# Install Pico SDK in Windows 10x64 June 2022

Largely based on RP2040 Development Setup on Windows https://len42.github.io/rp2040-dev-setup.html

1. Make two folders C:\Dev and C:\Pico

2. Install gcc-arm-11.2-2022.02-mingw-w64-i686-arm-none-eabi.exe from: https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/downloads to C:\Dev\ArmGNUToolchain - add path to environment variable during install.
https://developer.arm.com/-/media/Files/downloads/gnu/11.2-2022.02/binrel/gcc-arm-11.2-2022.02-mingw-w64-i686-arm-none-eabi.exe
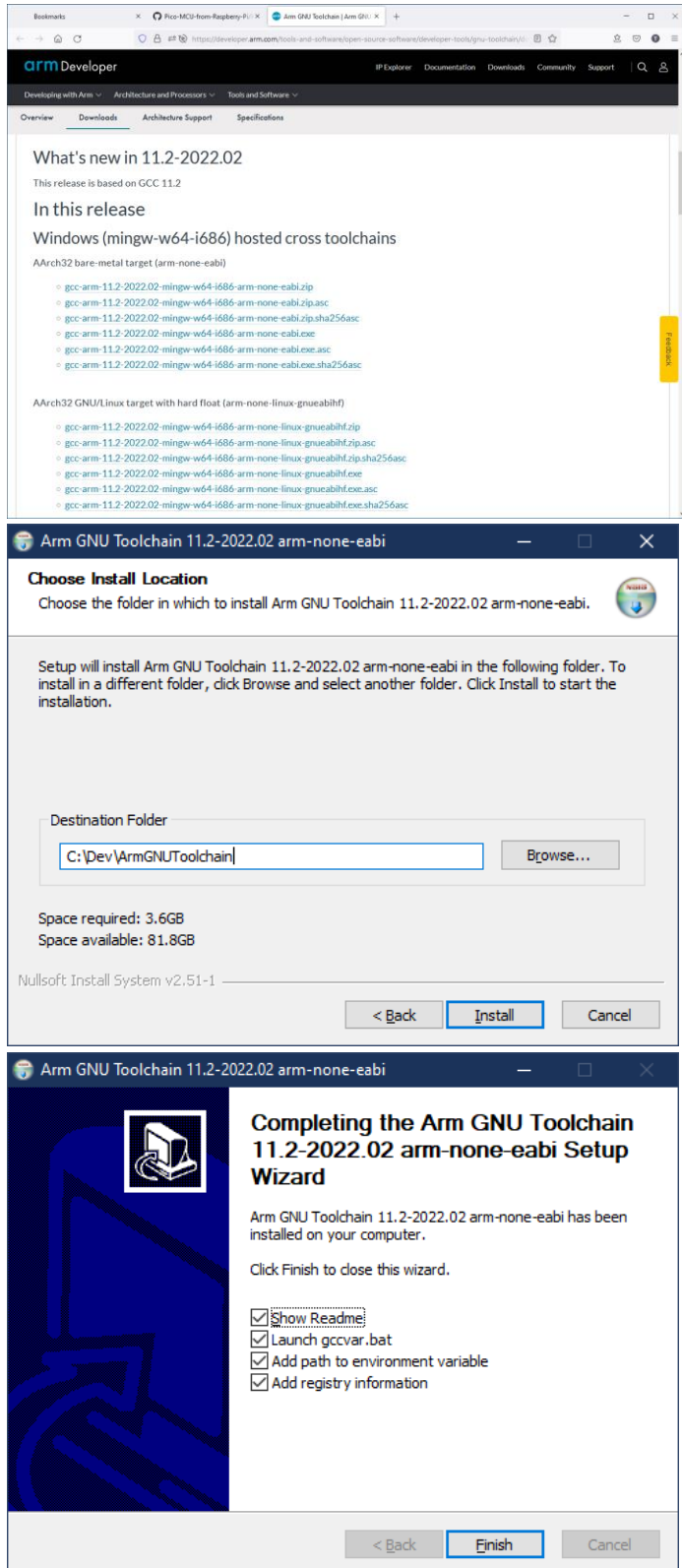
3. Install cmake-3.23.2-windows-x86_64.msi from https://cmake.org/download/ to C:\Dev\CMake\ - add Cmake to the system PATH for all users.

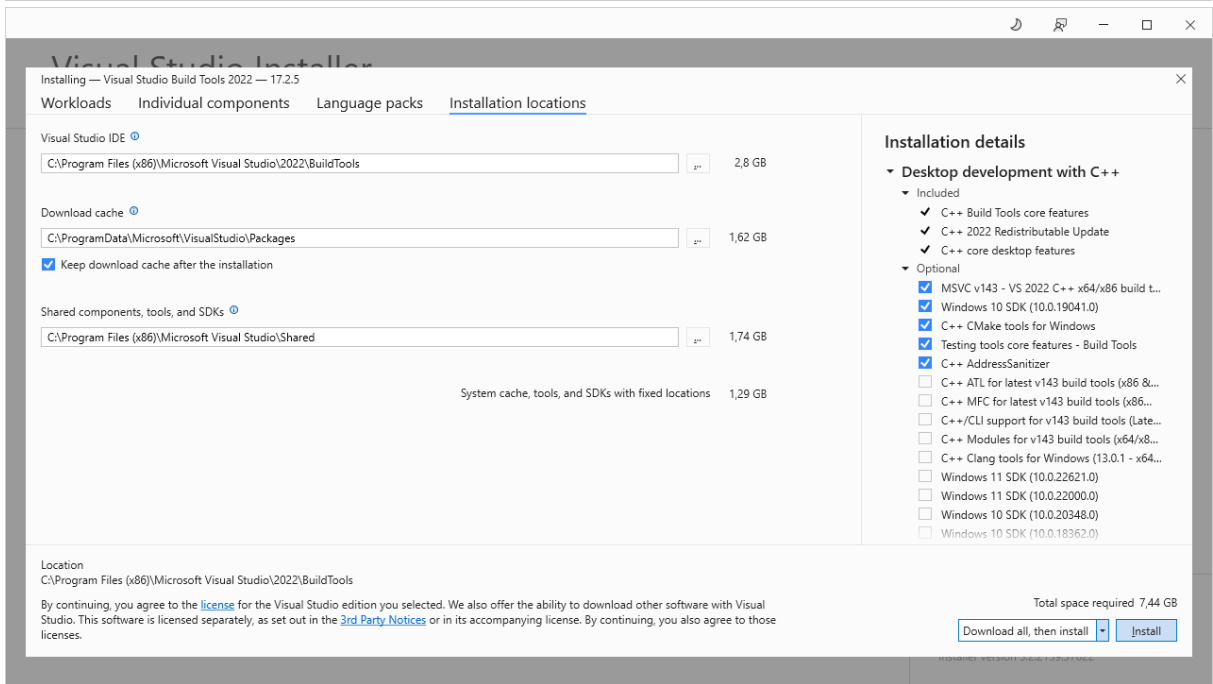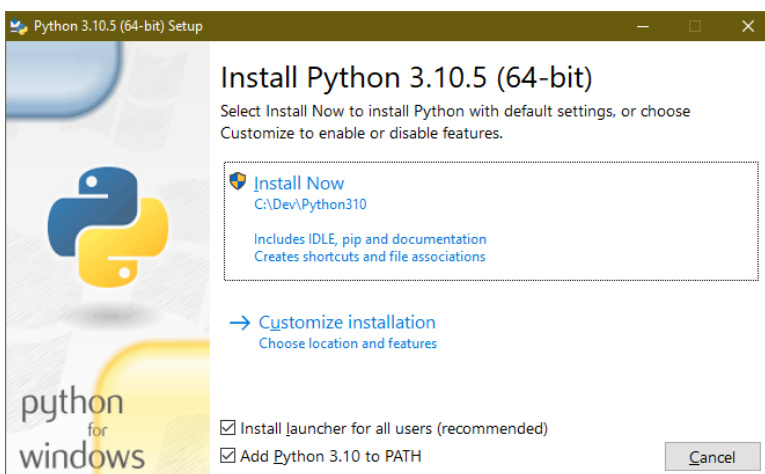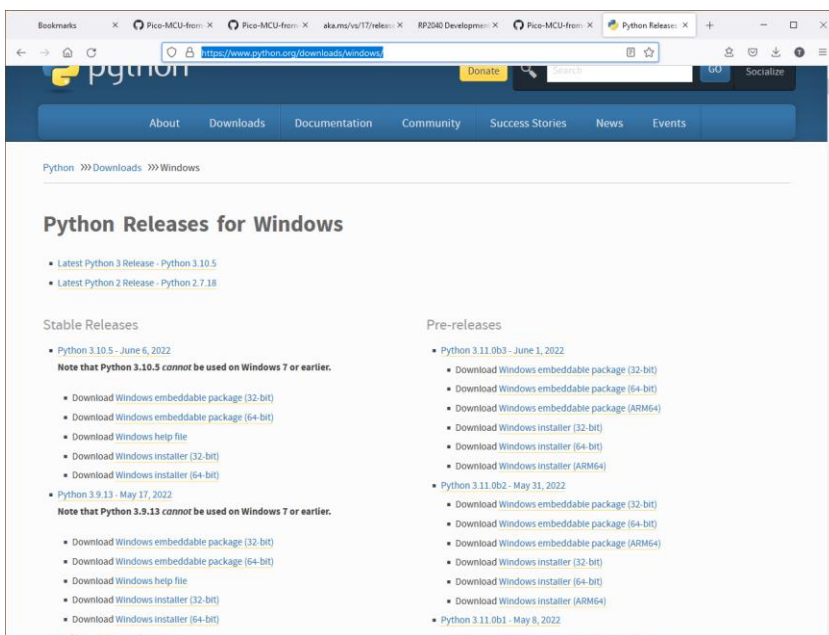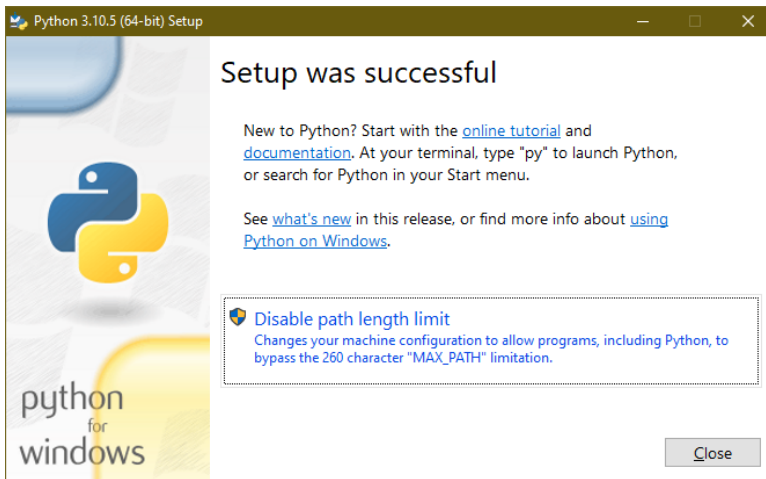https://github.com/Kitware/CMake/releases/download/v3.23.2/cmake-3.23.2-windows-x86_64.msi
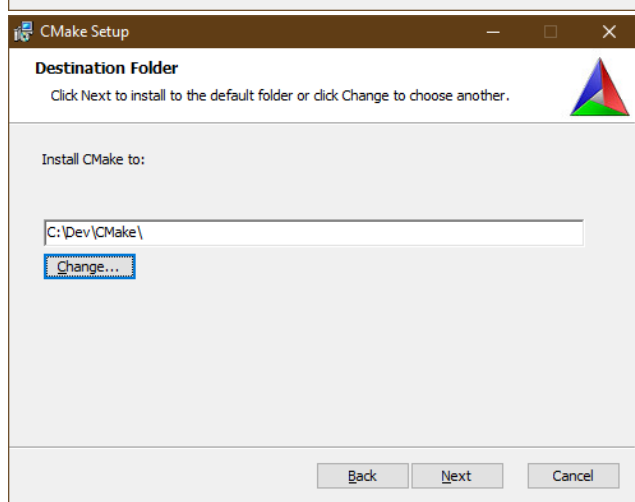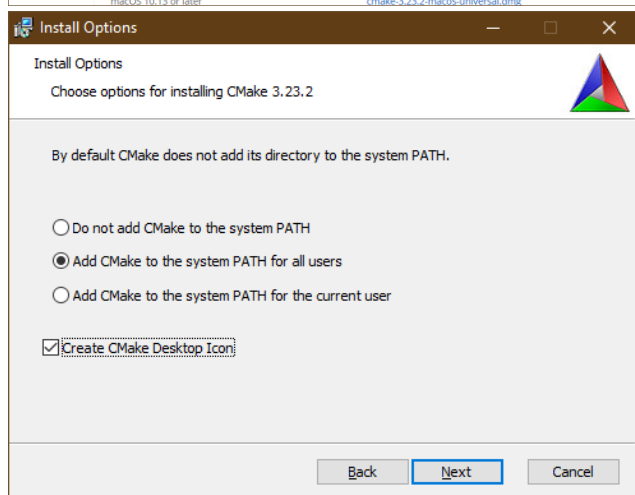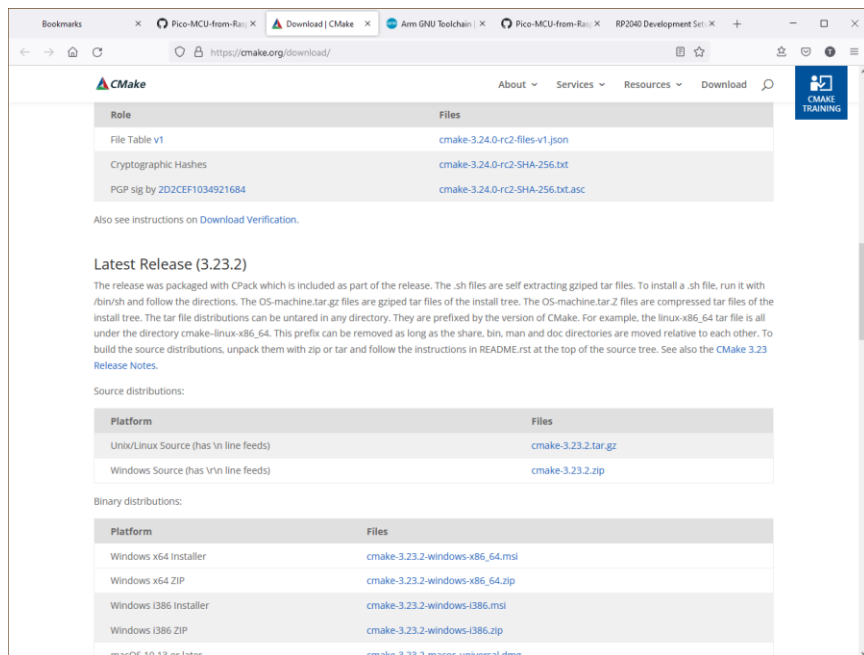
4. Install vs_BuildTools.exe from https://aka.ms/vs/17/release/vs_BuildTools.exe to the default folder - select C++ development tools. It was a 1.62 GB download.

Bookmarks | Pico-MCU-from-Raspberry | Pico-MCU-from-Raspberry | aka.ms/vs/17/release/vs_BuildTo | RP2040 Development Setup on V | +

https://aka.ms/vs/17/release/vs_BuildTools.exe

vs_BuildTools.exe
Completed — 1,6 MB

Show all downloads

Visual Studio Installer

Installing — Visual Studio Build Tools 2022 — 17.2.5

Workloads   Individual components   Language packs   Installation locations

ⓘ Need help choosing what to install? More info ✕

Desktop & Mobile (4)

☑ **Desktop development with C++**
Build modern C++ apps for Windows using tools of your choice, including MSVC, Clang, CMake, or MSBuild.

☐ **Mobile Development with .NET**
Tools for building cross-platform applications for iOS, Android and Windows using C# and F#.

☐ **.NET desktop build tools**
Tools for building WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.

☐ **Universal Windows Platform build tools**
Provides the tools required to build Universal Windows Platform applications.

Web & Cloud (4)

☐ **Web development build tools**
MSBuild tasks and targets for building web applications.

☐ **Azure development build tools**
MSBuild tasks and targets for building Azure applications.

Location
C:\Program Files (x86)\Microsoft Visual Studio\2022\BuildTools   Change...

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Installation details

▸ MSBuild Tools

▾ Desktop development with C++
  ▾ Included
    ✔ C++ Build Tools core features
    ✔ C++ 2022 Redistributable Update
    ✔ C++ core desktop features
  ▾ Optional
    ☑ MSVC v143 - VS 2022 C++ x64/x86 build t...
    ☑ Windows 10 SDK (10.0.19041.0)
    ☑ C++ CMake tools for Windows
    ☑ Testing tools core features - Build Tools
    ☑ C++ AddressSanitizer
    ☐ C++ ATL for latest v143 build tools (x86 &...
    ☐ C++ MFC for latest v143 build tools (x86...
    ☐ C++/CLI support for v143 build tools (Late...
    ☐ C++ Modules for v143 build tools (x64/x8...
    ☐ C++ Clang tools for Windows (13.0.1 - x64...
    ☐ Windows 11 SDK (10.0.22621.0)
    ☐ Windows 11 SDK (10.0.22000.0)

Total space required 7,48 GB

Install while downloading ▾   Install

Visual Studio Installer

Installing — Visual Studio Build Tools 2022 — 17.2.5

Workloads   Individual components   Language packs   Installation locations

Visual Studio IDE ⓘ
C:\Program Files (x86)\Microsoft Visual Studio\2022\BuildTools | 2,8 GB

Download cache ⓘ
C:\ProgramData\Microsoft\VisualStudio\Packages | 1,62 GB
☑ Keep download cache after the installation

Shared components, tools, and SDKs ⓘ
C:\Program Files (x86)\Microsoft Visual Studio\Shared | 1,74 GB

System cache, tools, and SDKs with fixed locations   1,29 GB

Location
C:\Program Files (x86)\Microsoft Visual Studio\2022\BuildTools

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Installation details

▾ Desktop development with C++
  ▾ Included
    ✔ C++ Build Tools core features
    ✔ C++ 2022 Redistributable Update
    ✔ C++ core desktop features
  ▾ Optional
    ☑ MSVC v143 - VS 2022 C++ x64/x86 build t...
    ☑ Windows 10 SDK (10.0.19041.0)
    ☑ C++ CMake tools for Windows
    ☑ Testing tools core features - Build Tools
    ☑ C++ AddressSanitizer
    ☐ C++ ATL for latest v143 build tools (x86 &...
    ☐ C++ MFC for latest v143 build tools (x86...
    ☐ C++/CLI support for v143 build tools (Late...
    ☐ C++ Modules for v143 build tools (x64/x8...
    ☐ C++ Clang tools for Windows (13.0.1 - x64...
    ☐ Windows 11 SDK (10.0.22621.0)
    ☐ Windows 11 SDK (10.0.22000.0)
    ☐ Windows 10 SDK (10.0.20348.0)
    ☐ Windows 10 SDK (10.0.18362.0)

Total space required 7,44 GB

Download all, then install ▾   Install

5. Install python-3.10.5-amd64.exe from https://www.python.org/downloads/windows/ to C:\Dev\Python310 - select Add Python to PATH and also select to remove the max path length.
https://www.python.org/ftp/python/3.10.5/python-3.10.5-amd64.exe

6. Install Git-2.36.1-64-bit.exe from https://git-scm.com/download/win to C:\Dev\Git - follow the instructions from https://len42.github.io/rp2040-dev-setup.html .

   Destination Location: Default (or not)
   Select Components: Default
   Default editor: Select one you like.
   Name of the initial branch: Let Git decide
   PATH environment: Git from the command line and also from 3rd-party software
   SSH executable: Use bundled OpenSSH
   HTTPS transport backend: Use the OpenSSL library
   Line ending conversion: Checkout as-is, commit as-is
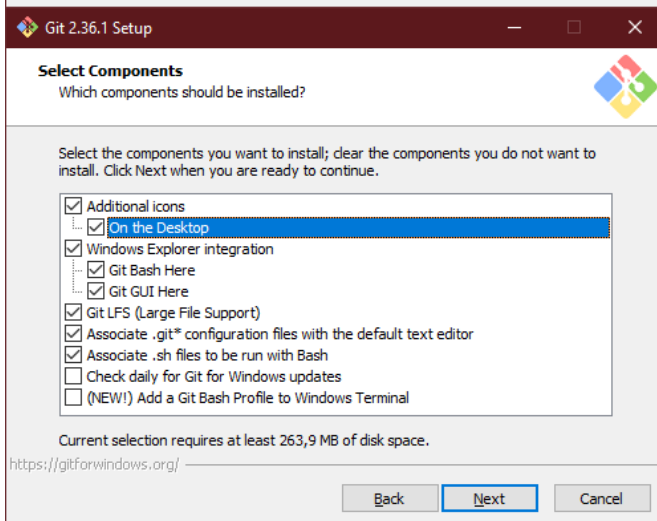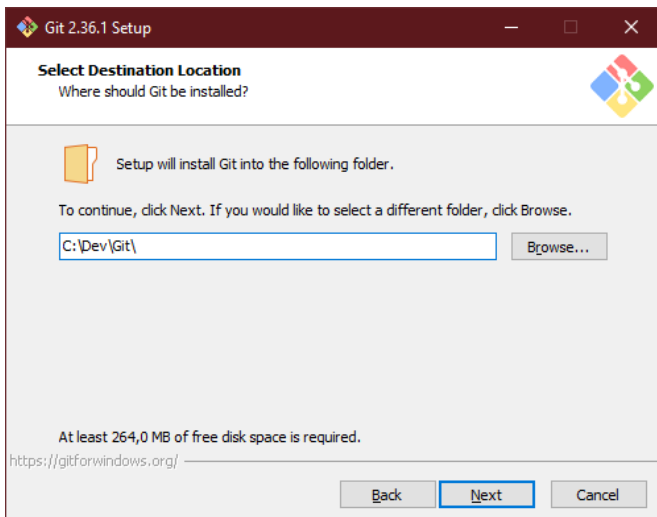   Terminal emulator for Git Bash: Select either option
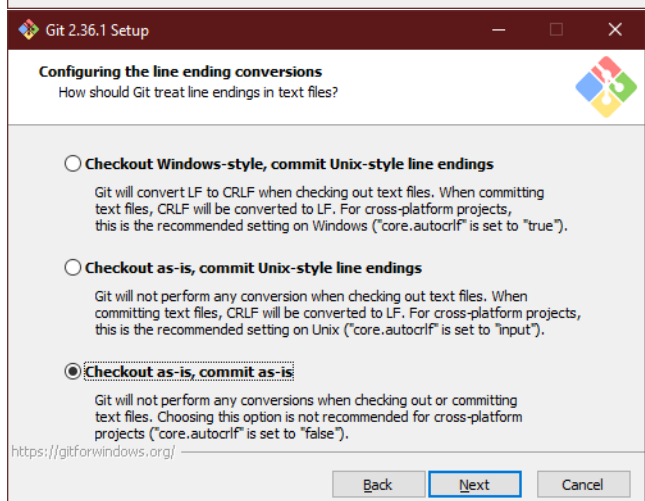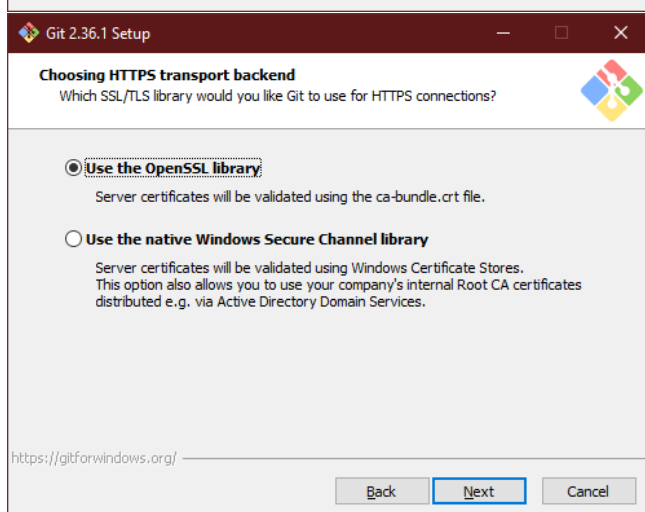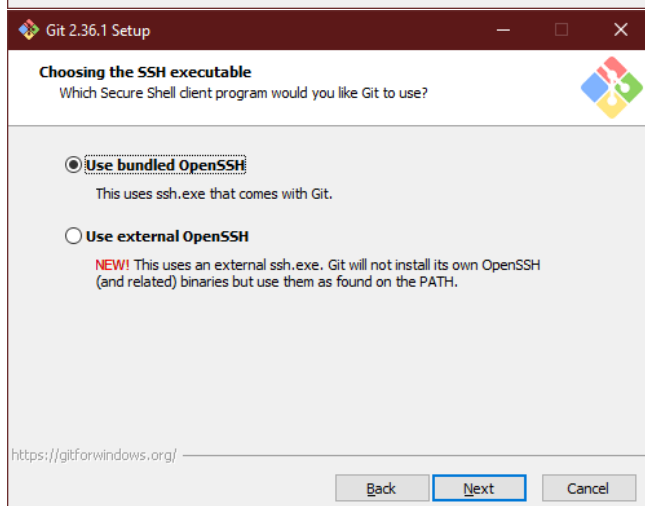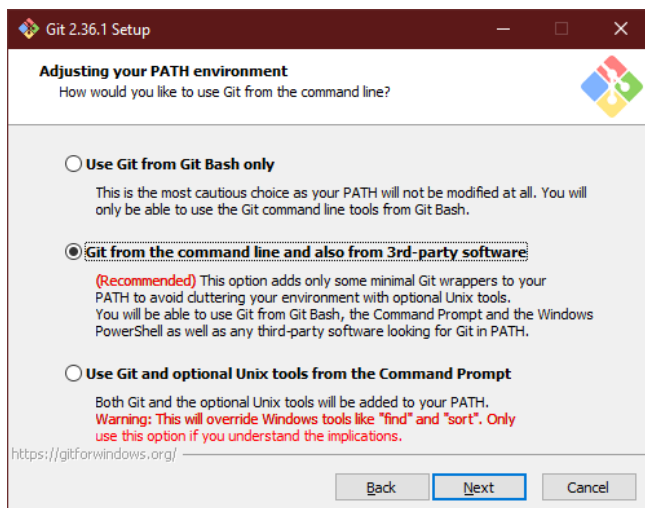   Default behavior of "git pull": Default (f-f or merge)
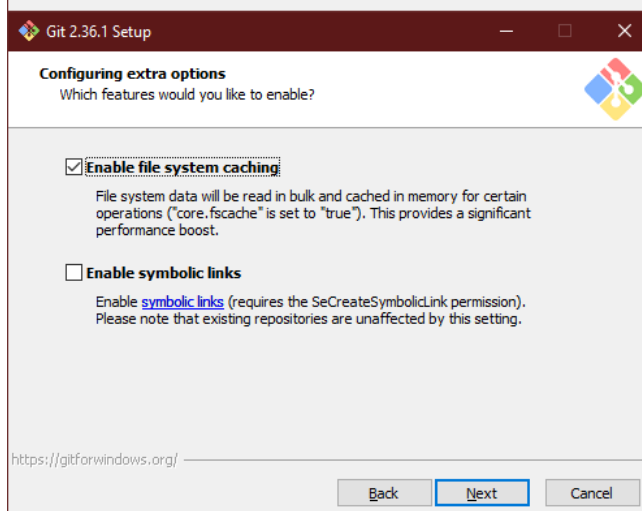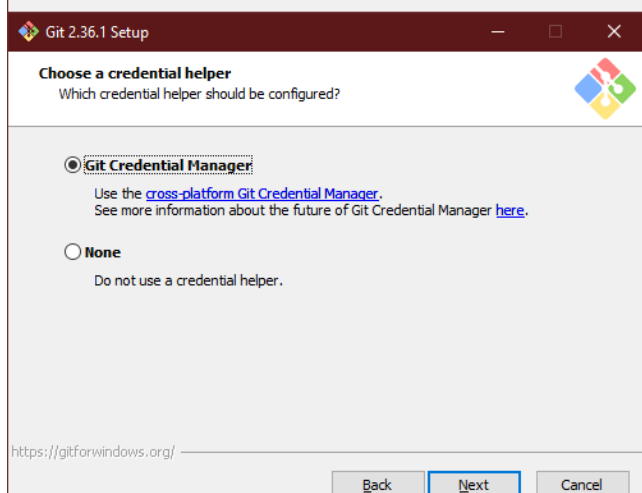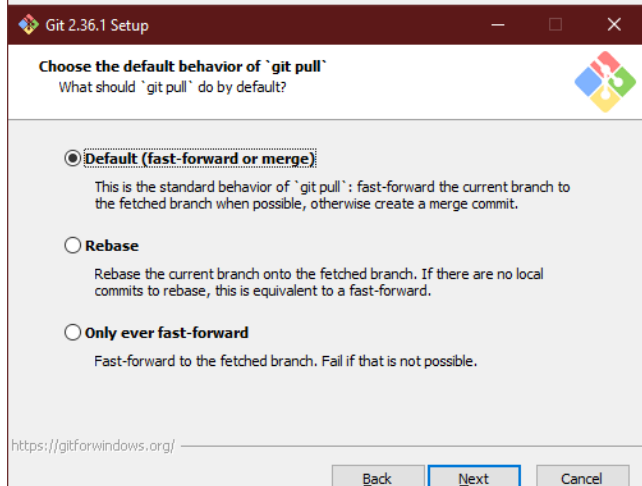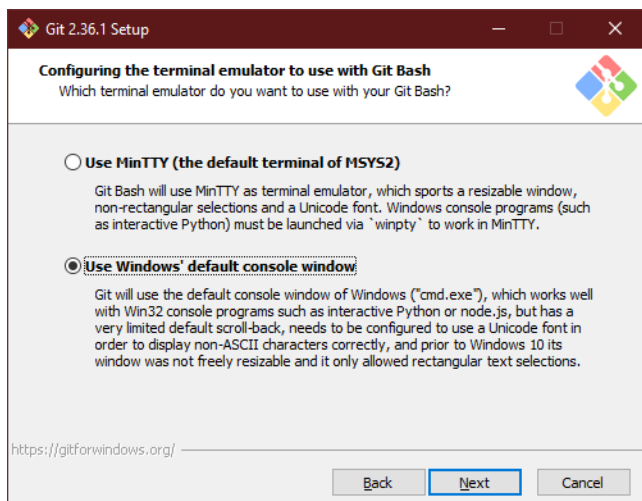   Credential helper: Default (Git Credential Manager Core)
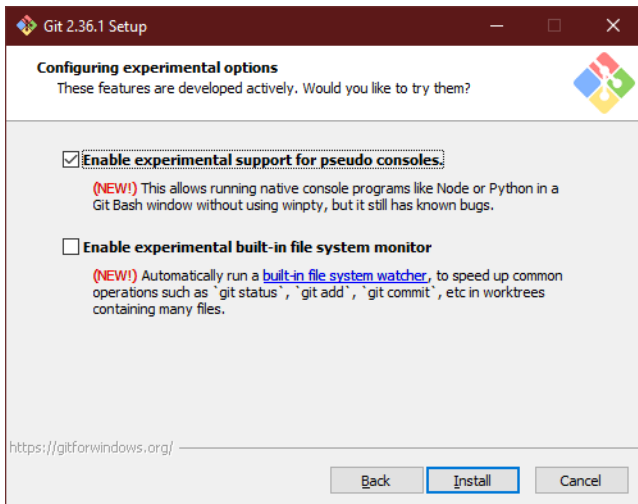   Extra options: Default (Enable file system caching on, Enable symbolic links off)
   Experimental options: Select "Enable experimental support for pseudo consoles"

Git 2.36.1 Setup

Select Destination Location
Where should Git be installed?

Setup will install Git into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\Dev\Git\   [Browse...]

At least 264,0 MB of free disk space is required.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Select Components
Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- ☑ Additional icons
  - ☑ On the Desktop
- ☑ Windows Explorer integration
  - ☑ Git Bash Here
  - ☑ Git GUI Here
- ☑ Git LFS (Large File Support)
- ☑ Associate .git* configuration files with the default text editor
- ☑ Associate .sh files to be run with Bash
- ☐ Check daily for Git for Windows updates
- ☐ (NEW!) Add a Git Bash Profile to Windows Terminal

Current selection requires at least 263,9 MB of disk space.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Choosing the default editor used by Git
Which editor would you like Git to use?

Use the Nano editor by default ▼

GNU nano is a small and friendly text editor running in the console window.

This is the recommended option for end users if no GUI editors are installed.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Adjusting the name of the initial branch in new repositories
What would you like Git to name the initial branch after "git init"?

◉ **Let Git decide**

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.

◯ **Override the default branch name for new repositories**

NEW! Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:

main

This setting does not affect existing repositories.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Adjusting your PATH environment
How would you like to use Git from the command line?

○ **Use Git from Git Bash only**

This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

◉ **Git from the command line and also from 3rd-party software**

(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

○ **Use Git and optional Unix tools from the Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Choosing the SSH executable
Which Secure Shell client program would you like Git to use?

◉ **Use bundled OpenSSH**

This uses ssh.exe that comes with Git.

○ **Use external OpenSSH**

NEW! This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?

◉ **Use the OpenSSL library**

Server certificates will be validated using the ca-bundle.crt file.

○ **Use the native Windows Secure Channel library**

Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Configuring the line ending conversions
How should Git treat line endings in text files?

○ **Checkout Windows-style, commit Unix-style line endings**

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

○ **Checkout as-is, commit Unix-style line endings**

Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

◉ **Checkout as-is, commit as-is**

Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?

○ **Use MinTTY (the default terminal of MSYS2)**

Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.

◉ **Use Windows' default console window**

Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Choose the default behavior of `git pull`
What should `git pull` do by default?

◉ **Default (fast-forward or merge)**

This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

○ **Rebase**

Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

○ **Only ever fast-forward**

Fast-forward to the fetched branch. Fail if that is not possible.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Choose a credential helper
Which credential helper should be configured?

◉ **Git Credential Manager**

Use the cross-platform Git Credential Manager.
See more information about the future of Git Credential Manager here.

○ **None**

Do not use a credential helper.

https://gitforwindows.org/

[Back] [Next] [Cancel]

Git 2.36.1 Setup

Configuring extra options
Which features would you like to enable?

☑ **Enable file system caching**

File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

☐ **Enable symbolic links**

Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.

https://gitforwindows.org/

[Back] [Next] [Cancel]

7. Use the windows admin cmd prompt to install the Pico SDK

cd c:\Pico
c:\Pico>git clone -b master https://github.com/raspberrypi/pico-sdk.git
c:\Pico>git clone -b master https://github.com/raspberrypi/pico-examples.git
c:\Pico>git clone -b master https://github.com/raspberrypi/pico-extras.git
c:\Pico>git clone -b master https://github.com/raspberrypi/pico-playground.git

c:\Pico>cd pico-extras
c:\Pico\pico-extras>git submodule update - -init
c:\Pico\pico-extras>cd ..
c:\Pico>cd pico-sdk
c:\Pico\pico-sdk>git submodule update - -init

8. Then run a build of all the pico-examples:
c:\Pico\setx PICO_SDK_PATH "C:\Pico\pico-sdk"

```
c:\Pico>cd pico-extras

c:\Pico\pico-extras>git submodule update --init
Submodule 'lwip' (https://git.savannah.nongnu.org/git/lwip.git) registered for path 'lib/lwip'
Cloning into 'C:/Pico/pico-extras/lib/lwip'...
Submodule path 'lib/lwip': checked out 'c385f31076b27efb8ee37f00cb5568783a58f299'

c:\Pico\pico-extras>cd ..

c:\Pico>cd pico-sdk

c:\Pico\pico-sdk>git submodule update --init
Submodule 'tinyusb' (https://github.com/hathach/tinyusb.git) registered for path 'lib/tinyusb'
Cloning into 'C:/Pico/pico-sdk/lib/tinyusb'...
Submodule path 'lib/tinyusb': checked out '4bfab30c02279a0530e1a56f4a7c539f2d35a293'

c:\Pico\pico-sdk>setx PICO_SDK_PATH "C:\Pico\pico-sdk"

SUCCESS: Specified value was saved.

c:\Pico\pico-sdk>_
```

9. Close the cmd window and run the VS Developer Command Prompt as admin.
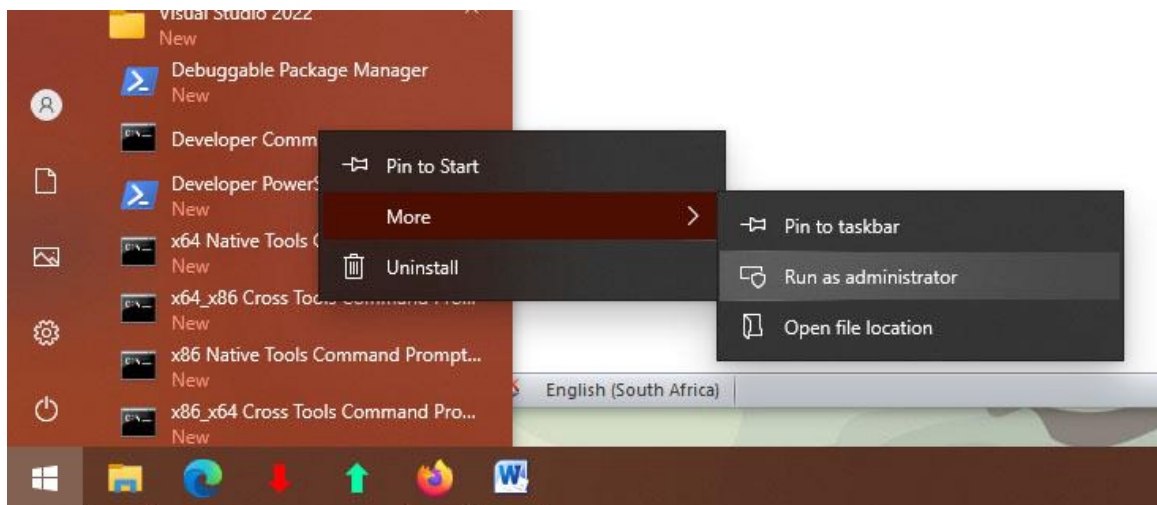
c:\Windows\System32>cd c:\Pico
c:\Pico>cd pico-examples\
c:\Pico\pico-examples>mkdir build
c:\Pico\pico-examples>cd build
c:\Pico\pico-examples\build>cmake -G "NMake Makefiles" ..
c:\Pico\pico-examples\build>nmake

Administrator: Developer Command Prompt for VS 2022

```
*****************************************************************
** Visual Studio 2022 Developer Command Prompt v17.2.5
** Copyright (c) 2022 Microsoft Corporation
*****************************************************************
[ERROR:team_explorer.bat] Directory not found : "C:\Program Files (x86)\Microsoft Visual Studio\2022\BuildTools\Common7\IDE\CommonExtensions\Microsoft\TeamFoundation\Team Explorer"

C:\Windows\System32>cd c:\Pico

c:\Pico>cd pico-examples

c:\Pico\pico-examples>mkdir build

c:\Pico\pico-examples>cd build

c:\Pico\pico-examples\build>cmake -G "NMake Makefiles" ..
Using PICO_SDK_PATH from environment ('C:\Pico\pico-sdk')
PICO_SDK_PATH is C:/Pico/pico-sdk
Defaulting PICO_PLATFORM to rp2040 since not specified.
Defaulting PICO platform compiler to pico_arm_gcc since not specified.
-- Defaulting build type to 'Release' since not specified.
PICO compiler is pico_arm_gcc
-- The C compiler identification is GNU 11.2.1
-- The CXX compiler identification is GNU 11.2.1
-- The ASM compiler identification is GNU
-- Found assembler: C:/Dev/ArmGNUToolchain/bin/arm-none-eabi-gcc.exe
Build type is Release
Defaulting PICO target board to pico since not specified.
Using board configuration from C:/Pico/pico-sdk/src/boards/include/boards/pico.h
-- Found Python3: C:/Dev/Python310/python.exe (found version "3.10.5") found components: Interpreter
TinyUSB available at C:/Pico/pico-sdk/lib/tinyusb/src/portable/raspberrypi/rp2040; enabling build support for USB.
-- Configuring done
-- Generating done
-- Build files have been written to: C:/Pico/pico-examples/build

c:\Pico\pico-examples\build>
```

Administrator: Developer Command Prompt for VS 2022 - nmake

```
c:\Pico\pico-examples>cd build

c:\Pico\pico-examples\build>cmake -G "NMake Makefiles" ..
Using PICO_SDK_PATH from environment ('C:\Pico\pico-sdk')
PICO_SDK_PATH is C:/Pico/pico-sdk
Defaulting PICO_PLATFORM to rp2040 since not specified.
Defaulting PICO platform compiler to pico_arm_gcc since not specified.
-- Defaulting build type to 'Release' since not specified.
PICO compiler is pico_arm_gcc
-- The C compiler identification is GNU 11.2.1
-- The CXX compiler identification is GNU 11.2.1
-- The ASM compiler identification is GNU
-- Found assembler: C:/Dev/ArmGNUToolchain/bin/arm-none-eabi-gcc.exe
Build type is Release
Defaulting PICO target board to pico since not specified.
Using board configuration from C:/Pico/pico-sdk/src/boards/include/boards/pico.h
-- Found Python3: C:/Dev/Python310/python.exe (found version "3.10.5") found components: Interpreter
TinyUSB available at C:/Pico/pico-sdk/lib/tinyusb/src/portable/raspberrypi/rp2040; enabling build support for USB.
-- Configuring done
-- Generating done
-- Build files have been written to: C:/Pico/pico-examples/build

c:\Pico\pico-examples\build>nmake

Microsoft (R) Program Maintenance Utility Version 14.32.31332.0
Copyright (C) Microsoft Corporation.  All rights reserved.

Scanning dependencies of target bs2_default
[  0%] Building ASM object pico-sdk/src/rp2_common/boot_stage2/CMakeFiles/bs2_default.dir/compile_time_choice.S.obj
[  0%] Linking ASM executable bs2_default.elf
[  0%] Built target bs2_default
[  0%] Generating bs2_default.bin
[  0%] Generating bs2_default_padded_checksummed.S
[  0%] Built target bs2_default_padded_checksummed_asm
[  0%] Creating directories for 'ELF2UF2Build'
[  0%] No download step for 'ELF2UF2Build'
[  0%] No update step for 'ELF2UF2Build'
[  0%] No patch step for 'ELF2UF2Build'
[  0%] Performing configure step for 'ELF2UF2Build'
-- The C compiler identification is MSVC 19.32.31332.0
-- The CXX compiler identification is MSVC 19.32.31332.0
-- Detecting C compiler ABI info
```

```
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_printf/printf
.c.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_bit_ops/bit
_ops_aeabi.S.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_bootrom/bootr
om.c.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_divider/div
ider.S.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_double/doub
le_aeabi.S.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_double/double
_init_rom.c.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_double/double
_math.c.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_double/doub
le_v1_rom_shim.S.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_int64_ops/p
ico_int64_ops_aeabi.S.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_float/float
_aeabi.S.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_float/float_i
nit_rom.c.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_float/float_m
ath.c.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_float/float
_v1_rom_shim.S.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_malloc/pico_m
alloc.c.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_mem_ops/mem
_ops_aeabi.S.obj
[100%] Building ASM object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_standard_li
nk/crt0.S.obj
[100%] Building CXX object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_standard_li
nk/new_delete.cpp.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_standard_link
/binary_info.c.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_stdio/stdio.c
.obj
[100%] Building C object watchdog/hello_watchdog/CMakeFiles/hello_watchdog.dir/C_/Pico/pico-sdk/src/rp2_common/pico_stdio_uart/st
dio_uart.c.obj
[100%] Linking CXX executable hello_watchdog.elf
[100%] Built target hello_watchdog

c:\Pico\pico-examples\build>
```