

FEC Software Design

General

FEC software supporting EBIS will largely make use of proven hardware and software components, originally designed for use in other parts of the accelerator complex. In some cases, existing software modules will need to be adapted to EBIS requirements. Those cases will be described in later sections of this document.

The platform for deploying FEC software will either be a VMEbus chassis running VxWorks, or PC running LINUX. In the latter case, the FEC software will be referred to as a “Manager”. In the former case, all components will be standard control system configurations, including VxWorks kernel, FEC “core software”, VMEbus chassis, CPU module, utility module, and other VMEbus modules. Event, RTDL, reset and network links will be extensions of existing controls infrastructure. Standard control system files servers will be used to store EBIS system configurations, executable modules, and provide other general file services required by FEC software.

The main activities associated with EBIS FEC software include:

- 1) Assigning addresses, channels, functional names, alarm thresholds, and other configuration tasks.
- 2) Creating and configuring basic “engineering” interfaces (pet pages) to support system configuration and checkout, and any system control/monitoring not supported by the EBIS application interface.
- 3) Creating and configuring loggers to record trending information, and data for correlation.
- 4) System installation, integration, and checkout

Timing pulses

EBIS has unique requirements for generating and managing timing pulses, which cannot reasonably be supported using existing controls hardware or software. To address these requirements, the V233 (qfg) module will be adapted from its present role as an analog function generator into a new role as a 16-channel pulse generator. This adaptation will preserve the register interface. Therefore existing software could be used to support this new role. A shortcoming of the existing software is that it is not designed to provide independent control of each of the 16 bits as separate channels. The software supports parameterizing each function via time-value pairs, where the value contains all 16 bits, and using linear interpolation between each pair. The functionality of separating the 16 bits into separate channels can be implemented via a software layer “above” the existing qfg software, implemented as FEC software, or a LINUX-based manager, or a

console-based library. This should be an early effort, since the interface to the timing pulse generator affects the design of the EBIS application.

Vacuum

Vacuum software will include interfacing with gauge, ion pump controllers, and other RS-232 devices, and interfacing with valve control/interlock PLCs via Ethernet. The PLC logic will be developed by the Vacuum group. Booster/NSRL LINUX-based Manager software will be reconfigured and reused for this purpose. The configuration of the Turbo-Molecular Pumping Station (TMPS) may differ slightly from existing TMPS configurations, requiring minor adaptation of existing software.

Power Supplies

Power supply software falls into two categories, each of which has unique FEC software requirements

- 1) Power supplies requiring DC reference points within a PPM environment, e.g. beam line magnet power supplies.
- 2) Power supplies requiring two or more different DC levels within an EBIS cycle, also within a PPM environment.

Power supplies in category 1 will be supported with PSC VMEbus modules, and standard PSC FEC software. The PSC FEC software does not currently support PPM. PPM support will have to be added for EBIS, but may be added sooner for Booster applications.

Many of the power supply roles in category 2 have only two levels with an EBIS cycle, one level as ions are introduced to EBIS, and a second level as ions are extracted from EBIS. Other roles require up to about a dozen different levels during the EBIS cycle. In either case, qfg modules will be used. In the latter case, an important EBIS requirement is that changes are coordinated among separate qfg modules. To support this requirement, qfg modules will be enhanced to be sensitive to a dedicated event code to initiate activating new functions. This requires a minor qfg driver change.

In order to support 6 qfg modules per VME chassis, more profound, EBIS-specific enhancements will be required for the qfg driver and ADO software. VMEbus bandwidth prohibits transferring every ADC reading from 6 qfg modules during normal operation. However, calculations show that for each of up to 12 separate levels, 16 ADC readings can be transferred, and published either independently, or as an average. Complete data arrays could be provided as a special diagnostic mode, temporarily interrupting normal operation. This has not been empirically verified, but there is high confidence in this approach among system experts. The qfg ADO already supports framework for “types” of PSs with unique algorithms. Extending this framework to support EBIS-specific roles is straightforward.

Analog Signal acquisition

A few dozen slowly-changing analog readings need to be continuously acquired and logged. These signals will be interfaced via spare PSI ADC channels associated with qfg modules. Existing qfg software is capable of reporting individual ADC readings (acquired concurrently with PS ADC readings) and therefore can be used without modification. To support ease of management of these readings (e.g. to be able to apply names to individual readings), a simple manager interface may be added.

Beam Diagnostics

Software to support Beam Diagnostics falls into two categories:

- 1) Digital, analog, and timing I/O implemented via VMEbus modules.
- 2) Oscilloscope signals acquired via Ethernet, supported by VMEbus-based signal muxes.

In each case, existing controls software can be used to support EBIS requirements.

RF

RF controls for EBIS will be an adaptation of existing RF controls. Historically, the RF group has assumed responsibility for implementing and configuring RF FEC software with support from the controls group.