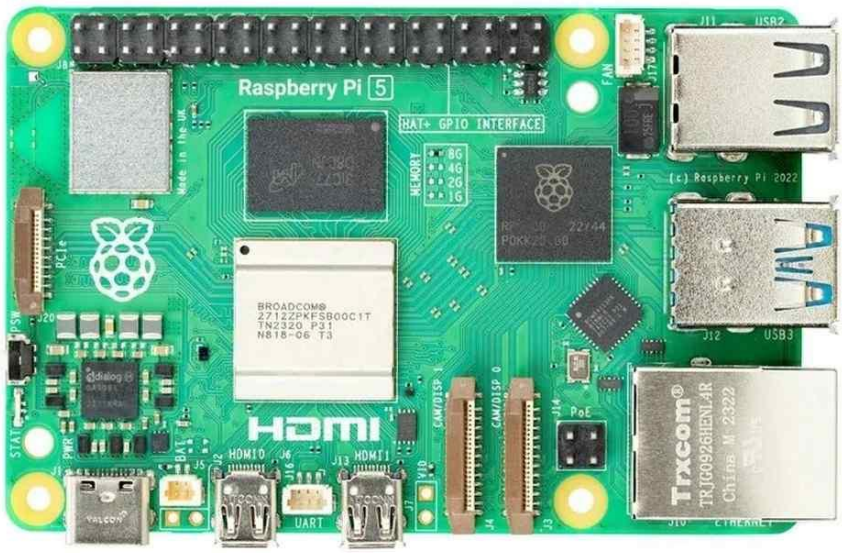
Raspberry Pi Tutorial

Using RPi.GPIO calls on a Raspberry Pi Model 5 or Bookworm OS



Bob Rathbone Computer Consultancy

www.bobrathbone.com

Version 2.0

27th of September 2024

Contents

Figures

Introduction

The Raspberry Pi Model 5 was introduced at the end of 2023. It only works with **Raspberry Pi Bookworm OS** or later.

However, the biggest impact for most developers is that the **RPi.GPIO** input/output library does not work on the **Raspberry Pi model 5** or on systems running **Bookworm**. This is because the **RPi Model 5** now has a separate chip called **RP1** for controlling I/O including the pins on the GPIO header (**j8**). In the case of Bookworm, the authors of **RPi/GPIO** have apparently not yet updated their software. This means that hundreds of thousands of programs or maybe even millions of programs need to be modified to use one of the newer libraries such as **gpiod** or **lgpio**. The **RP1** chip also controls USB ports, Gigabyte Ethernet, MIPI Camera Controllers and Low Speed Peripherals compatible with earlier versions of the Raspberry Pi.

My own product, the **Raspberry Pi Internet Radio** is also such a program and would have meant a lot of work to convert all the GPIO routines to say **GPIOD** which does run on the RPi Model 5. So, I decided to write a simple interface called **GPIOconverter** which converts **RPi.GPIO** calls to one of the newer GPIO interfaces. This is a so-called **software shim**. See the following link for more information: **https://en.wikipedia.org/wiki/Shim_(computing)**. **GPIOD** was advocated as the best way forward however I found that GPIOD was poorly documented and there didn't seem to be any examples of how to handle interrupts. I eventually settled on using the excellent **python3-lgpio** library for the **GPIOconverter** software. The architecture of the interface is shown below:

OUTPUT: User Program --> GPIO calls --> GPIOconverter --> LGPIO

INPUT: LGPIO events --> GPIOconverter --> User Program

The following illustration shows the location of the RPP1 I/O chip.

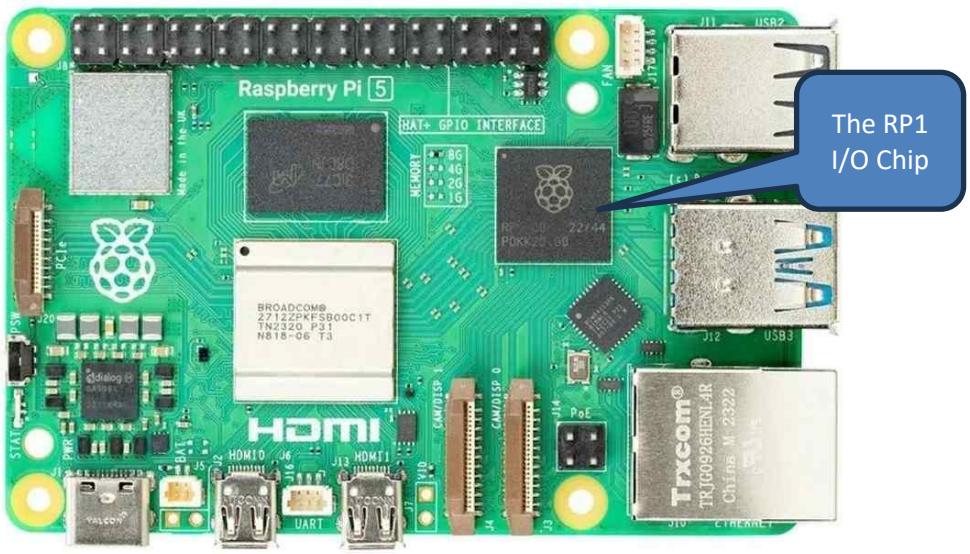


Figure 1 The Raspberry Pi Model 5 RP1 I/O chip

See *Appendix B - The RP1 general purpose I/O Chip* on page 10 for more information.

GPIO Hardware Notes

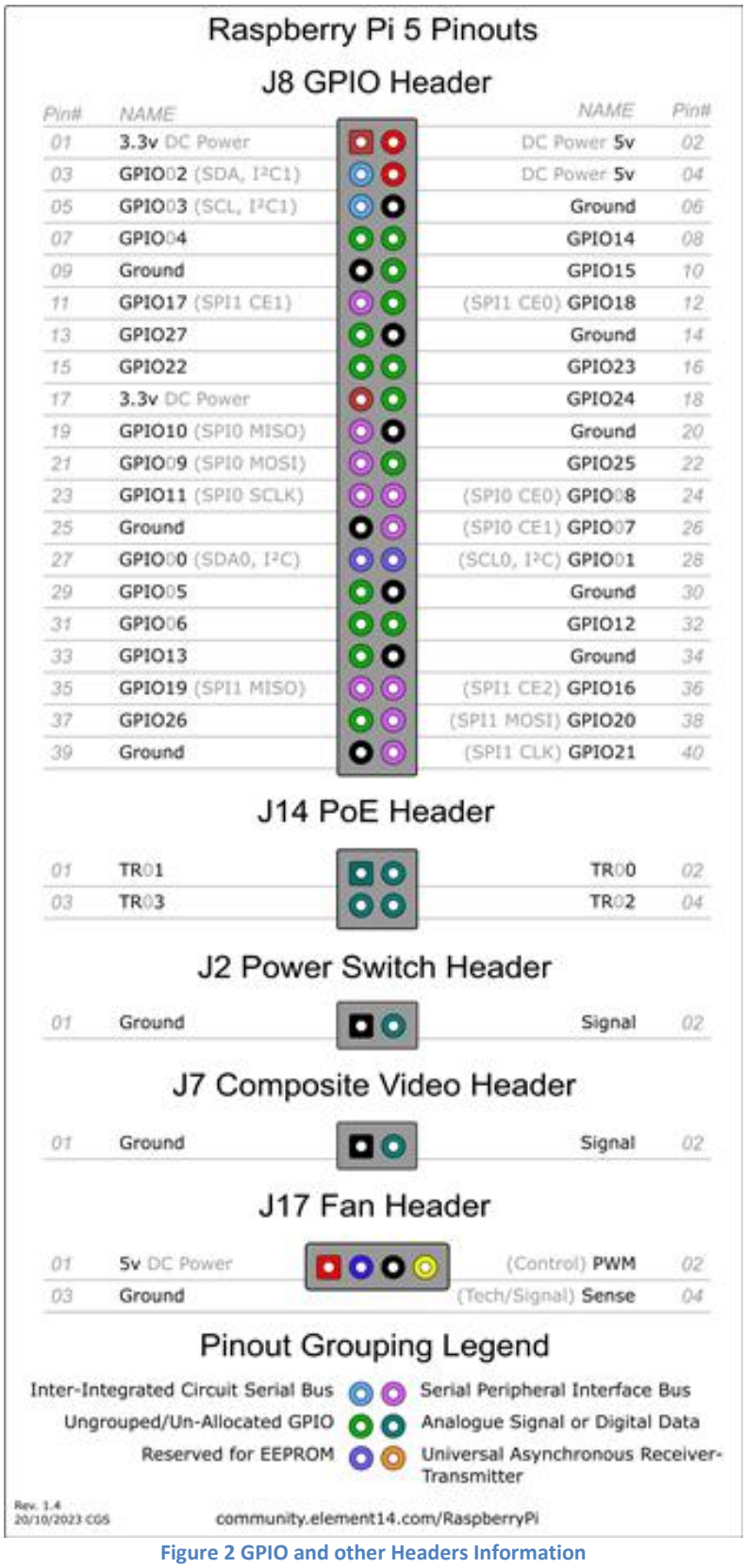


Figure 2 GPIO and other Headers Information

Conventions used in this tutorial

Installation of the radio program requires you to enter lines at the command line prompt. This requires you to log into the Raspberry PI as user '**pi**'. The default password is **raspberry**.

Note: Don't carry out any of the following commands just yet. They are just examples.

```
Raspberrypi login: pi
Password: raspberry
pi@raspberrypi:~$ Last login: Sun Apr  6 10:18:18 2014 from 192.168.2.100
pi@raspberrypi:~$
```

The prompt line is displayed ending with a **$** sign. The **pi@raspberrypi:~** string means user 'pi' on host machine called 'raspberrypi'. The ~ character means the user 'pi' home directory **/home/pi**. In this tutorial if you are required to do something as user **pi** then only the $ sign will be shown followed by the command as shown in the example below:

```
$ pinout
```

Some commands produce output which does not need to be shown. In such a case a ':' is used to indicate that some output has been omitted.

```
$ pinout
Description        : Raspberry Pi 5B rev 1.0
Revision           : c04170
: {Output ommited}
J8:
   3V3  (1) (2)  5V
 GPIO2  (3) (4)  5V
 GPIO3  (5) (6)  GND
 GPIO4  (7) (8)  GPIO14
   GND  (9) (10) GPIO15
GPIO17 (11) (12) GPIO18
GPIO27 (13) (14) GND
GPIO22 (15) (16) GPIO23
   3V3 (17) (18) GPIO24
GPIO10 (19) (20) GND
 GPIO9 (21) (22) GPIO25
GPIO11 (23) (24) GPIO8
   GND (25) (26) GPIO7
 GPIO0 (27) (28) GPIO1
 GPIO5 (29) (30) GND
 GPIO6 (31) (32) GPIO12
GPIO13 (33) (34) GND
GPIO19 (35) (36) GPIO16
GPIO26 (37) (38) GPIO20
   GND (39) (40) GPIO21
For further information, please refer to https://pinout.xyz/
```

END OF EXAMPLE COMMANDS.

Which OS works with my Raspberry Pi model

The following article contains a table showing which model Raspberry Pi's work with which Raspberry Pi OS:
https://en.wikipedia.org/wiki/Raspberry_Pi_OS

RPi.GPIO.py

The **GPIO.py code** is only for use with the **Raspberry Pi Model 5** or for earlier models such as the **Model 3B** or **4B** which are running the **Bookworm OS** or later. The code is designed to intercept traditional GPIO calls and convert them to LGPIO calls. See: https://abyz.me.uk/lg/py_lgpio.html

NOTE: On the **15th of March 2024** the **Raspberry Pi Foundation** released a new version of the **Bookworm** Operating System Firmware which completely broke the input functionality of the GPIO header pins. The error manifests itself as "Error: GPIO *nn* Failed to add edge detection" where *nn* is the GPIO number of the relevant pin. This means that it became necessary to either install **Bullseye OS** or to install an older release of **Bookworm** from December 2003 which has older firmware. However, the **GPIO.py** program now supports the **gpiochip0** circuitry embedded in the BCM processor. Once **GPIOconverter** is enable it intercepts calls to RPi.GPIO and converts them to **lgpio** calls. This only works on **Bookworm** or later. Earlier versions of the OS such as **Bullseye** continue to use the normal **RPi.GPIO** calls as normal.

Installation

Pre-requisites

Install package **python3-lgpio**

```
$ sudo apt install python3-lgpio
```

Don't include the "$" sign in the command you enter.

Downloading GPIOconverter

Log into the Raspberry Pi Model 5 and clone the **GPIOconverter** software and run:

```
$ cd
$ sudo apt install git
$ git clone https://github.com/bobrathbone/GPIOconverter
```

This creates a directory with the following files and directory RPi.

```
$ ls GPIOconverter/
Docs  README.md  RPi test_pwm.py
```

The **RPi** sub-directory contains the actual GPIOconverter files:

```
$ ls GPIOconverter/RPi/
GPIO.py  __init__.py  README
```

Installation

There are two methods of installing **GPIOconverter** software:

1) In the same directory as your GPIO program as a local package
2) Install as a systemwide package usually in **/usr/lib/python3/dist-packages/RPi**

Method 1 means that only your local **RP.GPIO** program(s) installation will be using **GPIOconverter**. All other GPIO programs will use the standard GPIO library in **/usr/lib/python3/dist-packages/RPi**

Installing GPIOconverter locally

Create a sub-directory called RPi in the directory where your GPIO code is installed. For example, for code in directory **/usr/share/radio**:

```
$ cd /usr/share/radio
$ mkdir RPi
```

Now copy the already downloaded **GPIOconverter** package files to **/usr/share/radio/RPi/**

```
$ cp ~/GPIOconverter/RPi/* /usr/share/radio/RPi/.
```

Enabling and disabling GPIOconverter

The package is active or inactive depending upon the presence of the **__init__.py** file. Normally this file will be present by default. For a Raspberry Pi model 5 only or earlier models running Bookworm OS. For example, for code found in the **/usr/share/radio/** directory:

```
$ touch /usr/share/radio/RPi/__init__.py
```

The instruction above will cause the code using the GPIO calls to see directory **RPi** as a package. For earlier models such as the 3B or 4B disable the package by removing the **__init__.py** file:

```
$ rm /usr/share/radio/RPi/__init__.py
```

Installing GPIOconverter as a systemwide package

Method 2: This has the advantage that all **RPi.GPIO** programs will use the **GPIOconverter** package wherever they are installed. The standard (and faulty) **RPi.GPIO** package is usually installed in the **/usr/lib/python3/dist-packages/RPi** directory. It is first necessary to move this out of the way and create a new **RPi** sub-directory.

```
$ cd /usr/lib/python3/dist-packages/
$ sudo mv RPi RPi-old
$ sudo mkdir RPi
```

Now make copy the **GPIOconverter** package files into the new **RPi** sub-directory.

```
$ cp ~/GPIOconverter/RPi/* /usr/lib/python3/dist-packages/RPi/.
```

All RPI.GPIO programs will now access the new RPi.GPIO package.

Known Issues

GPIO.setwarnings call limitations

The call **GPIO.setwarnings(True|False)** is currently not implemented in a compatible manner in **lgpio**. The **lgpio** package does not have the equivalent of **GPIO.setwarnings** but has the **lgpio.exceptions** call which can be set to True or False. So, the **setwarnings** call can either be ignored or be used to enable and disable lgpio exceptions. Set **IGNORE_WARNINGS** to **True** at the beginning of the **GPIO.py** program to prevent **lgpio exceptions** or control this with the **GPIO.setwarnings** call.

Using GPIOconverter with Rotary Encoders

There is a lot of Python software for Rotary Encoders which was originally written by Ben Buxton in 2011. You may find that the Rotary Encoder is sluggish and misses a high number of turns. In the case of the Ben Buxton code this can be corrected by changing the **HALF_STEP** flag from **False** to **True**.

```
# Enable this to emit codes twice per step.
# HALF_STEP == True: emits a code at 00 and 11
# HALF_STEP == False: emits a code at 00 only
HALF_STEP      = True
STATE_TAB = HALF_TAB if HALF_STEP else FULL_TAB
```

Fortunately, the above change doesn't seem to affect operation when running the code using the standard **RPi.GPIO** calls on say a Raspberry Pi Model 4B.

Support

It is not possible to provide support for the standard GPIO library as literally hundreds of thousands of programs are using GPIO routines. The code is provided as is and without any warranties or "fit for purpose" etc. However, do contact bob@bobrahbone.com for any errors or missing features in **GPIOconverter** on **Raspberry Pi Model 5** only.

Source files

The software is stored on **GitHub** at https://github.com/bobrathbone/GPIOconverter or is available as an archive (tar) for download from:
https://bobrathbone.com/raspberrypi/packages/GPIOconverter.tar.gz

The GPIO.py shim software

The **GPIO.py** file uses the Python 3 LGPIO library (python3-lgpio) to handle calls to and from the RP1 i/o chip. More information on LGPIO see: https://abyz.me.uk/lg/py_lgpio.html

Python code examples will be found at: https://abyz.me.uk/lg/examples.html#Python%20lgpio

The test_pwm.py program

The **test_pwm.py** program tests the **PWM** (Pulse-Width Modulation) function of **RPi.GPIO** which, which is a technique for creating variable-width pulses to encode or modulate a signal. PWM is used in many applications, including power supplies, DC-DC switching regulators, and DC motor control or in this case to brighten and dim a LED. To use it, amend the "led = 16" statement in the source code to use a different GPIO other than 16. Connect the +V pin of the LED to the GPIO pin being used and the other pin to GND (0-volts). Then run the program.

```
$ ./test_pwm.py
```

The LED should continuously brighten and dim the LED.

Appendix A Licences

The software and documentation for this project is released under the GNU General Public Licence.

The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software. Software that ensures that these rights are retained is called free software. The license was originally written by Richard Stallman of the Free Software Foundation (FSF) for the GNU project.

The GPL grants the recipients of a computer program the rights of the Free Software Definition and uses *copyleft* to ensure the freedoms are preserved whenever the work is distributed, even when the work is changed or added to. The GPL is a *copyleft* license, which means that derived works can only be distributed under the same license terms. This is in distinction to permissive free software licenses, of which the BSD licenses are the standard examples. GPL was the first *copyleft* license for general use.

See http://www.gnu.org/licenses/#GPL for further information on the GNU General Public License.

The licences for the source and documentation for this project are:

GNU General Public License.	See http://www.gnu.org/licenses/gpl.html
GNU AFFERO General Public License.	See http://www.gnu.org/licenses/agpl.html
GNU Free Documentation License.	See http://www.gnu.org/licenses/fdl.html

Acknowledgements

The people at https://abyz.me.uk/lg/index.html. who produced the **lgpio** package. No individuals are mentioned by name on their Web site but whoever they are, they have made an excellent product for General Purpose Input Output control on Linux Single Board Computers such as the Raspberry Pi Model 5 with extremely professional documentation. My compliments.

The GitHub member known only by the handle **fgmnts** who kindly implemented PWM support in **GPIOconverter**. See https://github.com/fgmnts

Appendix B - The RP1 general purpose I/O Chip

The RP1 general purpose I/O chip is a 12×12mm, 0.65mm-pitch BGA southbridge, which provides the majority of the I/O capabilities for Raspberry Pi 5.

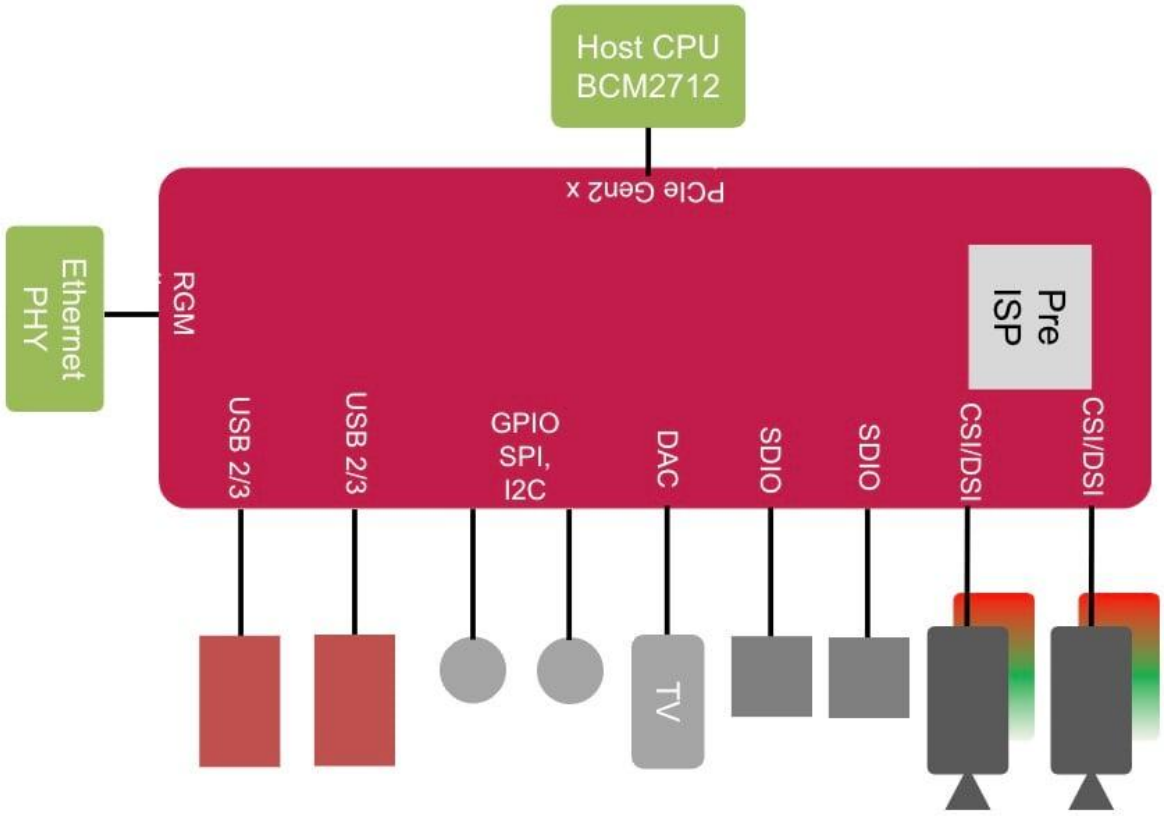


Figure 3 The RP1 General Purpose I/O Chip

The RPi chip provides:

- 4-lane PCIe 2.0 endpoint
- Gigabit Ethernet MAC
- 2× USB 3 host controllers
    - Each has 1× USB 3 and 1× USB 2 port
    - More than twice the usable USB bandwidth vs. Raspberry Pi 4
- 2× SDIO ports/eMMC (not used on Raspberry Pi 5)
- 2× MIPI transceivers (4-lane, supporting DSI and CSI-2)
- Video DAC (3-channel, supporting PAL/NTSC and VGA)
    - Only one channel (composite) used on Raspberry Pi 5
- Low-speed peripherals (SPI, UART, I2C, PWM, GPIO, I2S)
- Delta-sigma PWM audio out

More information on RP1 can be found in the RP1 Peripherals document.
See https://datasheets.raspberrypi.com/rp1/rp1-peripherals.pdf

Glossary

BGA Ball Grid Array – A popular surface mount for Integrated Circuits (ICs)

CSI Camera Serial Interface

DAC Digital to Analogue Converter (In this case for audio output cards)

DSI Display Serial Interface

GND Ground, 0 Volts

GPIO General Purpose IO (On the Raspberry PI)

I2C Two-wire serial communication protocol

I2S Electrical serial bus for connecting digital audio devices

PWM Pulse-Width Modulation

RP1 Input Output Controller Chip for Raspberry Pi Model 5 peripherals

SPI Serial Peripheral Interface – Interface us between digital components

USB Universal Serial Bus