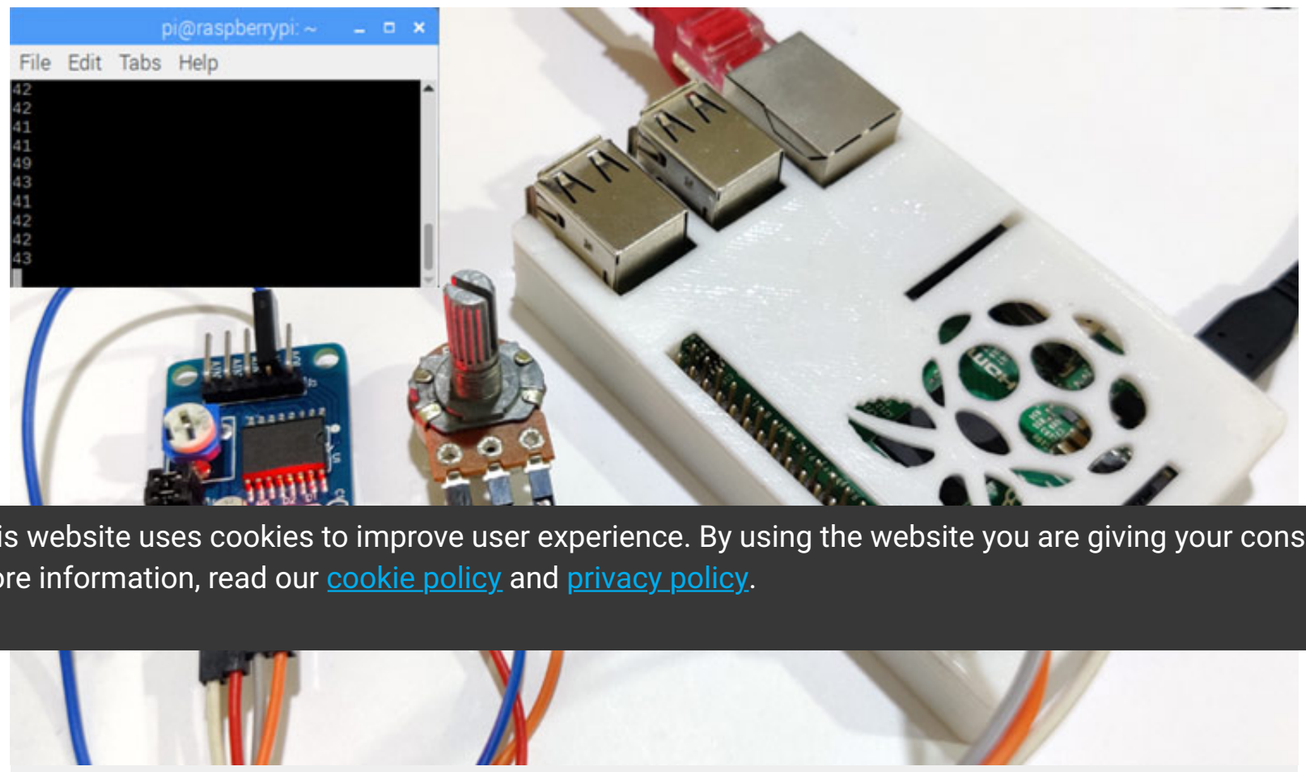
RASPBERRY PI

How to Interface PCF8591 ADC/DAC Analog Digital Converter Module with Raspberry Pi

By Abhishek Sharma Jul 25, 2019 0



Interfacing PCF8591 ADCDAC Analog Digital Converter Module with Raspberry Pi

Analog to digital conversion is a very important task in embedded electronics, as most of the sensors provide output as analog values and to feed them into microcontroller which only understand binary values, we have to convert them into Digital values. So to be able to process the analog data, microcontrollers need **Analog to Digital Converter**.

Some microcontroller has inbuilt ADC like Arduino, MSP430, PIC16F877A but some microcontroller don't have it like 8051, Raspberry Pi etc and we have to use some external Analog to digital converter ICs like ADC0804, ADC0808. Below you can find various examples of ADC with different microcontrollers:

- How to Use ADC in Arduino Uno?
- Raspberry Pi ADC Tutorial
- Interfacing ADC0808 with 8051 Microcontroller
- 0-25V Digital Voltmeter using AVR Microcontroller
- How to use ADC in STM32F103C8
- How to use ADC in MSP430G2
- How to use ADC in ARM7 LPC2148
- Using ADC Module of PIC Microcontroller with MPLAB and XC8

In this tutorial, we are going to learn **how to interface PCF8591 ADC/DAC module with Raspberry Pi.**

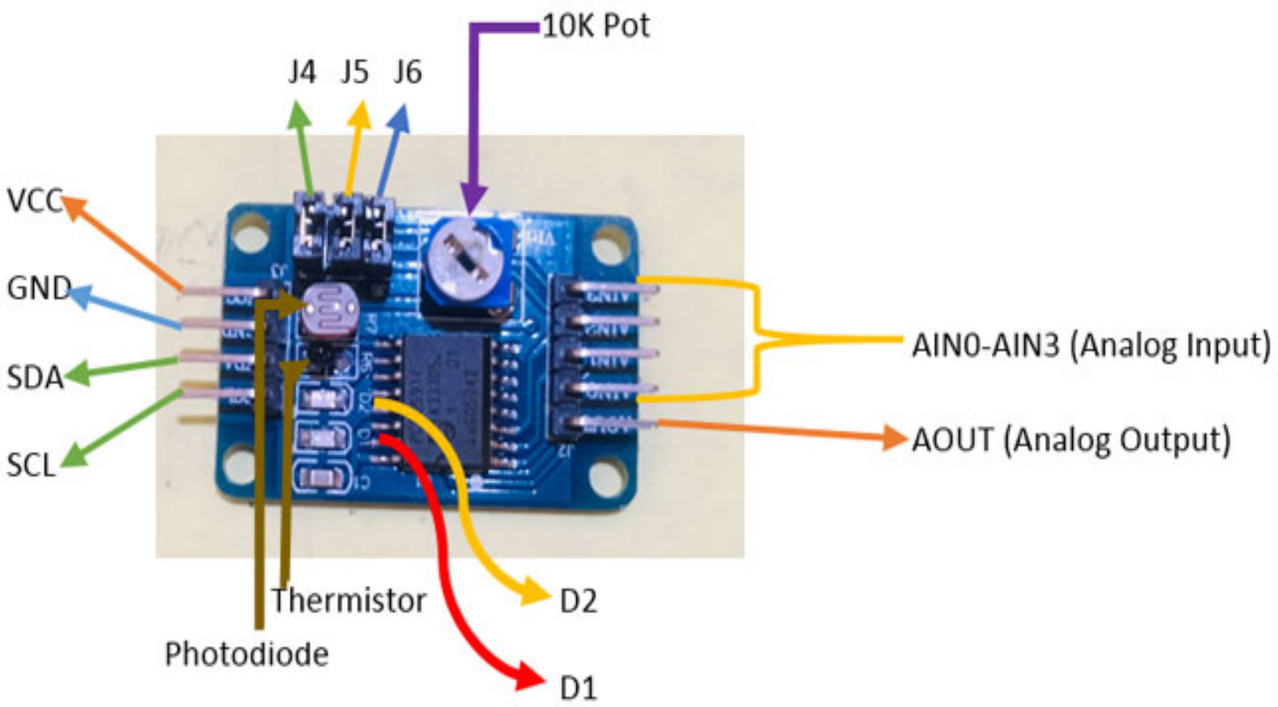
Required Components

1. Raspberry-pi
2. PCF8591 ADC Module
3. 100K Pot
4. Jumper Cables

It is assumed you have Raspberry Pi with latest Raspbian OS installed in it and you know how to SSH into the Pi using a terminal software like putty. If you are new to Raspberry Pi then follow this article to get started with Raspberry Pi. Still if you face any issue then there are tons of Raspberry Pi Tutorials that can help.

PCF8591 ADC/DAC Module

PCF8591 is an 8 bit analog to digital or 8 bit digital to analog converter module meaning each pin can read analog values up to 256. It also has LDR and thermistor circuit provided on the board. This module has four analog input and one analog output. It works on I^2C communication, so there are SCL and SDA pins for serial clock and serial data address. It requires **2.5-6V** supply voltage and have low stand-by current. We can also manipulate the input voltage by adjusting the knob of potentiometer on the module. There are also three jumpers on the board. J4 is connected to select the **thermistor access circuit**, J5 is connected to select the **LDR/photo resistor access circuit** and J6 is connected to select the adjustable voltage access circuit. There are two LEDs on board D1 and D2- D1 shows the output voltage intensity and D2 shows the intensity of supply voltage. Higher the output or supply voltage, higher the intensity of LED D1 or D2. You can also test these LEDs by using a potentiometer on VCC or on AOUT pin.



NEWS ARTICLES PROJECTS

Ring-Based Smart Pulse Oximeter with BLE Connectivity using nRF52832 SoC

ST Adds Support for LoRaWAN Firmware Update Over The Air in the STM32Cube Ecosystem

ESP32 now has Bluetooth LE 5.0 Certification with Improved Stability and Compatibility

Arm Cortex M23 MCU with Multi-Segment's Secured Storage for IoT Applications

Unregulated DC/DC Converters with 20kVDC Isolation

I2C pins in Raspberry Pi

In order to use **PCF8591** with Raspberry Pi, the first thing to do is knowing the Raspberry Pi I2C port pins and configuring I2C port in the Raspberry pi.

Below is the Pin Diagram of Raspberry Pi 3 Model B+, and **I2C pins GPIO2 (SDA) and GPIO3 (SCL)** are used in this tutorial.

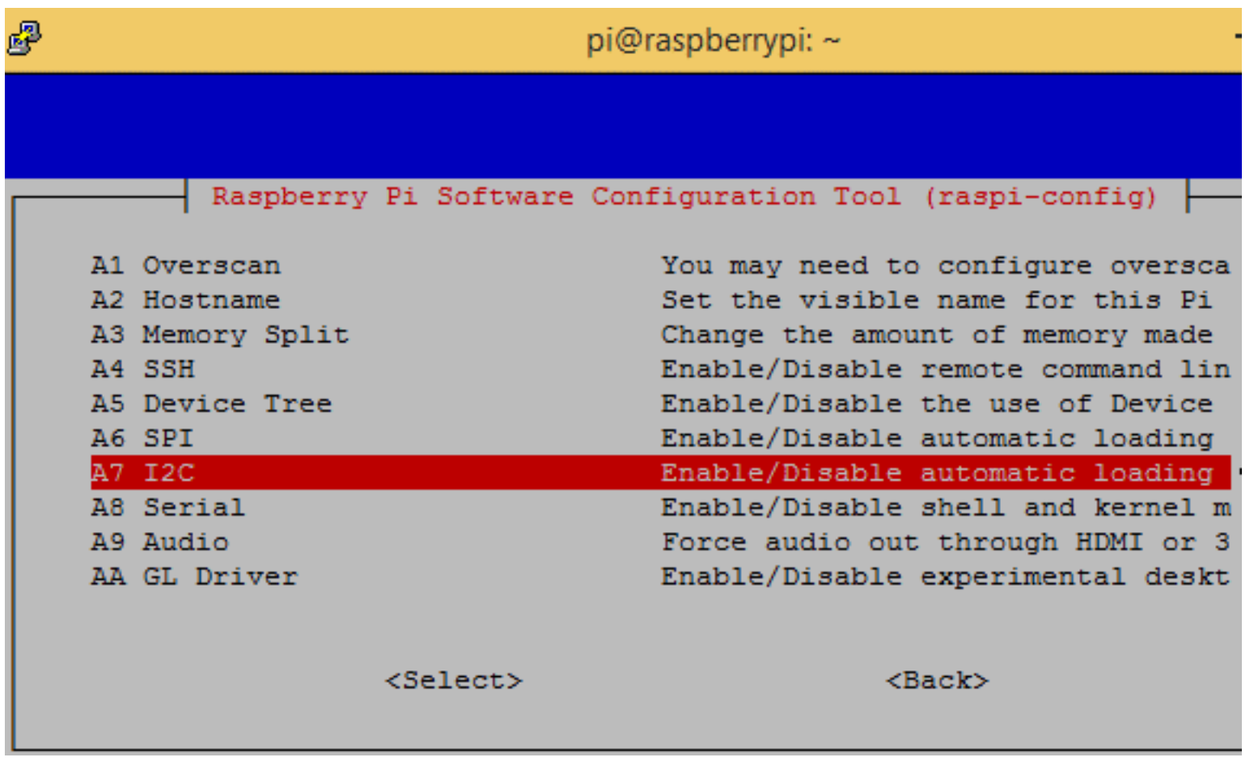
Configuring I2C in Raspberry Pi

By default, I2C is disabled in Raspberry Pi. So first it must be enabled. To enable the I2C in Raspberry Pi

1. Go to the terminal and type **sudo raspi-config.**

2. Now the Raspberry Pi Software Configuration Tool appears.

3. Select **Interfacing options** and then enable the I2C.



4. After enabling the I2C reboot the Pi.

Scanning I2C Address of PCF8591 using Raspberry Pi

Now in order to start communication with the PCF8591 IC, the Raspberry Pi must know its I2C address. To find the address first connect the SDA and SCL pin of PCF8591 to the SDA and SCL pin of Raspberry Pi. Also connect the +5V and GND pins.

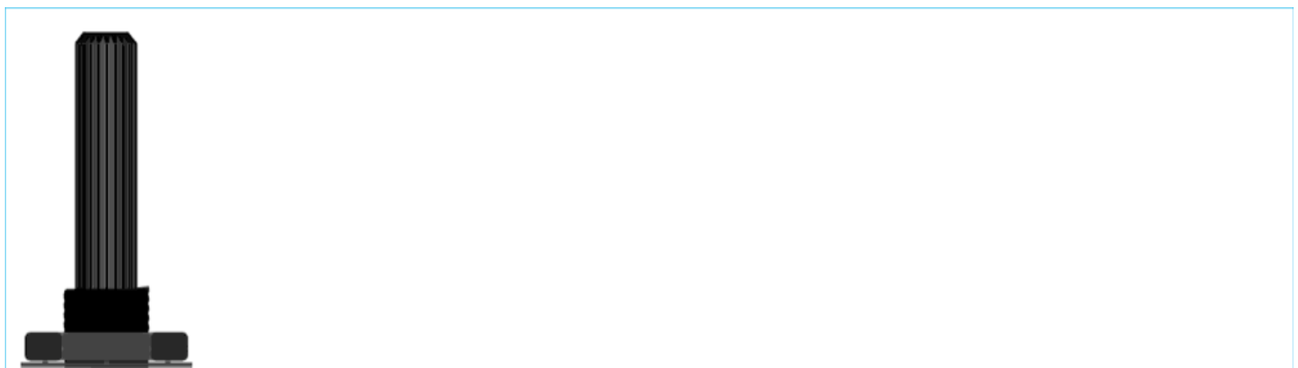
Now open the terminal and type below command to know the address of connected I2C device,

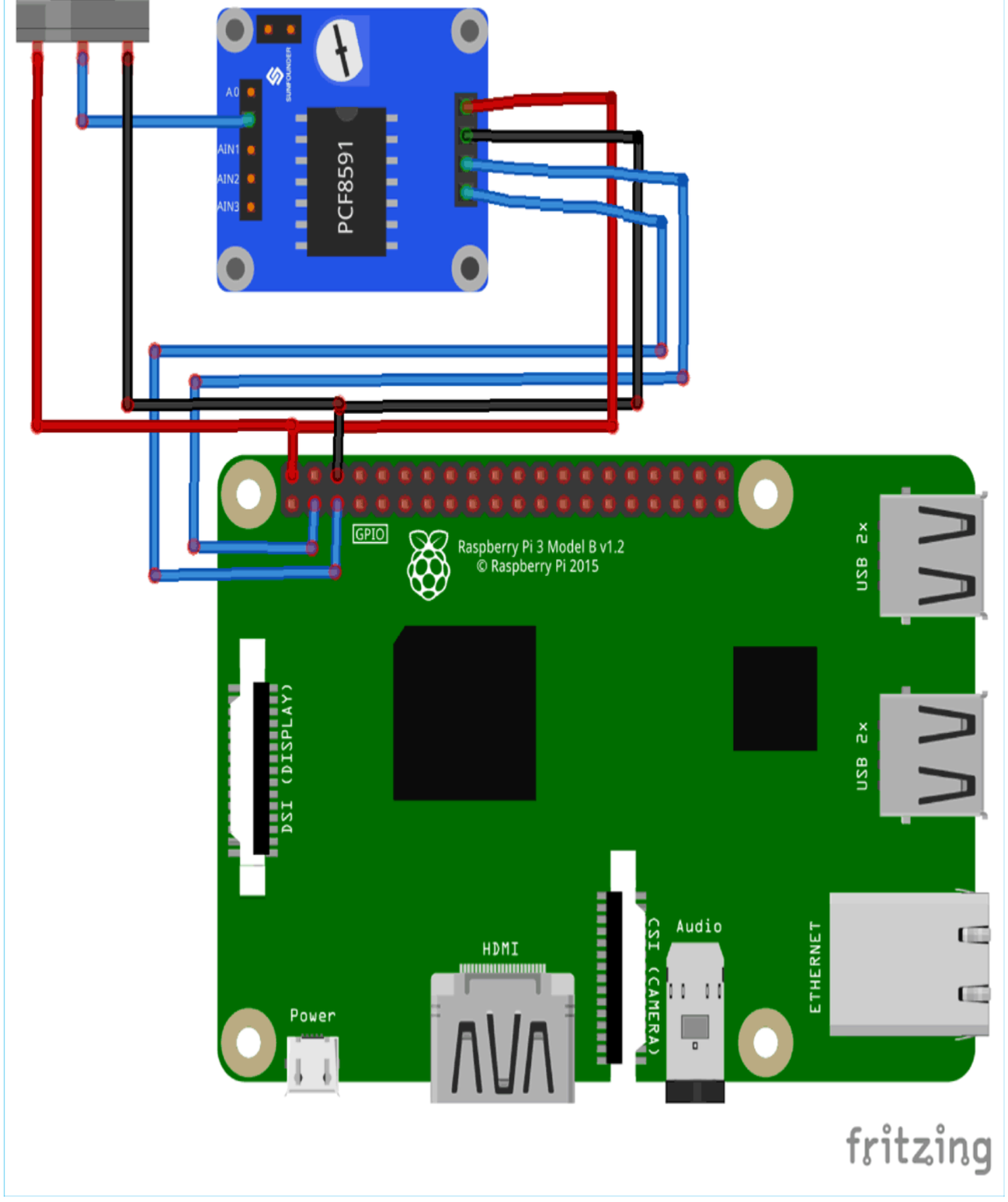
```
sudo i2cdetect –y 1 or sudo i2cdetect –y 0
```

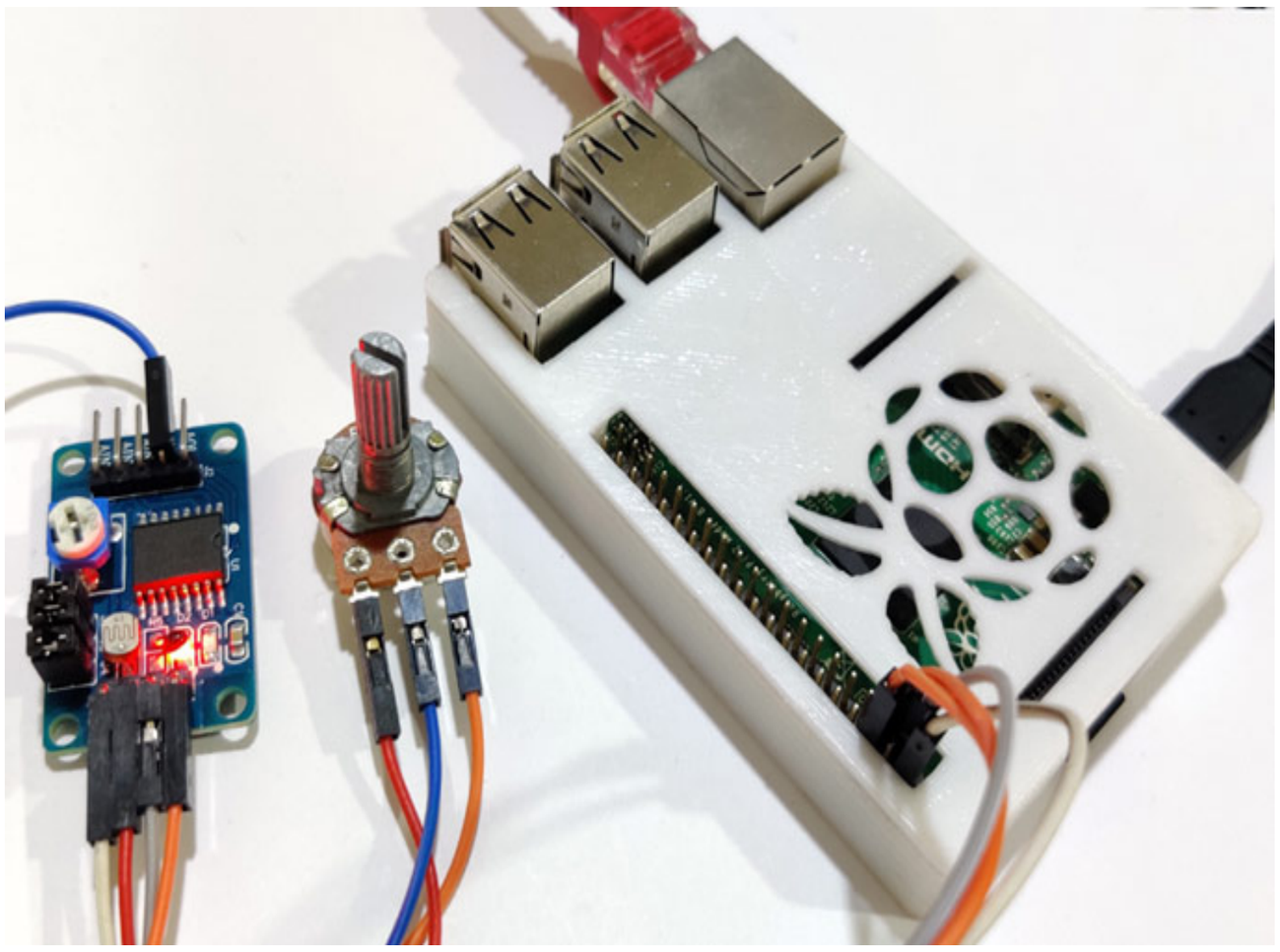
After finding the I2C address now its time to build the circuit and install the necessary libraries for using **PCF8591 with Raspberry Pi**.

Interfacing PCF8591 ADC/DAC Module with Raspberry Pi

Circuit diagram for **Interfacing of PCF8591 with Raspberry Pi** is simple. In this interfacing example, we will read the analog values from any of the analog pins and show it on Raspberry Pi terminal. We can change the values using a 100K pot.







Connect the VCC and GND to GPIO2 and GPIO of Raspberry Pi. Next, connect the SDA and SCL to GPIO3 and GPIO5 of respectively. Finally connect a 100K pot with AIN0. You can also add 16x2 LCD to display ADC values instead of showing it on Terminal. Learn more about interfacing 16x2 LCD with Raspberry Pi here.

Python program for Analog to Digital Conversion (ADC)

The **complete program and working video** is given at the end of this tutorial.

Firstly, import the *smbus* library for I^2C bus communication and time library to give a sleep time between printing the value.

```
import smbus
import time
```

Now define some variables. The first variable contains the address of the I^2C bus and second variable contains the address of first analog input pin.

```
address = 0x48
A0 = 0x40
```

Next, we've made an object of the function SMBus(1) of library smbus

```
bus = smbus.SMBus(1)
```

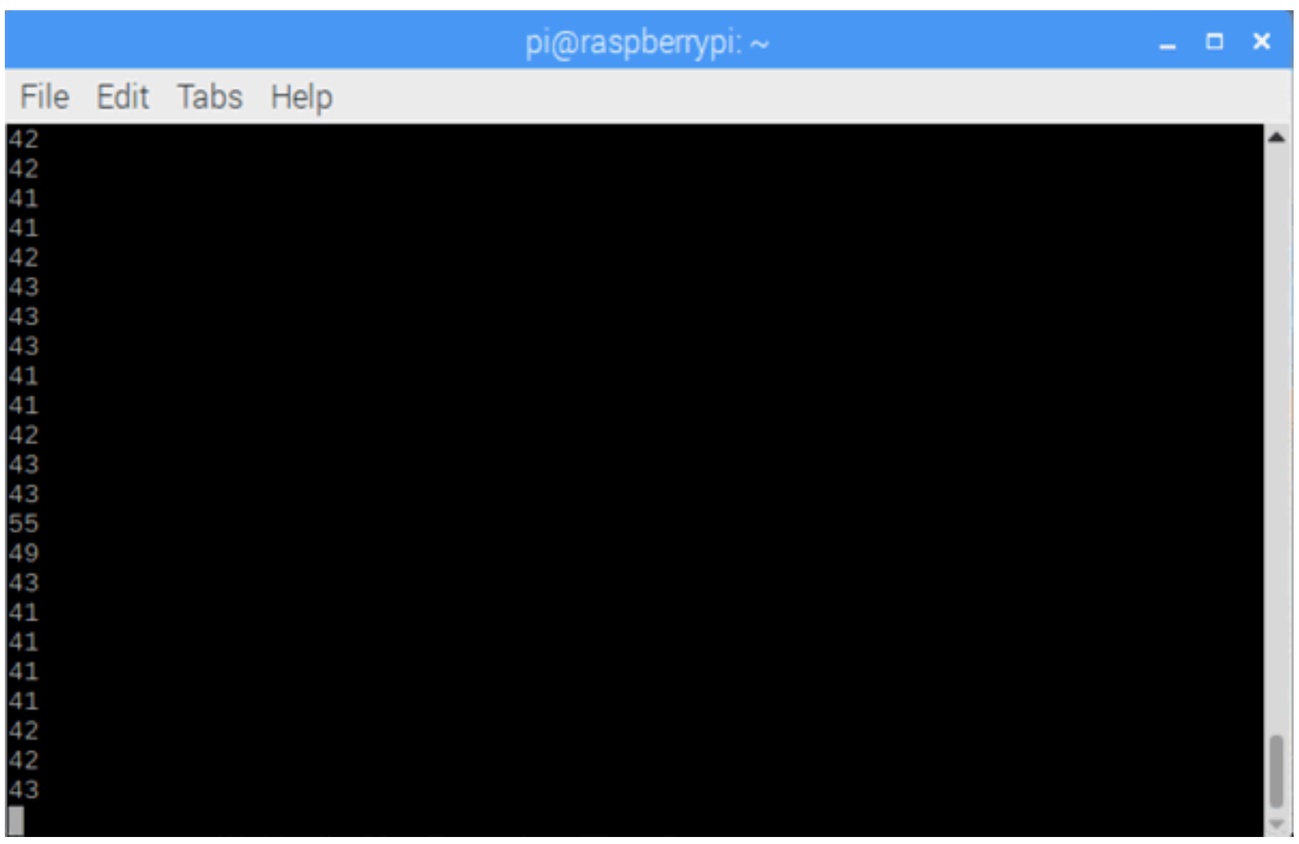
Now, in *while* the first line tells the IC to make the analog measurement at the first analog input pin. The second line stores the address read at analog pin in a variable *value*. Finally print the value.

```
while True:
    bus.write_byte(address,A0)
    value = bus.read_byte(address)
    print(value)
    time.sleep(0.1)
```

Now finally save the python code in some file with .py entension and run the code in raspberry Pi terminal by using below command"

```
python filename.py
```

Before running the code ensure that you have enabled the I^2C communication and all the pins are connected as shown in the diagram, otherwise it will show errors. The analog values must start showing up on terminal like below. Adjust the pot's knob, and you will see the gradual change in the values. Learn more about running the program in



Complete python code and Video is given below.

Code

```
import smbus
import time
address = 0x48
bus = smbus.SMBus(1)

while True:
    bus.write_byte(address,A0)
    value = bus.read_byte(address)
```

```
print(value)

time.sleep(0.1)
```

Video

Ein Fehler ist aufgetreten.

Schau dir dieses Video auf www.youtube.com an oder aktiviere JavaScript, falls es in deinem Browser deaktiviert sein sollte.

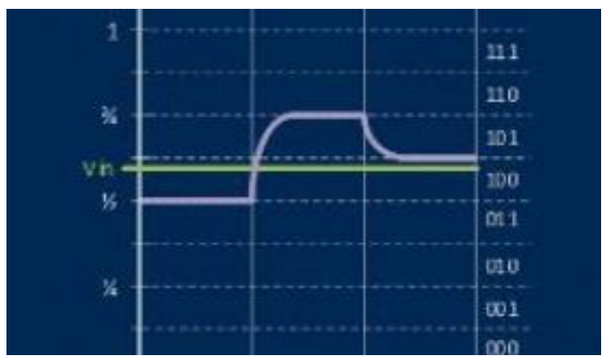
TAGS RASPBERRY PI INTERFACING ADC DAC PYTHON

RECOMMENDED POSTS

Dev Kit Weekly: Ayla Development Kit v2.0

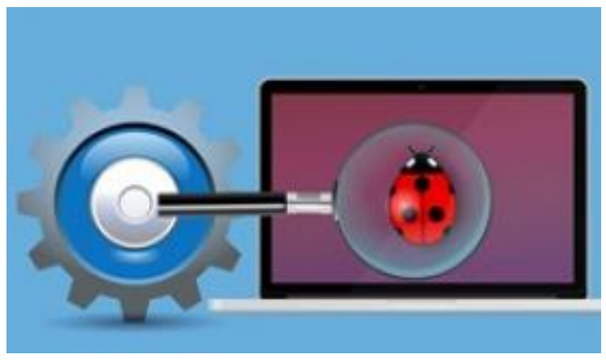
Dev Kit Weekly: BeagleBone AI

Industrial IoT Workloads Benefit from Edge to Cloud Intelligence

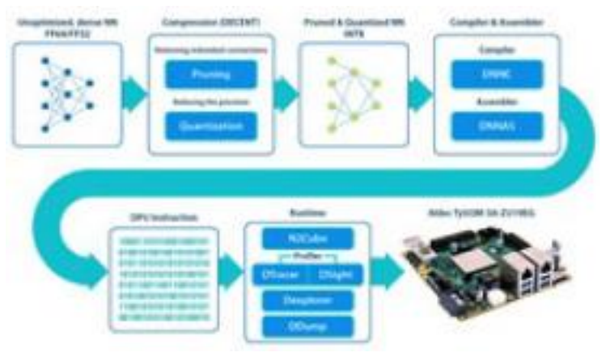


The Engineer's Answer to Faster Sampling at Lower Power

Modem or Module? Answering a Key Question for Cellular IoT Design Projects



Enhancing Debug and Trace Capabilities for System Developers



How to Develop High-Performance Object Recognition on FPGA Edge Devices

IoT Device Security Conference 2020

Get Our Weekly Newsletter!

Subscribe below to receive most popular news, articles and DIY projects from Circuit Digest

Email Address *

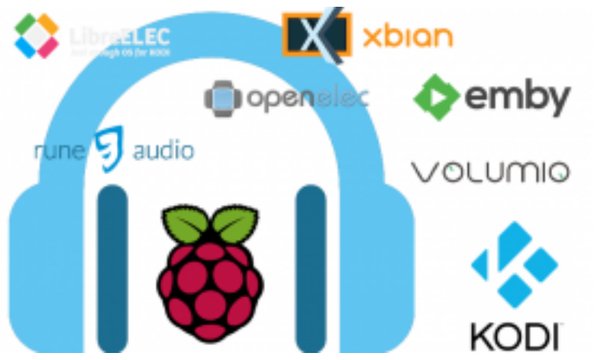
Name

Country

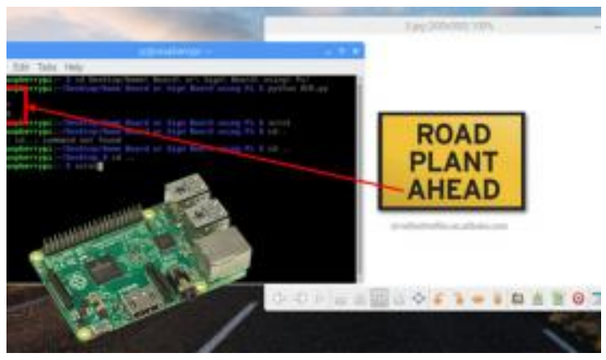
United States of America

Subscribe

RELATED CONTENT



Top 10 Media Server Software for Music Streaming on Raspberry Pi



Optical Character Recognition (OCR) using Tesseract on Raspberry Pi

Raspberry Pi Based Jarvis themed Speaking Alarm Clock

How to run Spotify on Raspberry Pi using Mopidy Music Server

How to build Raspberry Pi NAS Server using Samba

DIY Raspberry Pi Gaming Console using RetroPie

Wireless RF Communication using nRF24L01 Module

How to setup DietPi on Raspberry Pi

‹ PREVIOUS POST

DIY Arduino Power Supply Shield with 3.3v, 5v and 12v Output Options

NEXT POST ›

ESP32 based Bluetooth iBeacon

COMMENTS

LOG IN OR REGISTER TO POST COMMENT