# Raspberry Pi 12-24V interface HAT

## Preparations

This Raspberry Pi interface HAT is using several GPIO pins and an I²C device with a fixed address.

To ensure that this device will work for your system, make sure that these do not clash with GPIO and I²C devices that are already in use.

### Used GPIO pins

| Purpose | Direction | GPIO header pin no. | BCM GPIO no. | WiringPi no. |
|---|---|---|---|---|
| Digital outputs | OUTPUT | 7,29,31,36,11,12 | 4-6, 16-18 | 7,21,22,27,0,1 |
| Digital inputs | INPUT | 15,16,18,22,37,13 | 22-27 | 3,4,5,6,25,2 |
| Analog convert alert | INPUT | 26 | 7 | 11 |

### Used I²C addresses

| Purpose | Address | Device |
|---|---|---|
| 4x Analog input | 0x48 | ADS1115 |

## Installation

The Raspberry Pi 12-24V interface HAT is equipped with a 40p header, which means that you can place this on top of your Raspberry Pi.

After having added the interface HAT you can enable the ADS1115 analog inputs in the config file (/boot/config.txt) by adding the following under the [all] section

```
dtoverlay=ads1115
dtparam=cha_enable
dtparam=chb_enable
dtparam=chc_enable
dtparam=chd_enable
```

For more information refer to https://github.com/raspberrypi/linux/blob/rpi-4.9.y/arch/arm/boot/dts/overlays/README#L210

Make sure you have enabled the I2C interface on your Raspberry using `raspi-config` (see https://www.raspberrypi.org/documentation/configuration/raspi-config.md for more information).

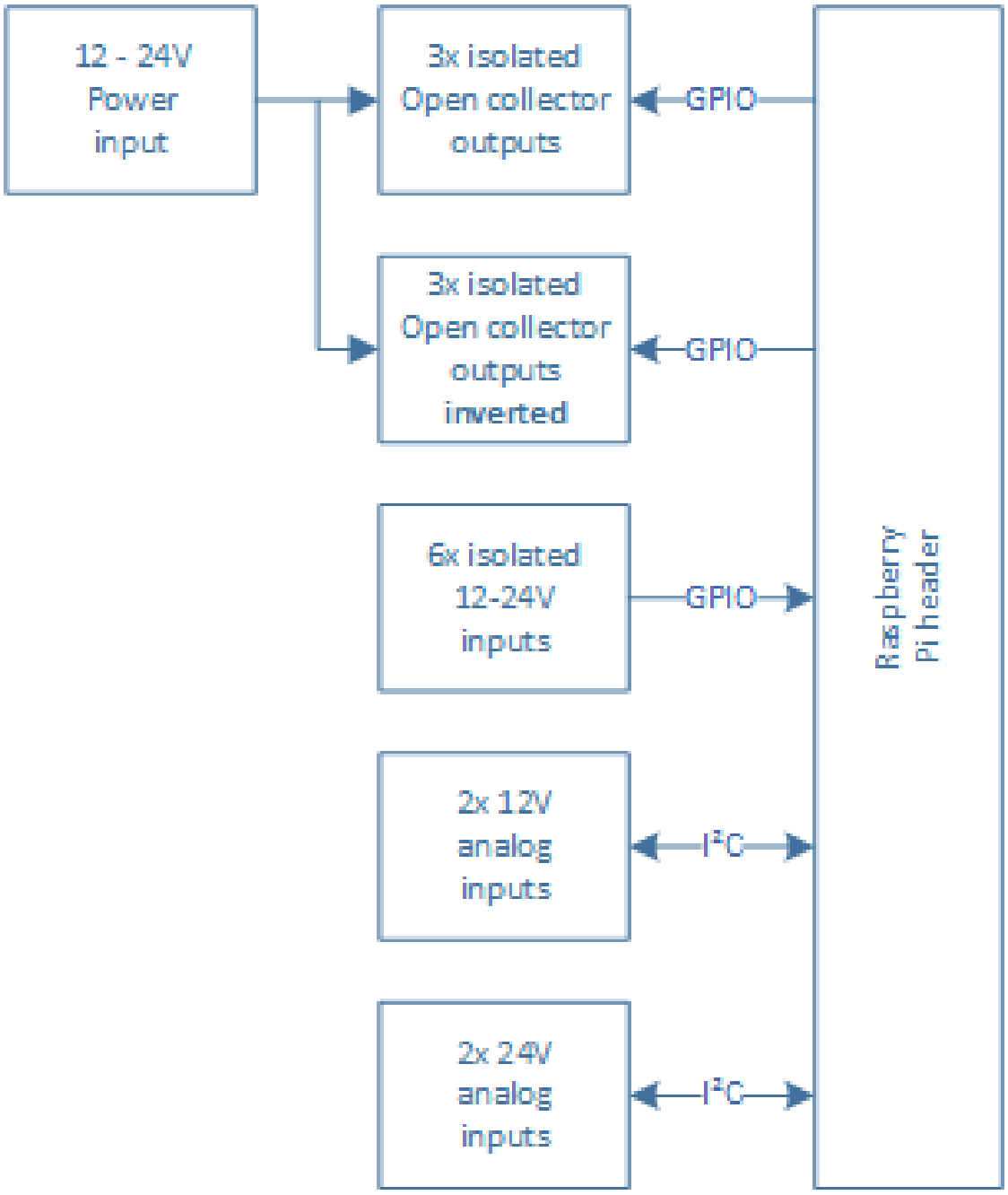
After a reboot you will find the device file entries in the folder `/sys/bus/i2c/devices` under a new created folder for the ADS1115 device. For example, a folder named `1-0048` in which you will find a file called '`name`'. Check the contents of this file (using `cat name`) to make sure it is related to the ADS1115.

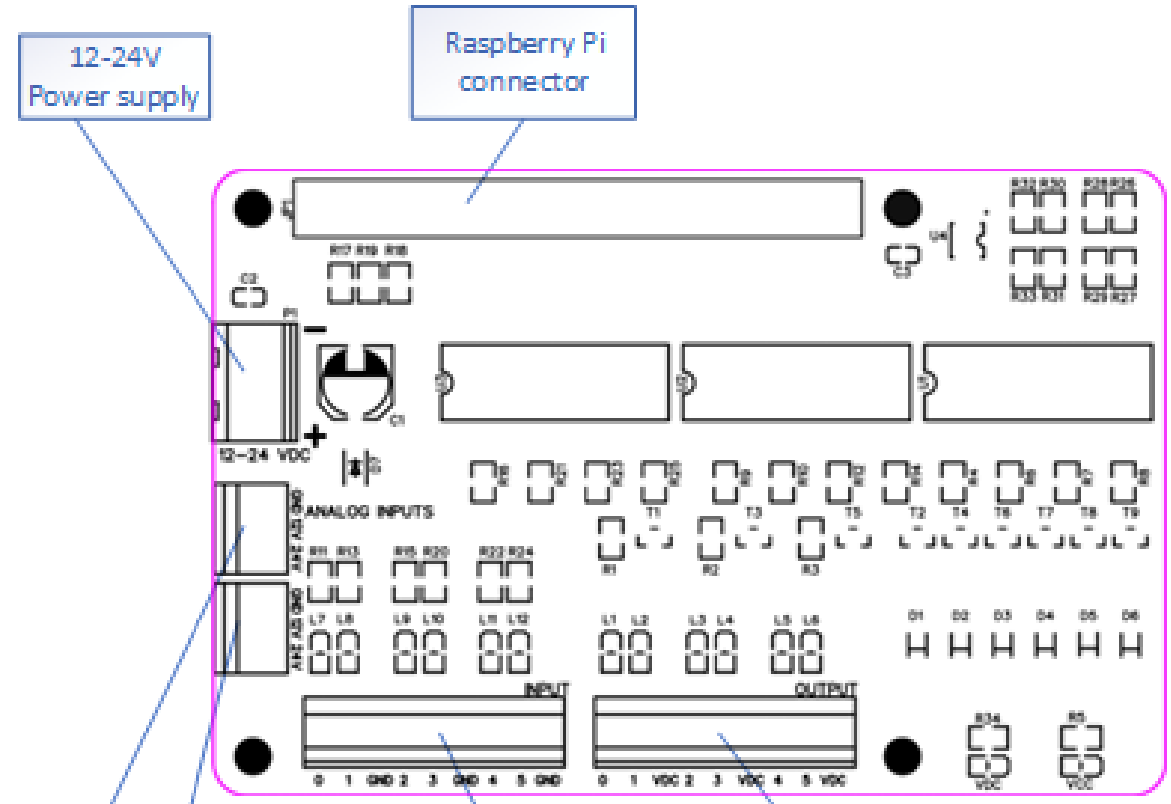
## Drivers

No specific drivers are required for this HAT as uses the standard GPIO pins.

When not using the ADS1115 as described above, please use either standard python libraries to read the analog inputs or standard I²C drivers (for example https://github.com/vincentrou/ads1115_lib).

# 12 – 24V Raspberry Pi interface HAT






12..24V analog inputs

ADS1115 I²C address 0x48

0..12V -> 0..4.8V o.p ADC
0..24V -> 0..4.9V o.p ADC

Bottom: 12V IN0
Bottom: 24V IN1

Top: 12V IN2
Top: 24V IN3

12..24V inputs

Based on BCM GPIO

0 -> GPIO22
1 -> GPIO23
2 -> GPIO24
3 -> GPIO25
4 -> GPIO26
5 -> GPIO27

Open collector outputs

Based on BCM GPIO

**Not inverted logic**
GPIO4 -> 0
GPIO5 -> 1
GPIO6 -> 2

**Inverted logic**
GPIO16 -> 3
GPIO17 -> 4
GPIO18 -> 5

Python Code example:

```
import ADS1115
import time
import os
from gpiozero import DigitalOutputDevice, DigitalInputDevice

ads = ADS1115.ADS1115()

# INPUTS
I22 = DigitalInputDevice(22)
I23 = DigitalInputDevice(23)
I24 = DigitalInputDevice(24)
I25 = DigitalInputDevice(25)
I26 = DigitalInputDevice(26)
I27 = DigitalInputDevice(27)

# OUTPUTS
O4 = DigitalOutputDevice(4)
O5 = DigitalOutputDevice(5)
O6 = DigitalOutputDevice(6)
O16 = DigitalOutputDevice(16)
O17 = DigitalOutputDevice(17)
O18 = DigitalOutputDevice(18)

I22.when_activated = O4.on
I22.when_deactivated = O4.off
I23.when_activated = O5.on
I23.when_deactivated = O5.off

O6.blink(0.5, 1)
time.sleep(0.2)
O16.blink(0.5, 1)
time.sleep(0.2)
O17.blink(0.5, 1)
time.sleep(0.2)
O18.blink(0.5, 1)

print("\033[?25l")
os.system('clear')
while True:
    print("\033[0;0H")

    print("AN0 = {:.0f} mV          ".format(ads.readADCSingleEnded(0) * 2.5))
    print("AN1 = {:.0f} mV          ".format(ads.readADCSingleEnded(1) * 4.9))
    print("AN2 = {:.0f} mV          ".format(ads.readADCSingleEnded(2) * 2.5))
    print("AN3 = {:.0f} mV          ".format(ads.readADCSingleEnded(3) * 4.9))

    I0 = I22.value
    I1 = I23.value
    I2 = I24.value
    I3 = I25.value
    I4 = I26.value
    I5 = I27.value
    print("I0 = {:d}".format(I0))
    print("I1 = {:d}".format(I1))
    print("I2 = {:d}".format(I2))
    print("I3 = {:d}".format(I3))
    print("I4 = {:d}".format(I4))
    print("I5 = {:d}".format(I5))

    print("O0 = {:d}".format(O4.value))
    print("O1 = {:d}".format(O5.value))
    print("O2 = {:d}".format(O6.value))
    print("O3 = {:d}".format(O16.value))
    print("O4 = {:d}".format(O17.value))
    print("O5 = {:d}".format(O18.value))
```