Estación Meteorológica

Ernesto chung, Pedro Romero

*Universidad Latina de Panamá; Ingeniería en Mecatrónica*

[ernesto2080@hotmail.com](mailto:ernesto2080@hotmail.com); [pedroromero96@gmail.com](mailto:pedroromero96@gmail.com)

**Abstract : this meteorological station was created with the purpose of being able to experience by itself how the devices that capture the information work in order to reach the personnel**

**Key Words: SenseHAT, MIME**

**Resumen: esta estacion meteorologica se creo con el fin de poder uno experimentar por si mismo como funciona los dispositivos que captan la información para asi hacerle llegar al personal**

**Palabras claves:SenseHAT, MIME**

1. Introducción

Una Raspberry Pi y un add on llamado SenseHat para determinar las principales variables meteorológicas de humedad, presión y temperatura. Luego el Raspberry Pi tomará dichos datos, los compilará a intervalos y enviará un correo electrónico con estos datos luego de un tiempo determinado. Además, se utilizará el SenseHat para mostrar la temperatura cada 5 segundos en su display de LEDs 8x8.

1. Justificacion

Usaremos la raspberry Pi para tener nuestra propia estación meteorológica y asi uno poder estar atentos a cualquier cambio meteorolico en su hogar o lugar de trabajo

1. Objetivo General

Crear un programa que utilice el Raspberry Pi y el SenseHAT para lograr mantener una observación constante sobre las principales variables ambientales en un lugar específico.

1. ESTADO DEL ARTE

El proyecto consiste en la utilización de un Raspberry Pi para mantener una observación constante sobre las principales variables climáticas en un

ambiente específico. Estas variables son temperatura, presión y humedad.

El Raspberry Pi no solo se mantiene leyendo la temperatura de manera constante sino que luego de un tiempo determinado (30 minutos como predeterminado) se envían

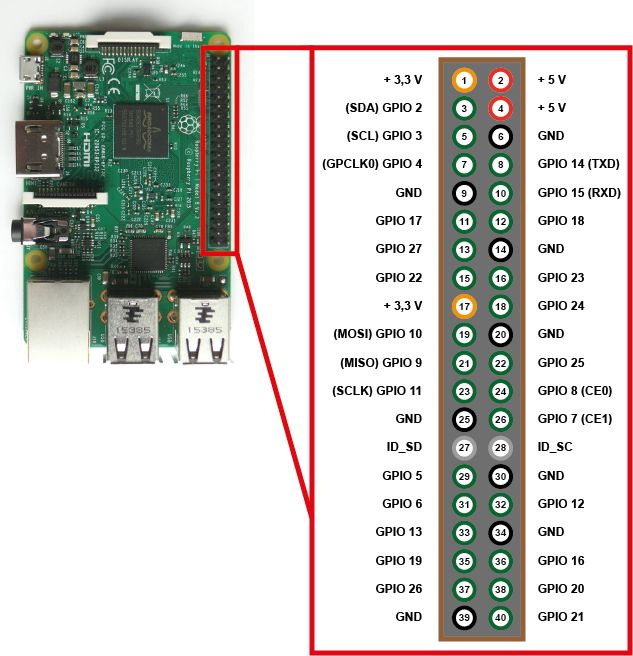
registros de ese periodo (se toman 6 registros divididos en tiempos iguales) al correo electrónico determinado en el programa.

Aparte de esto, cada 5 segundos se muestra la temperatura actual en el display 8x8 del SenseHat.

1. razones para querer monitorizar las condiciones atmosféricas de un entorno. Para el cultivo de ciertos vegetales en invernaderos resulta imprescindible el estudio y análisis de temperatura, humedad, luminosidad etc. Los animales en terrarios o acuarios deben cumplir con ciertas condiciones atmosféricas para garantizar la supervivencia de estos.
2. CONCLUSION

Se realizó un proyecto efectivo que permite observar de manera constante y a intervalos determinados las variables meteorológicas principales utilizando un Raspberry Pi y un SenseHAT.

1. Circuito Esquematico



BCM 8 (Atmel chip

select

BCM 25 (Atmel Prog Reset)

Masa

Masa

BCM 24 (IMU Interrupt)

BCM 23 (IMU Interrupt)

Masa

Masa

Masa

BCM 3(SDA)

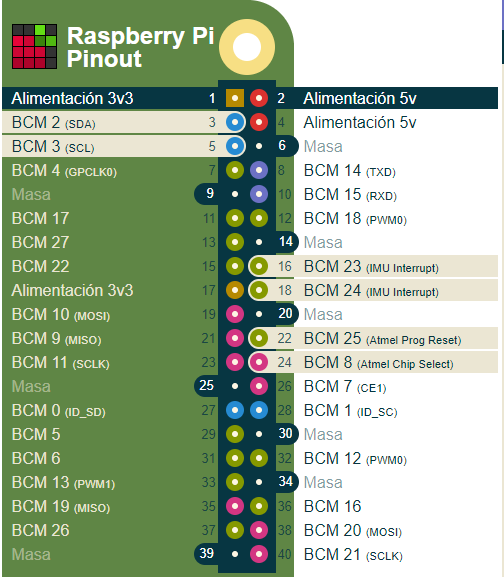
BCM 2 (SDA)

Alimentacion de 5v

Alimentacion de 3v3

Masa

Masa



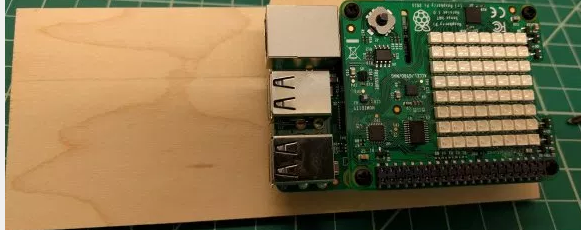
1. ENSAMBLAJE



Sensor de temperatura



Raspberry Pi 3



1. Librerias

import datetime. Proporciona clases Manipula la fecha y hora

import csv: crea objetos personalizados con forma de tabla a partir de los elementos de los archivos CSV.

import smtplib: módulo incorporado para enviar correos electrónicos utilizando el Protocolo simple de transferencia de correo (SMTP)

import os:nos permite acceder a funcionalidades dependientes del Sistema Operativo.

import sys: módulo proporciona acceso a algunas variables utilizadas o mantenidas por el intérprete y a funciones que interactúan fuertemente con el intérprete.

import time: manejar tareas relacionadas con el tiempo.

import SenseHat: importa todo los datos del sensor a la rapsberry

import Config: módulo para configurar programas Python que tiene como objetivo ofrecer más potencia y flexibilidad que el módulo ConfigParser existente .

1. CONCLUSION

Se realizó un proyecto efectivo que permite observar de manera constante y a intervalos determinados las variables meteorológicas principales utilizando un Raspberry Pi y un SenseHAT.

1. Bibliografia

<https://es.pinout.xyz/pinout/sense_hat>

<https://docs.python.org/2/library/datetime.html>

<https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/import-csv?view=powershell-6>

<https://realpython.com/python-send-email/>

<https://docs.python.org/3.4/library/email-examples.html>

<https://uniwebsidad.com/libros/python/capitulo-10/modulos-de-sistema>

<https://sites.google.com/site/cursodepython/modulo-sys>

<https://docs.python.org/2/library/sys.html>

<https://www.programiz.com/python-programming/time>

1. CODIGO

from \_\_future\_\_ import print\_function

import datetime

import csv

import smtplib

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

from email.mime.base import MIMEBase

from email import encoders

import datetime

import os

import sys

import time

from sense\_hat import SenseHat

from config import Config

sense = SenseHat()

mailmin = 30 # minutes

send\_mail = True

SINGLE\_HASH = "#"

HASHES "########################################"

SLASH\_N = "\n"

b = [0, 0, 255]

r = [255, 0, 0]

e = [0, 0, 0]

empty = [

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e,

e, e, e, e, e, e, e, e

]

def get\_cpu\_temp():

res = os.popen('vcgencmd measure\_temp').readline()

return float(res.replace("temp=", "").replace("'C\n", ""))

def get\_smooth(x):

if not hasattr(get\_smooth, "t"):

get\_smooth.t = [x, x, x]

get\_smooth.t[2] = get\_smooth.t[1]

get\_smooth.t[1] = get\_smooth.t[0]

get\_smooth.t[0] = x

xs = (get\_smooth.t[0] + get\_smooth.t[1] + get\_smooth.t[2]) / 3

return xs

def get\_temp():

t1 = sense.get\_temperature\_from\_humidity()

t2 = sense.get\_temperature\_from\_pressure()

t = (t1 + t2) / 2

t\_cpu = get\_cpu\_temp()

t\_corr = t - ((t\_cpu - t) / 1.5)

t\_corr = get\_smooth(t\_corr)

return t\_corr

def main():

global last\_temp

mailcount = 0

while 1:

current\_second = datetime.datetime.now().second

if (current\_second == 0) or ((current\_second % 5) == 0):

calc\_temp = get\_temp()

temp\_c = round(calc\_temp, 1)

hum = round(sense.get\_humidity(), 0)

pres = round(sense.get\_pressure() \* 0.0295300, 1)

print("Temp: %sC , Pressure: %s inHg, Humidity: %s%%" % (temp\_c, pres, hum))

sense.show\_message("%sC" % (temp\_c), text\_colour=[0, 0, 255], back\_colour=[0, 0, 0])

sense.set\_pixels(empty)

mailcount += 1

if mailcount == mailmin\*12:

print ("%s mail" % (mailcount))

with open('att.csv','w') as csvfile:

filewriter = csv.writer (csvfile, delimiter=',',

quotechar='|', quoting=csv.QUOTE\_MINIMAL)

filewriter.writerow(['Time','Temperature','Pressure','Humidity'])

filewriter.writerow(['5','%s'%(temp\_5),'%s'%(pres\_5),'%s'%(hum\_5)])

filewriter.writerow(['10','%s'%(temp\_10),'%s'%(pres\_10),'%s'%(hum\_10)])

filewriter.writerow(['15','%s'%(temp\_15),'%s'%(pres\_15),'%s'%(hum\_15)])

filewriter.writerow(['20','%s'%(temp\_20),'%s'%(pres\_20),'%s'%(hum\_20)])

filewriter.writerow(['25','%s'%(temp\_25),'%s'%(pres\_25),'%s'%(hum\_25)])

filewriter.writerow(['30','%s'%(temp\_30),'%s'%(pres\_30),'%s'%(hum\_30)])

mailcount = 0

fromaddr = "cisjuanrus@gmail.com"

toaddr = "pedroromero96@gmail.com"

msg = MIMEMultipart()

# storing the senders email address

msg['From'] = fromaddr

# storing the receivers email address

msg['To'] = toaddr

# storing the subject

date=datetime.datetime.now()

msg['Subject'] = "Temperatura de %s"%(date)

# string to store the body of the mail

body = "Incluido archivo CSV"

# attach the body with the msg instance

msg.attach(MIMEText(body, 'plain'))

# open the file to be sent

filename = "att.csv"

attachment = open("att.csv", "rb")

# instance of MIMEBase and named as p

p = MIMEBase('application', 'octet-stream')

# To change the payload into encoded form

p.set\_payload((attachment).read())

# encode into base64

encoders.encode\_base64(p)

p.add\_header('Content-Disposition', "attachment; filename= %s" % filename)

# attach the instance 'p' to instance 'msg'

msg.attach(p)

# creates SMTP session

s = smtplib.SMTP('smtp.gmail.com', 587)

# start TLS for security

s.starttls()

# Authentication

s.login(fromaddr, "cisruspass")

# Converts the Multipart msg into a string

text = msg.as\_string()

# sending the mail

s.sendmail(fromaddr, toaddr, text)

if mailcount == mailmin\*2:

temp\_5=temp\_c

pres\_5=pres

hum\_5=hum

if mailcount == mailmin\*4:

temp\_10=temp\_c

pres\_10=pres

hum\_10=hum

if mailcount == mailmin\*6:

temp\_15=temp\_c

pres\_15=pres

hum\_15=hum

if mailcount == mailmin\*8:

temp\_20=temp\_c

pres\_20=pres

hum\_20=hum

if mailcount == mailmin\*10:

temp\_25=temp\_c

pres\_25=pres

hum\_25=hum

if mailcount == mailmin\*12:

temp\_30=temp\_c

pres\_30=pres

hum\_30=hum

time.sleep(1)

print("Leaving main()")

if \_\_name\_\_ == "\_\_main\_\_":

try:

main()

except KeyboardInterrupt:

print("\nExiting application\n")

sys.exit(0)