

# PYTHON SCRIPTS FOR RF COMMISSIONING AT FRIB\*

H. Maniar, E. Daykin, D. Morris, A. Plastun, H. Ren, S. Zhao  
Facility for Rare Isotope Beams, Michigan State University, East Lansing, MI, USA

## Abstract

RF commissioning at FRIB involves QWR cavities ( $\beta=0.085$  and  $\beta=0.041$ ), HWR cavities ( $\beta=0.29$  and  $\beta=0.53$ ) and few room temperature devices. Each RF system has many process variables for LLRF and amplifier control located on different pages of CS-Studio. Efficient handling of all these PVs can be challenging for RF experts. Several scripts using Python have been developed to facilitate this process. User interface application has been developed using Qt Designer and PyQt package of Python, for ease of access of all scripts. These scripts are useful for mass actions (for multiple systems) including turning on/off LLRF controllers and amplifiers, resetting interlocks/errors, changing a PV value, etc. Python scripts are also used to quickly prototype the auto-start procedure for QWR cavities, which eventually is implemented on IOC driver. The application sends commands to IOC driver with device name, PV name and value to be changed. Future developments can be converting to state-notation language on IOC to add channel access security. This application intends to reduce time and efforts for RF commissioning at FRIB.

## INTRODUCTION

The Facility for Rare Isotope Beams (FRIB) at MSU will be a scientific user facility for nuclear physics research and will enable scientists to make discoveries about the properties of rare isotopes [1]. The FRIB linac consists of room temperature front end devices such as Low Energy Beam Transport (LEBT), Multi-Harmonic Buncher (MHB), Radio Frequency Quadrupole (RFQ) and Medium Energy Beam Transport (MEBT) bunchers. It also includes 104 Quarter-Wave Resonators (QWR) and 220 Half-Wave Resonators (HWR). Table 1 shows parameters of SRF cavities at FRIB. Figure 1 shows the layout of FRIB linac [2].

Table 1: FRIB Resonator Parameters

Resonator	QWR1	QWR2	HWR1	HWR2
$\beta$	0.041	0.085	0.29	0.53
f (MHz)	80.5	80.5	322	322
No. of cavities	12	92	72	148
Tuner type	Stepper	Stepper	Pneumatic	Pneumatic

CS-Studio screens have been developed for users to access all RF parameters related to Low Level Radio Frequency (LLRF) Controller and Amplifier. Each RF system contains around 400 process variables (PV) including parameters for interlocks, control parameters, cavity conditioning, calibration, attenuation, system configuration etc. All these PVs are accessible using Experimental Physics

\* Work supported the U.S. Dept. of Energy Office of Science under Cooperative Agreement DE-SC0000661.

and Industrial Control System (EPICS) channels. Input/Output Controller (IOC) driver handles read/write actions to these PVs.

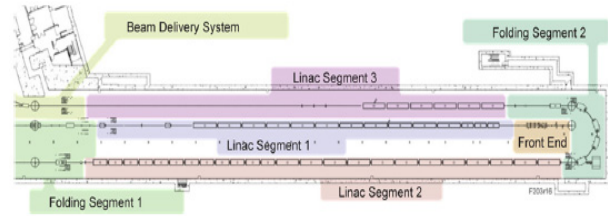


Figure 1: FRIB linac.

## PYTHON

Python offers several convenient features for scientific and engineering programming [3]. The Python EPICS package (PyEpics) is useful to interact with EPICS channel access PVs. The PyEpics package contains several function, modules and classes to interact with EPICS channel access [3]. Multiple Python scripts have been developed using PyEpics channel for RF commissioning. GUI framework has been also developed using PyQt library from Python.

### Python GUI

Qt Designer is the Qt tool helpful in designing and building graphical user interfaces. It allows user to use built-in widgets and forms to develop designs. It uses eXtensible Markup Language (XML) '.ui' format to store design files. Typically, Qt designer stores GUI files in XML format and does not generate C++ or Python code. PyQt is a library that includes *uic* Python module. Qt Designer's '*uic*' utility is useful in creating C++ code, whereas PyQt's '*pyuic*' utility is useful to generate Python files [3]. Any user interface files '.ui' can be converted to Python files '.py' using this command in terminal

```
'pyuic5 filename.ui -o filename.py'
```

### Python EPICS

This package can be called in any Python script using '*import epics*'

The main components of this module include functions such as *caget()*, *caput()*, *camonitor()* to simply read, write, monitor PVs. It also includes a *ca* module, useful for low-level epics channel access and a PV object, useful for higher-level epics channel access [4]. The get functions ask for data to be transferred over the network. For large data array, it can take a significant amount of time. Also for disconnected PVs, function will not return any value. For these reasons, *get* functions should be used with *timeout* or *wait* options. It also has options of *count* and *numpy* to return number of elements for array data.



## REFERENCES

- [1] FRIB, <http://www.frib.msu.edu>
- [2] D. Morris *et al.*, “RF System for FRIB Accelerator”, International Particle Accelerator Conference, Vancouver, BC, Canada, 2018
- [3] PyQt v5.13 reference guide, <http://www.riverbankcomputing.com>
- [4] Matthew Newville, “PyEpics: Epics Channel Access clients with Python,” Consortium for Advanced Radiation Sciences, University of Chicago, 2014
- [5] J. Vincent *et al.*, “On active disturbance rejection based control design for superconducting RF cavities,” Nuclear Instruments and Methods in Physics Research A, vol. 643, no. 1, pp. 11-16, 2011.

