FESA Class Technical Description

CryCup

Class name	CryCup
Class version	0.1.0
FESA version	Fesa3 Release 2.0.1
Manual version	0.2
Manual release date	March 23, 2015

Abstract

The *CryCup* FESA class is responsible for reading out the Faraday cups of the Cryring and calculating the total charge and number of ions detected by the cup.

The following conventions are used in the document:

- Names like file names, FESA class names, application names etc. are written in *italics*.
- Actual code fragements including class and interface names are written in `typewriter font`.

Contents

1 Introduction **3**

2 Interface **6**

2.1 Device Setting . . . 6
2.1.1 MeasureBaseline . . . 6
2.1.2 Init . . . 6
2.1.3 Reset . . . 7
2.1.4 Setting . . . 7
2.1.5 ExpertSetting . . . 8
2.1.6 IOSetting . . . 10
2.1.7 Power . . . 10
2.2 Device Acquisition . . . 11
2.2.1 BaselineAcquisition . . . 11
2.2.2 Status . . . 11
2.2.3 Acquisition . . . 12
2.2.4 Version . . . 13
2.3 Global Setting . . . 14
2.3.1 StartIO . . . 14
2.3.2 StopIO . . . 14
2.3.3 Arm . . . 14
2.3.4 Acquire . . . 15
2.3.5 DiagnosticSetting . . . 15
2.3.6 SummaryIOSetting . . . 16
2.3.7 SummarySetting . . . 17
2.4 Global Acquisition . . . 18
2.4.1 DeviceDescription . . . 18

3 Custom Types **19**

3.1 Enums . . . 19
3.2 State-Enums . . . 21
3.3 Bit-Enums . . . 22
3.4 Constants . . . 22

A Revision History **23**

Chapter 1

Introduction

The *CryCup* FESA class is responsible for reading out the Faraday cups of the Cryring and calculating the total charge and number of ions detected by the cup.

The current from the cup is amplified and converted in to a voltage signal by a Femto Variable Gain High Speed Current Amplifier (DHPCA-100). The voltage signal is sampled by a Struck SIS3302 ADC with 16bit dynamic range. The typical sampling frequency is 100 MHz but can be set in several steps down to 1 MHz. The sampling of the signal should start at least 100μs before the first ion hit the cup and should continue at least 100μs after the last ion hits the cup. Figure 1.1 shows the measurement of a test signal resembling an

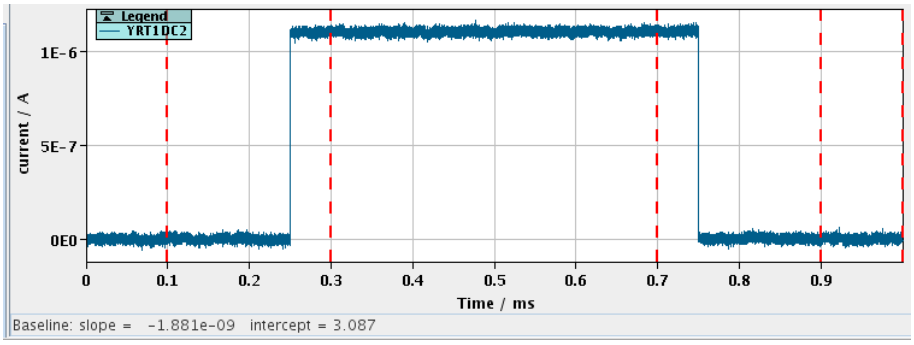

Figure 1.1: Acquired test signal resembling an ideal beam pulse.

idealistic beam pulse. It has been sampled at 100MHz over a time span of 1ms.

In general, operators are not interested in the time dependent signal. They only want to know integral values like the mean current in the pulse or the total number of ions. To calculate these values, 3 regions of interest are required. They are indicated in figure 1.1 by the vertical red dashed lines. The left and right region are used to determine the baseline of the ADC. These regions must not contain any signal from the beam. The middle region is used to integrate the signal after baseline subtraction to obtain values like mean current or total number of ions.

The regions of interest can be specified in two ways:

1. via the `ExpertSetting` property
   In this case, the start and end of each region (i.e. 6 values) in fraction of the full sampling time are specified.

2. via the machine events EVT_MEAS1 and EVT_MEAS2
   In this case the left region extends over the first 100μs and the right region over the last 100μs sampled. The middle region is determined by the two machine events.

As mentioned, each cup is connected connected via a Femto Variable Gain High Speed Current Amplifier (DHPCA-100) to a channel of a Struck SIS3302 ADC with 16bit dynamic range. The ADC is a 8 channel ADC and can thus handle the readout of up to 8 cups. The *CryCup* class is designed to work with more than one ADC. However all ADCs measure at the same time for the same time interval.

The settings of the Femto amplifer are controlled by a Struck SIS3820 Output Register with 32 bit output. For a single cup only 7 bits are required. Each output register can therefore handle up to 4 cups. The *CryCup* class is designed to work with more than one output register module.

The *CryCup* class is primarily designed to work for bunched beams. It requires three events for the typical 'Arm-Trigger-Readout' cycle. The 'Arm' event triggers the `ArmRTAction` which updates the output registers in case the settings for the Femto amplifier have changed and configures and arms the ADCs.

The 'Trigger' event will generate a hardware trigger for all ADCs handled by the class. It must come at least 1ms after the 'Arm' event and slightly before the beam actually hits the cup. Note, that all ADCs are triggered at the same time. That means all Faraday cups connected are read out at the same time, regardless whether they have beam or not.

The 'Readout' event should come at the end of the beam cycle. It triggers the readout of the ADC data and subsequent notification of subscribed clients. To reduce the load on the timing network, the 'Readout' event is generated in the ECA unit of the timing receiver based on the 'Trigger' machine event and a fixed delay (typically 10ms).

The *CryCup* class can also use the EVT_BEAM_ON, EVT_BEAM_OFF, EVT_MEAS1 and EVT_MEAS2 machine event to determine the regions mentioned above for integrating the signal.

As mentioned, the *CryCup* class is primarily designed to work with bunched beams. The ADCs must start sampling some time before the actual beam hits the cup and only stop some time after the beam bunch passed. The samples before and after the bunch are used to determine the baseline of the signal, which must be subtracted to obtain the true charge measured in the bunch. In general a time interval of $0.1\mu s$ – $1\mu s$ before and after the beam bunch is more than sufficient for baseline determination.

However, Faraday cups directly after the ion source will also need to measure a continuous beam. A special operating mode for the *CryCup* class has been introduced to handle this scenario. In `DC` mode, the Fesa class uses a previously determined baseline for each

measurement and determines the total charge and mean current over the full measurement time. To obtain correct values, the baseline must be determined by user request, when no beam is on the cup. It should be checked periodically.

Of course, it is always possible to use a predefined baseline stored in the instantiation file and read when the *CryCup* class starts.

Chapter 2

Interface

2.1 Device Setting

2.1.1 MeasureBaseline

Property	**multiplexed**	**visibility**
MeasureBaseline	false	operational
Description		
Set Action	**partial**	**transaction**
MeasureBaselineSetAction		

This command property triggers a measurement of the ADC's baseline. It is essential, that there is no beam on the cup, during the whole measurement time (typically less than 1s). This is only required, if the *CryCup* class is operated in `DC` mode. In `PULSED` mode with bunched beams, the baseline is automatically determined for each measurement.

2.1.2 Init

Property	**multiplexed**	**visibility**
Init	false	operational
Description		
Set Action	**partial**	**transaction**
InitSetAction	true	true

The `Init` property is currently without function.

2.1.3 Reset

Property	**multiplexed**	**visibility**
Reset	false	operational
Description		
Set Action	**partial**	**transaction**
ResetSetAction	true	true

The `Reset` property is currently without function.

2.1.4 Setting

Property	**multiplexed**	**visibility**
Setting	true	operational
Description		
Set Action	**partial**	**transaction**
SettingSetAction	true	true

The `Setting` property contains the operational settings for a single Faraday cup connected to a single ADC channel. For informational purposes it also returns the current sampling frequency (in Hz) and the sampling length of the ADC (in seconds).

Note: *The smapling length and frequency can only be set via the global device!*

The `gain` setting is in the range of 0 – 6 which results in actual gain factors according to the following table:

setting	gain	mode
0	10^2	low noise
1	10^3	low noise
2	10^4	low noise
3	10^5	low noise
4	10^6	low noise
5	10^7	low noise
6	10^8	high speed

The gain depends on the mode of the FEMTO amplifier. For the highest gain, the amplifier is switched automatically into the high speed mode. Otherwise the low noise mode is used.

The `ionCharge` specifies the charge of the ion hitting the Faraday cup. This value is required to convert the measured charge (or current) into number of ions.

Name	**Direct.**	**Type**	**Unit**
	Description		
frequency	OUT	double	Hz
	currently set ADC sampling frequency		
acquisitionLength	OUT	double	s
	currently set ADC sampling length		
gain	INOUT	int32_t	
	gain setting of the Femto amplifier (0–6)		
ionCharge	INOUT	int32_t	
	ion charge state required to convert current into particles		

2.1.5 ExpertSetting

Property	**multiplexed**	**visibility**
ExpertSetting	true	operational
Description		
Set Action	**partial**	**transaction**
ExpertSettingSetAction	true	true

The `ExpertSetting` property contains the expert settings for a single Faraday cup connected to a single ADC channel. Expert settings are those settings, which should not be changed ny the operator during normal operation. These include the ADC offsets, coupling and bandwidth of the FEMTO amplifier as well as definition of the analysis regions.

The `offset` setting defines the internal offset of the ADC. Typical values should be around 30000 to have the baseline in the middle of the ADC's dynamical range. Once set, there should be no reason for this setting to be changed during normal operation.

The `coupling` can be either 0 for DC coupling or 1 for AC coupling.

The `bandwidth` setting is in the range of 0 – 2 according to the following table:

setting	upper cut-off frequ. limit
0	full bandwidth
1	10 MHz
2	1 MHz

The `roi` settings specified three separate regions of interest used to determine the signal of the ADC in pulsed beam mode. Each region has a start and an end given in seconds

since the start of the measurement. Thus all three regions are defined by an array of six values. The first to entries define the first region, the next two the second region and the last to the third region.

The second region must contain the actual signal with beam from the cup. The first and third region should before and after the second region in a time of the measurement, when no beam is on the cup. They are used to determine the baseline of the ADC, which must be subtracted to get the actual signal current.

The three regions therefore depend on the timing of the beam cycle. Their limits are specified in fractions of the full sampling time and thus range from 0.0 (first sample) up to 1.0 (last sample).

Instead of using the `roi` settings, the signal measurement can be controlled by two machines events. This is the case if the field `roiFromEvents` is set to true. In this case the signal is integrated between these two machine events. If the baseline is not fixed, it is assumed, the the first 100μs and the last 100μs of the measured ADC data contain no signal and are thus used to determine the baseline.

The `opMode` setting defines the operation mode of the cup and FESA class. There are two operation modes: `PULSED` and `DC`. In `PULSED` mode, the FESA class expects a bunched beam, where the ADC also samples a short time before and after the bunch. This allows the determination of the ADC baseline with the three regions of interest as described above.

Baseline measurement can be suppressed by setting the `fixedBaseline` field to true. In this case a predetermined baseline stored in the instantiation file for each device is used.

Name	**Direct.**	**Type**	**Unit**
	Description		
offset	INOUT	int32_t	
	ADC baseline offset (0–65535)		
coupling	INOUT	COUPLING	
	coupling of the Femto amplifier (0/1)		
bandwidth	INOUT	BANDWIDTH	
	bandwidth limit of the Femto amplifier (0–2)		
opMode	INOUT	OP_MODE	
	beam operation mode of the cup		
roi	INOUT	double[6]	
	limits for three ranges of interest to determine baseline and signal in fractions of the sampling time		
roiFromEvents	INOUT	bool	
	if true, obtains the integration regions from the MEAS1 and MEAS2 events ignoring the roi set above		
fixedBaseline	INOUT	bool	
	if true, uses the predefined fixed baseline instead of calculating it from the data		

2.1.6 IOSetting

Property	multiplexed	visibility
IOSetting	true	operational
Description		
Set Action	**partial**	**transaction**
IOSettingSetAction	true	true

The `IOSetting` property is used to enable direct output to file of the measured data by the *CryCup* class. Direct output to file is an expert feature and not for standard operation. The file is a binary file in the BDIO tagged file format.

Note: *This property only enables output to file. The output must be explicitly started via the* `StartIO` *property of the global device.*

Name	Direct.	Type	Unit
	Description		
fileOutputActive	INOUT	bool	
	if trues write the measured data to file		
fileOutputTrace	INOUT	bool	
	if true also the full sampled ADC data is written instead of only the analyzed data		

2.1.7 Power

Property	multiplexed	visibility
Power	false	operational
Description		
Set Action	**partial**	**transaction**
PowerSetAction	true	true

The `Power` property can be used to switch the power of the FEMTO amplifier on or off.

Note: *Movement of the cups in or out of the beam is not handled by the* CryCup *class.*

Name	Direct.	Type	Unit
	Description		

power	INOUT	DEVICE_POWER	
	switch the power of the FEMTO amplifier on or off		

2.2 Device Acquisition

2.2.1 BaselineAcquisition

Property	**multiplexed**	**on change**	**subscribable**	**visibility**
BaselineAcquisition	false	true	true	operational
Description				
Get Action				
BaselineAcquisitionGetAction				

The `BaselineAcquisition` property returns the measured baseline and is notified after a baseline measurement was requested by the client. The baseline is determined by linear regression and has the form:

$$ADU = blIntercept + blSlope * sampleindex \tag{2.1}$$

Note: *This property is not notified, if the baseline is determined automatically for each measurement.*

Note: *The baseline actually used is always returned together with the data in the* `Acquisition` *property.*

Name	**Direct.**	**Type**	**Unit**
	Description		
blSlope	OUT	double	
	the slope of the base line		
blIntercept	OUT	double	
	the baseline value for sample 0		

2.2.2 Status

Property	**multiplexed**	**on change**	**subscribable**	**visibility**
Status	false	true	true	operational

Description
Get Action
StatusGetAction

This is the standard GSI status property. It defines 3 detailed status values:

Index	Label	Severity	Description
0	FileOut	INFO	if true, file output is in progress
1	MemoryOK	ERROR_ON_FALSE	if false, the FESA class could not allocate required memory
2	DataAcqOK	ERROR_ON_FALSE	if false, an error occured during data acquisition
3	SwitchOK	ERROR_ON_FALSE	if false, the power to the FEMTO amplifier is off
4	PowerOK	ERROR_ON_FALSE	if false, the FEMTO amplifier has no power
5	FCConnectOK	ERROR_ON_FALSE	if false, an error occured accessing the controller for the FEMTO amplifiers

2.2.3 Acquisition

Property	**multiplexed**	**on change**	**subscribable**	**visibility**
Acquisition	true	false	true	operational
Description				
Get Action				
AcquisitionGetAction				

The `Acquisition` property contains all the measured data and is notified by the `AcquireRTAction`. Besides the integrated values like charge or number of particles it also provides the measured ADC data in raw format as well as calibrated into a current.

Name	**Direct.**	**Type**	**Unit**
	Description		
frequency	OUT	double	Hz
	Sampling frequency of the ADC in Hz		

startTime	OUT	int64_t	$s/10^{-9}$
	Time of first data point in ns relative to start of cycle.		
blSlope	OUT	double	
	Slope of the baseline used		
blIntercept	OUT	double	
	Intercept of the baseline used		
vToAFactor	OUT	double	
	Conversion factor used to convert volts into amperes		
rawData	OUT	int32_t[]	
	Raw ADC data		
calData	OUT	double[]	A
	Calibrated ADC data: i.e. beam current. Calibration takes into account all device properties, but no beam properties.		
actualROI	OUT	double[6]	
	Actual ROI used to analyse the data		
roiFromEvents	OUT	bool	
	If true, the ROI has been determined by machine events		
roiCharge	OUT	double	C
	Total charge measured over the integration time.		
roiMeanCurrent	OUT	double	A/10
	Mean current over the integration time.		
roiMaxCurrent	OUT	double	A
	Maximum beam current during the integration time.		
roiMeanCurrentStddev	OUT	double	A/10
	Standard deviation of the mean beam current.		
roiParticles	OUT	double	
	Number of ions detected during the integration time. Requires valid ion charge setting from control system.		
charge	OUT	double	C
	Total charge measured between EVT_BEAM_ON and EVT_BEAM_OFF.		
meanCurrent	OUT	double	A/10
	Mean current measured between EVT_BEAM_ON and EVT_BEAM_OFF.		
maxCurrent	OUT	double	A
	Maximum beam measured between EVT_BEAM_ON and EVT_BEAM_OFF.		
meanCurrentStddev	OUT	double	A/10
	Standard deviation of the mean beam current measured between EVT_BEAM_ON and EVT_BEAM_OFF.		
particles	OUT	double	
	Number of ions detected between EVT_BEAM_ON and EVT_BEAM_OFF. Requires valid ion charge setting from control system.		

2.2.4 Version

Property	multiplexed	on change	subscribable	visibility
Version	false	false	false	operational
Description				
Get Action				
VersionGetAction				

This is the standard GSI version property.

2.3 Global Setting

2.3.1 StartIO

Property	multiplexed	visibility
StartIO	true	operational
Description		
Set Action	**partial**	**transaction**
StartIOSetAction	true	true

The `StartIO` command property starts the direct output of data to file by the *CryCup* class. Data is written to file — with one file per cycle — until the request number of cycles have been written or the `StopIO` command property has been set.

2.3.2 StopIO

Property	multiplexed	visibility
StopIO	true	operational
Description		
Set Action	**partial**	**transaction**
StopIOSetAction	true	true

The `StopIO` command property stops the direct output of data to file by the *CryCup* class.

2.3.3 Arm

Property	multiplexed	visibility
Arm	false	operational
Description		
Set Action	**partial**	**transaction**
ArmSetAction	true	true

The `Arm` command property allows the client to trigger the `ArmRTAction` which sets the Femto and ADC settings and arms the ADC. It is for expert debugging only!

2.3.4 Acquire

Property	multiplexed	visibility
Acquire	false	operational
Description		
Set Action	**partial**	**transaction**
AcquireSetAction	true	true

The `Acquire` command property allows the client to trigger the `AcquireRTAction` which reads out the ADC data. It is for expert debugging only!

2.3.5 DiagnosticSetting

Property	multiplexed	visibility
DiagnosticSetting	false	expert
Description		
Generic property which allows to diagnose any FESA classes		
Set Action	**partial**	**transaction**

This is the standard FESA `DiagnosticSetting` property.

Name	Direct.	Type	Unit
	Description		
enableDiagMode	INOUT	bool	

hostName	INOUT	char[32]	
portNumber	INOUT	int32_t	
requestConfig	IN	bool	
requestState	IN	bool	
fwkTopic	INOUT	DIAG_FWK_TOPIC	
customTopic	INOUT	DIAG_TOPIC	
traceDevices	INOUT	char[320]	
bypassActions	INOUT	char[320]	

2.3.6 SummaryIOSetting

Property	**multiplexed**	**visibility**
SummaryIOSetting	true	operational
Description		
Set Action	**partial**	**transaction**
SummaryIOSettingSetAction	false	false

The `SummaryIOSsetting` property configures the direct output of the data to file. For each beam cycle a file will be created containing the data of all devices handled by the instance of the FESA class.

Direct output to file is an expert feature and not for standard operation.

Name	**Direct.**	**Type**	**Unit**
	Description		
filePath	INOUT	char[MAX_PATH_LEN]	
	path for the files relative to the fixed base path		
fileOutputRequestCount	INOUT	int32_t	
	number of files (beam cycles) to be written		
fileOutputComment	INOUT	char[MAX_COMMENT_LEN]	
	comment written to each file		
fileBase	OUT	char[MAX_PATH_LEN]	
	The fixed base path for writing to file		

2.3.7 SummarySetting

Property	**multiplexed**	**visibility**
SummarySetting	false	operational
Description		
Set Action	**partial**	**transaction**
SummarySettingSetAction	true	true

The `SummarySetting` property contains the fields which modify all ADCs handled by an instance of the FESA class simultaneaously and thus may influence several Faraday cup devices together. Currently these settings are the sampling frequency, the sampling time and the pre-trigger time which are identical for all channels of a single ADC as well as for all ADCs handled by the instance of the FESA class. Pre-trigger tim and sampling length are specified in seconds. Their minimum and maximum value are also given by the proeprty.

To program the ADC, the FESA class converts the times into the number of samples based on the specified sampling frequency. The maximum and minimum values in samples (and not seconds) are a fixed property of the ADC used. Therefore, the minimum and maximum values in units of seconds as used in this property are dependent on the sampling frequency. Changing the sampling frequency may lead to illegal values of pre-trigger and/or sampling length.

In general, changing the sampling frequency is not recommended and generally not required.

Name	**Direct.**	**Type**	**Unit**
	Description		
frequency	INOUT	FREQUENCY	
	the sampling frequency of the ADC		
acquisitionLength	INOUT	double	s
	the acquisition length in time		
acquisitionLength_min	OUT	double	s
	the minimum acquisition length in time		
acquisitionLength_max	OUT	double	s
	the maximum acquisition length in time		
preTrigger	INOUT	double	s
	the time before the trigger which should be sampled		
preTrigger_min	OUT	double	s
	the minimum pre-trigger time		
preTrigger_max	OUT	double	s
	the maximum pre-trigger time		

2.4 Global Acquisition

2.4.1 DeviceDescription

Property	multiplexed	on change	subscribable	visibility
DeviceDescription	false	false	false	operational
Description				
GSI device description property				
Get Action				
none				

This is the standard GSI device description property.

Name	Direct.	Type	Unit
	Description		
propertyNames	OUT	char[][]	
deviceNames	OUT	char[][]	
globalDeviceName	OUT	char[]	
host	OUT	char[]	

Chapter 3

Custom Types

3.1 Enums

Name	Description		
	Symbol	Value	Description
TOL_CHECK_MODE			
	ABS	0	
	REL	1	

Name	Description		
	Symbol	Value	Description
DETAILED_STATUS_SEVERITY			
	INFO	0	
	WARNING_ON_FALSE	1	
	ERROR_ON_FALSE	2	

Name	Description		
	Symbol	Value	Description
FREQUENCY	enumeration for the possible ADC sampling frequencies		
	F_100MHZ	0	
	F_50MHZ	1	
	F_25MHZ	2	
	F_10MHZ	3	
	F_1MHZ	4	

Name	Description		
	Symbol	Value	Description
DEVICE_TYPE			

Name	Description		
	Symbol	Value	Description
	UNDEFINED	0	
	SCINT	1	
	IC	2	
	SEM	3	
	TRAFO	4	
	DCTRAFO	5	
	RTTRAFO	6	
	LXISCOPE	7	
	CUP	8	

Name	Description		
	Symbol	Value	Description
CAL_UNIT			
	A	0	
	cd	1	
	K	2	
	kg	3	
	m	4	
	mol	5	
	rad	6	
	s	7	
	V	8	
	m_s	9	
	Hz	10	
	NBCharges	11	
	A_s	12	
	C	90	
	Particles	91	
	Counts	92	
	ADC_value	93	
	arb_units	99	

Name	Description		
	Symbol	Value	Description
OP_MODE	Possible operation modes of the device.		
	PULSED	0	bunch beam operation
	DC	1	dc beam operation

Name	Description		
	Symbol	Value	Description
COUPLING	Possible coupling modes of the amplifier.		
	DC	0	
	AC	1	

Name	Description		
	Symbol	Value	Description
BANDWIDTH	Possible bandwidth settings of the amplifier.		
	FULL	0	
	F_10MHZ	1	
	F_1MHZ	2	

3.2 State-Enums

Name	Description		
	Symbol	Value	Description
DEVICE_STATUS			
	UNKNOWN	0	
	OK	1	
	WARNING	2	
	ERROR	3	

Name	Description		
	Symbol	Value	Description
DEVICE_POWER_STATE			
	UNKNOWN	0	
	ON	1	
	OFF	2	
	STANDBY	3	
	POWER_DOWN	4	
	POWER_UP	5	

Name	Description		
	Symbol	Value	Description
DEVICE_POWER			
	ON	1	
	OFF	2	

Name	Description		
	Symbol	Value	Description
DEVICE_CONTROL			
	REMOTE	0	
	LOCAL	1	

3.3 Bit-Enums

Name	Description		
	Symbol	Bit	Description
AQN_STATUS			
	b0	NOT_OK	
	b1	BAD_QUALITY	
	b2	DIFFERENT_FROM_SETTING	
	b3	OUT_OF_RANGE	
	b4	BUSY	
	b5	TIMEOUT	

3.4 Constants

Name	Type	Value	Description
MAX_ERROR_MESSAGE_LENGTH	uint32_t	256	
MAX_NUMBER_OF_ERROR_MESSAGES	uint32_t	16	
MAX_CYCLE_NAME_LENGTH	uint32_t	256	
MAX_VERSION_NAME_LENGTH	uint32_t	256	
MAX_DETAILED_STATUS_LABEL_LENGTH	uint32_t	30	
STATUS_FILEOUT	uint32_t	0	
STATUS_MEMORY	uint32_t	1	
STATUS_DATA_ACQ_OK	uint32_t	2	
DETAILED_STATUS_SIZE	uint32_t	3	
MAX_TIMESTAMP_LENGTH	uint32_t	32	
MAX_PATH_LEN	uint32_t	128	
MAX_FILE_LEN	uint32_t	320	
MAX_COMMENT_LEN	uint32_t	512	
DEVICE_NAME_LEN	uint32_t	64	
MAX_PROPERTY_LEN	uint32_t	512	

Appendix A

Revision History

Revision	Date	Author	Notes
0.1	2015-01-15	HBr	first draft
0.2	2015-03-23	HBr	updated to changes in the FESA class