Hello Friends, Welcome to the video tutorial on Raspberry Pi.

In this tutorial we will learn how to interface the ADC (IC MCP3008) with Raspberry Pi.

As Raspberry Pi does not have any analogue pins, so to interface the analogue sensors with RPi we need to convert the analogue values of sensors to digital values. This can be done by using an ADC i.e., IC MCP3008 which is a 16 pin IC. The MCP3008 is a successive approximation 10bit 8-channel Analogue to digital converter (ADC). It uses the SPI bus protocol which is supported by the RPi GPIO header.

Starting with the pins of MCP3008:

1. Pin 1 to 8 are 8 analogue channels to read the value of sensors.
2. Pin 9 and pin 14 are digital and analogue ground connection to internal digital and analogue circuitry respectively. These are connected to ground of Raspberry Pi.
3. Pin 10 is chip select pin used to initiate communication with the device when pulled low. When pulled high, it will end a conversion and put the device in low-power standby. This is connected to RPi pin 24 (chip select).
4. Pin 11 is SPI port serial data input pin which is used to load channel configuration data into the device. This is connected to RPi pin 19 (SPI\_MOSI i.e., master out slave in).
5. Pin 12 is SPI serial data output pin used to shift out the results of the A/D conversion. This is connected to RPi pin 21 (SPI\_MISO i.e., master in slave out).
6. Pin 13 is SPI clock pin used to initiate a conversion and clock out each bit of the conversion as it takes place. This is connected to RPi pin 23 (SPI\_SCLK).
7. Pin 15 and pin 16 are are Vref and VDD connected to +3.3V of Raspberry Pi.

Here, we will be using RPi as master and MCP3008 as slave SPI device.

First Experiment:

Hardware required for the experiment is

1. Breadboard : On which connections have to be made.
2. MCP3008 IC.
3. LM35 Temperature sensor.
4. RPi.

Connect the LM35 to pin CHO of MCP3008.

First define the problem statement:

Interfacing an ADC with RPi to read the temperature of LM35 (Temperature Sensor) connected to one of the channels of IC.

After the connections are made , next part is code.

In order to program the SPI devices we need to import spidev library.

#######Importing the libraries###################

import spidev

#module to control spi devices.

import time

import math

import sys

# Open SPI bus

spi = spidev . SpiDev ()

# to create spi object

spi .open(0 ,0) #Clock polarity , Clock Phase

def readadc (adcnum ):

# define a function to read SPI data from MCP3008 chip.

if adcnum > 7 or adcnum < 0:

return −1 #if user inputs channel number other than 0-7 then return -1 to indicate that he has

given wrong channel number.

r = spi . xfer2 ([1 , 8 + adcnum << 4 , 0]) # Transfer 3 Bytes.

adcout = (( r [1] & 3) << 8) + r [2]

#use AND operation with the 2nd byte to get the last 4 bits , and then make way for the 3rd data byte with the “move 8 bits to left” << 8 operation   
    return adcout

Now define a function to calculate temperature from LM35 data as read by the ADC.

def ConvertTemp(data , places ):

temp = data∗100 #As the voltages changes by 100 mV for per degree Celsius change in temperature.

temp = round(temp , places ) # Rounding off the decimal value to two decimal places.

return temp

try :

while True :

value = readadc (0)

# call the readadc(0) function to read the channel 0 of MCP3008

volts = ( value ∗ 3.3) / 1024

# convert the digital value to voltage taking 3.3V as reference.

temperature = ConvertTemp( volts ,2)

#call the ConvertTemp( volts ,2) function to convert the voltage value to the appropriate temperature in degree Celsius upto 2 decimal places .

Now print the results so obtained from above conversions.

print(”Temp : {} volts : {} deg C: {}” .format( value , volts , temperature ))

time . sleep (0.5)

except KeyboardInterrupt :

pass

spi.close()

sys.exit()

If the Keyboard interrupt is pressed then close the SPI bus .

We are ready with the code. Save the code and execute it by pressing F5.

You can see the temperature values to be printed on the Python Shell.

**Exercise**

Now you can similarly try to read the values of distance measured by the Sharp IR sensor using the MCP3008.