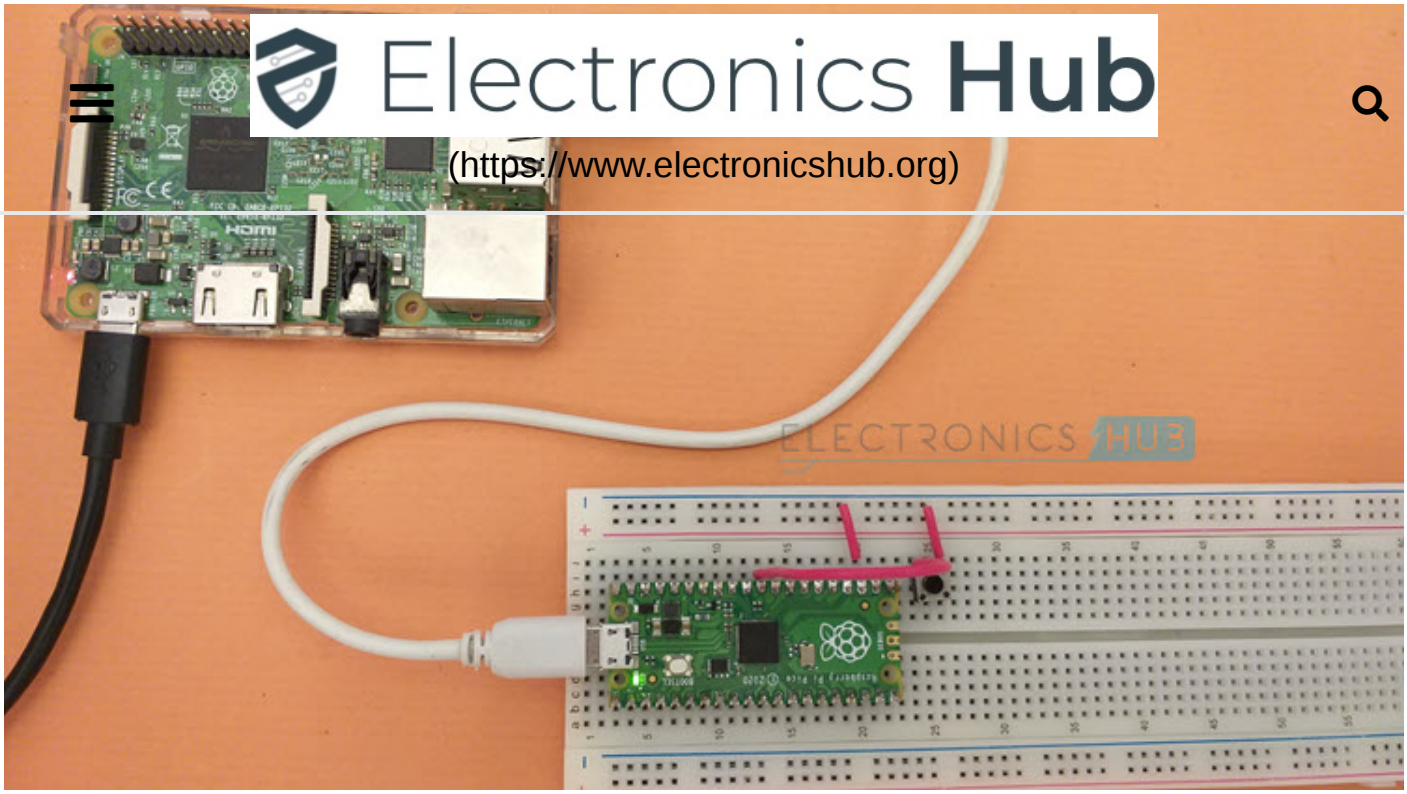
Programming Raspberry Pi Pico using C | Getting Started with C SDK

March 5, 2021 By Ravi Teja (Embedded Engineer)(https://www.electronicshub.org/author/raviteja/)

In this tutorial, we will learn how to Program Raspberry Pi Pico using C. We will see all the steps required for Programming Raspberry Pi Pico in C Programming Language like downloading Raspberry Pi Pico's C SDK and examples from github, installing all the necessary tools, building the Blinky C Program and finally loading the UF2 file to Pico and running the Blinky example.



(https://www.electronicshub.org/wp-content/uploads/2021/03/Program-Raspberry-Pi-Pico-using-C-Image.jpg)

Outline

Have You Read These?

Before proceed ; that I made for Raspberry P (https://www.electronicshub.org/getting-started-with-raspberry-pi-pico/)' and 'Programming Raspberry Pi Pico with MicroPython (https://www.electronicshub.org/raspberry-pi-pico-micropython-tutorial/)'.

These tutorials include some basics about Raspberry Pi Pico, the RP2040 Microcontroller, Pinout of Pico, programming environment, MicroPython and many other important topics.

I strongly recommend you to complete those two tutorials to get familiar with Raspberry Pi Pico.

Getting Started with Raspberry Pi Pico C SDK

If you are an old school hardware engineer (like I am) and has experience in writing firmware, then you will probably be more comfortable in C Programming Language rather than Python or MicroPython for writing applications / firmware. Ironically, MicroPython is written in C.

Even though the Raspberry Pi Foundation promotes Python as the primary programming language for all the Raspberry Pi SBCs, things are quire different when it comes to Microcontrollers.

I consider C as the low-level programming among all the high-level programming languages. If you want complete control of the hardware right down to the individual bit of a register, then C is ultimate choice. Hence, C is the preferred choice of programming for many microcontrollers and embedded systems.

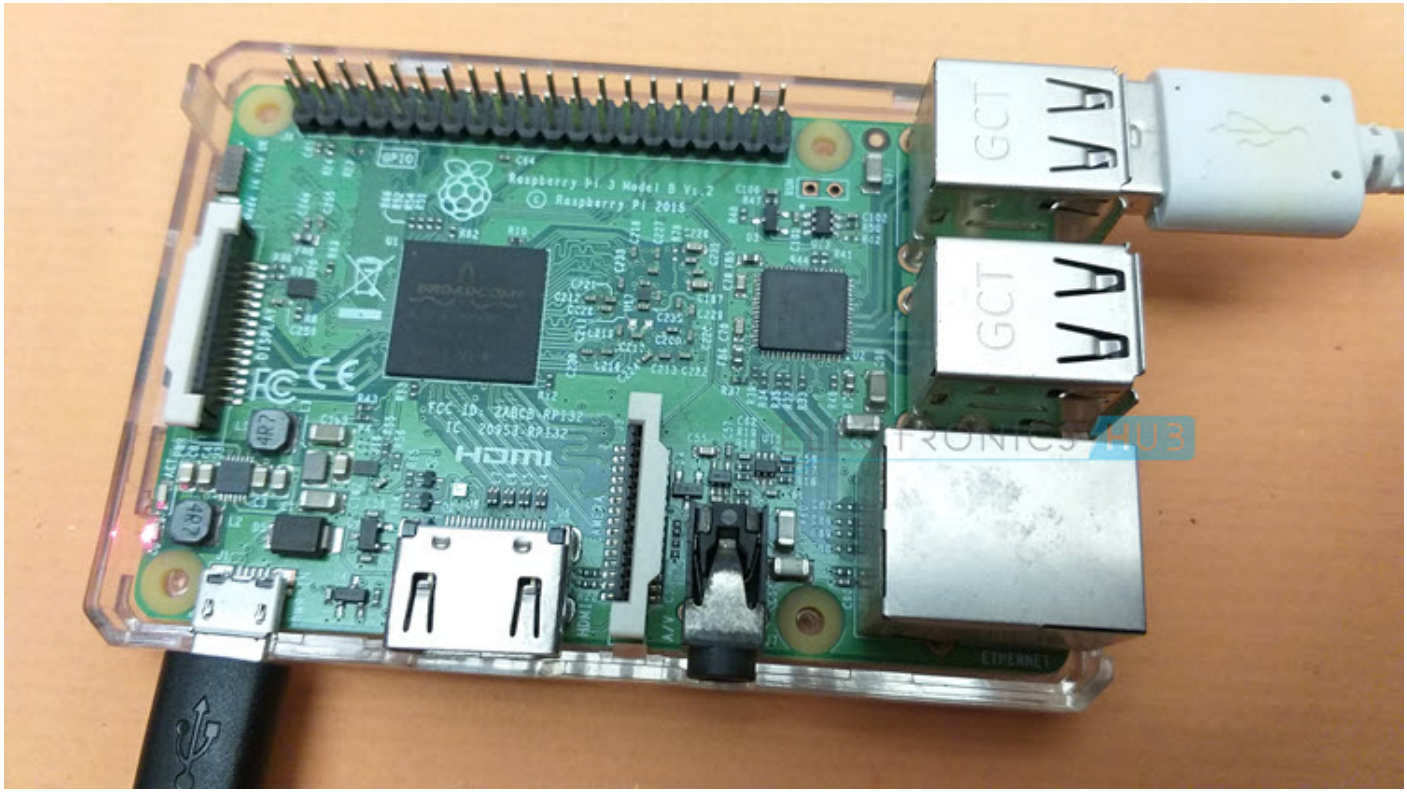
The RP2040 Microcontroller based Raspberry Pi Pico is no exception to this. While

investing and pi released SDK (Software Devel p application for RP2040 based boards. (https://www.electronicshub.org)

Let us see how to Program Raspberry Pi Pico using C Programming Language.

What Computer to use?

I already discussed this in the 'Programming Pico with MicroPython' tutorial. You can use any of the Linux, Windows or Mac based systems as the host computer. But the official documentation of Raspberry Pi Pico focuses on Raspberry Pi Computer running Raspberry Pi OS as the main host system (even though it mentions procedures for other systems).



(https://www.electronicshub.org/wp-content/uploads/2021/03/Raspberry-Pi-3-Model-B-for-Pico.jpg)

I do not have th i 3 Model B. So, when working v tions for Pico. In the future tutorials, I will explain how to work with other systems and environments like Windows, Visual Studio Code and others.

Get Pico's C SDK and Examples

Assuming you are also using a Raspberry Pi computer, login to Pi using either SSH or VNC (if you connect Pi to a monitor, then well and good). I will be using the VNC viewer to open the desktop of Raspberry Pi.

Open the terminal and create a directory called 'pico' at /home/pi/pico. Enter the following commands one after the other.

```
cd ~/
```

```
mkdir pico
```

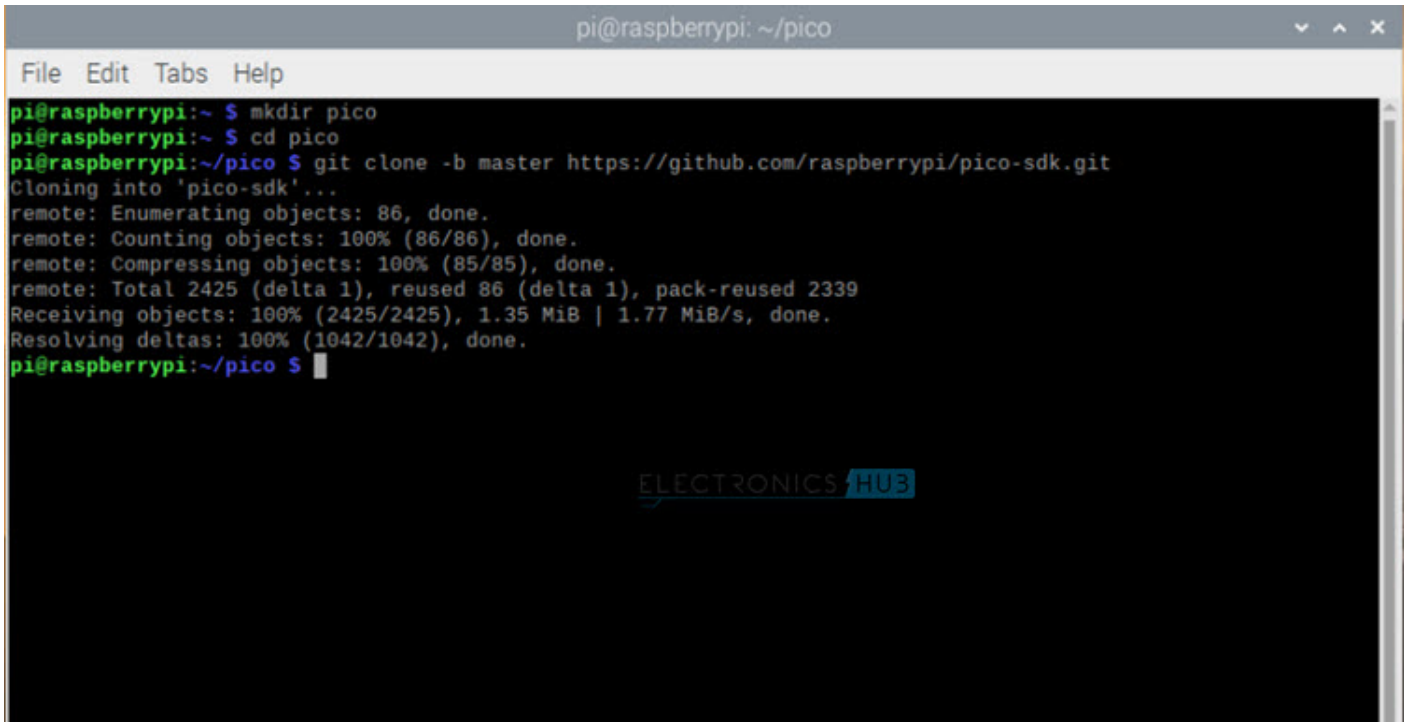
```
cd pico
```

(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Download-SDK-1.jpg)

Download C SDK

The official repository for Pico's C SDK is https://github.com/raspberrypi/pico-sdk (https://github.com/raspberrypi/pico-sdk). Let us clone this repo into our newly created 'pico' directory using the following command.

```
git clone -b master https://github.com/raspberrypi/pico-sdk.git
```



(https://www.electronicshub.org)

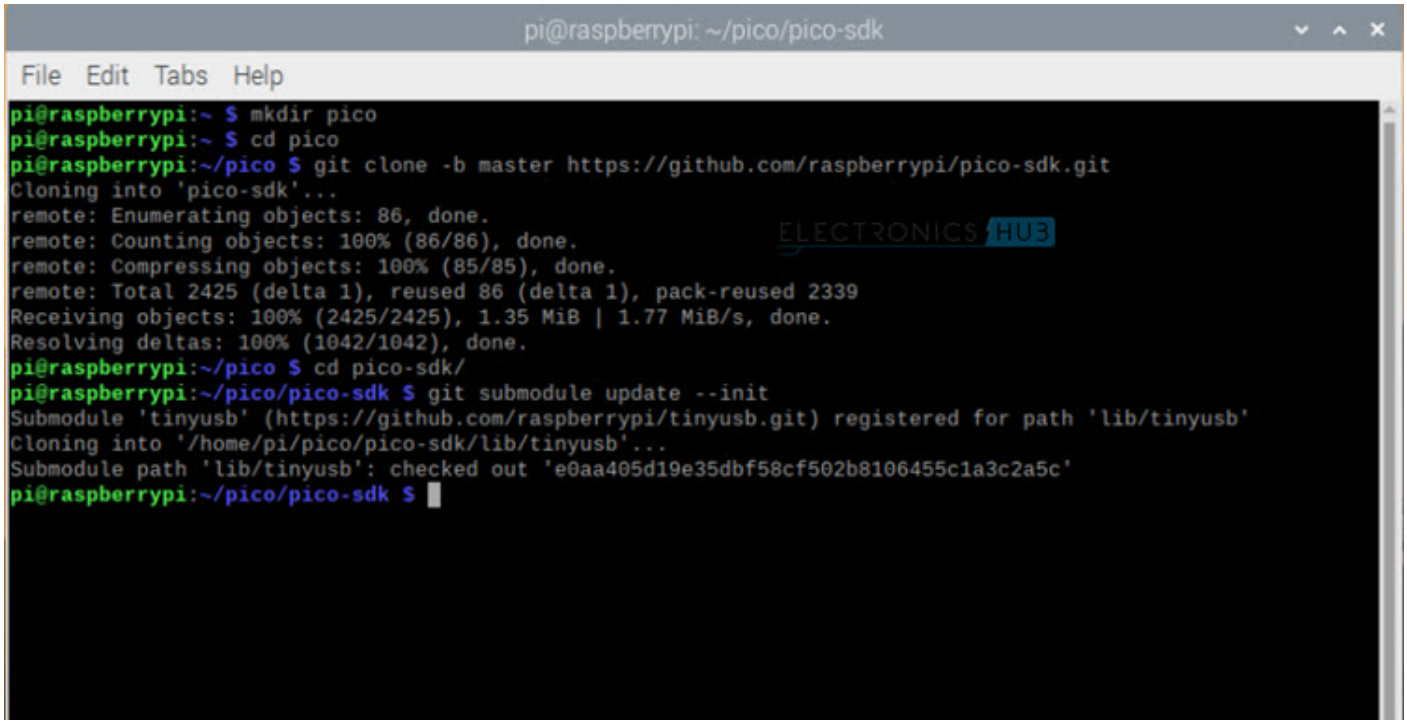
(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Download-SDK-2.jpg)

This will create a directory called 'pico-sdk' inside the 'pico' directory. Before proceeding with downloading the examples, you have to initialize the 'tinyusb' submodule in the 'pico-sdk' for USB related functions like USB CDC Serial to work.

So, open 'pico-sdk' directory and initialize the USB submodule using the following commands.

```
cd pico-sdk
```

```
git submodule update – -init
```



(https://www.electronicshub.org)

(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Download-SDK-3.jpg)

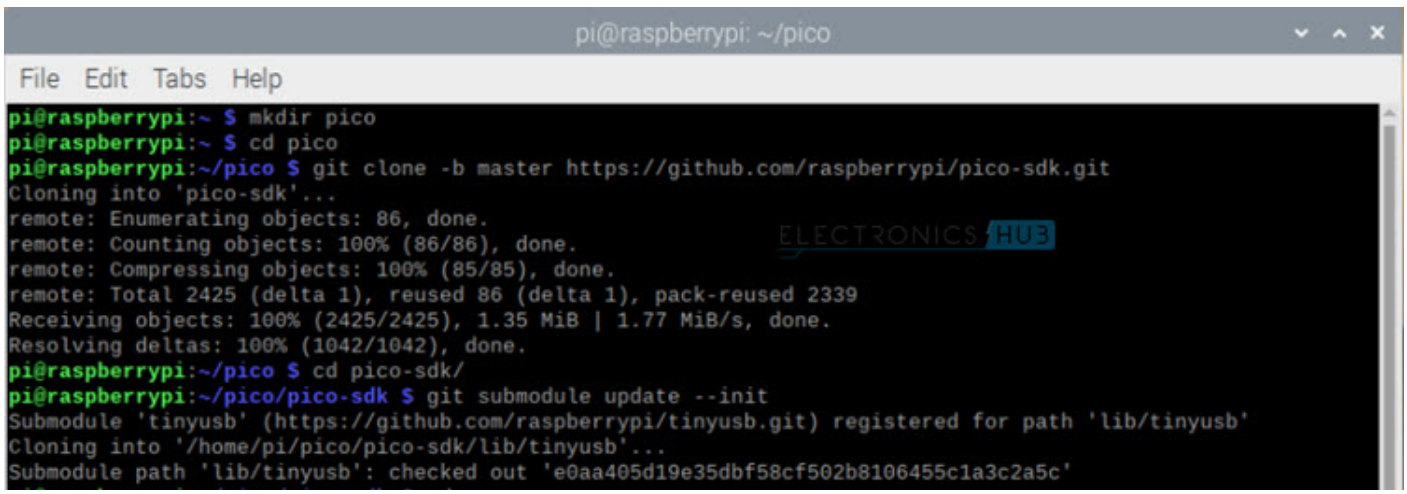
Download Examples

Raspberry Pi Foundation created a bunch of examples for different peripherals like GPIO, ADC, PWM, I^2C, SPI etc. of Pico. Instead of writing C programs from scratch, we can make use of these examples to get familiar with different functions and APIs. The official repo of Pico examples is https://github.com/raspberrypi/pico-examples (https://github.com/raspberrypi/pico-examples).

We will clone this repo into the 'pico' directory. So, go back to the 'pico' directory and clone the git using the following commands.

```
cd ..
```

```
git clone -b master https://github.com/raspberrypi/pico-examples.git
```

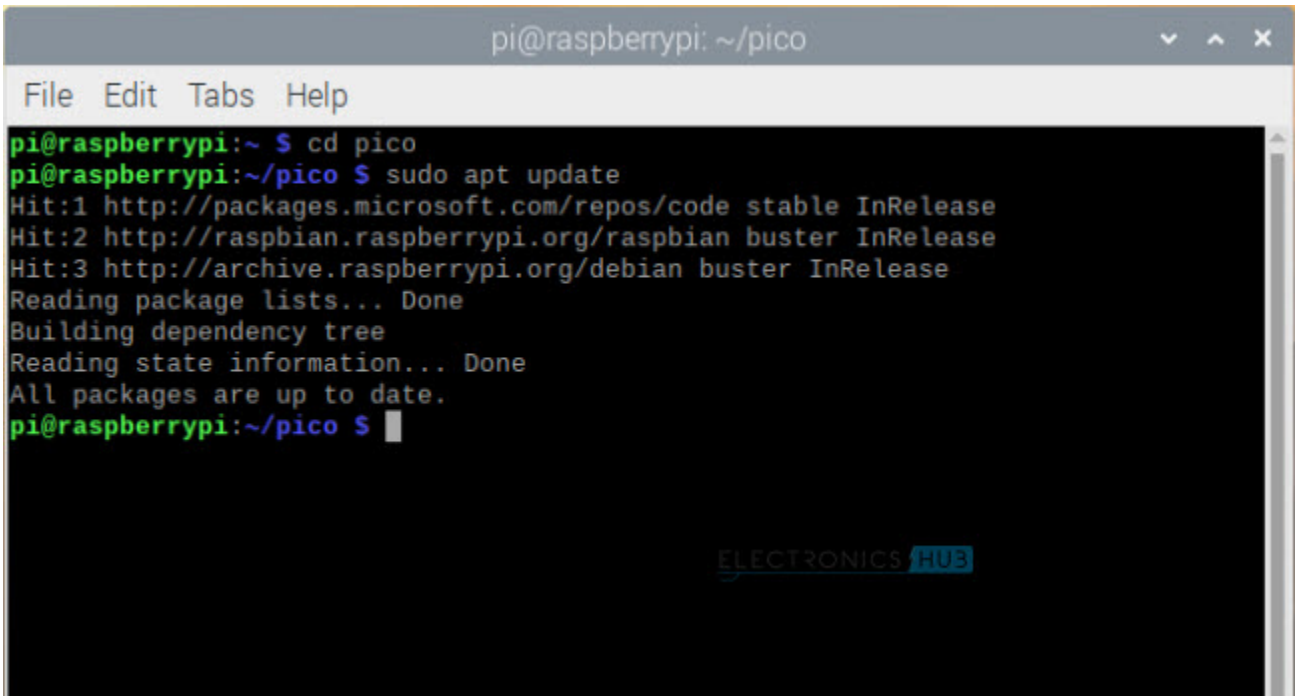


(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Download-Examples-1.jpg)

Installing the Toolchain

You just downloaded the C source files for Pico. To compile and build these files, you need additional tools like ARM GCC Compiler, ARM Cortex libraries, CMake build tool etc. We can install all these tools using 'apt'. First, update the package index using 'update' command.

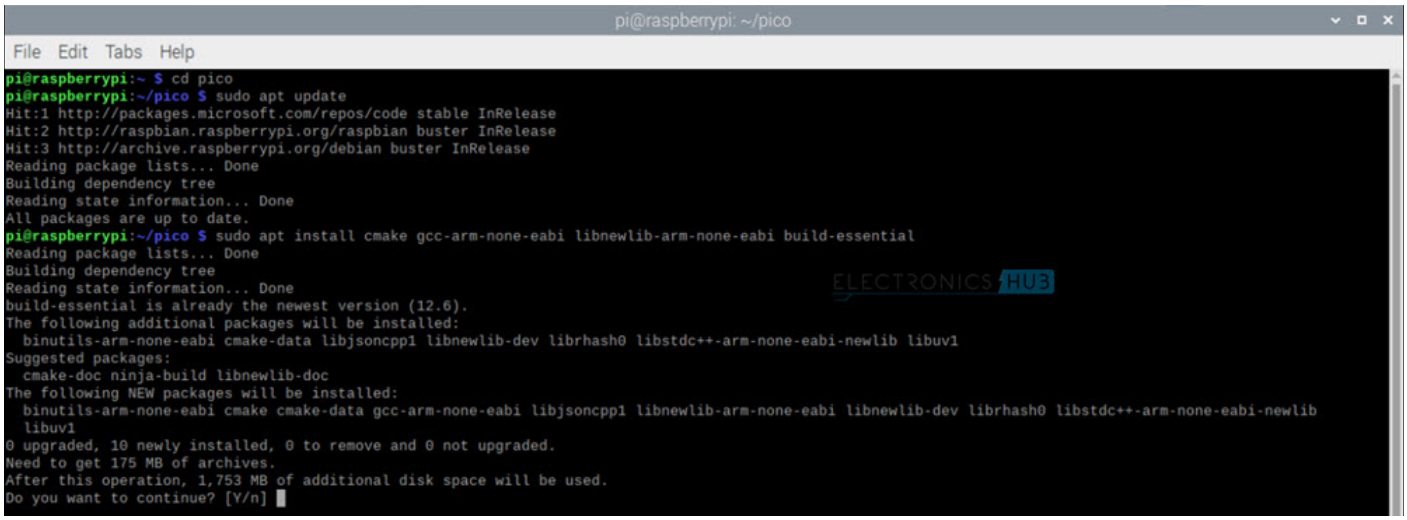
```
sudo apt update
```



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Install-Toolchain-1.jpg)

Now, you can install the necessary tools using the following command. If you already have any of these tools installed, 'apt' will not install them again.

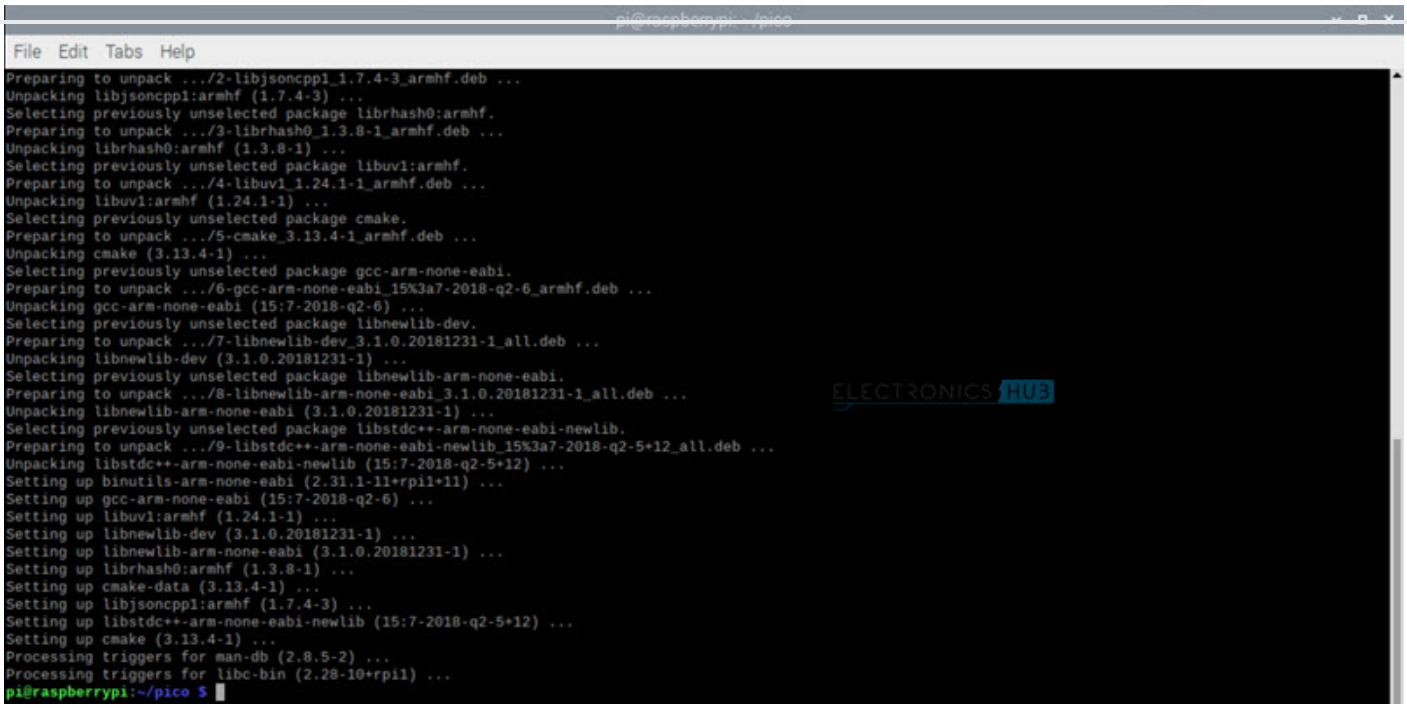
```
sudo apt install cmake gcc-arm-none-eabi libnewlib-arm-none-eabi build-essential
```



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Install-Toolchain-2.jpg)

WARNING: This is a very large installation which occupies approximately 1800 MB of disk space. Make sure you have enough space on the SD Card in which you installed the Raspberry Pi OS. I used a 16 GB card.

Also, it will take coffee and relax.



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Install-Toolchain-3.jpg)

Exploring Blink Example

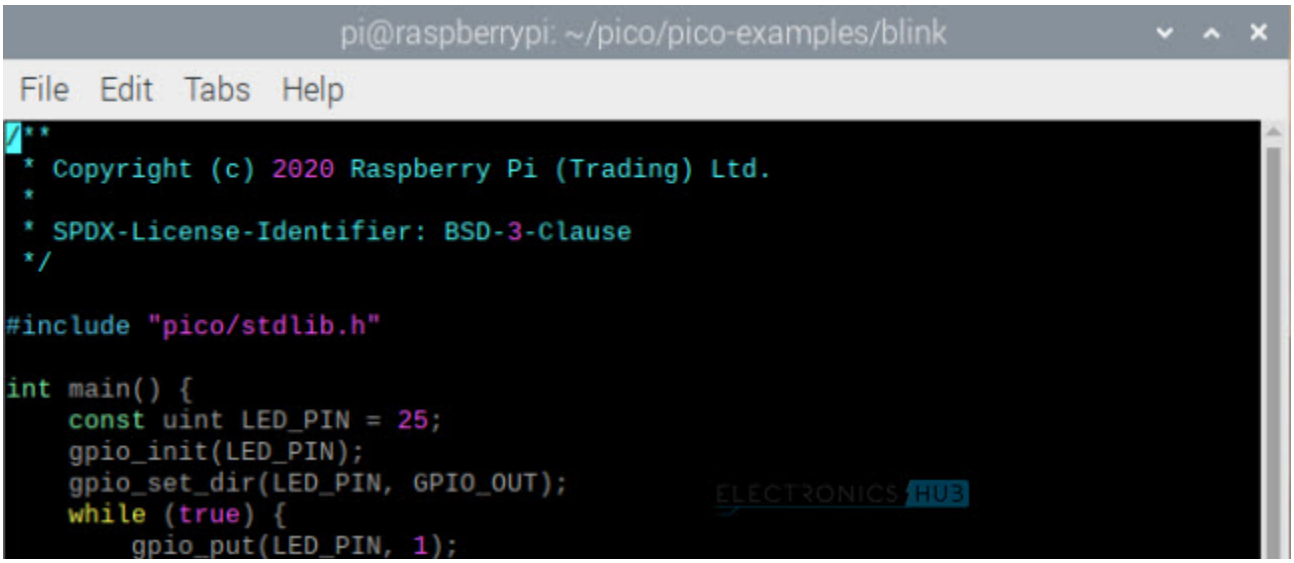
Let us now see the first C example program, which is nothing but the 'Blink' program, as expected. Assuming you are still in 'pico' directory, open the 'pico-examples' directory. This directory contains example for different peripherals and modules. Open 'blink' directory.

```
cd pico-examples/blink
```

(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-1.jpg)

Here, you can see the 'blink.c' source code for blinking the on-board LED. If you want to open it, use any editor like 'vim'. If you don't have 'vim', install it using sudo apt install vim command.

```
vim blink.c
```



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-2.jpg)

I made a copy of the code and added comments for easy understanding.

```c
/**
 * Copyright (c) 2020 Raspberry Pi (Trading) Ltd.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */

#include "pico/stdlib.h"

int main() {
    const uint LED_PIN = 25; /* On-board LED. Connected to GPIO 25 */
    gpio_init(LED_PIN); /* Initialize GPIO function on the Pin */
    gpio_set_dir(LED_PIN, GPIO_OUT); /* Set the Pin as Output */
    while (true) {
        gpio_put(LED_PIN, 1); /* Make the GPIO Pin HIGH */
        sleep_ms(250); /* Delay of 250ms */
        gpio_put(LED_PIN, 0); /* Make the GPIO Pin LOW */
        sleep_ms(250); /* Delay of 250ms */
    }
}
```

view raw (https://gist.github.com/elktros/9cdd468f0d1ae176d6529135913f0169/raw/a0a2310aec4466c00ce310b609b2f424422ebb64/Raspberry-Pi-Pico-C-Blink.c) Raspberry-Pi-Pico-C-Blink.c (https://gist.github.com/elktros/9cdd468f0d1ae176d6529135913f0169#file-raspberry-pi-pico-c-blink-c) hosted with ❤ by **GitHub (https://github.com)**

We have to now build this source code and generate a binary file.

Build the

Go back to 'pico-examples' directory and create a new directory called 'build'. This directory contains all the CMake build files for all the example projects. Open the newly created 'build' directory.

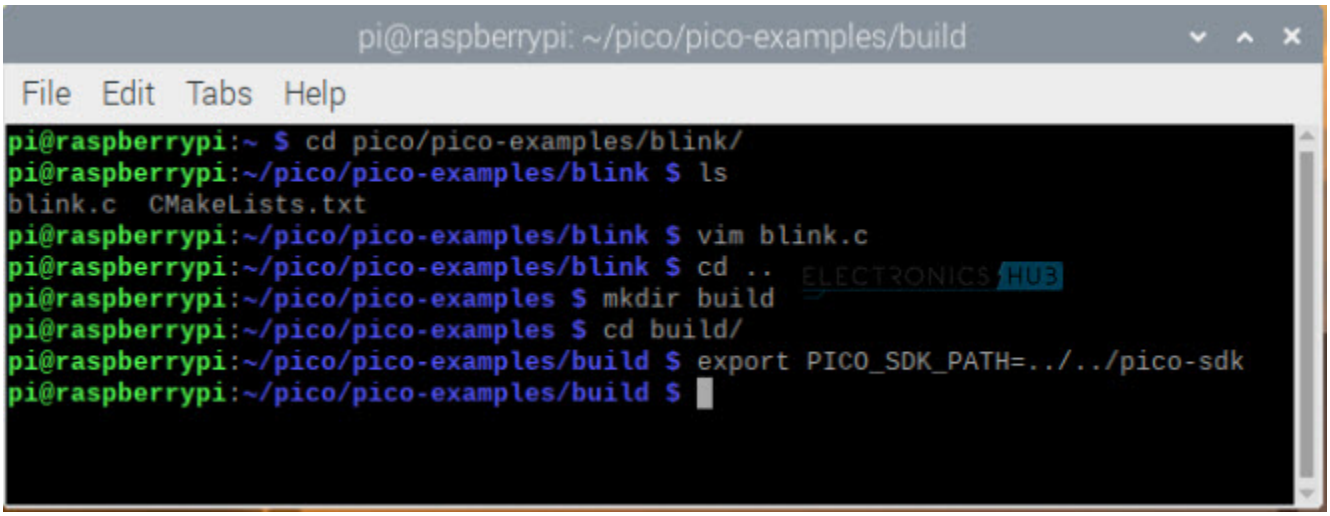
```
cd ..
```

```
mkdir build
```

```
cd build
```

Also, export the path for pico-sdk using the following command.

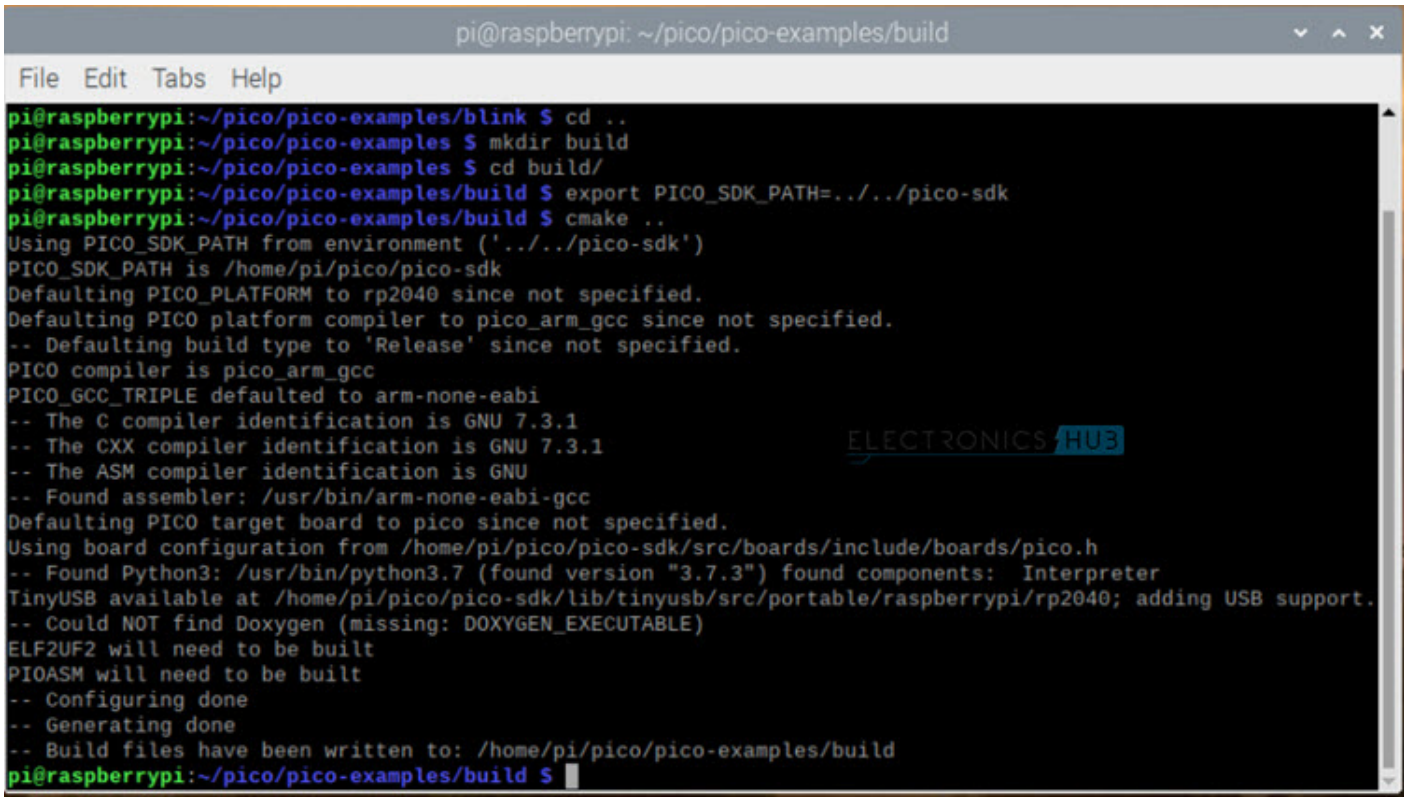
```
export PICO_SDK_PATH=../../pico-sdk
```



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-3.jpg)

You are now re/.


```
cmake ..
```

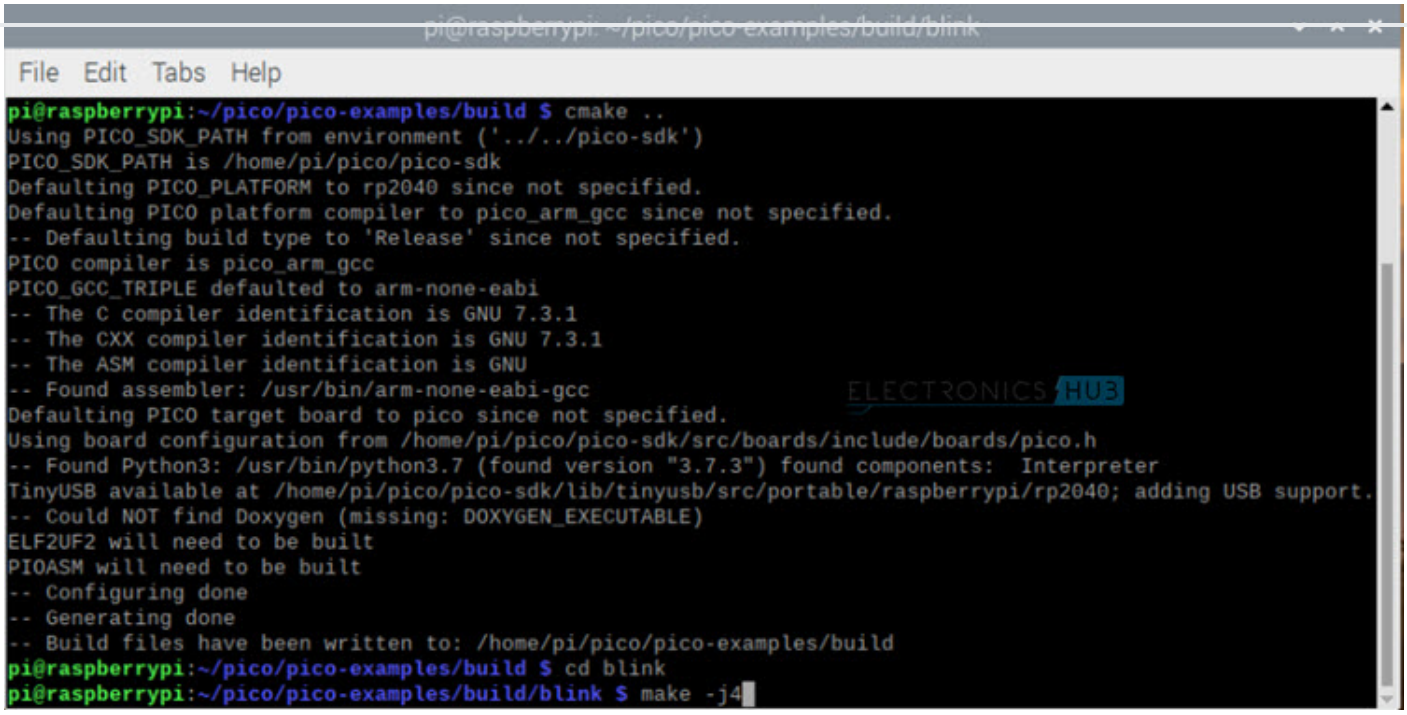


(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-4.jpg)

The ‘build’ directory is now populated with all the build related files for all the examples. But as we are interested only in the blink example, let us build that example using the make command. First, open the ‘blink’ directory in the ‘build’ directory and make the files.

```
cd blink
```

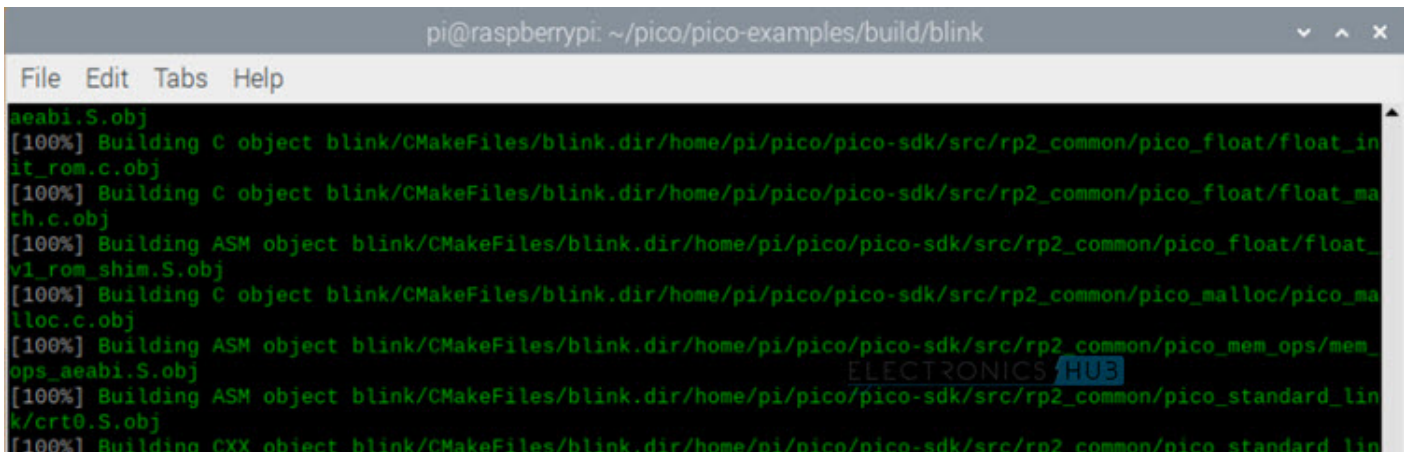
make -j4



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-5.jpg)

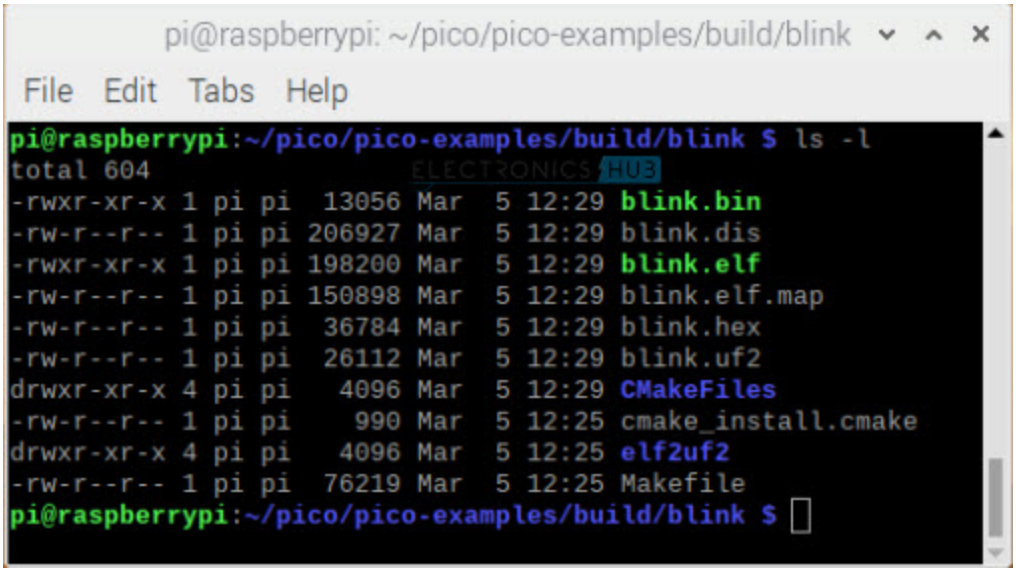
NOTE: The '-j' option lets you specify the number of simultaneous jobs to run. Since, my Raspberry Pi 3 has a quad core CPU, I chose 4 simultaneous jobs.

The 'make' command will compile all the source files and generates a bunch of binary files.



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-6.jpg)

If you list out the contents of the 'blink' directory in the 'build' directory, you can see the list of generated files.



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Build-Blink-7.jpg)

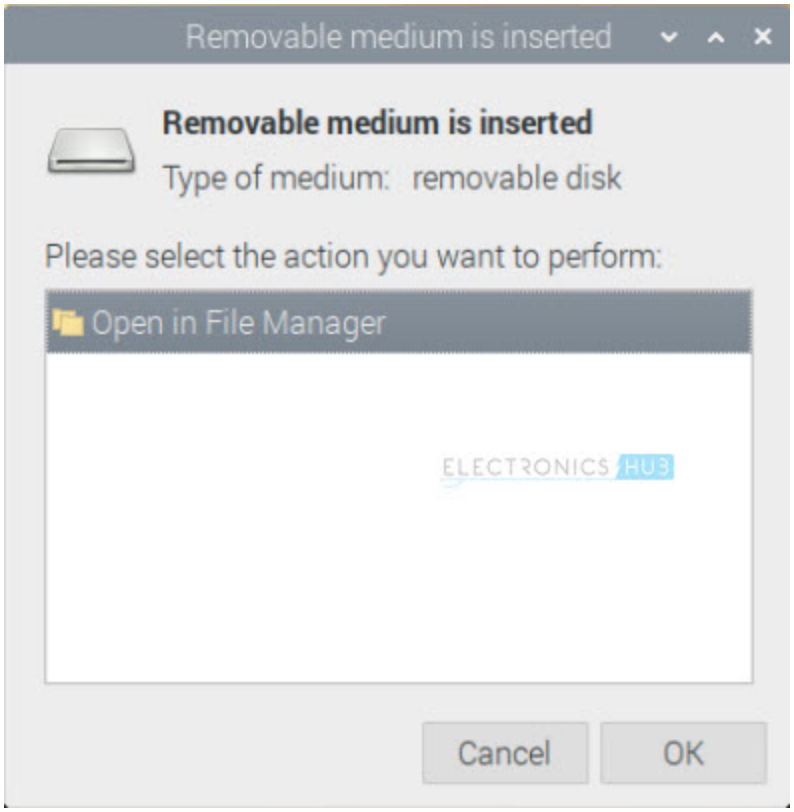
Of all these files, we are interested in the '.uf2' file named 'blink.uf2'. If you remember the MicroPython tutorial, this is file format of the MicroPython binary which can be easily uploaded in to Raspberry Pi Pico simply by dragging and dropping (after setting the Pico in bootloader mode).

Load the Blink and Run it

The simplest way to upload the program to Raspberry Pi Pico is to put it in Bootloader

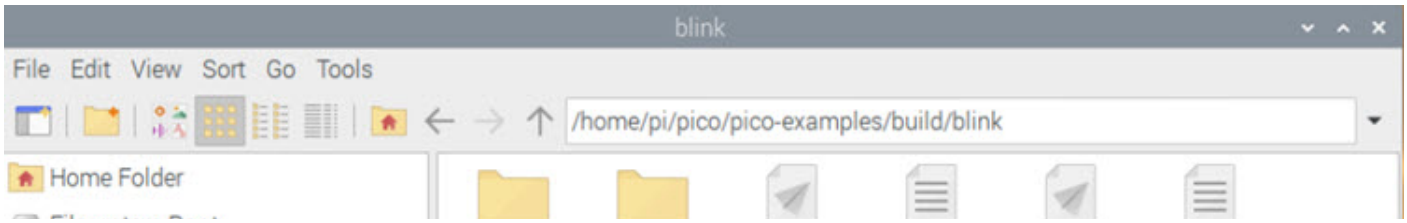
mode, which wi e can simply drag-and-drop t

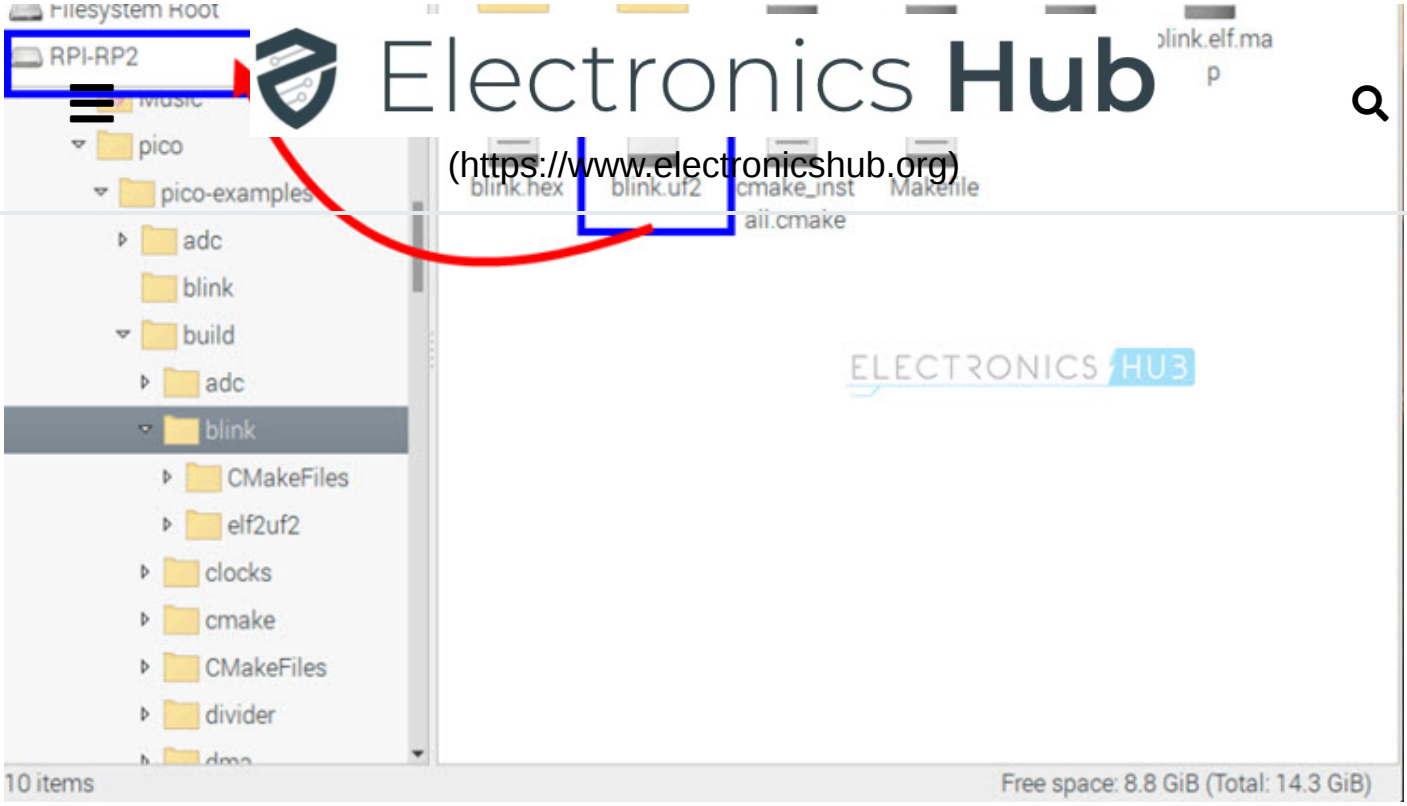
To put the Pico in Bootloader mode first plug-in the micro USB cable to Raspberry Pi Pico and hold the 'BOOTSEL' button while plugging in the USB cable to Raspberry Pi (or the host computer). The Pico will mount as a mass storage device.



(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Load-Blink-1.jpg)

Open the file manages, browse to the pico/pico-examples/build/blink directory. Simply drag and drop the 'blink.uf2' file onto Pico. The Pico will reset (and gets disconnected from the computer) and the LED starts blinking.





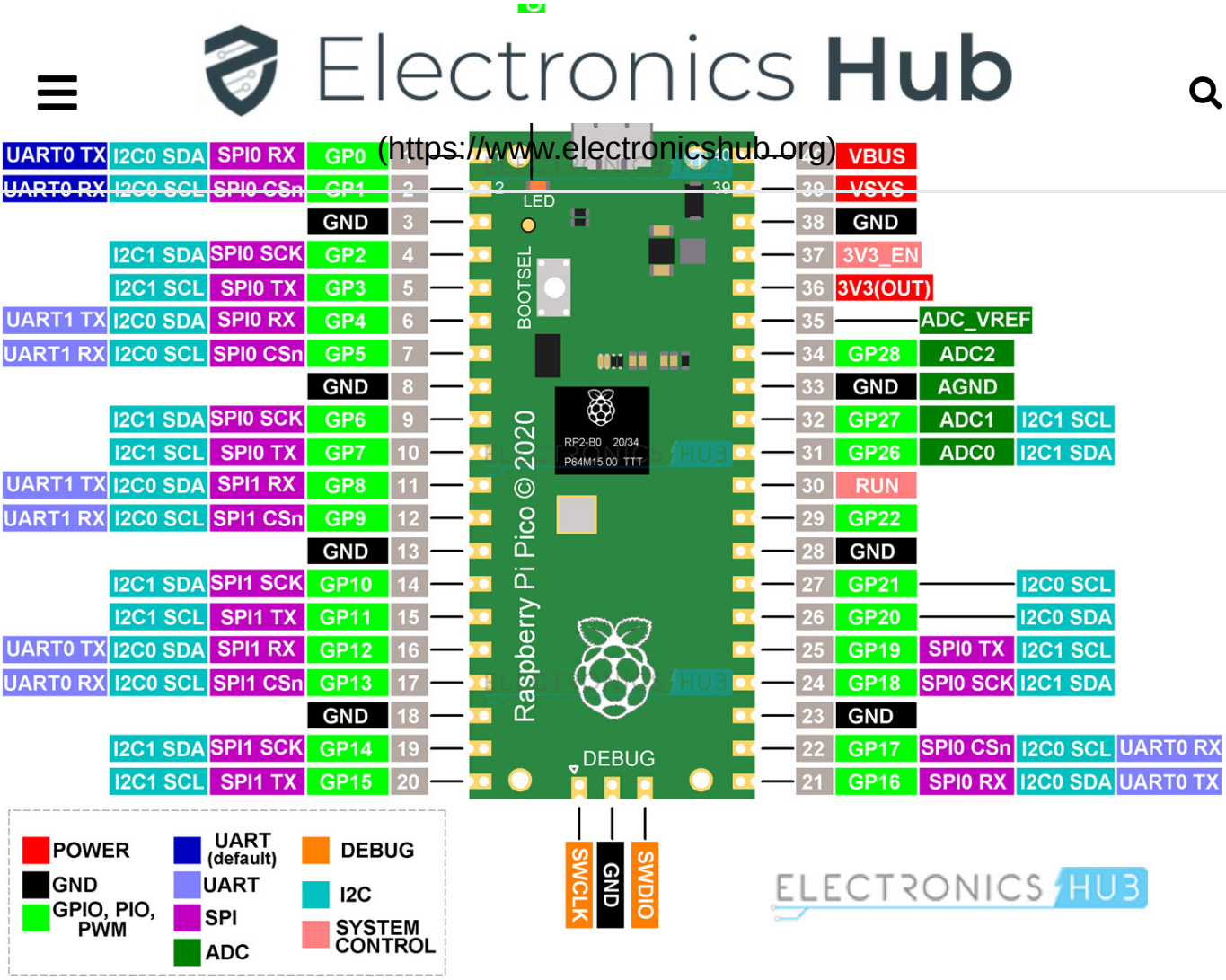
(https://www.electronicshub.org/wp-content/uploads/2021/03/Pico-C-Load-Blink-2.jpg)

Bonus: Alternative to Unplugging and Plugging USB Cable

Are you tired of unplugging and plugging the Raspberry Pi Pico to put into bootloader mode? Then I will show you a work around so that you don't have to do that every time you want upload a program to Pico.

What we are actually doing when unplugging and plugging Pico is resetting it. So, if we can find another way to reset Pico, then we can keep the Pico plugged in to the computer. As it turns out, there is rather a simple way to reset Pico.

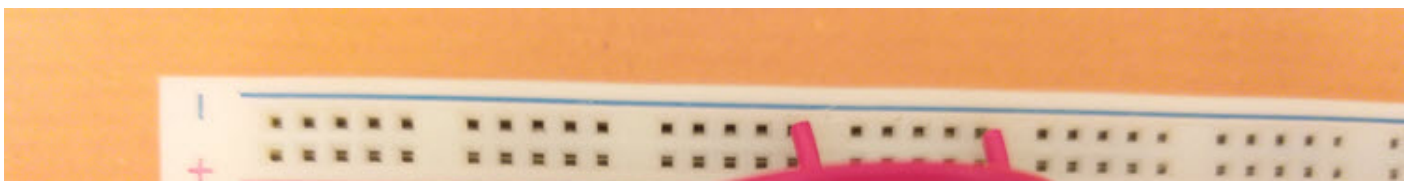
If you remember the Pinout of Raspberry Pi Pico, there is a pin called 'RUN'. This is pin number 30 on Raspberry Pi Pico.

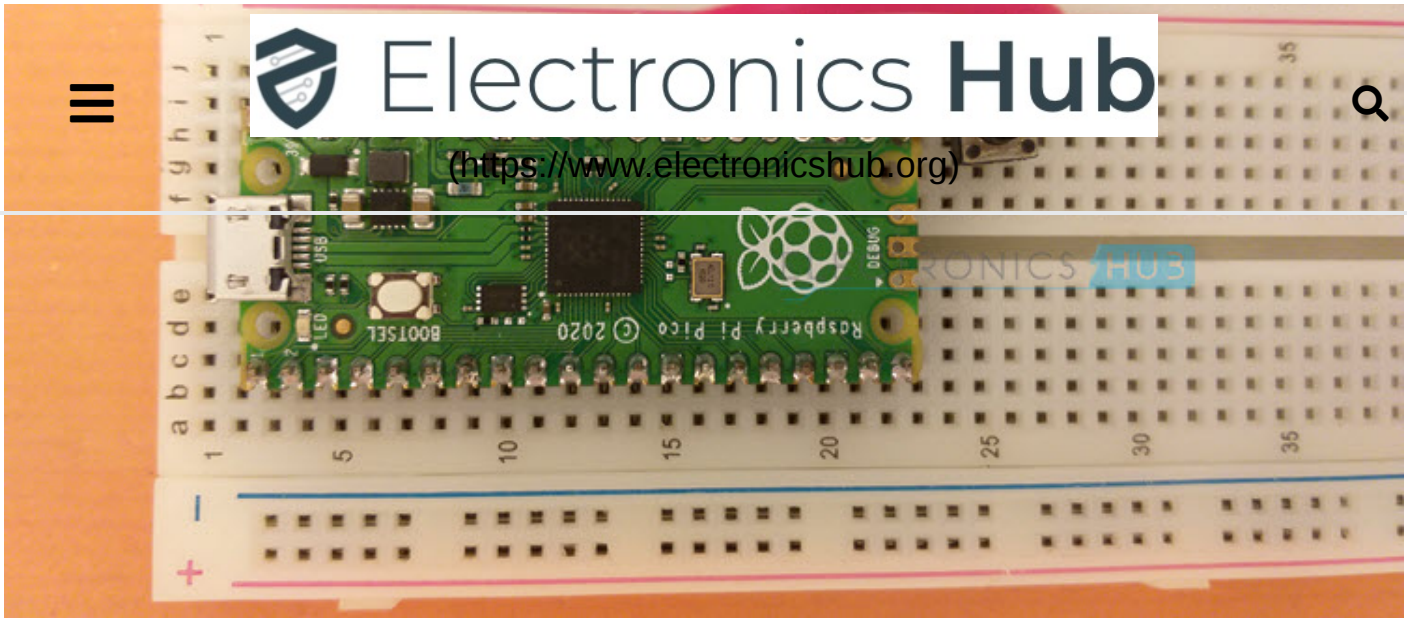


(https://www.electronicshub.org/wp-content/uploads/2021/02/Raspberry-Pi-Pico-Pinout.jpg)

This is actually the global reset pin of RP2040 Microcontroller. When this pin is pulled LOW, the RP2040 microcontroller will reset.

So, I connected a push button between RUN pin and GND. Whenever I push the button, it will reset the microcontroller. Let us call this button as Reset Button.





(https://www.electronicshub.org/wp-content/uploads/2021/03/Program-Raspberry-Pi-Pico-using-C-Reset.jpg)

Now, to put the Raspberry Pi Pico in Bootloader Mode, first push and hold the Reset Button. Then, push and hold the BOOTSEL button. Now, first release the Reset button and then after a second, release the BOOTSEL button.

That's it. Your Pico is now in Bootloader Mode. You can keep the USB cable connected to Pico and the host computer (Raspberry Pi) but you can easily reset or select bootloader mode.

If you just want to reset the Pico, simply push the Reset Button once and release it.

Conclusion

A complete beginner's guide to Getting Started with C SDK for Pico. You learned how to Program Raspberry Pi Pico using C Programming Language by understanding how to download the Pico C SDK, install all the necessary tools, build the source code and upload the binary to Raspberry Pi Pico.

6 Responses

Related Po

- How to Setup Raspberry Pi Pico Serial Programming?... (https://www.electronicshub.org/raspberry-pi-pico-serial-programming/)
- Learn How to Create Your Own New Project for... (https://www.electronicshub.org/raspberry-pi-pico-new-project/)
- How to Program Raspberry Pi Pico with Visual Studio Code? (https://www.electronicshub.org/program-raspberry-pi-pico-with-visual-studio-code/)
- Programming Raspberry Pi Pico with MicroPython | A... (https://www.electronicshub.org/raspberry-pi-pico-micropython-tutorial/)
- Learn how to Program and Debug Raspberry Pi Pico with SWD (https://www.electronicshub.org/programming-raspberry-pi-pico-with-swd/)
- Getting Started With Raspberry Pi Pico | An Introduction (https://www.electronicshub.org/getting-started-with-raspberry-pi-pico/)

Rudolf says: March 6, 2021 at 7:08 am (https://www.electronicshub.org/program-raspberry-pi-pico-using-c/#comment-2425442)

Hello, I can follow this tutorial until “git submodule update –init”. With this command I get only some git help, but no command execution.
What can be wrong?

Reply

↪ **Ravi Teja** says: March 6, 2021 at 7:08 am (https://www.electronicshub.org/program-raspberry-pi-pico-using-c/#comment-2425444)

It should be git submodule update ‘doubledash’init
git submodule update (-)(-)init without paranthesis.
There’s problem with text editor.

Reply

↪ **Ravi Teja** says: March 6, 2021 at 7:15 am (https://www.electronicshub.org/program-raspberry-pi-pico-using-c/#comment-2425446)

Take a look at the image next to the command.

Reply

john black says: March 11, 2021 at 6:38 am (https://www.electronicshub.org/program-raspberry-pi-pico-using-c/#comment-2425951)

There is an error in https://devconnected.com/how-to-add-and-update-git-submodules/ (https://devconnected.com/how-to-add-and-update-git-submodules/) git submodule update –init should read git submodule update –init –recursive. Otherwise thank you for a great article.

Reply

David Francis says:

April 7, 2021 at 11:22 am (https://www.electronicshub.org/program-raspberry-pi-pico-using-c/#comment-2428312)

I was hoping to find out how to load the SDK on to a Linux Mint computer, any suggestions?

Reply

↪ **Ravi Teja** says:

April 8, 2021 at 2:34 am (https://www.electronicshub.org/program-raspberry-pi-pico-using-c/#comment-2428355)

The installation on any Linux machine should be the same. You can follow the same steps for Linux Mint as well. Since Mint is based on Ubuntu, you might have to install an additional library “libstdc++-arm-none-eabi-newlib” (after installing the toolchain).

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Electronics Hub

(https://www.electronicshub.org)

Email *

Website

Post Comment

Electronics Hub

Electronics Hub

(https://www.electronicshub.org)

(https://www.electronicshub.org)

Get Our Latest Newletters

Get Great Content That You Love.
No Ads Or Spams. We Promise.

Enter your email **Sign Up**

Your Privacy Is Important To Us

General ()

Projects ()

Projects ()

Tutorials ()

(https://www.electronicshub.org)

Contact
(https://www.electronicshub.org/contact/)