1. **Introduction**

CopleyControl is a Device Server of copley controller to control the motors. It is based of PySerial Device Server, which transfer the ASCII messages to the controllers.

2. **Operating environment and software support:**

Debian 9 system.

Python 2.7.

Jive.

Tango, PyTango

CME 2 (copley control software)

3. **How to use them.**

3.1 Starten jive:

Command: “jive”

3.2 Starten PySerial DS:

Command: “python PySerial.py test”

In addition, a new device “pyserial/hhl/1” will be added in PySerial DS.

NOTE: Serial port is defined as “/dev/ttyS0” in the “\_\_init\_\_()” method in “PySerial.py”. If the serial port is different or the operating system is different, then the serial port should be changed manually.

3.3. Starten CopleyControl DS:

Command: “python CopleyControl.py test”

In addition, a new device “copley/hhl/1” will be added in CopleyControl DS.

4. **CopleyControl DS interface**

The copley control DS has 7 Commands and 8 attributes.

7 Commands are: Init, Move, SendCommandGetResult, State, Status, Stop, Write.

Init: init the parameters.

Currently, there is nothing in the init method. Due to the memory of the instrument, I did not initialize the parameters of the instrument, all original parameters are obtained from the instrument. If we want to change some parameters like position, acceleration, deceleration, velocity, we can change them in the Attributes.

Move: trigger the motor to move.

If nothing is changed, the motor moves according to the memory parameters of the instrument. The parameters including acceleration, deceleration, velocity and position(actually, relative position) can be also changed in Attributes using write method of each attribute. Once man write new values in Attributes, the new parameters take effect immediately. For example, write 200000 into Position, then the position is 200000. When we use “Move”, the motor moves relative position of 200000.

SendCommandGetResult: write a command such as “s r0xca 10000\n” and get the result of this command from the motor controller. (This is for testing)

State: if the motor is in motion, the State is “MOVING”;

if the motor is stationary, the State is “ON”;

if the motor is out of power, the State is “OFF”;

if error happens, the State is “Status is FAUlT”.

Status: if the motor is in motion, the State is “Status is MOVING”;

if the motor is stationary, the State is “Status is ON”;

if the motor is out of power, the State is “Status is OFF”;

if error happens, the State is “Status is FAULT”.

Stop: stop the motion of the motor immediately.

Write: write ASCII message to the controller directly. (This is for testing)

8 Attributes are: Acceleration, Deceleration, NodeID, Port, Position, State, Status, Velocity.

Acceleration:

Read: read the real acceleration

Write: write a new value of acceleration into the motor controller immediately.

Deceleration:

Read: read the real deceleration

Write: write a new value of deceleration into the motor controller immediately.

NodeID:

Read:

read the node\_id, which is set by the first time when you add a new device. It can not be changed.

Port:

Read:

read the serial port the motor connects with. It can be not changed in this interface, because it is set in PySerial DS. If you want to change the serial port, you should change the code of PySerial DS.

Position:

Read: read the real position

Write: write a new value of position into the motor controller immediately.

Velocity:

Read: read the real velocity

Write: write a new value of velocity into the motor controller immediately.

State:

Read:

if the motor is in motion, the State is “MOVING”;

if the motor is stationary, the State is “STANDBY”;

if the motor is out of power, the State is “OFF”;

if error happens, the State is “FAULT”.

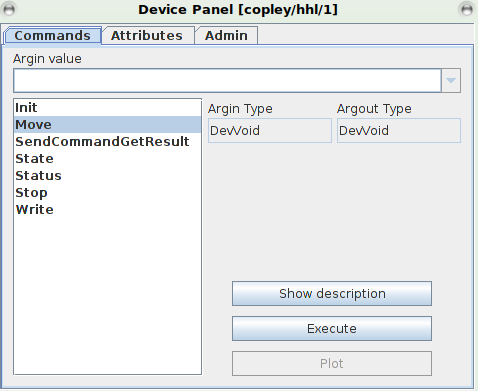
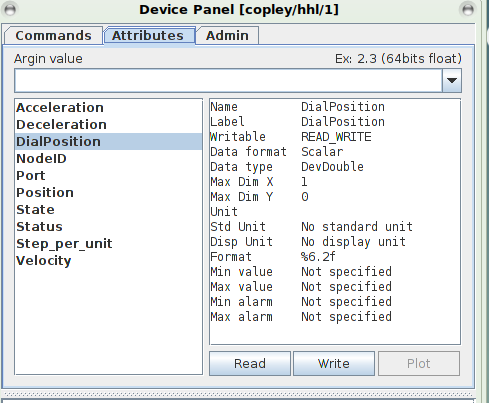
Status:

Read:

if the motor is in motion, the Status is “Status is MOVING”;

if the motor is stationary, the Status is “Status is STANDBY”;

if the motor is out of power, the Status is “Status is OFF”;



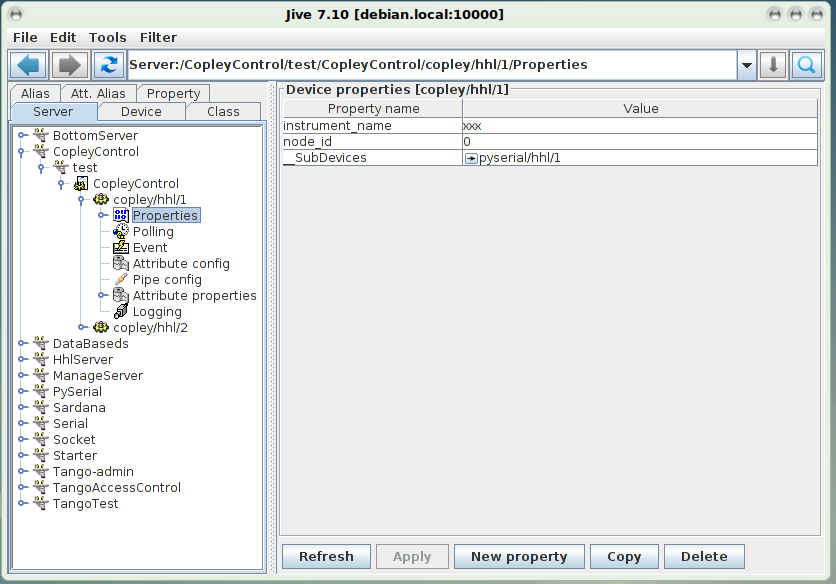
| **CopleyControl Class Commands** | | | |
| --- | --- | --- | --- |
| Commands | Input Type | Output Type | **Description** |
| Init | DevVoid | DevVoid | init the device |
| Move | DevVoid | DevVoid | trigger the motor to move. |
| SendCommandResult | DevString | DevString | write a command and get the result of this command from the motor controller. |
| State | DevVoid | State | if the motor is in motion, the State is “MOVING”;  if the motor is stationary, the State is “ON”;  if the motor is out of power, the State is “OFF”; |
| Status | DevVoid | DevString | if the motor is in motion, the State is “Status is MOVING”;  if the motor is stationary, the State is “Status is ON”;  if the motor is out of power, the State is “Status is OFF”; |
| Stop | DevVoid | DevVoid | stop the motion of the motor immediately. |
| Write | DevString | DevVoid | write ASCII message to the controller directly |

| **CopleyControl Class Attributes** | | | | |
| --- | --- | --- | --- | --- |
| Attributes | **R/W type** |  | Data type | Value, Range, Unit | **Description** |
| Acceleration | READ\_WRITE | DevDouble | Units: 10 counts/s2 | the acceleration |
| Deceleration | READ\_WRITE | DevDouble | Units: 10 counts/s2 | the deceleration |
| NodeID | READ | DevShort | Range: 0-127 | the CAN node address |
| Port | READ | DevString | Default value: “/dev/ttyS0” | the serial port |
| Position | READ\_WRITE | DevDouble | Units: Counts | the relative position |
| Velocity | READ\_WRITE | DevDouble | Units: 0.1 counts/s | the velocity |
| State | READ | State |  | device state.  if the motor is in motion, the State is “MOVING”;  if the motor is stationary, the State is “STANDBY”;  if the motor is out of power, the State is “OFF”; |
| Status | READ | DevString |  | device status.  if the motor is in motion, the State is “Status is MOVING”;  if the motor is stationary, the State is “Status is ON”;  if the motor is out of power, the State is “Status is OFF”; |

Device Properties are defined when we add Device(motor) to the CopleyControl DS at the first time. As for the node\_id, each control command consists of node\_id, so node\_id must be given. \_SubDevices “pyserial/hhl/1” is a device of PySerial DS which we connect.

instrument\_name :

we can define the name of the motor as you wish

 node\_id:

we must give the node\_id for the motor.

\_SubDevices:

It is set by the internal system. The system can check the connection between the devices.