Splitting, Cleaning, and Spikesorting
Post-Recording Tools For The 128 Channel DAQ2 Recorder

by
Dale Shuman

This document provides instructions on how to use a set of tools to process recordings made by the 128 channel DAQ2 recording system.

Some existing tools have been modified and new ones created. Tools created from existing tools, such as daq_split, retain the same name as the older ones but have a "2" added to the name, e.g., daq2_split. The existing utilities are unchanged and are available for processing older files. The new utilities do not have a "2" as part of the name.

The tools include:

split_all.sh	Split multiple recordings into individual channel files.
daq2_split	Program that does the actual splitting. Called by split_all.sh, can be used standalone.
make_chan.sh	Make the chanlist files used for cleaning. Clean_rec.sh needs these.
clean_rec.sh	Clean all the chan files for a recording.
make_label.sh	Make the label file that the spikesorter requires for a set of .chan files.
do_clean_data.sh	Workhorse script, called by clean_rec.sh, can be used standalone.
do_noclean_data.sh	Workhorse script, called by clean_rec.sh, can be used standalone.
do_clean_data2.m	Octave script used to clean chan files, called by clean_rec.sh.
daq2_unsplit	Combine a set of chan files into a single recording format file.

This document assumes you have created one or more experiment recordings. Instructions for running the daq2 recording software are provided in a separate document.

*** *Always split, clean, and run the SpikeSorter as ssu. It's just better that way. It's a permissions thing.*

One of the fundamental assumptions that all the tools make is that the initial name of the recording files conforms to a fixed format. The tools will not work correctly if this is not the case. The term "basename" is used throughout this document to refer to this. The basename format is:

YYYY-MM-DD

which is year-month-day. Additional fields are added to and removed from the basename at various stages in the processing.

What follows is a tutorial explaining how to use the tools using an example basename. Assume you have a set of three recordings for 2012-02-21.

2012-02-21_001_1-64.daq
2012-02-21_001_65-128.daq

2012-02-21_002_1-64.daq
2012-02-21_002_65-128.daq

2012-02-21_003_1-64.daq
2012-02-21_003_65-128.daq

SPLIT THE FILES

Each recording consists of one or two files. Each file contains 64 channels of data. The first processing step splits the channels into individual files.

- ☐ Login as ssu. (password = Obex@3040)
- ☐ In your directory of choice, typically in /raid/datamax on cisc3, create a dir with the basename of 2012-02-21. Copy the .daq files from the recorder, tape, or other source to this dir by your preferred method. This step may already be done. There is no need to make a second copy of the files.

- ☐ Open a command line window and cd to the dir with the .daq files.

- ☐ Run the split_all.sh script to split the .daq files into separate channels by typing:

    split_all.sh 001 002 003

This runs the daq2_split program for all the recordings. The script will look for both 1-64 and 65-128 files, but it is not an error if any of these do not exist. The daq2_split program will create split dirs if they do not exist and then split the data to separate chan files. Some user feedback is sent to the terminal window, such as percent complete.

After the script completes, you will have three new dirs with individual chan files in them. In our example, you will have:

split.001/
split.002/
split.003/

A typical file in split.001 would be: 2012-02-21_001_r_45.chan. The _r_ field means this is a raw file that has not been processed.

If you prefer to do this manually and have more control over the split operation and just split a few channels, or if you wish to process an older .daq file that does not have the 1-64 or 65-128 fields, you can invoke daq2_split directly. If, for example, you wanted to extract channel 23 from the 001 recording, you would type this:

daq2_split 2012-02-21_001_1-64 23

The daq2_split command is backwards compatible with .daq files from earlier recordings. These do not have the 1-64 and 65-128 fields. If these fields are missing, daq2_split assumes that it is dealing with a 1-64 channel file and names the output files accordingly.

*** *Splitting the files is I/O intensive! It's best if splitting is done on the machine where the data is mounted (i.e., cisc3).*

*** *The split files will take up quite a bit of space on the hard drive, so it's recommended that you wait to split a file until you are ready to SpikeSort the recording. If, for example, you want to split only recording #3, then run*
split_all.sh 003

CLEAN THE FILES

Most of the raw chan files (i.e., the array and nerve channels) must have a cleaning process applied to them to remove noise from the data. Others (analog traces like BP, CO2, trach pressure, and stimulus marker) require no processing and can be used as-is in down-stream processing.

- ☐ Before cleaning, you need to create channel list files for each recording that tell the tools used in this step what to do. In our example, on a command line while in the same directory as the .daq files, type:

  make_chan.sh 001 002 003

*** *Often, it is easier to just copy chan lists from time-adjacent recordings rather than start from scratch; look on datamax/"close dates" for existing chan lists. Of course, you should look in a copied file and edit it as necessary to fit the data.*

In August 2013, some of the channels on the 65-128 set were moved. The script prompts you with a question to choose to use the pre-August 2013 channel assignments or August 2013 and after assignments. Respond appropriately.

The script creates these files, with default values:

chanlist_001_1	chanlist_002_1	chanlist_003_1
chanlist_001_2	chanlist_002_2	chanlist_003_2
chanlist_001_3	chanlist_002_3	chanlist_003_3
chanlist_001_4	chanlist_002_4	chanlist_003_4
chanlist_001_5	chanlist_002_5	chanlist_003_5
chanlist_001_6	chanlist_002_6	chanlist_003_6
chanlist_001_7	chanlist_002_7	chanlist_003_7
chanlist_001_8	chanlist_002_8	chanlist_003_8
nocleanlist_001	nocleanlist_002	nocleanlist_003

You may want to edit these to exclude some channels or you may want to delete some of them completely if you know you will not need them. Note that there are lists for each recording, so you can alter them without affecting processing for other recordings.

*** *Be sure chan lists are appropriate for the current recording!*
*** *If you know that an array or nerve channel is "dead" (look on the day sheets for that recording), don't include it in the chanlist – and you can delete the split files for that channel, too. Save time and space!*

Some channels should not be cleaned. These are in the nocleanlist_* files. "Cleaning" for them means they are copied from the split dir to the clean dir. These are the analog traces that should not have noise on them: tracheal pressure, blood pressure, end-tidal CO2 waveform, and the stimulus marker channel.

Each channel file listed in the chanlist files must exist. The cleaner software will exit with an error if it cannot find a file.

Each chanlist file should contain a single line of channel numbers and two additional numbers that will be the name of files the cleaner creates. The default chanlist_001_1 file is this:

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 199 198

The last two items are used to name files that the cleaner software creates. These will contain pass one and pass two noise data that are calculated from a raw file to create a clean file. The convention is that the chanlist file number is the prefix for the 99 and 98, so, for example, in the chanlist3 file, they are 399 and 398.

- ☐ To process all the chan files, you need to run the clean script for each recording. In our example, to clean the 001 recording, on a command line while in the same directory as the .daq files, type:

clean_rec.sh 001

The script looks for the chanlist_001_* files and starts a cleaning process in the background for each set of channels in the chanlist files. The script expects to find chanlist_*_1 - chanlist_*_7 files, but it is not an error if any of these files do not exist. The cleaning for each chanlist runs concurrently. The OS will take advantage of multiple cores to keep all the CPUs busy. Output is sent to a log file named after the recording number and the channel list, such as chanlist_001_1.log. You can view updates to one of these in real time by typing:

tail -f chanlist_001_1.log

*** *Cleaning is CPU intensive. Each chan_list will be handled by one processor, so up to 9 processors could be tied up for quite a while. Consider using cisc 4 or cisc5 or ssh into cisc1. Do not run the cleaner within your Linux virtual machine.*

The files in the nocleanlist file are copied as-is to the destination directory. Since the split directories are usually deleted later, it is safer to decouple the split and clean dirs and not have the contents of one refer to the contents of the other using a symbolic link. A few duplicate files may exist, but probably not for long.

Once the 001 recording has been cleaned, you can move on to the next one and type:

clean_rec.sh 002

After that completes, type:

clean_rec.sh 003

It is possible to create a script that would iterate on a set of recording numbers. This is probably counterproductive because cleaning is a very CPU-intensive operation and if you give the CPUs too much work, they spend a lot of time switching between the jobs. You almost certainly want to wait for the "clean_rec.sh 001" command to finish before starting the next one. On the other hand, if you want to start a cleaning process for all the recordings and let it run unattended for a few hours (or a few days), you can issue all three clean_rec.sh commands in sequence.

The cleaning operations in our example will create three new dirs:

clean.001/
clean.002/
clean.003/

A typical file in clean.001/ would be: 2012-02-21_001_45.chan. Note that the _r_ field has been removed.

If the clean_rec.sh script does more than you want, you can get a copy of this from /usr/local/bin and edit your local copy to do a subset of the operations. The clean_rec.sh script calls do_clean_data.sh, which in turn runs an Octave script. The Octave script will create dirs if they do not exist, so you don't have to do this by hand. If you create variations of the clean_rec.sh script, don't forget to type:

./clean_rec.sh 001
./clean_rec.sh 002
./clean_rec.sh 003

to run your local copy instead of the global one in /usr/local/bin.

The do_clean_data.sh script processes the .chan files for all chanlist files. If you want to do this by hand for a single chanlist, on a command line while in the same directory as the .daq files, type:

do_clean_data.sh 2012-02-21_001 chanlist_001_1

SPIKESORTING

At this point, the recording data files have names and are in formats that the down-stream processing tools expect. Generally the next step is to spikesort some or all of the channels.

- ☐ The spikesorter expects an input file in the directory where the .chan files reside. A convenience script will create this for you. If you want to spikesort the cleaned recording 001 files, cd to the directory with the daq files, then cd to clean.001. The cleaned files for recording 001 will be here. To create the label file, type:

make_label.sh

This script will look for a .chan file in this dir and if it finds one, it extracts the basename and recording number and creates the label file from this. If there are no .chan files, the script prints an error message. In our example, the file is:

2012-02-21_001.lbl.

You can also type "make_label.sh some_other_name" to create a label file with a name you supply. Note that if this name does not conform to the naming conventions that the spikesorter expects, it may not be able to find .chan files unless you also rename them.

- ☐ To start spikesorting for recording 001, run spikesort_control_panel, giving the complete path and name of the .lbl file without an extension, like so:

spikesort_control_panel /raid/datamax/2012-02-21_001/clean.001/2012-02-21_001

Consult the spikesorting documentation for more information on spikesorting.

You can spikesort the raw data. The method is the same, cd to the split dir, use make_label.sh to create the label file, then run spikesort_control_panel, providing the complete path to the label file.

There is a variety of other existing tools, such as waveform.tcl, that can be used to access the .chan files. These should all function without any changes.

CLEANING OLDER RECORDINGS

The _r_ field is not present in older .chan files produced from DataMax recordings. A flag has been added to the current versions of the daq2 package to handle this case. The --no_r flag is recognized now by these functions:

clean_rec.sh
do_clean_data.sh
do_noclean_data.sh
do_clean_data2.m

The most typical case is to use clean_rec.sh to control the down-stream operations. In our example, you would use this:

clean_rec.sh 001 --no_r

Note there are two dashes in the --no_r flag. (This is a Linux convention for flags.)

You can invoke do_clean_data.sh, do_noclean_data.sh and do_clean_data2.m directly with these flags. You can read the scripts or invoke them with no arguments to see the expected value and order of the parameters. In general, the --no_r flag is expected to be the last argument in the argument list.

You may have to rename the directory where the raw .chan files reside to split.001, and so on, to conform to what the daq2 software expects.

*** *The easiest way to clean DataMax recordings is like this:*

do_clean_data.sh filename chanlist_00#_# --no_r

(e.g., do_clean_data.sh 2012-02-21_001 chanlist_001_1 –no_r)

UNSPLITTING

daq2_unsplit

The daq2_unsplit utility takes a set of 1-64 and 65-128 channel files and combines them into files that have the same format as a live recording. The intended use of this is to take a set of cleaned chan files and create a recording-format file for subsequent viewing and screen captures. The program must be run in a command line window in the directory where the .chan files are located. It can also be used to recombine any older recording that generated .chan files, such as Datamax recordings. It assumes that all the files in the current directory are from the same recording, and will extract the date and recording number from the first .chan file it finds in the directory to compose the basename for the output files.

The program is invoked like this:

daq2_unsplit [-1] [-2] [-t tag] [-f]

-1 This optional argument means to create a recording file from channels 1-64.
-2 This optional argument means to create a recording file from channels 65-128

If neither of these are present, both files are created if the chan files exist. If just one of these is present, the other recording file is not created.

By default, the output .daq file has "_clean" added to the name, such as 2012-02-21_001_clean_1-64.daq. The optional argument:

-t tag

allows you to over-ride this and use any tag text of your choice. Typing "-t allmine" would result in an output file name of 2012-02-21_001_allmine_1-64.daq

-f This optional argument tells daq2_unsplit to over-write existing .daq files without prompting for permission. The program will not over-write an existing file by default. If it detects one, it prompts the user for permission to over-write it. This flag is include for scripting when it may not be desirable to prompt for user input.

The play_daq2 playback utility can read and display the files that are created by daq2_unsplit. This is available in a separate package, which includes documentation covering how to use it.

CREATING .bin FILES

The Cambridge Electronic Design recording system includes the Spike2 Windows program. It can import data from other systems in a variety of formats. There are two command line programs to create files from .daq or .chan files that are suitable for importing into Spike2. The following describes them.

daq_to_bin -r recording_num [-f] [-t tag_text] [subset of chans, e.g. 2 3 7-19 119 127]

This utility must be run in the directory where the daq file(s) are located. It assumes it will find one or two files with the record number and .daq in the names. There are no assumptions about the directory name. If there are more than one 001 recording with a different name in the same directory, it will only find the first one. It is not an error if the 65-128 file is missing. Older .daq files without the channel range numbers are assumed to be channels 1-64 files. It assumes the .daq file has the general format of YYYY-MM-DD_RECNO.daq, YYYY-MM-DD_RECNO_1-64.daq, or YYYY-MM-DD_RECNO_65-128.daq. It creates a file based on the basename of the .daq file using the date fields and adds _spike2-number_of_chans.bin to the file name. For example, 2013-03-17_spike2_16.bin.

-r recording_num	The parameter can be "001" or just "1".
-f	The program will not over-write an existing file without asking first. This forces an over-write of an existing file, which is something you may want to do in non-interactive scripts.
-t tag_text	This is text that gets added after the number_of_chans field. If you are creating several .bin files with the same number of channels, you need to use this to create unique file names. For example, 2013-03-17_spike2_16_first.bin and 2013-03-17_spike2_16_second.bin.
channels	The program will include all of the channels in the output file by default. More typically, the user will want a subset of the channels. The numbers are 1-based. Ranges can be included. The channels are written to the output file in numeric order, the order in the channel list is not important. Duplicate numbers are ignored, the channel will only be included

chans_to_bin [-f] [-t tag_text] [subset of chans, e.g. 2 3 119 127]

This utility must be run in the directory where the .chan files are located. It assumes all of the .chan files are from the same recording. There are no assumptions about the directory name. It extracts the basename for the output file from the first .chan file it finds. It assumes the file names are of the general format YYYY-MM-DD_RECNO_r_CHAN_NUM.chan or YYYY-MM-DD_RECNO_CHAN_NUM.chan . Missing .chan files are not an error, a zero will be written for those channels. It creates a file based on the basename of the .chan file and adds _spike2-number_of_chans.bin to the file name. For example, 2013-03-17_spike2_16.bin.

-f	The program will not over-write an existing file without asking first. This forces an over-write of an existing file, which is something you may want to do in non-interactive scripts.
-t tag_text	This is text that gets added after the number_of_chans field. If you are creating several .bin files with the same number of channels, you need to use this to create unique file names.
channels	The program will include all of the channels in the output file by default. More typically, the user will want a subset of the channels. The numbers are 1-based. Ranges can be included. The channels are written to the output file in numeric order, the order in the channel list is not important. Duplicate numbers are ignored, the channel will only be included

This concludes the tutorial.

THE SHORT FORM

- ☐ 0. Login as ssu. (password = Obex@3040)
- ☐ 1. Create YYYY-MM-DD dir if it does not already exist.
- ☐ 2. Copy .daq files to it if they have not been copied from the recorder, tape, or other source.
- ☐ 3. Start a command shell and cd to dir with .daq files.
- ☐ 4. type: split_all.sh 001 002 003 etc., for all recordings
  *(*** It's recommended that you split only the recording(s) you're working on.)*
- ☐ 5. type: make_chan.sh 001 002 003 etc., for all recordings
  *(*** or copy and edit the chan_lists and the nocleanlist from an similar recording)*
- ☐ 6. Edit or delete chanlist* files as required
- ☐ 7. type: clean_rec.sh 001, wait for completion (or not)
  type: clean_rec.sh 002, wait for completion (or not)
  type: clean_rec.sh 003, wait for completion
  etc., for all recordings
  *(*** It's recommended that you clean only the recording(s) you're working on.)*

  Older .chan files produced by the daq_split program do not have the _r_ field.
  To process these files, use this command: clean_rec.sh 001 --no_r

  You may have to rename the split directories to conform to what the daq2 package expects.
- ☐ 8. cd to clean.001 dir
- ☐ 9. type: make_label.sh *(*** or copy .lbl, .wrk, and .num files from a similar recording)*
- ☐ 10. edit .lbl, .wrk. and .num files as required.
- ☐ 11. spikesort_control_panel /raid/datamax/YYYY-MM-DD/clean.001/name-of-label-file-without.lbl
  ("/raid/datamax/YYYY-MM-DD/clean.001/YYYY-MM-DD_001")
- ☐ 12. Go to 8, substituting 002 for 001, and so on for all recordings.

You have some flexibility here. While the order of some of the operations is fixed, they apply to a single recording. You could, for example, do all the steps for just recording 001, and then repeat the entire procedure for each subsequent recording, or do all the splitting first, then clean and spikesort 001, then repeat for 002, etc.