midas_to_phiat Documentation

Sam Porter (wporter@triumf.ca)
University of British Columbia
TRIUMF, on behalf of the TITAN Collaboration

Overview

- **Language:** Python 3
- midas_to_phiat.py converts MIDAS files output by the MPET DAQ to readable .csv files for input into PhIAT.m
- **Input:** A directory address (i.e. address to a folder) of MIDAS Files (ex: /Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_CAENOffset)
- **Output:**
  - .csv files containing position, time-of-flight and TDC trigger information
  - .csv files containing approximate count rates
  - One .csv file containing an address list of converted MIDAS files, their accumulations times (Taccs) and reference files assignments

BEFORE YOU CONTINUE! -- titan_data

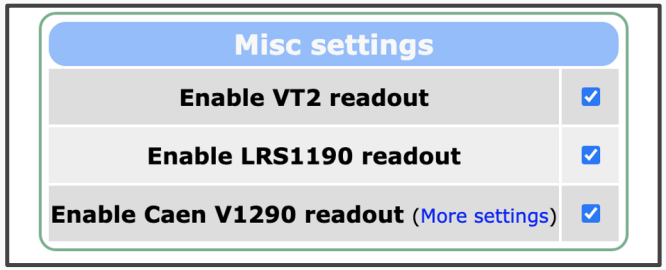
- midas_to_phiat.py requires you download the **titan_data package**
  - This can be downloaded here: https://bitbucket.org/ttriumfdaq/titan_data/src/master/

- On Line 20: Change the address in sys.path.append from '/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/titan_data' to the local address of titan_data on your computer (i.e. '/local/address/to/titan_data')

```
sys.path.append('/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/titan_data') # Address of titan_data module. NEW USERS NEED TO CHANGE THIS! #
```

A Few Other Initial Settings...

- **all_one_file:** If TRUE, resulting output data .csv files correspond to each MIDAS file
  - If FALSE, output .csv files correspond to each **loop** within in each MIDAS file (see TITAN DAQ documentation)
  - **Default is TRUE** for PI-ICR analysis
- **CAEN** and **VT2:** Set TRUE whichever TDC is being used to read out data; this is indicated in the MPET DAQ under Scan
  - **Default is CAEN = TRUE**
  - Note: Cannot both be true for data converting in script -- this will trigger an error




```
all_one_file = True # If TRUE, entire .mid becomes one .csv. If FALSE, e
i.e. scanned over voltages between events, change to False)
CAEN = True # If True, output data files are only CAEN TDC data
VT2 = False # If True, output data files are only VT2 TDC + LRS data
```

A Few Other Initial Settings...for Test Data

- **testing:** If you are using test data from MIDAS that **DOES NOT** include a reference file (i.e. a Tacc of 0), set to TRUE
  - Automatically assigns the last file a Tacc of 0, as needed to run through the script
- Otherwise, set to FALSE
  - This is the default for normal data

Getting Started

- midas_to_phiat.py is automatically run as part of PhIAT.m (see PhIAT.m documentation)
  - However, it can be run on its own as well:

```
# NOTE: To test run this script alone: python3 midas_to_phiat.py /path/to/directory/of/files
```

- The **input** is an address to a directory of MIDAS files (.mid or .mid.lz4)
  - The files in this directory are gathered together via **glob**

Data

- For every event (i.e. ion hit on the detector), four pieces of data are relevant:
  - **X Position** [mm]: X coordinate of ion hit on detector (i.e. delay line anode)
  - **Y Position** [mm]: Y coordinate of ion hit on detector (i.e. delay line anode)
  - **Time-of-Flight** [s]: Time-of-flight between ejection from the trap to ion hit on MCP
  - **Trigger** [int]: Number of trigger an ion hit came during (starting from 0 at file's beginning) -- CAEN 25ps ONLY
  - **Timestamp** [s]: Timestamp of event relative to initial trigger time -- VT2 ONLY

Data -- CAEN 25ps TDC

- **X/Y Position:** Determined via timing pulses in delay line anode
  - Correspond to **caen_tdc_parsed.pos_x_mm** (or pos_y_mm) in **../titan_data/mpet/__init__.py**
- **ToF:** Time between trap ejection and ion contact with MCP
  - Corresponds to **caen_tdc_parsed.mcp_tof_secs**
- **Trigger:** Trigger number at which an ion event occurred
  - Corresponds to **caen_tdc_parsed.trigger_count**

Data -- VT2 TDC

- **X/Y Position:** Determined via LeCroy 1190 Position Analyzer
  - Correspond to **pos_data.x** (or y) in **../titan_data/mpet/__init__.py**
  - **NOTE: The LC1190 DOES NOT give reliable position data...**
- **ToF:** Time between trap ejection and ion contact with MCP
  - Corresponds to **vt2_data.tof_secs**
    - **NOTE:** Since this is raw data (i.e. each ion hit does not yet correspond to one event), need to specify a **channel_ids** to get just one trigger value per ion hit
- **Timestamp:** Timestamp of event relative to initial trigger time
  - Corresponds to **vt2_data.time_secs**
    - **NOTE:** Since this is raw data (i.e. each ion hit does not yet correspond to one event), need to specify a **channel_ids** to get just one trigger value per ion hit

```
if tdc.channel_ids == [0]:
    event_tof_data_vt2.append(tdc.tof_secs) # ToF from trap ejection to initial MCP contact (1st ToF)
    event_timestamp_data_vt2.append(tdc.time_secs) # Timestamp corredsponding to initial MCP contact
```

Ion-Ion Interaction Data Cuts

- We want to minimize any ion-ion interaction effects in the trap on results, so we discard any data where **> N** total ions were in the trap together
  - N corresponds to **ion_ion_cut_min**
- In practice, we throw away data with **> N** events in the **same trigger window**
- **ONLY** works with CAEN 25ps TDC Data

Determining Ion Rates

- We want to determine the rate at which ion events are occurring
- We determine a gating time *t* (**gating_rate** [ms]) and determine the number of events that occur within each gating time window across the file
  - i.e. events between 0-*t*, events between *t*-2*t*, etc.
- We approximate the *global event time* for each event as the total trap time (**trap_time**) times the trigger number of that event
- We record the *approximate global gate time*, the *number of events*, and the *rate* of those events (num_events/*t*) for each gating time window

Ion Rate .csv

- A .csv file is printed with the recorded ion rate data:
  - Column 1: Approximate Global Gate Time [ms]
    - (gating window number)*t
  - Column 2: Number of Events in each Window [int]
  - Column 3: Rate of Incoming Events in each Window [events/ms]
    - num_events/t
- Saved as **original_MIDAS_filename.mid_ion_rate.csv**

	A	B	C
1	50	0	0
2	100	0	0
3	150	0	0
4	200	2	0.04
5	250	0	0
6	300	0	0
7	350	2	0.04
8	400	0	0
9	450	0	0
10	500	0	0
11	550	2	0.04
12	600	0	0
13	650	0	0
14	700	2	0.04
15	750	0	0
16	800	0	0
17	850	0	0
18	900	2	0.04
19	950	0	0
20	1000	0	0

Main Data .csv

- A .csv is printed with the data described on slides 7-9
  - Column 1: X Position [mm]
  - Column 2: Y Position [mm]
  - Column 3: Time-of-Flight [s]
  - Column 4: Trigger Number [int] (CAEN) OR Timestamp [s] (VT2)
- Saved as **original_MIDAS_filename.mid_.csv**
- Main Data and Ion Rate CSVs are created for each MIDAS file in the input directory

	A	B	C	D
1	1.62760417	1.62760417	2.44E-05	0
2	3.25520833	3.25520833	4.65E-05	0
3	1.62760417	1.62760417	2.44E-05	1
4	3.25520833	3.25520833	4.65E-05	1
5	1.62760417	1.62760417	2.44E-05	2
6	3.25520833	3.25520833	4.65E-05	2
7	1.62760417	1.62760417	2.44E-05	3
8	3.25520833	3.25520833	4.65E-05	3
9	1.62760417	1.62760417	2.44E-05	4
10	3.25520833	3.25520833	4.65E-05	4
11	1.62760417	1.62760417	2.44E-05	5
12	3.25520833	3.25520833	4.65E-05	5
13	1.62760417	1.62760417	2.44E-05	6
14	3.25520833	3.25520833	4.65E-05	6
15	1.62760417	1.62760417	2.44E-05	7
16	3.25520833	3.25520833	4.65E-05	7
17	1.62760417	1.62760417	2.44E-05	8
18	3.25520833	3.25520833	4.65E-05	8
19	1.62760417	1.62760417	2.44E-05	9
20	3.25520833	3.25520833	4.65E-05	9
21	1.62760417	1.62760417	2.44E-05	10
22	3.25520833	3.25520833	4.65E-05	10
23	1.62760417	1.62760417	2.44E-05	11
24	3.25520833	3.25520833	4.65E-05	11

Pairing Final and Reference Files

- For each final file (Tacc > 0), we need to pair it with the reference file (Tacc = 0) *closest to it in time*
- We determine the midpoint time of each file (i.e. file_start + (file_end - file_start)/2)
  - file_start is **event_time[0]** and file_end is **event_time[-1]** from titan_data/mpet/__init__.py
- We find the difference between the midpoint time of the final file and all reference files, and pair the actual file with the reference file of *smallest* difference

```
for ref_time in ref_time_list:
    diff = np.abs(ref_time - time)
    diff_list.append(diff)

min_index = diff_list.index(min(diff_list)) # Finding the number of ref file closest in time to each actual file
mindex_list.append(min_index+1)
```

File List .csv

- A .csv is printed with:
  - Column 1: Addresses of Main File CSVs
  - Column 2: Accumulation Times of each File [s]
  - Column 3: Reference File Assignment for each Final File
    - If file is a reference file, this is blank
    - Numbering of associated references correspondings to ordering of references in first column
- Saved as **inputDirectoryName_firstTaccFile_List.csv**
  - i.e. Test_CAENOffset_0.03File_List.csv
  - Saved OUTSIDE the input directory

File List .csv Example

	A	B	C
1	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run00103.mid_.csv	0.03	1
2	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001032.mid_.csv	0.03	1
3	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001033.mid_.csv	0.03	1
4	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001034.mid_.csv	0.03	1
5	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001035.mid_.csv	0.03	1
6	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001036.mid_.csv	0.03	1
7	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001037.mid_.csv	0.03	1
8	/Users/wsporter/Documents/Physics_Research/TITAN/PIICR_Analysis/Testing/Test_Set1/run001038.mid_.csv	0	
9			

- All reference files will come after all final files

- More on **titan_data** and the primary script referenced in this code (**../mpet/__init__.py**) can be found:
  - https://bitbucket.org/ttriumfdaq/titan_data/src/master/
  - TITAN DAQ documentation
- Please contact Sam Porter (wporter@triumf.ca) if you have any questions