

# Getting started with USB-RELAY SDK

## EN INSTALL

On windows >= 10:

### Step-1:

Install the PL2303 driver for windows follow the installation guide:  
(USB-RELAYDriverInstallationEN-FR).pdf

### Step-2:

Download and Install Visual Studio Code:

<https://code.visualstudio.com/>

### Step-3:

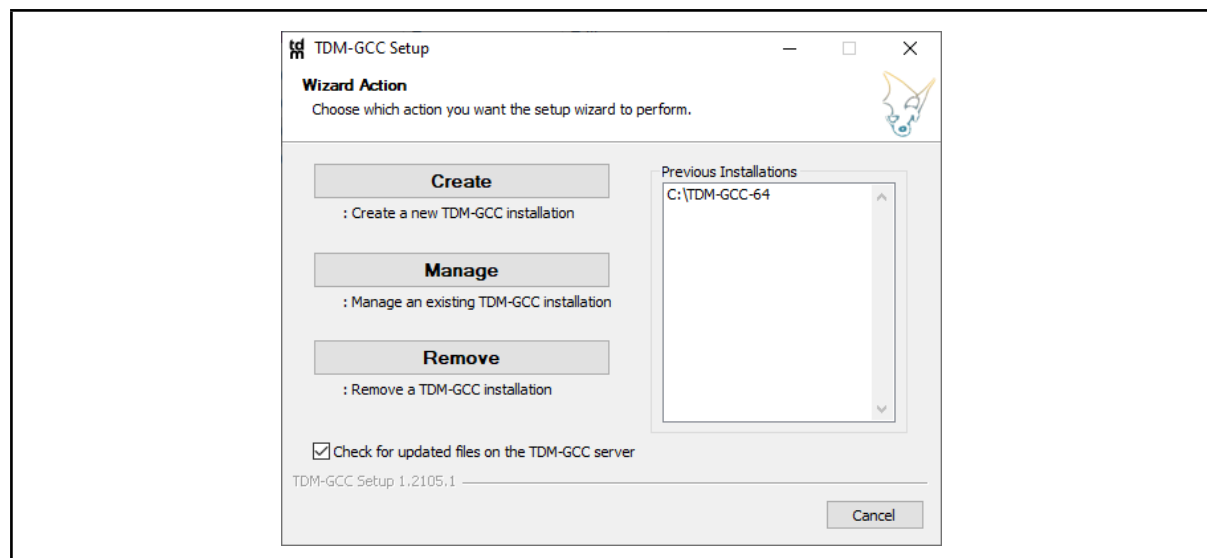
Download and Install TDM-GCC, version should be >= 10.3.0:

<https://jmeubank.github.io/tdm-gcc/download/>

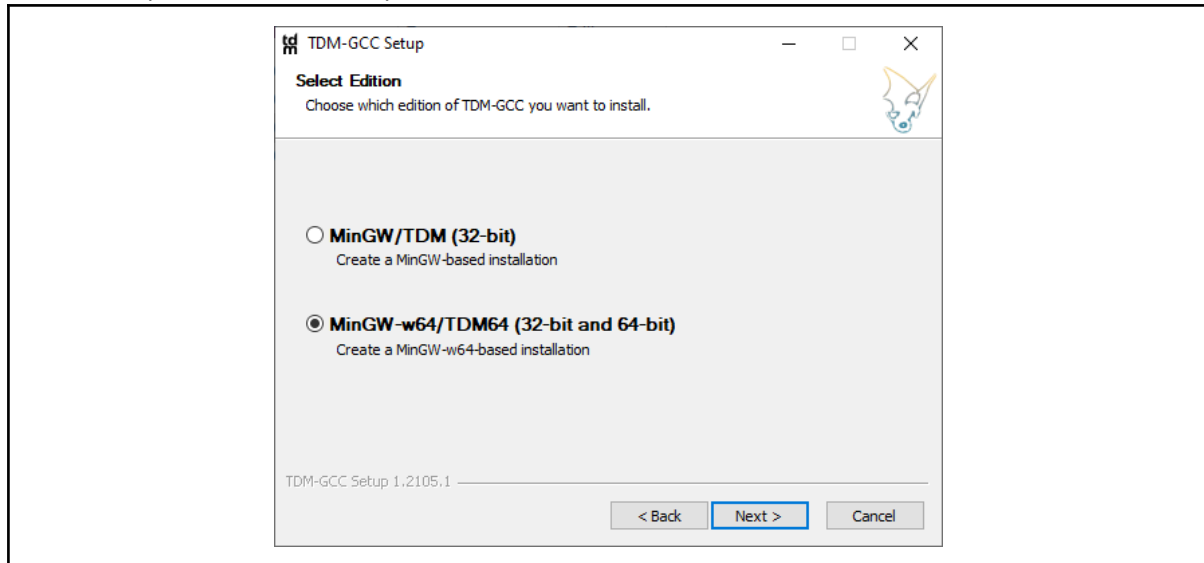
➡**Select** tdm64-gcc with MinGw and all libraries

<b>tdm-gcc-webdl.exe</b>	Minimal online installer. Select the components you want, and it downloads and unpacks them. Either edition, latest release only. (GCC 10.3.0)
<b>tdm64-gcc-10.3.0-2.exe</b>	64+32-bit MinGW-w64 edition. Includes GCC C/C++, GNU binutils, mingw32-make, GDB (64-bit), the MinGW-w64 runtime libraries and tools, and the windows-default-manifest package.
<b>tdm-gcc-10.3.0.exe</b>	32-bit-only MinGW.org edition. Includes GCC C/C++, GNU binutils, mingw32-make, GDB (32-bit), the MinGW.org mingwrt and w32api packages, and the windows-default-manifest package.

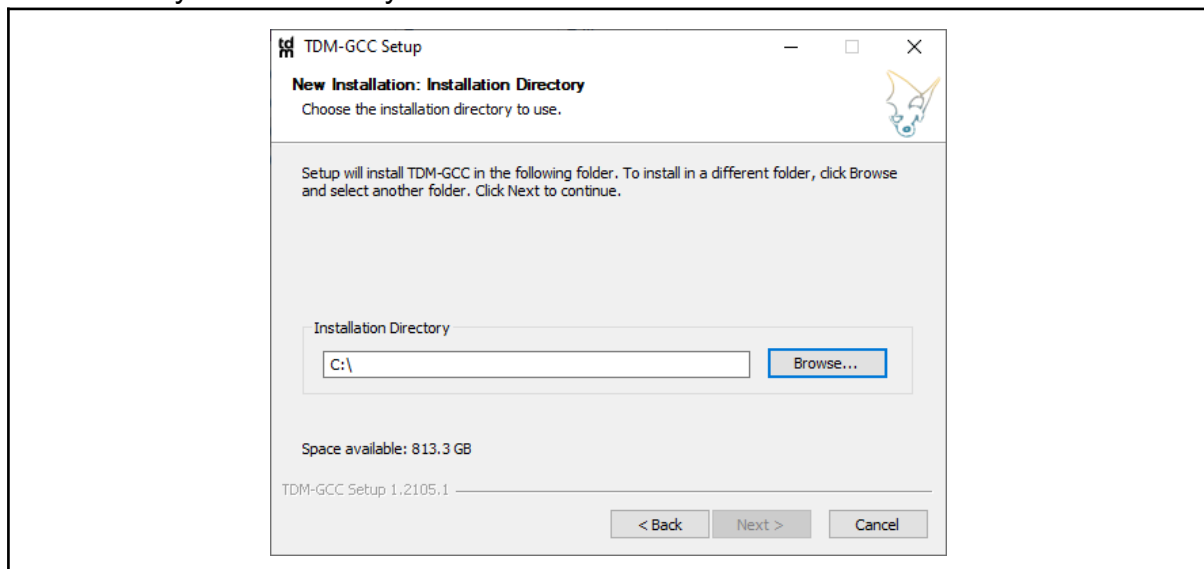
➡**Click on Create** to create a new TDM-GCC install



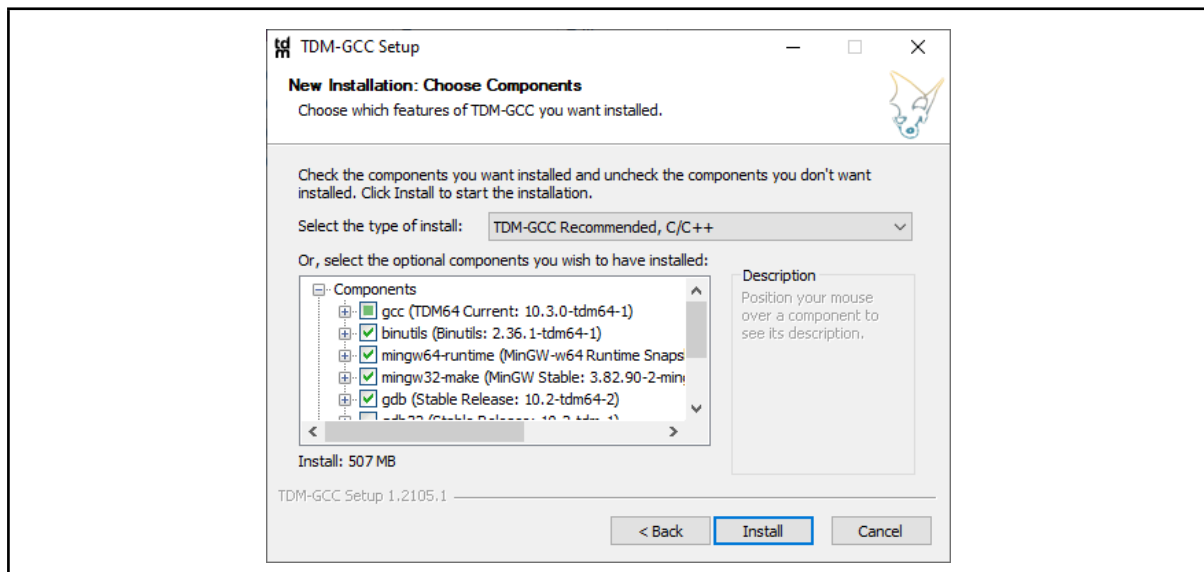
## ➡Select (32-bit and 64-bit) install



## ➡Install in your C directory



## ➡Select recommended installation and Click on Install



#### Step-4:

Download and Install Cmake, version should be  $\geq 3.30.2$ :

<https://cmake.org/download/>

#### ➡Download Windows x64 Installer

Binary distributions:

Platform	Files
Windows x64 Installer:	<a href="#">cmake-3.30.2-windows-x86_64.msi</a>
Windows x64 ZIP	<a href="#">cmake-3.30.2-windows-x86_64.zip</a>
Windows i386 Installer:	<a href="#">cmake-3.30.2-windows-i386.msi</a>
Windows i386 ZIP	<a href="#">cmake-3.30.2-windows-i386.zip</a>
Windows ARM64 Installer:	<a href="#">cmake-3.30.2-windows-arm64.msi</a>
Windows ARM64 ZIP	<a href="#">cmake-3.30.2-windows-arm64.zip</a>
macOS 10.13 or later	<a href="#">cmake-3.30.2-macos-universal.dmg</a>
	<a href="#">cmake-3.30.2-macos-universal.tar.gz</a>
macOS 10.10 or later	<a href="#">cmake-3.30.2-macos10.10-universal.dmg</a>
	<a href="#">cmake-3.30.2-macos10.10-universal.tar.gz</a>
Linux x86_64	<a href="#">cmake-3.30.2-linux-x86_64.sh</a>
	<a href="#">cmake-3.30.2-linux-x86_64.tar.gz</a>
Linux aarch64	<a href="#">cmake-3.30.2-linux-aarch64.sh</a>
	<a href="#">cmake-3.30.2-linux-aarch64.tar.gz</a>

(Note): You can install the GUI version it does not matter

#### Step-5:

Download Ninja Build System

<https://github.com/ninja-build/ninja/releases>

#### ➡Download ninja-win.zip

**v1.12.1** Latest

Bugfixes:

- Screen updates extremely slow on Windows [#2435](#)
- Dry run error if the build directory does not exist [#2431](#)
- New critical path scheduler performance improvements [#2443](#)

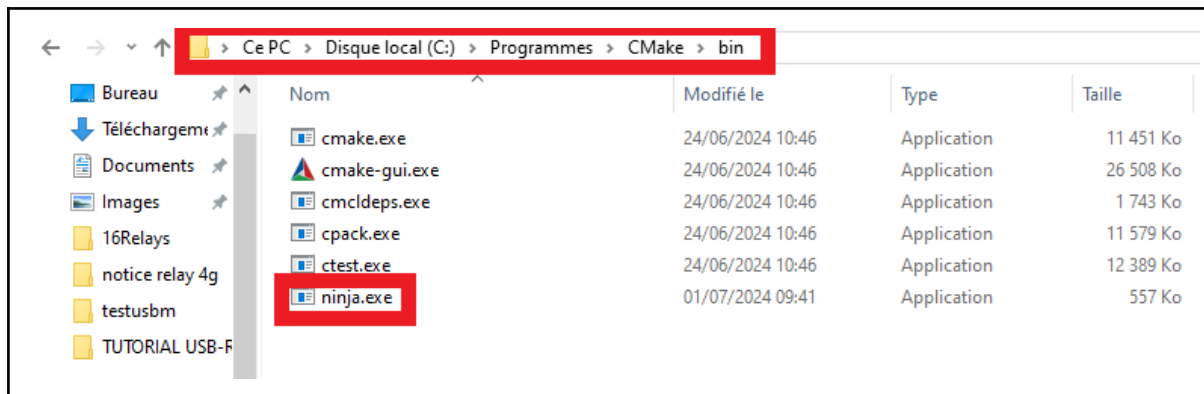
<https://github.com/ninja-build/ninja/milestone/11?closed=1>

**Assets** 7

<a href="#">ninja-linux-aarch64.zip</a>	119 KB	May 11
<a href="#">ninja-linux.zip</a>	120 KB	May 11
<a href="#">ninja-mac.zip</a>	275 KB	May 11
<a href="#">ninja-win.zip</a>	269 KB	May 11
<a href="#">ninja-winarm64.zip</a>	248 KB	May 11
<a href="#">Source code (zip)</a>		May 11
<a href="#">Source code (tar.gz)</a>		May 11

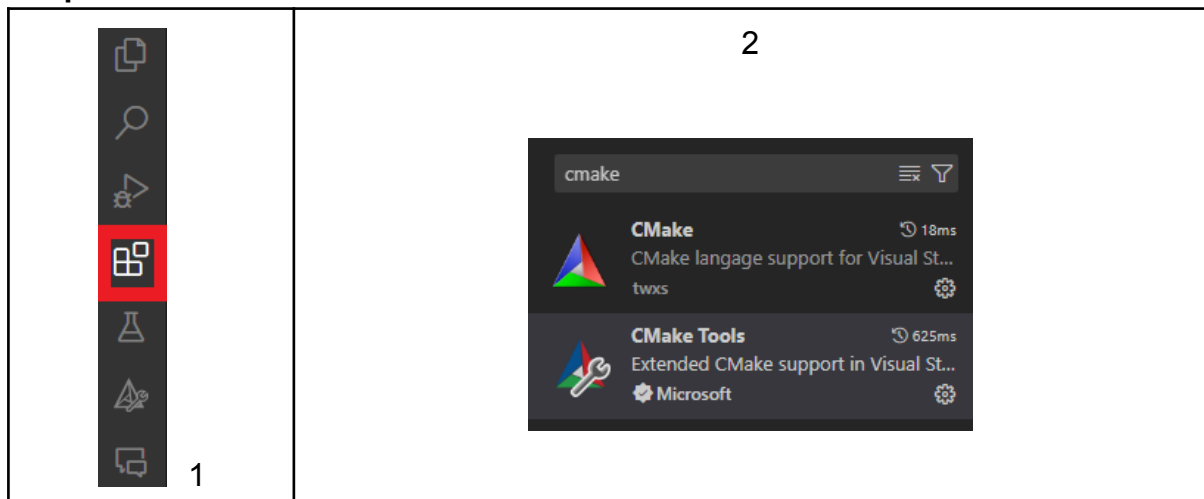
22 4 3 10 31 people reacted

- ➔ **Unpack the archive**, you should obtain a **ninja.exe** file
- ➔ **Open** the your Cmake installation folder and **Paste** **ninja.exe** in the Cmake **bin** folder

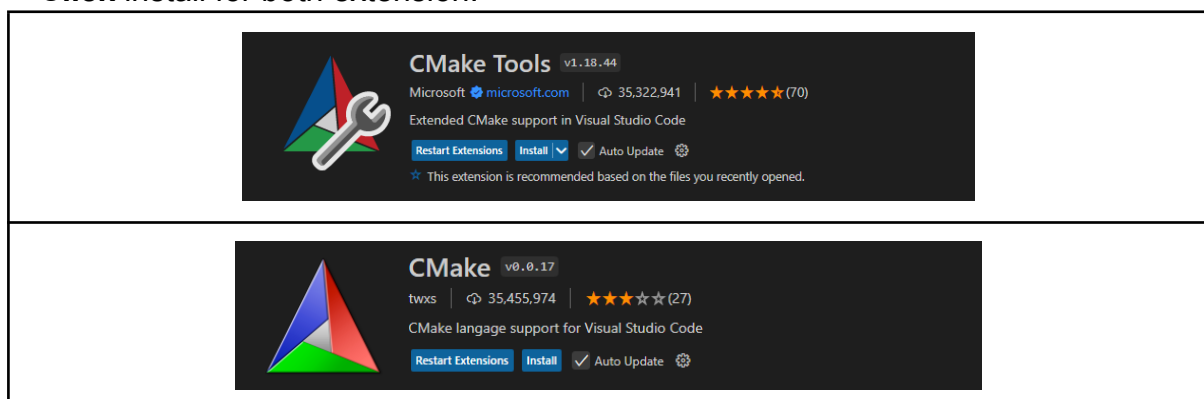


### Step-6:

- ➔ **Open** Visual Studio Code and **Install** Cmake and others extensions:



- ➔ **Click** install for both extension:



### Step-7:

- ➡ **Open** Visual Studio Code
- ➡ **Click** on **Open Folder** and **Select** USB-RELAY folder
- ➡ **Cmake** popup windows appear **Click** on **Configure Project**:  
Cmake will automatically detect your installation and build your project in build folder
- ➡ **Connect** your USB-RELAY board and **Set** the correct COM port and the default relay number in the example code located in: USB-RELAY/example/relaycontrol.cpp

```
Usbrelay* usbrelay = new Usbrelay("COM8",8);
```

- ➡ **Click** on **Build**(1) then **Launch**(2) to start the example sequence



Feel free to modify and change the Code

## On linux:

### Step-1:

Download and Install Visual Studio Code:

<https://code.visualstudio.com/>

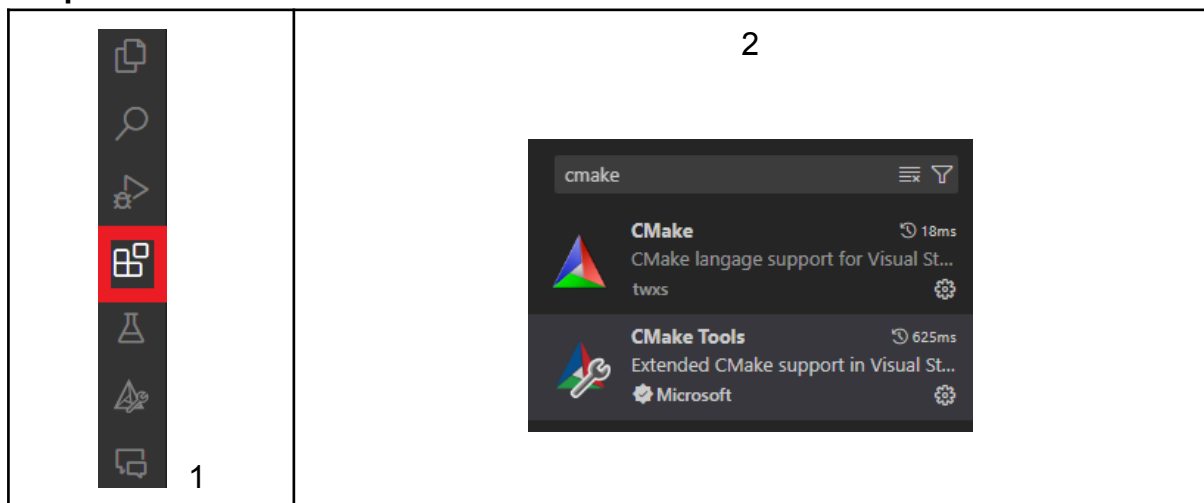
### Step-3:

Install Cmake with apt, run the following command in your terminal

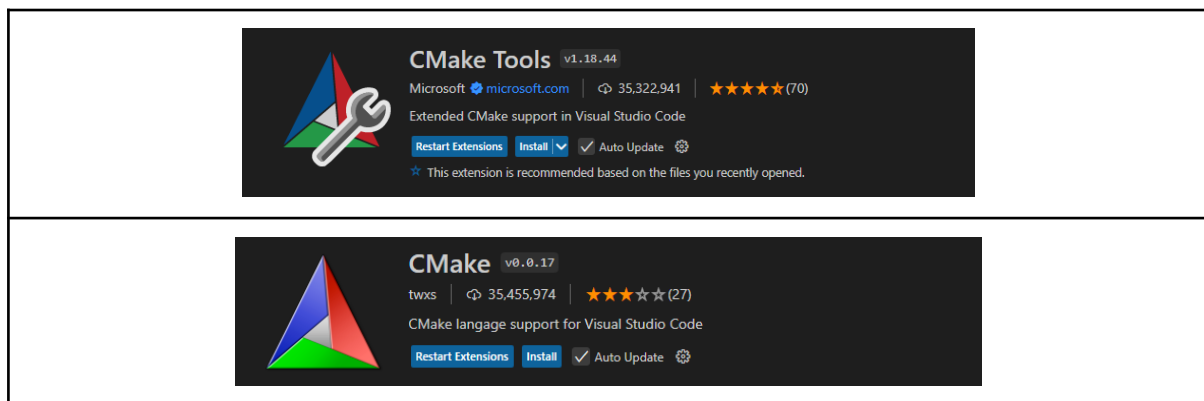
```
sudo apt-get install cmake
```

### Step-2:

➔ **Open** Visual Studio Code and **Install** Cmake and others extensions:



➔ **Click install** for both extension:

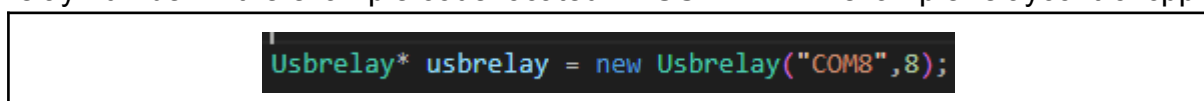


### Step-3:

➔ **Open** Visual Studio Code

➔ **Click** on **Open Folder** and **Select** USB-RELAY folder

➔ **Connect** your USB-RELAY board and **Set** the correct COM port and the default relay number in the example code located in: USB-RELAY/example/relaycontrol.cpp



➔ **Open** your terminal in the USB-RELAY folder, run the following commands to build the project:

```
mkdir build  
cd build  
cmake ..  
make -j4
```

➡ **Run** the examples script

```
sudo ./usbrelay
```

**Feel free to modify and change the Code**

## FR INSTALL

### Sur windows >= 10:

#### Etape-1:

Installez le pilote PL2303 pour Windows en suivant le guide d'installation :  
(USB-RELAYDriverInstallationEN-FR).pdf

#### Etape-2:

Télécharger et installer Visual Studio Code:

<https://code.visualstudio.com/>

#### Etape-3:

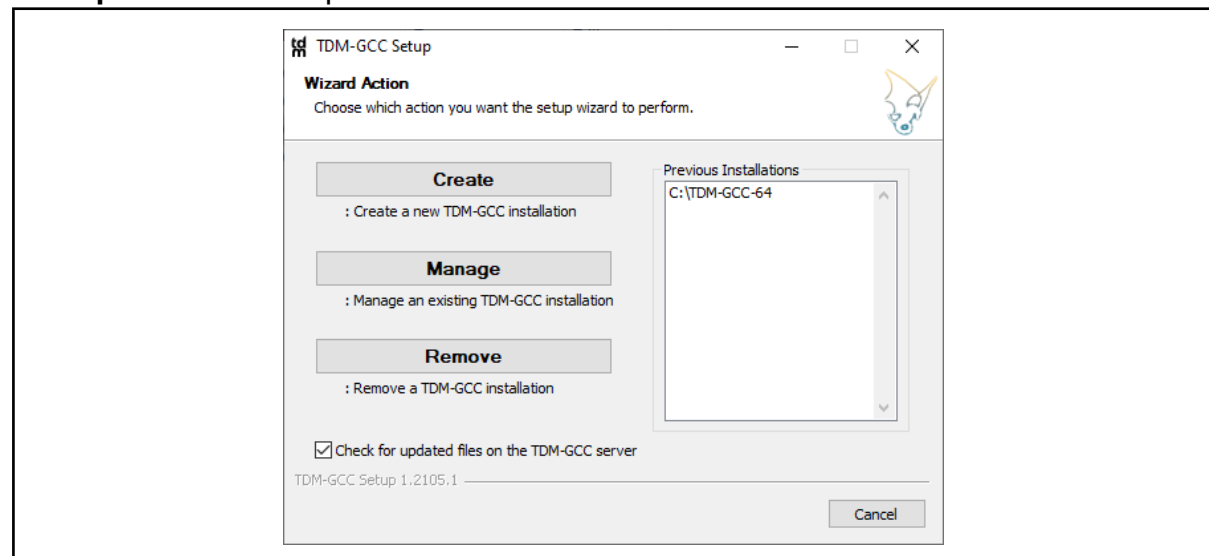
Télécharger et installer TDM-GCC, version >= 10.3.0:

<https://jmeubank.github.io/tdm-gcc/download/>

### ➔ Sélectionner tdm64-gcc avec MinGw et toutes les bibliothèques

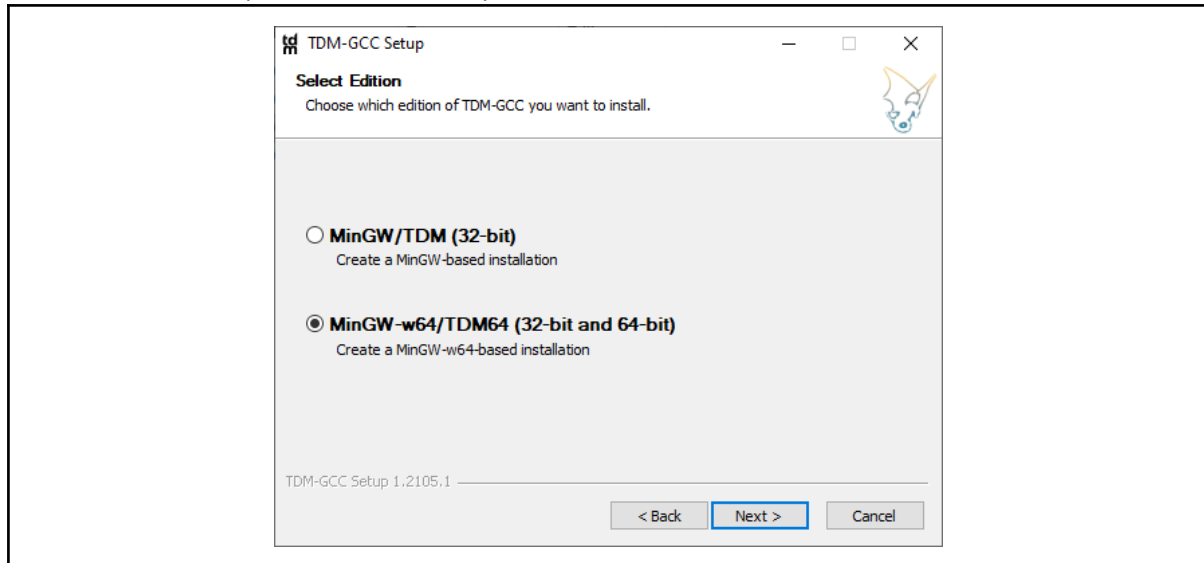
<a href="#">tdm-gcc-webdl.exe</a>	Minimal online installer. Select the components you want, and it downloads and unpacks them. Either edition, latest release only. (GCC 10.3.0)
<a href="#">tdm64-gcc-10.3.0-2.exe</a>	64+32-bit MinGW-w64 edition. Includes GCC C/C++, GNU binutils, mingw32-make, GDB (64-bit), the MinGW-w64 runtime libraries and tools, and the windows-default-manifest package.
<a href="#">tdm-gcc-10.3.0.exe</a>	32-bit-only MinGW.org edition. Includes GCC C/C++, GNU binutils, mingw32-make, GDB (32-bit), the MinGW.org mingwrt and w32api packages, and the windows-default-manifest package.

### ➔ Cliquer sur **Create** pour créer une nouvelle installation de TDM-GCC

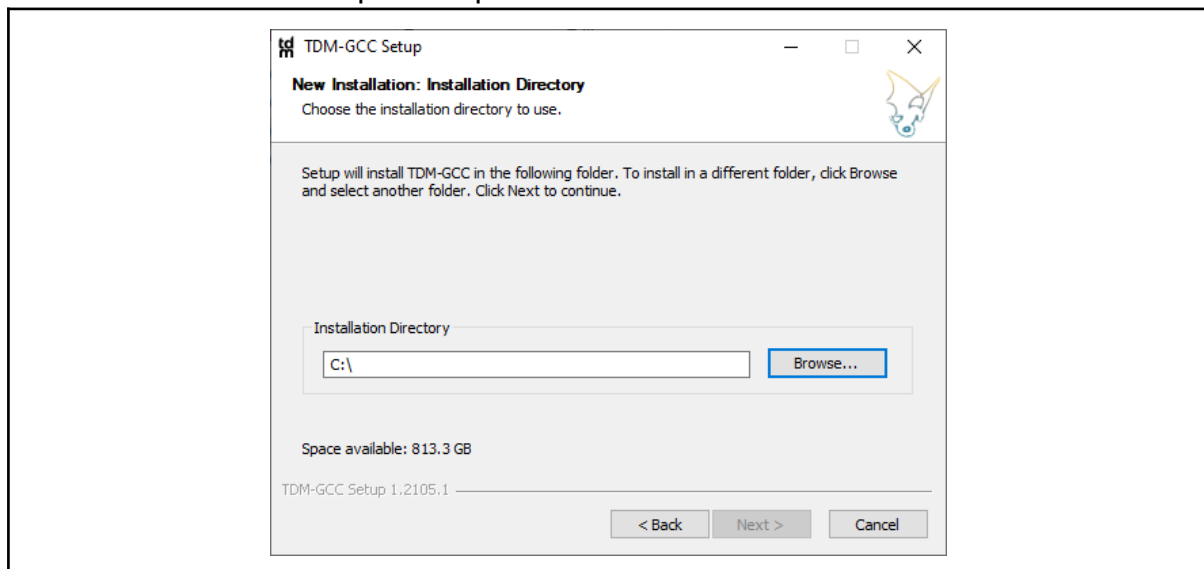




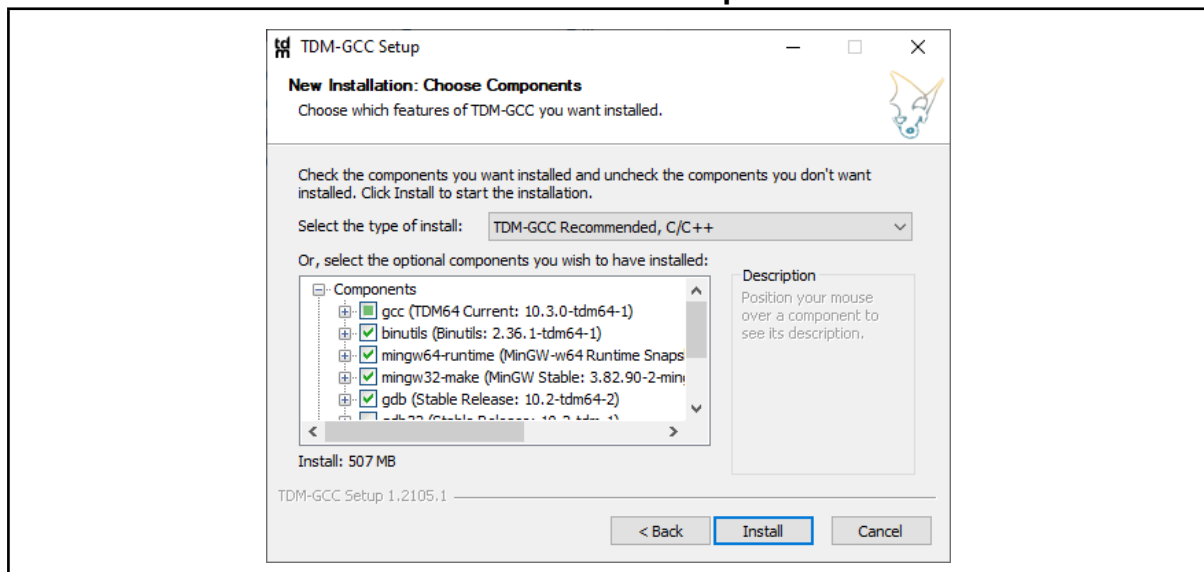
## ➡ Sélectionner (32-bit and 64-bit)



## ➡ Installation dans le répertoire par défaut



## ➡ Sélectionner l'installation recommandée et Cliquer sur install



#### **Etape-4:**

Télécharger et installer Cmake, version >= 3.30.2:

<https://cmake.org/download/>

#### **➡Télécharger Windows x64 Installer**

Binary distributions:

Platform	Files
Windows x64 Installer:	<a href="#">cmake-3.30.2-windows-x86_64.msi</a>
Windows x64 ZIP	<a href="#">cmake-3.30.2-windows-x86_64.zip</a>
Windows i386 Installer:	<a href="#">cmake-3.30.2-windows-i386.msi</a>
Windows i386 ZIP	<a href="#">cmake-3.30.2-windows-i386.zip</a>
Windows ARM64 Installer:	<a href="#">cmake-3.30.2-windows-arm64.msi</a>
Windows ARM64 ZIP	<a href="#">cmake-3.30.2-windows-arm64.zip</a>
macOS 10.13 or later	<a href="#">cmake-3.30.2-macos-universal.dmg</a>
	<a href="#">cmake-3.30.2-macos-universal.tar.gz</a>
macOS 10.10 or later	<a href="#">cmake-3.30.2-macos10.10-universal.dmg</a>
	<a href="#">cmake-3.30.2-macos10.10-universal.tar.gz</a>
Linux x86_64	<a href="#">cmake-3.30.2-linux-x86_64.sh</a>
	<a href="#">cmake-3.30.2-linux-x86_64.tar.gz</a>
Linux aarch64	<a href="#">cmake-3.30.2-linux-aarch64.sh</a>
	<a href="#">cmake-3.30.2-linux-aarch64.tar.gz</a>

#### **Etape-5:**

Télécharger Ninja

<https://github.com/ninja-build/ninja/releases>

#### **➡Télécharger ninja-win.zip**

**v1.12.1** Latest

Bugfixes:

- Screen updates extremely slow on Windows [#2435](#)
- Dry run error if the build directory does not exist [#2431](#)
- New critical path scheduler performance improvements [#2443](#)

<https://github.com/ninja-build/ninja/milestone/11?closed=1>

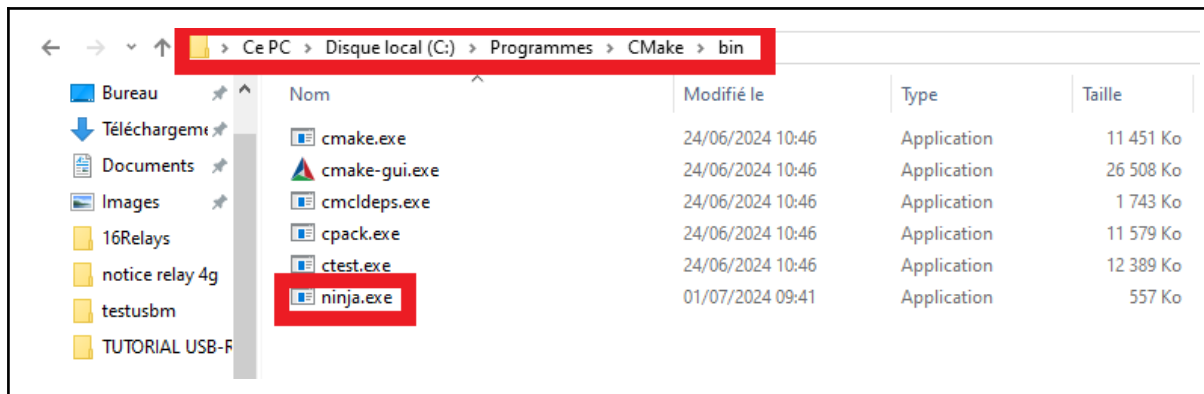
**Assets** 7

<a href="#">ninja-linux-aarch64.zip</a>	119 KB	May 11
<a href="#">ninja-linux.zip</a>	120 KB	May 11
<a href="#">ninja-mac.zip</a>	275 KB	May 11
<a href="#">ninja-win.zip</a>	269 KB	May 11
<a href="#">ninja-winarm64.zip</a>	248 KB	May 11
<a href="#">Source code (zip)</a>		May 11
<a href="#">Source code (tar.gz)</a>		May 11

👍 22 🍌 4 ❤️ 3 🚀 10 31 people reacted

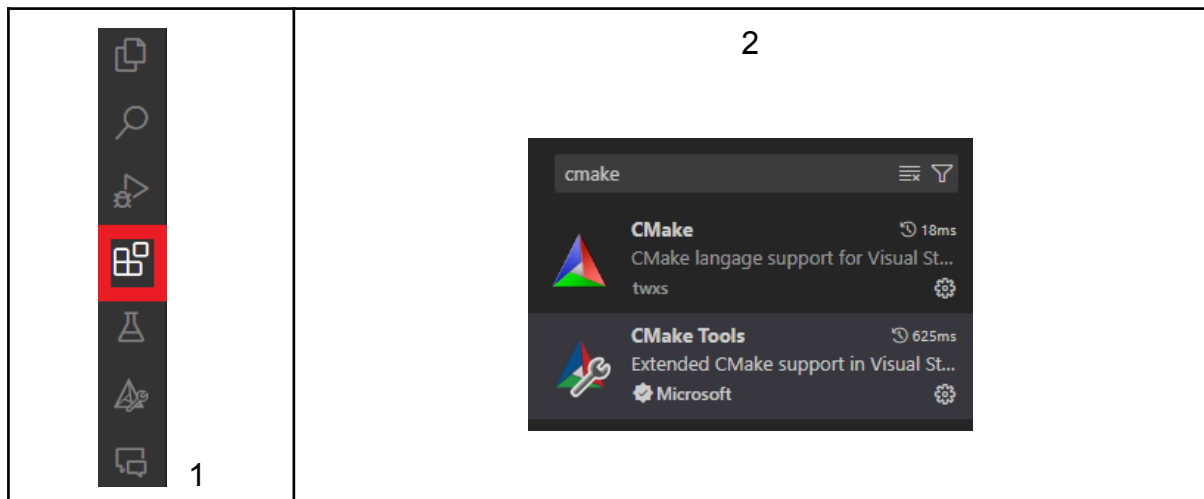
#### **➡Décompresser l'archive, vous devez obtenir un fichier ninja.exe**

➡ **Ouvrez** votre dossier d'installation de CMake et **Collez** ninja.exe dans le dossier bin de CMake

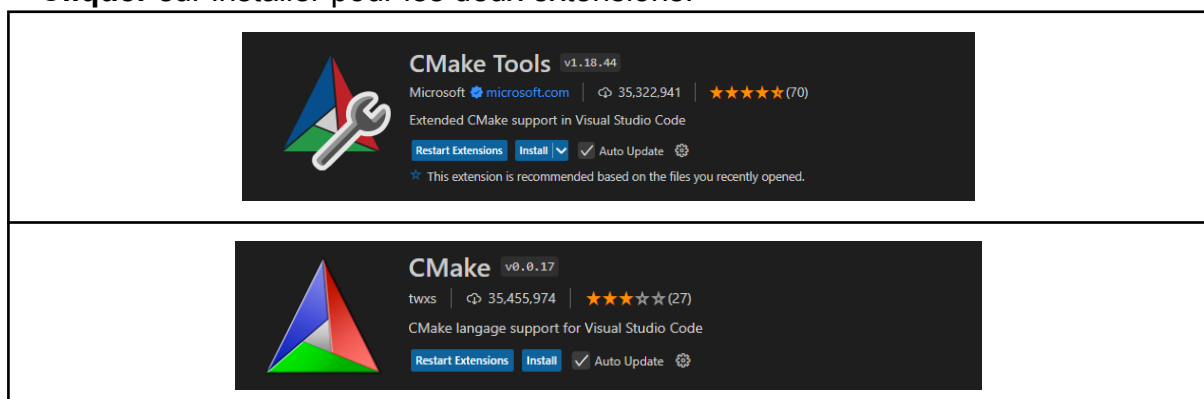


### Etape-6:

➡ **Ouvrir** Visual Studio Code et installer Cmake et les extensions de votre choix



➡ **Cliquer** sur installer pour les deux extensions:



### Etape-7:

- ➡ **Ouvrir** Visual Studio Code
- ➡ **Cliquer** sur **Ouvrir un dossier** et **Sélectionner** le dossier USB-RELAY
- ➡ **Un popup Cmake** va apparaître **Cliquer** sur Configurer le Projet:  
CMake détectera automatiquement votre installation et construira votre projet dans le dossier build.
- ➡ **Connectez** votre carte USB-RELAY et **définissez** le port COM correct et le nombre de relais par défaut dans le code d'exemple situé dans:  
**USB-RELAY/example/relaycontrol.cpp**

```
Usbrelay* usbrelay = new Usbrelay("COM8",8);
```

- ➡ **Cliquer** sur **Build**(1) puis **Start**(2) pour lancer le programme d'exemples



## On linux:

### Step-1:

Télécharger et installer Visual Studio Code:

<https://code.visualstudio.com/>

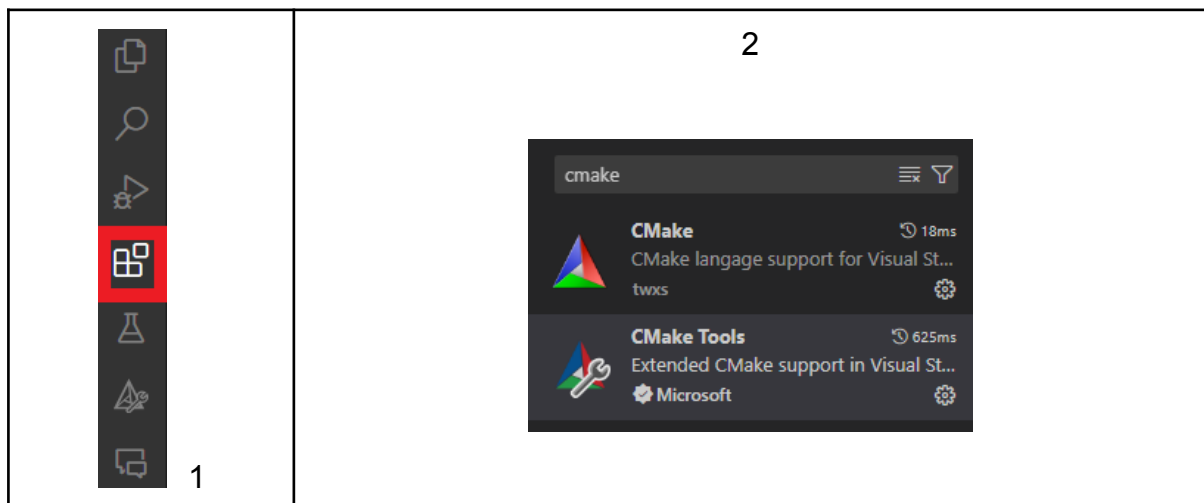
### Step-3:

Installer Cmake avec apt, exécuter la commande suivante dans votre terminal

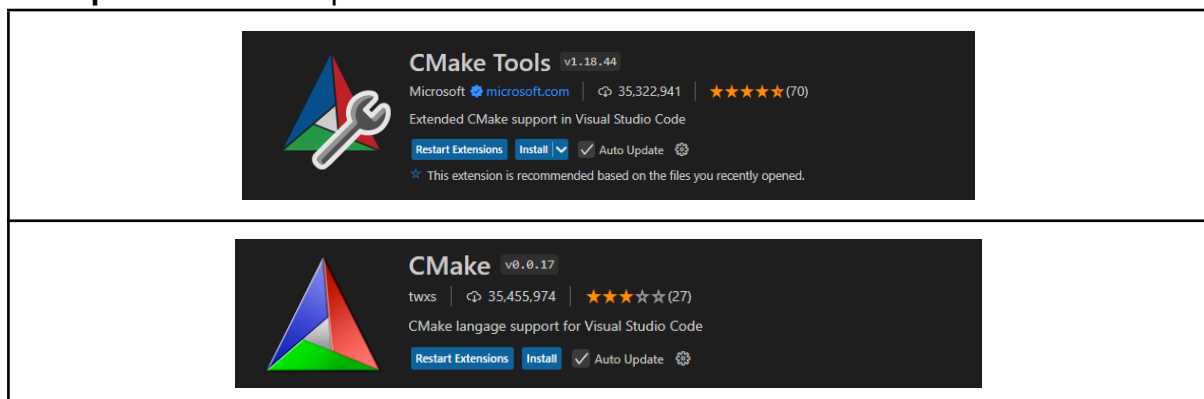
```
sudo apt-get install cmake
```

### Etape-2:

➔ **Ouvrir** Visual Studio Code et installer Cmake et les extensions de votre choix



➔ **Cliquer** sur installer pour les deux extensions:



### Step-3:

➔ **Ouvrir** Visual Studio Code

➔ **Cliquer** sur **Open Folder** and **Sélectionner le dossier USB-RELAY**

➡ **Connectez** votre carte USB-RELAY et **définissez** le port COM correct et le nombre de relais par défaut dans le code d'exemple situé dans:

**USB-RELAY/example/relaycontrol.cpp**

```
Usbrelay* usbrelay = new Usbrelay("COM8",8);
```

➡ **Ouvrez** votre terminal dans le dossier USB-RELAY et exécutez les commandes suivantes pour construire le projet:

```
mkdir build  
cd build  
cmake ..  
make -j4
```

➡ **Exécuter** le script d'exemple

```
sudo ./usbrelay
```