mks

XPS-RL

Universal High-Performance Motion Controller/Driver

ESP COMPATIBLE

RoHS Compliant

Newport®

Software Drivers Manual

V1.0.x

Table of Contents

Universal High-Performance Motion Controller/Driver XPS-RL

1.0 What Are .Net drivers for XPS Controller?

.Net drivers support the creation of a user application that operates on a PC host computer and communicates with XPS-RL motion controllers. These drivers implements a rich set of controller operations and conceals from the application the complexity of low-level communication and synchronization with the controller.

The aim of this document is to explain customers how to integrate the XPS-RL .Net drivers into their programming language such as C#, Labview, IronPython and Matlab.

A separate Labview library (one vi per command) is available on our Newport website (XPS-RL web page).

2.0 How to Install .NET Drivers for XPS Controller?

2.1 Requirements

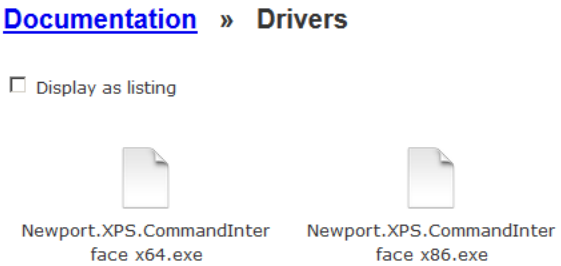
The PC host computer requires at least the .NET Framework 4.5.2 installed on it.

The .Net Framework is a programming infrastructure created by Microsoft for building, deploying, and running applications and services that use .NET technologies such as desktop custom applications.

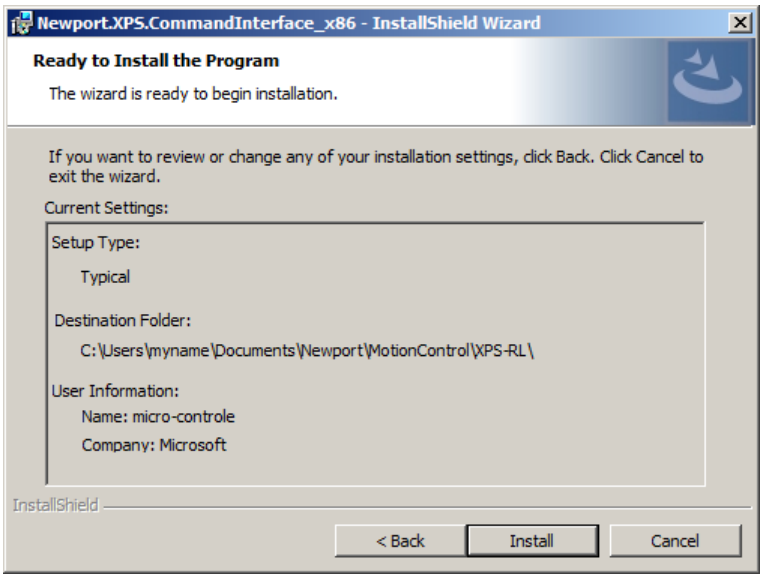
2.2 x86 Platform

First connect to the XPS-RL through the web site:

Once connected, go to the Documentation menu then Drivers submenu and download the Newport.XPS.CommandInterface x86.exe.



Once downloaded to the host PC, run the **Newport.XPS.CommandInterface_x86** executable file.



Once installed, the .Net assembly “Newport.XPS.CommandInterface.dll” V1.0.0.0 is located in GAC for x86 platforms:

C:\Windows\Microsoft.NET\assembly\GAC_32\Newport.XPS.CommandInterface\v4.0_1.0.0.0__9a267756cf640dcf

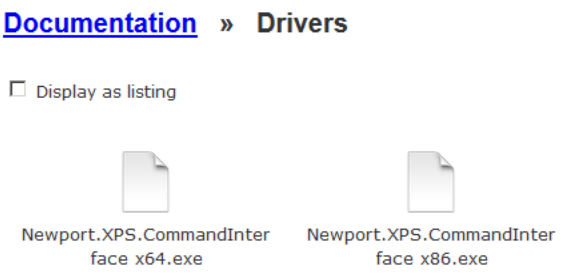
The sample application “XPSApplicationTest.exe” is located under:

C:\Users\myname\Documents\Newport\MotionControl\XPS-RL

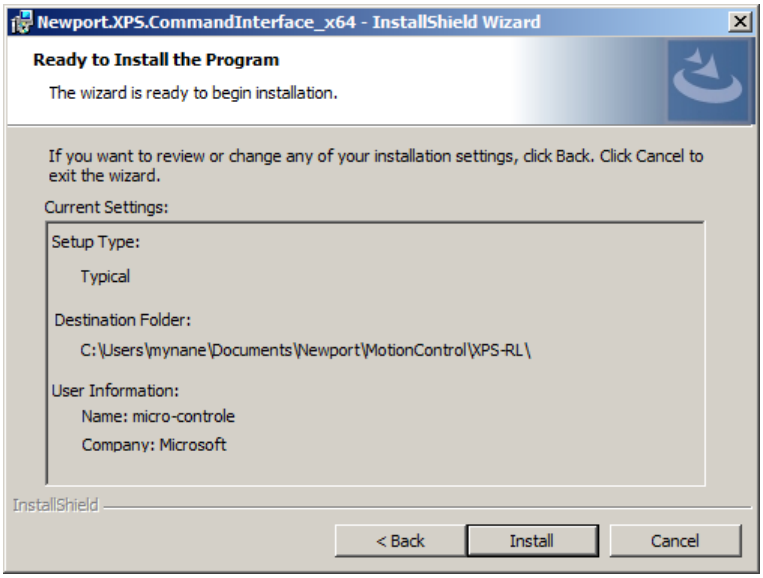
2.3 x64 Platform

First connect to the XPS-RL through the web site:

Once connected, go to the Documentation menu then Drivers submenu and download the Newport.XPS.CommandInterface x64.exe



Once downloaded to the host PC, run the **Newport.XPS.CommandInterface_x64** executable file.



Once installed, the .Net assembly “Newport.XPS.CommandInterface.dll” V1.0.0.0 is located in GAC for x64 platforms:

C:\Windows\Microsoft.NET\assembly\GAC_64\Newport.XPS.CommandInterface\v4.0_1.0.0.0__9a267756cf640dcf

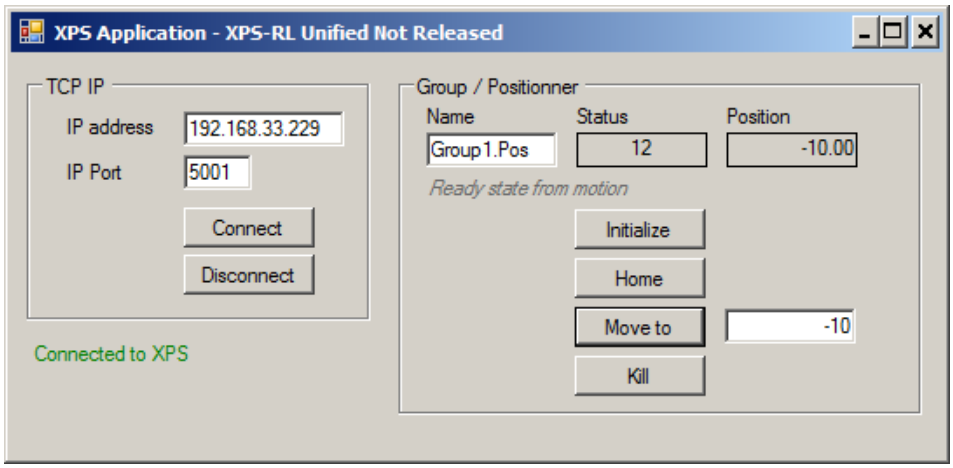
The sample application “XPSApplicationTest.exe” is located under:

C:\Users\myname\Documents\Newport\MotionControl\XPS-RL

3.0 How to Test .NET Drivers for XPS Controller?

Execute the XPS sample application "XPSApplicationTest.exe " from program files folder. This application uses **Newport.XPS.CommandInterface** assembly from GAC to test communication with XPS-RL controller.

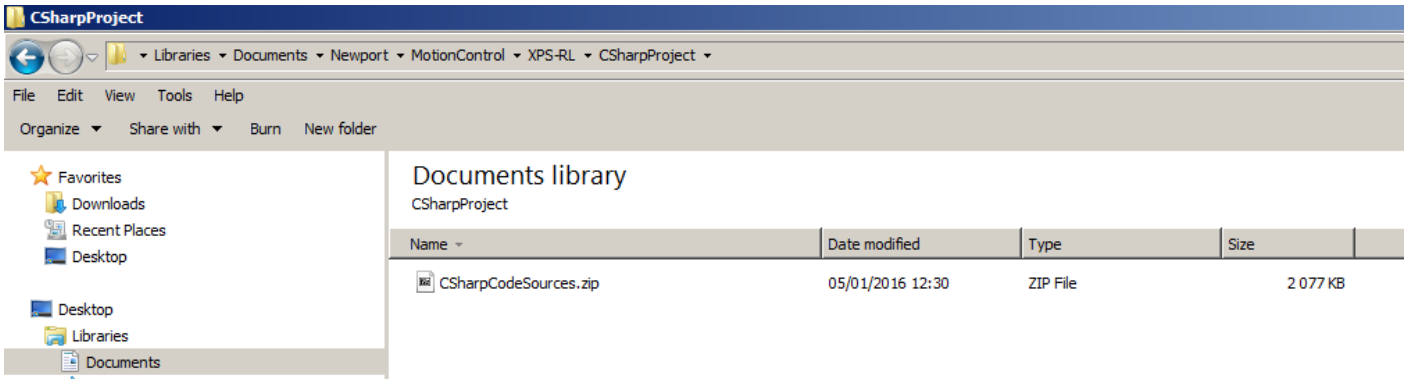
Several instances of this application can be running in parallel.



4.0 How to Access .Net C# Project

The C# project is available under User folder to show how to create a C# project for XPS-RL controller.

C:\Users\myname\Documents\Newport\MotionControl\XPS-RL\CSharpProject



This project is the one that has been used to create the provided XPS sample application.

5.0 How to Use XPS .NET Assembly from a Visual Studio C# Project?

Please refer to Microsoft Visual Studio web site to get more information to help you in your development (https://www.visualstudio.com/).

5.1 Add Reference to .NET Assembly

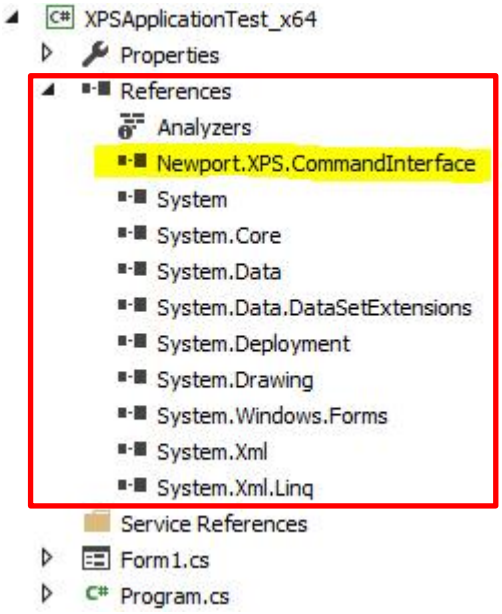
Add Newport.XPS.CommandInterface.dll in References of your project:

x64:

C:\Windows\assembly\GAC_64\Newport.XPS.CommandInterface\1.0.0.0__9a267756cf640dcf

x86:

C:\Windows\assembly\GAC_32\Newport.XPS.CommandInterface\1.0.0.0__9a267756cf640dcf



5.2 C# Code Sources

5.2.1 C# Header

```
using CommandInterfaceXPS; // Newport.XPS.CommandInterface .NET Assembly access
```

5.2.2 Add a Variable to Declare an "XPS" Object

```
CommandInterfaceXPS.XPS m_xpsInterface = null;
```

5.2.3 Create an Instance of "XPS" Object

```
m_xpsInterface = new CommandInterfaceXPS.XPS();
if (m_xpsInterface != null)
    ...
```

5.2.4 Open XPS Connection

```
if (m_xpsInterface != null)
    int returnValue = m_xpsInterface.OpenInstrument(m_IPAddress, m_IPPort,
    DEFAULT_TIMEOUT);
```

5.2.5 Call "XPS" Functions

```
if (m_xpsInterface != null)
{
    string XPSVersion = string.Empty;
    string errorString = string.Empty;
    int result = m_xpsInterface.FirmwareVersionGet(out XPSVersion, out
    errorString);
    if (result == CommandInterfaceXPS.XPS.FAILURE)
    ...
```

5.2.6 Close XPS Connection

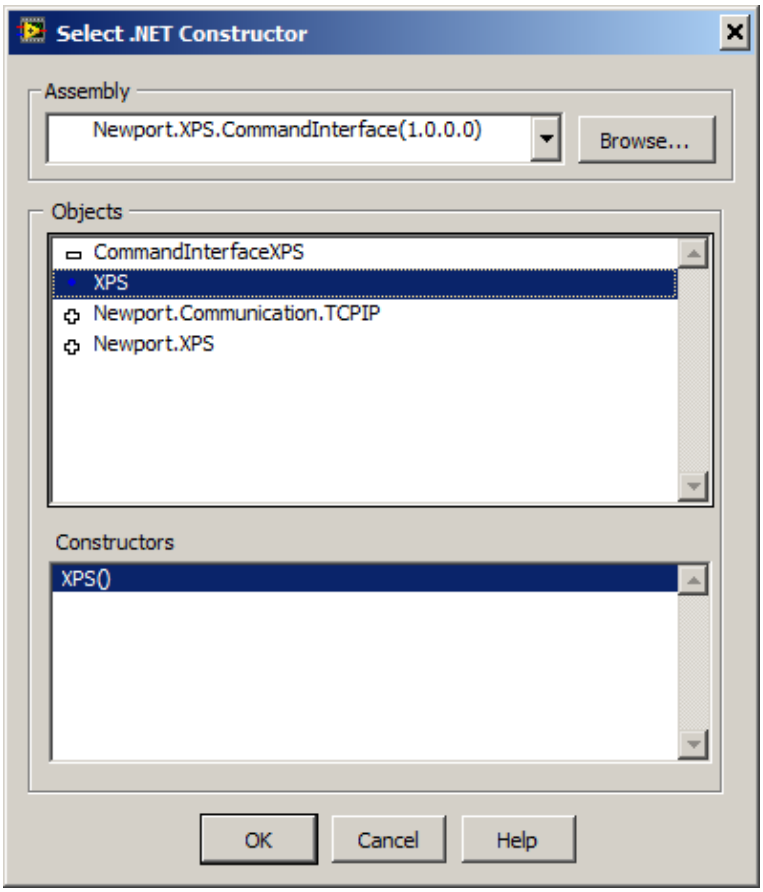
```
if (m_xpsInterface != null)
    m_xpsInterface.CloseInstrument();
```

6.0 How to use XPS .NET Assembly from a LabVIEW project?

Please refer to National Instruments web site to get more information to help you in your development (http://www.ni.com/labview/).

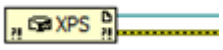
6.1 Add Reference to .NET Assembly

Select **CommandInterfaceXPS** and **XPS** constructor from a **.Net Constructor Node** (refer to Connectivity panel):

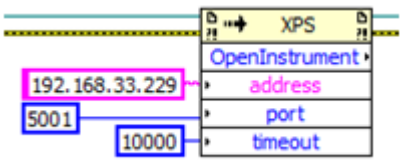


6.2 LabVIEW Code Sources

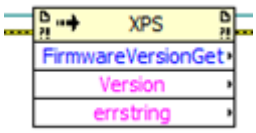
The instance of "XPS" object is created after configuration of **.Net Constructor Node**:



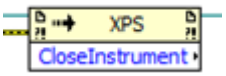
Open XPS connection (Use a **.Net Invoke Node** to select the XPS method "OpenInstrument"):



Call “XPS” functions (Use a **.Net Invoke Node** to select a XPS method):



Close XPS connection (Use a **.Net Invoke Node** to select the XPS method “CloseInstrument”):



Close .NET Reference:

7.0 How to Use XPS .NET Assembly Under IronPython?

Please refer to IronPython web site to get more information to help you in your development (http://ironpython.net/).

7.1 Add Reference to .NET Assembly

Add **Newport.XPS.CommandInterface.dll** in **References** of your script:

x64:

import sys

sys.path.append(r'C:\Windows\Microsoft.NET\assembly\GAC_64\Newport.XPS.CommandInterface\v4.0_1.0.0.0__9a267756cf640dcf')

x86:

import sys

sys.path.append(r'C:\Windows\Microsoft.NET\assembly\GAC_32\Newport.XPS.CommandInterface\v4.0_1.0.0.0__9a267756cf640dcf')

7.2 IronPython Code Source

7.2.1 IronPython Header

```
# The CLR module provide functions for interacting with the underlying
# .NET runtime
import clr

# Add reference to assembly and import names from namespace (IronPython)
clr.AddReferenceToFile("Newport.XPS.CommandInterface.dll")
from CommandInterfaceXPS import *
```

7.2.2 Create an Instance

```
# Create XPS interface
myXPS = XPS()
```

7.2.3 Open XPS Connection

```
def XPS_Open (address, port):
# Create XPS interface
myXPS = XPS()

# Open a socket
timeout = 1000
result = myXPS.OpenInstrument(address, port, timeout)
if result == 0 :
    print 'Open ', address, ":", port, " => Successful"
else:
    print 'Open ', address, ":", port, " => failure ", result

return myXPS
```

7.2.4 Call XPS Functions

```
def XPS_GetControllerVersion (myXPS, flag):
result, version, errString = myXPS.FirmwareVersionGet()
if flag == 1:
    if result == 0 :
        print 'XPS firmware version => ', version
    else:
        print 'FirmwareVersionGet Error => ',errString
return result, version

def XPS_GetControllerState (myXPS, flag):
result, state, errString = myXPS.ControllerStatusGet()
if flag == 1:
    if result == 0 :
        print 'XPS controller state => ', state
    else:
        print 'ControllerStatusGet Error => ',errString
return result, state
```

7.2.5 Close XPS Connection

```
def XPS_Close(myXPS):
myXPS.CloseInstrument()
```

8.0 How to Use XPS .NET Assembly Under Matlab?

Please refer to MathWorks web site to get more information to help you in your development (http://www.mathworks.com/products/matlab/).

8.1 Add Reference to .NET Assembly

```
% Make the assembly visible from Matlab
asmInfo = NET.addAssembly('Newport.XPS.CommandInterface')
```

8.2 Matlab Code Source

8.2.1 Create an Instance

```
% Make the instantiation
myxps=CommandInterfaceXPS.XPS();
```

8.2.2 Open XPS Connection

```
% Connect to the XPS controller
code=myxps.OpenInstrument('192.168.254.254',5001,1000);
```

8.2.3 Call XPS Functions

```
% Use API's
[code Version]=myxps.FirmwareVersionGet
[code]=myxps.GroupKill('Group1')
[code]=myxps.GroupInitialize('Group1')
[code]=myxps.GroupHomeSearch('Group1')
```

8.2.4 Close XPS Connection

```
% Disconnect from the XPS controller
code=myxps.CloseInstrument;
```

Service Form

Your Local Representative

Tel.: _____

Fax: _____

Name: _____

Company: _____

Address: _____

Country: _____

P.O. Number: _____

Item(s) Being Returned: _____

Model#: _____

Return authorization #: _____
(Please obtain prior to return of item)

Date: _____

Phone Number: _____

Fax Number: _____

Serial #: _____

Description: _____

Reasons of return of goods (please list any specific problems): _____

Newport®

Visit Newport Online at:
www.newport.com

North America & Asia
Newport Corporation
1791 Deere Ave.
Irvine, CA 92606, USA

Sales
Tel.: (800) 222-6440
e-mail: sales@newport.com

Technical Support
Tel.: (800) 222-6440
e-mail: tech@newport.com

Service, RMAs & Returns
Tel.: (800) 222-6440
e-mail: service@newport.com

Europe
MICRO-CONTROLE Spectra-Physics S.A.S
9, rue du Bois Sauvage
91055 Évry CEDEX
France

Sales
Tel.: +33 (0)1.60.91.68.68
e-mail: france@newport.com

Technical Support
e-mail: tech_europe@newport.com

Service & Returns
Tel.: +33 (0)2.38.40.51.55

mks

Newport® Ophir® Spectra-Physics®