

РЪКОВОДСТВО ЗА КОМПЛЕКТА



SmartHome
4SENIORS



Co-funded by
the European Union

Подкрепата на Европейската комисия за издаването на тази публикация не представлява одобрение на съдържанието, което отразява единствено възгледите на авторите, и Комисията не носи отговорност за използването на съдържащата се в нея информация.

ПРЕДУПРЕЖДЕНИЕ

Когато използвате този продукт, моля, обърнете внимание на следното:

1. Този продукт съдържа много малки части. Поглъщането или неправилното използване на някоя от тези части може да доведе до сериозни медицински усложнения, включително смърт. Потърсете незабавно медицинска помощ, ако се случи инцидент.
2. Използването на този продукт и неговите части в близост до електрически контакти за променлив ток или други вериги е строго забранено поради потенциалния риск от токов удар.
3. Използването на този продукт в близост до течности или огън е строго забранено.
4. Съхранявайте проводящи материали далеч от този продукт.
5. Не позволявайте на малки деца да използват този продукт без надзора на възрастни. Този продукт трябва да се съхранява на място, недостъпно за деца и домашни любимци.
6. Не съхранявайте и не използвайте този продукт в екстремни условия, като например висока или ниска температура, висока влажност, пряка слънчева светлина и др.
7. Не забравяйте да прекъсвате веригата, когато тя не е необходима.
8. Някои части на този продукт могат да станат топли на допир, когато се използват в определени схеми, което е нормално.
9. Неправилната употреба може да доведе до прегряване.
10. Използването на компоненти, които не са в съответствие със спецификацията, може да доведе до повреда на продукта.

Въведение

Комплектът SmartHome4SENIORS е създаден на базата на микроконтролера Raspberry Pi Pico (RPi Pico). Комплектът е създаден в рамките на едноименния проект, съфинансиран по програмата "Еразъм+", с номер на проекта 2021-1-DE02-KA220-ADU-000033587.

Автоматизацията на интелигентни домове е съществуваща тенденция, която помага на хиляди хора да опростят процесите в домакинствата си. Комплектът SmartHome4SENIORS е замислен и разработен с оглед на по-ниската степен на готовност на възрастните хора да възприемат и разбират технологичните решения за интелигентен дом.

Комплектът има за цел да насърчи безопасния и здравословен начин на живот на възрастните хора чрез практическо обучение. Комплектът предлага чудесна възможност на потребителите да навлязат в света на интелигентните решения за домашна автоматизация "Направи си сам".

Комплектът включва целия необходим хардуер (микроконтролер, електроника, сензори, периферни устройства и т.н.), включващ концепции за физически изчисления и програмиране.

Обхватът на настоящото ръководство е:

- Информирайте се за компонентите на комплекта и използването на основните електронни елементи.
- Напътства ви стъпка по стъпка за ефективно сглобяване на комплекта, като взема предвид съответните предпазни мерки.
- Предоставяне на уроци за използване на компонентите и свързването им с микроконтролера RPi Pico.

- Предоставяне на учебни материали с функционалността и обхвата на всеки електронен компонент.

Наслаждавайте се на четенето и се забавлявайте чрез практически упражнения и експерименти с комплекта SmartHome4SENIORS

Съдържание

Въведение	1
Съдържание	2
Включен в комплекта SmartHome4SENIORS	4
Обяснение на компонентите	6
1. Какво представлява макетната платка?	6
2. Какво представлява резисторът?	7
3. Какво представлява кондензаторът?	11
4. Какво представлява диодът?	12
5. Какво представлява джъмпер кабелът?	12
Подготовка на проекта.....	14
Сглобяване на комплекта SmartHome	24
Монтаж на електрониката на комплекта SmartHome	32
Основни уроци.....	39
0. "Здравейте, хора от SmartHome!"	39
1. Управление на светодиод	41
2. Бутон за натискане.....	43
3. Звуков сигнал.....	46

4. Потенциометър	49
Уроци за напреднали	52
5. Модул за светодиодни светофари	53
6. Фоторезистор LDR	56
7. Двигател за постоянен ток (малък вентилатор)	59
8. Сервомотор SG-90	62
9. OLED I2C SSD1306 дисплей	65
10. RFID четец RC522	70
Уроци със сензори	74
11. Сензор за дъждовни капки	74
12. Ултразвуков сензор HC-SR04	77
13. PIR сензор за движение	81
14. Сензор DHT11	Error! Bookmark not defined.
15. Сензор за пламък	Error! Bookmark not defined.
16. Сензор за откриване на газ на MQ-135	Error! Bookmark not defined.
ПРИЛОЖЕНИЕ: Обобщаваща таблица на MicroPython	
Error! Bookmark not defined.	

Включен в комплекта SmartHome4SENIORS

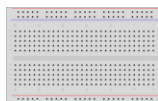
1 x Микроконтролер
Raspberry Pi Pico



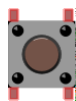
1 x модул за
захранване MB-102



1 x Бяла платка за хляб
830 бр.



1 x бутон за натискане
&
1 x капачка за бутон



1 x зумер



5 x резистори 220 Ohm
&
2 x 1k Ohm



1 x LDR фоторезистор



1 x 100 μ F кондензатор



4 x LED 3 мм
(синьо, зелено,
червено, жълто)



1 x RGB LED



1 x светодиоден модул на светофара



1 x вентилатор (DC мотор)



1 x модул за управление TIP-120



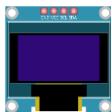
1 x диод 1N4007



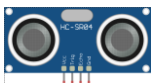
1 x сервомотор SG90



1 x OLED SSD1306 I2C дисплей



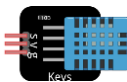
1 x Ултразвуков сензор HC-SR04



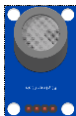
1 x PIR сензор за движение HC-SR501



1 x Цифров сензор за температура и влажност DHT11



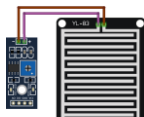
1 x Сензор за качество на въздуха MQ-135



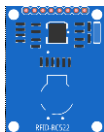
1 x сензор за откриване на пламък



1 x Сензор за дъждовни капки



1 x RFID четец RC522



1 x RFID етикет 3.56MHz



1 x Ротационен
потенциометър Linear
В1k Ohm



1 x кабел от USB към
micro-USB



6 x AA 1,5V батерии и 1
x държач за батерии



9 x джъмперни кабела



19 x метални болтове 2
мм и 19 x метални
гайки 2 мм

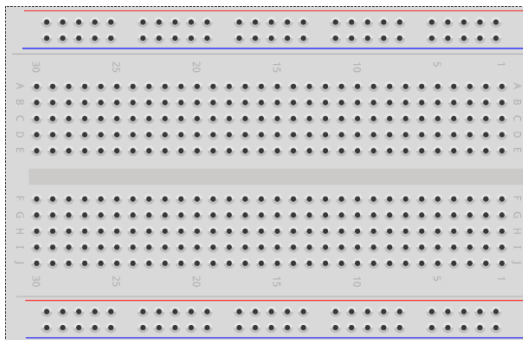


4 x винтове за метал 2
мм



Обяснение на компонентите

1. Какво представлява макетната платка?



Платката за хляб е пластмасова платка с малки отвори, които позволяват лесното поставяне на електронни компоненти (транзистори, резистори, чипове и др.) за създаване на прототип (изграждане и тестване) на електронна схема. Вътрешността е съставена от редици от

малки метални щипки, които държат кабелите, които трябва да бъдат свързани.

На повечето платки са изписани редове с цифри, букви и знаци плюс и минус. Целта на етикетите е да ви помогнат да откриете определени отвори на платката, за да можете да следвате указанията при изграждането на схема.

Дългите ленти от двете страни на платката обикновено се маркират с червено и синьо или червено и черно и съответно със знаци плюс (+) и минус (-). Тези редове се наричат шини или релси и обикновено се използват за подаване на електрическо захранване към схемата, когато е свързана към захранващ блок (батерия или външно захранване).

Положителната "шина" е маркирана в червено, има знак плюс (+) и осигурява захранването.

Отрицателната "шина" е маркирана в синьо или черно, има знак минус (-) и осигурява заземяването.

Предимства на използването на платка:

- Улеснява бързата проверка на прости и сложни вериги, както и лесната проверка на вериги в началния им етап.
- Лесно регулиране.
- Гъвкавост.
- Без пробиване на отвори.
- Не се изисква запояване.
- Лесно отстраняване на грешки в схемите и програмите.

2. Какво представлява резисторът?



Резисторът е малък пакет от съпротивления. Използването му в дадена верига намалява тока с точно определено количество. Съпротивлението на резистора се определя по модел от цветни ленти.

Color	1st Band	2nd Band	3rd Band (5-Band Only)	Multiplier (3rd or 4th Band)	Tolerance (Last Band)
Black	0	0	0	1	
Brown	1	1	1	10	± 1%
Red	2	2	2	100	± 2%
Orange	3	3	3	1000	
Yellow	4	4	4	10000	
Green	5	5	5	100000	± 0.5%
Blue	6	6	6	1000000	± 0.25%
Violet	7	7	7	10000000	± 0.1%
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1	± 5%
Silver				0.01	± 10%
None					± 1%

(Кредит за изображение: Future Owns) на разположение на <https://www.tomshardware.com/how-to/resistor-color-codes>

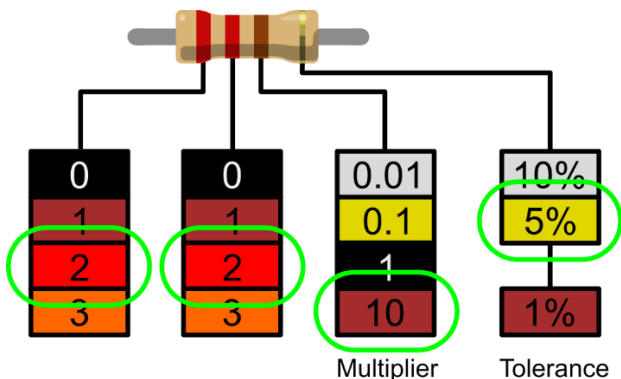
Обичайни цветови кодове на резисторите и тяхното използване:

Тип на резистора	4-лентов цветови код	5-лентов цветови код	Общи употреби
220Ohm	Червено-червено-кафяво-златно	Червено-червено-черен-черен-златен	Защита на LED светлината
1K Ohm (1Kiloohm)	Кафяво-черно-червено-златно	Кафяво-черно-черно-кафяво-златно	Защита на LED, делител на напрежение

Резисторите нямат полярност, така че могат да се използват във всякаква ориентация във веригата. Но за да определим правилните стойности на цветовете кодове на резисторите, трябва да разберем цветните ленти върху резистора.

При типичен четирилентов хоби резистор за ниво има три цвята в група. Това са първата, втората значеща цифра и множителят. Последната група е толерансът на резистора, границата на грешка, ако щете. За повечето любители толерансът от 5 % (злато) е идеален и обичаен.

Резистор 220 ома (4-лентов)

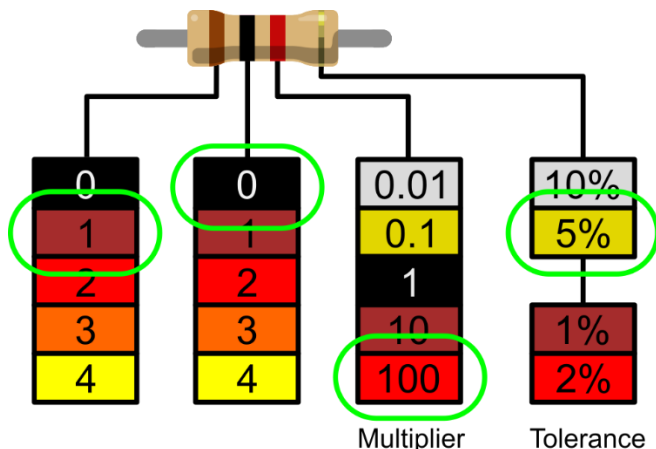


(Кредит за изображение: Future) на разположение на <https://www.tomshardware.com/how-to/resistor-color-codes>

1. Първата значеща цифра е червена и с помощта на дешифратора виждаме, че червеното има стойност 2.
2. Втората значеща цифра също е червена, така че получаваме 22.
3. Множителят е кафяв и се дешифрира до 10. Ако умножим 22 по 10, ще получим 220.
4. Последната лента - толерантност - е златна. Златото е 5%, което означава, че можем да приемем съпротивление с 5% допустима грешка.

За производителите, които се нуждаят от по-голяма прецизност, се предлагат и петлентови резистори с трета значеща цифра. Допълнителната цифра осигурява яснота, която може да бъде от съществено значение във вериги, чувствителни към съпротивлението, например научни и инженерни инструменти.

Резистор 1K Ohm (4-лентов)



(Кредит за изображение: Tom's Hardware) на разположение на <https://www.tomshardware.com/how-to/resistor-color-codes>

1. Линията 1st е кафява и с помощта на декодера можем да видим, че стойността е 1.
2. Линията 2nd е черна, така че получаваме 10.
3. Множителят е червен и се дешифрира до 100. Ако умножим 10 по 100, ще получим 1000.
4. Последната лента - толерантност - е златна. Златото е 5%, което означава, че можем да приемем съпротивление с 5% допустима грешка.

3. Какво представлява кондензаторът?



Кондензаторът е устройство, което съхранява електрическа енергия в електрическо поле. Той е пасивен електронен компонент с два терминала. Състои се от два електрически

проводника, които са разделени на определено разстояние. Пространството между проводниците може да бъде запълнено с вакуум или с изолационен материал, известен като диелектрик. (Уикипедия)

Кондензаторът 100uF е електролитен разединителен кондензатор. Тези кондензатори са чудесни потискатели на преходни процеси/напрежения и използването на такъв между захранването и земята на веригата гарантира безпроблемно подаване на енергия.

4. Какво представлява диодът?

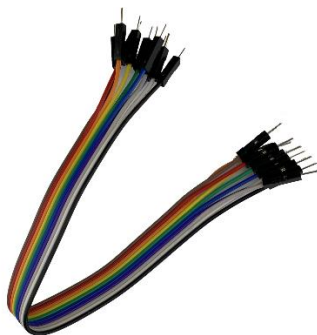


Диодът е електронен компонент с два терминала, който провежда ток предимно в една посока (асиметрична проводимост); той има ниско (в идеалния случай нулево) съпротивление в едната посока и високо (в идеалния случай безкрайно) съпротивление в другата.

Най-често срещаната функция на диода е да позволява преминаването на електрически ток в една посока (наречена директна посока на диода), като същевременно го блокира в противоположната посока (обратна посока). По този начин диодът може да се разглежда като електронна версия на възвратен клапан. Това еднопосочно поведение се нарича изправяне и се използва за преобразуване на променлив ток (AC) в постоянен ток (DC). В качеството си на токоизправители диодите могат да се използват за такива задачи като извличане на модулация от радиосигнали в радиоприемници. (Уикипедия)

<https://en.wikipedia.org/wiki/Diode>

5. Какво представлява джъмпер кабелът?



Кабелите за джъмпер са просто проводници, които имат щифтове за свързване във всеки край, което позволява да се използват за свързване на две точки една към друга без запояване. Обикновено джъмперните кабели се използват с дъски и други инструменти за прототипиране, за да се улесни промяната на схемата при необходимост. Цветовете вариации на проводниците могат да се използват като предимство за разграничаване на видовете връзки, като например заземяване или захранване.

Прекъсвачите обикновено се предлагат в три варианта: мъжки към мъжки, мъжки към женски и женски към женски. Разликата между тях е в крайната точка на проводника. Мъжките краища имат изпъкнал щифт и могат да се включват в различни неща, докато женските краища нямат такъв щифт и се използват за включване в различни неща. Най-често срещаните проводници са мъжки-мъжки. Когато свързвате два порта на платка, най-често се изисква мъжки проводник. (<https://blog.sparkfuneducation.com/what-is-jumper-wire>)

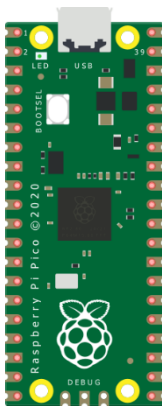
Подготовка на проекта

1. Readme Преди употреба

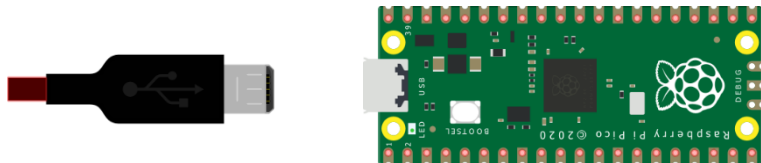
ЗАБЕЛЕЖКА: Тъй като всички експерименти са свързани с електрически вериги, неправилно свързване или късо съединение може да повреди вашата развойна платка RPi Pico. Моля, винаги проверявайте веригата отново, преди да свържете захранването.

2. Микроконтролер Raspberry Pi Pico

Това е Raspberry Pi Pico:

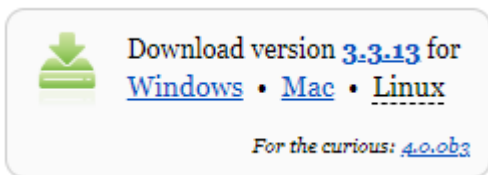


Включете кабела micro-USB в порта от лявата страна на платката.



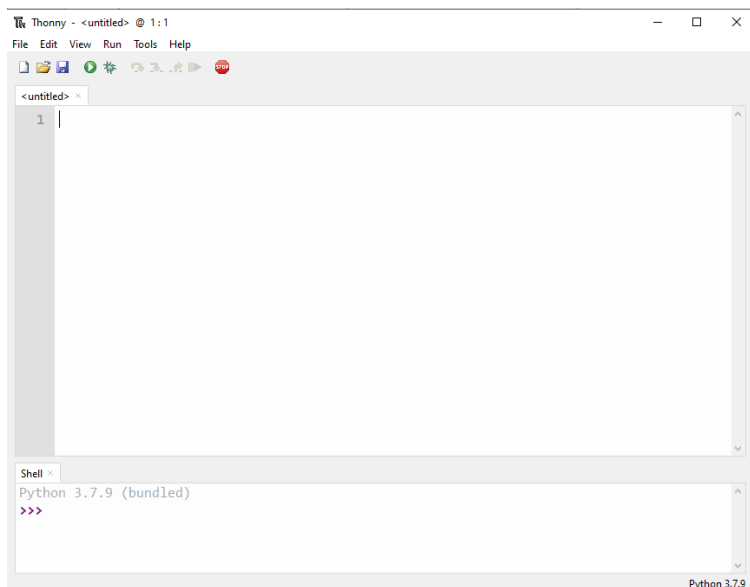
3. Инсталиране на Thonny IDE

Посетете <https://thonny.org> и изберете подходящата операционна система. Следвайте инструкциите, за да завършите инсталацията.



В това ръководство всички уроци са програмирани в Windows 10, като се използва микроконтролер RPi Pico и подходящ фърмуер.

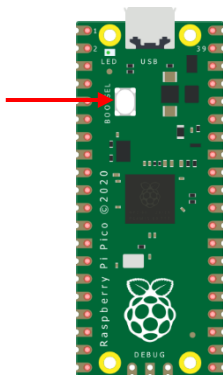
След като инсталацията приключи, отворете Thonny от компютъра.



4. Инсталиране на фърмуера

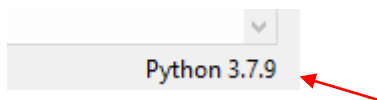
RPi Pico може да се програмира с помощта на вариант на Python, наречен MicroPython. За да използвате MicroPython на RPi Pico, първо трябва да инсталирате неговия фърмуер.

Стъпка 1: Намерете бутона BOOTSEL на вашия Raspberry Pi Pico.

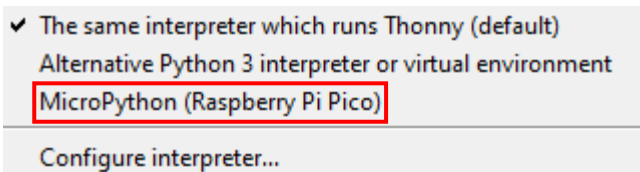


Стъпка 2: Натиснете бутона BOOTSEL и го задръжте, докато свързвате другия край на micro-USB кабела към компютъра.

Стъпка 3: В долния десен ъгъл на Thonny ще видите версията на Python, която използвате в момента.

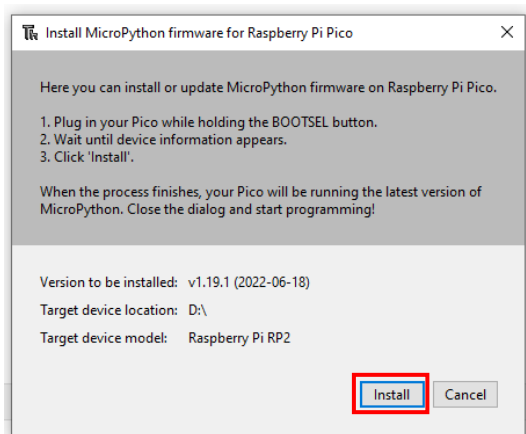


Кликнете върху версията на Python и изберете MicroPython (Raspberry Pi Pico)

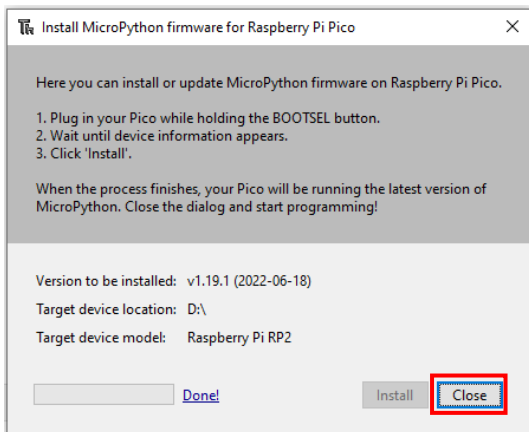


Ако не виждате тази опция, уверете се, че кабелът е свързан правилно в Pico и/или в компютъра ви.

Стъпка 4: Ще се появи диалогов прозорец, в който ще бъдете помолени да инсталирате най-новата версия на фърмуера на вашия Pico. Щракнете върху бутона **Инсталиране**, за да копирате фърмуера на вашия Pico.



Стъпка 5: Изчакайте инсталацията да приключи и щракнете върху **Затвори**.



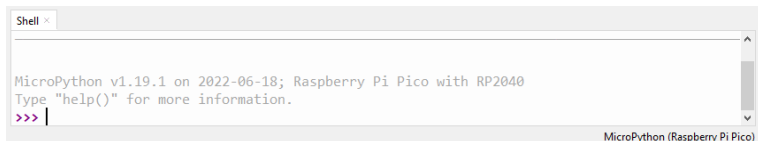
Не е необходимо да повтаряте процеса всеки път, когато свързвате Raspberry Pi Pico към компютъра си, така че следващия път просто го включете и сте готови за работа.

5. Въведение в програмирането на MicroPython

Сега ще използвате Thonny IDE, за да стартирате няколко прости кода на Python, за да се запознаете с Thonny's Shell и MicroPython.

Първо се уверете, че вашият Raspberry Pi Pico е свързан към компютъра и сте избрали интерпретатора MicroPython, както е обяснено в предишния раздел.

Панелът Shell в долната част на редактора на Thonny трябва да изглежда по следния начин:



Thonny е готов да комуникира с вашия Pico с помощта на REPL (read-eval-print loop) рамка, която ви позволява да пишете код директно в Shell и да получавате резултати.

Въведете следната команда:

печат ("Здравейте!")

След това натиснете клавиша Enter и вижте следния изход:

```
Shell <
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> print("Hello!")
Hello!
>>>
```

MicroPython (Raspberry Pi Pico)

MicroPython ви позволява да добавяте специфични за хардуера модули, като например `machine`, които можете да използвате за програмиране на вашия Pico. В следващия пример ще използвате модула `machine`, за да включите вградения светодиод на Pico.

Напишете следния код в Thonny's Shell:

```
от machine внос Pin
led = Pin(25, Pin.OUT)
led.value(1)
```

```
Shell <
>>> from machine import Pin
>>> led = Pin(25, Pin.OUT)
>>> led.value(1)
>>> |
```

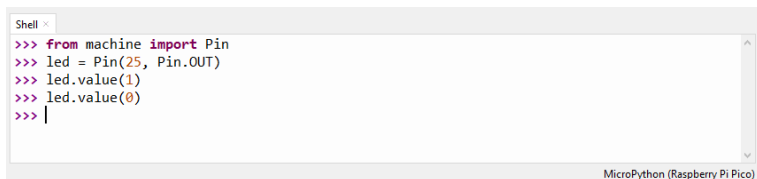
MicroPython (Raspberry Pi Pico)

Натиснете клавиша Enter и вграденият светодиод на Pico веднага ще се включи.



За да изключите светодиода, напишете следния код:

```
led.value(0)
```



```
Shell
>>> from machine import Pin
>>> led = Pin(25, Pin.OUT)
>>> led.value(1)
>>> led.value(0)
>>> |
```

MicroPython (Raspberry Pi Pico)

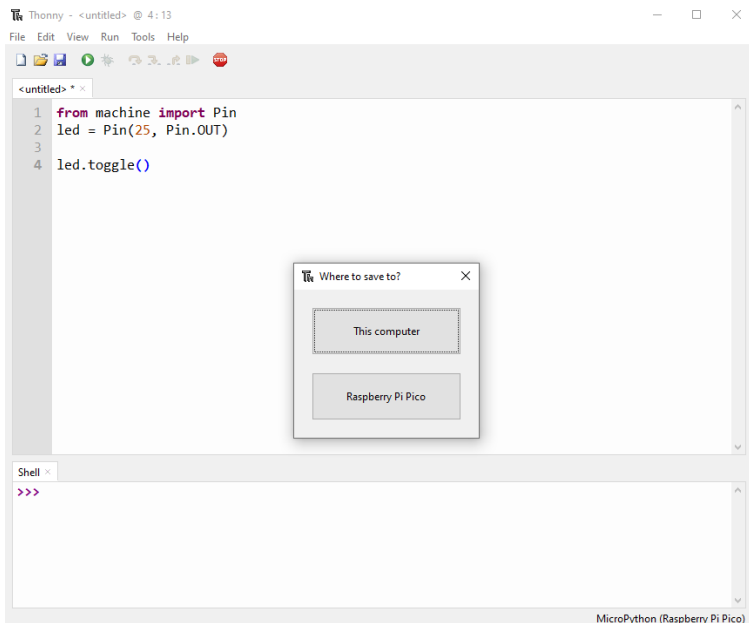
В останалата част на този раздел ще напишете първата си "истинска" програма, която ще накара вградения светодиод да мига всеки път, когато стартирате програмата си.

Thonny Shell е полезен за изпробване на бързи команди и за да се уверите, че всичко работи правилно. По-дългите програми обаче трябва да се записват в .py файл. С помощта на Thonny можете да записвате програми директно на Raspberry Pi Pico и след това да ги стартирате.

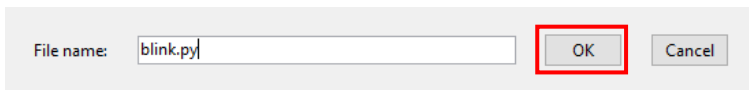
Отворете Thonny Python и в главния панел на редактора напишете следния код:

```
от machine внос Pin
led = Pin(25, Pin.OUT)
led.toggle()
```

Сега запазете програмата си, като щракнете върху иконата Save (Запазване) в горния ляв ъгъл или като натиснете Ctrl+S на клавиатурата.



Thonny ще ви попита къде искате да бъде запазена вашата програма. Изберете **Raspberry Pi Pico**. Запишете файла като *blink.py* и щракнете върху **OK**. Винаги трябва да добавяте разширението *.py*, за да може Thonny да разпознае файла като Python файл.



Сега всеки път, когато щракнете върху иконата Play, трябва да виждате как вграденият светодиод се включва и изключва.

Ако направите още една стъпка напред, можете да накарате вградения светодиод да мига с определена скорост.

Напишете следния код и запазете програмата си, като използвате същото име, както по-горе.

```
от machine внос Pin
от time внос sleep
led = Pin(25, Pin.OUT)
докато е вярно:
    led.toggle()
    заспиване (1)
```

Сега, когато стартирате програмата, вграденият светодиод ще мига на всеки 1 секунда, докато не спрем програмата. За да спрете програмата, можете да щракнете върху иконата STOP или да натиснете Ctrl+C на клавиатурата.

В следващите уроци ще научим как да добавяме и управляваме друга електроника и сензори и да създаваме програми, които могат да комуникират помежду си.

Сглобяване на комплекта SmartHome

Процесът на сглобяване изисква следните елементи:

- 6 x парчета шперплат
- 10 x метални болтове
- 10 x метални гайки



Процедурата е проста и се **извършва в 7 стъпки**. Всички парчета шперплат се маркират в десния страничен ъгъл със следното обозначение:

BP: Базова част

B: Част от задната страна

F: Предна част

L: лява част

R: Дясна част

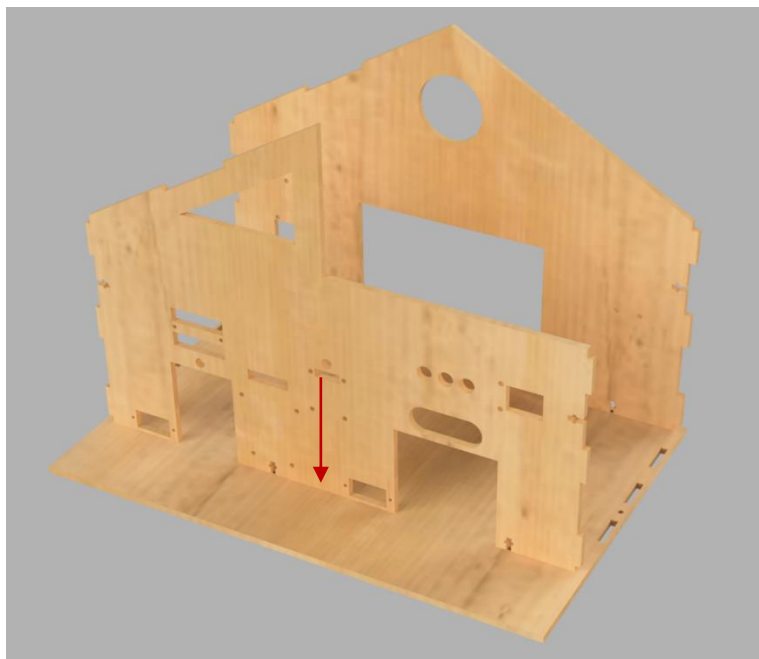
Стъпка 1: Поставете основата от шперплат върху хоризонтална повърхност.



Стъпка 2: Поставете парчето от задната страна, както е показано на изображението.



Стъпка 3: Поставете парчето от предната страна, както е показано на изображението.



Стъпка 4: Поставете лявата част на шперплата, както е показано на изображението.

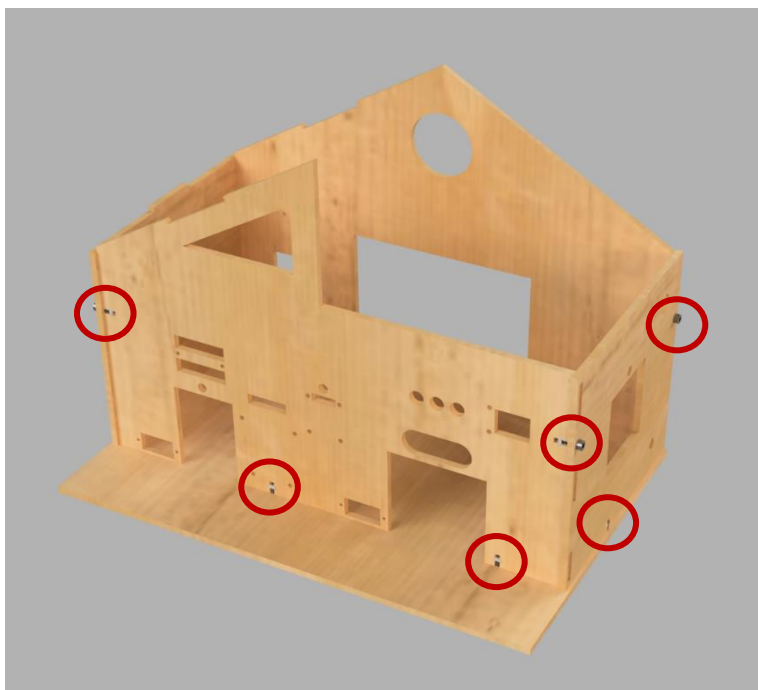


Стъпка 5: Поставете дясната страна на шперплата, както е показано на изображението.

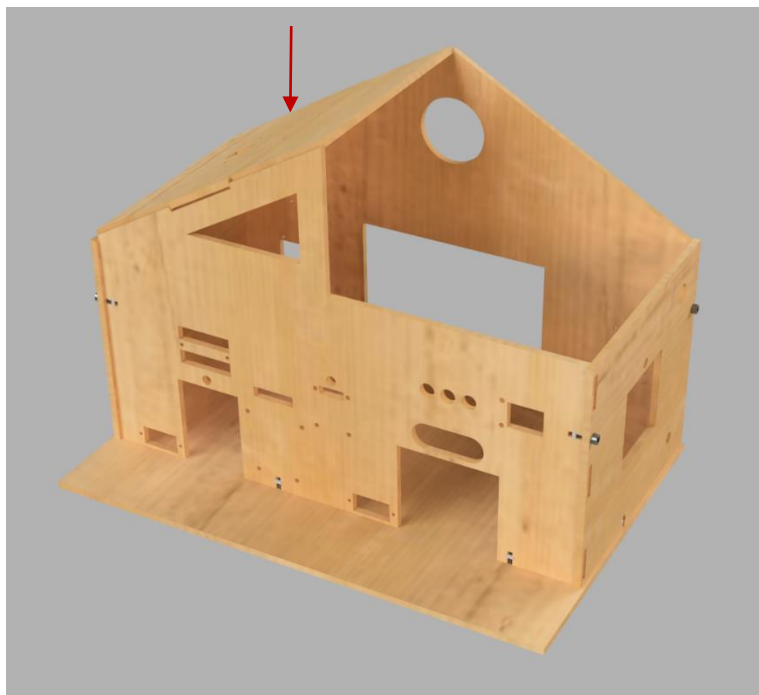


Стъпка 6: Затегнете парчетата шперплат, като използвате 10 болта и 10 гайки.

- 6 болта и гайки са поставени под основната част и отговарят за сглобяването на стените.
- На стените от лявата и дясната страна са поставени 4 болта и гайки, които поддържат стабилността на конструкцията.

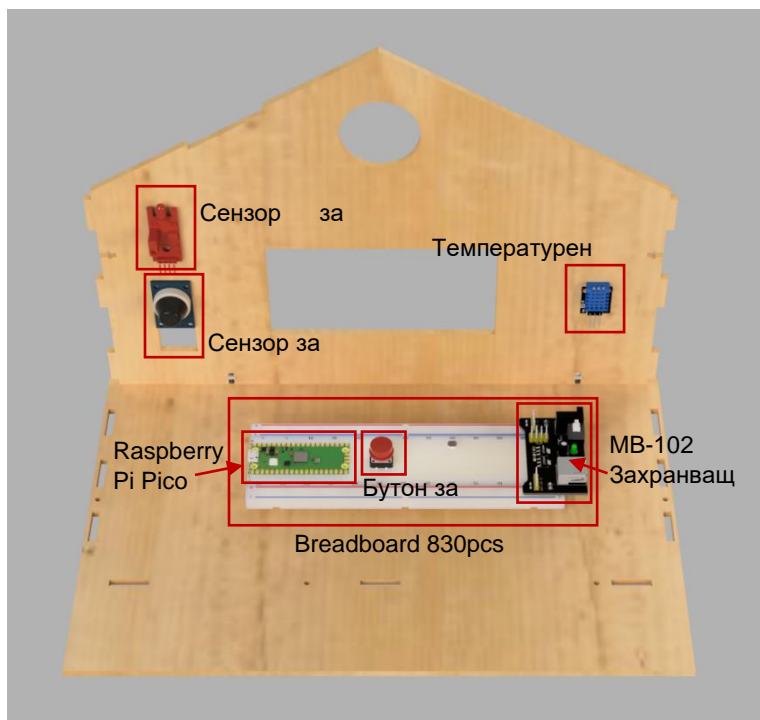


Стъпка 7: Поставете покрива от шперплат върху модела на къщата, както е показано на изображението:



Монтаж на електрониката на комплекта SmartHome

В модела на къщата SmartHome трябва да се монтират няколко електронни устройства. На следващите изображения е представено подробно каква електроника да се монтира и как да се монтира на различните слотове. Ще ви трябват също така отвертка с глава "Филипс" и клещи.



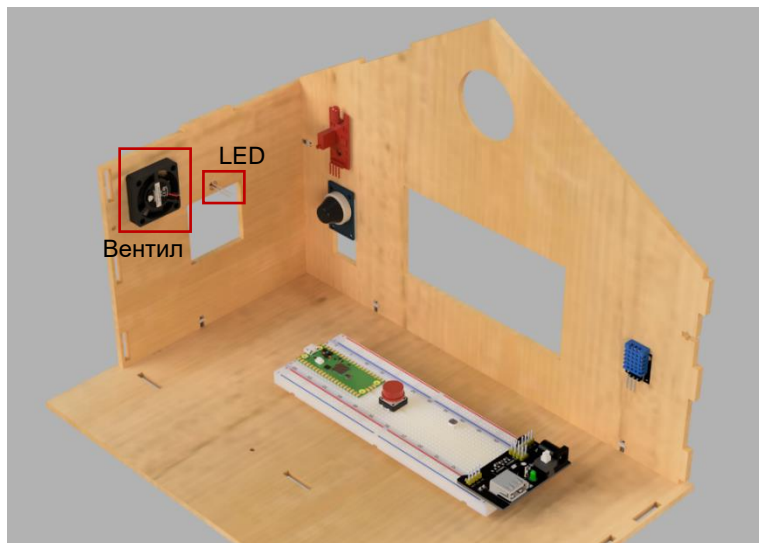
Поставете Raspberry Pi Pico върху платката, заедно със захранващия модул (MB-102) и бутона. Под дъската за хляб има залепваща повърхност, затова не забравяйте да

отлепите защитния слой и след това да я поставите върху основната част на къщата.

Има някои електронни устройства, които трябва да се монтират на задната стена. Това са сензорът за пламък, сензорът за качество на въздуха и сензорът за температура и влажност DHT11. Необходимо е да използвате следните елементи:

- **Сензор за пламък:** 1 x болт и 1 x гайка. Монтирайте го през средната част на сензора и се уверете, че сензорът (черната част) гледа нагоре.
- **Сензор за качество на въздуха:** 2 x болт и 2 x гайка. Монтирайте го през горния ляв и горния десен монтажен отвор.
- **DHT11:** 2 x болт и 2 x гайка. Трябва да огънете сензора (синята част) напред, за да получите достъп до монтажните отвори. След това монтирайте сензора през горния ляв и горния десен монтажен отвор.

Върху лявата част ще трябва да монтирате **вентилатор за постоянен ток** и **светодиодна** лампа. За **вентилатора за постоянен ток** ще използвате болтовете и гайките, които са включени в комплекта (4 x болтове и 4 x гайки). Поставете **светодиода** в монтажния отвор и триенето ще го задържи на място.



Върху дясната част ще трябва да монтирате 5V зумер, потенциометър и светодиод.

Поставете **зумера** в горния десен ъгъл (уверете се, че щифтовете му са от вътрешната страна на модела на къщата) и триенето ще го задържи на място.

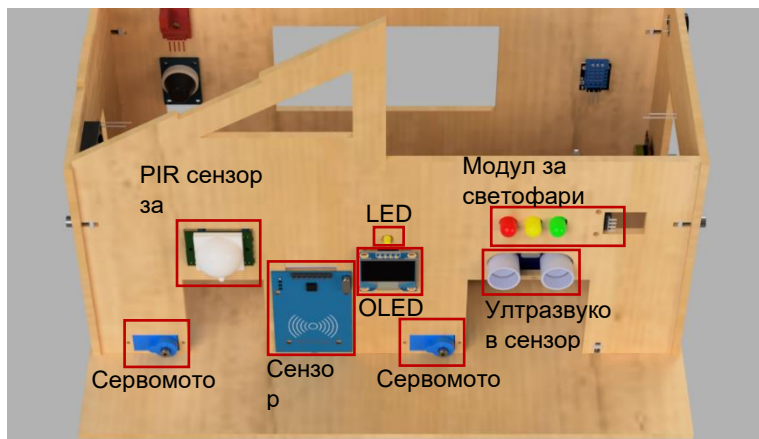
Потенциометърът вече има монтиран пръстен с гайка, който трябва да отвиете. Поставете потенциометъра от вътрешната страна на къщата към външната, така че лостът да изглежда както на изображението по-долу. След това завийте обратно пръстена на гайката, докато се затегне достатъчно, за да се задържи на място, когато завъртите лоста.

Накрая трябва да се инсталира **светодиод**. Поставете светодиода в монтажния отвор и триенето ще го задържи на място.

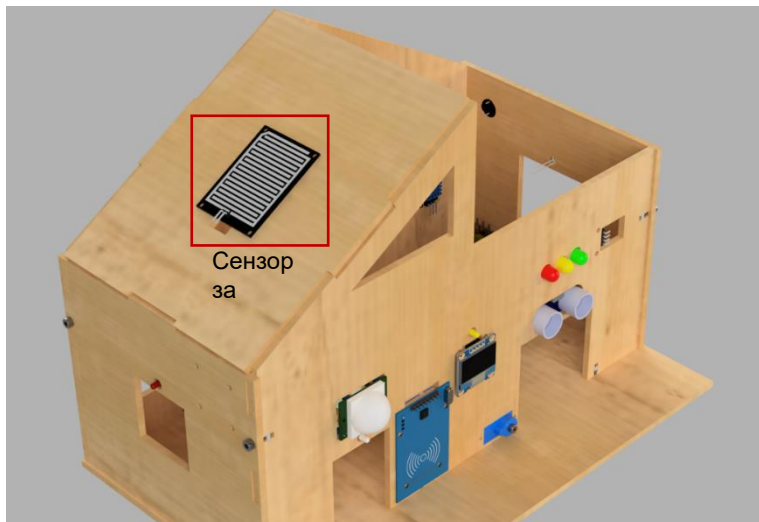


В предната част на модела на къщата трябва да се монтират няколко сензора и електроника.

- **Сервомотори:** 4 х винтове. Поставете ги от вътрешната страна на къщата към външната и ги монтирайте с помощта на винтовете, включени в опаковката.
- **PIR Сензор за движение:** 2 х болт и 2 х гайка. Монтирайте го през горния ляв и долния десен монтажен отвор.
- **Сензор RFID:** 2 х болт и 2 х гайка. Монтирайте го през горния ляв и долния десен монтажен отвор.
- **OLED дисплей:** 2 х болт и 2 х гайка. Монтирайте го през горния ляв и долния десен монтажен отвор.
- **Модул за светофари:** 2 х болт и 2 х гайка. Поставете модула от вътрешната страна на къщата към външната. След това го монтирайте през двата монтажни отвора от дясната страна.
- **Ултразвуков сензор:** Поставете го от вътрешната страна на къщата към външната. Триенето ще го задържи на мястото му.
- **LED:** Поставете светодиода в монтажния отвор и триенето ще го задържи на място.



Накрая на покрива трябва да се монтира **сензорът за дъждовни капки**. Използвайте 2 x болта и 2 x гайки и го монтирайте през монтажните отвори в горния ляв и долния десен ъгъл.



След като всички електронни устройства и сензори са монтирани на място, трябва да ги свържете с помощта на кабелите за джъмperi, които са включени в пакета. Тази процедура обаче ще бъде обяснена в раздела с уроци, където ще бъдат обяснени различните GPIO изводи на Raspberry Pi Pico и изводите на всеки електронен компонент или сензор.

В опаковката ще намерите и 6 батерии AA и държач за батерии. Поставете батериите върху държача за батерии и засега го дръжте настрана. В по-късен урок ще научите къде трябва да го поставите и как трябва да бъде свързан със захранващия модул MB-102.

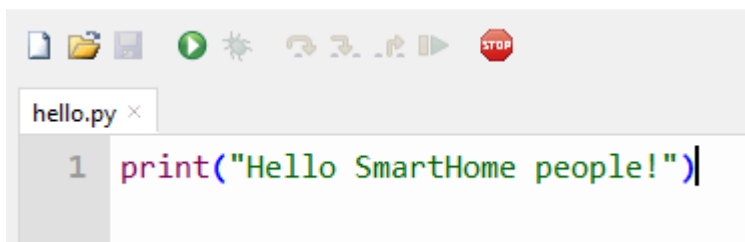
Основни уроци

0. "Здравейте, хора от SmartHome!"

В този основен урок ще научим как да отпечатаме просто съобщение в Thonny с помощта на програмиране в Python.

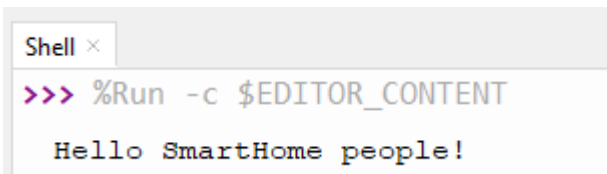
Функцията print()

1. Отидете в редактора на Thonny, напишете следния код и натиснете иконата за възпроизвеждане. Thonny ще ви помоли първо да запазите програмата си. Запишете я под името hello.py.



```
hello.py x
1 print('Hello SmartHome people!')
```

2. Проверете прозореца на обвивката.



```
Shell x
>>> %Run -c $EDITOR_CONTENT
Hello SmartHome people!
```

3. Добра работа! Току-що създадохте първата си програма на Python.

Функцията `print` е вградена функция на Python, която ни позволява да отпечатваме текст в обвивката. Тя може да приема и параметри. Създайте нова програма и копирайте следния код, след което натиснете Play и вижте как се появява текстът в шела.

```

hello.py x
1 print(1,2,3,4,5) #This is a comment!
2 print("I am ",2,"awesome") #1 line
3 print("Python is") #1 line
4 print("amazing") #1 line
5 print("I cant wait.....\n to learn more") # 2 lines of output!
  
```

Както можете да видите в прозореца на обвивката, всяка функция за печат отпечатва текст на отделен ред. Въпреки това, ако използвате "\n" (символ за нов ред), можете да смените реда в същата команда за печат.

```

Shell x
>>> %Run hello.py

1 2 3 4 5
I am 2 awesome
Python is
amazing
I cant wait.....
to learn more
  
```

Упражнение

Използвайте функцията `print`, за да отпечатате в 3 отделни реда "Goodbye SmartHome people!", като използвате само един оператор `print`.

1. Управление на светодиода

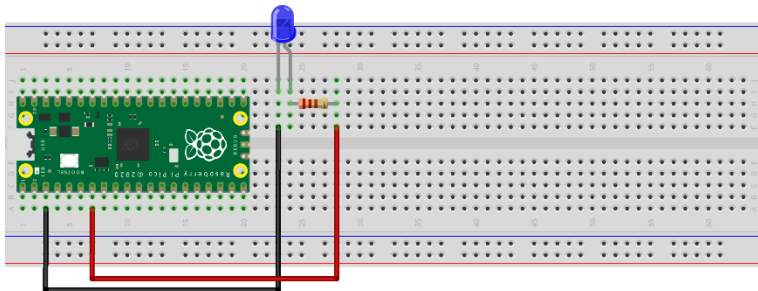
Описание

В този урок ще научите как да свържете и управлявате светодиодна лампа. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла си под името `led.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 2 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x LED (произволен цвят)
- 1 x Micro-USB кабел
- 1 x 220 Ohm резистор

Схема на свързване

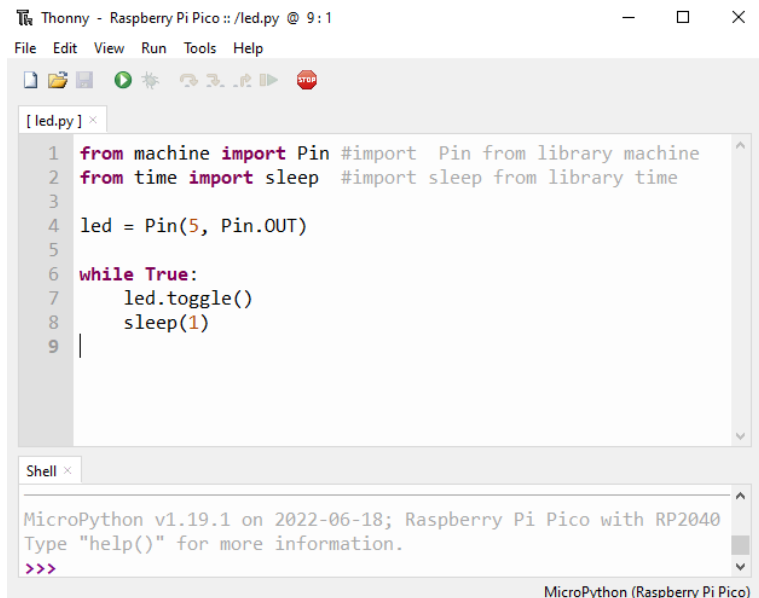


fritzing

- свържете по-дългия край (+) на светодиода с резистор 220 Ohm
- свържете резистора към GPIO5 (червен кабел)
- свържете по-късия край (-) на светодиода към извод GND.

Код

Код на MicroPython за урока:



```

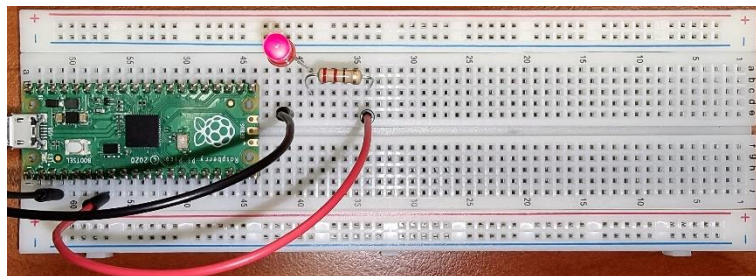
Thonny - Raspberry Pi Pico :: /led.py @ 9:1
File Edit View Run Tools Help

[ led.py ] x
1 from machine import Pin #import Pin from library machine
2 from time import sleep #import sleep from library time
3
4 led = Pin(5, Pin.OUT)
5
6 while True:
7     led.toggle()
8     sleep(1)
9 |

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
MicroPython (Raspberry Pi Pico)
  
```

Образици на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



2. Бутон за натискане

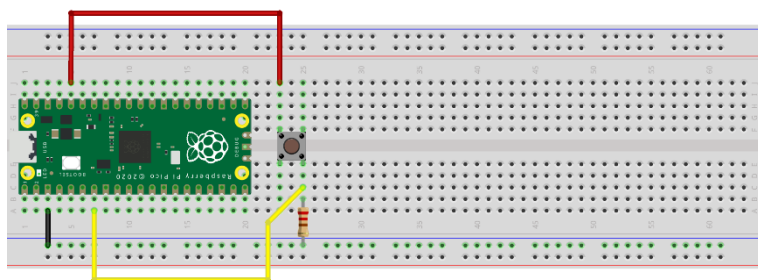
Описание

В този урок ще научите как да свържете и управлявате бутон. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла си под името `button.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x 220 Ohm резистор
- 1 x Micro-USB кабел
- 1 x капачка за бутон
- 1 x бутон за натискане (произволен цвят)

Схема на свързване



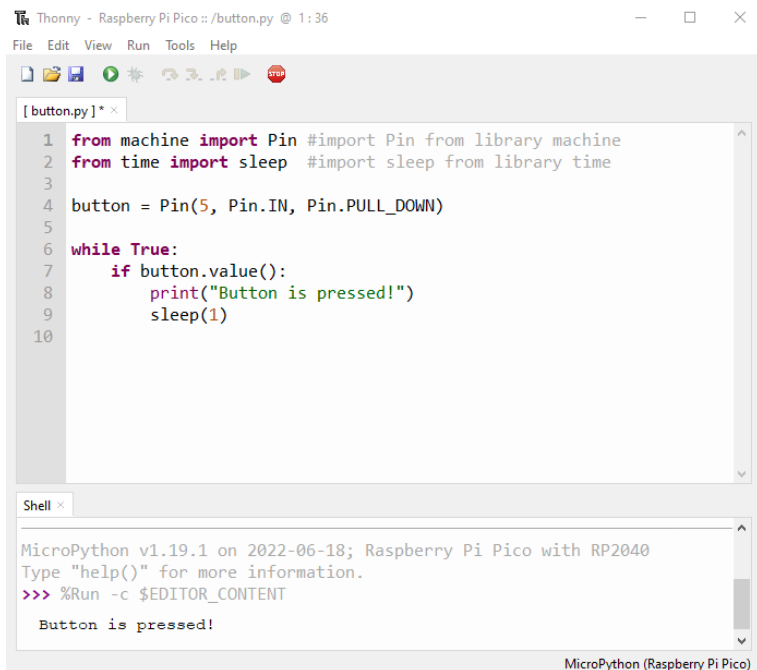
fritzing

- свържете горната лява страна на бутона към щифта 3v3 (червен кабел)
- свържете долния десен бутон към GPIO5 (жълт кабел)

- свържете щифт GND към шината (-) (черен кабел).
- свържете резистора 220 Ом към (-) релсата и долната дясна страна на бутона

Код

Код на MicroPython за урока:

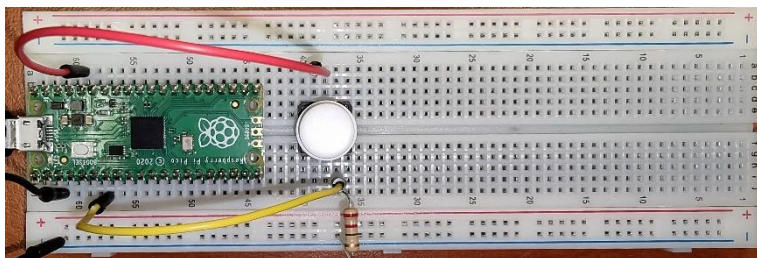


```

Thonny - Raspberry Pi Pico :: /button.py @ 1:36
File Edit View Run Tools Help
[button.py]* x
1 from machine import Pin #import Pin from library machine
2 from time import sleep #import sleep from library time
3
4 button = Pin(5, Pin.IN, Pin.PULL_DOWN)
5
6 while True:
7     if button.value():
8         print("Button is pressed!")
9         sleep(1)
10
Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
Button is pressed!
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



3. Звуков сигнал

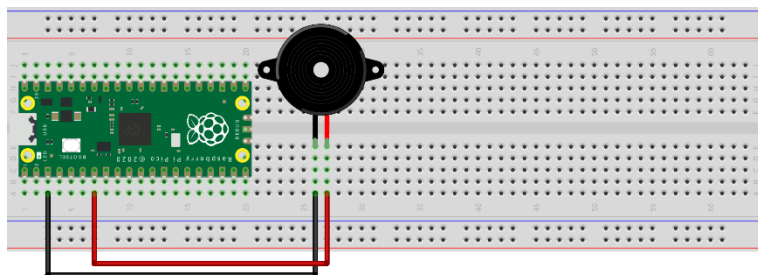
Описание

В този урок ще научите как да свържете и управлявате зумер. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла си под името `buzzer.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 2 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x зумер
- 1 x Micro-USB кабел

Схема на свързване

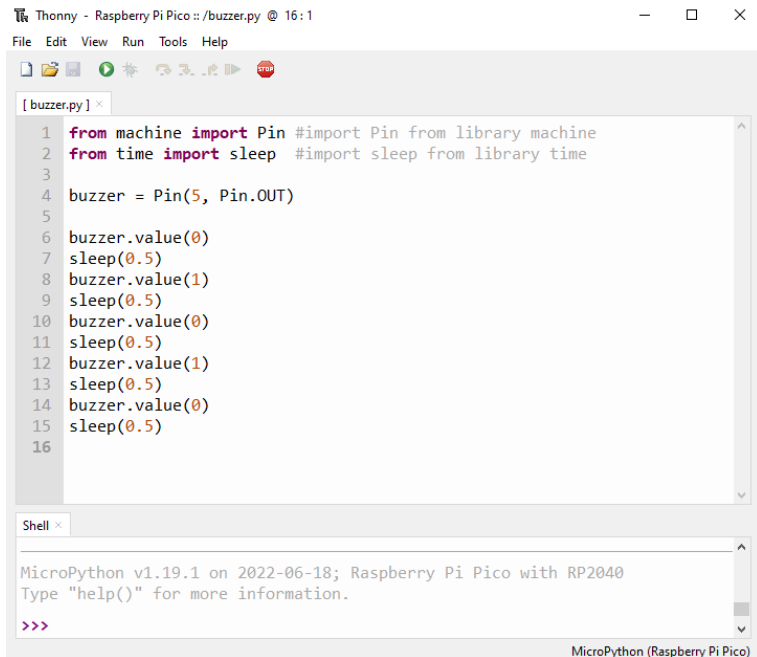


fritzing

- свържете по-дългия край (+) на зумера към щифта GPIO5
- свържете по-късия край (-) на зумера към извод GND.

Код

Код на MicroPython за урока:



```

Thonny - Raspberry Pi Pico ::/buzzer.py @ 16:1
File Edit View Run Tools Help

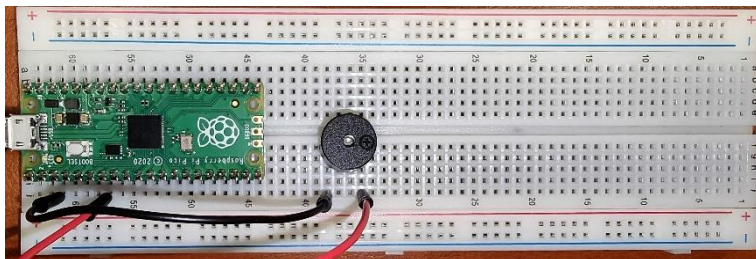
[ buzzer.py ] x
1 from machine import Pin #import Pin from library machine
2 from time import sleep #import sleep from library time
3
4 buzzer = Pin(5, Pin.OUT)
5
6 buzzer.value(0)
7 sleep(0.5)
8 buzzer.value(1)
9 sleep(0.5)
10 buzzer.value(0)
11 sleep(0.5)
12 buzzer.value(1)
13 sleep(0.5)
14 buzzer.value(0)
15 sleep(0.5)
16

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>

MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



4. Потенциометър

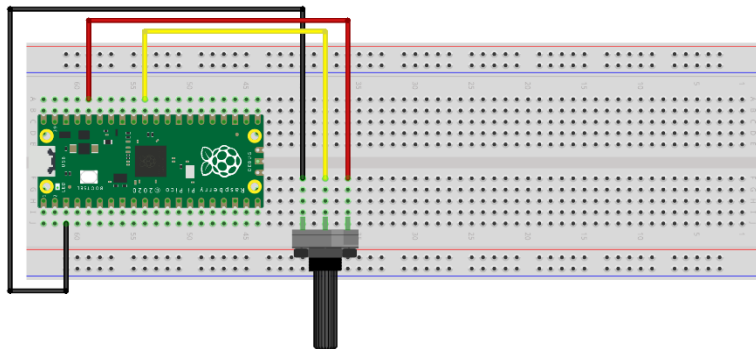
Описание

В този урок ще научите как да свържете и управлявате потенциометър. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла си под името `pot.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x потенциометър
- 1 x Micro-USB кабел

Схема на свързване



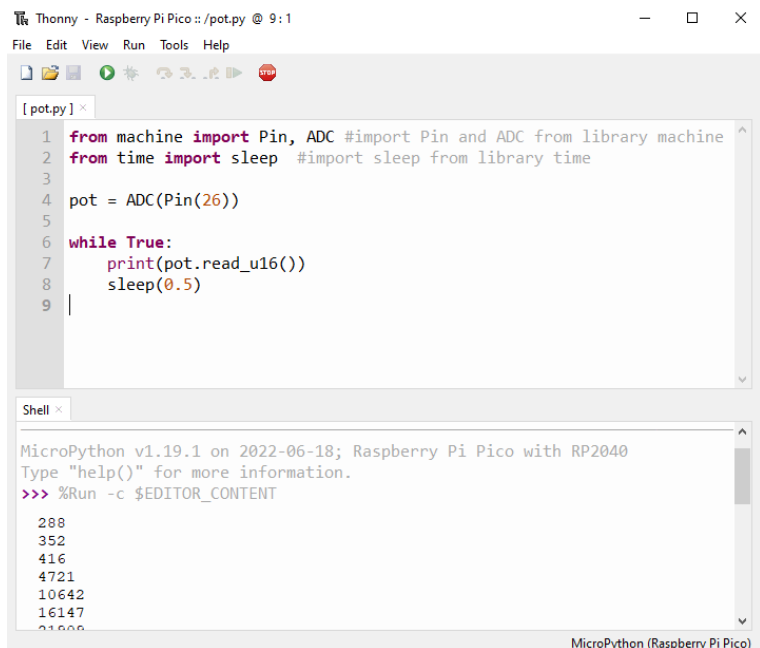
fritzing

- черният кабел трябва да бъде свързан към GND (Pin 3).
- жълтият кабел трябва да бъде свързан към GPIO26 ADC pin

- червеният кабел трябва да бъде свързан към пина за захранване 3V3
- завъртете потенциометъра наляво, така че да е изключен

Код

Код на MicroPython за урока:



```

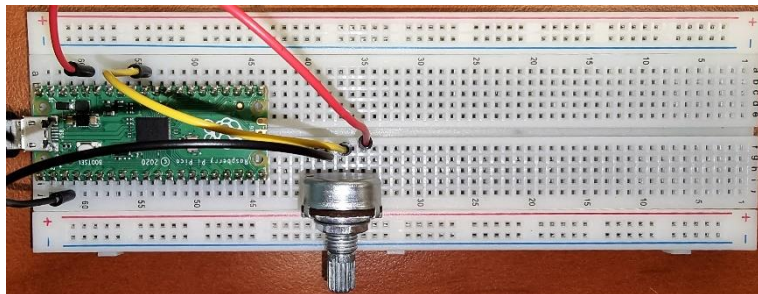
Thonny - Raspberry Pi Pico ::/pot.py @ 9:1
File Edit View Run Tools Help

[ pot.py ] x
1 from machine import Pin, ADC #import Pin and ADC from library machine
2 from time import sleep #import sleep from library time
3
4 pot = ADC(Pin(26))
5
6 while True:
7     print(pot.read_u16())
8     sleep(0.5)
9

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
288
352
416
4721
10642
16147
21000
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



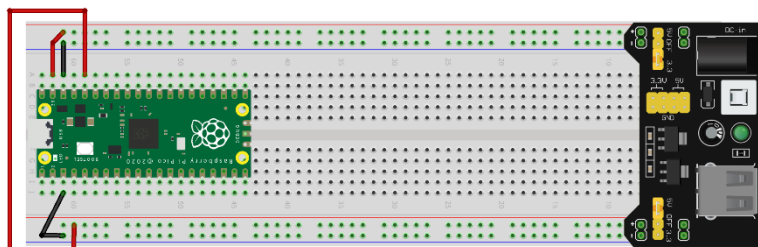
Уроци за напреднали

Преди да преминем към този и следващия раздел, трябва да направим няколко промени в нашата развойна платка. Това включва добавянето на няколко допълнителни компонента и тяхната свързаност.

Компоненти

- 1 x комплект батерии 6-AA
- 1 x захранващ модул MB-102
- 4 x мъжки-мъжки джъмперни проводници

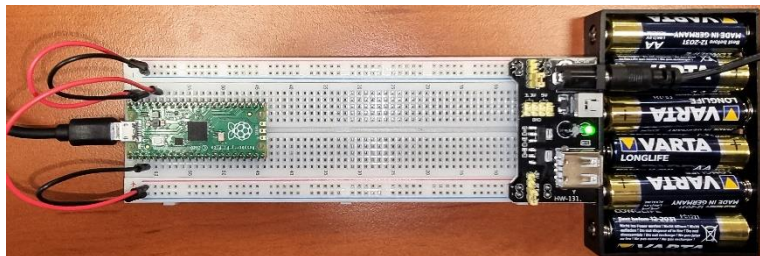
Моля, следвайте схемата за свързване на новите компоненти:



fritzing

- Тази настройка ще се използва за останалите уроци.
- връзки от горната страна: VSYS 5V ((+) червено) и GND ((-) черно)
- долни странични връзки: 3V3 ((+) червено) и GND ((-) черно)

Образци на изображения



5. LED светлини за движение модул

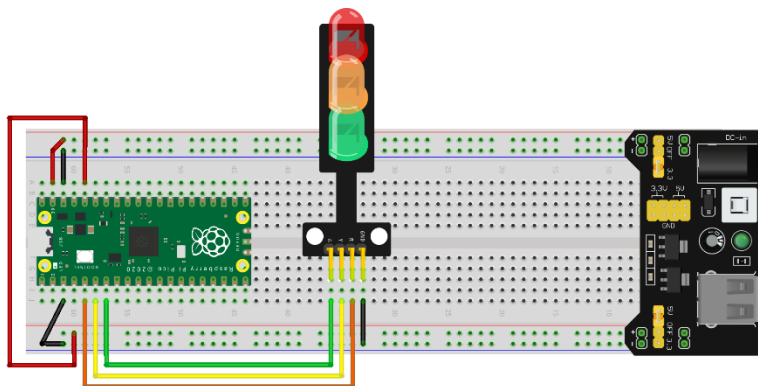
Описание

В този урок ще научите как да свържете и управлявате модула за светодиодни светофари. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла под името `traffic.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 4 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x модул за светодиодни светлини
- 1 x Micro-USB кабел

Схема на свързване

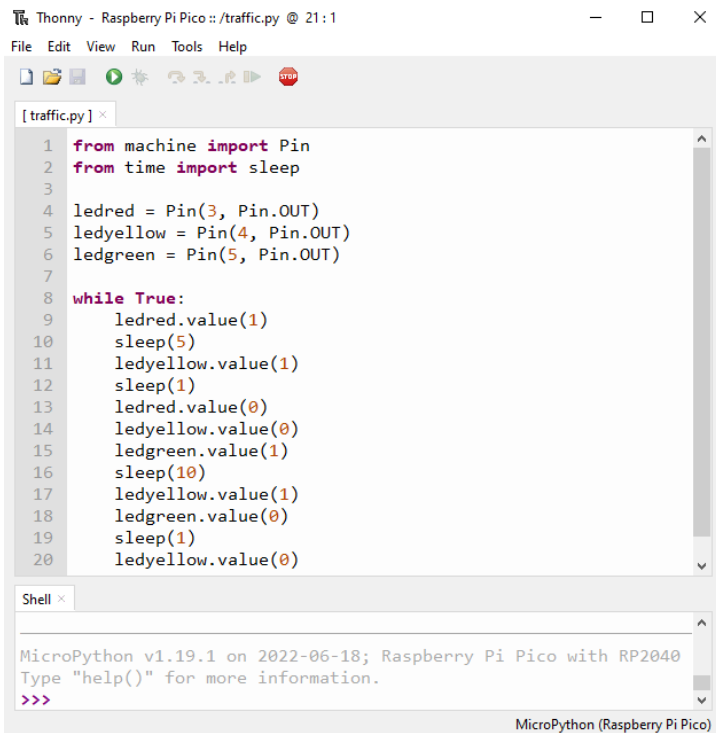


fritzing

- оранжевият кабел (R) е свързан към щифта GPIO3
- жълтият кабел (Y) е свързан към щифта GPIO4
- зеленият кабел (G) е свързан към щифта GPIO5
- черният кабел (GND) е свързан към GND шината ((-) черно)

Код

Код на MicroPython за урока:



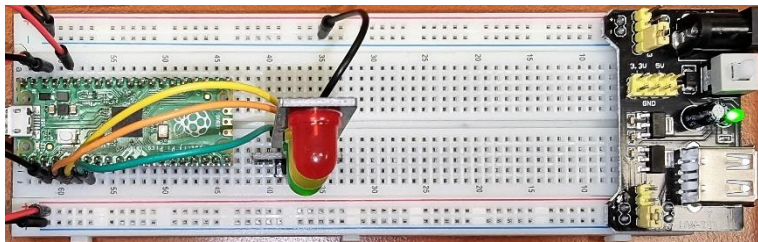
```

Thonny - Raspberry Pi Pico :: /traffic.py @ 21: 1
File Edit View Run Tools Help
[ traffic.py ] x
1 from machine import Pin
2 from time import sleep
3
4 ledred = Pin(3, Pin.OUT)
5 ledyellow = Pin(4, Pin.OUT)
6 ledgreen = Pin(5, Pin.OUT)
7
8 while True:
9     ledred.value(1)
10    sleep(5)
11    ledyellow.value(1)
12    sleep(1)
13    ledred.value(0)
14    ledyellow.value(0)
15    ledgreen.value(1)
16    sleep(10)
17    ledyellow.value(1)
18    ledgreen.value(0)
19    sleep(1)
20    ledyellow.value(0)

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



6. Фоторезистор LDR

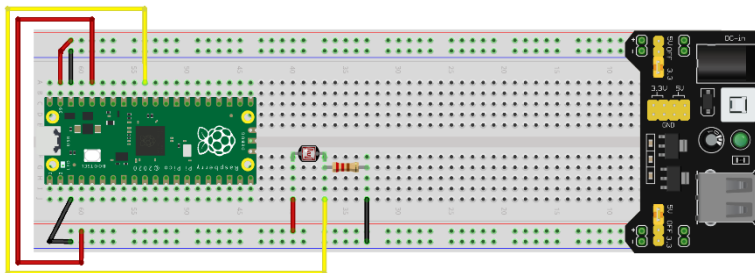
Описание

В този урок ще научите как да свържете и управлявате LDR фоторезистор. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла под името `ldr.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x LDR фоторезистор
- 1 x Micro-USB кабел
- 1 x 220 Ohm резистор

Схема на свързване

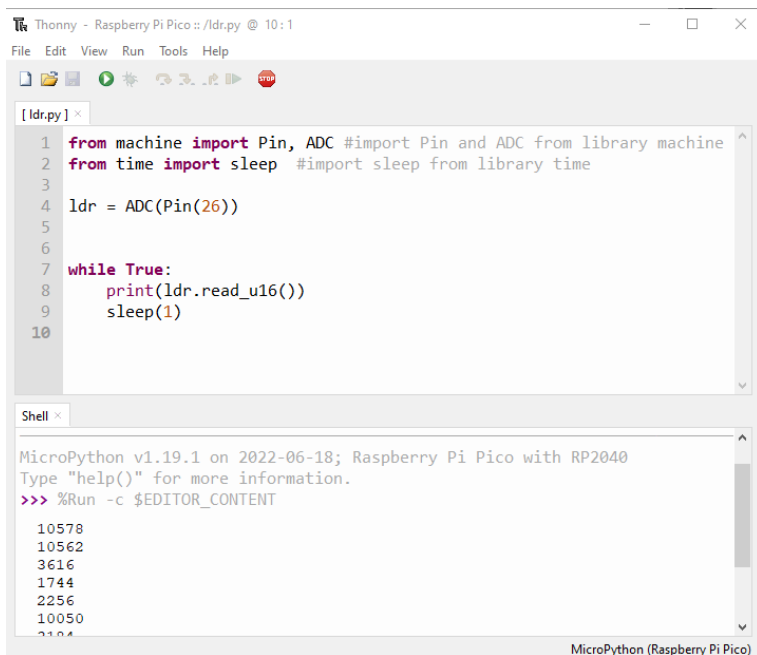


fritzing

- лявата страна на LDR е свързана към 3V шина ((+) червено)
- дясната страна на LDR е свързана с резистор 220 ома и с GPIO26 ADC (жълт кабел).
- дясната страна на резистора е свързана към GND шината ((-) черно)

Код

Код на MicroPython за урока:



```

Thonny - Raspberry Pi Pico :: /ldr.py @ 10: 1
File Edit View Run Tools Help

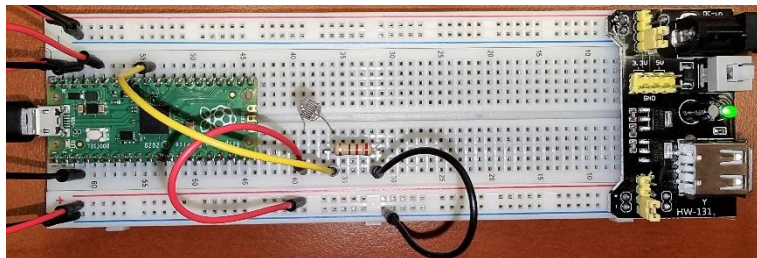
[ ldr.py ] x
1 from machine import Pin, ADC #import Pin and ADC from library machine
2 from time import sleep #import sleep from library time
3
4 ldr = ADC(Pin(26))
5
6
7 while True:
8     print(ldr.read_u16())
9     sleep(1)
10

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
10578
10562
3616
1744
2256
10050
2104

MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



7. Двигател за постоянен ток (малък вентилатор)

Описание

В този урок ще научите как да свържете и управлявате малък постояннотоков (DC) двигател. Отворете Thonny Python, след това отидете на File → Save as... (Файл Запази като...), изберете Raspberry Pi Pico и запазете файла си под името `fan.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x DC мотор (малък вентилатор)
- 1 x Micro-USB кабел
- 1 x транзистор TIP-120
- 1 x диод

Схема на свързване

Thonny - Raspberry Pi Pico :: /fan.py @ 12: 1

File Edit View Run Tools Help

```

[ fan.py ] x
1 from machine import Pin
2 from time import sleep
3
4 fan = Pin(5, Pin.OUT)
5 fan.value(0)
6
7 while True:
8     fan.value(1)
9     sleep(5)
10    fan.value(0)
11    sleep(5)
12
  
```

Shell x

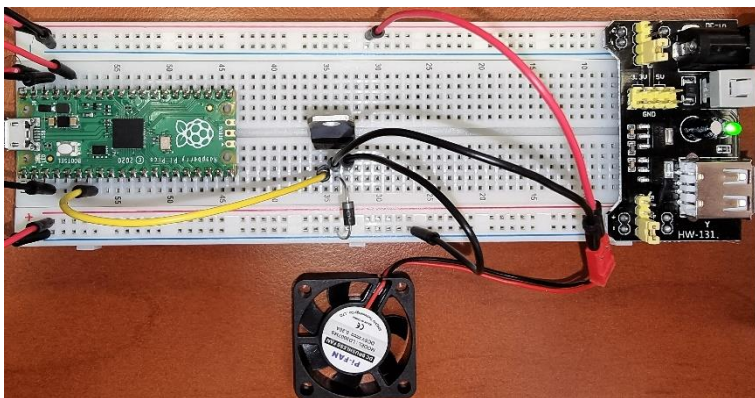
```

MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
  
```

MicroPython (Raspberry Pi Pico)

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



8. Сервомотор SG-90

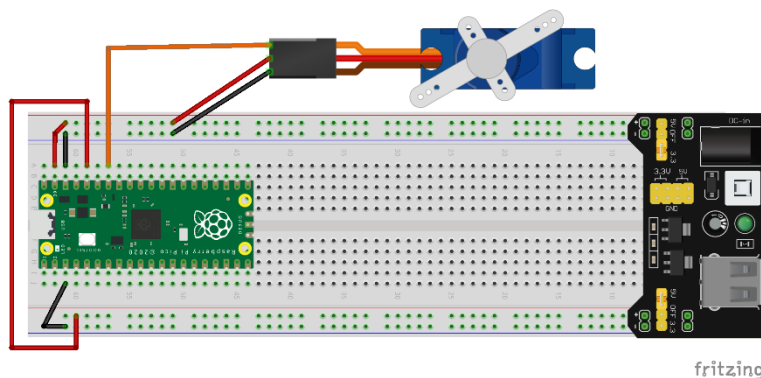
Описание

В този урок ще научите как да свържете и управлявате сервомотор. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла си под името `servo.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x сервомотор SG-90
- 1 x Micro-USB кабел

Схема на свързване

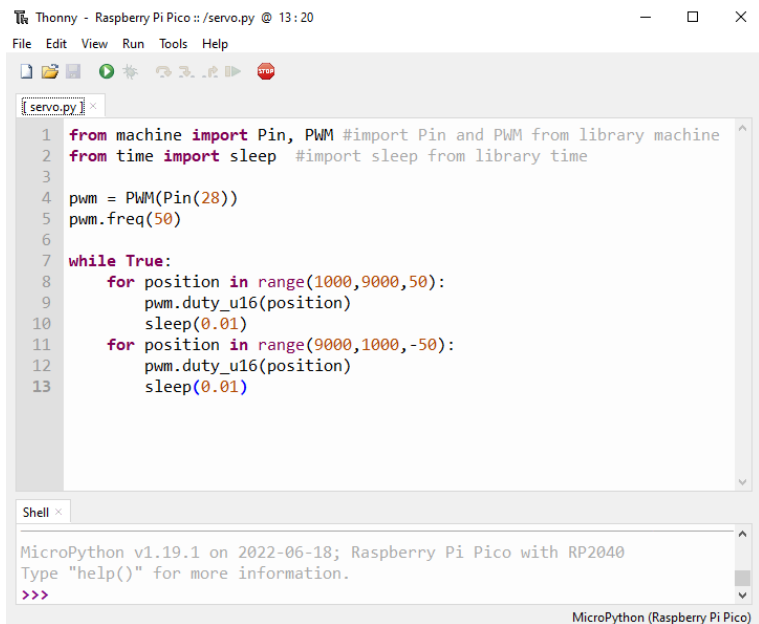


- червеният кабел е свързан към шината 5V (+)
- черният/кафявият кабел е свързан към шината GND (-)

- оранжевият кабел е свързан към GPIO28 ADC pin

Код

Код на MicroPython за урока:



```

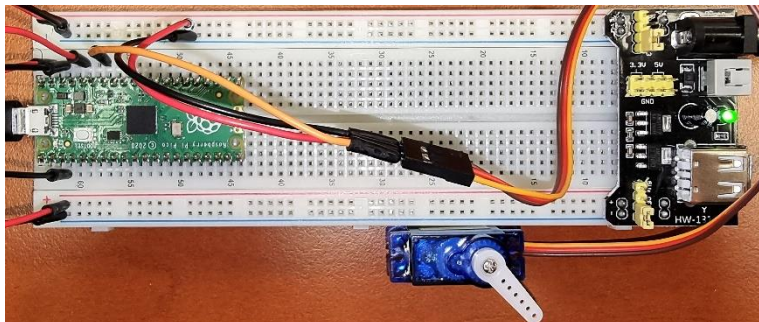
Thonny - Raspberry Pi Pico ::/servo.py @ 13:20
File Edit View Run Tools Help

1 from machine import Pin, PWM #import Pin and PWM from library machine
2 from time import sleep #import sleep from library time
3
4 pwm = PWM(Pin(28))
5 pwm.freq(50)
6
7 while True:
8     for position in range(1000,9000,50):
9         pwm.duty_u16(position)
10        sleep(0.01)
11    for position in range(9000,1000,-50):
12        pwm.duty_u16(position)
13        sleep(0.01)

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



9. OLED I2C SSD1306 дисплей

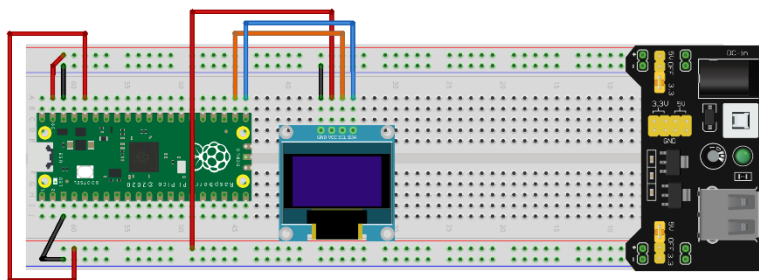
Описание

В този урок ще научите как да свържете и управлявате I2C ICC OLED дисплея. Отворете Thonny Python, след това отидете на File→ Save as..., изберете Raspberry Pi Pico и запазете файла под името `oled.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 4 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x OLED I2C SSD1306 дисплей
- 1 x Micro-USB кабел

Схема на свързване



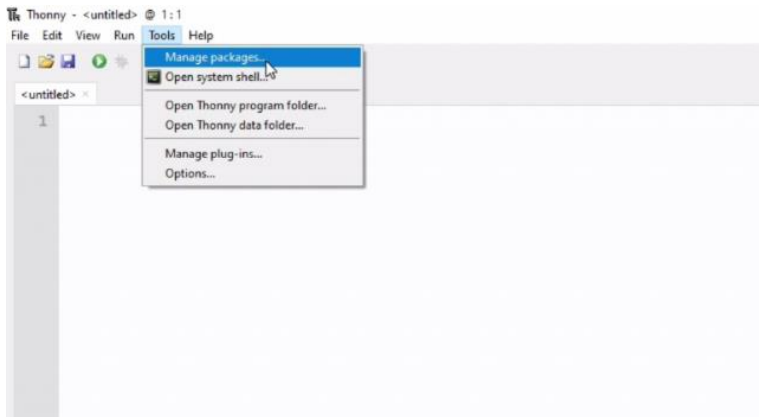
fritzing

- червеният кабел е свързан към шината 3v3 (+)
- черният кабел е свързан към шината GND (-)
- оранжевият кабел е свързан към GPIO17 I2C0 SCL pin
- синият кабел е свързан към GPIO26 I2C0 SDA pin

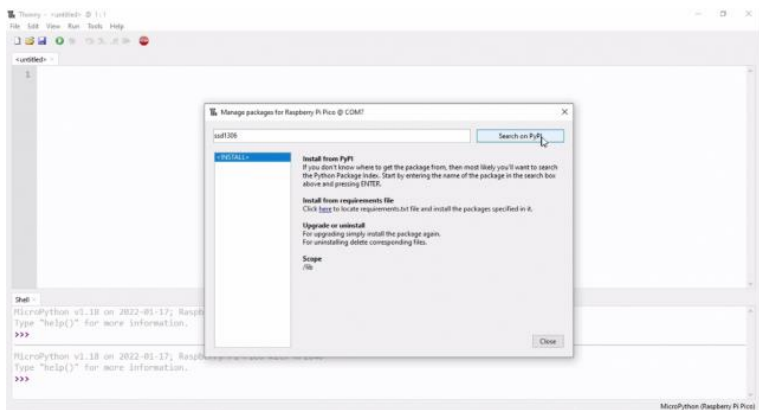
Код

Преди да започнем програмирането на OLED дисплея, първо трябва да добавим пакета SSD1306 към нашия RPi PiCo. За да направите това, моля, следвайте следващите стъпки:

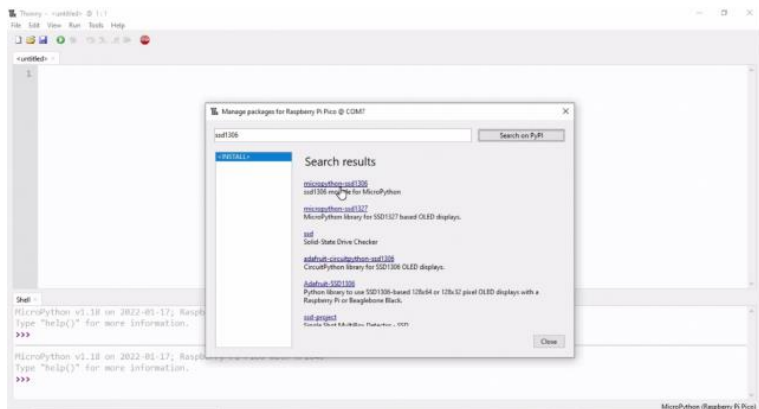
1. Отворете Thonny и отидете на **Tools** → **Manage packages...**



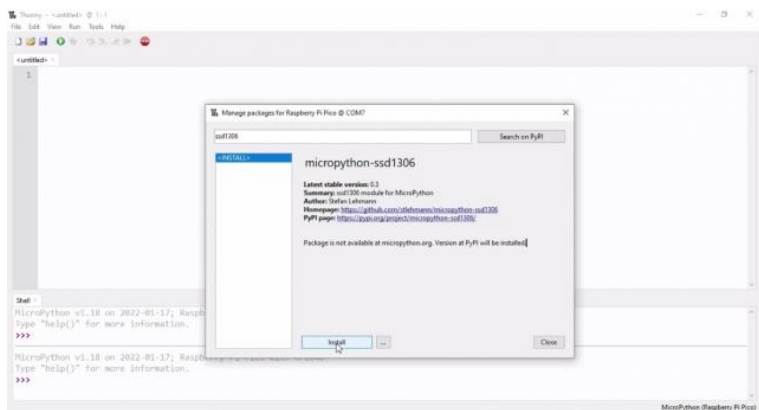
2. В прозореца за управление на пакети въведете **SSD1306** и щракнете върху *Търсене*.



3. След като търсенето приключи, кликнете върху *micropython-ssd1306*.



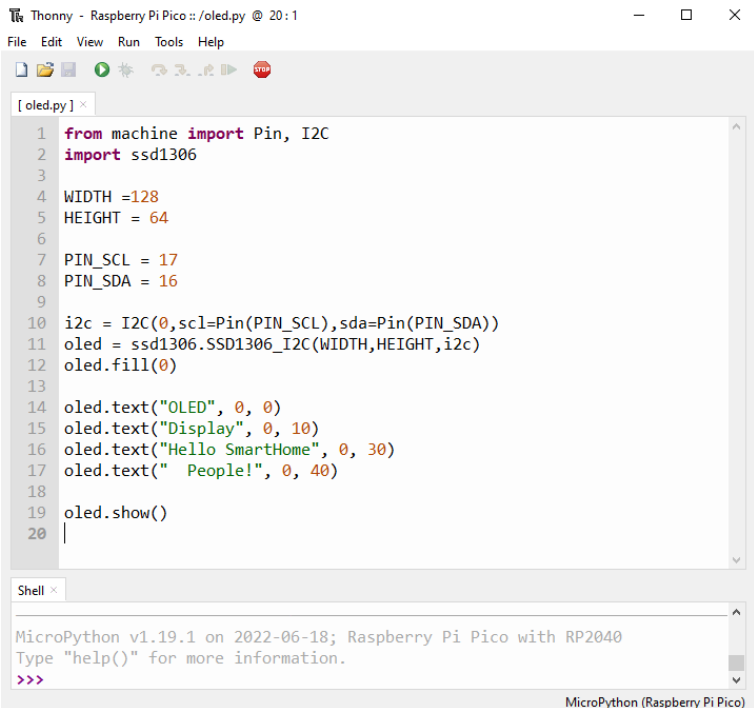
4. В следващия прозорец щракнете върху *Инсталиране*.



5. Изчакайте инсталирането на пакета, след което щракнете върху Затвори.

Сега сме готови да пристъпим към програмиране на OLED дисплея.

Код на MicroPython за урока:



```

1 from machine import Pin, I2C
2 import ssd1306
3
4 WIDTH =128
5 HEIGHT = 64
6
7 PIN_SCL = 17
8 PIN_SDA = 16
9
10 i2c = I2C(0,scl=Pin(PIN_SCL),sda=Pin(PIN_SDA))
11 oled = ssd1306.SSD1306_I2C(WIDTH,HEIGHT,i2c)
12 oled.fill(0)
13
14 oled.text("OLED", 0, 0)
15 oled.text("Display", 0, 10)
16 oled.text("Hello SmartHome", 0, 30)
17 oled.text(" People!", 0, 40)
18
19 oled.show()
20 |
  
```

Shell ×

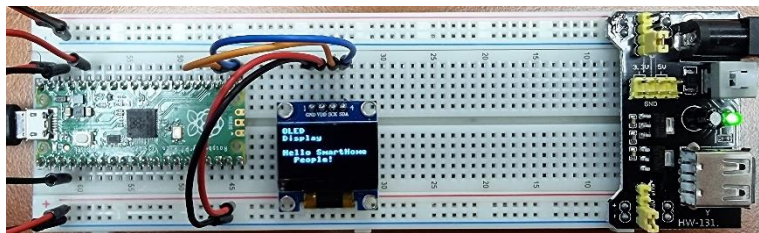
```

MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
  
```

MicroPython (Raspberry Pi Pico)

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



10. RFID четец RC522

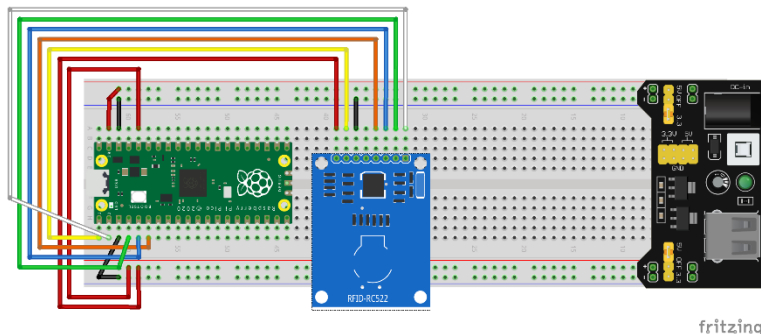
Описание

В този урок ще научите как да свържете и управлявате модул за четене на RFID. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла под името `rfid.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 7 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x RFID модул RC522
- 1 x Micro-USB кабел
- 1 x RFID етикет

Схема на свързване



- 3v3 (червен кабел) е свързан към шината 3v3 (+)
- GND (черен кабел) е свързан към GND шина (-)
- RST (жълт кабел) е свързан към щифт GPIO0
- SDA (белият кабел) е свързан към щифта GPIO1

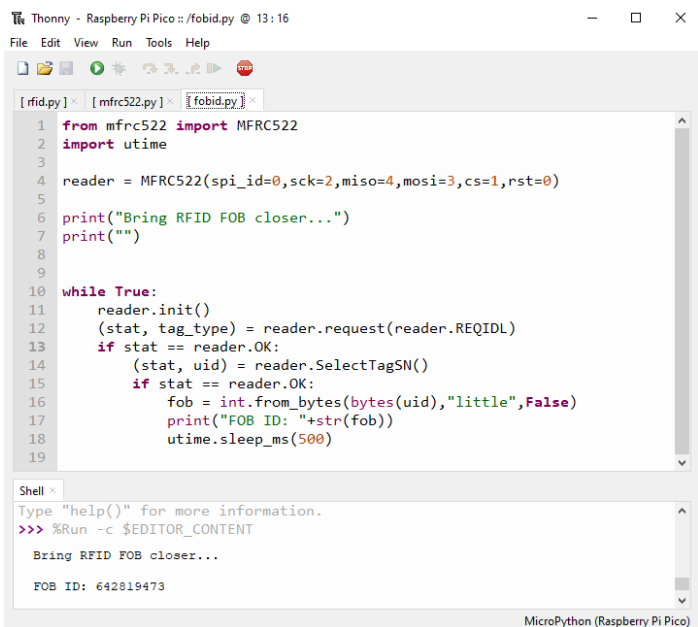
fritzing

- SCK (зеленият кабел) е свързан към щифта GPIO2
- MOSI (син кабел) е свързан към щифт GPIO3
- MISO (оранжев кабел) е свързан към щифт GPIO4

Код

Подобно на OLED дисплея, имаме нужда от допълнителна библиотека, която да управлява RFID модула, а именно библиотеката MFRC522. Можем да я изтеглим от <https://github.com/pimylifeup/MFRC522-python/blob/master/mfrc522/MFRC522.py>. Изтеглете файла и го отворете в Thonny Python. След това щракнете върху File (Файл) → Save as... (Запази като...), изберете Raspberry Pi PiCo и запазете файла под името `mfrc522.py`.

След това трябва да идентифицирате идентификатора на флашката, за да може RFID четецът да работи. За да направите това, трябва да създадете програма, която да чете RFID Fob и да дава неговия идентификатор. Вижте програмата по-долу:



```

1 from mfrc522 import MFRC522
2 import utime
3
4 reader = MFRC522(spi_id=0,sck=2,miso=4,mosi=3,cs=1,rst=0)
5
6 print("Bring RFID FOB closer...")
7 print("")
8
9
10 while True:
11     reader.init()
12     (stat, tag_type) = reader.request(reader.REQIDL)
13     if stat == reader.OK:
14         (stat, uid) = reader.SelectTagSN()
15         if stat == reader.OK:
16             fob = int.from_bytes(bytes(uid),"little",False)
17             print("FOB ID: "+str(fob))
18             utime.sleep_ms(500)
19
  
```

```

Shell x
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
Bring RFID FOB closer...
FOB ID: 642819473
  
```

MicroPython (Raspberry Pi Pico)

Кликнете върху Възпроизвеждане и сканирайте флашката. Това ще ви даде неговия идентификатор. Сега, обратно към програмата `rfid.py`, трябва да промените числото в ред 20, така че да съвпада с идентификатора на вашия fob. След това щракнете върху Save (Запази) и стартирайте програмата си. Код на MicroPython за урока:

```

Thonny - Raspberry Pi Pico :: /rfid.py @ 11:1
File Edit View Run Tools Help

[ rfid.py ] [ mfr522.py ] [ fobid.py ]

1 from machine import Pin
2 from mfr522 import MFRC522
3 import utime
4 import time
5
6 reader = MFRC522(spi_id=0,sck=2,miso=4,mosi=3,cs=1,rst=0)
7
8 print("Bring RFID FOB Closer...")
9 print("")
10
11 |
12 while True:
13     reader.init()
14     (stat, tag_type) = reader.request(reader.REQIDL)
15     if stat == reader.OK:
16         (stat, uid) = reader.SelectTagSN()
17         if stat == reader.OK:
18             fob = int.from_bytes(bytes(uid), "little", False)
19
20             if fob == 642819473:
21                 print("Fob ID: "+ str(fob))
22                 print("Fob is accepted")
23                 time.sleep(1)
24             else:
25                 print("Fob is not accepted")

Shell x
type - help() for more information.
>>> %Run -c $EDITOR_CONTENT
Bring RFID TAG Closer...
Fob ID: 642819473
Fob is accepted

MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



Уроци със сензори

11. Сензор за дъждовни капки

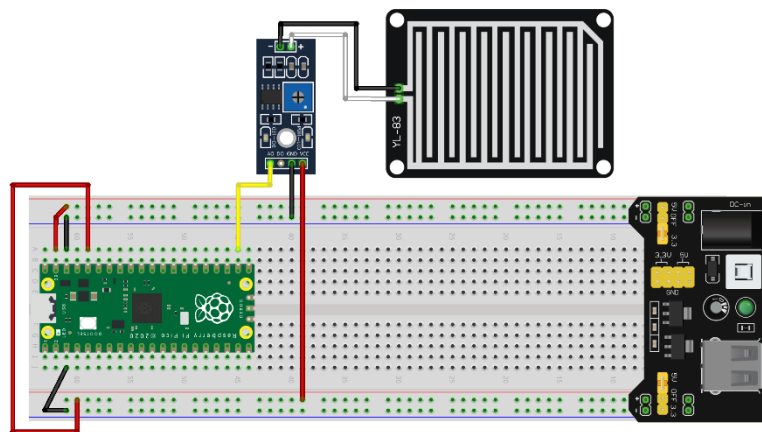
Описание

В този урок ще научите как да свържете и управлявате сензора за дъждовни капки. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла под името `raindrop.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x Сензор за дъждовни капки
- 1 x Micro-USB кабел

Схема на свързване



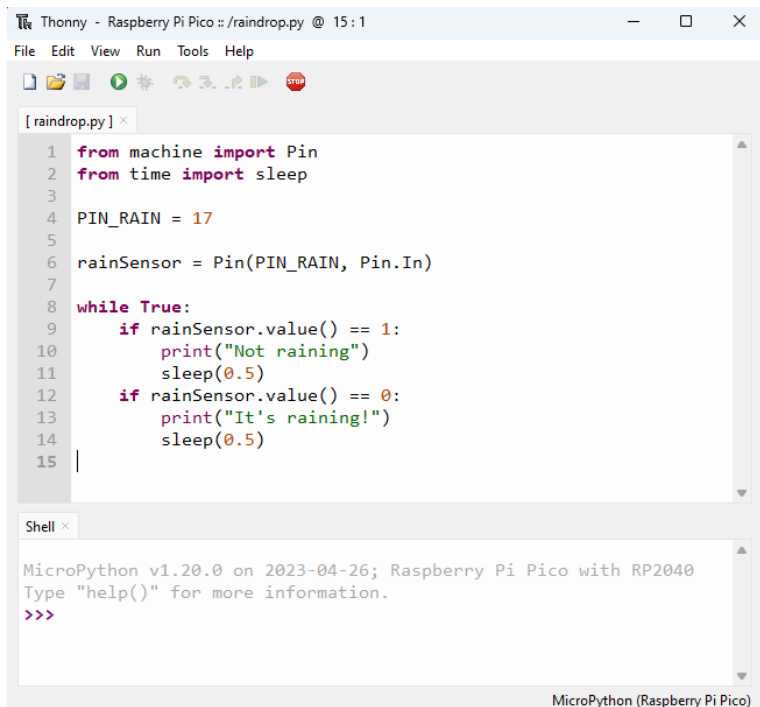
fritzing

- 3v3V (червен кабел) е свързан към шината 3v3V (+)
- GND (черен кабел) е свързан към GND шина (-)

- DO (оранжев кабел) е свързан към щифт GPIO17

Код

Код на MicroPython за урока:

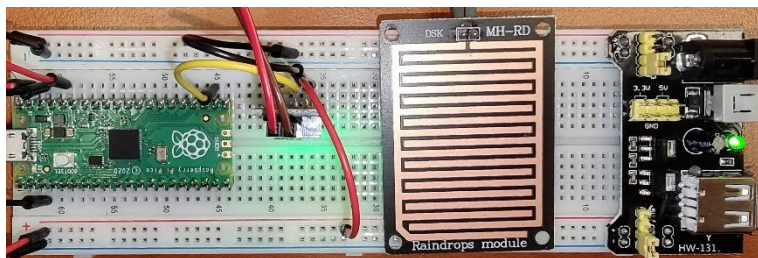


```

Thonny - Raspberry Pi Pico :: /raindrop.py @ 15:1
File Edit View Run Tools Help
[ raindrop.py ] x
1 from machine import Pin
2 from time import sleep
3
4 PIN_RAIN = 17
5
6 rainSensor = Pin(PIN_RAIN, Pin.In)
7
8 while True:
9     if rainSensor.value() == 1:
10        print("Not raining")
11        sleep(0.5)
12    if rainSensor.value() == 0:
13        print("It's raining!")
14        sleep(0.5)
15
Shell x
MicroPython v1.20.0 on 2023-04-26; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



12. Ултразвуков сензор HC-SR04

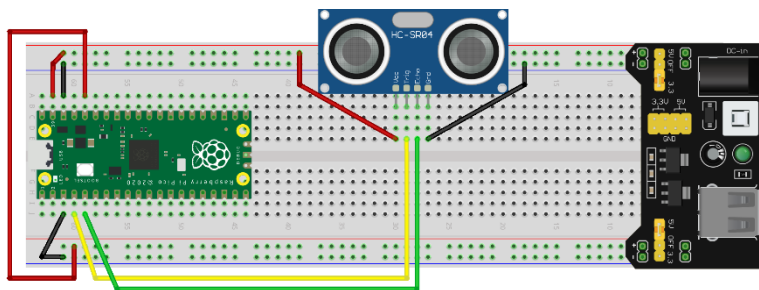
Описание

В този урок ще научите как да свържете и управлявате ултразвуковия сензор HC-SR04. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла под името `ultrasonic.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 4 x мъжки-мъжки джъмперни проводници
- 1 x Пълноразмерна платка
- 1 x HC-SR04 Ултразвуков сензор
- 1 x Micro-USB кабел

Схема на свързване



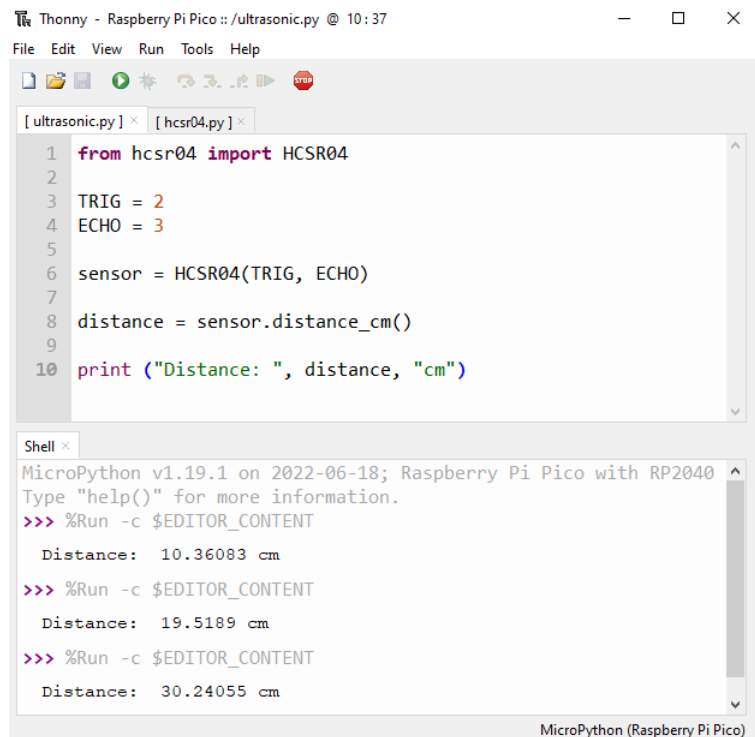
fritzing

- VCC (червен кабел) е свързан към 5V шина (+)
- GND (черен кабел) е свързан към GND шина (-)
- TRIG (оранжев кабел) е свързан към щифта GPIO2
- ECHO (зеленият кабел) е свързан към щифта GPIO3

Код

За да използваме ултразвуковия сензор HC-SR04, можем да разработим собствена програма или да използваме някоя от наличните библиотеки, например в [Github](#). Ако решите да изтеглите библиотеката *hcsr04.py*, трябва да я запишете във вашия Pico под същото име.

Код на MicroPython, използващ съществуваща библиотека:



```

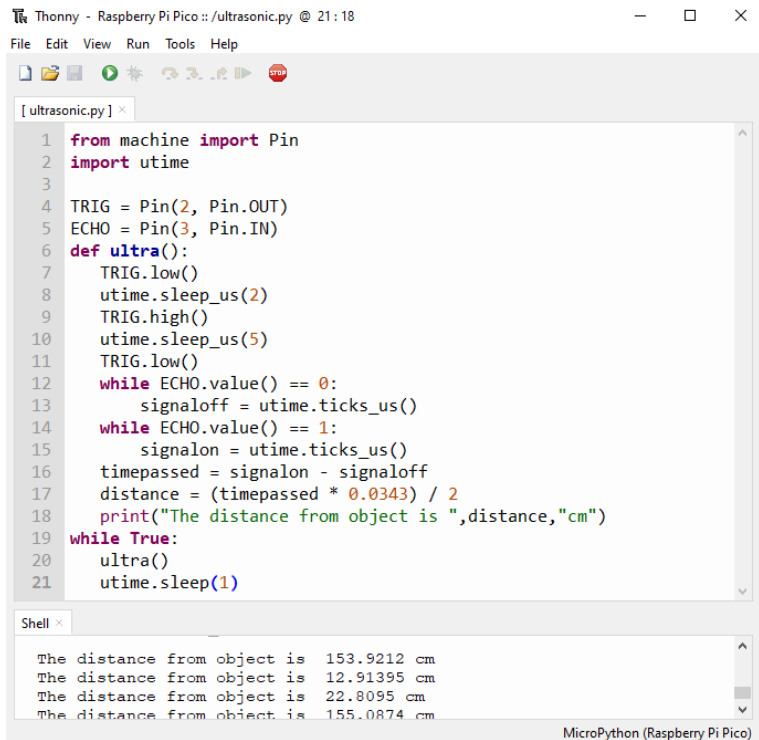
Thonny - Raspberry Pi Pico :: /ultrasonic.py @ 10:37
File Edit View Run Tools Help

[ ultrasonic.py ] x [ hcsr04.py ] x

1 from hcsr04 import HCSR04
2
3 TRIG = 2
4 ECHO = 3
5
6 sensor = HCSR04(TRIG, ECHO)
7
8 distance = sensor.distance_cm()
9
10 print ("Distance: ", distance, "cm")

Shell x
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
    Distance: 10.36083 cm
>>> %Run -c $EDITOR_CONTENT
    Distance: 19.5189 cm
>>> %Run -c $EDITOR_CONTENT
    Distance: 30.24055 cm
MicroPython (Raspberry Pi Pico)
  
```

Код на MicroPython без съществуваща библиотека:



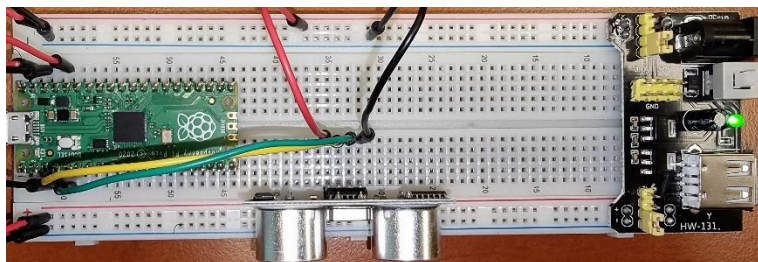
```

Thonny - Raspberry Pi Pico :: /ultrasonic.py @ 21:18
File Edit View Run Tools Help
[ultrasonic.py] x
1 from machine import Pin
2 import utime
3
4 TRIG = Pin(2, Pin.OUT)
5 ECHO = Pin(3, Pin.IN)
6 def ultra():
7     TRIG.low()
8     utime.sleep_us(2)
9     TRIG.high()
10    utime.sleep_us(5)
11    TRIG.low()
12    while ECHO.value() == 0:
13        signaloff = utime.ticks_us()
14    while ECHO.value() == 1:
15        signalon = utime.ticks_us()
16    timepassed = signalon - signaloff
17    distance = (timepassed * 0.0343) / 2
18    print("The distance from object is ",distance,"cm")
19 while True:
20    ultra()
21    utime.sleep(1)

Shell x
The distance from object is 153.9212 cm
The distance from object is 12.91395 cm
The distance from object is 22.8095 cm
The distance from object is 155.0874 cm
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



13. PIR сензор за движение

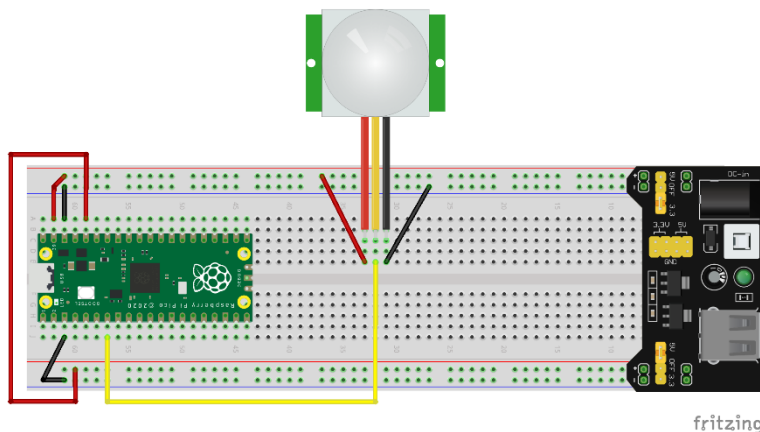
Описание

В този урок ще научите как да свържете и управлявате PIR сензора за движение. Отворете Thonny Python, след това отидете на File → Save as..., изберете Raspberry Pi Pico и запишете файла под името `motion.py`. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico
- 3 x мъжки-женски свързвачи проводници
- 1 x Пълноразмерна платка
- 1 x PIR сензор за движение
- 1 x Micro-USB кабел

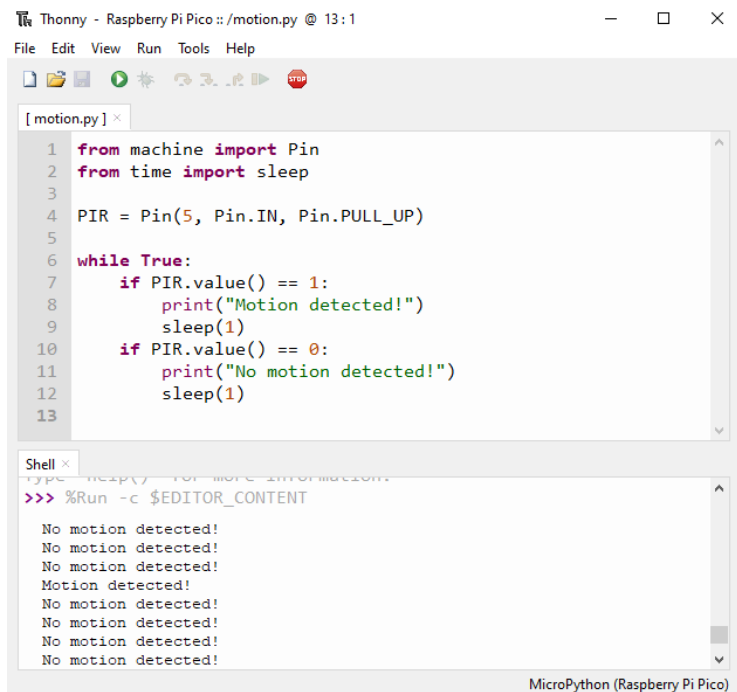
Схема на свързване



- VCC (червен кабел) е свързан към 5V шина (+)
- GND (черен кабел) е свързан към GND шина (-)
- OUT (оранжев кабел) е свързан към щифт GPIO5

Код

Код на MicroPython за урока:

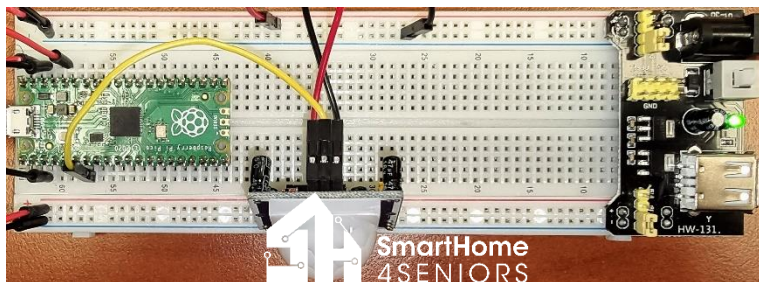


```

Thonny - Raspberry Pi Pico :: /motion.py @ 13:1
File Edit View Run Tools Help
[motion.py] x
1 from machine import Pin
2 from time import sleep
3
4 PIR = Pin(5, Pin.IN, Pin.PULL_UP)
5
6 while True:
7     if PIR.value() == 1:
8         print("Motion detected!")
9         sleep(1)
10    if PIR.value() == 0:
11        print("No motion detected!")
12        sleep(1)
13
Shell x
type --help -- for more information.
>>> %Run -c $EDITOR_CONTENT
No motion detected!
No motion detected!
No motion detected!
Motion detected!
No motion detected!
No motion detected!
No motion detected!
No motion detected!
No motion detected!
MicroPython (Raspberry Pi Pico)
  
```

Образци на изображения

Изображение на начина, по който изглежда урокът при използване на предоставения хардуер:



14. Сензор за пламък

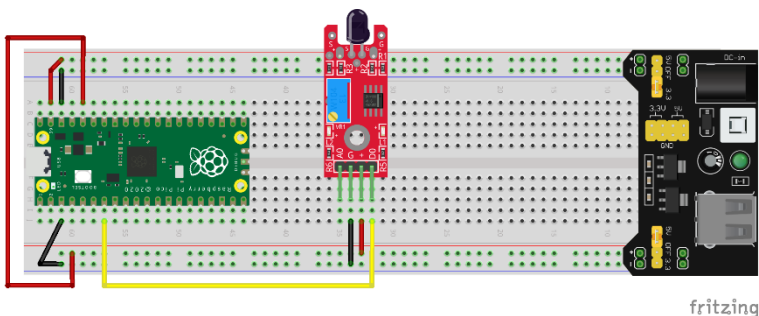
Описание

В този урок ще научите как да свържете и управлявате сензора за пламък KY-026. Отворете Thonny Python, след това отидете на File □ Save as..., изберете Raspberry Pi Pico и запазете файла си под името flame.py. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico - 3 x джъмпера от мъжки до мъжки тип
- 1 x Пълноразмерна платка за хляб - 1 x Датчик за пламък KY-026
- 1 x Micro-USB кабел

Схема на свързване



- VCC (червен кабел) е свързан към 5V шина (+)
- GND (черен кабел) е свързан към шината GND (-)
- DO (зелен кабел) е свързан към щифта GPIO5

Код

Код на MicroPython за урока:

```

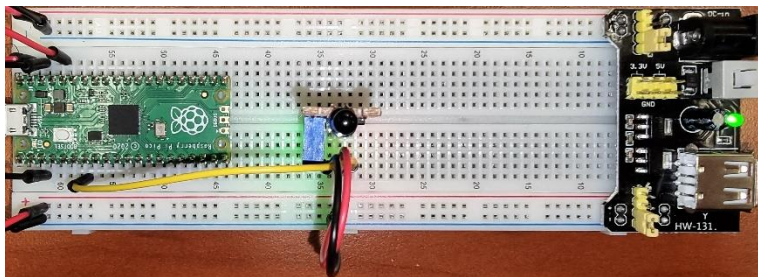
Thonny - Raspberry Pi Pico :: /flame.py @ 11:1
File Edit View Run Tools Help

[flame.py] -
1 from machine import Pin
2 from time import sleep
3
4 flame = Pin(5, Pin.IN)
5 sleep(2)
6
7 while True:
8     if flame.value() == 0:
9         print("Flame Detected")
10        sleep(3)
11

Shell -
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
Flame Detected
Flame Detected
Flame Detected
...
MicroPython (Raspberry Pi Pico)
  
```

Примерна снимка

Изображение на начина, по който изглежда урокът, като се използва предоставеният хардуер:



16. Сензор за откриване на газ MQ-135

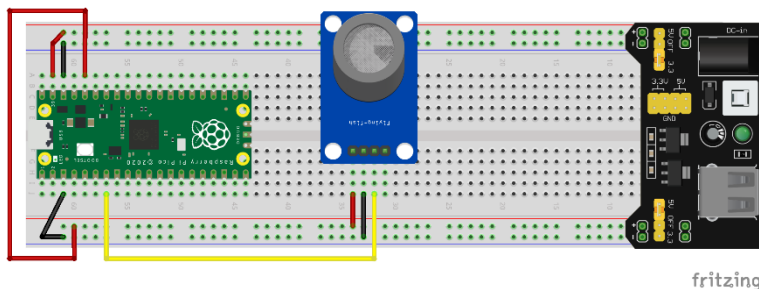
Описание

В този урок ще научите как да свържете и управлявате сензора за откриване на газ MQ-135. Отворете Thonny Python, след което отидете на File □ Save as..., изберете Raspberry Pi Pico и запишете файла си под името gas.py. След това е време да свържете електрониката и да напишете програмата си. Моля, следвайте инструкциите по-долу.

Необходими материали

- 1 x Raspberry Pi Pico - 3 x джъмперни проводници от мъжки до мъжки тип
- 1 x Пълноразмерна платка за хляб - 1 x MQ-135 за откриване на газ
- 1 x Micro-USB кабел

Схема на свързване



- VCC (червеният кабел) е свързан към шината 3v3 (+)
- GND (черен кабел) е свързан към шината GND (-)
- DO/OUT (жълт кабел) е свързан към щифта GPIO5

Код

Код на MicroPython за урока:

```

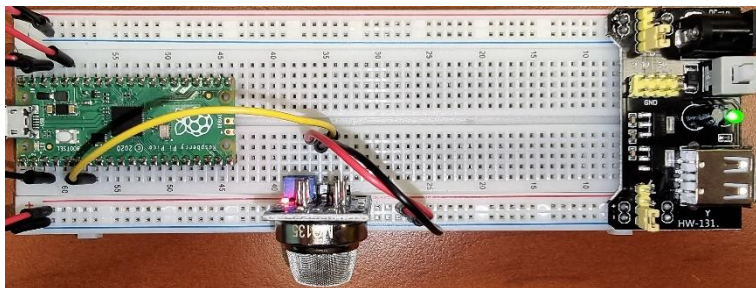
Thonny - Raspberry Pi Pico :: /gas.py © 11:1
File Edit View Run Tools Help

[ gas.py ]
1 from machine import Pin
2 from time import sleep
3
4 gas = Pin(5, Pin.IN)
5 sleep(2)
6
7 while True:
8     if gas.value() == 0:
9         print("Gas Detected")
10        sleep(3)
11

Shell
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> !Run -c $EDITOR_CONTENT
Gas Detected
Gas Detected
Gas Detected
MicroPython (Raspberry Pi Pico)
  
```

Примерна картина

Изображение на начина, по който изглежда урокът, като се използва предоставеният хардуер:



ПРИЛОЖЕНИЕ: Таблица за обобщение на MicroPython

Цифров изход		
Извикване на класа Pin	<code>from machine import Pin</code>	
Инициране на обекта за цифров изход	<code>led = Pin(pin_value, Pin.OUT)</code>	<code>pin_value</code> from 0 to 40
Отворен цифров изход (изход 3,3 V)	<code>led.value(1)</code>	ON
Затваряне на цифров изход (изход 0 V)	<code>led.value(0)</code>	OFF

Цифров вход		
Извикване на класа Pin	<code>from machine import Pin</code>	
Инициране на обекта за цифров изход	<code>button = Pin(pin_value, Pin.IN)</code>	pin_value от 0 до 40
	<code>button = Pin(pin_value), Pin.IN, Pin.PULL_UP)</code>	Активиране на съпротивлението PULL UP
	<code>button = Pin(pin_value), Pin.IN, Pin.PULL_DOWN)</code>	Активиране на съпротивлението PULL DOWN
Четене на входа	<code>value = button.value(1)</code>	Върнатата стойност може да бъде 0, ако изводът е с напрежение 0 V, или 1, ако изводът е с напрежение 3,3 V

Аналогов изход (широчинно-импулсна модулация - ШИМ)		
Извикване на класа PWM	<code>from machine import PWM</code>	
Инициране на аналоговия изход	<code>led = PWM(Pin(pin_value), frequency)</code>	pin_value от 0 до 40
Отчитане на входа	<code>led.duty(duty_cycle)</code>	честота в HZ, от 0 до 78125

Аналогов вход		
Извикване на класа ADC	<code>from machine import ADC</code>	
Инициране на аналоговия вход	<code>pot = ADC(Pin(pin_value))</code>	pin_value can be GPIO26, GPIO27 and GPIO28

Деклариране на напрежението, при което входът ще дава максималната си стойност (в ESP32 обикновено 3,3 V)	<code>pot.atten(ADC.ATTN_11DB)</code>	ADC.ATTN_0DB: full range voltage: 1.2 V ADC.ATTN_2_5DB: full range voltage: 1.5 V ADC.ATTN_6DB: full range voltage: 2.0 V ADC.ATTN_11DB: full range voltage: 3.3 V
Деклариране на обхвата на входната стойност (по подразбиране 12bit)	<code>pot.width(ADC.WIDTH_10BIT)</code>	ADC.WIDTH_9BIT: range 0 to 511 ADC.WIDTH_10BIT: range 0 to 1023 ADC.WIDTH_11BIT: range 0 to 2047 ADC.WIDTH_12BIT: range 0 to 4095
Четене на входа	<code>value = pot.read1()</code>	<code>value</code> is an integer from 0 to the maximum of the range specified by the <code>ADC.WIDTH_#BIT</code> statement (see previous)

Библиотеката на времето

Извикване на класа Sleep	<code>from time import sleep</code>	
Използване на функцията sleep	<code>sleep(sec)</code>	<code>sec</code> is the number of seconds for which the program will be delayed
Извикване на класа time	<code>import time</code>	
Използване на функцията time	<code>current_time = time()</code>	The <code>current_time</code> variable will take a numeric value, equal to the number of seconds since the last reset on the board.

Структура на изявлението if

<pre>if <expr1>: <statement1> elif <expr2>: <statement2></pre>	<expr#>: условието за контрол, което трябва да връща True или False <statement#>: set of commands to be executed when the adjacent condition is satisfied
--	--

<pre>elif <expr3>: <statement3> (...) else: <statementn></pre>	<p><expr#> (e.g. the set <statement2> is executed when <expr2> is satisfied)</p> <p><statementn>: set of instructions executed when none of the <expr#> conditions are satisfied</p>
--	--

Структура на цикъла While

<pre>while <expr>: <statement(s)></pre>	<p><expr>: the control condition that must return True or False</p> <p><statement#>: set of commands to be executed as long as <expr> condition is satisfied</p>
---	--

Структура на цикъла For

<pre>for <var> in <iterable>: <statement(s)></pre>	<p><iterable>: a collection of objects, for example a list containing numbers, alphanumeric, etc.</p> <p><var>: a variable to which the value of the next item in the collection <iterable> is assigned.</p> <p><statement(s)>: a set of instructions executed at each iteration</p>
<pre>for <var> in range(<start>, <end>, <step>): <statement(s)></pre>	<p>range(<start>, <end>, <step>): function that returns a sequence of numbers from <start> to <end>-1, with a <step> difference between two consecutive numbers (<start> and <step> parameters are optional).</p> <p><var>: variable to which the value of the next element of the sequence produced by range is assigned.</p> <p><statement(s)>: a set of instructions executed in each iteration</p>

Различни		
DHT11	<code>import dht</code>	Импортиране на библиотеката DHT
	<code>sensor = dht.DHT11(Pin(pin_number))</code>	Инициране на променливата на сензора с неговия асоцииран pin_number.
	<code>sensor.measure()</code>	Актуализиране на стойностите на сензора
	<code>temp = sensor.temperature()</code>	Запазване на текущата стойност на температурата
	<code>hum = sensor.humidity()</code>	Запазване на текущата стойност на влажността
OLED DISPLAY	<code>from machine import I2C</code>	Импортиране на библиотеката I2C
	<code>import ssd1306</code>	Импортиране на библиотеката ssd1306
	<code>i2c = I2C(-1, scl=Pin(1), sda=Pin(0))</code>	Инициране на променливата i2c на изводите SCL и SDA на Pico
	<code>oled_width = 128</code> <code>oled_height = 64</code> <code>oled = ssd1306.SSD1306_I2C(oled_width, oled_height, i2c)</code>	Инициране на екрана

	<pre>oled.text('Hello, World 1!', 0, 0) oled.text('Hello, World 2!', 0, 10) oled.text('Hello, World 3!', 0, 20)</pre>	Съхраняване на съобщения в буфера на екрана
	<pre>oled.show()</pre>	Показване на съобщенията (необходимо за показване на съобщенията, съхранени в буфера на екрана)
	<pre>display.pixel(3, 4, 1)</pre>	Задаване на пиксела, разположен на позиция (x,y) на екрана, с x=3 и y=4, в състояние 1 (т.е. показване)



Co-funded by
the European Union





SmartHome
4SENIORS

2021-1-DE02-KA220-ADU-000033587

The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Raspberry Pi Pico Pinout

