Migrating the LLRF controls from VME to \( \mu \)TCA hardware

O. Hensler, H. Schlarb, V. Ayvazyan, C. Schmidt, W. Jalmuzna
DESY (Hamburg Germany)

The Free electron LASer in Hamburg (FLASH)

FLASH the Free electron laser (FEL) in Hamburg at DESY (Germany) produces laser light of short wavelengths from the extreme ultraviolet down to soft X-rays. The reachd peak brilliance is one billion times more intense that that of the best synchrotron light sources today. FLASH is a high-gain FEL, which achieves laser amplification and saturation within a single pass of the electron bunch through an undulator. The requirements for amplitude and phase stability to achieve lasing are very high, so precise LLRF regulation is needed.

Introduction

In order to operate the complete LLRF hardware and firmware at FLASH and XFEL facilities successfully, a dedicated control system is required. The software stack, based on the DOOCS control system, from the low level firmware interface, front-end server and automation, up to high level calibration procedures and mass data storage will be presented. Additional attention will be given to the transition from VME to \( \mu \)TCA hardware.

The concept to automate the RF is based on a simple Finite State Machine FSM approach. The main purpose are to simplify the switching on/off procedure and faster recovery from trip.

This FSM is realised in the standard DOOCS server framework with the addition of the DOOCSFSM library, which provides simple classes for monitoring float values, recover set-values or resetting interlocks. The FSM is the central server for the automation, it starts up or switches off the whole RF. All actions in other servers are triggered by the FSM. Giving the operator one central location to look for the status or problems of the RF system.

The FSM runs with a repetition rate of 2Hz, checking several things, like interlocks, set-values or resetting interlocks. The FSM is the central server for the automation, it tries to recover the RF system. Tripaction() function is triggered to bring the system to a save condition. Then the FSM starts up or switches off the whole RF. All actions in other server are triggered by the FSM.

This FSM is realized in the standard DOOCS server framework with the addition of the DOOCSFSM library, which provides simple classes for monitoring float values, recover set-values or resetting interlocks. In case of a problem in a state, the so called tripaction() function is triggered to bring the system to a save condition. Then the FSM tries to recover the RF system. 2010-04-13

This two pictures are showing the \( \mu \)TCA test setup for ACC1 and the JDDD control panel, which is control automatically by reading out the shelf via IPMI.