"به نام یزدان پاک"

پیش گزارش آزمایش دوم

اعضای گروه:

محمد چوپان۹۸۳۱۱۲۵

محمد سپهر توکلی کرمانی ۹۸۳۱۱۱۱

تاریخ آزمایش : ۱۴۰۰/۰۷/۲۴

آزمایش ۲ :

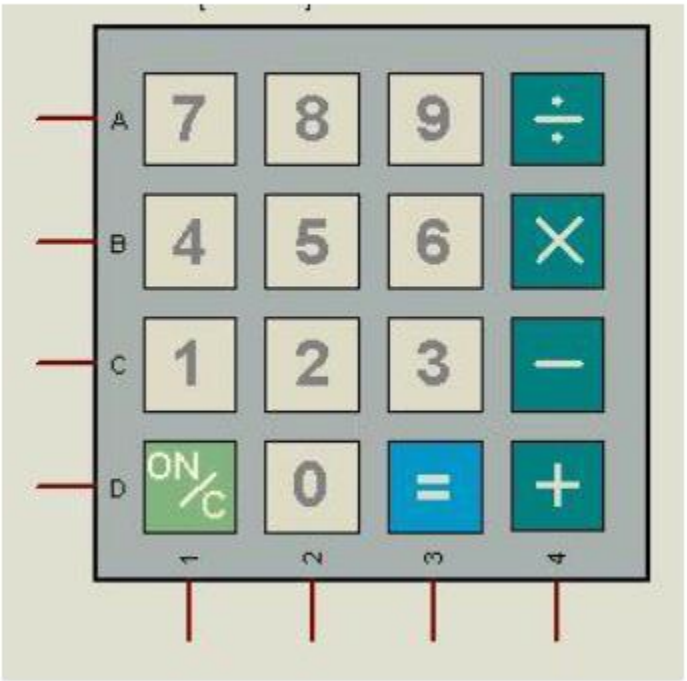
سوال ۱:

انواع keypad های ماتریسی و چگونگی کار کرد آن ها:

کیپد مجموعه از تاچ سوئیچ ها می‌باشند که به صورت سطری ،ستونی به طوری که تشکیل یک ماتریس دهند در کنار یک دیگر قرار گرفته اند.

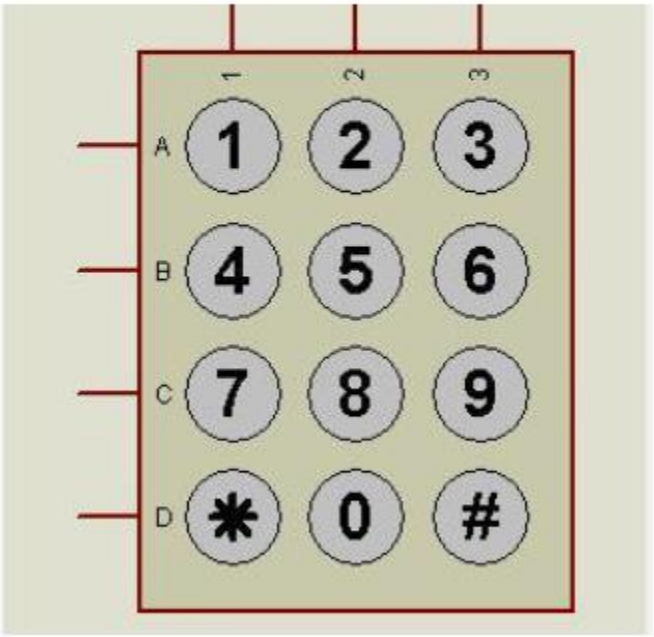
کیپد های ماتریسی انواع مختلفی دارند که انواع آن ها عبارت است از:

۱-



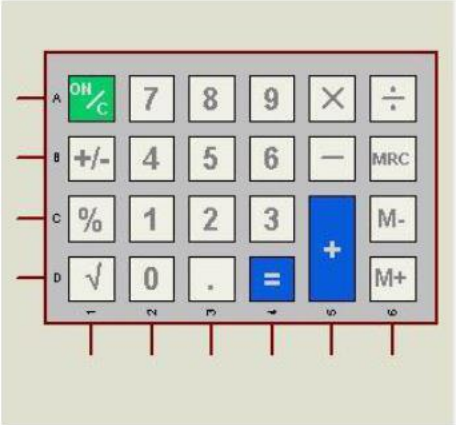
-٢

KEYPAD-PHONE



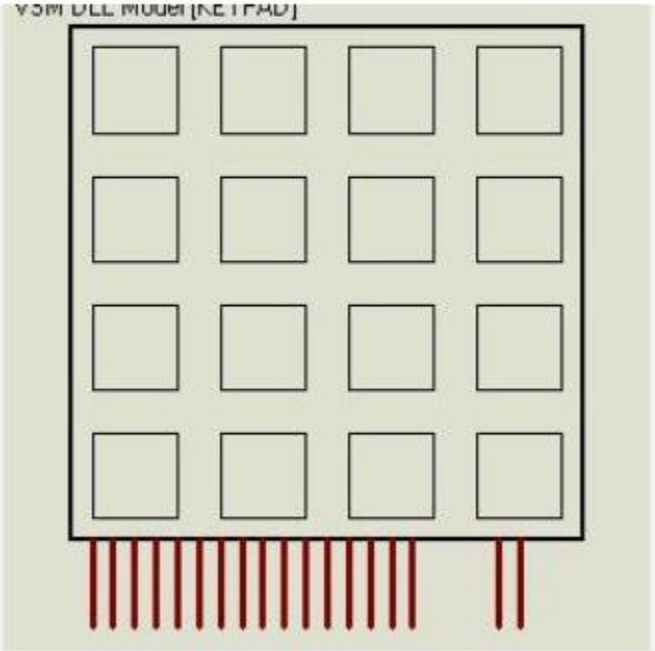
-٣

KEYPAD-CALCULATOR



-۴

KEYPAD-TRELLIS-C1



و انواع دیگر آن ها

حال نحوه کارکرد آن ها :

در صفحه کلیدهای ماتریسی ارتباط کلیدها به شکل ماتریس به هم متصل است. برای خواندن از این صفحه کلیدها تمامی پین‌ها را HIGH می‌کنیم. سپس یکی از ردیف‌ها (پین‌های R) را LOW کرده و در صورتی که یکی از ستون‌ها (پین‌های C) LOW شد، سطر و ستون دکمه فشرده شده مشخص می‌شود. این کار را در هر بار اسکن برای تمام ردیف‌ها تکرار می‌کنیم.

- پدیده‌ی نوسان (bounce) کلید چیست و چگونه می‌توان از بروز اشکالات ناشی از آن جلوگیری کرد؟

هنگام متصل شدن کلید دو صفحه رسانا بسیار به یکدیگر نزدیک می‌شوند. در این حالت مولکول‌های هوا می‌توانند رسانا شوند و باعث ایجاد جرقه و در نتیجه ایجاد ولتاژ نوسانی (شکل زیر) روی پین مربوطه شوند. برای جلوگیری از اتفاق می‌توان یک خازن را موازی با کلید بست و یا به صورت نرم‌افزاری پس از اینکه مدتی ولتاژ ثابت ماند تغییر وضعیت آن را در نظر گرفت.

- تعریف مختصر توابع مورد نیاز از کتابخانه Keypad.h مانند:

Keypad(makeKeymap(userKeymap)، row[]، col[]، rows، cols)

:

سازنده کلاس Keypad می‌باشد که با گرفتن نقشه‌ی کلیدها، شماره پین‌های هر سطر و ستون و تعداد کلیدهای هر کدام یک شی از این کلاس را برمی‌گرداند.

Char getKey() :

در صورتی که دکمه‌ای فشرده شده باشد کاراکتر آن را برمی‌گرداند. این دستور به صورت non-blocking است.

Char getKeys() :

در صورتی که وضعیت یکی از دکمه‌ها تغییر کرده باشد مقدار true را برمی‌گرداند.

char waitForKey() :

به شکل blocking منتظر فشرده شدن یک دکمه می‌ماند و در نهایت آن را برمی‌گرداند.

KeyState getState() :

وضعیت هر کلید را برمی‌گرداند که یکی از ۴ وضعیت IDLE، PRESSES، HOLD و RELEASED است.

boolean keyStateChanged() :

در صورتی که وضعیت یک دکمه تغییر کرده باشد true و در غیر این صورت false می‌دهد.

- نحوه و کاربردهای ارتباط سریال در آردوینو

تمام برد های آردوینو دارای حداقل یک عدد پورت سریال می باشد که اختصارا به آن ها UART USART,گفته می‌شود. در آردوینو از پین‌های دیجیتال ۰ و ۱ برای راه اندازی پورت سریال استفاده می‌شود و میتوان به وسیله‌ی درگاه USB بر روی برد ،آردوینو را به کامپیوتر متصل نمود)پین RX۰ (گیرنده اطلاعات) و پین ۱ TX (فرستنده اطلاعات)می باشد.(هنگامی که از پین های ۰ و ۱ به عنوان پروت سریال استفاده شود ، دیگر نمی‌توان از این پین ها به عنوان ورودی خروجی دیجیتال استفاده نمود.

- تعریف مختصر و نحوه کار با توابع ارتباط سریال مانند:

۱- begin() : نرخ داده را تنظیم میکند.

۲- end() : ارتباط سریال روی آن پورت را غیرفعال کرده و به شکل پورت ورودی و خروجی عادی می‌توان از آن استفاده کرد.

۳- find() :داده را از بافر میخواند تا زمانی که پیدا کند.

٤- parseInt() : عدد معتبر بعدی را می‌خواند و برمی‌گرداند.

٥- println() :چاپ میکند.

۶- read() : داده ها را میخواند.

۷- readStringUntil() : بافر را تا رسیدن به کاراکتر مشخص شده و می‌خواند و تعداد کاراکتر خوانده شده تا رسیدن به آن را برمی‌گرداند.

۸- write() : داده باینری را به شکل یک یا چند بایت ارسال می‌کند.

شرح آزمایش:

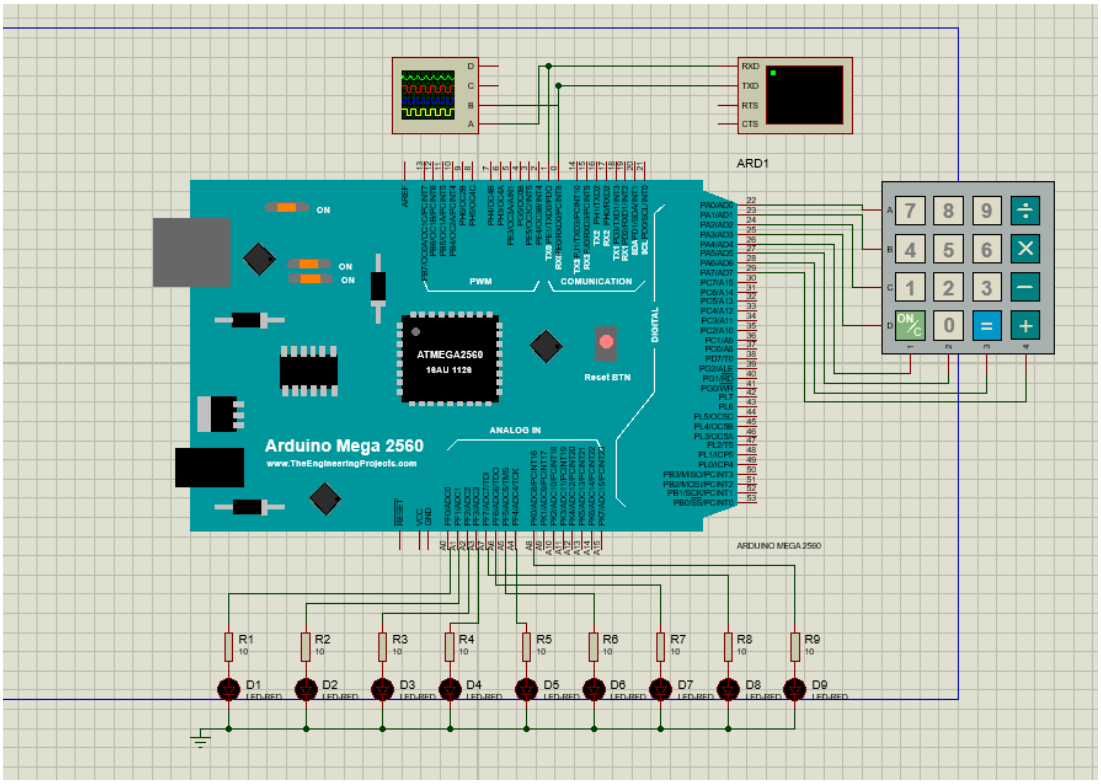
1. نخست 9 عدد LED و یک صفحه کلید را از میان componentهای پروتئوس اضافه و به برد Arduino Mega2560 وصل کنید. برنامه ای بنویسید که به تعداد عدد انتخاب شده در صفحه کلید از سمت چپ به راست LED ها را روشن کند.

ابتدا یک پروه جدید در proteus ساخته و سپس به آن برد Arduino atmega2560 را از کتابخانه آن اضافه می کنیم.

سپس مطابق دستور کار مدار داده شده را با virtual terminal,osiloscope ,resistance ,led و زمین و صفحه کلید مطابق شکل زیر پیاده سازی میکنیم.

پس از آن کد های مربوطه را در Arduino ide کامپایل کرده و فایل hex. آن را در proteus در برد قرار داده و اجرا میکنیم.

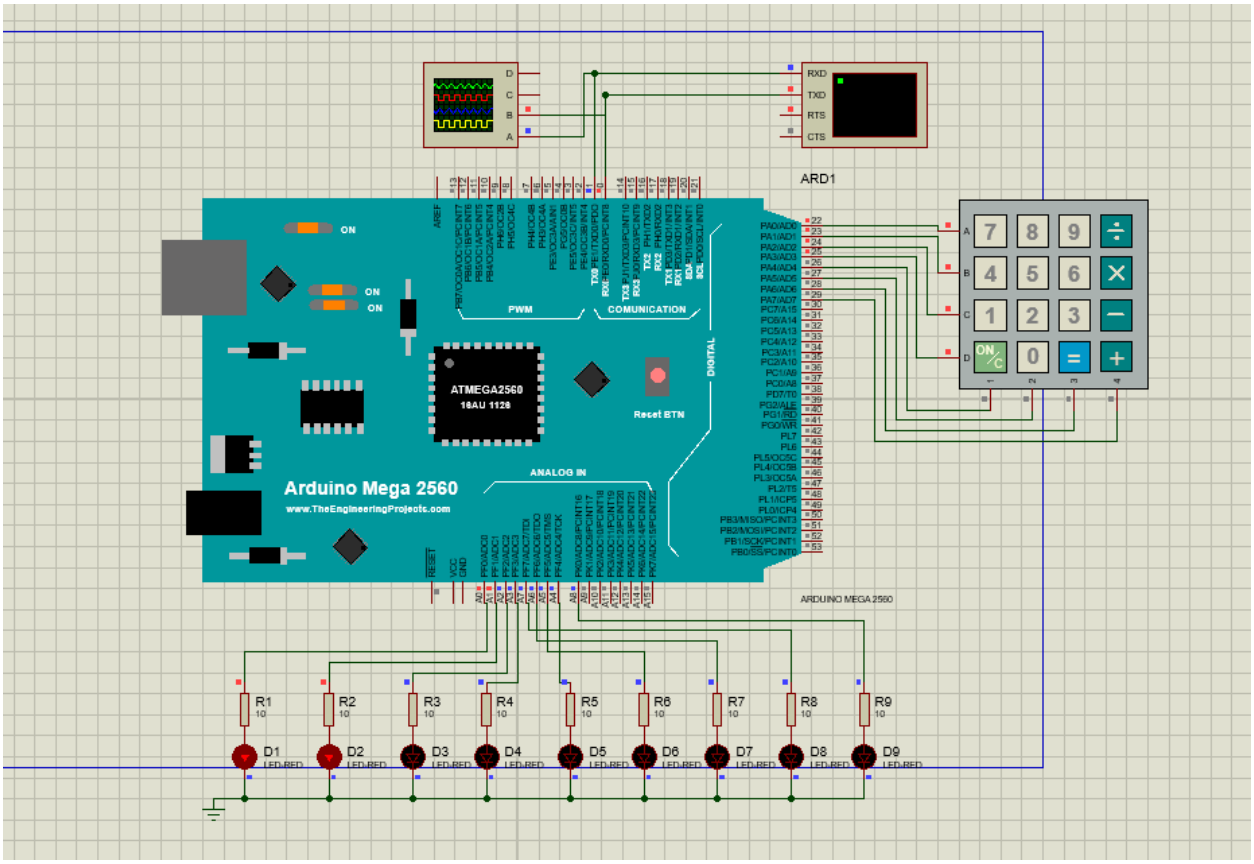
مدار طراحی شده برای هر ۳ قسمت :



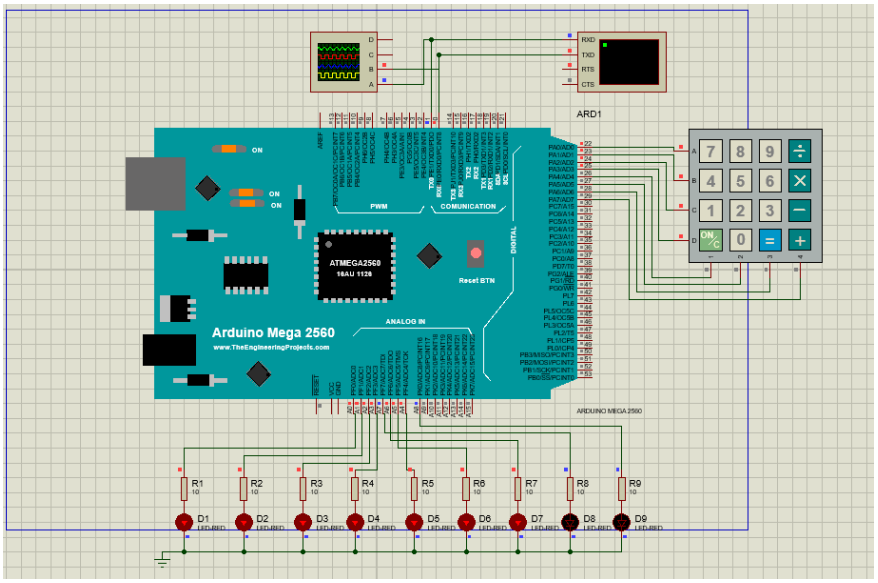
کد قسمت ۱:

```
#include <Keypad.h>//add library

const byte ROWS = 4; //four rows //my keypad rows
const byte COLS = 4; //four columns // my keypad columns
char keys[ROWS][COLS] = { // all my key pad charecters
  {'7','8','9', '/'},
  {'4','5','6', '*'},
  {'1','2','3', '-'},
  {'o','0','=', '+'}
};
byte rowPins[ROWS] = {22, 23, 24, 25}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {26, 27, 28, 29}; //connect to the column pinouts of the keypad

const byte ledPins[9] = {A0, A1, A2, A3, A4, A5, A6, A7, A8}; // connected len pins port

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS ); // define new key pad
void setup() {
  // put your setup code here, to run once:
  for (byte i = 0; i < 9; i++) // output pins led connected we use arra here
    pinMode(ledPins[i], OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
 char key = keypad.getKey();
  if (key){//read charecter and make it high
    for (char i = 0; i < (key-'0'); i++)
      digitalWrite(ledPins[i], HIGH);
  }
}
```

نتیجه مدار برای فشردن دکمه ۲:



نتیجه مدار برای فشردن دکمه ۷:



قسمت ۲:

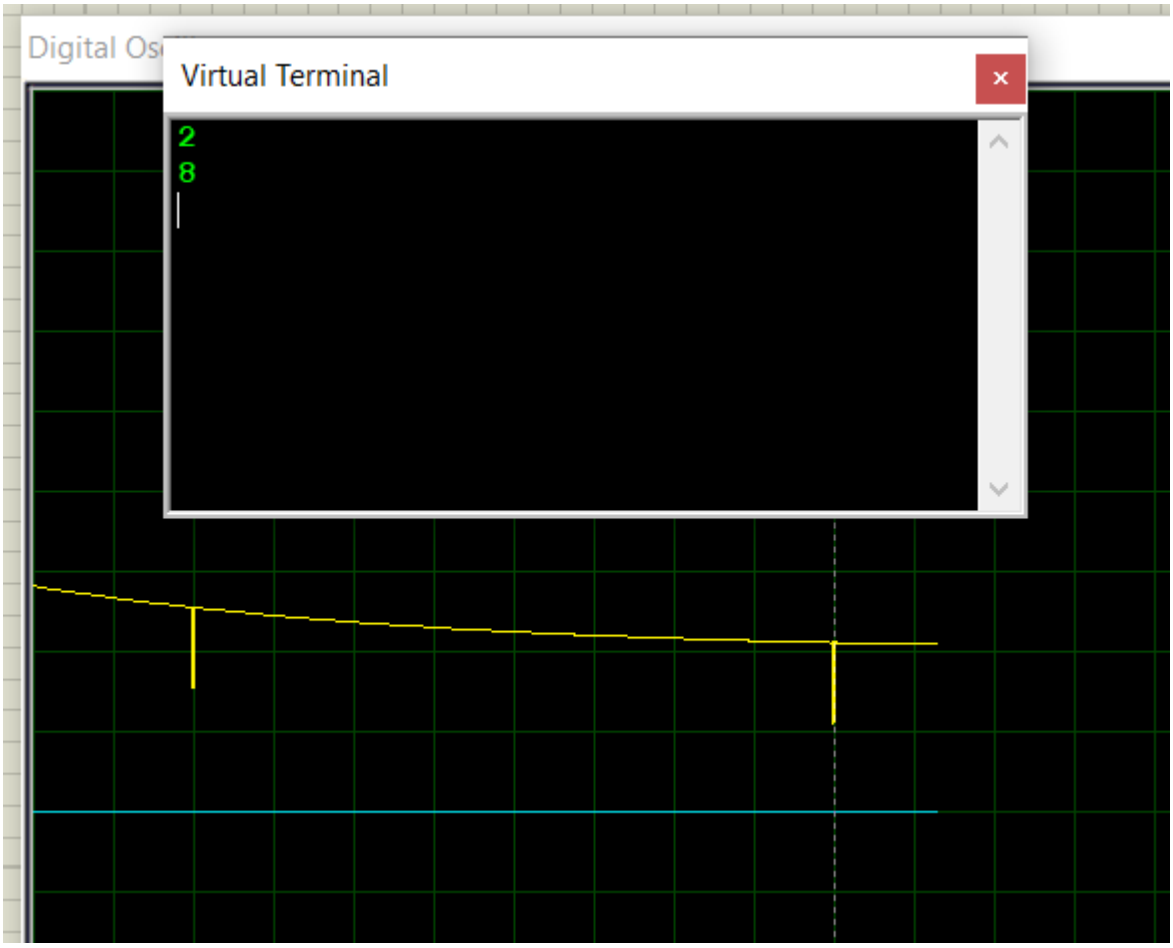
2. سپس ترمینال مجازی را (که در شکل زیر به صورت یک تبدیل USB-TTL نشان داده شده است) به پین‌های ارتباطی برد وصل کنید. برنامه‌ای بنویسید که کاراکتر روی دکمه فشرده شده را در ترمینال مجازی نشان دهد. حال این آزمایش را با اسیلوسکوپ متصل شده به سیم‌های ترمینال مجازی تکرار کنید. سیگنال فرستاده شده به ترمینال روی اسیلوسکوپ را ببینید. آیا می‌توانید آن را بررسی کنید؟

مدار همان مدار قسمت ۱ است اما کد آن متفاوت می باشد.

کد :

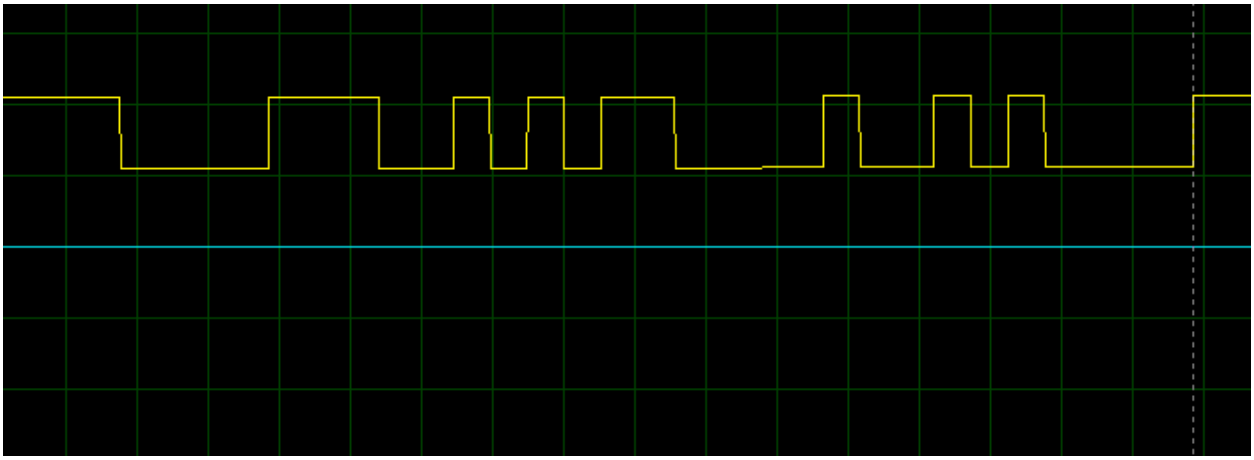
```
#include <Keypad.h>//add library

const byte ROWS = 4; //four rows //my keypad rows
const byte COLS = 4; //four columns // my keypad columns
char keys[ROWS][COLS] = { // all my key pad charecters
  {'7','8','9', '/'},
  {'4','5','6', '*'},
  {'1','2','3', '-'},
  {'o','0','=', '+'}
};
byte rowPins[ROWS] = {22, 23, 24, 25}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {26, 27, 28, 29}; //connect to the column pinouts of the keypad

const byte ledPins[9] = {A0, A1, A2, A3, A4, A5, A6, A7, A8}; // connected len pins port

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS ); // define new key pad
void setup() {
  // put your setup code here, to run once:
   Serial.begin(9600);// use virtual terminal serial
}
void loop() {
  // put your main code here, to run repeatedly:
 char key = keypad.getKey();
  if (key){//read charecter and make it high
    Serial.println(key);// show in virtual terminal
  }
```

نتیجه فشردن دکمه ۲ و ۸:

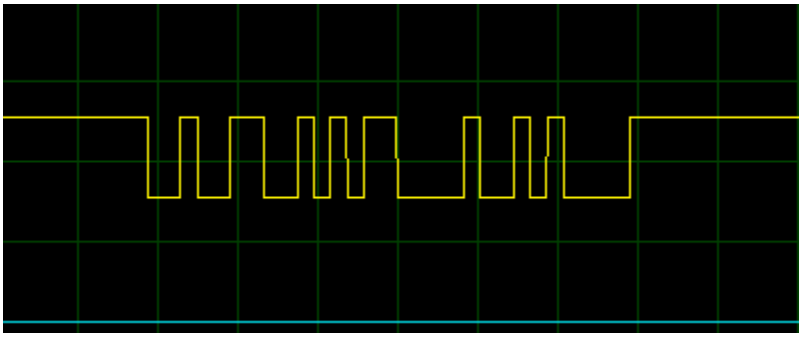


به صورت دقیق تر :

برای ۸:



برای ۲:



قسمت ۳:

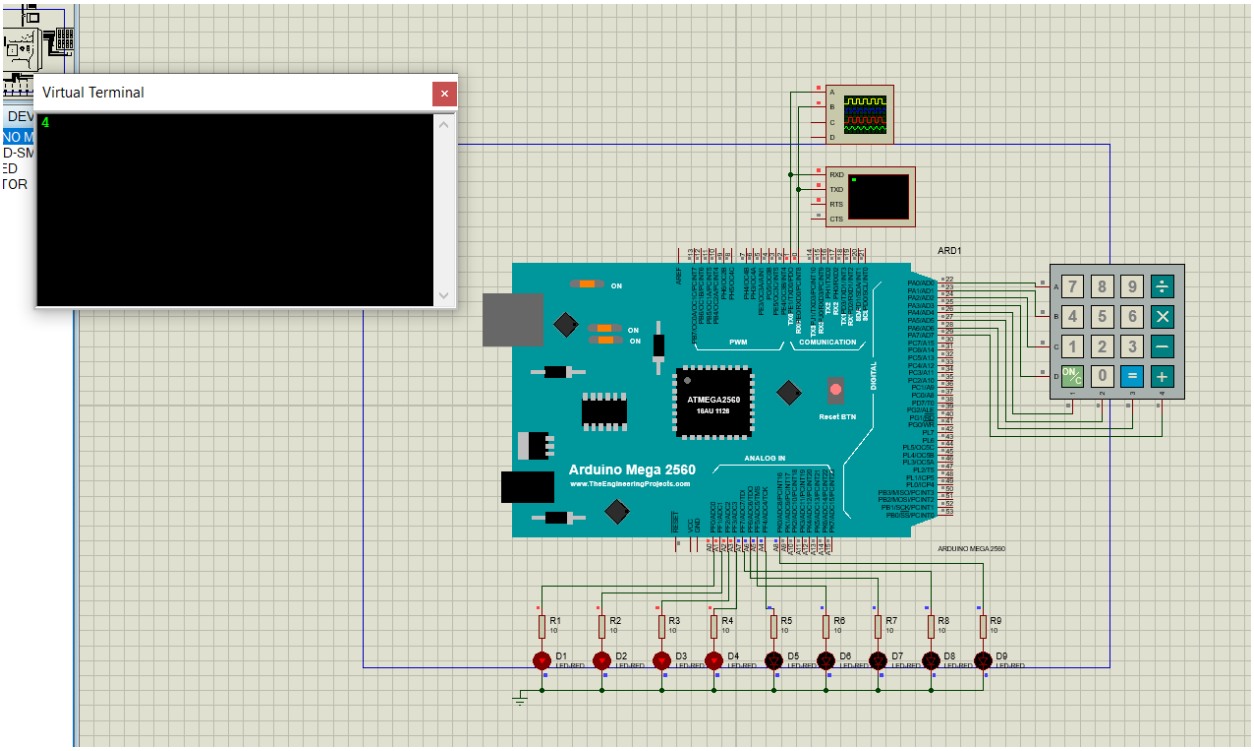
3. برنامه ای بنویسید که ترمینال مجازی، یک عدد بین 1 تا 9 را به عنوان ورودی بگیرد و به تعداد آن LED ها را از سمت چپ به راست روشن کند. در صورتی که عدد وارد شده بزرگتر از 9 بود پیام "Invalid number" را به عنوان خطا نمایش دهد.

در این قسمت به keypad نیازی نیست اما میتوانیم از همان مدار قسمت قبل استفاده کنیم.

کد :

```
const byte ledPins[9] = {A0, A1, A2, A3, A4, A5, A6, A7, A8};// connected pins to led
byte incomingByte = 0; // for incoming serial data

void setup() {
  // put your setup code here, to run once:
 Serial.begin(9600);

  for (byte i = 0; i < 9; i++)
    pinMode(ledPins[i], OUTPUT);// connect leds
}

void loop() {
  // put your main code here, to run repeatedly:
if (Serial.available() > 0) {
    // read the incoming byte:
    incomingByte = Serial.read();
    Serial.println('input number is :');
    Serial.println(incomingByte);
    if(incomingByte>'9'){
      Serial.println('invalid number');
      }

    // turn on corresponding led
    else{
      for (char i = 0; i < (incomingByte - '0'); i++)
        digitalWrite(ledPins[i], HIGH);

    }
  }
}
```

نتیجه وارد کردن ۴:



:۸