ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA034

SOFTWARE TOOLS FOR HARDWARE ELLIPTICAL CAVITY SIMULATOR MANAGEMENT AND CONFIGURATION

K. Klys, W. Cichalewski *, TUL, DMCS, Łódź, Poland

Abstract

The European Spallation Source (ESS) is currently in the middle of its construction phase. This facility linear accelerator consists of different sections. Superconducting part of this linac will be equipped with spokes and elliptical cavities (like M-Beta and H-Beta types). Various ESS linac components will be delivered by different in-kind partners from Europe. In order to provide a reliable development and evaluation platform hardware-based electronic cavity simulator have been built. This solution is especially useful for Low Level Radio Frequency (LLRF) systems development and integration in case of limited access to real superconducting structures. This contribution presents software tools developed for efficient cavity simulator parameters configuration and management. Solutions based on Python and EPICS framework are presented. Tool adaptation to ESS proposed E3 framework and experience from cavity simulator operation are also discussed.

INTRODUCTION

The ESS will be the most powerful neutron source on earth. In this facility, the spallation will be used to produce free neutrons. The tungsten target will be hit with protons with kinetic energy of 2.5 GeV. This leads to generation of neutrons' pulses. The superconducting part, which accelerates particles, will contain 120 cavities (84 H-Beta, 36 M-Beta) and 120 klystrons, operating at 704.42 MHz [1]. For each cavity the electrical field with appropriate gradient and frequency must be delivered. Its parameters (amplitude and phase) are controlled with LLRF control system [2].

The cavity simulator is a device which emulates the behaviour of superconducting High and Medium Beta cavities and klystrons basing on signals received from LLRF control system. It was created to minimize the risk of tests and measurements conducted on real facilities using LLRF system under development. In order to provide reliable results of simulations, the cavity simulator reflects phenomena like Lorentz force detuning, piezo compensation, beam loading π-modes, mechanical modes, amplifier non-linearity and others .

The device is composed of high performance Field-Programmable Gate Array (FPGA) evaluation board with data converters, Digital to Analog/Analog to Digital Converters (DAC/ADC) modules and specially designed Radio Frequency (RF) front-end [3]. The simulator can be controlled remotely with commands sent via Ethernet network. Those messages are realised using Standard Commands for Programmable Instruments (SCPI) syntax.

* wcichal@dmcs.pl

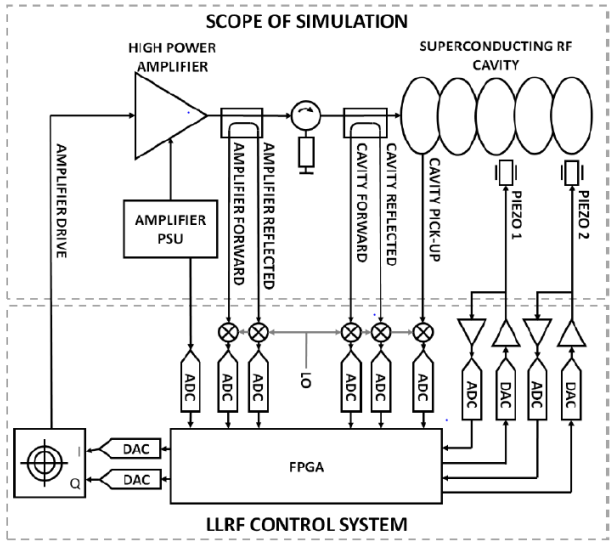

Figure 1: The cavity simulator area of simulation.

Figure 1 represents the scope of the cavity simulator's simulation and devices whose functioning is considered.

HARDWARE CAVITY SIMULATOR STRUCTURE AND COMMUNICATION

To simulate all mentioned effects and to cover the whole scope of cavities simulation, RF circuit has been designed (see Figure 2). The simulator's hardware is responsible for generating RF and base-band signals. The clock and refer-ence signals are there produced as well. The structure of hardware can be divided into several parts [4]:

- Data Conversion module,
- Down-conversion module,
- Piezo module,
- Reference Generation module,
- Local Oscillator (LO) Generation module,
- Power Supply module.

First module is used to convert different data types. It digitizes analog RF signals basing on down-conversion scheme. The generation of RF outputs is realised with vector modulator circuits. Down-conversion area decreases the frequency range of RF inputs to suitable level for ADCs. It makes use of active mixer circuit. Piezo part protects electronics against high voltage from piezo driver, lowering it 100 times. The module also emulates piezo element, functioning in 2 K

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA034

temperature as a sensor or actuator, depending on conditions. RF reference signal is distributed with Reference Generation section. It can produce RF signal itself or spread externally delivered. LO and clock signals are being created in LO Generation module. Power Supply conditions the voltage from off-the-rack AC-DC converter, it also adjusts the speed of fans accordingly to the power consumption [5].

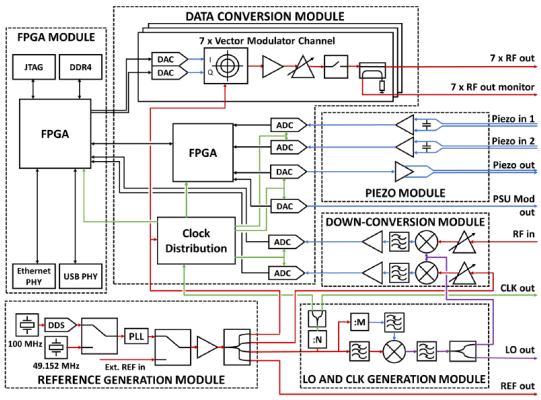

Figure 2: Hardware structure of the cavity simulator [5].

The cavity simulator can communicate with controlling PC using Ethernet network. Its IP address can be received through information sent via Universal Serial Bus (USB) to Universal Asynchronous Receiver-Transmitter (UART) converter. The raw Ethernet packages, containing the commands of special syntax, are sent with Transmission Control Protocol (TCP/IP) protocol.

PYTHON BASED SOLUTION

In order not to configure the cavity simulator with raw commands using Telnet and to facilitate monitoring of device's various parameters, the operator panel in Python language has been developed. It allows to read and modify all necessary parameters of the device. The Python has been chosen since it it is one of the EPICS supported language (PyEpics library). Furthermore, it possess modules to create graphical user part and to handle TCP/IP communication. The application is composed of 5 different tabs. The most crucial part of the created Graphical User Interface (GUI) is possibility of M-Beta and H-Beta cavities models' configuration. Each window provides options to read or write various cavity simulator settings. "Status" window is the first visible tab when program is being launched - Fig 3. It resembles the front panel of the cavity simulator. An operator can modify RF outputs signals' types and activate/deactivate them pressing button imitating the particular output connector.

Next tabs allow to alter H-Beta and M-Beta models' parameters (Q factor, detuning and gain), configure network parameters, set the emulated influence of piezo element on cavities model and send to the simulator any available command. To prevent user from transferring incorrect commands, one has been equipped with buttons, comboboxes and entries which facilitate the cavity simulator managing

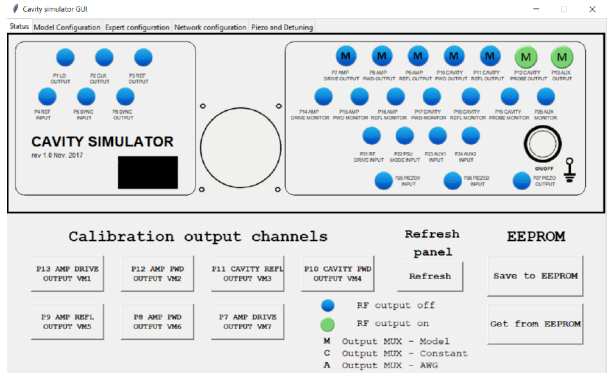

Figure 3: First tab of Python operator's panel.

and make the whole interface more transparent and clear. "Model Configuration" section (Fig 4.) provides entries and text fields to read and set model settings. For each mode of cavity model, there are factors that can be changed. Buttons to read configuration from file or to write it have been placed as well.

All commands generated with GUI are transferred to the cavity simulator using sockets and TCP protocol. To handle network exceptions, methods have been defined to inform an operator about any encountered problem. The verification mechanism, checking the status of the connection between the device and the client (in this case it is Python solution) has been implemented. It runs in the second thread, independently from interface layer and once per 5 minutes checks if it is possible to establish the connection sending command demanding an answer with temperature value from one of the sensors. These messages are presented in the appropriate window and saved to file with current data, hour and results.

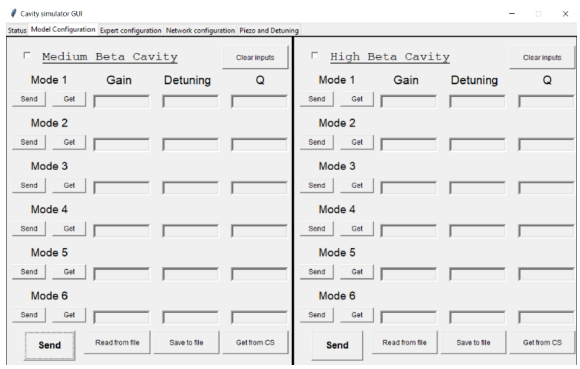

Figure 4: Model Configuration tab.

ESS E3 FRAMEWORK

The ESS EPICS Environment (E3) is a framework which enables compiling and launching EPICS applications. It has been developed in ESS facility for better managing and maintaining EPICS software and its dependencies.

E3 allows for different approaches of configuring modules and application for each user independently. It supports working with different software architecture and hardware.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA034

This environment ensures consistency of programs. It results in more efficient maintenance of different kinds of systems in the long term. E3 has been developed to reach coherence for various EPICS bases, modules and users.

On the other hand the framework has limited freedom. It provides the consistency for Integrate Control Systems (ICS) stakeholders in case of Input/Output Controllers' (IOCs) quality management. E3 is able to deal with different quality of open source modules, styles and codes. Additionally, the high level of the device integration results in avoiding low-level development.

From user point of view, E3 possesses several crucial features like built-in template builders to generate draft of IOC. An inexperienced operator can benefit from ready modules and pre-selection as well.

E3 BASED DEDICATED IOC FOR SIMULATOR MANAGEMENT

As an element of the distributed control system architecture of ESS facility, E3 EPICS based application has been created. It is composed of IOC with records allowing to configure all parameters of the cavity simulator. Each Process Variable (PV) corresponds to one setting of the device. In order to establish connection between the simulator and IOC, stream module has been employed. This is a communication based on byte stream over TCP/IP. It enables sending string messages to any machine if network parameters and protocol files, which determine the format of sent and received messages, are correctly configured.

In order to access the data stored in PVs, the OPerator Interface (OPI) has been developed in Control System Studio (CSS). This is an Eclipse based environment having features letting access live data directly from EPICS records. It was decided to use Best OPI Yet (BOY) mode since it ensures drag and drop approach with configurable elements, which can be connected to the data instantly. To manipulate declared components Jython (Python implementation on Java platform, it allows to import Java classes) language has been used. It provides possibility of modifying their features and behaviour with properly written script connected to the given interface element.

The structure of the tool is similar to Python based solution. There are separate windows which correspond to the given cavity simulator's scope of settings. Figure 5 presents "Status" window which provides buttons, diodes, switches and combo-boxes to read and change the cavity simulator's parameters like state of the RF outputs channels and the type of signal corresponding to them. The modification of one of the window's components cause change of the PV's value, which is sent to the cavity simulator as a proper string message. Depending on type of command (read or write) the response is obtained from the device, put into proper EPICS record and then transferred to the associated component on the interface.

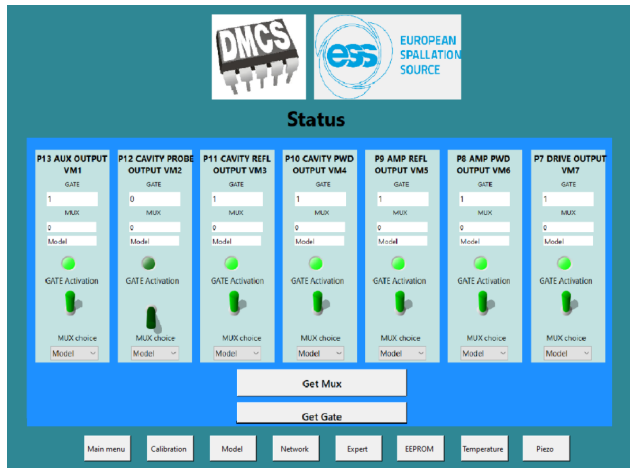

Figure 5: Panel of IOC for cavity simulator.

M- AND H-BETA CAVITIES CONFIGURATIONS EXAMPLES

The cavity is simulated as a parallel RLC circuit with the particular impedance. Its answer is emulated by Infinite Impulse Response (IIR) digital filter whose parameters can be altered by user in model configuration windows as a gain, Q and detuning [5]. To confirm the proper functioning of the both tools and the cavity simulator, the measurements with network analyser have been conducted. It was connected in the loop with the cavity probe and RF input. In this manner, the analyser was detecting any changes in the network caused by response of the device for model's modifications. Then, both H-Beta and M-Beta models were configured and tested. The parameters of cavities, which have been inserted into user interface, are the results of measurements carried out on real facilities in the temperature of 2 K (when resonant cavities are superconductive).

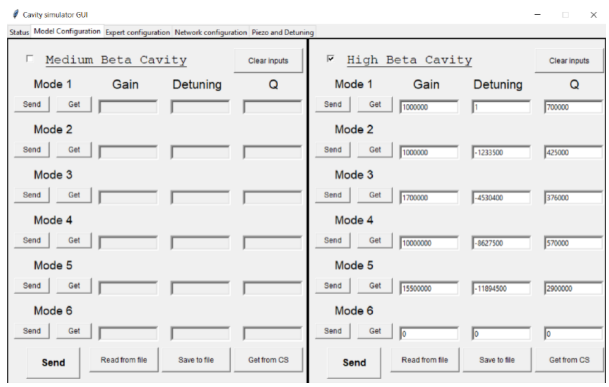

Figure 6: H-Beta cavity model configuration.

Figure 6 depicts fixed parameters of H-Beta cavity's model in Python tool. It can be seen how the organisation of the tab looks like and that some useful tools like saving/loading data to/from file have been provided. In Figure 7 the results of the tests with network analyser are presented. It can be observed that the response of the model is composed of 5 modes and one parasitic component which is an aliasing

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA034

effect of the first mode (between 3^{rd} and 4^{th} modes). The distances between modes correspond to values set in the GUI as detuning.

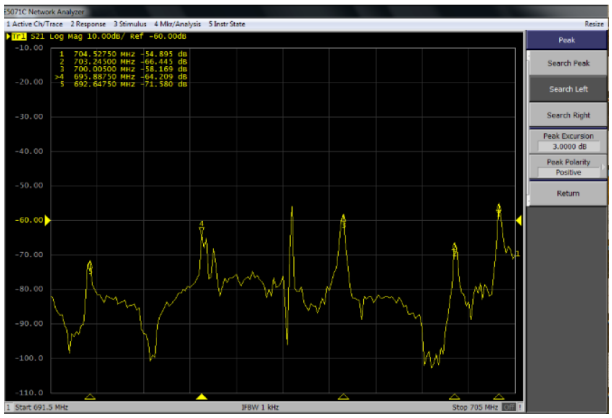

Figure 7: H-Beta cavity model response.

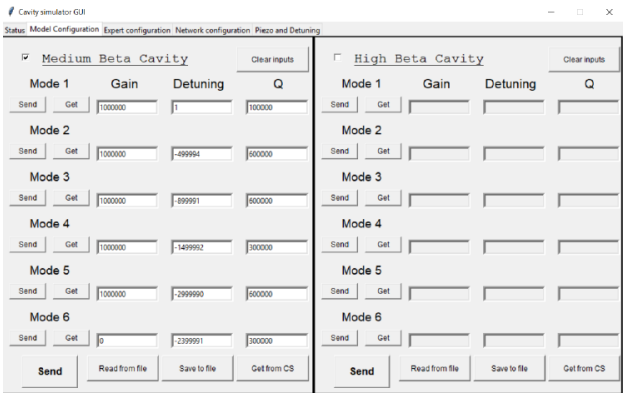

Figure 8: M-Beta cavity model configuration.

To move model parameters to the simulator, second section of the window was activated. All modes were configured and sent to the device (Figure 8). As a result, for M-Beta cavity model (Figure 9), it can be stated that the frequencies of visible modes are compliant with values from the Python tool. Detunings of modes are equal to values shown in entries.

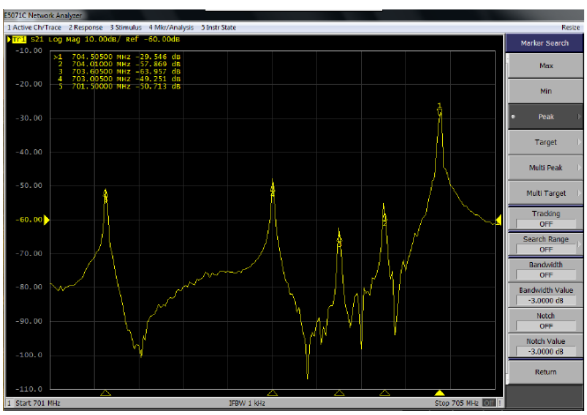

Figure 9: M-Beta cavity model response.

CONCLUSION

In this paper, software tools for hardware elliptical cavity simulator management and configuration are described. The overview of Python tool and E3 based solution with their features is presented. Many test have been performed to confirm that both GUIs are fully functional and they can not only substitute raw commands sent via Telnet but also facilitate access to the parameters of the simulator, their modification and monitoring.

The software is still under development to become a part of LLRF control system in ESS project and to fulfil all given requirements. Next step for IOC application will be associated with archiving data stored in records in case of the device failure. For Python solution it is planned to use test automation framework to create procedure for acceptance testing. Thanks to that, after installation of cavity simulator an operator could verify if it has been performed correctly.

ACKNOWLEDGMENTS

This work has been partially supported by the Polish Ministry of Science and Higher Education, decision number DIR/WK/2016/2017/03-1.

REFERENCES

[1] M. Aberg *et al.*, in *ESS technical Design Report*, Lund, Sweden, Apr. 2013.

[2] F. Kristensen, in *LLRF control system for ESS - Specification*, Lund, Sweden, Nov. 2013.

[3] M. Grzegrzółka, K. Czuba, and I. Rutkowski, "Rf front-end for cavity simulator for the european spallation source," in *22nd International Microwave and Radar Conference 2018 (MIKON)*, Poznan, Poland, May 2018, pp. 388–389.

[4] M. Grzegrzółka, K. Czuba, and I. Rutkowski, "Concept of a cavity simulator for the european spallation source," in *Proc. of ICALEPCS17*, Barcelona, Spain, 2017, paper THPHA123.

[5] M. Grzegrzółka, K. Czuba, and I. Rutkowski, "Cavity simulator for european spallation source," *IEEE Transactions on Nuclear Science*, vol. PP, pp. 1–1, Jan. 2019.