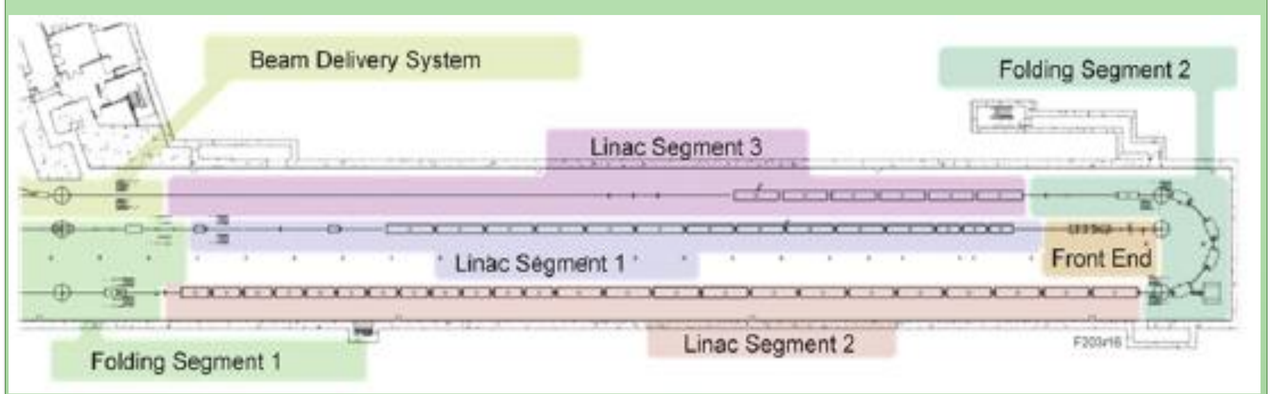
PYTHON SCRIPTS FOR RF COMMISSIONING AT FRIB

Harsh Maniar, Shen Zhao, Dan Morris, Alexander Plastun, Haitao Ren, Evan Daykin
Facility for Rare Isotope Beams, MSU, East Lansing, MI 48824, USA

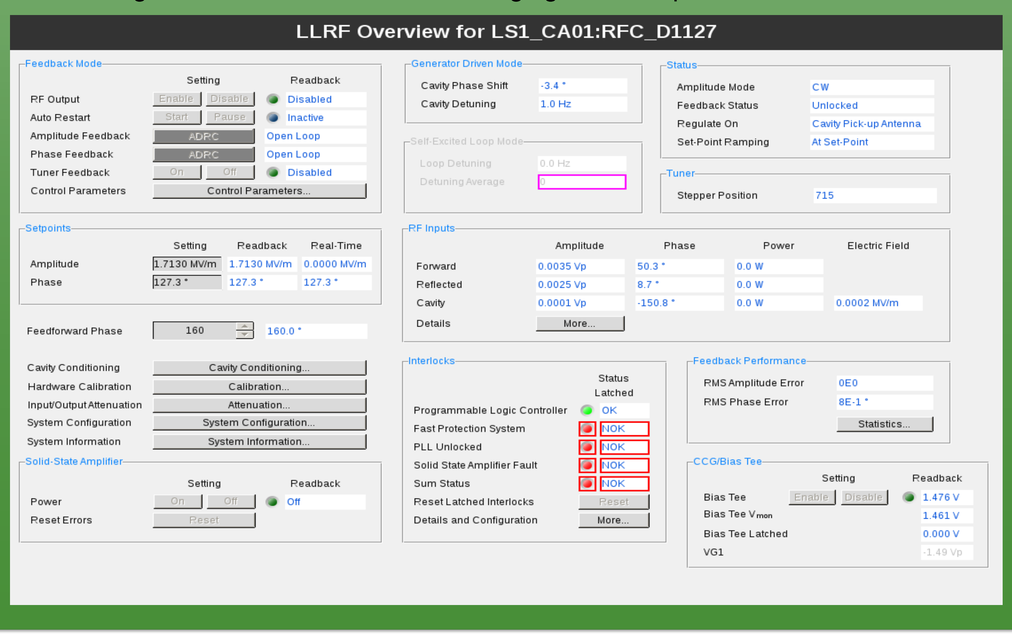
FRIB Linac



The FRIB linac consists of
- room temperature front end devices: Low Energy Beam Transport (LEBT), Multi Harmonic Buncher (MHB), Radio Frequency Quadrupole (RFQ) and Medium Energy Beam Transport (MEBT) bunchers
- 104 Quarter-Wave Resonators (QWR)
- 220 Half-Wave Resonators (HWR)

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

- CS-Studio engineering 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 and Industrial Control System (EPICS) channels. IOC driver handles read/ write actions to these PVs
- Efficient handling of all these PVs can be challenging for RF experts

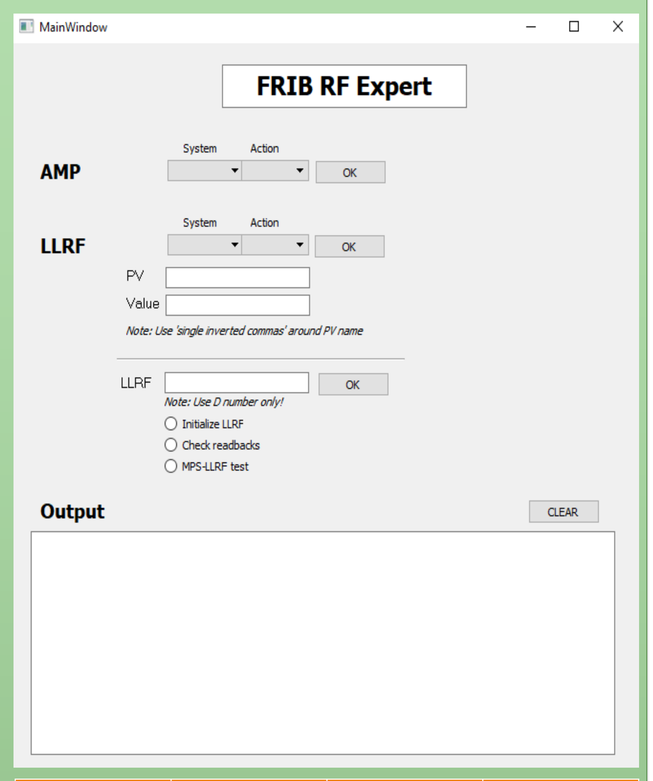


Python

- Python offers several convenient features for scientific and engineering programming. The Python EPICS package (PyEpics) is useful to interact with EPICS channel access PVs.
  - '*import epics*'
- The main components of this module include
  - functions such as *caget (), caput (), camonitor ()* to simply read, write, monitor PVs
  - *a ca* module, useful for low-level epics channel access
  - a *PV* object, useful for higher–level epics channel access
  - *timeout* and *wait* options for large data arrays and disconnected PVs
  - *count* and *numpy* options to return number of elements for array data and arithmetic functions
- Qt Designer is the Qt tool helpful in designing and building graphical user interfaces. Utilities from PyQt library helps to generate Python files
  - Built-in widgets and forms
  - XML *'.ui'* format to store design files
  - *'uic'* module to convert to C++ code
  - *'pyuic'* module to convert to Python code
- Any user interface files *'.ui'* can be converted to Python files *'.py'* using this command in terminal
  - *'pyuic5 filename.ui –o filename.py '*

RF Python Scripts

- Multiple Python scripts have been developed using PyEpics channel for RF commissioning. GUI framework has been also developed using PyQt library from Python.
- 'FRIB RF Expert' user interface application has been developed using Qt Designer and converted to Python files using '*pyuic*' module. This generated Python file serves as main file in application. All developed Python scripts for RF commissioning are nested under this main file.
- This application let users perform mass action to apply to multiple LLRF and amplifier systems
- **System:** Selection based on cavity number, cryomodule number, system type or linac segment type. Script searches for keyword in database where all device names for LLRF controller and Amplifier are stored
- **Action:** Options to turn On/ Off, clear/ reset interlocks for LLRF controller and amplifier
- **PV:** Manually input PV name and Value
- **Initialize LLRF:** Helpful to setup parameters such as cavity type, attenuation, interlocks, control parameters etc. to initial values before running RF for first time
- **Check Readbacks:** Helpful to make sure all set-points and readbacks match and there is no discrepancy in values
- **MPS-LLRF test:** Useful to run test between LLRF controller and Machine Protection System (MPS)
- Once any combination of system and action are selected, it shows a pop-up message to verify user actions
- Output window shows device names, PV names and values it has changed
- Python script is an easy way to prototype any state machine quickly and test the logic
- Once developed and tested, script's logic can be transferred to state-notation language and implement on IOC driver
- To provide most channel access security, it is recommended to implement state machines on IOC driver.



Linac	System Type	No. of Cryo-module	No. of Cavities
Linac Segment 1	CA	3	12
Linac Segment 1	CB	11	88
Folding Segment 1	CH	1	4
Linac Segment 2	CC	12	72
Linac Segment 2	CD	12	96
Folding Segment 2	CG	1	4
Linac Segment 3	CD	6	48

QWR Cavity Turn On Sequence

- For amplitude and phase regulation, the LLRF controller adopts the active disturbance rejection control (ADRC) algorithm
- Before locking cavity to desired amplitude and phase, it goes through different stages
- Python script was initially developed to automatically turn on QWR cavities. After testing it on multiple cavities and cryomodules, it was transitioned to state-notation language and implemented on IOC driver

	RF	Amplitude	Amplitude Set-point	Phase	Tuner
Stage 1	**ON**	Open	Initial	SEL	OFF
Stage 2	ON	Open	Initial	SEL	**ON**
Stage 2	ON	Open	Initial	**Open**	ON
Stage 2	ON	**Close**	Initial	Open	ON
Stage 2	ON	Close	Initial	Open	ON
Stage 2	ON	Close	**Final**	Open	ON

Future Work

- Future work involves transitioning this application to state-notation language and implement on IOC driver to add channel access security

Facility for Rare Isotope Beams
U.S. Department of Energy Office of Science
Michigan State University

NAPAC2019
NORTH AMERICAN PARTICLE ACCELERATOR CONFERENCE

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