

ProtoDUNE-II DAQ Expert Training

Wesley Ketchum

13 June 2024

Wes's Rules of Being a DAQ Expert

Wes's Rules of Being a DAQ Expert

1. DO. NOT. PANIC.

Wes's Rules of Being a DAQ Expert

1. DO. NOT. PANIC.
2. Be kind to everyone, *especially* the shifters.

Wes's Rules of Being a DAQ Expert

1. DO. NOT. PANIC.
2. Be kind to everyone, *especially* the shifters.
3. A - Always
B - Be
C - Collecting
D - Data
(except when you can't)

Wes's Rules of Being a DAQ Expert

1. DO. NOT. PANIC.
2. Be kind to everyone, *especially* the shifters.
3. A - Always
B - Be
C - Collecting
D - Data
(except when you can't)
4. Write it in the elog

Monitoring / trouble-shooting

DAQ overview super briefly

- Data ‘continuously’ flows from detector electronics to DAQ readout servers
 - Data ‘frames’ are timestamped with DTS (DUNE Timing System) timestamps, which are 16 ns ticks since epoch
- DAQ applications place data in (“latency”) buffers
 - Trigger/Timing data (TPs, TAs, TCs, HSIFrames) are created in separate/parallel streams, but are also timestamped with DTS and also ‘stream’ through the DAQ into latency buffers
- TriggerCandidates are used to create TriggerDecisions
 - Trigger decisions include what components to read out (all) and data window start and end times (3 ms total, 250 us before TC time and 2750 us after)
- TDs are forwarded to the DataFlowOrchestrator (DFO) to determine which dataflow application is responsible for that trigger record
- Dataflow applications issue requests for data in the readout window to all the sources of data in DAQ applications
- Data is pulled out of the latency buffer, packaged into a fragment, and sent to trigger record builder in dataflow application
 - And then data written to disk (HDF5 format)
- Special note: TPWriter application listens to ‘published’ sets of TriggerPrimitives and writes those (not the data request mechanism described above)

Some things that means

- The DAQ can “run” without sending out triggers
 - Should see data flowing through readout applications, triggers inputs and even trigger decisions produced (but ignored) until we “enable_triggers” in the DAQ
- It is very important to have good timestamps on all data, as that’s how we decide what data to keep on request
- In theory, we should only issue requests for data if we have capacity in dataflow applications to receive it
 - Trigger inhibit mechanism communicated through DFO
- Sources of data may not have data to respond to requests
 - Could be data was never there (e.g. no trigger data in that readout window)
 - Could be data was there, but no longer in latency buffer
 - Could be some other issue

Example

- Warning message received in recent run:

```
SourceID[subsystem: Detector_Readout id: 410]  
Trigger Matching result with empty fragment: TS  
match result for SourceID[subsystem:  
Detector_Readout id: 410]: Trigger/sequence  
number=114.0  
Oldest stored TS=107388637460944907  
Start of window TS=107388637425134132  
End of window TS=107388637425321632  
Estimated newest stored TS=107388637939509259  
Requestor=fragment_queue -- 100 similar messages  
suppressed, last occurrence was at 2024-Jun-12  
20:50:07,032905
```

You can see that the start and end of window is behind the oldest stored timestamp → we return an 'empty' fragment

SourceID

- Data sources are assigned unique SourceIDs in the DAQ
 - Subsystem type (DetectorReadout, HwSignalsInterface, Trigger, TRBuilder)
 - ID (just a number)
- Source IDs are specified as part of configuration
- Source IDs are not guaranteed to always correspond to the same component, but in practice this is often true
 - Source IDs are part of detector readout map config for DetectorReadout
 - 100s for APA1 WIBs, 200s for APA2 WIBs, etc. 8 streams per WIB
 - 1-14 for the DAPHNE
 - Source IDs in trigger typically count starting from 0, from TP sources through TA sources to TC sources

Example

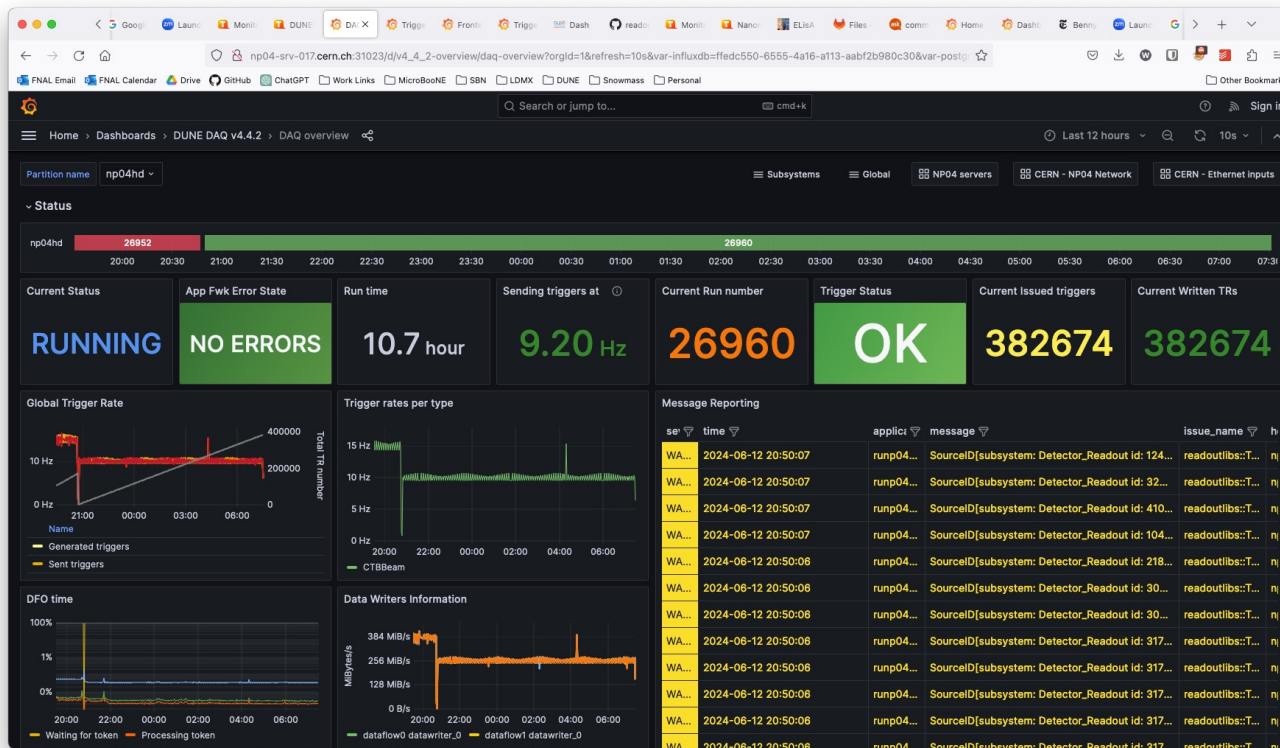
- Warning message received in recent run:

```
SourceID[subsystem: Detector_Readout id: 410]  
Trigger Matching result with empty fragment: TS  
match result for SourceID[subsystem:  
Detector_Readout id: 410]: Trigger/sequence  
number=114.0  
Oldest stored TS=107388637460944907  
Start of window TS=107388637425134132  
End of window TS=107388637425321632  
Estimated newest stored TS=107388637939509259  
Requestor=fragment_queue -- 100 similar messages  
suppressed, last occurrence was at 2024-Jun-12  
20:50:07,032905
```

You can see this if from SourceID Detector_Readout 410 → APA4, WIB 402, 3rd stream

Monitoring page

- Grafana page available within CERN network: <http://np04-srv-017.cern.ch:31023>
 - Need to setup a tunnel to it if you're outside CERN network
 - Remember to check partition on left, and the time / update on the right



Message reporting

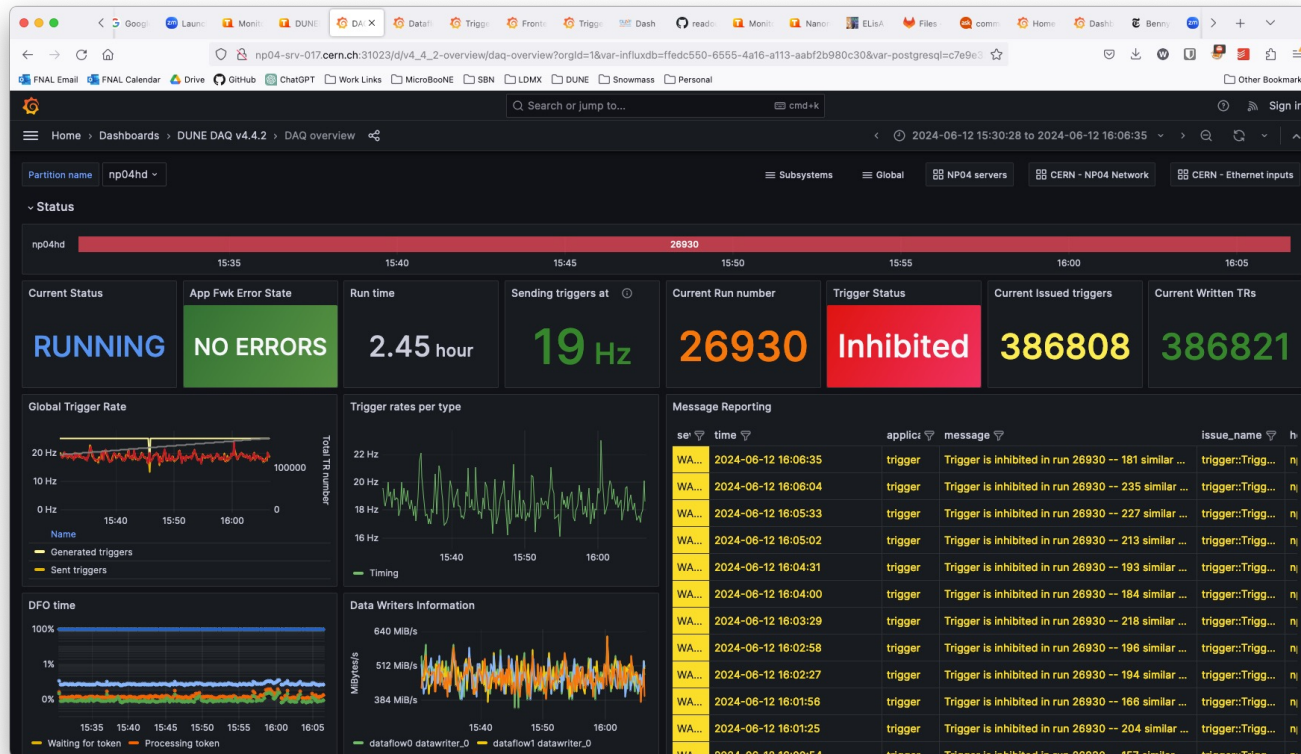
- The message reporting is incredibly important
 - You should treat any message reported as real, and pay attention to the severity
 - though not all ERRORS are fatal
 - Note the times, and the applications, and if similar messages are being suppressed
 - You can filter these to investigate more
 - And turn off updating to make it easier to copy / paste
 - Hopefully messages are meaningful!
- Expect shifters to let you know when they see something
 - ERRORS and WARNINGS should be checked on: don't assume a WARNING is OK unless we know it is
 - Typically, when things are really bad, we get lots of errors and warnings repeatedly → may be a good reason to stop and restart a run

Additional logging

- The message reporting doesn't show everything
 - E.g. something may fail to configure, and not leave much or anything in the messages
- Log files are written to the /log directory on the server each application runs on
 - Check nano04rc or 'Applications' monitoring window to see what application is running where in order to go in and look at logs

Trigger inhibits

- Should be clear on the 'Trigger Status' when we are inhibited
 - → we cannot keep up with the rate at which we are generating triggers
 - You should see a discrepancy in the generated triggers vs. written triggers
 - What do do? Ideally we would lower the trigger rate. But can handle this temporarily.



Other important monitoring pages

- Subsystem pages
 - Readout pages are particularly useful to see timestamps of data in buffers
 - Frontend Ethernet and Frontend FELIX readout to monitor data coming into the readout applications
 - Trigger Primitives: see TriggerPrimitive rates, check for hot channels / hot data sources
- Global
 - Applications shows all applications running → especially useful if you are having troubles starting applications due to ports in use, and need to check if there are other applications running
 - Generally we can solve that problem by picking a different partition, but usually want to try to find and close applications that are running that should not be
- You should look around!

Various TroubleShooting

- We do need to have a more consolidated trouble-shooting document
 - Here's a blank starter ...
<https://twiki.cern.ch/twiki/bin/edit/CENF/DAQExpertTroubleshooting?topicparent=CENF.DUNEDAQProto;nowysiwyg=1>
 - Maybe something we can start to fill out today
- There is some trouble-shooting information available on the wiki
 - <https://twiki.cern.ch/twiki/bin/view/CENF/DUNEDAQProto>
 - <https://twiki.cern.ch/twiki/bin/view/CENF/NanorcRunControl> has some FELIX info near the bottom
- I've also found searching in slack for specific errors to often be very useful
 - Though slack shouldn't replace documenting in elog
- Personal experience: do *read* messages being reported. Oftentimes they can include info on what the problem really is and how to fix it

Operations / running the DAQ

Logging in

- We take data as the np04daq user, and we do it from np04-srv-024
 - There is nothing *special* about np04-srv-024 → we can use a different server if need be
- Access as np04daq is controlled via Kerberos via `~np04daq/.k5login` file
 - List of valid Kerberos tickets there
- You should make sure you can login as np04daq user!
- If a shifter cannot login to np04-srv-024, make sure their username is in the k5login, and that they have a valid Kerberos ticket

At login...

- When logging in as np04daq,
`~np04daq/bin/np04daq_intro.sh` is run
 - On np04-srv-024 ...
 - Lists tmux sessions
 - On all servers ...
 - Lists DAQ area soft links
- You should know this is here and how to change it, but don't rely on shifters to have seen it
 - How often we re-login in at CR may not be often
 - Easy to miss for some if they don't know to look for it

Welcome message

---- Welcome to NP04 Run Control Server! ----

See <https://twiki.cern.ch/twiki/bin/view/CENF/NanorcRunControl> for further instructions.

Link to “Basic operator instructions”

Currently running tmux sessions are:

```
50l_timing_fanout_0: 1 windows (created Tue Jan 30 16:10:54 2024)
np02_coldbox: 1 windows (created Mon Feb 26 08:57:27 2024) (attached)
np02_coldbox_timing_tlu: 1 windows (created Tue Jan 30 16:05:35 2024)
np02_timing_fanout_2: 1 windows (created Tue Jan 30 15:59:56 2024)
np02_vd: 1 windows (created Fri Feb 23 11:11:13 2024)
np04_hd: 1 windows (created Fri Feb 23 11:28:49 2024)
np04_pds: 1 windows (created Sun Feb 11 18:16:23 2024)
np04_timing_fanout_1: 1 windows (created Tue Jan 30 15:55:25 2024)
```

List of tmux sessions

To attach to a session, do: `'tmux a -t <session_name>'`

If you are taking a run, please attach to the appropriate session and run from that environment.

If you need to do other other DAQ work,

The DAQ NP04_HD stable area is DAQ_NP04_HD_AREA (/nfs/sw/dunedaq/dunedaq-fddaq-v4.3.0-rc3)

The DAQ NP04_HD dev/integration area is DAQ_NP04_HD_DEV_AREA (/nfs/sw/dunedaq/dunedaq-fddaq-v4.3.0-rc1-dev)

Various DAQ working areas

The DAQ NP02_VD stable area is DAQ_NP02_VD_AREA (/nfs/sw/dunedaq/dunedaq-fddaq-v4.3.0-rc3-dev-wibfw)

The DAQ NP02_VD dev/integration area is DAQ_NP02_VD_DEV_AREA (/nfs/sw/dunedaq/dunedaq-fddaq-v4.3.0-rc3-dev-wibfw)

The DAQ NP02_COLDBOX area is DAQ_NP02_VD_DEV_AREA (/nfs/sw/dunedaq/dunedaq-fddaq-v4.3.0-rc3-dev-wibfw)

The DAQ DQM area is DAQ_DQM_AREA (/nfs/sw/dunedaq/dunedaq-fddaq-NFD_PROD4_240216_A9-dqm)

tmux

- We are using tmux sessions to run and control the DAQ
 - Advantages: we have been using it for a while, it's simple to setup / resetup
 - Disadvantages: maybe not the most 'shifter' friendly
- You should be unafraid of being able to do the following:
 - Attaching to and detaching from a tmux session
 - (detach via command line and "ctrl+\ d")
 - Starting a new session and killing a session by name
- tmux cheat sheet: <https://tmuxcheatsheet.com/>
 - **NOTE: "Ctrl + \ " instead of "Ctrl + b" for tmux commands as np04daq**
- You should be ready to guide shifters through attaching/detaching from tmux

tmux session names

- tmux session names are used in `setup_for_run.sh` script (more on that later) to define partition names and numbers for the run control

```
if [ "$TMUX_SESSION_NAME" == "np02_colddb" ]
then
    PARTITION_NUMBER=1
    PARTITION_NAME="np02vdcolddb"
elif [ "$TMUX_SESSION_NAME" == "np02_vd" ]
then
    PARTITION_NUMBER=2
    PARTITION_NAME="np02vd"
elif [ "$TMUX_SESSION_NAME" == "np02_vddev" ]
then
    PARTITION_NUMBER=3
    PARTITION_NAME="np02vddev"
elif [ "$TMUX_SESSION_NAME" == "np04_hd" ]
then
    PARTITION_NUMBER=4
    PARTITION_NAME="np04hd"
elif [ "$TMUX_SESSION_NAME" == "np04_hddev" ]
then
    PARTITION_NUMBER=5
    PARTITION_NAME="np04hddev"
else
    PARTITION_NUMBER=0
fi
```


DAQ areas

- DAQ common software areas are in /nfs/sw/dunedaq
 - Named “dunedaq-<version>-<optional-extension>”
 - Owned by np04daq user
- These are just ~normal DAQ areas (either static release or dev areas), so everything you know about DAQ setups apply
- Which are the ‘current’ areas in use should be linked to via the soft links from the ~np04daq area

Setting up a DAQ area

- Instructions here:
<https://twiki.cern.ch/twiki/bin/view/CENF/SetupDAQAreaEHN1>
- Basic steps (as np04daq user!)
 - Create the area
 - Remake configurations
 - (Test...)
 - Update tmux sessions and soft links
- As a DAQ expert, you should be able to do all of this
- Shifters should ~never setup a new DAQ area

Creating the area (1)

- As np04daq, go to /nfs/sw/dunedaq
- Follow instructions, or can use the “create_daq_area.sh” script
 - Should automatically detect release candidates or nightly releases and use the appropriate build type option
 - Specify versions *without* the a9 suffix
- Static vs. dev
 - Static releases ideally used for stable running
 - Dev releases should be used when we need to pull in specific updates for testing or rapid/necessary fixes before we have a release
- Naming conventions for the directories:
 - “static” areas: dunedaq-<release-name-without-a9-suffix>
 - e.g. dunedaq-fddaq-v4.4.2
 - “dev” areas: dunedaq-<release-name-without-a9-suffix>-dev
 - There may be need for additional dev areas for specific purposes
 - E.g. dunedaq-fddaq-v4.4.2-dev-crt for testing crt readout/DAQ

Creating the area (2)

- There are two setup scripts in /nfs/sw/dunedaq that should be (automatically) copied into the new area
 - setup.sh: script that does the basic setup of the area
 - setup_for_run.sh: script intended to perform basic setup and prepare for a run using nano04rc
 - Makes partition numbers/names environment variables, disables the web proxy, prints a lot of hopefully useful stuff
- The configuration repo should be (automatically) cloned into the area, and a soft link to the global_configs created
 - You will need to input your CERN credentials to pull down the np04daq-configs repo

Configurations

- Updating configurations across all various run conditions can be a pain: tools in np04daq-configs try to make that a little easier
 - <https://gitlab.cern.ch/dune-daq/online/np04daq-configs>
 - Ask for access if you don't have it
- Three main pieces:
 - Detector readout (DRO) map creation
 - Configuration parent/driver creation
 - Creation of all configurations using conf_gen commands

gen_dro.py

- Python script that creates a dro_map.json file
 - `python gen_dro.py <detector_configs> <daq_configs>
<dro_configs> comp,list output.json`

gen_dro.py

- Python script that creates a dro_map.json file

```
- python gen_dro.py <detector_configs> <daq_configs>  
  <dro_configs> comp,list output.json
```

- Inputs:

- detector_configs: folder or json file containing JSON blocs describing the detector components

```
"np04_APA1_WIB1_DAQ0":{  
  "det_id": 3,  
  "crate_id": 1,  
  "slot_id": 0,  
  "stream_ids": [0,1,2,3],  
  "tx_host": "np04-wib-101",  
  "tx_mac": "80:d3:36:00:52:10",  
  "tx_ip": "10.73.139.84"  
},  
"np04_APA1_WIB1_DAQ1":{  
  "det_id": 3,  
  "crate_id": 1,  
  "slot_id": 0,  
  "stream_ids": [64,65,66,67],  
  "tx_host": "np04-wib-101",  
  "tx_mac": "80:d3:36:00:52:11",  
  "tx_ip": "10.73.139.85"  
},  
"np04_APA1_WIB2_DAQ0":{  
  "det_id": 3,  
  "crate_id": 1,  
  "slot_id": 1,  
  "stream_ids": [0,1,2,3],  
  "tx_host": "np04-wib-102",  
  "tx_mac": "80:d3:36:00:52:12",  
  "tx_ip": "10.73.139.86"  
},
```

```
"np04_APA1_DAPHNE_TOP_DAQ0":{  
  "det_id": 2,  
  "crate_id": 1,  
  "slot_id": 4,  
  "stream_ids": [0],  
  "tx_host": "np04-daphne-002"  
},  
"np04_APA1_DAPHNE_TOP_DAQ1":{  
  "det_id": 2,  
  "crate_id": 1,  
  "slot_id": 4,  
  "stream_ids": [1],  
  "tx_host": "np04-daphne-002"  
},  
"np04_APA1_DAPHNE_TOP_DAQ2":{  
  "det_id": 2,  
  "crate_id": 1,  
  "slot_id": 4,  
  "stream_ids": [2],  
  "tx_host": "np04-daphne-002"  
},  
"np04_APA1_DAPHNE_TOP_DAQ3":{  
  "det_id": 2,  
  "crate_id": 1,  
  "slot_id": 4,  
  "stream_ids": [3],  
  "tx_host": "np04-daphne-002"  
},
```

gen_dro.py

- Python script that creates a dro_map.json file

```
- python gen_dro.py <detector_configs> <daq_configs>  
  <dro_configs> comp,list output.json
```

- Inputs:

- *detector_configs*: folder or json file containing JSON blocs describing the detector components
- *daq_configs*: same for the DAQ-side components

```
"np02-srv-001-ETH":{  
  "kind": "eth",  
  "protocol": "udp",  
  "mode": "fix_rate",  
  "rx_host": "np02-srv-001",  
  "rx_mac": "6c:fe:54:47:a1:28",  
  "rx_ip": "10.73.139.22",  
  "rx_pcie_dev": "0000:01:00.0"  
},  
"np02-srv-002-ETH":{  
  "kind": "eth",  
  "protocol": "udp",  
  "mode": "fix_rate",  
  "rx_host": "np02-srv-002",  
  "rx_mac": "6c:fe:54:47:98:20",  
  "rx_ip": "10.73.139.26",  
  "rx_pcie_dev": "0000:31:00.0"  
},  
"np02-srv-003-ETH":{  
  "kind": "eth",  
  "protocol": "udp",  
  "mode": "fix_rate",  
  "rx_host": "np02-srv-003",  
  "rx_mac": "6c:fe:54:4b:20:f1",  
  "rx_ip": "10.73.139.82",  
  "rx_pcie_dev": "0000:98:00.1"  
},  
"np04-srv-026-FLX":{  
  "kind": "flx",  
  "protocol": "half",  
  "mode": "var_rate",  
  "host": "np04-srv-026",  
  "cards": [0],  
  "slrs": [[0,1]],  
  "links": [[[0,1,2,3,4,5],[0,1,2,3,4,5]]],  
  "comment": "For DAPHNE Self-Triggered mode boards."  
},  
"np04-srv-030-FLX":{  
  "kind": "flx",  
  "protocol": "half",  
  "mode": "fix_rate",  
  "host": "np04-srv-030",  
  "cards": [0],  
  "slrs": [[0,1]],  
  "links": [[[0,1,2,3,4,5],[0,1,2,3,4,5]]],  
  "comment": "For DAPHNE Streaming mode boards."  
},  
"np04-srv-025-FLX":{  
  "kind": "flx",  
  "protocol": "half",  
  "mode": "fix_rate",  
  "host": "np04-srv-025",  
  "cards": [0],  
  "slrs": [[0,1]],  
  "links": [[[0,1,2,3,4,5],[0,1,2,3,4,5]]],  
  "comment": "For DAPHNE Streaming mode boards, colddb."}
```


gen_dro.py

- Python script that creates a dro_map.json file

```
python gen_dro.py <detector_configs> <daq_configs> <dro_configs>  
comp,list output.json
```

- Inputs:

- *detector_configs*: folder or json file containing JSON blocs describing the detector components
- *daq_configs*: same for the DAQ-side components
- *dro_configs*: JSON blobs pairing detector and daq components
 - Options to set `src_id` to avoid conflicts
 - Note FELIX breakdown by link

```
"np04_WIB_APA1": [  
  {  
    "src_id": 100,  
    "dettx_name": "np04_APA1_WIB1_DAQ0",  
    "daqrx_name": "np04-srv-028-ETH"  
  },  
  {  
    "src_id": 104,  
    "dettx_name": "np04_APA1_WIB1_DAQ1",  
    "daqrx_name": "np04-srv-028-ETH"  
  },  
  {  
    "src_id": 108,  
    "dettx_name": "np04_APA1_WIB2_DAQ0",  
    "daqrx_name": "np04-srv-028-ETH"  
  },  
  {  
    "src_id": 112,  
    "dettx_name": "np04_APA1_WIB2_DAQ1",  
    "daqrx_name": "np04-srv-028-ETH"  
  },  
  {  
    "src_id": 116,  
    "dettx_name": "np04_APA1_WIB3_DAQ0",  
    "daqrx_name": "np04-srv-028-ETH"  
  },  
  {  
    "src_id": 120,  
    "dettx_name": "np04_APA1_WIB3_DAQ1",  
    "daqrx_name": "np04-srv-028-ETH"  
  },  
],  
"np04_DAPHNE_APA1": [  
  {  
    "src_id": 1,  
    "dettx_name": "np04_APA1_DAPHNE_TOP_DAQ0",  
    "daqrx_name": "np04-srv-030-FLX_C0_S0_L0"  
  },  
  {  
    "src_id": 2,  
    "dettx_name": "np04_APA1_DAPHNE_TOP_DAQ1",  
    "daqrx_name": "np04-srv-030-FLX_C0_S0_L1"  
  },  
  {  
    "src_id": 3,  
    "dettx_name": "np04_APA1_DAPHNE_TOP_DAQ2",  
    "daqrx_name": "np04-srv-030-FLX_C0_S0_L2"  
  },  
  {  
    "src_id": 4,  
    "dettx_name": "np04_APA1_DAPHNE_TOP_DAQ3",  
    "daqrx_name": "np04-srv-030-FLX_C0_S0_L3"  
  },  
  {  
    "src_id": 5,  
    "dettx_name": "np04_APA1_DAPHNE_MID_DAQ0",  
    "daqrx_name": "np04-srv-030-FLX_C0_S0_L4"  
  },  
  {  
    "src_id": 6,  
    "dettx_name": "np04_APA1_DAPHNE_MID_DAQ1",  
    "daqrx_name": "np04-srv-030-FLX_C0_S0_L5"  
  },  
]
```

gen_dro.py

- Python script that creates a dro_map.json file

```
- python gen_dro.py <detector_configs> <daq_configs>  
  <dro_configs> comp,list output.json
```

- Inputs:

- *detector_configs*: folder or json file containing JSON blocs describing the detector components
- *daq_configs*: same for the DAQ-side components
- *dro_configs*: JSON blobs pairing detector and daq components
 - Options to set *src_id* to avoid conflicts
 - Note FELIX breakdown by link
- **comp,list**: comma separated list of named pairs from dro_configs to be included in the full output file

Recreating DRO maps

- A [recreate_dro_maps.sh script](#) does most of the various iterations of that
- Generally nothing needs to change here if the mapping of detector and DAQ components isn't changing
- Examples where you would need to make a change:
 - WIB FEMB off? Would need to find the right stream and remove it in detector_configs
 - Switch which readout server is reading an APA? Would need to update the proper dro_configs
 - Need some weird combination of detector resources? May need to add a line/few lines in the recreate_dro_maps script
- Final maps are written to DRO/dro_maps

Configuration generation

- `gen_daqconf_config.py` is a script for creating files to feed as input into `fddaq_conf_gen`
 - Makes it easier to keep track of and manage default and specialized DAQ sub-system configurations

Configuration generation

- Usage:

```
gen_daqconf_config.py generator_config.json output_config.json
```

Example...

```
gen_np04_daq_ctb_10Hz_config.json 379 B
```

```
1  {
2    "boot": { "config_name": "common"},
3    "ctb_hsi": { "config_name": "ctb_as_hsi_10Hz"},
4    "detector": { "config_name": "np04"},
5    "dataflow": { "config_base": "common_tp", "config_name": "np04_multiDF_6"},
6    "hsi": { "config_name": "ctb"},
7    "readout": { "config_name": "tp_hd"},
8    "timing": { "config_name": "common"},
9    "trigger": { "config_name": "hsi_passthrough"}
10 }
11
```

Configuration generation

- Usage:

```
gen_daqconf_config.py generator_config
```

```
gen_np04_daq_ctb_10Hz_config.json 379 B
```

```
1 {
2   "boot": { "config_name": "common"},
3   "ctb_hsi": { "config_name": "ctb_as_hsi_10Hz"},
4   "detector": { "config_name": "np04"},
5   "dataflow": { "config_base": "common_tp", "config_n":
6   "hsi": { "config_name": "ctb"},
7   "readout": { "config_name": "tp_hd"},
8   "timing": { "config_name": "common"},
9   "trigger": { "config_name": "hsi_assthrough"}
10 }
11
```

```
"common": {
  "use_ctb_hsi": false,
  "host_ctb_hsi": "np04-srv-012",
  "hlt_triggers": [
    { "id": "HLT_4",
      "description": "TEST HLT",
      "enable": true,
      "minc": "0x1",
      "mexc": "0x0",
      "prescale": "0x1"
    }
  ],
  "fake_trig_1": {
    "description": "Fake 1Hz LLT trigger",
    "enable": false,
    "fixed_freq": true,
    "beam_mode": false,
    "period": 62500000
  },
  "fake_trig_2": {
    "description": "Fake 1Hz LLT trigger",
    "enable": true,
    "fixed_freq": true,
    "beam_mode": false,
    "period": 62500000
  }
},
"ctb_as_hsi_10Hz": {
  "use_ctb_hsi": true,
  "fake_trig_2": {
    "description": "Fake 10Hz LLT trigger",
    "enable": true,
    "fixed_freq": true,
    "beam_mode": false,
    "period": 6250000
  }
},
},
```

“ctb_as_hsi_10Hz” assumes a “common” base configuration, and appends or overrides params in “ctb_as_hsi_10Hz” (all defined in ctb_hsi.configs)


Configuration generation

- Usage:

```
gen_daqconf_config.py generator_config.json output_config.json
```

```
gen_np04_daq_ctb_10Hz_config.json 379 B
```

```
1  {
2    "boot": { "config_name": "common"},
3    "ctb_hsi": { "config_name": "ctb_as_hsi_10Hz"},
4    "detector": { "config_name": "np04"},
5    "dataflow": { "config_base": "common_tp", "config_name": "np04_multiDF_6"},
6    "hsi": { "config_name": "ctb"},
7    "readout": { "config_name": "tp_hd"},
8    "timing": { "config_name": "common"},
9    "trigger": { "config_name": "hsi_passthrough"}
10 }
11
```



Can use “config_base” to use a different configuration as the “base” one (here: “common_tp” in dataflow_configs.tp, where the tpwriter is enabled), and then override to have more writer applications

“config_file” option also available to use file other than <subsystem>_configs.json

Configuration generation

- Everything needed to (re)create configurations should be in `recreate_*_configurations.sh` scripts
 - [recreate_wib_configurations.sh](#) for WIBs
 - Outputs sent to `WIB_CONFS` directory
 - `recreate_daphne_configurations.sh` for DAPHNEs
 - [recreate_np04_daq_configurations.sh](#) for FELIX, HERMES, and DAQ applications
 - Outputs routed to `FLXCARD_CONFS`, `HERMES_CONFS`, and `DAQ_CONFS/np04`
 - Goes through various DRO maps to create configurations
 - Uses proper “file-label” commands ...
 - Similar scripts for `np02_daq` and `np02_coldbox`
 - *Recent changes generate both ssh and k8s versions of configurations*
- There is poor error tracking here: **CHECK THAT CONFIGURATIONS GOT MADE**
 - Local “logs” directory can help find errors

Which hosts for which applications

- Most configs allow specifying what host to run the application on
- In many cases, we can easily move applications around
 - Trigger, dfo, wibmods, fakehsi, ...
- In other cases, we must be more careful
 - Readout applications require interface to detector electronics
 - 100G nics on np04-srv-028, 029, 021, and 022 for WIBs
 - FELIX cards on np04-srv-026, 030 for DAPHNE
 - **NOTE:** cpu_pinning files also tie readout applications to specific servers through thread naming based on source IDs
 - Dataflow applications need to have storage volumes specified for output
- When in doubt, try to stick to where applications typically run

global_configs

- We use application sets in nano04rc, and so have a “global” config that is used to define WIB, FLX_CARD, HERMES, and DAQ confs
 - Also sets “apparatus_id” for linkage to elog
- We want to try to have a reasonable naming on everything ...
 - np04_<components>_<descriptors>_<process_manager>.json

```
np04_TPC_noTPG_ctbFakeBeam_ssh.json 256 B
Blame Edit Lock Replace Delete
1 {
2   "apparatus_id": "np04_hd",
3   "np04_wib": "np04daq-configs/WIB_CONFS/np04_wib_conf",
4   "np04_hermes": "np04daq-configs/HERMES_CONFS/np04_hermes_TPC_ssh_conf",
5   "np04_daq": "np04daq-configs/DAQ_CONFS/np04/np04_daq_TPC_noTPG_ctbFakeBeam_ssh_conf"
6 }
7
```

Updating tmux sessions and soft links

- After testing that a running area is good to use, want to make it “the default” for users
- Update "DAQ_*_AREA" soft links from ~np04daq as appropriate
- Update tmux session
 - Kill and recreate tmux sessions from np04-srv-024
 - `tmux kill-session -t <session name>`
 - `tmux new -s <session name>`, then setup area for run within that tmux session (`setup_for_run.sh`)
 - Reminder: common session names are used for automatically picking up partition names and numbers
 - The area they use is *not* automatically defined

Session name	Use for ...	Area in use
np04_hd	Stable release running on NP04 (ProtoDUNE-II HD Module 0)	DAQ_NP04_HD_AREA
np02_vd	Stable release running on NP02 (ProtoDUNE-II VD Module 0)	DAQ_NP02_VD_AREA
np02_coldbox	Running for NP02 Coldbox NP02	DAQ_NP02_VD_AREA (or sometimes the DEV area)
np04_hddev	Dev/Integration release running on NP04 (ProtoDUNE-II HD Module 0)	DAQ_NP04_HD_DEV_AREA
np02_vddev	Dev/Integration release running on NP02 (ProtoDUNE-II VD Module 0)	DAQ_NP02_VD_DEV_AREA

Configuration listing

- Maintaining a list of configurations to use on the wiki
 - <https://twiki.cern.ch/twiki/bin/view/CENF/MonitoringRuns>
- Hope is that this is a ‘one-stop shop’ for shifters to be able to know what to run
 - We should tell shifters to look at this page, and guide them to pick the appropriate configuration from this list
- You should know how to edit / update this wiki!
 - See <https://twiki.cern.ch/twiki/bin/view/CENF/AccessRequirements>

Current configuration(s) to use

tmux Session	Configuration Name	Description	Grafana partition name
np04_hd	global_configs/np04/np04_TPC_noTPG_100MHz_ssh.json	NP04 TPC only readout, without TPG	np04hd
np04_hd	global_configs/np04/np04_TPC_TPG_100MHz_ssh.json	NP04 TPC only readout, with TPG	np04hd
np04_hd	global_configs/np04/np04_TPC_noTPG_10Hz_ssh.json	NP04 TPC only readout, without TPG, high trigger rate	np04hd
np04_hd	global_configs/np04/np04_TPC_noTPG_ctbFakeBeam_ssh.json	NP04 TPC only readout, CTB-generated triggers using beam spill structure, no TPG	np04hd
np04_hd	global_configs/np04/np04_TPC_TPG_ADCSimple_100MHz_ssh.json	NP04 TPC only readout, software-based trigger for ground blips	np04hd

nano04rc

- We use a 'np04' flavor of nanorc called nano04rc
 - Links to the elog, CERN microservices, etc.
- setup_for_run.sh has examples of the command to run
 - `nano04rc --partition-number $PARTITION_NUMBER --timeout 120 global_configs/<config_name>.json <username> $PARTITION_NAME`
 - An alias rc04 exists for this:
 - `rc04 global_configs/<config_name>.json <username>`
 - When we run with k8s, there's an additional option required
 - `--pm k8s://np04-srv-016:31000` (specified in rc04_k8s alias)
- **Reminder:** np04daq user should be used to run nano04rc, but shifter's CERN username should be used in command here
 - Connects shifter account to the elog
 - `change_user` to change username from within run control

Interactions with RC

- Usual instructions for starting / stopping a run
 - `boot` to start the applications
 - `start_run --message 'message to post to elog'`
 - **Remember:** `start_run` executes `conf`, `start`, then enable triggers. If we want to start without enabling triggers, instruct the shifter to do the commands separately
 - Configuration information automatically posted into `elog`
 - `stop_run --message 'message to post to elog'`
 - Bring us back to 'configured' state. Can start a run from here.
 - `shutdown` to close all applications
 - `exit` to exit `nano04rc` (needed to change configs)
- Use `--force` to `stop_run` or `shutdown` if an application is not responding
- Each command should have a help (`-h`) option
 - And `help` command in `nanorc` will print available commands

Run control

Run information

The screenshot shows a terminal window titled "np04daq@np04-srv-024:~/DAQ_NP04_HD_AREA". At the top, it indicates "Run #26960 ongoing". Below this, a box contains run information:

Type	PROD
Start time	12/06/2024 20:49:52
Duration	0:00:02.099482
Data storage enabled	True
Trigger rate	default from config (1Hz?)

Below the run information, a table titled "np04_hd applications in partition np04hd" lists various applications. The table has columns for name, state, host, pings, last cmd, and last succ. cmd. A tree structure is shown on the left side of the table, indicating the hierarchy of segments and applications within segments.

name	state	host	pings	last cmd	last succ. cmd
np04_hd	running				
np04_wib	running				
wib101	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib102	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib103	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib104	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib105	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib201	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib202	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib203	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib204	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib205	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib301	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib302	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib303	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib304	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib305	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib401	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib402	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib403	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib404	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
wib405	running - alive	np04-srv-011	True	enable_triggers	enable_triggers
np04_hermes	running				
hermes	running - alive	np04-srv-013	True	enable_triggers	enable_triggers
np04_daq	running				
ctbhsi	running - alive	np04-srv-012	True	enable_triggers	enable_triggers
dataflow0	running - alive	np04-srv-005	True	enable_triggers	enable_triggers
dataflow1	running - alive	np04-srv-005	True	enable_triggers	enable_triggers
dataflow2	running - alive	np04-srv-005	True	enable_triggers	enable_triggers
dataflow3	running - alive	np04-srv-005	True	enable_triggers	enable_triggers
dfo	running - alive	np04-srv-024	True	enable_triggers	enable_triggers
runp04srv021eth0	running - alive	np04-srv-021	True	enable_triggers	enable_triggers
runp04srv022eth0	running - alive	np04-srv-022	True	enable_triggers	enable_triggers
runp04srv028eth0	running - alive	np04-srv-028	True	enable_triggers	enable_triggers
runp04srv029eth0	running - alive	np04-srv-029	True	enable_triggers	enable_triggers
trigger	running - alive	np04-srv-018	True	enable_triggers	enable_triggers

At the bottom of the terminal, the prompt shows "wketchum@np04rc>" and the command "[np04_hd] 1:python*" is entered. The terminal also shows the system time "np04-srv-024" 05:40 13-Jun-24.

Tree of segments and applications within segments

Host the application is running on!

username shown at command line

Shifter / expert responsibilities

- Shifters should be able to ...
 - Login as np04daq and attach to or detach from tmux session
 - Start / stop runs for specified configurations
 - Post messages to the elog describing detector conditions
 - Monitor state of DAQ and detector
- DAQ experts should be able to
 - Create and setup DAQ software areas and tmux sessions
 - Create / modify configurations and communicate these to shifter
 - Update shifter instructions for run control as needed (e.g. if we should not do "start_run")
 - Help debug / provide instructions to shifters as needed

Exercises

- Login as np04daq on np04-srv-013
- ***In the /nfs/sw/dunedaq-testing*** area, create a new development software area based on fddaq-v4.4.2 with your name as suffix
 - Create all dro maps, WIB, DAPHNE, and NP04 DAQ configurations
- Create a tmux session (on np04-srv-013) and set it up to be ready to run
- Make or modify an existing configuration to ...
 - Move dataflow applications for the global_configs/np04/np04_TPC_noTPG_10Hz_ssh.json to run on four different storage servers
 - See <https://twiki.cern.ch/twiki/bin/view/CENF/ComputerAssignments> if you need to ...
 - Add trigger primitive generation to global_configs/np04/np04_TPC_noTPG_ctbFakeBeam_ssh.json
 - See global_configs/np04/np04_TPC_TPG_100mHz_ssh.json to help ...
 - Make a np04_TPC_noTPG_ctbFakeBeam_k8s configuration
 - Readout APA1 on np04-srv-021 and APA3 to readout on np04-srv-028
 - Pinning files!
 - Remove data streams from FEMB 1 on the third WIB of APA2

Other notes: DQM

- We are still working out the logistics for DQM, but ...
 - DQM applications should run for now in tmux sessions (likejustintime is running np04hd_dqm on np04-srv-004) where there is access to the data
 - They should run as np04daq user (to allow us to share control of them)
 - It should generally be safe to stop (ctrl-c) and restart DQM applications at any point if there's some problem
- Will have more information to share on that as that is better developed

Communication

- We use the #np04-shifterassistant for communication with shifters and on operations
 - Let's all try to be good about not cluttering it with too much detail
- I have created today a new channel #daq-experts-protodune2 for us to be able to communicate on DAQ expert issues
 - I know it's another channel / could overlap with a lot else – welcome to consider an existing channel if people prefer
- Propose that we plan to meet *daily* starting on Monday at 08:00
 - Standing meeting in 892-2-B19, aiming for 15 minutes, can have a zoom
 - Ideal to have ~all experts attend, but especially those on-call
- We should plan to have a DAQ expert available to meet the shifters at the beginning of every shift
 - Make sure shifters know how to run the DAQ, are aware of any issues, etc.
 - Hopefully over time this can be relaxed

DAQ Expert signup

- I began to add a 'DAQ expert' signup to the shift spreadsheet:
 - https://docs.google.com/spreadsheets/d/1_yYS8HMSpZ4zBtYlxE36sWsXQkeOvJc31goupQsWrf4/edit?usp=sharing
- We need to make sure we have an on-call expert available
 - An expert per week or M-TH, F-S rotation is sensible, though we can trade days as needed
 - This person should not also be taking detector shifts
 - (Though DAQ experts on shift should certainly help lessen the load of the on-call expert)
- For the first week of beam next week, it would be ideal to have a DAQ expert available with every shift
 - Ideally in CR with shifters, if not already on shift

Future training

- I expect we will be solidifying a lot over the next weeks, ahead of the full summer beam run
- Expect that we will have an updated training session ahead of the beam run
 - Hopefully we can couple that with some more detailed overview of the DAQ system
 - Your input into that will be very helpful!
 - What do you want to see / feel you need to know more about?
 - What can you help prepare that you think would be useful?

Closing notes

- DO. NOT. PANIC.
 - Few people are at their best when scrambling
 - No one knows everything – don't ever be afraid to ask for help!
 - Have you seen how often I do that? Even for stupid things?
- I am ~100% on call
 - Ping me on slack @wketchum
 - Though, I'm not likely to see it all the time, so after doing that ...
 - +1 312 792 1011 (whatsapp)
 - +41 78 222 87 58 (local cell)
 - 61778 (CERN phone)