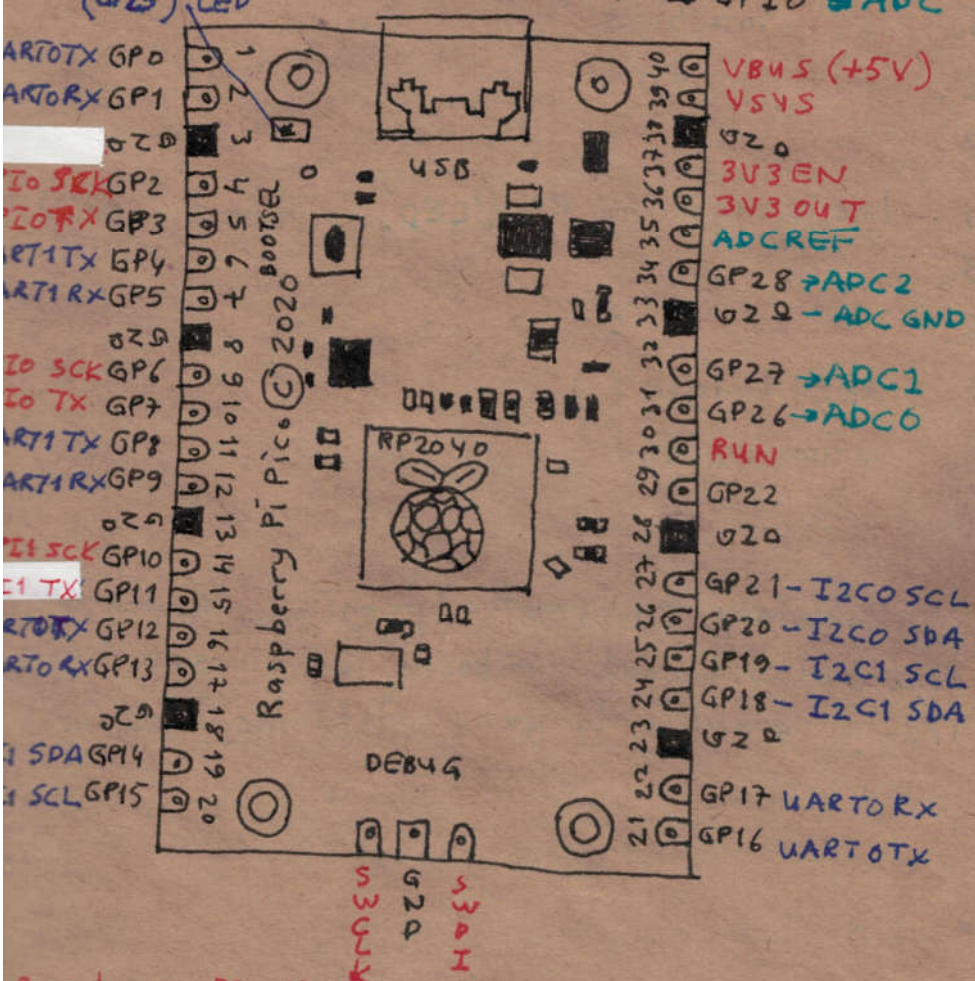
Raspberry pi Pico

Özellikleri

1. RP2040 mikrodenetleyici
2. Çift çekirdekli M0 işlemci 133MHz hızında çalışma
3. 264 kB SRAM program alanı
4. 2MB yerleşik Flash bellek
5. USB 1.1 desteği
6. Düşük güç tüketimli Uyku modları
7. USB üzerinden sürükle bırak program
8. 26 tane çok amaçlı GPIO pinleri
9. 2 x SPI, I2C, UART, 12 bit ADC 16 PWM kanalı
10. Çip üzerinde saat ve zamanlayıcı ve sıcaklık sensörü.

Raspberry Pi Pico Pinout



Raspberry Pi Pico W özellikleri:

- Kablosuz (802.11n), tekband 2.4 GHz
- Bluetooth 5.2
- Çip üzerinde sıcaklık sensörü
- Çip üzerinde saat ve zamanlayıcı

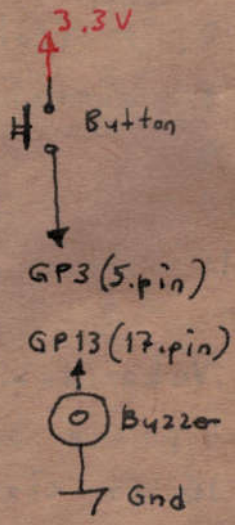
blink.py

```
1. from machine import Pin
2. import utime
3.
4. led = machine.Pin(25, machine.Pin.OUT)
5.
6. while True :
7.   led.toggle()
8.   utime.sleep(0.5)
9.
```

Yukarıdaki py (payton) programı Raspberry Pi Pico'nun üstündeki onboard led diyodunu (GP25) 1/2 sn aralıklarla yakıp söndürmeyi sağlar.

button_buzz.py

```
1. import machine
2. import utime
3.
4. buzzer = machine.Pin(13,machine.Pin.OUT)
5.buton=machine.Pin(3,machine.Pin.IN,machine.Pi
6. PULL_DOWN)
7. buzzer.value(1)
8. while True :
9.  if buton.value() == 1:
10.   buzzer.value(1)
11.   utime.sleep(0.15)
12.   print("Butona basıldı")
13.  else:
14.   buzzer.value(0)
15.   utime.sleep(0.15)
16.   print("Butona basılı değil")
```



Yukarıdaki program GP13 pinine bağlı olan buzzer
GP3 pinine bağlı butona basılınca öttüren ve shel
kısmına "Butona basıldı" yazan, butondan elimizi çe
kince buzzeri susturup "Butona basılı değil yaz

int_isi_sens.py

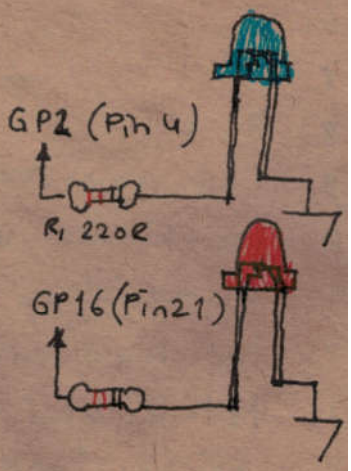
```
1. import machine
2. import utime
3. sicaklik_sensoru = machine.ADC(4)
4. sicaklik_cevirme = 3.3 /(65535)
5. while True:
6. okunan_sicaklik=sicaklik_sensoru.read_u16()
7. * sicaklik_cevirme
8. sicaklik = 27-(okunan_sicaklik-0,706)/
9. 0.001721
10. print (sicaklik)
11  utime.sleep (1)
```

Yukarıdaki program Raspberry Pi Pico içindeki ADC 4.pinine bağlı olan sıcaklık sensöründen aldığı 16 bit çözünürlükte olan değeri sicaklik_cevirme fonksiyonu ile shell'de gösterir.(1sn aralıklarla)

shell
27.51254
27.98069
27.51254

flip-FLOP.py

```
1. from machine import Pin
2. import utime
3.
4. led1 = Pin(2, Pin.OUT)
5. led2 = Pin(16, Pin.OUT)
6.
7. LED_state1 = True
8. LED_state2 = False
9.
10. while True:
11.     led1.value(LED_state1)
12.     led2.value(LED_state2)
13.     utime.sleep(0.5)
14.     led1.value(LED_state2)
15.     led2.value(LED_state1)
16.     utime.sleep(0.5)
```



Yukarıdaki program GP2 ve GP16 pinlerine bağlı olan led'leri (kırmızı ve mavi led)

CAKAR.py

```
1.  import machine
2.  import utime
3.
4.  led_mavi = machine.Pin(2, machine.Pin.OUT)
5.  led_kirmizi = machine.Pin(16, machine.Pin.OUT)
6.
7.  while True:
8.  # 1
9.  led_mavi.value(1)
10. led_kirmizi.value(0)
11. utime.sleep(0.05)
12. led_mavi.value(0)
13. led_kirmizi.value(1)
14. utime.sleep(0.05)
15. # 2
16. led_mavi.value(1)
17. led_kirmizi.value(0)
18. utime.sleep(0.05)
19. led_mavi.value(0)
20. led_kirmizi.value(1)
21. utime.sleep(0.05)
22. # 3
```

```
23. led_mavi.value(1)
24. led_kirmizi.value(0)
25. utime.sleep(0.05)
26. led_mavi.value(0)
27. led_kirmizi.value(1)
28. utime.sleep(0.05)
29. # 4
30. led_mavi.value(1)
31. led_kirmizi.value(0)
32. utime.sleep(0.05)
33. led_mavi.value(0)
34. led_kirmizi.value(1)
35. utime.sleep(0.05)
36. # 5
37. led_mavi.value(1)
38. led_kirmizi.value(0)
39. utime.sleep(0.05)
40. led_mavi.value(0)
41. led_kirmizi.value(1)
42. utime.sleep(0.05)
43. # 6
44. led_mavi.value(1)
```

```
45. led_kirmizi.value(0)
46. utime.sleep(0.05)
47. led_mavi.value(0)
48. led_kirmizi.value(1)
49  utime.sleep(0.5)
```

Yukarıdaki program GP2 (Pin 4)'e bağlı mavi led ve GP 16 (Pin 21)'e bağlı kirmizi ledi 6 defa arka arkaya 50 msnı aralıklarla yakıp söndürdükten sonra 1/2 sn kirmizi ledi yakıp söndürür ve tekrar 6 defa 50 ms aralıklarla yakıp söndürür. Bu işlemi devamlı olarak yapar.

buzzer.py

```
1.  from machine import Pin
2.  import utime
3.
4.  buzzer = machine.Pin(13,Pin.OUT)
5.  buzzer.value(1)
6.
7.  while True:
8.
9.      buzzer.value(1)
10.     utime.sleep(1)
11.     buzzer.value(0)
12.     utime.sleep(1.5)
13.
```

Yukarıdaki program buzzer pinini GP13 (pin 17) 1sn öttürdükten sonra 1.5sn susturur. Sonra tekrar 1sn öttürüp 1.5sn susturur. Bu islem devamlı tekrarlanır.

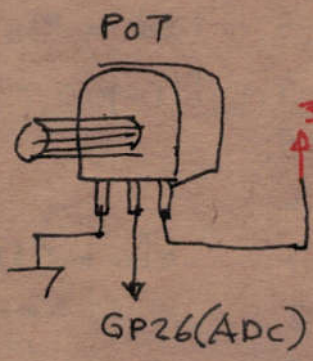
buton_led.py

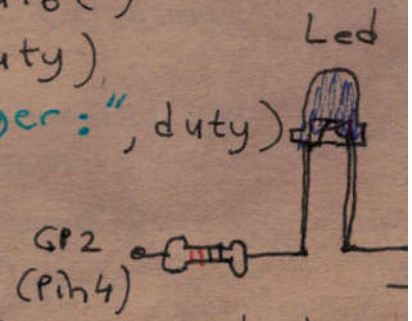
```
1. import machine
2. import utime
3.
4. led1 = machine.Pin(2, machine.Pin.OUT)
5. led2 = machine.Pin(16, machine.Pin.OUT)
6. buton = machine.Pin(3, machine.Pin.IN, machine.
7. Pin.PULL_DOWN)
8. led1.value(1)
9. led2.value(0)
10. print("led1 yandı")
11.
12. while True:
13.   if buton.value() == 1:
14.     led1.toggle()
15.     led2.toggle()
16.     utime.sleep(0.3)
17.     print("Butona basıldı")
```

Yukarıdaki programda GP3 (Pin5)'e bağlı butona basılmadan önce mavi ledi yakar. Butona basılınca mavi led söner, kırmızı led yanar. 0.3 sn buton sıçraması için bekler.

pot_deger.py

```
1. from machine import Pin, PWM, ADC
2. import utime
3.
4. pwm = PWM(Pin(2))
5. adc = ADC(Pin(26))
6.
7. pwm.freq(1000)
8.
9. while TRUE:
10.     duty = adc.read_u16()
11.     pwm.duty_u16(duty)
12.     print("okunan deger:", duty)
13.     utime.sleep(0,1)
```

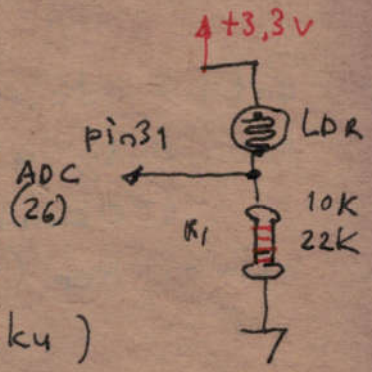




Yukarıdaki program ADC0 pin(31)'e bağlı olan Pot orta bacağından okuduğu 16 bit çözünürlükteki değere göre PWM pinine bağlı GP2 (pin4) ol mavi ledi duty değerine göre az parlak ya d daha çok parlak olarak yakar. Ayrıca shell kısmında okunan değer 0 ile 65535 arasında gösterilir.

LDR.py

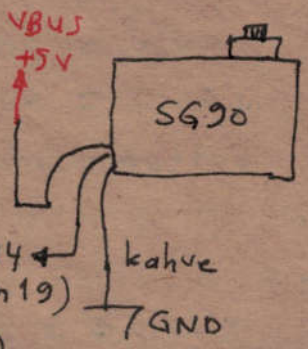
```
1.  from machine import ADC, Pin
2.  import utime
3.  led = Pin(2, Pin.OUT)
4.  ldr_pin = ADC(26)
5.
6.  while TRUE:
7.      oku = (ldr_pin.read_u16()/65535)*100
8.
9.      if oku > 4.00 :
10.         led.value(0)
11.     else
12.         led.value(1)
13.
14.     print("Deger :", oku)
15.     utime.sleep(0.5)
16
```



Yukarıdaki program ADC0(pin 31)'e bağlı olan LDR pininden okunan değer 4'ün üstünde iken yani ortam aydınlıkta ise ledi söndürür. Eğer LDR pininden okunan değer 4.00'ın altına gelirse yani ortam karanlık ise led'i yakar.

Servo.py

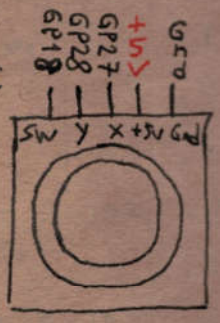
```
1. from machine import Pin, PWM
2. import utime
3.
4. servo = PWM(Pin(14))
5.
6. servo.freq(50)
7.
8. while TRUE:
9.  servo.duty_u16(1350)
10. print("Servo 0 derecede")
11. utime.sleep(2)
12. servo.duty_u16(8200)
13. print("Servo 180 derecede")
14.  utime.sleep(2)
```



Yukarıdaki program GP14 (Pin 19)'a bağlanmış olan SG90 servo motoru 2 sn aralıklarla 0° ve 180° arasında çalıştırır. Shell kısmında aşağıdaki gibi bir çıkış alınır.

Shell
Servo 0 derecede
Servo 180 derecede
Servo 0 derecede

Joystic.py




```
1. from machine import Pin, ADC
2. import utime
3.
4. xAxis = ADC(Pin(27))
5. yAxis = ADC(Pin(28))
6. button = machine.Pin(18, Pin.IN, Pin.PULL_UP)
7.
8. while TRUE:
9.     xValue = xAxis.read_u16()
10.    yValue = yAxis.read_u16()
11.
12.    buttonValue = button.value()
13.    print("xdeger:", xValue, "ydeger:", yValue,
14.    "Buton:", button.value)
15.    utime.sleep(0.5)
```

Yukaridaki programda joystic'in X axis için GP27 ve Y axis için GP28 'den okuduğu 16 bitlik ADC değerleri ve 18 pin (GP18) 'den okuduğu buton değeri 1/2 sn aralıklarla yazdırılır.

Shell

xdeger: 53004 ydeger: 50780 Buton: <bound method>
xdeger: 53116 ydeger: 50540 Buton: <bound method> 15

hiz.py

```
1. import machine
2.
3. a = machine.freq()
4. print ("pi pico hızı:", a)
5.
```

Yukarıdaki program raspberry pico pi'nin çalı
ma hızını göstermek için yazılmıştır. Bu prog-
ram ile machine hızı (frekansı) a değişkenine
aktarılır. Sonra print fonksiyonu ile bu hız
shell kısmından bir kereye mahsus yazdırılır

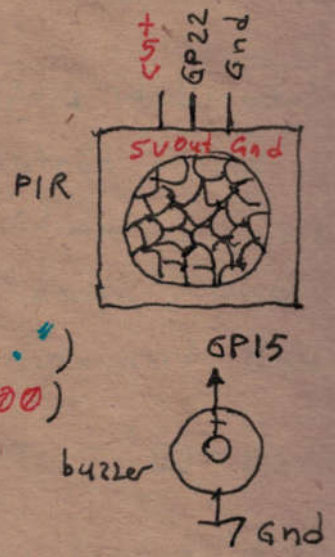
shell ⊠

```
pi pico hızı : 125000 000
```

Yazdırılan bu hız hz olarak anlaşılmalıdır. Yan
125000 000 milyon demek 125 Mhz anlamındadır

PIR_buzzer.py

```
1.  from machine import Pin, PWM
2.  import utime
3.
4.  PIR = Pin(22, Pin.IN, Pin.PULL_DOWN)
5.
6.  buzzer = PWM(Pin(15))
7.  buzzer.freq(2000)
8.  buzzer.duty_u16(0)
9.  while TRUE:
10.  if PIR.value() == 1:
11.      print("PIR çalışıyor...")
12.      buzzer.duty_u16(40000)
13.      utime.sleep(1)
14.      buzzer.duty_u16(0)
15.      else:
16.      print("Hareket bekleniyor...")
17.      utime.sleep(0.5)
```



Yukarıdaki program GP22 (Pin 29)'a bağlı PIR aktif olunca lojik 1 göndererek ekranda PIR çalışıyor yazar Ayrıca 1 sn buzzeri öttürür. GP15 (Pin 20)'ye bağlı buzzer dört defa öter. Sonra ekranda Hareket bekleniyor yazar. PIR hareket algılayınca "PIR çalışıyor" yazar 17

melody.py

```
1.  from machine import Pin, PWM
2.  from utime import sleep
3.  buzzer = PWM (Pin (15))
4.
5.  tones = {
6.  "B0" : 31,
7.  "C1" : 33,
8.  "CS1": 35,
9.  "D1" : 37,
10. "DS1": 39,
11. "E1" : 41,
12. "F1" : 44,
13. "FS1": 46,
14. "G1" : 49,
15. "GS1": 52,
16. "A1" : 55,
17. "AS1": 58,
18. "B1" : 62,
19. "C2" : 65,
20. "CS2": 69,
21. "D2" : 73,
22. "DS2": 78,
```

23. "E2" : 82,
24. "F2" : 87,
25. "FS2": 93,
26. "G2" : 98,
27. "GS2": 104,
28. "A2" : 110,
29. "AS2": 117,
30. "B2" : 123,
31. "C3" : 131,
32. "CS3": 139,
33. "D3" : 147,
34. "DS3": 156,
35. "E3" : 165,
36. "F3" : 175,
37. "FS3": 185,
38. "G3" : 196,
39. "GS3": 208,
40. "A3" : 220,
41. "AS3": 233,
42. "B3" : 247,
43. "C4" : 262,
44. "CS4": 277,
45. "D4" : 294,
46. "DS4": 311,
47. "E4" : 330,
48. "F4" : 349,

```
49. "FS4" : 370,
50. "G4"  : 392,
51. "GS4" : 415,
52. "A4"  : 440,
53. "AS4" : 466,
54. "B4"  : 494,
55. "C5"  : 523,
56. "CS5" : 554,
57. "D5"  : 587,
58. "DS5" : 622,
59. "E5"  : 659,
60. "F5"  : 698,
61. "FS5" : 740,
62. "G5"  : 784,
63. "GS5" : 831,
64. "A5"  : 880,
65. "AS5" : 932,
66. "B5"  : 988,
67. "C6"  : 1047,
68. "CS6" : 1109,
69. "D6"  : 1175,
70. "DS6" : 1245,
71. "E6"  : 1319,
72. "F6"  : 1397,
73. "FS6" : 1480,
74. "G6"  : 1568,
```

```
75.  "GS6" : 1661,
76.  "A6"  : 1760,
77.  "AS6": 1865,
78.  "B6" : 1976,
79.  "C7" : 2093,
80.  "CS7": 2217,
81.  "D7"  : 2349,
82.  "DS7": 2489,
83.  "E7"  : 2637,
84.  "F7"  : 2794,
85.  "FS7" : 2960,
86.  "G7"  : 3136,
87.  "GS7": 3322,
88.  "A7"  : 3520,
89.  "AS7" : 3729,
90.  "B7"  : 3951,
91.  "C8"  : 4186,
92.  "CS8" : 4435,
93.  "D8"  : 4699,
94.  "DS8" : 4978
95.  }
96.
97.  song = ["E5","G5","A5","P","E5","G5","B5",
98.  "A5","P","E5","G5","A5","P","G5","E5"]
99.
100. def playtone(frequency):
```

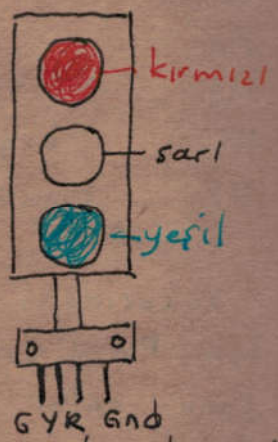
```
101.  buzzer.duty_u16(1000)
102.  buzzer.freq(frequency)
103.
104.  def bequiet():
105.  buzzer.duty_u16(0)
106.
107.  def playsong(mysong):
108.    for i in range(len(mysong)):
109.      if(mysong[i] == "p"):
110.        bequiet()
111.      else:
112.        playtone(tones[mysong[i]])
113.      sleep(0.3)
114.    bequiet()
115.    playsong(song)
```

Yukarıdaki program buzzer pini olan GP15 (20.pin) için PWM yardımıyla (pasif buzzer) üzerinden şarkı çalar. Şarkıyı bir kez daha çaldıktan sonra durur.

trafic_arac.py

```
1.  from machine import Pin
2.  import utime
3.
4.  led_k = Pin(10, Pin.OUT)
5.  led_s = Pin(11, Pin.OUT)
6.  led_y = Pin(12, Pin.OUT)
7.
8.  aracGecisSure = 5
9.  aracBeklemeSure = 5
10.
11. led_k.value(0)
12. led_s.value(0)
13. led_y.value(0)
14.
15. while TRUE:
16.
17.     led_y.value(1)  # Araçlara yeşil ışık yandı
18.     print("Araclara yeşil yanıyor, araçlar geçiyor.")
19.     utime.sleep(aracGecisSure)
20.     print("Yavaşla araçlara kırmızı yanacak.")
21.
22.     led_y.value(0)  # yeşil led 1/2 sn aralıklı
```

Trafik ışık modülü



```
23. utime.sleep(0.5) # olarak 4 defa yanıp söner.
24. led_y.value(1)
25. utime.sleep(0.5)
26. led_y.value(0)
27. utime.sleep(0.5)
28. led_y.value(1)
29. utime.sleep(0.5)
30. led_y.value(0)
31. utime.sleep(0.5)
32. led_y.value(1)
33. utime.sleep(0.5)
34. led_y.value(0)
35. utime.sleep(0.5)
36. led_y.value(1)
37. utime.sleep(0.5)
38. # Aynı İslemleri tekrar tekrar yazmak sıkıcı
39. led_y.value(0) #olduğu için bundan sonra
40. utime.sleep(0.5)# for döngüsü kullanacağız.
41.
42. led_s.value(1) # Sarı ışık yandı.
43. print("Dikkat araçlara kırmızı yanacak.")
44. utime.sleep(2) # 2sn yanar.
45. led_s.value(0) # Sarı ışık söndü.
46.
```

```
led_k.value(1) # Araçlara kırmızı yandı.
print("Araçlara kırmızı yanar, araçlar bekler.")
utime.sleep(aracBeklemeSure)
led_s.value(1) # Sarı ışık yandı.
print("Hazırlan araçlara yesil yanacak!")
utime.sleep(3)
led_k.value(0) # kırmızı ışık söner.
led_s.value(0) # Sarı ışık da söner.

# Yukarıdaki programda önce GP12'ye bağlı olan
#yeşil led yanıp söner ve tekrar 1/2 sn aralık-
#larla toplam 4 defa yanıp-söner. Sonra Sarı
#led GP11'e bağlı olan 2 sn yanıp söner. Daha
#sonra GP10'a bağlı kırmızı led yanar. Araç bek-
#leme süresi kadar 5 sn yanar. Kırmızı led
#yanarken sarı ışık yanar. 3 sn sonra kırmızı
#ve sarı led söner. Sonra tekrar GP12'ye bağlı
# olan yeşil led tekrar yanarak araç geçiş
# süresi kadar yandıktan sonra 4 defa yanıp
# söner.
```

ParkSensoru.p

```
1. from machine import Pin, PWM
2. import utime
3. trigger = Pin(8, Pin.OUT)
4. echo = Pin(9, Pin.IN)
5.
6. buzzer = PWM(Pin(15)) # PWM pini 15 (GP1
7. buzzer.freq(2000) # Frekans 2 kHz
8. buzzer.duty_u16(0) # PWM 0 gönderir
9.
10. led = Pin(2, Pin.OUT)
11.
12. def mesafe():
13.  trigger.low() # trigger'i 0 yap,
14.  utime.sleep_us(2) # 2 µsn bekle
15.  trigger.high() # trigger'i 1 yap
16.  utime.sleep_us(5) # 5 µsn bekle
17.  trigger.low() # trigger'i tekrar 0 yap
18.
19.  while echo.value() == 0: # sinyalin olmadigi
20.  SinyalKapali = utime.tick_us() # zaman sinyal ka
21.  while echo.value() == 1: # sinyalin oldug
22.  sinyalAcik = utime.tick_us() # zaman sinyal A
```

```
23.  gecenZaman = sinyalAcik - sinyalKapali
24.  uzaklik = (gecenZaman * 0.0343)/2
25.  print("uzaklik = ", uzaklik, "cm")
26.
27.  if uzaklik <=15 : #mesafe 15cm'den küçükse
28.  buzzer.duty_u16 (40000) #buzzeri öttür
29.  led.value (1)  #Led'i yak
30.  elif uzaklik <=25 :#mesafe 25cm'den küçükse
31.  buzzer.duty_u16(40000) #buzzeri öttür
32.  led.value(1)  # Led'i yak
33.  utime.sleep (0.1) # 100 msn bekle
34.  buzzer.duty_u16(0) # Buzzeri sustur
35.  led.value(0) # Led'i söndür
36.  utime.sleep(0.1) # 100 msn bekle
37.
38.  elif uzaklik <=35 : #mesafe 35'cm den küçükse
39.  buzzer.duty_u16(40000) # Buzzer'i öttür,
40.  led.value(1) # Led'i yak
41.  utime.sleep(0.2) # 200 msn bekle
42.  buzzer.duty_u16(0) # Buzzer'i sustur.
43.  led.value(0) # Led'i söndür.
44.  utime.sleep(0.2) # 200 msn bekle
```

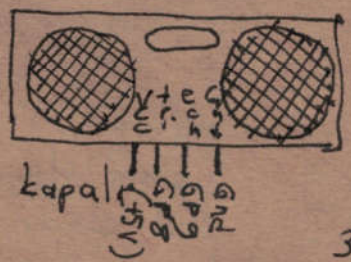
```
45.  elif uzaklik <= 50 : # mesafe 50cm'den küçükse
46.      buzzer.duty_u16(40000) # Buzzeri öttür.
47.      led.value(1) # Led'i yak
48.      utime.sleep(0.4) # 400 msn bekle
49.      buzzer.duty_u16(0) # Buzzeri sustur.
50.      led.value(0) # Led'i söndür.
51.      utime.sleep(0.4) # 400 msn bekle
52.
53.  elif uzaklik <= 80: # mesafe 80 cm'den küçükse
54.      buzzer.duty_u16(40000) # Buzzeri öttür.
55.      led.value(1) # Led'i yak
56.      utime.sleep(0.6) # 600 msn bekle
57.      buzzer.duty_u16(0) # Buzzer'i sustur.
58.      led.value(0) # Led'i söndür
59.      utime.sleep(0.6) # 600 msn bekle
60.
61.  elif uzaklik <=120: # mesafe 120cm'den küçükse
62.      led.value(1) # Led'i yak.
63.      utime.sleep(1) # 1 sn bekle
64.      led.value(0) # Led'i söndür.
65.      utime.sleep(1) # 1 sn bekle
66.  else: # mesafe 120 cm'den de büyükse
```

```
67.  buzzer.duty_u16(0) # Buzzeri sustur.
68.  led.value(0) # Led'i söndür.
69.
70.  while True:
71.      mesafe() # mesafeyi ölç.
```

Yukarıdaki program HCSR-04'ün trig ucunun GP8'e ve echo ucunun da GP9'a bağlanarak ölçüm yapıldığı ve çıkış olarak bir Led diyodun GP2'ye, bir buzzer'in ise (pasif) GP15 pinine bağlı olarak çalıştığı; mesafe yaklaştıkça Ledin ve buzzerin daha hızlı çalışmasını, mesafe arttıkça Ledin ve buzzer'in yavaş çalışmasını sağlar.

HCSR04-LCD.py

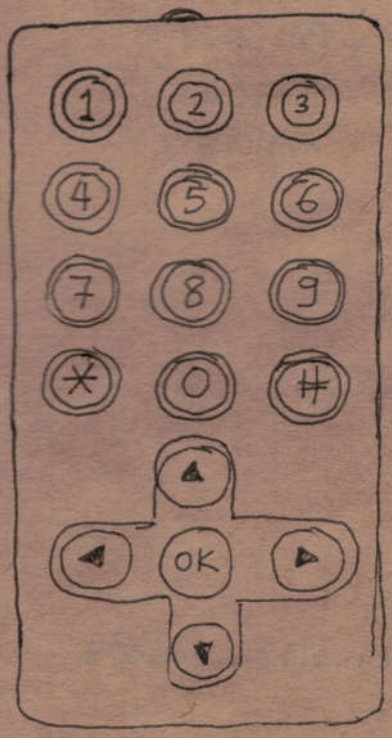
```
1. from machine import Pin, I2C
2. import utime
3. #HcSR04 Ultrasonic sensör ve LCD uygulaması
4. trigger = Pin(8, Pin.OUT) # sinyal gönderen pin
5. echo = Pin(9, Pin.OUT) # sinyal alan pin
6.
7. # - — — — LCD I2C için gereken kodlar - -- -
8. from lcd_api import LcdApi
9. from pico_i2c_lcd import I2cLcd
10. I2C_ADDR  = 0x27 # Lcd çalışmazsa 0x3F'yaz.
11. I2C_NUM_ROWS =2 # Lcd'nin satırı 2
12. I2C_NUM_COLS = 16 # Lcd'nin sütunu 16
13.
14. sda = Pin(20) # Lcd'nin SDA pini GP20'ye bağlı
15. scl = Pin(21) # Lcd'nin SCL pini GP21'e bağlı
16.
17. i2c = I2C(0, sda=sda, scl=scl, freq=400000)
18. lcd  = I2cLcd(i2C, I2C_ADDR, I2C_NUM_ROWS,
19. I2C_NUM_COLS)
20.
21. def mesafe():
22. trigger.low() # trig pini kapalı
```



```
23.  utime.sleep_us(2) # 2µsn bekle
24.  trigger.high() # triggeri açtık
25.  utime.sleep_us(5) #5µsn bekle
26.  trigger.low() # triggeri kapat
27.
28.  while echo.value()==0:#sinyal yoksa
29.    sinyalKapali =utime.ticks_us()
30.  while echo.value()==1:
31.    sinyalAcik = utime.ticks_us()
32.    gecenZaman = sinyalAcik - sinyalKapali
33.    uzaklik = (gecenZaman * 0.0343)/2
34.    print("UZAKLIK", uzaklik, "cm")
35.
36.    lcd.move_to(4,0) # LCD'nin 4.sütununa git
37.    lcd.putstr("UZAKLIK") # UZAKLIK yaz
38.    lcd.move_to(4,1) #LCD'nin 4.sütün 1.satıra
39.    lcd.putstr(str(uzaklik)) # Uzaklığı yaz.
40.    lcd.move_to(14,1) # LCD 14.sütun 1 satıra
41.    lcd.putstr("cm") # cm yaz.
42.
43.    while TRUE:
44.    mesafe() # mesafeyi ölç
45.    utime.sleep(1) # 1sn. bekle
46.    lcd.clear() # LCD'yi sil.
```

Yandaki sayfadaki program hem HCSR04 hemde LCD_I2C'nin birlikte çalıştığı, HCSR04'ün trig ucunun GP8'e ve echo ucunun da GP9'a bağlı olduğu; LCD'nin ise sda pininin GP20'ye scl ucunun ise GP21 bağlı olduğu çıkışında ise shell kısmında uzaklık = 25.673, LCD'de ise aynı şekilde UZAKLIK 25.673 şeklinde yazan programdır.

IR kodu okuyucu: Bu kodlar NEC kodu olup, farklı kumandalar için örneğin SONY. RC5 Philips gibi kodlar da kullanılır. TV, uydu gibi kumandalar için ayrı kodlar vardır.



IR KODU		BUTON
69	—	1
70	—	2
71	—	3
68	—	4
64	—	5
67	—	6
7	—	7
21	—	8
9	—	9
22	—	*
25	—	0
13	—	#
24	—	▲
28	—	OK
90	—	▶
8	—	◀
82	—	▼

32.

IR_RGB.py main.py

```
1. import time
2. import utime
3. from machine import Pin, freq, PWM
4. from ir_rx.print_error import print_error
5. from ir_rx.nec import NEC_8
6. # IR receiver pin ve RGB led tanımlanıyor.
7. rgb_k = Pin(4, Pin.OUT)
8. rgb_y = Pin(5, Pin.OUT)
9. rgb_m = Pin(6, Pin.OUT)
10. buzzer = Pin(13, Pin.OUT)
11. pin_ir = Pin(7, Pin.IN)
12.
13. Up = 24
14. Down = 82
15. Left = 8
16. Right = 90
17. Stop = 28
18.
19. def decodeKeyValue(data):
20.     return data
21.
22. def callback(data, addr, ctrl):
```

```
23. if data < 0 : # NEC gönderilen kod
24.   pass
25. else:
26.     print (data)
27.     if data == Up :
28.         ileri()
29.     elif data == Down :
30.         geri()
31.     elif data == Left :
32.         sol()
33.     elif data == Right :
34.         sag()
35.     elif data == Stop :
36.         dur()
37.
38.  def ileri():
39.    buzzer.value(1)
40.    rgb_k.value(0)
41.    rgb_y.value(1)
42.    rgb_m.value(0)
43.    utime.sleep(0.2)
44.    buzzer.value(0)
```

```
45.
46.  def geri():
47.      buzzer.value(1)
48.      rgb_k.value(1)
49.      rgb_y.value(1)
50.      rgb_m.value(1)
51.      utime.sleep(0.2)
52.      buzzer.value(0)
53.
54.  def sol():
55.      buzzer.value(1)
56.      rgb_k.value(0)
57.      rgb_y.value(0)
58.      rgb_m.value(1)
59.      utime.sleep(0.2)
60.      buzzer.value(0)
61.
62.  def sag():
63.      buzzer.value(1)
64.      rgb_k.value(1)
65.      rgb_y.value(0)
66.      rgb_m.value(0)
67.      utime.sleep(0.2)
68.      buzzer.value(0)
```

```
69.
70.  def dur ():
71.      buzzer.value(1)
72.      rgb_k.value(0)
73.      rgb_y.value(0)
74.      rgb_m.value(0)
75.      utime.sleep(0.2)
76.      buzzer.value(0)
77.
78.      ir = NEC_8(pin_ir, callback)
79.
80.      try:
81.          while True:
82.              pass
83.      except KeyboardInterrupt:
84.      ir.close()
```

#(main adlı program enerji verilince direkt olarak çalışı
Yukarıdaki program RGB led'in Rkırmızı pini GP04'e,
Ryeşil pini GP05'e ve Rmavi pini GP06'ya bağlandığı,
buzzer'in GP[illegible]'ye ve IR receiver modülün de GP07
ye bağlandığı zaman IR kumandadan ileri ▲ tuşuna
basılınca yeşil, (ok) stop tuşuna basılınca söndüğü. ba
arada buzzerin öttüğü, geri ▼ tuşuna basılınca beyaz
◀ geri tuşuna basınca mavi ve ▶ sag tuşuna basılınc
kırmızı led'in yanmasını sağlar.

hc06_buzzer.py

```
1.  from machine import Pin, UART, PWM
2.  import utime
3.
4.  uart = UART(0, 9600) # UART 9600 hızında
5.  led_mavi = Pin(2, Pin.OUT) # led mavi Pin2
6.  led_kirmizi = Pin(16, Pin.OUT) # led kirmizi Pin16
7.  buzzer = PWM(Pin(15)) # buzzer Pin15 PWM
8.  buzzer.freq(2000) # frekans 2kHz
9.  buzzer.duty_u16(0) # Buzzer sustu.
10.
11. while True:
12.   try:
13.
14.   if uart.any()>0: #Bluetooth veri gelirse
15.     data = uart.readline() # data'ya kaydet
16.
17.     if 'q' in data: # Gelen data "q" ise
18.       buzzer.duty_u16(40000) # Buzzeri öttür
19.       uart.write("*BUZZER ON*")
20.       utime.sleep(1) # 1sn BEKLE
21.       buzzer.duty_u16(0) # Buzzeri sustur.
22.       uart.write("BUZZER OFF*")
```

```
23.
24.     if '1' in data: # Gelen data "1" ise
25.         led_mavi.value(1) # mavi_led'i yak
26.         led_kirmizi.value(1)# kirmizi_led'i yak
27.         uart.write("*led_mavi ON*")
28.         uart.write("*led_kirmizi ON*")
29.
30.     if '0' in data: # Gelen data "0" ise
31.         led_mavi.value(0)
32.         led_kirmizi.value(0)
33.         uart.write("*led_mavi OFF*")
34.         uart.write("*led_kirmizi OFF*")
35.
36.     except: # Aksi bir durumda
37.         print("Hay Aksi Birşeyler Yanlış Gitti!")
```

Yukarıdaki program uart0 için Rx pininin GP0'a Tx pininde GP1'e bağlı olan HC06 modülü ile GP2 pinine bağlı mavi led ile GP16 pinine bağlı kirmizi ledi birlikte yakıp söndürür. Ayrıca GP15 pinine bağlı olan buzzer'i de öttürür. 1sn sonra buzzeri kendiliğinden söndürür. Herhangi bir Bluetooth terminal uygulaması ile çalışabilir. Yalnız Bluetoot taraması yapılarak HC-06 seçilmeli ve eşleştirme için "1234" yazılmalıdır. Bu yapıldıktan sonra HC-06'nın yanıp-sönen kırmızı lambasının devamlı olarak yanması beklenir. 98

hc06_lcd_i2c.py

```
1.  from machine import Pin, UART, I2C, PWM
2.  import utime
3.  uart = UART(0, 9600) # uart'o 9600 hızında
4.
5.  from lcd_api import LcdApi
6.  from pico_i2c_lcd import I2cLcd
7.
8.  I2C_ADDR = 0x27
9.  I2C_NUM_ROWS = 2
10. I2C_NUM_COLS = 16
11.
12. sda = Pin(20) # sda pini tanımlandı.
13. scl = Pin(21) # scl pini tanımlandı.
14.
15. i2c = I2C(0, sda=sda, scl=scl, freq=200000)
16. lcd = I2cLcd(i2c, I2C_ADDR, I2C_NUM_ROWS,
17. I2C_NUM_COLS)
18. led_mavi = Pin(2, Pin.OUT)
19. lcd_kirmizi = Pin(16, Pin.OUT)
20. buzzer = PWM(Pin(15))
21. buzzer.freq(2000)
22. buzzer.duty_u16(0) # Buzzer susta
```

```
23.
24. lcd.clear() # Lcd'yi sil.
25. lcd.move_to(4,0) #kursor 4.sütün 0.satıra
26. lcd.putstr("HC-06") # HC-06 yazar.
27. utime.sleep(1) # 1sn bekle
28. lcd.move_to(2,1) # kursor 2.sütun 1.satıra
29. lcd.putstr("Bluetooth")
30. utime.sleep(2) # 2 sn bekle
31.
32. while True:
33.  try:
34.
35.    if uart.any() > 0: # Eğer data gelirse
36.      data = uart.readline() # bunu kaydet.
37.
38.      if 'q' in data: # Eğer data q ise
39.       lcd.clear() # Ekranı temizle
40.       lcd.move_to(0,0) # kursor 0.sütun 0.satıra
41.       lcd.putstr("q datası geldi")
42.       uart.write("*q ok...")
43.       utime.sleep(1) # 1sn bekle.
44.       lcd.move_to(5,1) # kursor 5.sütün 1.satıra
45.       lcd.putstr("BUZZER") # BUZZER yazar
46.       buzzer.duty_u16(40000) # Buzzer öter. 40
```

```
47.         utime.sleep(1)  # 1sn bekle
48.         buzzer.duty_u16(0) # Buzzer sustu.
49.
50.     if '1' in data : # Eğer data 1 ise
51.         lcd.clear() # Lcd'yi sil.
52.         lcd.move_to(0,0) #kursor 0.sütun 0.satıra
53.         lcd.putstr("1 datası geldi")
54.         uart.write("*1 ok...")
55.         utime.sleep(1) # 1sn bekle
56.         lcd.move_to(5,1)#kursor 5.sütun 1 satıra
57.         lcd.putstr("ledler 1")
58.         led_mavi.value(1) # Mavi ledi yak.
59.         led_kirmizi.value(1) # kırmızı led'i yak.
60.
61.     if '0' in data : # Eğer data 0 ise
62.         lcd.clear() # Lcd'yi sil.
63.         lcd.move_to(0,0)#kursoru 0.satır 0.sütun
64.         lcd.putstr("0 datası geldi")
65.         uart.write("*0 ok...")
66.         utime.sleep(1) # 1sn bekle
67.         lcd.move_to(5,1)#kursor 5.sütun 1 satıra
68.         lcd.putstr("ledler 0")
69.         led_mavi.value(0)# Mavi Led'i söndür.
70.         led_kirmizi.value(0) # kırmızı Led'i söndür.
```

```
71.
72. except:
73.     print("Hay Aksi Birşeyler Yanlış Gitti!")
```

Yukarıdaki program HC-06'nın uart0'a bağlı olan Tx pini GP1'e ve Rx pini ise GP0'a, buzzer pini PWM pin GP15'e, mavi led GP2'ye, kırmızı led ise GP16'ya bağlandıktan sonra Lcd_I2C kütüphanesi yardımıyla önce ekranda "HC-06" ve alt satırında ise "Bluetooth" yazdıktan sonra Telefonumuza App Strol'den yüklenen serial Terminal uygulaması açılır. Daha sonra HC-06 taratılır ve eşleştirme yapılır. İlk olarak 1234 yazılarak eşleştirilir. Sonra uygulama açılarak Connected yazısı görülür. En sonund klavye ile "q" yazılıp gönder'e ▷ tıklanır. Ekranda ise 'q datası geldi' yazıp alt satırında 'BUZZER' yazıp 1sn buzzer öter. Aynı şekilde "1" yazılıp gönder'e ▷ tıklanır. Ekranda '1 datası geldi' yazıp alt satırında ise 'ledler 1' yazar. Kırmızı ve mavi ledler yanar. Yine klavyeden "0" yazılıp gönder'e ▷ tıklanır ise ekranda bu sefer '0 datası geldi' yazıp alt satırında 'ledler 0' yazar. Kırmızı ve mavi ledler söner. Bu şekilde gönderilen karakterler ile farklı işlemler yaptırılabilir.

IR_KOD_LCD.py

```
1. import utime
2. import time
3. from machine import Pin, freq, I2C, PWM
4. from ir_rx.print_error import print_error
5. from ir_rx.nec import NEC_8
6. from lcd_api import LcdApi
7. from pico_i2c_lcd import I2cLcd
8.
9. I2C_ADDR = 0x27
10. I2C_NUM_ROWS = 2
11. I2C_NUM_COLS = 16
12.
13. sda = Pin(20)
14. scl = Pin(21)
15.
16. i2c = I2C(0, sda=sda, scl=scl, freq 200000)
17. lcd = I2cLcd(i2c, I2C_ADDR, I2C_NUM_ROWS,
18. I2C_NUM_COLS)
19. buzzer = Pin(13, Pin.OUT) # buzzer pin 13'e bağlı
20. pin_ir = Pin(7, Pin.IN) # IR modül pin7'ye bağlı
21.
22. lcd.clear() # Lcd'yi sil.
```

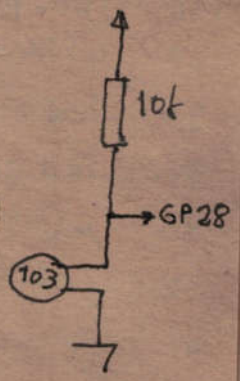
```
23.  lcd.move_to(0,0) # Lcd kursoru 0.sütun, 0.satıra
24.  lcd.putstr("Tamer CAGLAK")
25.  utime.sleep(1) # 1sn bekle
26.  lcd.move_to(0,1) # kursor 0.sütun 1.satıra
27.  lcd.putstr("IR kod okuyucu")
28.  utime.sleep(3) # 3 sn bekle
29.  lcd.clear() # Lcd'yi sil.
30.  lcd.move_to(0,0) #Kursoru 0.sütun 0.satıra
31.  lcd.putstr("IR KODU:")
32.
33.  def decodeKeyValue(data):
34.    return data
35.
36.  def callback(data, addr, ctrl):
37.    if data < 0 : # NEC protokolü kodu varsa
38.      pass
39.    else :
40.      print(data) # Data'yı shell kısmına yaz
41.      lcd.clear() # Lcd'yi sil.
42.      lcd.move_to(0,0)# kursor 0.sütun 0.satıra
43.      lcd.putstr("IR KODU")
44.      utime.sleep(0.1)
45.      lcd.move_to(0,1)# kursor 0.sütun 1.satıra
46.      buzzer.value(1) # Buzzeri öttür
47.      lcd.putstr(str(data))
48.      utime.sleep(0.3)
```

```
49.    buzzer.value (0) # Buzzer'ı sustur.
56.
51.  ir = NEC_8 (pin_ir, callback)
52.  # ir'nin NEC_8 kodu ile çalışacağı tanımlandı
53.
54.  try:
55.    while True:
56.      pass
57.  except keyboardInterrupt:
58.      ir.close()
```

Yukarıdaki program LCD-I2C bağlantısı için sda pini GP20 ve scl pini ise GP21'e bağlandıktan sonra buzzer pini GP13'e ve IR alıcı modül'de GP7'ye bağlanınca LCD ekranda önce "Tamer CAĞLAK" ve alt satırında "IR kod okuyucu" yazar. 3sn sonra ise LCD'yi silerek "IR KODU" yazar. NEC kumandadan herhangi bir tuşa basılınca altına kodun değerini yazar. Örneğin kumandadan '1' tuşuna basılınca IR KODU yazısının altına 69 kod bilgisini yazar. '0' tuşuna basılınca IR KODU yazısının altına ise 25 kod bilgisini yazar.

ntc10k_LCD_I2C.py

```
1.  import machine
2.  import utime
3.  import math
4.
5.  from machine import I2C, Pin
6.  from lcd_api import LcdApi
7.  from pico_i2c_lcd import I2cLcd
8.
9.  I2C_ADDR = 0x27
10. I2C_NUM_ROWS = 2
11. I2C_NUM_COLS = 16
12.
13. sda = Pin(20) # SDA pini GP20'ye bağlanacak.
14. scl = Pin(21) # SCL pini GP21'e bağlanacak.
15.
16. i2c = I2C(0, sda=sda, scl=scl, freq=200000)
17. lcd = I2cLcd(i2c, I2C_ADDR, I2C_NUM_ROWS,
18.  I2C_NUM_COLS)
19. thermistor = machine.ADC(28) # NTC ADC8'de
20. lcd.move_to(0, 0)
21. lcd.putstr("Tamer")
22. utime.sleep(2) # 2sn bekle
23. lcd_move_to(5, 1)
24. lcd.putstr("CAGLAK")
```



```
utime.sleep(2) # 2 sn bekle
lcd.clear(0)   # Lcd'yi sil.
lcd.move_to(0, 0)
lcd.putstr("NTC 10K") #
utime.sleep(2)
lcd.move_to(5, 1)
lcd.putstr("TERMOMETRE")
utime.sleep(2)
lcd.clear()

while True:
    temperature_value = thermistor.read_u16()
    Vr = 3.3 * float(temperature_value)/65535
    Rt = 10000 * Vr/(3.3 - Vr)
    tem = 1/(((math.log(Rt/10000))/3950)+
    (1/(273.15+25)))
    Cel = temp - 273.15
    Fah = Cel*1.8+32
    print('Celcius: %.2f C Fahrenheit: %.2f F'
    % (Cel, Fah))
    utime.sleep_ms(200)
    lcd.move_to(0, 0)
    string = " SICAKLIK \n "+str('{:.2f}'.format
    (Cel))+"C"
```

```
49. lcd.putstr(str(string)) # sıcaklığı yazdır.
50. utime.sleep(0.1)
```

Yukarıdaki program ADC28'e bağlı olan 10k NTC ile LCD_I2C'nin SDA pini GP20'ye ve SCL pini ise GP21'e bağlandığında sıcaklığı analog olarak 65535 çözünürlükte yani 16 bitlik hassaslıkta ölçer. Ekranda önce Tamer ve alt satırında CAĞLAK yazdıktan sonra 2 sn sonra 10k NTC ve alt satırında ise TERMOMETRE yazar. Daha sonra 2sn geçince tekrar ekran silinerek SICAKLIK ve alt satırında 33.85°C > 33.73 C şeklinde yazar, shell kısmında ise Celcius : 33.95 Fahrenheit : 92.98 şeklinde yazar.

Alarm_siren.py

```
1.  import machine
2.  import time
3.
4.  buzzer =machine.PWM(machine.Pin(15))
5.  led = machine.PWM(machine.Pin(16))
6.  led.freq(1000)
7.
8.  switch = machine.Pin(3,machine.Pin.IN)
9.
10. def noTone(pin):
11.   pin.duty_u16()
12.
13. def tone(pin,frequency):
14.   pin.freq(frequency)
15.   pin.duty_u16(30000)
16.
17. def interval_mapping(x, in_min, in_max, out_min,
18.  out_max):
19.  return (x-in_min)*(out_max-out_min)/(in_max
20. -in_min)+out_min
21.  def toggle(pin):
22.    global bell_flag
```

```
23.   bell_flag = not bell_flag
24.   print (bell_flag) # shel'de True yada False
25.   if bell_flag:
26.     switch.irq(trigger=machine.Pin.IRQ_
27.   FALLING , handler = toggle)
28.   else
29.     switch.irq (trigger=machine.Pin.IRQ_
30.   FALLING, handler = toggle)
31.
32. bell_flag = False
33. switch.irq (trigger=machine.Pin.IRQ_RISING,
34. handler = toggle)
35. while True:
36.   if bell_flag == True
37.     for i in range (0,100,2):
38.       led.duty_u16 (int(interval_mapping(i,0,100,0,
39.   65535)))
40.       tone(buzzer, int(interval_mapping(i,0,100,130,
41.   800)))
42.       time.sleep_ms(10)
43.   else:
44.     noTone(buzzer)
45.     led.duty_u16(0)
```

Yukaridaki program switche basinca buzzeri öttürür