



TANGO Device Server

BILT multi channel power supply User's Guide

Bilt Class

Revision: Bilt-Release_3_9 - Author: meier
Implemented in C++ - CVS repository: ESRF

Introduction:

The BILT multi channel power supply will be used with 3 channels for the ESRF fast feedback systems. The power supply can be controlled in 2 ways : By this device server for slow or human control or by the Libera Beam Position monitors for fast feedback with closed loop control. The devices are exported as one device per steerer magnet. Every device has three channels. The channels (poles) cannot be switched on individually but have their individual set points and measurement values.

Class Inheritance:

- Tango::Device_4Impl
  - Bilt

Properties:

Device Properties		
Property name	**Property type**	**Description**
IPAddress	Tango::DEV_STRING	IP Address of BILT controller. Each controller includes several magnets and each magnet includes 3 power supplies, for corrections in the X, Y and Z planes.
MagnetNumber	Tango::DEV_SHORT	Magnet number from 1 to 48
UpdatePeriod	Tango::DEV_LONG	The update period for the data reading from the power supply in milli seconds.

Device Properties Default Values:

Property Name	Default Values
IPAddress	0.0.0.0
MagnetNumber	0
UpdatePeriod	1000

There is no Class properties.

States:

States	
Names	**Descriptions**
ON	Magnet is ON, all three channels are ON.
OFF	Magnet is OFF, all three channels are OFF.
FAULT	Error detected by the BILT magnet.
ALARM	Alarm detected on one or more channels.
UNKNOWN	Communication fault

Attributes:

Scalar Attributes			
Attribute name	**Data Type**	**R/W Type**	**Expert**
Location: The pysical location of the power supply in the form:	DEV_STRING	READ	No

Spectrum Attributes			
Attribute name	**Data Type**	**X Data Length**	**Expert**
Currents: The measured currents of the three poles.	DEV_DOUBLE	3	No
Voltages: The measured voltages of the three poles.	DEV_DOUBLE	3	No
Impedances: Calculated impedances for the three poles.	DEV_DOUBLE	3	No
SetCurrentsRMS: Statistic when driven by the Libera. One RMS value for every pole of the magnet. RMS value of the AC setpoint applied on the DAC during the last second. X=sqr((sum(setAC)*sum(setAC)) / n - ((sum(setAC)/n) *(sum(setAC)/n)))	DEV_DOUBLE	3	No
SetCurrentsAverage: Statistic when driven by the Libera. One average value for every pole of the magnet. Average value of the AC setpoint applied on the DAC during the last second.	DEV_DOUBLE	3	No
FramesPerSecond: Number of settings applied during the last second	DEV_ULONG	3	No
ErrorsPerSecond: Errors detected during the last second	DEV_ULONG	3	No
ErrorCounters: Statistic when driven by the Libera. The total number of errors since the last reset for the three poles.	DEV_ULONG	3	No
LocalControls: Indicated whether the individual channels are in local control mode for the AC current control. True = local control activated.	DEV_BOOLEAN	5	No
Temperatures: Temperature measurements for the three poles.	DEV_DOUBLE	6	No

Commands:

More Details on commands....

Device Commands for Operator Level		
Command name	**Argument In**	**Argument Out**
Init	DEV_VOID	DEV_VOID
State	DEV_VOID	DEV_STATE
Status	DEV_VOID	CONST_DEV_STRING
On	DEV_VOID	DEV_VOID
Off	DEV_VOID	DEV_VOID
Reset	DEV_VOID	DEV_VOID
SetpointCheck	DEVVAR_DOUBLEARRAY	DEV_BOOLEAN
EnableAcCurrent	DEV_VOID	DEV_VOID
DisableAcCurrent	DEV_VOID	DEV_VOID

Device Commands for Expert Level Only		
Command name	**Argument In**	**Argument Out**
SetPoleAcCurrent	DEVVAR_DOUBLEARRAY	DEV_VOID
LocalPoleControl	DEVVAR_SHORTARRAY	DEV_VOID
SetPoleCurrent	DEVVAR_DOUBLEARRAY	DEV_VOID

1 - Init

- **Description:** This commands re-initialise a device keeping the same network connection.
  After an Init command executed on a device, it is not necessary for client to re-connect to the device.
  This command first calls the device *delete_device()* method and then execute its *init_device()* method.
  For C++ device server, all the memory allocated in the *nit_device()* method must be freed in the *delete_device()* method.
  The language device desctructor automatically calls the *delete_device()* method.

- **Argin:**
  DEV_VOID : none.

- **Argout:**
  DEV_VOID : none.

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

2 - State

- **Description:** This command gets the device state (stored in its *device_state* data member) and returns it to the caller.

- **Argin:**
  DEV_VOID : none.

- **Argout:**
  DEV_STATE : State Code

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

3 - Status

- **Description:** This command gets the device status (stored in its *device_status* data member) and returns it to the caller.

- **Argin:**
  DEV_VOID : none.

- **Argout:**
  CONST_DEV_STRING : Status description

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

4 - On

- **Description:** Turns on power supply.

- **Argin:**
  DEV_VOID :

- **Argout:**

DEV_VOID :

- **Command allowed for:**
  - Tango::OFF

5 - Off

- **Description:** Turns of power supply

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

6 - Reset

- **Description:** Reset faults and alarms

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

7 - SetPoleAcCurrent (for expert only)

- **Description:** Set the AC current for one Pole. [0] = pole number, [1] = current

- **Argin:**
  DEVVAR_DOUBLEARRAY : [0] = pole number, [1] = current

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

8 - LocalPoleControl (for expert only)

- **Description:** Set the AC current for one Pole. [0] = pole number, [1] = local control true/false

- **Argin:**
  DEVVAR_SHORTARRAY : [0] = pole number, [1] = local control true/false

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::ALARM

9 - SetpointCheck

- **Description:** Check the given current values against the specified limits.

- **Argin:**
  DEVVAR_DOUBLEARRAY : Setpoint values for the Poles

- **Argout:**
  DEV_BOOLEAN : true when settings are OK

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

10 - SetPoleCurrent (for expert only)

- **Description:** Set the current for one Pole. [0] = pole number, [1] = current

- **Argin:**
  DEVVAR_DOUBLEARRAY : [0] = pole number, [1] = current

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::OFF
  - Tango::FAULT
  - Tango::ALARM
  - Tango::UNKNOWN

11 - EnableAcCurrent

- **Description:** Enables the dynamic 10kHz AC current settings from the Liberas.

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::ALARM

12 - DisableAcCurrent

- **Description:** Disables the dynamic 10kHz AC current settings from the Liberas.

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**
  - Tango::ON
  - Tango::ALARM

TANGO is an open source project hosted by : SOURCEFORGE.NET®	Core and Tools : CVS repository on tango-cs project Device Servers : CVS repository on tango-ds project