

[Name: Homemade Thermometer](#)

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



Carcea Paul - 1305A - carceapaul9@gmail.com



Elevator pitch:

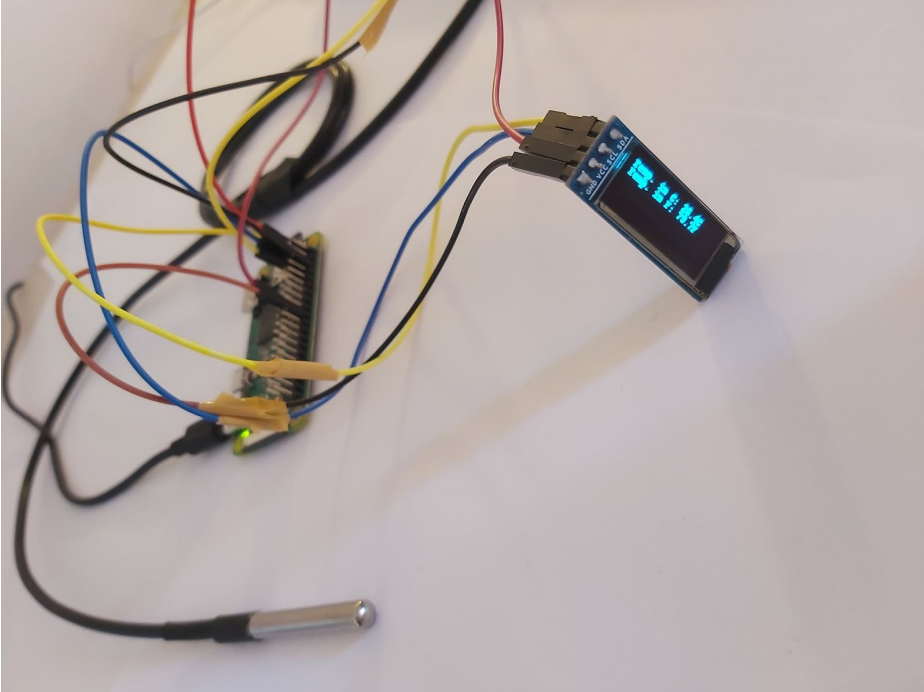
Have you ever thought of a homemade thermometer with a raspberry pi that updates every second? Well, we managed to do it. If you are curious to see how we did it, follow the project tutorial. This project it's easy to do and really exciting. Enjoy it!

Story:

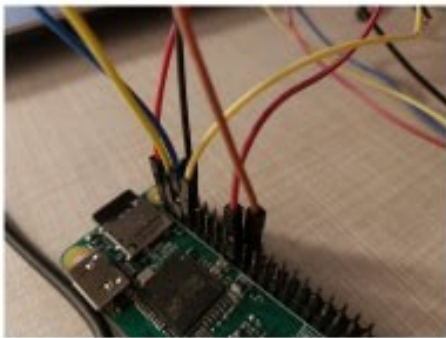
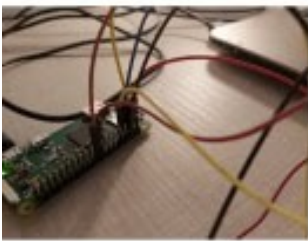
Our main target was to make an easy and fun project for the potential users, not too hard

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/11lbvDWrGnIDoQoaXObTtCojUQlcFv4Y8/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 k Ω (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 :

<u>OLED Pin</u>	<u>Pi GPIO Pin</u>	<u>Notes</u>
• <u>Vcc</u>	<u>17</u>	<u>3.3V</u>
• <u>Gnd</u>	<u>20</u>	<u>Ground</u>
• <u>SCL</u>	<u>5</u>	<u>I2C SCL</u>
• <u>SDA</u>	<u>3</u>	<u>I2C SCA</u>

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
```

```
pi@raspberrypi:~$ i2cdetect -y 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  3c  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

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
```

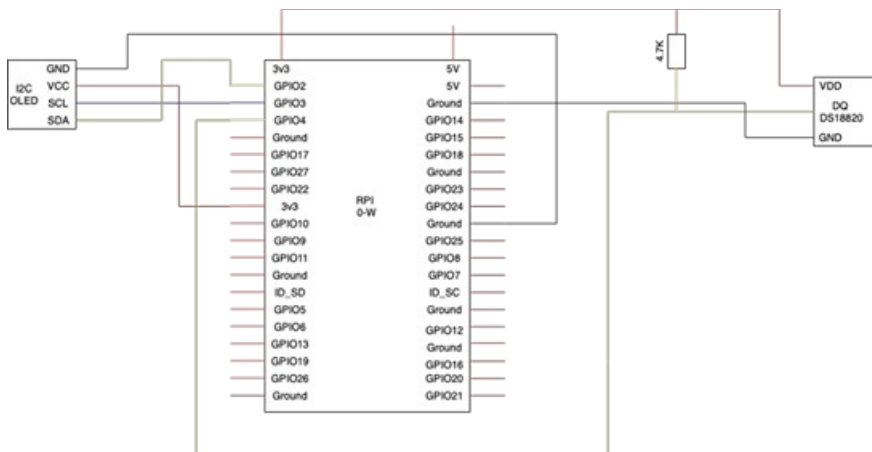
```
cat w1_slave
```

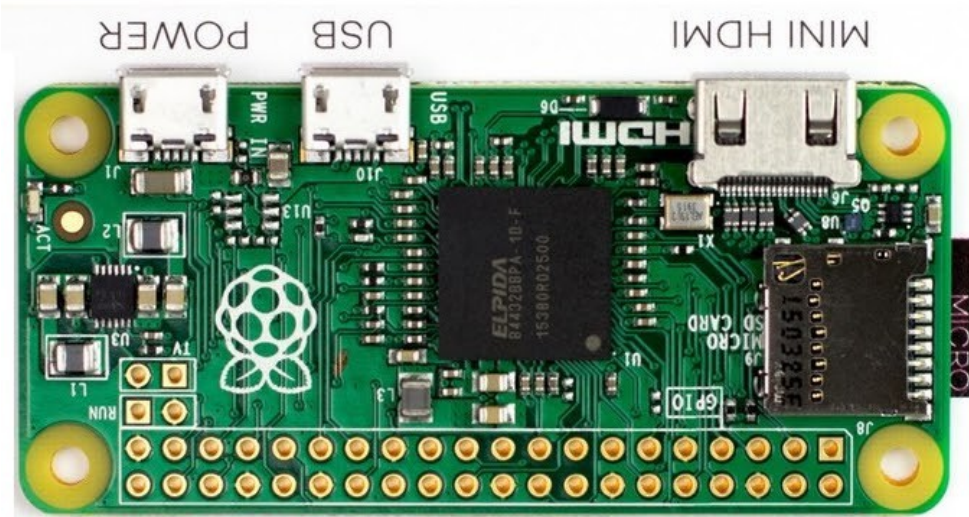
- The interface to the DS18B20 shows up here

```
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:





Pin Number	Function	Pin Number	Function
1	3V3	39	GPIO28
2	5V	40	GPIO29
3	GPIO2	1	GPIO1
4	5V	2	GPIO2
5	GPIO3	3	GPIO3
6	Ground	4	GPIO4
7	GPIO4	5	GPIO5
8	GPIO14	6	GPIO6
9	GPIO17	7	GPIO7
10	GPIO15	8	GPIO8
11	GPIO17	9	GPIO9
12	GPIO18	10	GPIO10
13	GPIO27	11	GPIO11
14	GPIO20	12	GPIO12
15	GPIO22	13	GPIO13
16	3V3	14	GPIO14
17	5V	15	GPIO15
18	GPIO19	16	GPIO16
19	GPIO21	17	GPIO17
20	GPIO23	18	GPIO18
21	GPIO24	19	GPIO19
22	GPIO25	20	GPIO20
23	GPIO26	21	GPIO21
24	GPIO27	22	GPIO22
25	GPIO28	23	GPIO23
26	GPIO29	24	GPIO24
27	GPIO30	25	GPIO25
28	GPIO31	26	GPIO26
29	GPIO32	27	GPIO27
30	GPIO33	28	GPIO28
31	GPIO34	29	GPIO29
32	GPIO35	30	GPIO30
33	GPIO36	31	GPIO31
34	GPIO37	32	GPIO32
35	GPIO38	33	GPIO33
36	GPIO39	34	GPIO34
37	GPIO40	35	GPIO35
38	GPIO41	36	GPIO36
39	GPIO42	37	GPIO37
40	GPIO43	38	GPIO38

Code:

```

1 import time
2 import glob
3 import Adafruit_GPIO.SPI as SPI
4 import Adafruit_SSD1306
5 from PIL import Image
6 from PIL import ImageDraw
7 from PIL import ImageFont
8 import subprocess
9
10 # Raspberry Pi pin configuration:
11 RST = None # on the PiOLED this pin isnt used
12 DC = 23
13 SPI_PORT = 0
14 SPI_DEVICE = 0
15
16 # 128x32 display with hardware I2C:
17 disp = Adafruit_SSD1306.SSD1306_128_32(rst=RST)
18
19 disp.begin()
20
21 # Clear display
22 disp.clear()
23 disp.display()

```



```

25 # Create blank image for drawing
26 width = disp.width
27 height = disp.height
28
29 image = Image.new('1', (width, height))
30
31 # Get drawing object to draw on image.
32 draw = ImageDraw.Draw(image)
33
34 # Draw a black filled box to clear the image.
35 draw.rectangle((0,0,width,height), outline=0, fill=0)
36
37 padding = -2
38 top = padding
39 bottom = height - padding
40 x = 0
41
42 # Load default font.
43 font = ImageFont.load_default()
44

```

```

45 base_dir = '/sys/bus/w1/devices/'
46 device_folder = glob.glob(base_dir + '28*')[0]
47 device_file = device_folder + '/w1_slave'
48
49 def read_temp_raw():
50     f = open(device_file, 'r')
51     lines = f.readlines()
52     f.close()
53     return lines
54
55 #CELSIUS CALCULATION
56 def read_temp_c():
57     lines = read_temp_raw()
58     while lines[0].strip()[-3:] != 'YES':
59         time.sleep(0.2)
60         lines = read_temp_raw()
61     equals_pos = lines[1].find('t=')
62     if equals_pos != -1:
63         temp_string = lines[1][equals_pos+2:]
64         temp_c = int(temp_string) / 1000.0
65         temp_c = str(round(temp_c, 1))
66     return temp_c

```

```

68 #FAHRENHEIT CALCULATION
69 def read_temp_f():
70     lines = read_temp_raw()
71     while lines[0].strip()[-3:] != 'YES':
72         time.sleep(0.2)
73         lines = read_temp_raw()
74     equals_pos = lines[1].find('t=')
75     if equals_pos != -1:
76         temp_string = lines[1][equals_pos+2:]
77         temp_f = (int(temp_string) / 1000.0) * 9.0 / 5.0 + 32.0
78         temp_f = str(round(temp_f, 1))
79     return temp_f
80
81 while True:|
82     # Draw a black filled box to clear the image.
83     draw.rectangle((0,0,width,height), outline=0, fill=0)
84     draw.text((x, top), "Temp. in C: " + read_temp_c() + "C", font=font, fill=255)
85     draw.text((x, top), "Temp. in F: " + read_temp_f() + "F", font=font, fill=255)
86     # Display image.
87     disp.image(image)
88     disp.display()
89     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