Guide to using a flight mill to measure speed and endurance of insects.

USDA-ARS
Flight Mill User Guide

Measuring stored-product insect flight activities October/2023

Flight.Mill

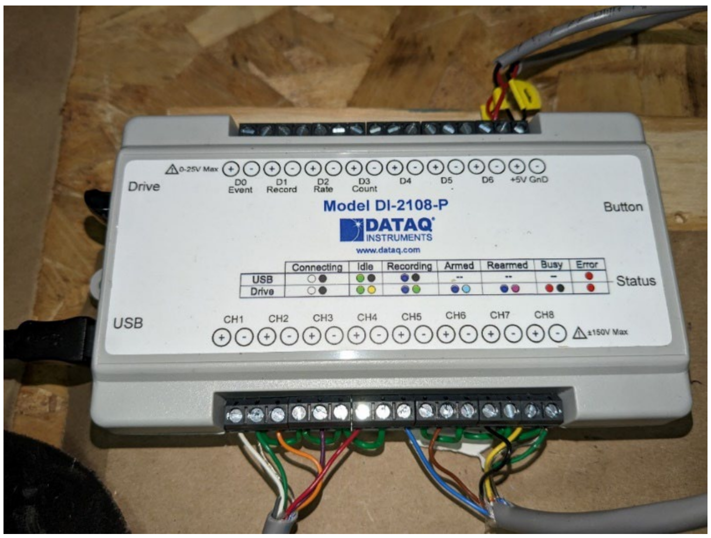
Contents

Flight Mill

The flight mill is made to study the flight activities of stored product insects. This unit was based on a similar unit which is housed at KSU entomology. The flight mill has 8 cells or units. The cells are numbered 1-8 and are matched with the data acquisition system. Cell number 1 is the top and left. Each cell contains a flight rod assembly which is held by a pair of magnets. The flight rod assembly has a vertical pin shaft which is housed inside a pipette tip with hot glue. The flight rod is horizontal with one end for a flag and the other end to hold an insect. The flag is used to interupt the LED/PT sensor which changes the analog signal for low voltage to higher voltage. An analog signal is recorded by the data acquisition system. The signal needs additional processing to determine the pulsed signal which is produced. The flight times per pulse event can be determined. In addition, the number of pulses can be determined per test period and other related parameter determined. The magnets are housed in red plastic and are adjusted so the flight rod flag passes through the LED/PT sensor. Also, the gap between the vertical pin shaft and the magnets is adjusted for minimal friction movement.

Data Logger Electronics

The signal/pc interface hardware for this project was the DATAQ Model DI-2108-P data acquisition interface and software. It has +5 VDC and ground terminals for supplying power to the 8-LED/PT sensors. The hardware includes an array of screw-terminals for connecting the wires from the 8-LED/PT sensors. This electronics is interfaced to a PC via a USB cable. Currently, a desktop computer is used for collecting the signal data from the 8-LED/PT sensors. The analog signals are converted to digital signals and transferred to the computer for data collections and processing.



Step 1: Data collection with WinDaq Software

The data acquistion software is only for collecting and storing the flight mill data. Separate software programs are required to compress the data file and then to process the data into summary information. To initiate the data acquisition software, **WinDaq**, click on the icon on the desktop.

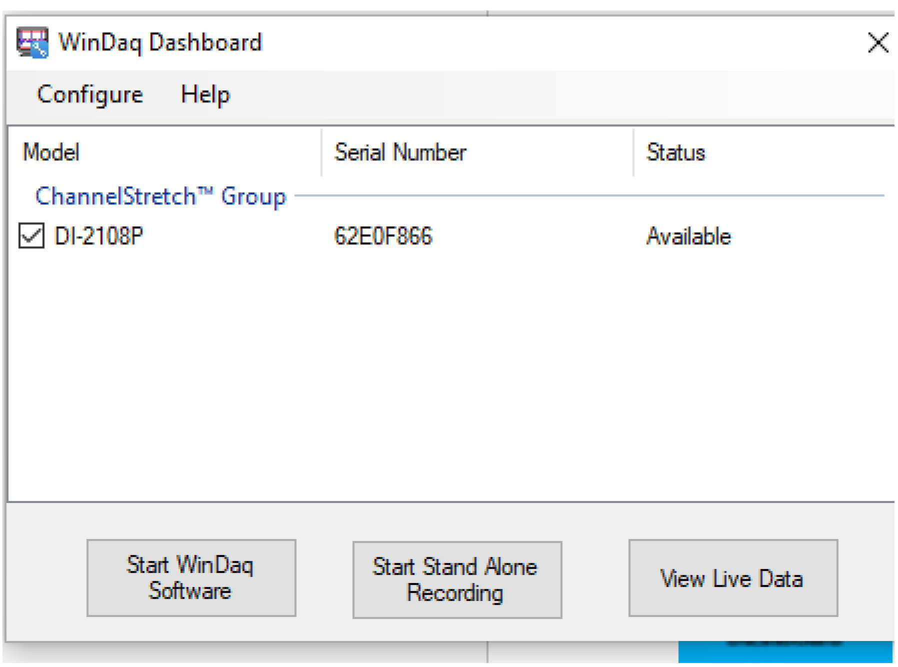
The **WinDaq Dashboard** is shown below.

By default, the DI-2108P is not checked.

Check the box for DI-2108P.

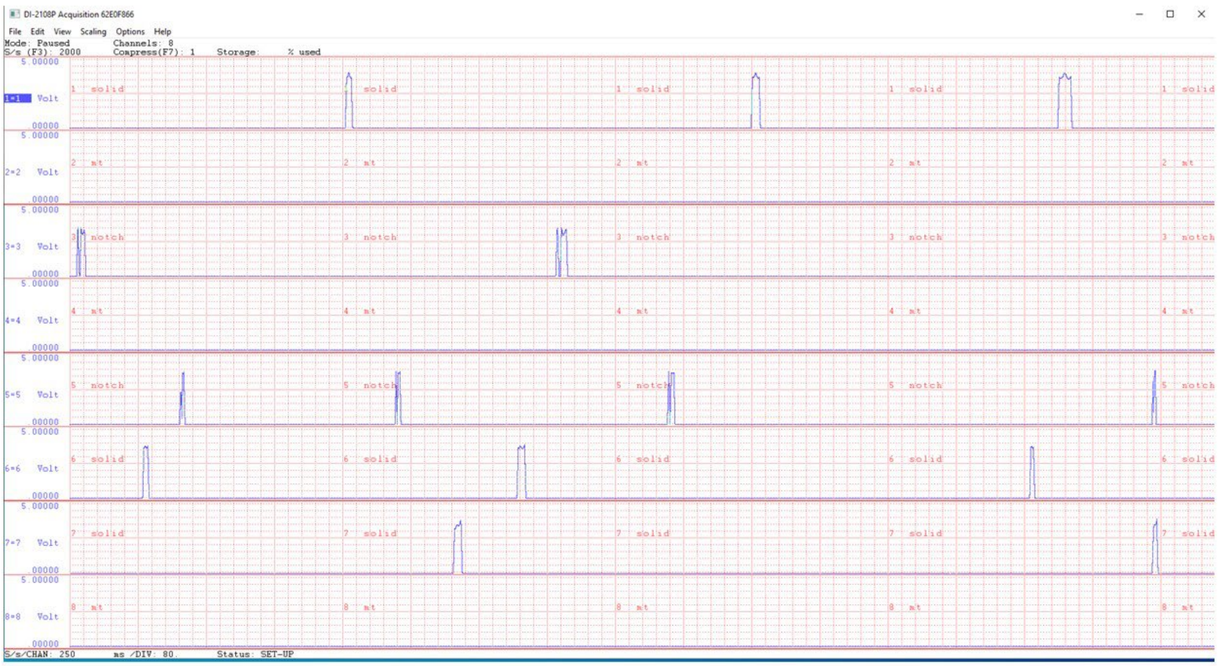
At the bottom of the dialog box are three button / options.

Click "**Start WinDaq Software**" to continue to the next step.



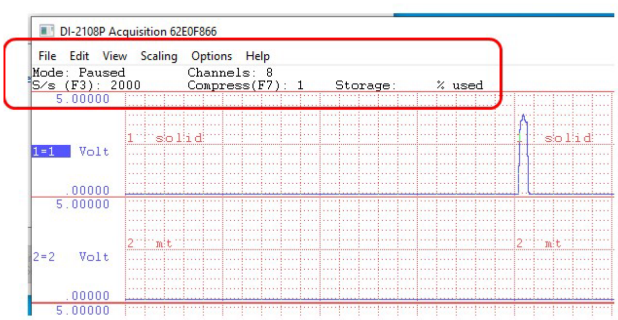
Basic Screen – DI-2108P Acquisition Software

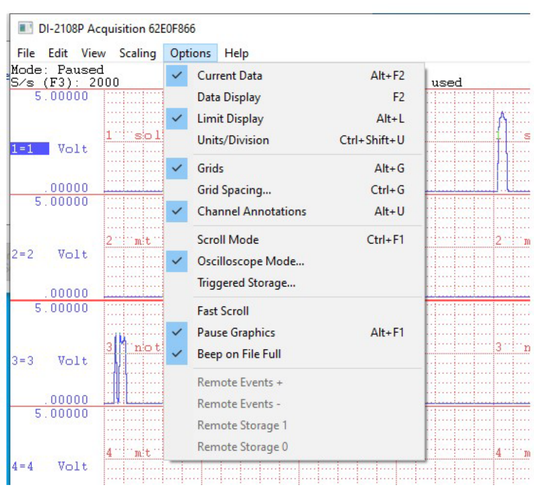
The data logging is like operating an oscilloscope. Below is an example of data being displayed in "free run" mode. In the example shown, five channels are active and show peak signals as the flight mill flag interrupts the LED/PT sensor. Three channels do not have flight rods in place.



Oscilloscope settings and options

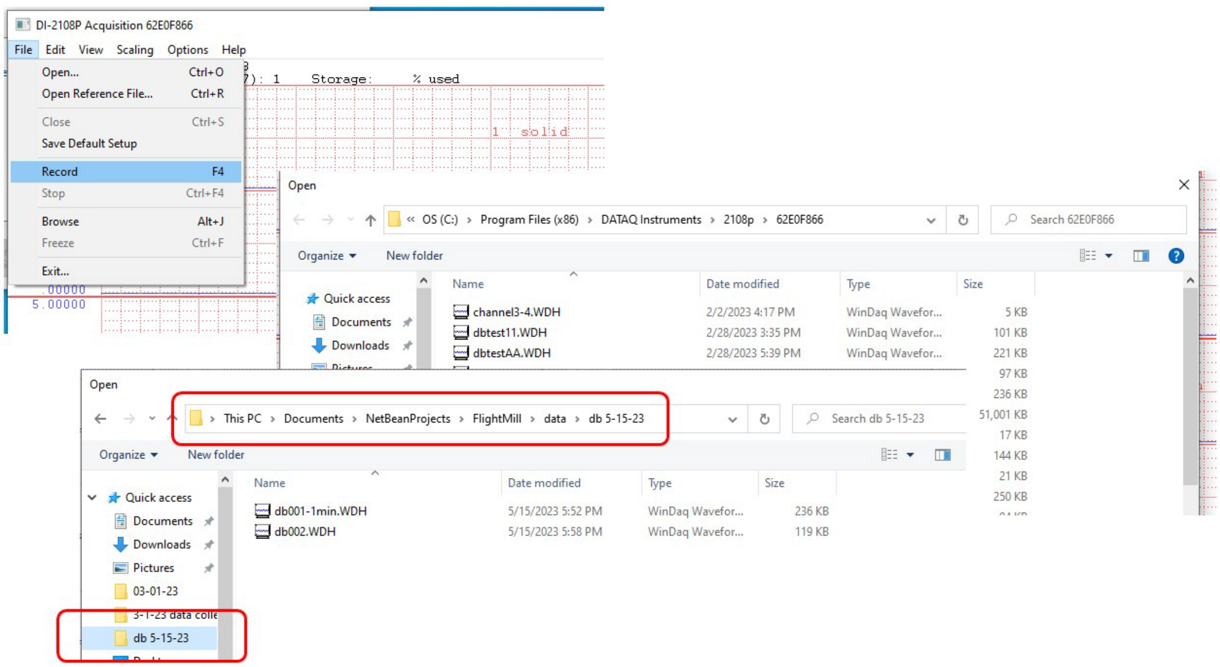
The display can be changed with many op. ns. The header gives info on the sampling rate, data compression, and % data collect. We are currently using 2000 samples/sec with 8 channels or 4ms/pt. Under the "options" menu are many settings that could change the way this digital oscilloscope looks and operates. If the data is not actively displayed, then the "Pause" button might be checked.



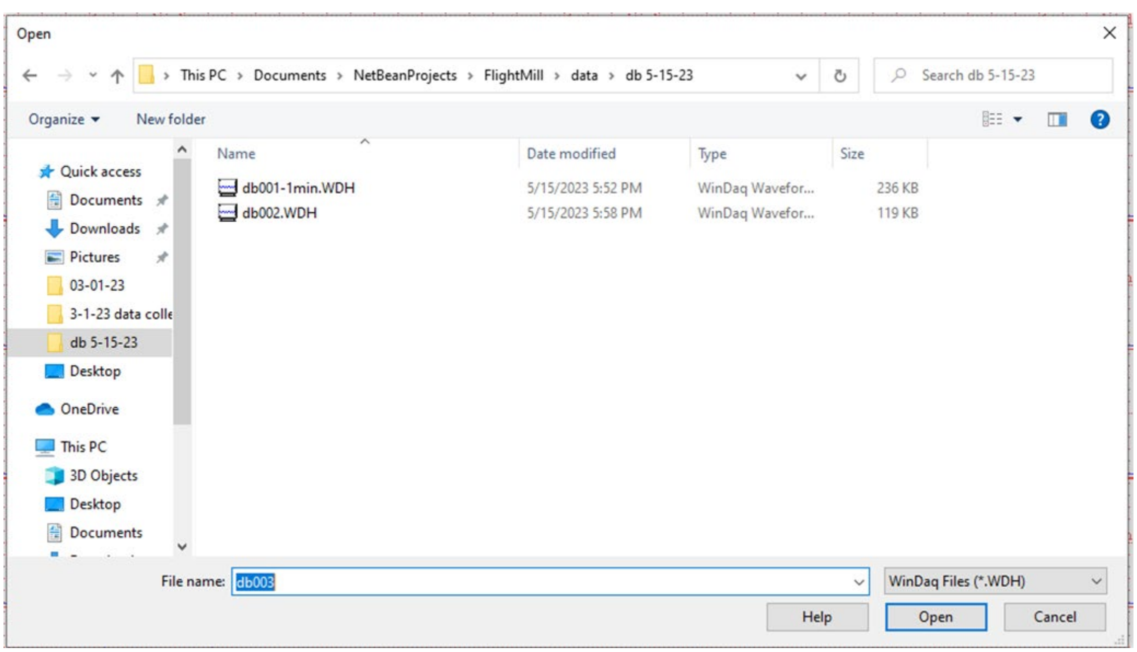


Recording LED/PT signal data

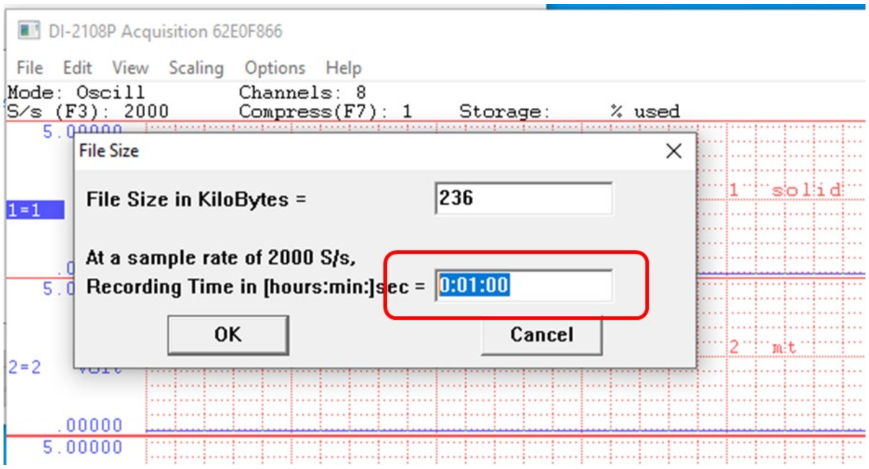
When you are ready to record data, select "File" menu, click "Record" option. Navigate to the desired storage location.



This file is in the internal format of WinDaq.

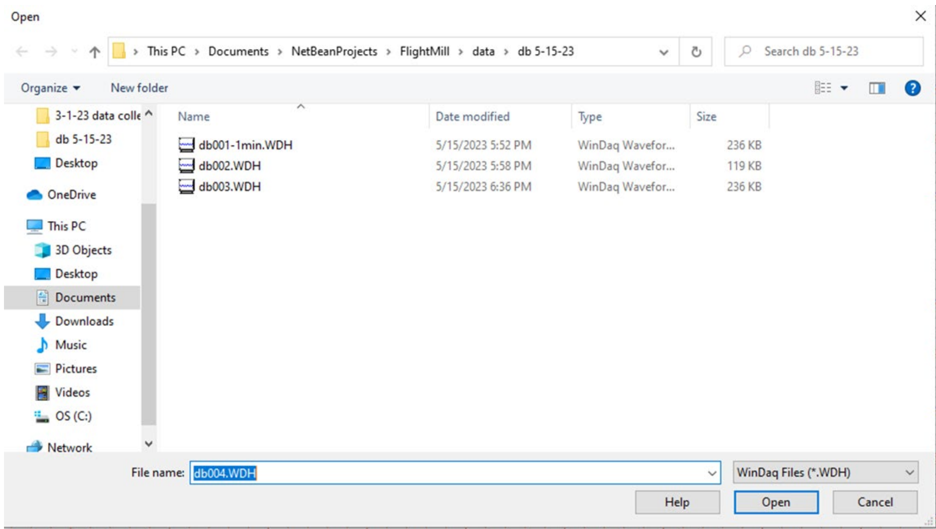


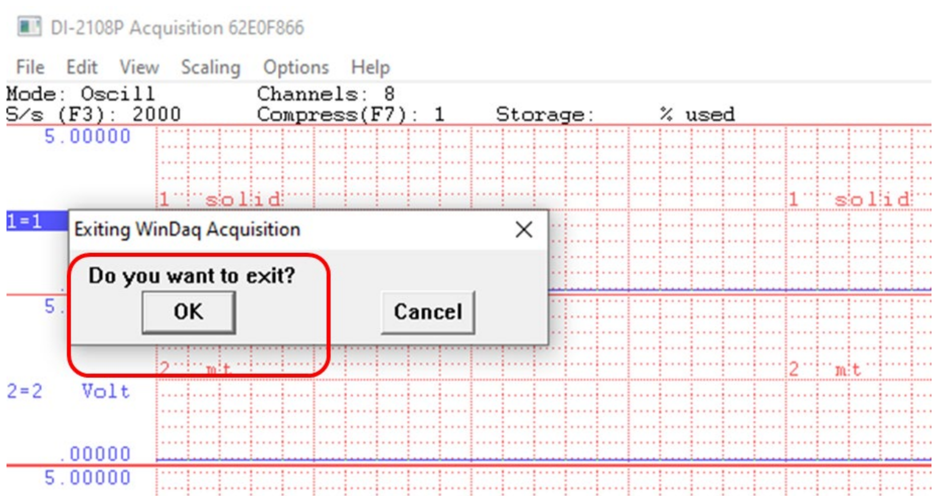
Specify the length of the session. This example shows one minute of recording time. After it starts, it shows that a .WDH file is being created. The recording is finished, either by timing out or being manually stopped.



The progress of the data collection is displayed in the header info line.

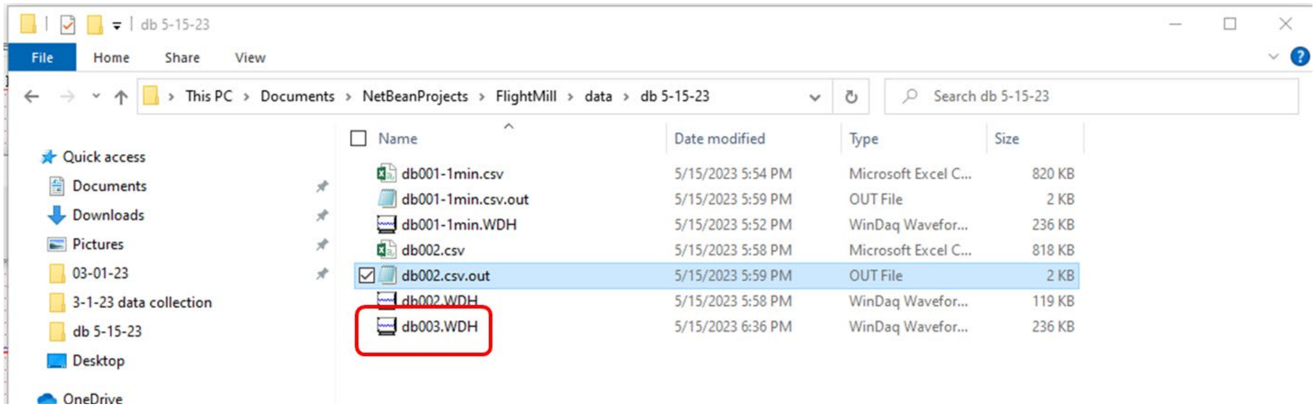
After the recording is completed, the user can continue to collect to the next file which is automatically incremented in the filename. Or the user can exit the data collection mode.



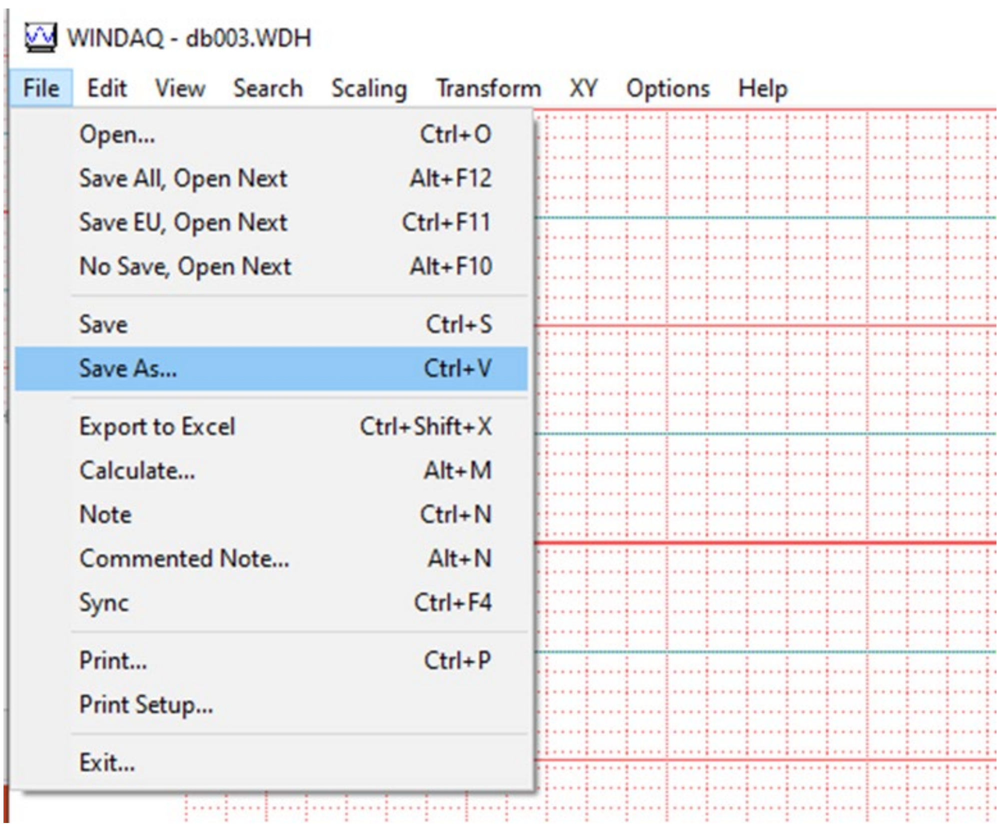


Step 2: Convert the .WDH file to a .CSV file.

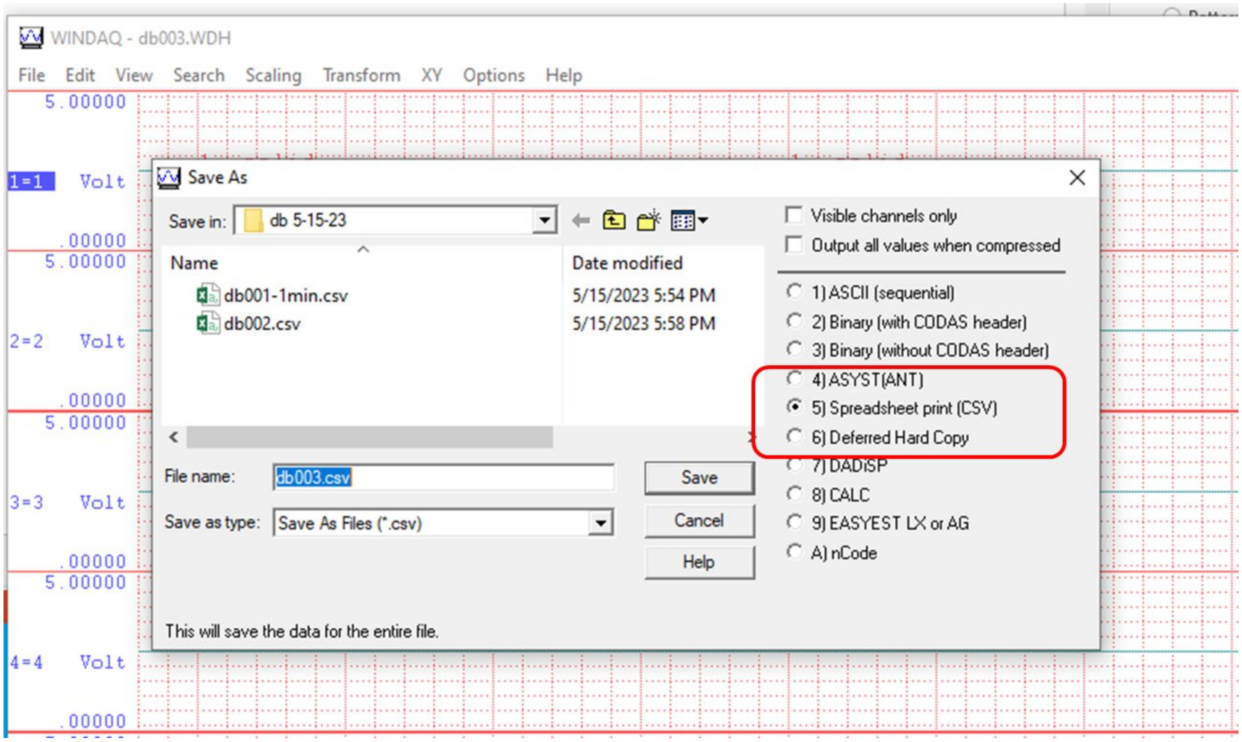
The WinDaq file needs to be converted to a text type file. It is not apparent, but the WinDaq software has two modes. The initial mode is for data collection. To get to the second mode of the software, the user needs to navigate to the directory which contains the ".WDH" file and double click on that file.



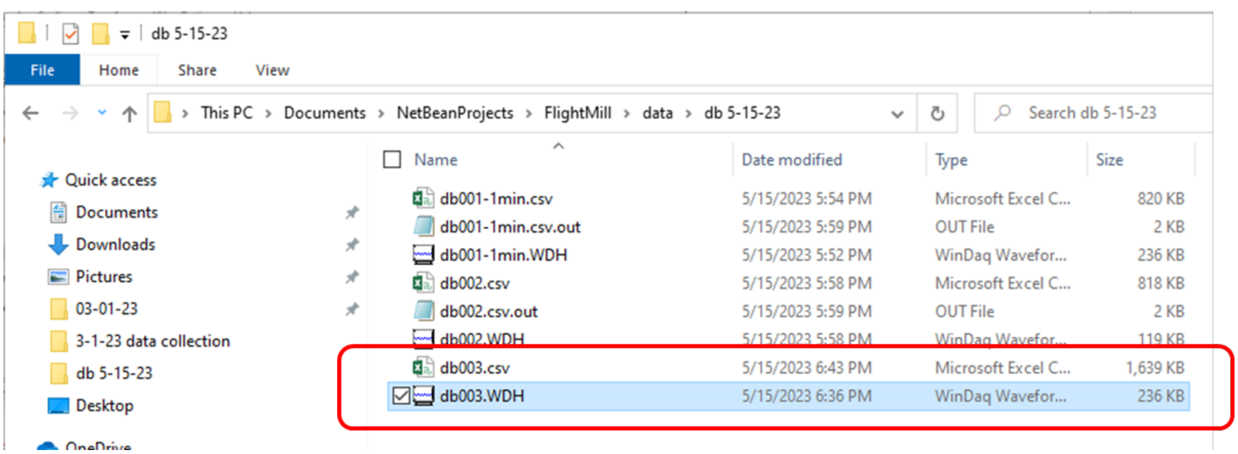
The file will open into a screen that looks like the previous screen; however, the menu options are a little different. In this software, select "File" and "Save As...."



A dialogue box with be displayed with save options. Select the radio button for “5) Spreadsheet print (CSV)” and “Save”.



After the file is saved, now 2 files for “datafile” should appear in the windows explorer. In this example, the files are db003.WDH and db003.csv.



Step 3: Compression Software with USDA-ARS software:

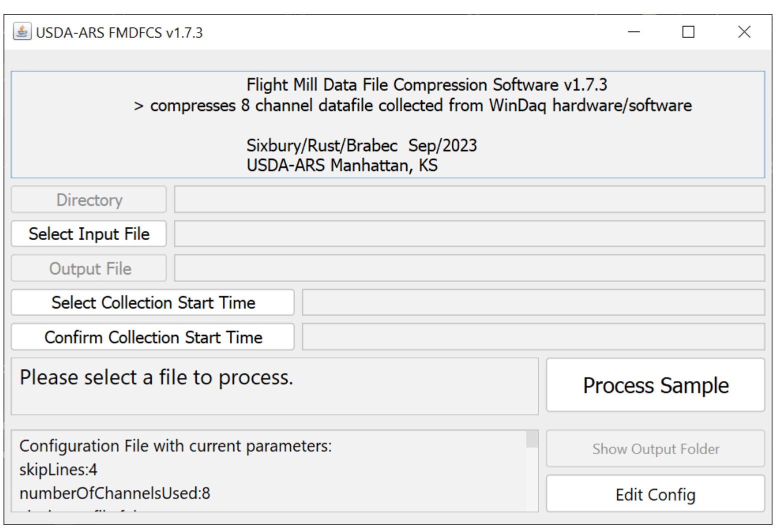
This software was written by a team from USDA-ARS of Manhattan, KS: Nicholas Sixbury and Bill Rust and Dan Brabec in the spring of 2023. The Flight Mill Data Compression software was developed to reduce the file size of the data collected with the flight mill and Windaq software. The Windaq datafile contains data information for each 4 ms and this file tends to be extremely large. It contains a lot of data that are basically ‘0’ values. The ‘0’ value data can be removed and processed resulting in peak-to-peak timing information. The compressed file saves data for every peak signal, as generated by the flight rod flag passing through the LED/PT detector. This program makes a csv file which contains a summary of the peak data. This peak data is stored into a file with the same name as the input filename, however, a “.out” suffix was added. Also, a graphical user interface (GUI) was developed to make usage easier.

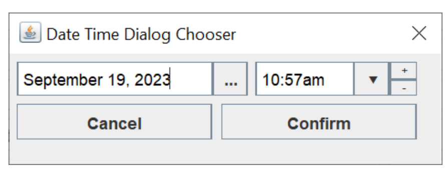
Below is the icon which will be located on the desktop of the computer. Double click this icon to initiate the program.

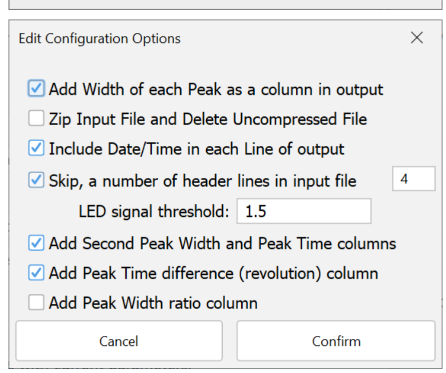
How to use the compression software:

1. **Open the program**, using the mouse and click on the icon.
2. Choose the data file to compress by clicking the “**Select Input File**” button.
3. Check your preferred options.
4. Select the time when data collection started on the flight mill machine.
5. Click the “**Process Sample**” button to process the selected file.

The processed file will be outputted to the labelled output file path.
The output file is placed in a subdirectory adjacent to the input file.
Each output file is timestamped, so output file should not overwrite each other.







Description of current options in the software.

- Input File
  - This is the file that will be loaded into the program and processed. To browse for a file to select, click the “Select Input File” button.

- Output File
  - This is the path of the output file. It will be automatically generated to place the output file near the input file, but you can manually change it yourself if desired.

- Add Width of each Peak as a column in output
  - This adds an additional column in the output file which displays across how many samplings a particular peak lasted. Peaks are defined as when the sample is above the threshold at a particular millisecond, and the width is the number of continuous milliseconds that channel was above the threshold.

- Zip input File and Delete Uncompressed File
  - This will replace the input file with a compressed version of itself.

- Include Date/Time in each Line of Output
  - This will add a new column to the output file with the date and time of each peak.

- Skip a number of header lines in input File.
  - This option allows you to tell the program to ignore a certain number of lines at the top of each input file, useful if you have four rows of header information which don’t contain data.

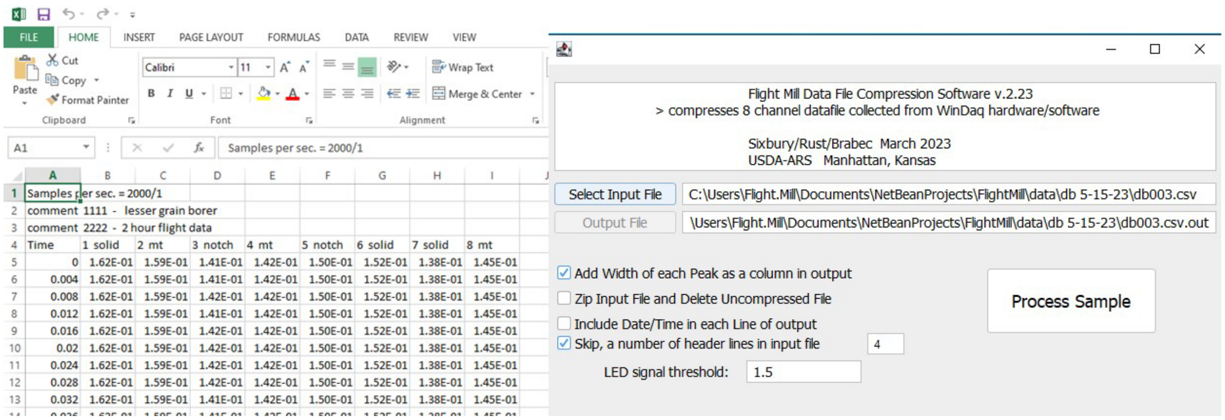
- LED signal threshold
  - Any reading above this threshold will be considered a peak. The possible readings range from ~0 to ~3, and the default threshold is 1.5.

- Add Second Peak Width and Peak Time Columns
  - This option allows you to get the information on the second peak in addition to the first peak when looking at a line with information from a notched flag. As such, this option adds two columns to the output.

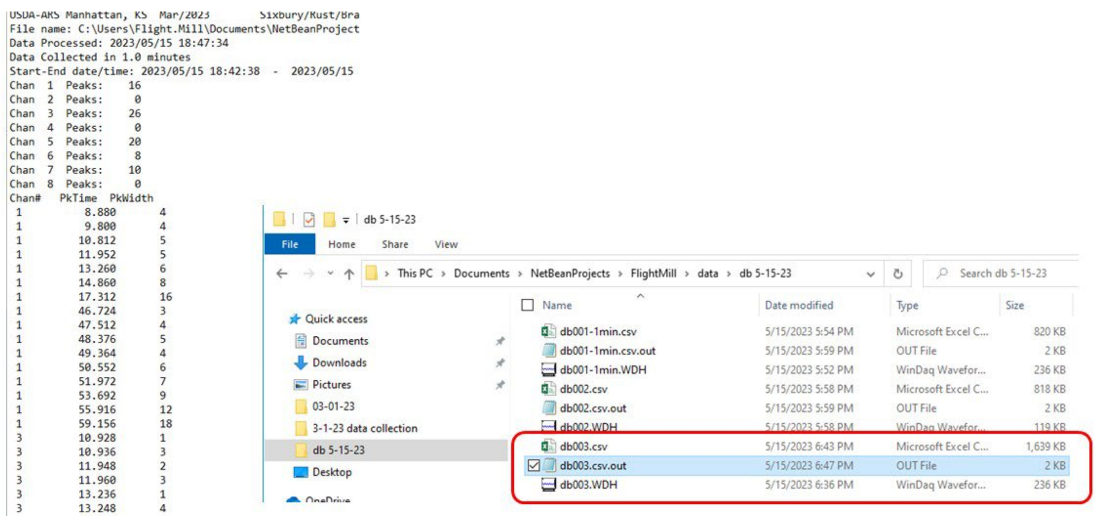
- Add Peak Time Difference (revolution) Column
  - This option adds an additional column with the difference between the current and previous peak time. Because of how the flight mill works, this gives you the amount of time it took the flag to make one full revolution around the flight mill.

- Add Peak Width Ratio Column
  - This option adds an additional column with the ratio of the first and second peak widths from a notched flag.

Before the file is compressed, the data is saved for every 4 ms and is a very large file.



After compression, the data is stored into peak sum info and is a much smaller file.



Over the course of development, additional columns and information were added to the output.

```
Chan  1  Peaks:  1129
Chan  2  Peaks:   191
Chan  3  Peaks:  1487
Chan  4  Peaks:     0
Chan  5  Peaks:  2324
Chan  6  Peaks:  2717
Chan  7  Peaks:   497
Chan  8  Peaks:    29
Chan#   PkTime1      PkTime2      Date and Time        PkWdth1     PkWdth2 Direc     Rev
 1        2.520        2.584      2023/09/14 16:02:02     8         21      1        2.520
 1        8.192        8.244      2023/09/14 16:02:08     6         17      1        5.672
 1       13.372       13.412      2023/09/14 16:02:13     5         13      1        5.180
 1       17.424       17.460      2023/09/14 16:02:17     5         13      1        4.052
 1       21.244       21.276      2023/09/14 16:02:21     5         12      1        3.820
 1       24.720       24.752      2023/09/14 16:02:24     4         12      1        3.476
 1       28.472       28.508      2023/09/14 16:02:28     5         12      1        3.752
 1       32.500       32.548      2023/09/14 16:02:32     6         16      1        4.028
```

Software Requirements

- Java Runtime Environment 8 (JRE)
- Flight mill data files as collected from Windaq software with following specifications:
    - 8 channels measured on flight mill.
    - Sample speed of once per 4 milliseconds.

Machine/Directory Specification

At the time of writing, the only machine hooked up to use the Flight Mill and software is a desktop computer found in the main building of the USDA Grain Research Center on College Avenue in Manhattan, Kansas. The computer name is ARSKSMHK4L118NI (HP39MN2), and the computer type is labelled as OptiPlex 7050. The specific directory where the software is located:

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v0.2.22 Released Mar 09, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v0.3.1 Released Jun 13, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v0.3.2 Released Jun 16, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.0.1 Released Aug 04, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.1.0 Released Aug 07, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.2.1 Released Aug 15, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.3.0 Released Aug 22, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.4.2 Released Aug 22, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.5.1 Released Aug 25, 2023

C:\\Users\Flight.Mill\Documents\Flight Mill Versions\v1.7.3 Released Sep 15, 2023

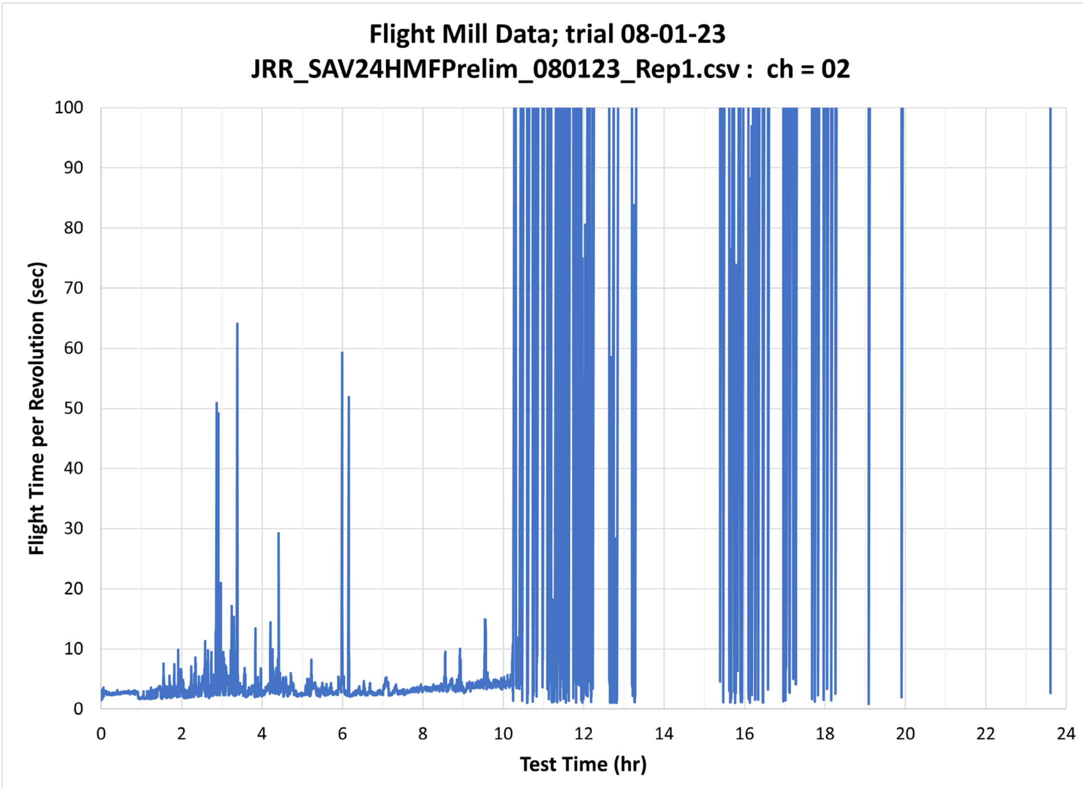
The desktop shortcut is labelled **FlightMill.jar.**

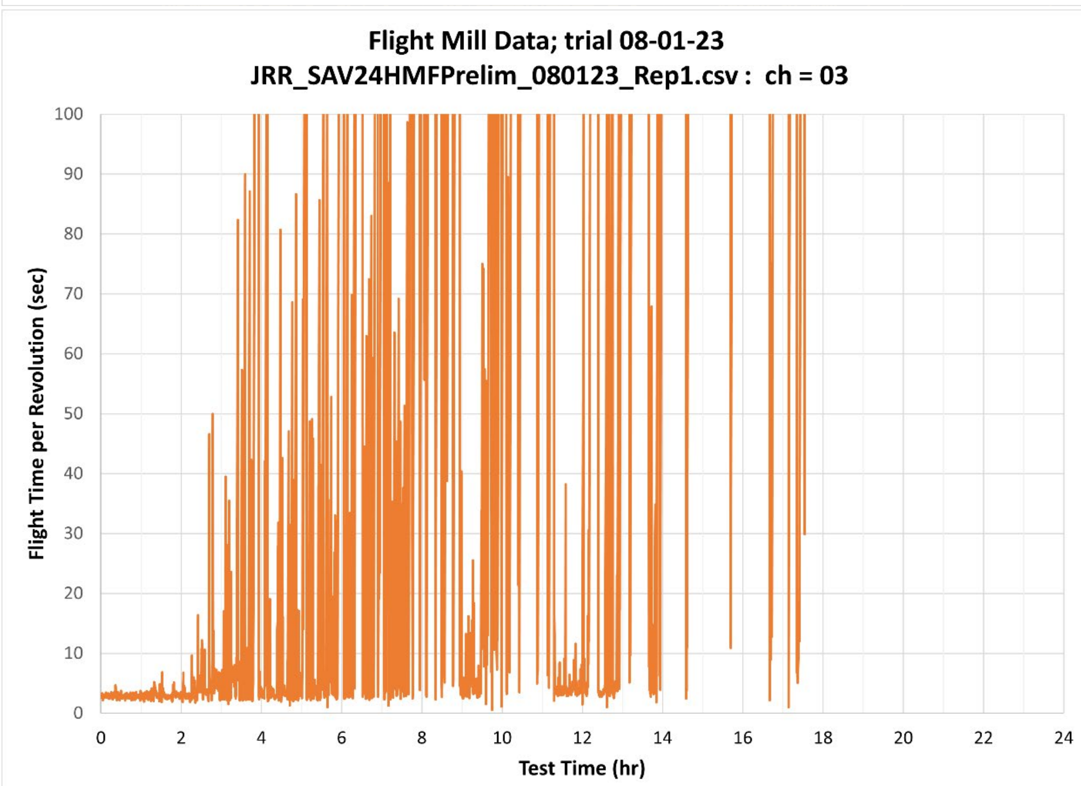
Step 4: Further Processing by Researcher / Summarizing Flight Data

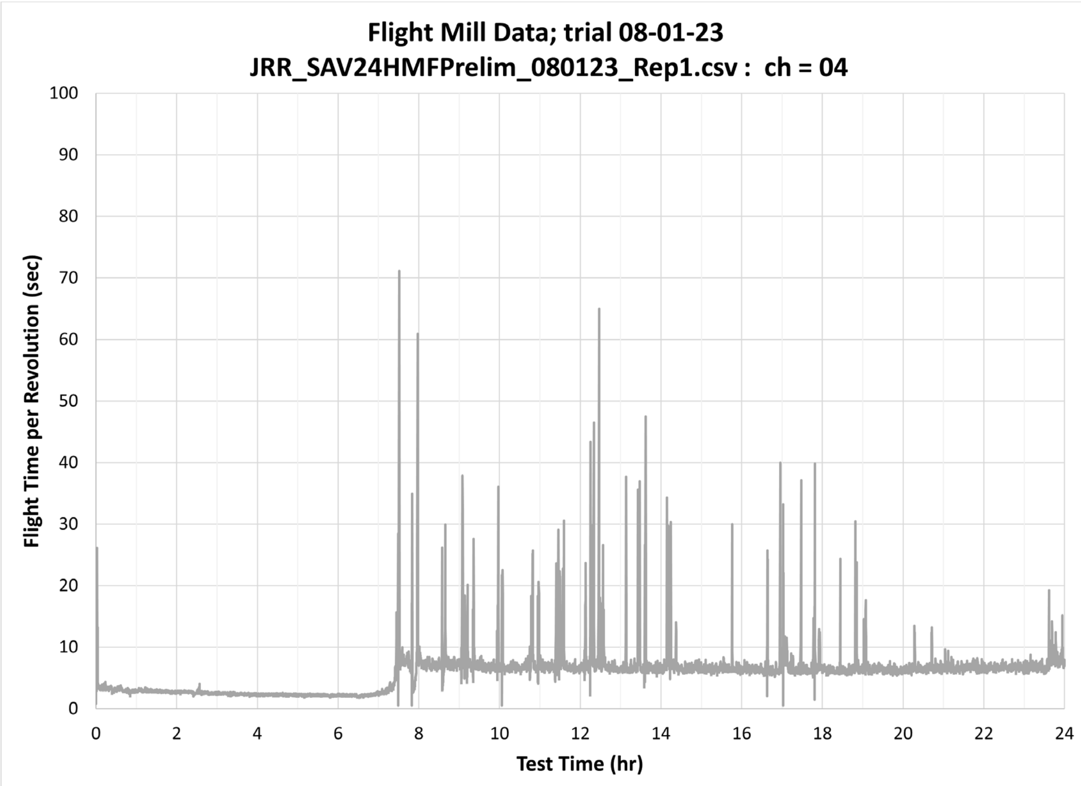
The following charts were created from the output of 24-hour data from the Flight Mill, only using six channels. The first six channels simply show the rate of activity over a 24-hour period.

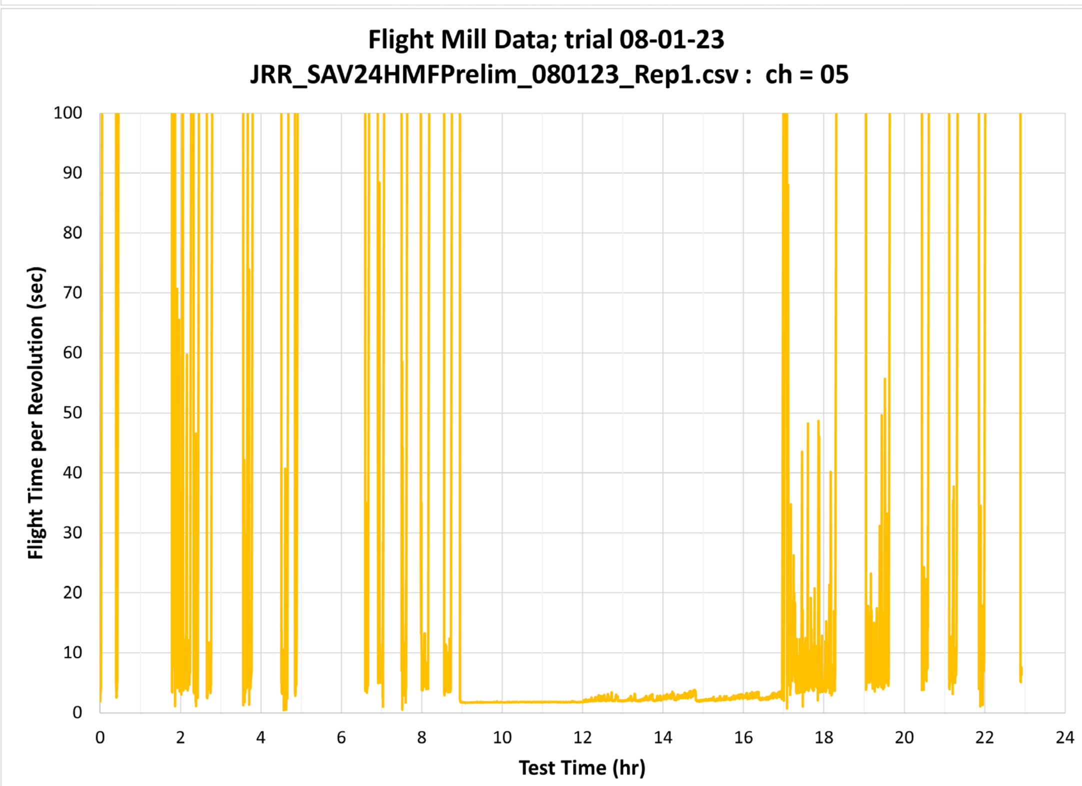
The last two contain information summarizing all the channels as a whole and are useful for debugging purposes and in understanding how the notched flags work. When using the notched flag, two peaks will be generated each time the flag passes the sensor. Work was done to find a way to identify if peaks came from the same flag pass or not, and it was found that a combination of peak width and the difference of time between the two peaks serve as a deterministic. For the most part, the time difference between the notches is less than a second, so the last two graphs focus on the length of time between notches.

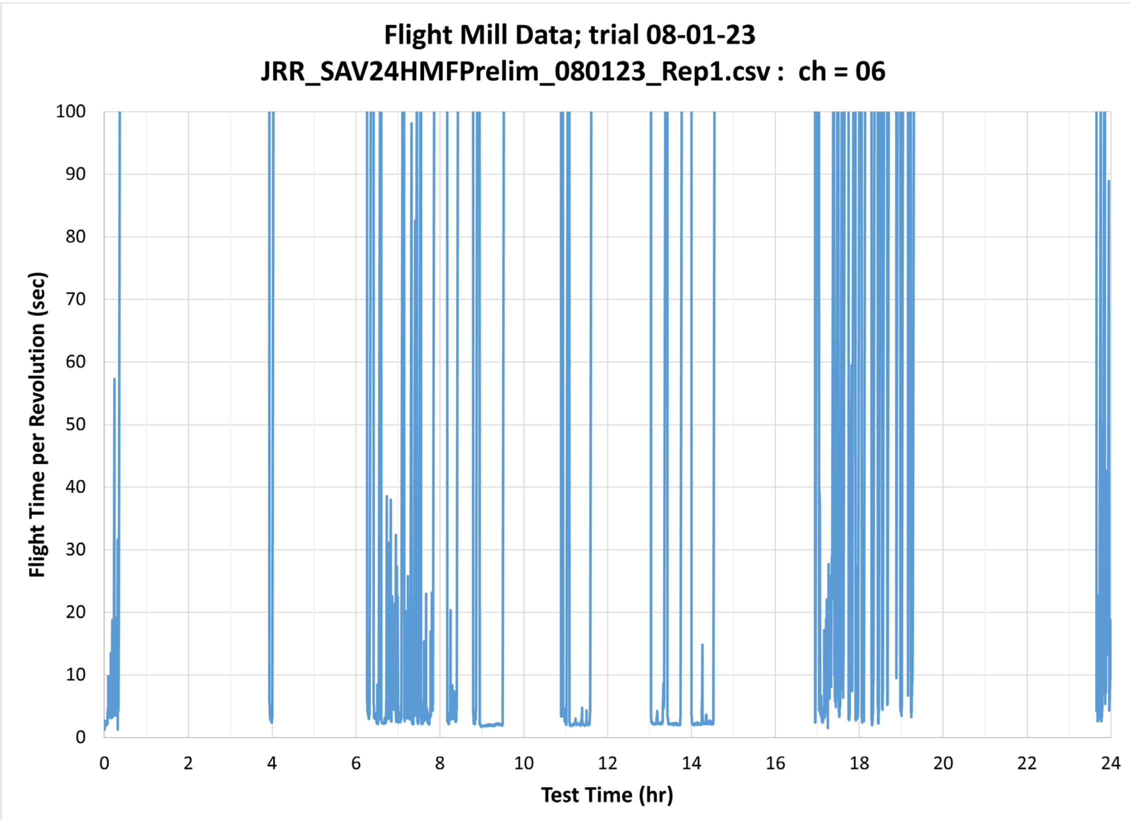
The data in the last graph, in particular, was used for setting various thresholds in the program in order to accurately detect notches.

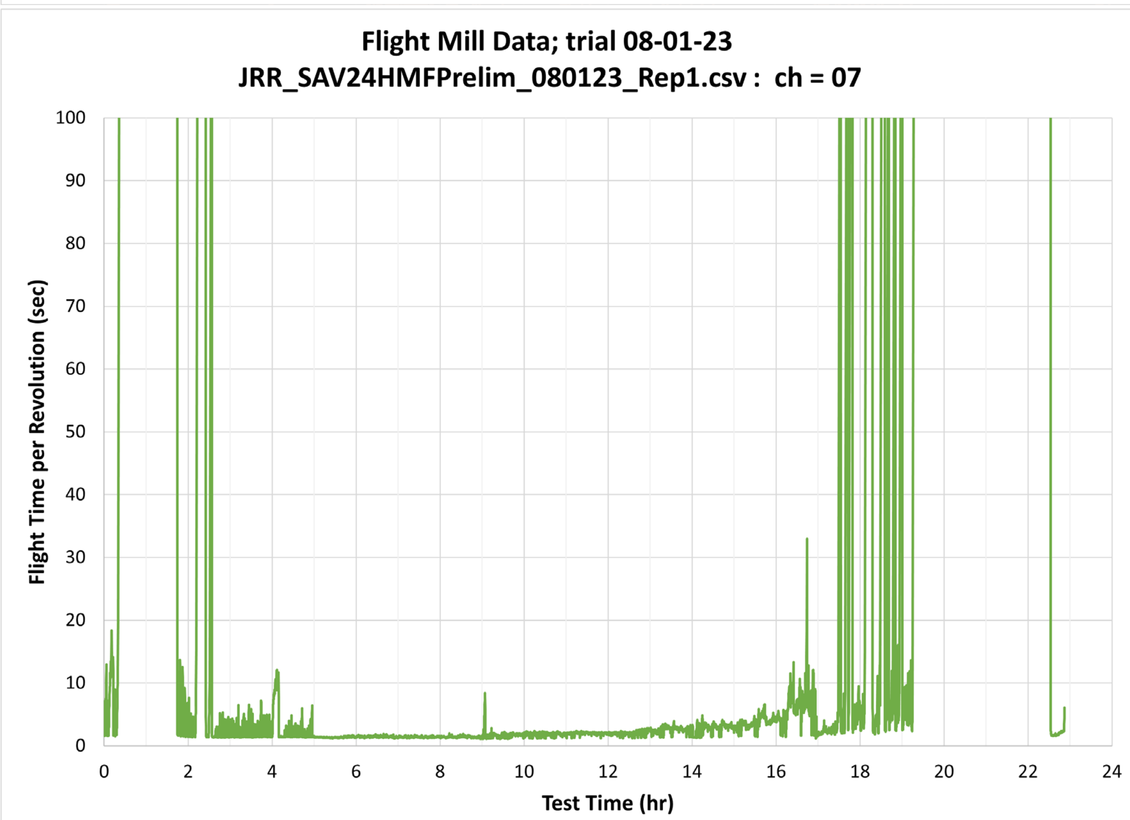

Flight Mill Data; trial 08-01-23
JRR_SAV24HMFPrelim_080123_Rep1.csv : ch = 02
Flight Time per Revolution (sec)
Test Time (hr)
0
10
20
30
40
50
60
70
80
90
100
0
2
4
6
8
10
12
14
16
18
20
22
24

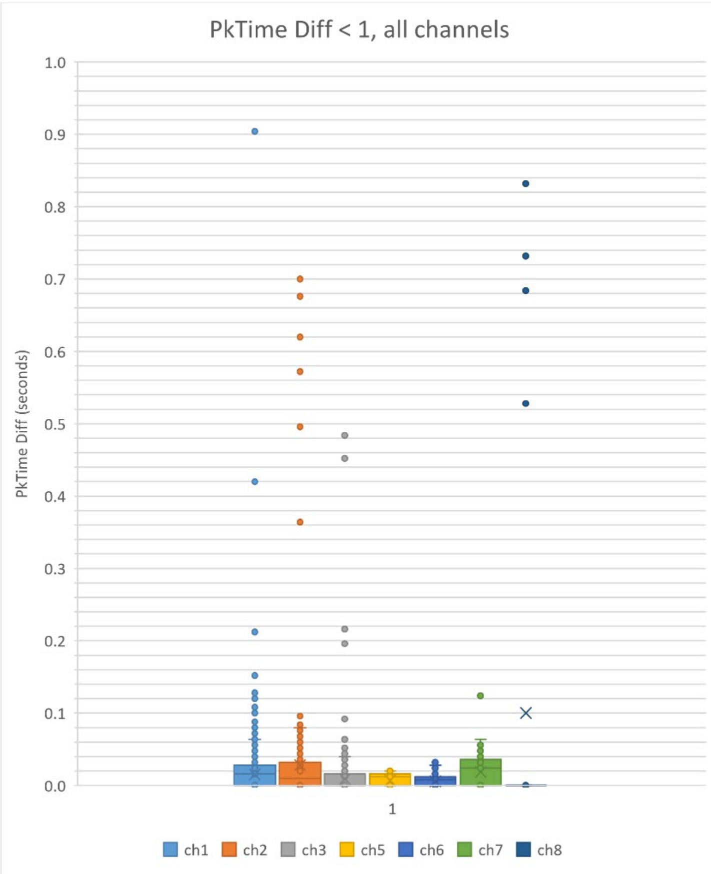

Flight Mill Data; trial 08-01-23
JRR_SAV24HMFPrelim_080123_Rep1.csv : ch = 03
Flight Time per Revolution (sec)
Test Time (hr)
0
10
20
30
40
50
60
70
80
90
100
0
2
4
6
8
10
12
14
16
18
20
22
24

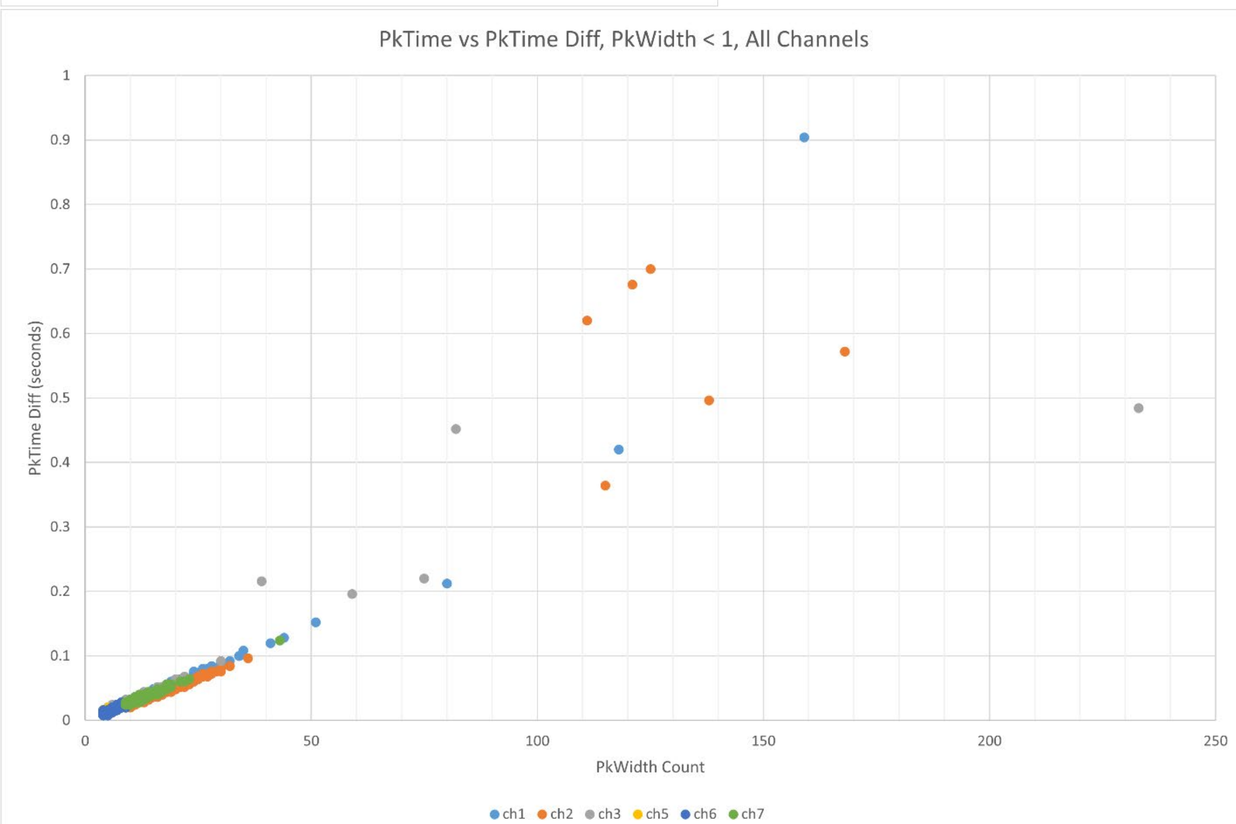

Flight Mill Data; trial 08-01-23
JRR_SAV24HMFPrelim_080123_Rep1.csv : ch = 04
Flight Time per Revolution (sec)
Test Time (hr)
0
2
4
6
8
10
12
14
16
18
20
22
24
10
20
30
40
50
60
70
80
90
100


Flight Mill Data; trial 08-01-23
JRR_SAV24HMFPrelim_080123_Rep1.csv : ch = 05
Flight Time per Revolution (sec)
Test Time (hr)
0
2
4
6
8
10
12
14
16
18
20
22
24
10
20
30
40
50
60
70
80
90
100


Flight Mill Data; trial 08-01-23
JRR_SAV24HMFPrelim_080123_Rep1.csv : ch = 06
Flight Time per Revolution (sec)
Test Time (hr)
0
10
20
30
40
50
60
70
80
90
100
0
2
4
6
8
10
12
14
16
18
20
22
24

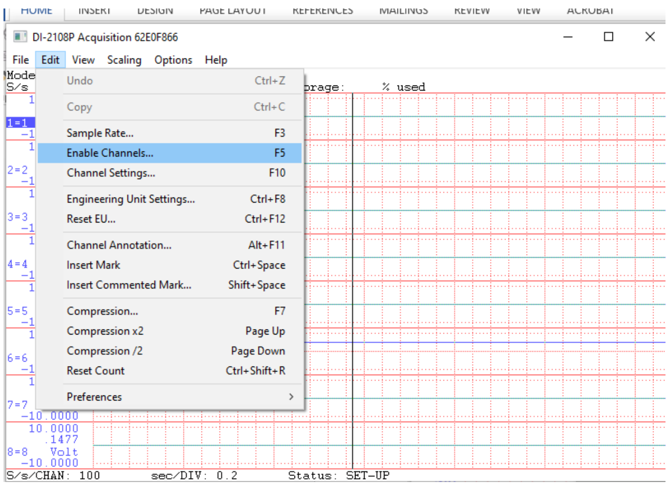

Flight Mill Data; trial 08-01-23
JRR_SAV24HMFPrelim_080123_Rep1.csv : ch = 07
Flight Time per Revolution (sec)
Test Time (hr)
0
10
20
30
40
50
60
70
80
90
100
0
2
4
6
8
10
12
14
16
18
20
22
24


PkTime Diff < 1, all channels
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
PkTime Diff (seconds)
1
ch1 ch2 ch3 ch5 ch6 ch7 ch8

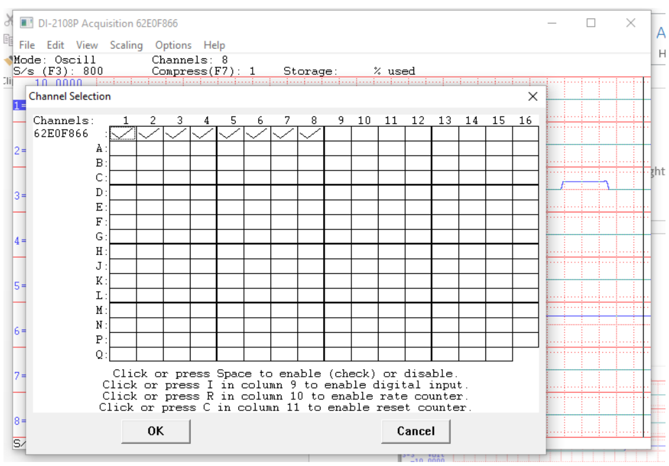

PkTime vs PkTime Diff, PkWidth < 1, All Channels
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
PkTime Diff (seconds)
0 50 100 150 200 250
PkWidth Count
ch1 ch2 ch3 ch5 ch6 ch7

Appendices

Select Number of Channels to Record

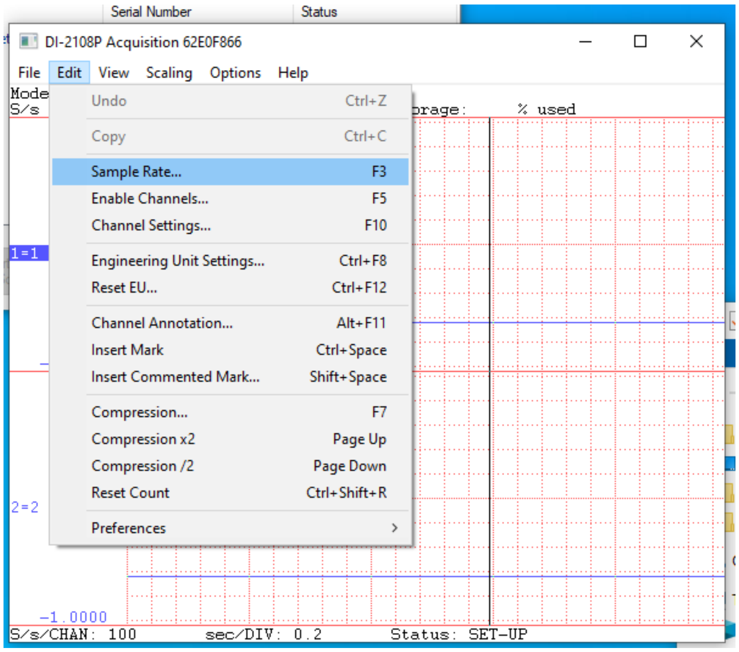


To change the number of channels being recorded, go to edit, enable channels, or press F5.

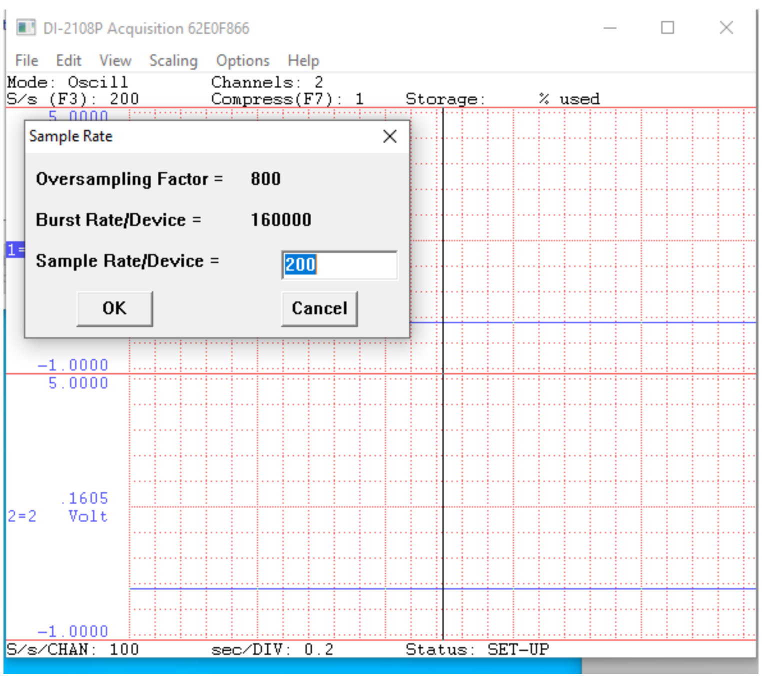


The checks across the top show the channels being recorded. Unchecking the column turns it off, i.e., data for that sensor won't be recorded.

Set Sample Rate

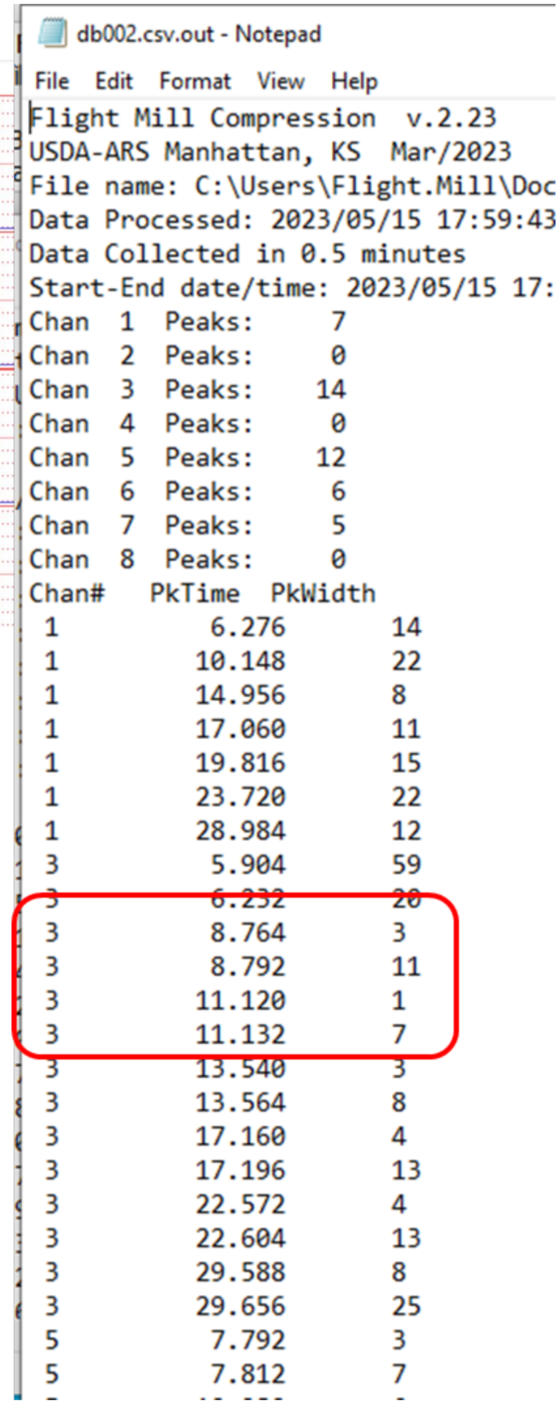


Sample rate is the number of times per second (hertz) that the value of the sensor is checked. The faster the insect flies the higher the sample rate needs to be. In practical terms, the higher the sample rate the wider the peak will be.



One problem with raising the sample rate is that it makes the data files much larger. But with today's storage devices that isn't the concern it once was.

Potential of notched flag vs solid flag


db002.csv.out - Notepad

File Edit Format View Help

```
Flight Mill Compression  v.2.23
USDA-ARS Manhattan, KS  Mar/2023
File name: C:\Users\Flight.Mill\Doc
Data Processed: 2023/05/15 17:59:43
Data Collected in 0.5 minutes
Start-End date/time: 2023/05/15 17:
Chan  1  Peaks:    7
Chan  2  Peaks:    0
Chan  3  Peaks:   14
Chan  4  Peaks:    0
Chan  5  Peaks:   12
Chan  6  Peaks:    6
Chan  7  Peaks:    5
Chan  8  Peaks:    0
Chan#   PkTime  PkWidth
 1        6.276       14
 1       10.148       22
 1       14.956       8
 1       17.060       11
 1       19.816       15
 1       23.720       22
 1       28.984       12
 3        5.904       59
 3        6.232       20
 3        8.764       3
 3        8.792       11
 3       11.120       1
 3       11.132       7
 3       13.540       3
 3       13.564       8
 3       17.160       4
 3       17.196       13
 3       22.572       4
 3       22.604       13
 3       29.588       8
 3       29.656       25
 5        7.792       3
 5        7.812       7
```

Early Software Development for file compression

Bill Rust (Feb 2023) did the early development of the compression software based on python examples provided.

At this point, the data should be analyzed by

1. Wrj_Jan2023_peaks_dat.py The previous filename was "standardize_peaks.py"
2. Wrj_Jan2023_flight_dat.py The previous filename was "analyze_flight.py."

Both are python language programs as indicated by the '*.py' extension. In order to run these programs, the Python interpreter needs to be ins. led on the computer and it was installed on the Flight Mill computer.

The 2 programs are currently set to process data from only 2 channels. The following snippet shows the top lines of the data file. Because we are using the data logger electronics, rather than the Arduino, the data file contains 3 header lines. These header lines are ignored with the modified version of the software.

<snippet of data file, top 15 lines>

Start a command prompt window. This can be done by entering "command" in the Window's task bar and selecting command prompt application.

Enter the program name at the prompt. A second line will display which asks for the filename to store the results.

<snippet of command prompt window with program name enters and filename.

To run these programs, first find a python programmer to modify the programs for the number of channels being analyzed, file locations, parameters of the flight mill, etc. After the modifications have been made, running them is simple.

First, open a command prompt, change the directory to the one containing the programs and type "python standardize_peaks.py". The program asks for the file to be processed. The program then goes through the data, normalizing it, e.g., changing readings below a threshold it has determined, to 0 and readings above to 1. It writes that data out into a file with the same name as the input file with ".out" appended to the file name.

The second program then takes the processed data and analyzes it. To run the second program type "python analyze_flight.py" in the same command prompt used by the previous program. Enter the new file name with ".out" appended. This program takes the data and finds the number of "bouts" that the insect flew. A bout is defined as a period of continuous flying. Since we don't know when the insect actually is stopped, when there is a gap in the recorded peaks of an arbitrary number of seconds, it is assumed that the insect stopped, and the current bout is ended. At the next peak, a new bout is started. Twenty seconds is the value for starting a new bout in the current code but can be changed by the python programmer. The program produces a list of bouts. The bouts are summarized by being put into buckets. The buckets are, currently, 1-5 minutes (a real bout of <1 minute is virtually impossible to occur so is omitted from the statistics), 5-15 minutes, 15-90 minutes, 1.5-4 hours, and 4+ hours. The program produces a summary of how many bouts are in each bucket. Changing the buckets requires a fair amount of code modification. The program also calculates the average speed of the insect for all bouts. Note that the list of bouts is not saved, just the summary of the bouts.

This document does not describe the modifications needed for the programs. To a programmer who knows python, the mods are trivial. To someone who does not know python (at a basic level) there are too many changes and things that could go wrong to document it all here.

After the python software is installed, the command prompt window needs to be started from the Window's task bar by typing "command prompt". Type "python" at a command prompt. If the program is installed type "exit()" to get out of it. If it is not installed, install it (good luck).

To run these programs, first find a python programmer to modify the programs for the number of channels being analyzed, file locations, parameters of the flight mill, etc. After the modifications have been made, running them is simple.

Continuing Development:

From there, Bill Rust rewrote the Python software as a Java CLI program. This program made it easier to adapt the program without rewriting the code, through the user of command line arguments.

After Bill Rust created an alpha version of the java program as a CLI, Nicholas Sixbury was given the project to make a GUI for the program and further polish the features while communicating with the team at the USDA that would be using the software. The GUI program described under step 3 is the one developed by Nicholas Sixbury on top of Bill Rust's CLI program.