

УНИВЕРЗИТЕТ У НИШУ  
ЕЛЕКТРОНСКИ ФАКУЛТЕТ У НИШУ  
КАТЕДРА ЗА МЕРЕЊА  
Предмет: МЕРЕЊЕ НЕЕЛЕКТРИЧНИХ ВЕЛИЧИНА

Тема семинарског рада:

## Аквизиција података са сензора помоћу Raspberry Pi 3 Model B+ платформе

Студент:  
Вељко Митић, бр. индекса 18776

Предметни наставник:  
проф. др Милан Динчић

У Нишу, јул 2024. године

## Садржај

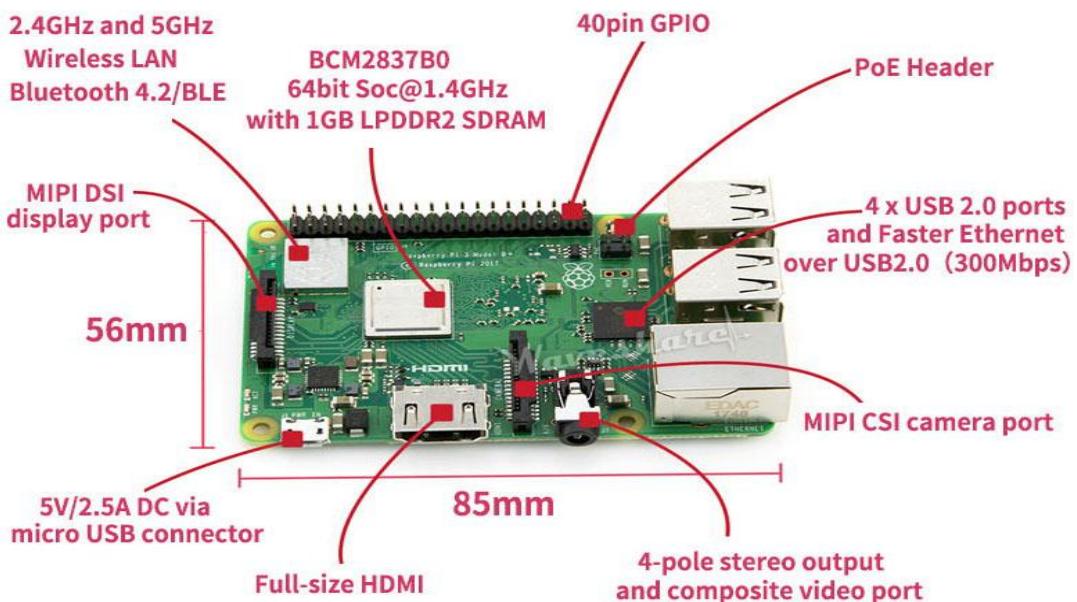
Увод .....	3
1. Raspberry Pi 3 Model B+ .....	3
1.1 Припрема microSD картице са инсталirаним оперативним системом за Raspberry Pi 3 Model B+ .....	5
1.2 Основне инструкције за кретање кроз командну линију (CLI) на Raspberry Pi Model B+ .....	7
1.3 Ажурирање и надоградња Raspberry Pi-а .....	8
1.4 Remote Access .....	8
1.4.1 SSH (Secure Shell) .....	8
1.4.2 RealVNC (Virtual Network Computing) .....	9
2. Повезивање сензора и Raspberry Pi 3 Model B+ .....	11
2.1 Повезивање сензора који дају дигитални излаз са Raspberry Pi .....	11
2.2 Повезивање сензора који дају аналогни излаз преко АД конвертора ...	11
2.2.1 PCF8591 АД конвертор .....	12
2.2.2 Примери АД конвертора веће резолуције .....	12
2.3 Повезивање сензора који имају I2C излаз са Raspberry Pi-ем .....	13
2.4 Библиотека Time .....	14
2.5 Разлика између BCM и BOARD .....	15
2.5.1 BCM Нумерација .....	15
2.5.2 BOARD Нумерација .....	15
2.6 Карактеристика GPIO пинова .....	16
3. Примери аквизиције података са сензора помоћу Raspberry Pi 3 Model B+ .	17
3.1 Детекција објекта .....	17
Повезивање сензора са Raspberry Pi-м .....	18
Код у Raspberry Pi-у .....	19
3.2 Детекција Светlostи .....	21
Повезивање сензора са Raspberry Pi-м .....	22
Код у Raspberry Pi-у .....	23
3.3 Мерење Температуре и Влажности .....	25
Повезивање сензора са Raspberry Pi-м .....	26
Код у Raspberry Pi-у .....	27
3.4 Паркинг Систем .....	33
Повезивање сензора са Raspberry Pi-м .....	33
Код у Raspberry Pi-у .....	35
4. Смештање мерних података на Raspberry Pi microSD картици или USB Flesh меморији .....	38
4.1 Смештање мерних података на microSD картицу (фолдер унутар Raspberry Pi уређаја) .....	38
Код у Raspberry Pi-у .....	39
4.2 Смештање мерних података на USB Flesh меморију .....	40
Код у Raspberry Pi-у .....	40
5. Гашење Raspberry Pi уређаја .....	41
Литература .....	42

## Увод

Raspberry Pi је напредна платформа која интегрише хардвер и софтвер за различите микрорачунарске примене. Користећи Raspberry Pi плочу заједно са одговарајућим софтервом и електронским компонентама, могуће је остварити бројне пројекте у областима аутоматизације, мониторинга и управљања системима. Ова плоча се широко користи у кућној аутоматизацији, медијским центрима и разним DIY пројектима.

### 1. Raspberry Pi 3 Model B+

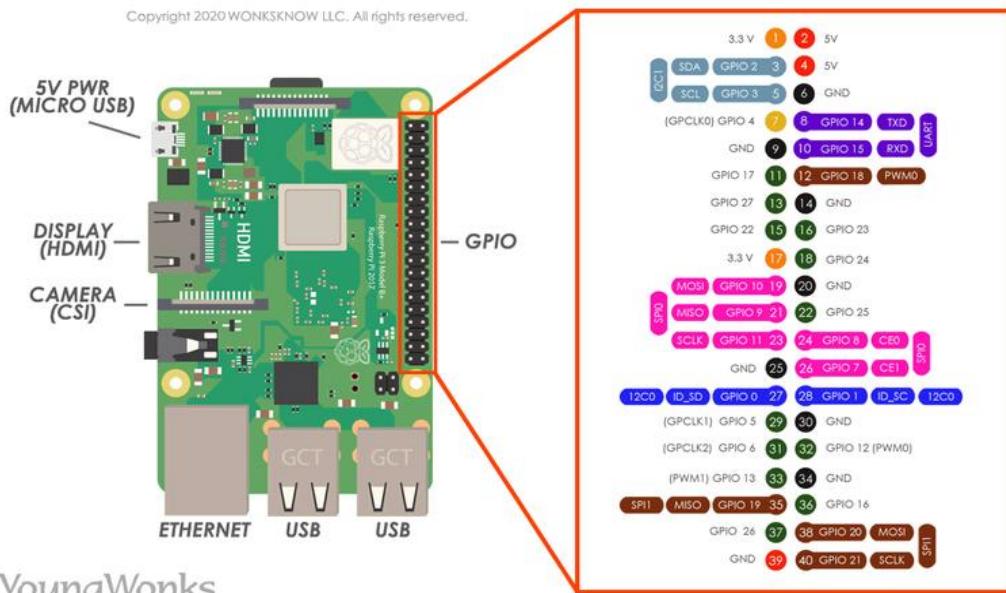
Кључна компонента Raspberry Pi 3 model B+ плоче је њен микропроцесор. Raspberry Pi 3 Model B+ је опремљен четворојезгарним ARM Cortex-A53 процесором који ради на такту од 1.4 GHz, уз 1GB RAM меморије. Оперативни систем и подаци се чувају на microSD картици. Плоча поседује улазне и излазне конекторе (GPIO пинове) који омогућавају повезивање и комуникацију са разноврсним сензорима, актуаторима и другим периферним уређајима.



Слика 1. Приказ Raspberry Pi 3 model B+ плочице

Raspberry Pi 3 Model B+ има више USB портова (четири USB 2.0 порта) који омогућавају повезивање различитих периферних уређаја, као што су тастатуре, мишеви и USB дискови. Плоча такође има HDMI порт за повезивање са монитором или телевизором, као и аудио излаз.

Напајање плоче се обезбеђује преко micro-USB конектора са напоном од 5V, а може се напајати и преко GPIO пинова. Овај модел има и уграђени Gigabit Ethernet који омогућава брузу жичану мрежну повезаност, као и dual-band 802.11ac Wi-Fi и Bluetooth 4.2 за бежичну комуникацију.



Слика 2. Приказ GPIO пинова

GPIO (General Purpose Input/Output) пинови омогућавају повезивање различитих уређаја и сензора. Raspberry Pi 3 Model B+ има 40 GPIO пинова који се могу конфигурисати као улазни или излазни пинови преко софтвера. Ови пинови подржавају различите комуникационе протоколе као што су I2C, SPI и UART.

За програмирање Raspberry Pi користи се оперативни систем Raspbian, који је заснован на Debian Linux-у и оптимизован за овај уређај. Такође подржава друге оперативне системе као што су Ubuntu и Windows 10 IoT Core. Програмски језици који се најчешће користе укључују Python, C, C++, Java и многе друге.

Развој софтвера за Raspberry Pi се обично врши у интегрисаном развојном окружењу као што је Thonny (за Python), Geany, или Visual Studio Code. Користећи ове алате, корисници могу писати, тестирати и извршавати свој код на Raspberry Pi-у.

Да би започели рад са Raspberry Pi, потребно је урадити следеће конфигурационе кораке:

1. Поставити microSD картицу са инсталirаним оперативним системом у одговарајући слот на плочи.
2. Повезати плочу са монитором преко HDMI порта.
3. Повезати тастатуру и миш преко USB портова.
4. Повезати плочу са извором напајања преко micro-USB конектора.
5. Укључити Raspberry Pi и конфигурисати мрежна подешавања и остале основне параметре.

Raspberry Pi 3 Model B+ је веома флексибилан и моћан алат који омогућава корисницима да реализују разноврсне пројекте и истражују свет микрорачунарских система.

## 1.1 Припрема microSD картице са инсталirаним оперативним системом за Raspberry Pi 3 Model B+

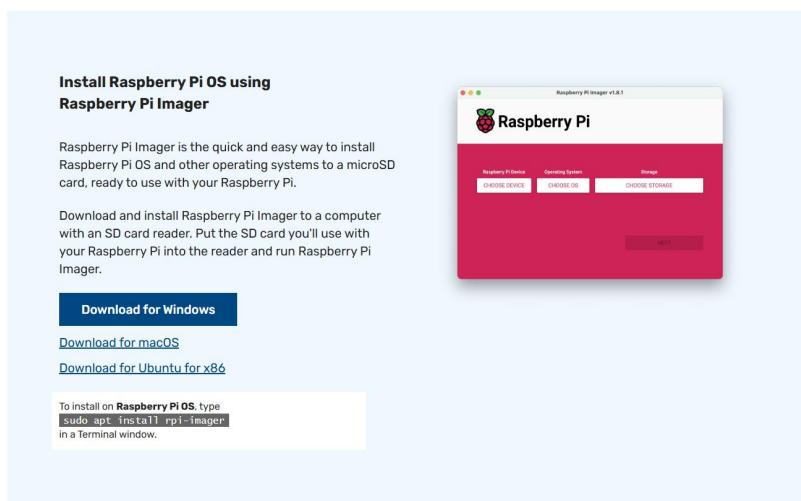
Да би сте припремили microSD картицу са инсталirаним оперативним системом за ваш Raspberry Pi 3 Model B+, следите следеће кораке:

### **Потребни алати и материјали:**

- **microSD картица:** Препоручена величина је најмање 8GB, али је боље користити веће картице (16GB или више) за боље перформансе.
- **Рачунар са читачем microSD картица.**
- **Интернет конекција:** За преузимање оперативног система.

### **Корак 1: Преузимање оперативног система**

- Посетите званични сајт Raspberry Pi: Raspberry Pi Downloads.

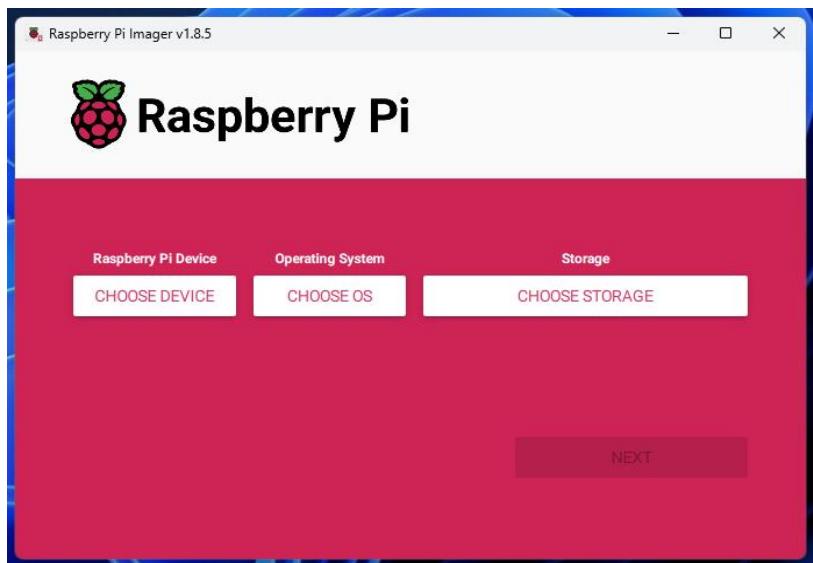


Слика 3. Приказ странице за скидање Raspberry Pi imager-a

- Преузмите Raspberry Pi Imager, званични алат за инсталацију оперативног система.
- Инсталирајте и покрените Raspberry Pi Imager на вашем рачунару.

### **Корак 2: Припрема microSD картице**

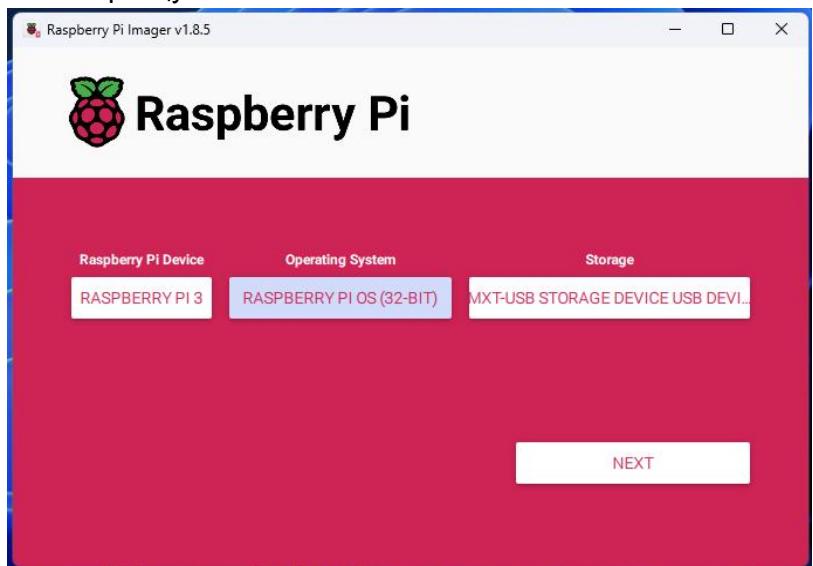
- Уметните microSD картицу у читач картица на вашем рачунару.
- Покрените Raspberry Pi



Слика 4. Приказ Raspberry Pi интерфејса

### **Корак 3: Избор и инсталација оперативног система**

- У Raspberry Pi Imager-у, кликните на "Operating System".
- Изаберите оперативни систем који желите да инсталаријате. Најчешће је препоручен избор "Raspberry Pi OS (32-bit)".
- Након што изаберете оперативни систем, кликните на "Storage" и изаберите вашу microSD картицу са листе.

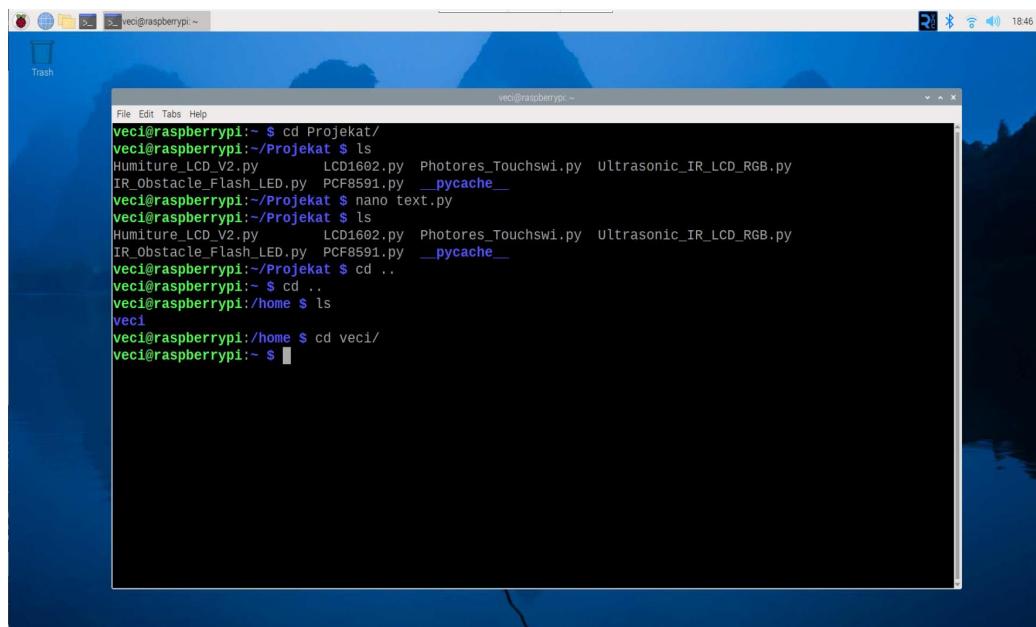


Слика 5. Одабир Raspberry Pi уређаја и система

- Кликните на "Next" да бисте започели процес инсталације. Овај процес ће форматирати microSD картицу и инсталирати изабрани оперативни систем. Ово може потрајати неколико минута.

## 1.2 Основне инструкције за кретање кроз командну линију (CLI) на Raspberry Pi Model B+

- **Преглед садржаја директоријума:**  
ls - Ова команда приказује листу датотека и поддиректоријума у тренутном директоријуму.
- **Промена директоријума:**  
cd <име\_директоријума> - Замени <име\_директоријума> са именом директоријума у који желите да пређете.
- **Пређи на родитељски директоријум:**  
cd .. - Ова команда вас враћа један ниво горе у директоријумској структури.
- **Пређи у почетни директоријум:**  
cd ~ - Ова команда вас враћа у ваш кућни директоријум.
- **Креирање новог директоријума:**  
mkdir <име\_директоријума> - Замени <име\_директоријума> са жељеним именом новог директоријума.
- **Брисање датотеке:**  
rm <име\_датотеке> - Замени <име\_датотеке> са именом датотеке коју желите да обришете.
- **Уређивање датотеке:**  
nano <име\_датотеке> - Ово отвара датотеку у Nano текст едитору. Замени <име\_датотеке> са именом датотеке коју желите да уређујете.



The screenshot shows a terminal window titled 'Terminal' with the command line 'veci@raspberrypi:~'. The window displays the following command history:

```
File Edit Tabs Help
veci@raspberrypi:~ $ cd Projekat/
veci@raspberrypi:/Projekat $ ls
Humiture_LCD_V2.py      LCD1602.py  Photores_Touchswi.py  ultrasonic_IR_LCD_RGB.py
IR_Obstacle_Flash_LED.py PCF8591.py  __pycache__
veci@raspberrypi:/Projekat $ nano text.py
veci@raspberrypi:/Projekat $ ls
Humiture LCD V2.py      LCD1602.py  Photores_Touchswi.py  ultrasonic_IR_LCD_RGB.py
IR_Obstacle_Flash_LED.py PCF8591.py  __pycache__
veci@raspberrypi:/Projekat $ cd ..
veci@raspberrypi:~/home $ ls
veci
veci@raspberrypi:~/home $ cd veci/
veci@raspberrypi:~ $
```

Слика 6. Приказ командног прозора

## 1.3 Ажурирање и надоградња Raspberry Pi-а

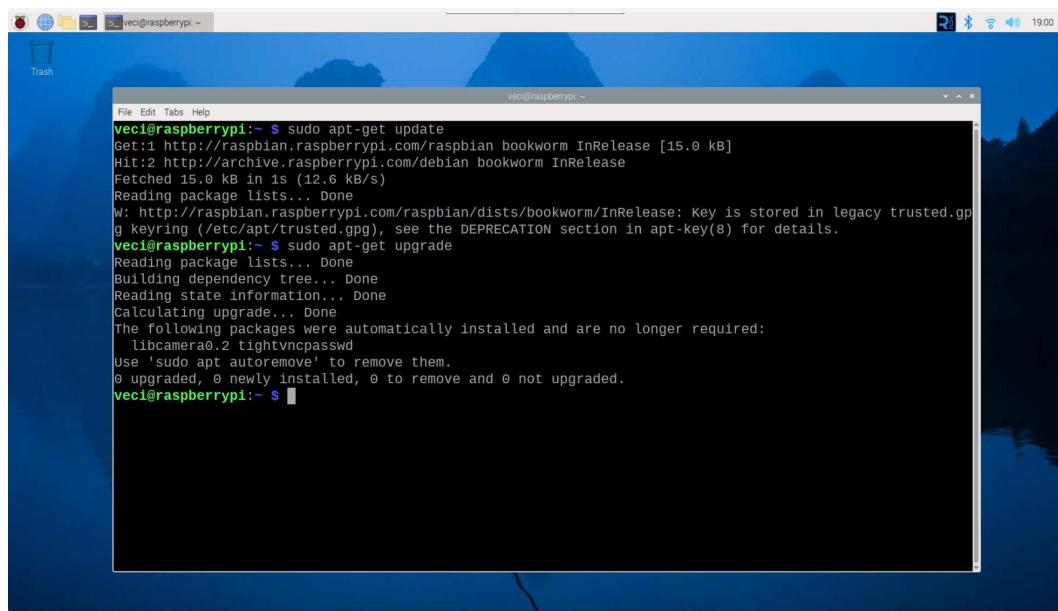
На самом почетку, након инсталације оперативног система на Raspberry Pi, препоручљиво је ажурирати и надоградити системске пакете како би се осигурала најновија верзија софтвера и побољшана сигурност уређаја.

### Ажурирање листе пакета

**sudo apt-get update** - Ова команда освежава листу доступних пакета и њихових верзија, али не инсталира или надограђује било који софтвер.

### Надоградња постојећих пакета

**sudo apt-get upgrade** - Ова команда инсталира најновије верзије свих тренутно инсталираних пакета.



```
File Edit Tabs Help veci@raspberrypi: ~
veci@raspberrypi: ~$ sudo apt-get update
Get:1 http://raspbian.raspberrypi.com/raspbian bookworm InRelease [15.0 kB]
Hit:2 http://archive.raspberrypi.com/debian bookworm InRelease
Fetched 15.0 kB in 1s (12.6 kB/s)
Reading package lists... Done
W: http://raspbian.raspberrypi.com/raspbian/dists/bookworm/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
veci@raspberrypi: ~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  libcamera0.2 tightvncpasswd
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
veci@raspberrypi: ~$
```

Слика 7. Приказ комадног прозора

## 1.4 Remote Access

Raspberry Pi 3 Model B+ омогућава извршавање кода директно на уређају или путем далинског приступа, што олакшава рад без директног повезивања периферних уређаја као што су тастатура, миш и монитор.

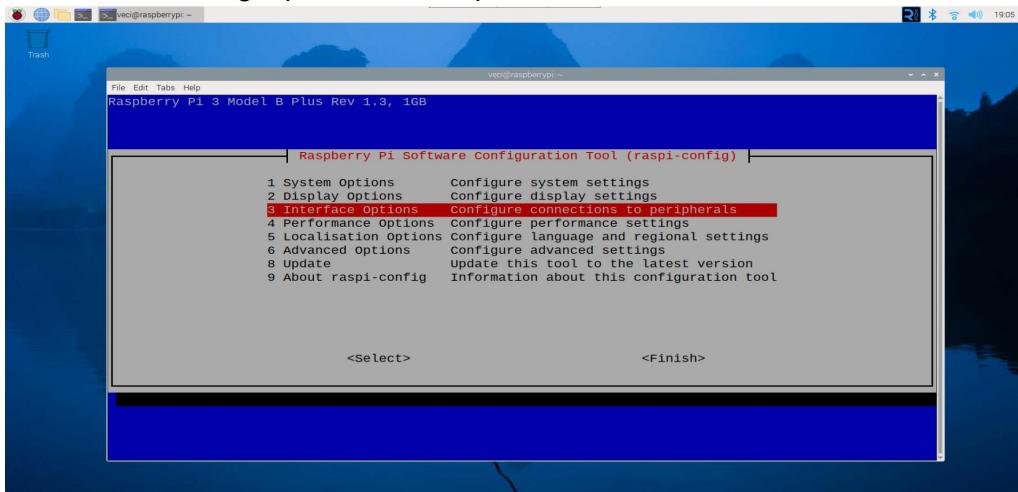
### 1.4.1 SSH (Secure Shell)

**Активирање SSH-а:** SSH је протокол који омогућава безбедан приступ Raspberry Pi-у преко мреже. Да би се омогућио SSH, потребно је:

- Повезати Raspberry Pi са интернетом.
- Отворите терминал на Raspberry Pi-у.
- Укџајте следећу команду да отворите Raspberry Pi конфигурацију:

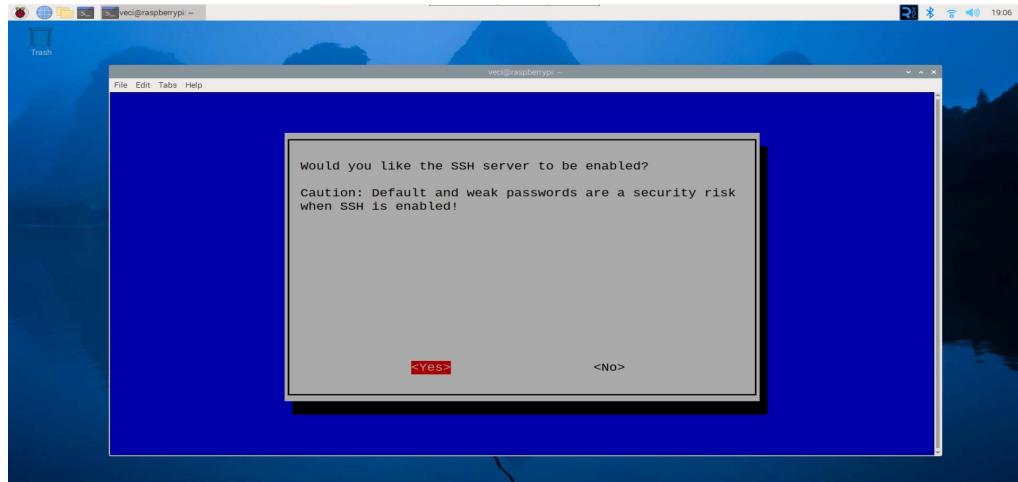
## **sudo raspi-config**

- Идите на Interfacing Options и изаберите SSH.



Слика 8. Одабир Interface опције

- Изаберите Enable да омогућите SSH.



Слика 9. Укључивање SSH опције

## **Приступ преко SSH-a:**

### **На Windows-y:**

- Отворите терминал (cmd).
- Укуцајте следећу команду, замењујући <IP\_адреса> са IP адресом вашег Raspberry Pi-a:  
**ssh pi@<IP\_адреса>**  
<IP\_адреса> - IP адресу добијате куцањем команде `hostname -l` у терминалу Raspberry Pi-a
- Унесите лозинку (raspberry ако је нисте променили).

### **1.4.2 RealVNC (Virtual Network Computing)**

RealVNC омогућава даљинско управљање Raspberry Pi-ом са графичким интерфејсом.

## Активирање VNC-а:

- Повежите Raspberry Pi на интернет (преко Wi-Fi или Ethernet кабла).
- **Омогућавање VNC-а:**
- Отворите терминал на Raspberry Pi-у.
- Укуцајте следећу команду да отворите Raspberry Pi конфигурацију:  
**sudo raspi-config**
- Идите на Interfacing Options и изаберите VNC.
- Изаберите Enable да омогућите VNC.

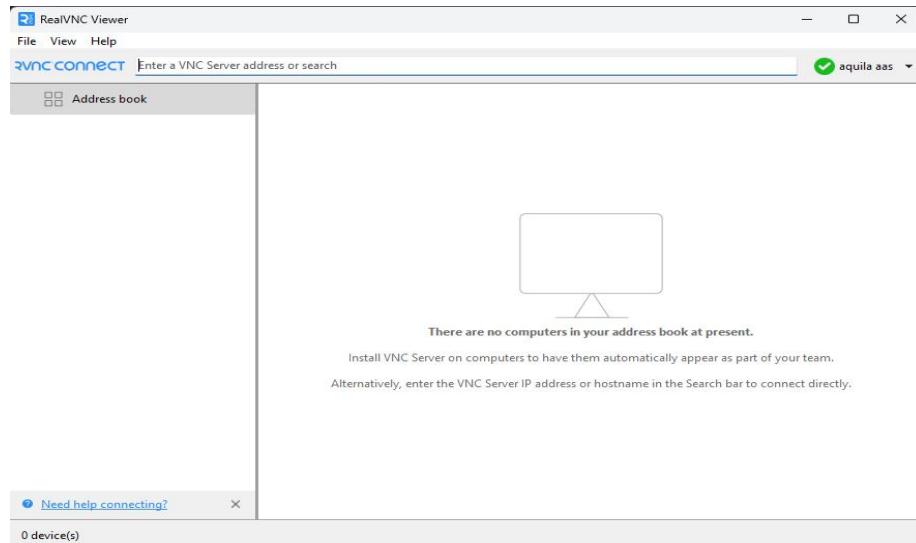
## Приступ преко VNC-а:

### Инсталирање VNC Viewer-а:

- Преузмите и инсталирајте VNC Viewer на ваш рачунар са RealVNC званичног сајта.

### Повезивање са Raspberry Pi-ом:

- Отворите VNC Viewer на вашем рачунару.
- У поље VNC Server унесите IP адресу вашег Raspberry Pi-а.
- Кликните на Connect.
- Унесите корисничко име (pi) и лозинку (raspberry ако је нисте променили).



Слика 10. Приказ VNC Viewer прозора

## 2. Повезивање сензора и Raspberry Pi 3 Model B+

### 2.1 Повезивање сензора који дају дигитални излаз са Raspberry Pi

Дигитални сензори су релативно једноставни за повезивање са Raspberry Pi, јер они дају бинарне вредности (0 или 1).

- **Повежите сензор:**

- Повежите VCC пин сензора са 3.3V или 5V пином Raspberry Pi.
- Повежите GND пин сензора са GND пином Raspberry Pi.
- Повежите излазни пин сензора (датапин) са једним од GPIO пинова Raspberry Pi.

- **Конфигуришите GPIO:**

- У Python скрипти, користите библиотеку RPi.GPIO за конфигурисање GPIO пина као улазног.
- Основне функције у програмском језику Python које се често користе за рад са GPIO пиновима на Raspberry Pi.

```
import RPi.GPIO as GPIO # Подешавање GPIO мода (можете користити BCM или BOARD нотацију за број пинова)
```

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(pin, GPIO.OUT) # Подешавање пина као излаза
```

```
GPIO.setup(pin, GPIO.IN) # Подешавање пина као улаза
```

```
GPIO.output(pin, GPIO.HIGH) # Постављање пина на HIGH (3.3V)
```

```
GPIO.output(pin, GPIO.LOW) # Постављање пина на LOW (0V)
```

```
value = GPIO.input(pin) # Читање стања пина (GPIO.HIGH или GPIO.LOW)
```

```
GPIO.cleanup() # Искључује GPIO мод и ослобађа све ресурсе
```

Ове функције пружају основне могућности за рад са GPIO пиновима на Raspberry Pi.

### 2.2 Повезивање сензора који дају аналогни излаз преко АД конвертора

Аналогни сензори дају континуирани излазни напон који мора бити конвертован у дигитални сигнал помоћу аналогно-дигиталног (АД) конвертора пре него што га Raspberry Pi може прочитати.

## **2.2.1 PCF8591 АД конвертор**

PCF8591 је 8-битни АД конвертор који има четири аналогна улазна канала и један аналогни излазни канал. Повезује се са Raspberry Pi преко I2C интерфејса.

На аналогни пин не сме да се доведе више од 3.3V

### **Резолуција:**

- PCF8591 има резолуцију од 8 бита, што значи да може претворити аналогне вредности у дигиталне у распону од 0 до 255.

### **Повезивање са Raspberry Pi преко I2C:**

- Повежите VCC пин PCF8591 са 5V пином Raspberry Pi.
- Повежите GND пин PCF8591 са GND пином Raspberry Pi.
- Повежите SDA пин PCF8591 са SDA пином Raspberry Pi (обично GPIO 2).
- Повежите SCL пин PCF8591 са SCL пином Raspberry Pi (обично GPIO 3).

Библиотеку за рад са PCF8591 АД/ДА нализимо на овом линку:

[https://github.com/sunfounder/SunFounder\\_SensorKit\\_for\\_RPi2/tree/master/Python](https://github.com/sunfounder/SunFounder_SensorKit_for_RPi2/tree/master/Python)

Инсталација библиотеке на Raspberry Pi плочици се врши командом

**wget**

[https://raw.githubusercontent.com/sunfounder/SunFounder\\_SensorKit\\_for\\_RPi2/master/Python/PCF8591.py](https://raw.githubusercontent.com/sunfounder/SunFounder_SensorKit_for_RPi2/master/Python/PCF8591.py)

Убацивање библиотеке у python пројекат се врши инструкцијом

**import PCF8591 as pcf**

## **2.2.2 Примери АД конвертора веће резолуције**

Постоје и други АД конвертори са већом резолуцијом који могу бити коришћени са Raspberry Pi.

### **MCP3008:**

- 10-битни АД конвертор са 8 аналогних улаза.
- Повезује се преко SPI интерфејса.

### **ADS1115:**

- 16-битни АД конвертор са 4 аналогна улаза.
- Повезује се преко I2C интерфејса.

## **ADS1015:**

- 12-битни АД конвертор са 4 аналогна улаза.
- Повезује се преко I2C интерфејса.

Сви ови конвертори нуде већу резолуцију од PCF8591 и могу бити коришћени за прецизније мерење аналогних сигнална.

## **2.3 Повезивање сензора који имају I2C излаз са Raspberry Pi-ем**

I2C (Inter-Integrated Circuit) је серијски комуникациони протокол који омогућава повезивање више уређаја користећи само две линије: **SDA** (Серијски Подаци) и **SCL** (Серијски Сат). Raspberry Pi има интегрисану подршку за I2C комуникацију, и можете користити ове пинове за повезивање I2C уређаја.

### **I2C Пинови на Raspberry Pi**

Raspberry Pi има два I2C интерфејса, али у већини случајева користићете I2C интерфејс број 1. Ево који пинови се користе за I2C на GPIO конектору:

- **SDA (Data Line)**: GPIO 2 (Pin 3)
- **SCL (Clock Line)**: GPIO 3 (Pin 5)

### **Број I2C уређаја који се могу повезати**

I2C протокол подржава вишеструке уређаје на истој I2C линији, али сваки уређај мора имати јединствену I2C адресу. На Raspberry Pi, можете теоретски повезати до 127 I2C уређаја (осим резервисаних адреса), али у пракси је тај број обично мањи због ограничења капацитета линије и комплексности система.

I2C уређаји користе 7-битне или 10-битне адресе за идентификацију на I2C магистрале. Најчешће се користе 7-битне адресе, што омогућава до 127 (0x00 до 0x7F) различитих адреса.

Неке I2C адресе су резервисане за специјалне намене и не могу се користити за обичне уређаје.

Адресе се често наводе у документацији производача уређаја.

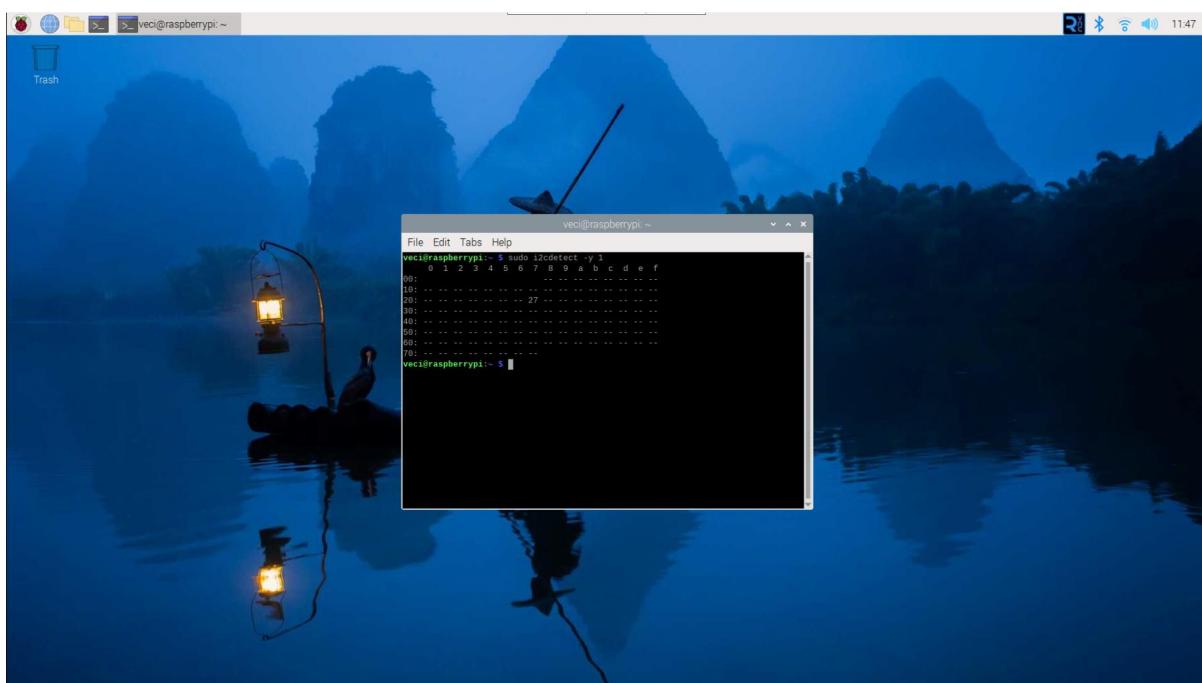
## Пronалажење Адресе I2C Уређаја

### I2C скенер:

- Можете користити i2cdetect алат на Raspberry Pi-у да скенирате магистралу и пронађете све повезане I2C уређаје. Ово је корисно када немате документацију или нисте сигурни у адресу уређаја.
- Покрените следећу команду да скенирате I2C магистралу 1 (најчешће коришћену магистралу на Raspberry Pi-у):

```
sudo i2cdetect -y 1
```

Ово ће приказати мапу I2C адреса и означити све адресе на којима је пронађен уређај.



Слика 11. Приказ командног простора

На слици можемо да видимо како изгледа приказ командног простора након извршења инструкције и адресу модула прикљученог на I2C пинове Raspberry Pi плочице (У овом случају приказана адреса представља адресу LCD1602 модула.)

## 2.4 Библиотека Time

Time је уграђена Python библиотека која омогућава рад са временом и каснијим функционалностима.

```
import time # импорт библиотеке
```

```
time.sleep(seconds) # Прекид извршавања програма на одређено време у секундама
```

```
start_time = time.time()

# Део кода чије се време извршавања мери

end_time = time.time()

execution_time = end_time - start_time
```

## 2.5 Разлика између BCM и BOARD

### 2.5.1 BCM Нумерација

BCM (Broadcom) нумерација пинова користи бројеве који одговарају Broadcom чипу на Raspberry Pi. Ово значи да пинови се идентификују користећи бројеве који одговарају пиновима на Broadcom чипу, а не физичким бројевима на GPIO header-у (мрежи пинова) на самом Raspberry Pi моделу.

Пример:

```
import RPi.GPIO as GPIO
setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT) # Постављање GPIO пина 17 као излаз
GPIO.output(17, GPIO.HIGH) # Постављање напона на пину 17 на високо
```

### 2.5.2 BOARD Нумерација

BOARD нумерација користи физичке бројеве пинова на GPIO header-у на Raspberry Pi моделу. Ово значи да сваки пин на header-у има своју унапред дефинисану нумерацију, која може варирати између различитих модела Raspberry Pi.

Пример:

```
import RPi.GPIO as GPIO
setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT) # Постављање GPIO пина 11 као излаз
GPIO.output(11, GPIO.LOW) # Постављање напона на пину 11 на ниско
```

У првом случају код BCM-а видимо да се пин иницијализује бројем 17 а у другом случају код BOARD-а видимо да се исти пин иницијализује са 11.

## **2.6 Карактеристика GPIO пинова**

GPIO (General Purpose Input/Output) пинови на Raspberry Pi 3 Model B+ имају следеће карактеристике:

### **Напон и струја**

- **Излазни напон:** 3.3V
- **Максимална струја по пину:** 2 mA до 16 mA (подразумевана 8 mA)
- **Укупна максимална струја за све пинове:** 50 mA

### **Напон и струја за улаз**

- **Максимални дозвољени улазни напон:** 3.3V
- **Максимални дозвољена улазна струја:** 0,5 mA
- **Прекорачење напона:** Важно је не прекорачити 3.3V јер у противном можете оштетити Raspberry Pi.
- GPIO пинови су веома осетљиви на напоне веће од 3.3V, тако да увек треба водити рачуна да не прекорачите ову вредност.

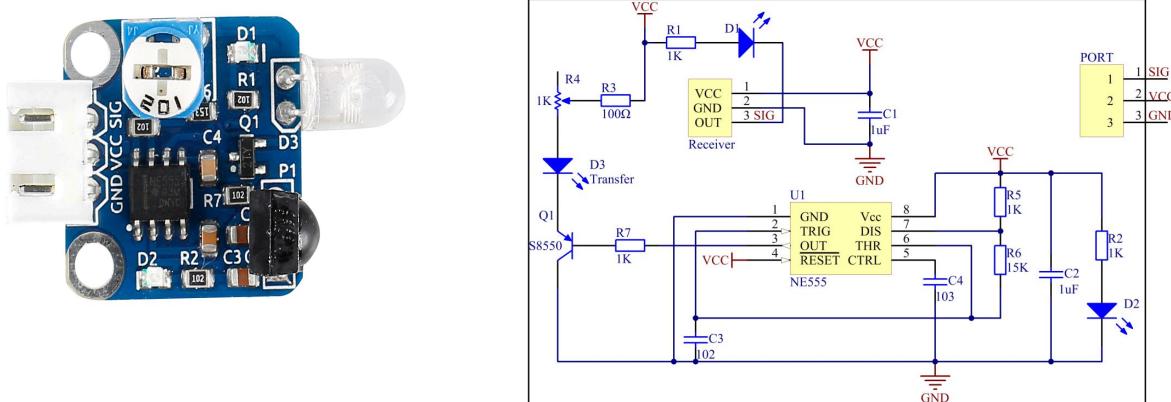
### 3. Примери аквизиције података са сензора помоћу Raspberry Pi 3 Model B+

#### 3.1 Детекција објекта

Циљ система детекција објекта јесте да уз помоћ IR\_Obstacle сензора детектује присуство објекта, да на основу тога испише одговарајућу поруку и упали Auto Flash LED модул.

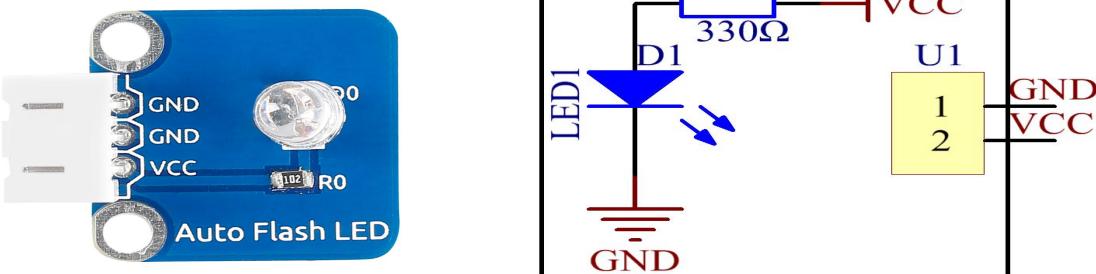
Потребне компоненте:

IR\_Obstacle - користи инфрацрвене (IR) зраке да би открио објекте у свом видокругу. Сензор емитује IR зраке, који се одбијају од објекта и враћају назад до пријемника. На основу времена потребног да се зраци врате, сензор одређује растојање до препреке и шаље сигнал микроконтролеру или другом уређају.



Слика 12. IR\_Obstacle (физички изглед и електрично коло)

Auto Flash LED - уређај који аутоматски контролише трептање LED диоде.

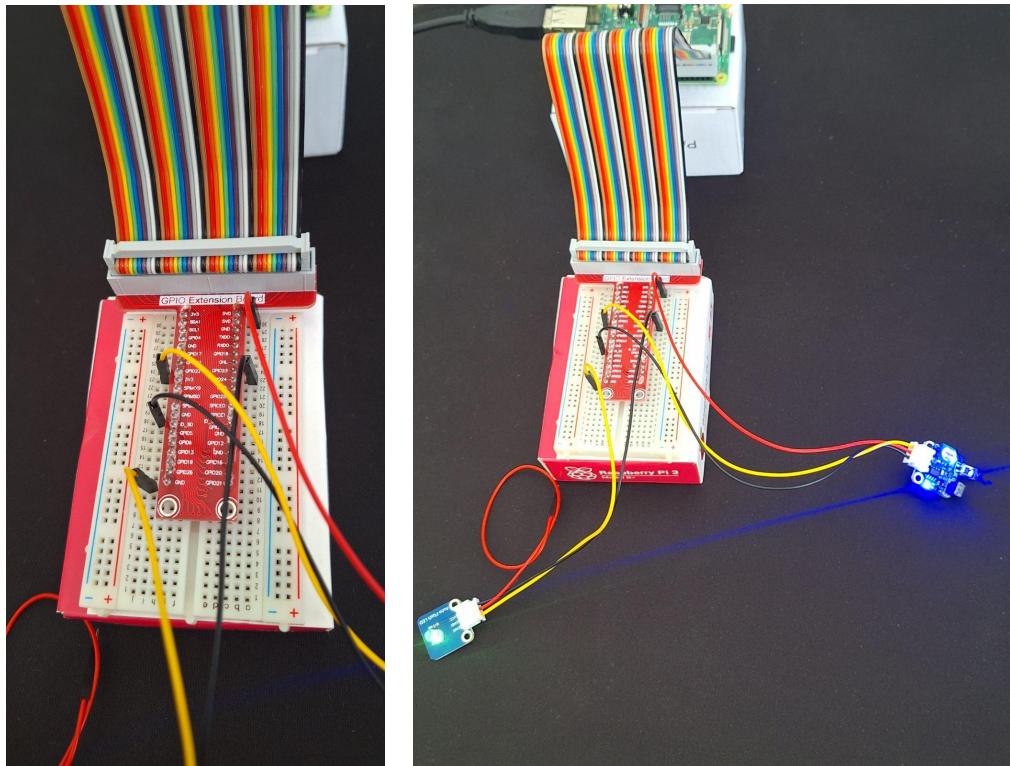


Слика 13. Auto Flash LED (физички изглед и електрично коло)

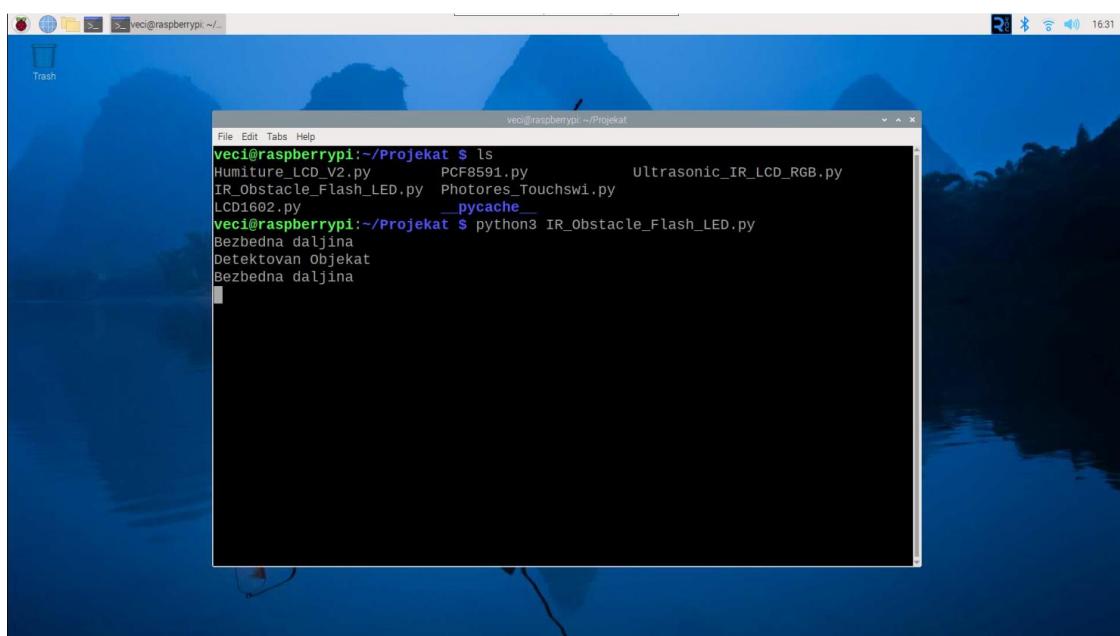
## Повезивање сензора са Raspberry Pi-м

IR_Obstacle	Raspberry Pi
SIG	GPIO 22
VCC	5V
GND	GND

Auto Flash LED	Raspberry Pi
GND	GND
GND	\
VCC	GPIO23



Слика 14. Приказ повезивања и система



Слика 15. Приказ командног прозора

## Код у Raspberry Pi-y

import RPi.GPIO as gpio - Увози библиотеку RPi.GPIO и даје јој алијас gpio  
import time - модул за рад са временом (кашњења у коду).

IRpin = 22 - Дефинише пин број 22 као пин на који је повезан ИР сензор.

LEDpin = 23 - Дефинише пин број 23 као пин на који је повезана ЛЕД диода.

def setup(): - Дефинише функцију setup која конфигурише пинове.

    gpio.setmode(gpio.BCM) - Поставља начин нумерације пинова на BCM (Броадцом)

    gpio.setup(IRpin, gpio.IN) - Поставља IRpin као улазни пин (за читање података са ИР сензора).

    gpio.setup(LEDpin, gpio.OUT) - Поставља LEDpin као излазни пин (за контролу ЛЕД диоде).

def loop(): - Дефинише главну петљу програма. Логика је следећа да се испис у командном прозору врши само уколико дође до промене стања (вредност коју даје сензор).

    state = gpio.input(IRpin) - Поставља почетно стање state на вредност очитану са ИР сензора.

    while True: - Покреће бесконачну петљу.

        ir\_value = gpio.input(IRpin) - Очијатва тренутну вредност са ИР сензора.

        if ir\_value == 0 and state == 0: - Ако је детектован објекат (вредност је 0) и претходно стање је било без објекта (state је 0)

            gpio.output(LEDpin, gpio.HIGH)

            print('Detektovan Objekat')

            state = 1

        elif ir\_value == 1 and state == 1: - Ако је објекат уклоњен (вредност је 1) и претходно стање је било детекција објекта (state је 1):

            gpio.output(LEDpin, gpio.LOW)

            print('Bezbedna daljina')

```
state = 0  
time.sleep(1)
```

def destroy(): - Дефинише функцију destroy која чисти (ресетује) GPIO поставке.

gpio.cleanup() - Ресетује све GPIO пинове који су коришћени.

if \_\_name\_\_ == '\_\_main\_\_': - Проверава да ли је скрипта покренута директно.

try: - Покушава да изврши следеће блокове кода.

```
setup()  
loop()
```

except KeyboardInterrupt: - Ако се деси прекид (Ctrl+C), извршава следећи блок кода.

destroy() - Позива функцију destroy за чишћење GPIO поставки.

У Python-у, if \_\_name\_\_ == '\_\_main\_\_': је конструкција која се користи да би се одвојио извршни код који треба да се изврши само када је скрипта покренута директно, а не када се увози као модул у неки други скрипт.

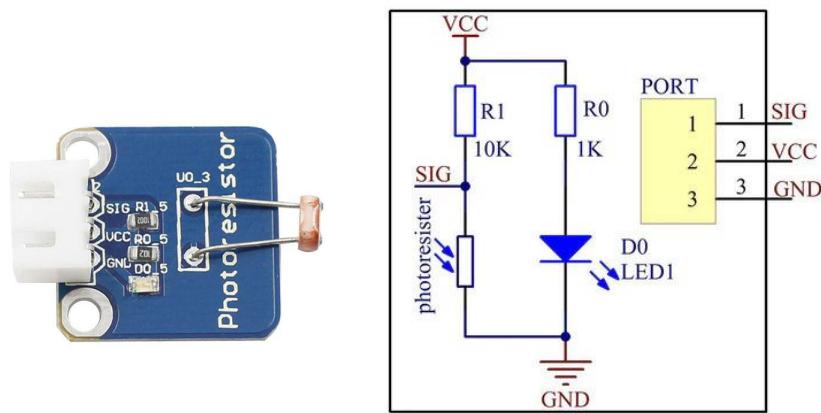
- Када Python извршава скрипту, променљива \_\_name\_\_ ће бити постављена на '\_\_main\_\_'. Ово је уgraђена променљива која садржи име текућег модула или, у случају извршавања као главни програм, име '\_\_main\_\_'.
- Конструкција if \_\_name\_\_ == '\_\_main\_\_': проверава да ли је име тренутног модула '\_\_main\_\_', што значи да се скрипта покреће директно.
- Ако је услов испуњен (\_\_name\_\_ је '\_\_main\_\_'), онда Python извршава блок кода који следи. Овај блок обично садржи почетне позиве функција или главну логику програма која треба да се изврши када се покрене програм.

### 3.2 Детекција Светлости

Циљ овог система јесте детекција светлости упомоћу photoотпорног сензора. Photoотпорни сензор се пали притиском на touch switch, а вредност коју photoотпорни сензор детектује се исписује у командном прозору.

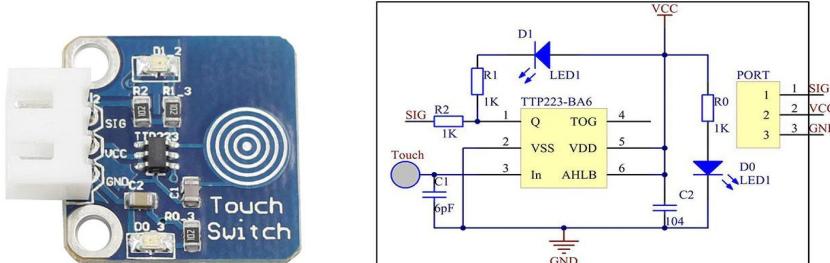
Потребне компоненте:

Photoresistor - пасивни електронски елемент који мења свој отпор у зависности од количине светлости која пада на њега. Његова основна примена је у детекцији светлости и контролним системима који реагују на промене у осветљењу.



Слика 16. Photoresistor (физички изглед и електрично коло)

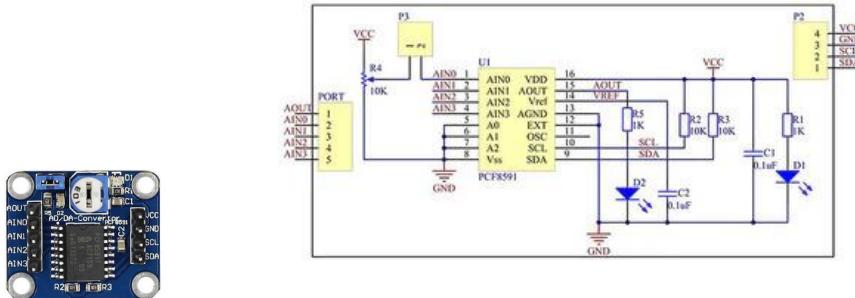
Touch Switch - електронски уређај који омогућава укључивање и искључивање кола једноставним додиром.



Слика 17. Touch Switch (физички изглед и електрично коло)

Пошто Raspberry Pi не располаже аналогним пиновима, за обраду аналогних сигнала користимо PCF8591 АД/ДА конвертор.

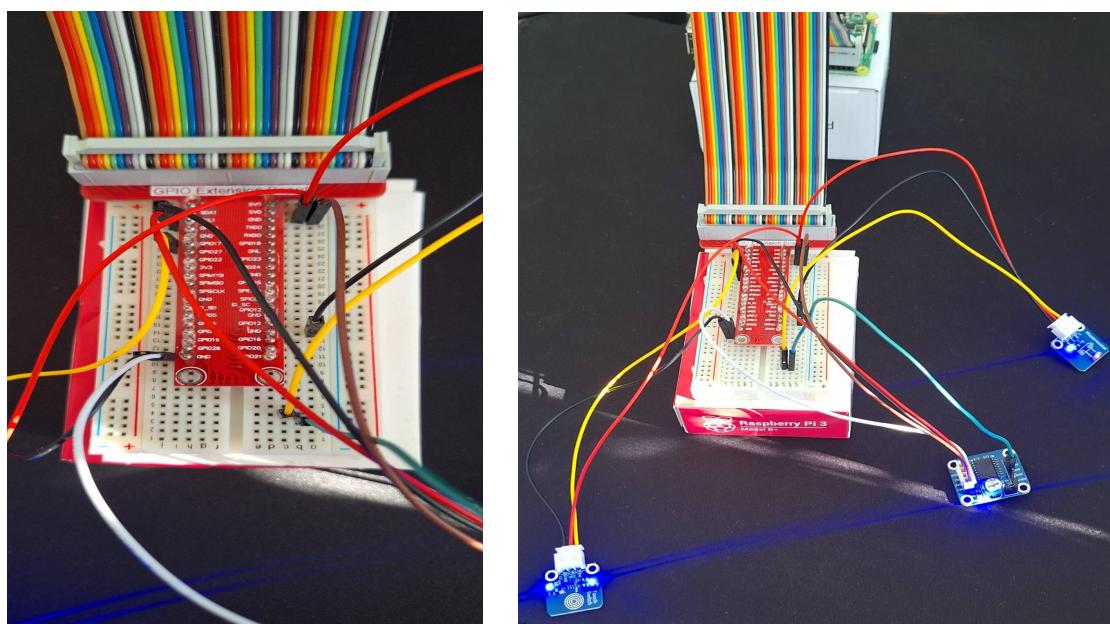
PCF8591 - интегрисани чип који омогућава аналогно-дигиталну (AD) и дигитално-аналогну (DA) конверзију сигнала.



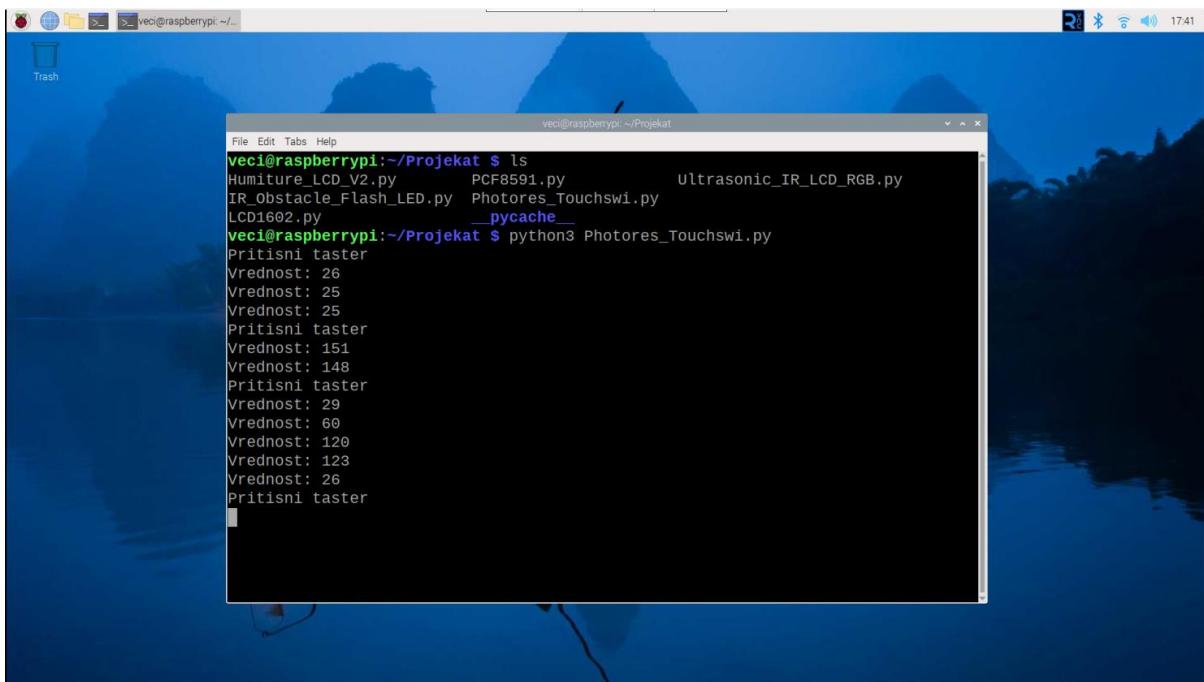
Слике 18. PCF8591 (физички изглед и електрично коло)

### Повезивање сензора са Raspberry Pi-м

Touch Switch	Raspberry Pi	Photoresistor	Raspberry Pi	PCF8591
SIG	GPIO17	SIG	/	AIN1
VCC	5V	VCC	5V	/
GND	GND	GND	GND	/
PCF8591	Raspberry Pi			
VCC	5V			
GND	GND			
SCL	SCL1			
SDA	SDA1			



Слика 19. Пrikaz повезивања и система



Слика 20. Приказ командног прозора

## Код у Raspberry Pi-у

```
import RPi.GPIO as gpio
```

import PCF8591 as pcf - Увози библиотеку за рад са АЦПЦФ8591 (Аналогно-дигитални конвертер) и даје јој алијас pcf.

```
import time
```

TCpin = 17 - Дефинише пин број 17 као пин на који је повезан тастер.

```
def setup():
```

```
    gpio.setmode(gpio.BCM)
```

gpio.setup(TCpin, gpio.IN) - Поставља TCpin као улазни пин (за читање стања тастера).

pcf.setup(0x48) - Иницијализује ПЦФ8591 са I2C адресом 0x48.

```
def loop():
```

set\_pin = -1 - Иницијализује променљиву set\_pin на -1, која прати стање тастера.

```
    while True:
```

```

dig_pin = gpio.input(TCpin) - Очитава тренутно стање тастера.

if dig_pin == 1:

    value = pcf.read(1) - Чита вредност са АЦПЦФ8591 са
    аналогног канала 1.

    print('Vrednost:', value)

    set_pin = 1

    time.sleep(1.2)

elif set_pin != 0 and dig_pin == 0:

    print('Pritisni taster')

    set_pin = 0

def cleanup():

    gpio.cleanup()

if __name__ == "__main__":
    try:
        setup()
        loop()
    except KeyboardInterrupt:
        cleanup()

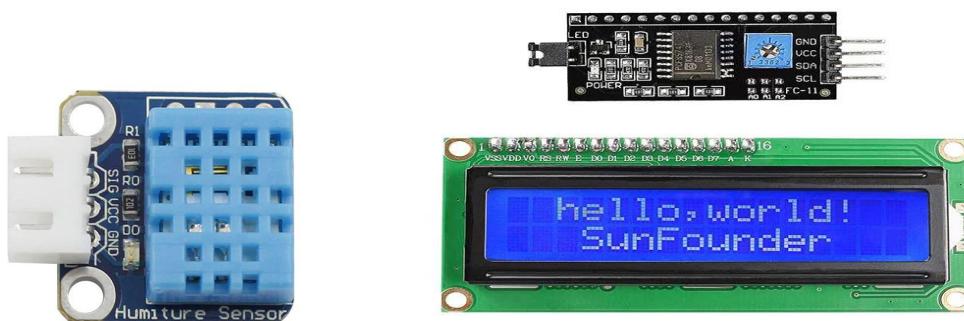
```

### 3.3 Мерење Температуре и Влажности

Циљ система јесте мерење температуре и влажности помоћу Humiture сензора, добијене вредности приказујемо на LCD екрану и у командном прозору.

Потребне компоненте:

Humiture - уређај који комбинује функције сензора температуре и влажности. Овај сензор је корисан за мерење и надгледање околне температуре и влажности у различитим апликацијама.



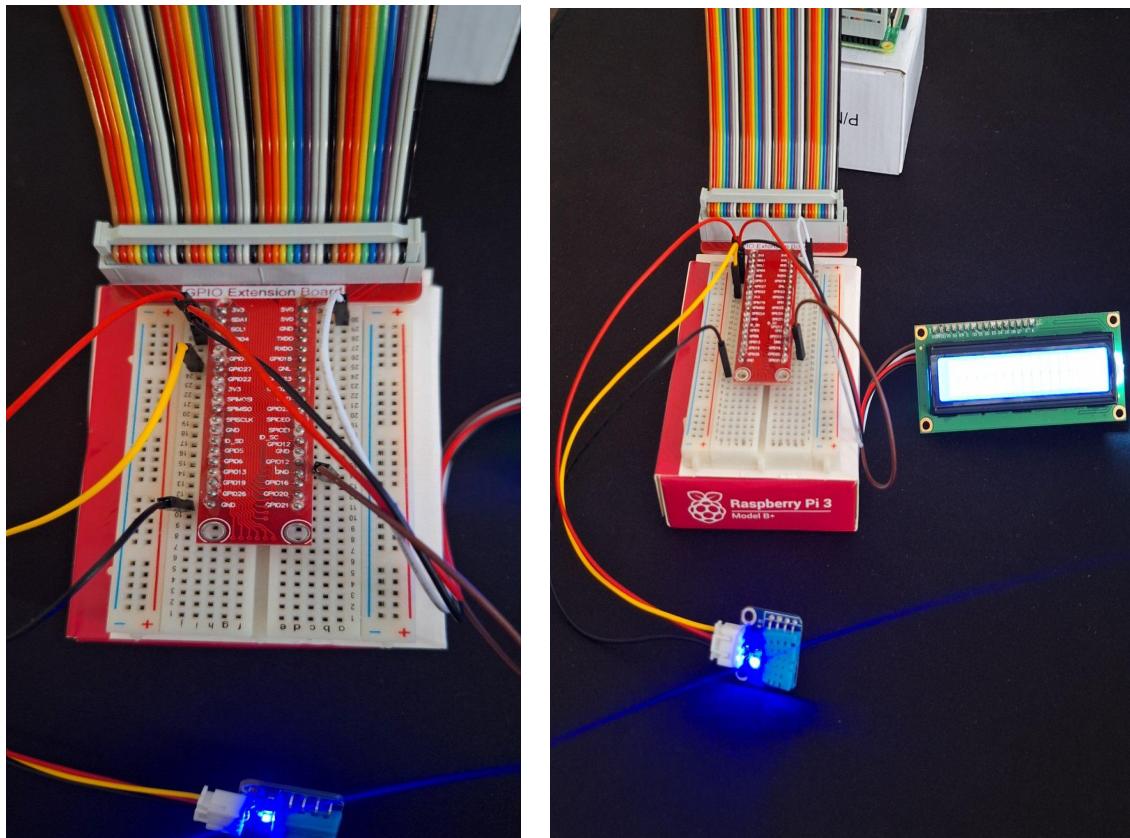
Слика 21. Физички изглед Humiture сензора и LCD1602 екрана

I2C LCD1602 – LCD дисплеј који се контролише преко I2C интерфејса, што олакшава његову интеграцију са микроконтролерским системима. Ово је корисно јер смањује број пинова који су потребни за управљање дисплејем, што је веома важно на микроконтролерима са ограниченим бројем доступних GPIO пинова.

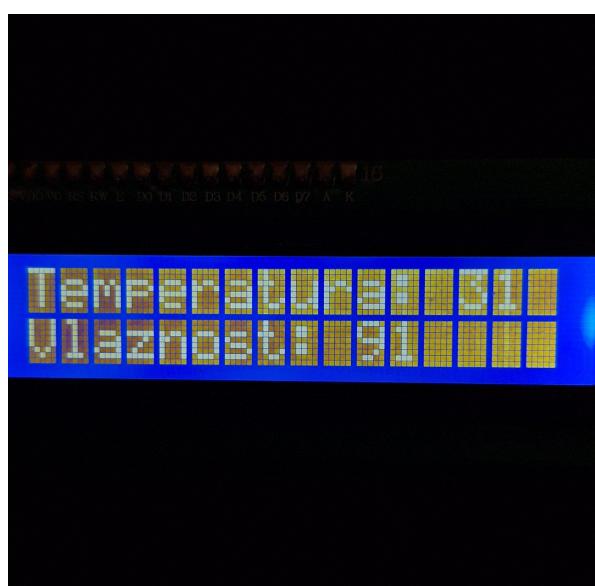
## Повезивање сензора са Raspberry Pi-м

LCD1602	Raspberry Pi
GND	GND
VCC	5V
SDA	SDA1
SCL	SCL1

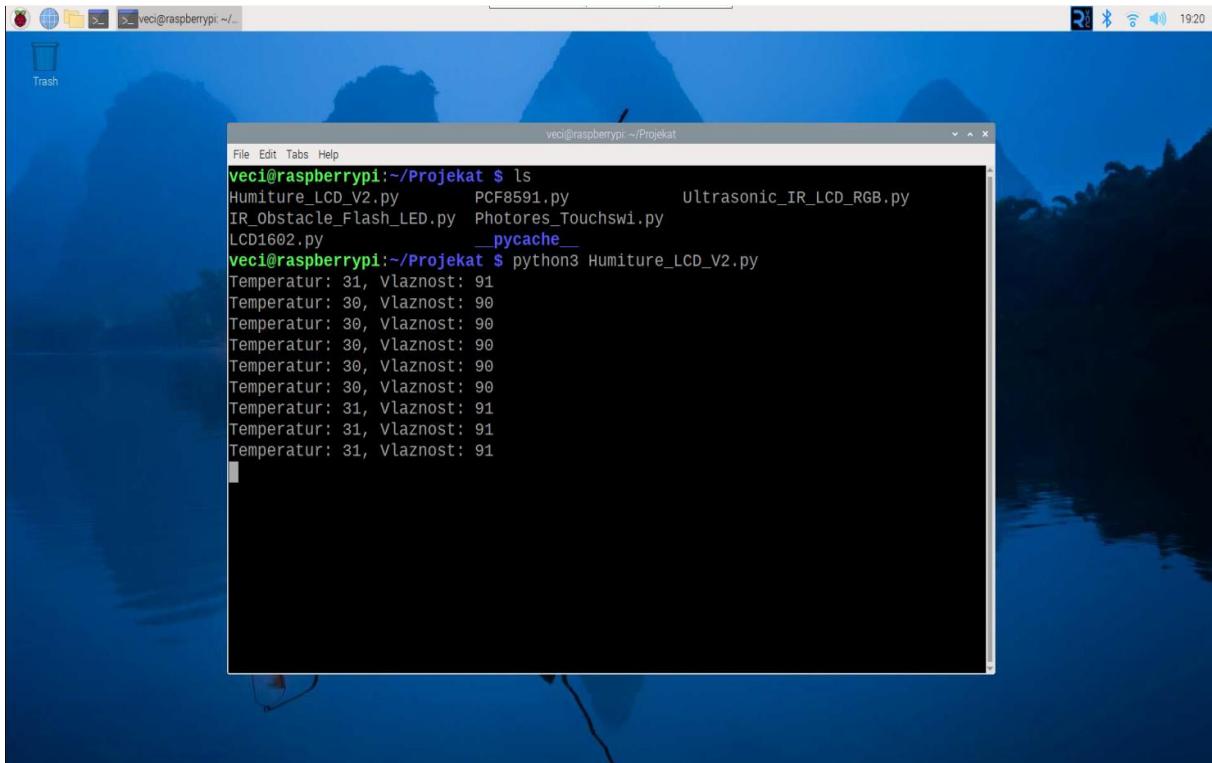
Humiture	Raspberry Pi
SIG	GPIO17
VCC	3V3
GND	GND



Слика 22. Приказ повезивања и система



Слика 23. Приказ LCD екрана



Слика 24. Приказ командног прозора

### Код у Raspberry Pi-у

```
import RPi.GPIO as gpio
```

```
import time
```

import LCD1602 as lcd - Увозимо библиотеку LCD1602 и дајемо јој алијас lcd. Ова библиотека се користи за управљање LCD екраном.

HTpin = 17 - Дефинишемо променљиву HTpin која представља GPIO пин број 17 на који је повезан сензор за температуру и влажност.

```
gpio.setmode(gpio.BCM)
```

max\_count = 100 - Дефинишемо максималан број поновљених вредности за сигнал, користи се за излазак из петље читања података.

Дефинишемо константе које представљају различита стања током читања сигнала са сензора.

```
stats_sc_low = 1
```

```
stats_sc_high = 2
```

```
stats_dc_first_low = 3
```

```
stats_dc_high = 4
```

```
stats_dc_low = 5
```

```
def read_ht():
```

    gpio.setup(HTpin, gpio.OUT) - Постављамо HTpin као излазни пин.

    gpio.output(HTpin, gpio.HIGH) - Постављамо HTpin на високу вредност (активирамо сензор).

    time.sleep(0.05) - Постављамо HTpin на ниску вредност (припремамо за читање).

```
    gpio.output(HTpin, gpio.LOW)
```

```
    time.sleep(0.02)
```

    gpio.setup(HTpin, gpio.IN, gpio.PUD\_UP) - Постављамо HTpin као улазни пин са pull-up отпорником.

    same\_count = 0 - Иницијализујемо променљиву same\_count на 0, која прати колико пута је узастопно очитана иста вредност.

    last = -1 - иницијализујемо променљиву last на -1, која памти последњу очитану вредност.

    data = [] - Иницијализујемо празну листу data која ће чувати све очитане вредности.

    while True: - У овој бесконачној петљи уписујемо податке које сензор шаље Raspberry Pi плочици упис се врши све док сензор не пошаље исту вредност 100 пута.

```
        curr_data = gpio.input(HTpin) - Читамо тренутну вредност са HTpin.
```

```
        data.append(curr_data)
```

```
        if last != curr_data:
```

```
            same_count = 0
```

```
            last = curr_data
```

```
        else:
```

```
            same_count += 1
```

```
            if same_count > max_count:
```

                break - наредба која служи да прекине извршење петње

```
stats = stats_sc_low
```

lengths = [] - Иницијализујемо празну листу lengths која ће чувати дужине временских интервала.

curr\_lengths = 0 - Иницијализујемо променљиву curr\_lengths на 0, која прати дужину текућег интервала.

for curr in data: - Започињемо петљу која обрађује сваки елемент у листи data.  
У овој for петљи борјимо колико пута се јединица понавља у низу (нпр. data(0,1,1,1,0,0,1,1,1,1,1,0) у овом случају на излаз добијамо низ са два елемента lengths (3,5)) низови су одвојени нулама.

```
curr_lengths += 1
```

Унутар петље проверавамо тренутно стање (stats) и вредност (curr)

```
if stats == stats_sc_low:
```

```
    if curr == gpio.LOW:
```

```
        stats = stats_sc_high
```

```
    else:
```

continue - наредба којом се прескаче текућа итерација петље и наставља са следећом итерацијом.

```
    if stats == stats_sc_high:
```

```
        if curr == gpio.HIGH:
```

```
            stats = stats_dc_first_low
```

```
    else:
```

```
        continue
```

```
if stats == stats_dc_first_low:
```

```
    if curr == gpio.LOW:
```

```
        stats = stats_dc_high
```

```
    else:
```

```
        continue
```

```
if stats == stats_dc_high:
```

```
    if curr == gpio.HIGH:
```

```
        stats = stats_dc_low
```

```

curr_lengths = 0

else:
    continue

if stats == stats_dc_low:
    if curr == gpio.LOW:
        stats = stats_dc_high
        lengths.append(curr_lengths)

else:
    continue

```

Проверавамо да ли је број елемената у низу lengths једнак 40. Ако није, враћамо False, што указује на грешку у читању података.

```

if len(lengths) != 40:
    #print("ERROR: Podaci nisu dobri")
    return False

```

```

bits = []
the_bytes = []
byte = 0
short_high = min(lengths)
long_high = max(lengths)

```

half = (long\_high + short\_high) / 2 - Израчунавамо просек најкраћег и најдужег интервала, што ће се користити за разликовање 0 и 1 у битовима.

for length in lengths: - У овој for петљи претходни низ елемената претворамо у битове тако да ако је елемет већи од half онда је 1 ако је мањи онда је 0.

```

bit = 0
if length > half:- Ако је дужина интервала већа од половине, постављамо bit на 1.

```

```
bit = 1
```

```
else:  
    bit = 0 - Ако није, bit остаје 0.  
  
    bits.append(bit)  
  
    for i in range(0, len(bits)):
```

- У овој for петљи предходно добијени низ битова претварамо у низ који ће имати 5 елемента где ће сваки елемет да представља једа бајт( 8-бита).

```
        byte = byte << 1 - Померамо битове улево за једну позицију.
```

```
        if bits[i]:
```

```
            byte = byte | 1 - Ако је бит 1, постављамо најнижи бит бајта на 1.
```

```
        else:
```

```
            byte = byte | 0 - Ако није, најнижи бит остаје 0.
```

```
        if ((i+1)%8 == 0):
```

```
            the_bytes.append(byte)
```

```
            byte = 0
```

```
chsuma = (the_bytes[0] + the_bytes[1] + the_bytes[2] + the_bytes[3]) & 0xFF -  
Израчунавамо контролни збир прва четири бајта
```

```
if the_bytes[4] != chsuma: - испитујемо да ли добијени збир одговара  
chsumi.
```

```
# print ("ERROR: Provera nije prosla")
```

```
return False
```

```
return the_bytes[2], the_bytes[0]
```

```
def lcd_show(temp, humi): - Дефинишемо функцију lcd_show() која приказује  
температуру и влажност на LCD екрану.
```

```
lcd.init(0x27, 1) - Иницијализујемо LCD екран са I2C адресом 0x27 и  
подешавањем фонтова.
```

```
lcd.write(0, 0, f'Temperatura: {temp}')
```

```
lcd.write(0, 1, f'Vlaznost: {humi}')
```

```
time.sleep(2)
```

`def main():` - Дефинишемо главну функцију `main()` која непрекидно чита податке са сензора и приказује их.

```
while True:
```

```
    res = read_ht()
```

```
    if res:
```

```
        temp, humi = res
```

```
        print(f"Temperatur: {temp}, Vlaznost: {humi}")
```

```
        lcd_show(temp, humi)
```

```
        time.sleep(1)
```

`def destroy():`

```
    gpio.cleanup()
```

```
if __name__ == '__main__':
```

```
    try:
```

```
        main()
```

```
    except KeyboardInterrupt:
```

```
        destroy()
```

### 3.4 Паркинг Систем

Циљ система јесте измерити тачну удаљеност до препреке и исписивати измерену вредност на LCD екрану, уколико удаљеност пређе критични праг исписти поруку упозорења и упалити светлећу диоду.

Потребне компоненте:

HC-SR04 - ултразвучни модул за мерење удаљености који се користи у различитим пројектима, укључујући роботика и аутоматизација. Он функционише на принципу ултразвучног таласа и његовог рефлексије од објекта.



Слика 25. Физички изглед HC-SR04 сензора

I2C LCD1602

IR\_Obstacle

Auto Flash LED

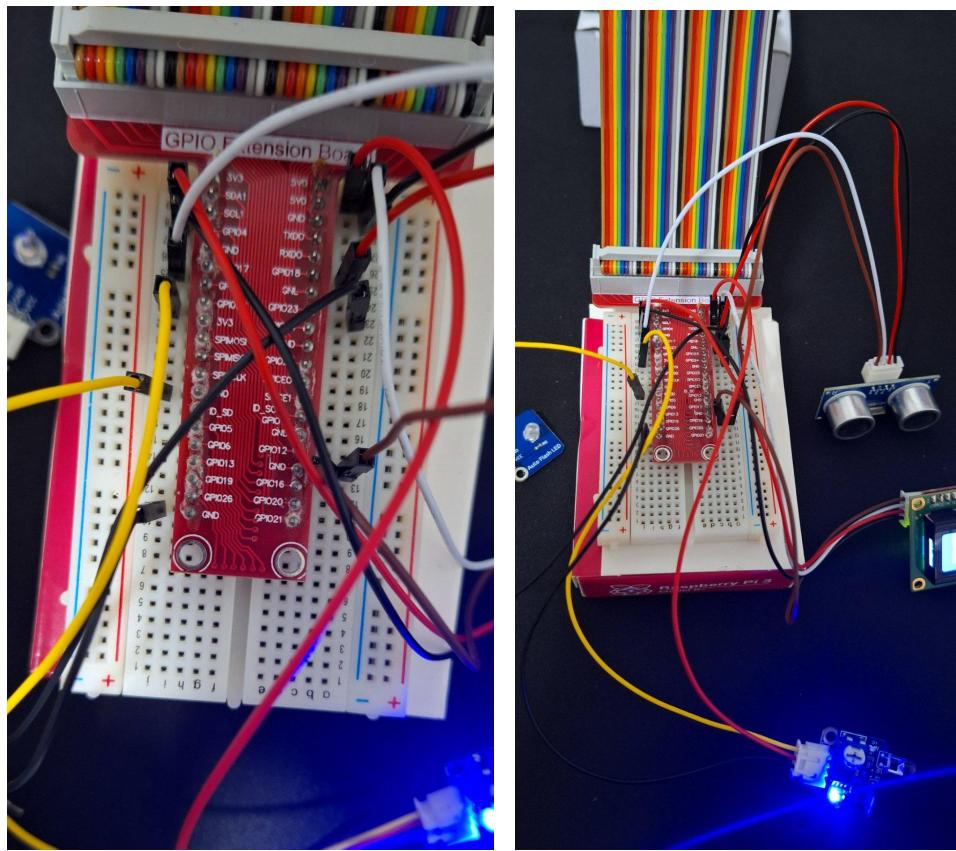
### Повезивање сензора са Raspberry Pi-м

Ultrasonic	Raspberry Pi
GND	GND
ECHO	GPIO17
TRIG	GPIO18
VCC	5V

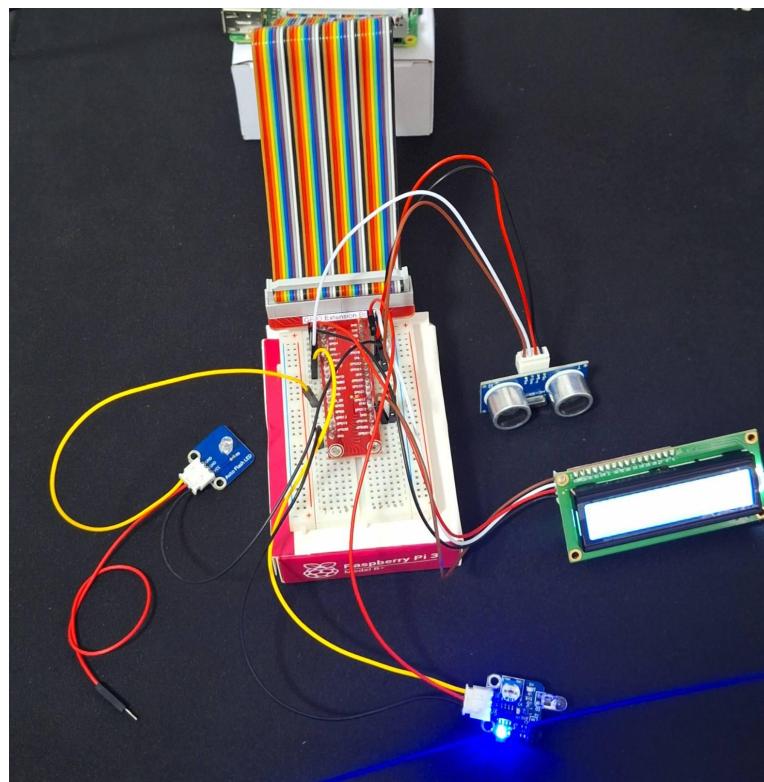
LCD1602	Raspberry Pi
GND	GND
VCC	5V
SDA	SDA1
SCL	SCL1

IR_Obstacle	Raspberry Pi
SIG	GPIO22
VCC	5V
GND	GND

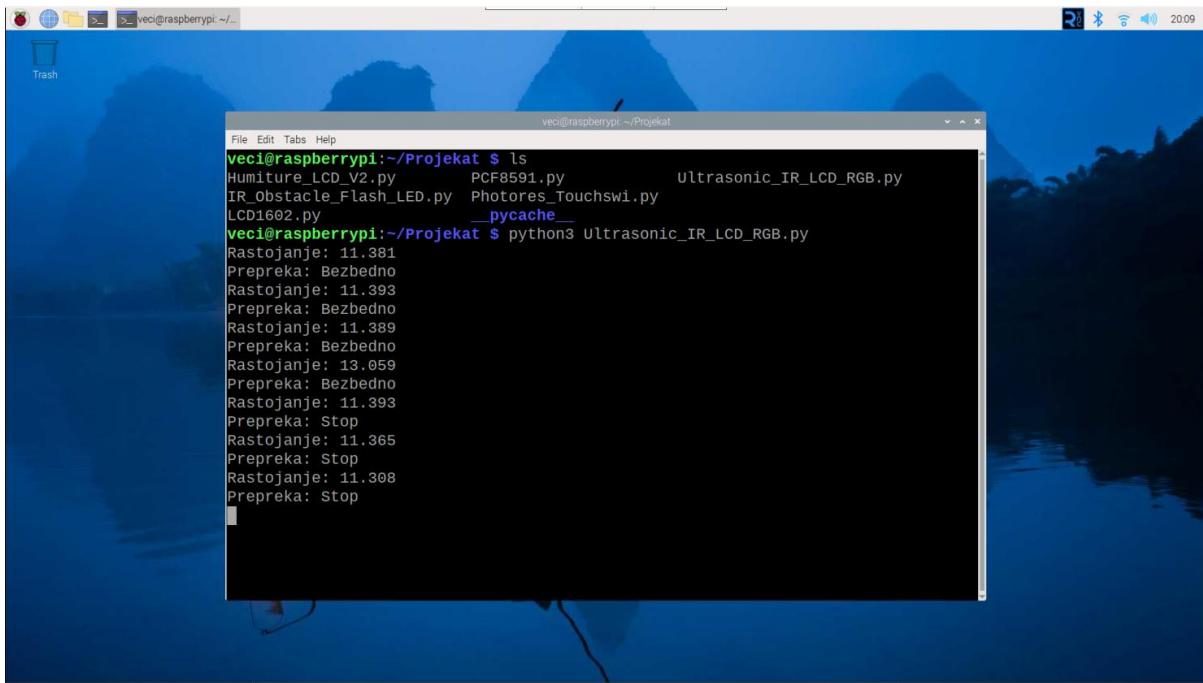
Auto Flash LED	Raspberry Pi
GND	GND
GND	/
VCC	5V



Слика 26. Приказ повезивања и система



Слика 27. Приказ система



Слика 28. Приказ командног прозора

### Код у Raspberry Pi-у

```
import RPi.GPIO as gpio  
  
import time  
  
import LCD1602 as lcd
```

Дефинишемо пинове на које су повезани сензори и излази:

IRpin = 22 - Пин за инфрацрвени сензор.

LCDpin = 23 - Пин за контролу LCD екрана.

ECHO = 17 - Пин за примање ултразвучног сигнала.

TRIG = 18 - Пин за слање ултразвучног сигнала.

def setup(): - Дефинишемо функцију setup() која ће иницијализовати GPIO пинове.

```
gpio.setmode(gpio.BCM)  
  
gpio.setup(IRpin, gpio.IN)  
  
gpio.setup(LCDpin, gpio.OUT)  
  
gpio.setup(ECHO, gpio.IN)  
  
gpio.setup(TRIG, gpio.OUT)
```

```
def distance():
```

    gpio.output(TRIG, gpio.LOW) - Постављамо TRIG пин на ниско (LOW) да бисмо ресетовали сензор.

```
    Time.sleep(0.000002)
```

    gpio.output(TRIG, gpio.HIGH) - Постављамо TRIG пин на високо (HIGH) да бисмо послали ултразвучни импулс.

```
    time.sleep(0.00001)
```

    gpio.output(TRIG, gpio.LOW) - Постављамо TRIG пин поново на ниско (LOW).

    while gpio.input(ECHO) == 0: - Чекамо да ECHO пин постане високо (HIGH).

```
        pass
```

        time1 = time.time() - Бележимо време почетка мерења. Тренутак када је ECHO сигнал посто висок.

        while gpio.input(ECHO) == 1: - Чекамо да ECHO пин постане ниско (LOW).

```
        pass
```

        time2 = time.time() - Бележимо време краја мерења. Тренутак када је ECHO сигнал постао низак (када је нестао)

    value = time2 - time1 - Израчунавамо време проласка ултразвучног сигнала.

    return value \* 340 / 2 \* 100 - Израчунавамо растојање (време проласка ултразвучног сигнала помножено са брзином звука, подељено са 2, и претворено у цетиметре).

```
def lcd_show(distance, detected):
```

    lcd.init(0x27, 1) - Иницијализујемо LCD екран са I2C адресом 0x27.

```
    lcd.write(0, 0, f"Rastojanje:{distance:.1f}")
```

```
    lcd.write(0, 1, str(detected))
```

```
    time.sleep(0.5)
```

```

def detected():
    ir_value = gpio.input(IRpin) - Читамо вредност са инфрацрвеног сензора.
    if ir_value == 1:
        gpio.output(LCDpin, gpio.LOW)
        return 'Bezbedno'
    else:
        gpio.output(LCDpin, gpio.HIGH)
        return 'Stop'

def main():
    while True:
        dis = distance()
        detec = detected()
        lcd_show(dis,detec)
        print(f'Rastojanje: {dis:.3f}')
        print(f'Prepreka: {detec:}')

def destroy():
    gpio.cleanup()

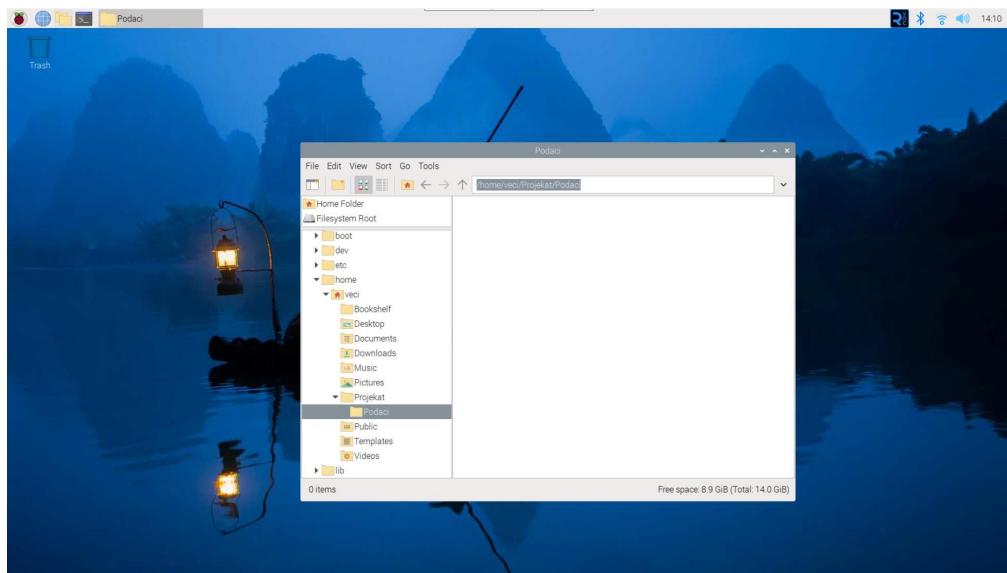
if __name__ == '__main__':
    try:
        setup()
        main()
    except KeyboardInterrupt:
        destroy()

```

## 4. Смештање мерних података на Raspberry Pi microSD картици или USB Flesh меморији

### 4.1 Смештање мерних података на microSD картицу (фолдер унутар Raspberry Pi уређаја)

На самом почетку потребно је наћи путању до фолдера у коме желимо да сместимо мерење податке добијене од сензора. Нпр. /home/veci/Projekat/Podaci



Слика 29. Приказ путање до фолдера

## Код у Raspberry Pi-y

import os - пружа функционалности за интеракцију са опретаивним системом, омогућава извршење различитих операција као што су рад са фајловима, директоријумима, процесима, окружењима и другим системским ресурсима.

```
import time
```

```
def distance(): # Функција враћа мерени податак
```

```
    return 10.5 # Пример вредности
```

```
def write_data_to_file(distance):
```

```
    # Путања до фолдера за смештање података
```

```
    folder_path = "/home/veci/Projekat/Podaci"
```

```
    # Провера да ли фолдер постоји ако не, креирај га
```

```
    if not os.path.exists(folder_path):
```

```
        os.makedirs(folder_path)
```

```
    filename = "data_sensor.txt" - Генерирање имена фајла
```

```
    filepath = os.path.join(folder_path, filename) - Генерирање пуне путање до фајла
```

```
    with open(filepath, 'w') as file: - Отварање фајла за писање
```

```
        file.write(f"Udaljenost: {distance:.2f} cm\n") - Упис података у фајл
```

```
        file.write(f"Vreme: {time.strftime('%Y-%m-%d %H:%M:%S')}\n") - Упис времена када су ти подаци настали
```

```
    print(f"Podaci su sačuvani u {filepath}")
```

```
def main(): # Главна функција која покреће мерење и писање података
```

```
    dist = distance()
```

```
    write_data_to_file(dist)
```

```
if __name__ == "__main__":
```

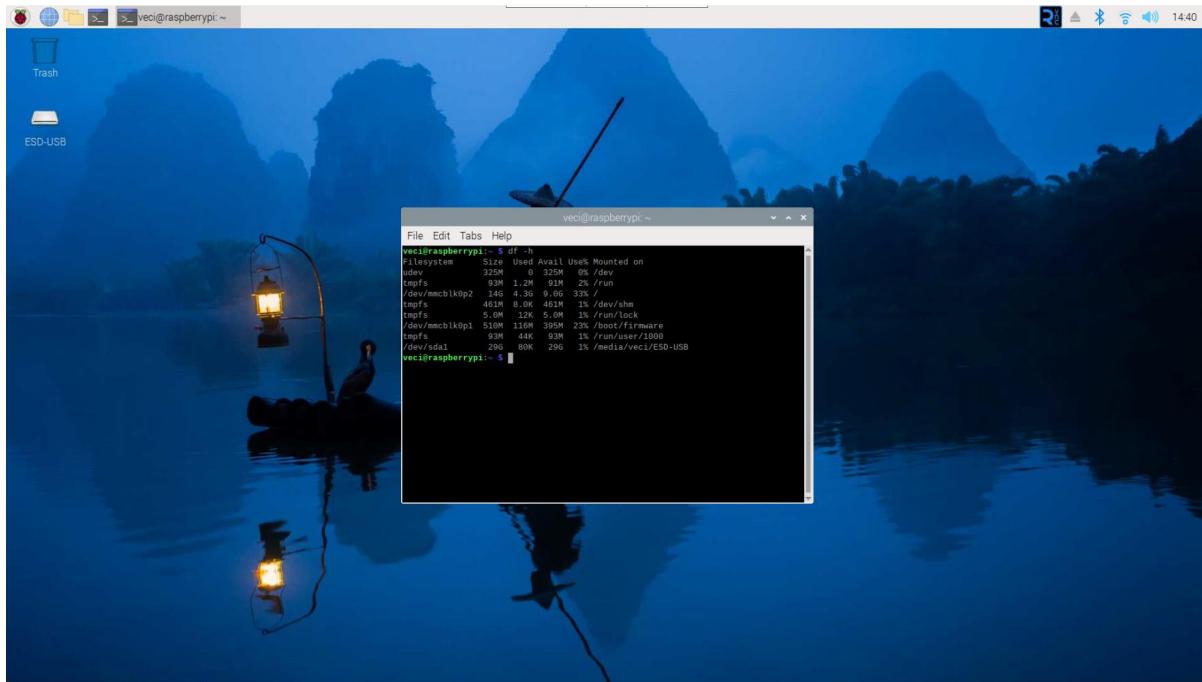
```
    main()
```

## 4.2 Смештање мерних података на USB Flesh меморију

На самом почетку потребно је наћи путању која води до USB Flesh меморију

Путању налазимо функцијом

**df -h**



```
File Edit Tabs Help
veci@raspberrypi: ~ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev           168M   10M  158M  6% /
tmpfs          93M  1.2M  91M   2% /run
/dev/mmcblk1p2  14G  4.3G  9.0G  33% /dev
tmpfs          461M  8.0K  461M   1% /var/tmp
tmpfs          1024M  16K  1024M  2% /run/lock
/dev/mmcblk1p1  518M  116M  395M  23% /boot/firmware
tmpfs          93M  44K  93M   1% /run/user/1000
/dev/sda1       29G  80K  29G   1% /media/veci/ESD-USB
veci@raspberrypi: ~ $
```

Слика 30. Приказ комнлог прозора

У тражимо путању где пише USB у овом случају је /media/veci/ESD-USB

### Код у Raspberry Pi-y

```
import os
import time
```

```
def distance(): # Функција која враћа измерену удаљеност са сензора
    return 10.5 # Пример вредности удаљености
```

```
def write_to_usb(distance):
    usb_path = "/media/veci/ESD-USB" # Путања до монтираног USB уређaja
    # Генеришемо име фајла
    filename = f"data_sensor.txt"
    filepath = os.path.join(usb_path, filename) # Отварање фајла на USB-у за писање
```

```

with open(filepath, 'w') as file:
    file.write(f"Удаљеност: {distance:.2f} cm\n")
    file.write(f"Време: {time.strftime('%Y-%m-%d %H:%M:%S')}\n")
print(f"Подаци су сачувани у {filepath}")

def main(): # Главна функција која покреће мерење и писање података
    dist = distance()
    write_to_usb(dist)

if __name__ == "__main__":
    main()

```

На крају потребно је искључити USB Flash меморију сигурно, то се врши наредбом

**sudo umount /media/veci/ESD-USB**

ову команду уписујемо у командни прозор.

## 5. Гашење Raspberry Pi уређаја

Када смо завршили са коришћењем Raspberry Pi уређаја потребно га је правилно угасити.

Прво је потребно укуцати команду за гашење у кодмандном прозору.

**sudo shutdown -h now**

Потребно је сачекати пар минута нако тога извлачимо кабал за напајање.

## **Литература**

1. Ибрахим, Д. 2019.: *Од основних до напредних пројеката.* infoelektronika
2. Sensor Kit V2.0 for Raspberry Pi B+ make it easy & make it fun

Пријето са: <https://docs.sunfounder.com/projects/sensorkit-v2-pi/en/latest/>

3. SunFounder\_SensorKit\_for\_RPi2

Пријето са: [GitHub - sunfounder/SunFounder\\_SensorKit\\_for\\_RPi2: SunFounder Sensor Kit V2.0 for Raspberry Pi 2](#)

4. Raspberry Pi Workshop for Beginners

[Raspberry Pi Workshop - Overview \(youtube.com\)](#)

5. Raspberry Pi and Python Tutorials

[\(481\) Introduction and Parts - Raspberry Pi and Python tutorials p.1 - YouTube](#)