



# FUNCTION REFERENCE

DLL for Monochromator 7IMS Series

## Table of Contents

<b>Properties .....</b>	<b>2</b>
1. Port.....	2
2. ConnectStatus.....	2
3. Busy .....	2
4. Language .....	2
5. CurrentSpeed .....	2
6. MonochromatorType .....	3
7. MonochromatorNo .....	3
8. CurrentWaveLen.....	3
9. FilterStatus.....	3
10. FilterUseCount.....	4
11. FilterZeroStep .....	4
12. FilterTotalStep .....	4
<b>Functions .....</b>	<b>4</b>
1. ConnectPort.....	4
2. ClosePort .....	5
3. RunToWaveLen.....	5
4. RunToStep .....	5
5. StopRun.....	5
6. RunToZero .....	6
7. FreeIncrease.....	6
8. FreeDecrease.....	6
9. SaveAllParameter.....	7
10. OpenAllParameter.....	7
11. SetStartWaveLen .....	7

## Properties

### 1. Port

**Definition:** short Port

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

**C# code:**

```
short mPort;  
mPort = IMSU1.Port;
```

### 2. ConnectStatus

**Definition:** bool ConnectStatus

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

**C# code:**

```
bool mConnect;  
mConnect = IMSU1.ConnectStatus;
```

### 3. Busy

**Definition:** bool Busy

**Description:** Read only. The busy status of the monochromator.

**C# code:**

```
bool mBusy;  
mBusy = IMSU1.Busy;
```

### 4. Language

**Definition:** short Language

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

**C# code:**

```
short mLanguage;  
mLanguage = IMSU1.Language;
```

### 5. CurrentSpeed

**Definition:** short CurrentSpeed

---

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

**C# code:**

```
short mCurrentSpeed;  
mCurrentSpeed = IMSU1.CurrentSpeed;
```

**6. MonochromatorType**

**Definition:** string MonochromatorType

**Description:** Read only. Get the monochromator type.

**C# code:**

```
string mMonochromatorType;  
mMonochromatorType = IMSU1.MonochromatorType;
```

**7. MonochromatorNo**

**Definition:** string MonochromatorNo

**Description:** Read only. Get the monochromator number.

**C# code:**

```
string mMonochromatorNo;  
mMonochromatorNo = IMSU1.MonochromatorNo;
```

**8. CurrentWaveLen**

**Definition:** double CurrentWaveLen

**Description:** Read only. Get the current wavelength.

**C# code:**

```
double mCurrentWaveLen;  
mCurrentWaveLen = IMSU1.CurrentWaveLen;
```

**9. FilterStatus**

**Definition:** short FilterStatus

**Description:** Read only. Get the status of filter wheel. 2:not set, 0:disabled, 1:enabled.

**C# code:**

```
short mFilterStatus;  
mFilterStatus = IMSU1.FilterStatus;
```

## 10. FilterUseCount

**Definition:** short FilterUseCount

**Description:** Read only. Get the actual number of the filters.

**C# code:**

```
short mFilterUseCount;  
mFilterUseCount = IMSU1.FilterUseCount;
```

## 11. FilterZeroStep

**Definition:** int FilterZeroStep

**Description:** Read only. Get the zero position of the filter wheel.

**C# code:**

```
int mFilterZeroStep;  
mFilterZeroStep = IMSU1.FilterZeroStep;
```

## 12. FilterTotalStep

**Definition:** long FilterTotalStep

**Description:** Read only. Get the total step of the filter wheel.

**C# code:**

```
long mFilterTotalStep;  
mFilterTotalStep = IMSU1.FilterTotalStep;
```

# Functions

## 1. ConnectPort

**Definition:** void ConnectPort(short sPort)

**Description:** Connect the monochromator through the assigned port number.

**Parameters:**

sPort: the port number

**Return:**

Null

**C# code:**

```
short mPort=3;  
IMSU1.ConnectPort(mPort);    // connect the monochromator through COM3 port.
```

## 2. ClosePort

**Definition:** void ClosePort ()

**Description:** Close the connection to the monochromator.

**Parameters:**

Null

**Return:**

Null

**C# code:**

```
IMSU1.ClosePort();    // close the connection.
```

## 3. RunToWaveLen

**Definition:** short RunToWaveLen(double dWaveLen)

**Description:** Move to the assigned wavelength.

**Parameters:**

dWaveLen: the assigned wavelength

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;
```

```
mRet = IMSU1.RunToWaveLen(100);    // move to 100nm wavelength
```

## 4. RunToStep

**Definition:** short RunToStep(int iStep)

**Description:** Move to the assigned step.

**Parameters:**

iStep: the assigned step

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;
```

```
mRet = IMSU1.RunToStep(1000);    // move to the position of 1000 steps
```

## 5. StopRun

**Definition:** short StopRun()

**Description:** Stop the grating moving.

**Parameters:**

Null

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;  
mRet = IMSU1.StopRun();    // stop moving
```

## 6. RunToZero

**Definition:** short RunToZero()

**Description:** Go back to the zero position.

**Parameters:**

Null

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;  
mRet = IMSU1.RunToZero();    // go back to the zero position
```

## 7. FreeIncrease

**Definition:** short FreeIncrease()

**Description:** Increase the wavelength freely to the maximum value.

**Parameters:**

Null

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;  
mRet = IMSU1.FreeIncrease();    // increase freely
```

## 8. FreeDecrease

**Definition:** short FreeDecrease()

**Description:** Decrease the wavelength freely to the minimum value.

**Parameters:**

Null

**Return:**

The wrong index. -1:error, 1:success

**C# code:**

```
short mRet;  
mRet = IMSU1.FreeDecrease();    // decrease freely
```

## 9. SaveAllParameter

**Definition:** short SaveAllParameter(string sFilePath)

**Description:** Save all the parameters into the assigned file.

**Parameters:**

sFilePath: the path and file name to save

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;  
mRet = IMSU1.SaveAllParameter("C:\\1.txt");    // save all the parameters into  
1.txt file
```

## 10. OpenAllParameter

**Definition:** short OpenAllParameter(string sFilePath)

**Description:** Get all the parameters from the assigned file and set up.

**Parameters:**

sFilePath: the path and file name to open

**Return:**

-1:error, 1:success

**C# code:**

```
short mRet;  
mRet = IMSU1.OpenAllParameter("C:\\1.txt");    // get the parameters from  
1.txt and set up.
```

## 11. SetStartWaveLen

**Definition:** short SetStartWaveLen(double dWavelen)

**Description:** Set the initial wavelength.

**Parameters:**



---

dWavelen: the initial wavelength

**Return:**

-1:error, 1:success

**C# code:**

short iRet;

iRet = IMSU1.SetStartWaveLen(0);     // Set the initial wavelength to 0nm