



TANGO Device Server

ocem septum powersupply control User's Guide

OcemPS Class

Revision: - Author: chaize
Implemented in C++ - CVS repository: ESRF

Introduction:

This class interfaces the septum control powersupply from Ocem. At ESRF it is used to control the storage ring injection septums. This class is a refurbished version of the former TACO ocem class.

Class Inheritance:

- Tango::DeviceImpl
  - PowerSupply
    - OcemPS

Properties:

Device Properties		
Property name	**Property type**	**Description**
Set_gain	Tango::DEV_DOUBLE	the current setpoint sent to the powersupply (setval) is equal to val = desired_current_in_A * set_gain + set_offset
Set_offset	Tango::DEV_DOUBLE	the current setpoint sent to the powersupply (setval) is equal to val = desired_current_in_A * set_gain + set_offset
Read_gain	Tango::DEV_DOUBLE	the current readpoint in A (Current) is equal to Current = read_from_PS * read_gain + read_offset
Read_offset	Tango::DEV_DOUBLE	the current readpoint in A (Current) is equal to Current = read_from_PS * read_gain + read_offset
Serialline	Tango::DEV_STRING	name of the serial line object on which the powersupply is connected

Device Properties Default Values:

Property Name	**Default Values**
Set_gain	0.5
Set_offset	-25
Read_gain	2.5
Read_offset	50
Serialline	/dev/ttyI1

There is no Class properties.

Attributes:

Scalar Attributes			
Attribute name	**Data Type**	**R/W Type**	**Expert**
Current: The powersupply current setting in Amps	DEV_DOUBLE	READ_WRITE	No
Voltage: The powersupply voltage in volts.	DEV_DOUBLE	READ_WRITE	No
CurrentSetPoint: The current set value as stored in the powersupply.	DEV_DOUBLE	READ	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

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:**

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:**

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:**

4 - On

- **Description:** Switch powersupply ON.

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**

5 - Off

- **Description:** Switch powersupply OFF.

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**

6 - Reset

- **Description:** Reset the powersupply to a well known state.

- **Argin:**
  DEV_VOID :

- **Argout:**
  DEV_VOID :

- **Command allowed for:**

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