ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP10

UPDATE ON THE STATUS OF THE FLUTE CONTROL SYSTEM

W. Mexner*, E. Blomley, E. Bründermann, C. Fehlinger, A.-S. Müller, R. Ruprecht,
T. Schmelzer, M. Schuh, N.-J. Smale, Karlsruhe Institute of Technology, Karlsruhe, Germany
S. Marsching, aquenos GmbH, Baden-Baden, Germany
I. Kriznar, Cosylab, Ljubljana, Slovenia

Abstract

The first phase of FLUTE, a new linac based test facility and THz source, is currently being commissioned at the Karlsruhe Institute of Technology (KIT). It consist of an RF photo gun and a traveling wave linac accelerating electrons to beam energies of 40 to 50 MeV. The control system is based on a virtualized infrastructure running Ubuntu Linux and Linux KVM. As base for the SCADA system we use EPICS 3.15 with Control System Studio (CSS) for the GUI. The long term data storage is provided by a Cassandra NoSQL database. This contribution will present the architecture and the current status of the FLUTE control system.

INTRODUCTION

FLUTE [1] is a new R&D linac accelerator (see Fig. 1) offering beam energies of 7 to 41 MeV for the development of accelerator technology, new diagnostics and instrumentation for fs bunches. It will be used as a test facility for the study of bunch compression with all related effects and instabilities like space charge, coherent synchrotron radiation (CSR) as well as the different generation mechanisms for coherent THz radiation. Furthermore it will serve as a broad band accelerator-based source for ultra-short and intensive THz pulses, e.g. for time- & frequency-domain spectroscopy of kinetic processes.

CONTROL SYSTEM OVERVIEW

The design of the FLUTE control system [2] is based on EPICS 3.15 with Control System Studio (CSS) as the main operators interface to the control system.

The DAQ has been driven by the demand to make systematic studies of the beam compression and the generation of THz radiation. The pulse synchronous (10 Hz) data-acquisition based on a Cassandra NoSQL database has to record for each pulse diagnostic information about RF pulses, laser pulses, pulse charge and of course the overall accelerator settings.

* wolfgang.mexner@kit.edu

Operator GUI Concept

The top level panels of the operator GUI have an synaptic approach. A 3D model of the Accelerator is used to for easy orientation (see Fig. 2). An error is visually connected to the device and to the position along the machine. The corresponding control panel for the device can be opened by a direct link on the panel. In addition all device panels are included in a list, which can be used to navigate to the devices, grouped by their task. As the accelerator is in its first phase the starting overview panel shows the injector section. For each of the next sections there are also synaptic overview panels planned as well as for the complete machine.

Stepper Motor Control

The FLUTE accelerator and the KARA Beamlines are using the same inhouse developed stepper motor control and driver concept. For motion control, we use the OMS (Pro-Dex) 5 axis Ethernet based MAXNET controllers and as 2 phase motor drivers we use the BCD130.x family of MIDDEX electronic allowing up to 12800 micro steps per revolution. EPICS hardware integration was done with the Pro-dex OMS asyn motor base axis support. For axes using an encoder, we experienced the problem that the motors would sometimes move randomly. We could solve these problems by applying a patch that synchronizes the motor with the encoder position before every movement.

Timing System

The timing system for FLUTE is based on the hardware components from Micro-Research Finland [3]. We use the VME form-factor for both the event generator (EVG) and the majority of event receivers (EVRs). However, we do not use the VMEbus for controlling these boards. Instead, we use their Ethernet interface to control them using a UDP/IP-based protocol.

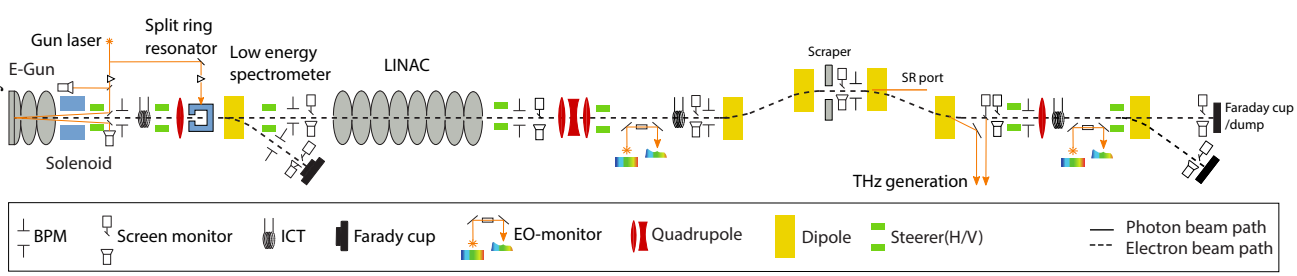

Figure 1: Scheme of the FLUTE accelerator with all installed and planned components [1].

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP10

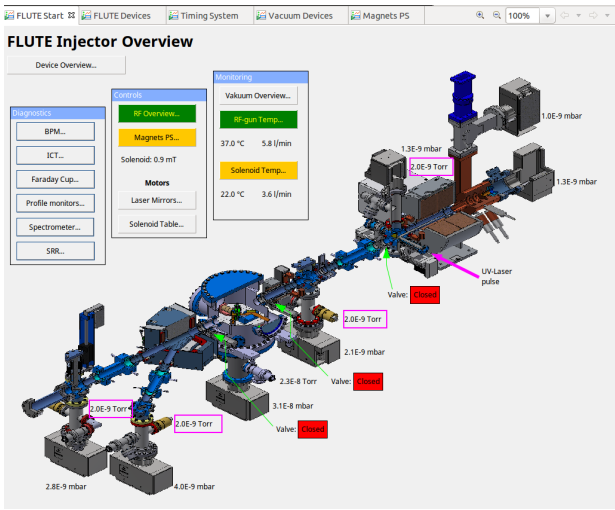

Figure 2: CSS Synaptic Operator view.

As the traditional EPICS device support for MRF devices [4] did not support this way of controlling the devices and its architecture did not include a layer for implementing different ways of transport, we developed our own EPICS device support [5] which includes such an abstraction layer.

Thanks to the flexibility of this new device support, we can essentially use the same code for controlling MRF boards over the network and controlling them locally over PCIe, which we need for some µTCA EVRs. The same device support is also used for the timing system of KIT's KARA facility, where it has proven to work reliably as well.

RF System

The RF uses a pulse type Klystron to produce a 4 µs long 3 GHz burst of 45 MW of power. This power is split between the E-Gun and the LINAC. The power supplies, for charging the Pulse Forming Network (PFN) and controlling the klystron coils are all slow control using PLCs, which are fully integrated into EPICS and CSS. However, MTCA based controllers such as the laser synchronization feedback loop, Low Level Radio Frequency feedback loop (LLRF) , and BPM electronics (readout via MTCA) which were developed for the FEL at DESY are not yet fully integrated to EPICS. As a stepping stone to full integration, these devices are presently used with DOOCS and JDDD installed on the devices. For standard configuration settings a remote desktop is used to access the JDDD panels. For the 'daily operator use', the more used configuration parameters have been integrated to EPICS and CSS using either gateways or Chimera TK adapter (the latter developed in cooperation by aquenos, DESY, HZDR and TUD). These methods are also used to transfer fast data capture to the operator display and database for storage. This is also true for the timing distribution system which, because of historical reasons, have a mix of the MRF VME and the X2 MTCA timing systems.

The software of the LLRF systems is based on the ChimeraTK framework [6]. Thanks to the use of this framework, essentially the same software can be used in control-system environments based on DOOCS, EPICS, or OPC-UA, making it possible to use the same hardware and software in a number of facilities like ELBE, the European XFEL, FLASH, FLUTE, and TARLA with only minor tweaks. This has helped to reduce the man-power needed to commission the system at FLUTE significantly.

SERVER INFRASTRUCTURE

Most services needed for FLUTE run within virtual machines on a virtualized infrastructure. We use the Linux Kernel-based Virtual Machine (KVM) [7] for hosting both Linux and Windows virtual machines (VMs). We use Ubuntu 14.04 LTS as the operating system for the VM hosts and manage the VMs using libvirt [8]. To ensure that basic services like network gateway (firewall) and DNS are always available, these essential services are provided by multiple VMs, hosted on different host systems. These host systems use a dedicated direct link for heartbeats, so that high-availability management is not affected by disruptions of the networking service.

This combination has proved very reliable: This virtualized infrastructure (based on two host machines) has been running for four years while needing virtually zero maintenance. In the near future, we plan to migrate the system to Ubuntu 18.04 LTS so that we can continue to apply critical security updates past April 2019.

We assessed that virtualization is not the best option for services with significant demands on the CPU or memory. For such services, sharing hardware resources does not make sense because it would only increase the demands on the virtualization hosts' hardware, while not profiting from the sharing of resources (which is the primary objective of virtualization in the first place).

For this reason, we also use three Supermicro Microcloud chassis [9], each containing 12 nodes. Each node is equipped with a Xeon E3 processor, 32 GBs of memory and two hard-disk drives. We use these nodes to run services like the Cassandra PV Archiver [10] that provides long-term storage of process data and the service for processing camera images.

NETWORK INFRASTRUCTURE

In order to improve the reliability and security of the network services, we use a dedicated network for all measurement and control equipment of the accelerator. The network is completely separated from the institute's office network. The VM hosts are based in both the dedicated accelerator network and the office network, so that selected VMs can provide services for transferring data between the two networks in a secure, well-controlled fashion. One example of such a service is the EPICS PV Gateway [11] that enables users to access EPICS process variables from their office computers, but limits all access to be read-only, thus ensuring no one can accidentally interfere with the operation of the accelerator.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP10

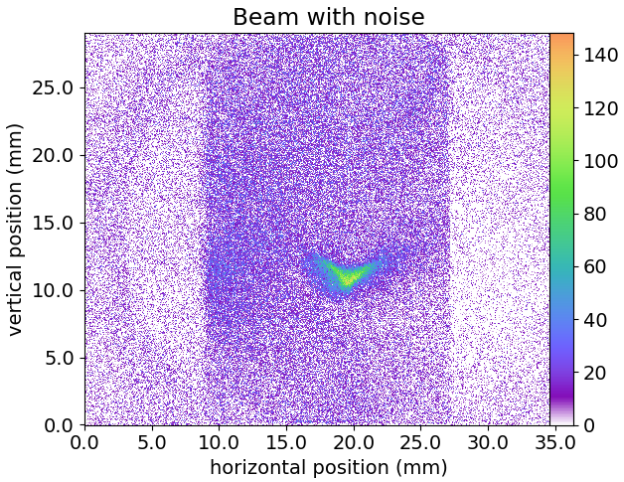

Figure 3: First FLUTE Beam.

CSS CLIENT DEPLOYMENT

The deployment of CSS consists of two separate, but closely related tasks: First, deploying CSS itself (the actual software along with its specific configuration for the environment), and second deploying the panels used inside CSS. The first task could be accomplished through means of the operating system (e.g. packaging and deploying CSS as a Debian package), but the panels change too frequently to make this a feasible approach for them. In addition to that, due to its heritage from Eclipse, CSS works around the concept of a "workspace" and no two instances using the same workspace can run at the same time.

These issues caused us to develop a couple of Python scripts that take care of installing or updating CSS, preparing a fresh workspace on each start, and cleaning up this workspace after CSS is closed. These scripts were originally developed for use at KIT's KARA facility, but they were quickly and easily adapted for use at FLUTE. These scripts are distributed alongside the panels as part of a Subversion repository. When the local working copy of the repository is updated, both the panels and the scripts are updated and thus the version of the panels and the version of CSS that is automatically installed by the scripts always match. By creating a fresh workspace on every start, the operators always start with the same, familiar view (Fig. 2), regardless of which panels have been opened or closed previously. This was a crucial factor to improve the acceptance of CSS as the user interface for operators.

SUMMARY AND OUTLOOK

In May 2018 first electron beam was produced and observed by different diagnostic systems such as the screen monitors (see Fig. 3), beam position monitors, integrated current transformer, and Faraday cup. Based on the recorded data the different systems can be calibrated which is needed for the full integration into the control system. At FLUTE the first user's experiment is foreseen in 2019 within the ARIES Transnational Access programme funded from the European Union's Horizon 2020 R&I programme under GA No 73 08 71.

ACKNOWLEDGEMENT

The authors would like to thank DESY's MSK group and KIT's Institute for Data Processing and Electronics for their support regarding the RF system. They would also like to thank PSI's diagnostics group for their help regarding beam diagnostics.

REFERENCES

[1] A. Malygin *et al.*, IPAC'18, Vancouver, 2018, TPHMF068. A. Malygin, A. Bernhard, E. Bründermann, A. Böhm, S. Funkner, I. Križnar, *et al.*, A. Malygin *et al.*, "Commissioning Status of FLUTE", in *Proc. IPAC'18*, Vancouver, BC, Canada, Apr/May 2018, pp. 4229–4231. doi:10.18429/JACoW-IPAC2018-THPMF068

[2] S. Marsching *et al.*, "Status of the FLUTE Control System", in *Proc. PCaPAC'14*, Karlsruhe, Germany, 2014, paper WPO013.

[3] Micro-Research Finland, http://www.mrf.fi/

[4] EPICS mrfioc2 device support, http://epics.sourceforge.net/mrfioc2/

[5] EPICS MRF device support provided by KIT, https://github.com/kit-ibpt/epics-mrf

[6] M. Killenberg *et al.*, "Abstracted Hardware and Middleware Access in Control Applications", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017. doi:10.18429/JACoW-ICALEPCS2017-TUPHA178

[7] Linux Kernel-based Virtual Machine, https://www.linux-kvm.org/

[8] libvirt Virtualization API, https://libvirt.org/

[9] Supermicro MicroCloud, https://www.supermicro.com/products/nfo/MicroCloud.cfm

[10] Cassandra PV Archiver, https://oss.aquenos.com/cassandra-pv-archiver/

[11] EPICS PV Gateway, https://epics.anl.gov/extensions/gateway/index.php