آزمایشگاه ریز پردازنده **آزمایش شماره ی 1**

نام و نام خانوادگی: هلیا سادات هاشمی پور **شماره دانشجویی:**9831106

نام استاد:استاد علیزاده **تاریخ:**14 مهر 1400

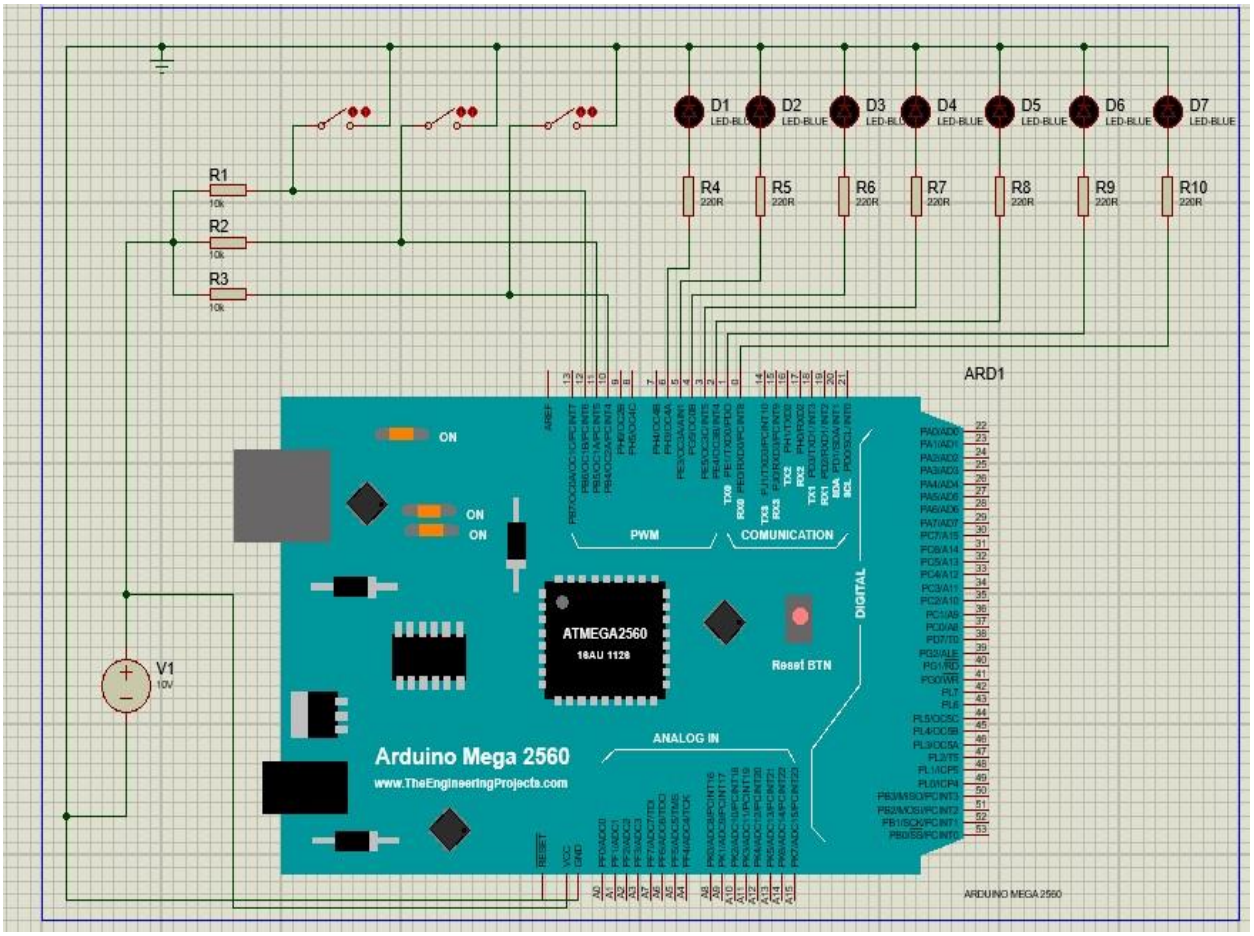
هدف آزمایش:

- آشنایی واحد PIO
- آشنایی با روش های سرکشی (Polling) و وقفه محور (Interrupt-Driven) برای مدیریت واحد های جانبی

مقایسه دو روش سرکشی و وقفه محور

.1-برنامه آزمایش را به روش سرکشی بنویسید و پس از آن که از درستی کارکرد مدار و برنامه خود مطمئن شدید گام دوم رو انجام دهید.

ابتدا یک پروژه ی جدید در محیط proteus می سازیم. با توجه به دستور کار یک برد آردینو ATmega 2560 ، چند مقاومت به اندازه های ۲۲۰ اهم و ۱۰ کیلو اهم، یکVSOURCE ۱۰ ولتی، سه کلید ، یک GROUND و ۷ لامپ LED-BLUE مطابق با شکل2.1 دستورکار در پروتئوس پیاده سازی می کنیم. سر مثبت منبع ولتاژ را به پایه ی VCC برد و سرمنفی آن را به پایه ی GND وصل می کنیم.مدار پیاده سازی شده به صورت زیر است.



مقاومت های R1و R2و R3را به صورت Pull-Up بسته ایم. یک پروژه ی جدید در آردوینو می سازیم و در بخش Tools برد ATmega2560را انتخاب می کنیم. سپس باتوجه به خواسته ی سوال،کد مورد نظر را پیاده سازی می کنیم.پس از اجرا کردن مسیر فایلی با پسوند hex. را کپی می کنیم سپس در نرم افزار پروتئوس بر روی برد ATmega 2560 کلیک کرده و در بخش PROGRAM FILE ، paste میکنیم.سپس برنامه خود را ران می کنیم.

توضیح درمورد کد آردوینو:

کد نوشته شده در فضای آردوینو به این شکل است که ابتدا LEDهای مورد نظر را براساس شماره LEDهای موجود در برنامه ی پروتئوس Define می کنیم.

با استفاده pinMode()وضعیت هر یک از پین ها(ورودی/خروجی)را مشخص میکنیم.با توجه به اینکه مدارمان به صورت Pull-Up می باشد؛ اگر سوییچ بسته باشدLOW می باشد و اگر سوییچ باز باشد HIGH است.

مطابق دستور کار، ابتدا همه ی LEDها خاموش می باشند. (با استفاده از digitalWrite()وضعیت خاموش بودن یا روشن بودن LEDها را با توجه به خواسته دستورکار مشخص می کنیم.)

- برای سوییچ نخست ، LED ها از سمت چپ به راست با تاخیر ۳۰۰ میلی ثانیه روشن می شوند.
- اگر سوییچ دوم بسته باشد. با استفاده از تابع strlen()به اندازه ی اسم خودم(helia) مشخص می کنیم که LEDها چند بار با هم روشن و خاموش شوند و بعد از اتمام آن روشن می مانند.
- با بسته شدن سوییچ سوم همه ی LEDها خاموش می شوند.

```
#define LED1 6
#define LED2 5
#define LED3 4
#define LED4 3
#define LED5 2
#define LED6 1
#define LED7 0
```

Define LEDS based on their pins number

```
const long interval_delay = 300;

void setup() {
  // put your setup code here, to run once:
 pinMode(LED1,OUTPUT);
 pinMode(LED2,OUTPUT);
 pinMode(LED3,OUTPUT);
 pinMode(LED4,OUTPUT);
 pinMode(LED5,OUTPUT);
 pinMode(LED6,OUTPUT);
 pinMode(LED7,OUTPUT);
 //INPUTS (SWITCHES)
 pinMode(6,INPUT);//button1
 pinMode(7,INPUT);//button2
 pinMode(8,INPUT);//button3
}
void loop() {
  // put your main code here, to run repeatedly:
//PULL UP CIRCUIT
 if(digitalRead(12)==LOW){//button1
   //at first LEDs are OFF(LOW)
  digitalWrite(LED1,LOW);
  digitalWrite(LED2,LOW);
  digitalWrite(LED3,LOW);
  digitalWrite(LED4,LOW);
  digitalWrite(LED5,LOW);
  digitalWrite(LED6,LOW);
  digitalWrite(LED7,LOW);
  //START
  digitalWrite(LED1,HIGH);
  delay(interval_delay);//DELAY FOR 0.3SEC
  digitalWrite(LED2,HIGH);
  delay(interval_delay);
  digitalWrite(LED3,HIGH);
  delay(interval_delay);
  digitalWrite(LED4,HIGH);
  delay(interval_delay);
  digitalWrite(LED5,HIGH);
  delay(interval_delay);
  digitalWrite(LED6,HIGH);
  delay(interval_delay);
  digitalWrite(LED7,HIGH);
  delay(interval_delay);
 }
```

Set that which pins are input/output

شرط اول

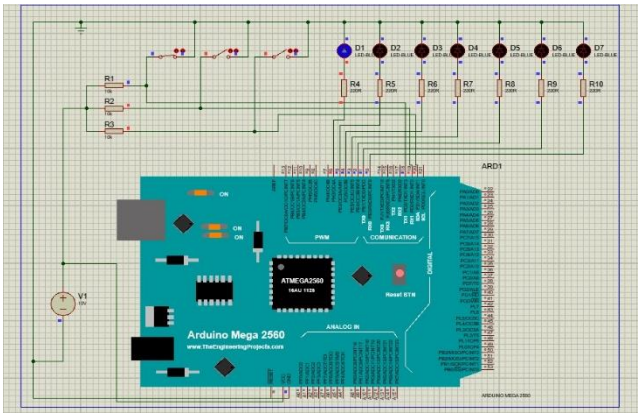
```
else if(digitalRead(11)==LOW){//button2
   //at first LEDs are low
   digitalWrite(LED1,LOW);
   digitalWrite(LED2,LOW);
   digitalWrite(LED3,LOW);
   digitalWrite(LED4,LOW);
   digitalWrite(LED5,LOW);
   digitalWrite(LED6,LOW);
   digitalWrite(LED7,LOW);
  for(int i=0;i< strlen("helia");i++){
   digitalWrite(LED1,HIGH);
   digitalWrite(LED2,HIGH);
   digitalWrite(LED3,HIGH);
   digitalWrite(LED4,HIGH);
   digitalWrite(LED5,HIGH);
   digitalWrite(LED6,HIGH);
   digitalWrite(LED7,HIGH);
   delay(interval_delay);
   digitalWrite(LED1,LOW);
   digitalWrite(LED2,LOW);
   digitalWrite(LED3,LOW);
   digitalWrite(LED4,LOW);
   digitalWrite(LED5,LOW);
   digitalWrite(LED6,LOW);
   digitalWrite(LED7,LOW);
   delay(interval_delay);
  }
   digitalWrite(LED1,HIGH);
   digitalWrite(LED2,HIGH);
   digitalWrite(LED3,HIGH);
   digitalWrite(LED4,HIGH);
   digitalWrite(LED5,HIGH);
   digitalWrite(LED6,HIGH);
   digitalWrite(LED7,HIGH);
   delay(2000);
```

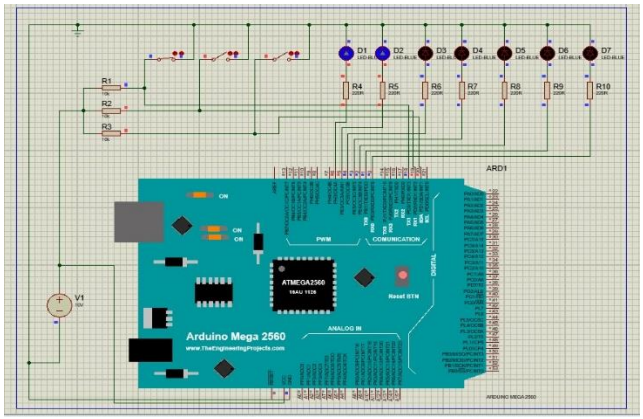
شرط دوم

```
  else if(digitalRead(10)==LOW){//button3
   digitalWrite(LED1,LOW);
   digitalWrite(LED2,LOW);
   digitalWrite(LED3,LOW);
   digitalWrite(LED4,LOW);
   digitalWrite(LED5,LOW);
   digitalWrite(LED6,LOW);
   digitalWrite(LED7,LOW);
  }

}
```

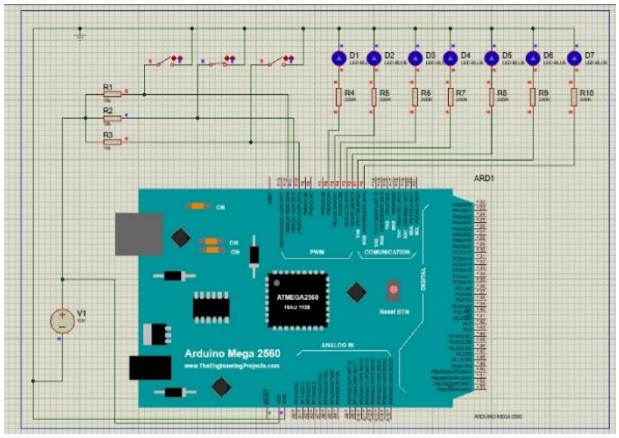
شرط سوم

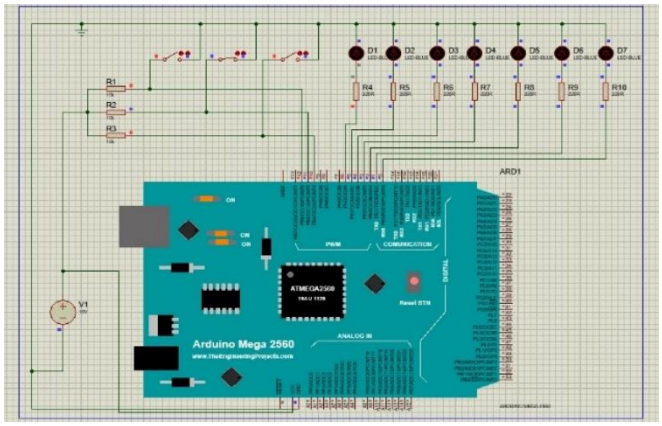
- با بسته شدن سوییچ اول LEDها به ترتیب از چپ به راست روشن می شوند.



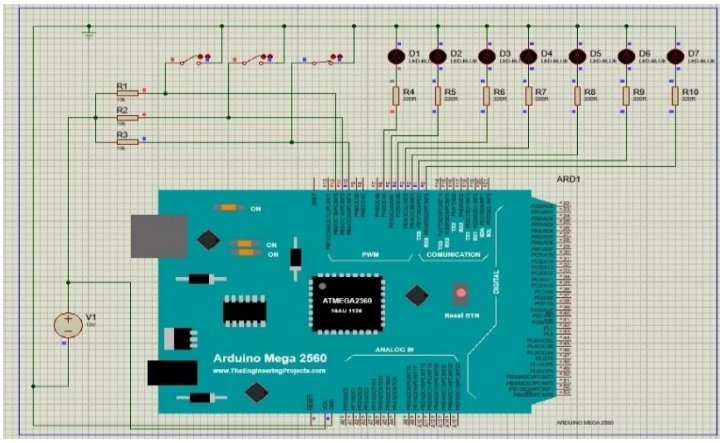


- با بسته شدن سوییچ دوم LED ها ،5 بار چشمک زده و در انتها روشن می مانند.





- با بسته شدن سوییچ سوم همه ی LEDها خاموش می شوند.



2 ..به پرسشهای زیر پاسخ دهید:

- **اگر دکمه را در حالت فشرده برای زمان طولانی نگه داریم چه اتفاقی خواهد افتاد؟ آیا با منطق کارکرد خواسته شده سازگار است؟ چه راه حلی برای این مشکل (در صورت وجود) می توان پیشنهاد کرد؟**

اگر سوییچ را به مدت طولانی نگه داریم چون در()loop به طور مداوم بسته بودن آن چک می شود پس از آنکه یک بار کارش را انجام داد، به صورت مداوم آن عملیات تکرار می شود؛ که چنین چیزی نباید اتفاق بیفد و انجام آن به صورت یک بار مد نظر ما می باشد.

راه حل پیشنهادی برای حل این مشکل این است که می توان یک شمارنده برای هرسوییچ نظر گرفت(counter1,counter2,counteter3) و در هر کدام از شرط ها (if) علاوه بر اینکه بسته بودن سوییچ مورد نظر چک می شد، تعداد دفعات انجام عملیات نیز چک شود یعنی شمارنده های هر سوییچ با تمام شدن یک دور از loop مورد نظر به اندازه ی یک واحد زیاد میشوند(counter++) اگر مقدار شمارنده ی مورد نظر از یک بیشتر بود و اگر کلید باز و بسته نشده بود حق اجرای دوباره را ندارد. توجه کنیم که شمارنده ی مورد نظر هر سوییچ با هر بار باز شدن آن مقدارش صفر می شود (به اصطلاح ریست می شود=counter=0)

- **فرض کنید می خواهیم برد مورد نظر علاوه بر فراهم کردن کارکرد خواسته شده در بالا، عمل دیگری را نیز به را نیز هر LED صورت زمان دار انجام دهد. برای نمونه در کنار کارکرد بالا، وضعیت روشن یا خاموش بودن یک5 ثانیه یک بار تغییر دهد. روشی برای افزودن این کارکرد تازه به برنامه پیشنهاد دهید.**

ابتدا برای اینکه همزمان با کارکرد اول،کارکرد دومی را که می خواهیم انجام شود را داشته باشیم؛ از کتابخانه یScheduler.h را به ابتدای برنامه اضافه می کنیم (با این کار می توان چندین کار به صورت همزمان را انجام داد)

یک متغیر از جنس بولین تعریف می کنیم(برایوضعیت روشن یا خاموش بودن LED) سپس در loop اصلی با استفاده از دستور ()digitalWrite و یک شرط برای اینکه بدانیم وضعیت فعلی LED به چه صورت است تا با توجه به آن وضعیت LED را تغییر دهیم. (اگر HIGH بود به LOW تغییرش دهیم)

حال برای اینکه ۵ ثانیه یک بار تغییر کند با استفاده ازدستور ()delayمی توان پنج ثانیه تاخیر را ایجاد کرد.

- **فرض کنید میخواهیم کارکرد دیگری را به دستگاه اضافه کنیم به این صورت که در صورت یک شدن یک پایه عملیات مشخصی را به عنوان پاسخ انجام دهد.(محدودیت زمانی برای پاسخ دادن وجود دارد) هیچ یک از اتفاق هاییک شدن پایه نباید از دست برود (بی پاسخ بماند). و یک شدن پایه نیز در هر زمانی ممکن است رخ دهد.آیا برنامهشما –که به روش سرکشی واحد های جانبی را بررسی میکند– می تواند در هر شرایطی (مثلا هنگام فشرده شدن کلید)این کارکرد را فراهم کند؟**

خیر،هنگامی که یک سوییچ بسته است (تا زمانی که باز نشود) عملیات مورد نظر را انجام می دهد. حال در این بین اگر یک شدن یک پایه دیگر رخ دهد چون برنامه درحال اجرای عملیات سوییچ مورد نظر است؛ برنامه نمی تواند آن را بررسی بکند.

- **فرض کنید به دلیل محدودیت در توان مصرفی می خواهیم پردازنده در هنگام بیکاری به خواب برود. در زمان خواب پردازنده هیچ دستوری را اجرا نمی کند. روش سرکشی چه قدر با این نیازمندی سازگاری دارد؟ آیا می توان با این روش هم به خواب رفت و هم کارکرد درست آزمایش را فراهم کرد؟**

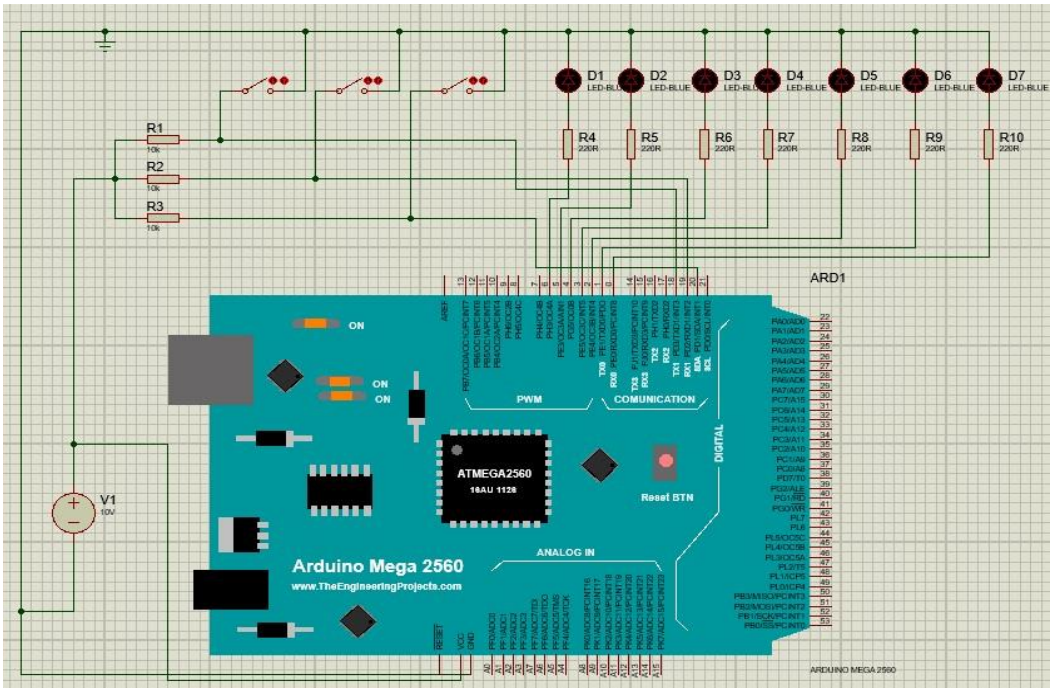
با روش سرکشی نمی توان هم به خواب رفت و هم کارکرد درستی را فراهم کرد؛ زیرا در این روش تمامی مراحل بصورت مداوم در حال اجرا هستند و در صورتی که به خواب برود چون وقفه ای وجود ندارد که سیستم را بیدار کند ،کارکرد درستی ارائه نمی شود.

3-با پاسخ به پرسش های بالا می توان دریافت که روش سرکشی برای کنترل واحد های جانبی با اینکه در برنامه های کوچک و به نسبت ساده قابل پیاده سازی است، همواره روش خوبی نیست و گاهی نمی تواند نیازمندی های ما را فراهم کند. اکنون آزمایش را به روش وقفه محور انجام دهید. پیاده سازی نیازمندی های خواسته شده در گام دوم را به روش سرکشی و وقفه محور مقایسه کنید.

مطابق با مداری که برای قسمت سرکشی بستیم در این قسمت هم همان مدار را می بندیم فقط برای سوییچ ها پین های مربوطه را (در پیش گزارش پین های وقفه رامشخص کردیم) به پین های ۱۸،۱۹،۲۰ وصل می کنیم.طبق دستور کار از دستور attachInterurpt()که دارای سه آرگومان هست استفاده می کنیم.

- آرگومان نخست آن پین وقفه مورد نظر می باشد.
- آرگومان دوم ISRمی باشد.
- سومین آرگومان mode هست که نشان می دهد چه زمانی وقفه داریم. با توجه به اینکه مدار ما به صورت Pull-Up - هست آرگومان سوم FALLING میشود .

چون ISR تا حد امکان باید کوتاه و سریع باشد و نمی توان بیشتر از چند میکرو ثانیه تاخیرداشت بنابراین برای اطمینان از اینکه مقادیر بین ISR برنامه به درستی به روزرسانی شده است ،دو متغیر به صورت global به نام های flag1 و flag2 را volatileتعریف می کنیم و هر جا نیاز به تاخیر بود مقدار آن را true میکنیم. برخلاف روش سرکشی اگر سوییچ ها به مدت طولانی بسته باشند فقط یک بار عملیات مربوطه اجرا می شود و از این روش می توان برای کاهش توان مصرفی پردازنده بدون آنکه مشکلی پیش آید به خواب برود. مدار آن به صورت زیر است.



```
#define LED1 6
#define LED2 5
#define LED3 4
#define LED4 3
#define LED5 2
#define LED6 1
#define LED7 0

volatile boolean flag1=false;
volatile boolean flag2=false;
const long interval_delay = 300;

void setup() {
  // put your setup code here, to run once:
  pinMode(LED1,OUTPUT);
  pinMode(LED2,OUTPUT);
  pinMode(LED3,OUTPUT);
  pinMode(LED4,OUTPUT);
  pinMode(LED5,OUTPUT);
  pinMode(LED6,OUTPUT);
  pinMode(LED7,OUTPUT);

  pinMode(18,INPUT);
  pinMode(19,INPUT);
  pinMode(20,INPUT);

  attachInterrupt(digitalPinToInterrupt(18), switch1Pressed, FALLING);
  attachInterrupt(digitalPinToInterrupt(19), switch2Pressed, FALLING);
  attachInterrupt(digitalPinToInterrupt(20), switch3Pressed, FALLING);
}
void switch1Pressed(){
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    digitalWrite(LED4, LOW);
    digitalWrite(LED5, LOW);
    digitalWrite(LED6, LOW);
    digitalWrite(LED7, LOW);
    flag1=true;
}

void switch2Pressed(){
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    digitalWrite(LED4, LOW);
    digitalWrite(LED5, LOW);
    digitalWrite(LED6, LOW);
    digitalWrite(LED7, LOW);
    flag2=true;
}
void switch3Pressed(){
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    digitalWrite(LED4, LOW);
    digitalWrite(LED5, LOW);
    digitalWrite(LED6, LOW);
    digitalWrite(LED7, LOW);
}
```

پین های وقفه

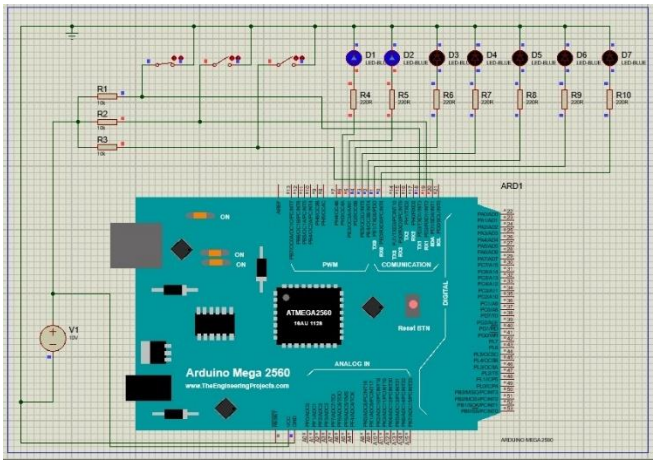
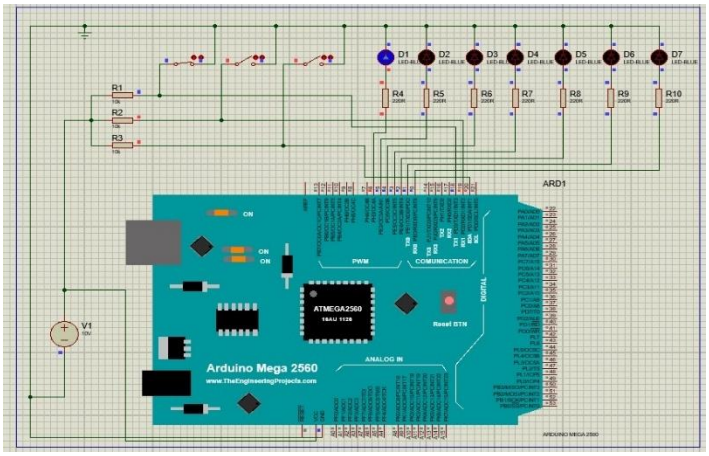
```
void loop() {
  // put your main code here, to run repeatedly:

  if(flag1==true){
    digitalWrite(LED1,HIGH);
    delay(interval_delay);
    digitalWrite(LED2,HIGH);
    delay(interval_delay);
    digitalWrite(LED3,HIGH);
    delay(interval_delay);
    digitalWrite(LED4,HIGH);
    delay(interval_delay);
    digitalWrite(LED5,HIGH);
    delay(interval_delay);
    digitalWrite(LED6,HIGH);
    delay(interval_delay);
    digitalWrite(LED7,HIGH);
    delay(interval_delay);
    flag1=false;
  }

  if(flag2==true){
    for (int i=0;i<strlen("helia");i++){
      digitalWrite(LED1, HIGH);
      digitalWrite(LED2, HIGH);
      digitalWrite(LED3, HIGH);
      digitalWrite(LED4, HIGH);
      digitalWrite(LED5, HIGH);
      digitalWrite(LED6, HIGH);
      digitalWrite(LED7, HIGH);

      delay(interval_delay);
      digitalWrite(LED1, LOW);
      digitalWrite(LED2, LOW);
      digitalWrite(LED3, LOW);
      digitalWrite(LED4, LOW);
      digitalWrite(LED5, LOW);
      digitalWrite(LED6, LOW);
      digitalWrite(LED7, LOW);
      delay(interval_delay);

    }
      digitalWrite(LED1, HIGH);
      digitalWrite(LED2, HIGH);
      digitalWrite(LED3, HIGH);
      digitalWrite(LED4, HIGH);
      digitalWrite(LED5, HIGH);
      digitalWrite(LED6, HIGH);
      digitalWrite(LED7, HIGH);
      flag2=false;
  }
}
```

شرط اول

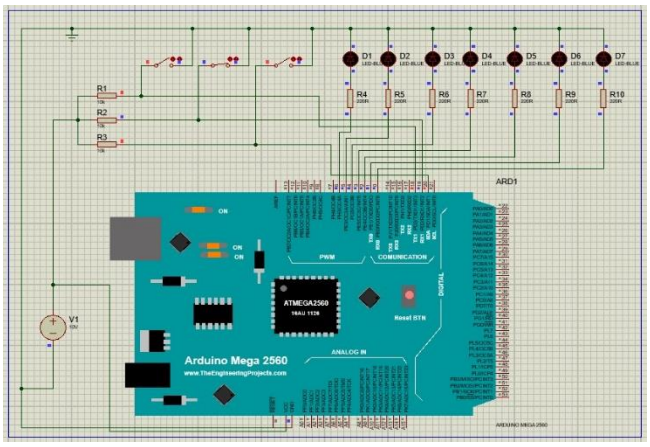
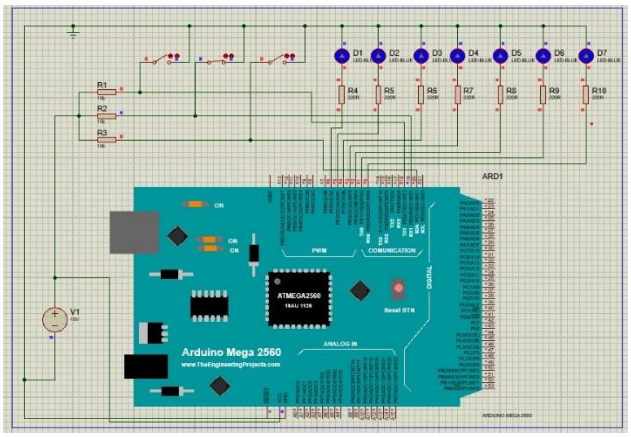
شرط دوم

شرط سوم

- با بسته شدن سوییچ اول LEDها به ترتیب از چپ به راست روشن می شوند.

- با بسته شدن سوییچ دوم LED ها ،5 بار چشمک زده و در انتها روشن می مانند.

- با بسته شدن سوییچ سوم همه ی LEDها خاموش می شوند.