Low Level RF Workshop 2022 - Poster #36

SPS & LHC setting-up tools using Python

D. Długosz*, *J.Egli, G. Hagmann, P. Baudrenghien, B. Bielawski, H. Timko, S. Novel Gonzalez, A. Butterworth*
CERN, Geneva, Switzerland

LLRF2022

Abstract

Commissioning of the LLRF systems of CERN accelerators consists of a number of time-consuming procedures, involving calibration and fine-tuning of numerous parameters. In the recent years, a system of Python scripts was developed to automate the setting up of the LLRF of the LHC and the SPS. Targeted at RF experts, the scripts provide high-level interface to the underlying physical system, managing I/O communication and performing data analysis and parameter optimisation. Designed with maintainability and portability in mind, the system can be expanded to be used with other accelerators at CERN.

In addition to the scripts, a dedicated IP (Intellectual Property) core was designed to be easily deployed in RF systems. The IP core is an excitation mechanism built into the FPGA firmware to perform measurements of transfer functions (Baseband Network Analyzer, BBNA) and to inject band-limited noise to study Coupled-Bunch Instability (CBI), growth rates and longitudinal diffusion. The excitation core works in conjunction with the embedded acquisition IP core (acqCore) to record the excitation and the response of a system. The recorded data allow offline computing of the transfer function.

LLRF Python framework

To automate the commissioning of the LLRF systems of CERN accelerators, a Python framework has been developed. For optimal, clean software structure, the base LLRF project implements generic functionalities, while accelerator-specific solutions have been realised in separate, derived frameworks: LHC-LLRF and SPS-LLRF.

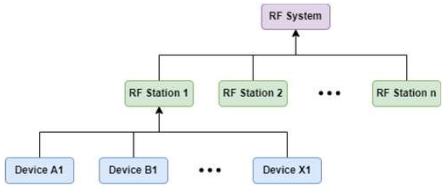

Figure 1 – Structural elements of the framework.

The basic element of the framework is a Python model of a *device*. Types of devices reflect the ones found in the hardware, e.g. cavity controller, timing unit, or RF feedback. Devices associated with each RF line are assembled into *RF stations*, which in turn constitute the complete model of an *RF system* of the given machine. The idea is depicted in Fig. 1.

Devices in the framework communicate with their hardware counterparts, permitting parameter read and write access. As shown in Fig. 2, the IO communication is facilitated by several software layers implemented in C++, Java, and Python. The *Controls Middleware* (CMW) and *Java API for Parameter Control* (JAPC) are standard layers of the CERN's control system, managing the access to hardware parameters in a role-based fashion [1, 2]. Next, JAPC is wrapped by libraries: *PyJapc* and *NiceJapc* to provide a user-friendly Python API.

Figure 2 – Software layers used in I/O communication with the hardware.

Additionally, the framework defines a functional element referred to as *routine*, which allows running commissioning procedures on any RF line in a standardised fashion. Among other functionalisies, the routines provide a safe execution *context*, which automates backing-up and restoration of the system state also in case of unexpected interruptions.

LHC setting-up tools

The LLRF system of the LHC consists of 16 identical RF lines, distributed over 4 RF cryomodules (2 per beam). The system is modelled in a dedicated sub-framework, which implements a well-defined sequence of routines to optimise a number of parameters during the commissioning procedure. Originally implemented in Matlab, the routines were translated to Python for compatibility with the CERN's software environment [3, 4].

The setting up starts with calibrating RF set point, feedback, and modulator parameters, which is followed by determining klystron saturation curves and calibration of the tuner. Many of the scripts perform sweeps through multiple values of a parameter to determine the optima to be used in operation. Two examples of such parameter scans are shown in Fig. 3.

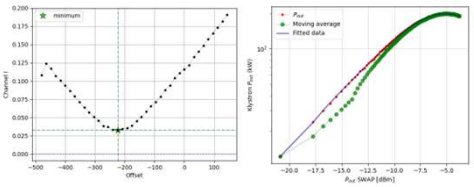

Figure 3 – Examples of parameter scans: RF modulator offset calibration (left) and klystron saturation curve measurement (right).

A number of parameters are also calibrated using transfer function (TF) fitting. The scripts record the response of the system to the injected excitation (noise) signal and calculate the frequency response. Next, an analytical model of the TF is fitted (see Fig. 4) to precisely determine values of parameters such as cavity detuning and quality factor, gains and delays of the feedback, as well as klystron characteristics.

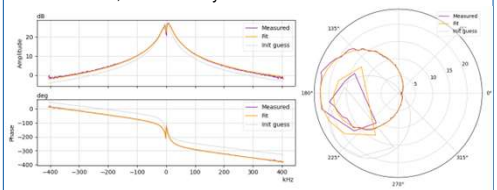

Figure 4 – Example of open-loop transfer function fitting results, shown on a Bode plot (left) and polar plot (right).

SPS setting-up tools

The LLRF system of the SPS has undergone a major upgrade during the recent Long Shutdown (2018-21) [5]. The Python setting-up tools for the SPS have played an important role in the development of the new electronics, helping with testing the system and tracking down any issues.

The central part of each of the 6 RF lines of the SPS is the cavity controller, which implements the feedback loop around each cavity, in particular the One Turn Feedback (OTFB).

Taking advantage of the new excitation IP core, the scripts facilitate the characterisation of various elements of the system using the frequency sweep technique (single-tone excitation). This allows for precise measurements in both wide-band and narrow-band settings with adjustable resolution. Examples are shown in Fig. 5.

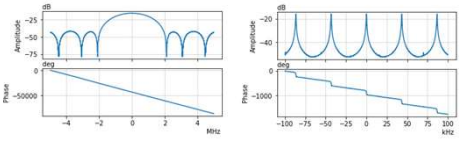

Figure 5 – Frequency characteristics of OTFB elements: cavity filter (left) and comb filter (right).

In addition to frequency scans, the SPS-LLRF sub-framework implements routines to set up several elements of the system. This includes calibration of system's delays as well as optimisation of cavity voltage and parameters of the amplifier linearisation polar loop (as shown in Fig. 6).

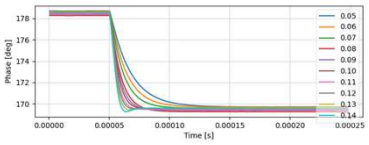

Figure 6 – Calibration of the gain of the polar loop's phase loop.

The SPS setting-up tools are still under development. The objective for the future work is to establish a series of routines which automate as much as possible the commissioning process of the LLRF system of the SPS, just like it is done for the LHC.

Firmware / IP Core

The excitation IP core was designed to be easily integrated in Xilinx FPGA firmware thanks to the universal AXI interfaces who allow a straightforward integration into projects.

To satisfy all requirements from LLRF system, the excitation can be used in three distinct modes:

The **Mode T** (for Tone) uses sinewave excitation. It allows measurement of transfer functions with single-tone excitation (sinewave) at a series of discrete frequencies and acquiring at another node in synchronism. This is useful if the response has a series of narrow resonances, such as the LHC or SPS One Turn Feedback.

The **Mode B** (for Buffer) uses buffer excitation. The buffer can be loaded with any data, such as white noise, band-limited noise or a step function. It is used for:

1. Measurement of transfer functions by injecting (white) noise at one node and acquiring at another node in synchronism. This is the simplest application. It is efficient if the response does not show narrow resonances. Example: The LHC RF feedback
2. Step responses by injecting a step excitation at one point and measure the corresponding response at another node. Example: The LHC Polar Loops

The **Mode BT** is a combination of the mode B and mode T. The applications are:

1. Measurement of transfer functions with narrow-band noise around a given revolution frequency harmonic and acquiring at another node in synchronism. This is useful if the response has a series of narrow resonances, located on the revolution harmonics, such as the LHC or SPS One Turn Feedback.
2. Excitation with narrow-band noise around a given revolution frequency harmonic for diffusion studies.
3. Excitation with narrow-band noise around a given revolution frequency harmonic and acquisitions at another node in synchronism for measurement of Coupled Bunch Instability growth rate.

Block diagram

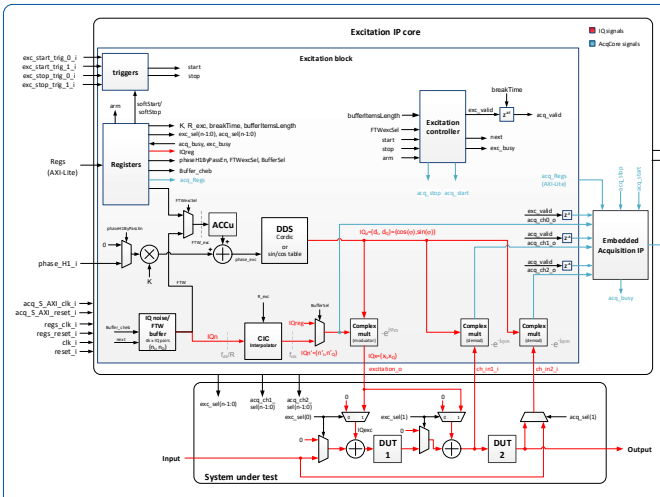

Figure 7 – excitation IP core architecture block diagram

*dominika.dlugosz@cern.ch

References

[1] A. Dworak, "The new CERN Controls Middleware", 2012 J. Phys.: Conf. Ser. 396 012017
[2] V. Baggiolini, "JAPC-the java API for parameter control", ICALEPCS2005, Geneva, Switzerland
[3] J. Banjac, "Developing Expert Tools for the LHC", CERN-THESIS-2017-182 (2017)
[4] D. Van Winkle et al., "Commissioning of the LHC Low Level RF System Remote Configuration Tools", International Particle Accelerator Conference 2010 Proceedings: Kyoto, Japan. 2010.
[5] G. Hagmann et al., "The CERN SPS Low Level RF upgrade Project", in Proc. 10th Int. Particle Accelerator Conf. (IPAC'19), Melbourne, Australia, May 2019, pp. 4005-4008. doi:10.18429/JACoW-IPAC2019-THPRB082