

آزمایش 8 درس ریزپردازنده و زبان اسمبلی  
آرمان حاتمی امیرحسین باریکلو

پاسخ سوالات دستور کار آزمایشگاه:

سوال اول : تفاوت آن ها در فرمول بدست آوردن مقدار  $v$  است به صورتی که در مدار اول مقاومت  
فتوسل مهم تر است در حالی که در مدار دوم مقاومت resistor

سوال دوم : دارای 3 پایه است یکی از آن ها به ولتاژ dc پایه دیگر به ground وصل میشود و در  
نهایت پایه vout به صورت آنالوگ به ما دما را میدهد

سوال سوم : در آردوینو مگا به ترتیب پایه های 50 برای miso پایه 51 برای mosi و پایه 52 برای  
sclk استفاده شده است پایه 53 نیز برای SS در نظر گرفته شده است

سوال چهارم : برای هر slave یک پین در نظر میگیریم که اگر آن را low کنیم ارتباط آغاز میشود  
سپس با استفاده از تابع transfer فرآیند انتقال را آغاز میکنیم و در نهایت دوباره پین آن slave را  
low میکنیم

سوال پنجم :  
توسط master

سوال ششم :  
تابع begin()  
ارتباط را راه اندازی میکند  
تابع transfer()  
داده ها را منتقل میکند  
تابع setclockdivider()  
فرکانس میکرو را بر پارامتر تقسیم و برای ارتباط spi استفاده میکند

سوال هفتم :  
turn on SPI in slave mode  
 $SPCR |= \_BV(SPE)$   
turn on interrupts

$$SPCR |=\_BV(SPIE)$$

سوال هشتم : مدیریت وقفه تولید شده در این تابع انجام میشود

لوازم مورد نیاز آزمایش:

1. سه دستگاه بورد آردینو 2560
2. ترمینال مجازی
3. اسیلسکوپ
4. مقاومت فتوسل برای اندازه گیری نور
5. سنسور lm35 برای اندازه گیری دما
6. یک resistor

شرح آزمایش :

برای شروع ابتدا هر 3 برد را با توجه به پایه های مناسب برای ارتباط spi در این نوع برد به یکدیگر متصل کردیم سپس همه سیم های ارتباطی بجز کلاک را به اسیلوسکوپ متصل کردیم پس از آن برای هر برد یک ترمینال مجازی قرار دادیم و به پین 13 هر برد وصل کردیم تا داده های دریافتی و ارسالی هر برد را در آنجا مشاهده کنیم پس از آن مقاومت فتوسل و lm35 را به ولتاژ مستقیم و سیم اتصال به زمین متصل کردیم و خروجی هر دو را به پین های A0 و A1 بردی که عنوان master در نظر گرفتیم وصل کردیم و سپس به سراغ برنامه نویسی برد ها رفتیم.

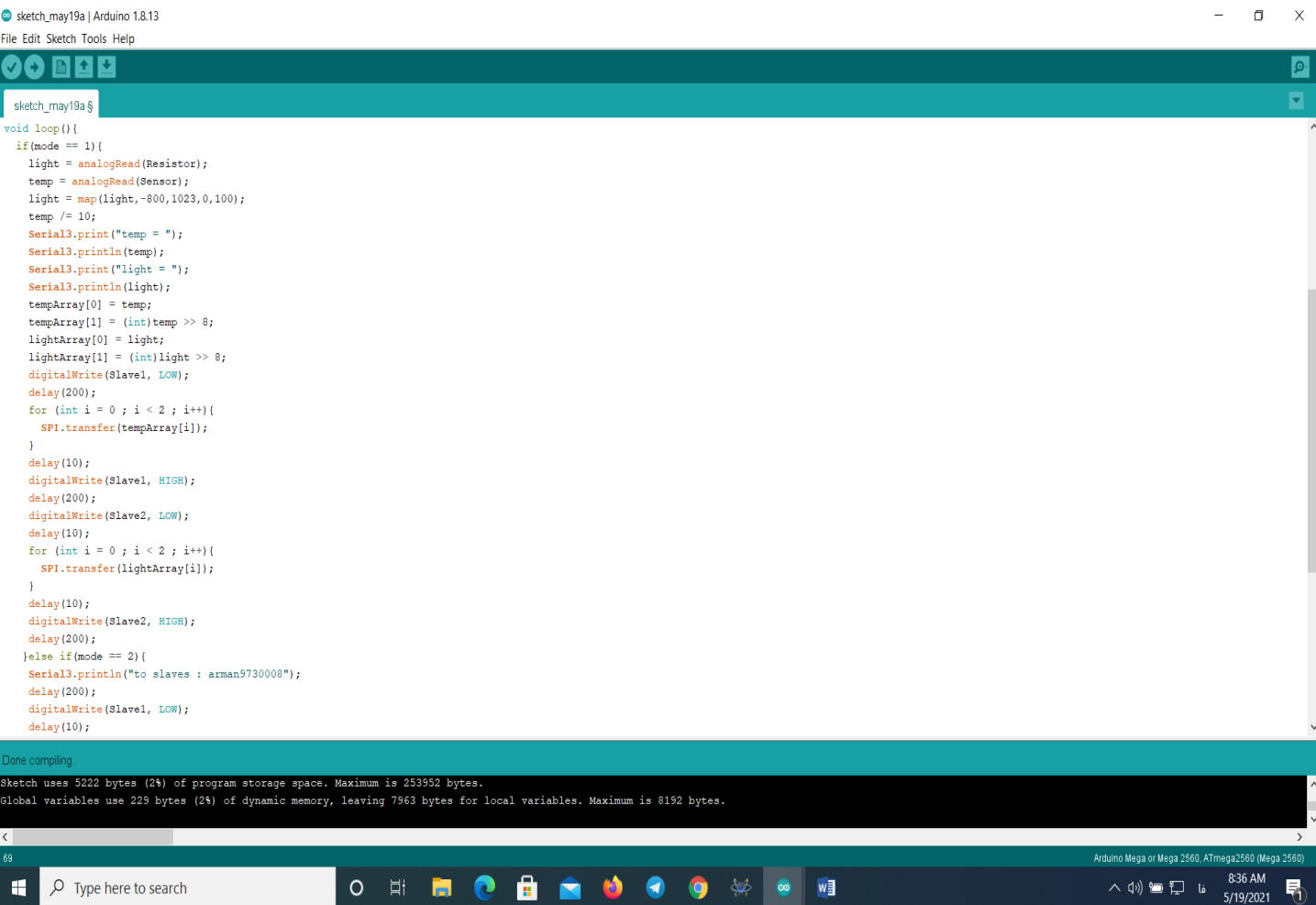
شرح کد آزمایش :

برای master 3 حالت داریم حالتی که بخواهیم hello arman 9730008 به هر دو slave بفرستیم یا حالتی که بخواهیم arman بفرستیم و hello arman در slave دیگر چاپ شود یا اینکه خروجی مقاومت فتوسل و lm35 را بخواهیم ارسال کنیم نحوه کلی برنامه نویسی master بدین شکل است اگر بخواهیم از A0 و A1 خروجی بگیریم از تابع readAnalog() استفاده میکنیم و سپس slave اول را low میکنیم و با استفاده از transfer این مقدار را برایش ارسال میکنیم و سپس آن را high میکنیم و همین کار را برای ارسال داده دیگر به slave دوم انجام میدهیم اگر بخواهیم یک رشته را منتقل کنیم این رشته را درون یک آرایه قرار میدهیم و به صورت کاراکتر به کاراکتر آن را transfer میکنیم به slave مورد نظر.

برای slave 2 حالت داریم یا اینکه میخواهیم رشته بخوانیم یا اینکه خروجی مقاومت فتوسل یا lm35 را دریافت کنیم برای این دو حالت ابتدا باید آردینو را با استفاده دستور  $SPCR |=\_BV(SPIE)$  به حالت slave ببریم و با استفاده از دستور  $SPCR |=\_BV(SPIE)$  اینترپت را برای آن فعال کنیم پس از این کار از آرایه buffer که از آن برای نوشتن مقادیر ارسال شده به این slave استفاده میشود استفاده

میکنیم و مقادیر دریافتی را چاپ میکنیم

عکس هایی از کد برنامه :



```
sketch_may19a | Arduino 1.8.13
File Edit Sketch Tools Help

sketch_may19a $
void loop() {
  if(mode == 1){
    light = analogRead(Resistor);
    temp = analogRead(Sensor);
    light = map(light,-800,1023,0,100);
    temp /= 10;
    Serial3.print("temp = ");
    Serial3.println(temp);
    Serial3.print("light = ");
    Serial3.println(light);
    tempArray[0] = temp;
    tempArray[1] = (int)temp >> 8;
    lightArray[0] = light;
    lightArray[1] = (int)light >> 8;
    digitalWrite(Slave1, LOW);
    delay(200);
    for (int i = 0 ; i < 2 ; i++){
      SPI.transfer(tempArray[i]);
    }
    delay(10);
    digitalWrite(Slave1, HIGH);
    delay(200);
    digitalWrite(Slave2, LOW);
    delay(10);
    for (int i = 0 ; i < 2 ; i++){
      SPI.transfer(lightArray[i]);
    }
    delay(10);
    digitalWrite(Slave2, HIGH);
    delay(200);
  }else if(mode == 2){
    Serial3.println("to slaves : arman5730008");
    delay(200);
    digitalWrite(Slave1, LOW);
    delay(10);
  }
}

Done compiling.
Sketch uses 5222 bytes (2%) of program storage space. Maximum is 253952 bytes.
Global variables use 229 bytes (2%) of dynamic memory, leaving 7963 bytes for local variables. Maximum is 8192 bytes.

Arduino Mega or Mega 2560, ATmega2560 (Mega 2560)
```

```
sketch_may19b
boolean done = false;
int number;
int mode = 2;

void setup() {
  Serial3.begin(9600);
  pinMode(input, INPUT_PULLUP);
  // turn on SPI in slave mode
  SPCR |= _BV(SPE);
  // turn on interrupts
  SPCR |= _BV(SPIE);
  SPI.attachInterrupt();
}

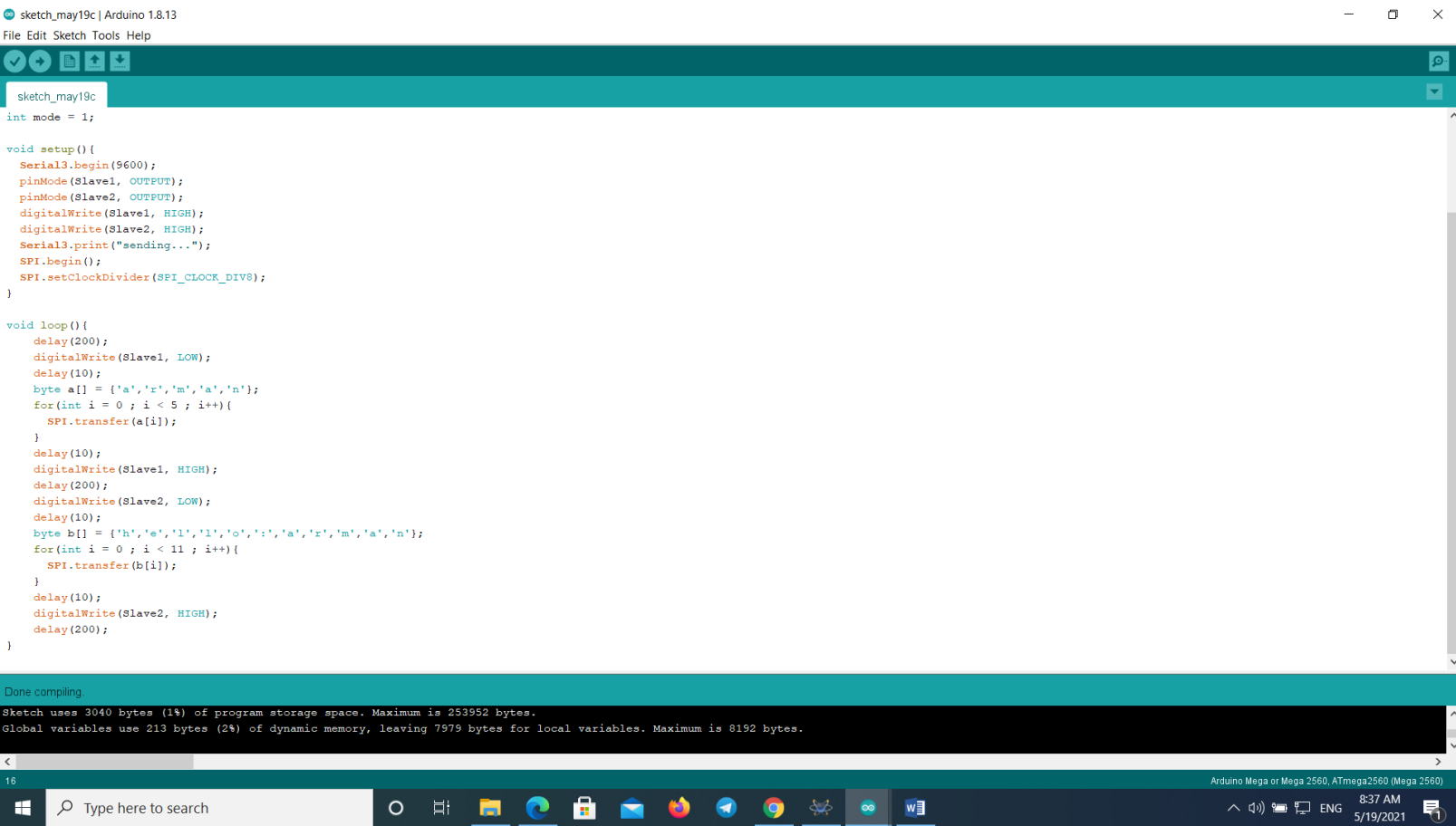
void loop() {
  if(mode == 1) {
    if(done) {
      done = !done;
      number = buffer1[0] + (buffer1[1] << 8);
      Serial3.println(number);
      index = 0;
    }
  } else if(mode == 2) {
    if(done) {
      done = !done;
      for(int i = 0; i < 12; i++) {
        char c = buffer1[i];
        Serial3.print(c);
      }
      Serial3.println("");
      index = 0;
    }
  }
}

ISR(SPI_STC_vect) {
```

Done compiling.

Sketch uses 2220 bytes (0%) of program storage space. Maximum is 253952 bytes.

Global variables use 202 bytes (2%) of dynamic memory, leaving 7990 bytes for local variables. Maximum is 8192 bytes.



عکس هایی از محیط پروتئوس برنامه :

