EXPLORER SET

RB-P-XPLR-SET

TURINYS

1. BENDRA INFORMACIJA

Gerbiamas kliente, dėkojame, kad pasirinkote mūsų gaminį. Toliau pateiksime, ką reikia turėti omenyje pradedant eksploatuoti ir naudojant gaminį.

Jei naudojimo metu susidurtumėte su netikėtomis problemomis, nedvejodami kreipkitės į mus.

2. PRIETAISO APŽVALGA IR KAIŠČIŲ PRISKYRIMAS

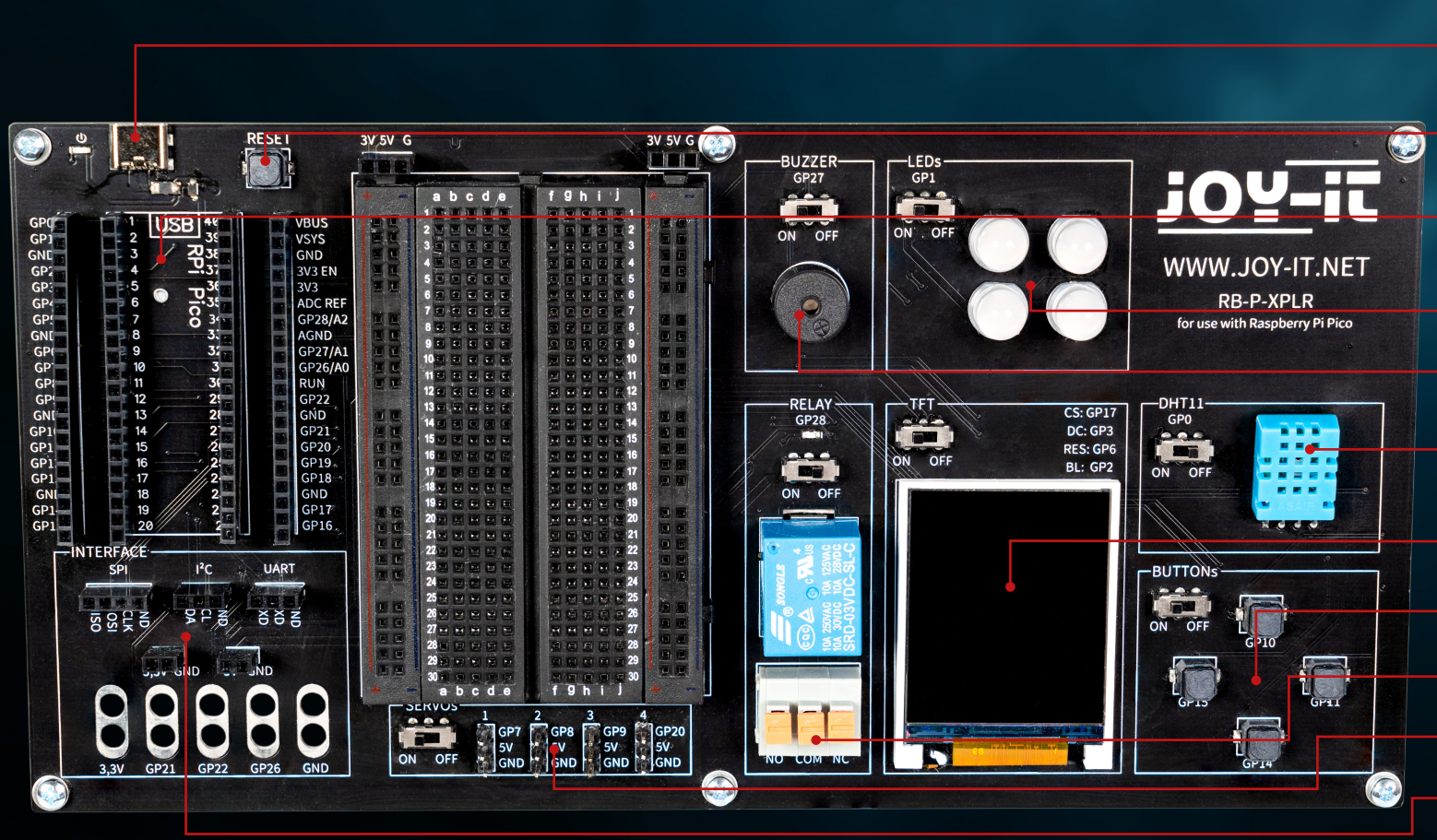
Mūsų "Explorer" plokštė - tai paprastas ir efektyvus būdas kurti "Raspberry Pi Pico" projektus.

Kadangi svarbiausi komponentai jau integruoti, taupysite laiką ir pastangas jungdami laidus. Explorer plokštė turi daugybę sąsajos jungčių, todėl galite prijungti savo projektus prie įvairių modulių ir įrenginių. Naudodami integruotą duonos plokštę galite greitai kurti ir įgyvendinti savo projektus.

Dėl galimybės visus modulius įjungti arba išjungti atskirai galite bet kada naudoti savo kaiščius, kurie taip pat atskirai išvedžioti į išorę, kitiems projektams arba eksperimentuoti su integruota duonos plokšte.

Visus integruotus komponentus galima išjungti atitinkamu jungikliu, jei jie nereikalingi. Tai reiškia, kad prireikus susijusius kaiščius galima naudoti ir kitiems komponentams.

Kairėje ir dešinėje "Raspberry Pi Pico" pusėje visi kaiščiai yra papildomai suprojektuoti. Komponentus čia galima prijungti tiesiogiai arba papildomais laidais nukreipti į integruotą plokštę.



5 V USB-C
Maitinimo šaltinis

Atstatymo mygtukas

Lizdas Raspberry Pi Pico

4x RGB LED

Signalizatorius

DHT11 Temperatūros
ir drėgmės jutiklis

1,8 colių TFT ekranas

4x mygtukai

Relė

4x servo sąsaja

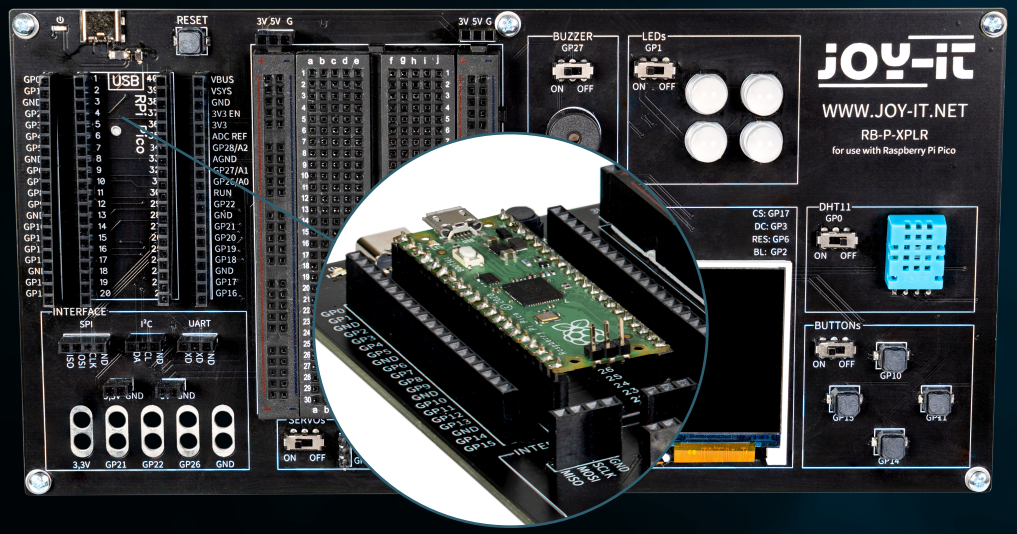
Sąsajos jungtys

Atkreipkite dėmesį, kad USB-C jungtis visada turi būti prijungta. Maitinimas per "Raspberry Pi Pico" mikro USB jungtį nega-
limas.

KAIŠČIŲ PRISKYRIMAS	
Signalizatorius	GP27
Šviesos diodai	GP1
Relė	GP28
1,8" TFT ekranas	CS: GP17, DC: GP3, RES: GP6, BL: GP2
DHT11	GP0
Mygtukai	GP10, GP11, GP14 & GP15
Servopavaros	GP7, GP8, GP9 & GP20
UART	RXD: GP13, TXD: GP12
I2C	SDA: GP4, SCL: GP5
SPI	MISO: GP16, MOSI: GP19, SCLK: GP18

3. RASPBERRY PI PICO

Pirmiausia įkiškite "Raspberry Pi Pico" į plokštės lizdą.



Dabar prijunkite mikro USB kabelį prie kompiuterio ir "Raspberry Pi Pico" programavimo įrenginio.

DĖMESIO! Explorer plokštėje esantis USB-C prievadas naudojamas tik maitinimui. Jis nenaudojamas duomenims į "Raspberry Pi" perduoti. Mūsų pavyzdžio programai perkelti galite naudoti pasirinktą tinkamą programavimo programą. Rekomenduojame naudoti **Thonny Python IDE**.

DĖMESIO! Jei esate naujokas mikrovaldiklių ir elektronikos pasaulyje, nesijaudinkite! Parengėme jums specialų pradedančiųjų vadovą. Šis vadovas specialiai pritaikytas pradedančiųjų poreikiams ir jame žingsnis po žingsnio paaiškinama, kaip naudotis "Raspberry Pi Pico".

Šiame vadove jus supažindinsime su visu procesu - nuo pagrindinės konfigūracijos iki vykdomų projektų. Mūsų vadove pateikiami lengvai suprantami paaiškinimai ir naudingi patarimai, padėsiantys greitai ir veiksmingai tobulinti savo įgūdžius naudojant "Raspberry Pi Pico". Mūsų vadovą galite atsisiųsti **čia**.

4. IŠSAMIAI APIE MODULIUS

Toliau visi "Explorer Board" moduliai paaiškinami atskirai, pateikiant pavyzdinius kodus. Čia galite atsisiųsti visus pavyzdinius kodus ir bibliotekas, taip pat pavyzdinį kodą, susiejantį visus modulius.

Naudojant kai kuriuos modulius naudojamos išorinės bibliotekos ir šrifto failas. Atsisiųskite bibliotekas ir įkelkite jas į "Raspberry Pi Pico" lib aplanką. Šrifto failą įdėkite į Raspberry Pi Pico šakninį katalogą.

4.1 SIGNALIZATORIUS

Zumeris skleidžia signalinį toną, panašų į garsiakalbį. Tačiau, kitaip nei garsiakalbis, jis tinka tik ribotam dažnių diapazonui, todėl neskleidžia gero garso muzikai ar kalbai atkurti. Tačiau jis idealiai tinka garsiems įspėjamiesiems signalams - pypsėjimams - generuoti. Kai elektrinis prietaisas generuoja įspėjamąjį toną, beveik visada tai yra garsinis signalas. Pavyzdžiui, žadintuvuose, dūmų detektoriuose arba automobiliuose esančiuose saugos diržų priminimo signaluose.

Zumeris prijungtas prie GPIO kaiščio GP27.

```
# Load libraries
from machine import Pin, PWM

buzzerPin = Pin(27)
buzzer = PWM(buzzerPin)

while True:
    # Activate buzzer for 1 sec
    buzzer.freq(1000)
    buzzer.duty_u16(1000)
    sleep(1)
    buzzer.duty_u16(0)
    sleep(1)
```

4.2 RGB ŠVIESOS DIODAI

RGB šviesos diodai - tai šviesos diodų tipas, kuriame derinama raudona, žalia ir mėlyna spalva, kad būtų sukurtos įvairios spalvos. Panašiai kaip garsinis signalas skleidžia tik paprastus tonus, RGB šviesos diodai negali rodyti sudėtingų vaizdų, tačiau jie puikiai maišo ir keičia spalvas. Kiekvieno RGB įrenginio šviesos diodo intensyvumas gali būti skirtingas, kad būtų išgauti skirtingi atspalviai - nuo švelnių pastelinių iki ryškių, sodrių spalvų. Todėl jie idealiai tinka nuotaikingam apšvietimui, dekoratyviniam apšvietimui ir tais atvejais, kai reikalingi vizualiniai signalai, pavyzdžiui, žaidimų įrenginiuose arba kaip būsenos indikatoriai elektroniniuose prietaisuose. Dėl jų universalumo ir energijos vartojimo efektyvumo jie tapo populiariu pasirinkimu šiuolaikinėse apšvietimo sistemose, nors, kaip ir švilpukas, jų paprastas veikimas reiškia, kad jie negali sukurti sudėtingų vaizdų ar raštų be papildomų valdymo blokų.

GPIO šviesos diodai yra prijungti prie GPIO kaiščio GP1.

```
# Load libraries
from machine import Pin, PWM
from utime import sleep
from neopixel import NeoPixel

ledPin = 1
ledCount = 4

# Initialize GPIOs
led = Pin(ledPin, Pin.OUT)
led = NeoPixel(Pin(ledPin, Pin.OUT), ledCount)

while True:
    # Turn LEDs white
    for i in range (ledCount):
        led[i] = (255, 255, 255)
    led.write()
    sleep(1)
    # Turn LEDs red
    for i in range (ledCount):
        led[i] = (255, 0, 0)
    led.write()
    sleep(1)
    # Turn LEDs blue
    for i in range (ledCount):
        led[i] = (0, 0, 255)
    led.write()
    sleep(1)
    # Turn LEDs green
    for i in range (ledCount):
        led[i] = (0, 255, 0)
    led.write()
    sleep(1)
```

4.3 RELĖ

Relės yra vieni iš seniausių elektromechaninių komponentų ir veikia kaip elektra valdomi jungikliai. Esant nedidelei įėjimo įtampai ir mažai srovei, išėjime galima įjungti ir išjungti didelę elektros apkrovą. Kai relė persijungia, užsidega ir raudonas šviesos diodas. Į gnybtų lizdą (nuspaudę oranžinę svirtelę) galite įkišti nupjautus kabelių galus ir naudotis trimis jungtimis.

Relė prijungta prie GPIO kaiščio GP28.

```
# Load libraries
from machine import Pin, PWM
from utime import sleep

relayPin = 28
# Initialize GPIOs
relay = Pin(relayPin, Pin.OUT)

while True:
    # Toggle Relay
    relay.on()
    sleep(1)
    relay.off()
    sleep(1)
```

4.4 TFT

Skystųjų kristalų ekrano (LCD TFT), turinčio apie 65 000 spalvų ir 1,8 colio įstrižainę, skiriamoji geba yra 128×160 taškų, jį galima valdyti per SPI. Jis tinka spalvotai grafikai ir vaizdams rodyti. Raidės ir kiti simboliai rodomi kaip grafika, sudaryta iš daugybės atskirų taškų.

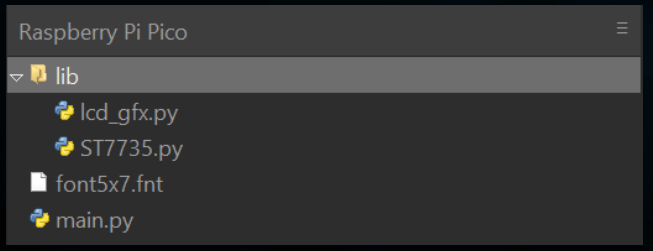
TFT yra prijungtas prie GPIO kontaktų GP17 (CS), GP3 (DC), GP6 (RES) ir GP2 (BL).

```
from machine import Pin, SPI
import ST7735

# Initialize LCD
spi = SPI(0, baudrate=8000000, polarity=0, phase=0, sck=Pin(18), mosi=Pin(19),
miso=Pin(16))
lcd = ST7735.ST7735(spi, rst=6, ce=17, dc=3)
backlight = Pin(2, Pin.OUT)

# Turn backlight on
backlight.high()
lcd.reset()
lcd.begin()

# Display content on the LCD
lcd.fill_screen(lcd.rgb_to_565(0, 255, 0))  # Fills the screen with a green color

# Display text
lcd.p_string(20, 50, ‚Hello, World!‘)
```

Be tekstų, taip pat galima rodyti stačiakampius, pvz:

```
# Draw red rectangle
lcd.draw_block(10, 10, 50, 50, lcd.rgb_to_565(255, 0, 0))
```

DĖMESIO! TFT ekranui reikalingi du atskiri bibliotekos failai ir šrifto failas; reikiamus failus galite atsisiųsti čia. Tada perkelkite visus failus iš aplanko Libraries į šakninį "Raspberry Pi Pico" katalogą, kad aplanko struktūra atrodytų taip:



4.5 DHT 11

DHT11 jutiklis gali nustatyti temperatūrą nuo 0 °C iki 50 °C (±2 °C tikslumu) ir santykinę drėgmę nuo 20 % iki 80 % (±5 %) (ne dažniau kaip kartą per sekundę). Meteorologijos stotys yra bene pagrindinė tokio jutiklio kaip DHT11 taikymo sritis. Norint patikrinti veikimą, pakanka priglausti burną prie jutiklio ir lėtai iškvėpti. Įkvėptas oras skiriasi nuo aplinkos temperatūros ir drėgmės, todėl vertės turėtų labai pasikeisti.

DHT11 yra prijungtas prie GPIO kaiščio GP0.

```
from machine import Pin
from dht import DHT11
from utime import sleep

# Initialize DHT11 Sensor
dhtPin = 0
dht = DHT11(Pin(dhtPin, Pin.IN))

while True:
    # Measure DHT11 values
    dht.measure()
    temp = dht.temperature()  # Temperature in Celsius
    humid = dht.humidity()    # Relative Humidity in %

    # Print the measurements
    print('Temperature:', temp, '°C')
    print('Humidity:', humid, '%')

    sleep(2)  # Wait for 2 seconds before the next reading
```

4.6 MYGTUKAI

Mygtukai yra interaktyvūs naudotojo sąsajų elementai, atliekantys paprastą, bet labai svarbią funkciją - naudotojo įvesties funkciją. Panašiai kaip RGB šviesos diodai gali atvaizduoti įvairias spalvas, mygtukai naudojami įvairioms komandoms ir veiksmams inicijuoti skaitmeninėje aplinkoje.

Mygtukai prijungti prie GPIO kontaktų GP10 (viršuje), GP11 (dešinėje), GP14 (apačioje) ir GP15 (kairėje).

```
from machine import Pin

# Define button pins
buttons = [10, 11, 14, 15]

# Initialize buttons
buttonOne = Pin(buttons[0], Pin.IN, Pin.PULL_DOWN)
buttonTwo = Pin(buttons[1], Pin.IN, Pin.PULL_DOWN)
buttonThree = Pin(buttons[2], Pin.IN, Pin.PULL_DOWN)
buttonFour = Pin(buttons[3], Pin.IN, Pin.PULL_DOWN)

# Define button handler functions
def buttonUp(pin):
    print("Button Up Pressed")

def buttonRight(pin):
    print("Button Right Pressed")

def buttonDown(pin):
    print("Button Down Pressed")

def buttonLeft(pin):
    print("Button Left Pressed")

# Attach interrupt handlers to buttons
buttonOne.irq(trigger=Pin.IRQ_RISING, handler=buttonUp)
buttonTwo.irq(trigger=Pin.IRQ_RISING, handler=buttonRight)
buttonThree.irq(trigger=Pin.IRQ_RISING, handler=buttonDown)
buttonFour.irq(trigger=Pin.IRQ_RISING, handler=buttonLeft)
```

4.7 SERVOPAVAROS

Servoelementą sudaro elektros variklis su pavarų dėže ir valdymo elektronika. Pavarų dėžės išėjimo pusėje yra krumpliaratis, ant kurio sumontuotas servo ragelis. Servoelementai naudojami modelių gamyboje, pavyzdžiui, lėktuvo ar laivo sparno ar vairo padėčiai valdyti. Vis daugiau servopavarų taip pat naudojama automobilių technikoje automatiškai uždarant duris, langų reguliatoriams, veidrodėliams ir kitiems reguliuojamiems elementams.

Servo jungtys yra GPIO kaiščiai GP7, GP8, GP9 ir GP20.

```
from machine import Pin, PWM
from utime import sleep

# Servo pin numbers
servoOnePin = 7
servoTwoPin = 8
servoThreePin = 9
servoFourPin = 20

# Initialize servos
servoOne = PWM(Pin(servoOnePin))
servoTwo = PWM(Pin(servoTwoPin))
servoThree = PWM(Pin(servoThreePin))
servoFour = PWM(Pin(servoFourPin))

# Servo degree positions in nanoseconds
deg0 = 500000
deg45 = 1000000
deg90 = 1500000
deg135 = 2000000
deg180 = 2500000

while True:
    # Move each servo through a range of angles
    for servo in [servoOne, servoTwo, servoThree, servoFour]:
        servo.duty_ns(deg0)
        sleep(1)
        servo.duty_ns(deg45)
        sleep(1)
        servo.duty_ns(deg90)
        sleep(1)
        servo.duty_ns(deg135)
        sleep(1)
        servo.duty_ns(deg180)
        sleep(1)
```

4.8 SĄSAJOS

Sąsajų jungtys elektronikos pasaulyje atlieka labai svarbų vaidmenį, panašiai kaip mygtukai vartotojo sąsajose. Jos užtikrina ryšį ir maitinimą tarp skirtingų elektroninių komponentų. Todėl mūsų "Explorer Board" sąsajos srityje galima rasti šias jungtis:

SPI (Serial Perhipheral Interface): Ši jungtis naudojama greitam nuosekliųjų duomenų perdavimui. Ją paprastai sudaro keturios linijos: MISO (Master In, Slave Out), MOSI (Master Out, Slave In), SCK (Serial Clock) ir SS (Slave Select). SPI idealiai tinka tais atvejais, kai reikia didelės duomenų perdavimo spartos, pavyzdžiui, valdant LCD ekranus arba SD korteles.

I2C (Inter-Integrated Circuit): I2C yra dviejų laidų sąsaja, kurią sudaro duomenų linija (SDA) ir laikrodžio linija (SCL). Ji dažniausiai naudojama mikrovaldiklių programose ryšiui tarp skirtingų integrinių grandynų palaikyti. Dėl savo paprastumo ji idealiai tinka programoms, kuriose nėra daug GPIO kontaktų.

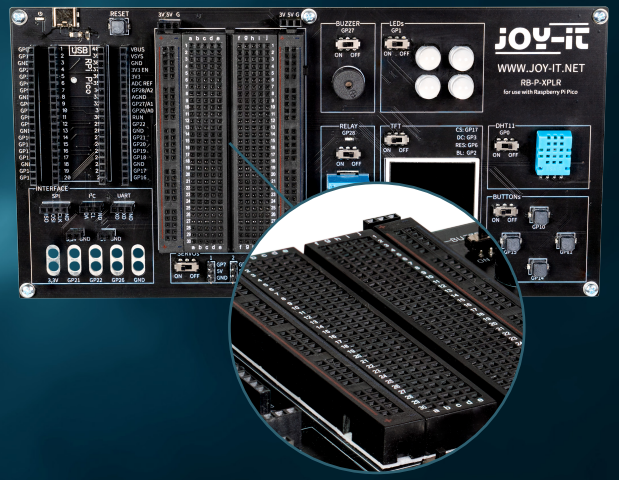
UART (Universal Asynchronous Receiver/Transmitter): Ši sąsaja leidžia palaikyti asinchroninį nuoseklųjį ryšį per dvi linijas: TX (Transmit) ir RX (Receive). UART dažnai naudojama ryšiui tarp mikrovaldiklių ir kompiuterių palaikyti arba tokiems moduliams, kaip GPS imtuvai ar "Bluetooth" moduliai, prijungti.

3,3 V ir 5 V jungtys: Šios jungtys užtikrina elektroninių komponentų maitinimą. 3,3 V įtampa dažnai naudojama šiuolaikiniuose mikrovaldikliuose ir jutikliuose, o 5 V - senesniuose ar daugiau energijos naudojančiuose įrenginiuose.

Krokodilo spaustukų jungtys: Šios jungtys idealiai tinka laikinoms jungtims arba bandymams. Jomis galima greitai ir lengvai prijungti įvairius komponentus ar matavimo prietaisus nelituojant. Iš viso "Explorer" plokštėje yra penkios tokios jungtys, kurias galima lanksčiai naudoti įvairiose srityse.

Kiekviena iš šių jungčių elektronikoje turi konkrečią paskirtį ir reikšmę, panašiai kaip skirtingų tipų mygtukai vartotojo sąsajoje atlieka skirtingas funkcijas. Jos užtikrina reikiamą lankstumą ir funkcionalumą kuriant ir plečiant elektronines sistemas.

4.9 BREADBOARD



Plokštės yra nepakeičiamas įrankis elektronikos pasaulyje, panašiai kaip sąsajos jungtys yra labai svarbios skirtingiems komponentams sujungti. Jos leidžia greitai ir be litavimo kurti ir išbandyti elektronines grandines, todėl yra ypač populiarios prototipų kūrimo ir švietimo tikslais.

Plokštę paprastai sudaro stačiakampio formos plastikinis blokas su daugybe įterptų skylučių, išdėstytų eilėmis. Šios skylutės viduje sujungtos metaliniais pėdsakais, kuriais galima lengvai prijungti ir sujungti komponentus ir laidus. Standartinis duonos plokštės išdėstymas apima dvi pagrindines sritis:

Pagrindinės sritys: Jas sudaro kelios lygiagrečios skylučių eilės, paprastai atskirtos centriniu grioveliu. Eilėje esančios skylutės tarpusavyje sujungtos elektra. Toks išdėstymas idealiai tinka integriniams grandynams (IC) ir kitiems komponentams įterpti.

Maitinimo juostelės: Plokštės krašte paprastai yra viena ar dvi eilės skylučių, kurios tarnauja kaip maitinimo juostelės. Jos yra sujungtos vertikaliai per visą plokštės ilgį ir yra patogus būdas užtikrinti maitinimą ir įžeminimą įvairiuose grandinės taškuose.

Plokštės lankstumas pasireiškia tuo, kad ją galima naudoti pakartotinai ir kurti grandines be nuolatinių pakeitimų. Dėl to ji idealiai tinka eksperimentams, nes galima lengvai ištaisyti klaidas ir pašalinti komponentus jų nepažeidžiant. Be to, tai puiki mokymosi priemonė, nes padeda suprasti grandinės logiką ir komponentų funkcijas praktiškai ir vaizdžiai.

Be to, galima įsigyti įvairių dydžių ir su skirtingu skaičiumi jungčių taškų, kad būtų patenkinti įvairūs reikalavimai. Mažesnės planšetės tinka paprastiems projektams ir eksperimentams, o didesnės - sudėtingesnėms grandinėms.

Nepaisant jų universalumo, planšetės turi ir apribojimų. Jos netinka labai aukštiems dažniams arba grandinėms, kurioms reikia didelės galios. Be to, kartais jungtys gali būti mažiau patikimos nei lituotos, ypač jei laikui bėgant susidėvi plokštė.

Apskritai planšetės yra būtina priemonė visiems, dirbantiems su elektronika - nuo pradedančiųjų, besimokančių pagrindų, iki patyrusių kūrėjų, norinčių greitai ir efektyviai kurti prototipus. Tai elektroninis menininko eskizų sąsiuvinio atitikmuo: vieta, kur galima tyrinėti idėjas ir eksperimentuoti prieš sukuriant galutinį kūrinį.

5. PROJEKTAI

Sveiki atvykę į skyrių apie novatoriškus elektronikos projektus su "Raspberry Pi Pico"! Šiame skyriuje susipažinsite su įvairiomis taikomosiomis programomis - nuo paprasto šviesos diodų valdymo iki sudėtingesnių sistemų, pavyzdžiui, automatizuotų meteorologinių stočių ir dinaminio apšvietimo sistemų, kūrimo. Kiekvienas projektas kruopščiai parengtas taip, kad įgytumėte praktinės patirties su įvairiais aparatinės įrangos komponentais.

Pradėkite pažinimo kelionę nuo pagrindinių projektų, kuriuose išmoksite naudotis "Raspberry Pi Pico" bendrosios paskirties įvesties ir išvesties (GPIO) jungtimis, ir tobulinkite savo įgūdžius pažengusiomis temomis, pavyzdžiui, valdydami servo variklius arba naudodami jutiklius aplinkos stebėjimui. Naudodami tokius komponentus, kaip sukamieji enkoderiai, ultragarsiniai jutikliai, švilpukai ir "Neopixel" šviesos diodai, išmoksite kurti interaktyvias ir reaktyvias sistemas.

Kiekviename projekte pateikiamas išsamus reikiamų komponentų įvadas, žingsnis po žingsnio instrukcijos apie aparatinės įrangos konfigūravimą ir aiškūs programinio kodo pavyzdžiai, padedantys suprasti elektronikos ir kompiuterinio programavimo principus. Taip pat paaiškinama, kaip integruoti išorinius jutiklius ir vykdiklius, kad būtų galima rinkti duomenis realiuoju laiku ir į juos reaguoti.

Šie projektai yra ne tik mokomieji, bet ir įdomūs, juose daug pritaikymo ir išplėtimo galimybių, todėl galite kurti savo kūrybinius sprendimus. Nesvarbu, ar esate pradedantysis, tik pradedantis tyrinėti skaitmeninės elektronikos pasaulį, ar patyręs kūrėjas, norintis praplėsti savo įgūdžius, šiame skyriuje rasite išteklių ir įkvėpimo, kurio reikia norint patobulinti savo techninius įgūdžius ir smagiai mokytis. Pasiruoškite plėsti savo programavimo ir elektronikos įgūdžius, nes būsite vedami per kiekvieną projektą, o kartu smagiai kurdami ir eksperimentuodami.

Kiekvieno projekto pabaigoje rasite atitinkamus pavyzdinius kodus. Failus taip pat galite atsisiųsti **čia**.

5.1 ATSTUMO RODYMAS

Mūsų pirmojo projekto tikslas - sukurti ultragarsinį tolimatį, kuris TFT ekrane atvaizduotų atstumus. Šis projektas yra puiki įžanga į duomenų jutiklių nustatymą ir vizualizavimą naudojant "Raspberry Pi Pico".

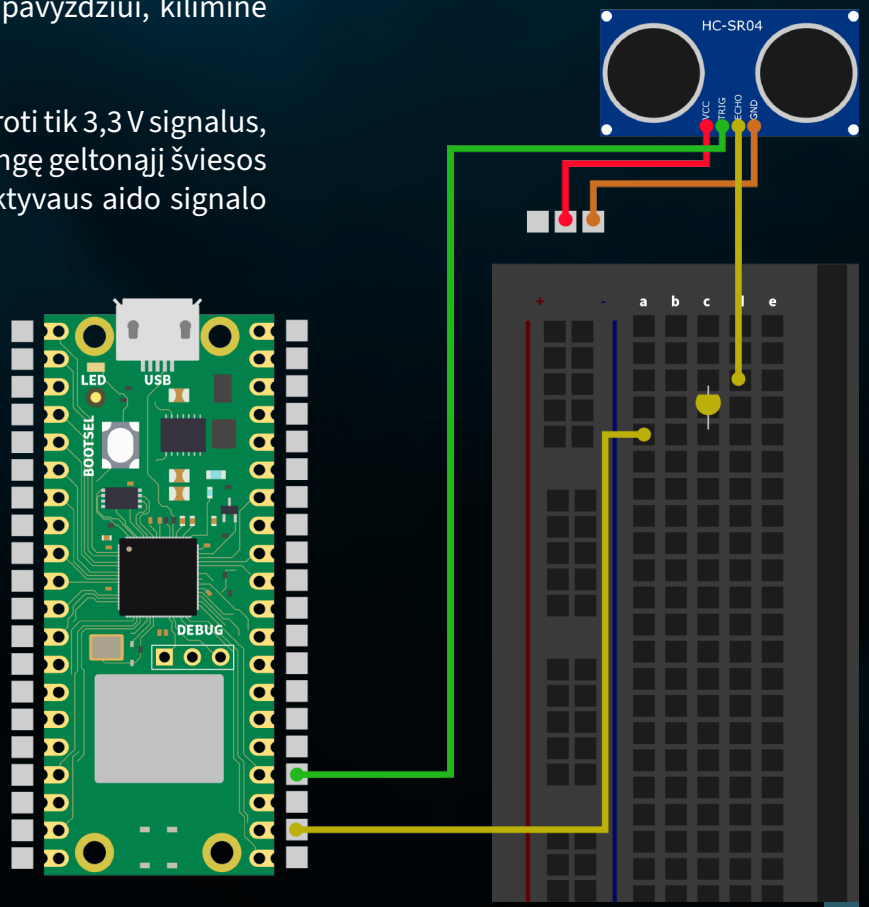
ULTRAGARSINIS JUTIKLIS: Siųstuvas skleidžia ultragarso bangą ir matuoja laiką, kol ji atsispindės ir pasieks siųstuvą. Kadangi garso greitis įvairiose terpėse, pavyzdžiui, ore (343 m/s esant 20 °C temperatūrai) ir vandenyje (1 484 m/s), yra žinomas, galima apskaičiuoti atstumą iki atsispindinčio paviršiaus (perpus sumažinus tranzito laiką, nes buvo išmatuotas atstumas ten ir atgal). Mikrokontrolerio impulsas paleidimo įėjime sukelia aštuonių trumpų ultragarso impulsų seką. Kai tik signalas vėl gaunamas, aido išėjimas trumpam tampa aukštas. Laikas tarp suveikimo ir aido signalo atitinka tranzito laiką. Atstumas gali būti matuojamas nuo 2 cm iki 400 cm ir yra gana tikslus, jei atspindintis paviršius yra kuo kietesnis ir lygesnis. Minkštos medžiagos, pavyzdžiui, kiliminė danga, gali net sutrukdyti atlikti matavimą.

Kadangi ultragarso jutikliui reikia 5 V maitinimo, o Raspberry Pi Pico be problemų gali apdoroti tik 3,3 V signalus, norint išvengti pažeidimų, būtina sumažinti signalo įtampą. Tai pasiekiame nuosekliai sujungę geltonąjį šviesos diodą ant plokštės ir panaudoję jį signalui žeminti. Šviesos diodas taip pat veikia kaip aktyvaus aido signalo indikatorius.

Daugiau informacijos apie šviesos diodus taip pat rasite 5.5 skyriuje.

RASPBERRY PI PICO	ULTRAGARSINIS JUTIKLIS
3V3	VCC
GP17	Trig
GP16	Echo
GND	GND

DĖMESIO! Šiame projekte būtina nustatyti relės ir DHT11 jungiklius į OFF, o TFT ekrano jungiklį į ON.



SANTRAUKA: Pirmajame projekte ultragarsiniu jutikliu matuojame atstumus ir vizualizuojame išmatuotą atstumą užpildydami TFT ekrano grafiką didesniu ar mažesniu laipsniu. Mūsų pavyzdyje visiškai užpildome ekraną nuo išmatuoto 100 cm atstumo.

```
# Load libraries
from machine import Pin, SPI
import ST7735
import time
import lcd_gfx

# Initialization of GPIO16 as input and GPIO17 as output
trig = Pin(17, Pin.OUT)
echo = Pin(16, Pin.IN, Pin.PULL_DOWN)

# Initialize LCD
spi = SPI(0, baudrate=8000000, polarity=0, phase=0, sck=Pin(18), mosi=Pin(19),
miso=Pin(16))
lcd = ST7735.ST7735(spi, rst=6, ce=17, dc=3)
backlight = Pin(2, Pin.OUT)
backlight.high()
lcd.reset()
lcd.begin()
lcd.fill_screen(lcd.rgb_to_565(255, 255, 255))

def translate(value, leftMin, leftMax, rightMin, rightMax):
    # Figure out how 'wide' each range is
    leftSpan = leftMax - leftMin
    rightSpan = rightMax - rightMin

    # Convert the left range into a 0-1 range (float)
    valueScaled = float(value - leftMin) / float(leftSpan)

    # Convert the 0-1 range into a value in the right range.
    return rightMin + (valueScaled * rightSpan)

# Endless loop for measuring the distance
while True:
     # Distance measurement is started using the 10us trigger signal
     trig.value(0)
     time.sleep(0.1)
     trig.value(1)

     # Now wait at the echo input until the signal has been activated
     # Then the time is measured for how long it remains activated
     time.sleep_us(2)
     trig.value(0)
     while echo.value()==0:
          pulse_start = time.ticks_us()
     while echo.value()==1:
          pulse_end = time.ticks_us()
     pulse_duration = pulse_end - pulse_start
```

Ultragarsinio jutiklio ir TFT ekrano inicijavimas

Pagalbinė funkcija verčių diapazonui reguliuoti

Atstumo matavimas

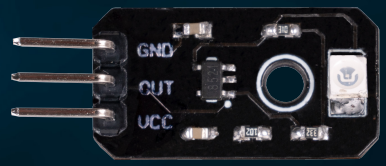
```
# Now the distance is calculated using the recorded time
distance = pulse_duration * 17165 / 1000000
distance = round(distance, 0)

# Serial output
print ('Distance:',"{:.0f}".format(distance),'cm')
time.sleep(1)

# Adjust measured value to LCD height
if(distance > 100):
    distance = 100
drawHeight = round(translate(distance, 0, 100, 0, 160))

# Fill the TFT display
lcd.fill_screen(lcd.rgb_to_565(255, 255, 255))
lcd.draw_block(0, 0, 128, drawHeight, lcd.rgb_to_565(0, 255, 0))
```

Vertės diapazono apskaičiavimas ir TFT ekrano aprašymas

5.2 METEOROLOGIJOS STOTIS

Pasinerkite į antrąjį mūsų elektronikos nuotykių projektą, kuriame sukursite savo orų stotį! Šiame projekte UV jutiklis derinamas su DHT11 temperatūros ir drėgmės jutikliu, kad ne tik sužinotumėte apie esamas oro sąlygas, bet ir išmatuotumėte UV spinduliuotę savo vietovėje. Visa ši informacija aiškiai rodoma spalvotame TFT ekrane, todėl orus ir UV spinduliuotę galite matyti iš karto.

UV JUTIKLIS: UV jutiklis yra nedidelis komponentas, padedantis matuoti nematomus ultravioletinius (UV) saulės spindulius. UV spinduliai - tai nematoma saulės šviesos dalis, kuri gali turėti įtakos mūsų odai ir sveikatai. Pagalvokite apie nudegimą saulėje arba odos įdegį - abu šiuos reiškinius sukelia UV spinduliai.

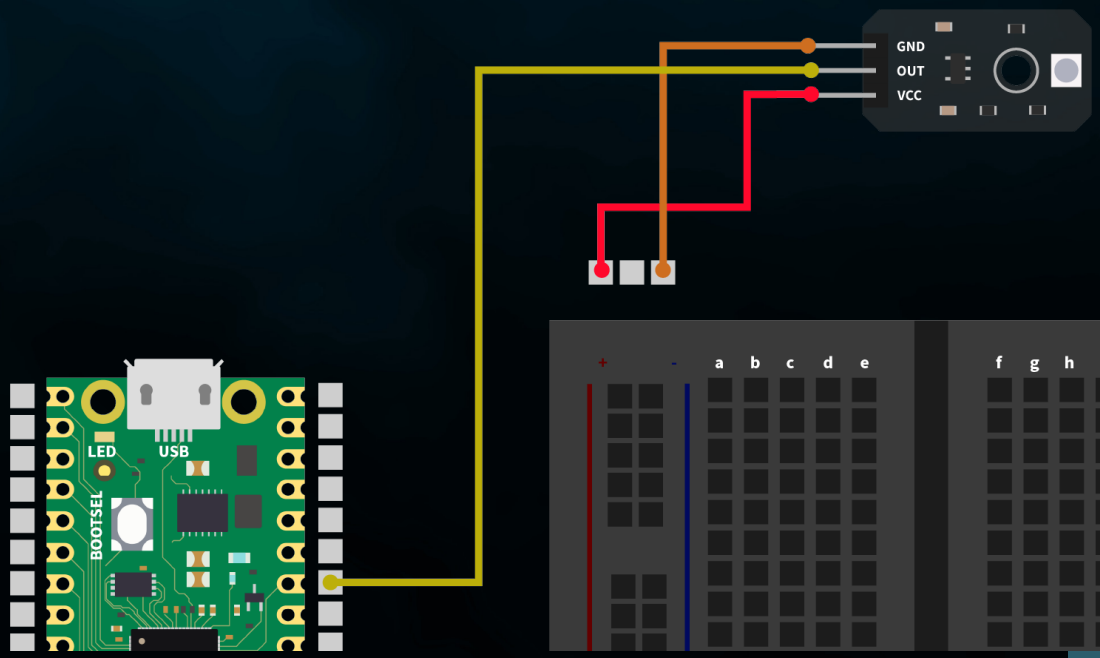


Jutiklio pagrindą sudaro medžiaga, reaguojanti į UV spinduliuotę. Kai UV spinduliai patenka į šią medžiagą, jutiklio elektrinė varža pakinta. Šį pokytį jutiklis paverčia signalu, kurį galime išmatuoti ir nuskaityti. Naudodami "Raspberry Pi Pico", šį signalą galime paversti verte, rodančia, kokia stipri šiuo metu yra UV spinduliuotė.

Pirmiausia prijunkite UV spinduliuotės jutiklį prie "Raspberry Pi Pico" naudodami pridedamus laidus. Nors, žinoma, galite jį prijungti ir prie duonos plokštės, tačiau tiesioginis kabelių sujungimas yra daug lankstesnis.

RASPBERRY PI PICO	UV JUTIKLIS
GND	GND
GP28	OUT
3V3	VCC

DĖMESIO! Šiame projekte būtina nustatyti relės jungiklį į OFF, o TFT ekrano ir DHT11 jutiklio jungiklius į ON.



SANTRAUKA: Mūsų meteorologijos stotis nuskaito DHT11 ir UV jutiklių duomenis ir pateikia juos TFT ekrane.

```
from machine import ADC, Pin, SPI
import utime
import dht
import ST7735  # Assuming this is the library for your TFT display

# Initialize DHT11 sensor
sensor_dht11 = dht.DHT11(Pin(0))

# Initialize UV sensor
uv_sensor = ADC(2)  # Assuming GP28 is ADC pin number 1 in your configuration

# Initialize LCD
spi = SPI(0, baudrate=8000000, polarity=0, phase=0, sck=Pin(18), mosi=Pin(19),
miso=Pin(16))
lcd = ST7735.ST7735(spi, rst=Pin(6), ce=Pin(17), dc=Pin(3))
backlight = Pin(2, Pin.OUT)
backlight.high()
lcd.reset()
lcd.begin()
lcd.fill_screen(lcd.rgb_to_565(255, 255, 255))

while True:
    lcd.fill_screen(lcd.rgb_to_565(255, 255, 255))

    # Read UV value
    uv_value = uv_sensor.read_u16()
    # Conversion in percent
    uv_percent = (uv_value / 65000) * 100
    print("UV Intensity (percent):", uv_percent)

    # DHT11 Read values
    sensor_dht11.measure()
    temp = sensor_dht11.temperature()
    humid = sensor_dht11.humidity()

    # Display values on LCD
    lcd.p_string(20, 20, "Temp: {}C".format(temp))
    lcd.p_string(20, 40, "Humid: {}%".format(humid))
    lcd.p_string(20, 60, "UV: {:.2f}%".format(uv_percent))  # Display of UV
intensity in percent with two decimal places

    utime.sleep(10)
```

TFT ekrano inicijavimas

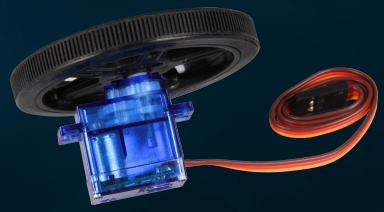
Jutiklio verčių matavimas

Išvestis ekrane

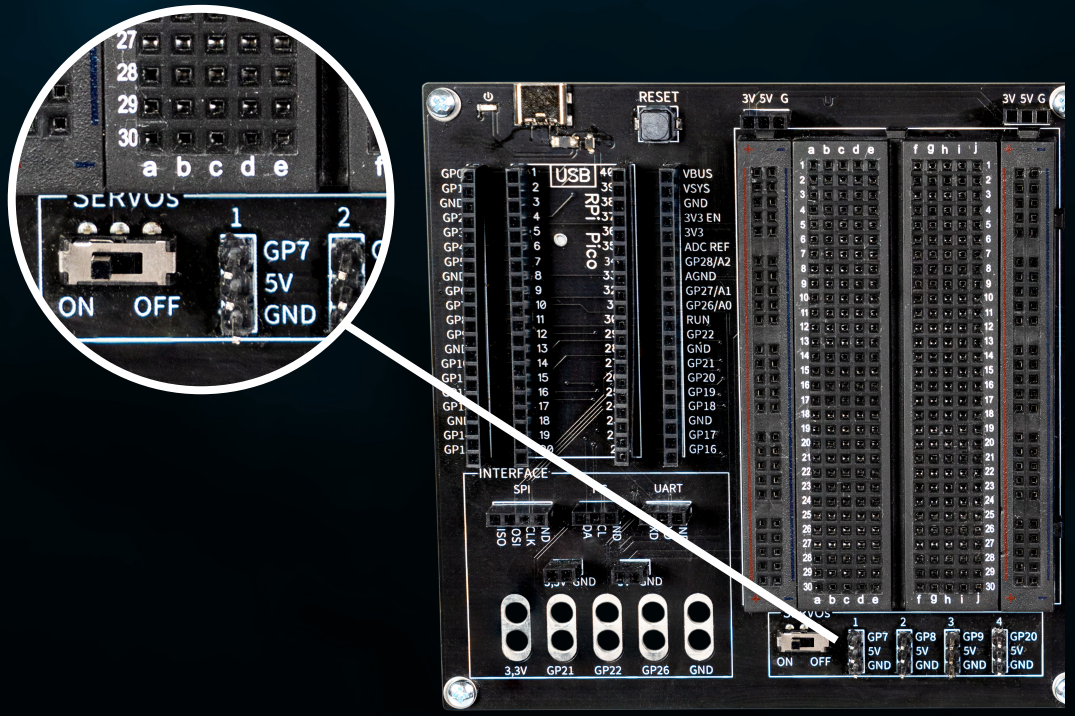
5.3 SERVO VALDYMAS

Kviečiame susipažinti su trečiuoju projektu iš mūsų įdomių elektronikos nuotykių su "Explorer" rinkiniu serijos! Šį kartą viskas apie judėjimą ir valdymą. Mūsų tikslas - suprogramuoti ir valdyti servo variklį taip, kad jo sukimosi kryptį būtų galima valdyti tiesiog paspaudus mygtuką. Šis projektas ne tik puikiai supažindina su variklių valdymo pasauliu, bet ir parodo, kaip sustiprinti sąveiką per vizualinį grįžtamąjį ryšį TFT ekrane.

SERVOVARIKLIS: Servoelementą sudaro elektros variklis su pavarų dėže ir valdymo elektronika. Pavarų dėžės išėjimo pusėje yra krumpliaratis, ant kurio sumontuotas servo ratukas. Servoelementai naudojami modelių gamyboje, pavyzdžiui, lėktuvo ar laivo sparno ar vairo padėčiai valdyti. Vis daugiau servopavarų taip pat naudojama automobilių technikoje automatiškai uždarant duris, langų reguliatoriams, veidrodėliams ir kitiems reguliuojamiems elementams.



Pirmiausia prijunkite servo variklį prie servo sąsajos su "Explorer" plokštės numeriu 1.



DĖMESIO! Šiame projekte būtina nustatyti TFT ekrano, mygtukų ir servopavarų jungiklį į padėtį ON.

SANTRAUKA: Valdome savo servo variklį ir leidžiame jam persijungti tarp kairiojo ir dešiniojo sukimosi, valdomo mūsų mygtukais.

Servo variklio ir mygtukų inicijavimas

```
from machine import Pin, PWM
from utime import sleep

# Servo pin numbers
servoOnePin = 7

# Key pin numbers
buttonLeftPin = 15
buttonRightPin = 11

# Initialization of the servo
servoOne = PWM(Pin(servoOnePin))

# Initialize the buttons with PULL_UP to use the default HIGH state
buttonLeft = Pin(buttonLeftPin, Pin.IN, Pin.PULL_UP)
buttonRight = Pin(buttonRightPin, Pin.IN, Pin.PULL_UP)

# Servo speeds in nanoseconds
leftSpeed = 1300000  # Moves the servo to the left
rightSpeed = 1700000  # Moves the servo to the right

# Servo frequency
servoOne.freq(50)  # Typical servo frequency of 50Hz

# Status of the servo
servoState = 'left'  # Starts with counterclockwise rotation

# Last state of the buttons to recognize edges
lastButtonLeft = buttonLeft.value()
lastButtonRight = buttonRight.value()

while True:
    # Read out the current status of the buttons
    currentButtonLeft = buttonLeft.value()
    currentButtonRight = buttonRight.value()

    # Check whether an edge from HIGH to LOW was detected (button was pressed)
    if lastButtonLeft == 1 and currentButtonLeft == 0:
        servoState = 'right'  # Changes the direction to the right when the left
button is pressed
    elif lastButtonRight == 1 and currentButtonRight == 0:
        servoState = 'left'  # Changes the direction to the left when the right
button is pressed
```

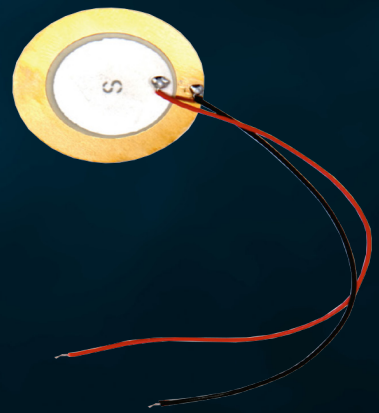
Mygtukų tikrinimas

```
    # Update the states for the next run
    lastButtonLeft = currentButtonLeft
    lastButtonRight = currentButtonRight

    # Controls the servo based on the current status
    if servoState == 'left':
        servoOne.duty_ns(leftSpeed)  # Moves the servo to the left
    else:
        servoOne.duty_ns(rightSpeed)  # Moves the servo to the right

    sleep(0.1)  # Short break for the control cycle
```

Servo valdymas

5.4 SAVADARBIS SKAMBUTIS

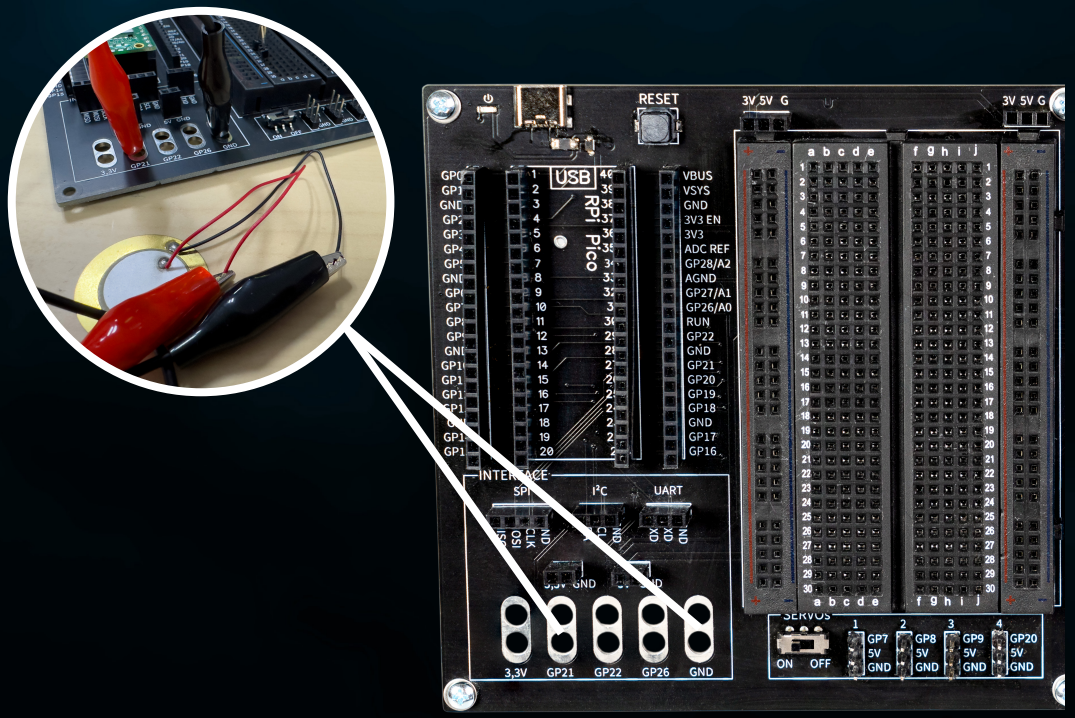
Ketvirtasis mūsų elektronikos nuotykių su "Explorer" rinkiniu projektas - apie garsą! Sukurdami savo švilpuko grandinę, pasinersime į akustinių signalų pasaulį. Šis projektas suteikia galimybę ne tik suprasti grandinių kūrimo pagrindus, bet ir išmokti kurti garsinius signalus naudojant paprastas priemones. Naudojant aligatoriaus spaustukus, šis konstravimas tampa ypač patogus ir prieinamas net tiems, kurie dar tik pradeda savo kelionę elektronikos srityje.



Pirmiausia kruopščiai prijunkite švilpuką prie "Explorer" plokštės naudodami krokodilo spaustukus. Šie gnybtai idealiai tinka greitam ir lanksčiam sujungimui be litavimo. Švilpukas, mažas komponentas, galintis skleisti garsą, tampa pagrindiniu mūsų projekto elementu. Įjungus srovę, švilpukas vibruoja ir skleidžia garsą.

Pirmiausia prijunkite aligatoriaus spaustuką prie "Explorer" plokštės GP21 aligatoriaus spaustuko jungties. Kitą krokodilo spaustuko galą prijunkite prie raudonojo švilpuko laido. Kitą krokodilo spaustuką prijunkite prie "Explorer" plokštės GND krokodilo spaustuko jungties. Kitą gnybto galą prijunkite prie juodo signalizatoriaus kabelio.

RASPBERRY PI PICO	SIGNALIZATORIUS
GP21	Red cable
GND	Black cable



DĖMESIO! Šiam projektui būtina nustatyti mygtukų jungiklį į ON.

SANTRAUKA: Prie savo "Raspberry Pi Pico" prijungiame išorinį garsiakalbį ir grojame paprastą, smagią melodiją.

```
from machine import Pin, PWM
import utime
# Note frequencies (in Hz)
notes = {
    'C4': 262,
    'D4': 294,
    'E4': 330,
    'F4': 349,
    'G4': 392,
    'A4': 440,
    'B4': 494,
    'C5': 523
}
# Melody and duration (in ms)
melody = ['C4', 'D4', 'E4', 'C4', 'C4', 'D4', 'E4', 'C4', 'E4', 'F4', 'G4', 'E4',
'F4', 'G4']
durations = [500, 500, 500, 500, 500, 500, 500, 500, 500, 500, 1000, 500, 500,
1000]
# Initialization of the buzzer
buzzer = PWM(Pin(21))
buzzer.freq(440) # Set a start frequency
# Function to play a note
def play_note(note, duration):
    if note in notes:
        buzzer.freq(notes[note]) # Set the frequency based on the note
        buzzer.duty_u16(32767) # Start the PWM signal
        utime.sleep_ms(duration) # Hold the note for the duration
    buzzer.duty_u16(0) # Stop the PWM signal (switch off note)
    utime.sleep_ms(50) # Short pause between the notes
# Play the melody
for note, duration in zip(melody, durations):
    play_note(note, duration)
buzzer.deinit() # Deactivate the PWM channel when finished
```

Pastabų sąrašas

Melodijos atmintis

Natos grojimo funkcija

5.5 JŪSŲ NUOSAVA GRANDINĖ

Penktajame elektronikos nuotykių su "Explorer" rinkiniu projekte tyrinėjame žavų šviesos valdymo pasaulį. Šį kartą kuriame grandinę, kurią galite naudoti šviesos diodams valdyti. Tai puiki proga suprasti elektroninių grandinių principus ir kartu įgyti galimybę valdyti apšvietimo efektus.

LEDS: Šviesos diodai - tai maži, bet galingi šviesos šaltiniai, naudojami daugelyje elektroninių projektų. Jie turi daug privalumų, palyginti su tradicinėmis lemputėmis, pavyzdžiui, ilgesnį tarnavimo laiką, mažesnes energijos sąnaudas ir galimybę šviesti įvairiomis spalvomis. Šviesos diodą sudaro puslaidininkinė medžiaga, kuri skleidžia šviesą, kai ja teka elektros srovė.

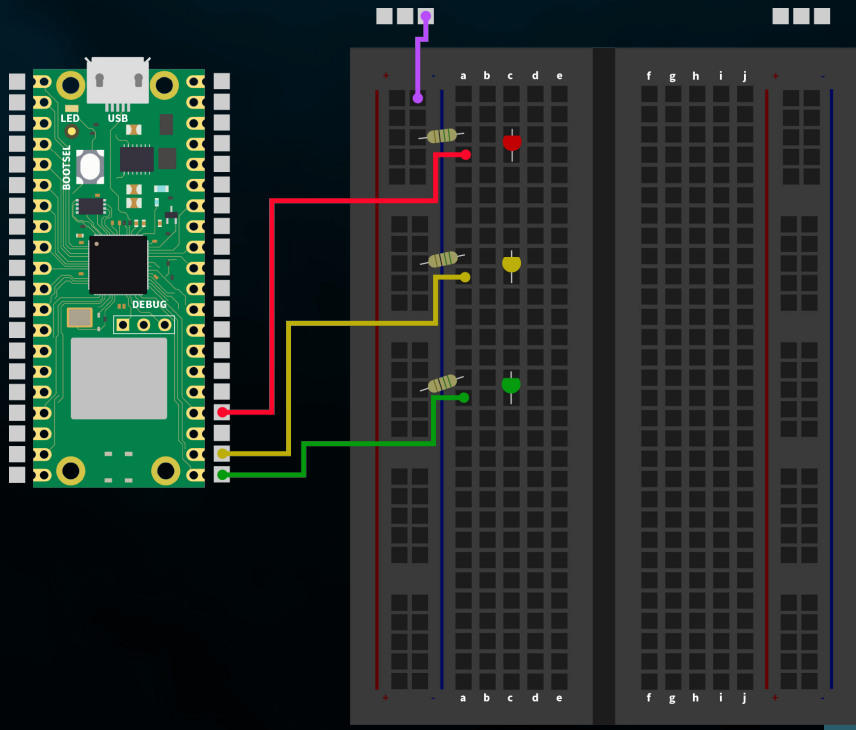
Svarbu atpažinti teisingą šviesos diodų poliariškumą, nes jie veikia tik tada, kai srovė jais teka tinkama kryptimi. Tai reiškia, kad teigiamasis maitinimo šaltinio polius turi būti prijungtas prie teigiamo šviesos diodo galo, o neigiamasis maitinimo šaltinio polius - prie neigiamo šviesos diodo galo.

Taip atpažįstamas šviesos diodo poliškumas:

ilgesnioji kojelė: daugumos šviesos diodų ilgesnioji kojelė yra anodas (+), t. y. teigiama jungtis. Trumpesnė koja yra katodas (-), t. y. neigiama jungtis.

Plokščias kraštas: Šviesos diodo korpuso šone gali būti plokščias kraštas. Šioje pusėje paprastai pažymimas katodas, t. y. neigiamas polius.

Pirmiausia atkurkite grandinę, kaip parodyta toliau pateiktoje schemoje. Tačiau įsitikinkite, kad šviesos diodų poliškumas yra teisingas. Taip pat kiekvienam šviesos diodui naudokite 56 Ω (žalias-mėlynas-juodas) nuoseklųjį rezistorių.



RASPBERRY PI PICO	LEDS
GP18	Raudonas šviesos diodas
GP17	Geltonas šviesos diodas
GP16	Žalias šviesos diodas

DĖMESIO! Šiame projekte būtina nustatyti TFT ekrano jungiklį į padėtį OFF.

SANTRAUKA: Valdome tris skirtingus šviesos diodus per "Raspberry Pi Pico" kaiščius, o kiekvienas šviesos diodas mirksi pakaitomis.

Šviesos diodų inicijavimas

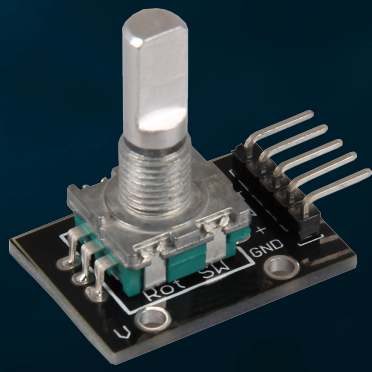
```python
from machine import Pin
import utime

# Initialize the LEDs
red_led = Pin(18, Pin.OUT)
yellow_led = Pin(17, Pin.OUT)
green_led = Pin(16, Pin.OUT)

# Flashing function for one LED
def blink_led(led, duration):
    led.value(1)  # Switch on LED
    utime.sleep(duration)
    led.value(0)  # Switch off LED
    utime.sleep(duration)

# Hauptschleife
while True:
    blink_led(red_led, 0.5)     # Red LED flashes for 0.5 seconds
    blink_led(yellow_led, 0.5) # Yellow LED flashes for 0.5 seconds
    blink_led(green_led, 0.5)   # Green LED flashes for 0.5 seconds
```

LED mirksėjimo funkcija

Pagrindinė kilpa

5.6 LED VALDYMAS

Šeštajame elektronikos nuotykių projekte sukamuoju koduotoju valdome šviesos diodų ryškumą ir spalvą - tai paprastas, bet įdomus būdas pasinerti į elektroniką. Sukamasis valdiklis, mūsų centrinis valdymo elementas, dėl savo dvigubos funkcijos leidžia žaismingai bendrauti. Ryškumo pokyčiai valdomi sukant, o spaudžiant valdiklį cikliškai keičiamos šviesos diodų spalvos - tai idealiai tinka iliustruoti elektronikos ir spalvų maišymo pagrindus.

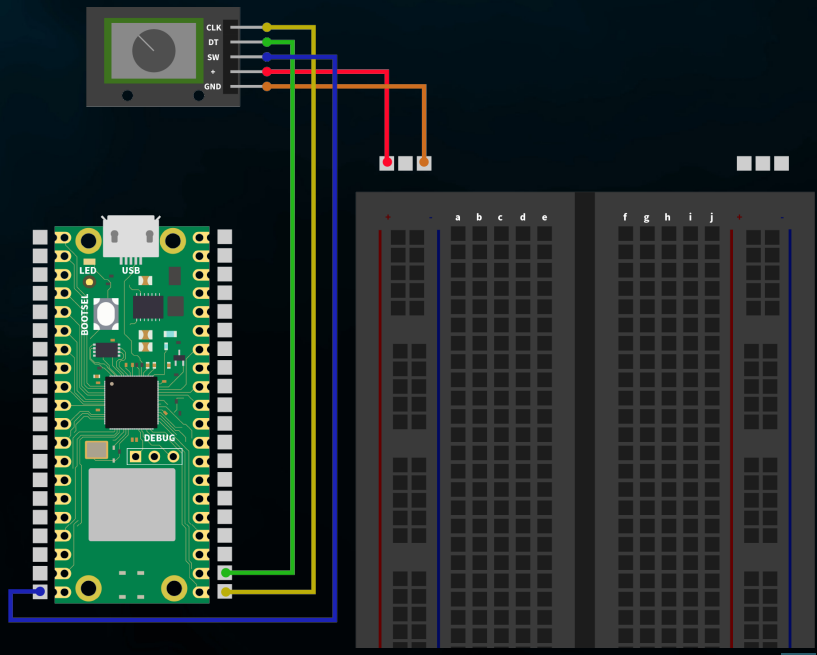


SUKAMASIS DAVIKLIS: Rotacinis kodavimo įtaisas - tai išmanus mažas prietaisas, kuris paverčia jūsų sukamuosius judesius elektroniniais signalais. Įsivaizduokite sukamąją rankenėlę, kokią žinote iš radijo imtuvo. Kai sukate šią rankenėlę, sukamasis kodavimo įtaisas gali išmatuoti, kiek ir kuria kryptimi ją pasukote. Šią informaciją galima naudoti, pavyzdžiui, garsumui keisti, naršyti meniu arba, mūsų projektuose, reguliuoti šviesos diodų ryškumą.

Sukamuosiuose davikliuose dažnai būna įmontuotas mygtukas - tai reiškia, kad jie gali veikti ir kaip slėgio jungiklis. Paspaudus sukamąjį mygtuką, sukamasis daviklis šį slėgį atpažįsta kaip atskirą signalą. Jį galima naudoti įvairioms funkcijoms, pavyzdžiui, prietaisui įjungti ir išjungti arba darbo režimams keisti.

APIBENDRINANT: Sukant ir spaudžiant sukamąjį valdiklį galima siųsti įvairias komandas elektronikos projektams. Tai intuityvus ir universalus įrankis, leidžiantis lengvai ir smagiai sąveikauti su jūsų projektais.

Pirmiausia prie "Raspberry Pi Pico" prijunkite sukamąjį daviklį taip:



RASPBERRY PI PICO	SUKAMASIS DAVIKLIS
GP16	CLK
GP17	DT
GP15	SW
3V3	+
GND	GND

DĖMESIO! Šiame projekte reikia nustatyti, kad TFT ekrano jungiklis būtų išjungtas, o šviesos diodų jungiklis - įjungtas.

SANTRAUKA: Keturių šviesos diodų spalvai ir ryškumui valdyti naudojame sukamąjį valdiklį. Sukant valdiklį keičiamas ryškumas, o spaudžiant valdiklį reguliuojama šviesos diodų spalva.

```
from machine import Pin
import utime
import neopixel

# Neopixel setup
NUM_LEDS = 4
PIXEL_PIN = 1
np = neopixel.NeoPixel(Pin(PIXEL_PIN), NUM_LEDS)

# Rotate encoder setup
PIN_CLK = Pin(16, Pin.IN, Pin.PULL_UP)
PIN_DT = Pin(17, Pin.IN, Pin.PULL_UP)
BUTTON_PIN = Pin(15, Pin.IN, Pin.PULL_UP)

# Global variables
counter = 0
PIN_CLK_LAST = PIN_CLK.value()
delayTime = 0.001
debounce_time_encoder = 0
debounce_time_button = 0

# Initialize colors
colors = [(255, 0, 0), (0, 255, 0), (0, 0, 255), (255, 255, 255)]  # Rot, Grün,
Blau, Weiß
color_index = 0

# Initialize brightness
brightness_levels = [0.2, 0.4, 0.6, 0.8, 1.0]
brightness_index = 0

def update_leds(color, brightness):
    dimmed_color = tuple([int(c * brightness) for c in color])
    for i in range(NUM_LEDS):
        np[i] = dimmed_color
    np.write()

def rotaryFunction(null):
    global counter, brightness_index, debounce_time_encoder
    PIN_CLK_CURRENT = PIN_CLK.value()
    if PIN_CLK_CURRENT != PIN_CLK_LAST and (utime.ticks_ms() - debounce_time_
encoder) > 300:
        if PIN_DT.value() != PIN_CLK_CURRENT:
            brightness_index = (brightness_index + 1) % len(brightness_levels)
        else:
            brightness_index = (brightness_index - 1) % len(brightness_levels)
        update_leds(colors[color_index], brightness_levels[brightness_index])
        debounce_time_encoder = utime.ticks_ms()
```

Šviesos diodų ir sukamojo valdiklio inicializavimas

Sukamojo daviklio funkcija

```
def counterReset(null):
    global color_index, debounce_time_button
    if (utime.ticks_ms() - debounce_time_button) > 300:
        color_index = (color_index + 1) % len(colors)
        update_leds(colors[color_index], brightness_levels[brightness_index])
        debounce_time_button = utime.ticks_ms()



PIN_CLK.irq(trigger=Pin.IRQ_FALLING | Pin.IRQ_RISING, handler=rotaryFunction)
BUTTON_PIN.irq(trigger=Pin.IRQ_FALLING, handler=counterReset)

update_leds(colors[color_index], brightness_levels[brightness_index])

while True:
    utime.sleep(delayTime)
```

Pastabų sąrašas

5.7 AUTOMATINIS RYŠKUMO VALDYMAS

Septintajame elektronikos nuotykių projekte naudojame fotodiodą, kad automatiškai valdytume šviesos diodų ryškumą. Fotodiodas šviesą paverčia elektriniu signalu, todėl šviesos diodai šviečia ryškiau, kai tamsu, ir blankiau, kai aplinkoje yra daugiau šviesos.

Prijungus fotodiodą prie Explorer Board ir užprogramavus jį Raspberry Pi Pico, šviesos diodai protingai prisitaiko prie aplinkos šviesumo. Šis projektas parodo, kaip iš paprastų komponentų galima sukurti reaktyvias ir energiją taupančias elektronines sistemas.

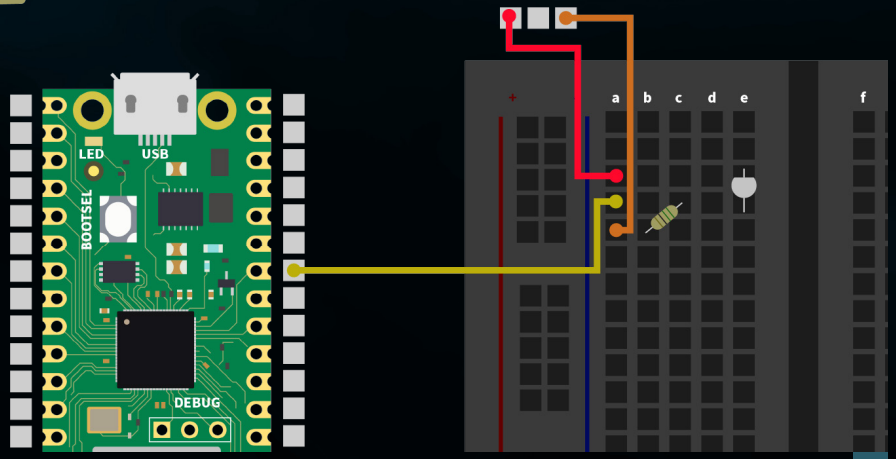
FOTODIODAS: Fotodiodas yra specialus puslaidininkis, kuris reaguoja į į jį krintančią šviesą generuodamas elektros srovę. Įsivaizduokite fotodiodą kaip mažą saulės bateriją: kai ant jo krinta šviesa, jis paverčia ją elektros signalu. Kuo daugiau šviesos patenka į fotodiodą, tuo stipresnis tampa signalas.

Fotodiodai yra labai jautrūs ir gali aptikti net ir nedidelį šviesos kiekį, todėl jie idealiai tinka projektams, kuriuose svarbu išmatuoti šviesos ryškumą ar buvimą. Pavyzdžiui, jie gali būti naudojami automatiniuose ryškumo valdikliuose, šviesos jutikliuose arba kaip apšvietimo valdymo sistemos dalis.

Trumpai tariant, fotodiodai yra veiksmingi šviesos detektoriai, leidžiantys elektroniniams prietaisams protingai reaguoti į aplinkos apšvietimo pokyčius.

Pirmiausia prijunkite fotodiodą per duonos plokštę taip. Atkreipkite dėmesį, kad čia taip pat reikia naudoti rezistorių. Čia naudokite 100 kΩ (rudos-juodos-geltonos spalvos) rezistorių.

RASPBERRY PI PICO	FOTODIODAS
3V3	Teigiamas katodas
GP28/A2	Neigiamas anodas



DĖMESIO! Šiame projekte būtina nustatyti relės jungiklį į OFF, o šviesos diodų jungiklį į ON.

SANTRAUKA: Naudodami fotodiodą matuojame aplinkos ryškumą ir reguliuojame keturių šviesos diodų ryškumą. Šviesos diodų intensyvumas keičiasi priklausomai nuo fotodiodo užfiksuotos šviesos: tamsesnėje aplinkoje šviesos diodai šviečia ryškiau ir atvirkščiai. Geriausia naudoti žibintuvėlį, kad pasiektumėte geriausią įmanomą rezultatą.

```
from machine import Pin, ADC
import neopixel
import utime

# Neopixel setup
NUM_LEDS = 4
PIXEL_PIN = 1
np = neopixel.NeoPixel(Pin(PIXEL_PIN), NUM_LEDS)

# Photodiode setup on ADC pin GP28 (A2)
fotodiode = ADC(2)

# Conversion function for brightness values of the photodiode into a suitable
brightness for the LEDs
def brightness_from_light(sensor_value):
    # Minimum and maximum sensor value
    min_sensor_value = 400
    max_sensor_value = 10000

    # Invert the sensor value within the actual range
    normalized_value = max_sensor_value - sensor_value + min_sensor_value

    # Scale the inverted value to a brightness range (0.05 to 0.5)
    # Adjust the scaling: Divide by (max_sensor_value - min_sensor_value)
    return max(0.05, min(0.5, normalized_value / (max_sensor_value - min_sensor_
value) * 0.45 + 0.05))

def update_leds(brightness):
    color = (255, 255, 255)  # White
    dimmed_color = tuple([int(c * brightness) for c in color])
    for i in range(NUM_LEDS):
        np[i] = dimmed_color
    np.write()


while True:
    # Read the sensor value from the photodiode
    light_value = fotodiode.read_u16()
    print(light_value)

    # Calculate the brightness based on the sensor value
    brightness = brightness_from_light(light_value)

    # Update the LEDs with the new brightness
    update_leds(brightness)

    # Waiting time to reduce the load on the CPU and for smoother brightness
transitions
    utime.sleep(0.5)
```

Šviesos diodų ir fotodiodo inicijavimas

Fotodiodo matavimas ir ryškumo valdymas

5.8 RGB LED VALDYMAS

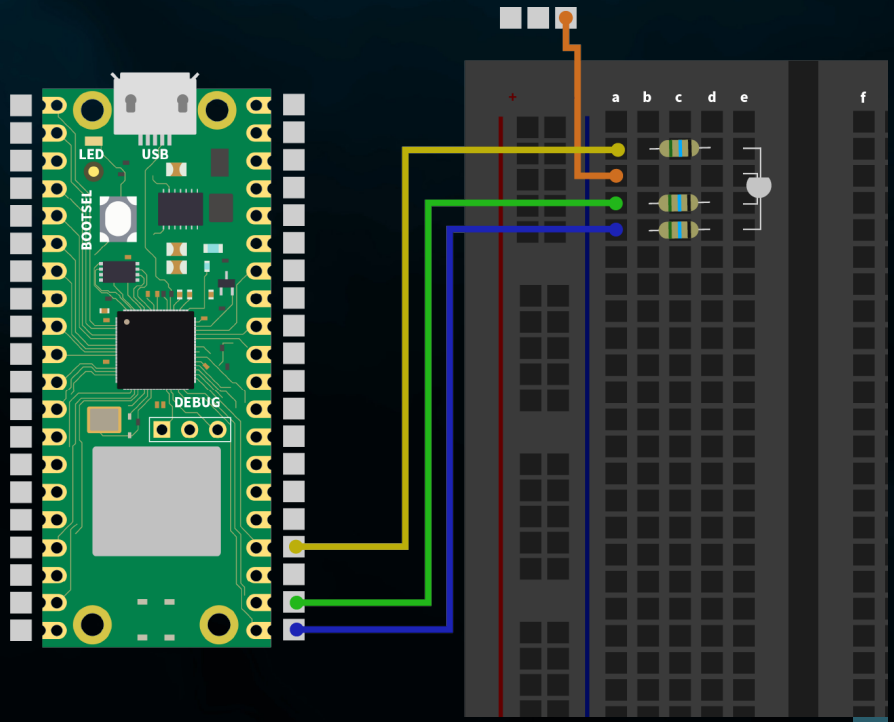
Aštuntajame ir paskutiniame elektronikos serijos projekte daugiausia dėmesio skirsime RGB šviesos diodų spalvų valdymui naudojant integruotus "Explorer" plokštės mygtukus. RGB šviesos diodai - tai specialūs šviesos diodai, kurie derina raudoną, žalią ir mėlyną (RGB) šviesą, kad galėtų atvaizduoti įvairias spalvas. Atskirai reguliuodami kiekvieno spalvinio komponento intensyvumą, galime sukurti beveik bet kokią spalvą.

Šiame projekte RGB šviesos diodą prijungiame prie duonos plokštės ir naudojame esamus mygtukus šviesos diodo spalvoms valdyti. Kiekvienam mygtukui priskiriama spalva (raudona, žalia, mėlyna).

RGB LED: RGB šviesos diodas sujungia raudoną, žalią ir mėlyną spalvas viename šviesos taške. Keičiant kiekvienos iš trijų spalvų ryškumą, galima sukurti beveik bet kokią spalvą. Tai daroma impulsų pločio moduliacija (PWM), kuri valdo kiekvienos spalvos intensyvumą. Taigi RGB šviesos diodai, turintys tik tris spalvas, leidžia sukurti platų spalvų spektrą, idealiai tinkantį spalvingiems apšvietimo projektams.

Pirmiausia prijunkite RGB šviesos diodą prie plokštės taip. Atkreipkite dėmesį, kad čia kiekvienam iš trijų spalvų kanalų taip pat reikia nuoseklaus rezistoriaus. Čia turėtumėte naudoti 56 Ω rezistorių (žalias-mėlynas-juodas).

RASPBERRY PI PICO	RGB-LED
GP18	Pirmasis smeigtukas
GND	Antrasis kaištis
GP17	Trečiasis kaištis
GP16	Ketvirtasis kaištis



DĖMESIO! Šiam projektui būtina nustatyti, kad TFT jungiklis būtų išjungtas (OFF), o mygtukai - įjungti (ON).

SANTRAUKA: Trys RGB šviesos diodo spalvų kanalai (raudona, žalia ir mėlyna) įjungiami ir išjungiami mygtukais (kairėje, viršuje ir dešinėje).

```
from machine import Pin
import utime

# Initialize the LED pins
red_led = Pin(18, Pin.OUT)
green_led = Pin(17, Pin.OUT)
blue_led = Pin(16, Pin.OUT)

# Initialize the button pins
button_red = Pin(15, Pin.IN, Pin.PULL_UP)
button_green = Pin(10, Pin.IN, Pin.PULL_UP)
button_blue = Pin(11, Pin.IN, Pin.PULL_UP)

# Save states of the LEDs
red_state = False
green_state = False
blue_state = False

def toggle_led(led, state):
    led.value(state)

while True:
    # Check the status of the red button
    if button_red.value() == 0:
        red_state = not red_state
        toggle_led(red_led, red_state)
        utime.sleep(0.2)  # Entprellung

    # Check the status of the green button
    if button_green.value() == 0:
        green_state = not green_state
        toggle_led(green_led, green_state)
        utime.sleep(0.2)  # Entprellung

    # Check the status of the blue button
    if button_blue.value() == 0:
        blue_state = not blue_state
        toggle_led(blue_led, blue_state)
        utime.sleep(0.2)  # Debouncing
```

Šviesos diodo ir mygtukų inicijavimas

Mygtukų testavimas ir šviesos diodo valdymas

6. INFORMAVIMO IR GRĄŽINIMO ĮSIPAREIGOJIMAI

MŪSŲ INFORMAVIMO IR GRĄŽINIMO ĮSIPAREIGOJIMAI PAGAL VOKIETIJOS ELEKTROS IR ELEKTRONINĖS ĮRANGOS ĮSTATYMĄ (ELEKTROG)

SIMBOLIS ANT ELEKTROS IR ELEKTRONINĖS ĮRANGOS: 
Šis perbrauktas šiukšlių konteineris reiškia, kad elektros ir elektroniniai prietaisai nepriklauso buitinėms atliekoms. Senus prietaisus turite atiduoti į surinkimo punktą. Prieš atiduodami turite atskirti panaudotas baterijas ir akumuliatorius, kurie nėra senojo prietaiso priedėlyje.

GRĄŽINIMO GALIMYBĖS:
Kaip galutinis naudotojas, įsigydami naują prietaisą, galite nemokamai grąžinti seną prietaisą (kuris iš esmės atlieka tą pačią funkciją kaip ir iš mūsų įsigytas naujas prietaisas), kad jis būtų sunaikintas. Mažus prietaisus, kurių išoriniai matmenys neviršija 25 cm, galima išmesti įprastu buitiniu būdu, neatsižvelgiant į tai, ar įsigijote naują prietaisą.

GALIMYBĖ GRĄŽINTI MŪSŲ ĮMONĖS BUVEINĖJE DARBO VALANDOMIS:
SIMAC Electronics GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn

GRĄŽINIMO GALIMYBĖ JŪSŲ VIETOVĖJE:
Mes jums atsiųsime siuntinio antspaudą, su kuriuo galėsite mums nemokamai grąžinti prietaisą. Norėdami tai padaryti, susisiekite su mumis el. paštu service@joy-it.net arba telefonu.

INFORMACIJA APIE PAKUOTĘ:
Saugiai supakuokite seną prietaisą, kad galėtumėte jį transportuoti. Jei neturite tinkamos pakavimo medžiagos arba nenorite naudoti savo, susisiekite su mumis ir mes atsiųsime jums tinkamą pakuotę.

7. PARAMA

Taip pat esame pasirengę padėti jums ir po pirkimo. Jei liko neatsakyta į klausimus ar iškilo problemų, mes taip pat esame pasiekiami el. paštu, telefonu ir bilietų palaikymo sistema.

E-Mail: service@joy-it.net
Ticket-System: http://support.joy-it.net
Telefonas: +49 (0)2845 9360 – 50 (pirmadieniais-ketvirtadieniais: 09:00-17:00 arba laikrodis, penktadieniais: 09:00-14:30 arba laikrodis)

Daugiau informacijos rasite mūsų svetainėje:

WWW.JOY-IT.NET