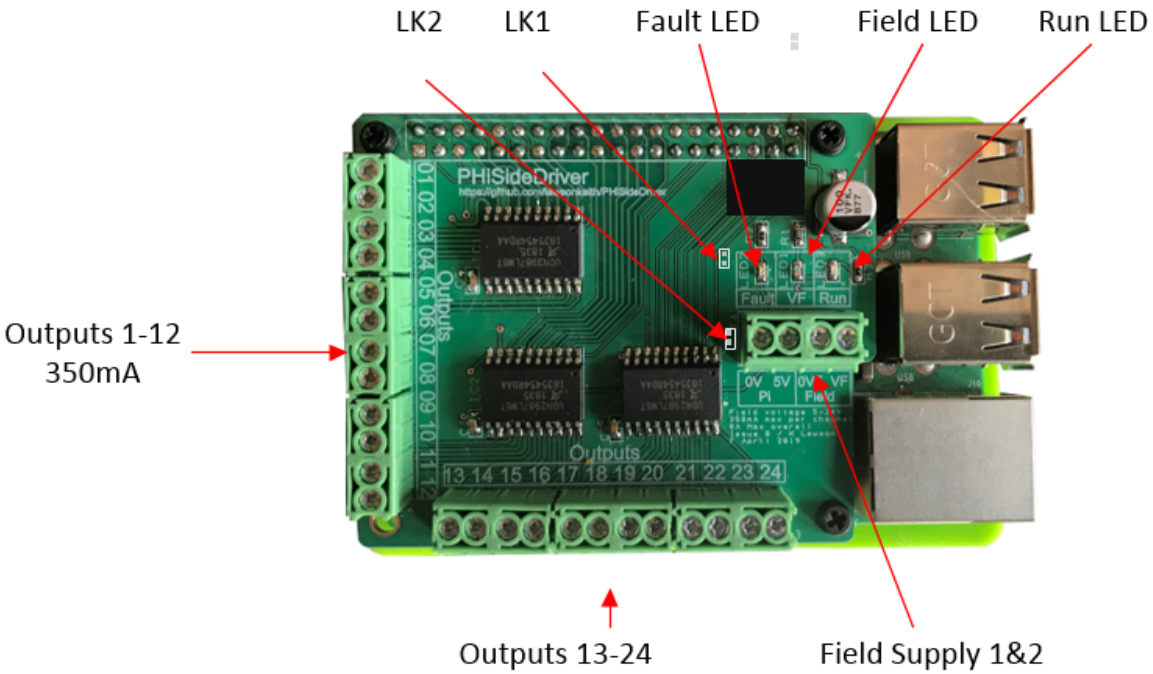
PIO DO24 PCB Manual

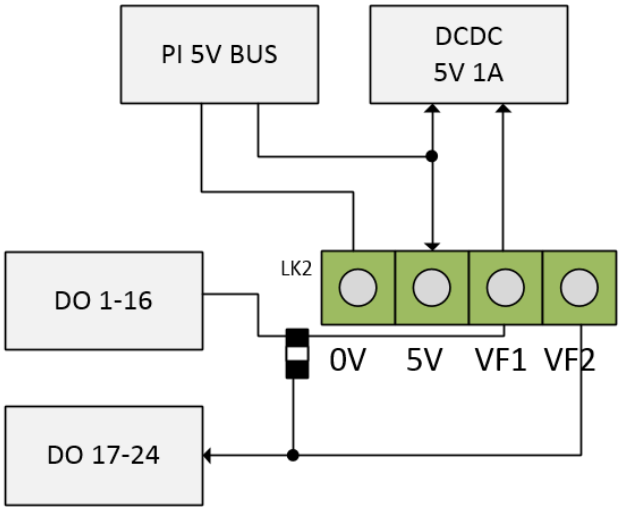
Description

The PIIO DO24 PCB sits on top of a raspberry PI PCB and can be used to interface it to light industrial and test / measurement applications. The board features 24 high side outputs that can be used to perform these tasks.



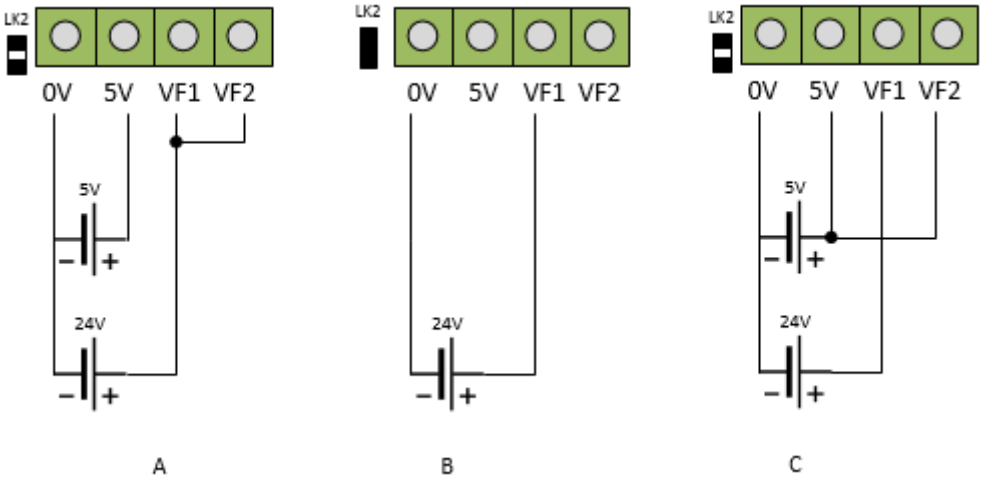
Powering the board

The board is powered via the 4 way connector block J4. How this is used depends on the board option you have purchased.



DCDC Not fitted

If you have not chosen to not have the DCDC fitted then then the 5V terminal can be used to power the PI assuming you have your own 5V supply. Alternatively you can power the pi via a micro USB and the J4 pin would then become an output for that supply.



In the above figure illustrates three powering options:

A/ External power supplies power the Pi and the field supply which runs at 24V.

B/ As A but the Pi is powered by the micro USB, the field supply is powered at 24V but LK2 is made so we don't have to wire VF2.

C/ In this instance the Pi is powered by an external 5V power supply which also feeds into VF2 so that DO17-24 operate at 5V. VF1 is powered by 24V so that DO1-16 operate at 24V.

DCDC fitted

If you have chosen to have the DCDC fitted then the 5V terminal on J4 is again an output for that supply but the Pi will be powered by VF1 which then feeds the on board DCDC converter.

Field supplies

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

- VF1 – Powers output 1-16 and the on board 5V DCDC
- VF2 – Powers output 17-24

It is possible to therefore possible to have the outputs of this board at different voltage levels i.e maybe one bank doing 5V interfacing and another handling 12V supplies.

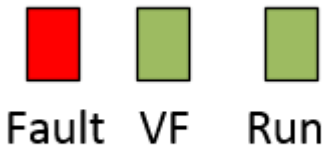
The Link LK2 may be soldered which joins VF1 and VF2 together – this may be useful for simplifying wiring.

On-board DCDC

A 1A or 2A DCDC Power supply is optionally fitted to the board to power the PCB. This has a maximum input voltage of 28V which limits the maximum fieldbus voltage used.

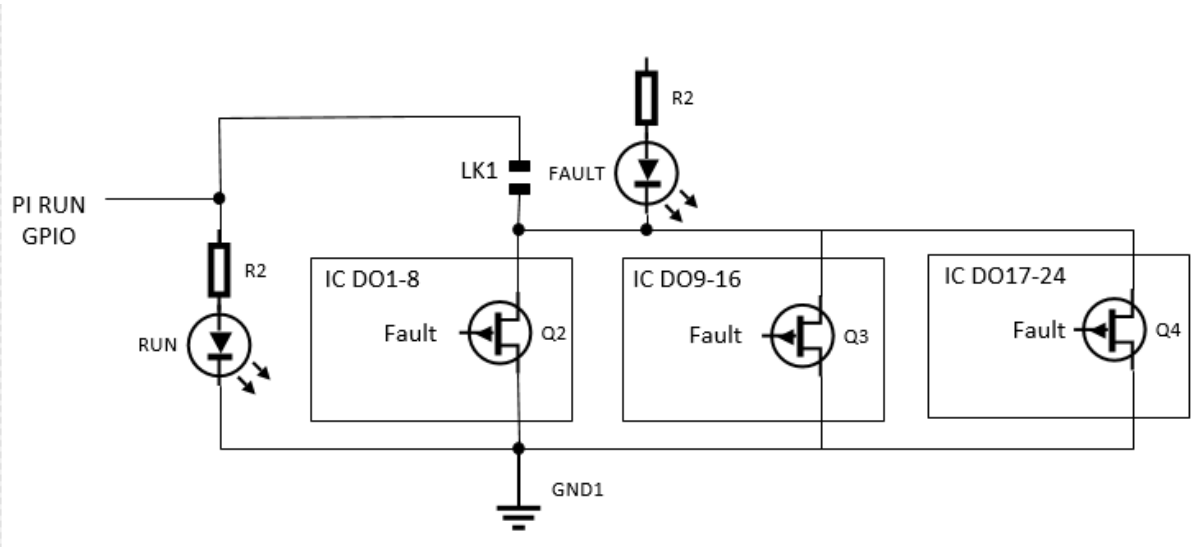
LEDs

There are three LEDs on the board.



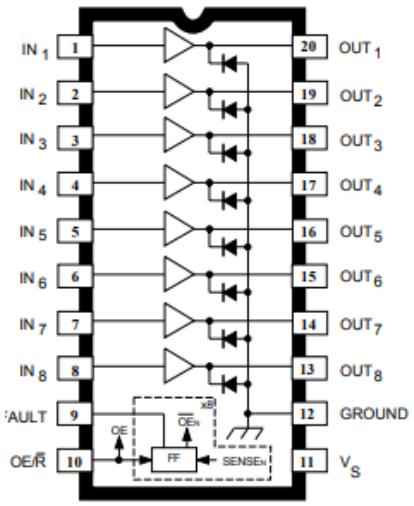
- **Fault** – Indicates a fault with one or more of the output high side driver ICs. This can be a short or over temp.
- **VF** – Indicated Field supply VF1 is powered.
- **Run** – software controlled to a GPIO Output, generally set to pulsing to indicate the program is running.

Note- when LK1 is made the fault output of the driver ICs is connected to the RUN LED GPIO Pin, this allows the RUN LED to be disabled so that pin can be changed to an input to relay the current fault status of the board back to the software. An example program details this, be careful with the configuration of the RUN LED pin though as it should always be set to an input if LK1 is made.



Digital outputs

The digital outputs are controlled via 3 UDN2987 High side driver ICS. These contain Darlington driver arrays and operator up to 35V.



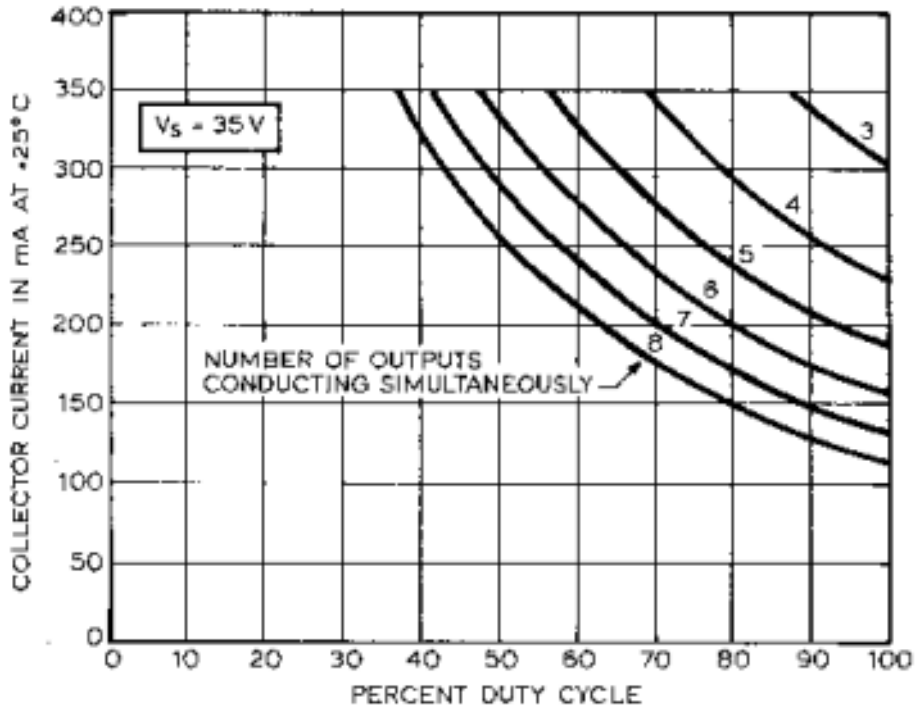
The drivers feature the following functionality:

- 35V V Max (VF1/2 – limited by DCDC if fitted on VF1 which has 28V max)
- Output enable input
- 8x 350mA Overcurrent protected High side Darlington drivers
- Internal Ground clamp diodes
- Thermal shutdown
- Output faults disable that channel and can be reset by toggling the OE Pin.

Application thermal considerations

The IC can pass 100mA on all 8 channels at a temperature of 25 Degrees Centigrade. The following data sheet describes how this can be uprated depending on channel duty cycle. Generally speaking though should be given to how much current you are asking the device to deliver and across how many pins. If you are powering two large loads arranging them so the load is shared across the 2 ICs will help greatly.

At +25°C



Dwg. No. A-13,268

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:

github repo board page

A video on how to install the library and set up your Pi can be found here.

basic config

Various example projects are documented here:

node red control

node red and python

python only control

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. Enable SPI / I2C and 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.

Software structure

The repository is structured as follows:

- **PiIO** – Fundamental drivers written in python 3
- **Docs** – Markdown documentation
- **Examples** – 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

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.

Basic_DO example

This program waits 5s then sets the corresponding output counting from 1-24. 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.

Concurrent_DO example

This program show the use of python threads.

Additional examples for the DIO board can easily be modified for usage on this board…

Fault_DIO example

This program shows how the fault status of the output drivers can be read back in software.

Nodered_DIO 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.

Nodered_direct_DIO 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.