DEEP UNDERGROUND
NEUTRINO EXPERIMENT

# ND DATA SELECTION STATUS

Alexander Booth
ND US DAQ
September 28, 2021

Queen Mary
University of London

# Overview

- All v2.8 DS work moved to a v2.10 development release of DUNE DAQ.
- Non-empty fragments from PACMAN output to hdf5 based on readout receiving a fake trigger from HSI.
- Can (somewhat) read output hdf5.

```
Path                          : TriggerRecord00008/TriggerRecordHeader
Size                          : (88, 1)
Data type                     : int8
Magic word                    : 0x33334444
Version                       : 2
Trigger number                : 8
Trigger timestamp             : 81638143650313600 (2021-09-27 12:14:33.006272)
No. of requested components   : 1
Run Number                    : 0
Error bits                    : 0
Trigger type                  : 1
Sequence number               : 0
Max sequence num              : 0
--------------------------------------------------------------------------------
Path                          : TriggerRecord00008/NDLArTPC/Region000/Element00
Size                          : (1048656, 1)
Data type                     : int8
Magic word                    : 0x11112222
Version                       : 3
Frag Size                     : 1048656
Trig number                   : 8
Trig timestamp                : 81638143650313600 (2021-09-27 12:14:33.006272)
Window begin                  : 81638143647813600 (2021-09-27 12:14:32.956272)
Window end                    : 81638143652813600 (2021-09-27 12:14:33.056272)
Run number                    : 0
Error bits                    : 0
Fragment type                 : 3
Sequence number               : 0
GeoID type                    : NDLArTPC
GeoID region                  : 0
GeoID element                 : 0
```

- New: 4 process setup now “properly” integrated with minidaq app.

```
python -m minidaqapp.nanorc.mdapp_multiru_gen --host-ru localhost -o . --number-of-data-producers 1 --frontend-type pacman
--trigger-window-before-ticks 2500000 --trigger-window-after-ticks 2500000 --trigger-rate-hz 1.0 --enable-raw-recording
mdapp_4proc_pacman_1Hz_pt1second_mode2
```

- Will require PRs in minidaqapp, dfmodules, readout.

# In-Out Tests

- Feed in known data, make sure that data is selected, make sure it is saved to hdf5.

- Focussing on known input.

```
usage: pacman-generator.py [-h] --input_file INPUT_FILE [--mode MODE] [--n_file_evals N_FILE_EVALS] [--n_pacman N_PACMAN]

optional arguments:
  -h, --help            show this help message and exit
  --input_file INPUT_FILE
                        Input h5 file.
  --mode MODE           Running mode, see function pacman.
  --n_file_evals N_FILE_EVALS
                        Number of times the input file is looped through.
  --n_pacman N_PACMAN   Number of PACMAN cards.
```

```
for n in range(n_file_evals):
    for i in word_lists:
        #data_socket.send(b"", zmq.SNDMORE)
        data_socket.send(pacman_msg_format.format_msg('DATA',i))
        print(pacman_msg_format.parse_msg(pacman_msg_format.format_msg('DATA',i)))
        message_count += 1
        print("Total messages sent:",message_count)
        if mode == 1:
            # 1 single message.
            break;
        elif mode == 2:
            # 10 messages spaced by 1 second each for each loop of file.
            next_sleep = 1;
            if message_count % 10 == 0:
                time.sleep(next_sleep);
                break;
        elif mode == 3:
            # 50 messages sent in groups of 5 at intervals of 1 second.
            if message_count % 50 == 0:
                time.sleep(next_sleep);
                break;
            if message_count % 5 == 0:
                next_sleep = 1;
            else: continue;
        else:
            next_sleep = random.randrange(1,3)
            if message_count != len(word_lists)*n_file_evals:
                print("Next message in: %ds" %(next_sleep))

        time.sleep(next_sleep)
```