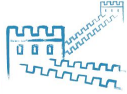
Manage the Physics Settings on the Modern Accelerator

Tong Zhang
Controls Physicist
Facility for Rare Isotope Beams, Michigan State University
E-mail: zhangt@frib.msu.edu



Presented at 18th International Conference on Accelerator and Large Experimental Physics Control System
14-22 October 2021, Shanghai, China

MICHIGAN STATE UNIVERSITY

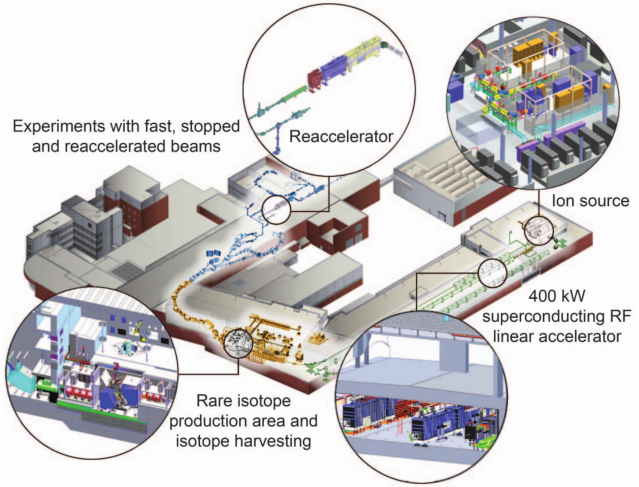
U.S. DEPARTMENT OF ENERGY | Office of Science

This material is based upon work supported by the U.S. Department of Energy Office of Science under Cooperative Agreement DE-SC0000661, the State of Michigan and Michigan State University. Michigan State University designs and establishes FRIB as a DOE Office of Science National User Facility in support of the mission of the Office of Nuclear Physics.

Outline

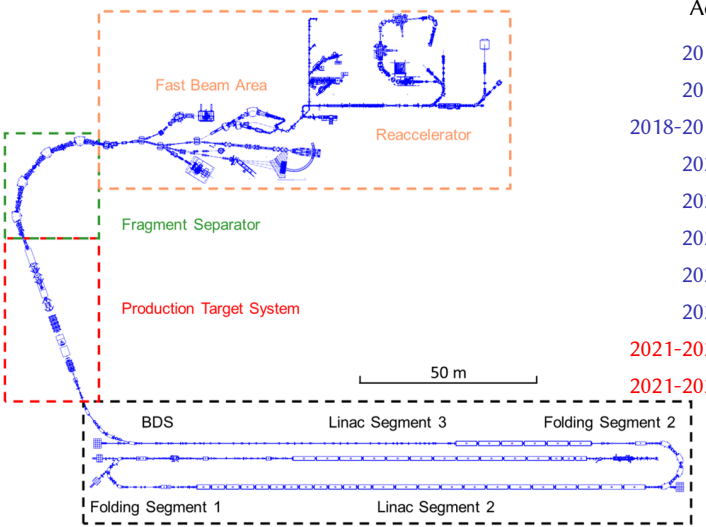
Outline

Facility for Rare Isotope Beams



▷ T. Glasmacher et al., "Facility for Rare Isotope Beams Update for Nuclear Physics News", Nuclear Physics News, 27 (2017) 2, 28-33.

Facility for Rare Isotope Beams



Accelerator Segments, staged commissioning:

- 2017 ECR ion source
- 2017 Front-End (FE), 0.5 MeV/u
- 2018-2019 Linac Segment 1 (LS1), 20 MeV/u
- 2020 Folding Segment 1 (FS1)
- 2020 Linac Segment 2 (LS2), 200 MeV/u
- 2021 Folding Segment 2 (FS2)
- 2021 Linac Segment 3 (LS3), > 200 MeV/u
- 2021 Beam Delivery System (BDS)
- 2021-2022 Target Systems, rare isotope beams!
- 2021-2022 Fragment Separator (ARIS)

Software Solution for High-level Physics Controls

PHANTASY

Physics High-level Applications aNd Toolkit for Accelerator SYstem

https://github.com/phantasy-project

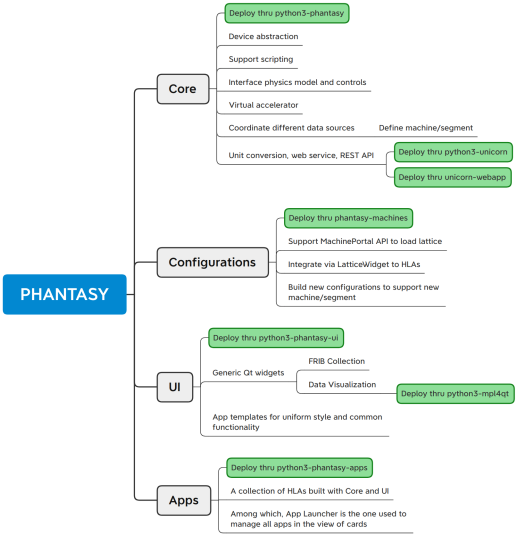
Features Highlight

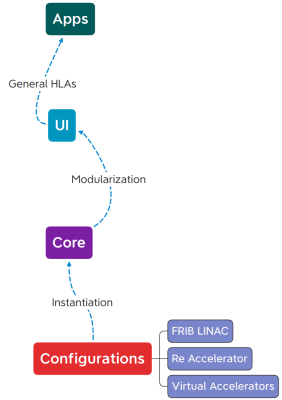
- Device configuration management: maintainability, portability
- Device abstraction: object-oriented
- Online modeling: physics model-depends machine representation
- Python interactive scripting environment for high-level controls: development and control
- Virtual accelerator based on EPICS and physics model: test physics algorithms
- Web service integration (channelfinder, scanserver, archiver appliance, UNICORN ...): extendability

Deployment

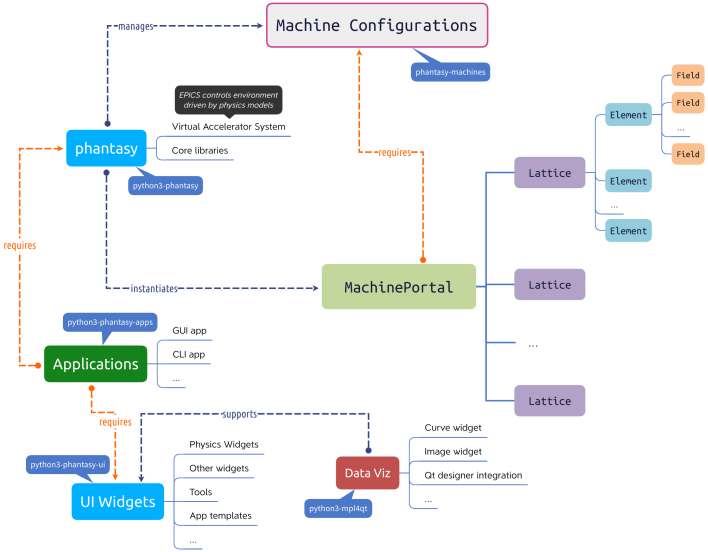
- Target OS: Debian Linux, to support PyPI, conda
- Only support Python 3.x
- Meta package: phantasy (~20 packages)
- Physics model engines: FLAME

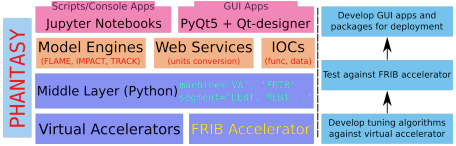
Evolution of PHANTASY Project





Evolution of PHANTASY Project





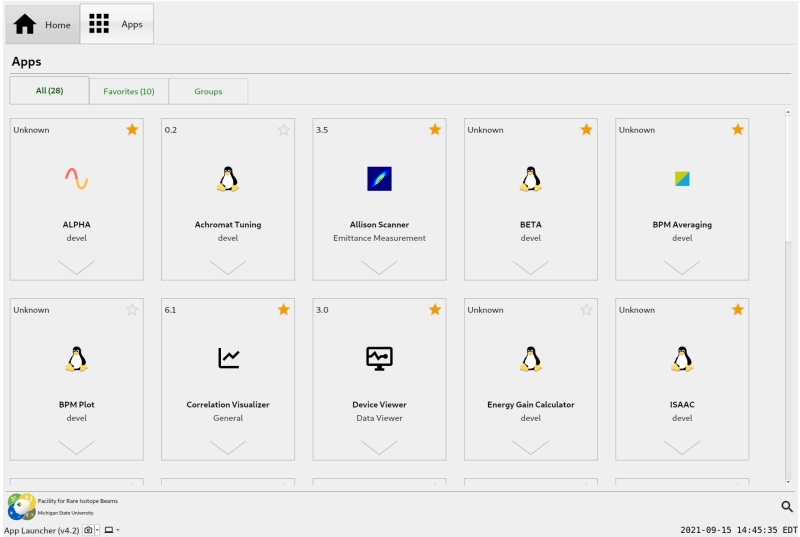
Code Facts

- Python 3, Debian Linux OS
- PHANTASY (core) ~20k LOC
- UI widgets: ~18k LOC (incl. dataviz, excl. ui)
- Apps: ~40k LOC (excl. ui)

Development Guide

- Device ⟶ Python object
- List of devices ⟶ Lattice
- Lattice manipulation
- Scripting/GUI apps (facility-agnostic)
- Package ⟶ deployment

Global Entry for All the Apps: Present and Manage the Information in A Better Way

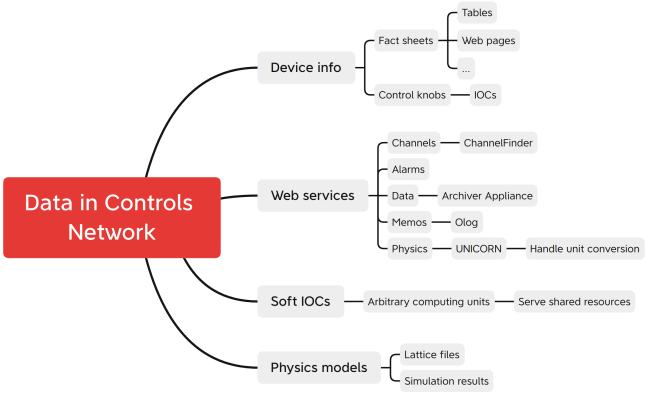


- Global entrypoint: right-clicking context menu → 'Physics Apps'
- Managed by a configuration file
- Present each app in one card, click to launch it
- User favored apps as the main page
- Another page for all apps (grouped)
- Support search
- Expanded card for more actions/info: documentation, contact...

Outline

Data in the Controls Network

- Device information: static properties → databases/files → handled in phantasy
- Device control knobs: IOCs → distributed → handled in phantasy
- Web services: request/response → distributed → Python client packages
- Soft IOCs: support HLAs → presented as additional attributes in phantasy
- Physics models: simulation, machine tuning → support FLAME

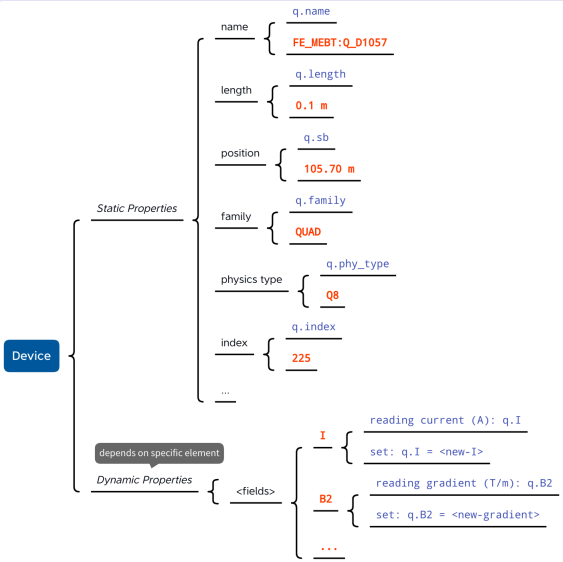


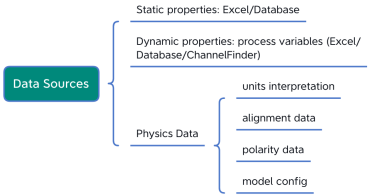
Manage the Data for Machine Tuning

Key Points to Address

- Device information accessibility
- Physics tuning repeatability
- Machine state data integrity

Device Information Accessibility






```
>>> from phantasy import MachinePortal
>>> mp = MachinePortal("FRIB", "LINAC")
>>> q = mp.get_element(type='QUAD', name='*1057*')[0]
```

Retrieve high-level object

Manage Data of Physics Settings

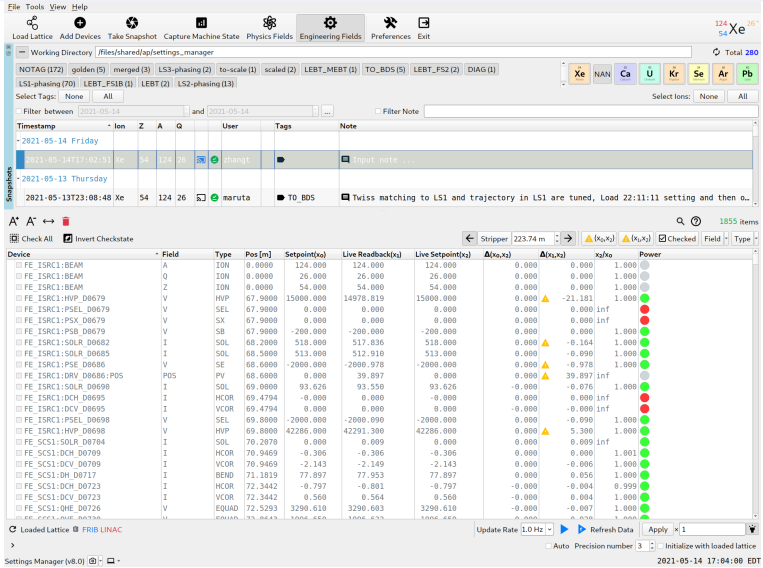
Requirements

- Manage accelerator physics settings segment by segment
- Correlate various kinds of accelerator data in one place
- Quickly model with physics code
- Scaling for working with different ion species
- Features for data accessibility and availability

Design Ideas

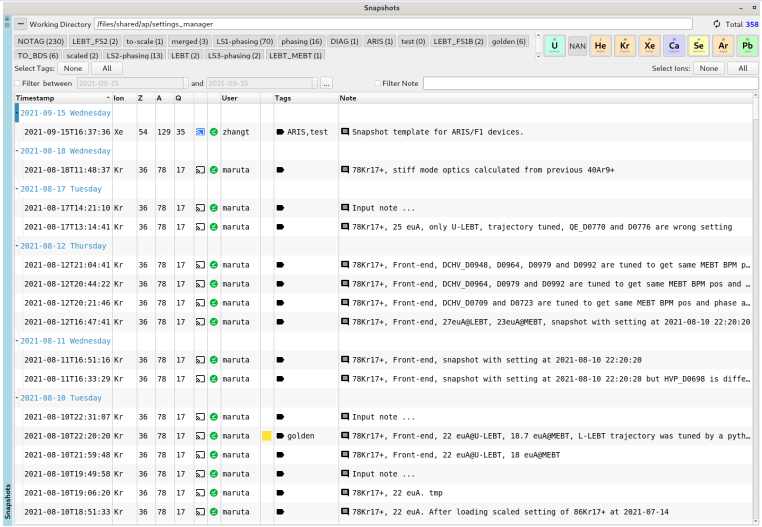
- Manage device entities rather than process variables
  - PHANTASY defines each device has multiple controllable ‘fields’
- Manage different data from different sources: live, archived
  - Aggregate data to expressive tables
- One snapshot should have all the information for:
  - Data investigation, visualization
  - Physics modeling

Settings Manager: Manage Physics Settings with Machine State and More



- Pull device info and live state, refer to phantasy project
- 'Take Snapshot': fetch machine state or not
- Fetched machine state presented as a new table along with the physics settings
- Both are stored in one new database record
- Users can export it to various file formats: XLSX, CSV, HDF5, etc.
- Filter buttons for snapshot records and physics settings
- Integration for Archiver Appliance through pyarchappl

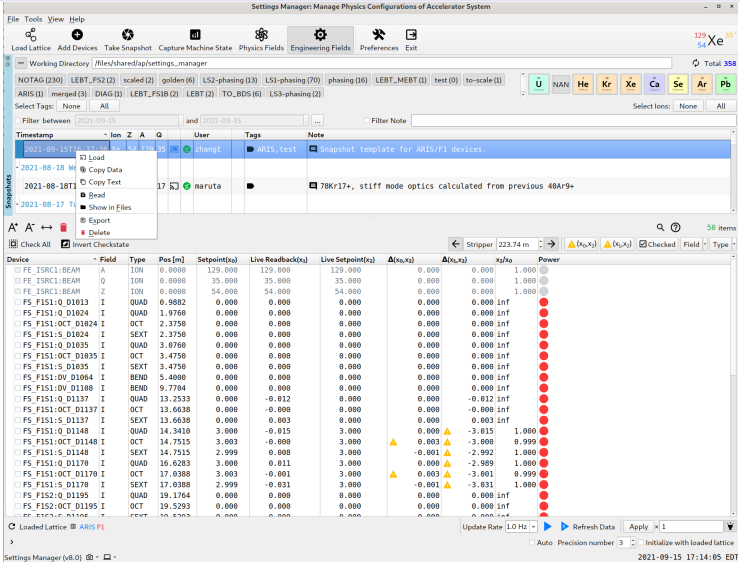
Settings Manager: Manage Physics Settings with Machine State and More



- Pull device info and live state, refer to phantasy project
- ‘Take Snapshot’: fetch machine state or not
- Fetched machine state presented as a new table along with the physics settings
- Both are stored in one new database record
- Users can export it to various file formats: XLSX, CSV, HDF5, etc.
- Filter buttons for snapshot records and physics settings
- Integration for Archiver Appliance through pyarchappl

Click to view one captured snapshot in xlsx

Settings Manager: Manage Physics Settings with Machine State and More



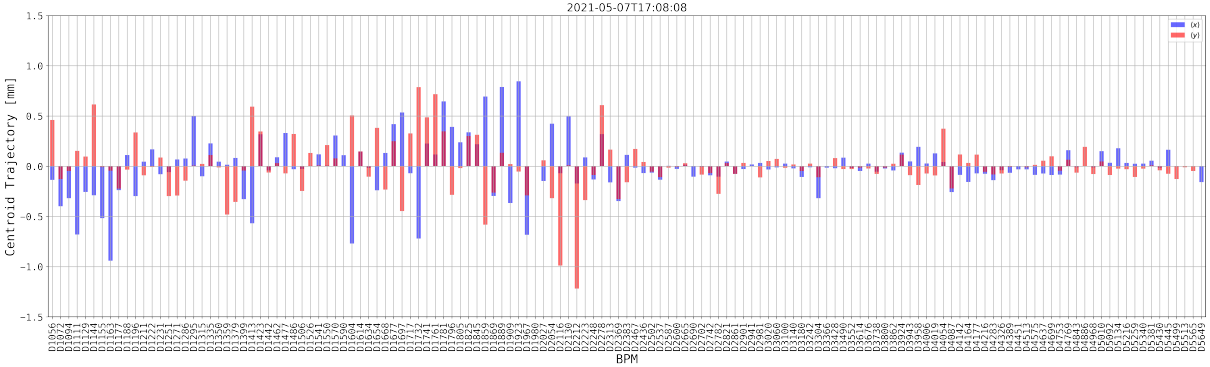
- Pull device info and live state, refer to phantasy project
- 'Take Snapshot': fetch machine state or not
- Fetched machine state presented as a new table along with the physics settings
- Both are stored in one new database record
- Users can export it to various file formats: XLSX, CSV, HDF5, etc.
- Filter buttons for snapshot records and physics settings
- Integration for Archiver Appliance through pyarchappl

Settings Manager: Manage Physics Settings with Machine State and More

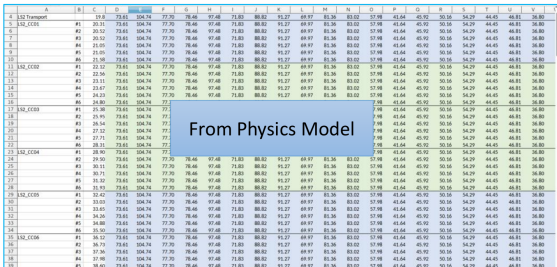
```
tong ~ pyarchappl-get -h
usage: pyarchappl-get [-h] [--url URL] [--pv PV_LIST] [--pv-file PV_FILE]
                      [--from FROM_TIME] [--to TO_TIME] [--resample RESAMPLE]
                      [--verbose] [-o OUTPUT] [-f FMT]
                      [--format-args FMT_ARGS]

Retrieve data from Archiver Appliance and export as a file.

optional arguments:
  -h, --help            show this help message and exit
  --url URL             URL of Archiver Appliance, default is FRIB FTC
                        archiver (default: None)
  --pv PV_LIST          List of PVs for retrieval, each define with --pv
                        (default: None)
  --pv-file PV_FILE     A file for PVs, one PV per line (skip line starts with
                        #), append each to pv_list (default: None)
  --from FROM_TIME      A string of begin time in ISO8601 format (default:
                        None)
  --to TO_TIME          A string of end time in ISO8601 format (default: None)
  --resample RESAMPLE   The offset string/object representing target
                        conversion, e.g. '1S' for resample with 1 second
                        (default: None)
  --verbose, -v         Verbosity level of the log output, 0: no output,
                        1(-v): output progress, 2(-vv): output progress with
                        description (default: 0)
  -o OUTPUT, --output OUTPUT
                        File path for output data, print to stdout if not
                        defined (default: None)
  -f FMT, --output-format FMT
                        File format for output data, supported: csv, hdf,
                        excel, html, ... (default: csv)
  --format-args FMT_ARGS
                        Additional arguments passed to data export function in
                        the form of dict, e.g. '{"key":"data"}' (for hdf
                        format) (default: {})

Examples:
# Retrieve raw PV data in the defined time frame
$ pyarchappl-get -o data.csv -v \
  --pv LS1_CA01:BPM_D1129:XPOS_RD --pv LS1_CA01:BPM_D1129:YPOS_RD  \
  --from 2021-04-15T20:10:00.000Z --to 2021-04-15T21:25:00.000Z \

# Align the timestamps, resample at 1 second
$ pyarchappl-get -o data.csv -v \
  --pv LS1_CA01:BPM_D1129:XPOS_RD --pv LS1_CA01:BPM_D1129:YPOS_RD  \
  --from 2021-04-15T20:10:00.000Z --to 2021-04-15T21:25:00.000Z \
  --resample 1S
```

- Pull device info and live state, refer to phantasy project
- 'Take Snapshot': fetch machine state or not
- Fetched machine state presented as a new table along with the physics settings
- Both are stored in one new database record
- Users can export it to various file formats: XLSX, CSV, HDF5, etc.
- Filter buttons for snapshot records and physics settings
- Integration for Archiver Appliance through pyarchappl

Settings Manager: An Example of Machine State Data (BPM group)

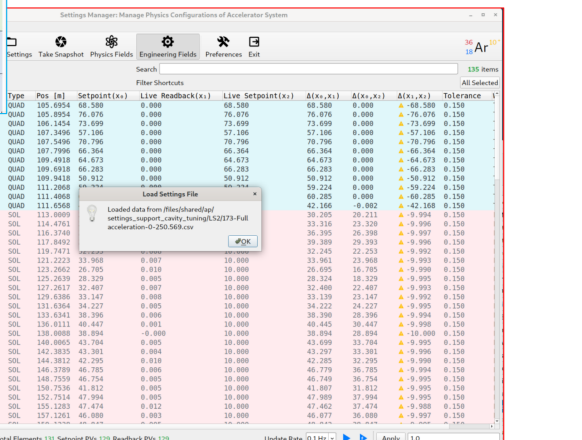


Settings Manager: Pre-generate Settings from Physics Model for Commissioning



Convert xlsx sheet settings as files/database records for Settings Manager to load

Load



Check device to apply the settings, w/ scaling factor if needed

Significantly saved effort to set the optics during cavity tuning to maintain beam envelope

Retrieve Archived Data: Machine State and Physics Settings

pvlist.txt buffers

```
# a column of 150 PVs
# each row for one PV
# line starts with # is bypassed
VA:LS1_CA01:BPM_D1129:X_RD
VA:LS1_CA01:BPM_D1129:Y_RD
VA:LS1_CA01:BPM_D1144:X_RD
VA:LS1_CA01:BPM_D1144:Y_RD
VA:LS1_WA01:BPM_D1155:X_RD
VA:LS1_WA01:BPM_D1155:Y_RD
VA:LS1_CA02:BPM_D1163:X_RD
VA:LS1_CA02:BPM_D1163:Y_RD
VA:LS1_CA02:BPM_D1177:X_RD
VA:LS1_CA02:BPM_D1177:Y_RD
VA:LS1_WA02:BPM_D1188:X_RD
VA:LS1_WA02:BPM_D1188:Y_RD
VA:LS1_CA03:BPM_D1196:X_RD
VA:LS1_CA03:BPM_D1196:Y_RD
VA:LS1_CA03:BPM_D1211:X_RD
VA:LS1_CA03:BPM_D1211:Y_RD
VA:LS1_WA03:BPM_D1222:X_RD
VA:LS1_WA03:BPM_D1222:Y_RD
VA:LS1_CB01:BPM_D1231:X_RD
VA:LS1_CB01:BPM_D1231:Y_RD
VA:LS1_CB01:BPM_D1251:X_RD
VA:LS1_CB01:BPM_D1251:Y_RD
VA:LS1_CB01:BPM_D1271:X_RD
VA:LS1_CB01:BPM_D1271:Y_RD
VA:LS1_WB01:BPM_D1286:X_RD
VA:LS1_WB01:BPM_D1286:Y_RD
VA:LS1_CB02:BPM_D1295:X_RD
VA:LS1_CB02:BPM_D1295:Y_RD
VA:LS1_CB02:BPM_D1315:X_RD
VA:LS1_CB02:BPM_D1315:Y_RD
VA:LS1_CB02:BPM_D1335:X_RD
VA:LS1_CB02:BPM_D1335:Y_RD
VA:LS1_WB02:BPM_D1350:X_RD
VA:LS1_WB02:BPM_D1350:Y_RD
VA:LS1_CB03:BPM_D1359:X_RD
VA:LS1_CB03:BPM_D1359:Y_RD
VA:LS1_CB03:BPM_D1379:X_RD
```

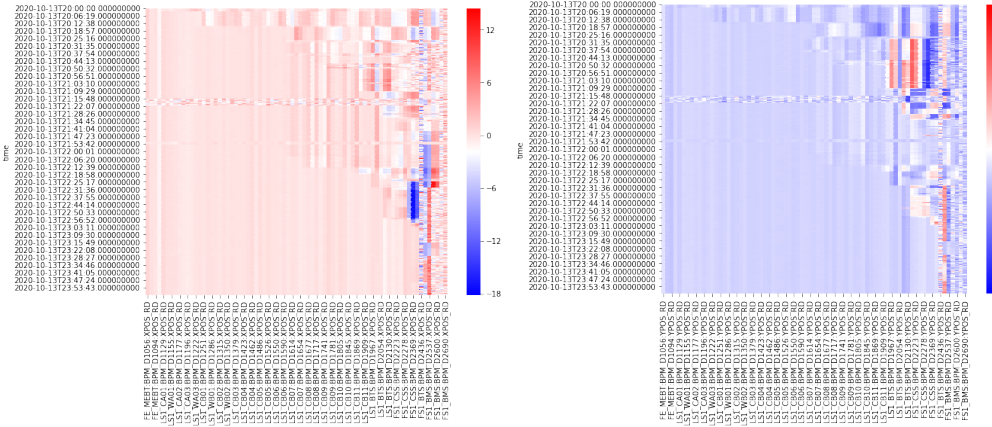
run.sh buffers

```
# CAUTION: run the following script after removing all the trailing comments!
# here is to explain the command
pyarchappl-get -vv \                          # show progress bar and more output
    --pv-file pvlist.txt \                    # load PVs from external file
    --from 2021-04-15T20:10:00.000Z \         # begin time in GMT timezone
    --to 2021-04-15T17:25:00.00-04:00 \       # end time in America/New_York, demo only
                                      \       # usually use the same timezone, i.e.
                                      \       # 2021-04-15T21:25:00.000Z
    --url http://127.0.0.1:17665 \            # define the address of a local archiver
    --resample 1S \                           # align data with interal of 1 second
    --output data.h5 \                        # save data as a hdf5 file
    --output-format hdf \                     # define file format
    --format-args '{"key":"Trajectory"}'      # required for 'hdf' format
```

140514 /home/tong/Dropbox/pyarchappl/main/scripts/examples

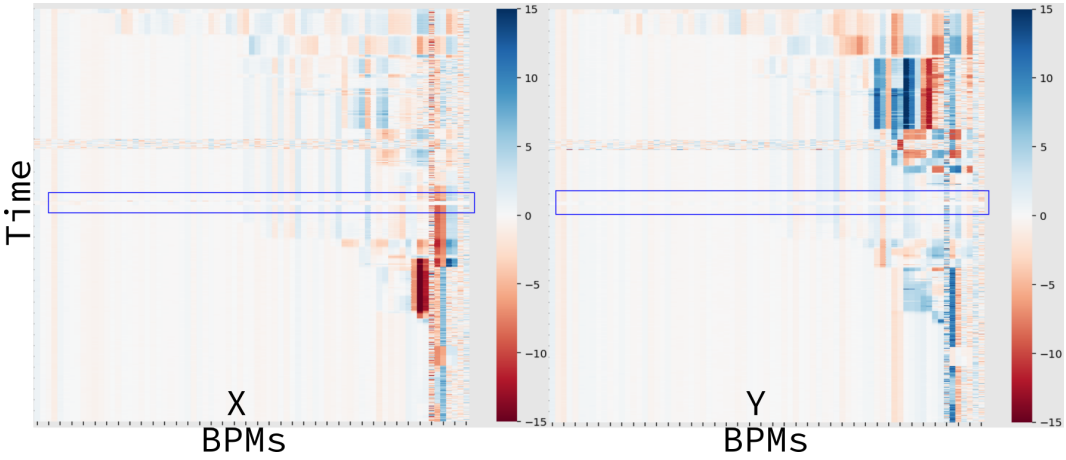
```
tong  master  …  main  scripts  examples  pyarchappl-get -vv \
                                           --pv-file pvlist.txt \
                                           --from 2021-04-15T20:10:00.000Z \
                                           --to 2021-04-15T17:25:00.00-04:00 \
                                           --url http://127.0.0.1:17665 \
                                           --resample 1S \
                                           --output data.h5 \
                                           --output-format hdf \
                                           --format-args '{"key":"Trajectory"}'
Fetched VA:FS1_BMS:BPM_D2702:Y_RD: 100%|                    | 150/150 [00:13<00:00, 11.28it/s]
[2021-04-20T17:01:33.889305] Fetched all, time cost: 14.8 seconds.
tong  master  …  main  scripts  examples  pyarchappl-get -vv --pv-file pvlist.txt
 --from 2021-04-15T20:10:00.000Z --to 2021-04-15T17:25:00.00-04:00 --url http://127.0.0.1:
17665 --resample 1S --output data.csv
Fetched VA:LS1_CB07:BPM_D1634:X_RD:  46%|                    | 69/150 [00:03<00:03, 22.37it/s]
```

Retrieve Archived Data: Machine State and Physics Settings



▷ pyarchappl project, “Pull, Visualize and Play the data from Archiver Appliance”

Retrieve Archived Data: Machine State and Physics Settings



▹ pyarchappl project, “Pull, Visualize and Play the data from Archiver Appliance”

Outline

Conclusions

- Overviewed the status of Python-based software framework “phantasy” project for the high-level physics controls for FRIB LINAC
- Overviewed the different data categories in the controls network
- Proposed and developed software for the data management for machine tuning
- Overviewed the developed tools for machine state data management

Thank you for your attention!