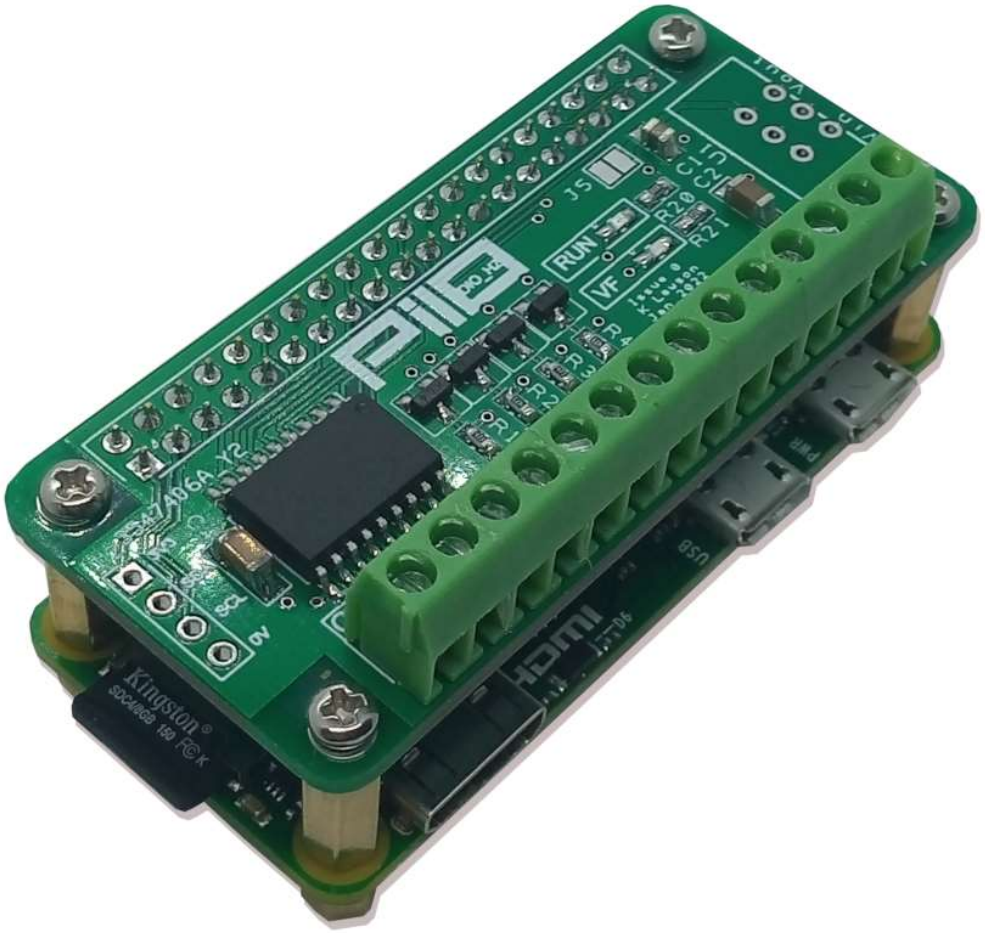


PiIO-DIO-HZ

Digital input / output board for raspberry Pi

User's Manual

Document Change Register				
Revision	**Date**	**Author**	**Change Description**	**Section**
0	Feb 2022	K Lawson	Initial revision	

1 Introduction

The PIIO DIO-HZ PCB sits on top of a raspberry PI Zero PCB and can be used to interface it to light industrial and test / measurement / hobby / home automation applications. The board features 4 high side inputs and 8 outputs that can be used to perform these tasks.

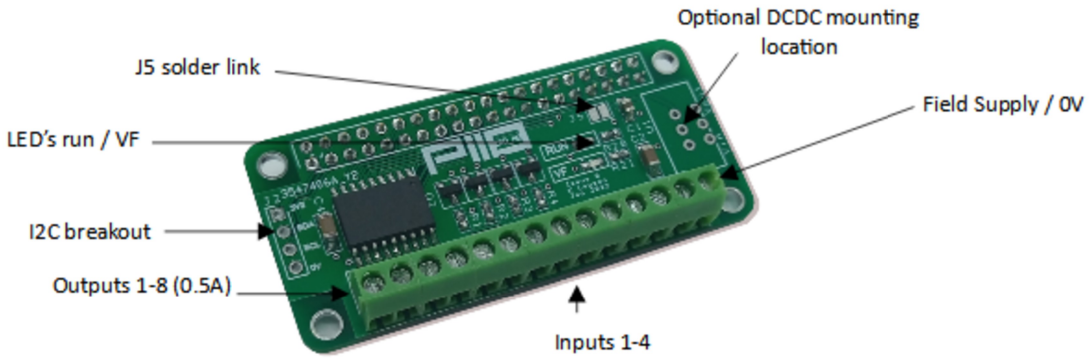


Figure 1: PCB layout

2 Powering the board

The board is powered via the 14 way connector block J1. How this is used depends on the board option you have purchased. Note solder link J5 which is open by default.

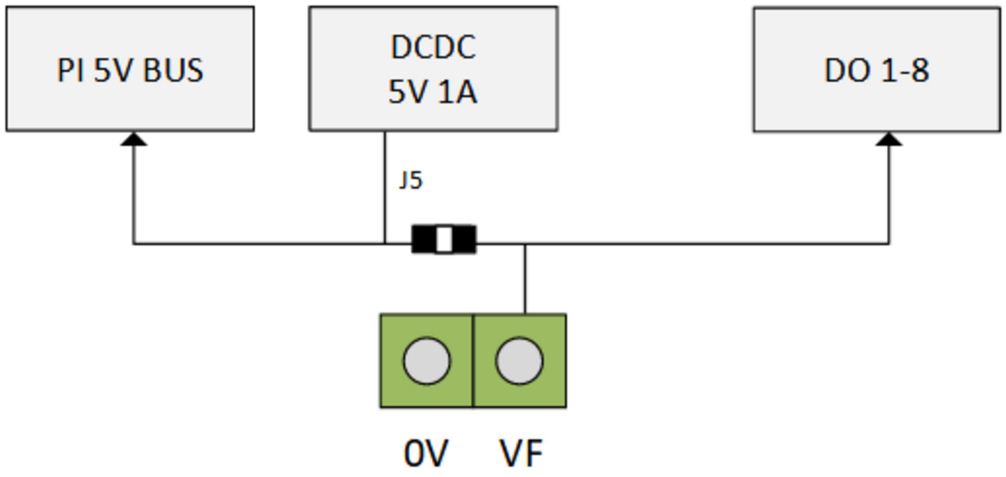
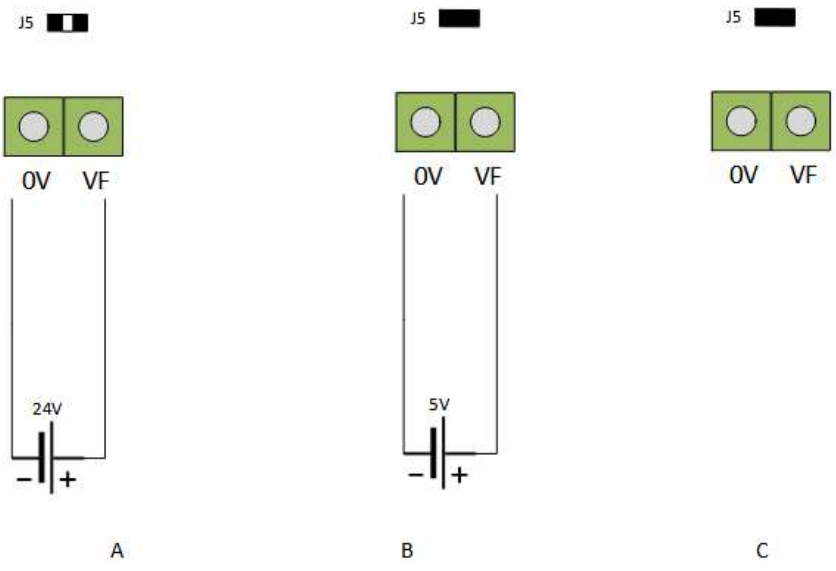


Figure 2: Power supply topology

2.1 DCDC Not fitted

If you have not chosen to not have the optional 5V DCDC fitted to the board then you need to power your Pi Zero with a micro USB or you can power the Pi and the digital outputs off a single 5V supply.



In the above figure illustrates three powering options:

A/ Pi powered by Micro USB (5V), J1 field supply powers digital outputs (24V).

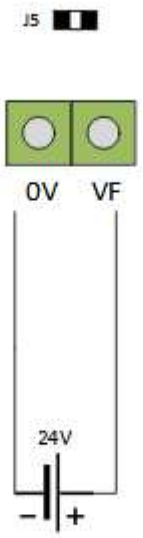
B/ Pi and digital outputs powered by external 5V power supply to J1, Note – digital outputs must run at 5V.

C/ As B but we use a micro USB to power everything rather than the terminals.

Note – the 5V terminal is connected straight to the PI power rail, any over-voltage will likely damage the Pi.

Note - Make sure you know what you are doing when you are powering up your device as there is no reverse polarity or back-feeding protection on these terminals. Always check with a multi meter before you apply power and preferably check voltages with PI disconnected first.

2.2 DCDC fitted



A

If you have chosen to have the DCDC fitted then 0V / VF terminals power the digital outputs and the onboard DCDC powers your Pi zero – no other power connections are needed.

2.3 Field supplies

The field supply input is used to power the digital outputs and the optionally fitted on-board DCDC for the PI.

- VF – Powers output 1-8 and the on board 5V DCDC

The Link J5 may be soldered which joins VF and the PI 5V bus together – this may be useful if you want to run everything at 5V.

Note – Be sure you understand exactly what you are doing if soldering J5, you are connecting the field and 5V bus, they cannot be at different voltages.

2.4 On-board DCDC

A 1A or 2A DCDC Power supply is optionally fitted to the board to power the PCB.

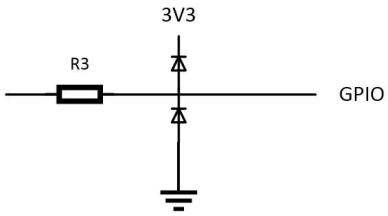
2.5 LEDs

There are two LEDs on the board.

- **VF** – Indicated Field supply VF1 is powered.
- **Run** – software controlled to a GPIO Output, generally set to pulsing to indicate the program is running.

3 Digital inputs

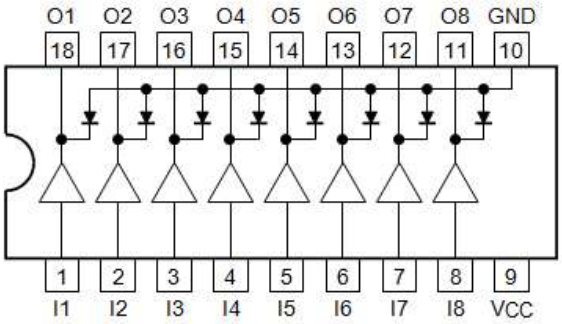
A simple clamp circuit allows digital inputs to be interfaced:



The inputs are designated as DI1-4 and are located at the bottom of the board. The clamp circuit allows the GPIO to read voltages 2-24V.

4 Digital outputs

The digital outputs are controlled via a TBD62783AFWG High side driver IC. This contains DMOS FET driver arrays and operator up to 50V.



The drivers feature the following functionality:

- 35V Max voltage
- Built in diodes for inductive loads
- 8x 500mA High side DMOS drivers
- Total current through device circa 2A (see thermal limitations doc)

4.1 Application thermal considerations

The device is thermally limited in operation (I.e cannot have all channels on max current simultaneously without overheating), the following table provides a guideline to the maximum number of outputs which can be powered at a certain current. This is purely a thermal limit to prevent the device overheating, you will need to validate your thermal position.

Number of outputs on	Max current through each (A)
1	.5
2	.5
4	.33
8	.25

5 Software

The software library is provided at [https://github.com/lawsonkeith/PiIO]. This is a python3 library and is designed to work on linux based systems such as Raspbian.

You can clone this repository and manually install it's dependencies as described in the following video:

https://www.youtube.com/watch?v=gdkUGLT3l4A

A video on how to install the library and set up your Pi can be found here. Note the node red packages are optional.

[https://www.youtube.com/watch?v=CY0j5Y8JfIU&t=7s]

Various example projects are documented here:

[https://www.youtube.com/watch?v=AFOfhobkOLQ]

[https://www.youtube.com/watch?v=7_d0eNJZd10]

[https://www.youtube.com/watch?v=kL8XjM-FGmY]

You will need to perform a number of tasks before your system is ready to use:

1. Update OS
2. Edit nano config file (if you're using nano as an editor)
3. Disable I2C,SPI and Serial, enable SSH in raspi-config
4. Clone the github repository
5. Install required Linux packages
6. Install required python packages
7. Test node red by importing a json flow into it (you don't need node red).

5.1 Basic install

Cones repo, installs some packages needed for the demos then installs they PiIO lib.

- [sudo apt-get update]
- [sudo apt-get upgrade]
- [git clone https://github.com/lawsonkeith/PiIO]
- [cd PiIO]
- [./install_packages.sh]

- [./install_py_packages.sh]
- [sudo python3 setup.py install]
- Use raspi-config to setup interfacing options.

5.2 Software structure

The repository is structured as follows:

- **PiIO** – Fundamental drivers written in python 3
- **Docs** – Markdown documentation
- **Examples/PiIO_H_boards/DIO_HZ** – Contains python3 examples
- **Images** – Contains pictures used in the repository
- **Manuals** – Contains all PDF manuals including this one
- **Install_packages**.sh – installs required linux packages
- **Install_py_packages**.sh – installs required python packages
- **Setup**.py – used to install the PiIO library

5.3 Basic_functs example

This example does not require any hardware but just shows the operation of some of the PiIO utility API.

- Alarm function
- Exponential moving average function
- Scale function
- Rising edge function
- Falling edge function
- Timed pulse function
- Timed on function
- Timed off function

You can run the program [python3 ./basic_fincts.py] and the program will step through and test each of these utility functions.

Definitions of these functions can be found in PiIO/PiIO.py.

5.4 DIO_H_basic example

This program scans the inputs of the board and if one is set sets the corresponding output. Pin 6 is controlled using PWM so outputs at 50% duty cycle. This is a python only program and has no node red user interface.

5.5 DIO_HZ_nodered example

This demonstrates how Node red can be used to provide a user interface via a web browser that can be used to scan inputs and write to outputs on the board. Again proportional control is used to drive the outputs, this example uses mqtt on node-red to communicate to the python program which allows for greater control than just using node-red to control the io direct.

5.6 DIO_HZ_Nodered_direct example

In this example the GPIO nodes of node red are used so that the node red user interface can control the PI GPIO but without having to run a python program on the PI. This provides an easier implementation but reduced functionality over how the IO can be controlled.

6 Certification

This board is intended for either educational use or to be used as a subcomponent. If it is incorporated into a final product then the user is responsible for undertaking any required certifications.