به نام خداوند مهربان

آزمایش ۷

امیررضارجبی ۹۸۳۱۱۲۶

۱-

پرسش: در چه کاربردهایی EEPROM به کار برده می‌شود؟ چرا در اینجا حافظه Flash یا RAM را به کار نمی‌بریم؟ تفاوت حافظه RAM با EEPROM در چیست؟

۲-

پرسش: اگر بخواهیم برای نگهداری مدهای کاری حافظه Flash را به کار ببریم، فرآیند نوشتن باید چگونه انجام شود که داده‌های دیگری که بر روی همان بلاک هستند از دست نروند؟

۳-

پرسش: اگر یک حافظه‌ی EEPROM بیرونی دارای 4KB حافظه و 2 پایه آدرس باشد، در این صورت می‌توان حداکثر چند KB حافظه EEPROM بیرونی بر روی یک باس مشترک داشت؟

۴-

پرسش: نمودار شماتیک برای اینکه دو AT24C02 را به یک باس مشترک وصل کنیم و حفاظت نوشتن غیر فعال باشد را رسم کنید. (آدرس‌دهی سخت‌افزاری دلخواه - باس را هم به پایه‌های میکروکنترلر متصل کنید)

۵-

پرسش: همخوانی این دنباله فریم ها را با پروتکل TWI بررسی کنید. (فریم‌های آدرس و داده را مشخص کنید، دستور خواندن یا نوشتن چگونه مشخص می‌شوند؟)

۶-

پرسش: فرکانس کلاک در کدام دستگاه پیکربندی می‌شود؟ کلاک را کدام دستگاه فراهم می‌کند؟ با توجه به زمان مورد نیاز برای انجام عملیات نوشتن، با فرض اینکه کلاک را 10KHz تنظیم کرده باشیم، در این صورت حداکثر با چه نرخی می‌توان عملیات نوشتن را انجام داد؟

۷-

پرسش: هر یک از تابع‌های نوشته‌شده را از راه لینک کتابخانه Wire، در مستندات آردوینو بررسی کنید و کد لازم را برای تولید دنباله‌ی فریم ها برای عملیات نوشتن و خواندن گفته‌شده (با این تابع ها) بنویسید.

پرسش ۱:

E^2PROM در میکروکنترل‌ها کارت‌های هوشمند و سیستم‌های ریموت برای ذخایر مقادیر نسبتا کمی از داده‌ها استفاده می‌شود و اجازه می‌دهد بایت‌ها تکی پاک و برنامه‌ریزی شوند.

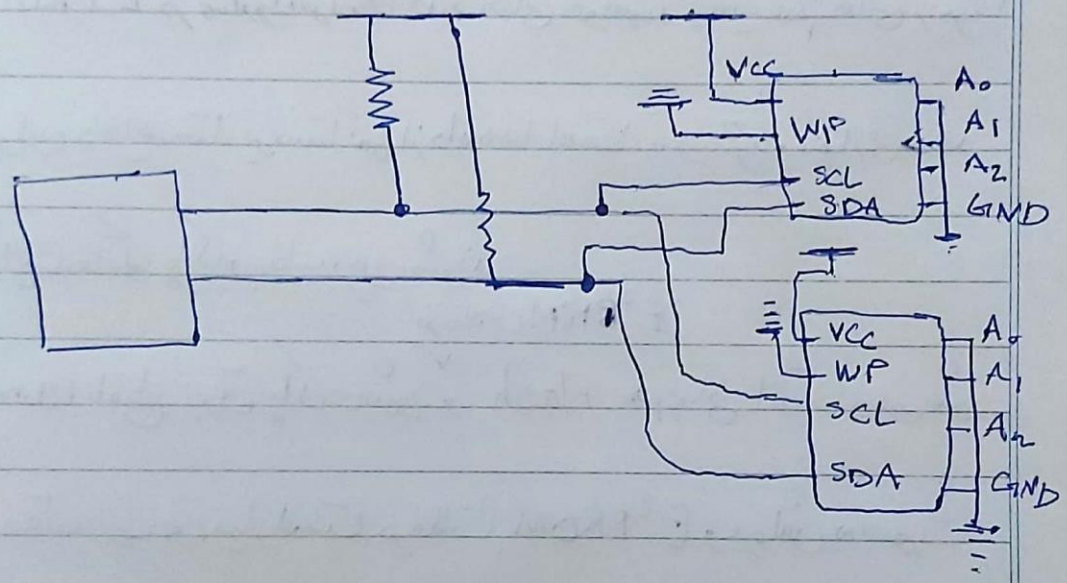
برخلاف E^2PROM

RAM با قطع برق پاک می‌شود و Flash هم‌زمانی استفاده می‌شود که مقادیر زیادی نیاز است برخلاف E^2PROM و حافظه‌اش به صورت بلوک داده‌ها پاک می‌شوند.

پرسش ۲: این که بلاک را بخوانیم و مقادیر مورد نیاز را تغییر داده و دوباره آن بلاک را بنویسیم

پرسش ۳: $2^2 = 4 \rightarrow 4 \times 4KB = 16KB$

پرسش ۴



پرسش ۵ : قبل دستور خواندن بیت R/W ۱ است

و اگر برای خواندن باشد قبل آن Dumy write داریم

در نوشتن با فرمت هم خوانی دارد

start - 7 - mode - Ack - 8 - Ack - ... - Stop

در خواندن هم همخوانی دارد

start - 7 - mode - 8 - Ack - stop - start - 7 - mde - Ack - 8 - Ack - ... - Stop

Farhang Gostar

پرسش ۶ : در برد آردوینو است پیکربندی و تولید کلاک نیز بالا است

$10 \times 10^9 / 29 = 345$

پرسش ۷) منظور کد مورد نظر داخل کدهای هست

begin : ارتباط را آغاز می کند

setclock : تغییر فرکانس

beginTransmission : ارتباط را برای شروع ارسال داده آغاز می کند

write : داده را می نویسد

endTransmission : ارتباط را تمام می کند

requestFrom : برای درخواست خواندن داده

available : تعداد بیت های قابل دریافت را می دهد

read : داده را می خواند

با توابع تعریف شده در بالای کد برای هندل کردن استفاده میشود
با زدن استپ و زمان آن و زدن دکمه منفی زمان ست میشود

```
#include <LiquidCrystal.h>
#include <Keypad.h>
#include <Wire.h>

#define RS 13
#define  E 12
#define D4 11
#define D5 10
#define D6 9
#define D7 8

const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
```

```
  {'7','8','9', 'S'},
  {'4','5','6', 'D'},
  {'1','2','3', '-'},
  {'G','0','=', '+'}
};
byte rowPins[ROWS] = {22, 23, 24, 25};
byte colPins[COLS] = {26, 27, 28, 29};

const byte LED_P[4] = {4, 5, 6, 7};

Keypad keypad = Keypad(makeKeymap(keys),
rowPins, colPins, ROWS, COLS);
LiquidCrystal lcd( RS, E, D4, D5, D6, D7 );

byte times[4] = {3, 2, 4, 2};
byte timeChange ;
int state ;
int level ;
```

```
String str ;
bool active ;
long tic ;
long toc ;
float level_time ;
float remain_time;
float total_time ;

void SAVE_STATE();
void LOAD_STATE();
void UPDATE();
void CALCULATE_TIME();
void eeprom_write(uint16_t memory_address,
uint8_t* data, int _size);
void eeprom_read(uint16_t memory_address,
uint8_t* data, int _size);

void setup() {
```

```
Serial.begin(9600);
Wire.begin();
pinMode(LED_P[0], OUTPUT);
pinMode(LED_P[1], OUTPUT);
pinMode(LED_P[2], OUTPUT);
pinMode(LED_P[3], OUTPUT);
lcd.begin(16, 2);
lcd.clear();
lcd.setCursor(0, 0);
str = "";
state = 2 ;
remain_time = times[0];
level = 0 ;
tic = millis();
toc = millis();
level_time = 3;
```

```
  active = false ;

  LOAD_STATE();

}



void loop() {
  char ch = keypad.getKey();
  if (ch) {
    // states : 0->finished , 1->paused , 2->idle , 3->active ,  4->change
    if (state==2 && (ch<='4' && ch>='1')) {
      state = 4; //change time
      str="";
      timeChange = ch -'0';
    } else if (state==2 && ch=='=') {
      state = 3;
```

```
  active = true;
  tic = millis();
  level = 0;
  level_time = times[0];
} else if (state==0 && ch=='=') {
  state = 2;
  for (int i = 0; i < 4; i++)
    digitalWrite(LED_P[i], LOW);
} else if (state==1 && ch=='=') {
  state = 3;
  level_time = remain_time;
  active = true;
  tic = millis();
} else if (state==4 && ch=='-') {
  state = 2;
  times[timeChange-1] = str.toInt();
} else if (state==3 && ch=='=') {
```

```
      state = 1;
      level_time = remain_time;
      active = false;
    } else if (state==4) {
      str += ch;
    }
  }

  UPDATE();
  CALCULATE_TIME();
}

void CALCULATE_TIME(){
  long now = millis();
  if ((now - toc) > 200) {
    SAVE_STATE();
    toc = now;
```

```
  }


  if (active) {
    remain_time = ((float)level_time-((float)(millis()-
tic)/1000));
      total_time = remain_time;
      for (int i = level+1 ; i < 4; i++){
        total_time += times[i];
      }
    if (remain_time <= 0) {
      level = (level+1)%4;
      tic = millis();
      level_time = times[level];
      remain_time = 0;
      if (level==0) {
        for (int i = 0; i < 4; i++)
          digitalWrite(LED_P[i], HIGH);
        active = false;
```

```
        state = 0;
      }
    }
  }
   delay(50);
}

void UPDATE(){

  lcd.clear();
  lcd.setCursor(0, 0);

  for (int i = 0; i < 4; i++)
    digitalWrite(LED_P[i], LOW);

  if (state==0) {
    for (int i = 0; i < 4; i++)
```

```
    digitalWrite(LED_P[i], HIGH);
} else if (state==1 || state==3 ) {
    digitalWrite(LED_P[level], HIGH);
    if (state==3) {
      active = true;
    }
}

if(state==0){
  lcd.print("FINISHED...");
} else if (state==1){
  lcd.print("PAUSED...");
  lcd.setCursor(0, 1);
  lcd.print("total time:" +(String) total_time);
} else if (state==3){
  lcd.print("ACTIVE...");
  lcd.setCursor(0, 1);
```

```
    lcd.print("total time:" +(String) total_time);
  } else if (state==2){
    lcd.print("IDLE...");
  } else if (state==4){
    lcd.print("CHANGE...");
    lcd.setCursor(0, 1);
    lcd.print("enter time :" + str);
  }

}

void SAVE_STATE(){
  uint8_t wr_data[8] = {0};
  for (int i = 0; i < 4; i++){
    wr_data[i] = (uint8_t) times[i];
  }
  wr_data[4] = (uint8_t) state;
```

```
  wr_data[5] = (uint8_t) level;
  wr_data[6] = (uint8_t) remain_time;
  eeprom_write(40, wr_data, 8);
}

void LOAD_STATE(){
  uint8_t re_data[8] = {0};
  eeprom_read(40, re_data, 8);
  if (re_data[0] != 255){
    for (int i = 0; i < 3; i++){
      times[i] = re_data[i];
    }
    state = re_data[4];
    level = re_data[5];
    remain_time = re_data[6];
    level_time =  times[level];
    total_time = remain_time;
```

```
    for (int i = level+1 ; i < 4; i++){
      total_time += times[i];
    }
    UPDATE();
  }
}

void eeprom_write(uint16_t memory_address,
uint8_t* data, int _size) {
  Wire.beginTransmission(0b1010000);
  Wire.write((uint8_t)((memory_address & 0xFF00)
>> 8));
  Wire.write((uint8_t)((memory_address & 0x00FF)
>> 0));
  for (int i = 0; i < _size; i++) {
    Wire.write(data[i]);
  }
  Wire.endTransmission();
```

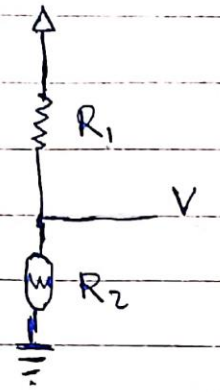
```
  delay(200);
}


void eeprom_read(uint16_t memory_address,
uint8_t* data, int _size) {
  Wire.beginTransmission(0b1010000);
  Wire.write((uint8_t)((memory_address & 0xFF00)
>> 8));
  Wire.write((uint8_t)((memory_address & 0x00FF)
>> 0));
  Wire.endTransmission();
  Wire.requestFrom(0b1010000, _size);
  for (int i = 0; i < _size; i++) {
    data[i] = Wire.read();
  }
}
```

آزمایش ۸

آزمایش ۸

پرسش ۱:

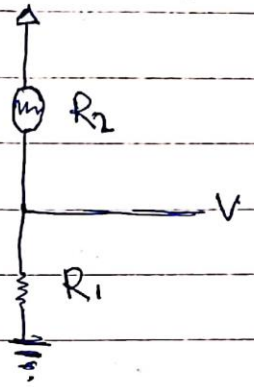


$V = \frac{R_2}{R_1+R_2} 5$

$\rightarrow R_2 = \frac{VR_1}{5-V}$

light ↑ R_2 ↓ V ↓

// ↓ R_2 ↑ V ↑



$V = \frac{R_1}{R_1+R_2} 5$

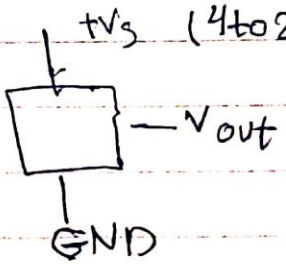
$R_2 = \frac{5R_1 - VR_1}{V}$

light ↑ R_2 ↓ V ↑

// ↓ R_2 ↑ V ↓

پرسش ۲:

LM35



دما را از -55 to +150 اندازه‌گیری می‌کند

ولتاژ آن از 5 تا -0.5 است

Farhang Tostar

پرسش ۳ : خیر I^2C

پرسش ۴ : بله هست full duplex امکان ارتباط دو طرفه دارند است که به طور همزمان باشد

پرسش ۵ : MISO:50 MOSI:51 SCK:52 SS:53

پرسش ۶ : باید خط slave مربوطه را LOW کرد - با تایپ

پرسش ۷ : Master کلاک را تعیین می کند

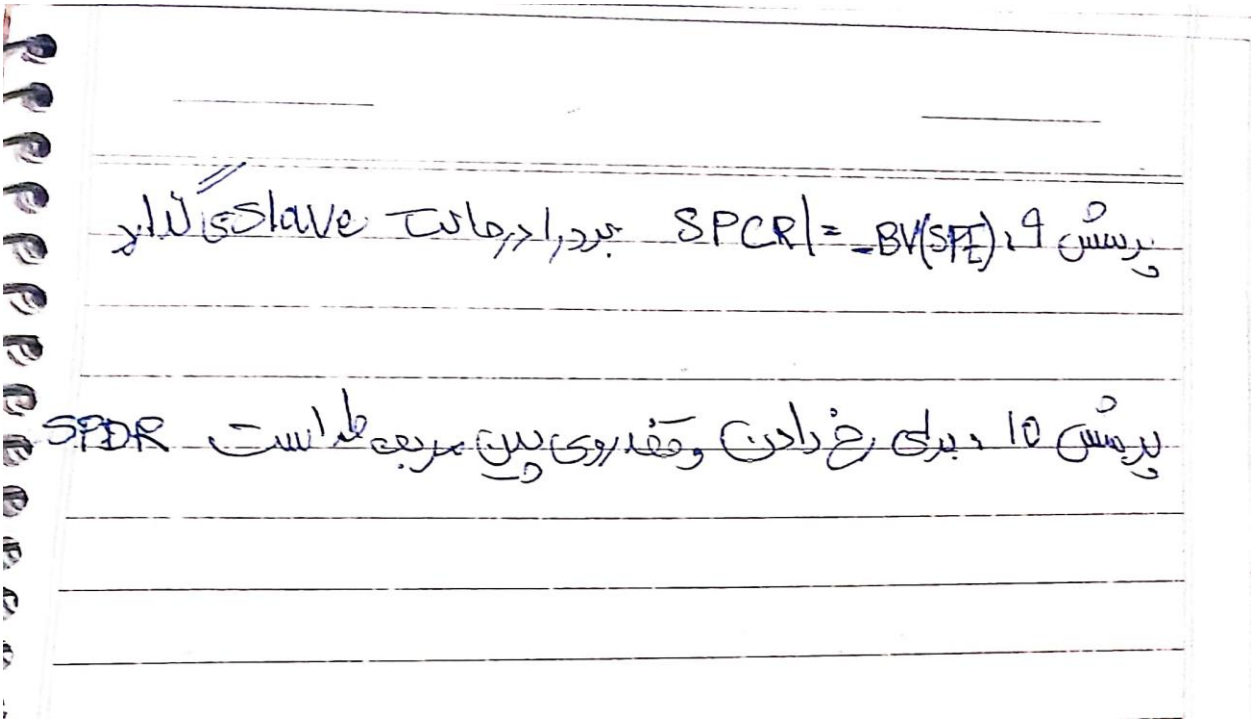
پرسش ۸ :

begin() : ارتباط را با ست کردن پین SS و SCK و MOSI به صورتی و low کردن MOSI و SCK و high کردن SS شروع می کند

setclockdivider : باید استفاده شود چون مرجع گفته است

transfer() : دیتا می گیرد و می دهد

attach : ISR برای آن پین مربوطه تنظیم می کند interrupt



کد قسمت ۱ و ۳ به درستی اجرا می شود در حین ارایه کد اجرا قسمت ۵ غلط بود که علت آن مقدار غیر ولید مقاومت بو که به اشتباه آن را مقدار دهی کرده بودم که بعد از تصحیح آن درست شد توضیحات مربوط به کد بخش ها را سر جلسه ارایه دادم در کنار این فایل های آزمایش را آپلود میکنم.

۱: دو ماژول مستر و اسلیو داریم که کد های آن در زیر آورده شده مستر اسم و شماره دانشجویی مرا میفرستد و اسلیو آن را دریافت و چاپ میکند

```
/// part1                    //Master
#include <SPI.h>

#define MESSAGE   "AMIRREZA: 9831126:)\n"

#define SS 46

void setup() {
  Serial.begin(9600);
  Serial.println("MASTER...");
  pinMode(SS, OUTPUT);
  digitalWrite(SS, HIGH);
  SPI.begin();
}
```

```
void loop() {
  digitalWrite(SS, LOW);
  delay(10);

  for (const char *p = MESSAGE ; char c = *p; p++) {
    SPI.transfer(c);
    Serial.print(c);
    delay(5);
  }
  Serial.println();

  digitalWrite(SS, HIGH);

  delay(1000);
}
```

```
/// SLAVE
#include <SPI.h>
#define MISO 50
#define MOSI 51
#define SCK 52
#define SS 53

volatile int ind = 0;
volatile bool isFinished = false;
char MESSAGE[100];

void setup() {
  Serial.begin(9600);
  Serial.println("SLAVE...");
```

```
  pinMode(MOSI, INPUT);
  pinMode(MISO, OUTPUT);
  pinMode(SCK, INPUT);
  pinMode(SS, INPUT_PULLUP);

  SPCR |= _BV(SPE);
  SPI.attachInterrupt();
}

void loop() {
  if (isFinished) {
    Serial.println(MESSAGE);
    ind = 0;
    isFinished = false;
  }
}
```

```
ISR (SPI_STC_vect)
{
  byte c = SPDR;
  if (ind < sizeof MESSAGE) {
    MESSAGE[ind++] = c;

    if (c == '\n') {
      isFinished = true;
    }
  }
}
```

قسمت ۳: مستر دوپیام که اولی حاوی اسم و دومی های + اسم هست هر پیام را به دو اسلیو جداگانه میفرستد و اسلیو ها آن را دریافت و چاپ میکنند علت ارسال دو پیام متفاوت به دو اسلیو آن است که گاهی اوقات اسلیو نمیخواهد تمام اطلاعات را بگیرد کد اسلیو بر روی دو برد آپلود میشود

```
/// part 3 MASTER
#include <SPI.h>

#define MESSAGE0   "Amirreza\n"
#define MESSAGE1   "Hi Amirreza:)\n"

#define SS0 45
#define SS1 46

void setup() {
  Serial.begin(9600);
  Serial.println("MASTER...");
```

```
  pinMode(SS0, OUTPUT);
  digitalWrite(SS0, HIGH);

  pinMode(SS1, OUTPUT);
  digitalWrite(SS1, HIGH);

  SPI.begin();
}

void sender(const char *message) {
  for (const char *p = message ; char c = *p; p++) {
    SPI.transfer(c);
    Serial.print(c);
    delay(5);
  }
  Serial.println();
}
```

```
void loop() {
  digitalWrite(SS0, LOW);
  sender(MESSAGE0);
  digitalWrite(SS0, HIGH);
  delay(100);

  digitalWrite(SS1, LOW);
  sender(MESSAGE1);
  digitalWrite(SS1, HIGH);
  delay(100);
}
```

```
/// SLAVE
#include <SPI.h>
#define MISO 50
#define MOSI 51
#define SCK 52
#define SS 53

volatile int ind = 0;
volatile bool isFinished = false;
char MESSAGE[100];

void setup() {
  Serial.begin(9600);
```

```
  Serial.println("SLAVE...");

  pinMode(MOSI, INPUT);
  pinMode(MISO, OUTPUT);
  pinMode(SCK, INPUT);
  pinMode(SS, INPUT_PULLUP);

  SPCR |= _BV(SPE);
  SPI.attachInterrupt();
}

void loop() {
  if (isFinished) {
    Serial.println(MESSAGE);
    ind = 0;
    isFinished = false;
  }
```

```
}

ISR (SPI_STC_vect)
{
  byte c = SPDR;
  if (ind < sizeof MESSAGE) {
    MESSAGE[ind++] = c;

    if (c == '\n') {
      isFinished = true;
    }
  }
}
```

قسمت ۵: مستر دما و شدت نور را از پین های آنالوگ خوانده و دما را به یک اسلیو و شدت نور را به اسلیو دیگری میفرستد علت فرستاده نشدن تمام اطلاعات به بک اسلیو آن است که آن اسلیو فقط دما یا شدت نور را میخواهد برای هر اسلیو کدی جدا زده شده است

```
/// part 5                /// MASTER
#include <SPI.h>


#define SST 45
#define SSL 46
#define TP A8
#define LP A1
int LPV;
uint8_t mLPV;
int TPV;
uint8_t mTPV;
```

```
void setup() {
  Serial.begin(9600);
  Serial.println("MASTER...");

  pinMode(SST, OUTPUT);
  digitalWrite(SST, HIGH);
  pinMode(SSL, OUTPUT);
  digitalWrite(SSL, HIGH);
  pinMode(TP, INPUT);
  pinMode(LP, INPUT);
  SPI.begin();
}

void loop() {
  delay(1000);
  LPV = analogRead(LP);
  mLPV = map(LPV, 0, 1023, 0, 100);
```

```
digitalWrite(SSL, LOW);
SPI.transfer (mLPV);
Serial.print("Light: ");
Serial.print(mLPV);
Serial.println("%");
delay(5);
digitalWrite(SSL, HIGH);
delay(1000);
TPV = analogRead(TP);
mTPV = map(TPV, 0, 1023, 0, 500);
digitalWrite(SST, LOW);
SPI.transfer (mTPV);
Serial.print("Temp: ");
Serial.print(mTPV);
Serial.println(" C");
delay(5);
digitalWrite(SST, HIGH);
```

```
  Serial.println();
}


////////////// LIGHT SLAVE
#include <SPI.h>
#define MISO 50
#define MOSI 51
#define SCK 52
#define SS 53
int value;
volatile boolean isFinished = false;

void setup() {
  Serial.begin(9600);
  Serial.println("SLAVE Light...");
```

```
  pinMode(MOSI, INPUT);
  pinMode(MISO, OUTPUT);
  pinMode(SCK, INPUT);
  pinMode(SS, INPUT_PULLUP);

  SPCR |= _BV(SPE);
  SPI.attachInterrupt();
}

void loop() {
  if (isFinished) {
    Serial.print("Light: ");
    Serial.print(value);
    Serial.println(" %");
    isFinished = false;
  }
}
```

```
ISR (SPI_STC_vect)
{
  byte number = SPDR;
  value = (uint8_t)number;
  isFinished = true;
}
```

```
//////          TEMP SLAVE
#include <SPI.h>
#define MISO 50
#define MOSI 51
#define SCK 52
#define SS 53
int value;
volatile boolean isFinished = false;

void setup() {
  Serial.begin(9600);
  Serial.println("SLAVE Temperature...");
```

```
  pinMode(MOSI, INPUT);
  pinMode(MISO, OUTPUT);
  pinMode(SCK, INPUT);
  pinMode(SS, INPUT_PULLUP);

  SPCR |= _BV(SPE);
  SPI.attachInterrupt();
}

void loop() {
  if (isFinished) {
    Serial.print("Temprature: ");
    Serial.print(value);
    Serial.println(" C");
    isFinished = false;
  }
}
```

```
ISR (SPI_STC_vect)
{
  byte number = SPDR;
  value = (uint8_t)number;
  isFinished = true;
}
```