

# Galil Motion Controller

*Auteur : J. Coquet*

*Version courante du document : 1.2*

*Date de création document : Aout 2003*

*Dernière modification le 05/01/2004 18:49*

## Historique des modifications

Date	Revision	Description	Author	Reader
08/2003	1.0	Initial Version	J. Coquet	A. Buteau
10/2003	1.1	Rewriting	Id/A.Thompson	Thompson
12/2003	1.2	Minor modification after LUCIA installation	A.Buteau	A.Buteau

<b>1</b>	<b><u>APPLICATION GOALS</u></b>	<b>2</b>
<b>2</b>	<b><u>TECHNICAL SCHEME OF THE APPLICATION</u></b>	<b>2</b>
<b>2.1</b>	<b><u>Hardware architecture</u></b>	<b>2</b>
<b>2.2</b>	<b><u>Software Architecture</u></b>	<b>3</b>
<b>3</b>	<b><u>WHAT GALILAXIS (AND THE EMBEDDED MICROCODE) CAN DO</u></b>	<b>3</b>
<b>4</b>	<b><u>KNOWN LIMITATIONS</u></b>	<b>4</b>
<b>4.1</b>	<b><u>Command limitations</u></b>	<b>4</b>
<b>4.2</b>	<b><u>Supported axis types</u></b>	<b>4</b>
<b>4.3</b>	<b><u>Positioning capabilities</u></b>	<b>4</b>
<b>5</b>	<b><u>TANGO SOFTWARE</u></b>	<b>5</b>
<b>5.1</b>	<b><u>Properties</u></b>	<b>5</b>
<b>5.2</b>	<b><u>Attributes</u></b>	<b>8</b>
<b>5.3</b>	<b><u>Commands</u></b>	<b>11</b>
<b>6</b>	<b><u>CONTROLBOX SETUP</u></b>	<b>14</b>
<b>6.1</b>	<b><u>Network configuration réseau</u></b>	<b>14</b>
<b>6.2</b>	<b><u>First setup of TANGO DeviceServer</u></b>	<b>14</b>
<b>7</b>	<b><u>GLOSSARY</u></b>	<b>16</b>

## 1 Application goals

The SOLEIL motion system, herein described, has been designed to allow the generic control of motorized axes.

The axes are intended to be controlled for *positioning* purposes.

*Speed control* is possible but without guaranteed precision. ( not tested, we have to build a validation platform to specify the accuracy)

Types of axes currently supported are : - :

- Stepper motor with optional encoder
- Brush or brushless motor ( « servo ») with at least 1 encoder ( possibility of 1 encoder on the motor and 1 encoder on the load )
- Piezo actuator with encoder or analog feedback ( for analog feedback : 12 bits resolution standard, 16 bit resolution on special hardware )

Limit switches must be present for finite movements with mechanical stops.

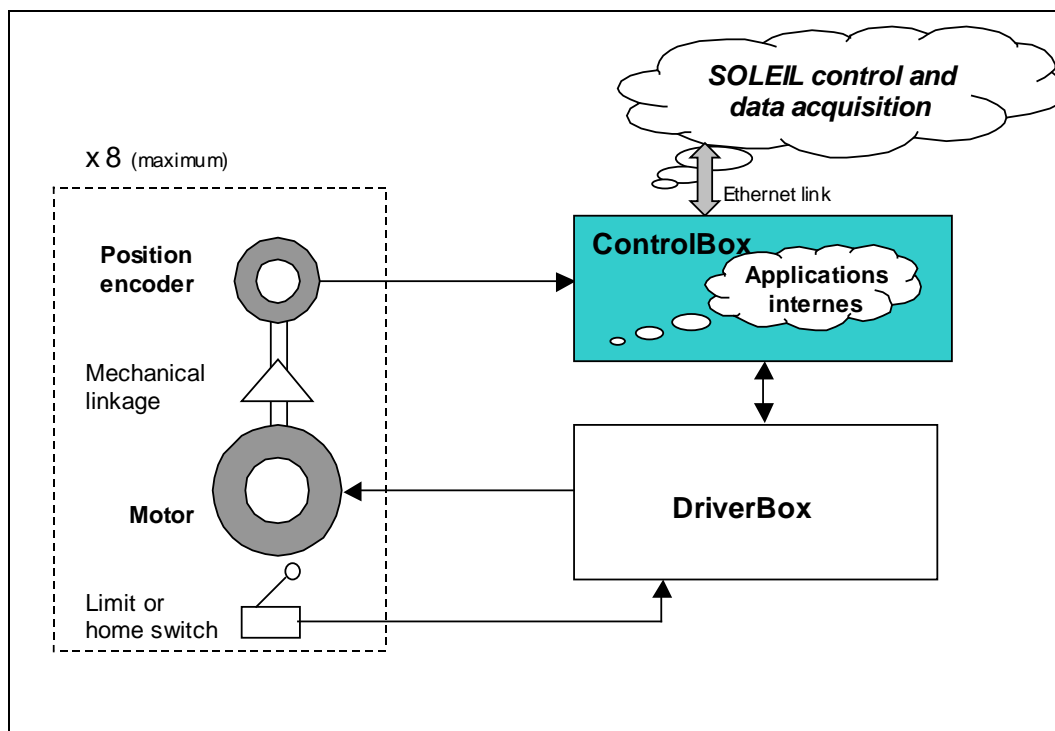
Supported encoders :

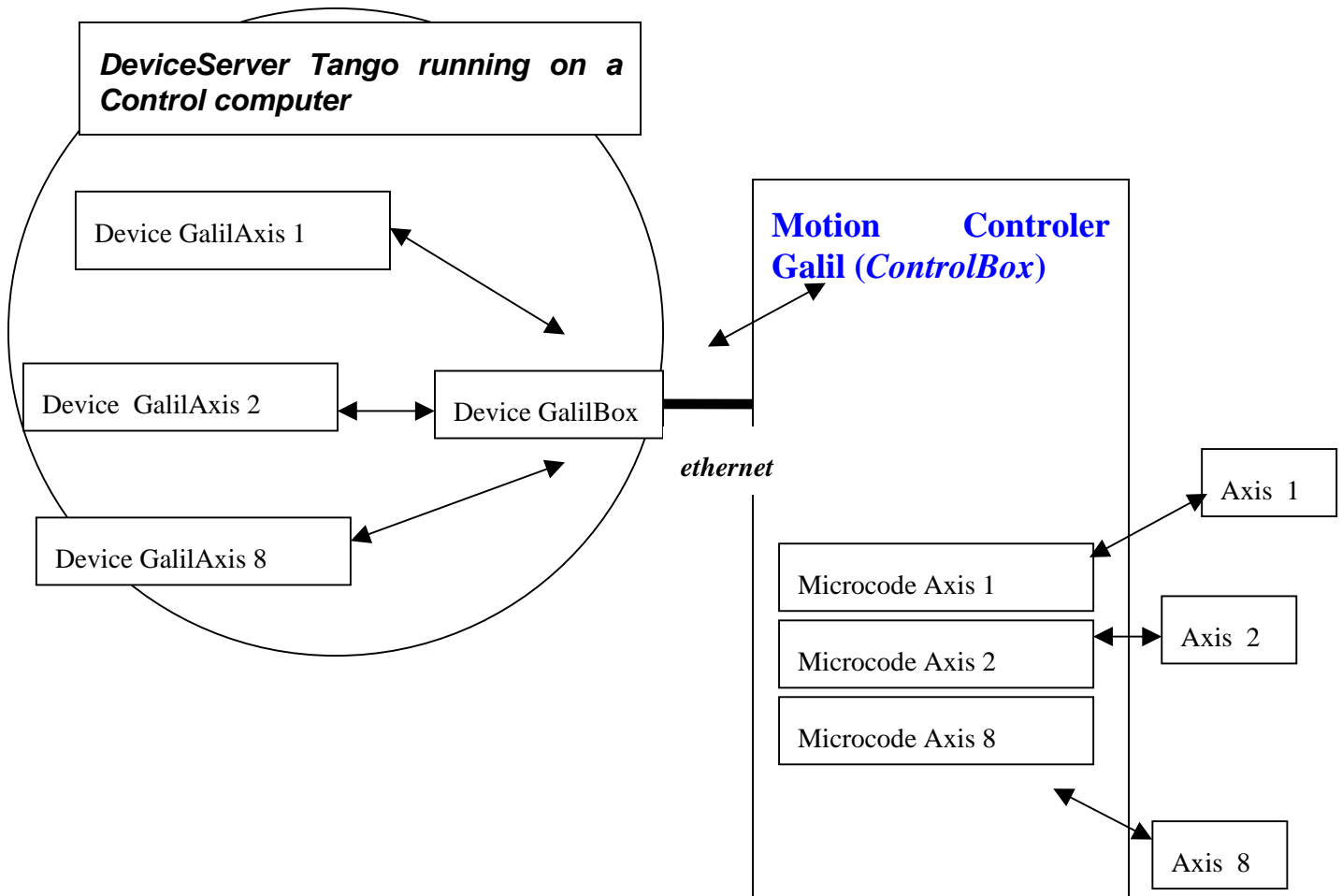
- incremental encoders (TTL or differential, RS422) with optional reference position ( Top 0 or index)
- absolute encoder norme SSI ( binary or gray code)
- sine/cosine encoder with TTL/RS422 conversion.

A HOME switch can be present ( with some limitations described hereafter)

## 2 Technical scheme of the application

### 2.1 Hardware architecture





- The « micro code » embedded in the ControlBox handles (independently of the Tango Control system) the axis positioning, jog, securities ( hardware and software) the initialization of the axis,...
- GalilBox ensures the serialization of commands and readings exchanged on the Ethernet link to guarantee the integrity of data and command .
- GalilAxis offers a control interface for the axis, independent from the Galil Motion Controller syntax.

### 3 What GalilAxis (and the embedded microcode) can do

GalilAxis has been coded for :

- Accurate positioning by closed loop control
- homing (go to a mechanical reference point such as a precision limit switch, a home switch, on the top 0 encoder)

Moreover, it offers:

- Manual movements (jog mode)

- latching of the encoder values on a TTL input of the ControlBox (without interrupting counting)

## 4 Known limitations

### 4.1 Command limitations

- **AxisOscillate**, oscillation between 2 positions, is not yet not operational, (to be implemented in future version)
- **GoToRelativePosition** is not accurate, it adds positioning errors. This problem is not resolved for the moment. (Improvements in the future version)
- Homing is partial, the ControlBox requiring a special mechanical design for Home switch : The home signal must be present on half of the total travel. **AxisFindEdge** Command and **AxisFindHome** command works only if this special design is present
- **AxisFindIndex** works only with an incremental encoder with optional Top 0.

### 4.2 Supported axis types

#### 4.2.1 Axes tested

- Stepper motor without encoder
- Stepper motor with incremental encoder ( or sin/cosine interpolation + TTL converter)
- Stepper motor with absolute encoder SSI
- Servo Motor( brush or brushless ) with incremental Encoder (Sine/cosine interpolation + TTL converter )

#### 4.2.2 Axes not tested but should be supported

- Motor Servo ( brush or brushless ) with absolute SSI Encoder
- Motor Piezo- with analog or encoder feedback

### 4.3 Positioning capabilities

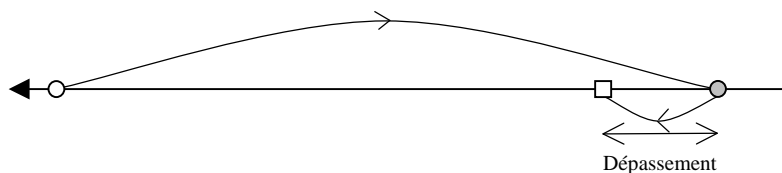
Software based, "microcode" embedded in the motion controller.

#### 4.3.1 Without encoder : for stepper motors only

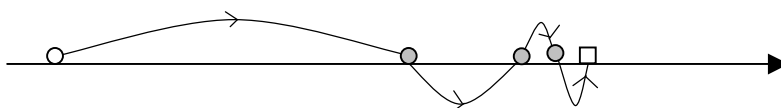
- Simple positioning
- Simple positioning with backlash compensation

#### 4.3.2 With encoders : all motor types

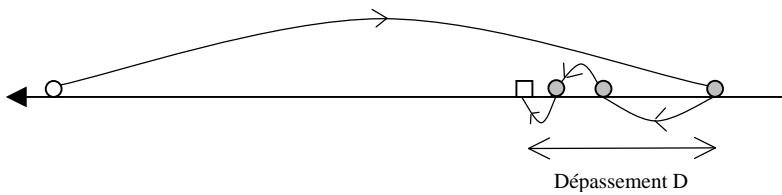
- Simple positioning
- Simple positioning with backlash compensation ( n retries )



- Positioning with successive closer moves ( n retries )



- Simple positioning with mechanical backlash compensation and successive closer moves ( n retries )



## 5 Tango Software

### 5.1 Properties

The properties contain the fixed parameters of the axis. They are filled by the user at install time, and are not supposed to be modified during runtime.

AxisNumber	<p>Number of the axis wired on the controlbox</p> <p>Can be the number or the letter ( Galil MC convention )</p> <p>Relation :</p> <p>1    A    X</p> <p>2    B    Y</p> <p>3    C    Z</p> <p>4    D    W</p> <p>5    E</p> <p>6    F</p> <p>7    G</p> <p>8    H</p> <p>Default : 1</p>
AxisBoxAttachement	<p>Tango name of the ControlBox on which the axis is wired</p> <p>Default : " "</p>
EncoderMotorRatio	<p>Number of encoder steps per motor step.</p> <p>Useful for stepper only.</p> <p>Incremental encoders are interpolated by 4</p> <p>For steppers without encoder and servo : set to 1</p> <p>Example : encoder 5000 steps/rev, stepper 200 steps/rev, driver tuned for 10 micro-steps by full step :</p> <p>encoder = 5000 x 4 = 20 000</p> <p>stepper = 200 x 10 = 2 000</p> <p>EncoderMotorRatio = encoder/micro-steps = 10</p> <p>Default : 1</p>

UserEncoderRatio	<p>User units divided by encoder step size.</p> <p>Example :</p> <p>User unit = 1 micrometer, 1 encoder = 100 nanometer, UserEncoderRatio = 10.0</p> <p>The formula used for calculation is :</p> $\frac{\text{AxisCurrentPosition}_{\text{unités\_utilisateurs}}}{(\text{AxisCurrentEncoderPosition}_{\text{unités\_codeur}} \times \text{UserEncoderRatio}) - \text{AxisOffset}} = *$ <p>Default : 1.0</p>
CurrentXXXXX	<p>All properties beginning by “Current” are used to store in the static database the last related attribute value. Not supposed to be modified.</p> <p>Example : <b>CurrentAcceleration</b> stores the last <b>AxisAcceleration</b>.</p> <p>No default.</p>
AxisEncoderType	<p>Encoder type.</p> <p>0 : no encoder.</p> <p>1 : incremental encoder</p> <p>2 : absolute encoder</p> <p>Default = 1.</p>
AxisEncoderDirection	<p>Useful only for incremental encoder</p> <p>Fixes the encoder type and the counting direction.</p> <p>For steppers only 1 encoder available : Main, Aux is used by the step counter.</p> <p>For servo : 2 encoders, Main and Aux main on the motor, aux on the load</p> <p>See Galil Reference manual, CE command</p> <p>Main encoder</p> <p>Aux encoder</p> <p>0 quadrature, normal direction</p> <p>0 quadrature, normal direction</p> <p>1 pulse and direction, normal direction</p> <p>4 pulse and direction, normal direction</p> <p>2 quadrature, reverse direction</p> <p>8 quadrature, reverse direction</p> <p>3 pulse and direction, reverse direction</p> <p>12 pulse and direction, reverse direction</p>

	Default : 0
AbsoluteEncoderDirection	<p>For absolute encoder.</p> <p>Defines the encoder counting direction</p> <p>1 : counts like the encoder</p> <p>-1 : counts reverse the encoder direction</p> <p>Default : 1</p>
AbsoluteEncoderOffset	<p>For absolute encoder</p> <p>Offset to subtract from the natural value of the encoder</p> <p>Pay attention to the overflow! You may turn the encoder shaft to find a value close to the real position of the axis and finely tune using this property.</p>
AxisMotorDirection	<p>Defines the motor type and direction</p> <p>1.0 : servo motor</p> <p>-1.0 : motor servo, reversed polarity</p> <p>2.0 : stepper, pulse and direction signals active low</p> <p>-2.0 stepper, pulse and direction signals active high</p> <p>2.5 : stepper, pulse and direction signals active low , normal direction</p> <p>-2.5 : stepper, pulse and direction signals active low , reverse direction</p> <p>Default : 2.0</p>
AxisForwardLimit	<p>software Limit in encoder unit (x 4 interpolation) of positive move.</p> <p>No move for a request higher than this limit, stop on that limit reached</p> <p>Low : - 2 147 483 648 high : 2 147 483 647</p> <p>limit active at n-1.</p> <p>Limit disabled at n = 2 147 483 647</p> <p>Default : 1 000 000</p> <p>ISN'T IT DANGEROUS TO SUPPLY AN ARBITRARY HIGH DEFAULT HERE? IT ASSUMES THAT HARDWARE LIMITS ARE PROPERLY CONFIGURED. 0 SEEMS TO ME A SAFER DEFAULT!</p>
AxisBackwardLimit	<p>software Limit in encoder unit (x 4 interpolation) of negative move.</p> <p>No move for a request lower the this limit, stop on that limit reached</p> <p>Range - 2 147 483 648 to 2 147 483 647</p> <p>limit active at n-1</p> <p>Limit disabled at n = -2 147 483 648</p> <p>Default : -1 000 000</p> <p>SAME COMMENT.</p>

AxisInitType	<p>Initialization type of the axis</p> <p>0 : no init possible : use AxisDefinePosition command.</p> <p>1 : Initialisation on Home limit switch. Need a long Home switch on half of the total travel.</p> <p>Useful for servo, less on steppers</p> <p>2 : Initialisation on the Limit Switch Backward (standard motorist direction, doesn't use AxisUserDirection)</p> <p>3 : Initialisation on Top 0 encoder. Needs an incremental encoder with Top 0.</p> <p>Default : 0</p>
AxisInitPosition	<p>Initial position in user units</p> <p>Loaded on AxisInit success.</p> <p>Default : 0</p>
MotoristMemo	<p>Use it to store important axis parameters such as driver location,.... Up to 255 characters.</p>
ServoPIDProp	<p>Servo Motors ( brush or Brushless ) only.</p> <p>Proportional coefficient of PID</p> <p>See KP reference</p>
ServoPIDInteg	<p>Servo Motors ( brush or Brushless ) only.</p> <p>Integral coefficient of PID</p> <p>See KI reference</p>
ServoPIDDeriv	<p>Servo Motors ( brush or Brushless ) only.</p> <p>derivative coefficient of PID</p> <p>See KD reference</p>
ServoPIDIntegLimit	<p>Servo Motors ( brush or Brushless ) only.</p> <p>Integral limit if Integral component of PID</p> <p>See IL reference</p>
ErrorLimit	<p>Limit of dynamic error position on moving.</p> <p>If ER exceeds ErrorLimit, motor stops and motor is disabled ( as MotorOFF command ).</p> <p>Not to be reached on normal operation, only if mechanical jam or encoder or driver fault.</p> <p>Useful for servo axes.</p> <p>-1 : disabled</p> <p>Max : 32767</p> <p>Default : -1</p>

## 5.2 Attributes

AxisCurrentPosition	Read/Write
---------------------	------------



	<p>Read : current position of the axis in user units.</p> <p>Write : starts a positioning on the axis.</p> <p>see AxisGetMotionStatus() and state() to control the positioning</p> <p>The formula used for calculation of Read value is :</p> $\text{AxisCurrentPosition}_{\text{unités\_utilisateurs}} = (\text{AxisCurrentEncoderPosition}_{\text{unités\_codeur}} \times \text{UserEncoderRatio}) - \text{AxisOffset}$
AxisCurrentEncoderPosition	<p>Read Only</p> <p>Read : current position of the axis in encoder units if any (or 0).</p>
AxisCurrentMotorPosition	<p>Read Only</p> <p>For stepper motors or servo + aux encoder</p> <p>Read : Current position of the motor in steps .</p>
AxisAcceleration	<p>Read/Write</p> <p>Read : Current acceleration of the motor.</p> <p>Write : new acceleration of the motor</p> <p>In user units/sec.<sup>2</sup></p>
AxisDeceleration	<p>Read/Write</p> <p>Read : Current deceleration of the motor.</p> <p>Write : new deceleration of the motor</p> <p>In user units/sec.<sup>2</sup></p>
AxisVelocity	<p>Read/Write</p> <p>Read : Current speed of the axis</p> <p>Write : new speed</p> <p>User units/sec.</p> <p><b>IS THERE A FLAG FOR WHEN THE AXIS HAS REACHED CONSTANT SPEED?</b></p>
AxisMotionAccuracy	<p>Read/Write</p> <p>Motion accuracy, the position is considered as correct if position reached &lt;= desired position +/- AxisMotionAccuracy</p> <p>Read current accuracy.</p> <p>Write : new accuracy</p> <p>In user units.</p> <p>Lower limit : 1 motor step or 1 encoder step</p>
AxisPercent	<p>Read/Write</p> <p>Coefficient for successive approach positioning.</p> <p>Read : current coefficient.</p> <p>Write : new coefficient</p>

	<p>In user units.</p> <p>Minimum : 0.0 ( no move )</p> <p>Maximum : 1.0 ( go direct to final position )</p>
AxisRetry	<p>Read/Write</p> <p>Number of successive retries when positioning errors increasing?</p> <p>Read : current number of retries ( decreased after each try ).</p> <p>Write : new retry number</p>
AxisBacklash	<p>Read/Write</p> <p>Compensation for mechanical backlash by doing final move in the direction defined by AxisDirection.</p> <p>Read : current backlash distance.</p> <p>Write : new backlash distance</p>
AxisDirection	<p>Read/Write</p> <p>Direction of final move ( for mechanical backlash compensation ).</p> <p>Write : modification of direction</p> <p>0 : no mechanical backlash compensation</p> <p>1 : final move direction = direction as defined by Userdirection</p> <p>1 : final move direction = reverse direction as defined by Userdirection</p>
AxisOffset	<p>Read/Write</p> <p>Offset subtracted from current position (AxisCurrentPosition).</p> <p>Doesn't change encoder or motor position.</p> <p>Read : current offset.</p> <p>Write : new offset</p> <p>In User units</p>
UserDirection	<p>Read/Write</p> <p>Positive direction defined by the user. Has an effect on move direction, AxisUserPosition. encoder and motor remain unchanged.</p> <p>Read : Current user direction of the axis.</p> <p>Write : new user direction</p> <p>1 : = motorist direction</p> <p>-1 : user direction reverse of motorist direction.</p>
AxisCurrentVelocity	<p>Read</p> <p>Instantaneous speed of axis measured by the axis Controller <i><b>only if an encoder is present</b></i></p>

	Without encoder value is always 0 In user units/sec.
IsInitialised	Read 0 : axis was not initialized at a known position. 1 : axis initialized. IsInitialised set to 0 when the ControlBox has been powered down or has been reset
LatchMode	Read/Write Capture of position mode. The encoder position is latched upon transition of a TTL input from the ControlBox. Read : 0 : latch not armed 1 : latch armed Write : 1 : arms latch
PositionLatched	Read Position value latched. Reading this attribute rearms the latch
LatchOccured	Read 1 : latch occurred 0 : latch armed, waiting.

### 5.3 Commands

AxisForward	Requests a continuous movement positive direction according to UserDirection. Stops on AxisStop or limit switch forward ( hardware or software ). Allowed if axis is in standby state.
AxisBackward	Requests a continuous movement negative direction according to UserDirection. Stops on AxisStop or limit switch backward( hardware or software ). Allowed if axis is in standby state.
AxisGoToPosition	Requests a positioning at absolute position “argin” in user units. Allowed if axis is in standby state.
AxisGoToRelativePosition	Requests a positioning at relative position “argin” in user units ( actual position +/- “argin” ). Allowed if axis is in standby state. Not very accurate, see Limits.
AxisStop	Stops the current movement.
AxisOscillate	Not actually useful. See Limits.

AxisGetMotionStatus	<p>Returns the simplified status of movement.</p> <p>-1 : positioning ended with error.</p> <p>0 : positioning ended OK.</p> <p>1 : positioning running.</p>
AxisGetErrorStatus	<p>Returns the error code.</p> <p>-1 : positioning ended with error.</p> <p>0 : positioning ended OK.</p> <p>1 : positioning running.</p> <p>2 ControlBox microcode not running</p> <p>3 limit switch forward ( according to UserDirection )</p> <p>4 limit switch backward ( according to UserDirection )</p> <p>5 Execute Microcode command and microcode already running</p> <p>6 Begin movement not possible due to Limit Switch</p> <p>7 Axis configured both as master and slave</p> <p>8 Jumpers need to be installed for stepper operation</p> <p>9 Axis not initialized</p> <p>10 Command error</p> <p>11 Invalid operand</p> <p>12 number out of range</p> <p>13 variable error</p> <p>14 subroutine more than 16 deep</p> <p>15 EEPROM checksum error</p> <p>20 begin not valid when moving</p> <p>21 decelerating or stopped by limit switch + direction</p> <p>22 decelerating or stopped by limit switch - direction</p> <p>23 stopped by ABORT input</p> <p>24 stopped by ABORT command</p> <p>25 decelerated or stopped on ON/OFF ERROR</p> <p>26 stopped after Home switch edge detected</p> <p>27 Stopped at the end of Home routine</p> <p>28 Stopped on selective abort input</p> <p>120 Bad Ethernet packet transmit</p> <p>121 Bad Ethernet packet received</p> <p>122 Ethernet buffer overflow</p> <p>123 lost Sync TCP</p>

	124 TCP handle already in use 125 no ARP response from IP Address 126 Lost TCP/IP communication
State	Returns the axis state OFF : no communication STANDBY : waiting for command MOVING : moving ALARM : axis in alarm, moving allowed FAULT : fault ( 2, 7, 14, 15 errors ) DISABLE : non handled errors, must not occur in normal operation..
Status	Error in plain text, similar to AxisGetErrorStatus
AxisInit	Initializes the axis according to property AxisInitType Executed after ControlBox power on, or reset. Executes according to AxisInitType : AxisFindHome AxisFindIndex Limit switch backward finding ( in motorist direction ). Then loads the value of AxisInitPosition property.
AxisFindEdge	Finds the edge of Home switch. Home switch must be special design type.
AxisFindIndex	Find top 0 of the incremental encoder if any.
AxisFindHome	Finds the edge of Home switch then stops and goes backward very slowly and stops on edge detection.. Home switch must be special design type.
AxisStartMicroCode	Expert only. Starts microcode execution NOINIT : parameters not resettled INIT : parameters are resettled at their default values
DirectCmd	Experts in ControlBox only; direct command to ControlBox. Only for use to other Dservers ( ex. GalilSlits ) needing to access a special function on the ControlBox.
MotorON	Enables the power amplifier.
MotorOFF	Disables the power amplifier. Only if the motor is stopped ( issue a AxisStop Command before MotorOFF ).
AxisGetHwConf	Display axis parameters (axis letter, encoder type, init type, etc)Useful for setup of an axis

### 6.1 Network configuration réseau

Each Controlbox must have a TCP/IP address.

At boot, the ControlBox tries to get a valid address from a DHCP server . If no one is available, the Galil tool (DMCSetup) must be used.

The PC running the DMCSetup must be connected to the ControlBox through a serial connection . Then

- Put the ENET switch in OFF position
- Open the serial line communication in DMCSetup
- Type the **IA** ( Internet Address ) **A,B,C,D** command. A,B,C,D are the IP number of the ControlBox
- Type the **BN** command( Burn Parameter)
- **RS** (ControlBox Reset)
- **IA ?** to verify address
- Put the ENET switch in ON position

### 6.2 First setup of TANGO DeviceServer

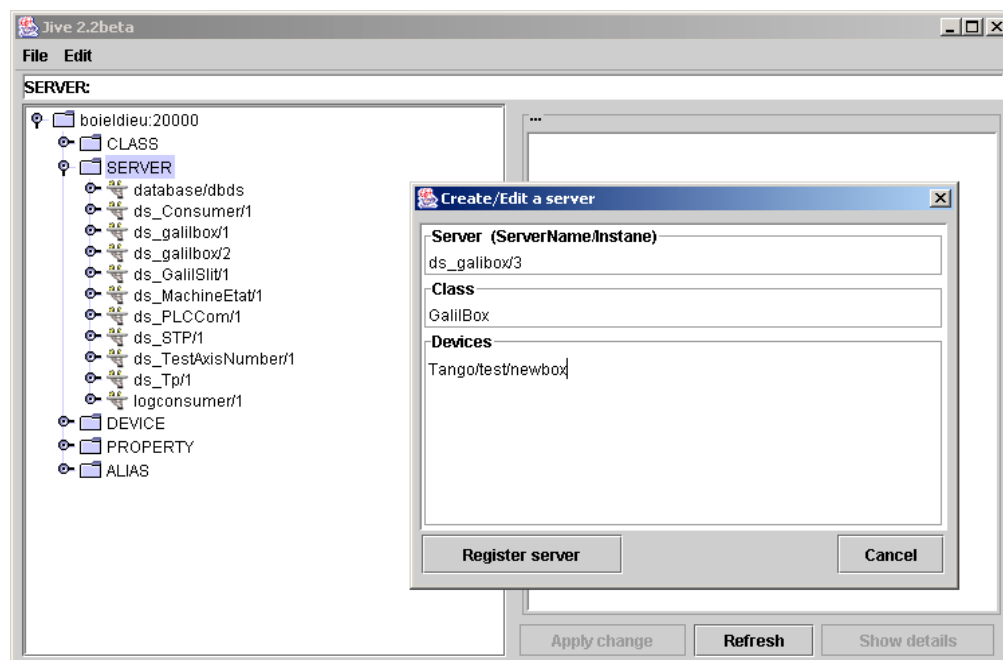
#### 6.2.1 Creation of the GalilBox Device

The ControlBox is supposed to be hardware ready (stepper jumpers installed, IP address assigned ), connected to the network, powered on.

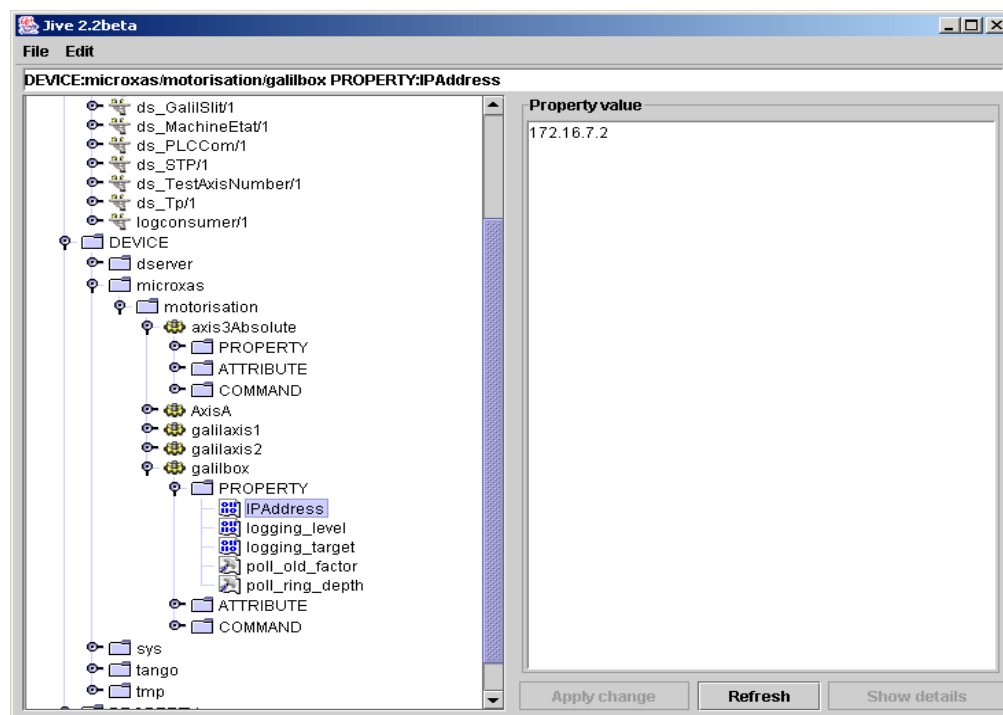
The Tango static DataBase is running.

Using Jive :

- Create a Dserver GalilBox and register it



- Add property IPAddress and assign the IP Address of the new ControlBox



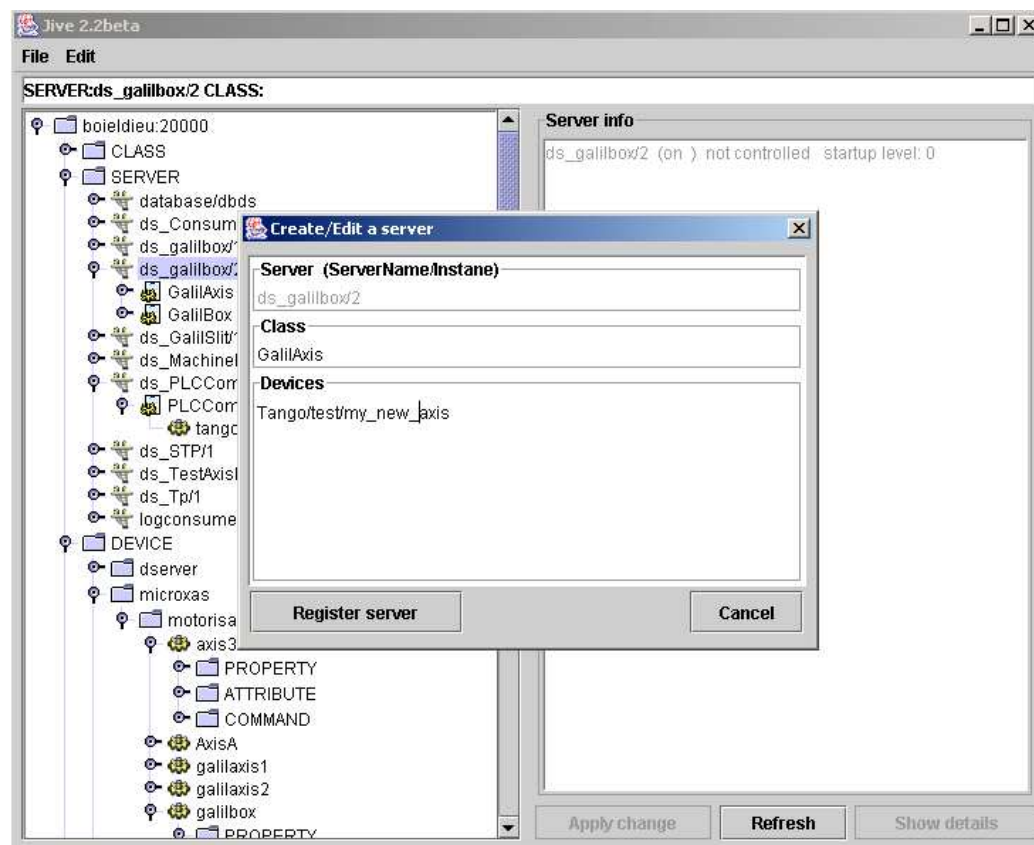
- Start the device server ( dos box or script )
- Test the device ( right click on DEVICE/Tango/test/newbox ) select ping command, the device responds ALIVE )

You have now the communication Device server installed and running.

## 6.2.2 Creation of a new axis : GalilAxis

You have a ControlBox and the related Dserver GalilBox running and want to add a new axis

- Make sure your axis hardware is correctly parameterized ( stepper jumper ).
- Select DSERVER/GalilBox/3, right click, “add class to server”
- Fill the form



- Create (or copy) the properties (from another axis)

The minimum set of properties to be filled in order to have a working axis are :

- AxisBoxAttachement
- AxisBoxAttachement.
- AxisEncoderType
- AxisNumber
- EncoderMotorRatio

Restart the Dserver GalilBox, the server GalilAxis should rise and respond.

## 7 Glossary

**ControlBox** : motion controller from Galil, 8 axis, ref. 2180, 2280, 2182 with SSI option

**DriverBox** : 19'' rack containing the power amplifiers and some electronics.

**Micro-pas** : some power amplifiers offer the possibility to do micro-stepping.

**Home switch with special feature** : Home signal is present on half of the total travel.

Permits the ControlBox automatic Home find. See Galil Doc.