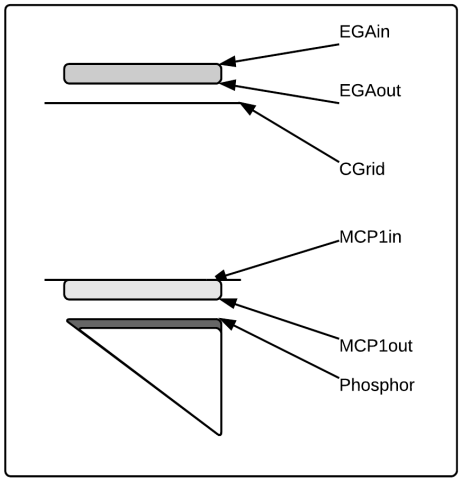
EDMS 1369556
Bogna Błaszczyk
CERN BE-BI-BL

FESA 3 BGI HV control

BGI_HVctrl Overview

This document describes high voltage control system implemented using FESA 3 framework[1] (version 1.2.1.) for ionisation gas monitors BGI. The main purpose of the class is to set and acquire voltage with respect to physical and logical constraints. In the typical detector setup 6 out 8 of the hv control card ports are used (in the picture), while MCP2in and MCP2out are disconnected. In one VME crate there are four hv cards and power supplies; for horizontal and vertical for beam 1 and beam2.
There are two main modes of operation: static, which ignores timing events and dynamic in which pulse by pulse modulation is applied to reference voltage.



Device Fields

Name	Type	Description	Default value	Multit-plexed[2]	Persistent [3]	Set/ acq
operatingMode	enum PpmMode: {DYNAMIC, STATIC}	In dynamic mode different HV values are applied for each cycle. In STATIC HV is totally independent of cycles.	STATIC	-	p	S
operatingMode_acq	enum PpmMode	Mode in the moment of acquisition	-	mux	-	A
rampSpeed_dynamic	double	[V/s] Maximum speed of raising high voltage on port for each cycle.	?	mux	p	S

[1] FESA3 Project Page

[2] Multiplexed = different value per each cycle.

[3] Persistent = saved between executions

Name	Type	Description	Default value	Multitplexed?	Persistent?	Set/acq
rampSpeed_static	double	[V/s] Maximum speed of raising high voltage on port.	?	-	p	S
rampSpeed_acq	double	[V/s] Ramp speed in the moment of acquisition	-	mux	-	A
isDataValid	bool	False if data was acquired but unsound.	-	mux	-	A
hvStateInCycle	enum StateInCycle: {MEASURE, STANDBY, OFF}	Behaviour per cycle, irrelevant in static mode.	STANDBY	mux	p	S
hvState	enum State {ON, OFF}	Hv behaviour.	ON	-	p	S
deviceStatus	enum Status: {ON, HV_OFF, SHUTDOWN, RAMP, MEASUREMENT, STANDBY, OFF_IN_CYCLE ERROR? }	Overall device status. Only data taken in MEASUREMENT is valid.	ON	-	-	A
lastShutdownCause	String (MAX_STRING_LENGTH)	What triggered the most recent shutdown; eg “EGP differential constraint not fulfilled”, “User request”	-	-	-	A
lastShutdownCauseInCycle	String (MAX_STRING_LENGTH)	What triggered the most recent shutdown for this cycle.	-	mux	-	A
lastShutdownCycle	String(32)	Name of the cycle in which the last shutdown occurred.	-	-	-	A
referenceHV_dynamic	double array [V]	Voltage values as set by the user for each cycle.	{1800, 1900, 2000, 2600, 0, 0, 5000}	mux	p	S

Name	Type	Description	Default value	Multitplexed?	Persistent?	Set/acq
referenceHV_static	double array [V]	Voltage values as set by the user.	{1800, 1900, 2000, 2600, 0, 0, 5000}	-	p	S
standbyHV	double array [V]	Reference voltage values used in standby mode.	{1800, 1900, 2000, 2600, 0, 0, 2000, 4000}	-	p	S
measuredHV	double array [V]	Voltage values as acquired from the hardware.	-	-	-	A
measuredCurrentIntensity	double array [A-6]	Current intensity values as acquired from the hardware	-	-	-	A
shouldCheckConstraints	bool	True, if differential constrained should be checked at all times and shutdown should be performed when they are not fulfilled.	true	-	p	S
Name	Type	Description	Default value	Multitplexed?	Persistent?	Set/acq
isPortUsed	bool array	True for the HV port in use during the operation. When false, no action will be done on this port.	{1, 1, 1, 1, 1, 0, 0, 1}	-	p	S
surveyFrequency	double [Hz]	Frequency of execution high voltage surveying procedure.	1.0	-	p	S
voltage_max	double array [V]	Maximal voltage allowed for each port.	{6000, 6000, 6000, 6000, 6000, 6000, 6000, 12000}	-	p	S
voltage_min	double array [V]	Minimal voltage allowed for each port.	8x {0}	-	p	S
maxDifferenceEGP	double [V]	Maximum difference between voltages on EGPin and EGPout.	800.0	-	p	S
maxDifferenceMCP1	double [V]	Maximum difference between voltages on MCP1in and MCP1out.	800	-	p	S

Name	Type	Description	Default value	Multitplexed?	Persistent?	Set/acq
maxDifference CGridEGPout	double [V]	Maximum difference between voltages on CGrid and EGPout	200	-	p	S
toleranceThreshold	uint32_t [%]	Value between 0 - 100, indicating how much feedback value of HV can vary from the reference.	10	-	p	S
adcWait	int32_t	[us] Length of wait for the ADC conversion.	25 000	-	p	S
beforeConvWait	int32_t	[us] Length of wait after conversion before reading the value from the register.	25 000	-	p	S
betweenReadoutsWait	int32_t	[us] Length of wait between reading consecutive ports.	25 000	-	p	S
fd	int32_t	File descriptor for BdiHvctl access driver.		-	-	A
next_cycle	char array	Dynamic mode: Name of the upcoming cycle as received from the SPS forewarning timing event.		-	-	A
extraction_received	bool	Dynamic mode: Describes whether the Extraction SPS event has been already received in this cycle.	false	-	-	A
Configuration fields						
hwAddress	logicalAddress	Lun and channel of the hv card.		Conf	Conf	Conf
VOLTAGE_ADC_IDS	int32_t array	Port identifiers to be passed to ADCs control register for voltage value acquisition.	{257,321,385,449, 257,321,385,449}			
CURRENT_ADC_IDS	int32_t array	Port identifiers to be passed to ADCs control register for	{289,353,417,481, 289,353,417,481}			

		current value acquisition.				
ADC_VOLTAGE_SCALING_FACTOR	double array	Scaling factor for each port of voltage value acquired from 12b ADC.	{1.628,1.628,1.628,1.628,1.628,1.628,1.628,3.256}			
ADC_CURRENT_SCALING_FACTOR	double array	Scaling factor of current value acquired from 12b ADC.	{1.221,1.221,1.221,1.221,1.221,1.221,1.221,2.442}			
DAC_VOLTAGE_SCALING_FACTOR	double array	Scaling factor of voltage value to be set to DAC.	{0.61425,0.61425,0.61425,0.61425,0.61425,0.61425,0.61425,0.30713}			
PORT_NAMES	char 2D array	Descriptive port names.	{EGPin,EGPout,CGrid,MCP1in,MCP1out,MCP2in,MCP2out,CGrid}			
hardwareDiagnosticMode	bool	If true, server will not be shut down when there is a problem with any of cards.	false	global field		
surveyFrequency	double [Hz]	Frequency of execution high voltage surveying procedure.	1.0	global field		

Metamodel documentation generation

Complete documentation of all the FESA fields, events and structures in the HTML format can be found in the class directory: BGI_HVctrl/docs/BGI_HVctrl.html

To regenerate the documentation:

- use Eclipse with FESA3 plugin.
- In class .design view press *Generate FESA3 Class Documentation* (3rd icon from the left) to create XML documentation base, if not yet present. You will create a file BGI_HVctrl/docs/BGI_HVctrl.xmldoc
- Open it, validate and generate HTML by pressing *Generate FESA3 class HTML documentation* (1st icon from the right).

Internal constants:

- IDs of HV ports as in the card :
  enum HvPorts {

0: EGPin, 1: EGPout, 2: CGrid, 3: MCP1in

4: MCP1out, 5: MCP2in, 6: MCP2out, 7: CGrid }

- Number of ports (also dimension for all the arrays):

  NR_OF_PORTS const = 8

Cycle division and timing events

Timing events in SPS

When set to the dynamic mode in SPS, 4 events are used:

- **Start cycle**

  SX.SCY-CT, C404
- **Extraction** Acquisition start slow ejection

  SEX.S-AMC-CT, C329
- **Extraction** Acquisition last fast extraction

  SEX.F-AMCLO-CT, C336
- Info about next cycle, at least 2.5 s before it starts

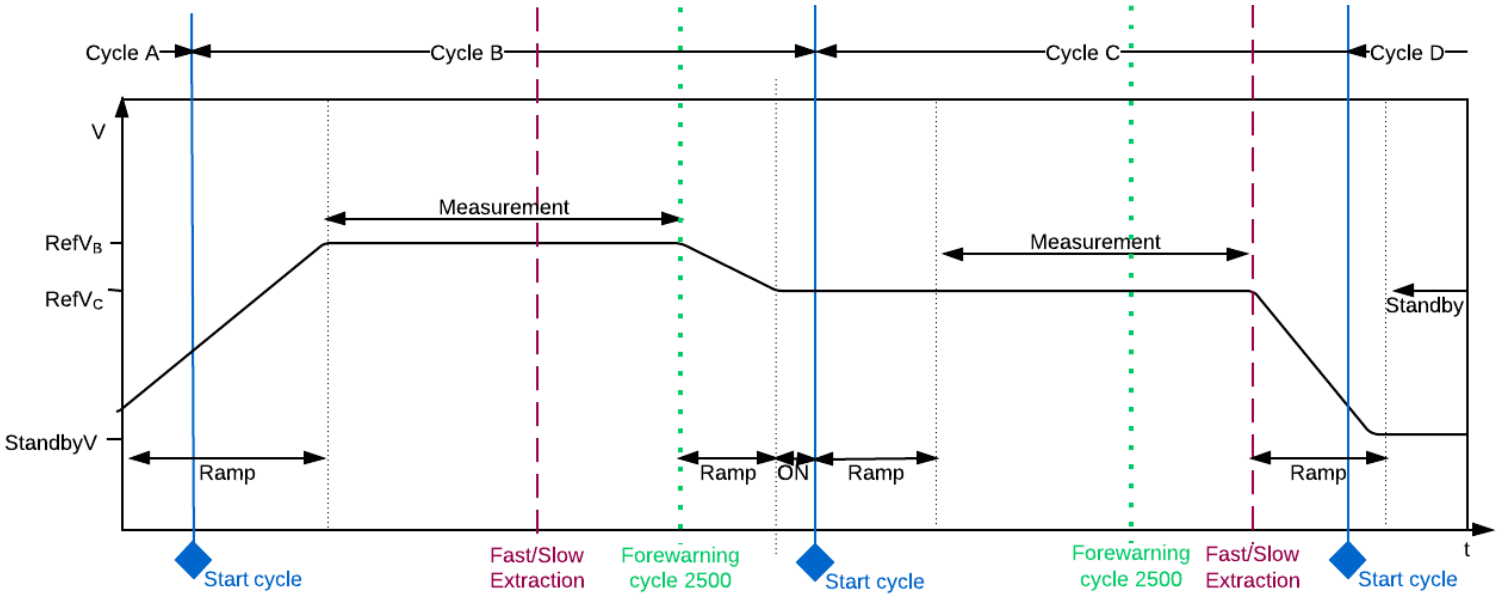
  Forewarning cycle 2500 [ms]

  SX.FCY2500-CTM, C342

After receiving extraction event, reference voltage is maintained till info about next cycle is received (forewarning). Knowing about next cycle, voltage is ramped to those values. If ramp finishes before the next cycle, state is set to ON, as voltage on ports is different than reference values for this cycle. Otherwise, ramp is continued in the next cycle. See more details in dynamic state diagram.

Example

Cycle A was set to OFF, cycle B to refV_B, cycle C to refV_C, cycle D to STANDBY



Constraints

Differential constraints

At any point (ramp, shutdown, measurement):

$$|measuredVoltage(EGPin) - measuredVoltage(EGPout)| <= maxDifferenceEGP$$

$$|measuredVoltage(MCP1in) - measuredVoltage(MCP1out)| <= maxDifferenceMCP1$$

$$|measuredVoltage(CGrid) - measuredVoltage(EGPout)| <= maxDifferenceCGridEGP$$

Applies also to reference voltage values set by user.

Boundary constraints

For each port, all the time:

$$referenceVoltage(port) <= voltage_{max}(port) \ and \ referenceVoltage(port) >= voltage_{min}(port)$$

Feedback

When voltage is supposed to be established (measurement, standby):

$$t = toleranceThreshold/100$$

$$measuredV(port) \in [referenceVoltage(port) * (1 - t), \ referenceVoltage * (1 + t)]$$

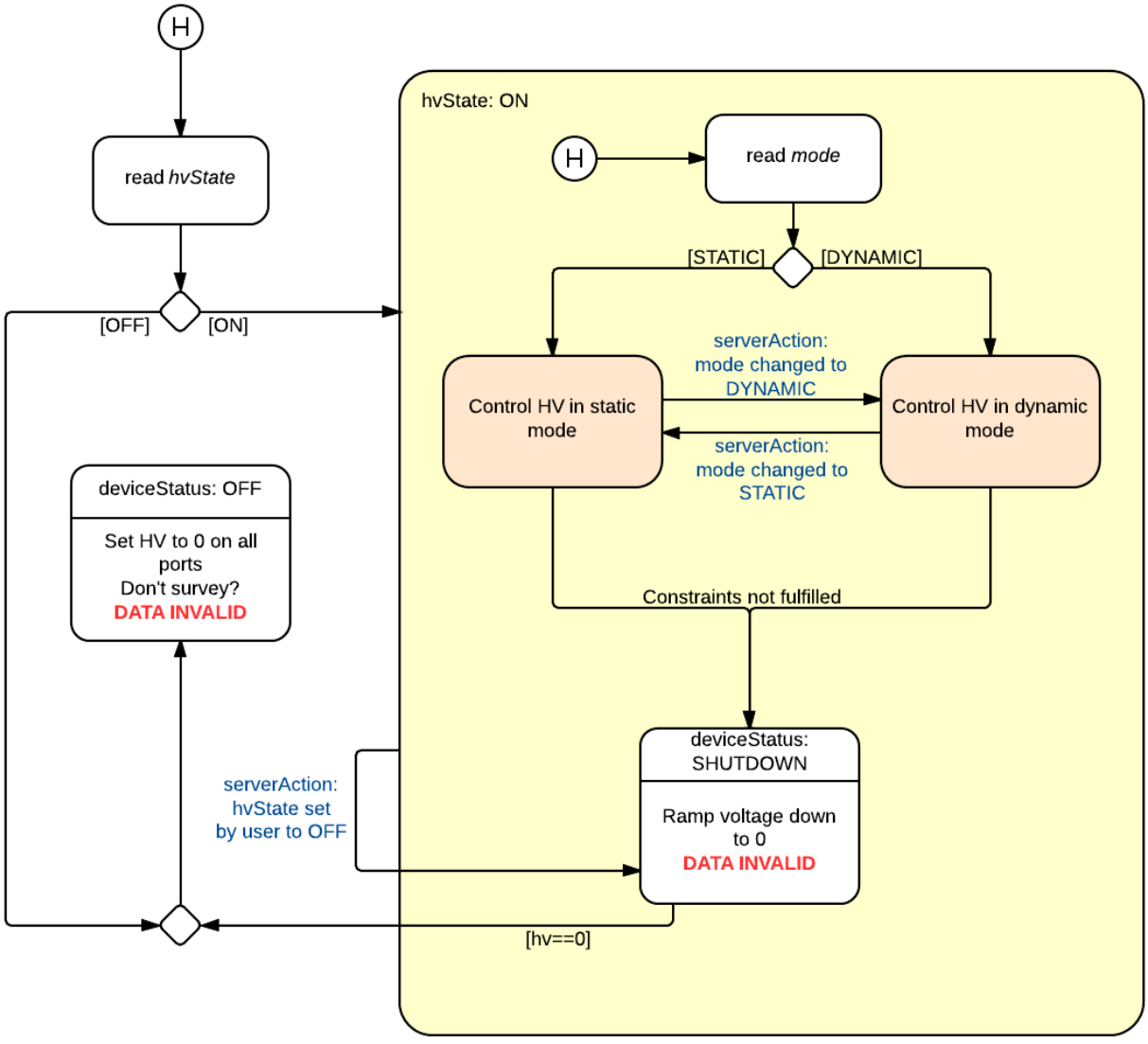
Ramp speed constraints

For each port, maximum speed of changing voltage on it is defined by value-item *rampSpeed* in property *ExpertSettings*. In DYNAMIC mode, *rampSpeed_dynamic* defines maximum speed separately for each cycle.

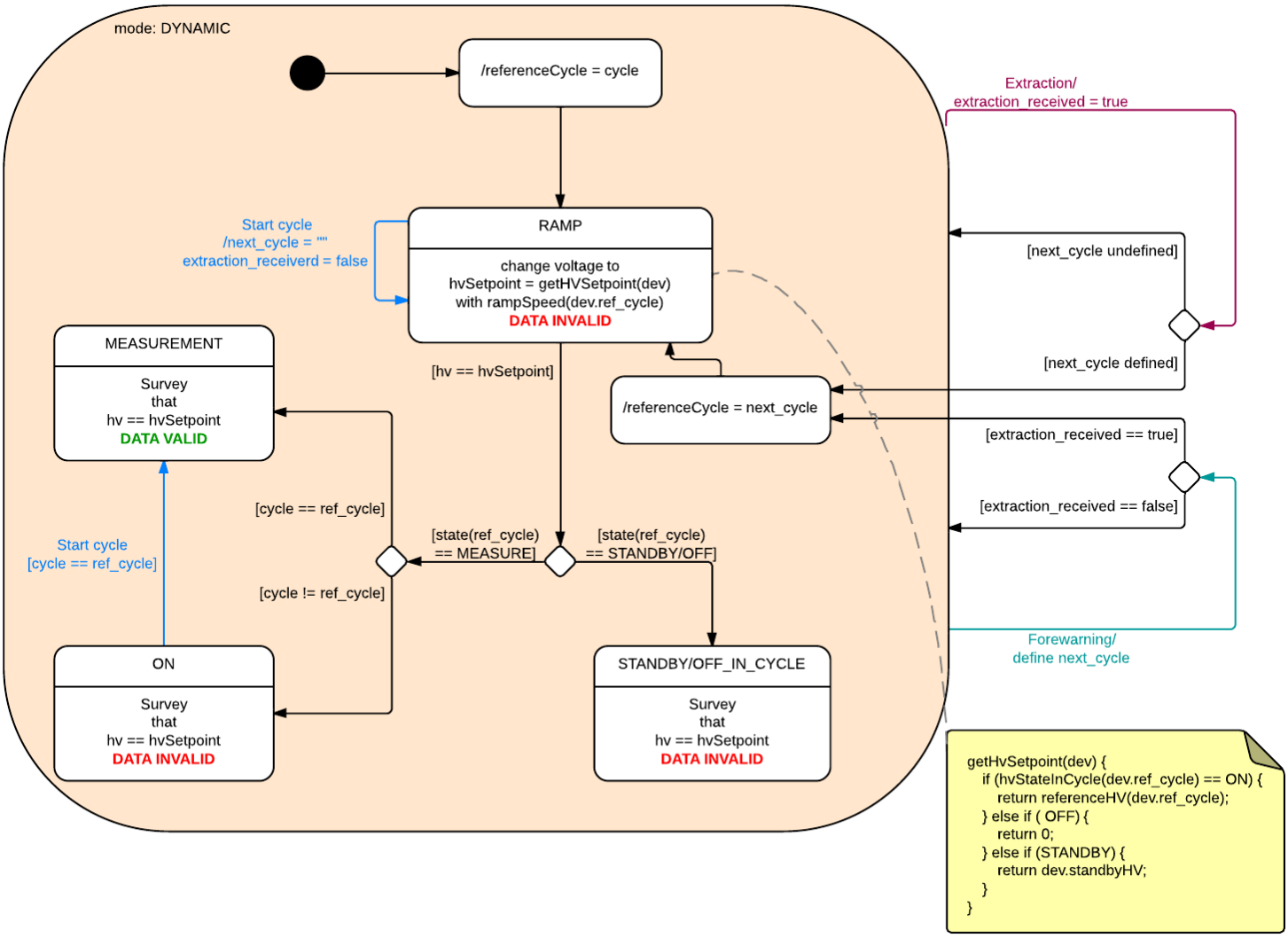
System state diagram

System functions as a finite state machine, result of a invoked action (ramp, survey, etc) depends on the state it is in. “Data invalid” marks states in which values on ports differ from the reference values, therefore making camera readouts unsound.

Overview:



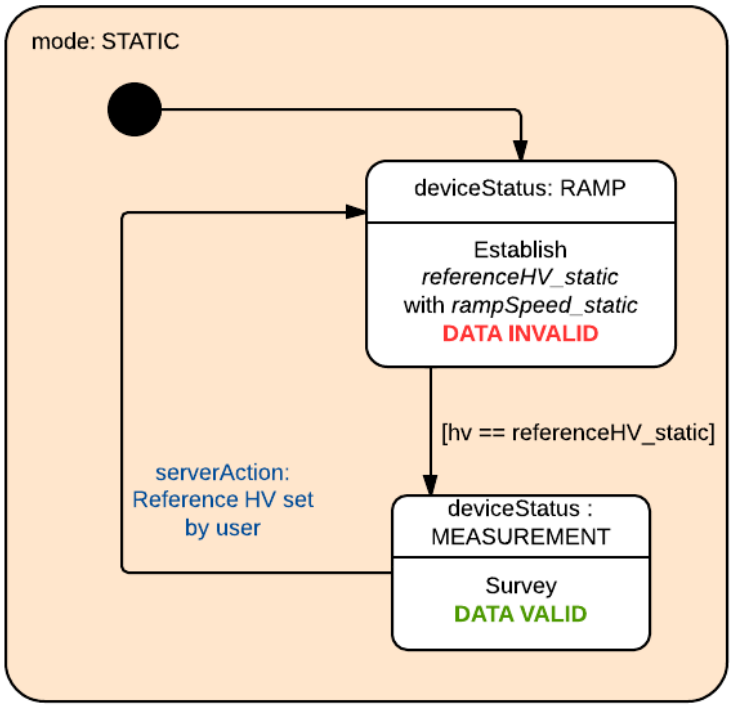
Detailed dynamic



In dynamic mode, high voltage setpoint is read from settings at beginning of every cycle and stored in an array field *hvSetpoint* to avoid changing the reference voltage while inside of the cycle. Field *referenceCycle* stores cycle name to which voltage is referenced at the moment as it is not always equivalent with the cycle currently played at the detector.

From the moment when both Extraction (either fast or slow) and Forewarning SPS timing events are received, voltage can be ramped to the value needed for next cycle. If this values are reached on ports before the next cycle arrives, device stays in ON mode, waiting for this event.

Detailed static



Interface

Properties with expert access are marked in **bold**.

- AcquisitionHV << *acquisition*>>
    - measuredVoltage : double array [N_PORTS]
    - measuredCurrent: double array [N_PORTS]
    - isDataValid : bool
    - operatingMode: enum PpmMode
    - rampSpeed

- Status <<*acquisition*>>
    - deviceStatus : enum Status
    - lastShutdownCause : enum/string
    - lastShutdownCauseInCycle
    - lastShutdownCycle
- **SetHV <<*setting*>>**
    - referenceVoltage_static : double array [N_PORTS]
- **SetHVDynamic <<*setting*>>**
    - referenceVoltage_dynamic : double array [N_PORTS]
- **ExpertSettings <<*setting*>>**

  partial setting: true
    - checkConstraints : bool
    - operatingMode : enum PpmMode
    - standbyHV : double array[N_PORTS]
    - rampSpeed_static : double
    - isPortUsed: bool array[N_PORTS]
    - voltage_max : double array [V]
    - voltage_min : double array [V]
    - maxDifferenceEGP : double
    - maxDifferenceMCP1 : double
    - maxDifferenceCGridEGPout : double
    - toleranceThreshold : uint32_t
- **ExpertSettingsDynamic <<*setting*>>**

  partial setting: true
    - hvStateInCycle: enum StateInCycle
    - rampSpeed_dynamic : double
- **ExpertSettingControlerCard <<*setting*>>**

  partial setting: true
    - adcWait : uint32_t
    - afterConvWait : uint32_t
    - betweenReadoutsWait : uint32_t
- State <<*setting*>>

  patial setting: true
    - state : enum State
- **ImmediateShutdown <<command>>**

Remarks:

Safe Set

Requirement is to have to have two ways of setting the reference voltage: with and without constraint check. In the latter case, when trying to set reference values not complying to differential constraints, exception is thrown. Preferably first option would be used by operators and experts for most of the time, to avoid mistakes. Second would be used in exceptional cases.
It is implemented by bool value-item *checkConstraints* in *ExpertSettings* property. When set to *true*, setting values by the user not complying to constraints is not allowed and if such values measured during runtime, shutdown is performed.

Errors

Logical errors:

- Differential constraints not met at any point when *checkConstraints* = true-> shutdown
- Ramp speed not met (within tolerance) -> shutdown
- Measured voltage too far from the reference voltage (within tolerance) -> shutdown
- Boundary constraints not met -> shutdown

Hardware errors:

- Cannot connect to the card (cannot acquire file descriptor) ->
    - State set to ERROR , error message (lastCauseOfShutdown) is put and turn off all devices on this crate.
- Cannot control the card (incorrect value or relayControl register) ->
    - State set to ERROR , error message (lastCauseOfShutdown) is put and turn off all devices on this crate.

Note: Setting *hardwareDiagnosticMode* global configuration variable to *true* prevents turning off all the devices on crate, when problem with any of the cards has been discovered. Use it carefully only for diagnostics and never in operation.

User errors:

- Reference voltage set not within [min_v, max_v] -> Exception is thrown

- Reference voltage not complying to constraints -> Exception is thrown [when *checkConstraints* = true]
- Survey frequency unfeasible with given adc waits -> throw exception but set value
- Waits too big in respect to given survey frequency -> throw exception but set value
- DYNAMIC mode set on non-multiplexed device -> throw exception

Use cases

#Set high voltage value independent of the changing cycles

Set mode to STATIC, define rampSpeed_static. Define referenceHV_static for every port.

#Pulse by pulse modulation in multiplexed environment

Set mode to DYNAMIC, define rampSpeed_dynamic for every cycle and referenceHV_dynamic for every cycle.

#Extension point: Stay idle for the selected cycles

(hvStateInCycles in STANDBY by default for all cycles). Change hvStateInCycles to STANDBY or OFF for selected cycles.

#Static mode in multiplexed environment *extends Use case 1: Static mode*

Timing events are ignored, hvStateInCycle not used, all dynamic properties values are ignored.

#Dynamic mode in non-multiplexed environment [Configuration error]

If such setting is discovered, exception is thrown.

#Read values of measured voltage and current

Use property AcquisitionHV. In proper functioning, returned values has been acquired with at most 1/surveyFrequency [s] delay.

#Turn off device

Change State to OFF to begin incremental shutdown procedure.

#Find cause of the last shutdown

In acquisition property ExpertStatus, lastShutdownCause returns name of the operation performed and first

violated constraints that cause the shutdown.

#Turn off selected ports

Use property *ExpertSettings*. By default, isPortUsed is set to *true* for all ports except MCP2in and MCP2out.
Change the value for selected ports.

#Modify differential constraints

In property ExpertSettings change the value of maxDifference*.

#Modify maximum and minimum values for ports

Use property ExpertSettings. Change the value of voltage_max and voltage_min for each port.

#Perform immediate shutdown (without steps)

Use command Immediate Shutdown.

#React to the Image class requests

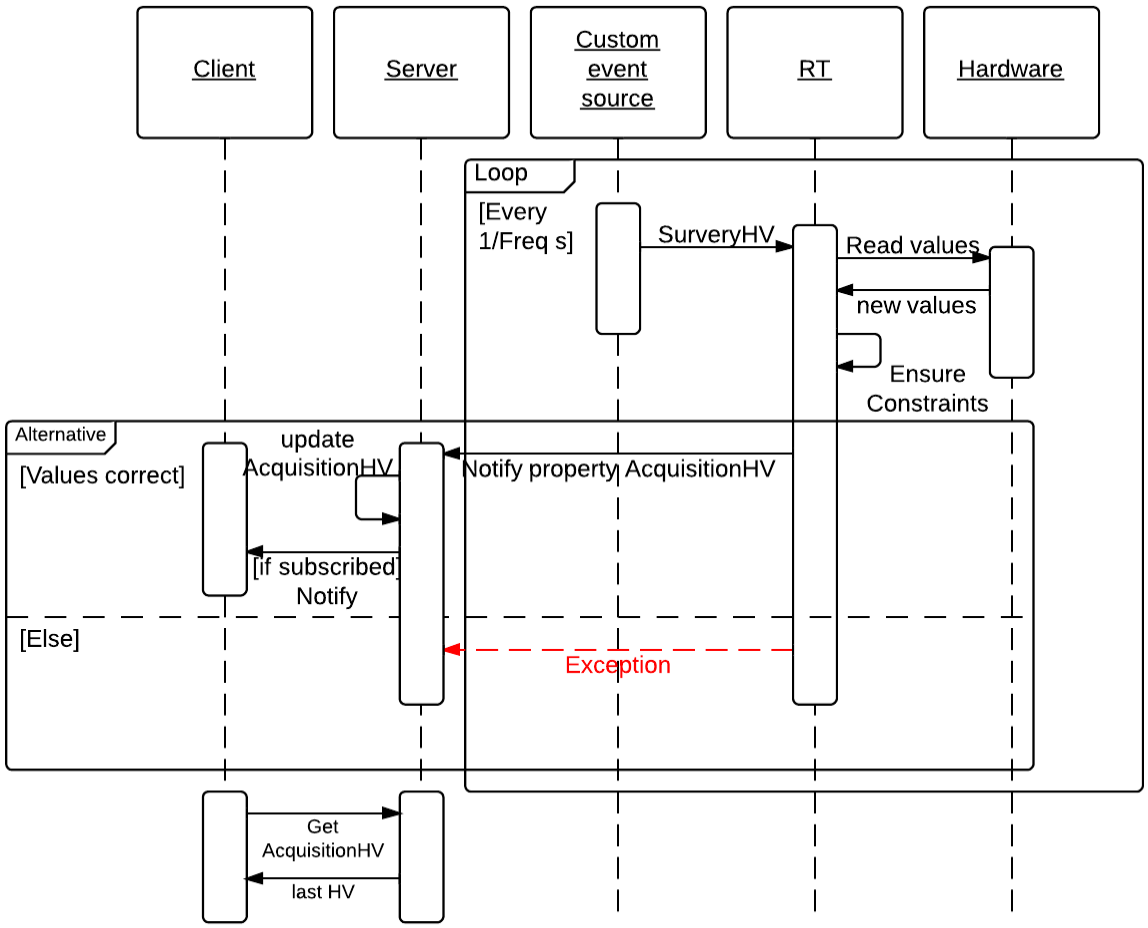
Set the image class property requestShutdown to true.

#Change voltage value in the dynamic mode

Changes will be applied in the next occurrence of the cycle for which you're setting the value (not immediately).

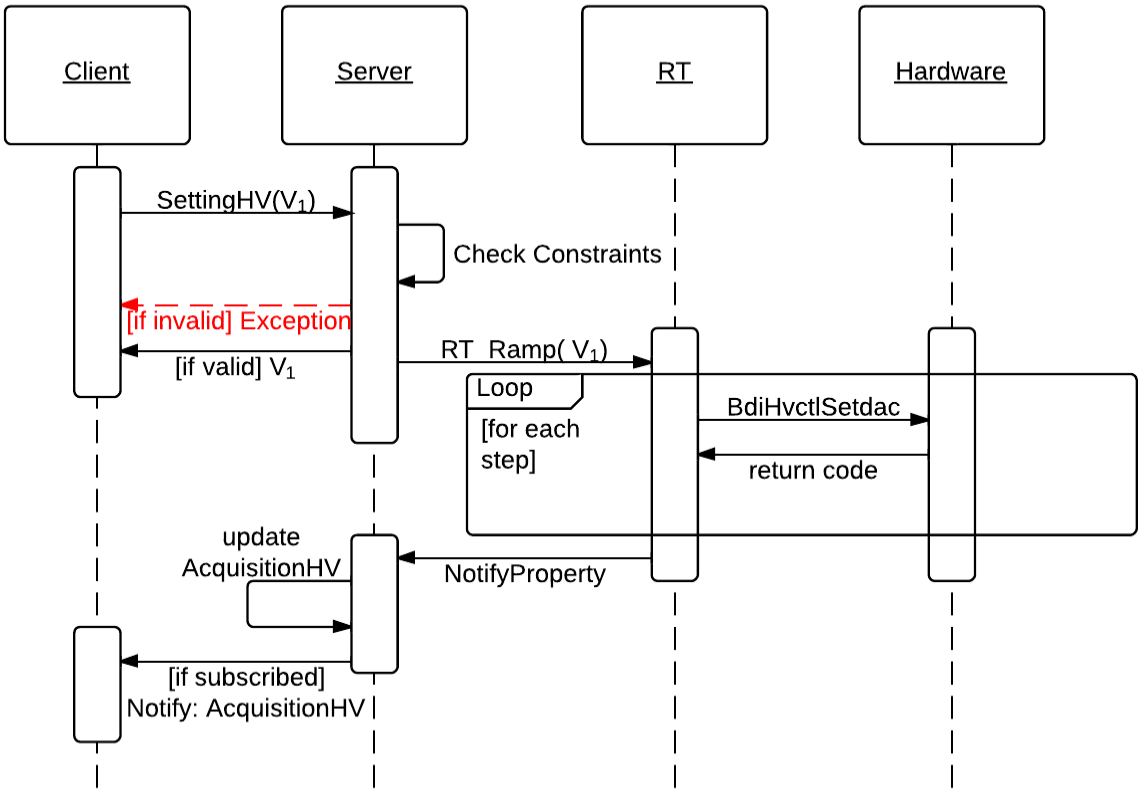
Actions outline

Get



When reading high voltage and current using property AcquisitionHV, the value returned immediately is the last one read by a Survey action, whether it is during a ramp or maintaining voltage values.

Set



When setting values on the ports, first it is ensured that they conform to the constraints, if so then the ramp procedure is invoked, setting values on ports gradually to avoid uncontrolled jumps and damaging voltage differences between ports.

Ramp

According to the doc breakdown voltage can occur due to:

- drop
- oscillation
- block
- power supply inertia

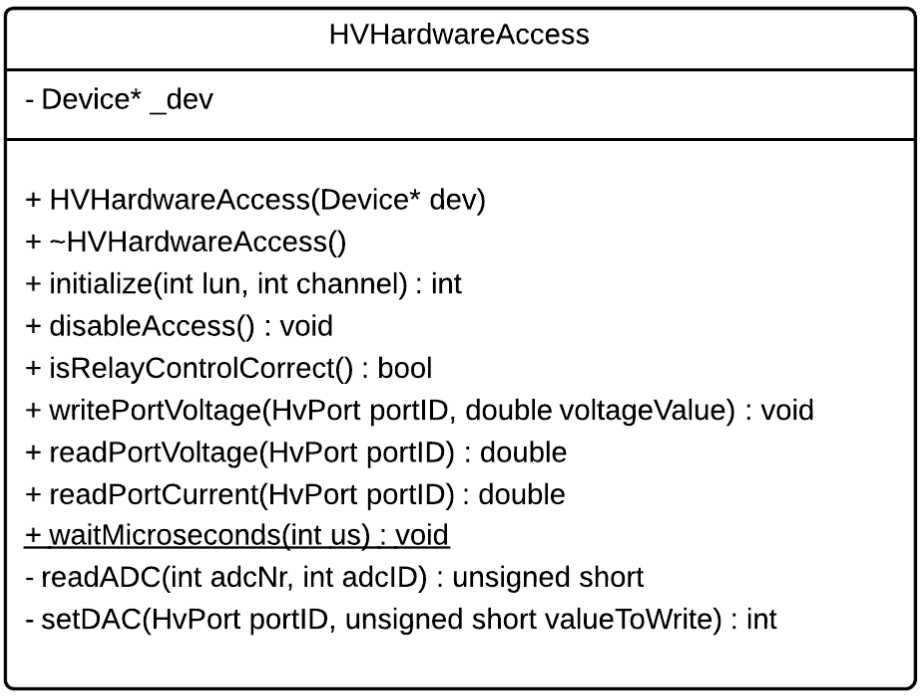
Ramp limitations:

- power supply inertia
- differential constraints should not be overpassed at any point

Hardware abstraction

In case of situation when there will be different hardware on particular instances (eg new cards mounted only in one accelerator). For each RT action can be defined selection-criterion, which selects devices for which it should be invoked. So you can have simultaneously instances using different drivers on different hardware.

HVHardware access is a class which communicates with card using driverGenII BdiHvctlIoctlAccess library:



Usage

`HVHardwareAccess`	Constructor initializes the class by storing a Device pointer.
`initialize`	`Obtain device descriptor and stores it in field Device.fd. Used only once per card at server startup.`
`disableAccess`	`Release device descriptor.`
`isRelayControlCorrect`	`Reads relay control register and returns true, if read value was correct.`

writePortVoltage	Convert and set voltage on selected port (immediately).
readPortVoltage	Read and convert voltage from selected port.
readPortCurrent	Read and convert current from selected port.
waitMicroseconds	Convenience wrapper for nanosleep

References

- FESA3 Documentation: https://wikis.cern.ch/display/FESA3/Documentation
- High voltage doc from Jan (translated to English by Ana) : EDMS1369845
- Timing events in SPS : EDMS1369846
- HV card design notes: \\cern.ch\dfs\Departments\AB\Groups\BI\Sections\BL\Perso\Jan\BWS\BWS-LHC\Electronics\VMECrate\Cards\HVControl