Document Title:	**On the Digitization of Analog Signals in the FAIR Accelerator Complex**
Description:	Detailed specification for the integration of time-domain digitizers with analog bandwidths and sampling frequencies ranging from DC to a few MHz or up to hundreds of MHz into the accelerator control system
Division/Organization:	FAIR
Field of application:	FAIR Project, existing GSI accelerator facility
Version	V 0.2

Abstract

This document describes the integration of time-domain digitizers with analog bandwidths and sampling frequencies ranging from a few MHz to hundreds of MHz. The primary aim is to provide a generic abstraction of the vendor-specific digitizer software interfaces, a limited range of generic data post-processing on the acquired data, and integration of these devices into the FAIR control systems by providing FESA standardised software interfaces.

Prepared by:	Checked by:	Approved by:
R. Steinhagen	FAIR-C2WG-ALL (e-group) R. Bär (CO) A. Bloch-Späth (OP) H. Bräuning (BI) S. Heymell (CO) T. Hoffmann (BI) S. Jülicher (CO) C. Omet (SIS-100 MP) D. Ondreka (SYS) V. Rapp (CO) H. Reeg (BI) A. Reiter (BI) P. Schütt (OP) M. Schwickert (BI) I. Lehmann (Exp. Link-Person) D. Severin (Exp. Link-Person)	R. Bär (Controls) M. Schwickert (BI) R. Steinhagen (FAIR Comm. & Control PL)

Document History:

Version	Prepared by:	Checked, Date	Released, Date	Comment
V 0.1	R. Steinhagen			Initial version based on preparatory work and discussions in the FAIR Commissioning and Control Working Group
V 0.1	R. Bär	2017-05-21		
V 0.2	R. Steinhagen			added comments from CSCO, prepared for first FC2WG circulation

Table of Contents

1 Purpose and Classification of the Document

The purpose of this document is summarise the functional requirements and required software interfaces for the digitization and acquisition of analog signals. The focus is put on time-domain type acquisitions with analog bandwidths and sampling frequencies ranging from DC to a few MHz or up to hundreds of MHz.

2 Scope of the technical System

The primary aim is to provide a generic abstraction of the vendor-specific digitizer software interfaces, a limited range of generic data post-processing on the acquired data, and integration of these devices into the FAIR control systems by providing FESA standardised software interfaces.

Two data acquisition modes shall be implemented and exposed via the FESA interface:

A.1) Streaming-mode acquisition of the sampled inputs at a medium raw acquisition rate (typically around 10-60 MS/s) while the data is merely being tagged and time-stamped by external triggers to decide which sub-chunk is exported to the user-level, or

A.2) (Rapid) block-mode acquisition, where a limited sequence of samples is acquired at the maximum raw digitizer sampling rate (typically a few hundred MHz) in response to an external trigger.

Based on the acquired data a limited set of post-processing of the raw data shall be provided:

B.1) scaling and shift of the raw measurement values using user-defined calibration constants,

B.2) re-alignment of time-base based on user-defined cable- and data-processing delays, as well as actual extraction event offset,

B.3) data aggregation and decimation (e.g. low-pass filtered and down-sampled) to user-defined rates, typically ranging from tens of kHz down to a few Hz,

B.4) perform a real-time Short-time Fourier Transform (STFT) on the data,

B.5) amplitude, phase and frequency detection of one input channel in relation to another reference input channel (i.e. performing digital band-pass filtering and I-Q demodulation on a combination of two input channels),

B.6) basic fitting module (initially for ‘B.4)’-type data) that implements a basic peak detection and chi-square type fit, and

B.7) comparison of the actual versus a user-defined target reference signal and issuing of a software-based interlock signal if the user-defined tolerance threshold is exceeded.

The different post-processing modules may be combined into a chain (e.g. first application of ‘B.1)’ → ‘B.5)’ → ‘B.3)’ → ‘B.6)’ → ‘B.7)’). Open-source signal processing and data fitting libraries shall be used for the post-processing wherever possible, notably the GNU-Radio[1] and ROOT[2] frame-works, in order to simplify further extensions, compactness, readability, re-usability, and maintainability of the implementation. The post-processing implies additional settings interfaces from the data supply to the digitizer front-end and data interfaces to user-level applications further outlined below.

[1] See www.gnuradio.org

[2] See https://root.cern.ch/

2.1 System Overview

Figures 1 provides an overview schematic of the targeted digitizer system integration architecture.

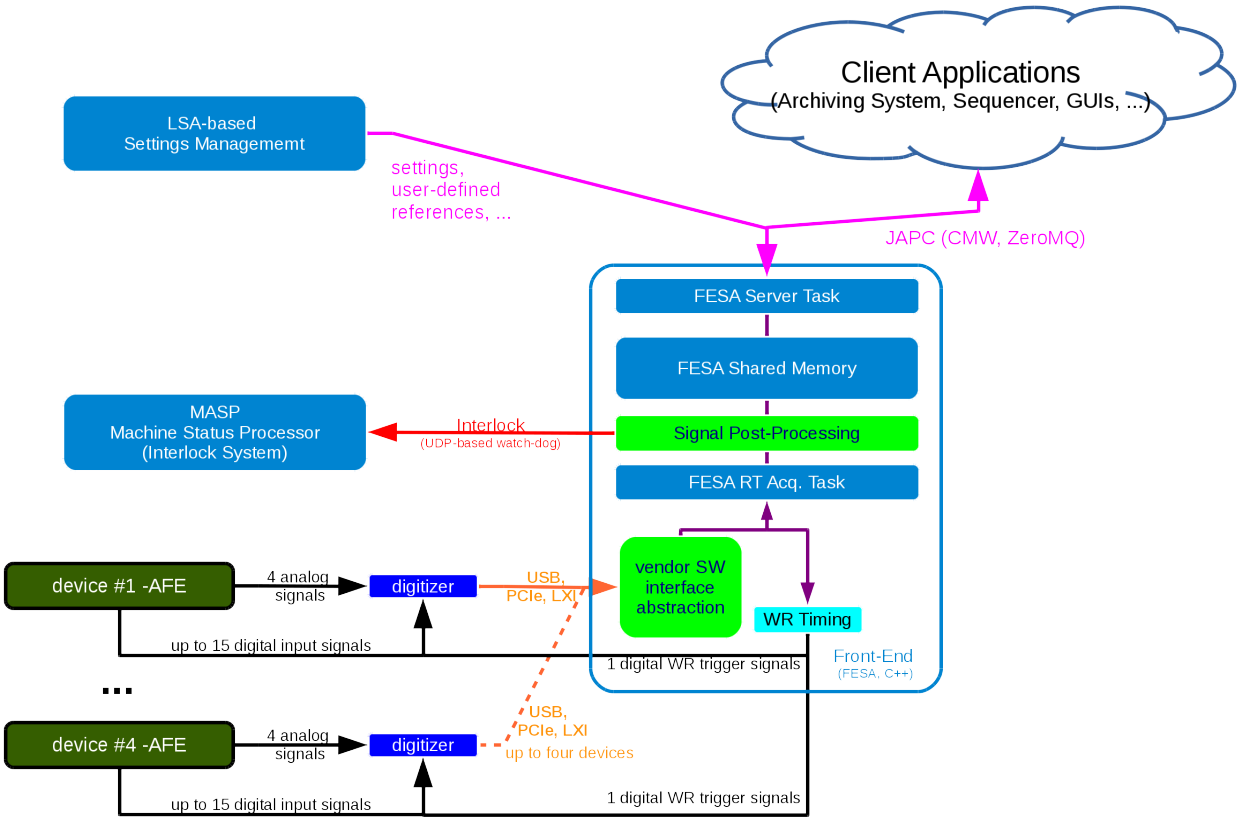


Figure 1: Digitizer System overview

The analog front-end (AFE) of the device is connected with up to four analog signals to the digitizer. The AFE is in the responsibility of the end-user equipment owner, and responsible for the coarse pre-conditioning of the analog signals to fit the digitizer input constraints (amplification, filtering, mixing, clamping, etc.). The digitizer's input amplitude may be fine adjusted within the digitizers capability.

Some of the targeted digitizers provide up to 16 digital external trigger ports, out of which one shall always be assumed to be connected to the front-end's timing receiver card, while the others may be connected to the user-defined device specific digital signals. The trigger source, logic levels, and trigger levels must be configurable via the front-end's software user interface.

Up to four digitizers may be controlled locally through a single Front-End Computer (FEC). The FECs primary task is to provide an abstraction layer for the vendor specific physical hardware (e.g. USB, PCIe, LXI, etc.), low-level driver software and external FESA interface to user-level applications (e.g. the Archiving System, Sequencer, GUIs, etc.). The abstraction is required to facilitate future easy follow-up of upgrades within the vendor specific product lines and/or integration of other vendor products with similar functionalities but different low-level driver or hardware interfaces. The abstraction interface shall be guided by the vendors native driver interface and extended by the device specific features where necessary. For devices that do not implement the given device feature, an empty implementation shall be used.

Besides static configurations that may set-up by local configuration files, the FEC and digitizer settings will be provided by the LSA-based settings management.

Based on the comparison of the actual signal versus a used-defined target reference signal, the FEC is required to issue a software-based interlock signal if the user-defined tolerance threshold is exceeded. Provisionally it can be foreseen that this interlock signal is based on a watch-dog-like mechanism of continuously sending a UDP stream containing the interlock state to the Machine Status Processor (MASP). The communication mechanism and module to the MASP will be provided by CSCO.

2.2 Primary Low-Level Data Acquisition Modes

The purpose of the digitization is to provide a generic facility wide continuous or repetitive monitoring of analog signals from a wide range of systems, and to monitor their performance during injection-, ramp-, fast- & slow-extraction and – if out of tolerance – to issue and transmit a signal to the interlock system. Figure 2 schematically illustrates the required top user-level measurements the low-level acquisitions need to provide data for.

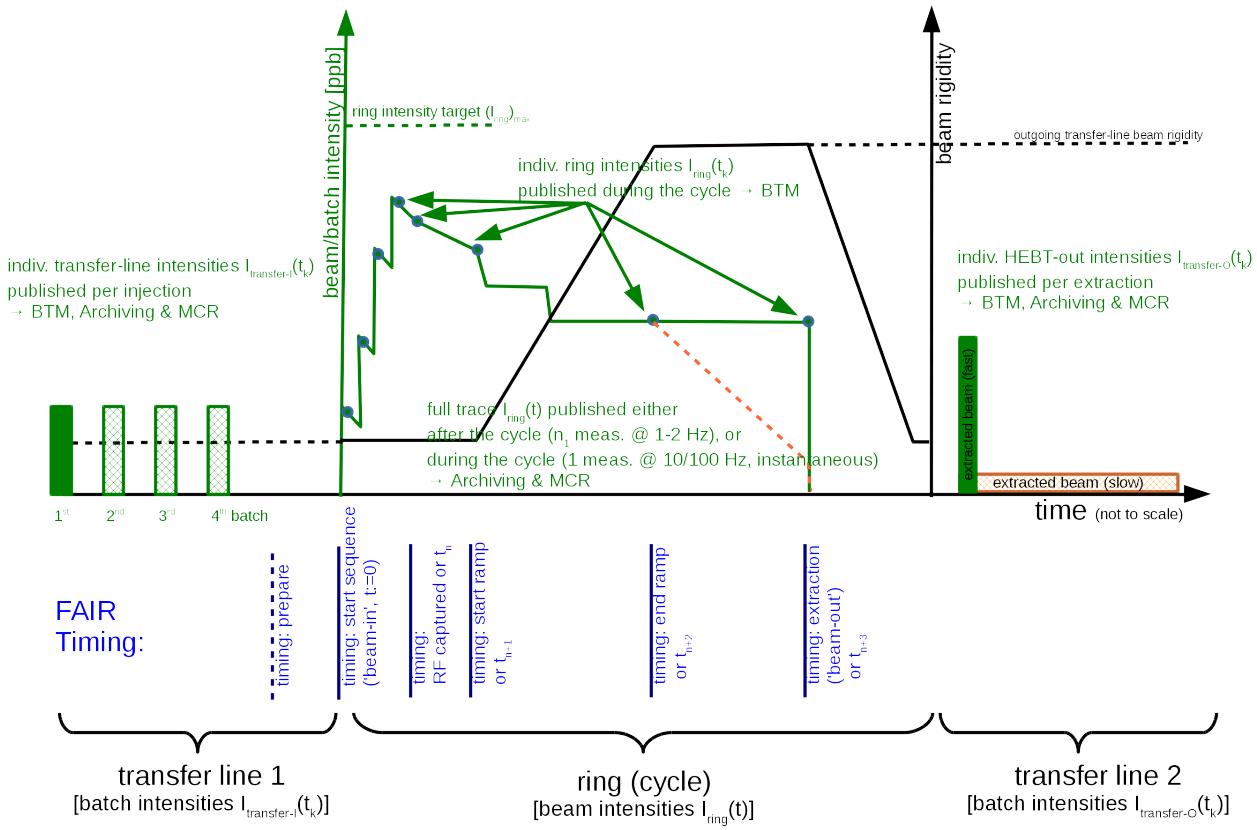


Figure 2: Schematic transfer line and ring-based data acquisition and synchronisation scheme. Here 'intensity' is used as an example but the signal can come from any current or voltage source.

Internally, two primary low-level raw data acquisition modes shall be provided (which also map directly to the typical low-level functionalities of nearly all digitizers). These modes are typically mutually exclusive and thus the FEC needs to provide only either one for a given digitizer setup:

2.3 Streaming-Mode Data Acquisition

In this mode, the digitizer data is passed continuously to the FEC, which enables long periods of data collection for slow(er) monitoring of systems (tens of MS/s). The external trigger shall be used to time-stamp and to align the time of the data stream to the FAIR-wide time-base and beam process or beam production chain structure.

Additional trigger may be used to extract smaller sub-portions of data based on the user-specified additional timing or external hardware triggers (e.g. at injections, extraction, or other user-defined trigger), or to monitor digital device process variables.

It may be necessary to configure the digitizer to internally aggregate the data (e.g. low-pass filter, computation of the minimum and maximum, etc.) and down-sample the raw acquisition rate to the rate that can be effectively streamed over the low-level hardware interface. The data should be transferred within the FEC to an internal sufficiently large circular buffer, to be used by the other post-processing modules or chains to compute their derived properties.

A schematic streaming-mode acquisition sequence is illustrated in Figure 3:

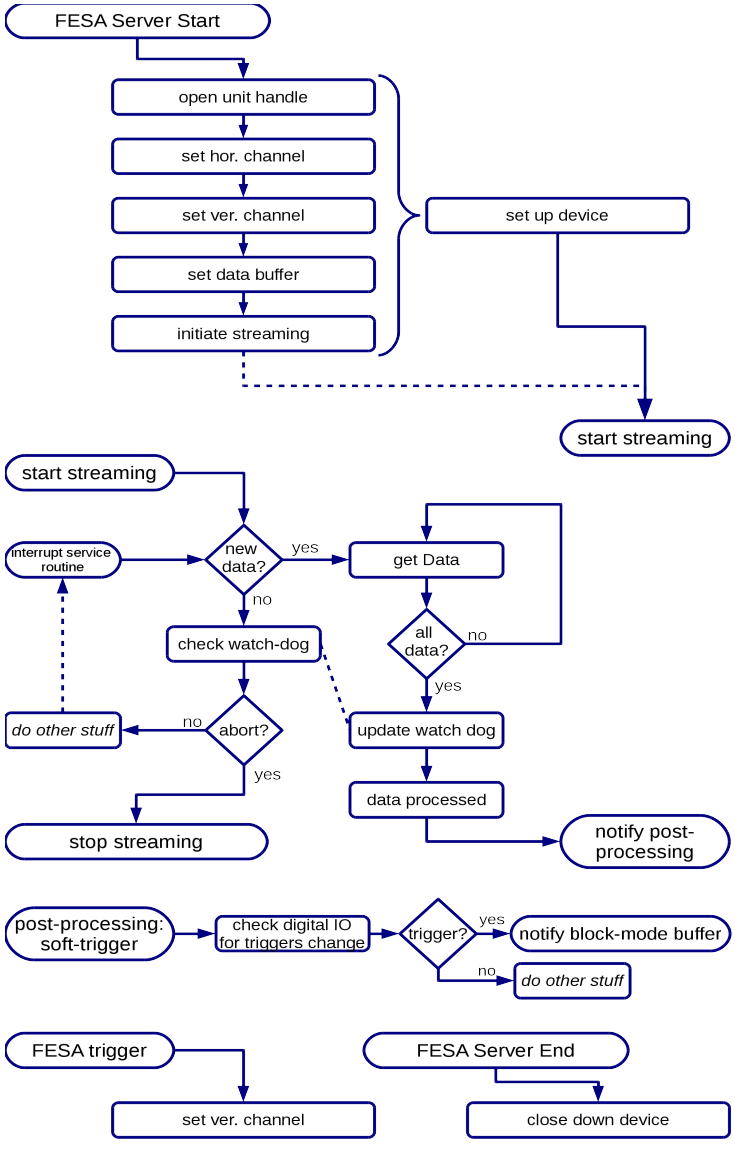


Figure 3: Schematic streaming-mode acquisition.

2.4 (Rapid) Block- or Sequence-Mode Data Acquisition

In this mode, the digitizer captures a limited amount of data samples at the user-defined sampling rate (often the digitizer's maximum) in response to a single or succession of several timing or external triggers. The mode requires that a digitizer-specific minimum delay between external triggers is respected and that the amount of requested data per trigger is constant. Provisionally it can be foreseen that the amount of samples is defined by the largest requested number and reduced in the post-processing for users that required less.

A schematic block-mode acquisition sequence is illustrated in Figure 4:

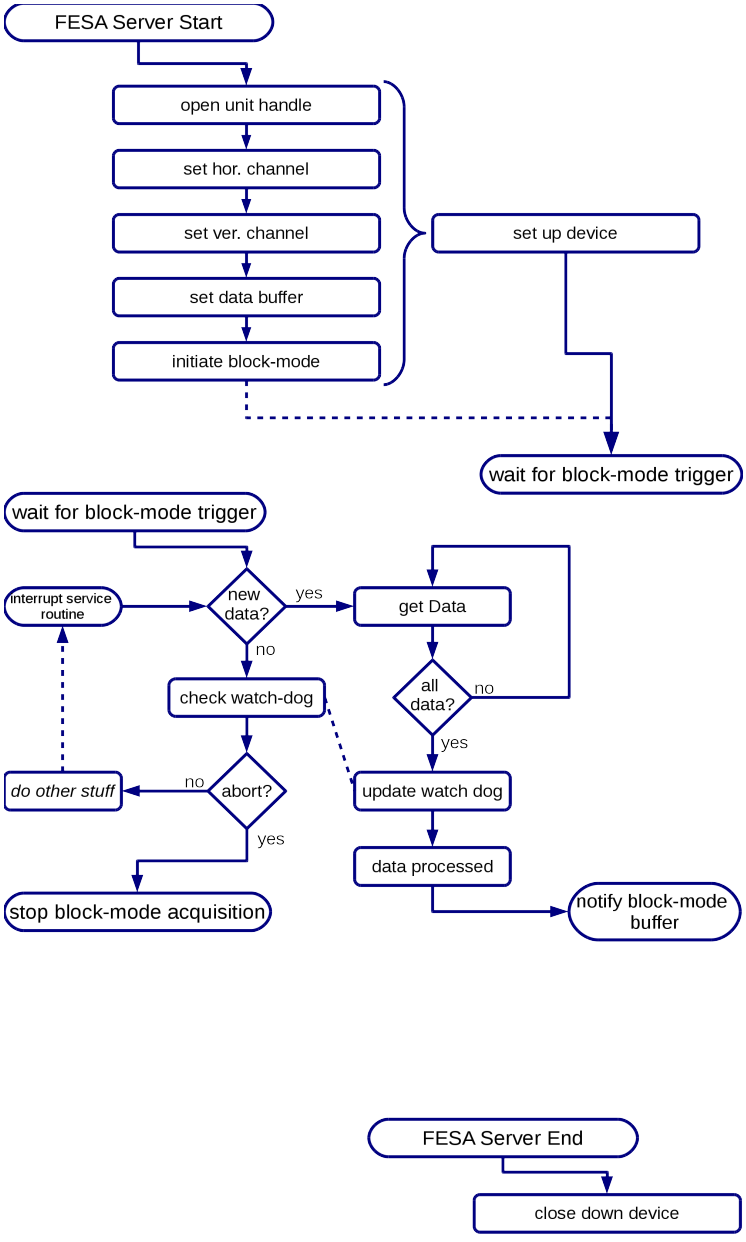


Figure 4: Schematic block-mode acquisition.

3 Required Post-Processing Modes

The acquired low-level digitizer signals require some further basic signal post-processing as an abstraction of the specific low-level measurement and to derive additional physical properties prior to being exported by the public FESA FEC interface. These signal post-processing steps shall be implemented as independent modules that may be combined and/or cascaded to more complex signal flow graphs.

The signal-flow graph for each digitizer input and given FESA setup may be considered as static, and could thus be set up, for example, during the FESA device initialisation. However, the module parameters are commonly multiplexed and shall be configurable during run-time. This implies additional settings interfaces detailed further below. An exemplary signal flow-graph is shown in Figure 5. Online flow-graph configuration or modification is not required and may be fixed at the FEC device start-up (e.g. via configuration file).

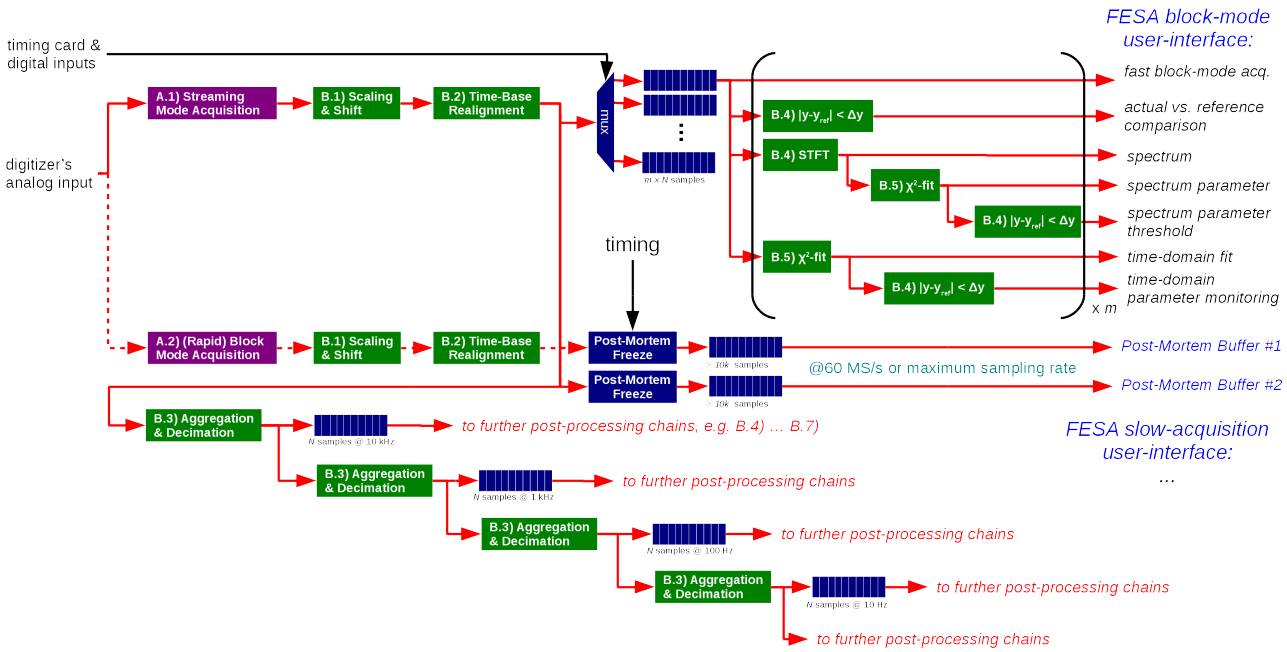


Figure 5: Signal flow-graph example to illustrate the (re-) usage of the different post-processing modules and possible daisy-chains. The flow-graph is specific and defined prior to the FEC start-up.

Open-source signal processing and data fitting libraries shall be used for the modules wherever possible, notably the GNU-Radio[3] and ROOT[4] frame-works, in order to simplify further extensions, compactness, readability, re-usability, and maintainability of the implementation. This is also beneficial in terms of performance optimisations and parallel processing. Each module must provide its own off-line unit test, and should kept in a separate source code sub-directory to ease code review, maintenance, and future upgrades.

3 See www.gnuradio.org

4 See https://root.cern.ch/

3.1 B.1) Input Scaling and Offset

The internal digitizer data format should be converted to a suitable floating point standard already at an early stage (e.g. see 'vendor software interface abstraction layer' in Figure 1). In addition, as a basic post-processing applied to all digitizer channels, the raw digitizer input need to be scaled to relate to the actual physical measurement device input range and property. The initial implementation shall implement a simple scaling of the digitizers ADC input (N.B. binned to [V] conversion) by a user-defined calibration constant per digitizer input channel (default: '1'). The user shall be able to chose on-line whether this calibration constant is static or multiplexed per beam-production-chain (BPC).

3.2 B.2) Time-Base Realignment

The time-base of each channel shall be re-aligned with a user-defined time offset to compensate for constant cable- and other post-processing delays. Provisionally, at least two separate user-defined delay settings variables should be provided: 'static cable delay' and 'post-processing delay', which internally may be summed.

In case the signal has been triggered by a bunch-to-bucket extraction event, the actual to be measured event may be delayed by a few milliseconds with respect to the emitted extraction timing system event. In this case, the time-base should be re-aligned accordingly based on the actual extraction event offset. Provisionally, it can be foreseen that the actual acquisition shall be triggered by a timing system's event, and the difference between the timing and actual extraction event being provided by a second timing event (or timing meta-information) in rapid succession after the first one (N.B. time-scale of a few ms to tens of ms). Further details on this are available in [10].

The actual extraction delay shall be published alongside the (summed) user-defined delays with the acquired data (see Section 4.3.1).

3.3 B.3) Data Aggregation and Decimation

The targeted digitizers operate natively with sampling frequency in the few hundred MHz range. However, continuous streaming to the host FEC is typically limited to around 50-60 Mhz, and many (but not all) users often only require data at rates in the few 10 kHz down to few Hz-range. Thus, the FESA server needs to implement several aggregation and decimation stages to both serve the triggered-high- and streaming-low sampling speed acquisition modes.

The aggregation result must not be limited to the computation of signal mean value but must also include the estimation of the corresponding signal's standard deviation for the given aggregation mode. For a given channel variable '*x*', the following definitions and short-hand notions are used in further subsection:

$$\mu \approx \langle x \rangle := \frac{1}{N} \sum_{i=0}^{N} x_i \tag{1}$$

with x_i being the *i*-th of the last *N* samples in the buffer, and the standard estimation being estimated by:

$$\sigma \approx \sqrt{\left| \langle x^2 \rangle - \langle x \rangle^2 \right|} \tag{2}$$

Systematic propagation of the measurement uncertainty must be applied between each different stage, e.g. if *f(a,b,c, …)* denotes the functional dependence of the input variables *a*, *b*, *c*, … and

their uncertainties given as σ_a, σ_b, σ_c, …, then the uncertainty on the new computed output variable can be estimated via:

$$\sigma_f = \sqrt{\left(\frac{\partial f(a,b,c,...)}{\partial a}\right)^2 \sigma_a^2 + \left(\frac{\partial f(a,b,c,...)}{\partial b}\right)^2 \sigma_b^2 + \left(\frac{\partial f(a,b,c,...)}{\partial c}\right)^2 \sigma_c^2 + ...} \quad (3)$$

In case no standard deviation of the measurement is available, the theoretic (e.g. vendor specified, user-configurable, and/or lab-measured ENOB) values should be used as a starting point.

The following three aggregation and decimation modes need to be implemented:

3.3.1 Digitizer Streaming to FEC

In case the low-level streaming-mode acquisition is chosen, most digitizer offer functionalities for the aggregation and decimation to intermediate sampling frequencies (N.B. often in favour of trading bandwidth for vertical resolution). For the initial implementation the following aggregation modes shall be exposed for the data conversion of the streaming at intermediate frequencies to the FEC via the 'vendor software interface abstraction layer':

- min/max value over the last N values, with the following estimates being used: for the mean $\mu \approx 0.5 \cdot (x|_{min} + x|_{max})$, and $\sigma \approx (x|_{max} - x|_{min})/\sqrt{N}$ as an estimate for the standard deviation.
- simple decimation returning only the first of the last N values, with μ equal to the decimated value, and σ being estimated by the ENOB from digitizer specification folded with the chosen input range and user-calibration factor,
- average returning the arithmetic mean μ of the last N values, and σ being estimated by the ENOB from digitizer specification, propagated via equation (3) using N (N.B. equals to individual measurement error scaled by $/\sqrt{N}$), and folded with the chosen input range and user-calibration factor.

Further low-pass filtering and decimation should be done in software via the FEC.

3.3.2 FEC-based low-pass Filter and Decimation

In case the signal of interest is in the bottom-half of the base-band of the digitizer range (i.e. f in [DC, f_{max}]), further aggregation shall be performed with a user-configurable low- or band-pass filter and decimation to the desired acquisition rate. For convenience and testability the corresponding GNU Radio *'Low Pass Filter'* and *'Band Pass Filter'* functions shall be used. The function shall be applied for both computations of μ and σ (according to equations (1) and (2)). It provides the following configuration parameters that shall be exposed via the FESA interface:

- Decimation/Interpolation (type: int): if re-sampling is not required this parameter is set to 1.
- Gain (type: real): the gain of the filter (default: 1).
- Sample Rate (type: real): the sample rate of the filter, in Hz.
- (High/Low) Cutoff Freq (type: real): the cut-off frequency of the filter, in Hz.
- Transition Width (type: real): width between the pass band and stop band. Small transition width will increase the length of the FIR filter.
- Window: specifies the window function that will be applied to the FIR filter (e.g. Hamming, Hann, Blackman, Rectangular, Kaiser).

- Beta (type: float): Beta parameter for the Kaiser window (default: 6.76).

The filters may be internally cascaded between the different required sampling ranges, and their intermediate results stored in circular buffers to reduce the computational overhead.

3.3.3 Band-pass Filtering and Down-Conversion

In case the signal of interest is not in the bottom-half of the base-band of the digitizer range (i.e. f in [f_{min}>0, f_{max}]), further aggregation shall be performed with a user-configurable band-pass filter, down-mixing of the signal to base-band, and decimation to the desired acquisition rate. For convenience and testability GNU Radio's channelizer functionality and corresponding *'Frequency Xlating FIR Filter'* function shall be used[5]. The function shall be applied for both computations of μ and – provided the FEC the performance permits this – σ (according to equations (1) and (2)). In addition to the band-pass filter parameters outlined above, it provides the following configuration parameters that shall be exposed via the FESA interface:

- Decimation (type: integer): if re-sampling is not required this parameter is set to 1.
- Taps: a valid *'firdes'* expression or list containing the taps (if designing filters using another method).
- Center Frequency (type: float): the centre frequency. this is the frequency that will be shifted down to 0 Hz before the channel selection filter is applied. This may either be a user-defined constant or frequency function per BPC, most commonly the time-dependent function $f_s(t)$, most often being the n-th harmonic of the revolution frequency f_{rev}.
- Sample Rate (type: float): specifies the sample rate, in Hz.

A schematic illustration of the algorithm is shown in Figure 6:

[5] an exemplary implementation documentation is given at: http://blog.sdr.hu/grblocks/xlating-fir.html

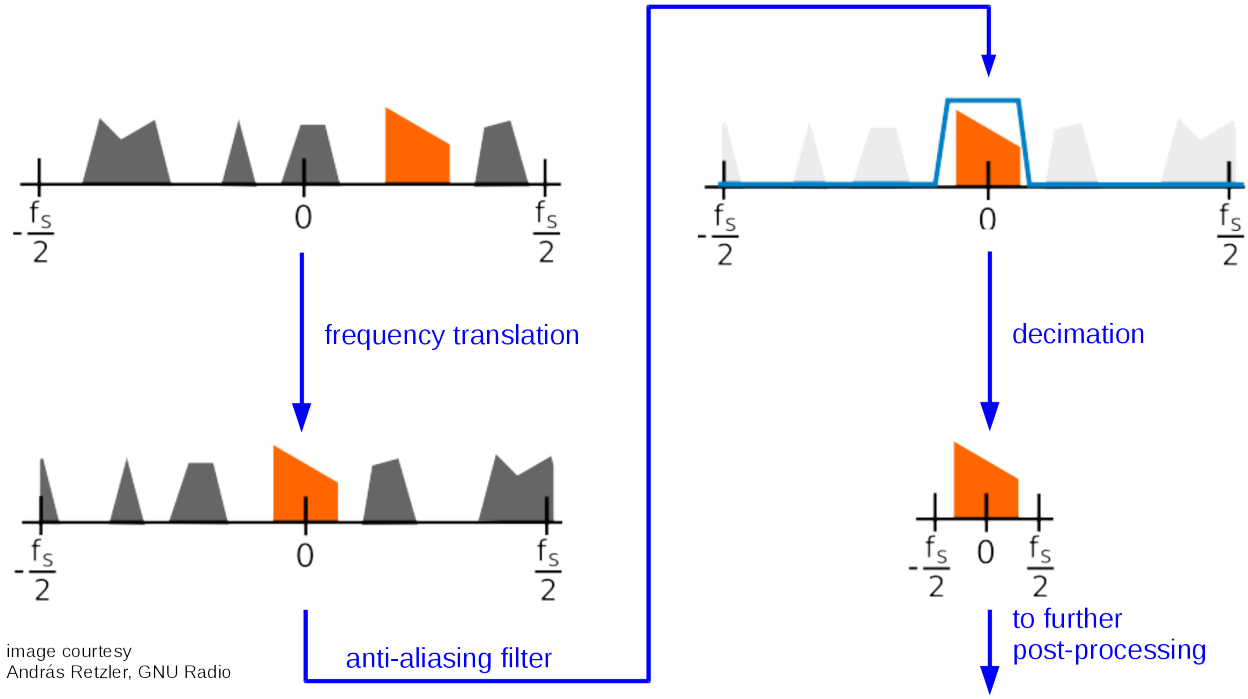


Figure 6: Schematic function of GNU Radio's *'Frequency Xlating FIR Filter'* function.

3.4 B.4) Short-Time Fourier Transform

Perform a real-time Short-time Fourier Transform (STFT) on the data obtained through the previous modules:

$$STFT\{[x]\}(m,f_k) := \sum_{n=-m/2}^{n=+m/2} x_n \cdot w_n \cdot e^{i2\pi f_k} \quad \text{and} \quad f_k = f_{min} + k \cdot \Delta f = f_{min} + k \cdot \frac{f_{max} - f_{min}}{n_f} \tag{4}$$

with x_n being the n-th sample of the input, w_n one of the user-selectable windowing function (e.g. Hamming, Hann, Blackman, Rectangular, Kaiser), m the acquisition length in terms of samples, f_i in $[f_{min}, f_{max}]$ for which the frequency component in the signal should be evaluated, and n_f *the* frequency binning (N.B. $k = 0 \ldots n_f$).

The following configuration parameters that shall be exposed via the FESA interface:

- Acquisition Period ΔT: the time between blocks of data for which the STFT should be computed. Typical default periods are between 1 to 5 ms.
- Acquisition Length *m*: segmentation length of the acquisition buffer upon which the STFT shall be computed. Typical lengths are between 0 to 50 % overlap between segments, corresponding to minimum segment lengths of about 1-2 ms. The maximum length shall be limited by the FEC CPU performance. Provisionally, it can be foreseen to limit this to 256k samples (↔ 262 µs at 1 MHz sampling frequency, or minimum frequency resolution of about 4 Hz).

- Equivalent Sample Rate $f_s(t)$: specifies the sampling rate, in Hz. This frequency may either be a user-defined constant f_s or time-dependent function $f_s(t)$, most often being the n-th harmonic of the revolution frequency f_{rev}. In case $f_s(t)$ is a function of time, the resulting spectra should be exported in terms of normalised frequency rather than 'Hz'[6].
- Minimum/Maximum [f_{min}, f_{max}] and frequency binning n_f of the input.
- Window: specifies the window function that will be applied to the FIR filter (e.g. Hamming, Hann, Blackman, Rectangular, Kaiser).
- Beta (type: real): Beta parameter for the Kaiser window (default: 6.76).

Concerning the numeric implementation, depending on the frequency range [f_{min}, f_{max}], frequency binning n_f and equivalent sampling frequency $f_s(t)$, the following numerical implementations may be considered:

- FFT – Fast Fourier Transform, in case [f_{min}, f_{max}] ≈ [0, $f_s/2$] and $f_s(t)$ is constant,
- Goertzel Transform, in case only a few terms between [f_{min}, f_{max}] are requested to be computed, or
- DFT — Discrete Fourier Transform, in case $f(t)$ changes from one time-slice to the next.

Wherever possible, performance optimised versions of these algorithms either from GNU Radio or other common libraries shall be used (e.g. FFTW). Other custom implementation may be considered if the standard library performance is deemed to be insufficient.

3.5 B.5) RF-Amplitude, Phase, and Frequency Detection

A specific digitizer application includes the monitoring of RF amplitude, phase, and frequency in relation to an external frequency reference $U_{ref}(t)$. For the initial implementation it shall be assumed that the external reference is provided through one of the other analog digitizer inputs.

The amplitude, phase and frequency detection can be measured via the IQ-demodulator illustrated in Figure 7.

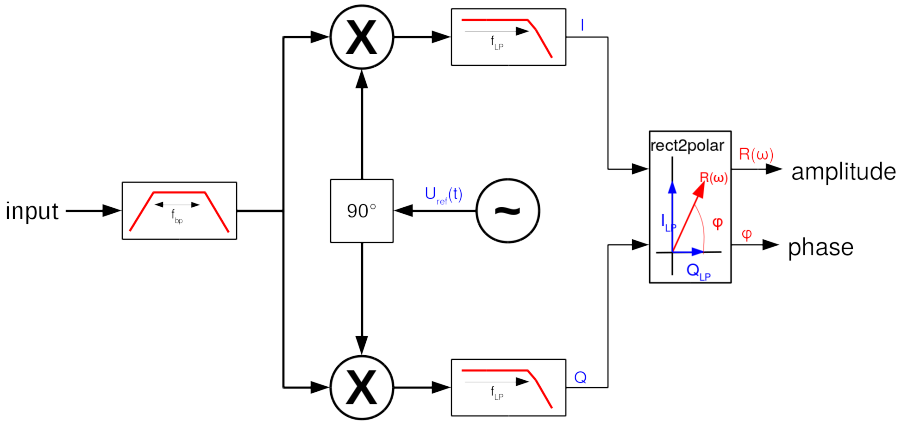


Figure 7: Schematic IQ-demodulator for the detection of amplitude and phase for a given reference signal $U_{ref}(t)$. The '90°' block indicates a Hilbert-Transformer providing a phase-synchronous in- and quadrature reference signal to the mixer.

It is recommended to re-use GNU Radio's band- and low-pass filter mentioned in the previous post-processing blocks.

Due to the possibly substantial cable and post-processing delay, the corresponding amplitude and phase outputs need to be corrected by user-defined calibration constants related to the amplitude

[6] This is required for the analysis of longitudinal and transverse Schottky and tune spectra.

(similar to 'B.1'), and delays due to cable length or post-processing. The actual phase and amplitudes correction after the rectangle-to-polar conversion are given by

$$\begin{aligned} A &= cal_{user} \cdot A_0 \\ \varphi &= \varphi_0 - \left[\varphi_{user} + \left(\frac{\partial \varphi}{\partial f} \right)_{user} \cdot f \right] \end{aligned} \tag{5}$$

with cal_{user}, φ_{user}, $(\partial\varphi/\partial f)_{user}$ being user-defined calibration constants. The input signal can be assumed to be a fairly pure harmonic signal (i.e. second-order terms contributing less than a few percent to the total signal), thus the frequency can be estimated via counting the positive to negative input samples for a known observation period *N*, which should be equal to the effective time constant of the low-pass filter shown in Figure 7.

3.6 B.6) Basic Chi-square-based Fitting

In addition to the raw and filtered digitizer inputs, it is required to perform further fitting to derive second order parameter which are further treated similar to the direct digitizer inputs. The aim of the fitting is to find a set of parameter that of a function *f* that describes the input well, with n_p the number of free parameters to be fitted (aka. 'degree of freedoms'). A commonly used optimisation criteria is to minimise the χ^2-function (or: chi-square function), given as

$$\chi^2 = \sum_{i=0}^{N} \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2 \tag{6}$$

with y_i the measured (post-processed) digitizer input signal, x_i the time (or frequency) coordinates, and *N* the number of samples to be fitted. The related 'goodness' of the fit *G* is estimated by

$$G := \frac{\chi^2}{n_p} \tag{7}$$

and should generally be around '1' for a fit to be considered 'good'. The actual acceptance criteria for G should be evaluated against a user-defined threshold

$$\Delta G = |G - 1| \tag{8}$$

If this criteria is not met, the fitted parameters should be kept, but the result value flagged as erroneous.

The FEC shall implement two fitting modules that shall be extensible in the future: one implementing a generic chi-square-based fitting that can be used for time- or frequency domain data, and another module more specific for frequency-domain data, implementing a basic peak detection and peak-width fitter. The result of their parameter are in turn new post-processed signals that may be post-processed in a similar way as outlined above.

3.6.1 Generic Chi-square based fitting

In order to simplify the implementation, and to avoid re-implementation of common numeric code, the emphasis must be laid on the use of the standardised ROOT fitting framework, encapsulation and unit testing of the fitting module.

Provisionally, it can be foreseen that the fitting is primarily limited to individual block-mode acquisition, the time-domain data spanning the whole beam-in-beam out sequence, or individual STFT magnitude spectra. The following ROOT fitting interface shall be exported through the FESA user-interface:

- selection of the input signal: (rapid) block mode acquisition, beam-in-to-beam-out BPC data (typically limited to 1-10 kHz data streams), STFT data blocks:
- Npar (type: integer): number of parameters n_p: the number of free fitting parameters,
- ParNames (type: 1D-array of strings): parameter names that later shall be mapped to the new channel/signal names,
- Parameters (type: 1D-array of floats): initial parameter values (typically supplied as default, or user-defined values through the settings management),
- ParErrors (type: 1D-array of floats): initial parameter error estimates
- FixParameter (type: 1D-array of boolean): list of parameter that should be fixed
- ParLimits (type: 2 floats): parameter min/max range
- Fitting range (type: 2 floats): depending the input channel this shall be given either in time coordinates [t_{min}, t_{max}] or frequency range [f_{min}, f_{max}] (generic: [x_{min}, x_{max}])
- Function representation (type: string): ROOT allows a wide range of string-based function description (see class documentation for 'TFormula' for reference[7]), some examples include, e.g.
  - 'gaus' for a simple fit of a Gaussian function,
  - sin(x)/x
  - [0]*sin(x) + [1]*exp(-[2]*x)
  - x + y**2
  - 'gaus + pol0(3) + expo(4)', 'gaus+(x>30 && x<90)*pol2(3)', or
  - '(x<[0]?0:1*[1])' for fitting the time offset and amplitude of a Heaviside step function.
- The return values follow the same convention and return:
  - parameter (type: 1D-array of floats)
  - parameter error (type: 1D-array of floats)
  - chi-square (type: float):
  - validity flag (type: boolean): based on the rejection criteria mentioned in equation (8).

[7] https://root.cern.ch/doc/master/classTFormula.html

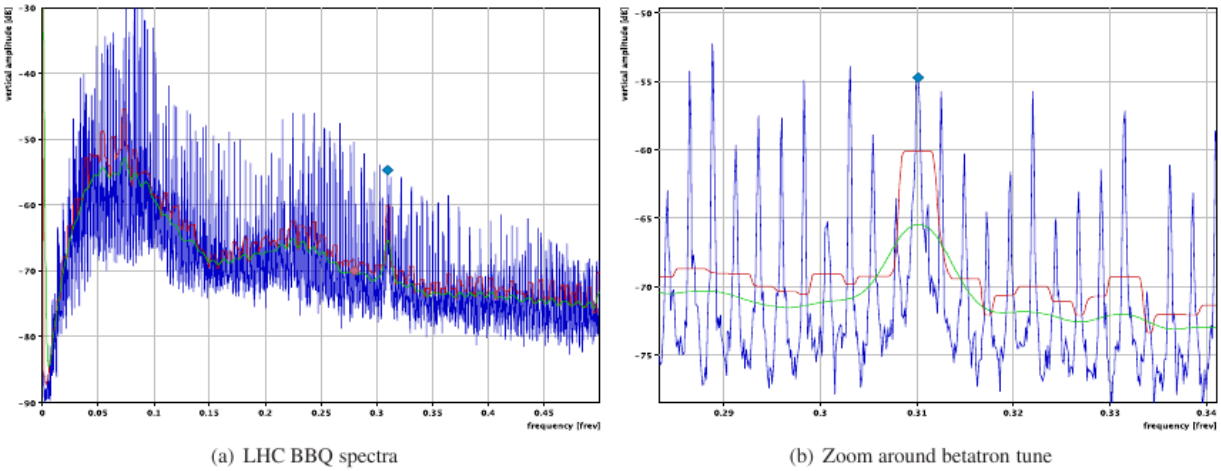
3.6.2 Fitting of spectral peaks

A common post-processing step of frequency-domain data is the peak detection and peak width measurement. While a wide range of specialised spectral fitting aspects are covered by the generic fitting routine outlined in section 3.6.1 above, an additional more basic but also more robust fitting routine is required from an engineering point-of-view. Based on operational experience at CERN and similar systems, the following post-processing algorithm[8] shall be implemented:

1. calculate the raw-spectra $S_{raw}(f)$ based on the n-sample oscillations data,
2. compute (averaged) magnitude spectra $|S_{raw}(f)|$,
3. apply a n_{med}-wide median-filter → $|S_{med}(f)|$,
4. apply a $\pm n_{lp}$-wide sliding average-filter → $|S_{lp}(f)|$,
5. find highest peak $Q_{est.}$ in $|S_{lp}(f)|$ within the given
6. boundaries $Q_{est.} \in [f_{min}, f_{max}]$,
7. find highest peak Q_{raw} in $|S_{raw}(f)|$ around the previous '$Q_{est.} \cdot n/2 \pm n_{med}/2$' estimate,
8. refine the binning-limited Q_{raw} estimate by fitting the tune resonance to a Gaussian distribution[9].
9. compute the full-width-half-maximum estimate at the peak, and compute the Gaussian-equivalent width as:

$$\sigma \approx \frac{\mathrm{FWHM}}{2\sqrt{2\ln 2}} \tag{9}$$

A visual example of the algorithm is given in Figure 8.



(a) LHC BBQ spectra (b) Zoom around betatron tune

Figure 8: Raw ($|S_{raw}(f)|$, blue), intermediate median- ($|S_{med}(f)|$, red) and low-pass filtered magnitude spectra ($|S_{lp}(f)|$, green) as used in the revised peak fitter algorithm.

[8]R.J. Steinhagen, M. Gasior, and S. Jackson, *"Advancements in the Base-Band-Tune and Chromaticity [..]"*, Proceedings of DIPAC2011, Hamburg, Germany, 2011

[9]R.J. Steinhagen, "Tune and Chromaticity Diagnostics", CAS, Dourdan, France, 2008, pp.317

3.7 B.7) Actual vs. Reference Monitoring and Interlock Generation

In order to automatically detect device errors or parameter drifts, a comparison of the actual *y(t)* versus required used-defined target reference signal $y_{ref}(t)$ shall be provided. Here, *y(t)* may be any of the raw digitizer input channels or post-processed variables.

A software-based interlock signal shall be issued if the user-defined minimum $y_{min}(t)$ or maximum $y_{max}(t)$ tolerance thresholds are exceeded. This interlock shall be latched within and for subsequent executions of the BPC for which the violation was detected. A FESA interface must be provided to reset this latched interlock.

4 User/Top-Level Data Acquisition Modes

The data obtained through the low-level acquisition and post-processing modes described above, shall be exported using FESA conform user-level interfaces. It can be assumed that for single-pass devices (e.g. injection kickers, or other transfer-line devices) the digitizers will be triggered by external timing events and thus may internally use the block-mode acquisition. Other devices (e.g. RF-cavities, electro-static injection/extraction septa voltages, etc.), shall be generally performed and processed on a continuous basis to allow the use of post-triggers or triggers where sub-portions of the requested data may overlap.

In addition, for machine study and machine protection (post-mortem) purposes three copies of the fast-sampling buffers are needed that cover at least 5 ms (↔ ~2 MB at 100 MS/s). The buffer used for machine studies would need to be filled with dedicated start timing event. The two post-mortem buffers would be circular buffers that are filled on a fast basis and frozen once the corresponding post-mortem event has been triggered. The details of the post-mortem system need to be further discussed but the data structures and information should be kept synchronised with the other modes of acquisition to ease further integration into the controls and operation environment. It can be assumed that the post-mortem trigger is supplied either via the regular external timing trigger or – if necessary – dedicated digital trigger.

4.1 User/top-Level Data Acquisition Modes

The digitizer and derived data that is published using the FESA client interface must be provided in at least three different acquisition variants simultaneously (though some of them could be implemented as down-sampled or averaged copies of the higher bandwidth acquisition):

- Slow-acquisition (FESA property: e.g. 'Acquisition') – some of them could be implemented as down-sampled copies of the higher bandwidth streaming-mode acquisition:
  - beam-in-to-beam-out sequence[10]-based acquisition: the post-processed and averaged digitizer input would be acquired with up to 1 kHz and down-sampled to the rate needed for the given user or application. Some expected typical sampling rates are: SIS18 typ. 1 kHz, SIS100 typ. 100 Hz (1k Hz for MDs), CR/HESR ~ 10/100 Hz (1k Hz for MDs). The user may chose further down-sampled copies with decadic sampling steps (e.g. 1 kHz, 100 Hz, 10 Hz) specified given via the filter on the FESA property. For this mode the data would be published as a complete trace as a function of time at the end of the sequence. Since the interface is the same, the different possible data rate selections could be implemented via the FESA property filter functionality.

[10] here 'sequence' corresponds to the FESA sequence or beam-process definition. The choice whether to multiplex on sequences or beam-processes shall be made with the instantiation of the FEC device.

- Continuous real-time data publication during the sequence: some of the beam-in-to-beam-out sequence lengths may become very long or undefined, notably the dedicated storage rings ESR and HESR, as well as for very long slow extraction from SIS18 and SIS100 (up to 100 seconds). While for, for example, archiving the data block-wise at the end of the sequence is preferred, in this particular case the information would often be needed already while the sequence is being executed. Typically update rates are around 10-25 Hz for user-level applications or software-based real-time feedbacks, or other automated online steering applications, for example.
- Individual samples at specified times within the beam-in-to-beam-out sequence: this mode is very similar to the previous insofar that the post-processed and averaged digitizer input data is immediately published with the difference that the sampling and number of samples is different (e.g. property notification at 1 ms, 13 ms, 42 ms, 152 ms after the first injection). This mode would cover only up to 10 measurement points in the sequence that are – within 1 ms granularity – arbitrarily spaced. This acquisition mode could use a copy of the data already stored in the sequence measurement buffer, as one implementation option. The total latency for the communication between digitizer to the UDP-packet having left the FEC should be targeted to be less than 10 ms with 98-99% probability, and 100 ms worst case delay (soft real-time requirements). The timing will be provided through a BPC-multiplexed timing vector containing the delay offsets of the data to be taken and published with respect to the injection event.

- Fast block-mode acquisition – could be implemented either as tagged sub-portions that are extracted from the low-level streaming-mode acquisition (see Figure 3), or using the (rapid) block-mode acquisition (if the low-level streaming-mode is not requested or higher sampling speeds are required, see Figure 4). These are triggered by up to 10 external user-defined timing triggers per each beam-in-beam-out sequence (ie. trigger for each injection, start/end of the ramp, extraction etc.). Each trigger should be time-stamped using the FAIR timing system reference. The horizontal channel settings can be assumed to be fixed per digitizer channel setup. For overlapping burst-mode segments the user will ensure that the vertical channel setting are either coherent with respect to each trigger within the sequence, or – for well separated triggers – provide a separate
  - ‘arm’ trigger in response to which the channel can be set up,. and
  - actual start trigger (ie. trigger for each injection, start/end of the ramp, extraction etc.).
- Post-Mortem: for post mortem analysis, data needs to be stored in different buffers to be frozen by external timing events. The exact requirements in this domain are not finalized yet. Provisionally, it can be foreseen that:
  - a first circular buffer will store the post-processed digitizer data measured every millisecond over the last second of beam or full beam-in-to-beam-out sequence, whatever is more convenient. If possible, the same data format as for the regular sequence-based data acquisition should be kept to ease controls integration.
  - a second circular buffer shall store the post-processed data measured by the digitizer over the last 5 ms. If possible, the same data format as for the regular fast block-mode acquisition should be kept to ease controls integration.

Depending on whether the FESA design is chosen to be sequence- or beam-process-based, the acquisition shall be published using the chosen FESA data structure. In addition, the data should be also available in a dedicated non-BPC-multiplexed buffer to allow data retrieval in case timing is unavailable, or no beam-production-chain or timing pattern is being defined.

4.2 Data Acquisition Multiplexing and Indexing

The above mentioned acquisition mode data should be retained for each sequence and buffered for the length of at least one timing pattern (length of multiple beam-production-chains). The acquisition window is typically set from the first injection until the last part of the beam is extracted. In order to correlate measurements across different digitizers, each beam-in-to-beam-out sequence data should be tagged with an absolute UTC (ns-level) time stamp of the first injection and other fast acquisitions, and ms-level precision relative time-stamp since the first injection for the slow data acquisition modes. For further details see [2].

4.3 Expected Data Acquisition Variables

It is expected that the digitizer data will be provided via FESA/CMW and its 'Device-Property model'. Where possible, the variables should be reported in natural SI-units without metric prefixes. Following specific variables and rates are proposed to be acquired from an operation and archiving point of view, in addition to what has been described in [2]:

4.3.1 Time-Domain FESA Property User-Interface

Variable Name	Brief Description	Variable Type	SI unit	Expected Sampling
acqTiggerName	trigger name[11]	enum or String	[]	per sequence
acqTiggerTimeStamp	UTC trigger time-stamp	Integer (64 bits)	[s]	per sequence
acqLocalTimeStamp	time-stamp w.r.t. beam-in trigger	Integer (64 bits)	[s]	per sequence
channelTimeBase	time scale	1D-array of integers	[s]	per sequence
channelUserDelay	user-defined delay	2D-array of floats (32 bits)	[s]	per sequence
channelActualDelay	actual trigger delay (i.e. during bunch-to-bucket transfer)	2D-array of floats (32 bits)	[s]	per sequence
channelValue	value of digitizer input or post-processed signals	2D-array of floats (32 bits)	[user-defined, see channelUnit]	per sample[12]
channelName	name of digitizer input or post-processed signals	1D-array of strings	[]	per sequence
channelUnit	S.I. unit of post-processed signal	1D-array of strings	[]	per sequence
channelCalibration	raw- to post-processed signal calibration factor	1D-array of floats	[channelUnit/V]	per sequence
channelError	r.m.s. error of post-processed signal	2D-array of floats (32 bits)	[see channelUnit]	per sample

[11] e.g. INJECTION1,…, INJECTION8, EXTRACTION, USER1, … USER3

[12] sample rate is machine dependent: e.g. SIS18/SIS100: 1 kHz

digitalInputValue	ext. digital input state	1D-array of integers	[1: high, 0: low]	per sample
digitalInputName	ext. digital input name[13]	1D-array (16) of Strings	[]	per sequence
channelInterlockState	result of actual-vs-reference comparison	1D-array of Booleans	[1: interlock, 0: no interlock]	per sample
Status	status bit-mask	Integer (32 bits)	[a.u.]	per sequence
Temperature[14]	temperature of AFE, etc.	1D-array of floats (32 bits)	[°C]	per sequence or [1..0.1 Hz]

4.3.2 Frequency-Domain FESA Property User-Interface

Variable Name	Brief Description	Variable Type	SI unit	Expected Sampling
acqTiggerName	trigger name	enum or String	[]	per sequence
acqTiggerTimeStamp	UTC trigger time-stamp	Integer (64 bits)	[s]	per sequence
acqLocalTimeStamp	time-stamp w.r.t. beam-in trigger	Integer (64 bits)	[s]	per sequence
channelTimeBase	time scale	1D-array of integers	[s]	per sequence
channelFrequencyBase	frequency scale	2D-array of floats	[Hz or f_{rev}]	per sequence
channelMagnitude	magnitude specta of digitizer input or post-processed signals	3D-array of floats	[channelUnit / √Hz]	per sample
channelPhase	phase specta of digitizer input or post-processed signals	3D-array of floats	[rad]	per sample
channelName	name of digitizer input or post-processed signals	1D-array of strings	[]	per sequence
channelUnit	S.I. unit of post-processed signal	1D-array of strings	[]	per sequence

...to be defined

[13] defined during the initial FEC device set-up

[14] 'Temperature' where available, e.g. of the AFE, digitizer or other components that may cause measurement drifts, or that are available to the FEC.

5 Settings Supply

In order to calculate some of the derived measurement properties from the digitizer input, some additional information such as the reference revolution frequency, number of bunches, or selected charge state needs to be provided from the data supply model. A tentative list of standard parameters in addition to what has been described in [2], are outlined below:

Variable Name	Brief Description	Variable Type	SI unit
Global Beam Parameters			
ionMassNumber	Ion mass number A	Float (32 bits)	[u]
IonAtomicNumber	Ion atomic number Z	Integer (32 bits)	[]
ionChargeState	Ion charge state	Float (32 bits)	[e]
kineticEnergy	Kinetic energy E_{ion}	Float (32 bits)	[MeV/u]
maxIntensity	maximum expected ion intensity	Float (32 bits)	[ppp]
Device Specific Settings			
calFactor	calibration factor from digitizer input to post-processed varible	1D-array of floats (32 bits)	[‘see calFactorUnit’/V]
calFactorUnit	calibration factor unit from digitizer input to post-processed varible	1D-array of strings	[‘user-defined’/V]
Extraction Beam Parameters			
harmonicNumber	Harmonic number w.r.t. f_{rev}	Integer (32 bits)	[]
maxNumberOfBunches	expected number of bunches	Integer (32 bits)	[]
bunchLength	r.m.s. bunch length	Float (32 bits)	[s]
spillLength	expected maximum spill length	Float (32 bits)	[s]
Ring Specific Parameters			
numberOfInjections	number of expected injections	Integer (32 bits)	[]
revolutionFrequency	f_{rev} frequency (rings)	2D-array of floats (32 bits)	(t [s], frev [Hz])
Specific Parameters for Beam-Detector combination			
timeOfFlight	Time-of-flight (source resp. extraction-reference to device	Float (32 bits)	[s]
flightPath	path length acc-det.	Float (32 bits)	[m]
BeamSpotSizeX	Full horizontal beam spot size	Float (32 bits)	[m]
BeamSpotSizeY	Full vertical beam spot size	Float (32 bits)	[m]

The specific gain settings is derived based upon above information within the front-end computer.

6 Quality Assurance, Tests, and Acceptance

Since the system described here fully consists of software which must be well integrated into the accelerator control system, the chapter 4 “Quality Assurance, Tests and Acceptance” of the Common Specification for the Accelerator Control System [9] applies. The following subsections briefly describe the necessary measures, for details refer to [9].

6.1 Quality Assurance system of the contractor

The contractor has to fulfil the QA requirements specified in the General Specification [1].

Due to the complexity of the accelerator control system and therefore the need for early integration tests, an incremental and iterative development approach is advised. Based on the main requirements an overall solid system architecture is setup and in a first iteration a vertical slice through all layers is built realizing the most crucial functionality. Then, in well-defined iteration cycles, components are replaced by newer versions, implementing more functionality. For further details see [9].

Additionally the long lifetime of the FAIR facility and in this time the occurring computer hardware and software technology changes generate the necessity for continuous adaptation and extension of the control system. All delivered components must therefore be developed with the focus on maintainability and exchangeability.

The contractor must obtain the approval of the contracting body for all technical concepts and detailed design reports before start of system implementation. The contractor must obtain the approval of the main contractor for the integration into the control system as well as the adherence to guidelines and standards before start of system implementation.

6.2 Factory Acceptance Test (FAT)

After finishing one iteration of a component, successful internal function testing and before delivery, the Factory Acceptance Test (FAT) is used to ensure that the component meets the functional and non-functional requirements defined by the contracting body and is ready to be shipped. For details on the FAT procedure see [9].

6.3 Site Acceptance Test (SAT)

The Site Acceptance Test (SAT) after each iteration of a component ensures that the component fulfils all user requirements and that it is ready to use within the users environment. Each iteration cycle must repeat tests from previous iterations. For details on the SAT procedure see [9].

7 Documentation

Chapter 5 “Documentation” of the Common Specification for the Accelerator Control System [9] and chapter 10 “Documentation and Training” of the Detailed Specification Data Acquisition Error: Reference source not found fully apply.

8 Related Documents

1. General Specification, F-GS-F-01e-General_Specification (Mandatory)
2. Detailed Specification Data Acquisition (F-DS-BD-40e), https://edms.cern.ch/document/1179740 (Mandatory)
3. FAIR specification document F-TC-C-06e, “Multiplexing and Data Indexing Concept” (Mandatory)
4. FAIR Technical Concept F-TC-C-07, “Accelerator and Beam Modes” (Mandatory)
5. FESA Guideline, https://edms.cern.ch/document/1235310 (Mandatory)
6. Detailed Specification Settings Management System (F-DS-C-03e), https://edms.cern.ch/document/1176027 (Recommended)
7. Detailed Specification Archiving System (F-DS-C-11e), https://edms.cern.ch/document/1176039 (Recommended)
8. Detailed Specification Post Mortem System (F-DS-C-13e), https://edms.cern.ch/document/1176041 (Recommended)
9. Common Specification for the Accelerator Control System (F-CS-C-01e), https://edms.cern.ch/document/1174186 (Mandatory)
10. Technical Concept of the FAIR Bunch To Bucket Transfer System (F-TC-C-05e), https://edms.cern.ch/document/1514162/6

9 Appendix

9.1 Abbreviations, Terms and Definitions

AFE	**Analog Front-End**
	Actual State, actual measured state of the accelerator or beam
	Accelerator & Beam Mode, deliberate user-driven states (references or 'desired target') that follow and track the normal operational sequences (e.g. 'no beam' → 'pilot beam' → 'intensity ramp-up' → 'adjust' → 'stable beams for physics')
BP	**Beam Process**, specific atomic operation/procedure (injection, ramp, extraction…) that occurs in a defined section of an accelerator or transfer-line
BPC	**Beam Production Chain,** organisational control system structure to manage parallel operation and beam transfer through the FAIR accelerator facility. It describes a beam from the ion source to the target accelerator or experiment.
	Beam Pattern, sequence of BPCs typically executed periodically. Beam patterns can be changed within a few minutes if necessary.
BTM	**Beam Transmission Monitoring System**, collection and aggregation of the sum of all DCCTs, FCTs, and BLM for the purpose of tracking the beam particle intensities and transmission losses along the BPC
FB	**Feedback**
FEC	**Front-End Controller**
FESA	**Frond-End-Software-Architecture [5]**
MD	**Machine Development Studies**
MP	**Machine Protection (System)**
OP	**Operation**
ppp	**Particles per pulse** (unit used for beam intensities)
RF Bucket	stable phase space around the synchronous particle, defining the finite longitudinal bunch length