|  |
| --- |
| Difference Between NAAC & NBA Accreditation - Haq Se EngineerPREC LONIJai Shriram Engineering College (@JSREC09) / Twitter**JAI SHRIRAM ENGINEERING COLLEGE**  **TIRUPPUR – 638 660**  Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai  Recognized by UGC & Accredited by NAACandNBA (CSE and ECE) |

**DEPARTMENT OF**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**IBM - Naan Mudhalvan**

**Internet of Things**

**Group 3**

**Phase 5 - Project Submission**

**NAME : SATHISH KUMAR.D**

**NM ID : au711221106030**

**YEAR : III ECE**

# Temperature Monitor

Temperature and humidity are vital data points in today’s industrial world. Monitoring environmental data for server rooms, commercial freezers, and production lines is necessary to keep things running smoothly. There are lots of solutions out there ranging from basic to complex and it can seem overwhelming on what your business needs and where to start.

We’ll walk through how to build and use a Raspberry Pi temperature sensor with different temperature sensors. This is a good place to start since these solutions are inexpensive, easy to do, and gives you a foundation to build off of for other environmental monitoring.

# ****RASPBERRY PI****

Raspberry Pi which having inbuilt wi-fi, which makes Raspberry Pi to suitable for IoT applications, so that by using IoT technology this monitoring system works by uploading the temperature value to the thingspeak cloud by this project you can able to learn to how to handle cloud-based application using API keys.

In this monitoring system, we used thingspeak cloud, the cloud which is suitable to view the sensor logs in the form of graph plots. Here we created one field to monitor the temperature value, that can be reconfigurable to monitor a number of sensor values in various fields. This basic will teach you to how to work with a cloud by using LM35 as a temperature sensor, to detect the temperature and to upload those values into the cloud.

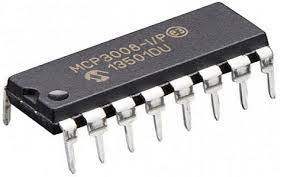
**HARDWARE REQUIRED**

* SD card
* Power supply
* VGA to HDMI converter :



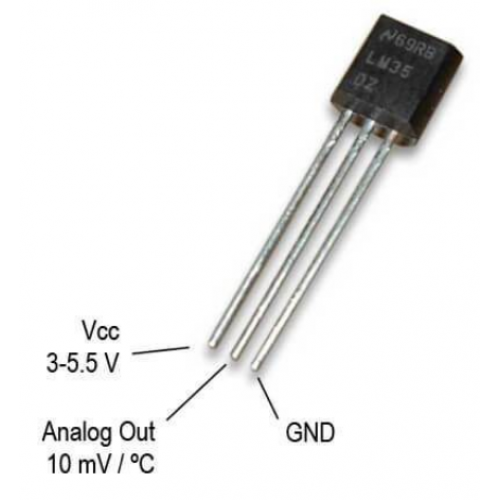
The converters register the analog signals delivered to them from the computer or any other device and then convert those signals into digital so that the HDMI monitor can read them. This process occurs without any obvious breaks or input, so you must connect the devices.

* MCP3008 (ADC IC):



MCP3008 is one of the famous Analog to Digital converter IC. This chip will add 8 channels of 10-bit analog input to your microcontroller or microcomputer project. It's super easy to use, and uses SPI so only 4 pins are required.

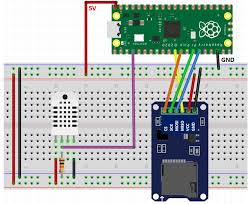
* Temperature sensor(LM35):



LM35 is a temperature sensor that outputs an analog signal which is proportional to the instantaneous temperature. The output voltage can easily be interpreted to obtain a temperature reading in Celsius. The advantage of lm35 over thermistor is it does not require any external calibration. The coating also protects it from self-heating. Low cost (approximately $0.95) and greater accuracy make it popular among hobbyists, DIY circuit makers, and students. Many low-end products take advantage of low cost, greater accuracy and used LM35 in their products.

**SOFTWARE REQUIRED:**

* Raspbian Stretch OS
* SD card Formatter
* Win32DiskImager (or) Etcher

****

**CODE:**

import time

import urllib2

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

myAPI = "paste your write API key here"

t=0

m=0

baseURL = '[https://api.thingspeak.com/update?api\_key=%s](https://api.thingspeak.com/update?api_key=%25s)' % myAPI

# read SPI data from MCP3008 chip, 8 possible adc's (0 thru 7)

def readadc(adcnum, clockpin, mosipin, misopin, cspin):

if ((adcnum > 7) or (adcnum < 0)):

return -1

GPIO.output(cspin, True)

GPIO.output(clockpin, False) # start clock low

GPIO.output(cspin, False) # bring CS low

commandout = adcnum

commandout |= 0x18 # start bit + single-ended bit

commandout <<= 3 # we only need to send 5 bits here

for i in range(5):

if (commandout & 0x80):

GPIO.output(mosipin, True)

else:

GPIO.output(mosipin, False)

commandout <<= 1

GPIO.output(clockpin, True)

GPIO.output(clockpin, False)

adcout = 0

# read in one empty bit, one null bit and 10 ADC bits

for i in range(12):

GPIO.output(clockpin, True)

GPIO.output(clockpin, False)

adcout <<= 1

if (GPIO.input(misopin)):

adcout |= 0x1

GPIO.output(cspin, True)

adcout >>= 1 # first bit is 'null' so drop it

return adcout

# change these as desired - they're the pins connected from the

# SPI port on the ADC to the Cobbler

SPICLK = 18

SPIMISO = 23

SPIMOSI = 24

SPICS = 25

# set up the SPI interface pins

GPIO.setup(SPIMOSI, GPIO.OUT)

GPIO.setup(SPIMISO, GPIO.IN)

GPIO.setup(SPICLK, GPIO.OUT)

GPIO.setup(SPICS, GPIO.OUT)

try:

while True:

temperature = readadc(0, SPICLK, SPIMOSI, SPIMISO, SPICS)

print("Temperature value")

print(temperature)

urllib2.urlopen(baseURL +"&field1=%s" % (str(temperature)))

time.sleep(15)

except KeyboardInterrupt:

print("stopping")

# ****The benefits of monitoring temperature:****

# Temperature monitoring in industrial processes is essential for ensuring quality, efficiency, and safety in various industries. It helps control the temperature of industrial processes, resulting in consistent product quality and reducing the risk of spoilage or contamination.

# By monitoring and controlling temperatures, companies can also prolong the life of critical equipment and reduce the risk of equipment damage due to overheating. Furthermore, temperature monitoring can help improve energy efficiency, thus reducing costs and increasing profitability. It also helps maintain a safe working environment by detecting and preventing hazardous temperature conditions.

# Moreover, temperature monitoring provides real-time data, enabling companies to make prompt corrective actions to improve process control.

# Finally, it is a requirement for regulatory compliance in industries such as food and pharmaceuticals. All of these benefits make temperature monitoring a critical aspect of industrial processes.