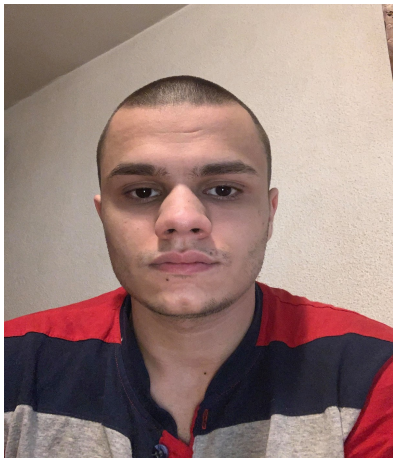
Name: Homemade Thermometer

Lela Roxana-Andreea -1305A - roxana.lela15@gmail.com

Carcea Paul - 1305A - carceapaul9@gmail.com



Elevator pitch:

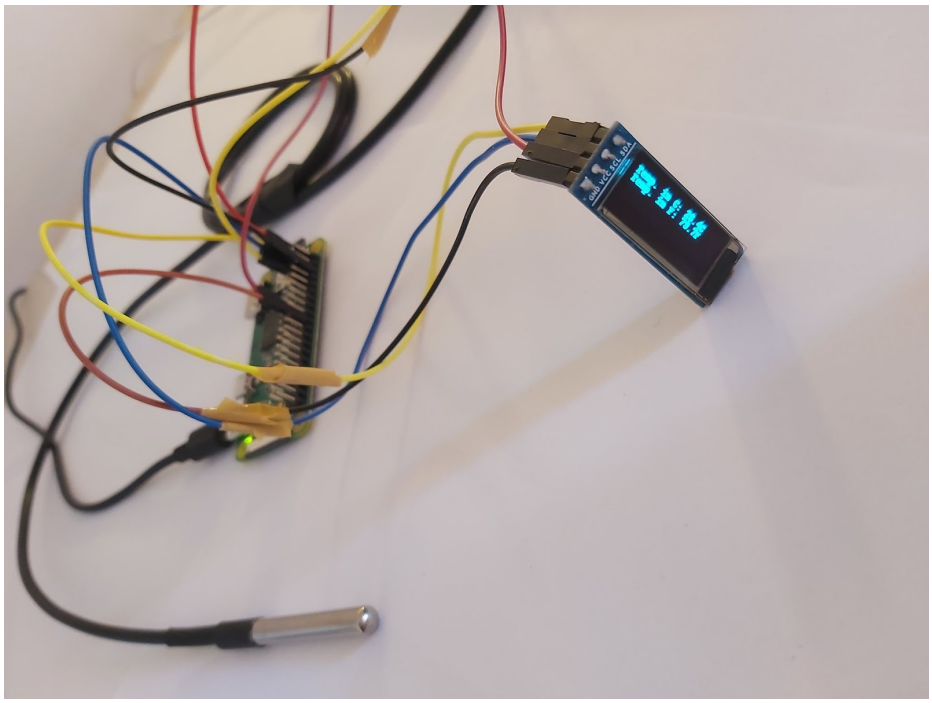
Have you ever thought of a homemade thermometer with a raspberry pi that updates every second? Well, we managed to do it. If you are curious to see how we did it, follow the project tutorial. This project it's easy to do and really exciting. Enjoy it!

Story:

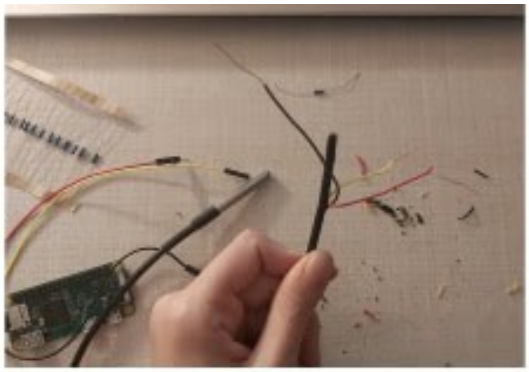
Our main target was to make an easy and fun project for the potential users, not too hard

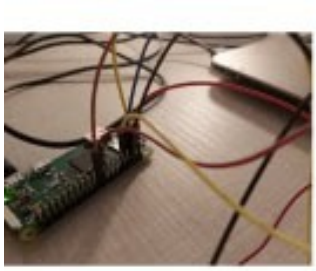
and to have a good time making it. We build a simple digital thermometer useful in any circumstances.

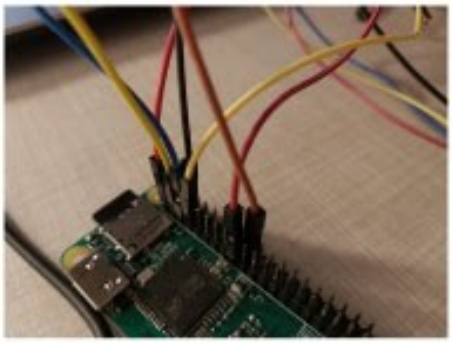
Cover image:



Intermediate steps:







Demo:

https://drive.google.com/file/d/11IbvDWrGnlDoQoaXObTtCojUQlcFv4Y8/view?usp=sharing

Components:

- Raspberry Pi ZERO WH V1.3 1GHz 512MB
- SD card - 8GB
- USB cable (for power)
- DS18B20 Temperature Sensor
- 0.91 inch 128x32 OLED display module
- 4.7 kΩ (used as pull-up resistor)

- some jumper wires

Software:

- Raspberry PI Raspbian
- Python 2

Steps:

I started off by creating an SD card with the latest Raspbian image. Then I made sure this was up-to-date by running the following commands :

sudo apt update

sudo apt upgrade

Display Module Setup

My screen had four pins, two for power and two for the I2C interface.

I connected them directly to the Raspberry Pi's GPIO header using the following scheme :

OLED Pin Pi GPIO Pin Notes

- Vcc 17 3.3V
- Gnd 20 Ground
- SCL 5 I2C SCL
- SDA 3 I2C SCA

Enable I2C Interface

The I2C interface is disabled by default so you need to enable it. You can do this within the raspi-config tool on the command line by running :

sudo raspi-config

Next:

sudo apt install -y python-dev

sudo apt install -y python-smbus i2c-tools

sudo apt install -y python-pil

sudo apt install -y python-pip

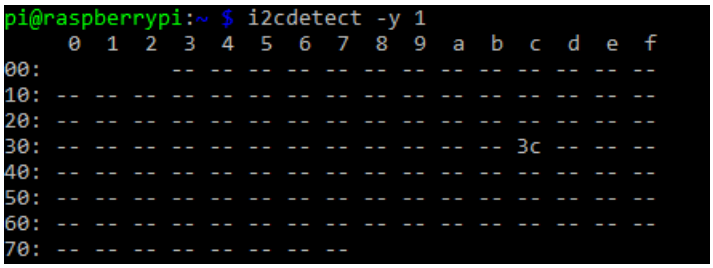
sudo apt install -y python-setuptools

(python2)

Finding the OLED Display Module's Address

With the I2C libraries installed I used the i2cdetect command to find the module on the I2C bus.

i2cdetect -y 1



Install OLED Python Library

sudo apt install -y git

git clone https://github.com/adafruit/Adafruit_Python_SSD1306.git

cd Adafruit_Python_SSD1306

sudo python setup.py install

DS18B20 Circuit Diagram

- black wire -> pin 6 --> GND
- red wire -> pin 1 --> 3.3 V
- yellow wire -> pin 7 --> GPIO4 (with a 4.7 kΩ pull-up resistor)

Set the sensor

sudo nano /boot/config.txt

dtoverlay=w1-gpio

sudo reboot

sudo modprobe w1-gpio

sudo modprobe w1-therm

cd /sys/bus/w1/devices

ls

cd 28-0517a1ac64ff

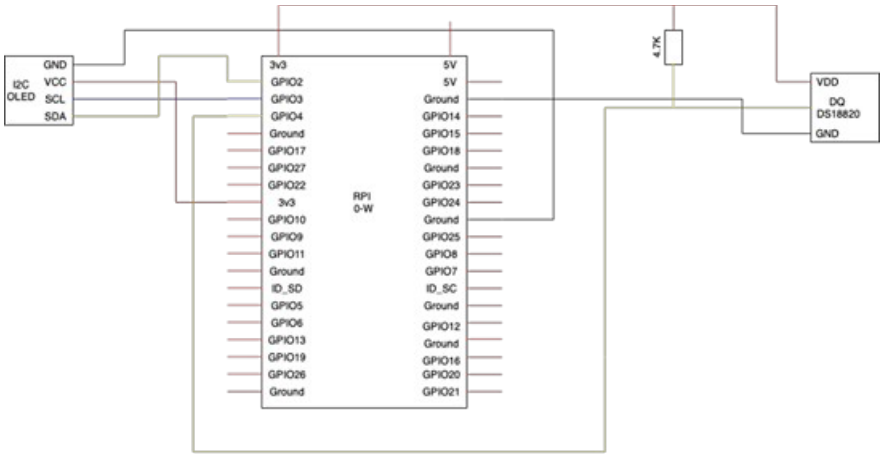
cat w1_slave

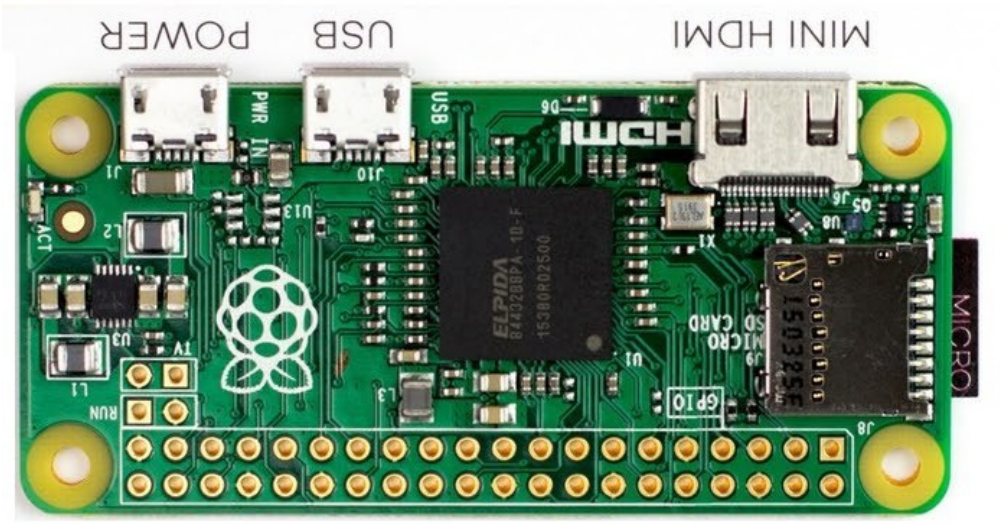
- The interface to the DS18B20 shows up here

```
pi@raspberrypi:/sys/bus/w1/devices/28-0517a1ac64ff $ ls
driver  hwmon  id  name  power  subsystem  uevent  w1_slave
pi@raspberrypi:/sys/bus/w1/devices/28-0517a1ac64ff $ cat w1_slave
a4 01 4b 46 7f ff 0c 10 da : crc=da YES
a4 01 4b 46 7f ff 0c 10 da t=26250
pi@raspberrypi:/sys/bus/w1/devices/28-0517a1ac64ff $
```

- The 28-0517a1ac64ff directory represents this particular thermometer.
- The 28 identifies a DS18B20 device.
- The 0517a1ac64ff represents the unique device id for this particular thermometer. This allows you to have multiple thermometers on the same data line.
- The second line, now it is t=26250, is the temp in Celsius multiplied by 1000.

Schematics:





Code:

```python
import time
import glob
import Adafruit_GPIO.SPI as SPI
import Adafruit_SSD1306
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
import subprocess

# Raspberry Pi pin configuration:
RST = None     # on the PiOLED this pin isnt used
DC = 23
SPI_PORT = 0
SPI_DEVICE = 0

# 128x32 display with hardware I2C:
disp = Adafruit_SSD1306.SSD1306_128_32(rst=RST)

disp.begin()

# Clear display
disp.clear()
disp.display()
```

```
# Create blank image for drawing
width = disp.width
height = disp.height

image = Image.new('1', (width, height))

# Get drawing object to draw on image.
draw = ImageDraw.Draw(image)

# Draw a black filled box to clear the image.
draw.rectangle((0,0,width,height), outline=0, fill=0)

padding = -2
top = padding
bottom = height-padding
x = 0

# Load default font.
font = ImageFont.load_default()

```

```
base_dir = '/sys/bus/w1/devices/'
device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

#CELSIUS CALCULATION
def read_temp_c():
    lines = read_temp_raw()
 while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = int(temp_string) / 1000.0
        temp_c = str(round(temp_c, 1))
        return temp_c
```

```
#FAHRENHEIT CALCULATION
def read_temp_f():
    lines = read_temp_raw()
 while lines[0].strip()[-3:] != 'YES':
    time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_f = (int(temp_string) / 1000.0) * 9.0 / 5.0 + 32.0
    temp_f = str(round(temp_f, 1))
        return temp_f

while True:
    # Draw a black filled box to clear the image.
    draw.rectangle((0,0,width,height), outline=0, fill=0)
    draw.text((x, top),"Temp. in C: " + read_temp_c() + "C",  font=font, fill=255)
    draw.text((x, top),"Temp. in F: " + read_temp_f() + "F",  font=font, fill=255)
    # Display image.
    disp.image(image)
    disp.display()
    time.sleep(.1)
```

References:

https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

https://learn.adafruit.com/adafruit-pioled-128x32-mini-oled-for-raspberry-pi/usage

https://cdn.sparkfun.com/assets/learn_tutorials/6/7/6/PiZero_1.pdf