

Getting started with Pico on PC

Version: 2.0.3

Date: 18-Aug-2024

Important – these instructions are for a PC running Windows 10 or 11; they will not work for a Linux computer or for a Mac.

Contents

Introduction	2
Installing the software	3
Download the installer	3
Run the installer	3
Checkpoint 1	4
The Windows Command Prompt	5
Commands	5
dir (Directory Listing)	5
cd (Change Directory)	6
Environment variables	6
Keyboard shortcuts	8
TAB	8
Up and Down cursors	8
Opening the Pico - Developer Command Prompt	9
Checkpoint 2	9
Starting Visual Studio Code	11
Making it easier	12
Checkpoint 3	12
Selecting the default VS Code Terminal	13
Checkpoint 4	13
Install the project generator	15
Checkpoint 5	16
Setting up your work area	17
Creating a new project	19
Create a project	19
Checkpoint 6	19

Open the project	20
Working with a project	22
Exploring the project	22
Edit your code	22
Building your program	23
Checkpoint 7	23
Errors	23
Delivering a program to the Pico	24
Connecting the Pico to a Terminal	25
Source code	25
Serial Monitor	26
GitHub Desktop	27
Commit	28
Push	28
Cloud synchronisation	29
Putty	30
Debugging your program.....	32
Enable debugging.....	32
Set up the Debug Probe.....	32
Set up Picoprobe.....	33
Debugging a Project	34
Breakpoints	35
Troubleshooting	37
Previous installation	37

Introduction



As you step through this guide, please ensure that you check that your setup passes each of the checkpoints marked like this.

If you have already set up your computer, but you are encountering problems, please start by walking through each of the checkpoints.

Installing the software

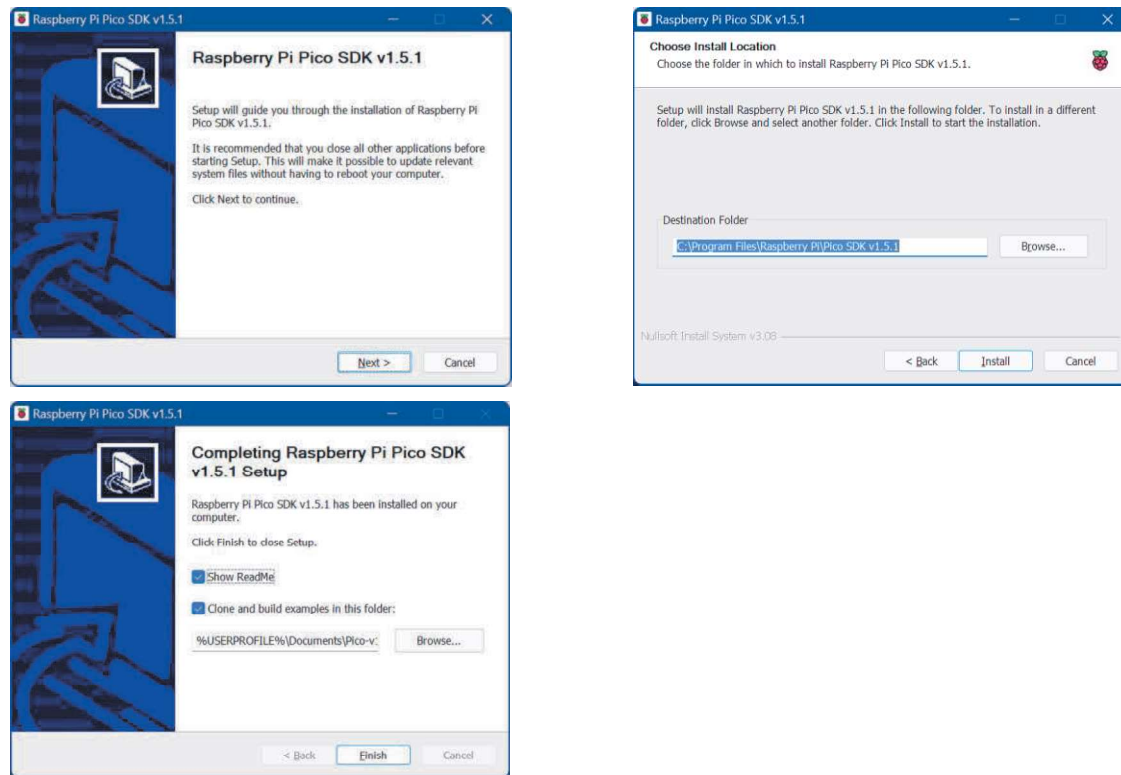
Download the installer

Go to <https://github.com/raspberrypi/pico-setup-windows/releases/tag/v1.5.1> and download the installer, which is named **pico-setup-windows-x64-standalone.exe**.

Run the installer

Note - You must have administrator privileges on your PC to run the installation program.

Follow the default settings as you install the program:



When the installation is complete, the examples will be copied and built:



```
pico-setup - call "C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-setup.cmd" "C:\Users\jc454533\Documents\Pico-v1.5.1" 1
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/gcc-arm-none-eabi/bin/arm-none-eabi-g++.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- The ASM compiler identification is GNU
-- Found assembler: C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/gcc-arm-none-eabi/bin/arm-none-eabi-gcc.exe
Build type is Debug
Using regular optimized debug build (set PICO_DEOPTIMIZED_DEBUG=1 to de-optimize)
Defaulting PICO target board to pico since not specified.
Using board configuration from C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/pico-sdk/src/boards/include/boards/pico.h
-- Found Python3: C:/Users/jc454533/AppData/Local/Programs/Python/Python310/python.exe (found version "3.10.5") found components: Interpreter
TinyUSB available at C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/pico-sdk/lib/tinyusb/src/portable/raspberrypi/rp2040; enabling build support for USB.
Compiling TinyUSB with CFG_TUSB_DEBUG=1
BTstack available at C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/pico-sdk/lib/btstack
cyw43-driver available at C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/pico-sdk/lib/cyw43-driver
Pico W Bluetooth build support available.
lwIP available at C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/pico-sdk/lib/lwip
mbedtls available at C:/Program Files/Raspberry Pi/Pico SDK v1.5.1/pico-sdk/lib/mbedtls
-- Configuring done
-- Generating done
-- Build files have been written to: C:/Users/jc454533/Documents/Pico-v1.5.1/pico-examples/build
Building blink
[51/51] Linking CXX executable blink\blink.elf
Building "hello_world/all"
[118/118] Linking CXX executable hello_world\usb\hello_usb.elf
Press any key to continue . . .
```

Checkpoint 1



Did the examples build process run and complete successfully? Do the last five lines resemble the lines in the picture above?

The Windows Command Prompt

Important: if you have not used the Windows command line environment before, read this section.

From time to time in this course, you will need to work with the Windows Command Prompt.

The Windows Command Line, often referred to as the Command Prompt or simply "cmd," is a text-based interface used to interact with the operating system. It allows users to execute commands, navigate through directories, and manage files directly from the keyboard, providing a more powerful and efficient way to perform tasks compared to the graphical interface.

When you open the Command Prompt, you're greeted with a prompt that typically displays the current directory path, followed by a greater-than symbol (>), and then a blinking cursor. For example:

```
C:\Users\YourName>
```

This prompt indicates that you are in the `C:\Users\YourName` directory and ready to enter commands.

Commands

Two fundamental commands in the Windows Command Line are `dir` and `cd`:

`dir` (Directory Listing)

The `dir` command is used to list the contents of the current directory. When you type `dir` and press Enter, you'll see a list of all files and subdirectories within the current directory, along with details such as file sizes and dates.

```
C:\repos\cc2511\JSmith>dir
Volume in drive C is Windows
Volume Serial Number is F845-B6B2

Directory of C:\repos\cc2511\JSmith

10/08/2024  11:55 AM    <DIR>          .
10/08/2024  11:52 AM    <DIR>          ..
10/08/2024  11:55 AM    <DIR>          Lab2
10/08/2024  11:55 AM    <DIR>          Lab3
               0 File(s)                0 bytes
               4 Dir(s)  334,560,444,416 bytes free
```

If you type a name after the command, you will see a listing of the named sub-folder:

```
C:\repos\cc2511\JSmith>dir Lab2
Volume in drive C is Windows
Volume Serial Number is F845-B6B2

Directory of C:\repos\cc2511\JSmith\Lab2

10/08/2024  11:55 AM  <DIR>          .
10/08/2024  11:55 AM  <DIR>          ..
10/08/2024  11:55 AM                169 .gitignore
10/08/2024  11:55 AM                388 CMakeLists.txt
10/08/2024  11:55 AM                364 main.c
10/08/2024  11:55 AM            2,802 pico_sdk_import.cmake
               4 File(s)              3,723 bytes
               2 Dir(s)  334,540,525,568 bytes free
```

cd (Change Directory)

The `cd` command is used to change the current directory. To navigate to a different folder, type `cd` followed by the path of the directory you want to enter. If the directory is within the current one, you can simply type its name. e.g.:

```
C:\repos\cc2511\JSmith>cd Lab2

C:\repos\cc2511\JSmith\Lab2>
```

To move up one level, use: `cd ..`

```
C:\repos\cc2511\JSmith\Lab2>cd ..

C:\repos\cc2511\JSmith>
```

To navigate to a specific directory on the current disk, type the full path, starting with a back-slash, e.g:

```
C:\>cd \repos\cc2511\JSmith

C:\repos\cc2511\JSmith>
```

Environment variables

A command environment can include a number of variables, which are aliases for other values – often directory paths. You can see a list of all the active environment variables by typing `set` (followed by Enter).


```
C:\repos\cc2511\JSmith>set
ALLUSERSPROFILE=C:\ProgramData
APPDATA=C:\Users\jc359743\AppData\Roaming
CHROME_CRASHPAD_PIPE_NAME=\\.\pipe\crashpad_26844_CNGVVOEALKTKXUUP
CMAKE_GENERATOR=Ninja
CommonProgramFiles=C:\Program Files\Common Files
CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files
CommonProgramW6432=C:\Program Files\Common Files
COMPUTERNAME=8384BY3
ComSpec=C:\Windows\system32\cmd.exe
DriverData=C:\Windows\System32\Drivers\DriverData
EFC_1468=0
errors=0
FPS_BROWSER_APP_PROFILE_STRING=Internet Explorer
FPS_BROWSER_USER_PROFILE_STRING=Default
HOME=C:\Users\jc359743
HOMEDRIVE=C:
HOMEPATH=\Users\jc359743
```

The variables are listed in alphabetical order. Scroll up and down to find the variable you are interested in.

In this course we will be using the variables below extensively:

```
PICO_repos_PATH=C:\Users\jc359743\Documents\Pico-v1.5.1
PICO_SDK_PATH=C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk
```

You can substitute the value of an environment variable into a command by enclosing it in percent symbols, e.g.:

```
C:\Users\jc359743>dir "%PICO_repos_PATH%\ppgen"
Volume in drive C is Windows
Volume Serial Number is F845-B6B2

Directory of C:\Users\jc359743\Documents\Pico-v1.5.1\ppgen

07/08/2024  05:14 PM    <DIR>          .
07/08/2024  05:14 PM    <DIR>          ..
07/08/2024  05:14 PM                1,928 .gitignore
07/08/2024  05:14 PM                1,099 LICENSE
07/08/2024  05:14 PM                3,935 ppgen.py
07/08/2024  05:14 PM                 33 README.md
07/08/2024  05:14 PM    <DIR>          templates
               4 File(s)              6,995 bytes
               3 Dir(s)  335,478,661,120 bytes free
```

Keyboard shortcuts

TAB

Use the TAB key to complete the name of a folder or file in the current command. If there are multiple files that match the current input, they will be listed in alphabetical order: hit TAB once for the first match, press it again for the second, and so on.

Using the TAB key has 2 major advantages:

- 1) It saves on typing.
- 2) It reduces errors.

Up and Down cursors

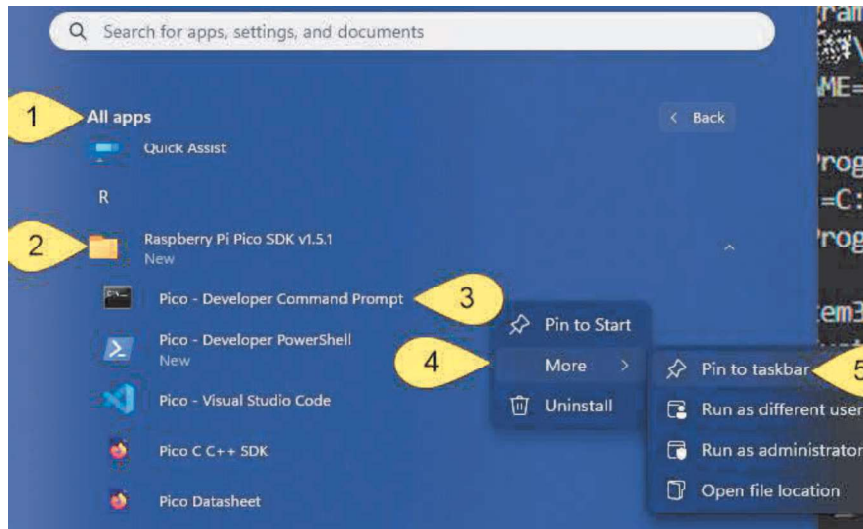
Use the UP cursor key to display the previous command entered at this command line. Keep pressing UP to scroll through a history of the commands. You can use DOWN to scroll forward through the list.

This saves time and mistakes during repetitious command entries.

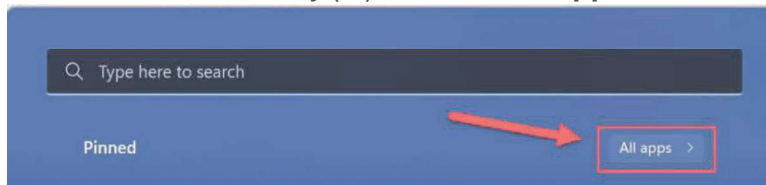
Opening the Pico - Developer Command Prompt

We will use a specific variant of the Windows command prompt in this course, which has the requires environment variables set up automatically. This is listed under **All apps**, in the **Raspberry Pi Pico SDK v1.5.1** folder, as **Pico - Developer Command Prompt**.

Save time by pinning it to the task bar, so that you won't need to find it every time you need it.



1. Press the Windows key (⊞) and click **All apps**:



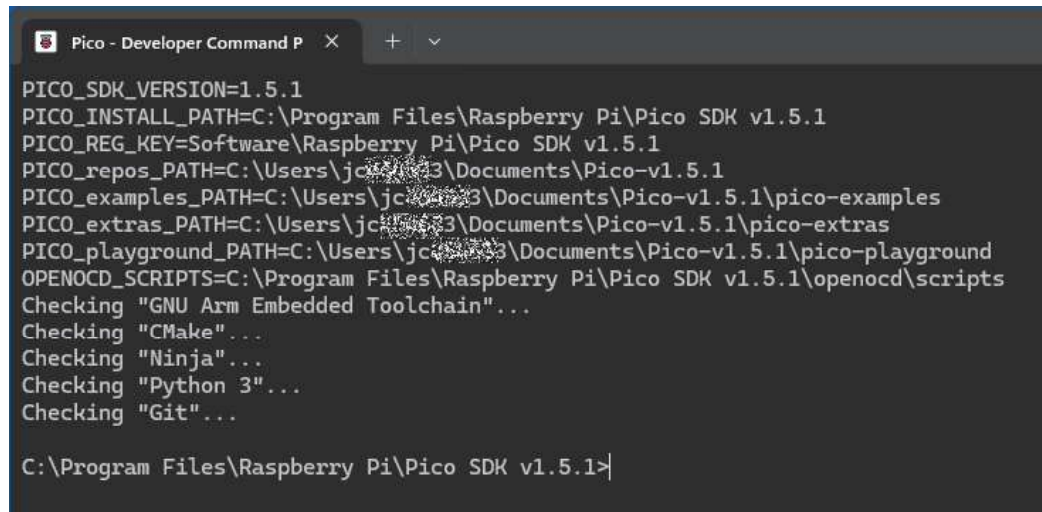
Scroll down to the **Raspberry Pi Pico SDK v1.5.1** folder and click on it to open it.

2. Right-click on the **Pico - Developer Command Prompt** item.
3. Click **More >**.
4. Then click **Pin to taskbar**.

Checkpoint 2



1. Find the **Pico - Developer Command Prompt** icon on the task bar and click it.
You should see a window resembling the following:



```
PICO_SDK_VERSION=1.5.1
PICO_INSTALL_PATH=C:\Program Files\Raspberry Pi\Pico SDK v1.5.1
PICO_REG_KEY=Software\Raspberry Pi\Pico SDK v1.5.1
PICO_repos_PATH=C:\Users\jch\Documents\Pico-v1.5.1
PICO_examples_PATH=C:\Users\jch\Documents\Pico-v1.5.1\pico-examples
PICO_extras_PATH=C:\Users\jch\Documents\Pico-v1.5.1\pico-extras
PICO_playground_PATH=C:\Users\jch\Documents\Pico-v1.5.1\pico-playground
OPENOCD_SCRIPTS=C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\openocd\scripts
Checking "GNU Arm Embedded Toolchain"...
Checking "CMake"...
Checking "Ninja"...
Checking "Python 3"...
Checking "Git"...

C:\Program Files\Raspberry Pi\Pico SDK v1.5.1>
```

2. Note that the title bar contains “***Pico – Developer Command Prompt***”.
3. Note that a set of environment variables, including PICO_repos_PATH, are created at the start of the environment.

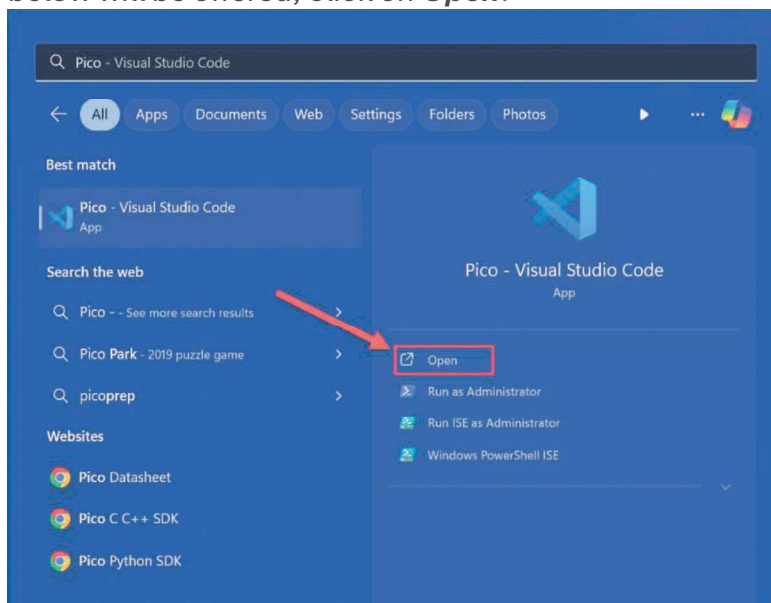
Starting Visual Studio Code

In your Start Menu, look for the “Pico – Visual Studio Code” shortcut, in the “Raspberry Pi” folder. The shortcut sets the required environment variables and then launches Visual Studio Code.

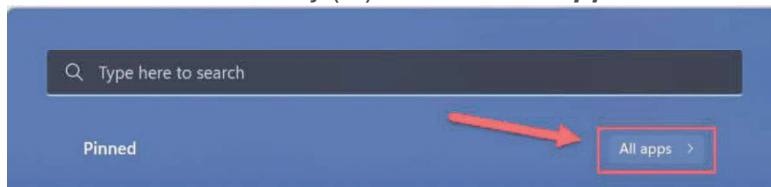
Note – you may already have a copy of VS Code on your computer. Do not start the existing application: make sure that you use the shortcut “Pico – Visual Studio Code” as described above.

On Windows 11, there are two ways you can start VS Code:

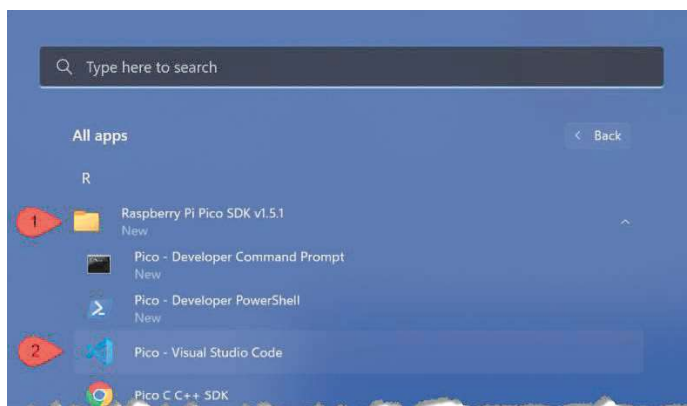
- 1) Press the Windows key (⊞) and type **Pico –** in the search bar. The application shown below will be offered, click on **Open**:



- 2) Press the Windows key (⊞) and click **All apps**:

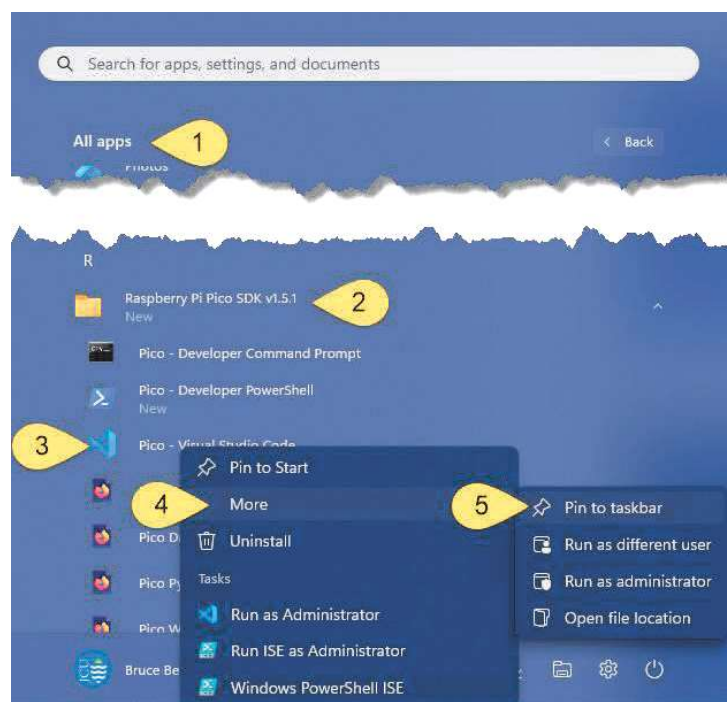


Then scroll down until you can see **Raspberry Pi Pico SDK v1.5.1**. Click on this to expand it, and click on **Pico – Visual Studio Code**:



Making it easier

Tip: Before you click on the Pico – Visual Studio Code menu item, right-click on the icon and then select **More > Pin to taskbar**.



Now you will be able to start the application from the task bar whenever you need it.

Checkpoint 3



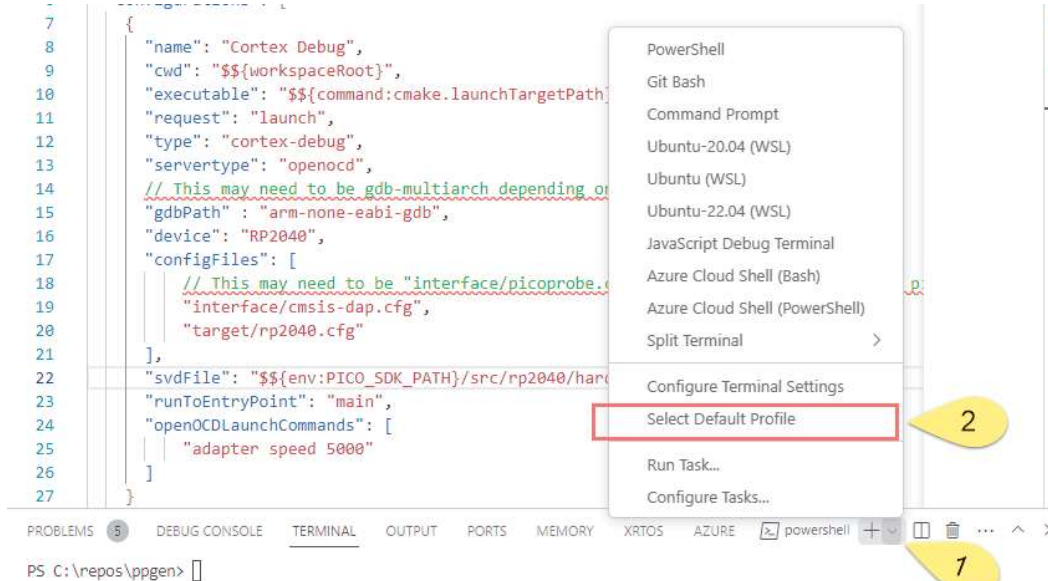
1. Is there an icon on the Task Bar that looks like this?
2. If you float the mouse over the icon, does it show the text Pico – Visual Studio Code?
3. If you click on it, does Visual Studio Code start up?



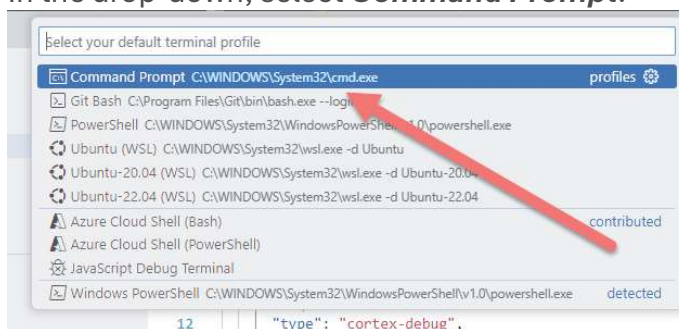
Selecting the default VS Code Terminal

Once you have started VS Code, change the default Terminal window to be the CMD terminal instead of Powershell or Git Bash. You can set this terminal as the default with the following steps:

- 1) Start Pico - Visual Studio Code (as described above).
- 2) In VS Code use the menu command **Terminal > New Terminal**. Note that this opens a PowerShell window, which is probably not what you want.
- 3) Click the down-arrow next to the + on the Terminal window toolbar.



- 4) Click **Select Default Profile**.
- 5) In the drop-down, select **Command Prompt**.



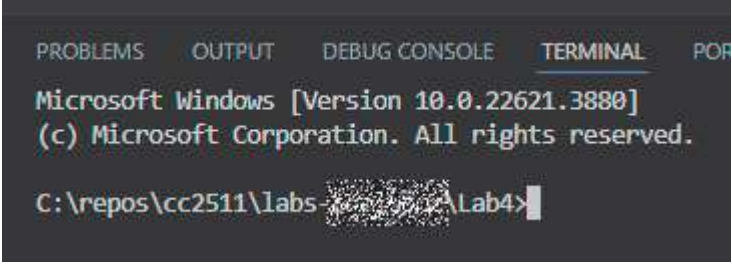
- 6) You can close the PowerShell window by clicking the trash icon on the Terminal window toolbar.
- 7) Next time you open a terminal window it will be a standard Command window.

Checkpoint 4



1. If VS Code is not open, start it, using the **Pico – Visual Studio Code** icon on the Taskbar.
2. Use the View > Terminal menu (or Ctrl+`) to open a new Terminal window. Does it look something like the picture below, with a Windows version in the

top line and no **PS** at the start of the prompt?



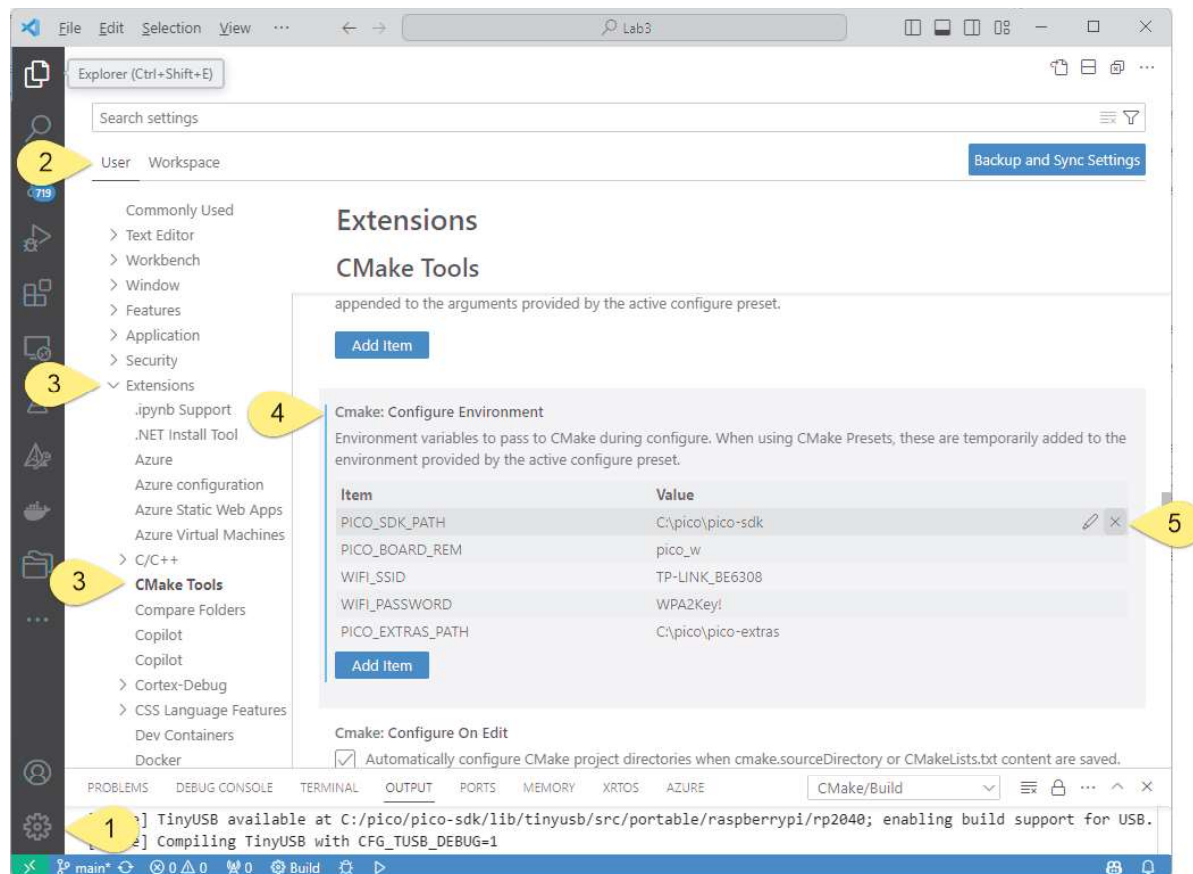
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
Microsoft Windows [Version 10.0.22621.3880]  
(c) Microsoft Corporation. All rights reserved.  
C:\repos\cc2511\labs-2511\Lab4>
```


Troubleshooting

Previous installation

If you have had a previous installation of the Pico SDK on your computer you may encounter problems while building and/or debugging within VS Code.

Follow these steps to fix the build process:



1. Open the Settings panel - either:
 - a. Click on the gear symbol at the bottom-left of VS Code and then click **Settings**; or
 - b. Press **Ctrl+,** (comma)
2. Select **User** at the top of the panel.
3. Select **Extensions > CMake Tools** in the list.
4. Locate **Cmake: Configure Environment**
5. If PICO_SDK_PATH is present, delete it.
6. Locate **Cmake: Cmake Path**, and set it to cmake.

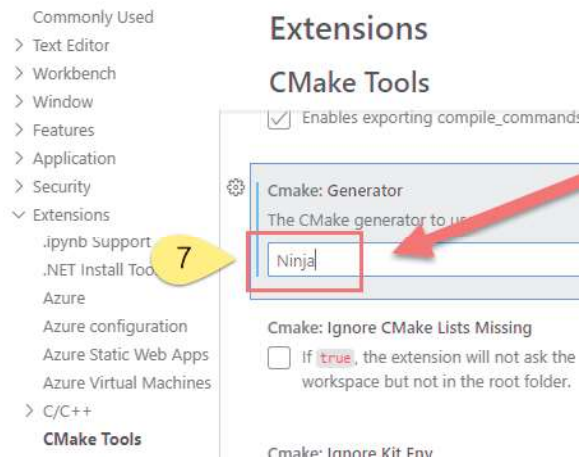
Cmake: Cmake Path

Name/path of the CMake executable to use.

cmake

6

7. Locate **Cmake: Generator**, and set it to Ninja.



Follow these steps to fix the debugging process:

1. Open the Settings panel.
2. Select **User** at the top of the panel.
3. Select **Extensions > Cortex-Debug** in the list.
4. Locate **Cmake: Openocd Path** and click **Edit in settings.json**.
5. Delete the line that starts with “cortex-debug.openocdPath”

```
"cortex-debug.openocdPath.windows": "${env:PICO_SDK_PATH}\\..\\openocd_picoprobe\\openocd.exe",
"workbench.startupEditor": "none",
"cortex-debug.openocdPath": "${env:PICO_SDK_PATH}\\..\\openocd_picoprobe\\openocd.exe",
"editor.tabSize": 2,
"cortex-debug.armToolchainPath": "",
"workbench.colorTheme": "Default Light+",
```

Delete this

6. Ensure that you have copied the latest version of **debugprobe_on_pico.uf2** (provided along with this document) to your debugging Pico device.