FUNCTION REFERENCE

DLL for motion controller

Optics Focus Instruments Co., Ltd.

2021-2-9

Table of Contents

Properties

1. **Port**

Define: short Port

Description: Read only. The port number connected successfully. It can be changed by function ConnectPort.

C# code:

```
short sPort;
sPort = scu1.Port;
```

2. **ConnectStatus**

Define: bool ConnectStatus

Description: Read only. The connect status between the motion controller and computer. It can be changed by function ConnectPort and ClosePort.

C# code:

```
bool bConnect;
bConnect = scu1.ConnectStatus;
```

3. **BusyStatus**

Define: bool BusyStatus

Description: Read only. The busy status of the motion controller.

C# code:

```
bool bBusy;
bBusy = scu1.BusyStatus;
```

4. **Language**

Define: short Language

Description: Read/Write. Get or set the current language.

C# code:

```
short sLanguage;
sLanguage = scu1.Language;
```

5. **SpeedGrades**

Define: short SpeedGrades

Description: Read/Write. Get or set the current speed grade.

C# code:

```
short sSpeed;
sSpeed = scu1.SpeedGrades;
```

6. **ParameterFlag**

   Define: bool ParameterFlag

   Description: Read only. Whether the parameter has set or not.

   C# code:

   ```
   bool bParameterFlag;
   bParameterFlag = scu1.ParameterFlag;
   ```

7. **CurrentAxis**

   Define: short CurrentAxis

   Description: Read only. Get the index of the current axis.

   C# code:

   ```
   Short sCurrentAxis;
   sCurrentAxis = scu1.CurrentAxis;
   ```

Functions

1. **ConnectPort**

   Define: void ConnectPort(short sPort)

   Description: Connects the motion controller through the assigned port number.

   Parameters:

   sPort**:** the port number

   Return:

   Null

   C# code:

   ```
   short sPort=3;
   scu1.ConnectPort(sPort);      // connects the motion controller through COM3 port.
   ```

2. **ClosePort**

   Define: void ClosePort ()

Description: Closes the connection to the motion controller.

Parameters:

Null

Return:

Null

C# code:

scu1.ClosePort();	// closes the connection.

3. **GetCurrentStep**

Define: long GetCurrentStep(short sIndex)

Description: Gets the current step of the assigned axis.

Parameters:

sIndex**:** the index of axis

Return:

The current step

C# code:

```
long lCurrentStep;
lCurrentStep = scu1.GetCurrentStep(0);	// gets the current step of X axis
```

4. **GetCurrentPosition**

Define: double GetCurrentPosition(short sIndex)

Description: Gets the current position of the assigned axis (unit is mm or degree)

Parameters:

sIndex**:** the index of axis

Return:

The current position

C# code:

```
double dCurrentPosition;
dCurrentPosition = scu1.GetCurrentPosition(0);	// gets the current position of X axis
```

5. **GetActualSpeed**

Define: double GetActualSpeed(short sIndex)

Description: Gets the actual speed of the assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The actual speed

C# code:

```
double dActualSpeed;
dActualSpeed = scu1.GetActualSpeed(0);       // gets the actual speed of X axis
```

6. **GetPulseEquivalent**

Define: double GetPulseEquivalent(short sIndex)

Description: Gets the pulse equivalent of the assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The pulse equivalent

C# code:

```
double dPulseEquivalent;
dPulseEquivalent = scu1.GetPulseEquivalent(0);       // gets the pulse equivalent
of X axis
```

7. **GetType**

Define: short GetType(short sIndex)

Description: Gets the type index of assigned axis (0:Null, 1:linear stage, 2:rotation stage, 3:goniometer, 4:lab jack)

Parameters:

sIndex**:** the index of axis

Return:

The type index

C# code:

```
short sType;
sType = scu1.GetType(0);       // gets the type index of X axis
```

8. **GetUnit**

Define: short GetUnit(short sIndex)

Description: Gets the unit index of assigned axis (0:mm, 1:degree, 2:step)

Parameters:

sIndex**:** the index of axis

Return:

The unit index

C# code:

```
short sUnit;
sUnit = scu1.GetUnit(0);        // gets the unit index of X axis
```

9. **GetMotorAngle**

Define: float GetMotorAngle(short sIndex)

Description: Gets the stepper angle of the motor of assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The stepper angle of motor of axis

C# code:

```
float fMotorAngle;
fMotorAngle = scu1.GetMotorAngle(0);        // gets the stepper angle of X axis
```

10. **GetSubsection**

Define: short GetSubsection(short sIndex)

Description: Gets the subdivision of assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The subdivision of axis

C# code:

```
short sSubsection;
sSubsection = scu1.GetSubsection(0);        // gets the subdivision of X axis
```

11. **GetPitch**

Define: float GetPitch(short sIndex)

Description: Gets the pitch of lead screw of assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The pitch of lead screw of axis

C# code:

```
float fPitch;
fPitch = scu1.GetPitch(0);        // gets the pitch of X axis
```

12. GetTranRatio

Define: int GetTranRatio(short sIndex)

Description: Gets the transmission ratio of assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The transmission ratio of axis

C# code:

```
int iTranRatio;
iTranRatio = scu1.GetTranRatio(0);      // gets the transmission ratio of X axis
```

13. GetTravel

Define: double GetTravel(short sIndex)

Description: Gets the travel range of assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The travel range

C# code:

```
double dTravel;
dTravel = scu1.GetTravel(0);       // gets the travel range of X axis
```

14. GetNegativeTravel

Define: double GetNegativeTravel(short sIndex)

Description: Gets the negative travel range of the assigned axis (for goniometer)

Parameters:

sIndex**:** the index of axis

Return:

The negative travel range

C# code:

```
double dNegativeTravel;
dNegativeTravel = scu1.GetNegativeTravel(0);       // gets the negative travel range of X axis
```

15. GetPositiveTravel

Define: double GetPositiveTravel(short sIndex)

Description: Gets the positive travel range of the assigned axis (for goniometer)

Parameters:

sIndex**:** the index of axis

Return:

The positive travel range

C# code:

```
double dPositiveTravel;
dPositiveTravel = scu1.GetPositiveTravel(0);       // gets the positive travel range of X axis
```

16. GetZeroOffset

Define: long GetZeroOffset(short sIndex)

Description: Gets the offset of zero position of assigned axis

Parameters:

sIndex**:** the index of axis

Return:

The offset of zero position

C# code:

```
long lZeroOffset;
lZeroOffset = scu1.GetZeroOffset(0);       // gets the offset of zero position of X axis
```

17. SetType

Define: short SetType(short sIndex, short sType)

Description: Sets the stage type of assigned axis

Parameters:

sIndex**:** the index of axis

sType**:** the type index of stage (0:Null, 1:linear stage, 2:rotation stage, 3:goniometer, 4:lab jack)

Return:

-1 // error

1 // success

C# code:

```
short sType;
sType = scu1.SetType(0,1);       // sets the stage type of X axis to linear stage
```

18. SetUnit

Define: short SetUnit(short sIndex, short sUnit)

Description: Sets the unit of assigned axis

Parameters:

sIndex**:** the index of axis

sUnit**:** the unit index (0:mm, 1:degree, 2:step)

Return:

-1 // error

1 // success

C# code:

```
short sUnit;
sUnit = scu1.SetUnit(0,0);       // sets the unit of X axis to mm
```

19. SetMotorAngle

Define: short SetMotorAngle(short sIndex, float fMotorAngle)

Description: Sets the stepper angle of the motor of assigned axis

Parameters:

sIndex**:** the index of axis

fMotorAngle**:** the stepper angle (0.9 or 1.8)

Return:

-1 // error

1 // success

C# code:

```
short sMotorAngle;
sMotorAngle = scu1.SetMotorAngle(0,0.9);       // sets the stepper angle of X axis
to 0.9
```

20. SetSubsection

Define: short SetSubsection(short sIndex,short sSubsection)

Description: Sets the subdivision of assigned axis

Parameters:

sIndex**:** the index of axis

sSubsection**:** the subdivision (1, 2, 4 or 8)

Return:

-1 // error

1 // success

C# code:

```
short sSubsection;
sSubsection = scu1.SetSubsection(0,2);     // sets the subdivision of X axis to 2
```

21. SetPitch

Define: Short SetPitch(short sIndex, float fPitch)

Description: Sets the pitch of lead screw of assigned axis

Parameters:

sIndex**:** the index of axis

fPitch**:** the pitch of lead screw (>0)

Return:

-1 // error

1 // success

C# code:

```
short sPitch;
sPitch = scu1.SetPitch(0,1);           // sets the pitch of X axis to 1
```

22. SetTranRatio

Define: short SetTranRatio(short sIndex, int iTranRatio)

Description: Sets the transmission ratio of assigned axis

Parameters:

sIndex**:** the index of axis

iTranRatio: the transmission ratio (>0)

Return:

-1 // error

1 // success

C# code:

```
int sTranRatio;
sTranRatio = scu1.SetTranRatio(0,1);        // sets the transmission ratio of X axis to 1
```

23. SetTravel

Define: short SetTravel(short sIndex, double dTravel)

Description: Sets the travel range of assigned axis

Parameters:

sIndex**:** the index of axis

dTravel**:** the travel range

Return:

-1 // error

1 // success

C# code:

```
short sTravel;
sTravel = scu1.SetTravel(0, 10);       // sets the travel range of X axis to 10mm or degree
```

24. SetNegativeTravel

Define: short SetNegativeTravel(short sIndex, long sNegativeTravel)

Description: Sets the negative travel range of assigned axis (for goniometer)

Parameters:

sIndex**:** the index of axis

sNegativeTravel**:** the negative travel range

Return:

-1 // error

1 // success

C# code:

```
short sNegativeTravel;
sNegativeTravel = scu1.SetNegativeTravel(0, 10);       // sets the negative travel range of X axis to 10 degree
```

25. SetPositiveTravel

Define: short SetPositiveTravel(short sIndex, long sPositiveTravel)

Description: Sets the positive travel range of assigned axis (for goniometer)

Parameters:

sIndex**:** the index of axis

sPositiveTravel**:** the positive travel range

Return:

-1 // error

1 // success

C# code:

```
short sPositiveTravel;
sPositiveTravel = scu1.SetPositiveTravel(0, 10);      // sets the positive travel range of X axis to 10 degree
```

26. SetZeroOffset

Define: short SetZeroOffset(short sIndex, long lZeroOffset)

Description: Sets the offset step of zero position of assigned axis

Parameters:

sIndex**:** the index of axis

lZeroOffset**:** the offset step

Return:

-1 // error

1 // success

C# code:

```
short sZeroOffset;
sZeroOffset = scu1.SetZeroOffset(0, 1000);      // sets the offset of X axis to 1000 steps
```

27. RefreshCurrentPosition

Define: short RefreshCurrentPosition()

Description: Refresh the current position and display

Parameters:

Null

Return:

-1 // error

1 // success

C# code:

```
short sRefreshCurrentPosition;
sRefreshCurrentPosition = scu1.RefreshCurrentPosition();        // refresh the current position
```

28. RunToZero

Define: short RunToZero(short sIndex, short sMode)

Description: Returns to the zero position according to the return mode

Parameters:

sIndex**:** the index of axis

sMode**:** the return mode (0: moves to the offset of zero position after returning to the mechanical zero, 1: moves to the current position after returning to the mechanical zero)

Return:

-1 // error

1 // success

C# code:

```
short sRunToZero;
sRunToZero = scu1.RunToZero(0,0);       // returns X axis to zero position by mode 0
```

29. RunToStep

Define: short RunToStep(short sIndex, long lStep)

Description: Moves the assigned axis to the assigned steps

Parameters:

sIndex**:** the index of axis

lStep**:** the steps to move

Return:

-1 // error

1 // success

C# code:

```
short sRunToStep;
sRunToStep = scu1.RunToStep(0,10000);       // moves X axis to 10000 steps
```

30. RunToPosition

Define: short RunToPosition(short sIndex, double dPosition)

Description: Moves the assigned axis to the assigned position

Parameters:

sIndex**:** the index of axis

dPosition**:** the position to move

Return:

-1 // error

1 // success

C# code:

```
short sRunToPosition;
sRunToPosition = scu1.RunToPosition(0,10);       // moves X axis to 10mm or
degree
```

31. StopRun

Define: short StopRun()

Description: Stops moving

Parameters:

Null

Return:

-1 // error

1 // success

C# code:

```
short sStopRun;
sStopRun   = scu1.StopRun();       // stops moving
```

32. StopDelay

Define: void StopDelay()

Description: Stops delay

Parameters:

Null

Return:

Null

C# code:

```
scu1.StopDelay();       // stops delay
```

33. SaveParam

Define: short SaveParam(short sIndex)

Description: Refreshes and saves the parameters of the assigned axis

Parameters:

sIndex**:** the index of axis

Return:

-1 // error

1 // success

C# code:

```
short sSaveParam;
sSaveParam = scu1.SaveParam(0);      // saves the parameters of X axis
```

34. DisplayParameterInterface

Define: void DisplayParameterInterface()

Description: Displays the interface of parameters settings.

Parameters:

Null

Return:

Null

C# code:

```
scu1.DisplayParameterInterface();      // displays the interface of parameters
settings
```

35. DisplayOperationInterface

Define: void DisplayOperationInterface()

Description: Displays the interface of operation.

Parameters:

Null

Return:

Null

C# code:

```
scu1.DisplayOperationInterface();      // displays the interface of operation
```

Code Examples for Positioning Stages

Please run your own program as administrator. The parameters need permission to be saved into a file. Please refer to the below examples to use the necessary functions to set and save the parameters before executing other operations.

Please get the parameters of specific stage from our website and get the subdivision from the back panel of our motion controller. The default subdivision is 2. Please get the methods of setting subdivision from the user manual of motion controller.

1. **Motorized Linear Stage:**

```
sC3U1.SetType(0, 1);                        //Set parameters for translation stage
sC3U1.SetUnit(0, 0);                                          //Set mm as the unit
sC3U1.SetMotorAngle(0, Convert.ToSingle(0.9)); //Set the motor angle to 0.9
sC3U1.SetSubsection(0, 2);                                 //Set the subdivision to 2
sC3U1.SetPitch(0, 1);                                      //Set the pitch to 1
sC3U1.SetTravel(0, 50);                                    //Set the travel range to 50
sC3U1.SaveParam(0);                                        //Save all parameters
```

2. **Motorized Rotary Stage:**

```
sC3U1.SetType(0, 2);                            //Set parameters for rotation stage
sC3U1.SetUnit(0, 1);                                          //Set degree as the unit
sC3U1.SetMotorAngle(0, Convert.ToSingle(0.9));      //Set the motor angle to 0.9
sC3U1.SetSubsection(0, 2);                                    //Set the subdivision to 2
sC3U1.SetTranRatio(0, 180);                         //Set the transmission ratio to 180
sC3U1.SaveParam(0);                                           //Save all parameters
```

3. **Motorized Lab Jack:**

```
sC3U1.SetType(0, 4);                               //Set parameters for lab jack
sC3U1.SetUnit(0, 0);                                          //Set mm as the unit
sC3U1.SetMotorAngle(0, Convert.ToSingle(0.9));     //Set the motor angle to 0.9
sC3U1.SetSubsection(0, 2);                                    //Set the subdivision to 2
sC3U1.SetPitch(0, 1);                                         //Set the pitch to 1
sC3U1.SetTranRatio(0, 1);                             //Set the transmission ratio to 1
sC3U1.SetTravel(0, 50);                                       //Set the travel range to 50
sC3U1.SaveParam(0);                                           //Save all parameters
```

4. **Motorized Goniometer:**

4.1 Not set new zero position

```
sC3U1.SetType(0, 3);                        //Set parameters for goniometer stage
sC3U1.SetUnit(0, 1);                        //Set degree as the unit
sC3U1.SetMotorAngle(0, Convert.ToSingle(0.9));   //Set the motor angle to 0.9
```

```
sC3U1.SetSubsection(0, 2);                //Set the subdivision to 2
sC3U1.SetTranRatio(0, 90);                //Set the transmission ratio to 90
sC3U1.SetTravel(0, 20);                   //Set the travel range to 20
sC3U1.SaveParam(0);                       //Save all parameters
```

4.2 Set new zero position

```
sC3U1.SetType(0, 3);                      //Set parameters for goniometer stage
sC3U1.SetUnit(0, 1);                        //Set degree as the unit
sC3U1.SetMotorAngle(0, Convert.ToSingle(0.9));   //Set the motor angle to 0.9
sC3U1.SetSubsection(0, 2);                  //Set the subdivision to 2
sC3U1.SetTranRatio(0, 90);                  //Set the transmission ratio to 90
sC3U1.SetZeroOffset(0, sC3U1.GetCurrentStep(0));   //Set the current position as zero position
sC3U1.SetNegativeTravel(0, 10);             //Set the negative travel to 10
sC3U1.SetPositiveTravel(0, 10);             //Set the positive travel to 10
sC3U1.SaveParam(0);                         //Save all parameters
```