به نام خدا

**آزمایشگاه شماره 2 کار با Arduino محمد شریفی صادقی و امیرحسین هدایتی**

**چکیده**

در این آزمایش روش دیگر ایجاد تاخیر در برنامه مورد بحث قرار می­گیرد و برنامه LED چشمک زن آزمایش قبلی با این روش پیاده­سازی می­شود در ادامه 2 برنامه چشمک زن در محیط شبیه­سازی Proteus توسعه داده می­شود و در انتها مفهوم ریسمان یا Thread در Arduino مورد بحث قرار می­گیرد.

**مقدمه و معرفی**

در بخش اول راه حل جایگزین تابع delay در Arduino مورد بحث قرار می­گیرد. در بخش دوم برنامه چشمک زن آزمایش اول با راه حل بخش اول دوباره پیاده سازی می­شود. در بخش سوم 8 LED به وسیله Push Button و برد Arduino خاموش و روشن می­شود. در بخش نهایی برنامه چشمک زن به جای استفاده از تاخیر با استفاده از ریسمان یا Thread پیاده سازی می­شود.

**روش‌ها و تجهیزات مورد استفاده**

* محیط توسعه‌ی Arduino (IDE)
* برد Arduino UNO یا محیط شبیه‌سازی Proteus

**روش آزمایش**

برای انجام این آزمایش کافیست برنامه نوشته شده در Arduino را Compile کرده

و فایل hex تولید شده را به برد Arduino داخل محیط Proteus بدهیم یا با کمک

خود ArduinoIDE به برد فیزیکی Arduino بدهیم

**نتایج**

در بخش اول می­خواهیم جایگزینی برای تابع delay پیدا کنیم زیراکه این تابع کل پردازنده را برای مدتی از کار می­اندازد اما ما می­توانیم در این زمان بهره های دیگری از پردازنده ببریم یکی از راه حل ها استفاده از تابع millis می­باشد که مقدار میلی ثانیه گذشته از شروع برنامه را نشان می­دهد و با اندازه گیری لحظه­ای این مقدار می­توانیم در زمان های دلخواه که قبلا با delay ایجاد می­کردیم کار خود را پیش ببریم با این فرق که پردازنده همیشه در حال ایجاد وقفه نیست.

در بخش دوم می­خواهیم کد آزمایش اول را ایندفعه با تابع millis بزنیم که کد آن به شرح زیر است:

int previousMillis;

const long interval = 100; // millis

int state;

void setup() {

  pinMode(13, OUTPUT);

  state = LOW;

  previousMillis = 0;

}

void loop() {

  int currentMillis = millis();

  if (currentMillis - previousMillis >= interval){

      previousMillis = currentMillis;

      if(state == LOW){

        state = HIGH;

      }else{

        state = LOW;

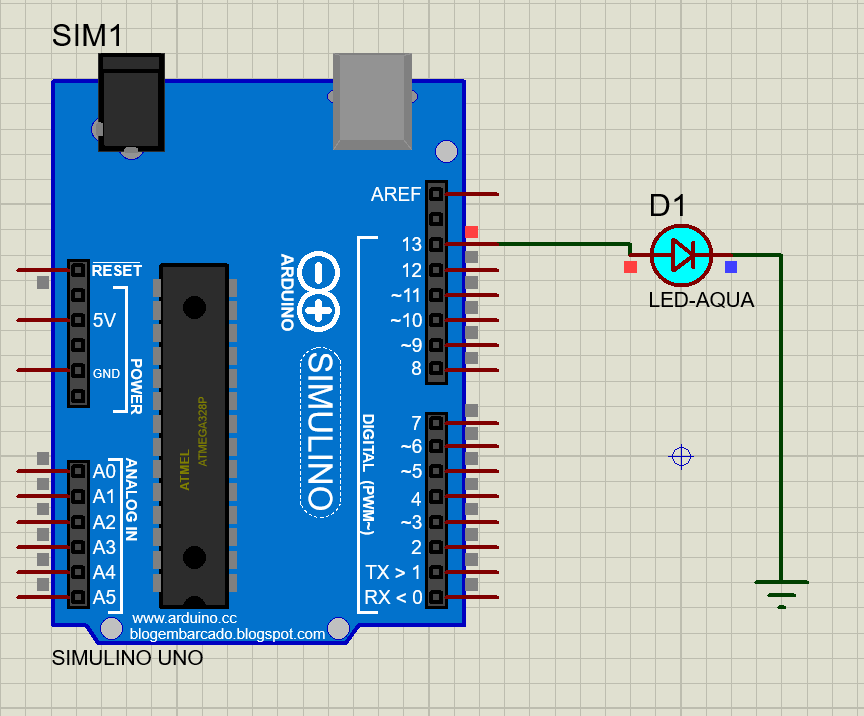
      }

      digitalWrite(13, state);

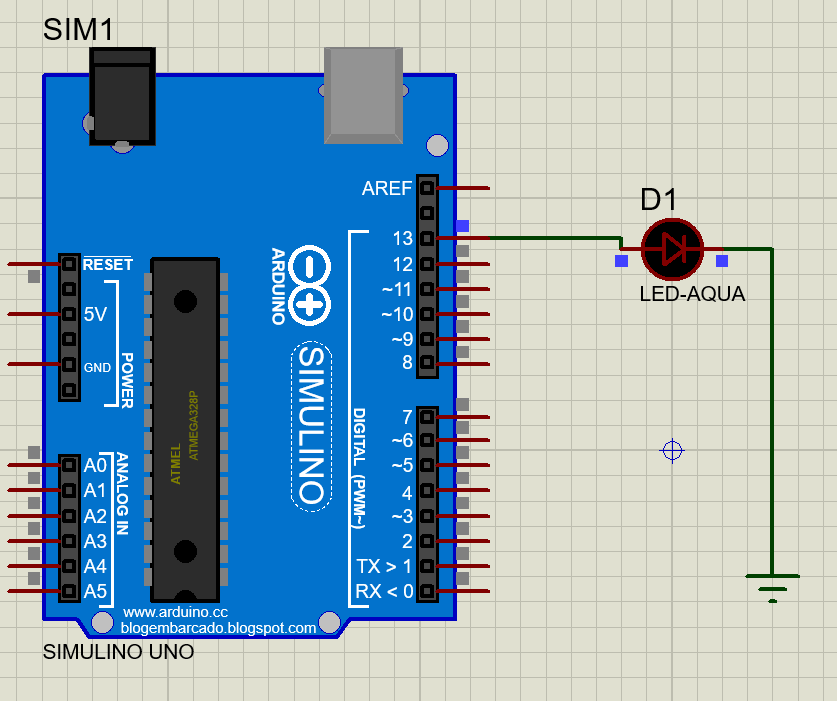
  }

}

و در زمان اجرا در حالت روشن:



و در حالت خاموش:



در بخش سوم قصد داریم تمرین اول را با 8 چراغ و یک کلید برای شروع پیاده­سازی کنیم که کد آن به شرح زیر است:

int previousMillis;

const long interval = 500; // millis

int state;

int start;

void setup(){

  pinMode(1, INPUT);

  pinMode(2, OUTPUT);

  pinMode(3, OUTPUT);

  pinMode(4, OUTPUT);

  pinMode(5, OUTPUT);

  pinMode(6, OUTPUT);

  pinMode(7, OUTPUT);

  pinMode(8, OUTPUT);

  pinMode(9, OUTPUT);

  state = LOW;

  previousMillis = 0;

  start = 0;

}

void loop(){

  start = digitalRead(1);

  if (start == 1){

    int currentMillis = millis();

    if (currentMillis - previousMillis >= interval){

      previousMillis = currentMillis;

      if(state == LOW){

        state = HIGH;

      }else{

        state = LOW;

      }

      digitalWrite(2, state);

      digitalWrite(3, state);

      digitalWrite(4, state);

      digitalWrite(5, state);

      digitalWrite(6, state);

      digitalWrite(7, state);

      digitalWrite(8, state);

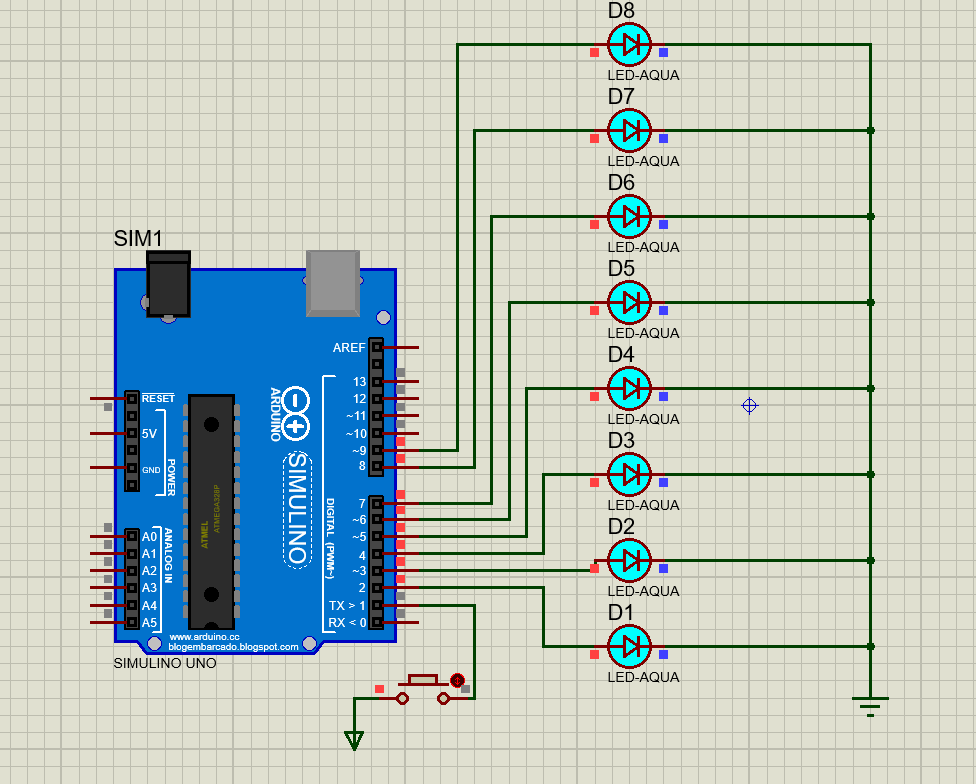
      digitalWrite(9, state);

    }

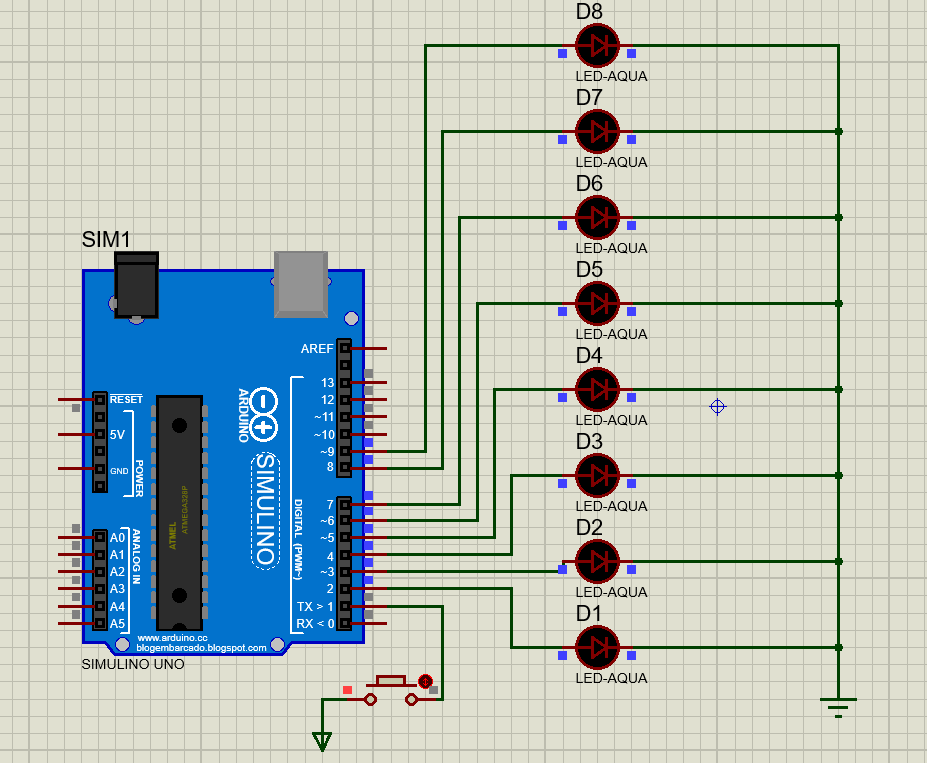
  }

}

و در زمان اجرا در حالت روشن :



و حالت خاموش:



در بخش بعدی قصد داریم 8 چراغ به صورتی نوبتی خاموش و روشن شوند که کد مربوطه به شرح زیر می باشد:

int previousMillis;

const long interval = 500; // millis

int start;

int turn;

void setup(){

  pinMode(1, INPUT);

  pinMode(2, OUTPUT);

  pinMode(3, OUTPUT);

  pinMode(4, OUTPUT);

  pinMode(5, OUTPUT);

  pinMode(6, OUTPUT);

  pinMode(7, OUTPUT);

  pinMode(8, OUTPUT);

  pinMode(9, OUTPUT);

  turn = 2;

  start = 0;

  previousMillis = 0;

}

void loop(){

  start = digitalRead(1);

  if(start == 1){

    int currentMillis = millis();

    if (currentMillis - previousMillis >= interval){

      previousMillis = currentMillis;

      digitalWrite(turn, HIGH);

      if (turn == 2){

        digitalWrite(9, LOW);

      }

      if (turn > 2){

        digitalWrite(turn - 1, LOW);

      }

      if (turn == 9){

        turn = 2;

      }else{

        turn = turn + 1;

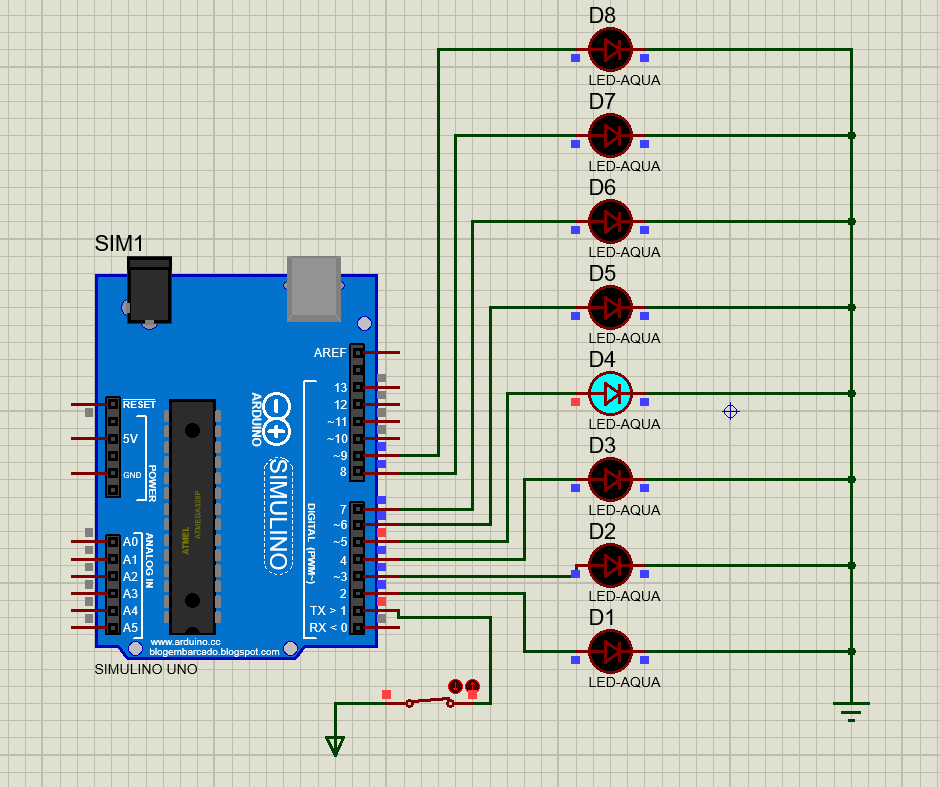
      }

    }

  }

}

و در زمان اجرا:



و در بخش نهایی در Arduino تنها با استفاده از ترفندهای نرم افزاری قادر به همگام سازی فرآیند ها هستیم در این بخش از کتابخانه Protothreads استفاده می­کنیم و کد 2 چراغ چشمک زن با سرعت های متفاوت به شرح زیر است:

#include "protothreads.h"

pt slowBlinking;

pt fastBlinking;

int fastBlinkThread(struct pt\* pt)

{

  PT\_BEGIN(pt);

  while(true)

  {

  digitalWrite(2, HIGH);

  PT\_SLEEP(pt, 250);

  digitalWrite(2, LOW);

  PT\_SLEEP(pt, 250);

  }

  PT\_END(pt);

}

int slowBlinkThread(struct pt\* pt)

{

  PT\_BEGIN(pt);

  while(true)

  {

  digitalWrite(3, HIGH);

  PT\_SLEEP(pt, 1000);

  digitalWrite(3, LOW);

  PT\_SLEEP(pt, 1000);

  }

  PT\_END(pt);

}

void setup()

{

  PT\_INIT(&slowBlinking);

  PT\_INIT(&fastBlinking);

  pinMode(2, OUTPUT);

  pinMode(3, OUTPUT);

}

void loop()

{

  PT\_SCHEDULE(slowBlinkThread(&slowBlinking));

  PT\_SCHEDULE(fastBlinkThread(&fastBlinking));

}

در اینجا هر Thread یا ریسمان یک تابع دریافت می­کند که مسئول اجرای آن می­شود

**مراجع**

[**Protothreading guide**](https://www.digikey.com/en/maker/blogs/2022/how-to-write-multi-threaded-arduino-programs)