SF Innovations Ltd

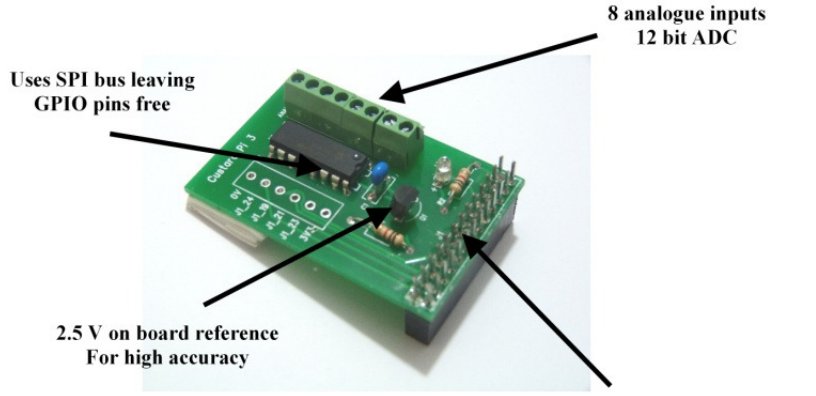
Custard Pi 3 - 8 Analogue input board for the Raspberry Pi GPIO

User Instructions (13th December 2016)

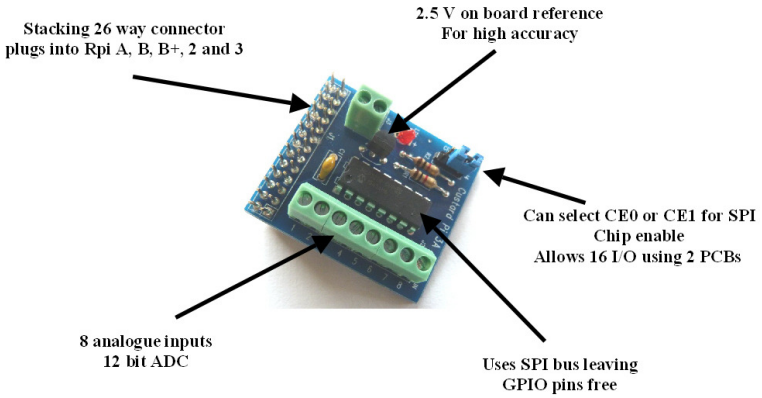
Contents

Introduction

The Raspberry Pi GPIO allows the control of external electronics. There are two rows of 13 pins which are brought out to a 26 way header on the edge of the board. The Custard Pi 3 board simply plugs into the Raspberry Pi GPIO connector and provides eight 12 bit analogue inputs.



Custard Pi 3



Custard Pi 3A

The Custard Pi 3 uses a stacking 26 way connector. What this means is that when it is plugged into the Raspberry Pi GPIO, these pins are still available for other accessories. The Custard Pi 3 can also be plugged into the 40 pin GPIO provided on the later raspberry Pi's such as the A+. B+, Raspberry Pi 2 and Zero models.

CE Compliance and Safety Information

* This product simply plugs into the Raspberry Pi GPIO to allow users to interface their products and projects to the Raspberry Pi. It uses the 3.3V supply from the Raspberry Pi to power the ADC and the DAC. For this reason it is outside the scope of the LVD Directive.

* The connection of incompatible devices to this product may cause damage and invalidate the warranty.

* All devices connected to this product must comply with all relevant standards to ensure that safety and performance is not compromised.

* This product complies with the Class B limit for Electromagnetic Radiation when used with the Raspberry Pi.

* This product provides reasonable protection against harmful interference in a residential installation. However there could be deterioration in performance in the presence of strong RF fields. For this reason, this product should not be used in any safety critical applications. Any wires connected to this product should be less than 2 metres in length. Avoid handling the PCB while it is powered. Only handle by the edges to avoid the risk of ESD damage.

* If this product is being incorporated in a commercial product which uses either all CE marked products or some CE marked and some non CE marked product, it is the responsibility of the system integrator to assess the end product for compliance with the relevant EU Directives.

* This product complies with the requirements of the RoHS regulations.

Circuit Description

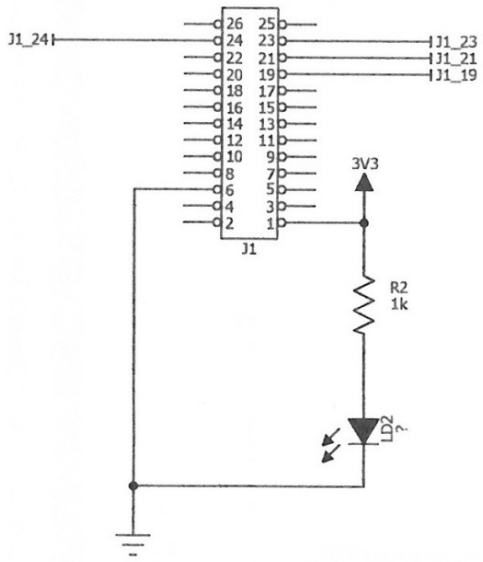
When the Custard Pi 3 is plugged into the GPIO, an LED comes ON, showing that the 3.3V rail is working correctly.

The diagram below shows the connections to the GPIO. Pin 1 is the 3.3V supply and pin 6 is the 0V connection. The other 4 pins shows are the connections to the SPI bus and are defined as follows.

Pin 19: SPI MOSI - Data into chip
Pin 21: SPI MISO - Data out of chip
Pin 23: SPI SCLK - Clock
Pin 24: SPI CS1 - Chip select (low to select)

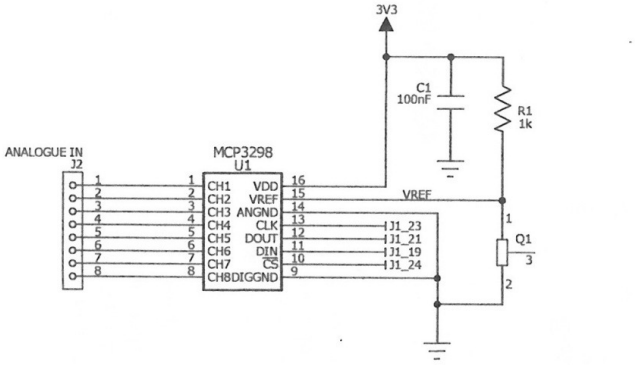
Note CPi3: Pin 26 (SPI CS2) is not used as we only use one SPI device on this card. It is available for use for communication with another SPI device.

Note CPi3A: This has a jumper link that can be used to select CE0 or CE1. This means that 2 cards can be used on a Raspberry Pi to provide up to 16 analogue inputs.



GPIO connections

The connections to the 8 channel Analogue to Digital Convertor (ADC) is shown below. The MCP3208 is a 12-bit convertor (this means it uses 4096 bits to represent voltages from 0 to 2.5V).

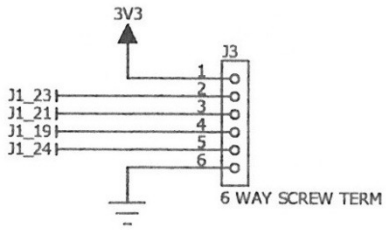


The 8 analogue channels are not voltage limited or protected in any way. It is up to the user to make sure that the analogue voltage presented to the Custard Pi 3 is not larger than 2.5V or more negative than 0V.

Q1 is the external 2.5 precision voltage reference used. Some example conversions from bits to voltages are shown below.

4096 bits represents 2.5V
2048 bits represents 1.25V
1024 bits represents 0.625V
100 bits represents 0.061V (or 61mV)

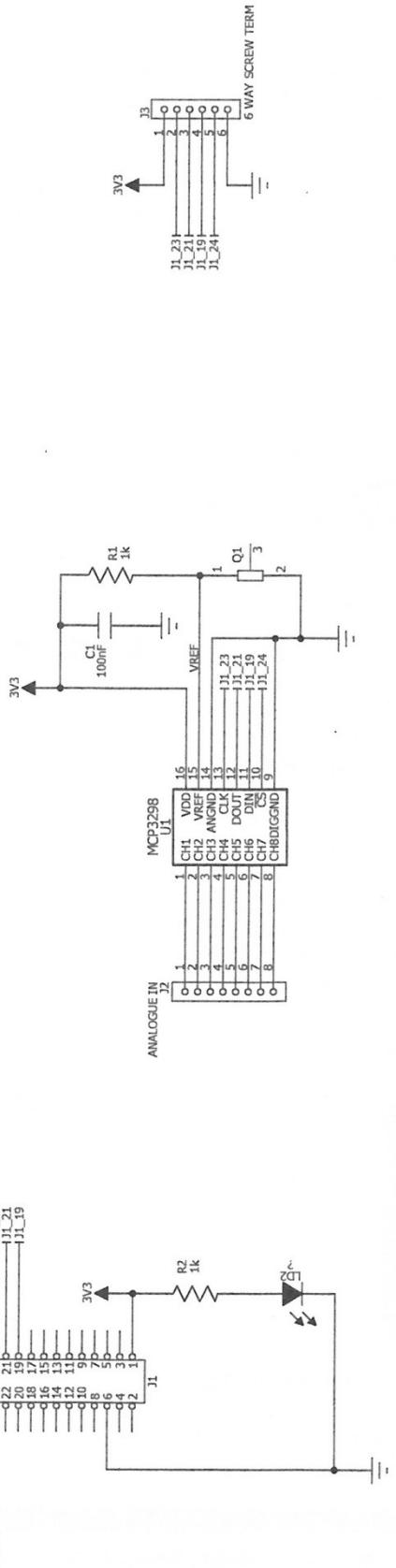
Some of the signals on the Custard Pi 3 are brought out to a series of via holes and can be used for other purposes, such as providing a 3.3V or 0V supply. This is shown below.



Parts List

Qty	Description	Ref	Source
1	Printed Circuit Board (PCB)		
1	26 way stackable connector	J1	
4	4 x 2 way screw terminal connectors	J2	Rapid 21-3000
1	1 x 100nF capacitor	C1	Rapid 11-3454
1	1 x MCP3208	U1	Farnell 1084269
1	1 x LED	LD2	Rapid 55-0816
2	2 x 1k resistors	R1, R2	Rapid 62-0370
1	1 x LT1009	Q1	Rapid 82-4015

Schematic


J3
6 WAY SCREW TERM
3V3
J1_23
J1_21
J1_19
J1_24
R1
1k
Q1
C1
100nF
VREF
MCP3298
U1
VDD
VREF
ANGND
CLK
DOUT
DIN
CS
DIGGND
CH1
CH2
CH3
CH4
CH5
CH6
CH7
CH8
ANALOGUE IN
J2
R2
1k
J1

Sample code for testing analogue inputs

This program reads each analogue channel in turn and prints the result to the screen. Supply suitable analogue voltages between 0 and 2.5V to the channels and run the program.

Note: we use the terminology Channel 0 to 7. On the PCB they are marked as Channel 1 to 8.

```
#1/usr/bin/env python
#Sample Python program to test 8 analogue inputs on Custard Pi 3
#www.sf-innovations.co.uk

import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)

GPIO.setup(24, GPIO.OUT)    #pin 24 is chip enable
GPIO.setup(23, GPIO.OUT)    #pin 23 is clock
GPIO.setup(19, GPIO.OUT)    #pin 19 is data out
GPIO.setup(21, GPIO.IN)     #pin 21 is data in

#set pins to default state
GPIO.output(24, True)
GPIO.output(23, False)
GPIO.output(19, True)

#set address channels 0 to 7
#1st bit selects single/differential
#2nd bit channel address
#3rd bit channel address
#4th bit channel address
#5th bit 1 bit delay for data
#6th bit 1st null bit of data

#read each channel in turn
for y in range (0,8):
    if y == 0:
        word= [1 ,1, 0, 0, 0, 1, 1]     #set channel 0
    if y == 1:
        word= [1 ,1, 0, 0, 1, 1, 1]     #set channel 1
    if y == 2:
        word= [1 ,1, 0, 1, 0, 1, 1]     #set channel 2
    if y == 3:
        word= [1 ,1, 0, 1, 1, 1, 1]     #set channel 3
    if y == 4:
        word= [1 ,1, 1, 0, 0, 1, 1]     #set channel 4
    if y == 5:
        word= [1 ,1, 1, 0, 1, 1, 1]     #set channel 5
    if y == 6:
        word= [1 ,1, 1, 1, 0, 1, 1]     #set channel 6
    if y == 7:
        word= [1 ,1, 1, 1, 1, 1, 1]     #set channel 7

    GPIO.output(24, False)  #enable chip
    anip=0  #clear variable

#clock out 7 bits to select channel
    for x in range (0,7):
        GPIO.output(19, word[x])
        time.sleep(0.01)
        GPIO.output(23, True)
        time.sleep(0.01)
        GPIO.output(23, False)

#clock in 11 bits of data
```

```
    for x in range (0,12):
        GPIO.output(23,True)    #set clock hi
        time.sleep(0.01)
        bit=GPIO.input(21)      #read input
        time.sleep(0.01)
        GPIO.output(23,False)   #set clock lo
        value=bit*2**(12-x-1)   #work out value of this bit
        anip=anip+value         #add to previous total
#    print x, bit, value, anip

    GPIO.output(24, True)       #disable chip

    volt = anip*2.5/4096        #use ref voltage of 2.5 to work out voltage
    print "voltage ch", y, ("%.2f" %round(volt,2)) #print to screen

GPIO.cleanup()
import sys
sys.exit()

#TO TEST
#Note: we use the terminology Channel 0 to 7. On the PCB they are marked as Channel 1 to
8.
#Connect Channels 0 to 7 to 2.5V and voltage should be 2.5V printed to screen
#Connect two equal resistors in series from 2.5V to 0V.
#Connect mid point of resistors to Channels 0 to 7 in turn
#Voltage should be 1.25V printed to screen
```

End of Document