PicoBricks

Yayım 0.1

R&D TEAM

16 Şub 2023

İçindekiler:

Not: Bu proje aktif gelişim aşamasındadır.

Tüyo: Eğer sorunuz varsa, Community üzerinden teknik destek ekibimizle iletişime geçebilirsiniz.

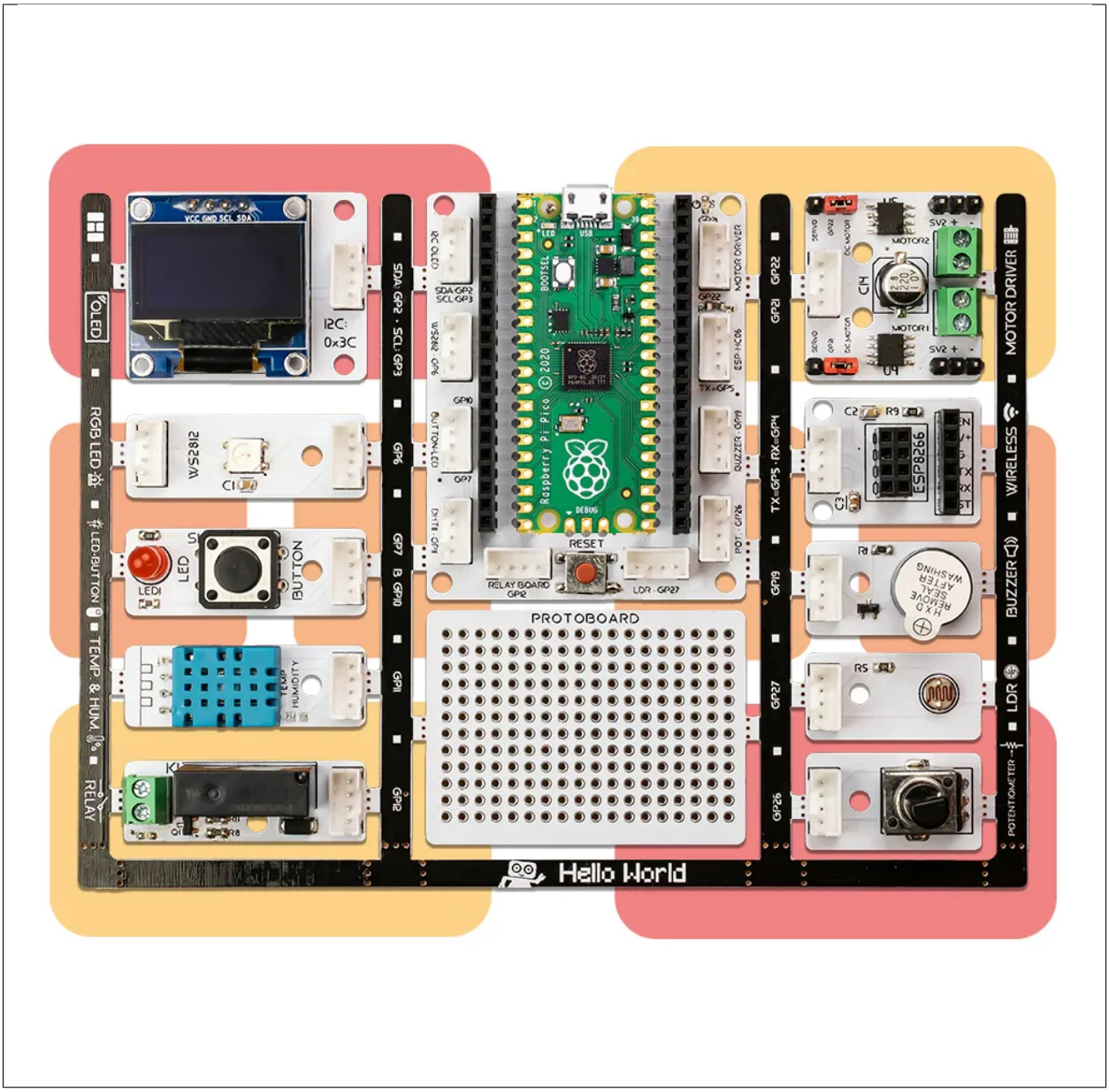
BÖLÜM 1

PicoBricks Nedir?

1.1 PicoBricks

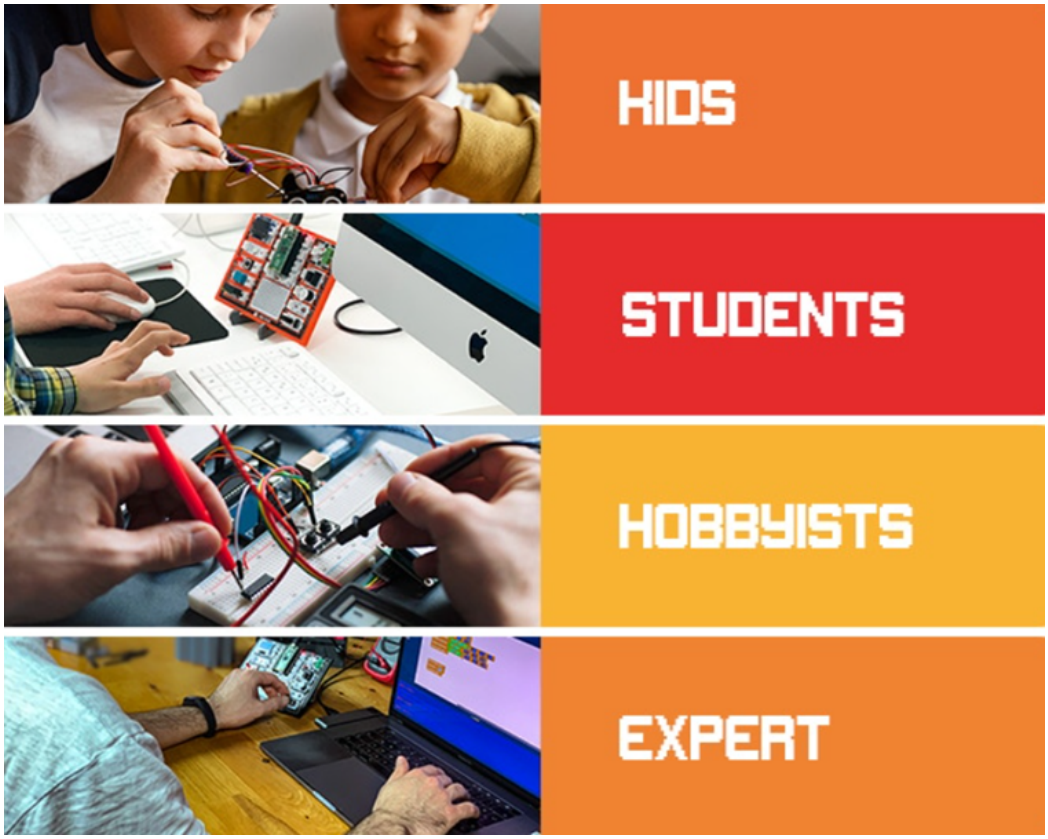
PicoBricks, maker projelerinde kullanılmak üzere tasarlanmış bir elektronik geliştirme `kartı + yazılımıdır`. On ayrılabilir modül dahil, PicoBricks çok çeşitli projeler oluşturmak için kullanılabilir. Ayrıca kendi modüllerinizi eklemek için kullanabileceğiniz bir protokol içerir!

PicoBricks, elektronik ve kodlama ile ilgilenen herkes içindir. Modüler donanım tasarımı, Scratch benzeri blok kodlama ortamı ve simülatör sayesinde daha önce deneyimi olmayan yeni başlayanlar, başlamayı kolay bulacaktır. Tecrübesi olanlar elektroniği daha derinlemesine inceleyebilir veya Python'da kodlamayı keşfedebilir. Ve en uzman makerlar bile PicoBricks ile fikirleri ne kadar çabuk keşfedebileceklerini ve prototipler oluşturabileceklerini takdir edeceklerdir. Diğer kartların aksine, PicoBricks her seviyedeki makerlar için inanılmaz bir esnekliğe sahiptir! Bricks IDE, farklı senaryolar için örnek kodlara sahiptir. `MicroBlocks` veya PicoBricks'in sürükle-bırak, blok kodlama oluşturucusu ile sıfırdan kahramana kodlamayı öğrenin. MicroBlocks, şimdiye kadar oluşturulmuş en kolay kodlama deneyimidir ve maker endüstrisinde yaygın olarak bilinir.



Her Seviye İçin Geliştirme Ortamı Desteği

Modüler donanım tasarımı, Scratch benzeri blok kodlama ortamı ve simülatör sayesinde daha önce deneyimi olmayan yeni başlayanlar, başlamayı kolay bulacaktır. Tecrübesi olanlar elektroniği daha derinlemesine inceleyebilir veya Python'da kodlamayı keşfedebilir. Ve en uzman yapımcılar bile Pico Bricks ile fikirleri ne kadar çabuk keşfedebileceklerini ve prototipler oluşturabileceklerini takdir edeceklerdir.PicoBricks, elektronik ve kodlamayla ile ilgilenen herkes içindir.



1.2 PicoBricks Modülleri Ayrılabilirdir

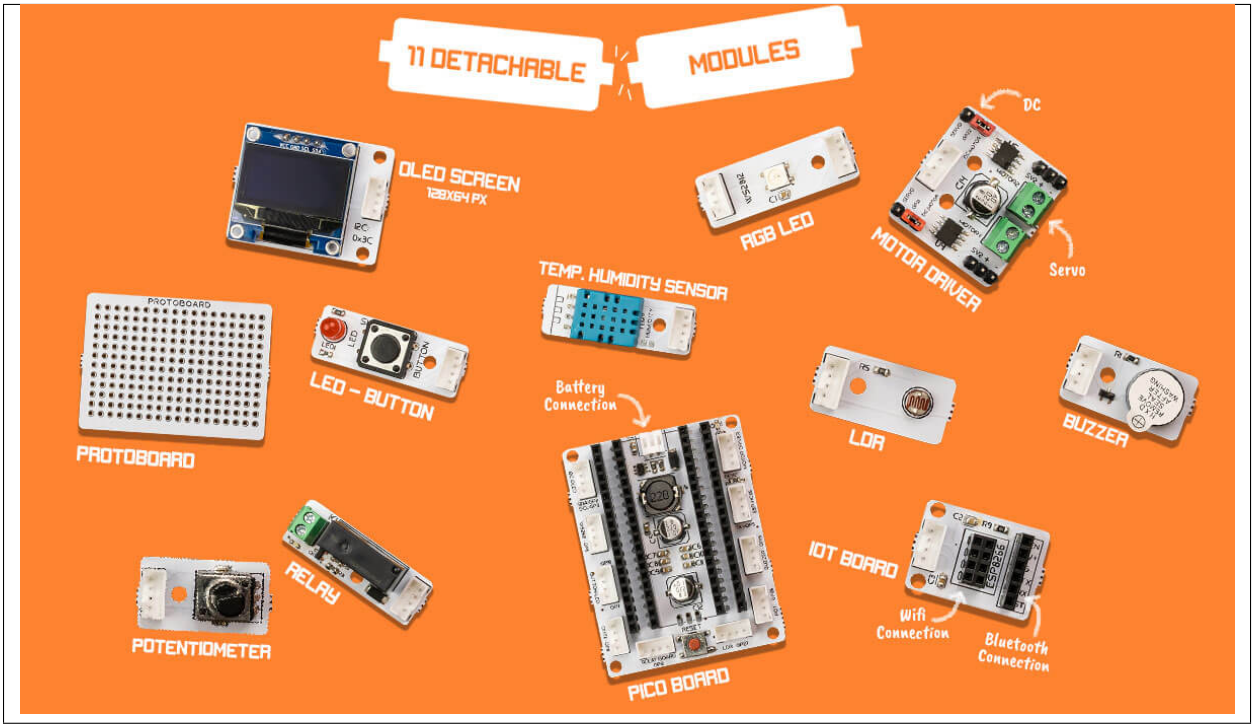
PicoBricks modüllerini keşfettikten sonra, bazılarını bir projede kullanmak isteyebilirsiniz. PicoBricks kompakt olmasına rağmen, her şeyi projenize dahil etmenize gerek yoktur. PicoBricks modüllerini ayırarak, standart, kullanımı kolay konektörlü kabloları kullanarak yalnızca ihtiyacınız olan modülleri bağlayabilirsiniz.

1.3 Ayrılabilen Modüller Tekrar Bir Araya Gelebilir

After dividing the Pico Bricks into modules, you can easily reassemble them again on the specially designed base plate.PicoBricks modüllerini ayırdıktan sonra özel olarak tasarlanmış plaka üzerinde tekrar kolayca monte edebilirsiniz.

BÖLÜM 2

Bricks



2.1 Pico Board

Raspberry Pi Pico gömülü sistem projelerinizde, prototiplemede kullanabileceğiniz bir mikrodenetleyicidir. Gücünü `RP2040 mikrodenetleyiciden` alan Raspberry Pi Pico, çift çekirdekli ARM Cortex M0+ işlemciye sahiptir. Düşük maliyetli Raspberry Pi Pico, düşük güç tüketimi ve yüksek performansı ile öne çıkıyor. Hem C/C++ hem de MicroPython ile programlanabilen Raspberry Pi Pico, her yaştan kullanıcıya hitap ediyor.

2.2 OLED Ekran

Minyatür OLED ekran modülleri, Raspberry Pi projelerinize küçük bir ekran eklemenin harika bir yoludur.

2.3 Sıcaklık ve Nem Sensörü

Çevredeki havayı ölçmek için kapasitif bir nem sensörü ve bir termistör kullanır. Veri pinine dijital bir sinyal verir. Kullanımı oldukça basittir ancak verileri almak için dikkatli zamanlama gerektirir.

2.4 LED-Buton

Işık yayan diyot (LED) modülleri, bir LED yayıcı zinciri içeren bağımsız cihazlardır. Boyutları sayesinde küçük LED modülleri, alanın sınırlı olduğu uygulamalarda mükemmel aydınlatma armatürleri oluşturur.

2.5 RGB LED

RGB LED modülleri çeşitli renklerde ışık yayabilir. Kırmızı, yeşil ve mavi olmak üzere üç ana renk karıştırılabilir ve parlaklığa göre her türlü rengi oluşturabilir, böylece devreyi kontrol ederek bir RGB LED'in renkli ışık yaymasını sağlayabilirsiniz.

2.6 Motor Sürücü

Motorların hızını, frekansı değiştirerek ayarlayan elektrikli ekipmanlardır. Hız kontrolünün yanı sıra üstün koruma, kontrol ve haberleşme özellikleri ile donatılmıştır.

2.7 LDR

LDR sensör modülü, ışığın yoğunluğunu algılamak için kullanılır. Işık olduğunda LDR'nin direnci ışığın yoğunluğuna göre düşük olacaktır. Işığın yoğunluğu ne kadar yüksek olursa, LDR'nin direnci o kadar düşük olur. Sensör, LDR'nin ışığa karşı hassasiyetini değiştirmek için ayarlanabilen bir potansiyometre düğmesine sahiptir.

2.8 Röle

Röleler, devreleri elektromekanik veya elektronik olarak açan ve kapatan anahtarlardır. Röleler, başka bir devredeki kontakları açıp kapatarak bir elektrik devresini kontrol eder.

2.9 Potensiyometre

Potansiyometre, elektrik akımının akışını kontrol etmek için direncin manuel olarak değiştirildiği 3 uçlu bir değişken direnç olarak tanımlanır.

2.10 ESP8266

ESP8266, tam TCP/IP yığınına sahip düşük bir `Wi-Fi mikrodenetleyici` dir. Farklı sensörlerle arayüz oluşturmak için birçok GPIO'ya (genel amaçlı giriş çıkış pinleri) sahiptir. İyi işlevselliği nedeniyle ESP8266, IoT ürünlerinin prototiplenmesinde çok kullanılır. Arduino ile çapraz işlevselliği, Arduino IDE ile programlamayı kolaylaştırır.

2.11 Buzzer

Buzzer, ses üreten bir sinyal cihazıdır.

2.12 Protoboard

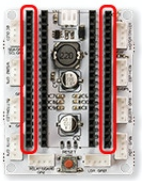
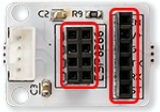
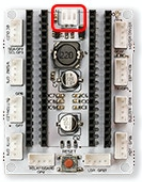
Protoboard, genellikle elektronik prototipleme için bir yapı temelidir.

BÖLÜM 3

Bağlantı Yöntemleri

PicoBricks ile devrelerinizi kablo karmaşası olmadan oluşturabilirsiniz. Aynı şekilde MicroBlocks programı ile kodlama yaparak kod karmaşasından kurtulabilirsiniz. Bu özelliği ile sıradan bir robotik kodlama eğitimi dersine başlarken karşılaşabileceğimiz birçok sorunu PicoBricks'in modüler yapısı sayesinde ortadan kaldırmış oluyoruz. Robotik kodlama eğitimine girdikten sonra PicoBricks üzerindeki modülleri gerekli yerlerden koparıp Raspberry Pi Pico'ya **Connector Kablolar** ile bağlayarak daha fazla proje yapma imkanı buluyoruz.

3.1 Lehimleme yok! Sınırsız bağlantı yöntemi var.

CONNECTION TYPE	WHAT CAN YOU CONNECT?
Header 	Sensors, extra motors, extra LEDs, anything that can be connected to Raspberry Pi Pico.
Easy Connector 	After the modules are detached, they are used for connecting modules to mainboard.
Terminals  	You can use them for the direct cable connections. DC Motor
Servo Headers 	Servo Motor
IoT Board 	ESP-01 Wi-Fi module and Bluetooth modules
Protoboard 	You can use protoboard with your advanced projects where you need soldering.
Battery Input 	To run your projects without being connected to a computer. Compatible with 2.5/5V inputs.
RGB LED Connection 	To connect more addressable LEDs.


NO SOLDERING!
NO SOLDER SMOKE
NO DANGER

BÖLÜM 4

Geliştirme Ortamları

4.1 MicroBlocks Blok Tabanlı Programlama

MicroBlocks, micro:bit, Adafruit Circuit Playground Express ve diğerleri gibi eğitici mikrodenetleyici kartlarıyla fiziksel hesaplamayı öğrenmek için ücretsiz, Scratch benzeri blok programlama dilidir. MicroBlocks canlı bir ortamdır. Bir bloğa tıklayın ve hemen panoda çalışır. Komutları deneyin. Sensör değerlerini gerçek zamanlı olarak görün ve grafiklendirin. Artık kodun derlenmesi ve indirilmesi için beklemenize gerek yok.

4.2 Başlangıç Seviyesi için Thonny(MicroPython) IDE

PicoBricks'in kalbi Raspberry Pi Pico'dur. Thonny, Raspberry Pi Pico ve PicoBricks'i kodlamak için harika bir seçimdir.

Tüyo: Eğer kodunuzu main.py adıyla keydederseniz, her boot yaptığınızda çalışacaktır.

4.3 Arduino IDE

Picobricks bize Arduino C ile kodlama fırsatı sunuyor. Picobricks'in kalbinde yer alan Raspberry Pi Pico'yu yaygın olarak kullanılan Arduino IDE ile kodlamaya başlamak oldukça kolay.

BÖLÜM 5

Başlangıç

5.1 Arduino IDE

5.1.1 Arduino IDE Kurulumu

PicoBricks size Arduino C ile kodlama imkânı sunuyor. Yaygın olarak kullanılan Arduino IDE ile Picobricks'in kalbinde yer alan Raspberry Pi Pico'yu kodlamaya başlamak oldukça kolaydır.

Arduino IDE 1.8.x kurulum dosyasını *Arduino Web Sitesinden <https://www.arduino.cc/en/software>* adresinden bilgisayarınıza indirin ve kurun.

Öncelikle Raspberry Pi Pico'yu Arduino IDE'ye eklemeniz gerekir. Arduino IDE'yi başlatın. Ardından ``Araçlar>Pano>Pano Yöneticisi'ne gidin.

1. alana "`Raspberry Pi Pico`" yazın. Bir süre bekledikten sonra Arduino Mbed OS RP2040 Boards seçeneğine tıklayın ve 2. alanda kur butonuna tıklayın.

Tüm bu kurulumlar sırasında sizden isteyeceği onayları kabul etmeniz gerekmektedir. Kurulum tamamlandığında ve kapat butonuna tıkladığınızda Pico'yu Arduino IDE'ye eklemiş olacaksınız.

5.1.2 Arduino IDE ile Kod Yazma ve Çalıştırma

Pico'yu Arduino IDE ile kodlamak istediğinizde, ilk seferinde `BOOTSEL tuşuna basılı tutarak` bilgisayarınıza bağlamanız yeterlidir.

Bu sayede Pico bootloader modunda bağlanacak ve bilgisayarınız tarafından harici bellek olarak tanınacaktır. Bootsel butonuna basılı tutarak Pico'yu bilgisayarınıza bağlayın. Pico'yu bilgisayarın flash belleği olarak gördükten sonra `Tools>Board>Arduino Mbed OS RP2040 boards> Raspberry Pi Pico` yolunu izleyerek kartınızı aktif ediniz.

Aşağıdaki 1 numaralı alana kodu yazın ve `File>Save` yolunu izleyerek `"Blink"` ismiyle bilgisayarınızın herhangi bir yerine kaydedin.

Kaydetme işleminden sonra 1. alandaki `"Yükle"` butonuna basarak kodu derleyip Pico'ya kaydetmemiz gerekiyor. En altta Done uploading yazısını gördüğümüzde Pico'da kodumuz çalışacak ve dahili LED 1 sn aralıklarla yanıp sönecektir. Önemli Not: Picobricks'i Arduino IDE ile kodlarken MicroPython veya MicroBlocks firmware'inden ilk geçişte


bricks
ide

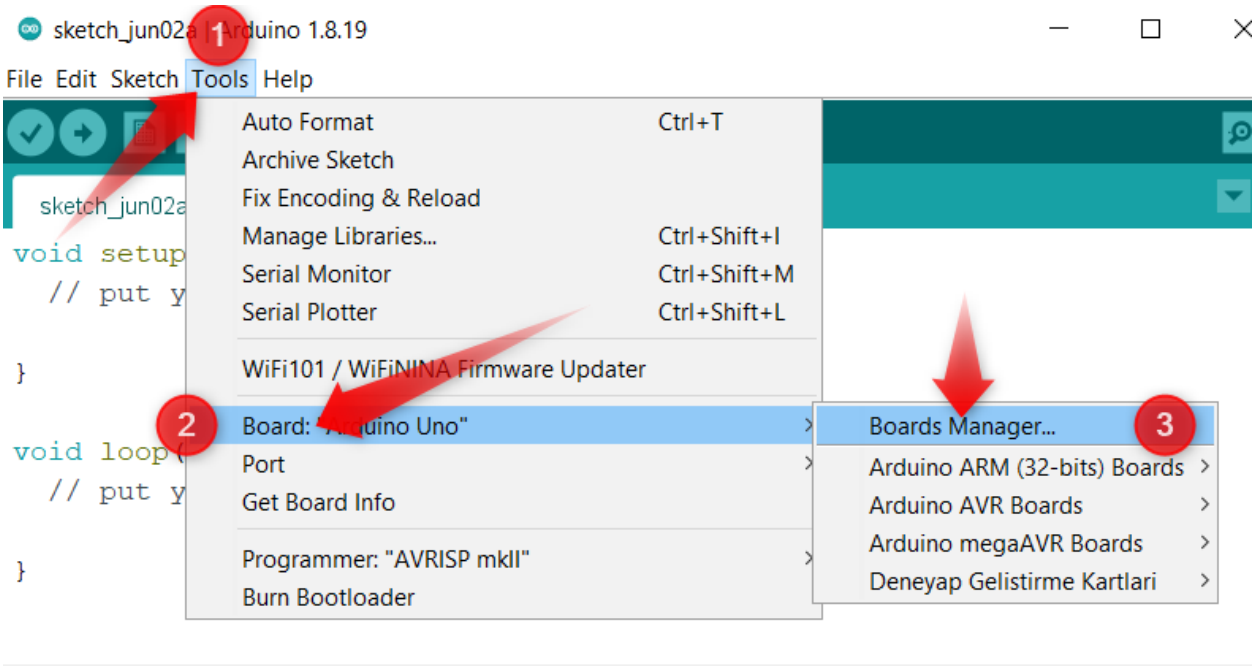

MicroPython

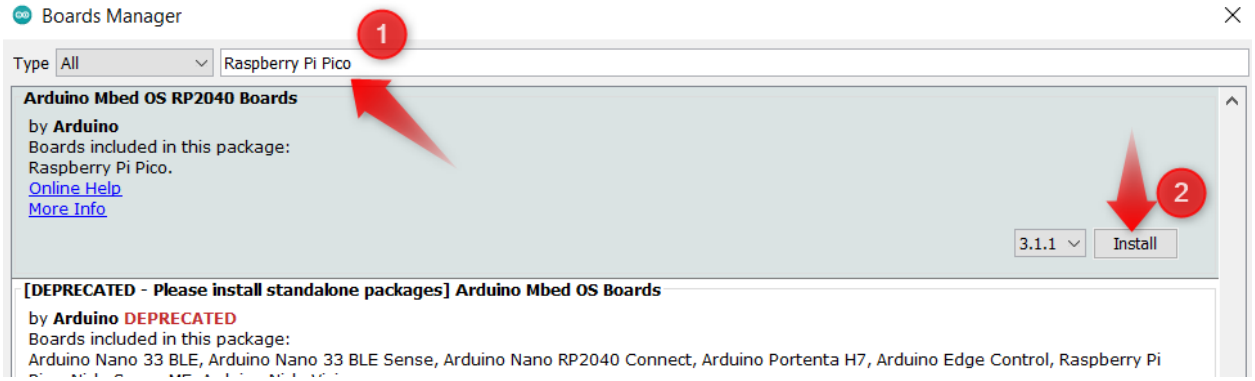

circuit
python

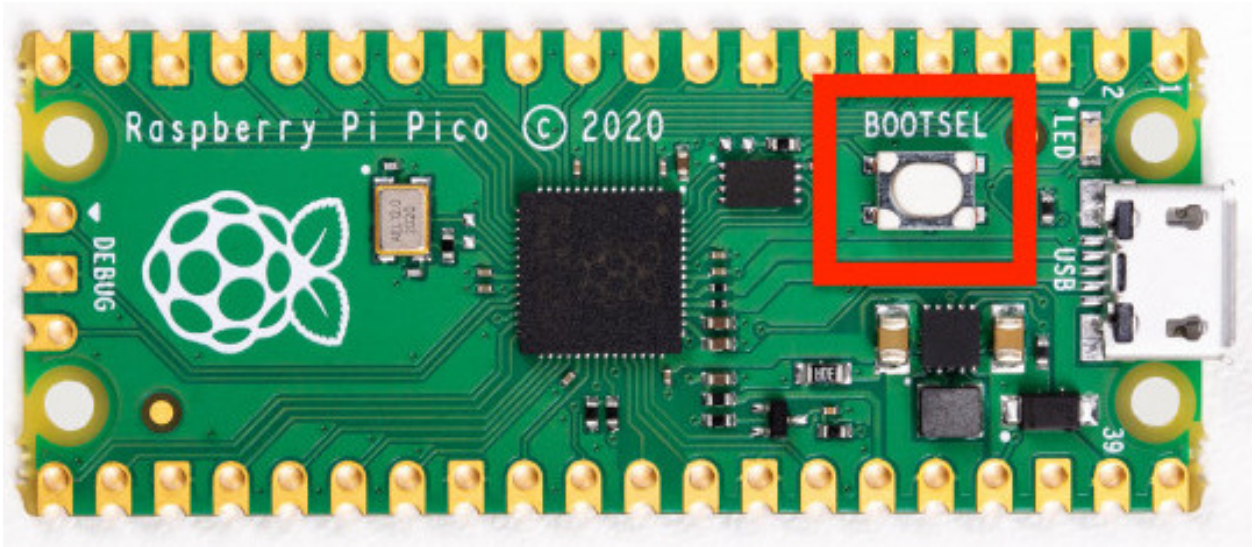

ARDUINO

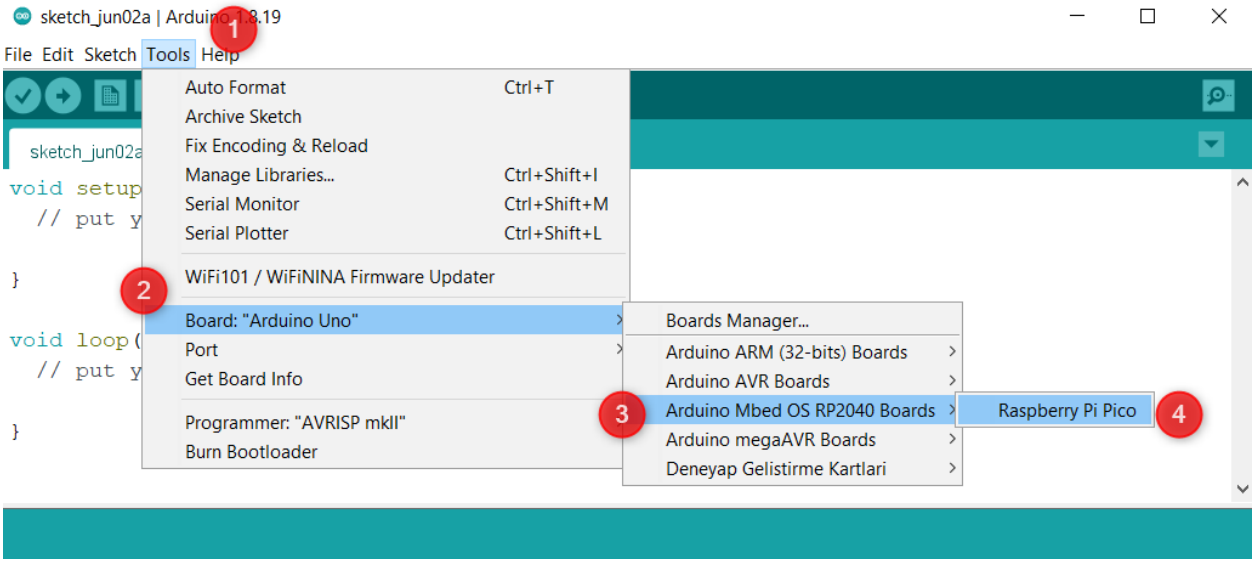

MicroBlocks

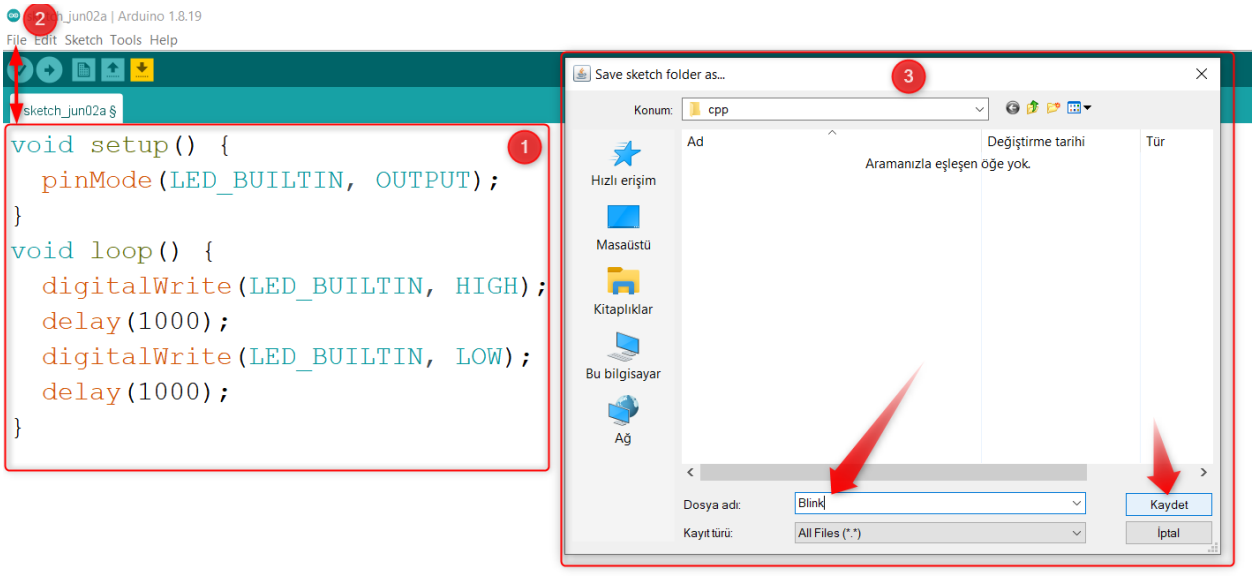

Thonny IDE


sketch_jun02a | Arduino 1.8.19
File Edit Sketch Tools Help
1
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
2
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
3
Boards Manager...
Arduino ARM (32-bits) Boards
Arduino AVR Boards
Arduino megaAVR Boards
Deneyap Gelistirme Kartlari
void setup
// put y
}
void loop
// put y
}

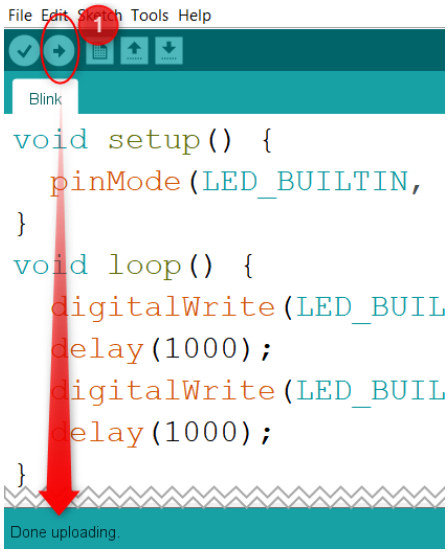

Boards Manager
1
Type All
Raspberry Pi Pico
Arduino Mbed OS RP2040 Boards
by Arduino
Boards included in this package:
Raspberry Pi Pico.
Online Help
More Info
2
3.1.1
Install
[DEPRECATED - Please install standalone packages] Arduino Mbed OS Boards
by Arduino DEPRECATED
Boards included in this package:
Arduino Nano 33 BLE, Arduino Nano 33 BLE Sense, Arduino Nano RP2040 Connect, Arduino Portenta H7, Arduino Edge Control, Raspberry Pi


Raspberry Pi Pico © 2020
BOOTSEL
LED
DEBUG
USB


sketch_jun02a | Arduino 1.8.19
File Edit Sketch Tools Help
1
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
2
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino ARM (32-bits) Boards
Arduino AVR Boards
3
Arduino Mbed OS RP2040 Boards
Arduino megaAVR Boards
Deneyap Gelistirme Kartlari
Raspberry Pi Pico
4


2
File Edit Sketch Tools Help
sketch_jun02a §
1
void setup() {
pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
digitalWrite(LED_BUILTIN, HIGH);
delay(1000);
digitalWrite(LED_BUILTIN, LOW);
delay(1000);
}
Save sketch folder as...
3
Konum: cpp
Hızlı erişim
Masaüstü
Kitaplıklar
Bu bilgisayar
Ağ
Ad
Değiştirme tarihi
Tür
Aramanızla eşleşen öğe yok.
Dosya adı: Blink
Kayıt türü: All Files (*.*)
Kaydet
İptal

BOOTSEL butonuna basarak bilgisayarınıza bağlayınız. Sonraki kod yüklemeleri için BOOTSEL tuşuna basmanız gerekmez. Keyifli projeler :)



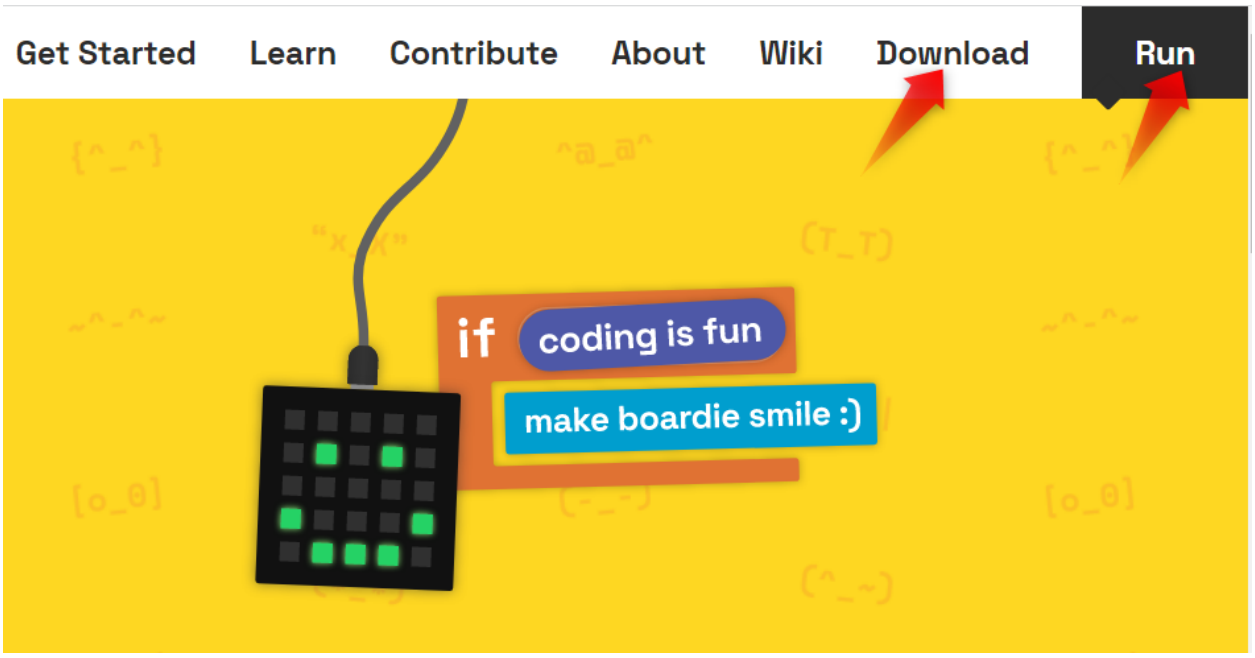

```
void setup() {
// put your setup code here, to run once:
pinMode(7, OUTPUT); // initialize digital pin 7 as an output
  }
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(7, HIGH); //turn the LED on by making the voltage HIGH
  delay(500); //wait for a half second
  digitalWrite(7, LOW); //turn the LED on by making the voltage LOW
  delay(500); //wait for a half second
  }
```

5.2 MicroBlocks Blok Tabanlı Programlama Dili

5.2.1 MicroBlocks Nedir?

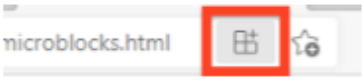
MicroBlocks, micro:bit, Adafruit Circuit Playground Express ve diğerleri gibi eğitici mikrodenetleyici kartlarıyla fiziksel hesaplamayı öğrenmek için kullanılan ücretsiz, Scratch benzeri blok programlama dilidir. MicroBlocks canlı bir ortamdır. Bir bloğa tıklayın ve hemen panoda çalışsın. Komutları deneyin. Sensör değerlerini gerçek zamanlı olarak görün ve grafiklendirin. Artık kodun derlenmesi ve indirilmesi için beklemenize gerek yok. Bir motoru kontrol ederken aynı anda bir animasyon görüntülemek ister misiniz? MicroBlocks, her görev için ayrı komut dosyaları yazmanıza ve bunları aynı anda çalıştırmanıza olanak tanır. Kodunuzun yazılması daha basit ve anlaşılması daha kolaydır. MicroBlocks birçok farklı panoda çalışır, ancak komut dosyalarınız taşınabilirdir. Butonlar, sensörler ve gösterge blokları, ilgili donanıma sahip tüm kartlarda aynı şekilde davranır. Kodu MicroBlocks'ta çalıştırdıktan sonra, USB bağlantısını kesebilir ve Picobricks'i farklı bir güç kaynağıyla besleyebilirsiniz. Karttaki kod otomatik olarak çalışacaktır.

Picobricks'i MicroBlocks ile programlamak için tarayıcıda MicroBlocks Web Sitesini açalım (Google Chrome ve Edge tarayıcıları önerilir).



MicroBlocks'u bir Chrome veya Edge tarayıcısında çalıştırmak için herhangi bir şey yüklemeniz gerekmez; ekranın sağ üst köşesindeki menüden `RUN` butonuna tıklayarak çevrimiçi düzenleyiciyi çalıştırabilirsiniz. Alternatif olarak, İndir butonuna tıklayarak işletim sisteminize uygun çevrimdışı bir sürümü indirebilir ve bilgisayarınıza kurabilirsiniz.

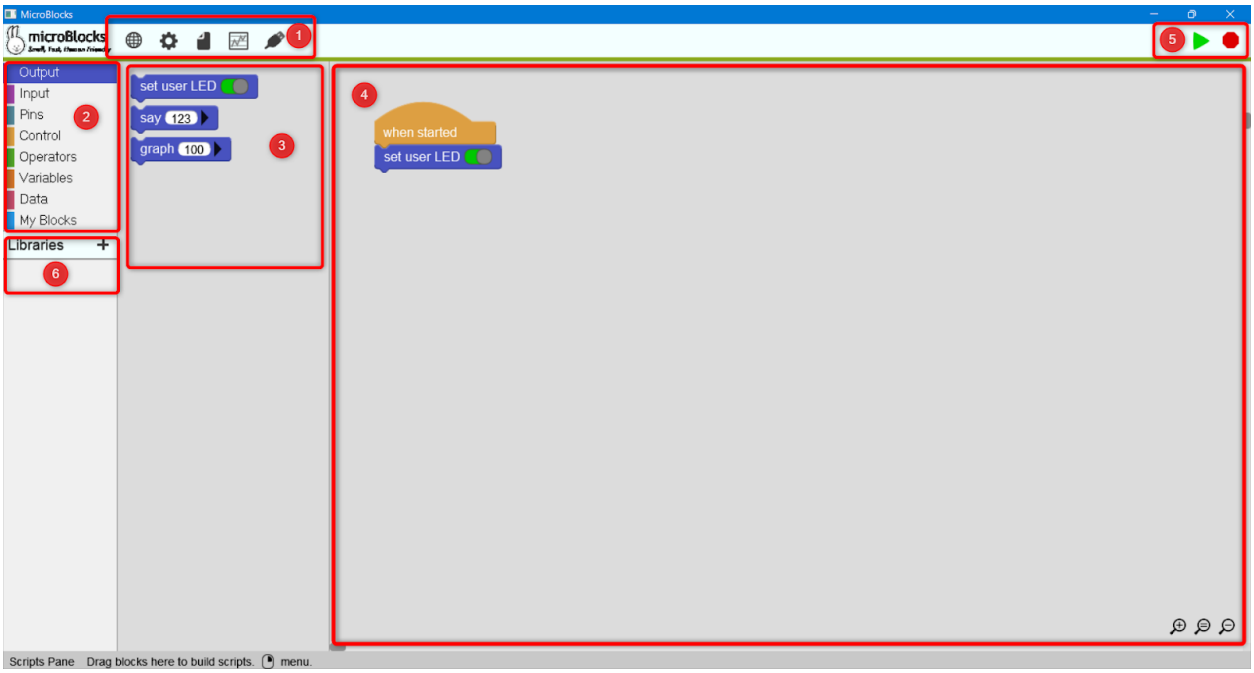
MicroBlocks Web editörünü tarayıcınıza kaydedebilir ve internet erişimi olmadan kullanabilirsiniz. MicroBlocks Web uygulamasını kaydetmek için tarayıcınızda MicroBlocks'u çalıştırın, ardından tarayıcınızın URL çubuğunun sağ üst köşesindeki yükle butonuna tıklayın.

Google Chrome	Microsoft Edge
	

5.2.2 IDE'ye Giriş

MicroBlocks programını açtığınızda aşağıda gösterilen IDE görüntüsünü göreceksiniz. IDE bileşenlerinin açıklamasını aşağıdan inceleyebilirsiniz. IDE'nin ayrıntılı ve en güncel açıklaması için lütfen WIKI'deki Kullanıcı Kılavuzumuza bakın.

- `Menu Bar:` Bu bölümde soldan sağa doğru ilk buton programın dil seçeneğini değiştirmemizi sağlıyor. İkinci buton MicroBlocks'un çalışma ayarlarını ve firmware güncelleme seçimini görebileceğimiz menü iken üçüncü buton File ile ilgili seçenekleri sunuyor. Dördüncü buton, verileri çizmek için grafik bloğu tarafından kullanılan bir grafik penceresini açarken, en sağdaki beşinci buton, USB arabirimi aracılığıyla Picobricks'e bağlanmak için kullanılır.
- `Block Categories:` Bu alan, MicroBlocks'ta programlama için kullanılan blok kategorilerini içerir. Kategoriler farklı renkler kullanılarak gruplandırılmıştır. Kategoriler seçildikçe bloklar paletinde (Alan 3) ilgili bloklar listelenecektir.

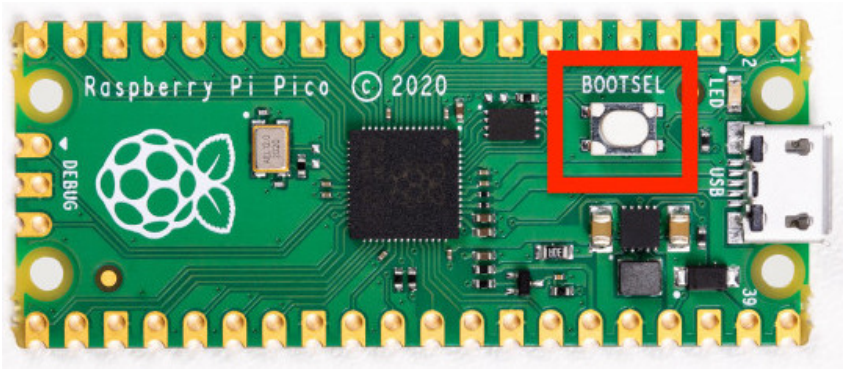


- `Blocks Palette:` Blok kategorileri alanında seçimler yapıldıkça bu alanda belirli fonksiyonları olan bloklar listelenir. Bu alandaki bloklar 4 numaralı Scripting alanına sürüklenerek kod yazılır.
- `Scripting Area:` Tüm kodlama aktivitelerinin yapıldığı alandır. Kullanıcılar, komut dosyaları ve özel bloklar (işlevler) oluşturmak için blokları bu alana sürükleyip bırakır.
- `Start/Stop Buttons:` Bu alan, MicroBlocks programlarını kontrol etmek için kullanılan Start ve Stop olmak üzere iki simge içerir.
- `Library List:` Bu alanın içeriği, kullanıcının komutları ve mikro cihazlarının gereksinimlerine bağlı olarak yüklenen çeşitli kütüphaneleri sunar.

5.2.3 MicBlocks-PicoBricks Bağlantı ve Çalıştırma

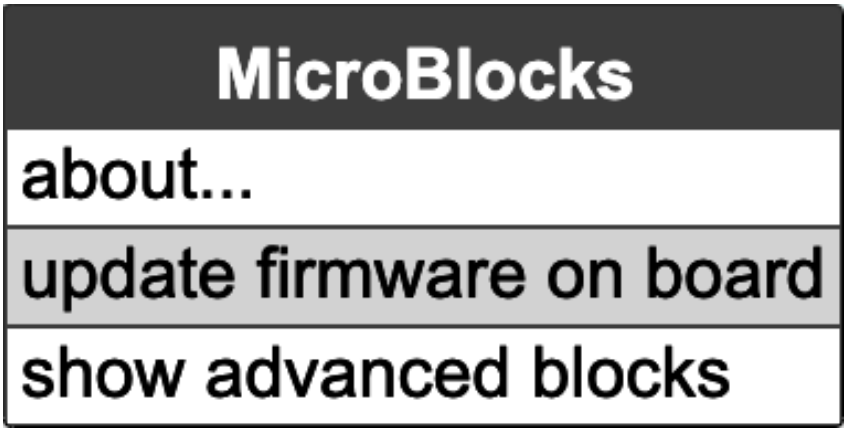
- Yazılımı Çevrimdışı Düzenleyicide Bağlama ve Güncelleme

Picobricks'i çevrimdışı editörüne bağlamak için, Raspberry Pi Pico üzerinde `BOOTSEL butonunu basılı tutarken` USB kablosu ile kartı bilgisayarınıza bağlamanız gerekir.

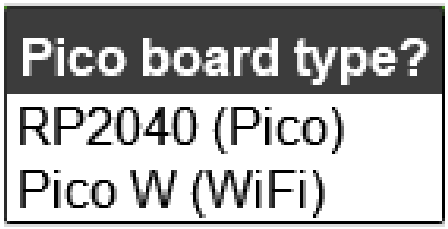


MicroBlocks çevrimdışı editörünü açın ve MicroBlocks menüsünden MicroBlocks butonuna (dişli simgesi) tıklayın,

ardından cihaz yazılımını güncelle seçeneğine tıklayın.



RP2040(Pico) öğesini seçin. Kurulumu yalnızca birkaç saniye sürecek ve bittiğinde, MicroBlocks otomatik olarak Picobricks'e bağlanacaktır.



- Çevrimiçi Editörde Bağlantı ve Güncelleme

Picobricks'i çevrimiçi editöre bağlamak için fazladan birkaç adım gereklidir. Güvenlik nedeniyle tarayıcı, kullanıcıya sormadan kartın USB sürücüsüne erişemez. Önce menüden MicroBlocks butonuna (dişli simgesi) tıklayın, ardından kartta yazılımı güncelle seçeneğine tıklayın ve açılan listeden RP2040 (Pico) seçeneğine tıklayarak kart tipini seçin.

Kart seçildiğinde, aşağıdaki Yazılım Yükleme penceresi açılacaktır.

Şimdi Pico üzerindeki BOOTSEL butonunu basılı tutarken kartı bilgisayarınıza bağlayın.

Görüntülenen mesajdaki "Tamam" butonuna tıklayın ve vm_pico.uf2 adlı yazılım dosyası seçili olarak RPI-RP2 sürücüsüne konumlandırılan sistem dosyası yöneticisi penceresini otomatik olarak açacaktır.

Kaydet butonuna tıklayın, aygıt yazılımı güncellemesi tamamlanacaktır.

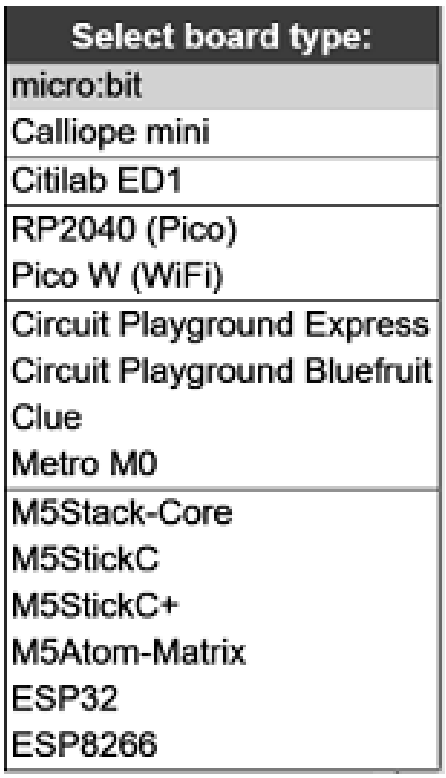
- Normal işlemler için bağlanma (Yazma ve Düzenleme programları):

Bağlan butonuna tıklandığında mikro cihazların takılı olduğu sistem USB portları görüntülenecektir. Bu pencerede önce Pico cihazını seçip ardından Bağlan butonlarına tıklayarak Picobricks'i MicroBlocks'a bağlayabilirsiniz. Bağlantı başarılı olduğunda, USB simgesinin arkasında yeşil bir daire görünecektir.

PicoBricks modüllerinden herhangi birini kullanmak için öncelikle PicoBricks kitaplığını Microblocks düzenleyicisine aktarmanız gerekir. Bunun için Add Library butonuna basmanız gerekmektedir.

Dosya Aç penceresinde, desteklenen cihazların listesini açmak için Kitler ve Panolar butonuna tıklayın. Açılır listeden PicoBricks'e ve ardından Aç butonuna tıklayın.

Her şey yolunda giderse, PicoBricks kitaplığı ve kod blokları, Kod blokları panelinde görüntülenecektir. Tüm Proje örneklerinde, PicoBricks ile başlayan bir blok adı gördüğünüzde, PicoBricks Kitaplığı menüsünde yer alacaktır.

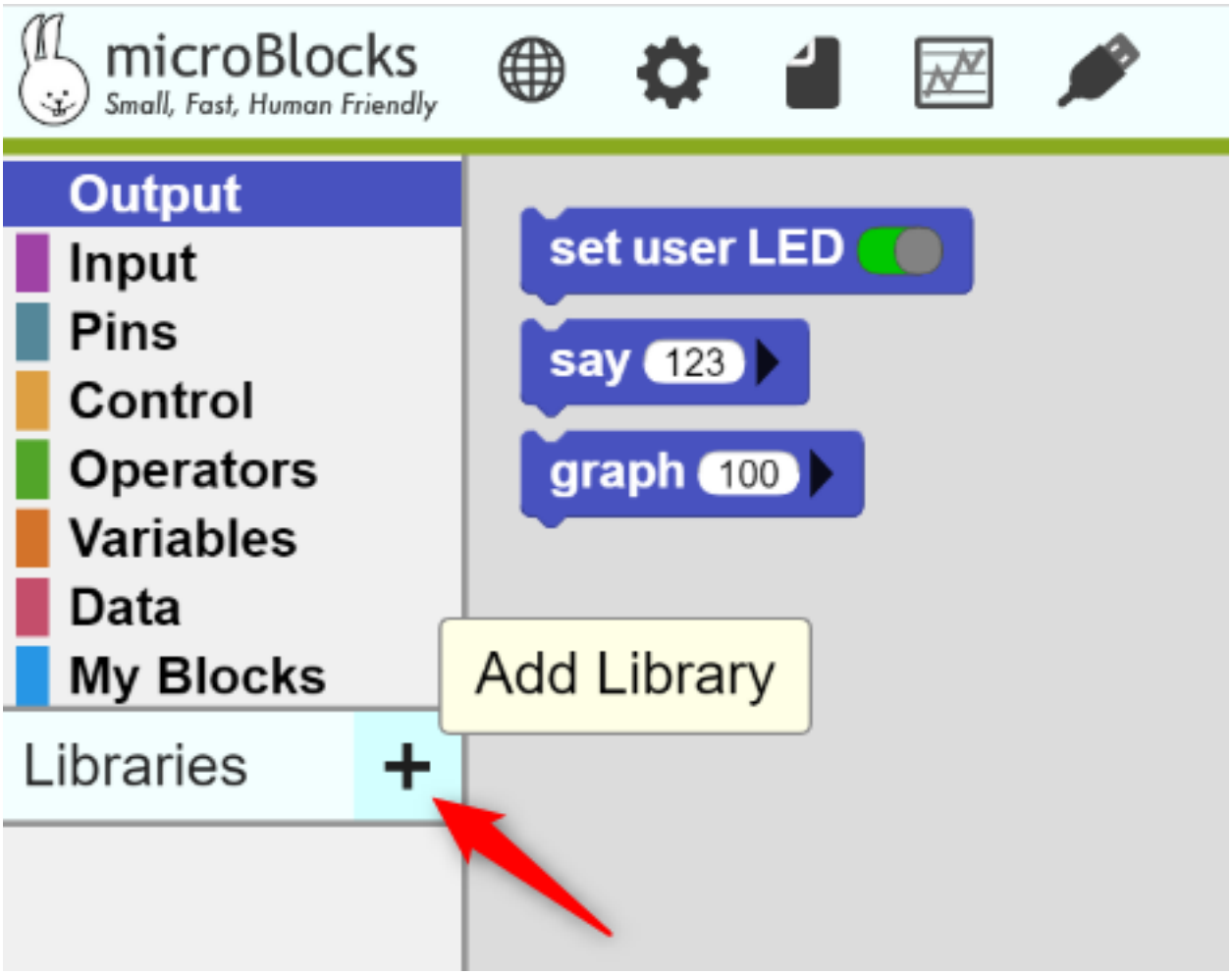

Select board type:
micro:bit
Calliope mini
Citilab ED1
RP2040 (Pico)
Pico W (WiFi)
Circuit Playground Express
Circuit Playground Bluefruit
Clue
Metro M0
M5Stack-Core
M5StickC
M5StickC+
M5Atom-Matrix
ESP32
ESP8266

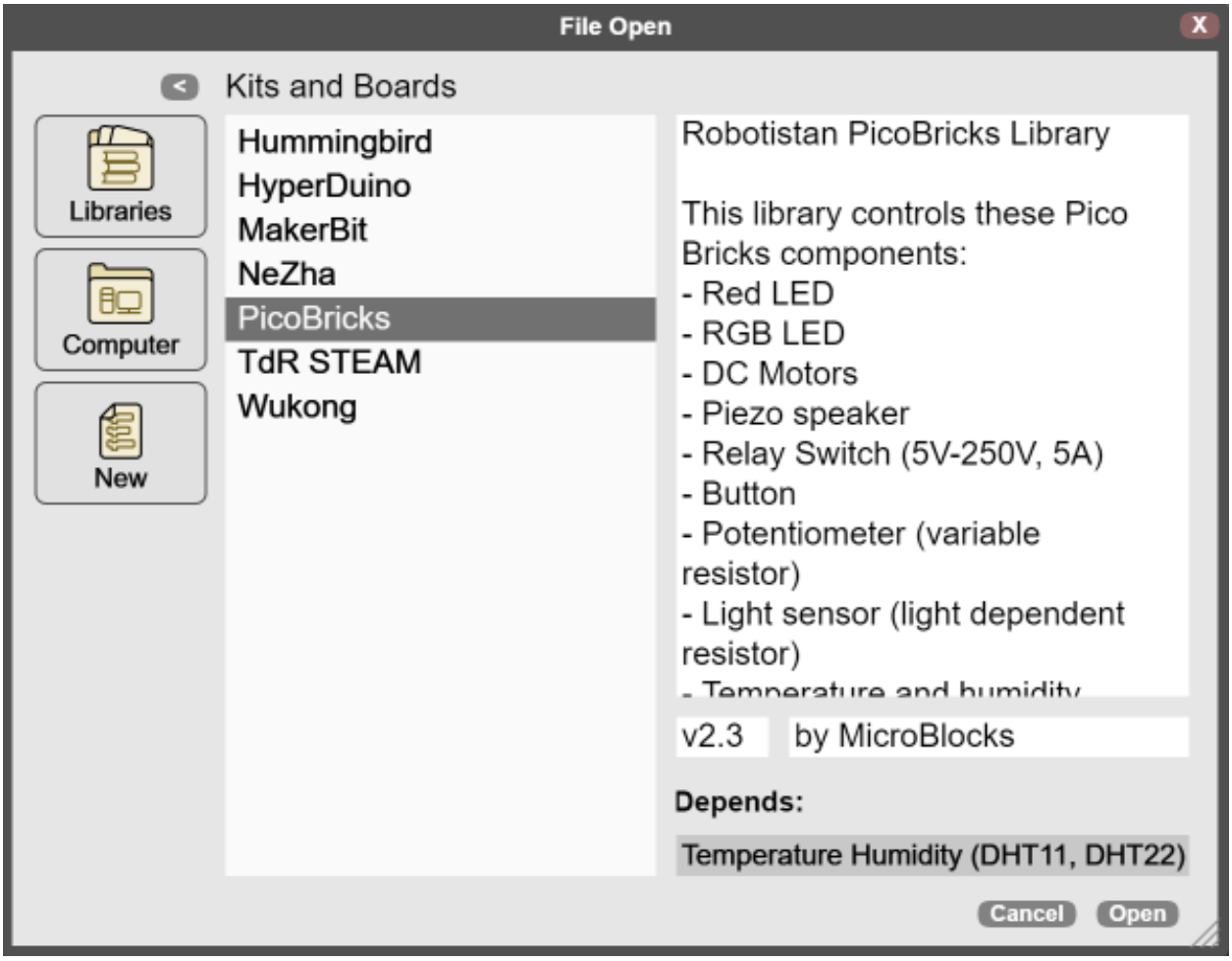

Firmware Install
X
Connect USB cable while holding down the white BOOTSEL button before proceeding.
You will be asked to save the firmware file.
Select RPI-RP2 as the destination drive, then click Save.
OK

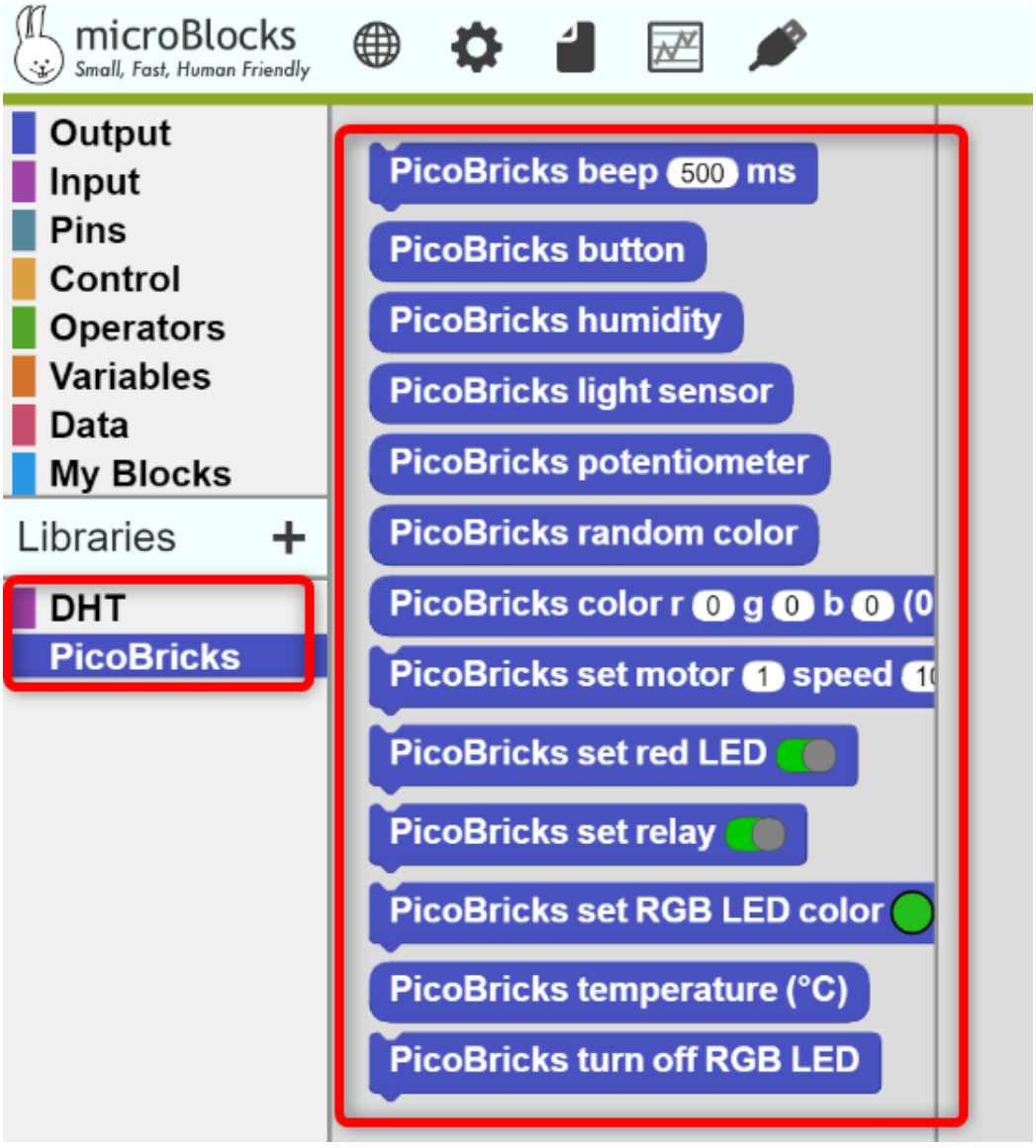
:align:center:width:720:figclass:align-center

```
\relax
```

Şekil 1: MicroBlocks, gerçek zamanlı bir kodlama editörüdür. Kodu yazdıktan sonra derleme ve karta yükleme işlemi yoktur. Kod bloklarına tıkladığınızda kod çalışacaktır.




File Open
Kits and Boards
Libraries
Computer
New
Hummingbird
HyperDuino
MakerBit
NeZha
PicoBricks
TdR STEAM
Wukong
Robotistan PicoBricks Library
This library controls these Pico Bricks components:
- Red LED
- RGB LED
- DC Motors
- Piezo speaker
- Relay Switch (5V-250V, 5A)
- Button
- Potentiometer (variable resistor)
- Light sensor (light dependent resistor)
v2.3
by MicroBlocks
Depends:
Temperature Humidity (DHT11, DHT22)
Cancel
Open


microBlocks
Small, Fast, Human Friendly
Output
Input
Pins
Control
Operators
Variables
Data
My Blocks
Libraries
DHT
PicoBricks
PicoBricks beep 500 ms
PicoBricks button
PicoBricks humidity
PicoBricks light sensor
PicoBricks potentiometer
PicoBricks random color
PicoBricks color r 0 g 0 b 0 (0
PicoBricks set motor 1 speed
PicoBricks set red LED
PicoBricks set relay
PicoBricks set RGB LED color
PicoBricks temperature (°C)
PicoBricks turn off RGB LED

5.2.4 PicoBricks Aygıt Yazılımı

PicoBricks Aygıt Yazılımını daha önce güncellediyseniz, USB simgesine tıklayarak bağlanabilirsiniz. MicroBlocks'u Picobricks'e ilk kez bağlayacaksanız bölüm 1.1.2'deki adımları takip edebilirsiniz.

IDE'nin veya Pico Firmware'in yeni bir sürümü olup olmadığını zaman zaman kontrol etmek iyi bir fikirdir. Menüde MicroBlocks Seçenekleri (dişli simgesi)/hakkında öğesini seçerek çalıştırdığınız sürümü doğrulayabilirsiniz. Aşağıdaki gibi bir ekran görmelisiniz:

5.3 Başlangıç Seviyesi İçin Thonny (MicroPython) IDE

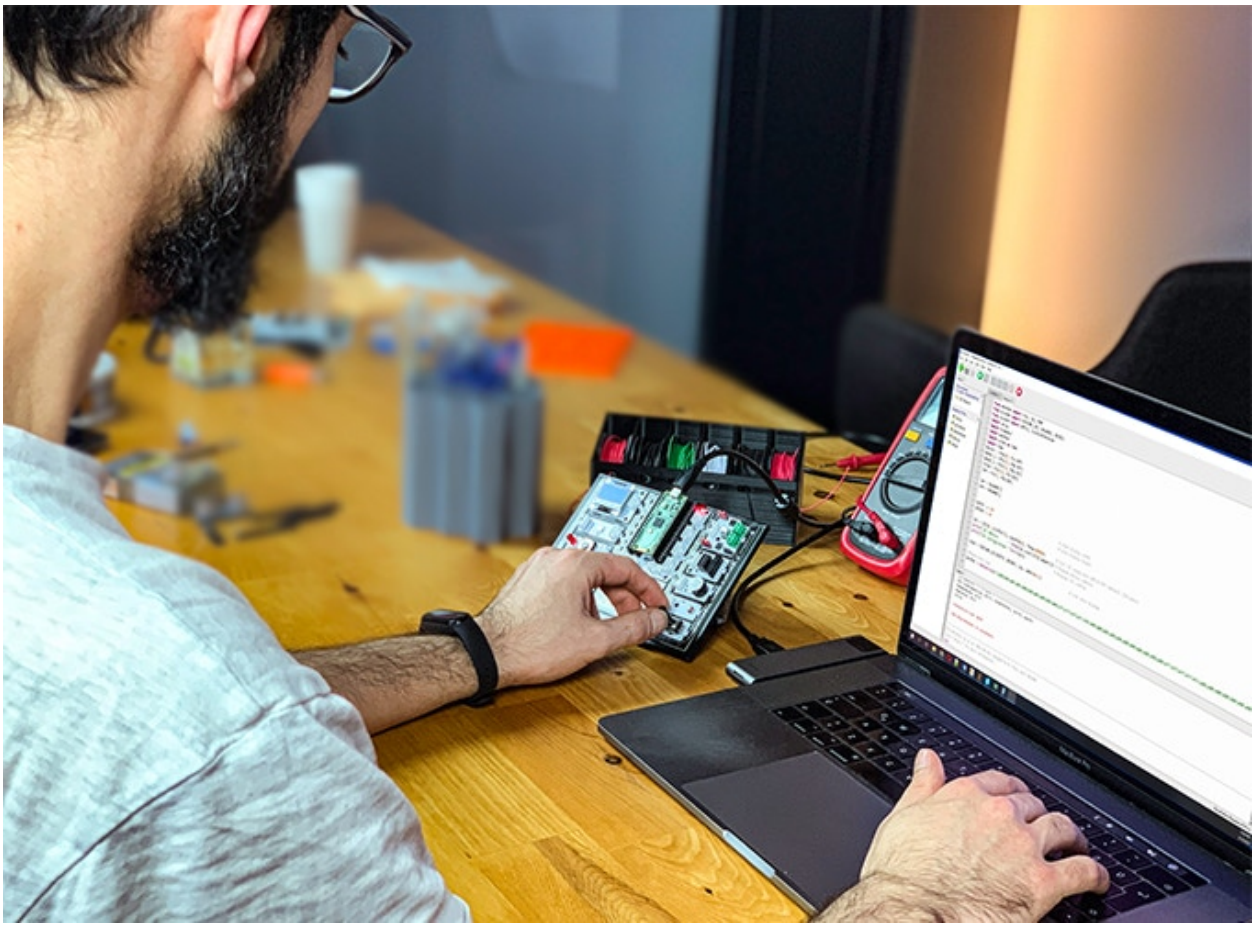
5.3.1 Thonny IDE Kurulumu

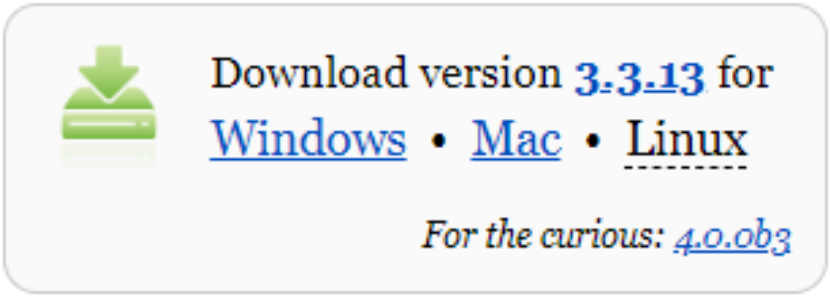
Raspberry Pi Pico, PicoBricks'in kalbindedir. Thonny, Pico kartını ve PicoBricks'i kodlamak için harika bir seçimdir. Thonny Websitesini ziyaret edebilirsin. Uygun versiyonu seçin ve bilgisayarınıza indirin. Ardından kurulumu gerçekleştirin. Thonny IDE'yi " $ pip install thonny " komutunu kullanarak da yükleyebilirsiniz.

```
$ pip install thonny
```

5.3.2 Thonny IDE Arayüzü

A: Boş bir komut dosyası açın. B: Mevcut bir kod dosyası açabilirsiniz. C: Üzerinde çalıştığınız kod dosyasındaki değişiklikleri kaydedebilirsiniz. D: Belirttiğiniz yorumlama ortamında yazdığınız kodu çalıştırır. E: Kodunuzdaki hataları kontrol edebilirsiniz. F: Hata ayıklamak için kod satırlarını çalıştırabilirsiniz. G: Hata ayıklama sırasında kod satırındaki komutlar arasında gezinebilirsiniz. H: Hata ayıklamadan çıkabilirsiniz. I: Hata ayıklama modundan çalışma moduna geçebilirsiniz. J: Kodun yürütülmesini durdurabilirsiniz.

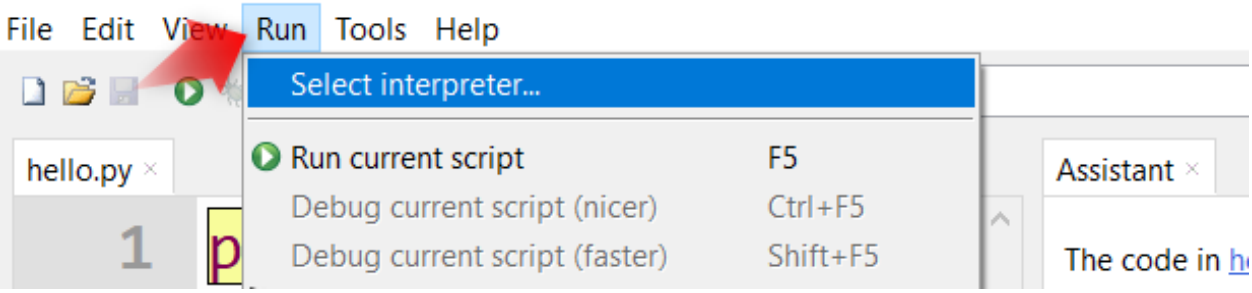



Download version 3.3.13 for
Windows • Mac • Linux
For the curious: 4.0.0b3

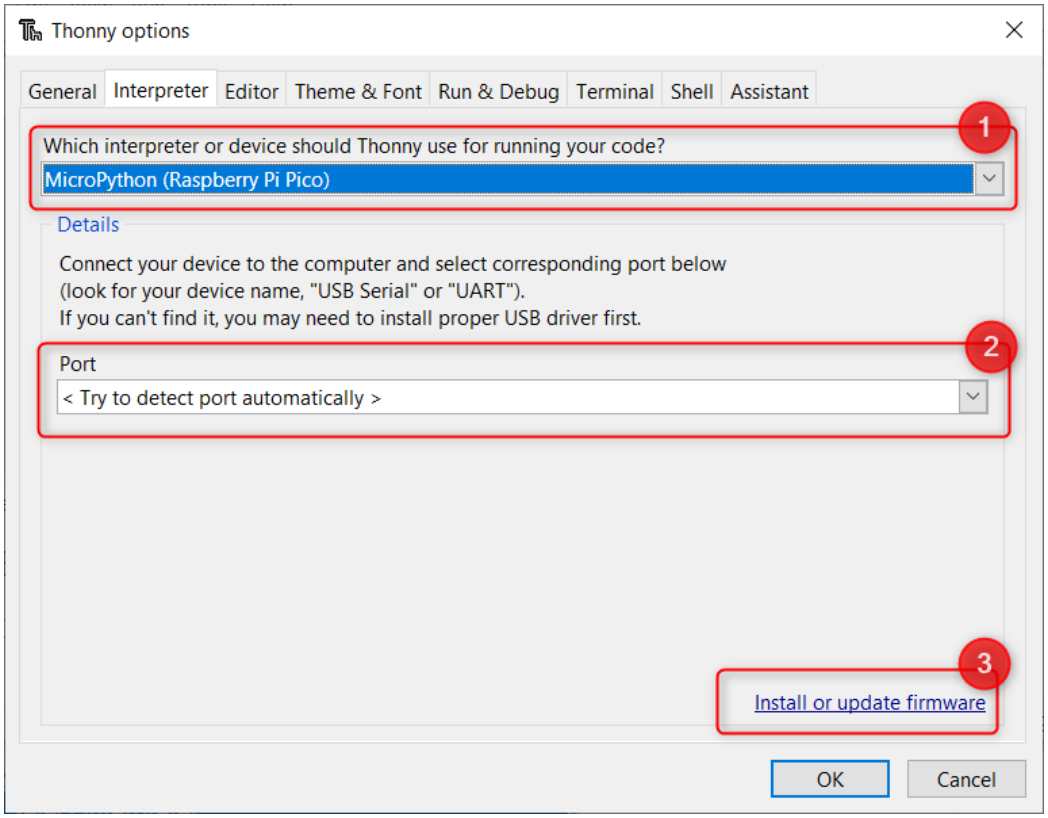

A B C D E F G H I J
STOP

5.3.3 MicroPython Aygıt Yazılımını Raspberry Pi Pico'ya Yükleyin

Raspberry Pi Pico'nun yazacağımız MicroPython kodunu anlaması için ona özel bir işletim sistemi kurmamız gerekiyor. Biz buna aygıt yazılımı diyoruz. Thonny editörünü açın ve Çalıştır menüsünden Tercüman seç'e tıklayın.



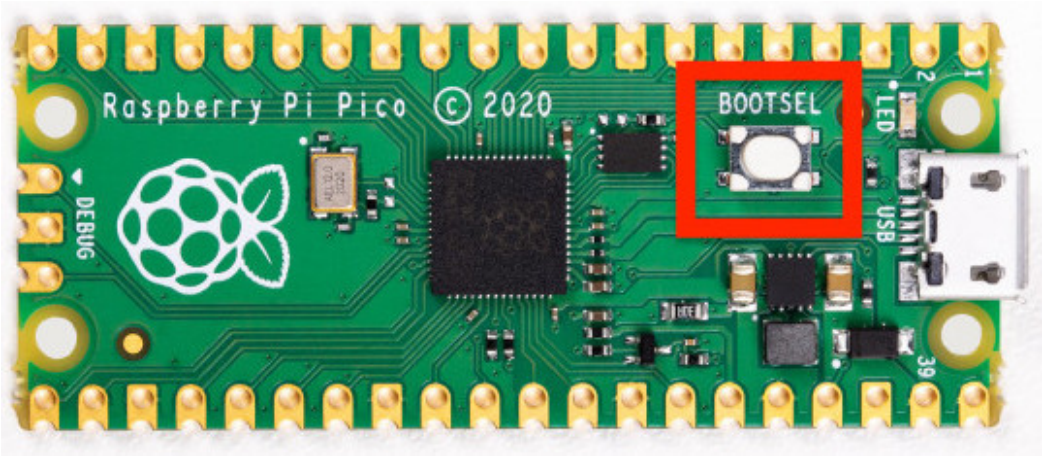
1. alanda gösterilen listeden Raspberry Pi Pico'yu seçin. 2. alanı resimdeki gibi bırakın, 3. alana tıklayın.

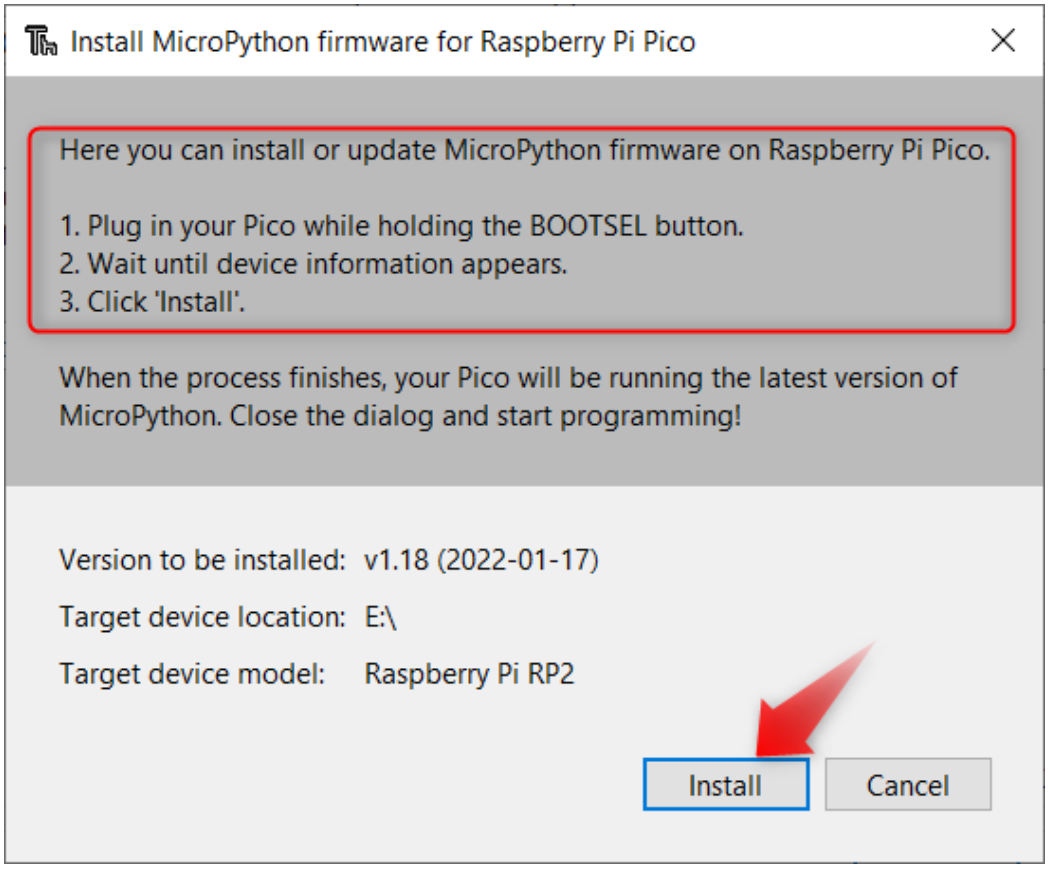


Üzerindeki ``BOOTSEL butonuna" basılı tutarak Pico'yu bir kablo ile bilgisayarınızın USB portuna bağlayın.

Yükle butonu etkinleştirildikten sonra butonu bırakabilirsiniz. "Yükle butonuna" basın ve aygıt yazılımının yüklenmesini bekleyin.

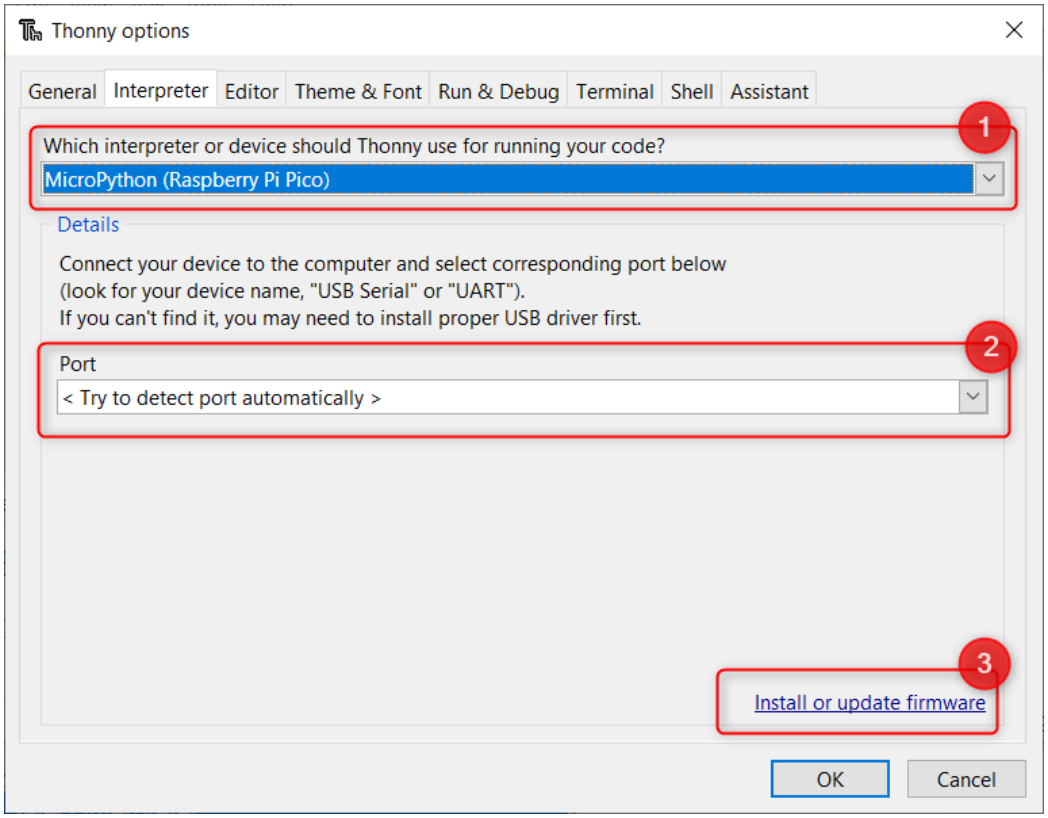
Kurulum tamamlandıktan sonra, kurulumu tamamlamak için Kapat butonuna tıklayın.


Raspberry Pi Pico © 2020
BOOTSEL
LED
USB
DEBUG
1
2
39


Install MicroPython firmware for Raspberry Pi Pico
Here you can install or update MicroPython firmware on Raspberry Pi Pico.
1. Plug in your Pico while holding the BOOTSEL button.
2. Wait until device information appears.
3. Click 'Install'.
When the process finishes, your Pico will be running the latest version of MicroPython. Close the dialog and start programming!
Version to be installed: v1.18 (2022-01-17)
Target device location: E:\
Target device model: Raspberry Pi RP2
Install
Cancel

5.3.4 Raspberry Pi Pico'da Kod Yükleme ve Çalıştırma

Pico'nun kablosunu doğrudan bilgisayarın USB bağlantı noktasına takın. Bootsel butonuna basılı tutmanız gerekmez. Thonny'deki Çalıştır menüsünden ``Tercüman seç'` seçeneğini seçin. 1. bölümde Raspberry Pi Pico'nun seçili olduğundan emin olunuz. Tamam butonuna tıklayarak pencereyi kapatınız.

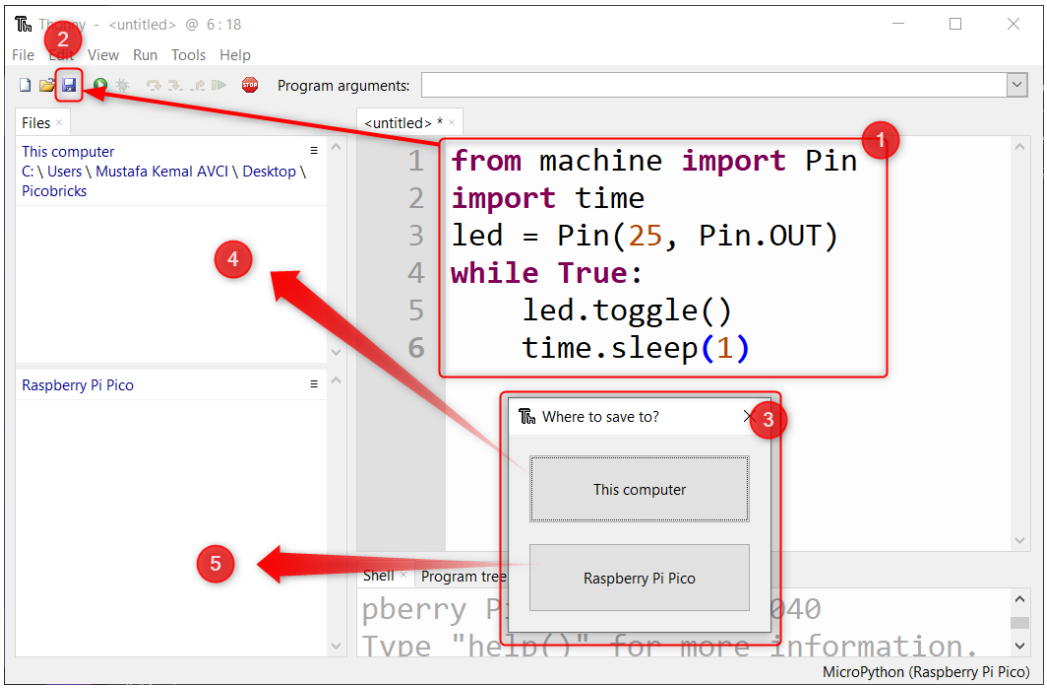


Görünüm menüsünden Dosyalar seçeneğini etkinleştirin. Ekranın sol tarafına uzun bir dosya gezgini sekmesi yerleştirilecektir. 1. bölümde Raspberry Pi Pico'yu görüyorsanız Thonny Pico'ya sorunsuz bir şekilde bağlanmış demektir, kodunuzu yazmaya, kaydetmeye ve çalıştırmaya hazırsınız demektir.

Thonny'de yazdığınız MicroPython kodu, Raspberry Pi Pico ve benzeri mikro kontrol kartları için düzenlenmiş kütüphanelerden oluşuyor ve MicroPython olarak adlandırılıyor. Sözdizimi ve neredeyse tüm kitaplıklar MicroPython ile aynı şekilde çalışır. Yazılım dünyasının `"merhaba dünya"` uygulaması, fiziksel programlamaya `"blink"` uygulamasıdır. 1. alanda gösterilen kodu yazın. 2. alandaki kaydet butonuna tıklayın. Thonny, 3. alandaki pencerede size kodunuzu bilgisayarınızın çalışma dizinine mi yoksa Pico'nun belleğine mi kaydetmek istediğinizi soracaktır. Bilgisayarınızı seçerseniz, ortaya çıkan dosya 4. alanda, Pico'yu seçerseniz, ortaya çıkan dosya alanda görünecektir.

Kayıt penceresinden Raspberry Pi Pico'yu seçin, Dosya Adı alanına `“blink.py”` yazın ve Tamam butonuna tıklayın. `"blink.py"` dosyasını Pico'nun dosya gezgininde gördükten sonra tıklayın. klavyedeki F5 tuşuna veya araç çubuğundaki yeşil Çalıştır butonuna basın ve kod dosyası Pico tarafından çalıştırılacaktır. Pico üzerindeki dahili LED'in 1 saniye aralıklarla yanıp söndüğünü görüyorsanız, ilk kodunuzu başarıyla yazıp çalıştırmışsınız demektir. Tebrikler :)

Not: Yazdığınız kodun Pico açılır açılmaz çalıştır komutu vermeden çalışmasını istiyorsanız, kodunuzu Pico'nun ana dizinine `"main.py"` adıyla kaydetmelisiniz.


File Edit View Run Tools Help
Program arguments:
Files
This computer
C: \ Users \ Mustafa Kemal AVCI \ Desktop \ Picobricks
Raspberry Pi Pico
<untitled> *
1 from machine import Pin
2 import time
3 led = Pin(25, Pin.OUT)
4 while True:
5 led.toggle()
6 time.sleep(1)
Where to save to?
This computer
Raspberry Pi Pico
Shell
Program tree
Type "help()" for more information.
MicroPython (Raspberry Pi Pico)
1
2
3
4
5

BÖLÜM 6

Projeler

PicoBricks'in 11 farklı komponenti ile birçok projeyi hayata geçirebilirsiniz. Bunun dışında 8 farklı bağlantı tipi ile sınırsız sayıda komponent ekleyerek hayallerinizin ötesinde projelere imza atabilirsiniz.

PicoBricks ile eğleceli projeler yapabilirsiniz.

Ev otomasyon sistemlerini PicoBricks ile kurabilirsiniz..

İşte PicoBricks'i ayırdıktan sonra yapabileceğiniz bazı projeler.

6.1 Blink

6.1.1 Giriş

Gerçek hayatta işi yeni öğrenmeye başlayan çalışan, önce en temel görevi üstlenir. Temizlik görevlisi ise süpürgenin kullanımını, aşçı ise mutfak araç gereçlerini, garson ise tepsi taşımayı… Bu örnekleri arttırabiliriz. Yazılım geliştirmeye yeni başlayanların ilk yazdıkları kod "`Hello World`" olarak bilinir. Kullandıkları dil de ekrana ya da konsol penceresine program başlar başlamaz "`Hello World`" yazdırmak, programlamaya atılan ilk adımdır. Bir bebeğin emeklemeye başlaması gibi…

```
>>> print("Hello World")
>>> Hello World
```

Robotik kodlamaya diğer adıyla fiziksel programlamaya atılan ilk adım ise Blink uygulamasıdır. Robotik kodlamaya göz kırpmak anlamını taşır. Basit olarak bir LED'in bağlantısını devre kartına yaparak yapılan kodlama ile LED'in sürekli yanıp sönmesini sağlanır. Robotik kodlama alanında kendini geliştirmiş kişilere bu seviyeye nasıl geldiklerini sorun. Size verecekleri cevap şöyle başlar; her şey bir LED yakmak ile başladı!

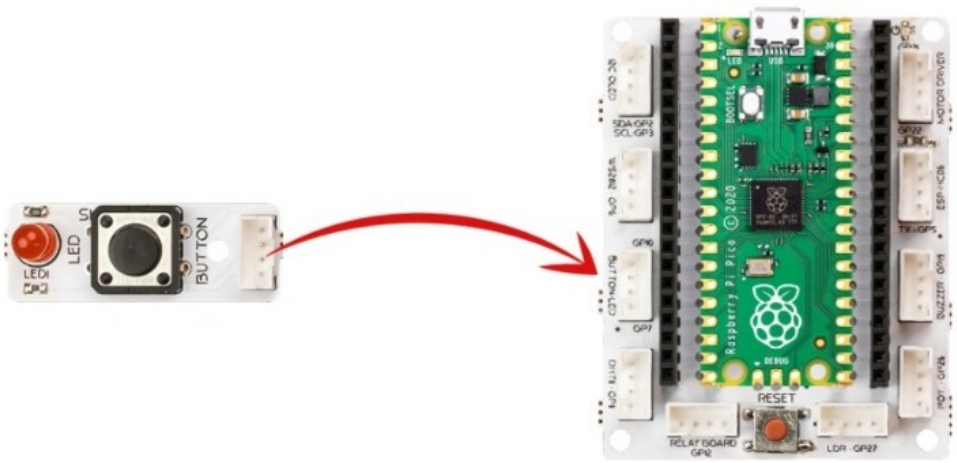
LED'ler elektronik cihazların dilidir. LED'ler sayesinde programcı cihazın görevin hangi aşamasında olduğunu, varsa sorunun ne olduğunu, hangi seçeneklerin aktif olduğunu kullanıcılara ifade eder. Bu projede Picobricks ile üzerinde yer alan LED'lerin çeşitlerini öğrenip onları nasıl yakıp söndüreceğini öğreneceksin.

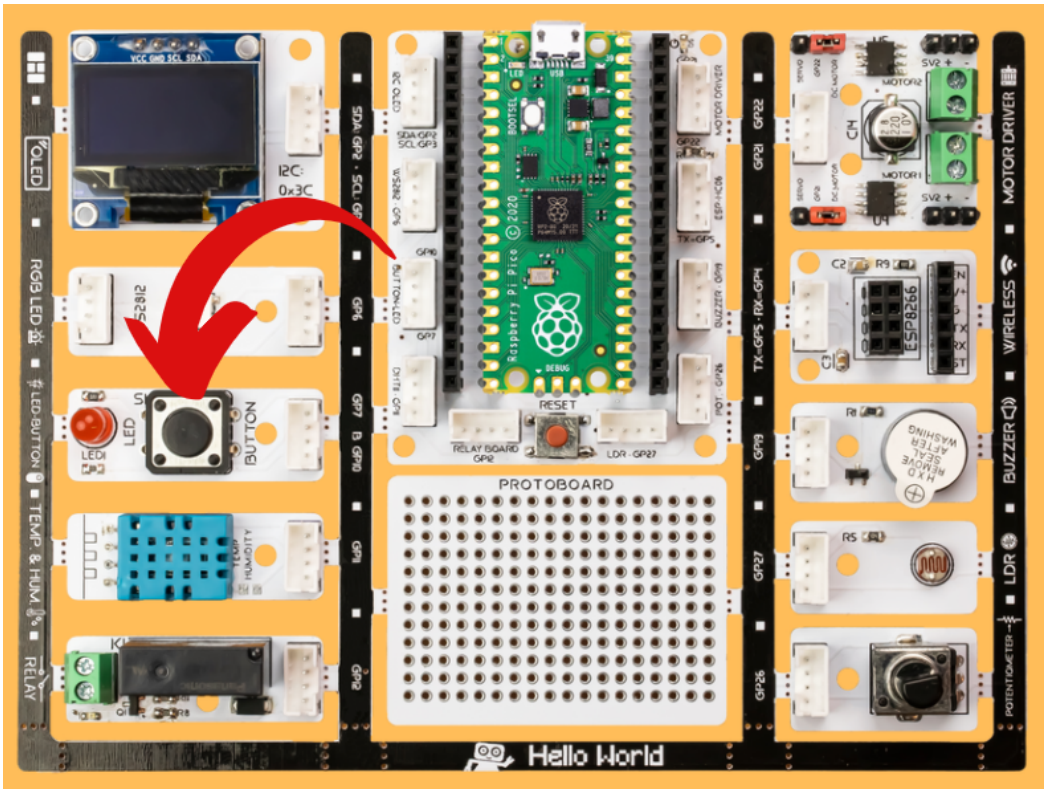
6.1.2 Proje Detayları ve Algoritması

Picobricks üzerinde 1 adet 5mm kırmızı LED, 1 adet WS2812B RGB LED bulunmaktadır. Normal LED'ler tek renk yanabilirken RGB renkler hem ana hem ara renkler olmak üzere farklı renklerde yanabilmektedir. Bu projede Picobricks üzerindeki kırmızı LED kullanacağız.

Projede Picobricks üzerindeki kırmızı LED yakılması, belirli bir süre geçtikten sonra söndürülmesi, tekrar belirli bir süre geçtikten sonra yakılması ve bu işlemlerin sürekli tekrarlanması için gerekli kodlar yazacağız.

6.1.3 Bağlantı Diyagramı





Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.1.4 Projenin MicroPython Kodu

```
from machine import Pin #to access the hardware
on the pico
import utime #time library
led = Pin(7,Pin.OUT) #initialize digital pin 7 as an output for LED
while True: #while loop
    led.toggle() #LED on&off status
    utime.sleep(0.5) #wait for a half second
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.1.5 Projenin Arduino C Kodu

```
void setup() {
 // put your setup code here, to run once:
pinMode(7,OUTPUT); // initialize digital pin 7 as an output
 }
 void loop() {
  // put your main code here, to run repeatedly:
    digitalWrite(7,HIGH); //turn the LED on by making the voltage HIGH
    delay(500); //wait for a half second
    digitalWrite(7,LOW); //turn the LED on by making the voltage LOW
    delay(500); //wait for a half second
 }
```

6.1.6 Projenin MicroBlocks Kodu

1) `Kontrol` kategorisinden `When started` bloğunu Scripting alanına sürükleyip bırakın. Bu blok, `Start` butonuna her tıklandığında altındaki kodu çalıştırır.

2) Ardından, `Kontrol` kategorisinden `forever` bloğunu sürükleyin ve `when started` bloğunun altına ekleyin. `Forever` bloğu, içine yerleştirilen blokları durmadan çalıştırır.

3) PicoBricks `set red LED block` öğesini sürükleyin ve `forever` bloğunun içine bırakın. Bloktaki boolean DOĞRU(yeşil) seçimi, LED'in yanmasını sağlar. Alternatif olarak, YANLIŞ(kırmızı) seçimi LED'in sönmesine neden olur.

"Start" butonuna basarak kırmızı LED'in yanıp yanmadığını test edin.

4) Şimdi, kırmızı LED'i kapatmak için bloktaki boolean kontrolüne bir kez tıklayarak kırmızıya ayarlayın. Bu ayar YANLIŞ anlamına gelir ve LED'i kapatmalıdır.

Start butonuna tekrar basarak LED'in sönüp sönmediğini test edin.

5) Şimdi LED'in belirli zaman aralıklarında kendi kendine yanıp sönmesi için kodumuzu değiştireceğiz.

`500 milisaniye bekleyin` bloğunu "Kontrol" kategorisinden sürükleyin ve `PicoBricks set red LED` bloğunun altına ekleyin.

6) Daha sonra `500 milisaniye bekle` bloğunun altına tekrar `Picobricks set red LED block` ekleyin ve boolean kontrolünü False olarak ayarlayın.

En alta başka bir `500 milisaniye bekle` bloğu ekleyin.

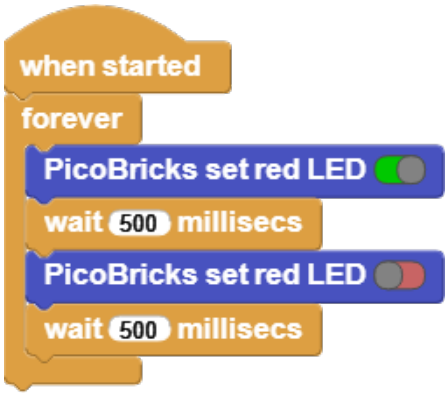
`Start` butonuna bastığınızda kırmızı LED'in 500 milisaniye aralıklarla yanıp söndüğünü göreceksiniz. `500 milisaniye bekle` bloğundaki 500 sayısı milisaniyeyi temsil eder. Bu sayıyı istediğiniz gibi değiştirebilirsiniz. Bir saniye 1000 milisaniyedir.

6.2 Action-Reaction

6.2.1 Giriş

Bu projede Picobricks'in buton-LED modülünü kodlayarak projelerinde kullanıcıdan bir komutun nasıl alınacağını ve bu komuta nasıl tepki verileceğini öğreneceksin.

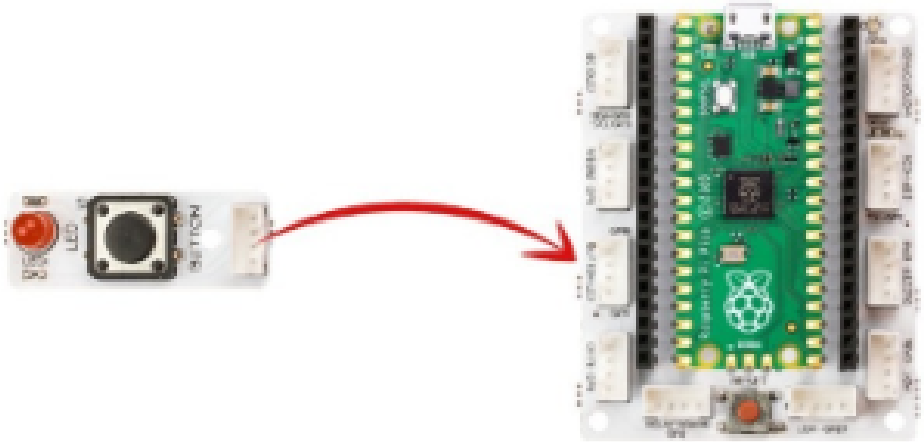
Newton hareket yasalarında açıkladığı gibi her etkiye karşı bir tepki oluşur. Elektronik sistemler kullanıcılardan komutlar alırlar ve görevlerini yerine getirirler. Genellikle bu iş için bir tuş takımı, dokunmatik ekran veya bir buton kullanılır. Elektronik cihazlar görevlerinin sona erdiğini ve görev süresince olan bitenden kullanıcıyı haberdar etmek için sesli, yazılı veya görsel olarak tepki verirler. Bu tepkilerin kullanıcıyı haberdar etmenin yanı sıra olası arızada hatanın nerede olabileceğinin anlaşılmasında yardımcı olabilmektedir.



6.2.2 Proje Detayları ve Algoritması

Picobricks'te 1 buton var ve `anahtar` gibi çalışır, basıldığında akım iletir ve serbest bırakıldığında akım vermez.

6.2.3 Bağlantı Diyagramı

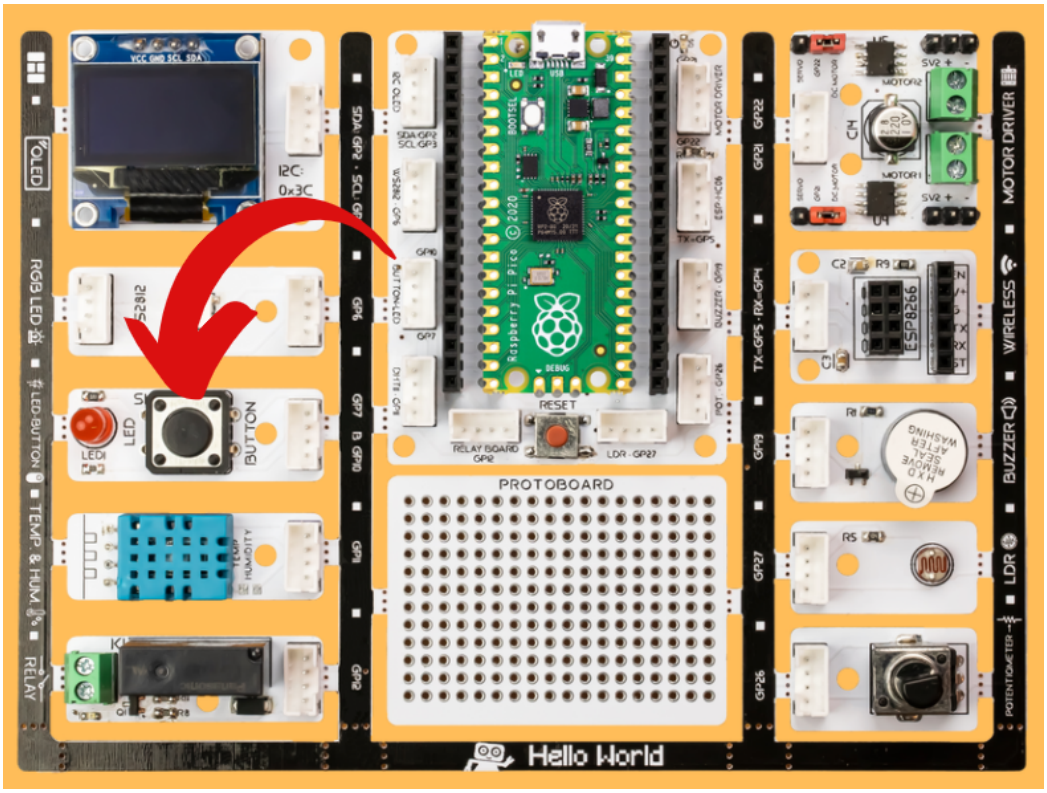


Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.2.4 Projenin MicroPython Kodu

```
from machine import Pin#to acces the hardware picobricks
led = Pin(7,Pin.OUT)#initialize digital pin as an output for led
push_button = Pin(10,Pin.IN,Pin.PULL_DOWN)#initialize digital pin 10 as an input
while True:#while loop
 logic_state = push_button.value();#button on&off status
 if logic_state == True:#check the button and if it is on
     led.value(1)#turn on the led
```

(sonraki sayfaya devam)



(önceki sayfadan devam)

```
else:
    led.value(0)#turn off the led
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.2.5 Projenin Arduino C Kodu

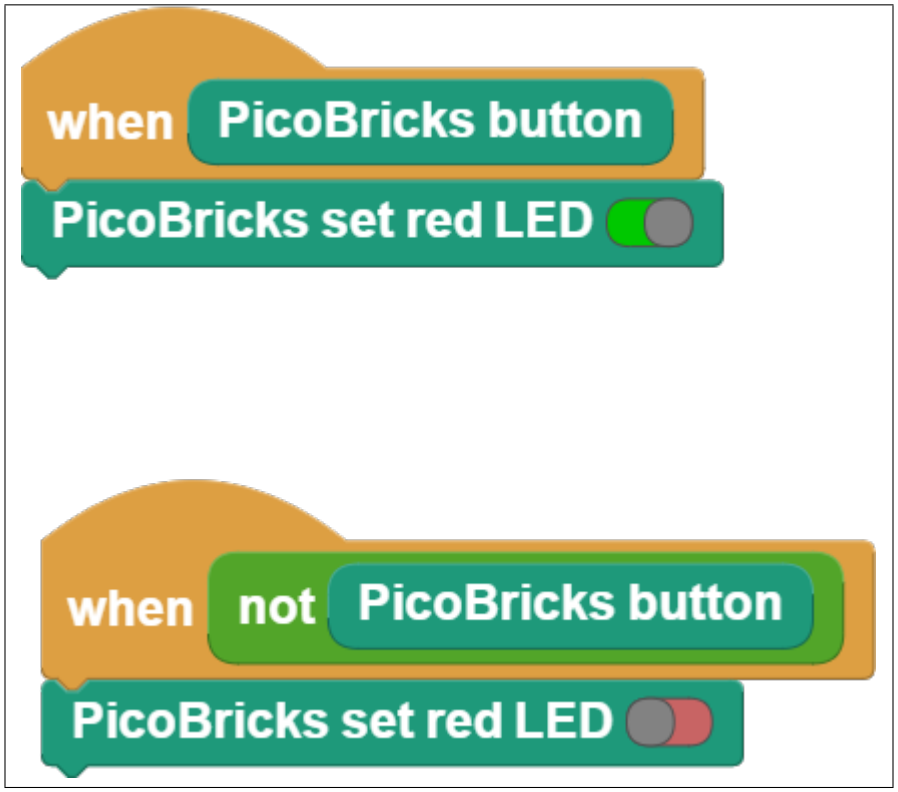
```
 void setup() {
// put your setup code here, to run once:
pinMode(7,OUTPUT);//initialize digital pin 7 as an output
pinMode(10,INPUT);//initialize digital pin 10 as an input


 }
    void loop() {
// put your main code here, to run repeatedly:
if(digitalRead(10)==1){//check the button and if it is on
  digitalWrite(7,HIGH);//turn the LED on by making the voltage HIGH
}
else{
  digitalWrite(7,LOW);//turn the LED off by making the voltage LOW
}
delay(10);//wait for half second
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
}
```

6.2.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.3 Autonomous Lighting

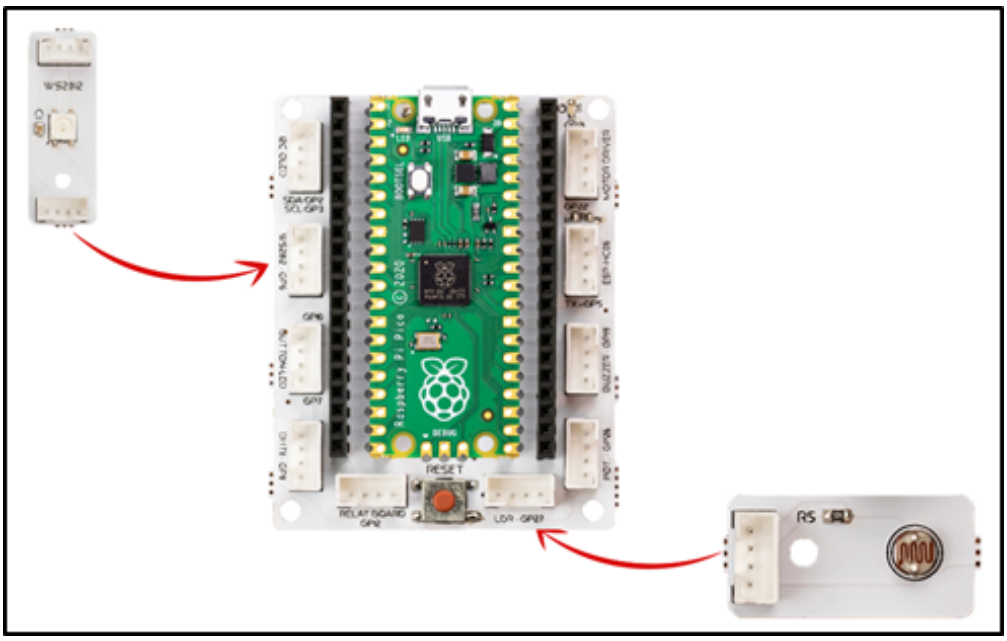
6.3.1 Giriş

Bu projede PicoBricks ile hava karardığında aydınlatmanın otomatik yandığı sistemlerin çalışma sistemlerini anlamak için ışık miktarı düştüğünde LED'in yanmasını sağlayacağız.

6.3.2 Proje Detayları ve Algoritması

Elektronik sistemlerin verilen görevi, topladığı veriler doğrultusunda karar vererek otomatik olarak yapmasına otonom olma durumu denir. Elektronik sistemlerin çevresinden veri toplamasını sağlayan bileşenlerine sensör (algılayıcı) denir. Ortamdaki ışık seviyesinin hangi düzeyde olduğu, hava sıcaklığının kaç derece olduğu, su akış hızının kaç lt/dk olduğu, ses şiddetinin ne kadar olduğu gibi birçok veri sensörler tarafından toplanarak elektrik sinyalleri olarak yani veri olarak PicoBricks'e iletilir. Picobricks'te birçok sensör yer almaktadır. Sensörlerden verinin nasıl alındığını ve bu verilerin nasıl yorumlanılıp kullanılacağını bilmek, kitap okumanın kelime dağarcığını geliştirmesi gibi proje fikirlerini geliştirecektir.

6.3.3 Bağlantı Diyagramı

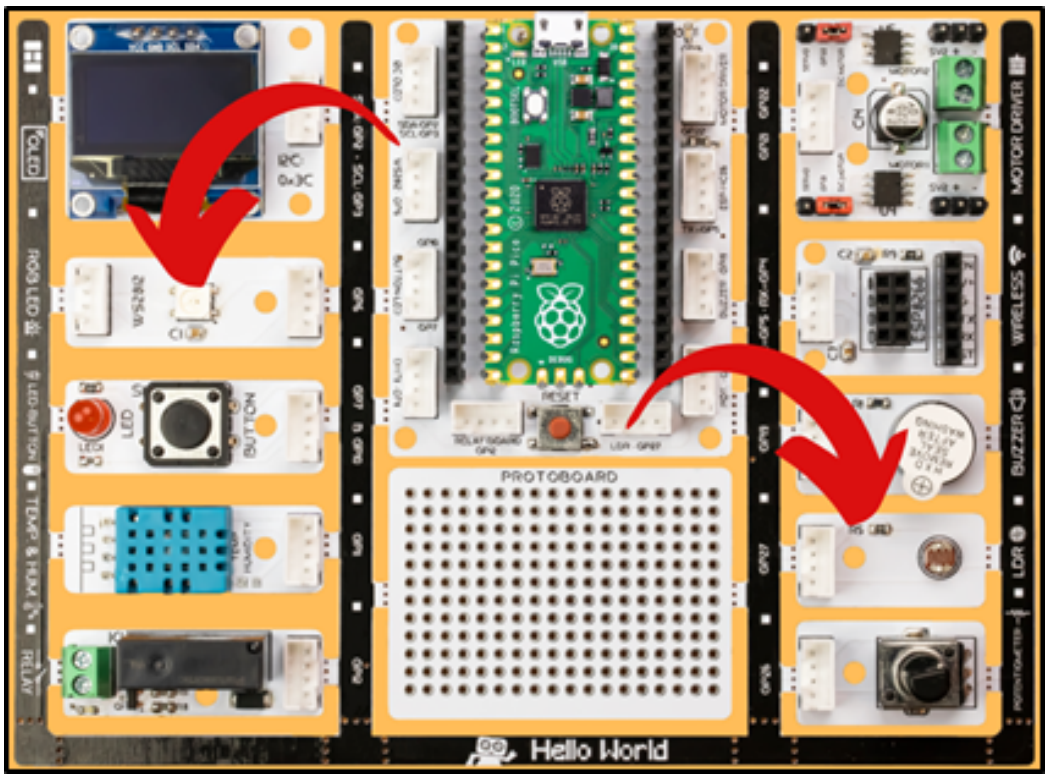


Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.3.4 Projenin MicroPython Kodu

```
import time
from machine import Pin, ADC
from picobricks import  WS2812
#define the library

ldr = ADC(Pin(27))
ws = WS2812(6, brightness=0.4)
#define the input and output pins

#define colors
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 0, 255)
```

(sonraki sayfaya devam)



(önceki sayfadan devam)

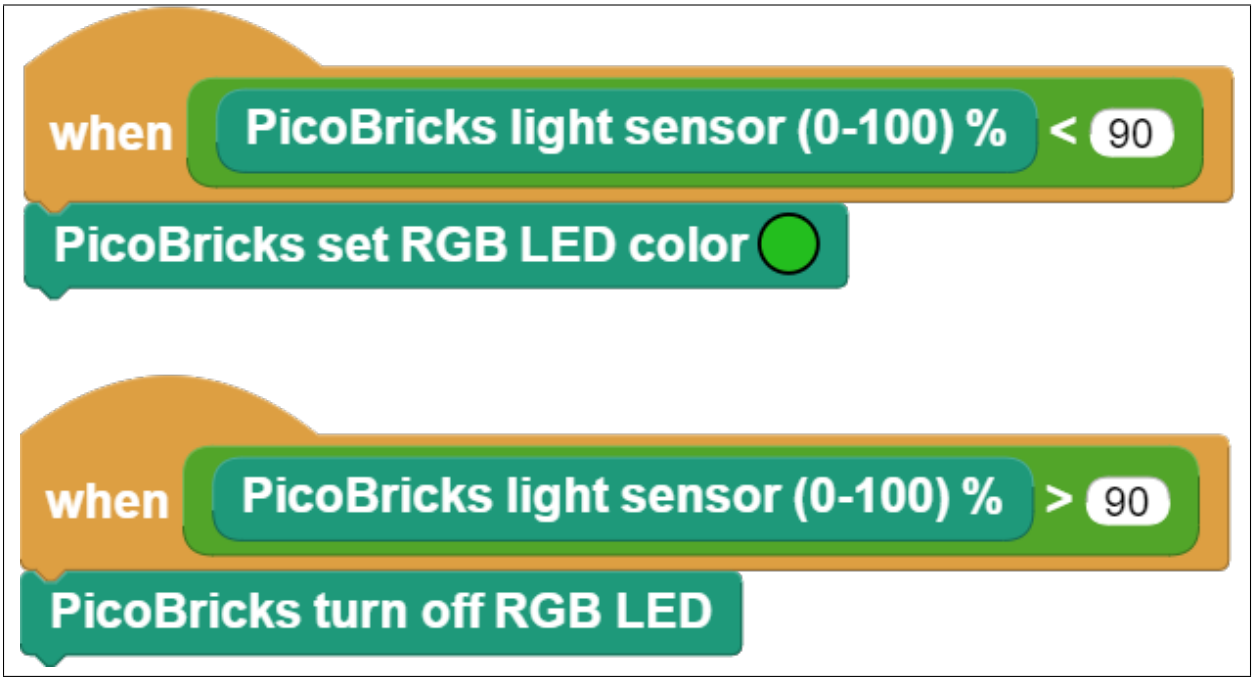
```
COLORS = (RED, GREEN, BLUE)
#RGB color Code

while True:#while loop
 print(ldr.read_u16()) #print the value of the LDR sensor to the screen.

 if(ldr.read_u16()>10000):#let's check the ldr sensor
     for color in COLORS:

         #turn on the LDR
         ws.pixels_fill(color)
         ws.pixels_show()

 else:
     ws.pixels_fill((0,0,0))  #turn off the RGB
     ws.pixels_show()
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptınızda çalışacaktır.

6.3.5 Projenin Arduino C Kodu

```
#include <Adafruit_NeoPixel.h>
#define PIN            6
#define NUMLEDS        1
#define LIGHT_SENSOR_PIN 27

Adafruit_NeoPixel leds = Adafruit_NeoPixel(NUMLEDS, PIN, NEO_GRB + NEO_KHZ800);
//define the libraries

int delayval = 250; // delay for half a second

   void setup()
   {
   leds.begin();
   }

   void loop()
   {
   int analogValue = analogRead(LIGHT_SENSOR_PIN);
   for(int i=0;i < NUMLEDS;i++)
   {
   if (analogValue > 200) {
       // pixels.Color takes RGB values, from 0,0,0 up to 255,255,255
       leds.setPixelColor(i, leds.Color(255,255,255));
       leds.show(); // This sends the updated pixel color to the hardware.
       delay(delayval);
   }
    else {
      leds.setPixelColor(i, leds.Color(0,0,0));  //white color code.
      leds.show(); // This sends the updated pixel color to the hardware.
   }
 }
 delay(10);
}
```

6.3.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.4 Thermometer

6.4.1 Giriş

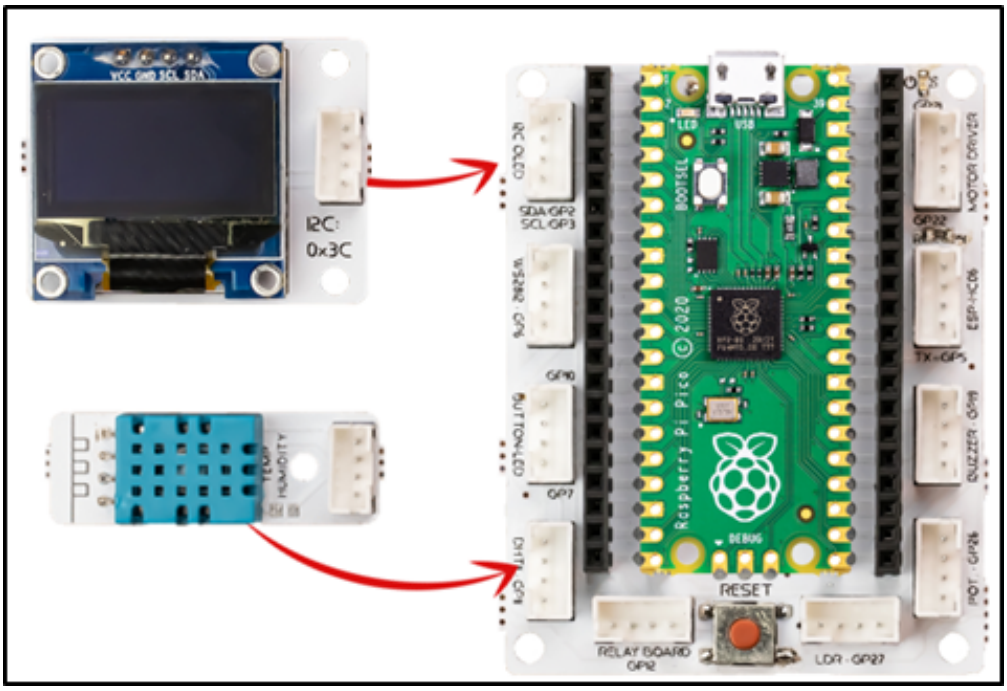
Bu projede Picobricks ile ortam sıcaklığını OLED ekranda gösterecek bir termometre hazırlayacaksın.

Sensörler elektronik sistemlerin duyu organlarıdır. Hissetmek için derimizi, görmek için gözümüzü, duymak için kulağımızı, tatmak için dilimizi ve koklamak için burnumuzu kullanırız. Picobricks'te hali hazırda birçok duyu organı(sensör) vardır. Ayrıca yenileri de eklenebilir. Nem, sıcaklık, ışık ve daha birçok sensörü kullanarak çevreyle etkileşim sağlayabilirsiniz. Picobricks ortam sıcaklığını başka hiçbir çevre bileşenine ihtiyaç duymadan ölçebilir. Ortam sıcaklığı seralarda, kuluçka makinelerinde, ilaçların taşınmasında kullanılan ortamlarda kısaca sıcaklık değişiminin sürekli takip edilmesi gereken durumlarda kullanılmaktadır. Projelerinde sıcaklık değişimi üzerine bir işlem yapacaksan ortam sıcaklığını nasıl ölçeceğini bilmelisin.

6.4.2 Projenin Detayları ve Algoritması

Picobricks'te DHT11 modülü bulunmaktadır. Bu modül ortamdaki sıcaklık ve nem miktarını algılayarak mikrokontrolcüye veriler gönderebilmektedir. Bu bu projede Picobricks üzerindeki DHT11 sıcaklık ve nem sensörünün ölçtüğü sıcaklık değerlerini Picobricks OLED ekrana yazdırmak için gerekli kodları yazacağız.

6.4.3 Bağlantı Diyagramı

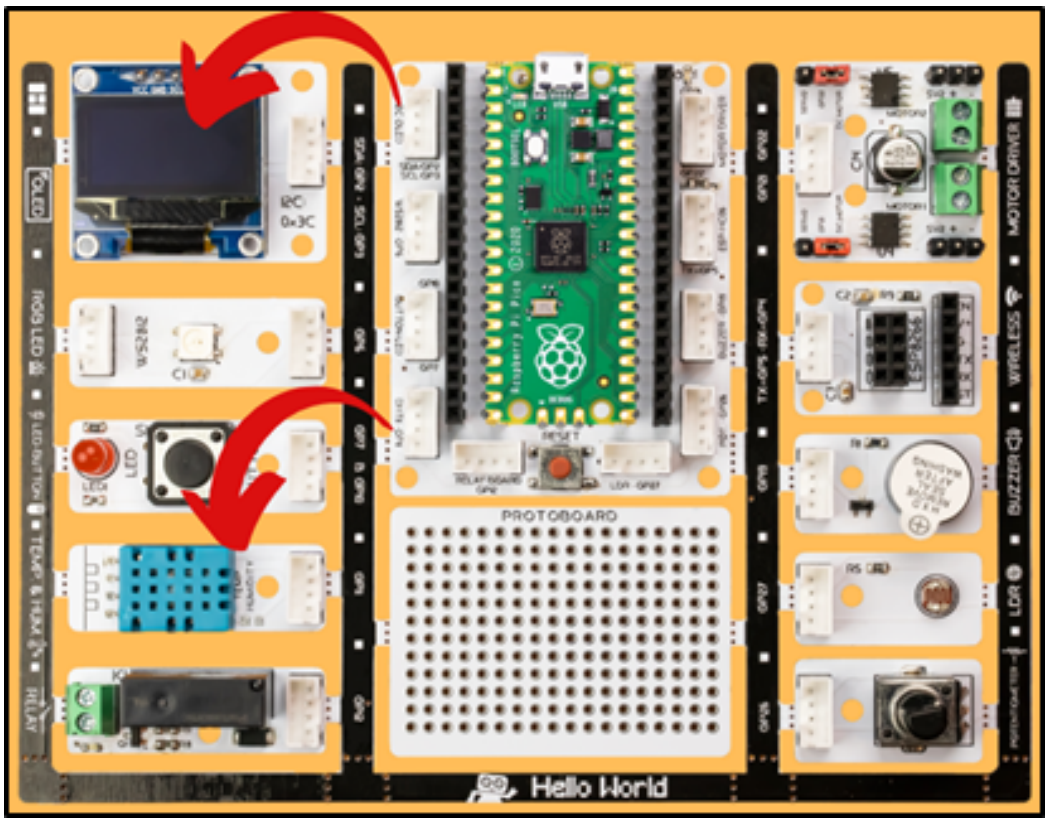


Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.4.4 Projenin MicroPython Kodu

```
from machine import Pin,I2C,ADC #to acces the hardware picobricks
from picobricks import SSD1306_I2C, DHT11 #oled library
import utime #time library
#to acces the hardware picobricks
WIDTH=128
HEIGHT=64
#define the weight and height picobricks

sda=machine.Pin(4)
scl=machine.Pin(5)
#we define sda and scl pins for inter-path communication
i2c=machine.I2C(0, sda=sda, scl=scl, freq=2000000)#determine the frequency values
oled=SSD1306_I2C(WIDTH, HEIGHT, i2c)
pico_temp=DHT11(Pin(11))
current_time=utime.time()
```

(sonraki sayfaya devam)



(önceki sayfadan devam)

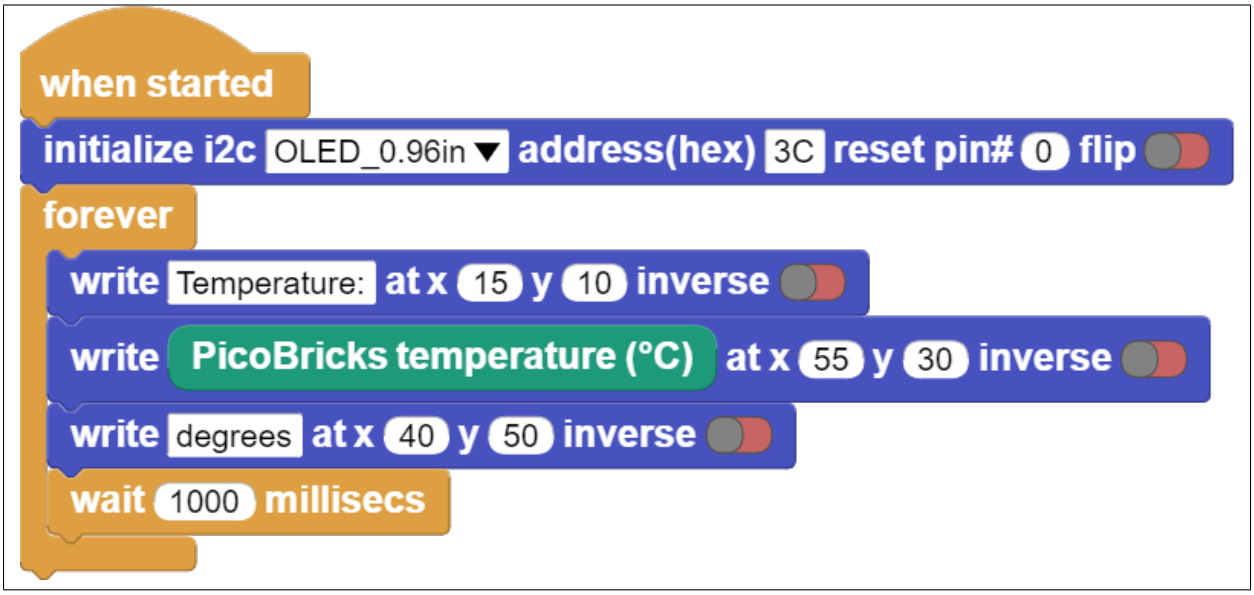
```
while True:
  if(utime.time() - current_time > 2):
      current_time = utime.time()
      pico_temp.measure()
      oled.fill(0)#clear OLED
      oled.show()
      temperature=pico_temp.temperature
      humidity=pico_temp.humidity
      oled.text("Temperature: ",15,10)#print "Temperature: " on the OLED at x=15 y=10
      oled.text(str(int(temperature)),55,25)
      oled.text("Humidty: ", 30,40)
      oled.text(str(int(humidity)),55,55)
      oled.show()#show on OLED
      utime.sleep(0.5)#wait for a half second
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.4.5 Projenin Arduino C Kodu

```
#include <Wire.h>
#include <DHT.h>
#include "ACROBOTIC_SSD1306.h"
#define DHTPIN 11
#define DHTTYPE DHT11
//define the library

DHT dht(DHTPIN, DHTTYPE);
float temperature;
//define the temperature veriable

void setup() {
//define dht sensor and Oled screen
Serial.begin(115200);
dht.begin();
Wire.begin();
oled.init();
oled.clearDisplay();
   }

void loop() {
temperature = dht.readTemperature();
Serial.print("Temp: ");
Serial.println(temperature);
oled.setTextXY(3,1);
oled.putString("Temperature: ");
//print "Temperature: " on the OLED at x=3 y=1
oled.setTextXY(4,3);
oled.putString(String(temperature));
//print the value from the temperature sensor to the oled screen at x=4 y=3
Serial.println(temperature);
delay(100);
   }
```

6.4.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.5 Graphic Monitor

6.5.1 Giriş

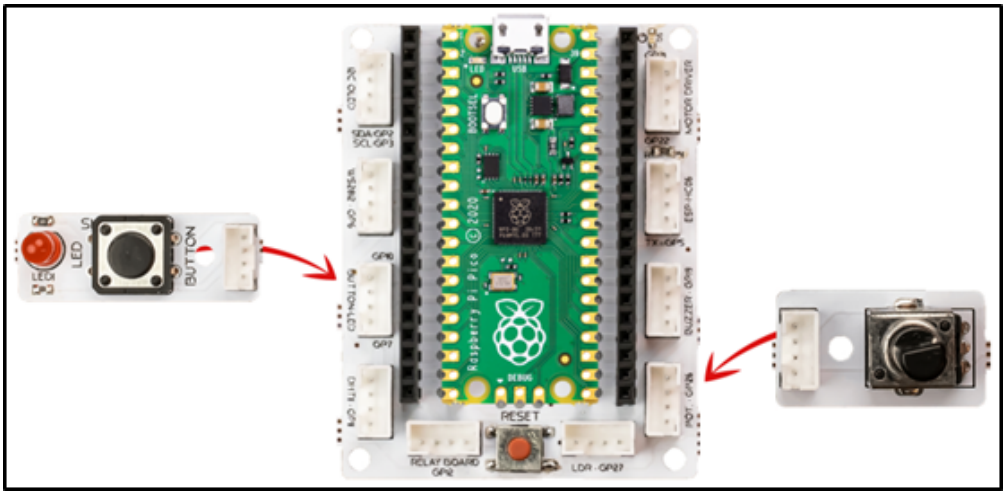
Bu projede potansiyometre ile kırmızı LED'in parlaklığı artırıp azalttığımız bir proje hazırlayacağız. Ayrıca bu işlem sırasında meydana gelen elektriksel değişimi Microblocks grafik monitöründen eş zamanlı olarak takip edeceğiz. Picobricks başladığında potansiyometre değeri sürekli okunarak LED'in parlaklık değerini ayarlanacaktır. Elektrik sinyalinin frekansının değiştirilerek etkisinin azaltıldığı uygulamalara PWM denmektedir. Potansiyometreden okuduğumuz analog değerleri PWM sinyalleri olarak kırmızı LED'e göndereceğiz ve aydınlatma şiddetini ayarlayabileceğiz.

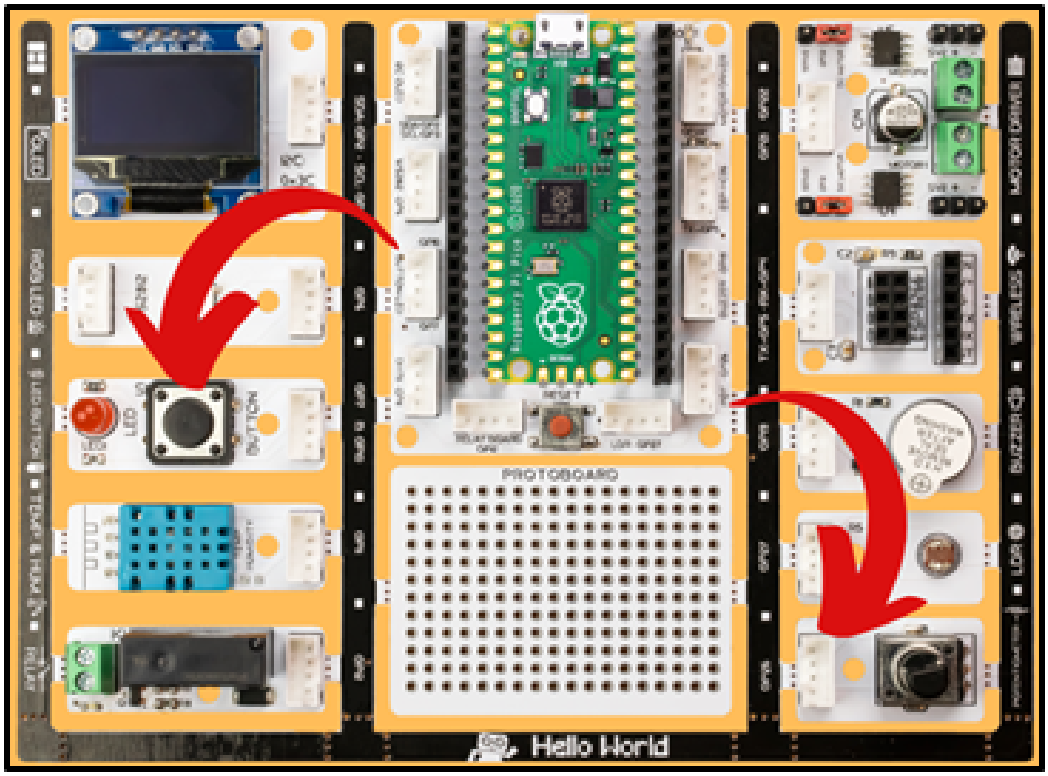
6.5.2 Proje Detayları ve Algoritması

Çevremizdeki elektronik eşyalara baktığımızda değiştirilebilir birçok özelliklerinin olduğunu mühendisler tarafından kullanıcının en çok işine yarayacak şekilde tasarlandıklarını fark edersin. Aydınlatma sistemleri, pişirme sistemleri, ses sistemleri, temizlik sistemleri gibi. bir çok sistem kullanıcısı tarafından çalışma şekli , miktarı, yöntemi vb. özellikleri değiştirilebilir şekilde programlanır. Robotik projelerde ses seviyesini değiştirme, motor hızını değiştirme , ışığın parlaklığını değiştirme işlemlerinde elektrik geriliminin daha düşük veya yüksek etki yaratacak şekilde gönderilmesi sağlanır. Bileşene giden elektrik sinyalinin sıklığı azaltılarak daha düşük seviyede çalışması giden elektrik sinyallerinin sıklığı artırılarak yüksek seviyede çalışması sağlanabilir.

Ekranı olmayan sistemlerde bazı sensörleri ve sistemin çalışmasında görev alan değişkenleri takip etmek için gerçek zamanlı grafik monitörler kullanılır. Arızanın tespit edilmesi için grafik monitörler oldukça kolaylık sağlamaktadır.

6.5.3 Bağlantı Diyagramı





Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.5.4 Projenin MicroPython Kodu

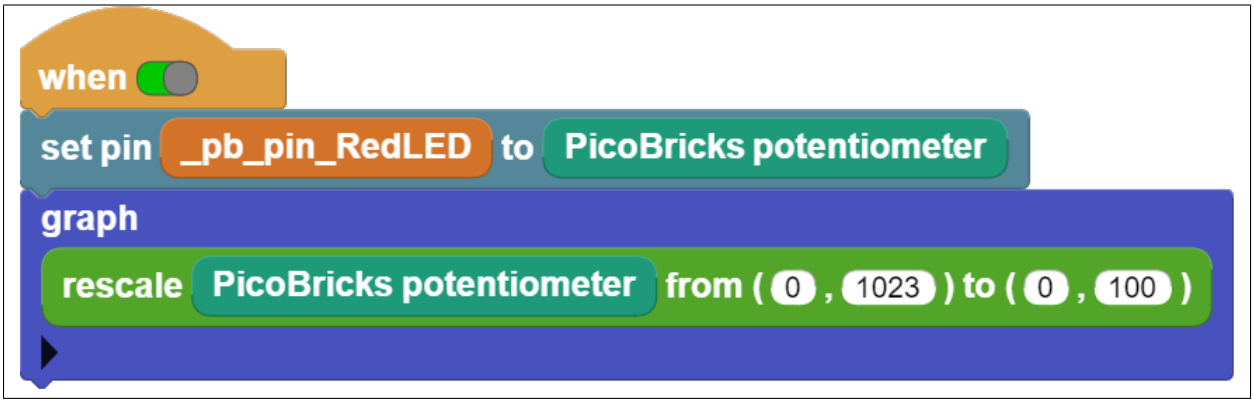
```
from machine import Pin,ADC,PWM
from utime import sleep
#define libraries

led=PWM(Pin(7))
pot=ADC(Pin(26,Pin.IN))
#define the value we get from the led and pot.
led.freq(1000)

while True:#while loop

 led.duty_u16(int((pot.read_u16())))
 print(str(int((pot.read_u16()))))
 #Turn on the LED according to the value from the potentiometer.

 sleep(0.1)#delay
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.5.5 Projenin Arduino C Kodu

```
void setup() {
// put your setup code here, to run once:
pinMode (7,OUTPUT);//initialize digital pin 7 as an output
pinMode (26,INPUT);//initialize digital pin 26 as an input
Serial.begin(9600);//start serial communication


   }

void loop() {
// put your main code here, to run repeatedly:
int pot_val = analogRead(26);
int led_val = map(pot_val, 0, 1023, 0, 255);
analogWrite(7, led_val);
Serial.println(led_val);
//turn on the LED according to the value from the potentiometer

delay(100);//wait


}
```

6.5.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.6 Dominate The Rhythm

6.6.1 Giriş

Kullanıcı şarkıyı başlatmak için butona bastığında, rthm değişkenine göre hesaplanan süre boyunca notaların çalmasını sağlayacak nota kodlarını hazırlayacağız.

6.6.2 Projenin Detayları ve Algoritması

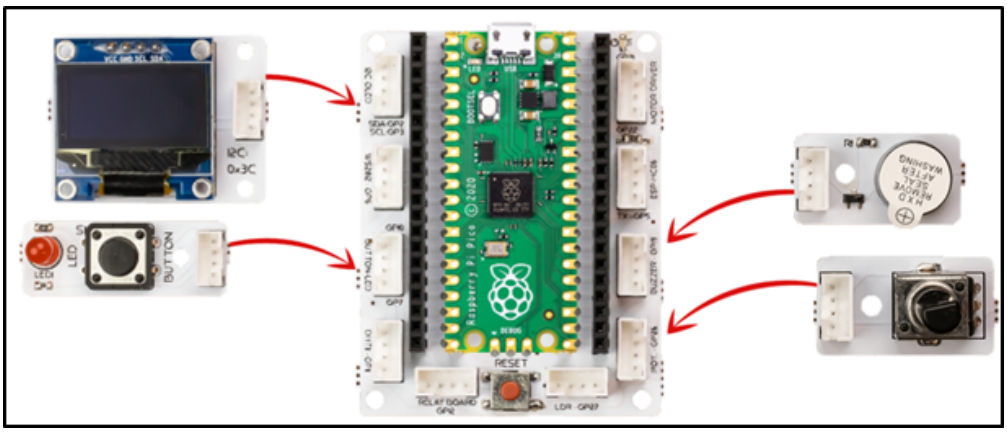
Hayatımızdaki birçok olgu dijitalleştirilmiştir. Bunlardan biri de seslerdir. Sesin tonu ve şiddeti elektriksel olarak işlenebilmektedir. Yani notaları elektronik olarak çıkarabiliriz. Müziği oluşturan seslerin en küçük birimine nota denir. Her notanın bir frekansı ve şiddeti vardır. Yazacağımız kodlarla frekans ve şiddet uygulayarak hangi notanın çalınmasını ve ne kadar sürmesi gerektiğini ayarlayabiliriz.

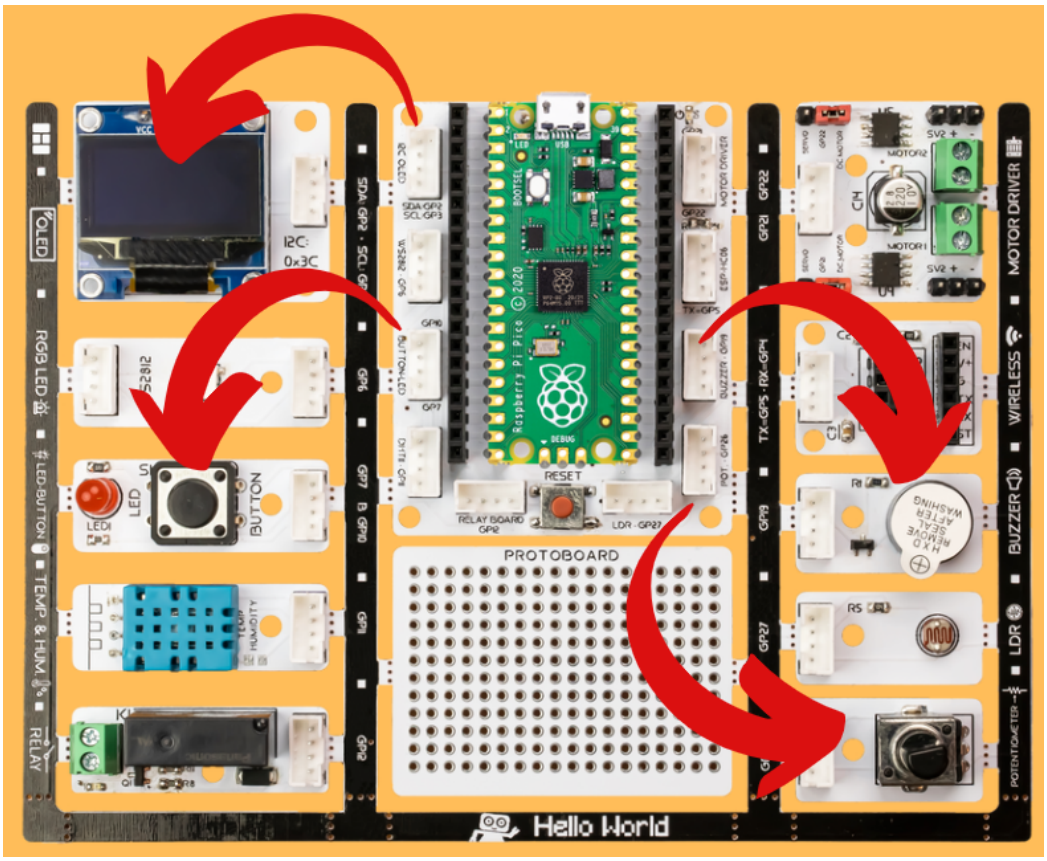
Bu projede PicoBricks ile bir şarkının melodisini buzzer modülünü kullanarak çalacak, potansiyometre modülü ile ritmi ayarlayabilecek bir müzik sistemi hazırlayacağız. Programlama terminolojisinde önemli bir yere sahip değişken kullanımını da bu projede öğreneceksin. Notalarını bildiğimiz her şarkıyı Picobricks ile çalabilirsin. Şarkıyı başlatmak için buton-LED modülünü, şarkının hızını ayarlayabilmek için potansiyometre modülünü notaları çalabilmek için buzzer modülünü kullanacağız. Potansiyometre analog giriş modülüdür. Değişken dirençtir. Üzerinden geçen akım miktarı çevrildikçe bir musluğun açılıp kapatılması gibi artar ve azalır. Bu akım miktarını kodlarla kontrol ederek şarkının hızını ayarlayacağız. Buzzer'lar üzerlerinden geçen akımın şiddetine göre ses seviyelerini, gerilim frekansına göre ses tonlarını değiştirmektedirler. MicroBlocks ile kolayca buzzer modülünden istediğimiz notaları, tonlarını ve sürelerini ayarlayarak kodlayabiliriz.

Projede butona basılma durumunu kontrol edeceğiz. Butona basıldığında melodinin çalmaya başlamasını sağlayacağız. Melodinin çalması sırasında notaların çalınma sürelerini aynı oranda artırıp azaltabilmek için rithm adında bir değişken kullanacağız. Picobricks başladıktan sonra kullanıcının potansiyometre ile ister melodi çalarken ister çalmadan önce rithm değişkenini ayarlayabilmesini sağlayacağız. Picobricks açık olduğu sürece potansiyometre değerini (0-1023) 128'e bölüp rithm değişkene atayacağız. Değişkenler, kullanıcı tarafından ya da sensörler tarafından değiştirilebilecek değerleri kodlarımızda kullanmak istediğimizde kullandığımız veri yapılarıdır. Kullanıcı şarkıyı başlatmak

için butona bastığında ise notaların süreleri rithm değişkenine göre hesaplanan süre boyunca çalmasını sağlayacak nota kodlarını hazırlayacağız.

6.6.3 Bağlantı Diyagramı





Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.6.4 Projenin MicroPython Kodu

```
from machine import Pin,PWM,ADC,I2C #to acces the hardware picobricks
from utime import sleep #time library
from picobricks import SSD1306_I2C
import utime

WIDTH=128
HEIGHT=64
#define the weight and height picobricks

sda=machine.Pin(4)
scl=machine.Pin(5)
#we define sda and scl pins for inter-path communication
i2c=machine.I2C(0, sda=sda, scl=scl, freq=2000000)#determine the frequency values
oled=SSD1306_I2C(WIDTH, HEIGHT, i2c)

button= Pin(10,Pin.IN,Pin.PULL_DOWN)
pot=ADC(Pin(26))
buzzer= PWM(Pin(20))
#determine our input and output pins
pressed = False
rithm = 0

tones = {
"A3": 220,
"D4": 294,
"E4": 330,
"F4": 349
   }
#define the tones

mysong = ["A3","E4","E4","E4","E4","E4","E4","F4","E4","D4","F4","E4"]#let's define the␣
↪tones required for our song in the correct order into a sequence
noteTime = [1,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,1]#define wait times between tones␣
↪into an array


def playtone(frequency):
 buzzer.duty_u16(6000)
 buzzer.freq(frequency)
#define the frequencies of the buzzer
def playsong(pin):
 global pressed
 pressed = True
#play the tones with the right cooldowns
#An finally we need to tell the pins when to trigger, and the function to call when they␣
↪detect an event:
button.irq(trigger=Pin.IRQ_RISING, handler=playsong)
note_count = 9999
played_time = 0
while True:
 current_time = utime.ticks_ms()
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
oled.show()
oled.text("Press the button",0,0)

if (note_count < len(mysong)):
    oled.fill(0)
    oled.text("Dominate ",30,10)
    oled.text("the ",45,25)
    oled.text("Rhythm ",35,40)
    rithm=((pot.read_u16()/65535.0)*20) +1
    if (current_time - played_time)/1000.0 >= noteTime[note_count]/rithm:
        played_time = utime.ticks_ms()
        playtone(tones[mysong[note_count]])
        note_count += 1
else:
    buzzer.duty_u16(0)

if pressed:

    note_count = 0
    pressed = False
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.6.5 Projenin Arduino C Kodu

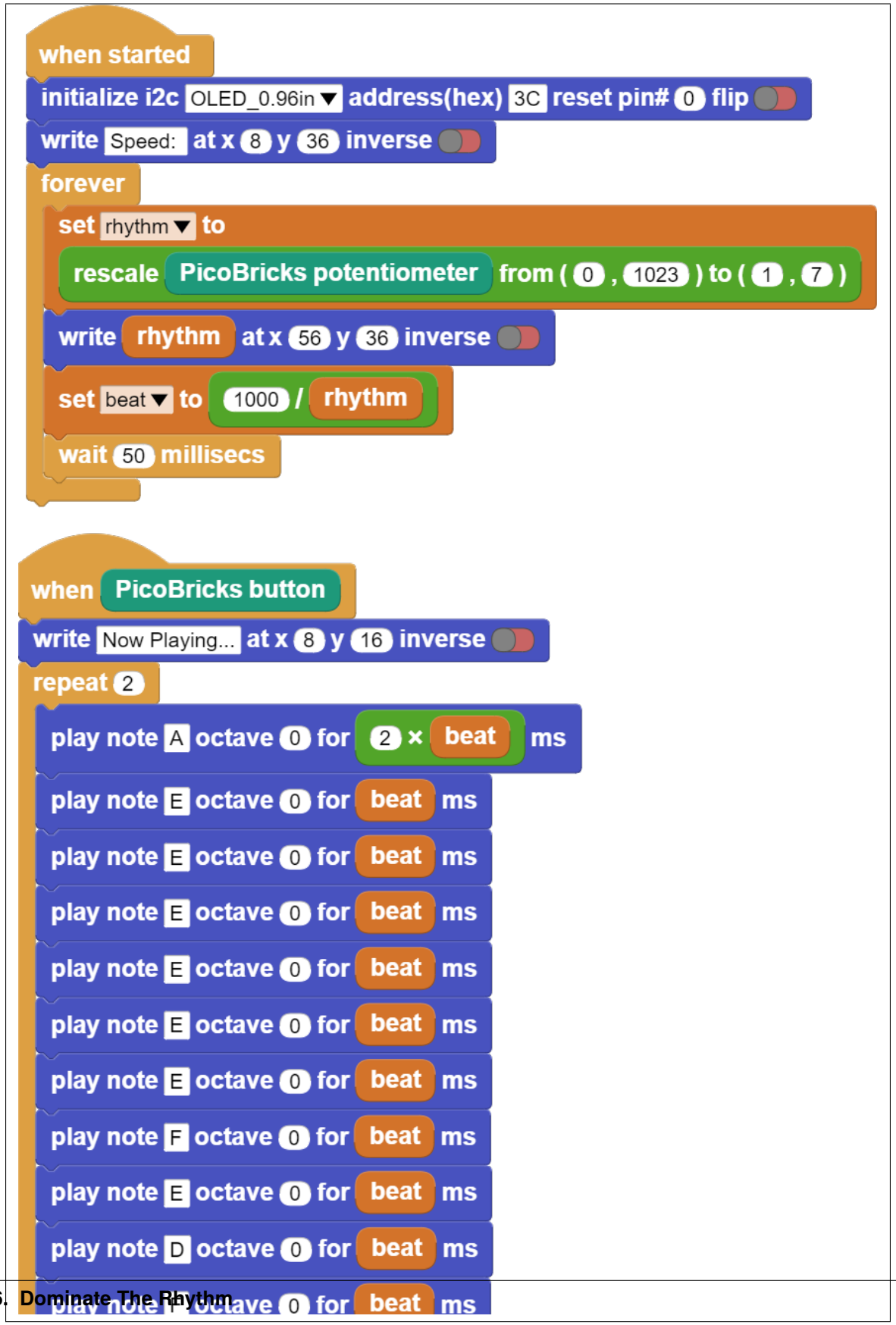
```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int buzzer = 20;
int pot =26;
int button= 10;
//define the buzzer, pot and button

int Re = 294;
int Mi = 330;
int Fa = 349;
int La = 440;
//DEFİNE THE TONES
void setup()
   {
Wire.begin();
oled.init();
oled.clearDisplay();

pinMode(buzzer,OUTPUT);
pinMode(26,INPUT);
pinMode(button,INPUT);
//determine our input and output pins
   }
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
void loop()
   {
int rithm = (analogRead(pot))/146;
String char_rithm = String(rithm);
oled.setTextXY(3,4);
oled.putString("Speed: ");
oled.setTextXY(3,10);
oled.putString(char_rithm);

//print "Speed: "  and speed value on the OLED at x=3 y=4

delay(10);

if (digitalRead(button) == 1){

 oled.clearDisplay();
 oled.setTextXY(3,2);
 oled.putString("Now playing...");
 //print "Speed: "  and speed value on the OLED at x=3 y=4
 tone(buzzer, La); delay (1000/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Fa); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (500/(rithm+1));
 tone(buzzer, Re); delay (500/(rithm+1));
 tone(buzzer, Fa); delay (500/(rithm+1));
 tone(buzzer, Mi); delay (1000/(rithm+1));

 //play the notes in the correct order and time when the button is pressed

 oled.clearDisplay();
 //clear the screen
   }
 noTone(buzzer);
 //stop the buzzer
}
```

6.6.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.7 Show Your Reaction

6.7.1 Giriş

Bu projede her programlama dilinde kullanılan rastgelelik durumunu öğreneceksin. Picobricks ile OLED ekran, Buton-LED ve Buzzer modülünü kullanarak elektronik bir sistem geliştireceğiz. PicoBricks açılır açılmaz bir zamanlayıcı çalışmaya başlar. Bu zamanlayıcı ile saniyenin binde 1'ini ölçebiliriz.

6.7.2 Projenin Detayları ve Algoritması

Şimdi de dikkat ve refleks geliştiren bir oyun hazırlayacağız. Hızlı hareket etmek ve dikkatin uzun süre sağlanabilmesi çocukların önemli gelişimsel özelliklerindendir. Okul öncesi ve ilkokul dönemindeki çocukların dikkat sürelerini ve reflekslerini artırıcı etkinlikler yapması çocukların hoşlarına gittiği gibi ebeveynlerinin ve öğretmenlerinin de istediği bir durumdur. Hazırlayacağımız elektronik sistemi dikkat artırıcı ve refleks geliştirici bir oyun olacak. Projeyi bitirdikten sonra siz de arkadaşlarınla yarışabilirsin. :)

Zamanlayıcılar günlük yaşamda birçok elektronik sistemde kullanılırlar. Zaman ayarlı aydınlatmalar, fırınlar, ütüler, mutfak robotları…

Projemiz çalışmaya başladığında OLED ekranda karşılama mesajı görüntüleyeceğiz. Ardından kullanıcıya oyuna başlaması için yapması gerekeni ekrana yazdıracağız. Oyuna başlayabilmek için butona basılacak butona basıldıktan sonra 3'ten geriye doğru ekranda sayılarak oyuncunun hazırlanmasını isteyeceğiz. Geri sayımın bitiminden sonra 2-10 saniye arasında rastgele bir süre içinde kırmızı LED yanacak. Kırmızı LED yandıktan sonra hemen zamanlayıcıyı sıfırlayacağız. Tekrar butona basılır basılmaz zamanlayıcıyı ölçeceğiz. Elde ettiğimiz bu değer milisaniye cinsinden olacak. Oyuncunun tepki süresi olarak bu değeri ekranda göstereceğiz.

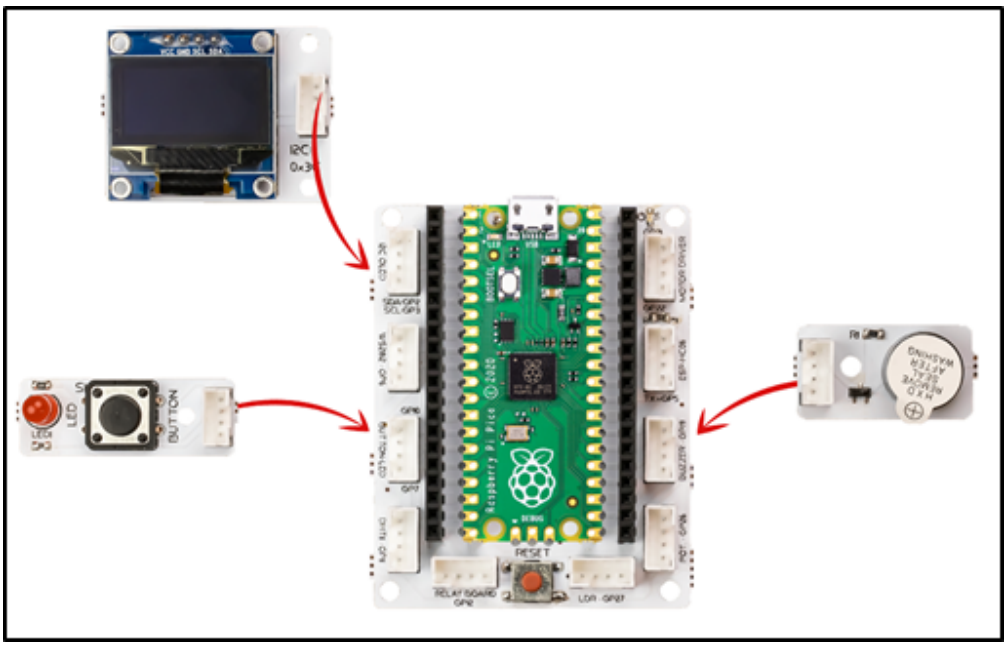
6.7.3 Bağlantı Diyagramı

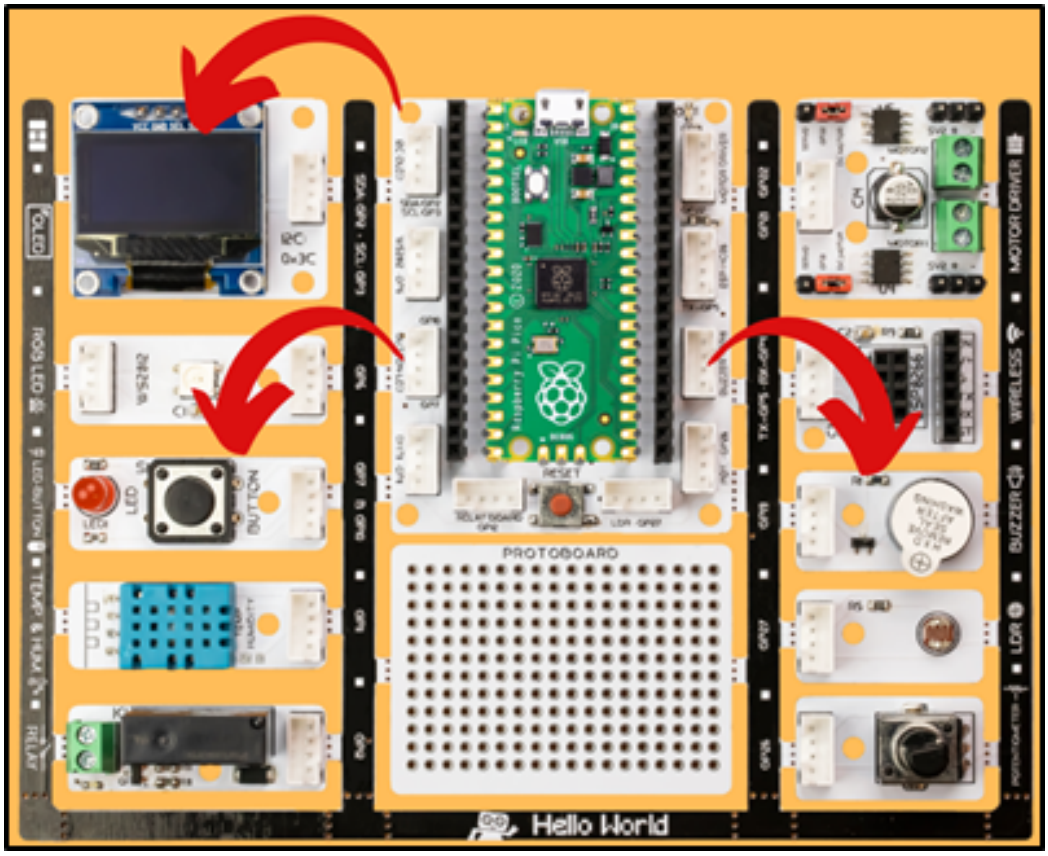
Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.7.4 Projenin MicroPython Kodu

```
from machine import Pin, I2C,Timer
from picobricks import SSD1306_I2C
import utime
import urandom
#define the library
WIDTH=128
HEIGHT=64
#define the width and height values
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=2000000)
```

(sonraki sayfaya devam)


LED
BUTTON
RESET


OLED
LED
BUTTON
PROTOBOARD
MOTOR DRIVER
WIRELESS
BUZZER
LDR
Hello World

(önceki sayfadan devam)

```
oled= SSD1306_I2C(WIDTH, HEIGHT, i2c)
button = Pin(10,Pin.IN,Pin.PULL_DOWN)
led=Pin(7,Pin.OUT)
#define our input and output pins
while True:
 led.value(0)
 oled.fill(0)
 oled.text("press the button",0,10)
 oled.text("TO START!",25,25)
 oled.show()
 #print "Press the button" and "TO START!" on the OLED screen
 while button.value()==0:
     pass
 oled.fill(0)
 oled.text("Wait For LED",15,30)
 oled.show()
 #write "wait for LED" on the screen when the button is pressed
 utime.sleep(urandom.uniform(1,5))
 led.value(1)
 timer_start=utime.ticks_ms()
 #wait for a rondom second and turn on the led
 while button.value()==0:
     pass
 timer_reaction=utime.ticks_diff(utime.ticks_ms(), timer_start)
 pressed=True
 oled.fill(0)
 oled.text("Your Time",25,25)
 oled.text(str(timer_reaction),50,50)
 oled.show()
 led.value(0)
 utime.sleep(1.5)
 #print the score and "Your Time" to the screen when the button is pressed.
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.7.5 Projenin Arduino C Kodu

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"
//define the library

int buzzer=20;
int button=10;
int led=7;
int La=440;

int old_time=0;
int now_time=0;
int score=0;
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

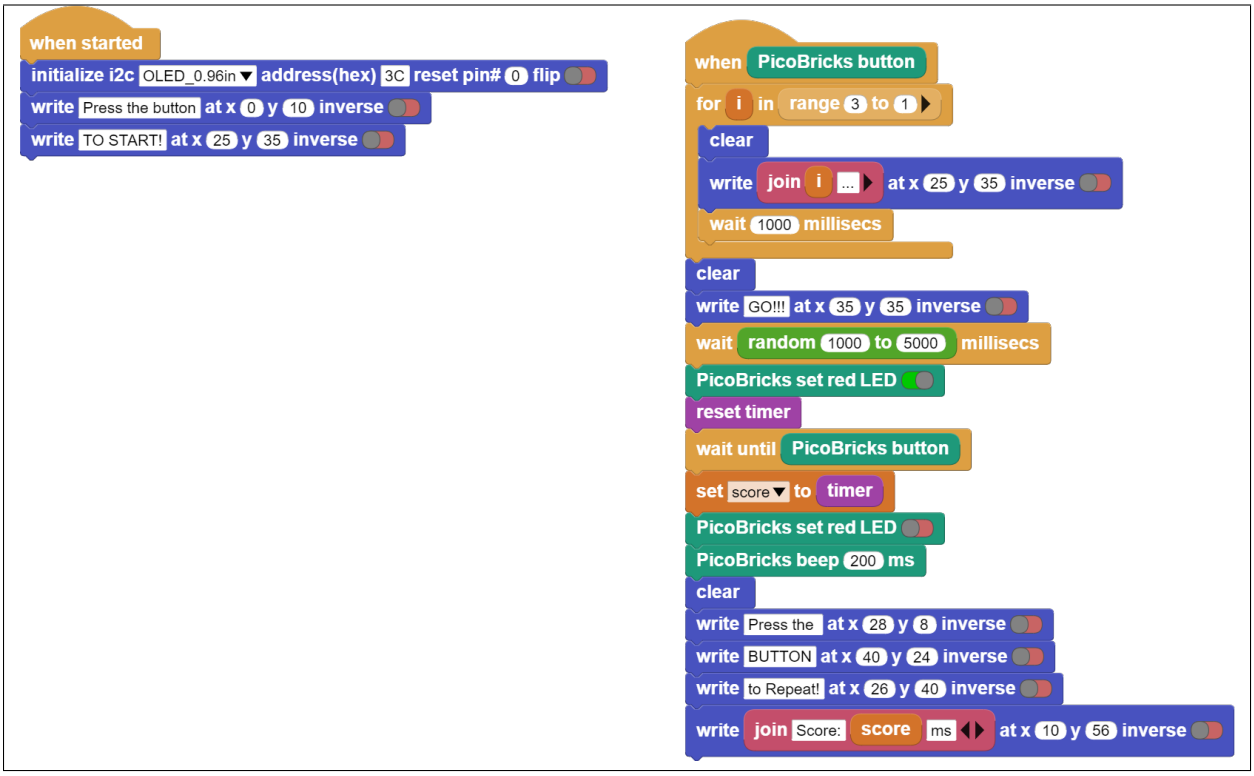
```
String string_score;
//define our integer and string veriables


void setup() {
// put your setup code here, to run once:
Wire.begin();
oled.init();
oled.clearDisplay();

pinMode(led,OUTPUT);
pinMode(buzzer,OUTPUT);
pinMode(button,INPUT);
Serial.begin(9600);
//define the input and output pins

 }

void loop() {
// put your main code here, to run repeatedly:
oled.setTextXY(3,0);
oled.putString("Press the button");
oled.setTextXY(5,4);
oled.putString("TO START");

if(digitalRead(button)==1){
 for(int i=3; i>0; i--){

   String string_i=String(i);
   oled.clearDisplay();
   oled.setTextXY(4,8);
   oled.putString(string_i);
   delay(1000);

 }
 //count backwards from three

 oled.clearDisplay();
 oled.setTextXY(4,6);
 oled.putString("GO!!!");
 //print "GO!!" on the OLED at x=4 y=6

 int random_wait= random(1000, 5000);
 delay(random_wait);
 //wait for a random second between 1 and 5

 digitalWrite(led, HIGH);
 old_time=millis();
 //turn on LED
 while(!(digitalRead(button)==1)){

   now_time=millis();
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
  score=now_time-old_time;
  string_score= String(score);
  //save score as string on button press

}
digitalWrite(led, HIGH);
tone(buzzer, La);
delay(200);
noTone(buzzer);
//turn on LED and buzzer

oled.clearDisplay();
oled.setTextXY(1,4);
oled.putString("Press the");
//print "Press the" on the OLED at x=1 Y=4

oled.setTextXY(2,3);
oled.putString("RESET BUTTON");
//print "RESET BUTTON" on the OLED at X=1 Y=4

oled.setTextXY(3,3);
oled.putString("To Repeat!");
//print "To Repeat!" on the OLED at X=3 Y=3

oled.setTextXY(6,3);
oled.putString("Score: ");
oled.setTextXY(6,9);
oled.putString(string_score);
oled.setTextXY(6,13);
oled.putString(" ms");
Serial.println(score);
//print score value to screen

delay(10000);
oled.clearDisplay();
//wait ten seconds and clear the screen
}

    }
```

6.7.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.8 My Timer

6.8.1 Giriş

Bu projede Picobircks ile OLED ekran, buton ve potansiyometre modüllerini kullanarak kendi zaman ölçme aygıtını yapacaksın. Bir Timer…

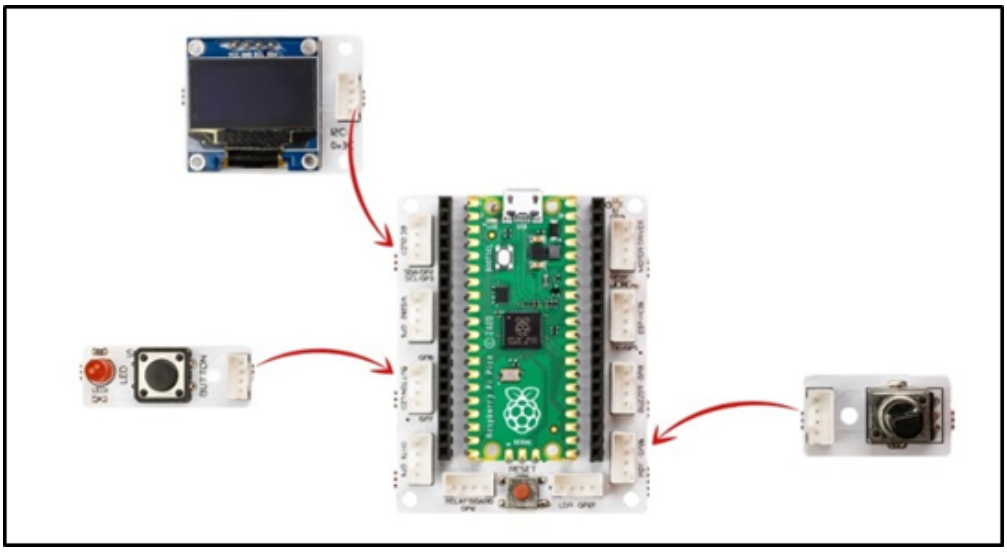
6.8.2 Projenin Detayları ve Algoritması

Zaman ölçmek günlük hayatımızda farkına varmadan yaptığımız basit ama önemli bir iştir. Ameliyattaki cerrah, toplantısına yetişmeye çalışan bir iş insanı, kazanmaya çalışan sporcu, sınavı bitirmeye çalışan bir öğrenci ya da satranç müsabakası… Zaman ölçmek için akıllı kol saatleri, telefonlar hatta profesyonel kronometreler kullanılmaktadır. Elektronik sistemler içinde zaman oldukça doğru kullanılması gereken bir değişkendir. Örneğin bir Çamaşır makinesi; tamburun ne kadar süre saat yönünde ne kadar saat yönü tersine döneceği, deterjanı eritip alabilmek için kaç sn su akması gerektiği hep zaman ölçerek yapılan görevlerdir. Zamanın önemli olduğu projeler geliştirmek için onu nasıl kullanacağını bilmelisin.

PicoBricks başladığında ekrana projeyi tanıtan ve yönerge içeren bir ifade yerleştirelim. Kullanıcı potansiyometreyi çevirdikce `0-60 dakika` aralığında bir süre belirleyecek. Kullanıcı potansiyometre ile süreye karar verdikten sonra

Picobricks'in butonuna bastığında dakika saniye ve salise ekranda geri doğru saymaya başlayacak. Eğer zaman geriye doğru akarken butona basılırsa Timer duracak ve kalan süreyi ekranda gösterecek. Butona basılmadan dakika, saniye ve salise sıfır değerine ulaşırsa ekrana sürenin dolduğunu ifade eden bildirim gösterilecek ve program durdurulacak.

6.8.3 Bağlantı Diyagramı

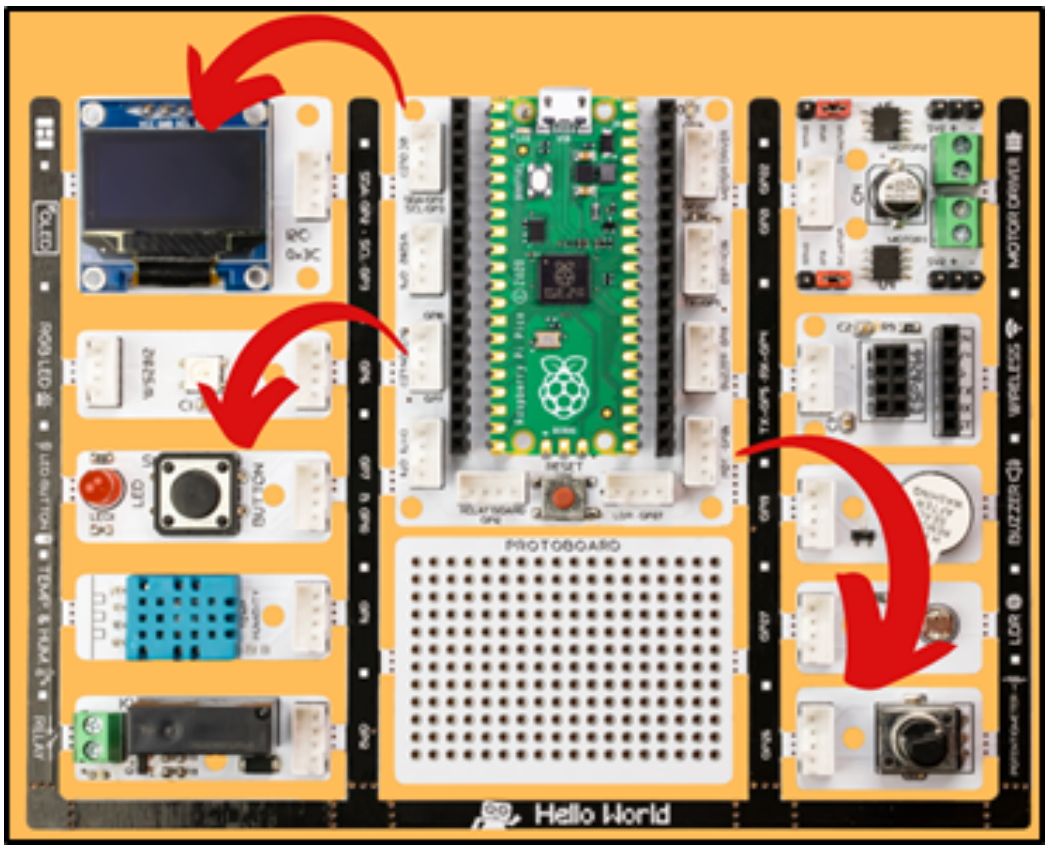


Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.8.4 Projenin MicroPython Kodu

```
from machine import Pin, I2C, ADC, Timer #to acces the hardware picobricks
from picobricks import SSD1306_I2C #oled library
import utime #time library

WIDTH  = 128
WEIGHT = 64
#define the width and height values

sda=machine.Pin(4)
scl=machine.Pin(5)
#we define sda and scl pins for inter-path communication
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)#determine the frequency values

oled = SSD1306_I2C(128, 64, i2c)
pot = ADC(Pin(26))
button = Pin(10,Pin.IN,Pin.PULL_DOWN)
#determine our input and output pins

oled.fill(0)
oled.show()
#Show on OLED
```

(sonraki sayfaya devam)



(önceki sayfadan devam)

```
time=Timer()
time2=Timer()
time3=Timer()
#define timers

def minute(timer):
 global setTimer
 setTimer -=1

def second(timer):
 global sec
 sec-=1
 if sec==-1:
     sec=59

def msecond(timer):
 global msec
 msec-=1
 if msec==-1:
     msec=99
#We determine the increments of the minute-second and millisecond values.
sec=59
msec=99
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
global setTimer

while button.value()==0:
 setTimer=int((pot.read_u16()*60)/65536)+1
 oled.text("Set timer:" + str(setTimer) + " min",0,12)
 oled.show()
 utime.sleep(0.1)
 oled.fill(0)
 oled.show()
#If the button is not pressed, the value determined by the potentiometer is printed on␣
↪the OLED screen.

setTimer-=1

time.init(mode=Timer.PERIODIC,period=60000, callback=minute)
time2.init(mode=Timer.PERIODIC,period=1000, callback=second)
time3.init(mode=Timer.PERIODIC,period=10, callback=msecond)
#We determine the periods of minutes, seconds and milliseconds.
utime.sleep(0.2)#wait for 0.2 second

while button.value()==0:
 oled.text("min:" + str(setTimer),50,10)
 oled.text("sec:" + str(sec),50,20)
 oled.text("ms:" + str(msec),50,30)
 oled.show()
 utime.sleep(0.008)
 oled.fill(0)
 oled.show()
 if(setTimer==0 and sec==0 and msec==99):
     utime.sleep(0.1)
     msec=0
     break;
 #When the button is pressed, it prints the min-sec-ms values to the OLED screen in the␣
↪determined x and y coordinates.

 oled.text(str(setTimer),60,10)
 oled.text(str(sec),60,20)
 oled.text(str(msec),60,30)
 oled.text("Time is Over!",10,48)
 oled.show()
 #Print the minutes, seconds, milliseconds and "Time is Over" values to the X and Y␣
↪coordinates determi
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.8.5 Projenin Arduino C Kodu

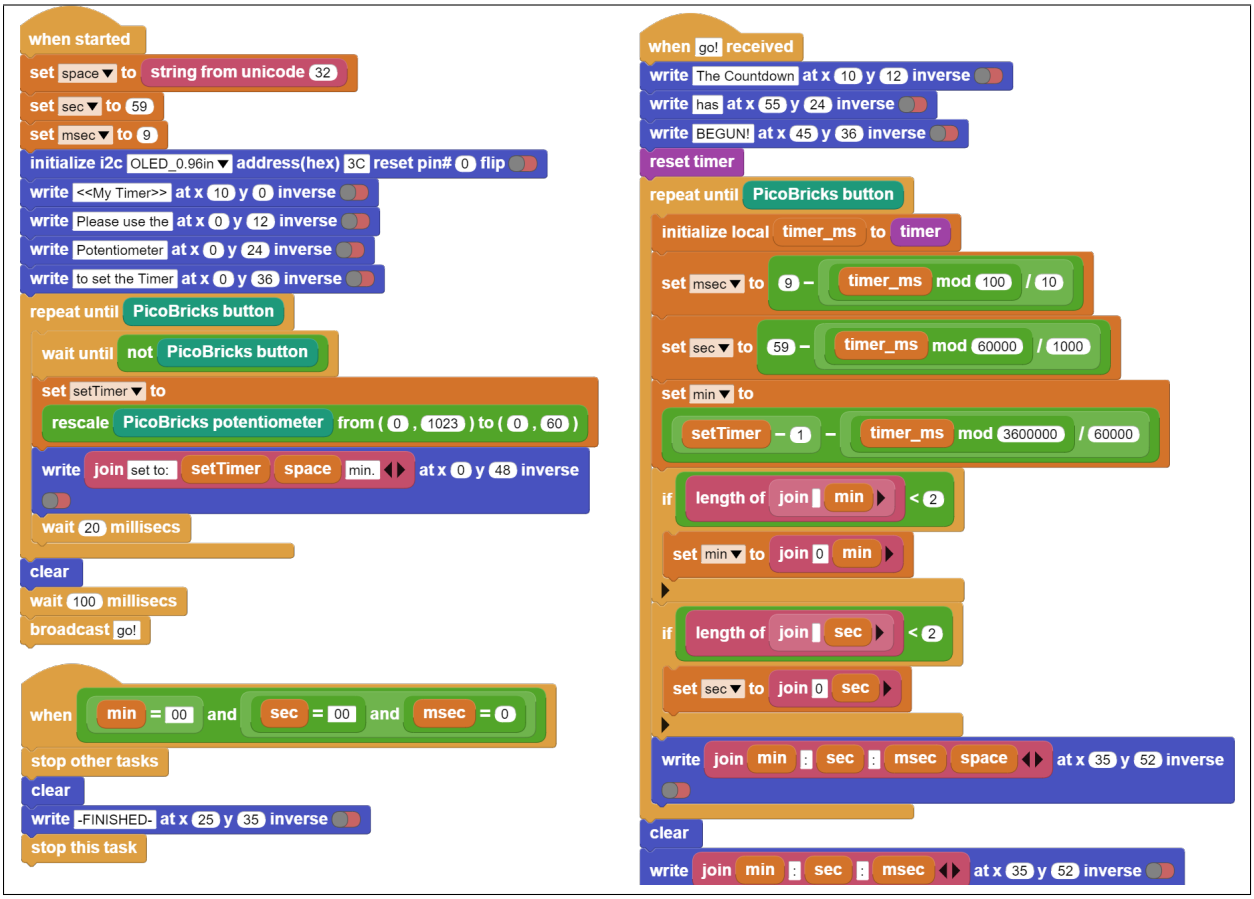
```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int minute;
int second = 59;
int milisecond = 9;
int setTimer;

void setup() {
// put your setup code here, to run once:
pinMode(10,INPUT);
pinMode(26,INPUT);

Wire.begin();
oled.init();
oled.clearDisplay();


  }

void loop() {
// put your main code here, to run repeatedly:
oled.setTextXY(1,2);
oled.putString("<<My Timer>>");
oled.setTextXY(3,1);
oled.putString("Please use the");
oled.setTextXY(4,1);
oled.putString("Potantiometer");
oled.setTextXY(5,0);
oled.putString("to set the Timer");
delay(3000);
oled.clearDisplay();

  while(!(digitalRead(10) == 1))
  {
  setTimer = (analogRead(26)*60)/1023;
  oled.setTextXY(3,1);
  oled.putString("set to:");
  oled.setTextXY(3,8);
  oled.putString(String(setTimer));
  oled.setTextXY(3,11);
  oled.putString("min.");
  }
  oled.clearDisplay();
  oled.setTextXY(1,1);
  oled.putString("The Countdown");
  oled.setTextXY(2,3);
  oled.putString("has begin!");

  while(!(digitalRead(10) == 1))
      {
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
  milisecond = 9- (millis()%100)/10;
  second = 59-(millis()%60000)/1000;
  minute = (setTimer-1)-((millis()%360000)/60000);

  oled.setTextXY(5,3);
  oled.putString(String(minute));
  oled.setTextXY(5,8);
  oled.putString(String(second));
  oled.setTextXY(5,13);
  oled.putString(String(milisecond));
  oled.setTextXY(5,6);
  oled.putString(":");
  oled.setTextXY(5,11);
  oled.putString(":");
      }
  oled.setTextXY(5,3);
  oled.putString(String(minute));
  oled.setTextXY(5,8);
  oled.putString(String(second));
  oled.setTextXY(5,13);
  oled.putString(String(milisecond));
  oled.setTextXY(5,6);
  oled.putString(":");
  oled.setTextXY(5,11);
  oled.putString(":");
  delay(10000);

  if (minute==0 & second==0 & milisecond==0){

  oled.setTextXY(5,3);
  oled.putString(String(minute));
  oled.setTextXY(5,8);
  oled.putString(String(second));
  oled.setTextXY(5,13);
  oled.putString(String(milisecond));
  oled.setTextXY(5,6);
  oled.putString(":");
  oled.setTextXY(5,11);
  oled.putString(":");
  oled.putString("-finished-");
  oled.setTextXY(7,5);
  delay(10000);
  }


      }
```

6.8.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.9 Alarm Clock

6.9.1 Giriş

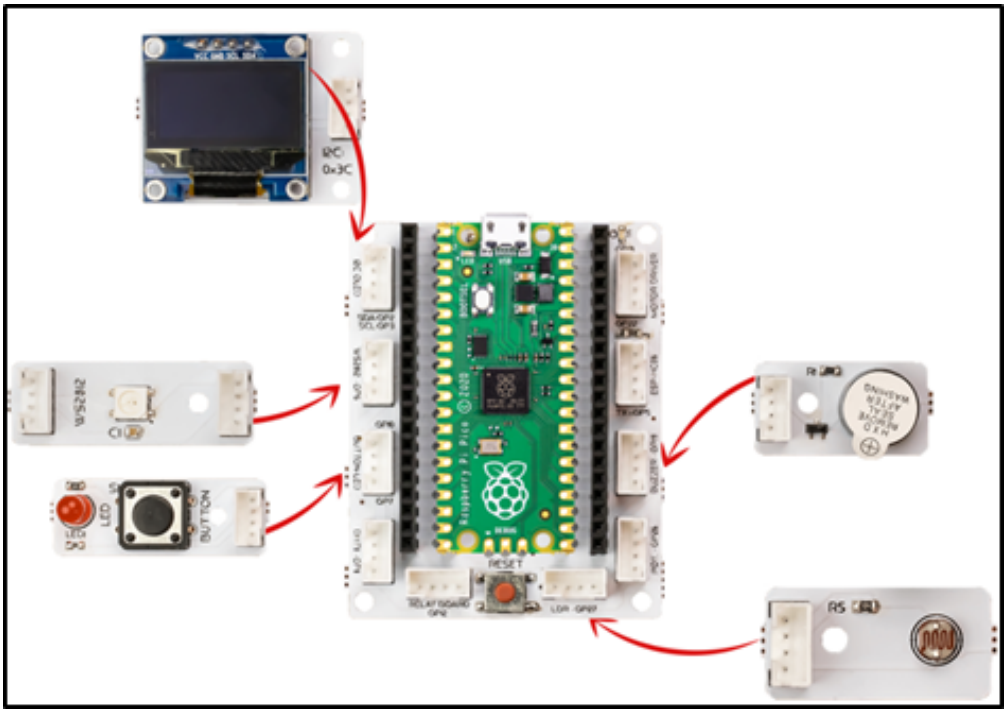
Bu projede Picobricks'teki ışık sensörünü kullanarak gün ışığına göre ayarlanan bir saat alarmı hazırlayacağız. Bu projede basit bir alarm uygulaması yapacağız. Tasarlayacağımız alarm sistemi sabah olduğunda otomatik olarak çalacak şekilde kurgulanmıştır. Bunun için projede LDR sensör kullanacağız.

6.9.2 Proje Detayları ve Algoritması

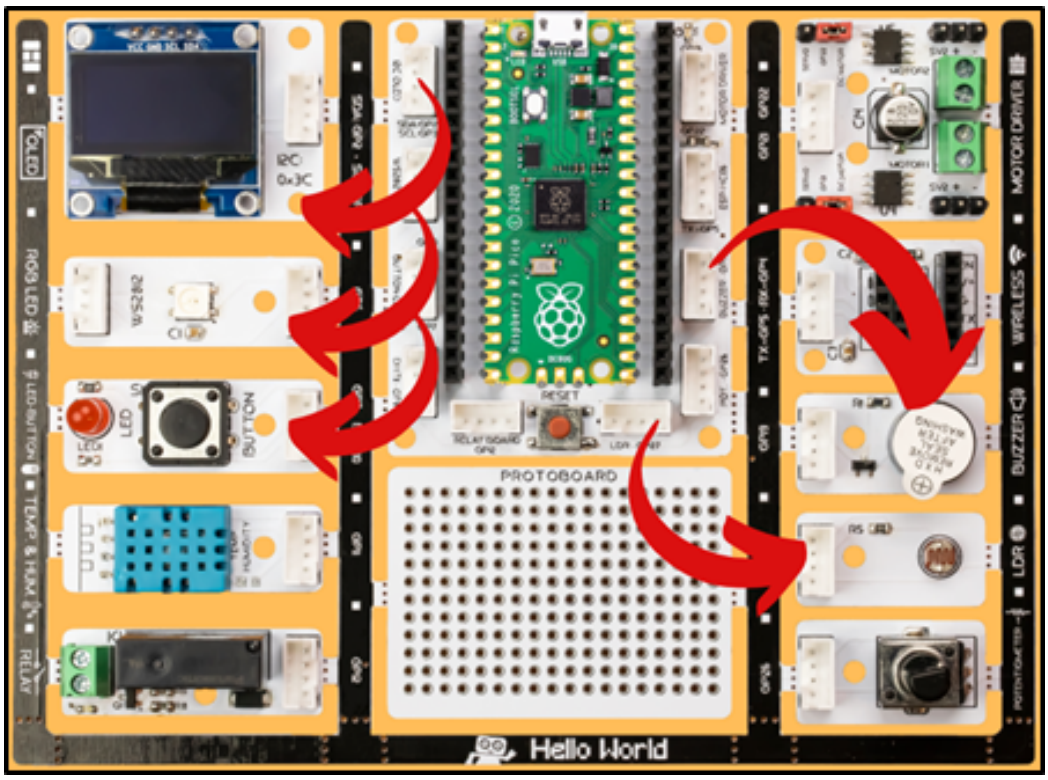
Küresel ısınma dünyamızın iklimini her geçen gün daha kötü etkiliyor. Ülkeler küresel ısınmanın etkilerini azaltmak için birçok tedbiri devreye sokuyor ve anlaşmalar imzalıyorlar. Yenilenebilir enerji kaynaklarının kullanılması ve enerjinin verimli kullanılması fabrikalardan evimizin odalarına kadar her yerde dikkat edilmesi gereken bir konudur. Şehirlerde yol ve park aydınlatmalarının insan hatasından dolayı açık kalması, yüksek enerji tüketen aydınlatma araçlarının kullanılması gibi birçok sebep enerji verimliliğini düşürmektedir. Ortamın ışık, sıcaklık ve nem değerlerini ölçerek sadece gerek duyulduğunda ve doğru miktarlarda kullanılmasını sağlayan birçok elektronik ve dijital sistem mühendisler tarafından geliştirilmekte ve programlanmaktadır.

Gece olduğunda OLED ekranda kullanıcıya iyi geceler mesajı görüntülenecek, sabah olduğunda ise buzzer sesi ile alarm çalacak, ekranda kullanıcıya günaydın mesajı gösterilecek ve ışıklı bildirim amacıyla RGB LED beyaz renkte yanacak. Kullanıcının alarmı durdurması için ise Picobricks'in butonuna basması gerekecek. Alarm durdurulana kadar devam eden bu işlemlerden sonra butona basıldığında buzzer ve RGB LED kapanacak ve OLED ekranda kullanıcıya iyi günler mesajı gösterilecek.

6.9.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.



6.9.4 Projenin MicroPython Kodu

```
from machine import Pin, I2C, ADC, PWM#to access the hardware on the pico
from picobricks import SSD1306_I2C#OLED Screen Library
import utime
from picobricks import WS2812#ws8212 library

#OLED Screen Settings
WIDTH  = 128
HEIGHT = 64

sda=machine.Pin(4)
scl=machine.Pin(5)
#initialize digital pin 4 and 5 as an OUTPUT for OLED Communication

i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
neo = WS2812(pin_num=6, num_leds=1, brightness=0.3)#initialize digital pin 6 as an␣
↪OUTPUT for NeoPixel

oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)
ldr = ADC(Pin(27))#initialize digital pin 6 as an OUTPUT for NeoPixel
button = Pin(10,Pin.IN,Pin.PULL_DOWN)#initialize digital pin 10 as an INPUT for button
buzzer = PWM(Pin(20, Pin.OUT))#initialize digital pin 20 as an OUTPUT for buzzer
buzzer.freq(1000)

BLACK = (0, 0, 0)
WHITE = (255, 255, 255)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
#RGB black and white color code
oled.fill(0)
oled.show()

neo.pixels_fill(BLACK)
neo.pixels_show()

if ldr.read_u16()<4000:
wakeup = True
    else:
wakeup = False

while True:
while wakeup==False:
    oled.fill(0)
    oled.show()
    oled.text("Good night",25,32)
    oled.show()
    #Show on OLED and print "Good night"
    utime.sleep(1)
    if ldr.read_u16()<4000:
        while button.value()==0:
            oled.fill(0)
            oled.show()
            oled.text("Good morning",15,32)
            oled.show()
            #Print the minutes, seconds, milliseconds and "Goog morning" values to the X␣
↪and Y coordinates determined on the OLED screen.
            neo.pixels_fill(WHITE)
            neo.pixels_show()
            buzzer.duty_u16(6000)
            utime.sleep(1)
            #wait for one second
            buzzer.duty_u16(0)
            utime.sleep(0.5)
            #wait for half second
            wakeup=True
        neo.pixels_fill(BLACK)
        neo.pixels_show()
oled.fill(0)
oled.show()
oled.text("Have a nice day!",0,32)
#Print the minutes, seconds, milliseconds and "Have a nice day!" values to the X and Y␣
↪coordinates determined on the OLED screen.
oled.show()
if ldr.read_u16()>40000:
    wakeup= False

utime.sleep(1)
#wait for one second
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.9.5 Projenin Arduino C Kodu

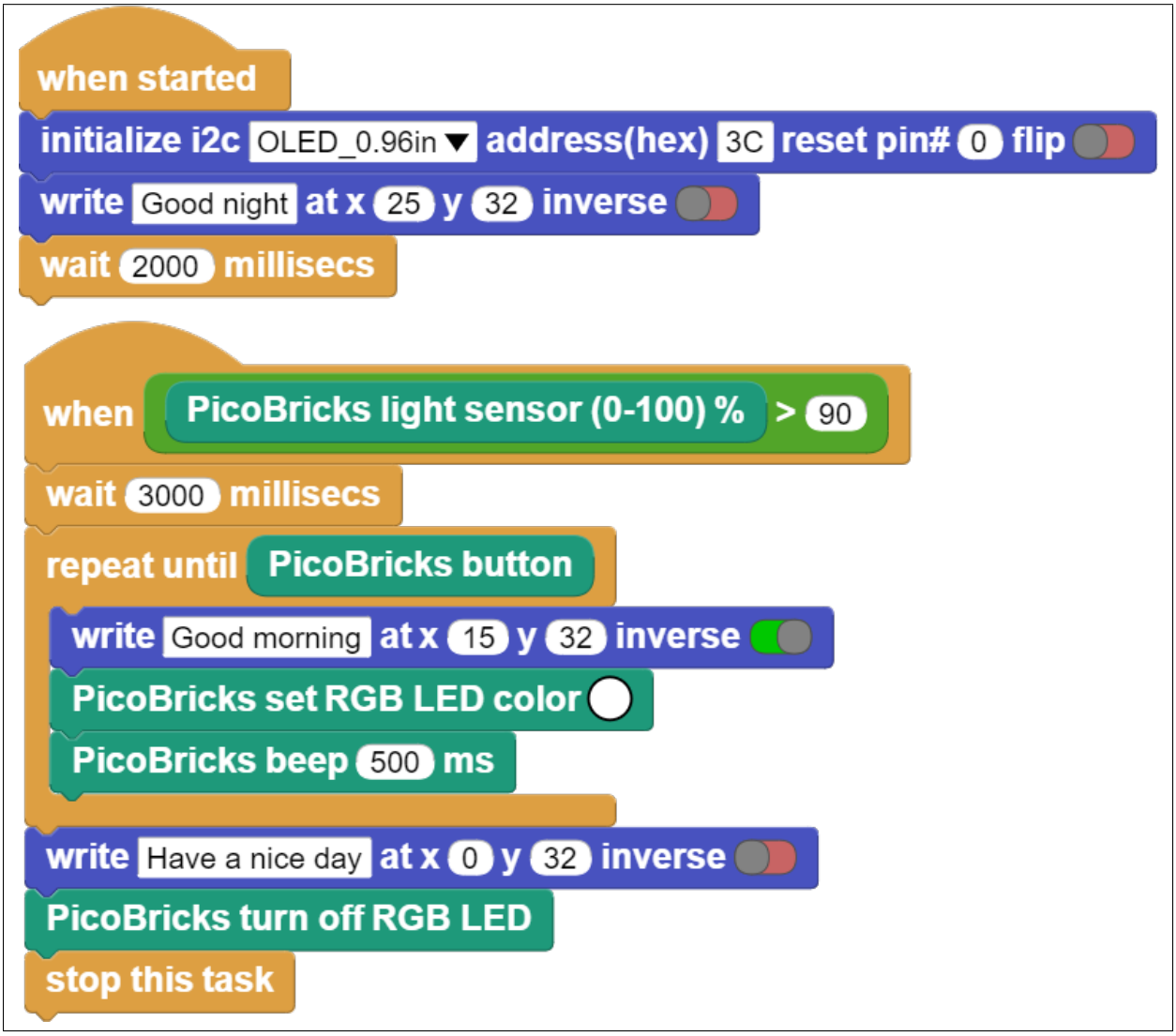
```
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define PIN        6

#define NUMPIXELS 1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"
int button;
void setup() {
// put your setup code here, to run once:
Wire.begin();
oled.init();
oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
clock_prescale_set(clock_div_1);
#endif
pinMode(10,INPUT);
pinMode(27,INPUT);
pinMode(20,OUTPUT);

pixels.begin();
pixels.setPixelColor(0, pixels.Color(0, 0, 0));
pixels.show();

    }

void loop() {
// put your main code here, to run repeatedly:
oled.setTextXY(4,3);
oled.putString("Good night");

if (analogRead(27)<200){

  while(!(button == 1)){

    button=digitalRead(10);

    oled.setTextXY(4,2);
    oled.putString("Good morning");
    pixels.setPixelColor(0, pixels.Color(255, 255, 255));
    pixels.show();
    tone(20,494);
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
  }
    oled.clearDisplay();
    oled.setTextXY(4,1);
    oled.putString("Have a nice day");
    noTone(20);
    pixels.setPixelColor(0, pixels.Color(0, 0, 0));
    pixels.show();
    delay(10000);
}


}
```

6.9.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.10 Know Your Color

6.10.1 Giriş

Bu projede her programlama dilinde kullanılan rastgelelik durumunu öğreneceksin. Picobricks'in RGB LED, OLED ekran ve buton modülü ile eğlenceli bir oyun hazırlayacağız. Projede inşa edeceğimiz oyun kullanıcının renkleri doğru veya yanlış bilmesi üzerine kurgulanacaktır.

6.10.2 Proje Detayları ve Algoritması

Elektronik sistemler üzerinde LED'ler sıklıkla kullanılır. Her butonun her seçeneğin yanında küçük LED'ler bulunabilmektedir. Tek bir LED'i değişik renklerde yanmasını sağlayarak birden fazla LED'in yaptığı işi tek bir LED ile yapılması sağlanabilmektedir. Bu türde çalışan LED'lere RGB LED denir. Adını Red, Green, Blue renk isimlerinin baş harflerinden alır. Bu LED'in diğer avantajı da 3 ana rengin karışımlarında da yanabilmesidir. Mor, turkuaz, turuncu...

PicoBricks üzerindeki RGB LED'de kırmızı, yeşil, mavi ve beyaz renklerden birisi rastgele olarak yanacak, aynı anda OLED ekranda yine bu dört renkten birisinin adı rastgele olarak yazılacaktır. Kullanıcı cevap hakkını kullanmak için 1,5 saniye içerisinde Picobricks'in butonuna basmalıdır. Oyun 10 kere tekrarlanacak, her tekrarda renkler eşlediğinde kullanıcı butona basarsa ya da eşleşmediğinde kullanıcı butona basmazsa 10 puan alacaktır. Renkler eşleşmediği halde kullanıcı butona basarsa 10 puan kaybedecektir. On tekrar sonunda kullanıcının aldığı puan OLED ekranda gösterilecektir. Kullanıcı dilerse butona basmayarak cevap hakkını kullanmayabilir.

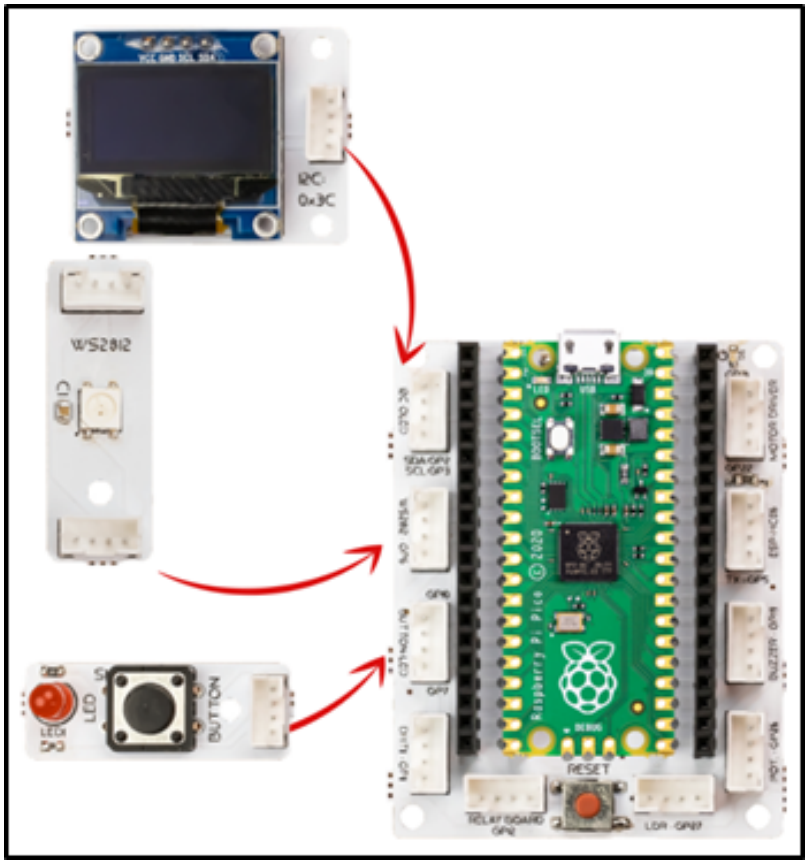
6.10.3 Bağlantı Diyagramı

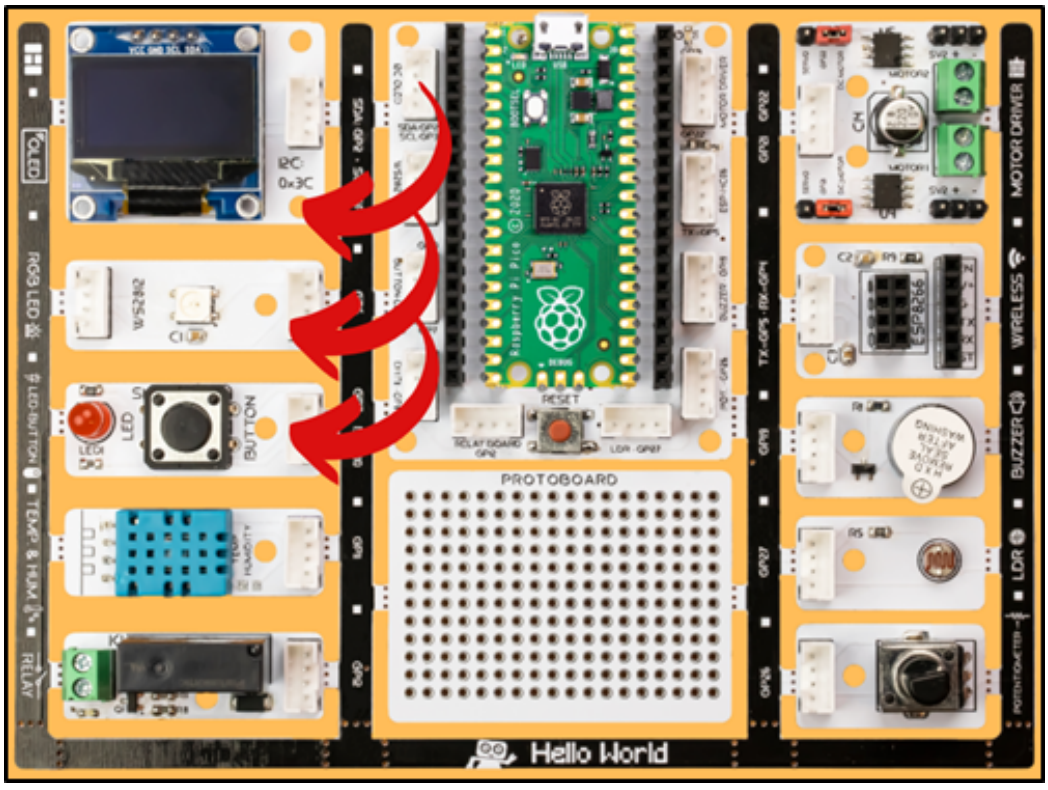
Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.10.4 Projenin MicroPython Kodu

```
from machine import Pin, I2C
from picobricks import SSD1306_I2C
import utime
import urandom
import _thread
from picobricks import WS2812

WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
ws = WS2812(pin_num=6, num_leds=1, brightness=0.3)

oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)
```

(sonraki sayfaya devam)


WS2812
LED
BUTTON
RESET


PROTOBOARD
Hello World
MOTOR DRIVER
WIRELESS
BUZZER
LDR
RELAY

(önceki sayfadan devam)

```
button = Pin(10,Pin.IN,Pin.PULL_DOWN)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 0, 255)
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)

oled.fill(0)
oled.show()

ws.pixels_fill(BLACK)
ws.pixels_show()

global button_pressed
score=0
button_pressed = False

def random_rgb():
global ledcolor
ledcolor=int(urandom.uniform(1,4))
if ledcolor == 1:
    ws.pixels_fill(RED)
    ws.pixels_show()
elif ledcolor == 2:
    ws.pixels_fill(GREEN)
    ws.pixels_show()
elif ledcolor == 3:
    ws.pixels_fill(BLUE)
    ws.pixels_show()
elif ledcolor == 4:
    ws.pixels_fill(WHİTE)
    ws.pixels_show()

def random_text():
global oledtext
oledtext=int(urandom.uniform(1,4))
if oledtext == 1:
    oled.fill(0)
    oled.show()
    oled.text("RED",45,32)
    oled.show()
elif oledtext == 2:
    oled.fill(0)
    oled.show()
    oled.text("GREEN",45,32)
    oled.show()
elif oledtext == 3:
    oled.fill(0)
    oled.show()
    oled.text("BLUE",45,32)
    oled.show()
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
elif oledtext == 4:
    oled.fill(0)
    oled.show()
    oled.text("WHITE",45,32)
    oled.show()

def button_reader_thread():
while True:
    global button_pressed
    if button_pressed == False:
        if button.value() == 1:
            button_pressed = True
            global score
            global oledtext
            global ledcolor
            if ledcolor == oledtext:
                score += 10
            else:
                score -= 10
    utime.sleep(0.01)

_thread.start_new_thread(button_reader_thread, ())

oled.text("The Game Begins",0,10)
oled.show()
utime.sleep(2)

for i in range(10):
random_text()
random_rgb()
button_pressed=False
utime.sleep(1.5)
oled.fill(0)
oled.show()
ws.pixels_fill(BLACK)
ws.pixels_show()
utime.sleep(1.5)
oled.fill(0)
oled.show()
oled.text("Your total score:",0,20)
oled.text(str(score), 30,40)
oled.show()
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.10.5 Projenin Arduino C Kodu

```
#include <Adafruit_NeoPixel.h>
#define PIN        6
#define NUMPIXELS 1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
#define DELAYVAL 500
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h" //define libraries
int OLED_color;
int RGB_color;
int score = 0;
int button = 0;



void setup() {
// put your setup code here, to run once:
Wire.begin();
oled.init();
oled.clearDisplay();


pixels.begin();
pixels.clear();
randomSeed(analogRead(27));

    }

void loop() {
// put your main code here, to run repeatedly:
oled.clearDisplay();
oled.setTextXY(3,1);
oled.putString("The game begins");
pixels.setPixelColor(0, pixels.Color(0, 0, 0));
pixels.show();
delay(2000);
oled.clearDisplay();

for (int i=0;i<10;i++){
button = digitalRead(10);
random_color();
pixels.show();
unsigned long start_time = millis();
while (button == 0) {
    button = digitalRead(10);
    if (millis() - start_time > 2000)
      break;
}
if (button == 1){

    if(OLED_color==RGB_color){
      score=score+10;
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

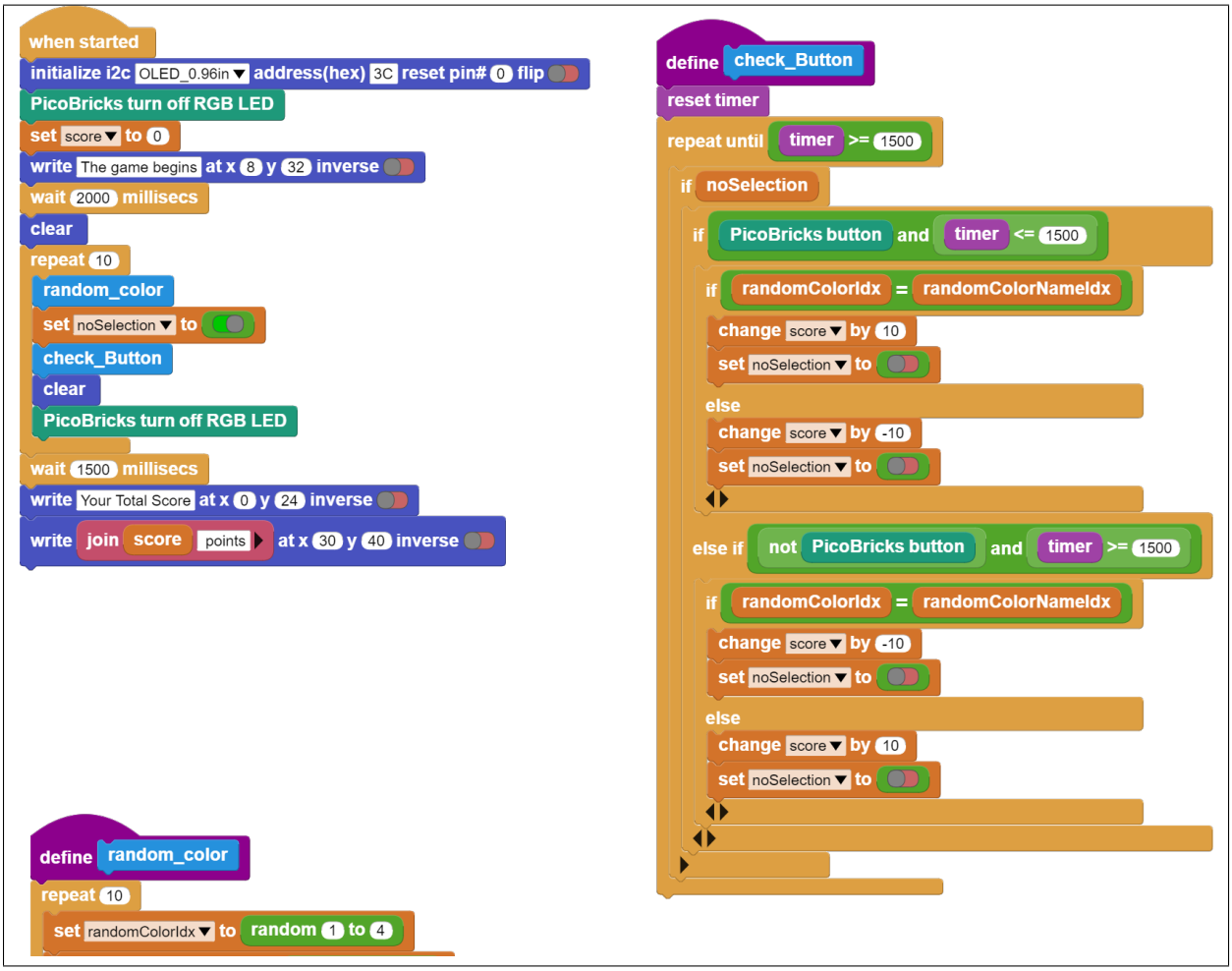
```
    }
    if(OLED_color!=RGB_color){
      score=score-10;
    }
    delay(200);
}
oled.clearDisplay();
pixels.setPixelColor(0, pixels.Color(0, 0, 0));
pixels.show();
    }

String string_scrore=String(score);
oled.clearDisplay();
oled.setTextXY(2,5);
oled.putString("Score: ");
oled.setTextXY(4,7);
oled.putString(string_scrore);
oled.setTextXY(6,5);
oled.putString("points");
// print final score on OLED screen

delay(10000);
    }

void random_color(){

OLED_color = random(1,5);
RGB_color = random(1,5);
// generate numbers between 1 and 5 randomly and print them on the screen
if (OLED_color == 1){
  oled.setTextXY(3,7);
  oled.putString("red");
    }
if (OLED_color == 2){
  oled.setTextXY(3,6);
  oled.putString("green");
    }
if (OLED_color == 3){
  oled.setTextXY(3,6);
  oled.putString("blue");
    }
if (OLED_color == 4){
  oled.setTextXY(3,6);
  oled.putString("white");
    }
if (RGB_color == 1){
  pixels.setPixelColor(0, pixels.Color(255, 0, 0));
    }
if (RGB_color == 2){
  pixels.setPixelColor(0, pixels.Color(0, 255, 0));
    }
if (RGB_color == 3){
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
  pixels.setPixelColor(0, pixels.Color(0, 0, 255));
    }
if (RGB_color == 4){
  pixels.setPixelColor(0, pixels.Color(255, 255, 255));
}


}
```

6.10.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.10.7 Projenin Videosu

6.11 Magic Lamp

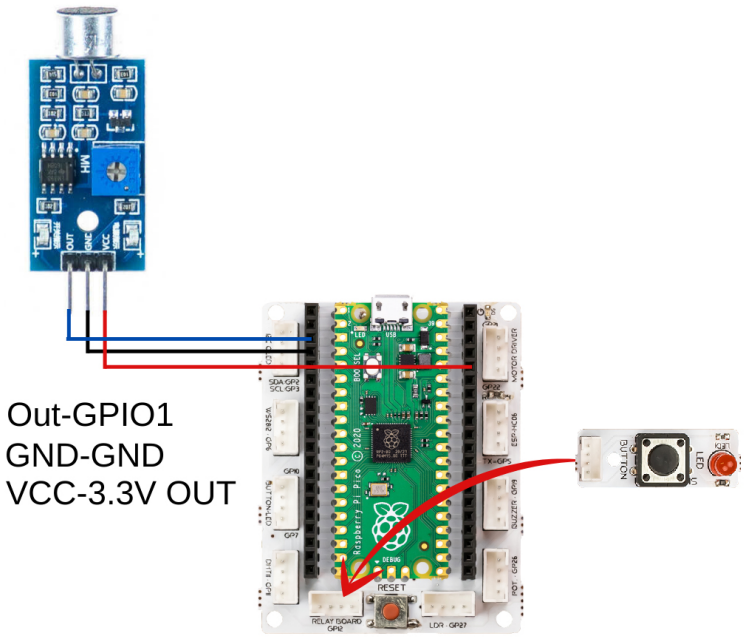
6.11.1 Giriş

Bu projede PicoBricks kartı üzerindeki LED modülünü sesle açıp kapatacağız. PicoBricks ses seviyesi sensörü kullanarak inşa edeceğimiz projemizde alkış sesi çıkararak açma-kapama işlemlerini gerçekleştireceğiz. Önceki projelerde olduğu gibi sensörlerin kullanıldığı projelerde kodları yazmaya başlamadan önce sadece sensörü çalıştırarak yapmak istediğimiz işlemlerde sensörün hangi değerleri gönderdiğini görmek daha sonra bu değerleri baz alarak projenin kodlarını yazmak ilerlemenizi kolaylaştıracaktır.

6.11.2 Proje Detayları ve Algoritması

Çoğumuz, filmlerde alkış sesiyle sihirliymişçesine yanıp sönen lambaları veya açılıp kapanan kapıları görmüşüzdür. Çekimlerde bu kapıları kapatan , lambaları söndüren set yardımcıları bulunmaktadır. Peki ya biz bunu otomatik olarak gerçekleştirsek nasıl olur? Ortamda meydana gelmesini beklediğimiz ses şiddeti değişikliğini elektrik sinyaline dönüştüren sensörler vardır. Bunlara ses sensörü denmektedir. Bu projede alkış ile yanıp sönebilen bir aydınlatma lambası düzeneği hazırlarken PicoBricks buton modülü ve ses seviyesi sensörünü kullanmayı kontrolünü öğreneceksin.

6.11.3 Bağlantı Diyagramı

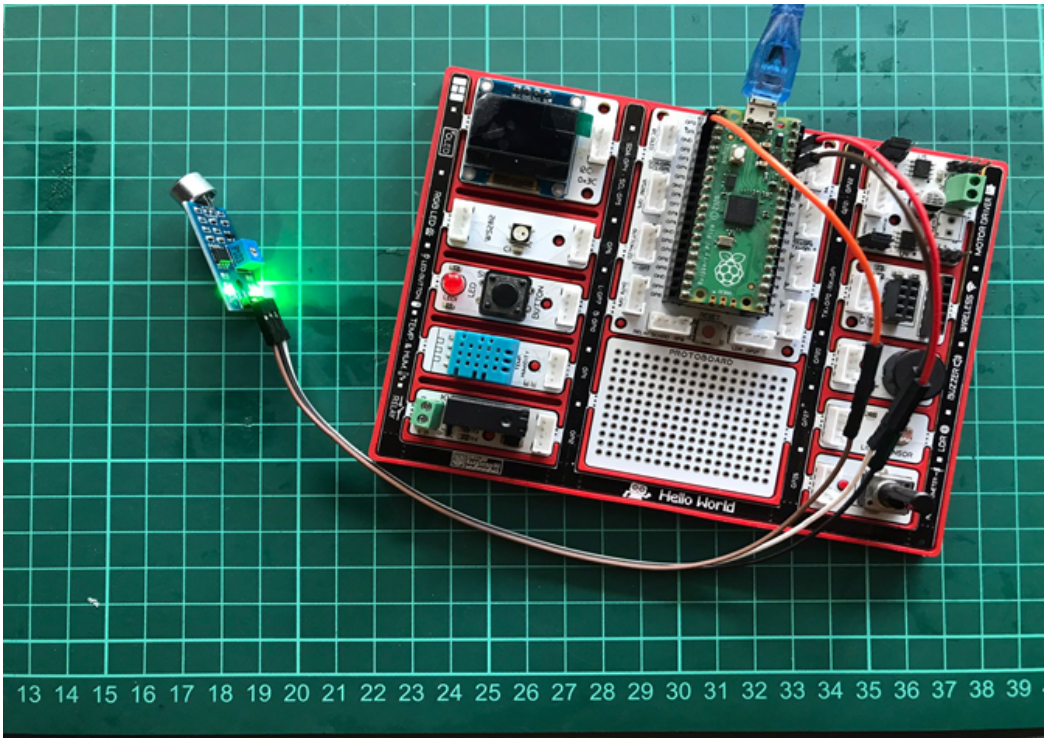


PicoBricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.11.4 Projenin MicroPython Kodu

```
from machine import Pin #to access the hardware on the pico
sensor=Pin(1,Pin.IN) #initialize digital pin 1 as an INPUT for Sensor
led=Pin(7,Pin.OUT)#initialize digital pin 7 as an OUTPUT for LED
while True:
#When sensor value is '0', the relay will be '1'
print(sensor.value())
if sensor.value()==1:
    led.value(1)
else:
    led.value(0)
```

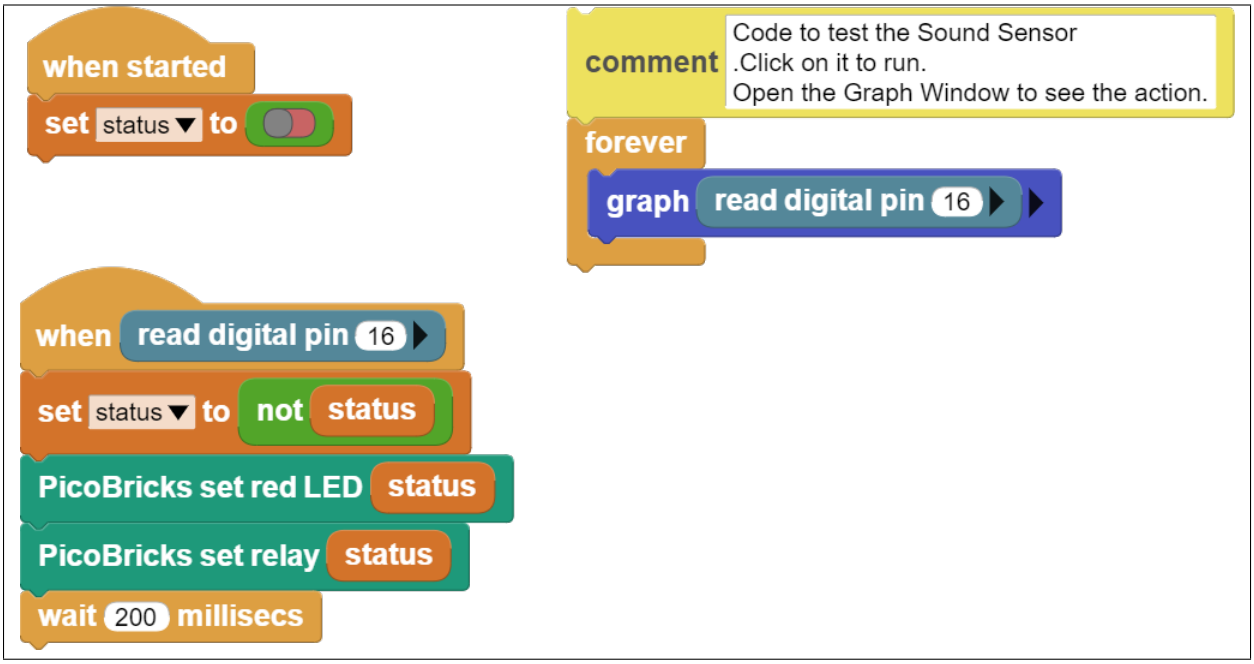
Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.



6.11.5 Projenin Arduino C Kodu

```
void setup() {
// put your setup code here, to run once:
pinMode(1,INPUT);
pinMode(7,OUTPUT);
//define the input and output pins
    }

void loop() {
// put your main code here, to run repeatedly:


Serial.println(digitalRead(1));

if(digitalRead(1)==1){
digitalWrite(7,HIGH);
delay(3000);
    }
else{
digitalWrite(7,LOW);
delay(1000);
}
}
```

6.11.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.12 Smart Cooler

6.12.1 Giriş

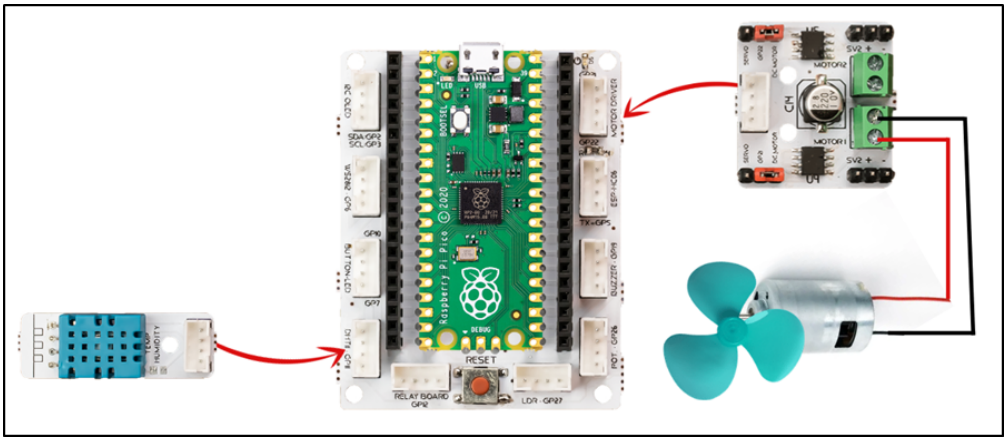
Projemizde öncelikle Picobricks üzerindeki DHT11 sıcaklık ve nem sensörünün ölçtüğü sıcaklık değerlerini görüntüleyeceğiz. Daha sonra sonra bir sıcaklık sınırı belirleyerek DHT11 modülünden gelen sıcaklık değeri bu sınıra ulaştığında Picobricks'e bağlı DC motorun dönmeye başlaması, sıcaklık değeri belirlediğimiz sınırın altına indiğinde ise DC motorun durması için gerekli kodları yazacağız.

6.12.2 Projenin Detayları ve Algoritması

Yaz aylarında serinlemek için kış aylarında ısınmak için klimalar kullanılır. Klimalar ısıtma ve soğutma derecesini bulunduğu ortamın sıcaklığına göre ayarlamaktadır. Fırınlar yemeği pişirirken kullanıcının ayarladığı sıcaklık değerine çıkmaya ve o sıcaklığı korumaya çalışırlar. Bu iki elektronik cihazda sıcaklığı kontrol etmek için özel sıcaklık sensörleri kullanmaktadır. Ayrıca seralarda sıcaklık ve nem birlikte ölçülür. Bu iki değer istenen düzeyde dengede kalabilmesi için fan ile hava akımını sağlanmaya çalışılır.

PicoBricks'te sıcaklığı ve nemi ayrı ayrı ölçebilir ve bu ölçümler ile çevreyle etkileşime girebilirsiniz. Bu projede PicoBricks ile sıcaklığa göre fan hızını otomatik ayarlayan bir serinletme sistemi hazırlayacağız. Böylelikle DC motor çalışma sistemini ve motor hız ayarı yapmayı öğreneceksin.

6.12.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.12.4 Projenin MicroPython Kodu

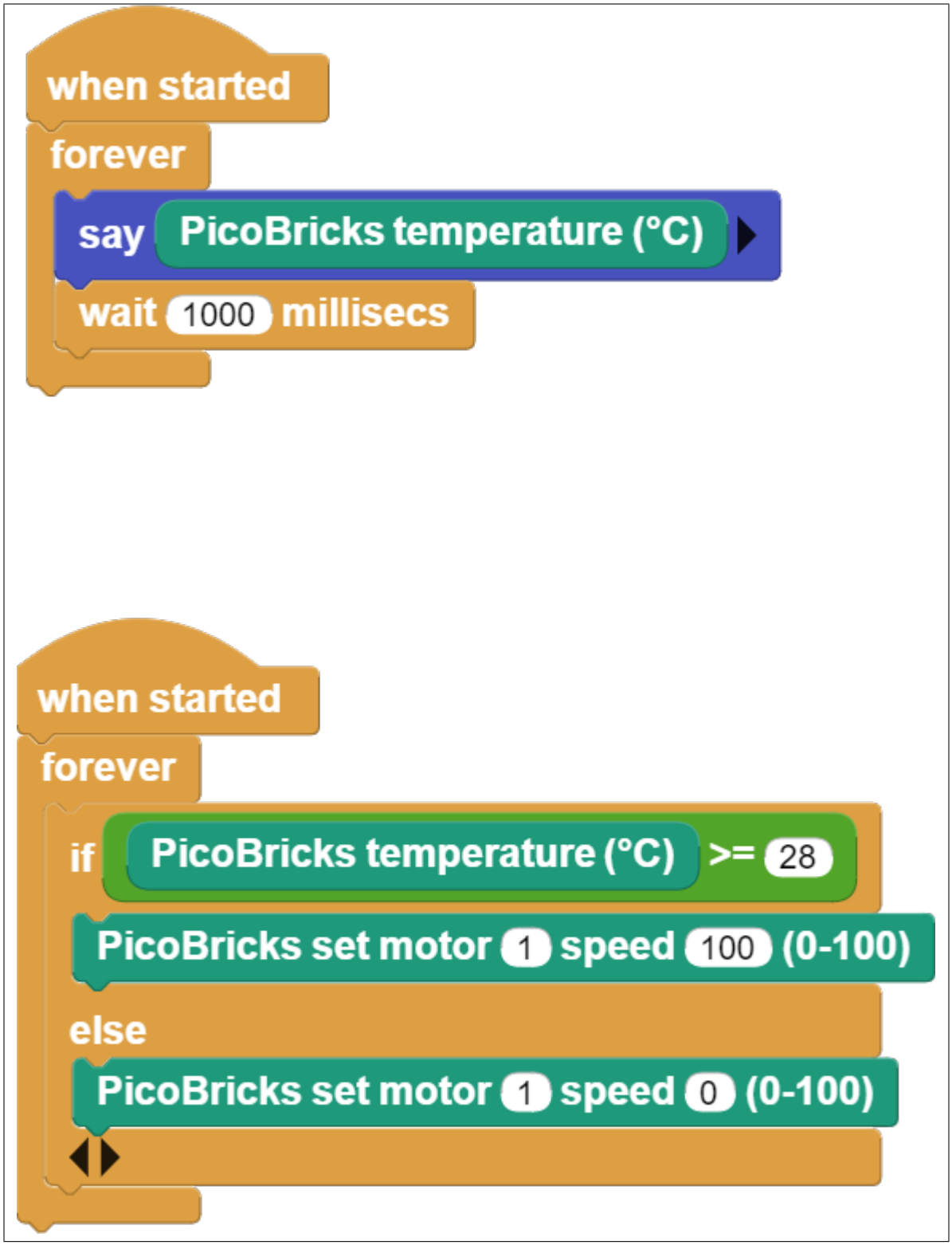
```
from machine import Pin
from picobricks import DHT11
import utime

LIMIT_TEMPERATURE = 20 #define the limit temperature

dht_sensor = DHT11(Pin(11, Pin.IN, Pin.PULL_DOWN))
m1 = Pin(21, Pin.OUT)
m1.low()
dht_read_time = utime.time()
#define input-output pins

while True:
if utime.time() - dht_read_time >= 3:
    dht_read_time = utime.time()
    dht_sensor.measure()
    temp= dht_sensor.temperature
    print(temp)
    if temp >= LIMIT_TEMPERATURE:
        m1.high()
        #operate if the room temperature is higher than the limit temperature
    else:
        m1.low()
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.12.5 Projenin Arduino C Kodu

```
#include <DHT.h>
#define LIMIT_TEMPERATURE     27
#define DHTPIN 11
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);
float temperature;

void setup() {
// put your setup code here, to run once:
Serial.begin(115200);
dht.begin();
pinMode(21,OUTPUT);

    }

void loop() {
// put your main code here, to run repeatedly:
delay(100);
temperature = dht.readTemperature();
Serial.print("Temp: ");
Serial.println(temperature);
if(temperature > LIMIT_TEMPERATURE){
digitalWrite(21,HIGH);
}
else{
digitalWrite(21,LOW);
    }


}
```

6.12.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.13 Buzz Wire Game

6.13.1 Giriş

Bu projede Picobricks ile buzzer ve LED modül kullanarak dikkat ve konsantrasyon geliştirici Buzz Wire Game'i iletken bir tel yardımıyla elektronik olarak hazırlayacağız. Bu projeyi hazırlarken bir buton değil, `button` gibi kullanılacak bir giriş tekniği öğrenmiş olacaksınız.

6.13.2 Proje Detayları ve Algoritması

Projeler her zaman sorunları çözmek ve işleri kolaylaştırmakla ilgili olmak zorunda değildir. Siz de eğlenmek ve kendinizi geliştirmek için projeler hazırlayabilirsiniz. Dikkat ve konsantrasyon, birçok insanın geliştirmek istediği özelliklerdir. Bununla yapabileceğimiz uygulamalar oldukça ilgi çekici. Picobricks ile Buzz Wire Oyunu yapmaya ne dersiniz? Bilgisayarların 0'lar ve 1'lerle çalıştığı tabirini duymuşsunuzdur. 0 elektriğin yokluğunu, 1 ise varlığını temsil eder. 0 ve 1'ler belirli sayıda ve dizilişle bir araya gelerek anlamlı veriler oluşturur. Elektronik sistemlerde, bir durumu doğrudan kontrol etmek için 0'lar ve 1'ler kullanılabilir. Kapı kapalı mı değil mi? Işık açık mı kapalı mı? Sulama sistemi açık mı, değil mi? Bu tür bilgileri elde etmek için bir durum kontrolü yapılır. Bu projemizde Picobricks ile buzzer ve LED modül kullanarak dikkat ve konsantrasyon geliştirici Buzz Wire Game'i iletken bir tel yardımıyla elektronik olarak hazırlayacağız. Bu projeyi hazırlarken buton olmayan, buton gibi kullanılacak bir giriş tekniği öğrenmiş olacaksınız.

Projeyi hazırlamak için 2 adet erkek-erkek jumper kablo ve 15 cm uzunluğunda iletken bükülebilir tele ihtiyacın var. Oyuncu hazır olduğunda oyunu başlatmak için butona basması istenecek. Butona basıldığında oyuncunun elinde jumper kablo iletken tele değerse Picobricks bunu algılayıp sesli ve yazılı uyarı verecek. Oyun başladıktan bitene kadar geçen sürede OLED ekranda gösterilecek. Kullanıcı butona bastıktan sonra timer'ı resetliyoruz. Daha sonra Picobricks'in GPIO1 nolu pinine bağlı iletken tel'e 3.3V'luk gerilim vereceğiz. Oyuncunun elinde tuttuğu kablonun bir ucu Picobricks üzerinde GND pinine bağlı olacak. Eğer oyuncu elindeki jumper kabloyu iletken tele dokundurursa GPIO1 nolu pin Pasif/Kapalı/0 konumuna düşecektir. Daha sonra Oyunun bittiğini bildirir ışıklı, yazılı ve sesli geri bildirimde bulunulacak ardından OLED ekran üzerinde geçen süre milisaniye cinsinden gösterilecek. 5 saniye sonunda oyuncunun yeniden başlamak için butona basması istenecek.

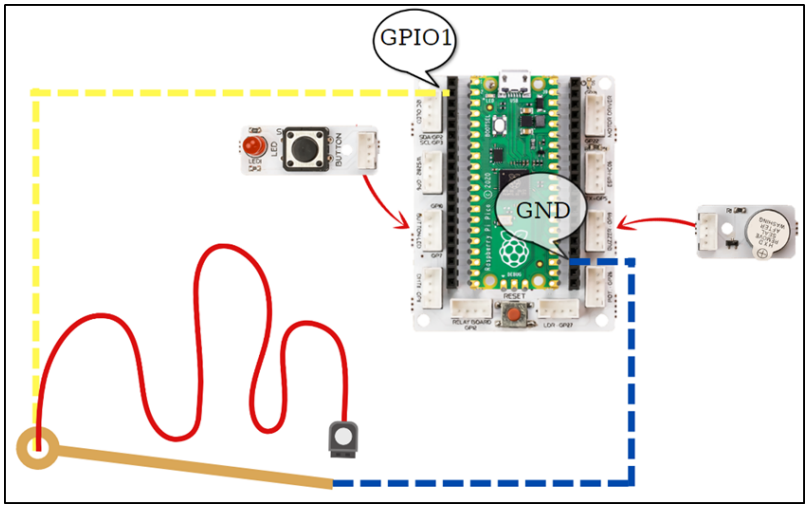
6.13.3 Bağlantı Diyagramı

Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.13.4 Projenin MicroPthon Kodu

```
from machine import Pin, I2C, Timer #to access the hardware on the pico
from picobricks import SSD1306_I2C #OLED Screen Library
from utime import sleep # time library

#OLED Screen Settings
WIDTH  = 128
HEIGHT = 64
```

(sonraki sayfaya devam)



(önceki sayfadan devam)

```
sda=machine.Pin(4)#initialize digital pin 4 and 5 as an OUTPUT for OLED Communication
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)

wire=Pin(1,Pin.OUT)#initialize digital pin 1 as an OUTPUT
led = Pin(7,Pin.OUT)#initialize digital pin 7 and 5 as an OUTPUT for LED
buzzer=Pin(20, Pin.OUT)#initialize digital pin 20 as an OUTPUT for Buzzer
button=Pin(10,Pin.IN,Pin.PULL_DOWN)#initialize digital pin 10 as an INPUT for button
endtime=0


while True:
led.low()
oled.fill(0)
oled.show()
oled.text("<BUZZ WIRE GAME>",0,0)
oled.text("Press the button",0,17)
oled.text("TO START!",25,35)
oled.show()
#When button is '0', OLED says 'GAME STARTED'
while button.value()==0:
    print("press the button")
oled.fill(0)
oled.show()
oled.text("GAME",25,35)
oled.text("STARTED",25,45)
oled.show()
wire.high()
timer_start=utime.ticks_ms()
 #When wire is '1', OLED says 'GAME OVER'
while wire.value()==1:
    print("Started")
endtime=utime.ticks_diff(utime.ticks_ms(), timer_start)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
print(endtime)
oled.fill(0)
oled.show()
oled.text("GAME OVER!",25,35)
oled.text(endtime + "ms" ,25,45)
oled.show()
led.high()#LED On
buzzer.high()#Buzzer On
sleep(5)#Delay
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.13.5 Projenin Arduino C Kodu

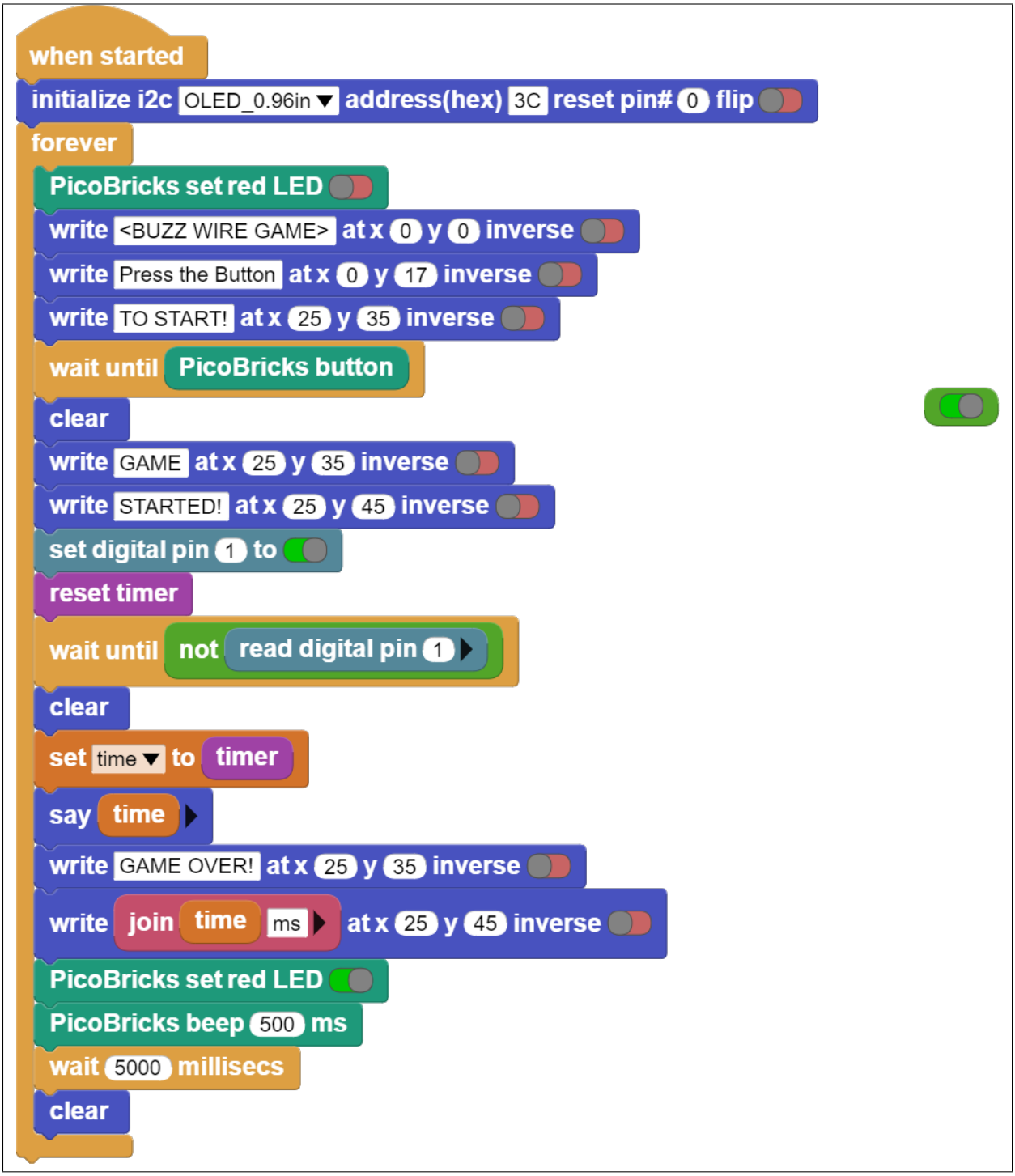
```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int Time=0;
unsigned long Old_Time=0;

void setup() {
// put your setup code here, to run once:
pinMode(20,OUTPUT);
pinMode(7,OUTPUT);
pinMode(1,OUTPUT);
pinMode(10,INPUT);

Wire.begin();
oled.init();
oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
clock_prescale_set(clock_div_1);
#endif


    }

void loop() {
// put your main code here, to run repeatedly:
digitalWrite(7,LOW);

oled.setTextXY(2,1);
oled.putString("BUZZ WIRE GAME");
oled.setTextXY(4,2);
oled.putString("Press Button");
oled.setTextXY(5,3);
oled.putString("TO START!");
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
while (!(digitalRead(10)==1)){

    }

oled.clearDisplay();
oled.setTextXY(3,6);
oled.putString("GAME");
oled.setTextXY(5,4);
oled.putString("STARTED");

digitalWrite(1,HIGH);
Old_Time=millis();

while(!(digitalRead(1)==0)){

Time=millis()-Old_Time;
    }

String(String_Time)=String(Time);

oled.clearDisplay();
oled.setTextXY(3,4);
oled.putString("GAME OVER");
oled.setTextXY(5,4);
oled.putString(String_Time);
oled.setTextXY(5,10);
oled.putString("ms");

digitalWrite(7,HIGH);
digitalWrite(20,HIGH);
delay(500);
digitalWrite(20,LOW);
delay(5000);

Time=0;
Old_Time=0;
oled.clearDisplay();


    }
```

6.13.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.14 Dinosaur Game

6.14.1 Giriş

Bu projede PicoBricks ile Servo motorların nasıl kontrol edilebildiğini öğreneceksin. Proje kodlarını yazarken öncelikle LDR sensörü bilgisayar ekranına sabitleyerek beyaz ve siyah zemindeki sensör verilerine okuyacak daha sonra bu verilere göre servo motorun hareket etmesi için gerekli kodları yazacağız.

6.14.2 Projenin Detayları ve Algoritması

Geliştirilecek elektronik sistemler görevini itme, çekme, döndürme, kaldırma, indirme gibi hareketle sonuçlanan işlerle yerine getireceklerse projede aktuatör olarak pnomatik sistemler ya da elektrik motorlu sistemler kullanılır. Picobricks, projelerinde yazdığın kodları harekete geçirebilecek sistemler üretebilmen için iki farklı motor tipini desteklemektedir. DC motor ve DC motorların elektronik olarak hareketlerinin düzenlendiği Servo motorlar. Servo motorlar dönüş açısı değeri verildiğinde o açıya dönen motorlardır. Model uçakların kanatlarında, uçağa yön verebilmek için kanat uçlarındaki yaprakları servo motorlar ile hareket ettirilmektedir. RC teknelerde aracın yönünü değiştirmek için de aynı mantıkla servo motorlar kullanılmaktadır. Ayrıca tam tur dönebilen, akıllı sürekli servo olarak bilinen gelişmiş servo motorlar, evlerimizde kullandığımız akıllı süpürgelerin tekerleklerinde de kullanılmaktadır.

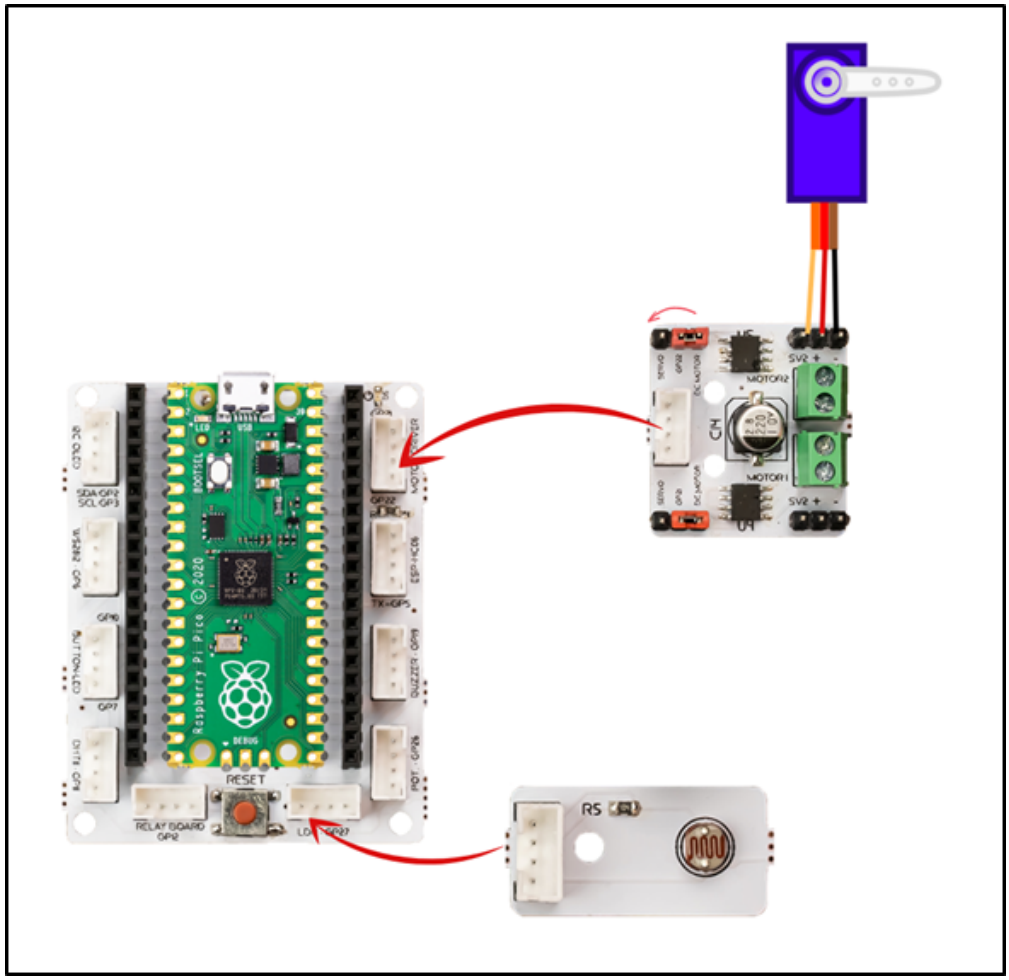
Bu projede Google Chrome offline dinasour game'i otomatik olarak Picobricks'e oynatacağız. Oyunda Picobricks engelleri algılayarak otomatik olarak dinazorun hareketlerini kontrol edecek. Oyun esnasında dinazorun karşısına çıkan engelleri algılamak için picobricks LDR sensör kullanacağız. LDR sensör yüzeyine temas eden ışık miktarını ölçerek analog sinyaller gönderebilmektedir. Sensörü bilgisayar ekranına sabitleyerek beyaz ve siyah renkler arasındaki ışık miktarı farkından yararlanarak dinazorun önüne engel gelip gelmediğini algılayabiliriz. Engel algılandığında ise servo motor kullanarak klavyedeki boşluk tuşuna otomatik olarak basılmasını sağlayabiliriz. Bu sayede dinazor engelleri kolaylıkla aşacaktır.

6.14.3 Bağlantı Diyagramı

Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.14.4 Projenin MicroPython Kodu

```
from machine import Pin, ADC, PWM#to access the hardware on the pico
from utime import sleep #time library

ldr=ADC(27) #initialize digital pin 27 for LDR
servo=PWM(Pin(21)) #initialize digital PWM pin 27 for Servo Motor
servo.freq(50)

while True:
sleep(0.01)
#When LDR data higher than 40000
if ldr.read_u16()>4000:
    servo.duty_u16(2000)# sets position to 180 degrees
    sleep(0.1)#delay
    servo.duty_u16(1350) # sets position to 0 degrees
    sleep(0.5)#delay
```

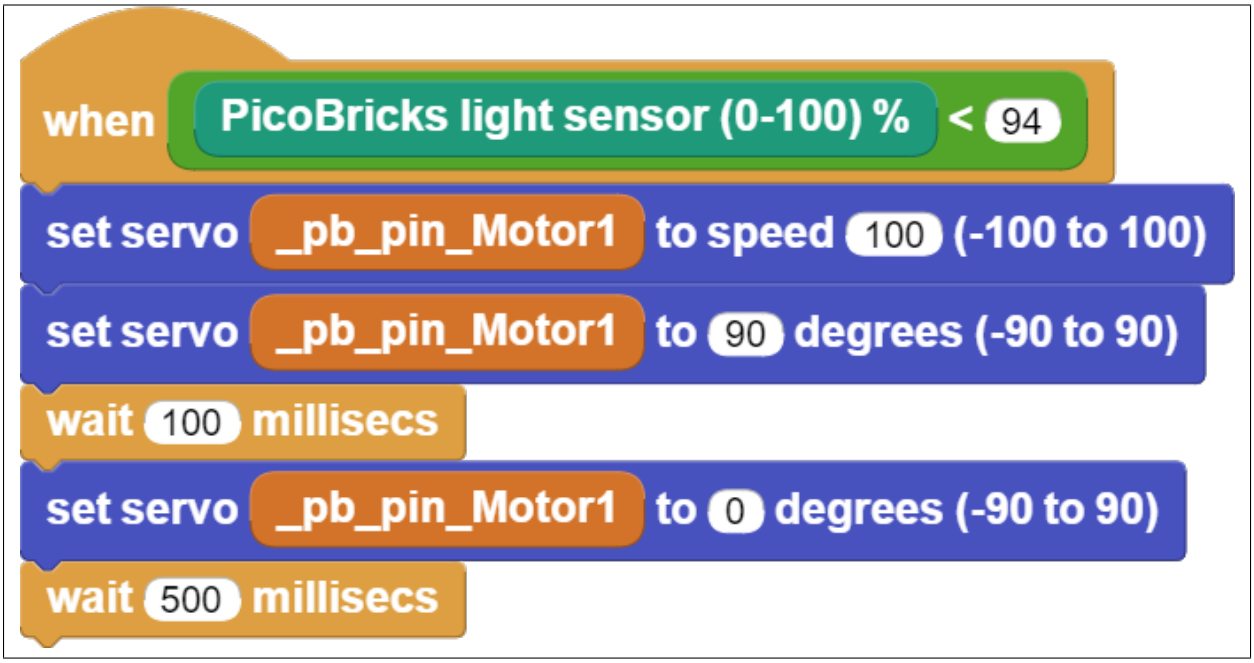

RESET
RS

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.14.5 Projenin Arduino C Kodu

```
#include <Servo.h>
Servo myservo;

void setup() {
// put your setup code here, to run once:
myservo.attach(22);
myservo.write(20);
pinMode(27,INPUT);



    }

void loop() {
// put your main code here, to run repeatedly:
int light_sensor=analogRead(27);

if(light_sensor>100){

int x=45;
int y=20;

myservo.write(x);
delay(100);
myservo.write(y);
delay(500);
    }


}
```

6.14.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.15 Night and Day

6.15.1 Giriş

Bu oyun dikkatini ve refleksini kullanacağın bir oyun olacak.

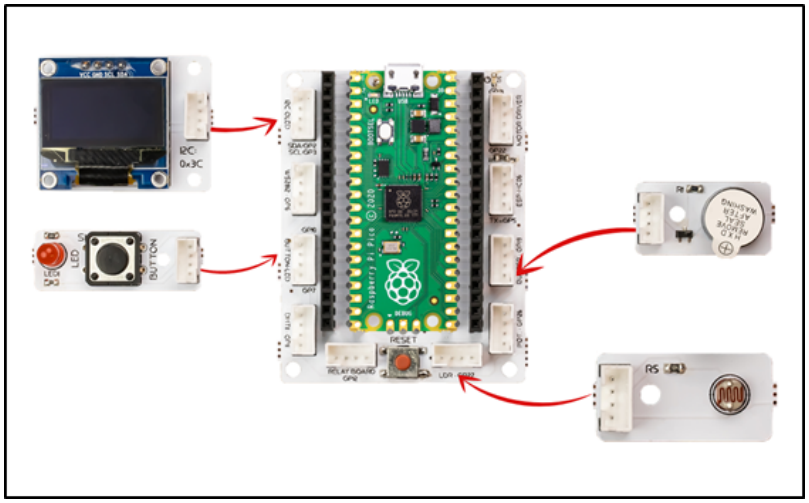
6.15.2 Projenin Detayları ve Algoritması

Okulda oynadığınız Gece Gündüz oyununu elektronik olarak oynamaya ne dersin? Öğretmen gece dediğinde kafamızı öne eğip masanın üzerindeki kolumuza yasladığımız, gündüz dediğinde başımızı kaldırdığımız bir oyundur gece-gündüz oyunu. Bu projede 0,96" 128x64 piksel I2C OLED ekranı kullanacağız. OLED ekranlar yapay ışık kaynağı olarak kullanılabildikleri için ekran üzerindeki karakterleri mercek ve ayna kullanarak büyütebilir ve istediğiniz düzleme yansıtabilirsin. Akıllı gözlükler ve otomobil camlarına bilgilendirme, yol ve trafik bilgisi yansıtabilen sistemler OLED ekranlar kullanılarak yapılabilmektedir. Işık sensörleri bulundukları ortamın ışık seviyelerini ölçebilen, fotodiyot da denilen sensörlerdir. Işığa maruz kalan sensörün elektrik geçirgenliği değişmektedir. Biz de kodlayarak ışık sensörünü kontrol edip, ışık miktarının etkilediği elektronik sistemler geliştirebilmekteyiz.

Önce oyuncunun oyuna başlaması için butona basmasını isteyeceğiz. Ardından PicoBricks'in OLED ekranında NIGHT ve DAY ifadelerini rastgele olarak 2'şer saniye boyunca gösterilmesini sağlayacağız. Oyuncu, eğer OLED ekranda yazan kelime NIGHT ise 2 saniye içinde LDR sensörünün üzerini eliyle kapatmalı, eğer OLED ekranda GÜNDÜZ kelimesi yazıyorsa LDR sensörünün üzerinden elini kaldırmalı. Oyuncunun her doğru tepkisi 10 puan kazanmasını sağlayacak, yanlış tepkide ise oyun bitecek ve ekranda oyunun bitmesini bildiren yazılı ifade yer alacak, buzzerdan

farklı tonda bir ses çalacak ve OLED ekranda puan bilgisi yer alacaktır. Eğer oyuncu toplam 10 doğru tepki verip 100 puan elde ederse "Congratulation" yazısı ile oyuna yeniden başlayabilmek için RESET butonuna basılmasını bildiren ifade OLED ekranda gösterilip buzzerdan farklı tonda notalar çalacaktır.

6.15.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.15.4 Projenin MicroPython Kodu

```
from machine import Pin, I2C, Timer, ADC, PWM
from picobricks import SSD1306_I2C
import utime
import urandom
#define the libraries
WIDTH = 128
HEIGHT = 64
#OLED Screen Settings
sda=machine.Pin(4)
scl=machine.Pin(5)
#initialize digital pin 4 and 5 as an OUTPUT for OLED Communication
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)
buzzer = PWM(Pin(20))
buzzer.freq(440)
ldr=ADC(Pin(27))
button=Pin(10,Pin.IN,Pin.PULL_DOWN)
#define the input and output pins
oled.text("NIGHT and DAY", 10, 0)
oled.text("<GAME>", 40, 20)
oled.text("Press the Button", 0, 40)
oled.text("to START!", 40, 55)
oled.show()
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
#OLED Screen Texts Settings
def changeWord():
global nightorday
oled.fill(0)
oled.show()
nightorday=round(urandom.uniform(0,1))
#when data is '0', OLED texts NIGHT
if nightorday==0:
    oled.text("---NIGHT---", 20, 30)
    oled.show()
else:
    oled.text("---DAY---", 20, 30)
    oled.show()
#waits for the button to be pressed to activate

while button.value()==0:
print("Press the Button")
sleep(0.01)

oled.fill(0)
oled.show()
start=1
global score
score=0
while start==1:
global gamerReaction
global score
changeWord()
startTime=utime.ticks_ms()
#when LDR's data greater than 2000, gamer reaction '0'
while utime.ticks_diff(utime.ticks_ms(), startTime)<=2000:
    if ldr.read_u16()>20000:
        gamerReaction=0
    #when LDR's data lower than 2000, gamer reaction '1'
    else:
        gamerReaction=1
    sleep(0.01)
#buzzer working
buzzer.duty_u16(2000)
sleep(0.05)
buzzer.duty_u16(0)
if gamerReaction==nightorday:
    score += 10
#when score is 10, OLED says 'Game Over'
else:
    oled.fill(0)
    oled.show()
    oled.text("Game Over", 0, 18, 1)
    oled.text("Your score " + str(score), 0,35)
    oled.text("Press RESET",0, 45)
    oled.text("To REPEAT",0,55)
    oled.show()
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
    buzzer.duty_u16(2000)
    sleep(0.05)
    buzzer.duty_u16(0)
    break;
if score==100:
    #when score is 10, OLED says 'You Won'
    oled.fill(0)
    oled.show()
    oled.text("Congratulation", 10, 10)
    oled.text("Top Score: 100", 5, 35)
    oled.text("Press Reset", 20, 45)
    oled.text("To REPEAT", 25,55)
    oled.show()
    buzzer.duty_u16(2000)
    sleep(0.1)
    buzzer.duty_u16(0)
    sleep(0.1)
    buzzer.duty_u16(2000)
    sleep(0.1)
    buzzer.duty_u16(0)
    break;
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.15.5 Projenin Arduino C Kodu

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"
//define the library


#define RANDOM_SEED_PIN     28
int Gamer_Reaction=0;
int Night_or_Day=0;
int Score=0;
int counter=0;

double currentTime=0;
double lastTime=0;
double getLastTime(){
return currentTime=millis()/1000.0-lastTime;
    }

void _delay(float seconds){
long endTime=millis()+seconds*1000;
while (millis()<endTime) _loop();
    }

void _loop(){
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
}

void loop(){
_loop();
}
//define variable

void setup() {
// put your setup code here, to run once:
pinMode(10,INPUT);
pinMode(27, INPUT);
pinMode(20,OUTPUT);
randomSeed(RANDOM_SEED_PIN);
Wire.begin();
oled.init();
oled.clearDisplay();
//define the input and output pins

oled.clearDisplay();
oled.setTextXY(1,3);
oled.putString("NIGHT and DAY");
oled.setTextXY(2,7);
oled.putString("GAME");
oled.setTextXY(5,2);
oled.putString("Press the BUTTON");
oled.setTextXY(6,4);
oled.putString("to START!");
//write "NIGHT an DAY, GAME, Press the BUTTON, to START" on the x and y coordinates␣
↪determined on the OLED screen

Score=0;
//define the score variable

while(!(digitalRead(10)==1))  //until the button is pressed
    {
    _loop();
}
_delay(0.2);

while(1){  //while loop
if(counter==0){
  delay(500);
  Change_Word();
  lastTime=millis()/1000.0;
    }
while(!(getLastTime()>2)){
  Serial.println(analogRead(27));
  if(analogRead(27)>200){
    Gamer_Reaction=0;

  }
  else{
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

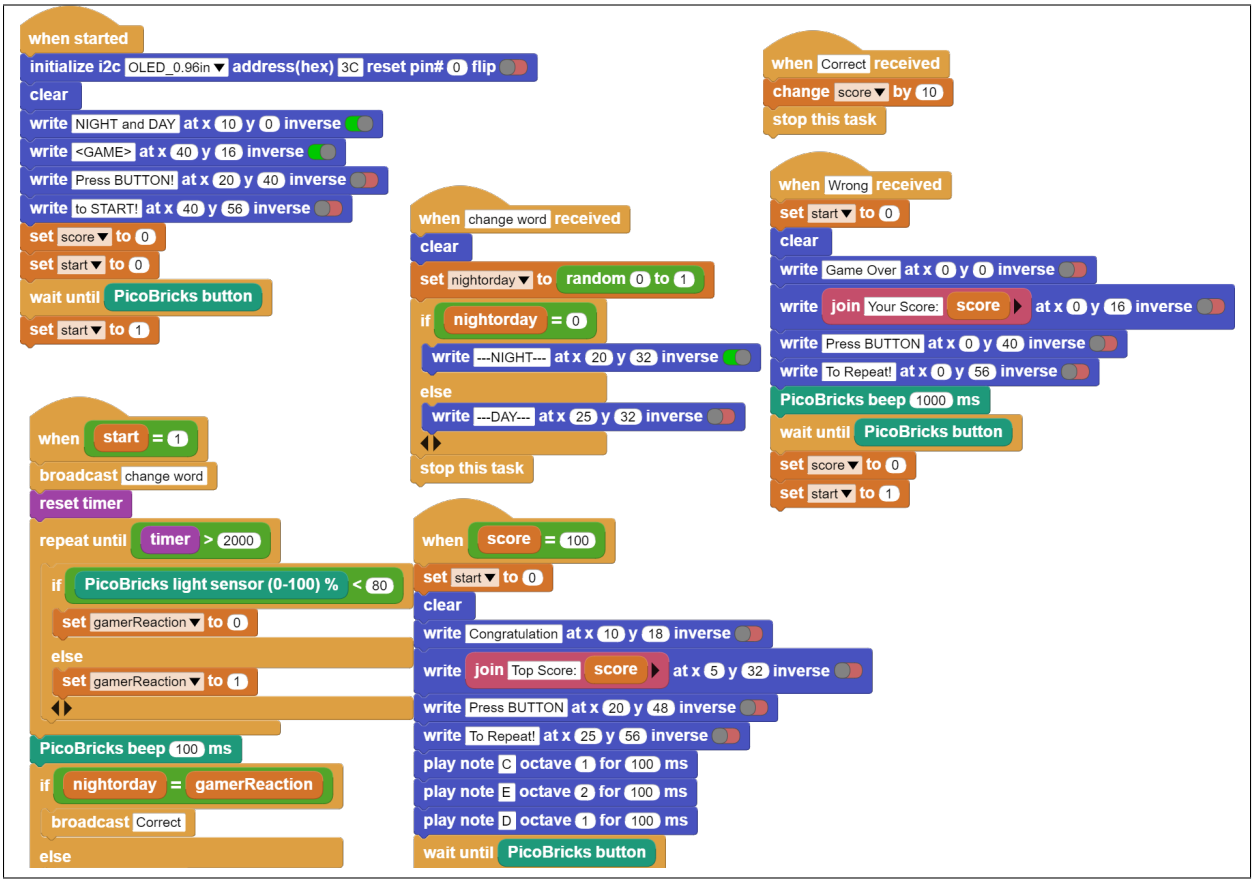
```
    Gamer_Reaction=1;
  }
}
//determine the gamer reaction based on the value of the LDR sensor
digitalWrite(20,HIGH);   //turn on the buzzer
delay(250);  //wait
digitalWrite(20,LOW);  //turn off the buzzer

if(Night_or_Day==Gamer_Reaction){  //if the user's reaction and the Night_or_Day␣
↪variable are the same
Correct();

}
else{
Wrong();
}
_loop();

if(Score==100){
  oled.clearDisplay();
  oled.setTextXY(1,1);
  oled.putString("Congratulation");
  oled.setTextXY(3,1);
  oled.putString("Your Score");
  oled.setTextXY(3,13);
  String String_Score=String(Score);
  oled.putString(String_Score);
  oled.setTextXY(5,3);
  oled.putString("Press Reset");
  oled.setTextXY(6,3);
  oled.putString("To Repeat!");
  //write the "Congratulation, Your Score, press Reset, To Repeat!" and score variable␣
↪on the x and y coordinates determined on the OLED screen
  for(int i=0;i<3;i++){
    digitalWrite(20,HIGH);
    delay(500);
    digitalWrite(20,LOW);
    delay(500);

}
//turn the buzzer on and off three times
counter=1;

    }
    }
}

void Correct(){
Score+=10;
oled.clearDisplay();
oled.setTextXY(3,4);
oled.putString("10 Points");
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
//increase the score by 10 when the gamer answers correctly
}

void Change_Word(){

oled.clearDisplay();
Night_or_Day=random(0,2);
if(Night_or_Day==0){
oled.setTextXY(3,6);
oled.putString("NIGHT");

}
else{
oled.setTextXY(3,7);
oled.putString("DAY");
}

}
//write "NIGHT" or "DAY" on random OLED screen

void Wrong(){
oled.clearDisplay();
oled.setTextXY(1,3);
oled.putString("Game Over");
oled.setTextXY(3,1);
oled.putString("Your Score");
oled.setTextXY(1,13);
String String_Score=String(Score);
oled.putString(String_Score);
oled.setTextXY(5,3);
oled.putString("Pres Reset");
oled.setTextXY(6,3);
oled.putString("To Repeat");
// write the score variable and the expressions is quotation marks to the coordinates␣
↪determined on the OLED screen.

digitalWrite(20,HIGH);  //turn on the buzzer
delay(1000);   //wait
digitalWrite(20,LOW); //turn off the buzzer
counter=1;
}
```

6.15.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.16 Voice Controlled Car

6.16.1 Giriş

Bu projede, setin içinden çıkan robot araba kiti montajlayıp cep telefonu üzerinden kontrol edeceğiz.

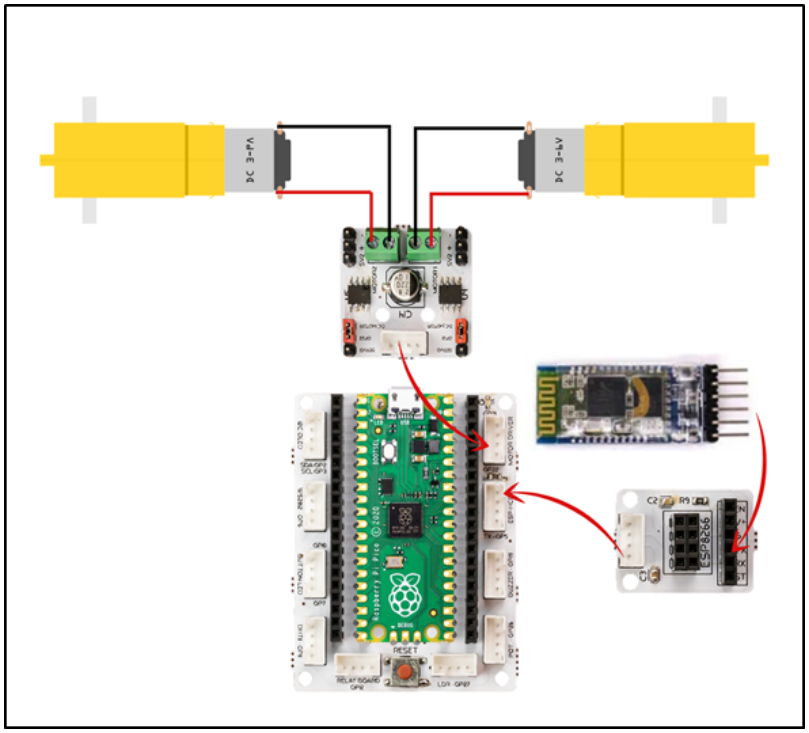
6.16.2 Proje Detayları ve Algoritması

Gelişen ve gelişmeye devam eden `yapay zeka` uygulamaları insan özelliklerini tanımakta, öğrenmekte ve onun gibi davranmaya çalışmaktadır. Yapay zekayı en kısa haliyle öğrenebilen yazılımlar olarak ifade edebiliriz. Bazen görüntüyü bazen sesi bazen ise sensörlerden topladığı verileri kullanarak öğrenirler. Geliştiricilerinin belirlediği algoritmalar sayesinde bunu yaparlar ve elde ettikleri sonuçlara göre kullanıdıkları alanlarda karar verme süreçlerinde yardımcı olurlar. Kısaca karar verme sürecinin hızlı ve hatasız yapılması gereken durumlarda artık yapay zeka uygulamaları kullanılmaktadır. Pazarlama alanından savunma sanayisine, eğitimden sağlığa, ekonomiden eğlenceye her alanda yapay zeka verimi arttırıp maliyetleri düşürmektedir.

PicoBricks ile yapacağımız bu projede konuşarak kontrol edebileceğiniz 2WD araba yapacağız. PicoBricks 2 adet 6V DC motoru ve bluetooth ile kablosuz iletişim kurmanızı sağlamaktadır.

HC05 bluetooth modülü PicoBricks ile cep telefonu arasında kablosuz olarak iletişim kurabilmemizi sağlayan bir modüldür. Projede cep telefonuna yüklenen mobil uygulama sayesinde telefondan gönderilen komutlar HC05 modülü üzerinden PicoBrickse iletilecek ve bu verilere göre de robot araba hareket edecektir. Cep telefonundan ileri, geri, sağ, sol butonları ile robot arabayı yönlendirebileceğimiz gibi sesli komutla da PicoBrickse veriler gönderebiliriz. Projede robot arabanın hareketlerini kontrol etmek için sesli komutlar vereceğiz.

6.16.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.16.4 Projenin MicroPython Kodu

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

uart = UART(0,9600) #If connection cannot be established, try 115200.
m1 = Pin(21, Pin.OUT)
m2 = Pin(22, Pin.OUT)

m1.low()
m2.low()

while True:
sleep(0.05)
if uart.any():
    cmd = uart.readline()
if cmd==b'F':
    m1.high()
    m2.high()
elif cmd==b'R':
    m1.high()
    m2.low()
elif cmd==b'L':
    m1.low()
    m2.high()
elif cmd==b'S':
    m1.low()
    m2.low()
cmd=""
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.16.5 Projenin Arduino C Kodu

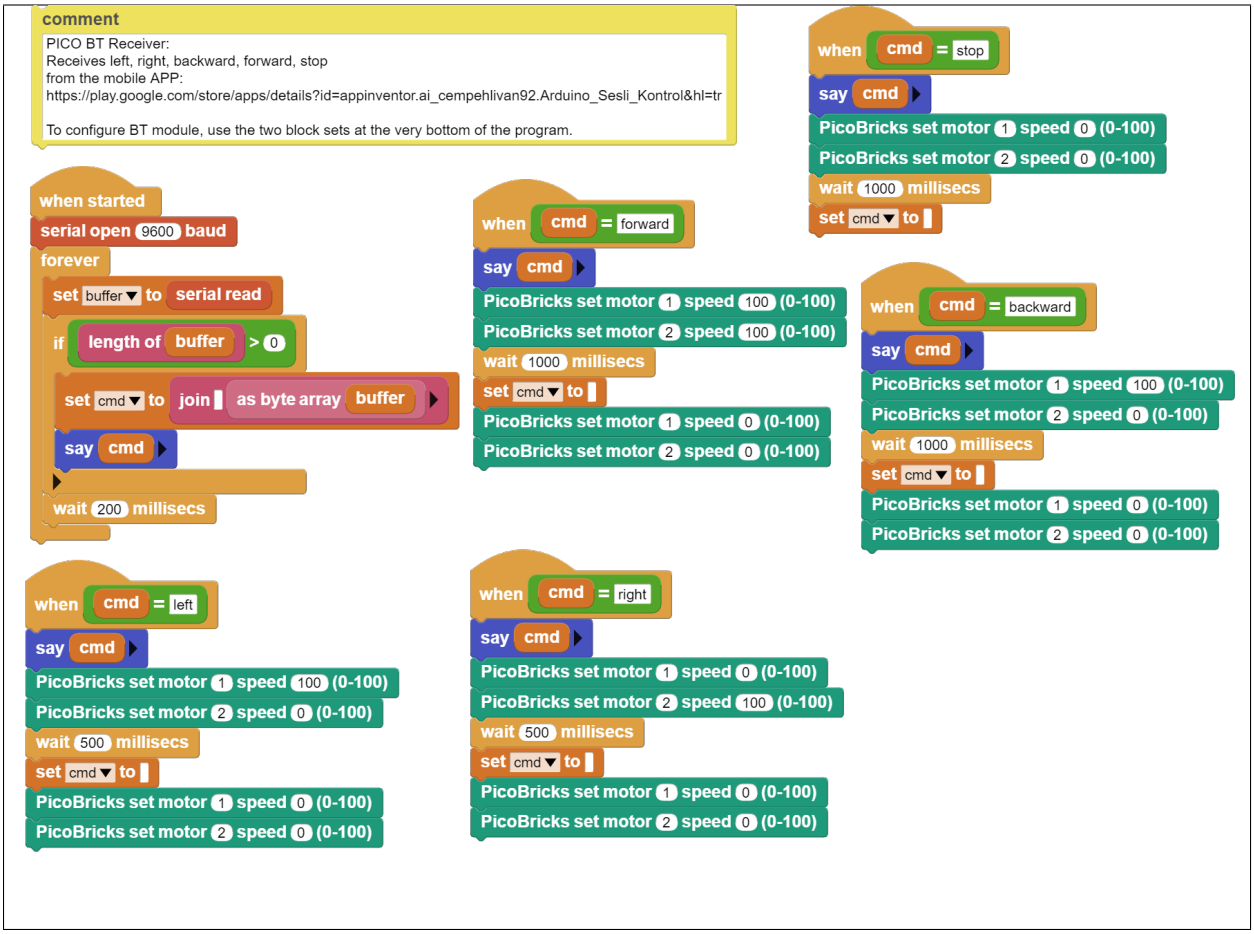
```
void setup() {
Serial1.begin(9600);
    }

void loop() {
if (Serial1.available() > 0) {

  char sread = Serial1.read();
  Serial.println(sread);

  if (sread == 'f') {
  Forward();
    }
  else if(sread == 'r'){
  Turn_Right();
    }
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
  else if(sread == 'l'){
  Turn_Left();
    }
  else if(sread == 's'){
  Stop();
    }
    }
    }

  void Forward(){
  digitalWrite(21,HIGH);
  digitalWrite(22,HIGH);
  delay(1000);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
    }
  void Turn_Left(){
  digitalWrite(21,LOW);
  digitalWrite(22,HIGH);
  delay(500);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
    }
  void Turn_Right(){
  digitalWrite(21,HIGH);
  digitalWrite(22,LOW);
  delay(500);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
    }
  void Stop(){
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
  delay(1000);
    }
```

6.16.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.17 Two Axis Robot Arm

6.17.1 Giriş

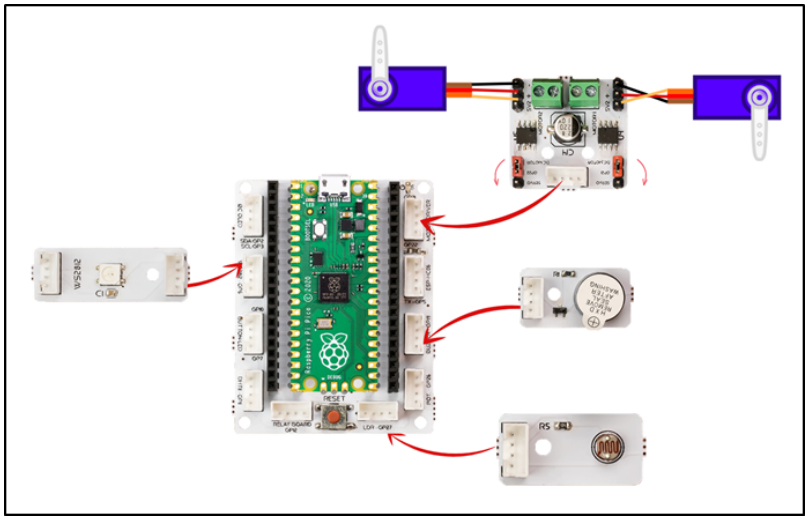
Bu projede, PicoBricks ile iki eksenli robot kol yapımını öğreneceğiz.

6.17.2 Proje Detayları ve Algoritması

Robot kollar endüstriyel alanda insan gücünün yerini almıştır. Bir insanın taşıyamayacağı ağırlık ve büyüklükteki yükleri taşıma ve döndürme işlerini fabrikalarda robot kollar üstlenmektedir. Milimetrenin binde biri hassasiyetinde konumlandırılabilmeleri de insan elinin sergileyebileceği hassasiyetin üzerindedir. Otomobil fabrikalarının üretim videolarını izlediğinde robot kolların ne kadar hayati bir öneme sahip olduğunu göreceksin. Robot denmesinin sebebi programlanarak sonsuz tekrarla aynı işi yapabilmelerinden kaynaklanmaktadır. Kol denmesinin sebebi ise bizim kollarımız gibi eklemli bir yapıya sahip olmasından kaynaklanmaktadır. Bir robot kolun kaç farklı doğrultuda dönme ve hareket etme kabiliyeti varsa o kadar eksenli olarak ifade edilmektedir. Robot kollar alüminyum ve çeşitli metalleri oyma ve şekil verme işlerinde de kullanılmaktadır. 7 eksenli CNC Router olarak geçen bu cihazlar bir heykeltraşın çamura şekil vermesi gibi metallere şekil verebilmektedirler. Robot kollarda kullanılma amacına göre bir tür elektrik motoru olan step motor ve servo motorlar kullanılmaktadır. PicoBricks servo motorlarla projeler yapmanıza olanak sağlamaktadır.

Kuruluma hazırlık amacıyla öncelikle servo motorları 0 dereceye ayarlamak için kodlarını yazıp yükleyeceğiz. LDR sensörünün üzerine bir cisim konulduğunda robot kol aşağı eğilecek ve açık olan kıskacını kapatacak. Kıskaç kapandıktan sonra robot kol tekrar yukarı kalkacak. Robot kolun her hareketinin sonucunda buzzer dan kısa bir bip sesi çıkacak. LDR sensörünün üzerine cisim yerleştirildiğinde RGB LED kırmızı renkte yanacak. Cisim robot kol tarafından tutulup havaya kaldırıldığında ise RGB LED yeşil renkte yanacak. Servo motorun hareketleri çok hızlıdır. Hareketi yavaşlatmak için 30 milisaniye aralıklarla 2 şer derece toplamda 90 derece hareket ile servo motorları kodlayacağız. Kıskacın kapanması için bunu yapmayacağız.

6.17.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.17.4 Projenin MicroPython Kodu

```
from machine import Pin, PWM, ADC
from utime import sleep
from picobricks import WS2812
#define libraries

ws = WS2812(6, brightness=0.3)
ldr=ADC(27)
buzzer=PWM(Pin(20, Pin.OUT))
servo1=PWM(Pin(21))
servo2=PWM(Pin(22))
#define LDR, buzzer and servo motors pins

servo1.freq(50)
servo2.freq(50)
buzzer.freq(440)
#define frequencies of servo motors and buzzer

RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLACK = (0, 0, 0) # RGB color settings
angleupdown=4770
angleupdown2=8200

def up():
global angleupdown
for i in range (45):
    angleupdown +=76
    servo2.duty_u16(angleupdown)
    sleep(0.03)
buzzer.duty_u16(2000)
sleep(0.1)
buzzer.duty_u16(0)
# servo2 goes up at specified intervals
def down():
global angleupdown
for i in range (45):
    angleupdown -=76
    servo2.duty_u16(angleupdown)
    sleep(0.03)
buzzer.duty_u16(2000)
sleep(0.1)
buzzer.duty_u16(0)
# servo2 goes down at specified intervals

def open():
global angleupdown2
for i in range (45):
    angleupdown2 +=500
    servo1.duty_u16(angleupdown2)
    sleep(0.03)
buzzer.duty_u16(2000)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
sleep(0.1)
buzzer.duty_u16(0)
# servo1 works for opening the clamps
def close():
global angleupdown2
for i in range (45):
    angleupdown2 -=500
    servo1.duty_u16(angleupdown2)
    sleep(0.03)
buzzer.duty_u16(2000)
sleep(0.1)
buzzer.duty_u16(0)
# servo1 works for closing the clamps
open()
servo2.duty_u16(angleupdown)
ws.pixels_fill(BLACK)
ws.pixels_show()
while True:
if ldr.read_u16()>20000:
    ws.pixels_fill(RED)
    ws.pixels_show()
    sleep(1)
    buzzer.duty_u16(2000)
    sleep(1)
    buzzer.duty_u16(0)
    open()
    sleep(0.5)
    down()
    sleep(0.5)
    close()
    sleep(0.5)
    up()
    ws.pixels_fill(GREEN)
    ws.pixels_show()
    sleep(0.5)
    # According to the data received from LDR, RGB LED lights red and green and servo␣
↪motors move
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.17.5 Projenin Arduino C Kodu

```
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define PIN        6
#define NUMPIXELS 1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

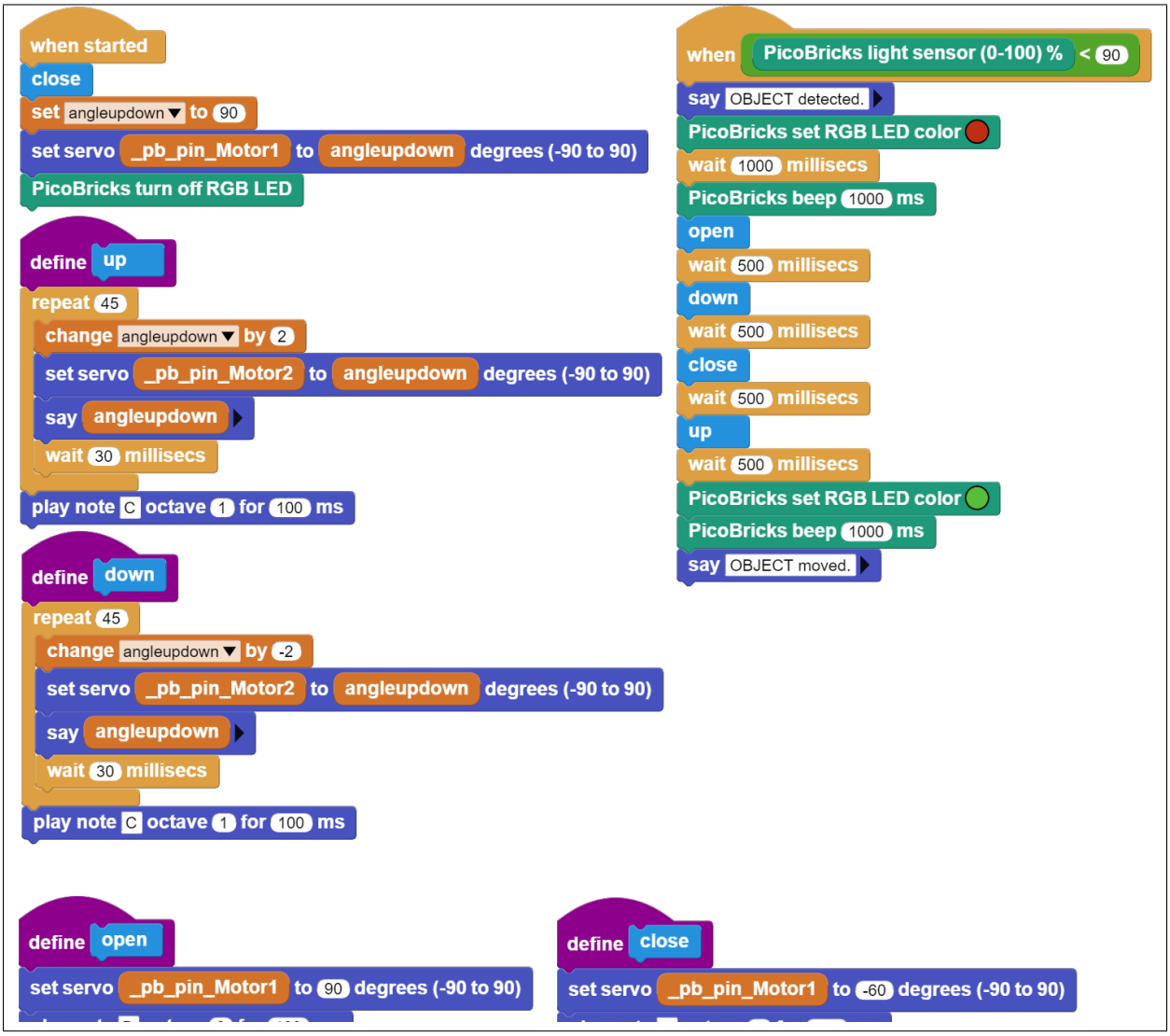
```
#define DELAYVAL 500
// define required libraries
#include <Servo.h>
Servo myservo1;
Servo myservo2;

int angleupdown;

void setup() {

pinMode(20,OUTPUT);
pinMode(27,INPUT);
// define input and output pins

pixels.begin();
pixels.clear();

myservo1.attach(21);
myservo2.attach(22); // define servo motor pins
Open();
angleupdown=180;
myservo2.write(angleupdown);

    }

void loop() {
if(analogRead(27)>150){

pixels.setPixelColor(0, pixels.Color(255, 0, 0));
pixels.show();
delay(1000);
tone(20,700);
delay(1000);
noTone(20);

Open();
delay(500);
Down();
delay(500);
Close();
delay(500);
Up();
pixels.setPixelColor(0, pixels.Color(0, 255, 0));
pixels.show();
delay(10000);
pixels.setPixelColor(0, pixels.Color(0, 0, 0));
pixels.show();
Open();
angleupdown=180;
myservo2.write(angleupdown);
// If the LDR data is greater than the specified limit, the buzzer will sound, the RGB␣
↪will turn red and servo motors will work
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
// The RGB will turn green when the movement is complete

    }
}

void Open(){
myservo1.write(180);
    }

void Close(){
myservo1.write(30);
    }

void Up(){

for (int i=0;i<45;i++){

angleupdown = angleupdown+2;
myservo2.write(angleupdown);
delay(30);
}
}

void Down(){

for (int i=0;i<45;i++){

angleupdown = angleupdown-2;
myservo2.write(angleupdown);
delay(30);
    }
    }
```

6.17.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.18 Smart House

6.18.1 Giriş

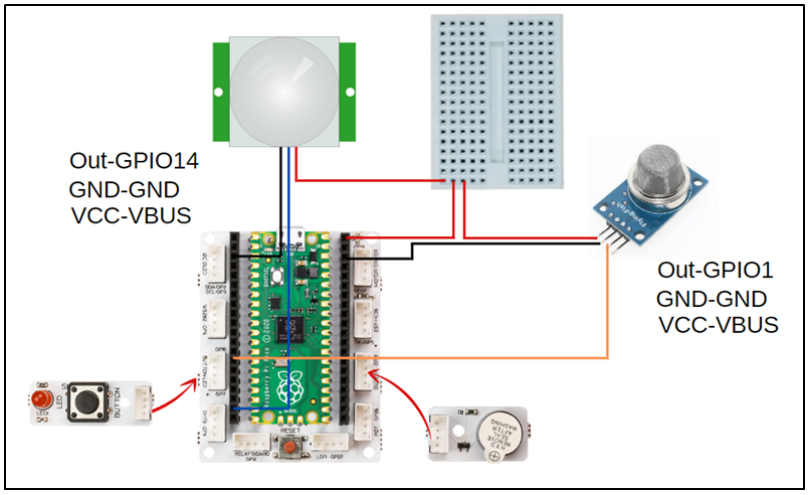
Bu sensör HC-SR501 , PIR sensörü olarak da ifade edilmekte, insan vücudunun yansıttığı kızılötesi dalgaların değişimlerini yakalayarak hareketi tespit etmektedirler. HC-SR501 PIR sensörü hareket algıladığında 3 saniye boyunca dijital çıkış vermektedir. Proje maketinde Picoboard, buzzer ve buton LED modülünü kullanacağız.

6.18.2 Projenin Detayları ve Algoritması

İşyerleri, fabrikalar, evlerimiz hatta hayvan barınakları… Yaşam alanlarımızı istemediğimiz davetsiz misafirlere karşı korumak için kullanılabilecek farklı elektronik sistemler bulunmaktadır. Bu sistemler ev ve iş yeri güvenlik sistemleri olarak üretilmekte ve pazarlanmaktadır. Güvenlik kameralarının ürettiği görüntülerin işlenip yorumlandığı sistemler bulunduğu gibi insan bedenini ve hareketlerinin sensörler ile tespit edip harekete geçen güvenlik sistemleri de mevcuttur. Güvenlik sistemleri bir tür alarmlı saat gibi kurulur ve belirlenen zaman diliminde tanımlanmayan bir hareketlilik algılandığında sesli ve ışıklı uyarılar verir. İşletme veya ev sahibine bildirimde bulunur ayrıca güvenlik birimlerine de otomatik bildirimlerde bulunabilmektedir. Gaz kaçağı yangın vb. durumlarda zehirlenmeleri önlemek için ev ve işyerlerinde gaz sensörleri kullanılır. Olumsuz bir durumda yüksek sesle alarm vererek ortamda yaşayan insanlar uyarılır. PicoBricks ile HC-SR501 ve MQ-2 gaz sensörü kullanarak maket bir akıllı ev projesi hazırlayacağız.

Tüm parçalar maketin içinde olmalı. Picobricks başladığında alarm sisteminin devreye girmesi için butona basılması gerekmektedir. Butona basıldıktan sonra elin maketi içinden çekilmesi için 3 saniye bekleme koymalıyız. 3 saniyenin sonunda kırmız LED yanar ve alarm sistemi devreye girer. Alarm sistemi devredeyken bir hareket algıladığında kırmızı LED yanıp sönmeye başlar ve buzzer'dan alarm sesi çalınır. Susturmak için Picobricks'in yeniden başlatılması gerekmelidir. MQ-2 sensörü ise sürekli devrededir. Zehirli bir gaz algıladığında ise buzzer ve kırmızı LED ile bunu bildirecektir.

6.18.3 Bağlantı Diyagramlarını



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.18.4 Projenin MicroPython Kodu

```
from machine import Pin, PWM
from utime import sleep
# define libraries
PIR=Pin(14, Pin.IN)
MQ2=Pin(1,Pin.IN)
buzzer=PWM(Pin(20,Pin.OUT))
redLed=Pin(7,Pin.OUT)
button=Pin(10,Pin.IN,Pin.PULL_DOWN)
# define output and input pins

activated=0
gas=0

while True:
if button.value()==1:
    activated=1
    gas=0
    sleep(3)
    redLed.value(1)
    buzzer.duty_u16(0)
if MQ2.value()==1:
    gas=1
if activated==1:
    if PIR.value()==1:
        buzzer.duty_u16(6000)
        buzzer.freq(440)
        sleep(0.2)
        buzzer.freq(330)
        sleep(0.1)
        buzzer.freq(494)
        sleep(0.15)
        buzzer.freq(523)
        sleep(0.3)
if gas==1:
    buzzer.duty_u16(6000)
    buzzer.freq(330)
    sleep(0.5)
    redLed.value(1)
    buzzer.freq(523)
    sleep(0.5)
    redLed.value(0)
    # LED will light and buzzer will sound when PIR detects motion or MQ2 detects toxic␣
↪gas
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.18.5 Projenin Arduino C Kodu

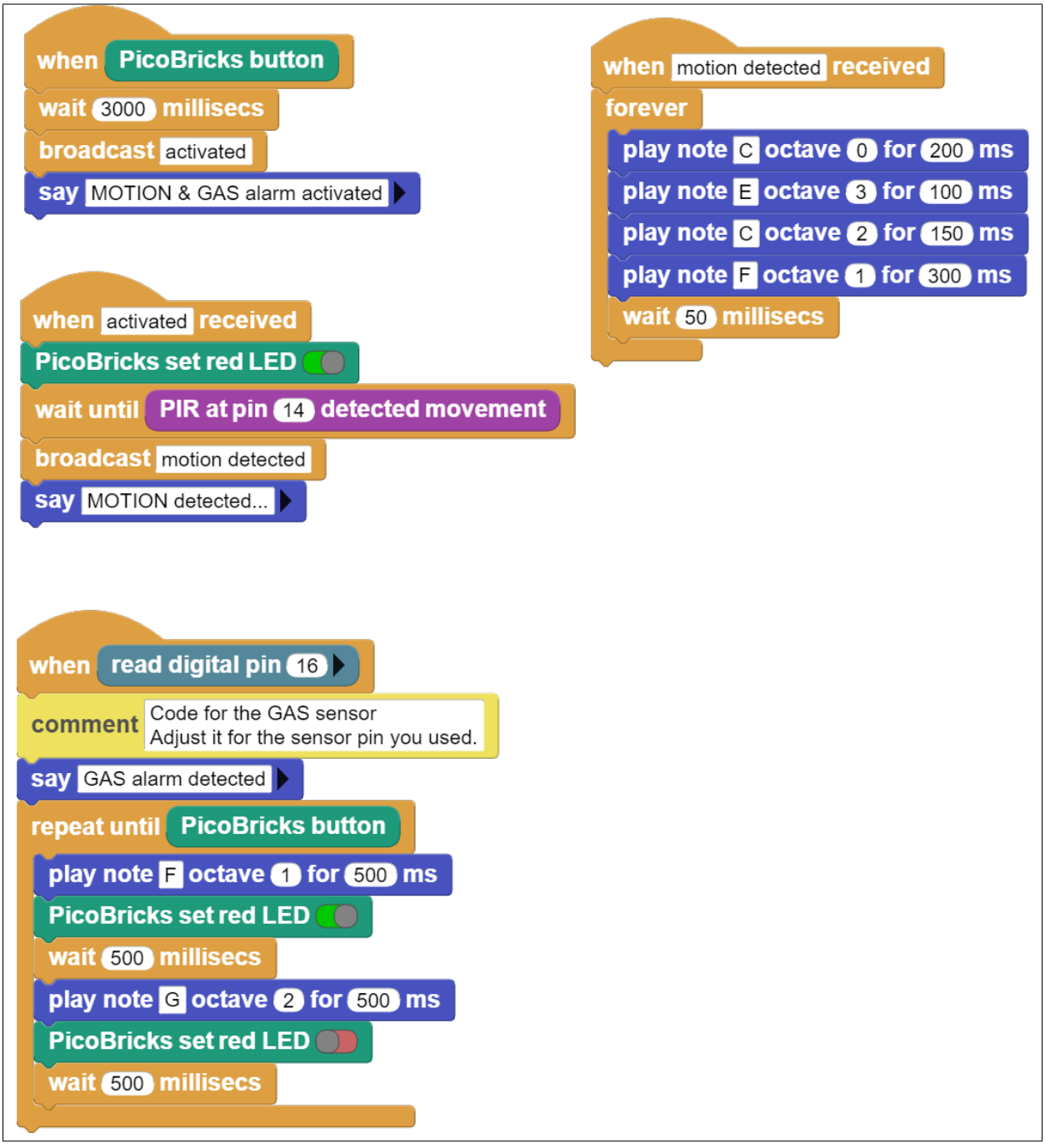
```
void actived (){
digitalWrite(7,1);
while(!(digitalRead(14) == 1))
    {
_loop();
    }
motion_detected();
    }

void motion_detected (){
while(1) {
  // buzzer settings
  tone(20,262,0.25*1000);
  delay(0.25*1000);
  tone(20,330,0.25*1000);
  delay(0.25*1000);
  tone(20,262,0.25*1000);
  delay(0.25*1000);
  tone(20,349,0.25*1000);
  delay(0.25*1000);
// sound the buzzer when PIR detected a motion
  _loop();
    }
    }

void _delay(float seconds) {
long endTime = millis() + seconds * 1000;
while(millis() < endTime) _loop();
    }

void _loop() {
    }

void loop() {
_loop();
    }

void setup() {

pinMode(10,INPUT);
pinMode(1,INPUT);
pinMode(20,OUTPUT);
pinMode(7,OUTPUT);
pinMode(14,INPUT);
// define input and output pins

while(1) {
  if(digitalRead(10) == 1){
      _delay(3);
      actived();
  }
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
  if(digitalRead(1) == 1){
      while(!(digitalRead(10) == 1))
      {
        _loop();
        tone(20,349,0.5*1000);
        delay(0.5*1000);
        digitalWrite(7,1);
        _delay(0.5);
        tone(20,392,0.5*1000);
        delay(0.5*1000);
        digitalWrite(7,0);
        _delay(0.5);
      }
  }
  _loop();
    }
    }
```

6.18.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.19 NFC Akıllı Kapı

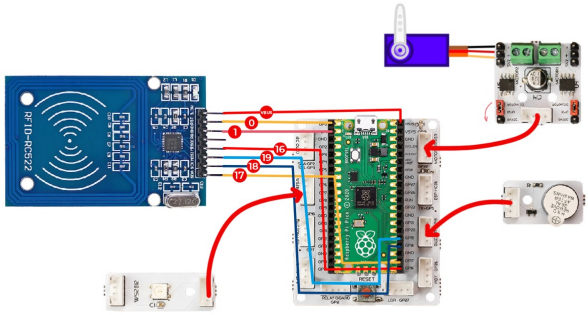
6.19.1 Giriş

Bu projede, kartlı giriş sistemi olan bir model ev yapacağız.

6.19.2 Proje Detayları ve Algoritması

`Güvenlik sistemlerinin` kapsamına bina ve oda girişlerinde yetkileri kontrol edebilen teknolojiler de girmektedir. Hastanelerin ameliyathanelerine sadece görevli personellerin girebildiği kartlı giriş sistemleri ilk akla gelen örneklerden biridir. Ayrıca askeri güvenlik merkezlerinde her düzeyden kişi veya personelin girmemesi gereken alanların giriş kapıları kartlı ve şifreli giriş teknolojileriyle donatılmaktadır. Bina ve oda girişlerinde kullanılan bu elektronik sistemler yetkisiz kişilerin girişini engellediği gibi giriş çıkış bilgilerinin kayıt altında tutulmasını da sağlamaktadır. Şifreli giriş, kartlı giriş, parmak izi tarama, yüz tarama, retina taraması ve ses tanıma teknolojileri elektronik giriş sistemlerinde kullanılan doğrulama yöntemleridir. RFID ve NFC gibi sistemler bugün temassız ödeme teknolojilerinin temel halleridir. Kredi kartlarındaki temassız ödeme teknolojisi teknik olarak farklı olsa da çalışma mantığı aynıdır. Okuyucu ve kart arasında yer alacak maksimum mesafe, kullanılan teknolojileri birbirinden ayıran özelliklerin başında gelir. Alışveriş mağazalarından çıkarken özellikle giyim mağazalarında ürünlerin üzerindeki NFC etiketler girişteki okuyuculara takılırsa öterler. O sistemlerde de bir tür `RFID teknolojisi` kullanılmaktadır. Bu projede maket ev üzerinde kartlı giriş sistemi hazırlayacağız. Kullanacağımız elektronik bileşenler MFRC522 RFID okuyucu ve 13.56 Mhz'lik kartlardır.

6.19.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.19.4 Projenin MicroPython Kodu

```
from machine import I2C, Pin, SPI, PWM
from mfrc522 import MFRC522
from ws2812 import NeoPixel
from utime import sleep

servo = PWM(Pin(21))
servo.freq(50)
servo.duty_u16(1350) #servo set 0 angle 8200 for 180.

buzzer = PWM(Pin(20, Pin.OUT))
buzzer.freq(440)

neo = NeoPixel(6, n=1, brightness=0.3, autowrite=False)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLACK = (0, 0, 0)

sck = Pin(18, Pin.OUT)
mosi = Pin(19, Pin.OUT)
miso = Pin(16, Pin.OUT)
sda = Pin(17, Pin.OUT)
rst = Pin(15, Pin.OUT)
spi = SPI(0, baudrate=100000, polarity=0, phase=0, sck=sck, mosi=mosi, miso=miso)
homeowner = "0x734762a3"
rdr = MFRC522(spi, sda, rst)

while True:

(stat, tag_type) = rdr.request(rdr.REQIDL)
if stat == rdr.OK:
    (stat, raw_uid) = rdr.anticoll()
    if stat == rdr.OK:
        buzzer.duty_u16(3000)
        sleep(0.05)
        buzzer.duty_u16(0)
        uid = ("0x%02x%02x%02x%02x" % (raw_uid[0], raw_uid[1], raw_uid[2], raw_uid[3]))
        print(uid)
        sleep(1)
        if (uid==homeowner):
            neo.fill(GREEN)
            neo.show()
            servo.duty_u16(6000)
            sleep(3)
            servo.duty_u16(1350)
            neo.fill(BLACK)
            neo.show()

        else:
            neo.fill(RED)
            neo.show()
            sleep(3)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
            neo.fill(BLACK)
            neo.show()
            servo.duty_u16(1350)
```

6.19.5 MicroPyhton ID Kart Kodu

```
from machine import Pin, SPI
from mfrc522 import MFRC522
import utime
#define libraries
sck = Pin(18, Pin.OUT)
mosi = Pin(19, Pin.OUT)
miso = Pin(16, Pin.OUT)
sda = Pin(17, Pin.OUT)
rst = Pin(15, Pin.OUT)
spi = SPI(0, baudrate=100000, polarity=0, phase=0, sck=sck, mosi=mosi, miso=miso)
rdr = MFRC522(spi, sda, rst)
#define MFRC522 pins

while True:
(stat, tag_type) = rdr.request(rdr.REQIDL)
if stat == rdr.OK:
    (stat, raw_uid) = rdr.anticoll()
    if stat == rdr.OK:
        uid = ("0x%02x%02x%02x%02x%02x" % (raw_uid[0], raw_uid[1], raw_uid[2], raw_uid[3]))
        print(uid)
        utime.sleep(1)
        #read the card and give the serial number of the card
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.19.6 Projenin Arduino C Kodu

```
#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>
#include <Adafruit_NeoPixel.h>
//Define libraries.


#define RST_PIN    26
#define SS_PIN     17
#define servoPin   22
#define PIN        6
#define NUMPIXELS  1
#define buzzer     20
//define pins of servo,buzzer,neopixel and rfid.
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
Servo motor;
MFRC522 rfid(SS_PIN, RST_PIN);

byte ID[4] = {"Write your own ID."};

void setup() {
pixels.begin();
motor.attach(servoPin);
Serial.begin(9600);
SPI.begin();
rfid.PCD_Init();
pinMode(buzzer, OUTPUT);

    }

void loop()
    {
pixels.clear();

if ( ! rfid.PICC_IsNewCardPresent())
return;
if ( ! rfid.PICC_ReadCardSerial())
return;

if
(rfid.uid.uidByte[0] == ID[0] &&
rfid.uid.uidByte[1] == ID[1] &&
rfid.uid.uidByte[2] == ID[2] &&
rfid.uid.uidByte[3] == ID[3] )
{
    Serial.println("Door Opened.");
    printid();
    tone(buzzer,523);
    delay(200);
    noTone(buzzer);
    delay(100);
    tone(buzzer,523);
    delay(200);
    noTone(buzzer);
    pixels.setPixelColor(0, pixels.Color(0, 250, 0));
    delay(200);
    pixels.show();
    pixels.setPixelColor(0, pixels.Color(0, 0, 0));
    delay(200);
    pixels.show();
    motor.write(180);
    delay(2000);
    motor.write(0);
    delay(1000);
 //RGB LED turns green and the door opens thanks to the servo motor if the correct card␣
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
↪is read to the sensor.
    }
    else
    {
  Serial.println("Unknown Card.");
  printid();
  tone(buzzer,494);
  delay(200);
  noTone(buzzer);
  delay(100);
  tone(buzzer,494);
  delay(200);
  noTone(buzzer);
  pixels.setPixelColor(0, pixels.Color(250, 0, 0));
  delay(100);
  pixels.show();
  pixels.setPixelColor(0, pixels.Color(0, 0, 0));
  delay(100);
  pixels.show();
  //RGB LED turns red and the door does not open if the wrong card is read to the sensor
    }
rfid.PICC_HaltA();
    }
void printid()
    {
Serial.print("ID Number: ");
for(int x = 0; x < 4; x++){
Serial.print(rfid.uid.uidByte[x]);
Serial.print(" ");
    }
Serial.println("");
    }
```

6.19.7 Arduino Kart ID Kodu

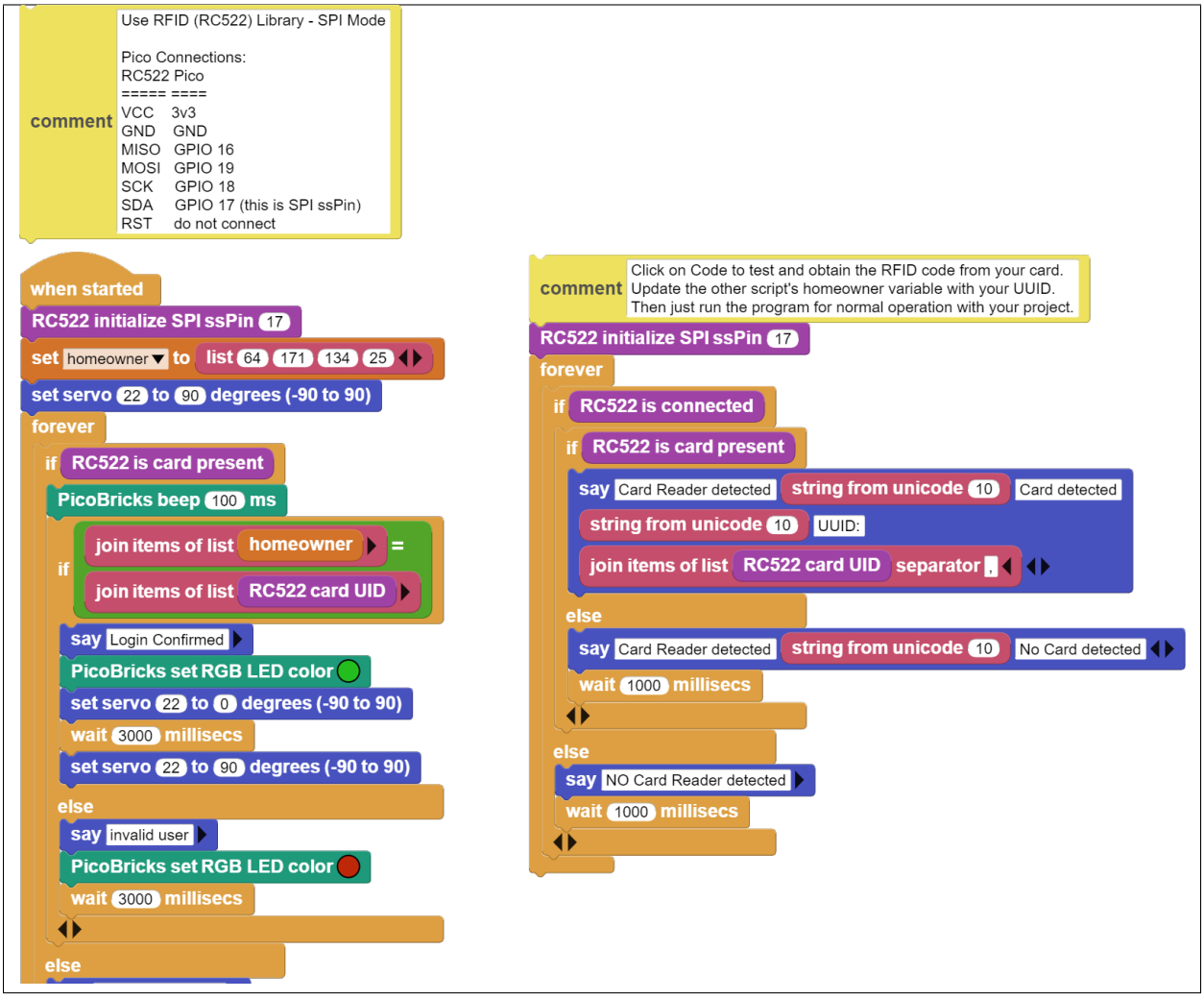
```
#include <SPI.h>
#include <MFRC522.h>
//define libraries

int RST_PIN = 26;
int SS_PIN = 17;
//define pins

MFRC522 rfid(SS_PIN, RST_PIN);

void setup()
    {
Serial.begin(9600);
SPI.begin();
rfid.PCD_Init();
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
    }

void loop() {

if (!rfid.PICC_IsNewCardPresent())
return;
if (!rfid.PICC_ReadCardSerial())
return;
rfid.uid.uidByte[0] ;
rfid.uid.uidByte[1] ;
rfid.uid.uidByte[2] ;
rfid.uid.uidByte[3] ;
printid();
rfid.PICC_HaltA();
    //Reading your ID.
    }
void printid()
    {
Serial.print("Your ID: ");
for (int x = 0; x < 4; x++) {
Serial.print(rfid.uid.uidByte[x]);
Serial.print(" ");
    }
Serial.println("");
    }
```

6.19.8 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.20 Automatic Trash Bin

6.20.1 Giriş

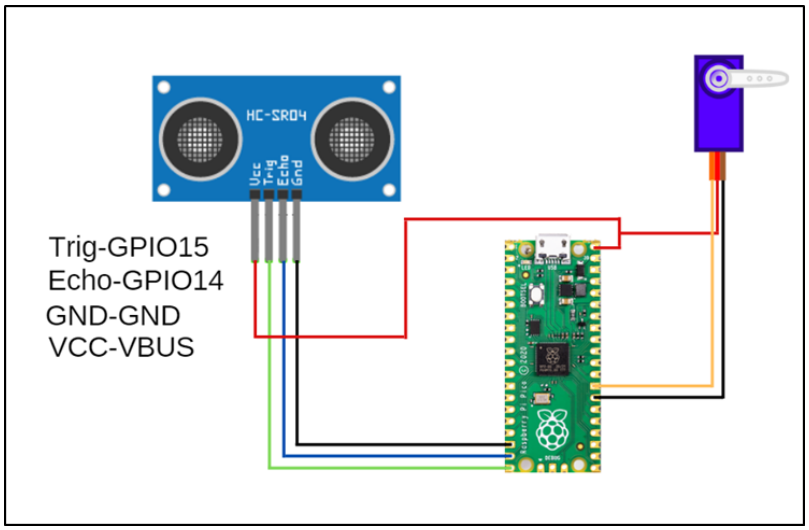
Bu projemizde PicoBricks ile ultrasonik sensör ve servo motor kullanarak odanıza mobil ve otomatik şık bir çöp kutusu yapacaksınız.

6.20.2 Proje Detayları ve Algoritması

Covid-19 pandemisi insanların günlük yaşamdaki rutinlerini bir çok alanda değiştirdi. Temizlik, çalışma, alışveriş, sosyal hayat gibi bir çok alanda insanlar uymak zorunda olduğu bir dizi yeni kurallarla tanıştı. Covid-19 yeni iş alanlarının doğmasına ve gelişmesine zemin oluşturduğu gibi bazı ürünlerinde ön plana çıkmasını sağlamıştır. El hijyenin çok önemsendiği bir dönemde kimse çöpünü atmak için çöp kovasının kapağına dokunmak istemezdi. Yanına yaklaşıldığında kapağı otomatik açılan dolduğunda ise tek bir hareketle içindeki poşeti büzüp çıkarıp atmaya hazır hale getiren çöp kovaları maliyetinin çok üstünde fiyatlara alıcı buldu. Ayrıca otomatik dezenfektan makineleri elimizi altına tuttuğumuzda belirli bir miktar sıvıyı avucumuzun içine boşaltmakla temassız bir hijyen sağlıyordu. Otomatik dezenfektan sıkıcılarda maliyetinin çok üzerindeki fiyatlara raflarda yer aldı. Bu iki üründe çalışma sistemi açısından benzerlikler yer almaktadır. Otomatik dezenfektan sıkıcılarda doğrudan elektrik motorlu bir pompa sıvıyı dışarı aktardığı gibi bazı modellerde de servo motorun gücü ile pompalanma sistemine dayanan cihazlar bulunmaktaydı. Otomatik çöp kovalarında ise kapağı açan servo motor kullanılmakta el hareketini algılamak için ise kızılötesi ya da ultrasonic sensörler kullanılmaktaydı.

Bu projede HC-SR04 ultrasonic mesafe sensörü ve SG90 servo motor kullanılacaktır. Çöp kutusunun kapağının önüne kullanıcı elini yaklaştırdığında mesafe sensörü yakınlığı algılayacak ve Picobrickse gönderecektir. Picobricks de bu bilgiye göre servo motor çalıştırarak çöp kovasının kapağını açacak kısa bir süre sonra tekrar aşağı indirecektir.

6.20.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.20.4 Projenin MicroPython Kodu

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

servo=PWM(Pin(21,Pin.OUT))
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)

servo.freq(50)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
servo.duty_u16(1920) #15 degree

def getDistance():
trigger.low()
utime.sleep_us(2)
trigger.high()
utime.sleep_us(5)
trigger.low()
while echo.value() == 0:
   signaloff = utime.ticks_us()
while echo.value() == 1:
   signalon = utime.ticks_us()
timepassed = signalon - signaloff
distance = (timepassed * 0.0343) / 2
print("The distance from object is ",distance,"cm")
return distance

while True:
sleep(0.01)
if int(getDistance())<=10:
    servo.duty_u16(4010) #70 degree
    utime.sleep(0.3)
    servo.duty_u16(1920)
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.20.5 Projenin Arduino C Kodu

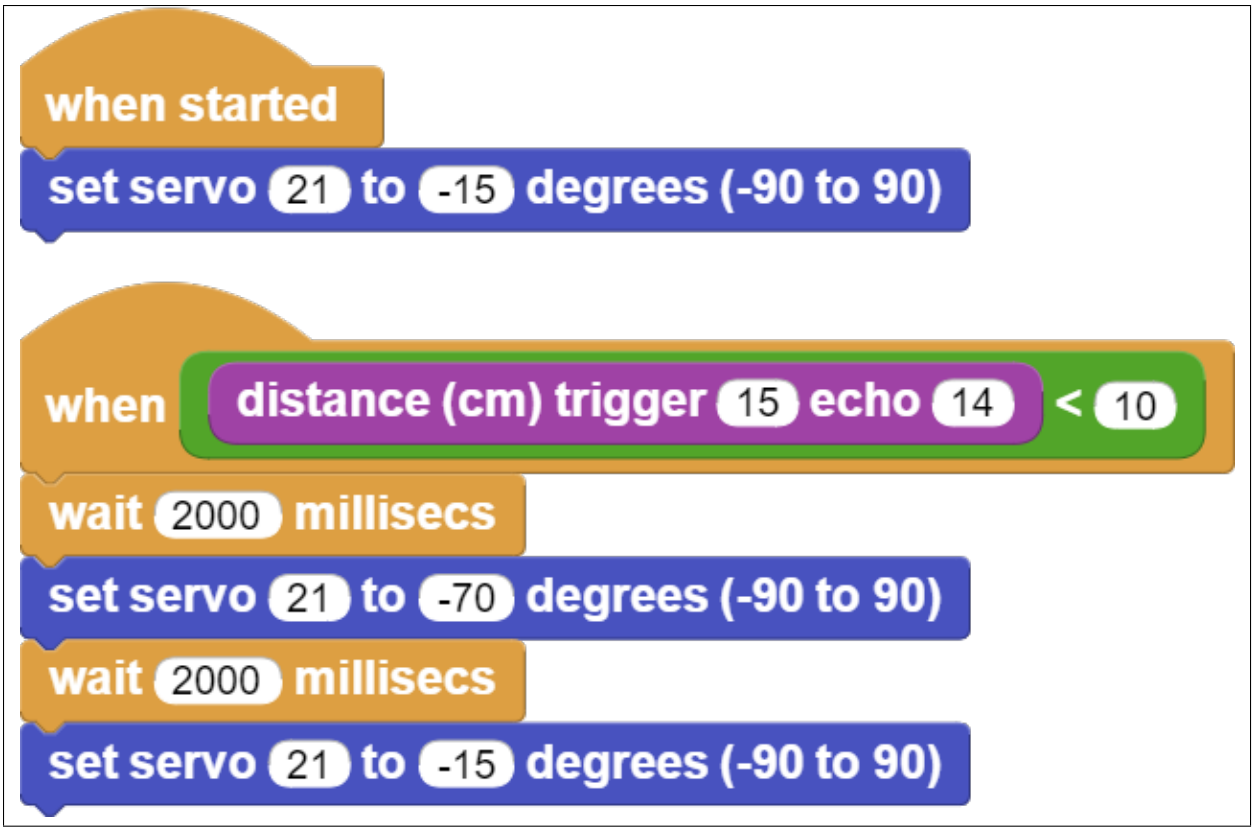
```
#include <Servo.h>
#define trigPin 14
#define echoPin 15
Servo servo;
void setup() {
Serial.begin (9600);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
servo.attach(21);
    }

void loop() {
long duration, distance;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = (duration/2) / 29.1;
if (distance < 80) {
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
Serial.print(distance);
Serial.println(" cm");
servo.write(179);
    }

else if (distance<180) {
Serial.print(distance);
Serial.println(" cm");
servo.write(100);
    }


    }
```

6.20.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.21 Digital Ruler

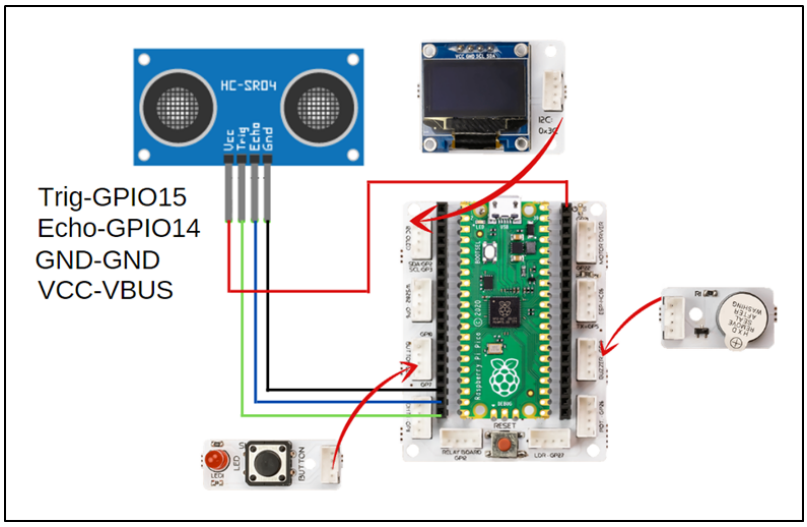
6.21.1 Giriş

Projemizde PicoBricks ile ultrasonik sensör kullanarak mesafe değerini butona basıldığında OLED ekranda göstereceğimiz dijital bir cetvel hazırlayacağız.

6.21.2 Projenin Detayları ve Algoritması

Uzunluk ölçmek için birçok araç kullanılmaktadır. Bu araçların başında cetveller gelmektedir. Ölçeceğimiz yere ve büyüklüğüne göre ölçü aletimiz farklılaşmaktadır. İnşaat ve mimaride şerit metreler , küçük ve milimetrik hassasiyet gerektiren nesneler için kumpaslar kullanılmaktadır. Ayrıca hem büyük hem de hassas ölçüm yapılması gereken bir alan ölçülmek isteniyorsa lazer ve kızılötesi sistemler ile çalışan mesafe ölçerler kullanılmaktadır. Sağlık sektöründe kullanılan ultrasonografi cihazları da benzer mantıkla çalışmakta ancak ölçümlerini görsellere dönüştürmektedir. Ultrasonik sensörler yaydıkları ses dalgalarının geri dönüş sürelerine göre mesafe tespiti yaparlar.

6.21.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.21.4 Projenin MicroPython Kodu

```
from machine import Pin, PWM, I2C
from utime import sleep
from picobricks import SSD1306_I2C
import utime
#define the libraries
redLed=Pin(7,Pin.OUT)
button=Pin(10,Pin.IN,Pin.PULL_DOWN)
buzzer=PWM(Pin(20,Pin.OUT))
buzzer.freq(392)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)
#define input and output pins
WIDTH  = 128
HEIGHT = 64
#OLED screen settings
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
#initialize digital pin 4 and 5 as an OUTPUT for OLED communication
oled = SSD1306_I2C(128, 64, i2c)
measure=0
finalDistance=0

def getDistance():
trigger.low()
utime.sleep_us(2)
trigger.high()
utime.sleep_us(5)
trigger.low()
while echo.value() == 0:
   signaloff = utime.ticks_us()
while echo.value() == 1:
   signalon = utime.ticks_us()
timepassed = signalon - signaloff
distance = (timepassed * 0.0343) / 2
return distance
#calculate the distance
def getMeasure(pin):
global measure
global finalDistance
redLed.value(1)
for i in range(20):
    measure += getDistance()
    sleep(0.05)
redLed.value(0)
finalDistance = (measure/20) + 1
oled.fill(0)
oled.show()
oled.text(">Digital Ruller<", 2,5)
oled.text("Distance " + str(round(finalDistance)) +" cm", 0, 32)
oled.show()
#print the specified distance to the specified x and y coordinates on the OLED screen
print(finalDistance)
buzzer.duty_u16(4000)
sleep(0.05)
buzzer.duty_u16(0)
measure=0
finalDistance=0
#sound the buzzer
button.irq(trigger=machine.Pin.IRQ_RISING, handler=getMeasure)
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.21.5 Projenin Arduino C Kodu

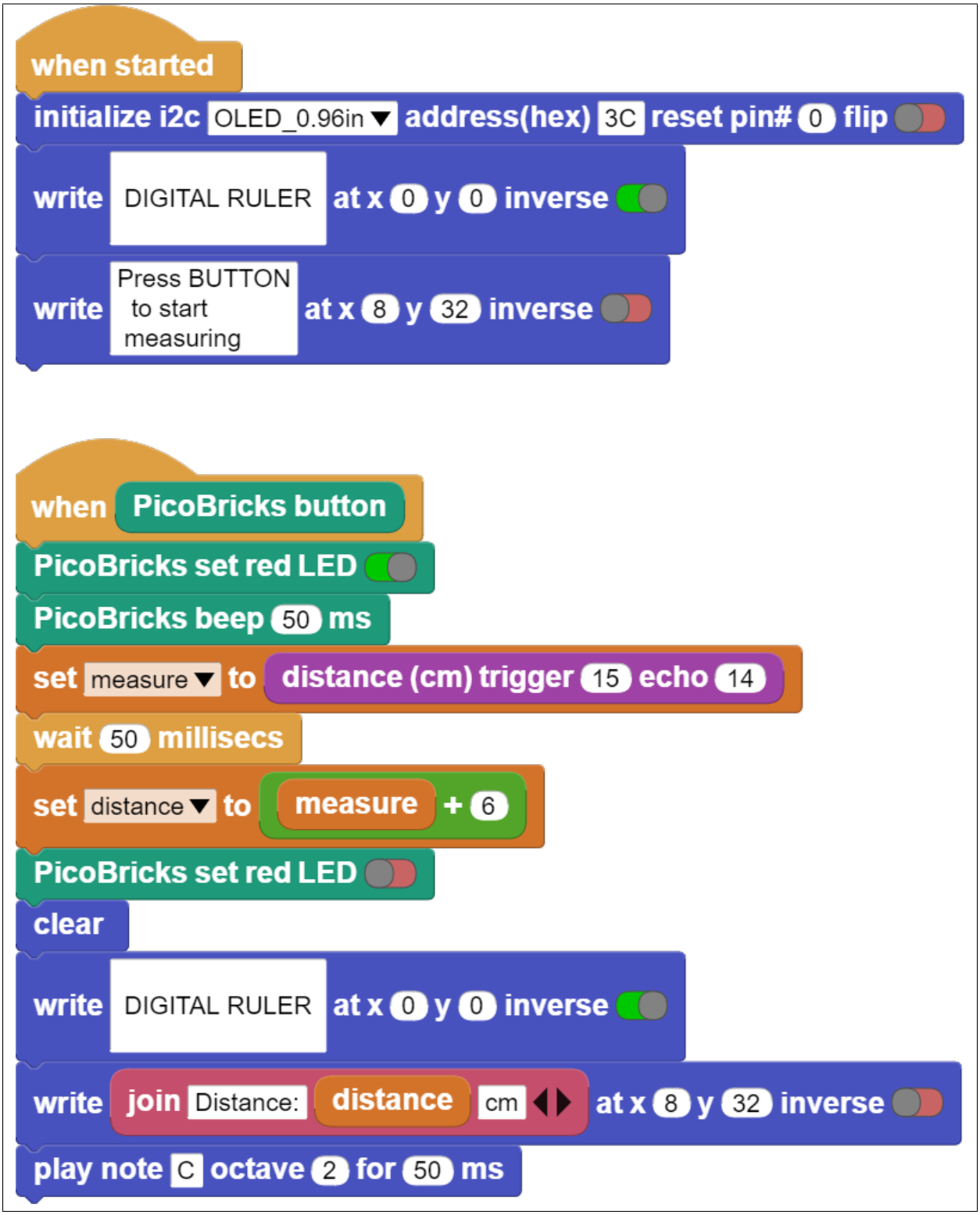
```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"
#include <NewPing.h>
// define the libraries
#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

#define T_B 493

int distance = 0;
int total = 0;

void setup() {
pinMode(7,OUTPUT);
pinMode(20,OUTPUT);
pinMode(10,INPUT);
// define input and output pins
Wire.begin();
oled.init();
oled.clearDisplay();


    }

void loop() {

delay(50);
if(digitalRead(10) == 1){

int measure=0;
digitalWrite(7,HIGH);
tone(20,T_B);
delay(500);
noTone(20);

for (int i=0;i<20;i++){

  measure=sonar.ping_cm();
  total=total+measure;
  delay(50);
    }

distance = total/20+6; // calculate the distance
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
digitalWrite(7,LOW);

delay(1000);
oled.clearDisplay();
oled.setTextXY(2,1);
oled.putString(">Digital Ruler<");
oled.setTextXY(5,1);
oled.putString("Distance: ");
oled.setTextXY(5,10);
String string_distance=String(distance);
oled.putString(string_distance);
oled.setTextXY(5,12);
oled.putString("cm"); // print the calculated distance on the OLED screen

measure=0;
distance=0;
total=0;
    }
}
```

6.21.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız

yeterlidir.

6.22 Air Piano

6.22.1 Giriş

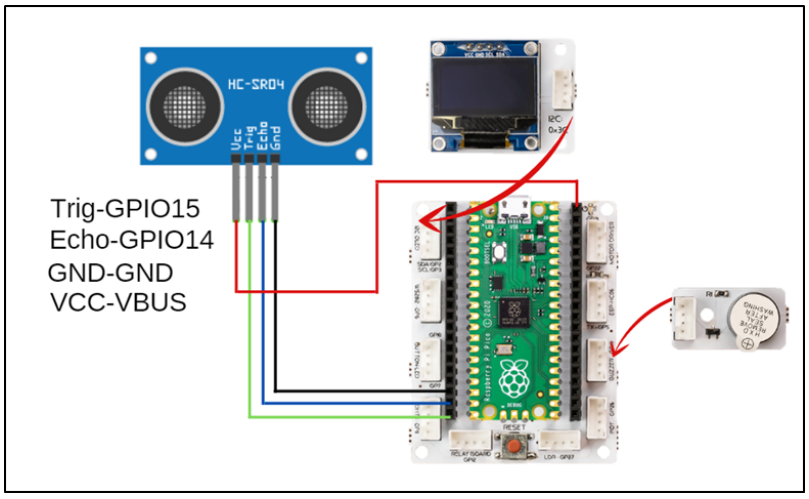
Bu projede PicoBricks ile 8 nota çalabilen basit bir piyano yapacağız. Bu piyanonun hoparlörü buzzer olacak. Piyanonun tuşlarının görevini ise ultrasonik sensör üstlenecek.

6.22.2 Proje Detayları ve Algoritması

Elektronik teknolojisinin gelişimi ile üretimi zor, pahalı, yüksek kaliteli ses üreten müzik aletleri dijitalleşmiştir. Piyanolar bu enstrümanların başında gelmektedir. Dijital piyanoların her bir tuşu farklı frekansta elektrik sinyalleri üretir. Böylelikle hoparlörlerinden 88 farklı notayı çalabilmektedir. Dijital enstrümanların tuşlarının gecikme süresi, hoparlörün kalitesi, sesin çözünürlüğü gibi faktörler kaliteyi etkileyen faktörler olarak ortaya çıkmıştır. Elektro gitarlarda tuşlar yerine tellerdeki titreşimler dijitalleştirilir. Üflemeli enstrümanlarda ise ses çıkışına takılan yüksek çözünürlüklü mikrofonlar sayesinde çalınan notalar elektrik sinyallerine dönüştürülüp kayıt edilebilmektedir. Elektronik teknolojisindeki bu gelişim yüksek maliyetli müzik aletlerine ulaşım kolaylaştırmış, müzik eğitimi daha geniş çeşitliliğe kavuşmuş ve daha geniş kitleye yayılmıştır.

Bu projede `HC-SR04 Ultrasonik mesafe sensörü` ve PicoBricks üzerindeki `buzzer modülünü` kullanarak piyano uygulaması yapacağız. Mesafe sensöründen gelen değerlere göre buzzer ın farklı notalar çalmasını sağlayacak ve sensöre elimizi yaklaştırıp uzaklaştırarak melodiler oluşturacağız. Ayrıca OLED ekrana anlık olarak mesafe çalınan nota bilgilerini yazdıracağız.

6.22.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.22.4 Projenin MicroPython Kodu

```
from machine import Pin, PWM, I2C
from utime import sleep
import utime
from picobricks import SSD1306_I2C
import _thread
#define the libraries

buzzer=PWM(Pin(20,Pin.OUT))
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)
#define the input and Output pins

WIDTH  = 128
HEIGHT = 64
#OLED screen settings

sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
#initialize digital pin 4 and 5 as an OUTPUT for OLED communication

oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)

measure=0

def getDistance():
trigger.low()
utime.sleep_us(2)
trigger.high()
utime.sleep_us(5)
trigger.low()
while echo.value() == 0:
   signaloff = utime.ticks_us()
while echo.value() == 1:
   signalon = utime.ticks_us()
timepassed = signalon - signaloff
distance = (timepassed * 0.0343) / 2
return distance
#calculate distance

def airPiano():
while True:
    global measure

    if measure>5 and measure<11:
        buzzer.duty_u16(4000)
        buzzer.freq(262)
        sleep(0.4)

    elif measure>10 and measure<16:
        buzzer.duty_u16(4000)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
        buzzer.freq(294)
        sleep(0.4)

    elif measure>15 and measure<21:
        buzzer.duty_u16(4000)
        buzzer.freq(330)
        sleep(0.4)

    elif measure>20 and measure<26:
        buzzer.duty_u16(4000)
        buzzer.freq(349)
        sleep(0.4)

    elif measure>25 and measure<31:
        buzzer.duty_u16(4000)
        buzzer.freq(392)
        sleep(0.4)

    elif measure>30 and measure<36:
        buzzer.duty_u16(4000)
        buzzer.freq(440)
        sleep(0.4)

    elif measure>35 and measure<41:
        buzzer.duty_u16(4000)
        buzzer.freq(494)
        sleep(0.4)
    else:
        buzzer.duty_u16(0)

_thread.start_new_thread(airPiano, ())
#play the tone determined by the value of the distance sensor

while True:
measure=int(getDistance())
oled.text("Distance " + str(measure)+ " cm", 5,30)
oled.show()
sleep(0.01)
oled.fill(0)
oled.show()
#write the specified texts to the determined x and ye coordinates on the OLED screen
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.22.5 Projenin Arduino C Kodu

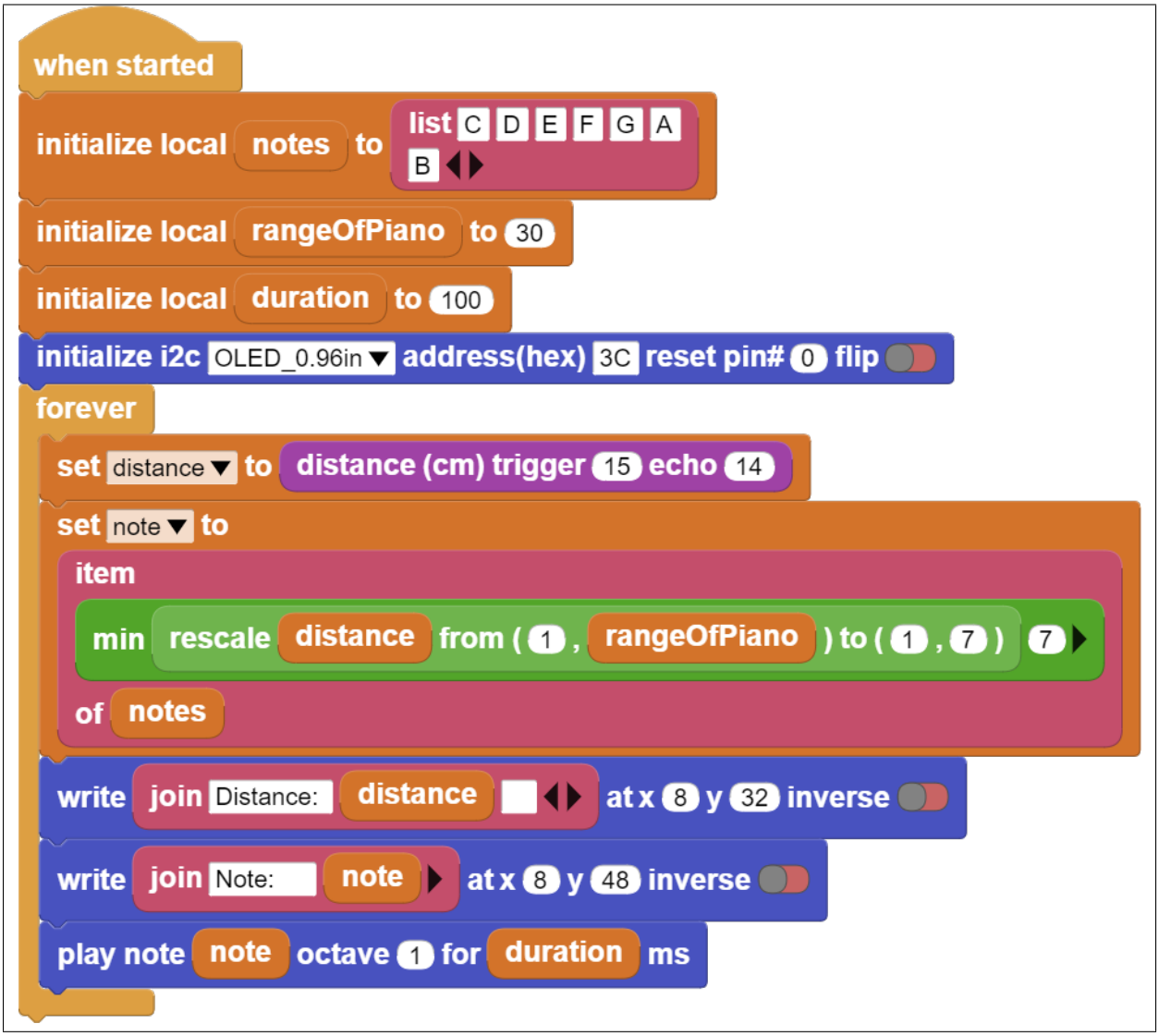
```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"
#include <NewPing.h>

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

#define T_C 262
#define T_D 294
#define T_E 330
#define T_F 349
#define T_G 392
#define T_A 440
#define T_B 493

const int Buzzer = 20;

void setup() {
pinMode(Buzzer,OUTPUT);

Wire.begin();
oled.init();
oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
clock_prescale_set(clock_div_1);
#endif
    }

void loop() {

delay(50);
int distance=sonar.ping_cm();

if(distance>5 & distance<11)
    {
tone(Buzzer,T_C);
    }

else if(distance>10 & distance<16)
    {
tone(Buzzer,T_D);
    }

else if(distance>15 & distance<21)
    {
tone(Buzzer,T_E);
    }
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
else if(distance>20 & distance<26)
    {
tone(Buzzer,T_F);
    }

else if(distance>25 & distance<31)
    {
tone(Buzzer,T_G);
    }

else if(distance>30 & distance<36)
    {
tone(Buzzer,T_A);
    }

else if(distance>35 & distance<41)
    {
tone(Buzzer,T_B);
    }

else
    {
noTone(Buzzer);
    }

oled.clearDisplay();
oled.setTextXY(2,4);
oled.putString("Distance: ");
oled.setTextXY(4,6);
String string_distance=String(distance);
oled.putString(string_distance);
oled.setTextXY(4,8);
oled.putString("cm");
    }
```

6.22.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.23 Maze Solver Robot

6.23.1 Giriş

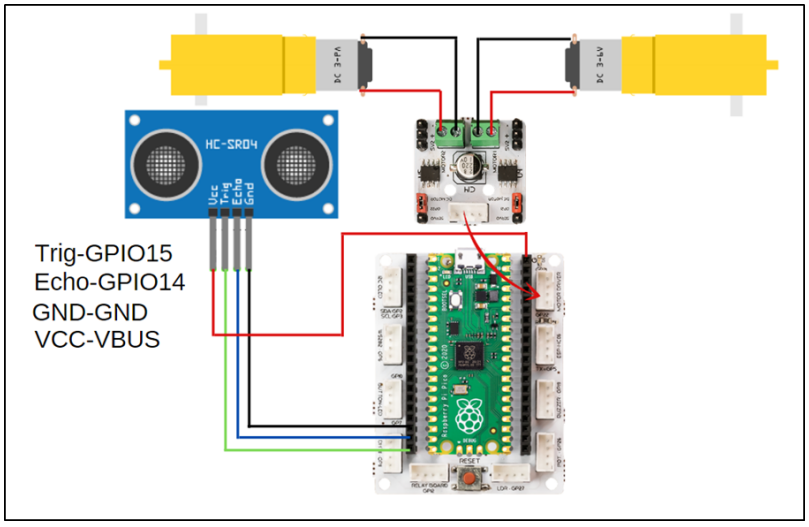
Maze solver robot projesinde setin içerisinden çıkan 2WD robot araba kitini kullanacağız.

6.23.2 Projenin Detayları ve Algoritması

Kodlama eğitimi programlama dillerinin tarihi kadar eskidir. Günümüzde kodlama eğitimini yaygınlaştırmak, heyecanlı ve eğlenceli hale getirmek için farklı ürünler kullanılmaktadır. Bunların başında eğitsel robotlar gelmektedir. Robotları hazırlamak ve kodlamak çocukların `mühendislik` ve `kodlama` becerilerini geliştirmektedir. Kodlama eğitimini yaygınlaştırmak, öğretmen ve öğrencileri teşvik etmek için kurum ve kuruluşlar tarafından robotik yarışmalar düzenlenmektedir. Bu yarışmalardan biri de Labirent Çözen Robot yarışmalarıdır. Bu robotlar önce labirentte dolanarak varış noktasını öğrenir ve başlangıç noktasına geri dönerler. Daha sonra labirente tekrar başladıklarında en kısa yoldan en hızlı şekilde varış noktasına ulaşmaya çalışırlar. Robotlar labirenti öğrenirken mesafe sensörlerinden faydalanırlar. Kızılötesi ya da ultrasonik sensörler bu robotlarda görev almaktadır. Ev ve işyerlerinde kullanılan akıllı robot süpürgeler de labirent çözen robotların algoritmalarına yakın mantıkla çalışmaktadır. Engelleri sürekli kontrol edip haritalayan algoritmaları sayesinde süpürme işini eksiksiz ve çarpmadan yapmaya çalışırlar. Akıllı süpürgelerin çoğunda mesafe ölçme ve engel algılamak için lazer ile yüksek hassasiyetli ölçüm yapan LİDAR ve kızılötesi sensörler görev almaktadır. Bu projede PicoBricks ile labirent çözen robot yarışmalarına hazırlanabileceğiniz basit bir robot yapacağız.

Robotun önündeki mesafeyi algılayarak hareketlerine kendi kendine karar verebilmesi için HC-SR04 ultrasonic mesafe sensörü kullanacağız. Labirent içerisinde robot araba önündeki mesafeyi algılayarak önü boş ise ilerleyecek. Eğer mesafe 5 cm den küçük ise araba sağa dönecek, tekrar mesafeyi ölçecek eğer sağ taraftaki mesafe 5 cm den büyük ise ilerleyerek yoluna devam edecek, küçükse sola dönerek ilerleyecek. Bu şekilde sağa ve sola dönerek labirent içerisinde boş olan yollardan aracın ilerlemesini ve labirentten çıkmasını sağlayacağız.

6.23.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.23.4 Projenin MicroPython Kodu

```
from machine import Pin
from utime import sleep
import utime
#define libraries

trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)
#define sensor pins

m1 = Pin(21, Pin.OUT)
m2 = Pin(22, Pin.OUT)
#define dc motor pins

m1.low()
m2.low()
signaloff = 0
signalon = 0

def getDistance():
trigger.low()
utime.sleep_us(2)
trigger.high()
utime.sleep_us(5)
trigger.low()
while echo.value() == 0:
   signaloff = utime.ticks_us()
while echo.value() == 1:
   signalon = utime.ticks_us()
timepassed = signalon - signaloff
distance = (timepassed * 0.0343) / 2
return distance
#calculate distance

measure=0
while True:

measure=int(getDistance())
print(measure)
if measure>5:
    m1.high()
    m2.high()
    sleep(1) #if the distance is higher than 5, the wheels go straight
else:
    m1.low()
    m2.low()
    sleep(0.5)
    m1.high()
    m2.low()
    sleep(0.5)
    measure=int(getDistance())
    if measure<5:
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
        m1.low()
    m2.low()
    sleep(0.5)
    m1.low()
    m2.high()
    sleep(0.5)
    #If the distance is less than 5, wait, move in any direction; if the distance is␣
↪less than 5, move in the opposite direction
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.23.5 Projenin Arduino C Kodu

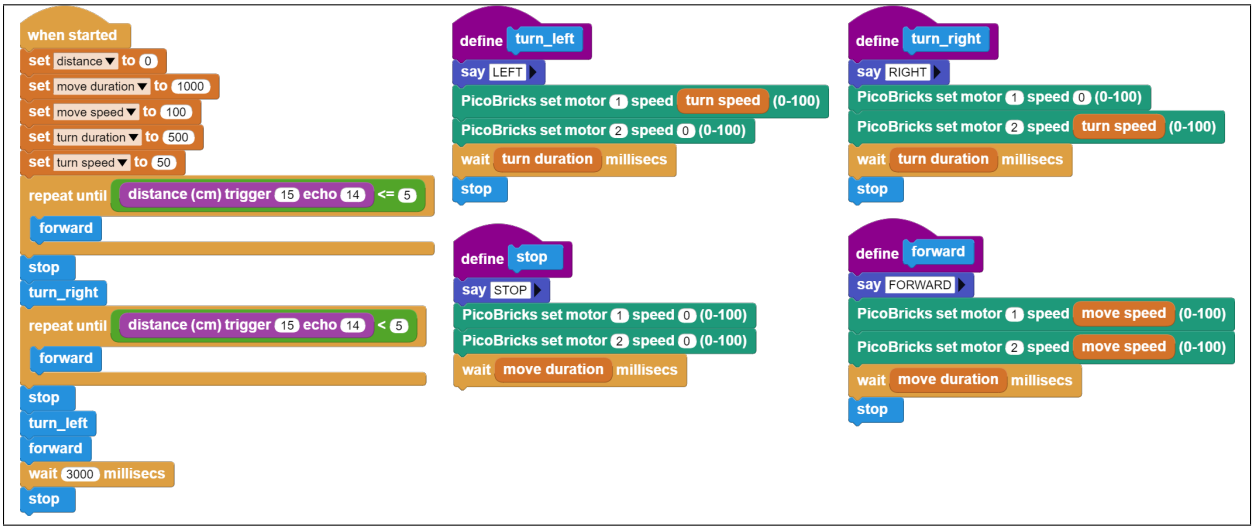
```
#include <NewPing.h>

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400
//define sensor pins

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
pinMode(21,OUTPUT);
pinMode(22,OUTPUT); //define dc motor pins
    }

void loop() {

delay(50);
int distance=sonar.ping_cm();
Forward();

if(distance<5){

Stop();
delay(1000);
Turn_Right();
delay(1000);
int distance=sonar.ping_cm();

if(distance<5){
  Stop();
  delay(1000);
  Turn_Left();
  delay(500);
  // If the distance is less than 5, wait, turn right; if the distance is less than 5␣
↪again, move in the opposite direction
        }
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
    }
}

void Forward(){
digitalWrite(21,HIGH);
digitalWrite(22,HIGH); //if the distance is higher than 5, go straight
}
void Turn_Left(){
digitalWrite(21,LOW);
digitalWrite(22,HIGH); //turn left
}
void Turn_Right(){
digitalWrite(21,HIGH);
digitalWrite(22,LOW);  //turn right
}
void Stop(){
digitalWrite(21,LOW);
digitalWrite(22,LOW); //wait
}
```

6.23.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

6.24 Smart Greenhouse

6.24.1 Giriş

Bu projede PicoBricks ile `IOT teknolojisini` barındıran basit bir sera hazırlayacağız. PicoBricks'i ESP8266 WiFi modülü ile bu serada kullanacağız. Bu sayede serayı internet üzerinden takip edebildiğimiz bir nesne haline getireceğiz.

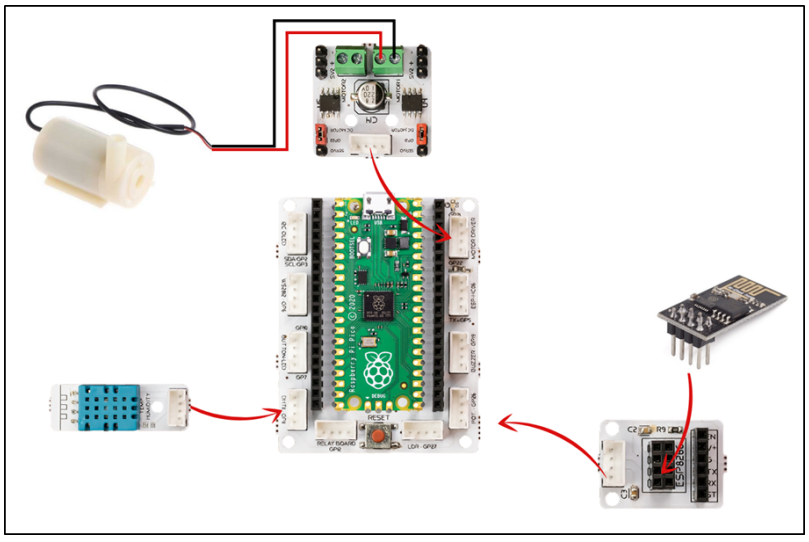
6.24.2 Projenin Detayları ve Algoritması

Küresel ısınmanın etkisiyle iklimlerde meydana gelen hızlı değişimler tarım alanındaki faaliyetlerde verimin düşmesine sebep olmaktadır. 1500'lü yıllarda Daniel Barbaro tarihte bilinen ilk serayı yaptı. Seralar kontrol edilebilen hava, su, ısı ve ışık koşulları sunabilen bitki yetiştirmeye elverişli ortamlardır. Seralarda ısıyı dengelemek için ısıtıcılar , sulama yapmak için elektrikli su motorları, nem oranını ayarlamak ve tozlaşmayı sağlayabilmek için fanlar kullanılmaktadır. Teknolojinin de gelişmesiyle üretici seranın durumunu istediği yerden telefonuyla takip edebilmekte ve yapılması gereken işleri yapabilmektedir. Bu teknolojinin genel adı Nesnelerin İnterneti (IOT) dir.

Seralarda sıcaklık, nem ve oksijen miktarını ölçmek için özel sensörler kullanılır. Ayrıca sulamaya karar vermek için toprak nemini ölçen özel sensörler kullanılmaktadır. Sulama verimliliğini artırmak için ise elektronik kontrollü damla sulama sistemleri kullanılmaktadır.

Hazırlayacağın sera maketinin içinde toprak nemi sensörü, üstten asılı olarak DHT11 sıcaklık ve nem sensörü bulunacaktır. Maketin dışındaki su tankının içine dalgıç pompa konulacak, pompanın ucundan çıkan hortum ise seradaki toprağa gidecektir. Picoboard sera maketinin dışında uygun bir yere yerleştirilecektir. Picobricks başladığında ESP 8266 wifi modülü sayesinde WiFi yayını yapmaya başlar. Aynı ağa bağlı akıllı telefondan Esp8266'nın IP adresini girdiğimizde Serayı kontrol edeceğimiz web sayfası ile karşılaşırız. Burada sıcaklık ve nem değerlerini görebiliriz. Dilersek sulama komutu vererek sulama işlemini başlatabiliriz.

6.24.3 Bağlantı Diyagramı



Picobricks modüllerini herhangi bir kablo bağlantısı olmadan programlayabilir ve çalıştırabilirsiniz. Modülleri karttan ayırarak kullanacaksanız modül bağlantılarını verilen konektör kablolar ile yapmalısınız.

6.24.4 Projenin MicroPython Kodu

```
import utime
import uos
import machine
from machine import Pin, ADC
from picobricks import DHT11
from utime import sleep

dht_sensor = DHT11(Pin(11))
smo_sensor=ADC(27)
m1 = Pin(22, Pin.OUT)
m1.low()

print("Machine: \t" + uos.uname()[4])
print("MicroPython: \t" + uos.uname()[3])

uart0 = machine.UART(0, baudrate=115200)
print(uart0)

def Connect_WiFi(cmd, uart=uart0, timeout=5000):
print("CMD: " + cmd)
uart.write(cmd)
utime.sleep(7.0)
Wait_ESP_Rsp(uart, timeout)
print()

def Rx_ESP_Data():
recv=bytes()
while uart0.any()>0:
    recv+=uart0.read(1)
res=recv.decode('utf-8')
return res

def Send_AT_Cmd(cmd, uart=uart0, timeout=2000):
print("CMD: " + cmd)
uart.write(cmd)
Wait_ESP_Rsp(uart, timeout)
print()

def Wait_ESP_Rsp(uart=uart0, timeout=2000):
prvMills = utime.ticks_ms()
resp = b""
while (utime.ticks_ms()-prvMills)<timeout:
    if uart.any():
        resp = b"".join([resp, uart.read(1)])
print("resp:")
try:
    print(resp.decode())
except UnicodeError:
    print(resp)

Send_AT_Cmd('AT\r\n')          #Test AT startup
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
Send_AT_Cmd('AT+GMR\r\n')      #Check version information
Send_AT_Cmd('AT+CIPSERVER=0\r\n')
Send_AT_Cmd('AT+RST\r\n')      #Check version information
Send_AT_Cmd('AT+RESTORE\r\n')  #Restore Factory Default Settings
Send_AT_Cmd('AT+CWMODE?\r\n')  #Query the WiFi mode
Send_AT_Cmd('AT+CWMODE=1\r\n') #Set the WiFi mode = Station mode
Send_AT_Cmd('AT+CWMODE?\r\n')  #Query the WiFi mode again
Send_AT_Cmd('AT+CWJAP="ID","Password"\r\n', timeout=5000) #Connect to AP
utime.sleep(3.0)
Send_AT_Cmd('AT+CIFSR\r\n')    #Obtain the Local IP Address
utime.sleep(3.0)
Send_AT_Cmd('AT+CIPMUX=1\r\n')
utime.sleep(1.0)
Send_AT_Cmd('AT+CIPSERVER=1,80\r\n')    #Obtain the Local IP Address
utime.sleep(1.0)

while True:
res =""
res=Rx_ESP_Data()
utime.sleep(2.0)
if '+IPD' in res: # if the buffer contains IPD(a connection), then respond with HTML␣
↪handshake
    id_index = res.find('+IPD')

    if '/WATERING' in res:
        print('Irrigation Start')
        m1.high()
        utime.sleep(10)
        m1.low()
        print('Irrigation Finished')
        connection_id =  res[id_index+5]
        print("connectionId:" + connection_id)
        print ('! Incoming connection - sending webpage')
        uart0.write('AT+CIPSEND='+connection_id+',200'+'\r\n')
        utime.sleep(1.0)
        uart0.write('HTTP/1.1 200 OK'+'\r\n')
        uart0.write('Content-Type: text/html'+'\r\n')
        uart0.write('Connection: close'+'\r\n')
        uart0.write(''+'\r\n')
        uart0.write('<!DOCTYPE HTML>'+'\r\n')
        uart0.write('<html>'+'\r\n')
        uart0.write('<body><center><H1>CONNECTED...<br/></H1></center>'+'\r\n')
        uart0.write('<body><center><H1>Irrigation Complete.<br/></H1></center>'+'\r\n')
        uart0.write('</body></html>'+'\r\n')
    elif '/SERA' in res:
        #sleep(1) # It was used for DHT11 to measure.
        dht_sensor.measure() # Use the sleep() command before this line.
        temp=dht_sensor.temperature
        hum=dht_sensor.humidity
        smo=round((smo_sensor.read_u16()/65535)*100)
        sendStr="\"TEMP\":{}, \"Humidity\":{}, \"S.Moisture\":{}%".format(temp,hum,smo)
        sendText="{"+sendStr+"}"
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
        strLen=46+len(sendText)
        connection_id =  res[id_index+5]
        print("connectionId:" + connection_id)
        print ('! Incoming connection - sending webpage')
        atCmd="AT+CIPSEND="+connection_id+","+str(strLen)
        uart0.write(atCmd+'\r\n')
        utime.sleep(1.0)
        uart0.write('HTTP/1.1 200 OK'+'\r\n')
        uart0.write('Content-Type: text/html'+'\r\n')
        uart0.write(''+'\r\n')
        uart0.write(sendText+'\r\n')

    elif '/' in res:

        print("resp:")
        print(res)
        connection_id =  res[id_index+5]
        print("connectionId:" + connection_id)
        print ('! Incoming connection - sending webpage')
        uart0.write('AT+CIPSEND='+connection_id+',200'+'\r\n')
        utime.sleep(3.0)
        uart0.write('HTTP/1.1 200 OK'+'\r\n')
        uart0.write('Content-Type: text/html'+'\r\n')
        uart0.write('Connection: close'+'\r\n')
        uart0.write(''+'\r\n')
        uart0.write('<!DOCTYPE HTML>'+'\r\n')
        uart0.write('<html>'+'\r\n')
        uart0.write('<body><center><H1>CONNECTED.<br/></H1></center>'+'\r\n')
        uart0.write('<center><h4>INFO:Get Sensor Data</br>WATERING:Run Water Pump</h4></
↪center>'+'\r\n')
        uart0.write('</body></html>'+'\r\n')
    utime.sleep(4.0)
    Send_AT_Cmd('AT+CIPCLOSE='+ connection_id+'\r\n') # once file sent, close connection
    utime.sleep(3.0)
    recv_buf="" #reset buffer
    print ('Waiting For connection...')
```

Tüyo: Eğer kodunuzun adını main.py olarak kaydederseniz, kodunuz her BOOT yaptığınızda çalışacaktır.

6.24.5 Projenin Arduino C Kodu

```
#include <DHT.h>
#define RX 0
#define TX 1

#define LIMIT_TEMPERATURE     30
#define DHTPIN                11
#define DHTTYPE               DHT11
#define smo_sensor            27
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
#define motor                 22
#define DEBUG true

DHT dht(DHTPIN, DHTTYPE);
int connectionId;

void setup() {
Serial1.begin(115200);
dht.begin();
pinMode(smo_sensor, INPUT);
pinMode(motor, OUTPUT);

sendData("AT+RST\r\n", 2000, DEBUG); // reset module
sendData("AT+GMR\r\n", 1000, DEBUG); // configure as access point
sendData("AT+CIPSERVER=0\r\n", 1000, DEBUG); // configure as access point
sendData("AT+RST\r\n", 1000, DEBUG); // configure as access point
sendData("AT+RESTORE\r\n", 1000, DEBUG); // configure as access point
sendData("AT+CWMODE?\r\n", 1000, DEBUG); // configure as access point
sendData("AT+CWMODE=1\r\n", 1000, DEBUG); // configure as access point
sendData("AT+CWMODE?\r\n", 1000, DEBUG); // configure as access point
sendData("AT+CWJAP=\"WIFI_ID\",\"WIFI_PASSWORD\"\r\n", 5000, DEBUG); // ADD YOUR OWN␣
↪WIFI ID AND PASSWORD
delay(3000);
sendData("AT+CIFSR\r\n", 1000, DEBUG); // get ip address
delay(3000);
sendData("AT+CIPMUX=1\r\n", 1000, DEBUG); // configure for multiple connections
delay(1000);
sendData("AT+CIPSERVER=1,80\r\n", 1000, DEBUG); // turn on server on port 80
delay(1000);
    }

void loop() {
if (Serial1.find("+IPD,")) {
delay(300);
connectionId = Serial1.read() - 48;
String serialIncoming = Serial1.readStringUntil('\r');
Serial.print("SERIAL_INCOMING:");
Serial.println(serialIncoming);

if (serialIncoming.indexOf("/WATERING") > 0) {
  Serial.println("Irrigation Start");
  digitalWrite(motor, HIGH);
  delay(1000); // 10 sec.
  digitalWrite(motor, LOW);
  Serial.println("Irrigation Finished");
  Serial.println("! Incoming connection - sending WATERING webpage");
  String html = "";
  html += "<html>";
  html += "<body><center><H1>Irrigation Complete.<br/></H1></center>";
  html += "</body></html>";
  espsend(html);
}
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

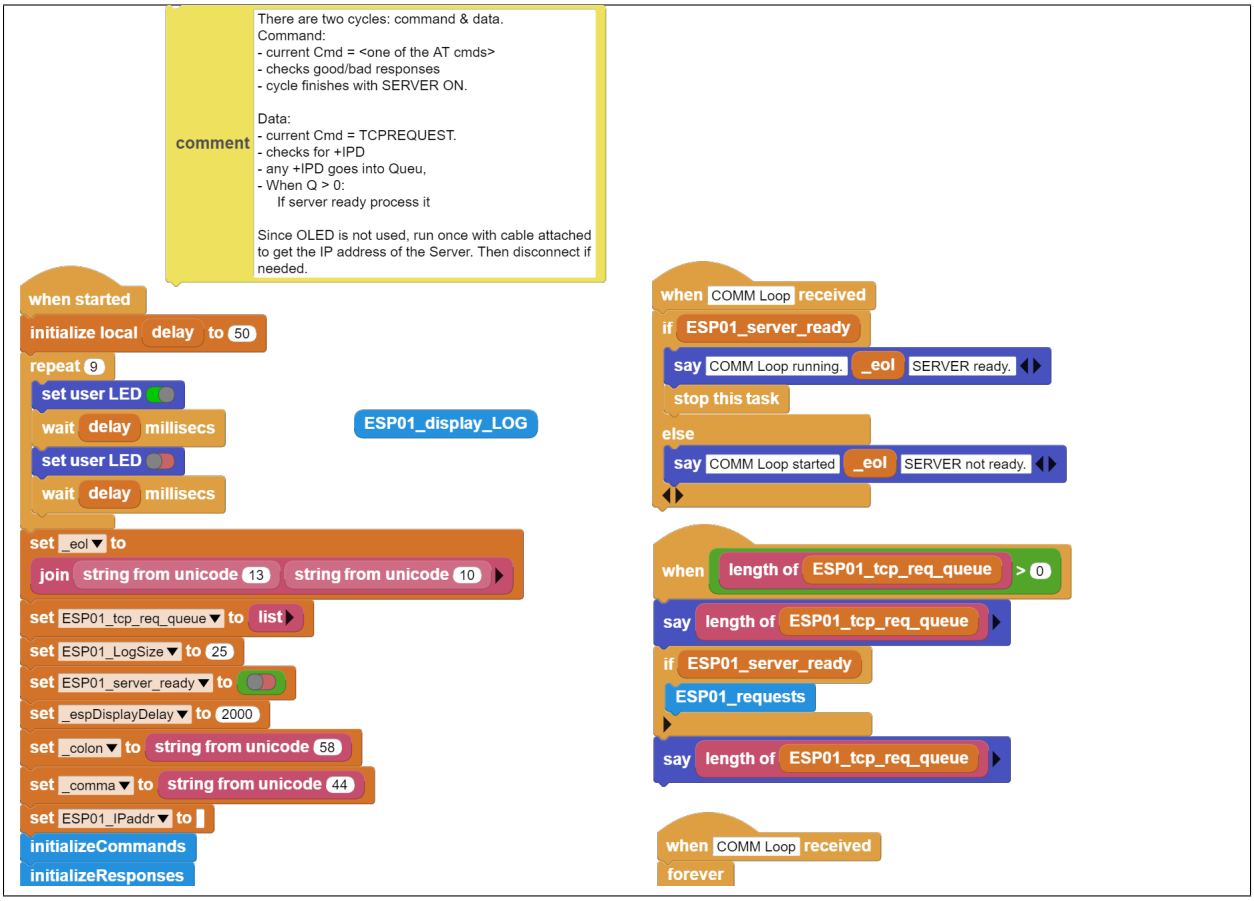
```
if (serialIncoming.indexOf("/SERA") > 0) {
  delay(300);

  float smo = analogRead(smo_sensor);
  float smopercent = (460-smo)*100.0/115.0 ; //min ve max değerleri değişken.
  Serial.print("SMO: %");
  Serial.println(smo);

  float temperature = dht.readTemperature();
  Serial.print("Temp: ");
  Serial.println(temperature);

  float humidity = dht.readHumidity();
  Serial.print("Hum: ");
  Serial.println(humidity);

  Serial.println("! Incoming connection - sending SERA webpage");
  String html = "";
  html += "<html>";
  html += "<body><center><H1>TEMPERATURE<br/></H1></center>";
  html += "<center><H2>";
  html += (String)temperature;
  html += " C<br/></H2></center>";

  html += "<body><center><H1>HUMIDITY<br/></H1></center>";
  html += "<center><H2>";
  html += (String)humidity;
  html += "%<br/></H2></center>";

  html += "<body><center><H1>SMO<br/></H1></center>";
  html += "<center><H2>";
  html += (String)smopercent;
  html += "%<br/></H2></center>";

  html += "</body></html>";
  espsend(html);
}
else
  Serial.println("! Incoming connection - sending MAIN webpage");
String html = "";
html += "<html>";
html += "<body><center><H1>CONNECTED.<br/></H1></center>";
html += "<center><a href='/SERA'><h4>INFO:Get Sensor Data</a></br><a href='/WATERING'>
↪WATERING:Run Water Pump</a></h4></center>";
html += "</body></html>";
espsend(html);
String closeCommand = "AT+CIPCLOSE=";  ////////////////close the socket connection////
↪esp command
closeCommand += connectionId; // append connection id
closeCommand += "\r\n";
sendData(closeCommand, 3000, DEBUG);
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
    }

    }
    ////////////////////////////sends data from ESP to webpage/////////////////////////
↪//

void espsend(String d)
    {
String cipSend = " AT+CIPSEND=";
cipSend += connectionId;
cipSend += ",";
cipSend += d.length();
cipSend += "\r\n";
sendData(cipSend, 1000, DEBUG);
sendData(d, 1000, DEBUG);
    }

    //////////////gets the data from esp and displays in serial monitor/////////////////
↪/////

String sendData(String command, const int timeout, boolean debug)
    {
String response = "";
Serial1.print(command);
long int time = millis();
while ( (time + timeout) > millis())
    {
while (Serial1.available())
{
  char c = Serial1.read(); // read the next character.
  response += c;
}
}

if (debug)
{
Serial.print(response); //displays the esp response messages in arduino Serial monitor
}
return response;
}
```

6.24.6 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görseli MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

BÖLÜM 7

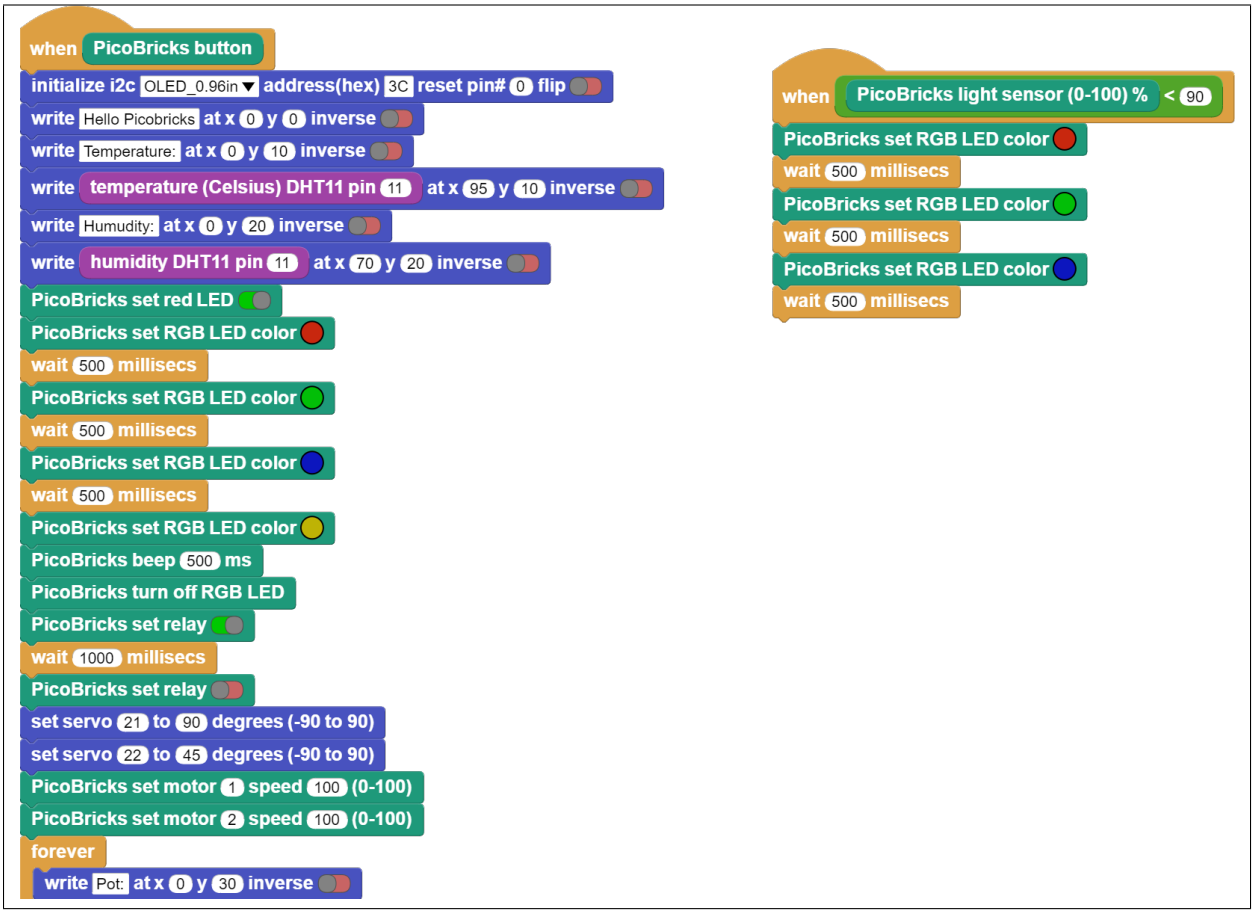
Picobricks Eğitim

PicoBricks eğitim içeriğinde bulabileceğiniz projelere buradan kolayca ulaşabilirsiniz. Picobricks ile kodlama öğrenme maceranıza buradaki projelerle hızlı bir şekilde başlayabilir ve kendinizi geliştirebilirsiniz.

7.1 MicroBlocks Test Kodu

Bu test kodu ile PicoBricks üzerindeki tüm modülleri test edebilir ve nasıl çalıştığını öğrenebilirsiniz.

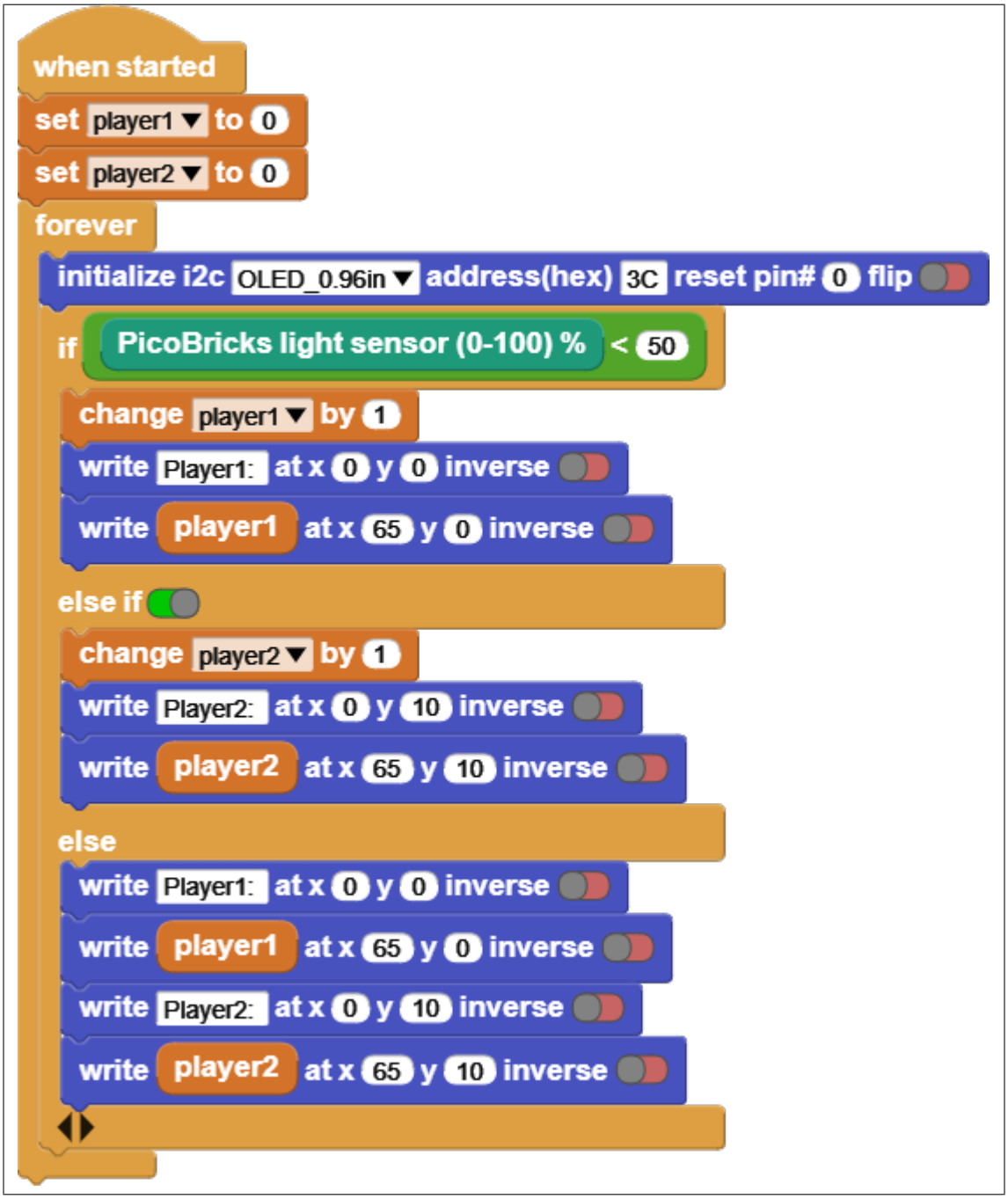
7.1.1 MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.2 2 Player Score

7.2.1 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

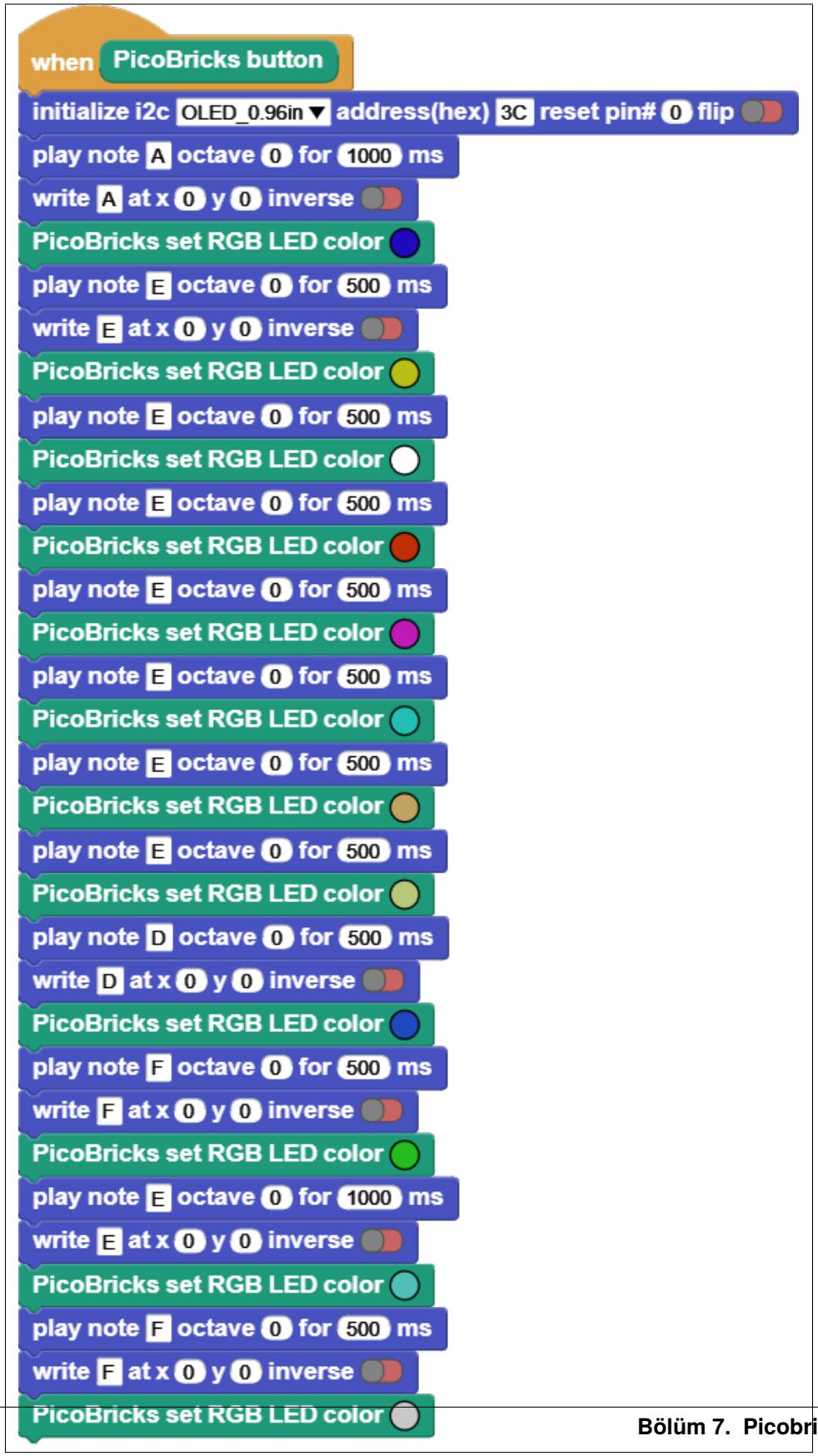
7.3 Colourful Music

7.3.1 Projenin Detayları ve Algoritması

Öğrendiğimiz bilgileri kullanarak eğlenceli bir proje yapalım. Projemizde OLED ekran, buton, buzzer ve RGB LED kullanacağız.

Butona basıldığında buzzer ile aşağıda verilen notalar çalınacak, aynı anda hangi notanın çalıştığı OLED ekrana yazılacak ve notalarla eş zamanlı olarak RGB LED'in rengi değişecektir.

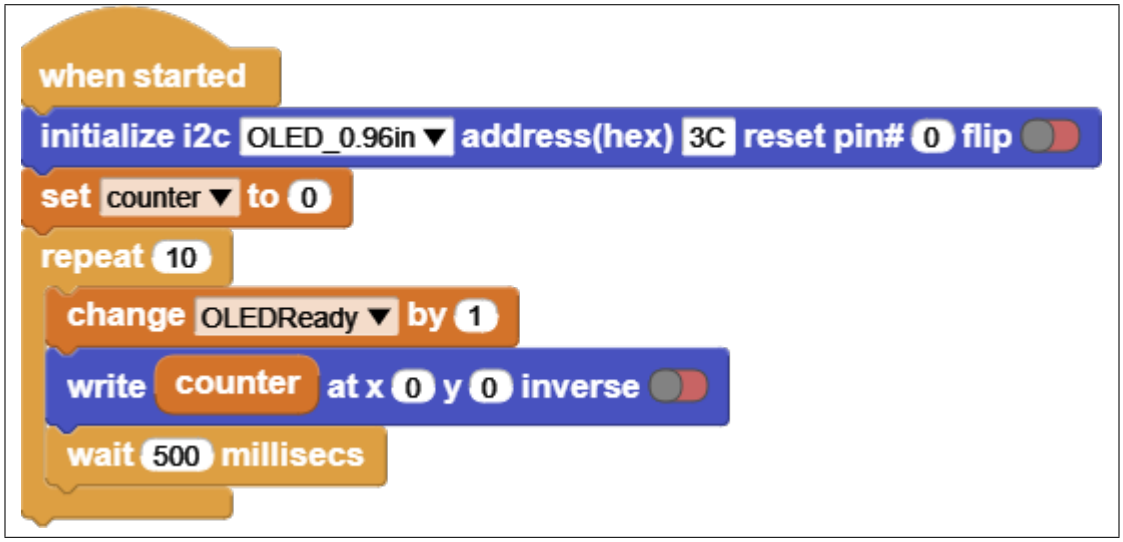
7.3.2 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.4 Counter Project

7.4.1 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.5 Propeller Button

7.5.1 Projenin MicroBlocks Kodu

Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.6 Rock-Paper-Scissor

7.6.1 Giriş

Bu projemizde, PicoBricks'i kullanarak basit ve elektronik bir taş-kağıt-makas oyunu yapacağız.

7.6.2 Projenin Detayları ve Algoritması

Bu proje sadece PicoBricks gerektirdiği için, başlangıç seviyesindeki kullanıcılar için harika!

Kurallar:

- Makas, kağıdı keser.
- Kağıt, taşı sarar.
- Taş, makası kırar.

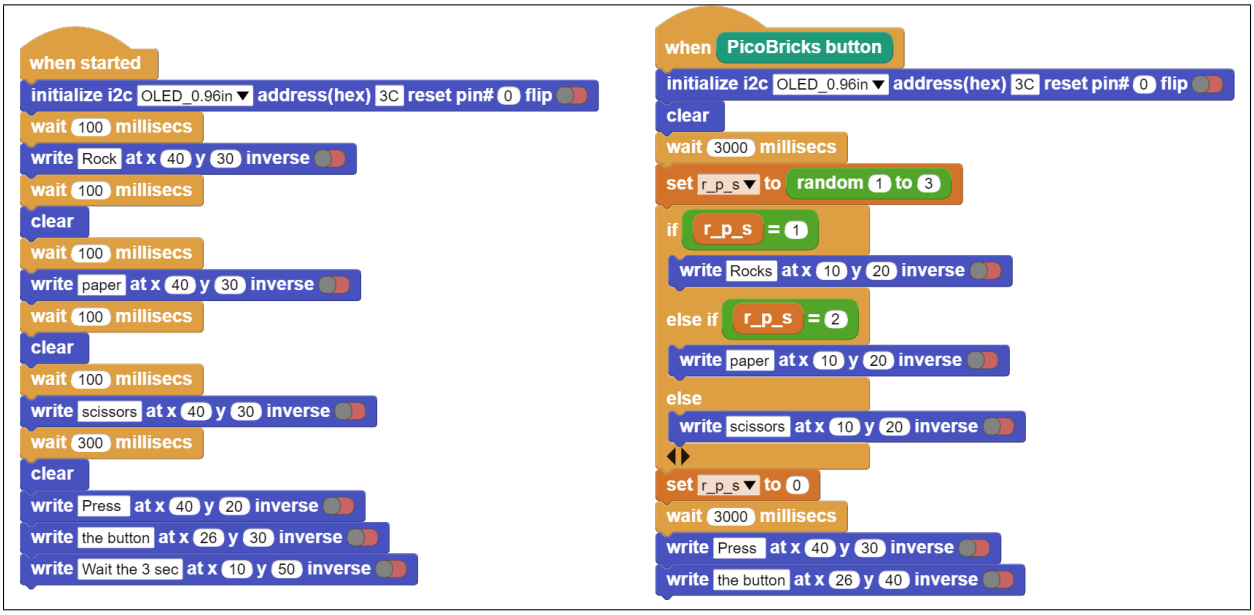
Bu proje nasıl daha iyi yapılabilir?

Bir düşünelim:

- Daha büyük bir buton eklenebilir
- Daha büyük bir OLED ekran eklenebilir
- Yazı yerine görseller kullanılabilir

Hayal gücünü kullan…

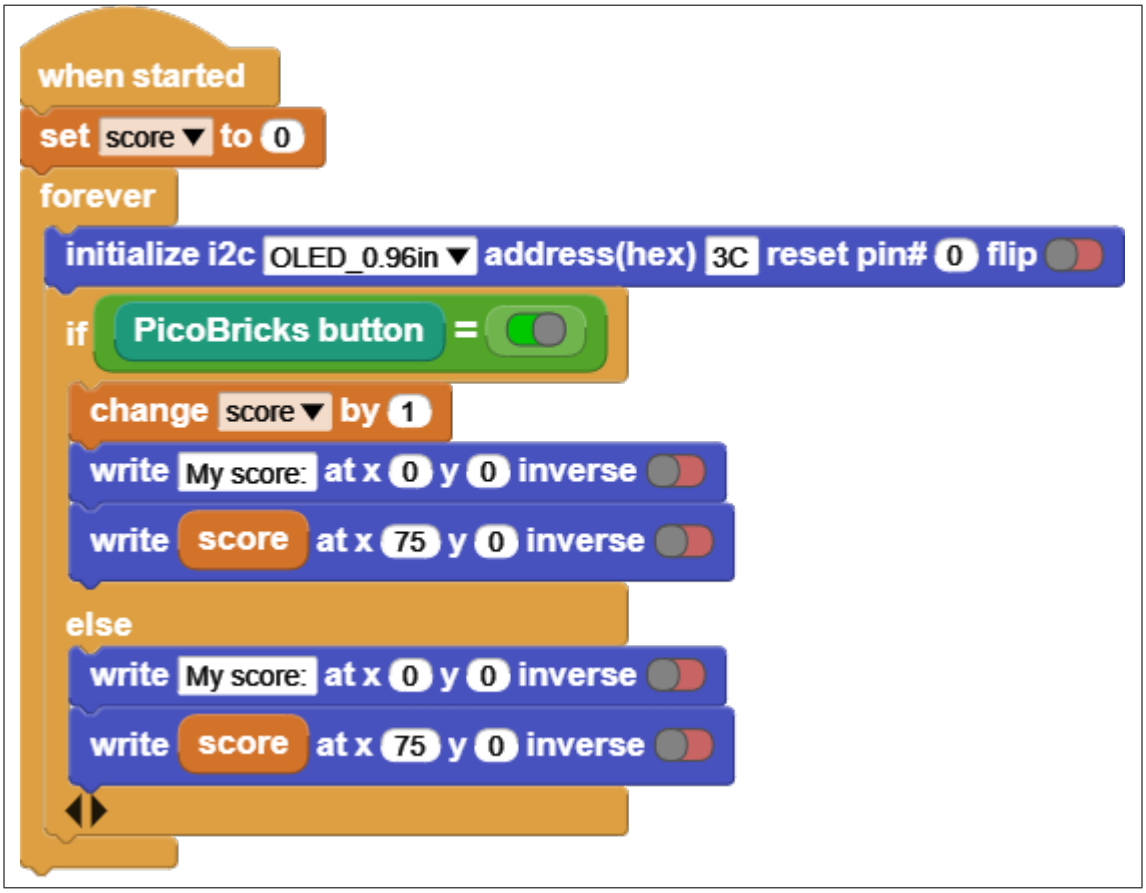
7.6.3 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.7 Score Game

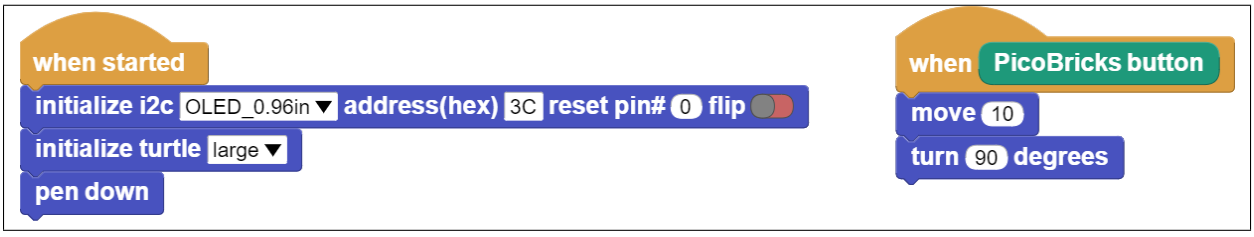
7.7.1 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.8 Turtle Project

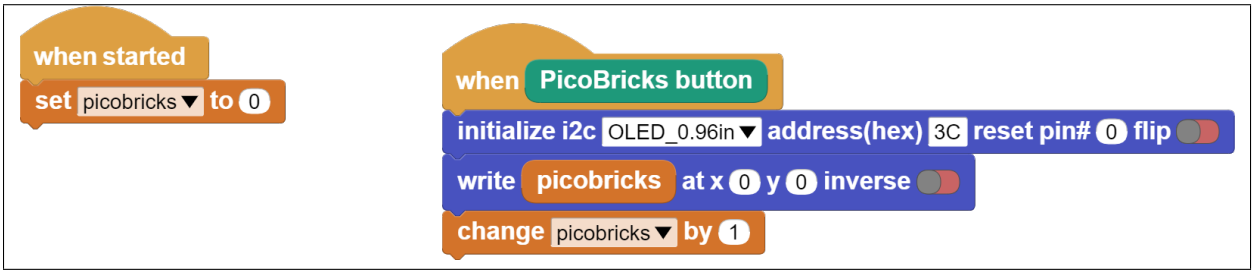
7.8.1 Projenin MicroBlocks Kodu



Not: MicroBlocks ile kodlama yapmak için yukarıdaki görüntüyü MicroBlocks Run sekmesine sürükleyip bırakmanız yeterlidir.

7.9 Variable Button

7.9.1 Coding the Project with MicroBlocks



Not: To code with MicroBlocks, simply drag and drop the image above to the MicroBlocks Run tab.

BÖLÜM 8

Bootloader

8.1 BOOTSEL

Pico'nun BOOTSEL modu, RP2040 içindeki salt okunur bellekte bulunur ve yanlışlıkla üzerine yazılamaz. Ne olursa olsun, Pico'nuzu prize taktığınızda BOOTSEL düğmesini basılı tutarsanız, üzerine yeni bir UF2 dosyası sürükleyebileceğiniz bir sürücü olarak görünecektir. Kartı programlamanın başka bir yolu yoktur. Ancak, Flash belleğinizin boş olduğundan emin olmak isteyebileceğiniz bazı durumlar vardır. Bunu, toplu depolama modundayken özel bir UF2 ikili dosyasını Pico'nuza sürükleyip bırakarak yapabilirsiniz.

- Raspberry Pi web sitesinden MicroPython `UF2 dosyasını` indirin
- Pico'nuzdaki `BOOTSEL` düğmesini basılı tutun ve bilgisayarınızın USB portuna takın.
- Explorer'ı açın ve diğer herhangi bir sabit sürücüde yaptığınız gibi `RPI-RP2 dizinini` açın
- `UF2 dosyasını` sürükleyip `RPI-RP2 dizinine` bırakın

8.2 Flash Belleği Sıfırlama

Raspberry Pi Pico harika bir teknoloji ürünü ama bir kusuru var: `sıfırlama düğmesi yok`. Bu ihmal ne kadar önemli? Bazen kodumuz kaybolabilir veya Pico'muza yeni aygıt yazılımı yüklememiz gerekir.

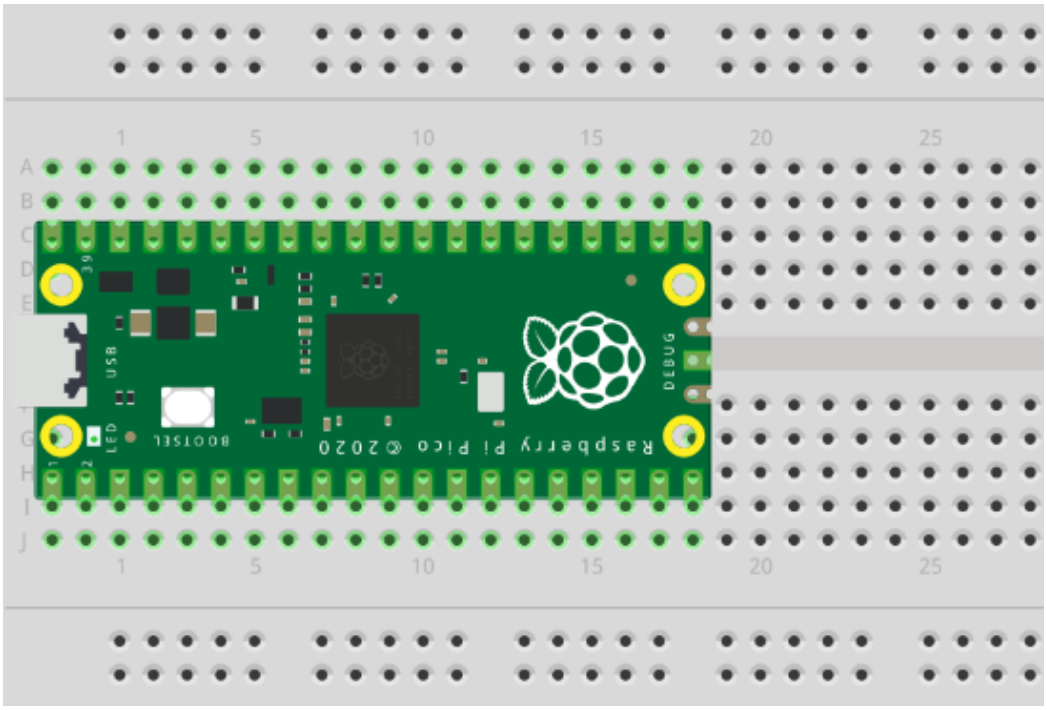
Bu olduğunda, sıfırlamak için Pico'nun fişini çekip tekrar takmamız gerekir. Sınırlı sayıda yerleştirme için derecelendirilen mekanik bir bağlantı olan mikro USB kablosunu çok fazla çıkarırsak, onu yıpratabiliriz. Açma/kapama butonlarıyla elektrikli bir USB hub'ına bağlı Pico'muz varsa, bunun üzerindeki düğmeye basabiliriz, ama ya basmazsak.

Tüyo: Çok az ekipman ve sıfır kodla, Pico'muzu bir sonraki projeye hazır hale getirmek için basit bir buton oluşturabiliriz.

8.3 Sıfırlama Butonu Projesi

Gerekli olanlar
Raspberry Pi Pico
2 x Erkek-Erkek Jumper Kablo
Breadboard
Buton

1) Raspberry Pi Pico'yu mikro USB bağlantı noktası `breadboardun ucunda` asılı kalacak şekilde devre tahtasına yerleştirin.

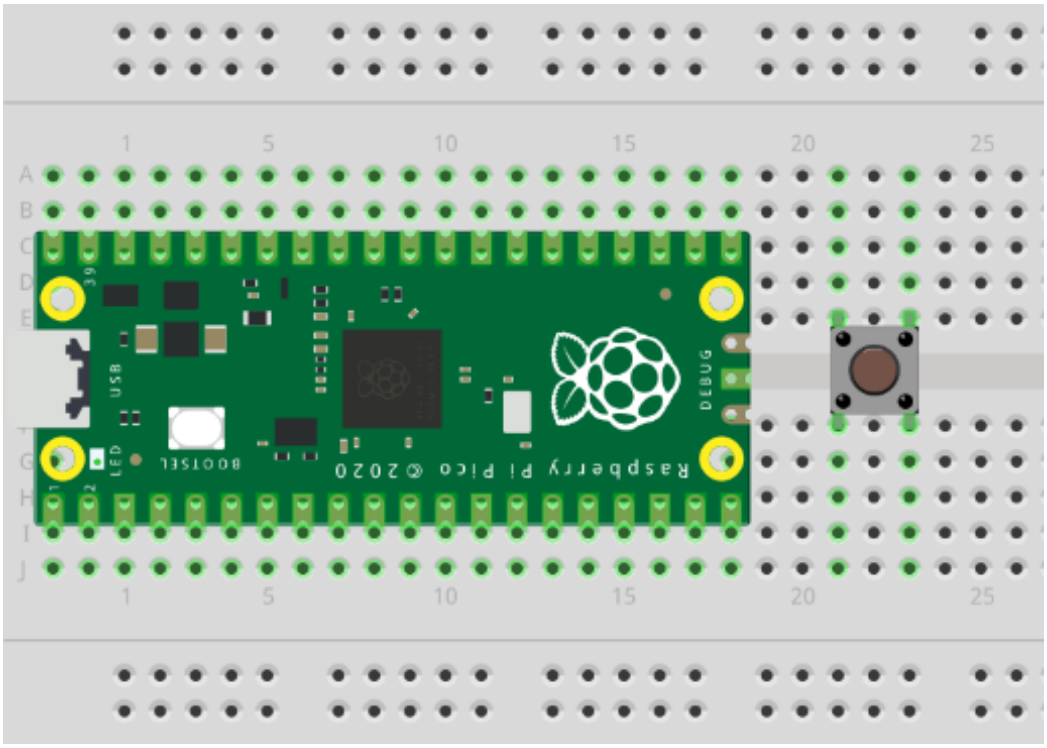


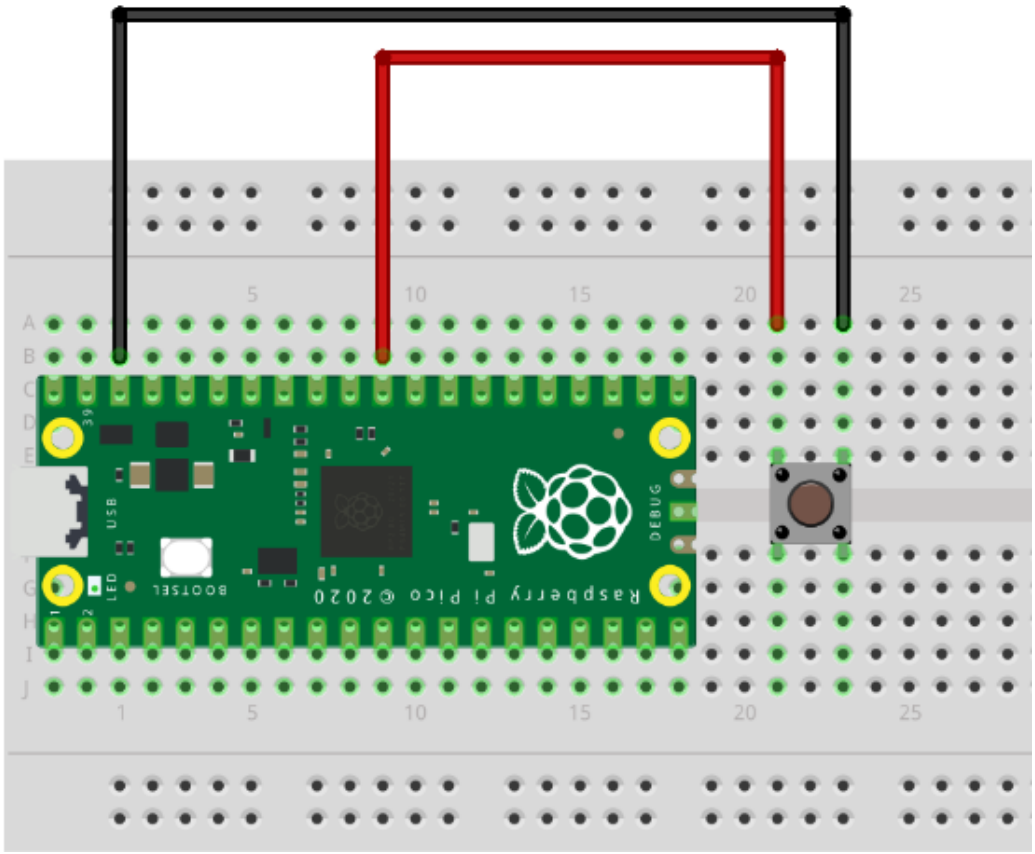
2) Görselde gördüğünüz gibi bir `buton` yerleştirin.

3) Jumper kablolardan birini GND pinine ve butonun sağ ayağına, diğerini RUN pinine ve butonun sol ayağına bağlayın.

Not: Sıfırlama butonumuz kullanıma hazır.

Tüyo: Daha fazla bilgi için Raspberry Pi Websitesini ziyaret edebilirsiniz.


1 5 10 15 20 25
A
B
C
D
E
F
G
H
I
J
USB
LED
BOOTSEL
DEBUG
Raspberry Pi Pico ©2020


5 10 15 20 25
A
B
C
D
E
F
G
H
I
J
USB
LED
BOOTSEL
DEBUG
Raspberry Pi Pico ©2020

BÖLÜM 9

MicroBlocks Kılavuzu

9.1 Blokların Özeti

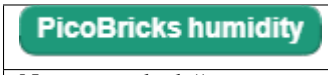
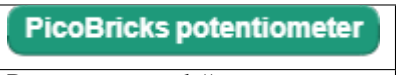
Her blok için kısa bir açıklama girişi ve ayrıntılı bir blok ve komponent açıklaması vardır. Detaylara ve örnek kodlara ulaşmak için kısa açıklama tablosundaki blok resimlerine tıklayabilirsiniz.

Blokların nasıl kullanılacağına dair örnek kodlar verilmiştir. Bunları denemek için tek yapmanız gereken MicroBlocks IDE'de bir tarayıcı oturumu açmak ve bunları düzenleyici programlama alanına sürükleyip bırakmaktır. Ardından üzerlerine tıklayıp sonuçları görebilirsiniz.

Tüyo: Aşağıdaki örnek kodlardan herhangi birini test etmek için, bunları MicroBlocks IDE'ye sürükleyip bırakmanız yeterlidir.

	
RGB LED'in rengini belirler	Santigrat(°C) cinsinden sıcaklığı verir

	
Hoparlörden beep sesi çıkarır	Butonun durumunu 1/0 olarak verir

	
Nemin yüzde değerini verir	Potansiyometre değerini verir

	
RGB LED için rastgele renk belirler	Kırmızı LED'i 1/0 olarak ayarlar

PicoBricks set motor 1 speed 100 (0-100)	PicoBricks set relay
DC motorları kontrol eder	Röleyi 1/0 olarak ayarlar

PicoBricks color r 0 g 0 b 0 (0-255)	PicoBricks light sensor (0-100) %
RGB renk değerini verir	Işık seviyesini yüzde değeri olarak verir

PicoBricks turn off RGB LED
RGB LED'i kapatır

9.2 Kütüphane Blokları ile Çalışmak

Kitaplık, kullanıcının PicoBricks Board ve üzerindeki modüller üzerinde tam denetime sahip olmasına izin veren bir dizi basit hizmetten oluşur.

PicoBricks Kitaplığı'nda iki farklı blok şekli vardır:

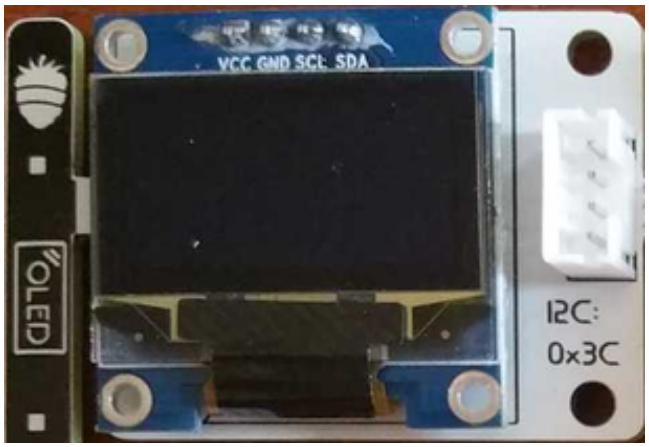
- oval: Bir tür bilgiyi geri döndüren `reporter bloklarıdır`. Kullanıcı normalde bunları bir proje değişkenine atar veya diğer blokların uygun bir giriş yuvasında kullanır. Dönüş bilgisi türü, desteklenen veri türlerinden herhangi biri olabilir.
- dikdörtgen: Programlanmış bir işlevi yerine getiren ve herhangi bir bilgi döndürmeyen `komut bloklarıdır`.

9.3 Kütüphane Bileşenleri Üzerine Notlar

PicoBricks kartı, önceden monte edilmiş on bileşen içerir. Ancak Kütüphane'de yalnızca yedi tanesi için blok vardır; OLED ekranı, Motor Kontrolü ve Kablosuz İletişim bileşenleri kütüphane tarafından `doğrudan kapsanmaz`.

Bu bölümde, bu bileşenlerden nasıl yararlanılabileceğini ve bunlarla ilgili bazı bilgileri açıklayacağız.

- OLED ekran modülü, OLED Graphics adı verilen MicroBlocks grafik kütüphanesi kullanılarak programlanmıştır. IDE'de `Libraries+ / Graphics / OLED Graphics.ubl` altında bulunur.



Not: Kitaplık bloklarının ve işlemlerinin ayrıntıları için, OLED Kütüphane bölümüne bakın.

Tüyo: Kütüphanenin gelişmiş kullanımını ve veri aktarımını açıklayan bir proje için lütfen SNAP to MicroBlocks bölümüne bakın.

- Motor modülü, Servo Motorları ve DC Motorları kontrol etmek için kullanılır. DC motor direkt olarak kütüphane tarafından desteklenir. Ancak bir uyarı var: motor bağlantıları yalnızca tel kablo üzerinden yapıldığından, DC motorların yönünü programlı olarak tersine çevirmek mümkün değildir. DC motorların yönünü değiştirmenin tek yolu, kabloların bağlanma şeklini değiştirmektir.

Uyarı: Servo motorların programlanması MicroBlocks servokütüphanesi tarafından desteklenmektedir. Servo motor kütüphanesi, IDE içinde Libraries+ / Servo.ubl'de yer almaktadır.

Not: DC Motor ve Servo Motor kontrolünün detaylı projesini incelemek için PicoBricks Servo ve DC Motor Kontrol bölümüne bakınız.

- Kablosuz Haberleşme Modülü, PicoBricks'in bir WiFi modülü veya bir Bluetooth modülü kullanarak diğer ortamlarla iletişim kurmasını sağlar ve Pico'nun Seri IO bağlantı noktalarına bağlanır.

9.4 MicroBlocks ile Projeler

MicroBlocks, sizi keşfetmeye davet eden canlı bir programlama sistemidir. Metnin görünmesini görmek için MicroBlocks IDE'de bir bloğa tıklayın.

Hem yeni başlayanlar hem de uzmanlar, MicroBlocks'un PicoBricks'in her yönünü keşfeden harika bir araç olduğunu göreceklerdir.

BÖLÜM 10

Datasheet

- ESP8266
- Raspberry-Pi-Pico
- OLED-Screen
- Buzzer
- DHT11
- Relay
- WS2812
- LDR
- L9110

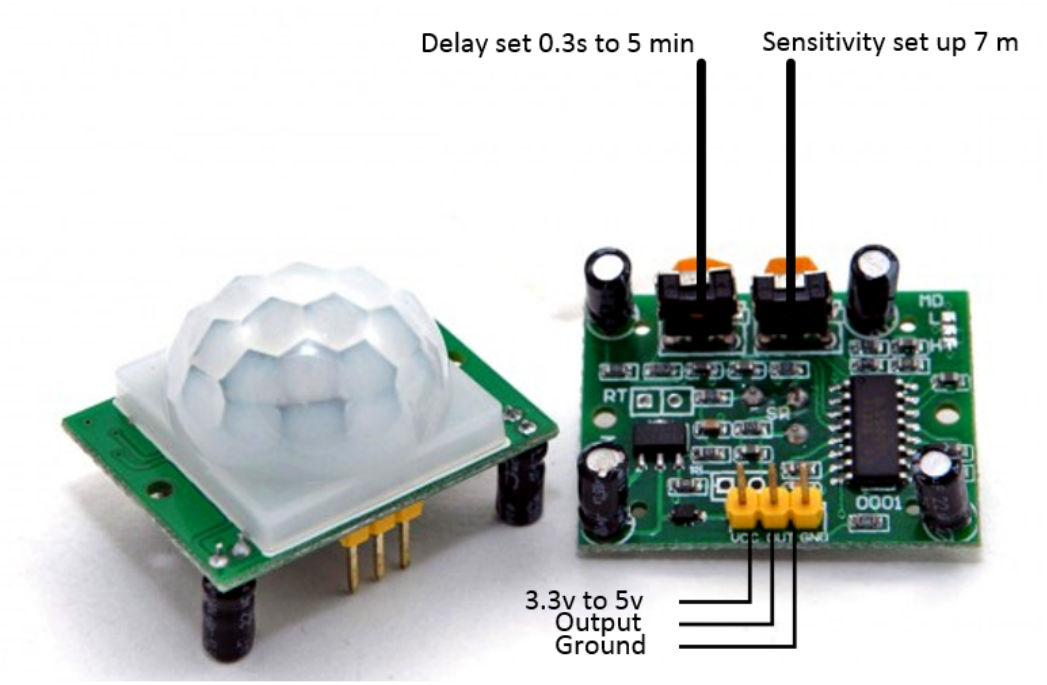
BÖLÜM 11

Sorun Giderme

11.1 PIR sensörü geç tepki veriyor

PIR sensörünüz çok geç tepki veriyorsa veya dokunmadan tepki vermiyorsa PIR sensörünüzün hassasiyetini ayarlamalısınız. PIR sensörünüzün hassasiyetini ayarlamak için, aşağıdaki adımları izleyebilirsiniz.

- Bu ayar, PIR'ınız çok hassassa veya yeterince hassas değilse gereklidir (saat yönünde çevirmek daha hassas hale getirir). Ancak, devre doğru şekilde bağlandığında herhangi bir algılama yoksa, sensörünüz arızalıdır.



11.2 `no module named 'picobricks'` hatasını almak

Kodunuzu çalıştırdığınızda `no module named 'picobricks'` hatasını alıyorsanız, `picobricks.py` kütüphanesini yüklemelisiniz.

- `picobricks.py` kütüphanesini yüklemek için, aşağıdaki adımları takip edebilirsiniz.
- 1. Thonny'yi açın.
- 2. `BOOTSEL` butonuna basarak PicoBricks'i bilgiarınıza bağlayınız.
- 3. Buradan, `picobricks.py` kütüphanesini kopyalayın ve Thonny kodlama alanına yapıştırın.
- 4. Son olarak,``ctrl+s`` tuşlarına basarak Raspberry Pi Pico kartının içine kaydet

---

Not: Bu içerik yaşadığın problemi çözmek için yeterli değilse, `support@robotistan.com` adresinden bize mail gönderebilirsin.

BÖLÜM 12

Haklar ve Lisanslar

12.1 Apache License 2.0

Buradaki tüm orijinal kaynak kodları Telif Hakkı (C) 2022 PicoBricks/Robotistan'a aittir. Bu kaynak kodu, lisans dosyasında açıklandığı gibi Apache License 2.0 altında lisanslanmıştır.

Apache Lisansı, Sürüm 2.0 ("Lisans"); bu dosyayı Lisans ile uyum dışında kullanamazsınız. Lisansın bir kopyasını şu adresten edinebilirsiniz:

http://www.apache.org/licenses/LICENSE-2.0

Yürürlükteki yasa gerektirmedikçe veya yazılı olarak kabul edilmedikçe, Lisans kapsamında dağıtılan yazılım, açık veya zımni olarak HİÇBİR TÜRDE GARANTİ VEYA KOŞUL OLMAKSIZIN, "OLDUĞU GİBİ" ESASINDA dağıtılır. Lisans kapsamındaki izinleri ve sınırlamaları yöneten belirli dil için Lisansa bakın.

Not: Lisanslar hakkında daha fazla bilgi için PicoBricks GitHub adresine bakabilirsiniz.

12.2 Uluslararası Attribution-NonCommercial-ShareAlike 4.0

Creative Commons Corporation ("Creative Commons") bir hukuk firması değildir ve yasal hizmetler veya yasal tavsiye sağlamaz. Creative Commons kamu lisanslarının dağıtımı, bir avukat-müvekkil ilişkisi veya başka bir ilişki oluşturmaz. Creative Commons, lisanslarını ve ilgili bilgileri "olduğu gibi" sunar. Creative Commons, lisansları, bunların hüküm ve koşulları kapsamında lisanslanan herhangi bir materyal veya ilgili herhangi bir bilgi ile ilgili hiçbir garanti vermez. Creative Commons, kullanımlarından kaynaklanan zararlar için mümkün olan en geniş ölçüde sorumluluk kabul etmez.

Not: Lisanslar hakkında daha fazla bilgi için PicoBricks GitHub adresine bakabilirsiniz.