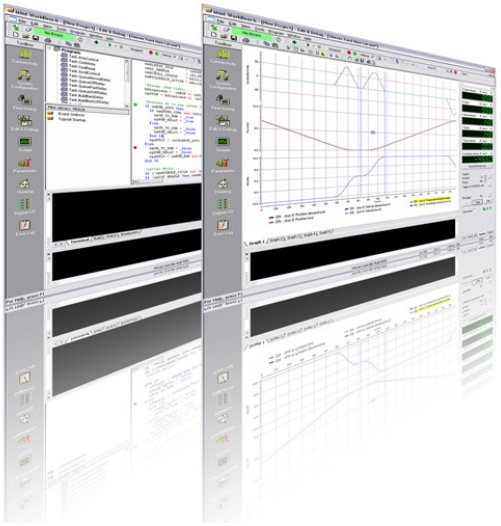
Motion Control Products

Application note
Using the Mint ActiveX control in Visual Studio

AN00127

Rev D (EN)

Visual Studio is the latest development environment from Microsoft. It supports a variety of different languages, including Visual Basic, Visual C++ and C#. All these languages fully support ActiveX controls, and hence the Mint ActiveX/COM components can be included in the development of host applications



Introduction

This document aims to show the reader how to get started with automating ABB's Mint motion control and drive products using Microsoft Visual Studio. Later, we will also discuss some of the differences that may be encountered when moving from Visual Basic 6 to Visual Studio.

Compatibility

Before we start with the steps on how to use the Active Control, we should first mention about some compatibility issues. Since Microsoft Visual Studio 2017, language Visual C++ cannot support ActiveX wizards. So, for C++ applications using Visual Studio 2017 or Visual Studio 2019, it is needed to import interface definition files (the mintcontrollerXXXX.h and mintcontrollerXXXX.cpp) manually. These are attached for convenience (in the Zip File - PART 2) or can be sourced by the user from Visual Studio 2015 to allow all necessary controls to work.

Note: In the attached ZIP file there are two versions of the interface definition files 5864 and 5860, both can be used with their respective active control versions.

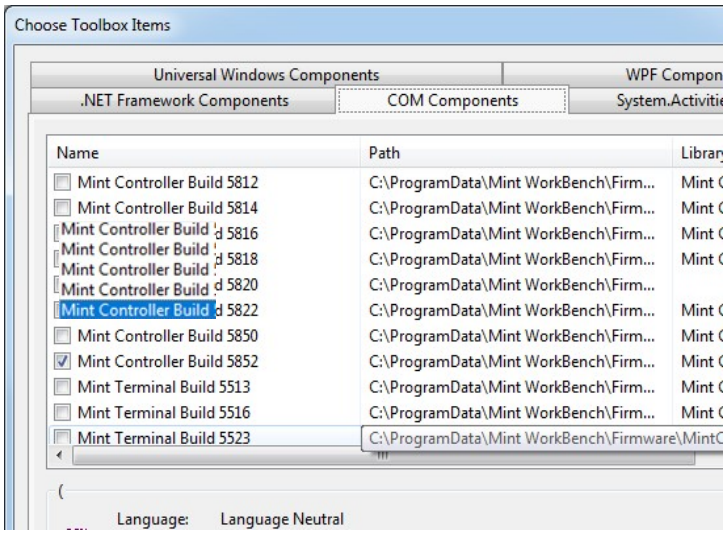
Microsoft Software version	Compatibility
Visual Basic 6	Fully
Visual Studio 2015	Fully
Visual Studio 2017	Needs files from Visual Studio 2015
Visual Studio 2019 (before v16.8)	Needs files from Visual Studio 2015

The ActiveX controls support both 32 bit and 64 bit platforms.

Note: The last ActiveX control was released in 2018 so some "newer drive keywords may be missing. No further development will be made on the ActiveX control by ABB.

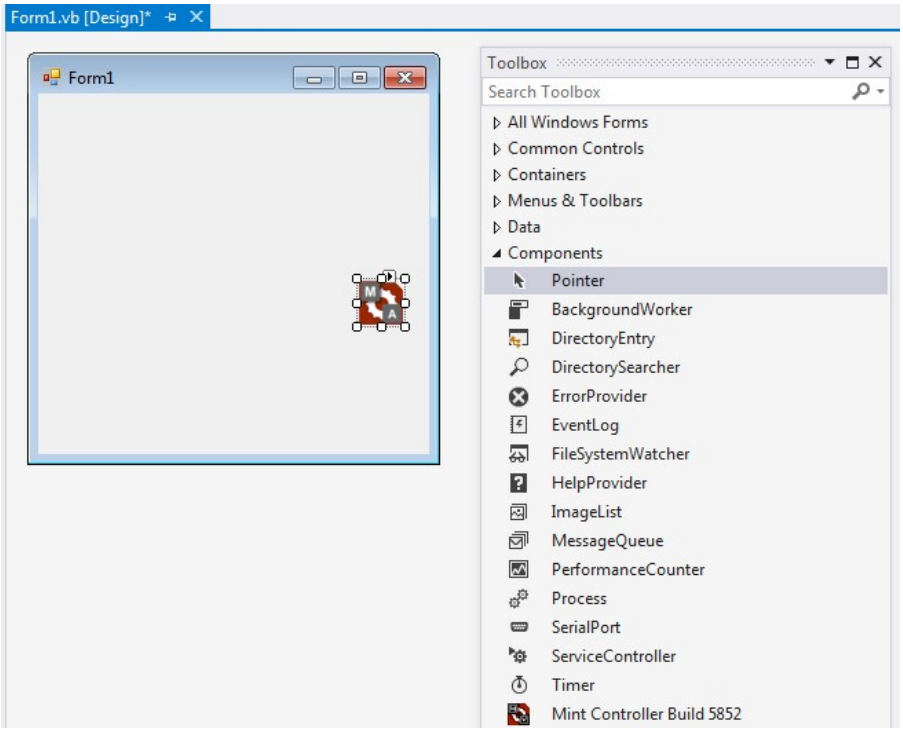
Including the Mint controls in a project

The first step to using the Mint ActiveX control in Microsoft Visual Studio is to add the control to the toolbox. To do this, select “Choose Toolbox Items…” from the Tools menu. In the resulting dialog, select the “COM Components” tab, which will list all COM (ActiveX) components that are registered on your computer. Listed amongst these should be the Mint Command Prompt control, the MintController control and the MintTerminal control. There will probably be several versions of each control registered, just choose the version you want to work with (if you’re not sure then choose the latest) by clicking in the small box to the left of the name (this will become ticked). When you have added the controls, you wish to use press OK.

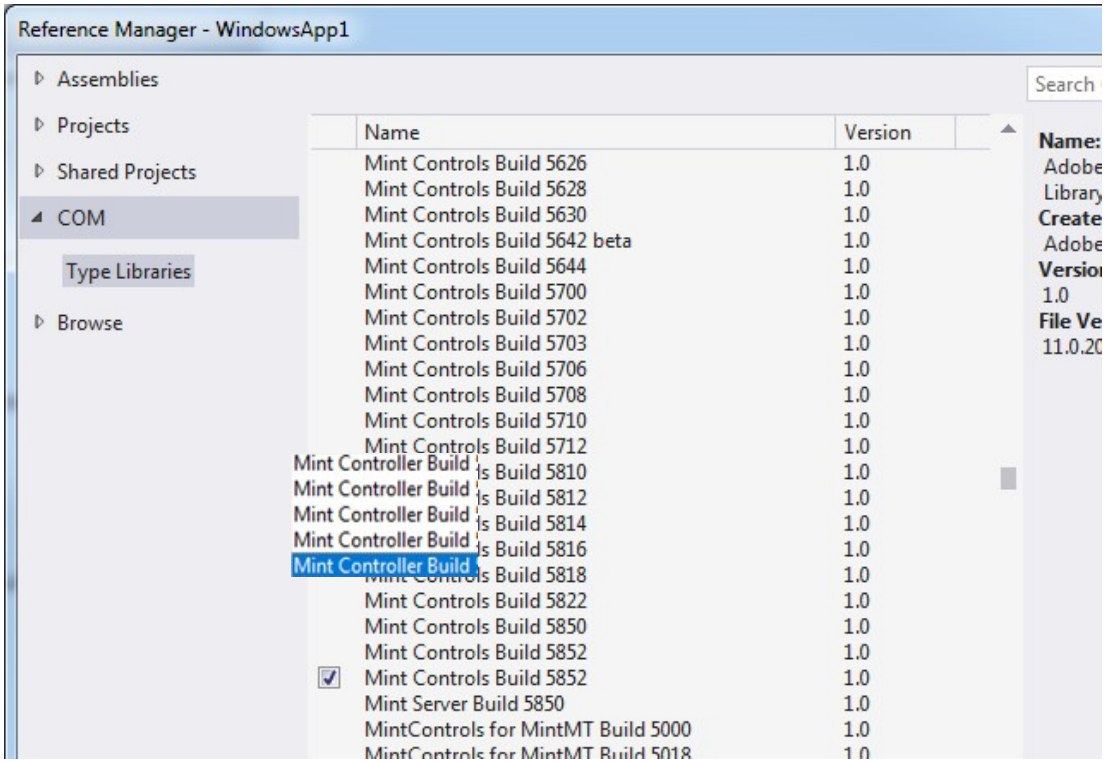


The ‘Mint Controller’ item is essential, it’s this control that you need to connect to the controller and issue motion commands or read/write data. The Terminal and Command Window controls are optional and are only needed if your host application needs to recreate the Terminal and Command Window features provided by Mint Workbench.

Once you have added the Mint controls to the Visual Studio (VS) toolbox you will find the ActiveX controls in the ‘Components’ section of the toolbox dialog. You can add a control to your project by selecting it in the toolbox and drawing it on your form.



Alternatively, the Mint controls can be added to the VS project as a reference instead of adding them as a toolbox item. Use the Project > Add Reference… menu item to do this and then select the COM>Type Libraries menu…



It is assumed the reader is familiar with the use of references in VS projects – if further information is required please refer to application *'AN00135 - Using Mint ActiveX Control with VBA*' or contact your local ABB support team.

New features in Visual Studio

If you have previously written host applications with the Mint ActiveX control in Visual Basic 6 then you will notice a few differences when using Microsoft Visual Studio.

Property Naming
In Visual Studio, ActiveX properties that have parameters (e.g. Speed) now have a set_ or get_ in front of the property name. For example:

VB6:
Controller.SPEED(0) = 20
value = Controller.SPEED(0)

VB .NET:
Controller.set_SPEED(0, 20)
value = Controller.get_SPEED(0)

So, when using the Mint ActiveX help file with Visual Studio, remember to add a set_ or get_ to the start of the name of any property with a parameter in the VB examples.

If you add the controls by reference (rather than drawing an object on a form) then the set_ and get_ prefixes are not used).

Method Calling
In Visual Basic 6, method calls could be made without using brackets:

VB6:
Controller.SetNextMoveESLink 2, 1, 57600, True

This is no longer supported, so all method calls must include brackets around the arguments:

VB .NET:
Controller.SetNextMoveESLink(2, 1, 57600, True)

Note that if you accidentally use the VB6 syntax, the Visual Basic .NET editor will automatically add the brackets to it.

Capture
One of the Mint ActiveX properties, CAPTURE causes a naming conflict in Visual Basic .NET and so is not accessible (note that other Visual Studio languages are not affected by this). From version 5212 of the Mint ActiveX control, MintController has a new property CAPTURESTATE which is functionally equivalent to CAPTURE but can be called from Visual Basic .NET:

VB6:
Controller.CAPTURE = 1

VB .NET:
Controller.CAPTURESTATE = 1

Note that CAPTURESTATE doesn't have any parameters, so it isn't prefixed with set_.

Using the Command Prompt and Terminal ActiveX controls (52xx builds)

In Visual Basic 6 you could easily connect the Terminal or Command Prompt controls to the MintController control using the setMintController method:

VB6:
```
Controller.SetMintDriveIILink 2, 1, 57600, True  ' create a controller connection
Terminal.setMintController Controller ' connect the controller to the terminal
```

This no longer works in Visual Studio. Instead, the old setMintControllerID function should be used:

VB .NET:
```
Controller.SetMintDriveIILink( 2, 1, 57600, True)
Terminal.setMintControllerID( Controller.ControllerID )
```

For the command prompt you also need to set up the compiler and symbol table:

VB .NET:
```
Controller.SetMintDriveIILink( 2, 1, 57600, True)
Prompt.setMintControllerID( Controller.ControllerID )
Prompt.setCompiler( 10 ) ' See below.  Must call this before setting symbol table.
Dim str As String = ""
Controller.GetSymbolTableForController( False, str )
Prompt.setSymbolTable( str )
```

The Compiler target format that is written with the setCompiler() call is linked to the version of the ActiveX control being used. The following table shows which compiler target format to use for a given ActiveX control.

ActiveX Control	Compiler
Builds 50xx	8
Builds 51xx	9
Builds 52xx	10
Builds 53xx	10
Builds 542x	13
Builds 545x	14
Builds 56xx	14
Builds 58xx	14

Using the Command Prompt and Terminal ActiveX controls (55xx builds and later)

For both Visual Basic 6 and Visual Studio the terminal and command prompt initialization was simplified. Both environments allow you to use:

```
Controller.setUSBControllerLink     (or whatever link function is called)
Prompt.setUSBControllerLink         (or whatever link function is called)
Terminal.setUSBControllerLink       (or whatever link function is called)
```

These calls handle all the processing of the compiler etc… so it's much simpler for the user.

It is always recommended to use the latest version of the Mint ActiveX control as we're always striving to simplify operation for the user!

Examples

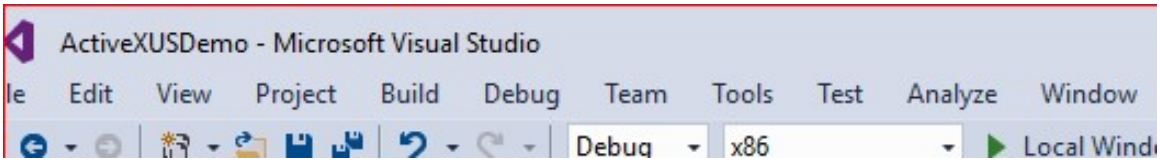
In the attached Zip File (PART 2), there are two example projects. One is written in Visual C++ 2015, the other is Visual Basic.net 2019

Example 1: ActiveXUSDemo.zip, it is a VC++ demo

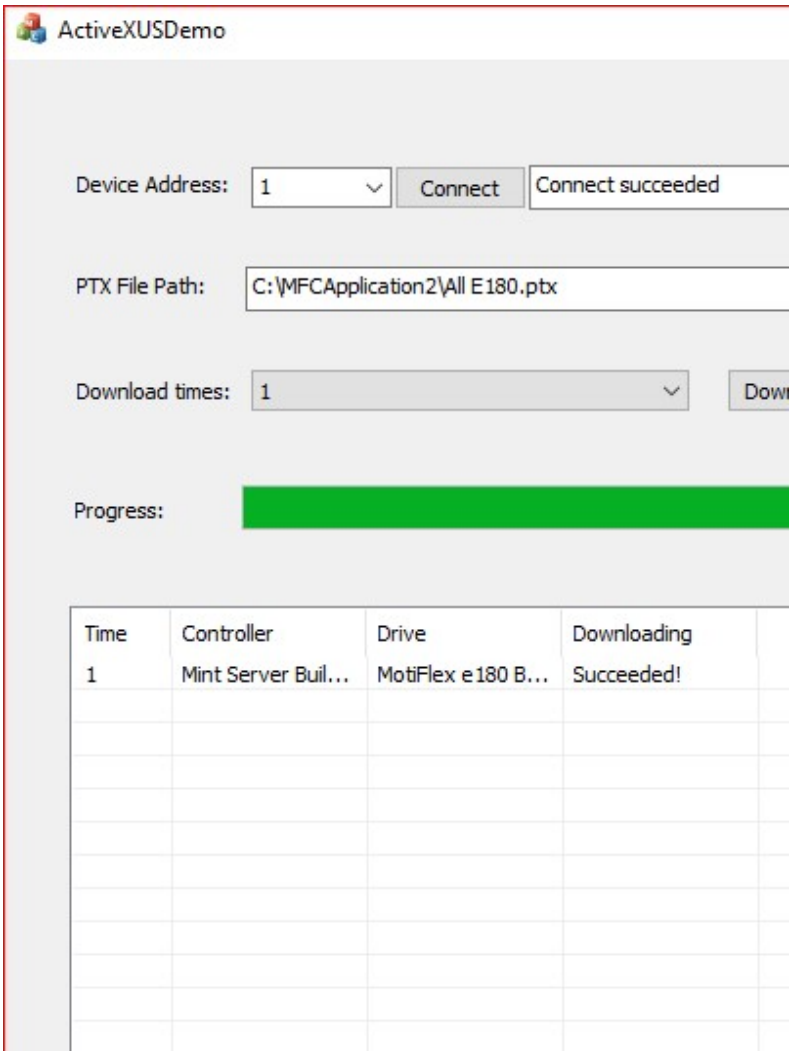
This application is built to download parameters to a connected ABB Servo device and is built in 32 bit compatibility mode.

How to use it:

1. You can see at the top, x86 (32 bit) mode is selected;



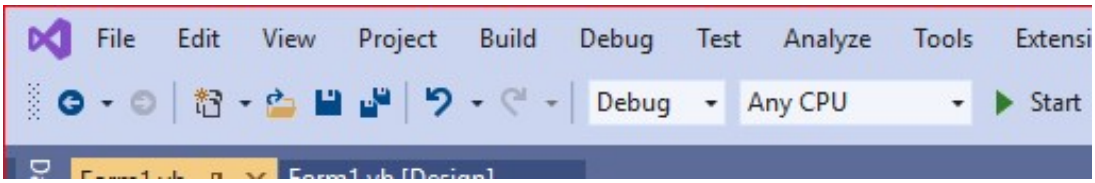
2. To use the program – run it
3. Select the Nextmove's address from the device address drop down
4. Hit the connect button
5. Select a .ptx file from local drive
6. Select download times, default is 1
7. Hit the download button, this app will download the. ptx to the drive;



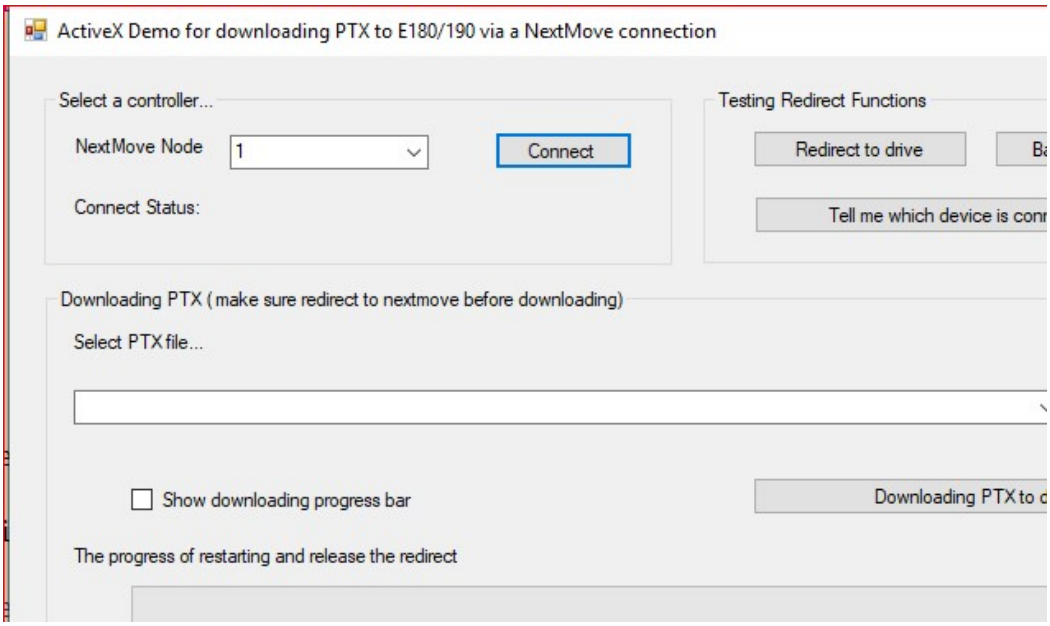
Example 2: NextMoveDownloadDemo

This is another example using vb.net 2019 which can be used to download the .ptx file to a Nextmove or other ABB Servo device, how to use it:

1. the build configuration is set as default “Any CPU”;



2. To use the program – run it
3. Select Node address
4. Press Connect button
5. After connection is established, select a .ptx file by the … button
6. Press downloading button to download the selected file.
7. Also, if the next move is connected, we could also test the redirecting functions. Hit the Redirect first, then hit the ‘tell me …’ button to learn which drive it is connecting now. After that, we could hit the ‘Back to NextMove’ button to release the redirecting.;



Contact us

For more information please contact your
local ABB representative or one of the following:

new.abb.com/motion
new.abb.com/drives
new.abb.com/drivespartners
new.abb.com/PLC