Running ModusToolbox™ using Windows Subsystem for Linux (WSL)

ModusToolbox™ tools package 3.8.0 or later

About this document

Scope and purpose

This document provides instructions for how to set up WSL and ModusToolbox™ software to provide a working environment with much better performance than Windows alone. It also provides information on how to enable the development in VS Code for ModusToolbox™ from WSL.

Intended audience

This document is intended for users who are knowledgable about using the Linux/Ubuntu environment, as WSL generally only provides a lightweight terminal interface. It provides some basic information about using the command line instructions, plus a guide for getting started with Linux and Bash.

Table of contents

Table of contents

About this document **1**
Table of contents **2**
1 Introduction **3**
2 Install (one-time setup actions) **4**
2.1 Install WSL2 4
2.2 Update Linux 6
2.3 Install Linux GUI app drivers 7
2.4 Install USB support 7
2.5 Install Qt support 7
2.6 Install ModusToolbox™ software 8
2.7 Install and test Visual Studio Code 8
2.8 Install Git tools (optional) 9
3 Run and configure (subsequent actions) **10**
3.1 Check network settings 10
3.2 Configure USB support 10
3.3 Configure serial UART ports 11
3.4 Use VS Code 11
3.5 Install/configureJ-Link (optional) 12
3.6 Install/run UART terminal 12
4 Troubleshooting and tips **13**
4.1 WSL network configuration issue 13
4.1.1 Manually update resolv.conf 13
4.1.2 Use resolv.conf update script 14
4.2 USB connection issue 14
4.2.1 Linux USB attach manual process 15
4.2.2 Linux USB attach script 16
4.3 Killing a hung WSL session 17
4.4 Setting default browser to Windows host 17
Revision history **18**
Disclaimer **19**

1 Introduction

WSL provides a way to use ModusToolbox™ tools in a Linux environment to provide a faster experience within a Windows system. We have also seen substantial performance improvement using WSL vs. Cygwin modus-shell, including builds taking ¼ of the time and opening tools taking ½ of the time.

To learn more about WSL, refer to this website: https://learn.microsoft.com/en-us/windows/wsl/about.

Installing and configuring WSL can be a little tricky with several factors such as WSL version, Windows version, networking issues, and VPN/firewall issues. The basic process involves installing the software once, configuring settings for your environment, and then repeating the configuration for subsequent sessions, as needed. The following shows an outline of the basic process, and the rest of this document describes the process in detail:

- Install (one-time setup actions):
  - Install WSL, USB, and miscellaneous software
  - Install ModusToolbox™ software
- Run and configure (subsequent actions):
  - Connect the kit
  - Run Ubuntu/PowerShell
  - Configure USB/UART
  - Create/develop ModusToolbox™ applications
- Troubleshooting and tips

Note: *We use WSL2 and assume Power Shell 7 is installed. Windows Terminal might be useful to install from the Microsoft Store.*

2 Install (one-time setup actions)

This section has the basic process to set up WSL on your system. This process assumes you are running Windows 11, and do not have any network policy restrictions. If you do have various security settings and/or IT policies, refer to the troubleshooting section later in this document. This process should work on Windows 10 build 19044+; however, Windows 11 is recommended. You only need to do these set-up steps once.

The main resource for setting up and using WSL is the Microsoft website: https://learn.microsoft.com/en-us/windows/wsl. This guide is based on our experience setting this up on our machines at Infineon.

2.1 Install WSL2

Part 1

To install WSL, open PowerShell as Administrator and type the following command:

```
wsl --install
```

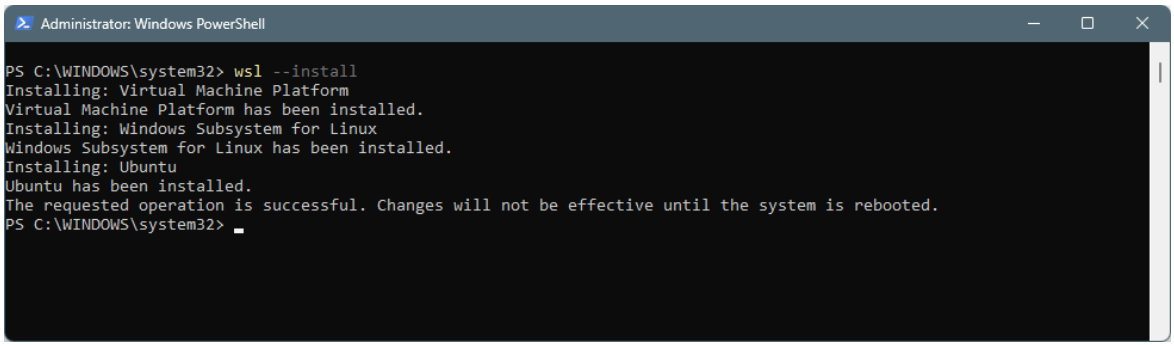
If the above command doesn't work, you may have to type:

```
wsl --install -d Ubuntu
```

If you use different a distribution, you need to supply the right name; for example:

```
wsl --install -d Ubuntu-24.04
```

This will install WSL2 by default. Various messages will display until the installation of WSL is complete.



You may also need other commands such as an update. You can also do `wsl --setdefault Ubuntu-24.04`, which will make that distribution start when you type `wsl`.

If you installed WSL1 previously, you'll need to upgrade the version. See https://learn.microsoft.com/en-us/windows/wsl/install for more details about various WSL options.

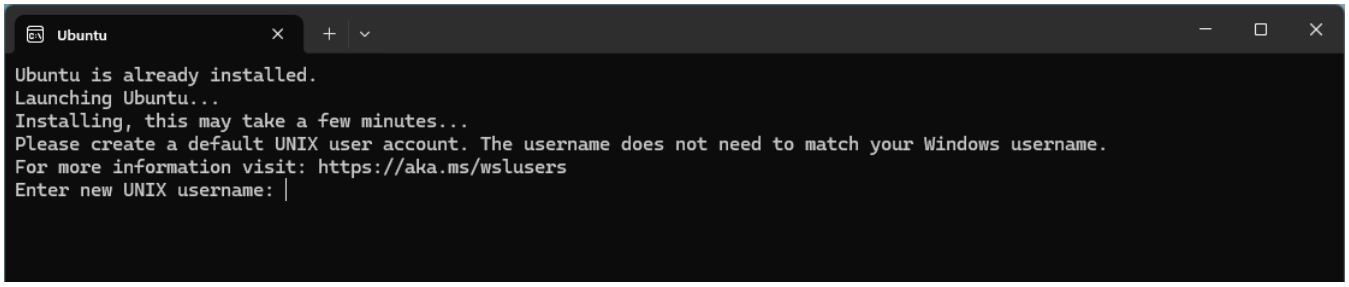
You can list your installed Linux distributions and check the version of WSL each is set to by entering the command:

```
wsl -l -v
```

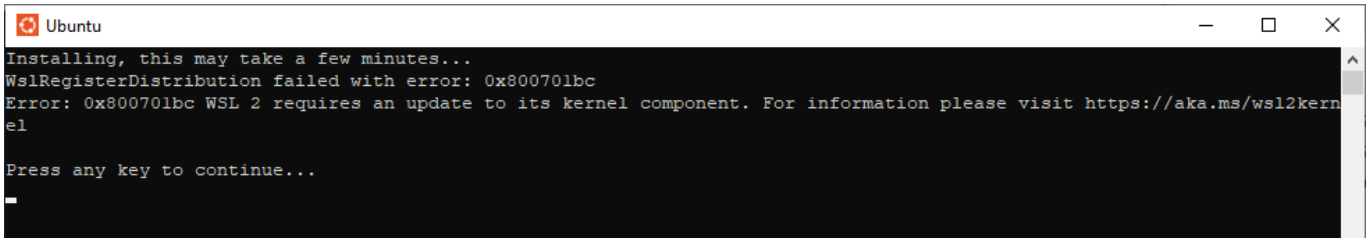
Install (one-time setup actions)

Part 2

As the prompt in the previous image shows, you need to reboot your computer. After rebooting is complete, the system launches the Ubuntu distribution in a separate window and prompts you to create a user name and password. These do not have to be the same credentials that you use on Windows.



In the case of some older Windows versions, you may instead see the following error.



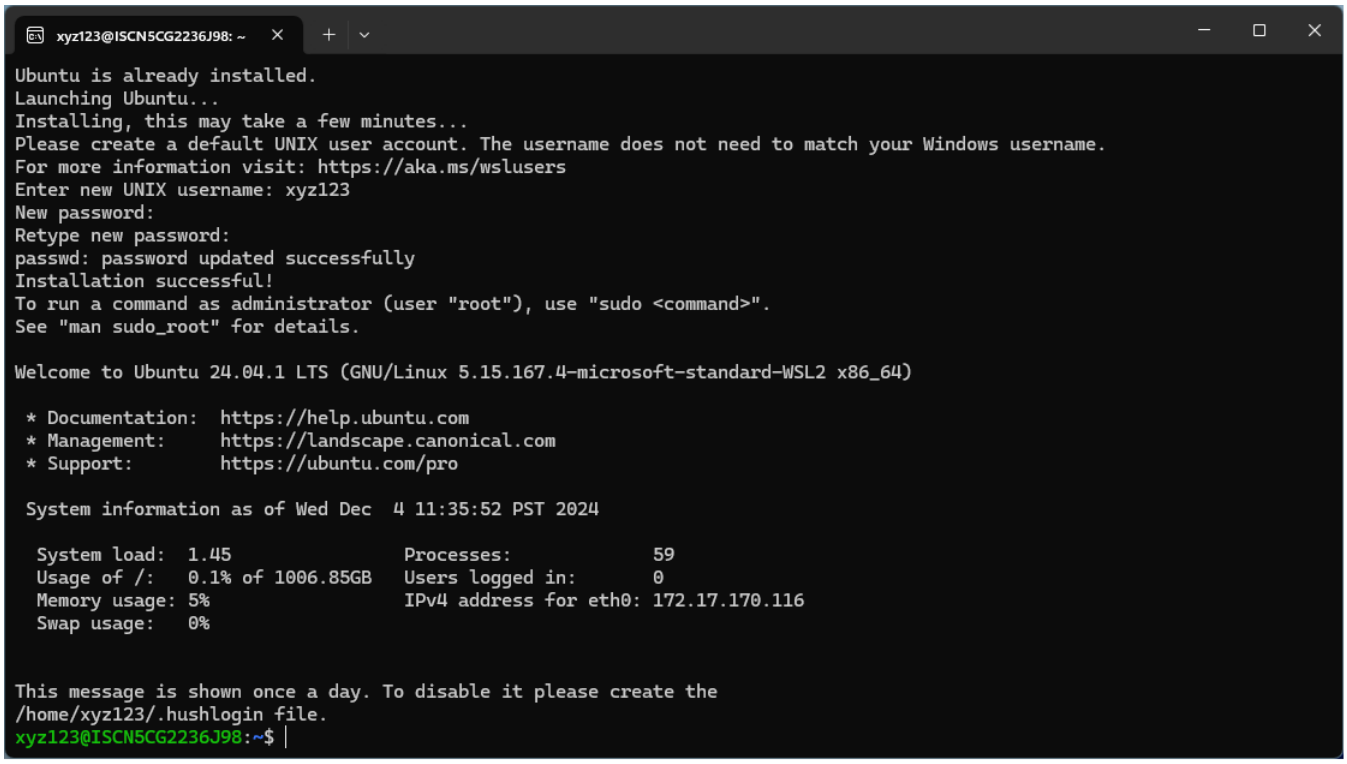
As instructed, press any key to close Ubuntu. Then in PowerShell, type the following command:

```
wsl --update
```

This command will update the system. When it finishes, it should launch Ubuntu again. Then it will prompt you to enter a user name and password. If Ubuntu doesn't launch, you can do it manually from the **Start** menu.

Install (one-time setup actions)

Follow the prompts to complete installation and account setup:



After the Ubuntu account is set up, exit Ubuntu by typing the following in PowerShell:

```
wsl --terminate Ubuntu
```

Note: *In some cases, Ubuntu might open inside the PowerShell terminal rather than a separate window. If that happens, type "exit" first before using the "--terminate" command.*

When you have completed the installation, terminate the Ubuntu terminal, exit PowerShell, and then relaunch PowerShell as Adminstrator.

2.2 Update Linux

Before using the Linux environment, it is a good idea to update things first.

Open the Ubuntu terminal again and type the following command:

```
$ sudo apt update && sudo apt upgrade
```

Note: *If you see an error like the following, it means the update failed; refer to WSL network configuration issue.*

```
Err:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
  Temporary failure resolving 'security.ubuntu.com'
W: Failed to fetch http://archive.ubuntu.com/ubuntu/dists/jammy/InRelease
Temporary failure resolving 'archive.ubuntu.com'
```

Install (one-time setup actions)

2.3 Install Linux GUI app drivers

If you want to run Linux GUI apps such a Google Chrome or Microsoft Edge, refer to this section on the Microsoft website: https://learn.microsoft.com/en-us/windows/wsl/tutorials/gui-apps

These steps are optional, but they might be helpful for a more cohesive experience as described on the site.

2.4 Install USB support

In order to interact with connected devices in the Linux environment, you need to install a program called "usbipd-win" on the Windows side and the associated Linux tools for Ubuntu.

1. In PowerShell, terminate Ubuntu using this command:

   ```
   wsl --terminate Ubuntu
   ```

2. Then, exit PowerShell.
3. Go to https://github.com/dorssel/usbipd-win/releases to download the MSI file and install it on your Windows PC.
4. Reopen the Ubuntu terminal and run the following commands:

   ```
   sudo apt install linux-tools-generic hwdata
   sudo update-alternatives --install /usr/local/bin/usbip usbip /usr/lib/linux-tools/*-generic/usbip 20
   ```

2.5 Install Qt support

Qt support is needed to run many of the ModusToolbox™ GUI tools.

1. In Ubuntu, use the following command to install Qt:

   ```
   $ sudo apt install qt6-base-dev
   ```

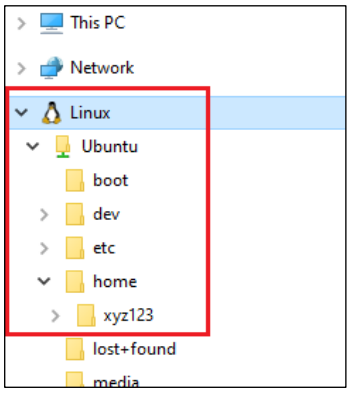
2. For high DPI monitors (see https://doc.qt.io/qt-5/highdpi.html), add the following lines in your Linux *.bashrc* file:

   ```
   export QT_AUTO_SCREEN_SCALE_FACTOR=0
   export QT_SCALE_FACTOR=2.0
   ```

2.6 Install ModusToolbox™ software

After you have the WSL environment set up, you can download and install the ModusToolbox™ software. You can install a web browser on Ubuntu if you prefer (see Install Linux GUI app drivers), or you can download using your Windows web browser and then move the files to the Ubuntu file system.

Note: You can access the Ubuntu file system from Windows Explorer under Linux/Ubuntu, and your home directory is under Linux/Ubuntu/home/<username>.



Refer to the KBA for installing the software without the Setup program for details. You will need to download and install several packages separately.

2.7 Install and test Visual Studio Code

ModusToolbox™ software supports several IDEs and the command line. For this application note, we'll use Visual Studio Code (VS Code). There are several ways to do this, but it is explained here: https://code.visualstudio.com/docs/remote/wsl

You should install VS Code on your Windows PC. If it is not already installed, close Ubuntu and exit PowerShell, and then install VS Code on Windows. Refer to https://code.visualstudio.com/docs/?dv=win64user for more information about how to download and install as needed. Refer also to the Visual Studio Code for ModusToolbox™ user guide for details about the extensions you need to install. Reboot your computer after installation is complete.

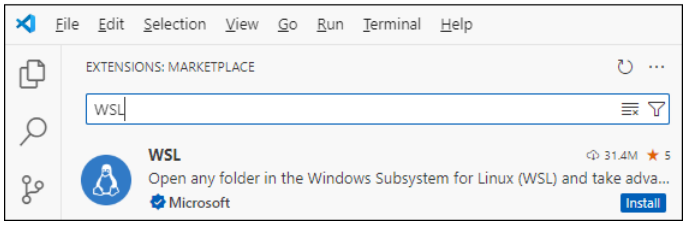
Then, open PowerShell as Administrator and open the Ubuntu terminal, navigate to where you want to open the tool, and type the following command. VS Code will install the server component into the WSL environment.

```
$ code .
```



Install (one-time setup actions)

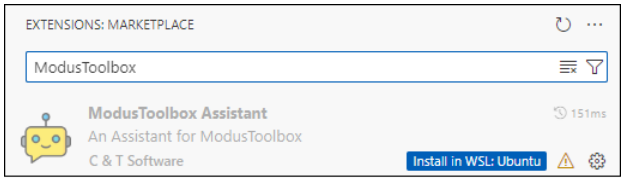
When this completes, VS Code will open. Go to the **Extensions** tab and install WSL if not already installed, or click **Reload Required** if it is already installed.



You may need to open a window in WSL to get this prompt.

Install the required extensions in WSL, including Cortex-Debug, Arm Assembly, and clangd.

You can also search for and install the ModusToolbox Assistant extension and click **Install in WSL**:



From there, you can create a ModusToolbox™ application. If you see any troubles with accessing the manifest, see WSL network configuration issue.

If everything works as expected, proceed to the Run and configure (subsequent actions) section to connect to the kit and program the application.

2.8 Install Git tools (optional)

Installing and configuring these Git tools could be helpful for your ModusToolbox™ development environment, but they are not required.

1. Install meld as a graphical diff tool, and configure Git to use it. This requires GUI app drivers described in section 1.4.

```
$ sudo apt install meld
$ git config --global diff.tool meld
$ git config --global merge.tool meld
$ git config --global difftool.prompt false
$ git config --global mergetool.keepBackup false
```

2. For Git behavior similar to Windows Gitbash, apply the following Git configuration commands:

```
$ git config --global user.name <Your Name>
$ git config --global user.email <Your.Email@example.com>
$ git config --global push.default current
$ git config --global pull.rebase false
$ git config --global branch.autoSetupRemote true
$ git config --global init.defaultbranch master
$ git config --global autocrlf true
```

3. Create SSH keys for your WSL installation using `ssh-keygen`. You may wish to install them in your online Git tooling profile (gitlab, github, bitbucket, etc. - for example per the instructions at https://docs.gitlab.com/ee/user/ssh.html). This removes the need for a password manager or credential storage when using git in WSL.

3 Run and configure (subsequent actions)

After installing all the software and configuring everything, it is best to reboot your computer one more time to ensure everything is finalized. Then you can configure the USB settings for your kit and begin using ModusToolbox™ software in WSL.

3.1 Check network settings

If you have network issues as described under WSL network configuration issue, then make sure that you resolve this. IP addresses may change after rebooting your computer, so you may need to update them. It is a good idea to run `nslookup www.google.com` when you open a new Ubuntu window to make sure the connection is working before proceeding.

3.2 Configure USB support

This process is needed to connect your kit through Ubuntu to program the device and debug the application.

1. Open Power Shell 7 as Administrator and the Ubuntu terminal (this is where the Windows Terminal becomes useful).
2. Make sure your kit is connected to the computer, and enter the following command in the Power Shell 7 to see a list of devices:

   ```
   usbipd list
   ```

Note: *For older versions of usbipd-win, the command is `usbipd wsl list`.*

3. Take a note of the BUSID for device you want to use. In this case it is 1-2; your kit might be different:

   ```
   BUSID  VID:PID    DEVICE                                                STATE
   1-2    04b4:f155  KitProg3 CMSIS-DAP, KitProg3 bridge, KitProg3 USB-UART (C...  Not shared
   1-4    17ef:3076  Billboard Interface, USB Input Device                 Not shared
   ```

4. Then, run these commands to bind and attach the desired device:

   ```
   usbipd bind -b 1-2
   usbipd attach -b 1-2 --wsl Ubuntu
   ```

   After a kit/USB port is bound once, it will always be available to WSL going forward, but you will need to attach it from PowerShell each time it is reconnected. If the "attach" command fails, it is likely there are security settings within your company blocking this. To work around this issue, see USB connection issue.

Note: *For older versions of usbipd-win, the command is `usbipd wsl attach --busid 1-2`. The "bind" command was not required previously, but it is in version 4.0.*

5. Then on Ubuntu, run the "lsusb" command to see the devices:

   ```
   xyz123@ISCNR90T1JHZ:~$ lsusb
   Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
   Bus 001 Device 004: ID 04b4:f155 Cypress Semiconductor Corp. KitProg3 CMSIS-DAP
   ```

Run and configure (subsequent actions)

3.3 Configure serial UART ports

On common Linux distributions, the serial UART ports (usually /dev/ttySx or /dev/ttyUSBx devices) belong to the root user and to the dialout or plugdev groups. Standard users may not be allowed to access these devices. On the Ubuntu terminal, type the following:

```
$ dmesg | grep -i serial
```

If you see messages like this, then you should be all set:

```
[    0.397799] Serial: 8250/16550 driver, 4 ports, IRQ sharing disabled
[    0.469610] usb usb2: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[    0.471140] usb usb2: SerialNumber: vhci_hcd.0
```

If you see errors or nothing, then an easy way to allow the current user access to the Linux machine's serial ports is by adding the user to the dialout and plugdev groups. This can be done using the following command:

```
$ sudo usermod -a -G dialout,plugdev $USER
```

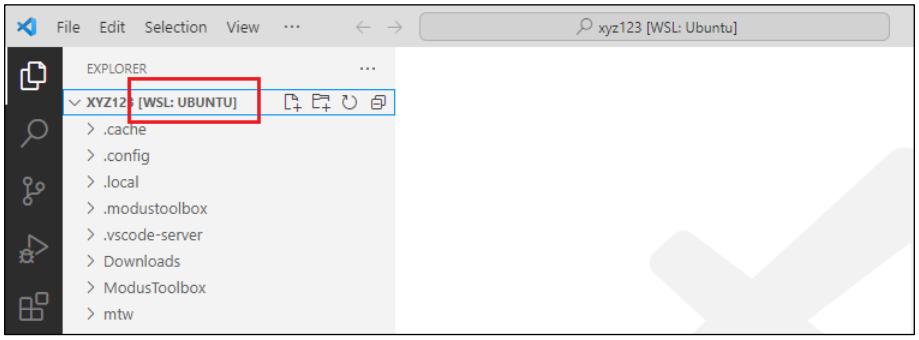
3.4 Use VS Code

Once the WSL environment is set up and configured, you can create and develop ModusToolbox™ applications.

From the Ubuntu terminal, run the the following to lauch VS Code:

```
code . &
```

When VS Code loads, look for **WSL: UBUNTU** under the Explorer view. This indicates that you are running VS Code in WSL.



From here you can create a new application. Then, you can build and program it, and then make any changes required.

You can also launch various tools such as Library Manager, BSP Assistant, Device Configurator from the ModusToolbox Assistant extension, or use the Ubuntu terminal with various "make" arguments. For example:

```
make library-manager
make bsp-assistant
make device-configurator
```

Refer to the Visual Studio Code for ModusToolbox™ user guide for more details.

Run and configure (subsequent actions)

3.5 Install/configureJ-Link (optional)

If you want to program/debug your application using J-Link instead of the KitProg3, you will have to install the software and configure your project in VS Code.

1. Download the J-Link software from the SEGGER website (https://www.segger.com/downloads/jlink/JLink_Linux_x86_64.deb).

   ```
   sudo dpkg -i JLink_Linux_x86_64.deb
   ```

2. Edit the application *Makefile* to add: `BSP_PROGRAM_INTERFACE=JLink`
3. Open a terminal and in the project directory and run: `make vscode`

You may have to restart VS Code and/or reboot your computer. When you follow process to Configure USB support, you'll see the JLink device instead of the KitProg.

Then, when you launch the VS Code Debugger, you'll see JLink configs. Refer to section 7 in the Visual Studio Code for ModusToolbox™ user guide for more details.

3.6 Install/run UART terminal

You will likely need a UART terminal for many of the ModusToolbox™ code examples. We recommend the program "screen" since it should be installed by default. To run screen, open the Ubuntu terminal and type the following, for example:

```
screen /dev/ttyACM0 115200
```

To kill the screen terminal and return to the command line, type **Ctrl-a, k**. For more information about using this tool, refer to https://linuxize.com/post/how-to-use-linux-screen/

You can also install PuTTY, but refer to this article for a common font error: https://devicetests.com/fix-unable-to-load-font-error-putty-ubuntu

4 Troubleshooting and tips

This section provides additional information for cases when you work with a VPN or firewall, when programs hang, or when you want to set the default browser to the Windows host.

- WSL network configuration issue
- USB connection issue
- Killing a hung WSL session
- Setting default browser to Windows host

4.1 WSL network configuration issue

If you're running on a company network with VPN and/or firewall, you may encounter connectivity issues that prevent you from updating or installing Linux packages. See https://learn.microsoft.com/en-us/windows/wsl/troubleshooting#wsl-has-no-network-connectivity-once-connected-to-a-vpn for how to resolve this issue.

For your convenience, here are the high-level steps to solve the problem (partially copied from https://randomwits.com/blog/wsl-2-vpn-internet):

1. In PowerShell, run the following command and note the DNS server IP addresses:

   ```
   PS C:\WINDOWS\system32> Get-DnsClientServerAddress -AddressFamily IPv4 | Select-Object -ExpandProperty ServerAddresses
   10.24.9.6
   10.64.177.25
   ```

   These IP addresses will vary for your computer, so change as appropriate in the following commands.

2. In Ubuntu, edit the */etc/wsl.conf* file to add the following lines:

   ```
   [network]
   generateResolvConf=false
   ```

4.1.1 Manually update resolv.conf

1. Edit the */etc/resolv.conf* file to add lines for the IP addresses found in the prior section. For example:

   ```
   nameserver 10.24.9.6
   nameserver 10.64.177.25
   ```

Note: *In addition to the nameservers provided by the Windows adapter, consider adding both private and public nameservers if your IT policies allow it. You may be able to avoid having to update this file when changing networks.*

Note: *If you experience difficulty editing this file, run these commands in the Ubuntu terminal:*

```
$ sudo unlink /etc/resolv.conf
$ sudo rm /etc/resolv.conf
$ sudo bash -c 'echo "nameserver 10.24.9.6" >> /etc/resolv.conf'
$ sudo bash -c 'echo "nameserver 10.64.177.25" >> /etc/resolv.conf'
$ sudo chattr +i /etc/resolv.conf
```

2. In the Ubuntu terminal, run the following to see if things are working. If this doesn't work, then reboot the system and try again:

   ```
   $ nslookup www.google.com
   ```

4.1.2 Use resolv.conf update script

You can simplify the resolve.conf connection using the following script that automates updating the file as an example. Edit this file according to your company and site settings. Then, run this whenever your connection changes (i.e., between home and work).

This script should be placed in a directory in your system path (i.e., *~/.local/bin/vpn_resolv_fix.sh*) and must be given executable priviledges (i.e., `chmod +x ~/.local/bin/vpn_resolv_fix.sh`).

```
#!/usr/bin/bash
#
# Helper script for ModusToolbox in WSL
#
# Updates resolv.conf per home/work network
#
sudo chattr -i /etc/resolv.conf
sudo rm /etc/resolv.conf

# Try to find the IP if we're on PPP network, i.e. at home via VPN.
# Update the PPP Adapter name to match your system.
WINHOST_IP="$(/mnt/c/WINDOWS/system32/ipconfig.exe -all | grep -A 10 "PPP adapter
MyCompany" | tail -n 1 | grep -Eo '\b([0-9]{1,3}\.){3}[0-9]{1,3}\b')"

# If the variable is empty, it means we're at work.  Find the IP from a different
interface.
if [ -z "$WINHOST_IP" ]; then
    WINHOST_IP="$(/mnt/c/WINDOWS/system32/ipconfig.exe -all | grep -A 16 "Wireless
LAN adapter Wi-Fi" | tail -n 1 | grep -Eo '\b([0-9]{1,3}\.){3}[0-9]{1,3}\b')"
fi

sudo echo "nameserver $WINHOST_IP" > /etc/resolv.conf
sudo chattr +i /etc/resolv.conf
```

Ensure that you correct the line breaks as needed when copying the script out of this document.

4.2 USB connection issue

By default, usbipd creates a rule to allow access to the TCP port 3240, which facilitates the use of the USB device in WSL. Depending on your firewall settings, you may not be able to make the connection work.

For example, if you run the "attach" command in PowerShell, you might see the following warnings:

> usbipd: warning: A group policy blocks the 'usbipd' firewall rule for the public network profile, which includes WSL.
>
> usbipd: warning: A third-party firewall may be blocking the connection; ensure TCP port 3240 is allowed.

There are a few potential issues. One could be that the installation was corrupted, and the usbipd rule was not created correctly. You can try to re-install or repair to resolve this.

Ensure that your Wi-Fi connection is not public. Public networks may have more stringent firewall rules that interfere, whereas private network settings will usually allow usbipd to work without issue.

Your IT department may have rules to block certain ports and may not allow this type of connection. If the following instructions do not work for you, you will need to contact your IT department to help resolve those types of problems.

4.2.1 Linux USB attach manual process

Without any port blocking issues, you only need to run "bind" and "attach" commands on the Windows side using PowerShell. However, if there are port blocking issues, you will need to run an "attach" command from the Linux side.

1. In PowerShell, run the "ipconfig.exe /all" command and note the IP address for the correct connection. This will vary from system to system and types of connection. For example, if you are on your company's Wi-Fi network, the correct IP address might look like this:

```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : my.company.com
   Link-local IPv6 Address . . . . . : XYZ
   IPv4 Address. . . . . . . . . . . : 10.14.44.28
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.14.44.1
```

Note: When connected to a VPN, the IP address is likely different. In our test case, the correct IP address was as shown here:

```
PPP adapter iconnect.infineon.com - my.vpn.com:

   Connection-specific DNS Suffix  . : my.vpn.com
   Description . . . . . . . . . . . : my.vpn.com
   Physical Address. . . . . . . . . :
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 10.64.237.88(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.255
   Default Gateway . . . . . . . . . :
```

2. In the Ubuntu terminal, enter the following "attach" command using the IP address found above, plus the original BUSID obtained from Configure USB support section:

```
$ sudo usbip attach --remote=10.14.44.28 --busid 1-2
```

Note: On Linux, the program is called usbip without the d as it is on Windows.

3. If the "attach" command fails, you will need to work with your IT department to troubleshoot the problem and determine the correct IP address to use.

Note: We found that running "ipconfig /renew" and then "ipconfig /all" ensured the correct IP address was used. Also, make sure there are no spaces in the "`--remote=#.#.#.#`" option.

4. If the "attach" command succeeds, run the "lsusb" command to see the device:

```
$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 04b4:f155 Cypress Semiconductor Corp. KitProg3 CMSIS-DAP
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

You should now be able to proceed with configuring serial UART ports. If you use a different kit and/or USB port in the future, you will need to bind that kit in Powershell. After a kit/USB port is bound once, it will always be available to WSL going forward, but you will need to attach it from Linux each time it is reconnected.

4.2.2 Linux USB attach script

You can simplify the USB connection using the following script that automates the Linux side of the process.

This script should be placed in a directory in your system path (i.e., *~/.local/bin/usb_attach.sh*) and must be given executable priviledges (i.e., `chmod +x usb_attach.sh`). Change the USER PARAMETERS as applicable for your system.

```
#!/usr/bin/bash

# This script attaches to a USB device that is already bound to usbipd in Windows.

# ============= USER PARAMETERS ===============

# Enter the name of the PPP adapter that appears when you use a VPN.
PPP_ADAPTER_NAME="PPP adapter MyCompany"
PPP_IP_OFFSET=7    # Number of lines below adapter name that IP address is found on.

# Enter the name of the adapter that provides your primary/preferred IP address when not on VPN.
OFFICE_ADAPTER_NAME="Wireless LAN adapter Wi-Fi"
OFFICE_IP_OFFSET=8  # Number of lines below adapter name that IP address is found on.

# This can be generically the registered USB name, or a serial number to identify a specific device.
# It simply needs to match something in the same line as the BUS ID in the output of 'uspid --list'
USB_DEVICE_SEARCH_NAME="Cypress Semiconductor Corp"

# ========== END USER PARAMETERS =============

# Try to find the IP if we're on PPP network, i.e. at home
WINHOST_IP="$(/mnt/c/WINDOWS/system32/ipconfig.exe -all | grep -A $PPP_IP_OFFSET "$PPP_ADAPTER_NAME" |
tail -n 1 | grep -Eo '\b([0-9]{1,3}\.){3}[0-9]{1,3}\b')"

# If the variable is empty, it means we're at work. Find the IP from a different interface.
if [ -z "$WINHOST_IP" ]; then
        WINHOST_IP="$(/mnt/c/WINDOWS/system32/ipconfig.exe -all | grep -A $OFFICE_IP_OFFSET
"$OFFICE_ADAPTER_NAME" | tail -n 1 | grep -Eo '\b([0-9]{1,3}\.){3}[0-9]{1,3}\b')"
fi

echo "searching for Cypress device..."
echo "/usr/local/bin/usbip list -r $WINHOST_IP | grep '$USB_DEVICE_SEARCH_NAME' | grep -Eo '[0-9]\-[0-
9]'"

BUSID="$(/usr/local/bin/usbip list -r $WINHOST_IP | grep "$USB_DEVICE_SEARCH_NAME" | grep -Eo '[0-9]\-
[0-9]')"

if [ ! -z "$BUSID" ]; then
        echo "attaching USB at $BUSID to remote Windows host at $WINHOST_IP"
        sudo usbip attach -r $WINHOST_IP --busid $BUSID
        if [ $? == 0 ]; then
                echo "Your device is available, see lsusb output for more details."
        else
                echo "Device could not be attached. Troubleshooting tips:"
                echo "  - Group privileges may be required, add your user to the dialout group or (not
recommended) run with sudo."
                echo "  - The Windows IP could not be discovered. Check the script's adapter name and
offsets are correct for your networking setup."
                echo "  - The USB device may already be attached."
        fi
else
        echo "No Cypress USB devices discovered. Troubleshooting tips:"
        echo "  - Make sure your device is plugged into a USB port and the power LED is on."
        echo "  - Check your Windows usbipd status in Powershell with 'usbipd list'."
fi
```

Ensure when copying the script out of this document to correct line breaks, as needed.

4.3 Killing a hung WSL session

There are times that WSL will fail to respond, often when running a GUI application that fails on a remote procedure call. At these times, often the `wsl.exe --shutdown` command will hang and not complete. To kill a non-responsive WSL session, open an admin Powershell and execute the following command:

```
PS C:\WINDOWS\system32> taskkill /t /f /im wslservice.exe
```

This force kills the the Ubuntu terminal and any processes that were running in it without requiring a reboot. Wait a few seconds, and then restart the Ubuntu terminal.

4.4 Setting default browser to Windows host

URL links from the Ubuntu terminal can be opened from a browser in Windows using the following steps. Adapt the steps to your preferred browser path. The steps here are for Google Chrome:

1. Use the following command to create a symlink from the Windows browser of your choice to a "winbrowser" command. Take care to properly escape the path:

   ```
   $ ln -s "/mnt/c/Program\ Files/Google/Chrome/Application/chrome.exe"
   ~/.local/bin/winbrowser
   ```

2. Then, append the following lines to your ~/.bashrc file:

   ```
   # Open links using the default Windows browser. winchrome is a symlink to the
   Windows Chrome browser.
   export BROWSER='winbrowser'
   ```

Re-source your *.bashrc* file or terminate the Ubuntu terminal. Then, URLs in the terminal, ModusToolbox™ files, or other tools should open in your other Windows browser.

Revision history

Document revision	Date	Description of changes
**	2023-10-12	New document
*A	2023-12-21	Updated to add more information and example scripts for the networking/USB issues.
*B	2024-12-10	Updated for newer versions of software.
*C	2025-06-30	Updated to include changes for version 3.5 tools package.
*D	2026-03-16	Updated the instructions to reference the separate KBA for installing ModusToolbox™ packages without using the Setup program.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2026-03-16

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2026 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-38846 Rev. *D

Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.