be created in directory outdir. If -i is used incomplete station day volumes will be created as well.

When using StaDayVols.py to send data there is NO need to use the –i option, the purpose of this tool is for verification of what is complete or what is missing before sending data to PASSCAL for archiving. By typing in the command line StaDayVols.py –h you will get the usage of this tool:

```
usage: StaDayVols.py [options]
options:
  -h,       --help  show this help message and exit
  -v        Verbose; will also describe complete Vols
  -d         DB      CSS database to use for station description
  -f         INDIR    Dir to search for MSEED files
  -o         OUTDIR   Dir to place created vols, will be created if doesn't exist
  -i        Used with -o, will also create incomplete vols from files that exist
```

Once you have verified that your data is complete, please contact us by sending an email with your experiment name and network code to data_group@passcal.nmt.edu. For example: XO –Terra data 2004-2005.

Sending data - DoFTP

There are two options, using the command line or a GUI. We recommend use of **GUI_DoFTP** to submit data to the PIC (current version 2008.038 or later version). See Appendix K for more details on gui_DOFTP, which is a python-based package available

from PASSCAL as part of our software release:
http://www.passcal.nmt.edu/software/software.html.

```
<my_cpu:EXPT> my_path Gui_DoFTP
```

(Where my_path is where you have installed **DoFTP**.)

DoFTP will:

- Descend the specified directory path, identify, and pack ALL miniseed files found
- Create .tar and .md5 (similar to check sums) files of the data
- Send the dataless and its .md5 file
- Build a report (list) of all data files sent
- Start an FTP session to PIC and send the data

Note:

- Be as specific as possible when specifying the path to the data, so unintended files are not packed
- The software requires at least as much free disk space as the size of the data set to be sent. That is, if you have 100 GB of data to send, DoFTP will need at least another 100 GB of free space to build the tar files.

To use in the command line option, type **con_DoFTP**

```
<my_cpu:EXPT> my_path con_DoFTP
```

(Where my_path is where you have installed **DoFTP**.)

-# print version of this program
-a force ACTIVE FTP mode (default is PASSIVE mode)
-f ftp the tarred data or resume ftp from the last broken pt.
-r gives an integer from 1 to 366 (default is today's julday)
-t set FTP timeout with a positive integer (default: no timeout)
- help print help information

e.g.: `./con_DoFTP -a -f -r 366 -t 15 /Users/kxu/FA_tremor`

Document created by Eliana Arias-Dotson (2008),
modified March 10, 2009
modified April 20, 2009
Modified Eliana Arias-Dotson 2/8/11