

\*\*\*\*\*Draft\*\*\*\*\*

**Pico Tools**  
**SDK OPENOCD**  
**11/08/21**

\*\*\*\*\*Draft\*\*\*\*\*

```
/opt/  
pico-extras  pico-sdk  
pico-examples pico-playground
```

Backups:

```
devel@mypi3-16:/media/devel/1b763776-4e1d-499c-9f24-a116a58c161f $ ls pico/  
installed-openocd081421-71510a.img  pico-examples.img  pico-playground.img  
openocd081421-71510a.img           pico-extras.img    pico-sdk.img
```

```
cd opt  
sudo unsquashfs -d pico-sdk pico-sdk.img  
sudo unsquashfs -d pico-examples pico-examples.img
```

Openocd was download from “<https://github.com/raspberrypi/openocd.git>”  
An img file “openocd081421-71510a.img” was created of the

```
commit 71510a77a61c6eb2b1266e00010f8f258785a54b (grafted, HEAD -> rp2040, origin/rp2040,  
origin/HEAD)  
Author: Peter Lawrence <12226419+majbthrd@users.noreply.github.com>  
Date: Thu Jun 3 14:06:09 2021 -0500
```

```
tcl/boards: add pico-debug.cfg
```

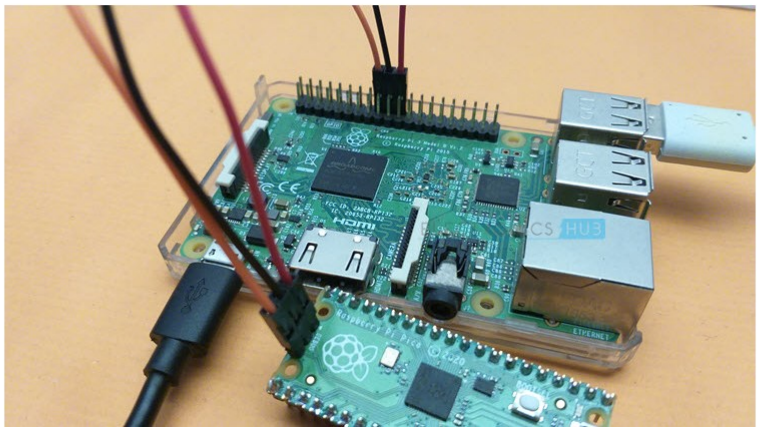
The folder “/home/devel/local/openocd” was created from the file “installed-openocd081421-71510a.img” in “/home/devel” and executing the command “sudo unsquashfs -d openocd ../installed-openocd081421-71510a.img” in “/home/devel/local” folder.

```
ls -la local/openocd/bin/openocd
```

| Rpi   | Pico          |
|---|---------------|
| SWDIO----18 red-----black-----black-----    | blue----swclk |
| grd-----20 yellow----red-----blue-----      | green---grd   |
| SWCLK---22 orange-----orange-----green----- | black---swdio |

The following table shows all the necessary connections between Raspberry Pi and Raspberry Pi Pico that you need to make.

| Raspberry Pi Pico | Raspberry Pi     |
|-------------------|------------------|
| SWDIO             | GPIO 24 (PIN 18) |
| SWD GND           | GND (PIN 20)     |
| SWCLK             | GPIO 25 (PIN 22) |



How to Program and Debug Raspberry Pi Pico C/C++ Getting started with Raspberry Pi Pico

https://www.electronicshub.org/programming-raspberry-pi-pico-with-swd/


# Electronics Hub

LEARN TUTORIALS DIY PRODUCT REVIEWS GUIDES & HOWTO ABOUT

## Raspberry Pi Pico SWD Programming and Debug

Like all ARM Cortex processors, the Raspberry Pi Pico also has dedicated hardware for debugging via the SWD Interface. The two wires required for SWD Debugging are called SWDIO (bidirectional SWD Data) and SWCLK (SWD Clock).

On the Raspberry Pi Pico, the SWD Pins are separated from the rest of the GPIO Pins and are placed at the bottom of the Board.



The 2-wire SWD Interface of RP2040 on the Raspberry Pi Pico board allows you to do the following:

- Upload program into External Flash or Internal SRAM.
- Control the state of execution of the processor i.e., run, halt, step, set breakpoints, etc.
- Access processor memory and I/O peripherals (which are memory mapped) through the system bus.

Pico The labels swclk grd swdio are only on the bottom side.

pico-pi-sdk-build.txt has the instruction from the video below.

Revisit the video below

<https://www.youtube.com/watch?v=UZwq3eb5My0>

Downloaded the getting-started

<https://datasheets.raspberrypi.com/pico/getting-started-with-pico.pdf>

```
export PATH=~/.local/openocd/bin:$PATH
```

```
echo $PATH
```

```
/home/devel/.local/openocd/bin:/home/devel/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games
```

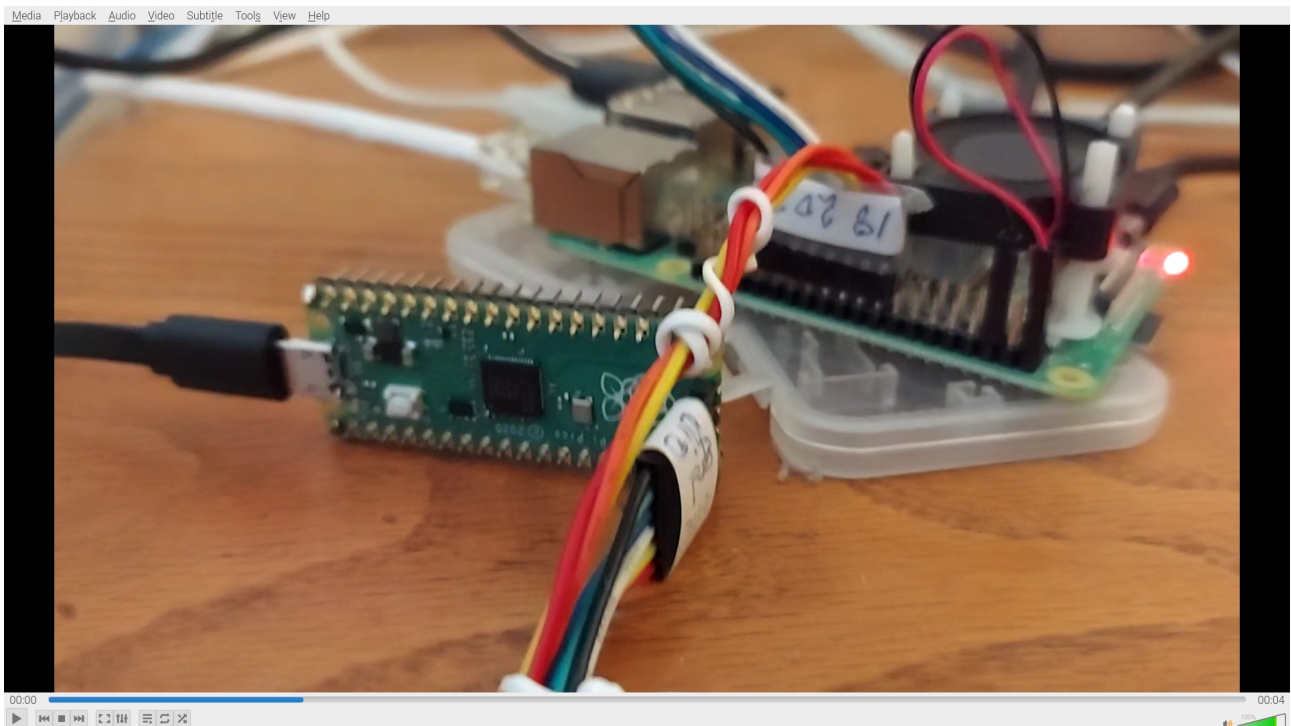
```
which openocd
```

```
/home/devel/.local/openocd/bin/openocd
```

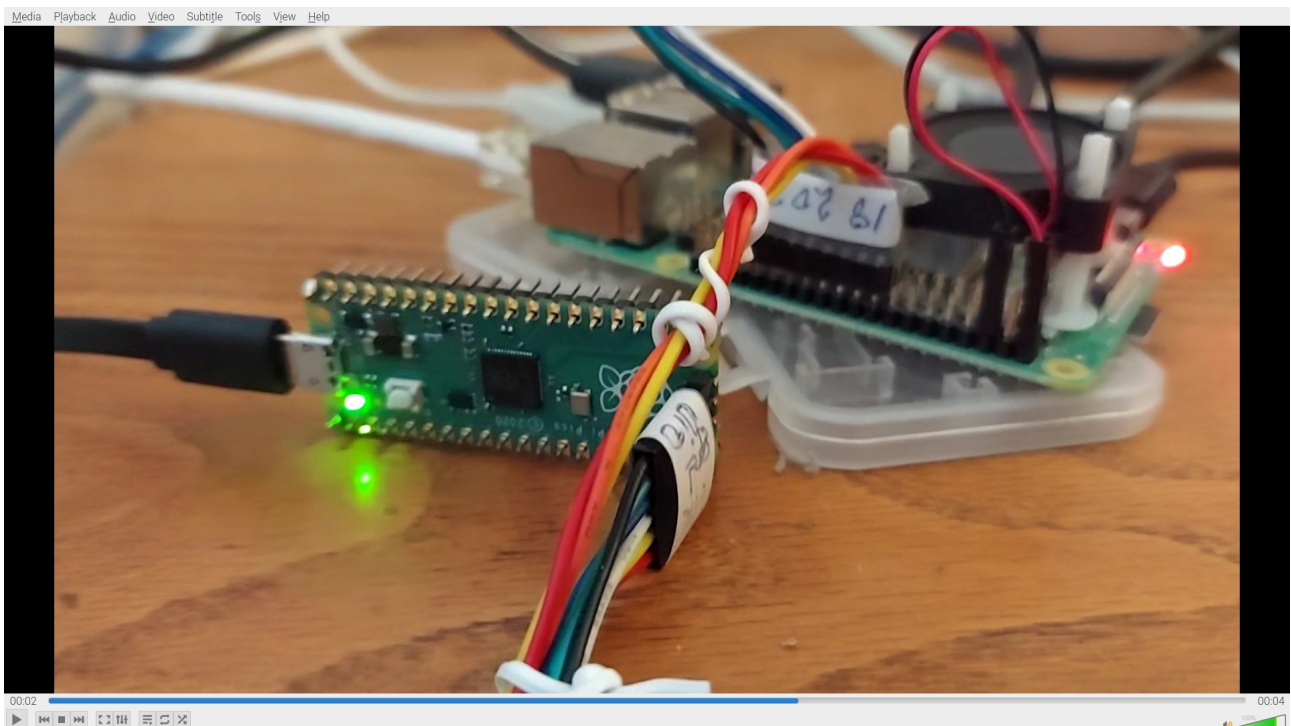
```
/opt/pico-examples/build
```

```
openocd -f interface/raspberrypi-swd.cfg -f target/rp2040.cfg -c "program blink/blink.elf verify reset exit"
```

Raspberry Pico will start to blink.



Blink on.



Using openocd to start the hello world. The window on the left is running minicom. Window on the right loads the hellp\_usb program ***openocd -f interface/raspberrypi-swd.cfg -f target/rp2040.cfg -c "program hello\_world/usb/hello\_usb.elf verify reset exit"***

The screenshot displays two terminal windows from a Linux environment.

The left terminal window has tabs labeled "File Edit Tabs Help". It contains the following output:

```
Hello, world!  
Hello, world!  
Hello, world!  
Hello, world!  
Hello, world!  
Hello, world!  
Hello, world!  
Hello, world!  
Hello, world!
```

The right terminal window also has tabs labeled "File Edit Tabs Help". It shows the execution of a program called `verify` located at `./opt/pico-examples/build`. The output includes several error-like messages indicating target halts due to debug requests, followed by successful completion messages:

```
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
** Programming Finished **  
** Verify Started **  
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
target halted due to debug-request, current mode: Thread  
XPSR: 0x00000000 pc: 0x00000178 msp: 0x20041f00  
** Verified OK **  
** Resetting Target **  
Shutdown command invoked  
dev@lmyp3-20:/opt/pico-examples/build $
```