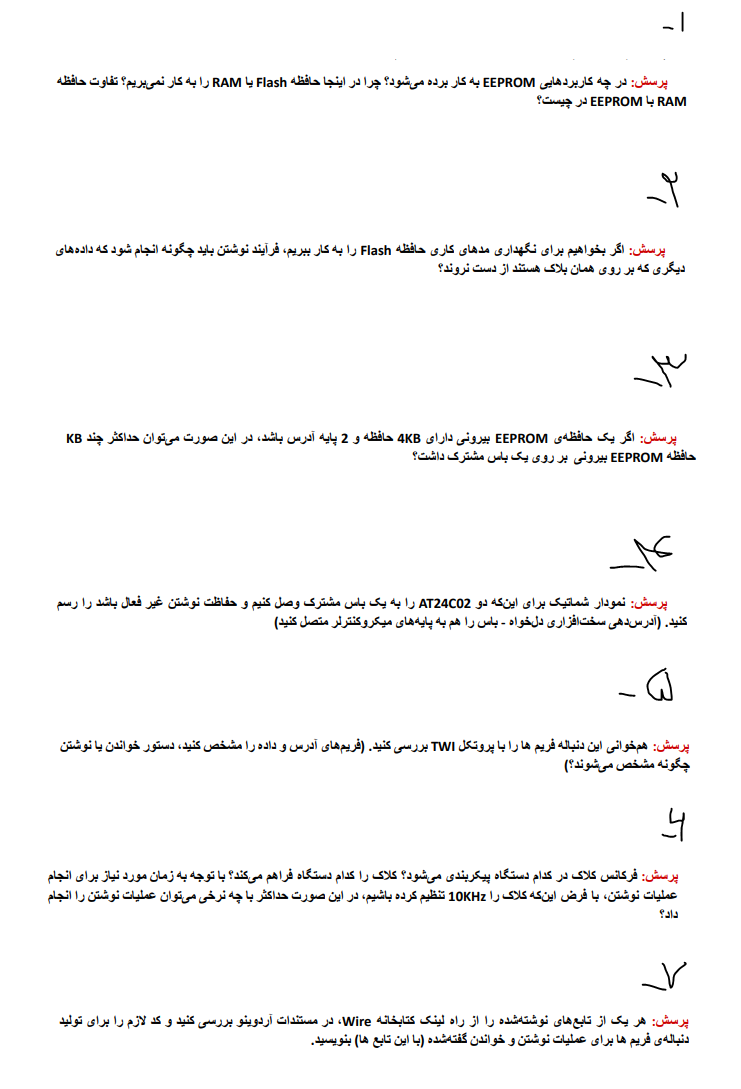
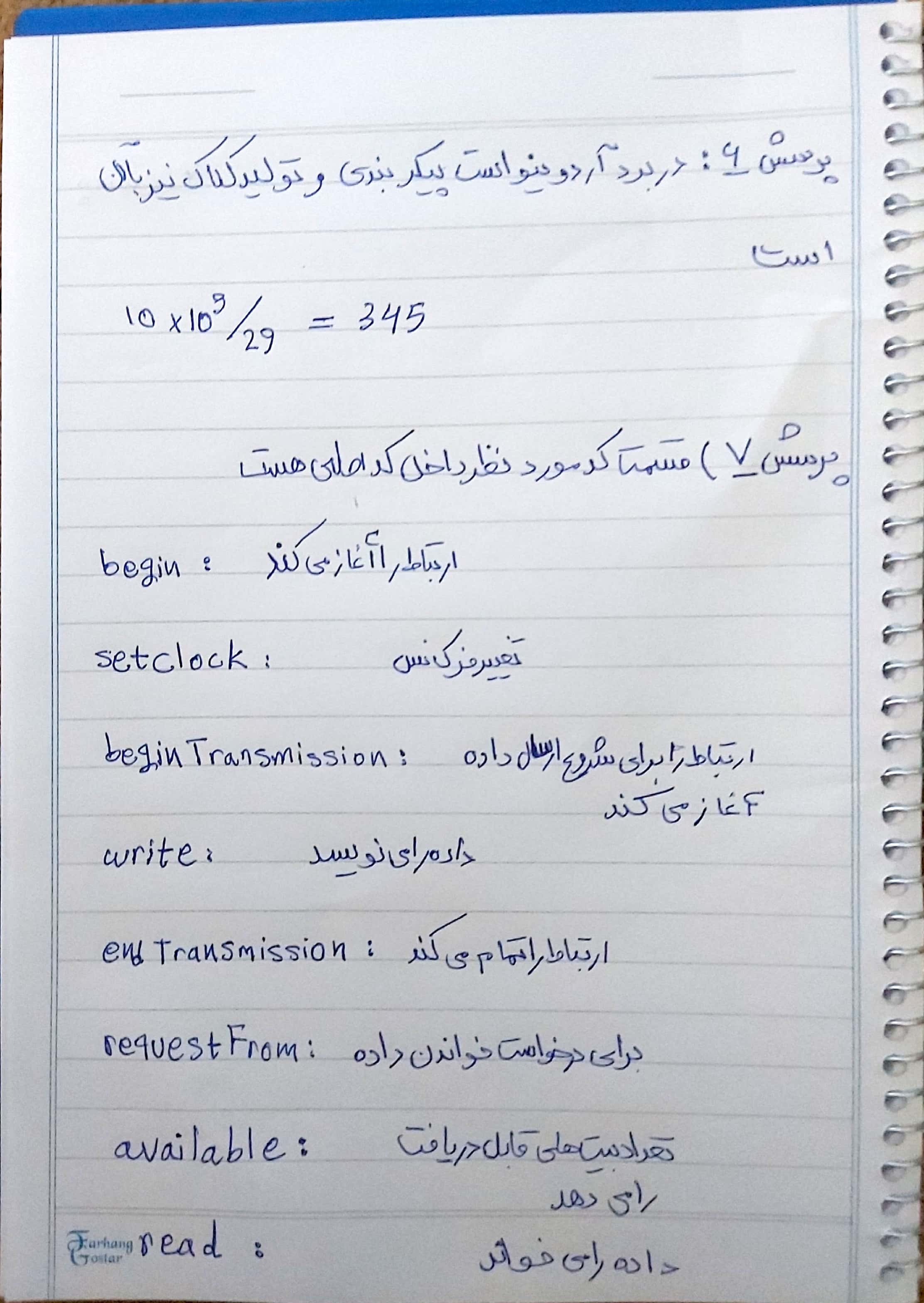
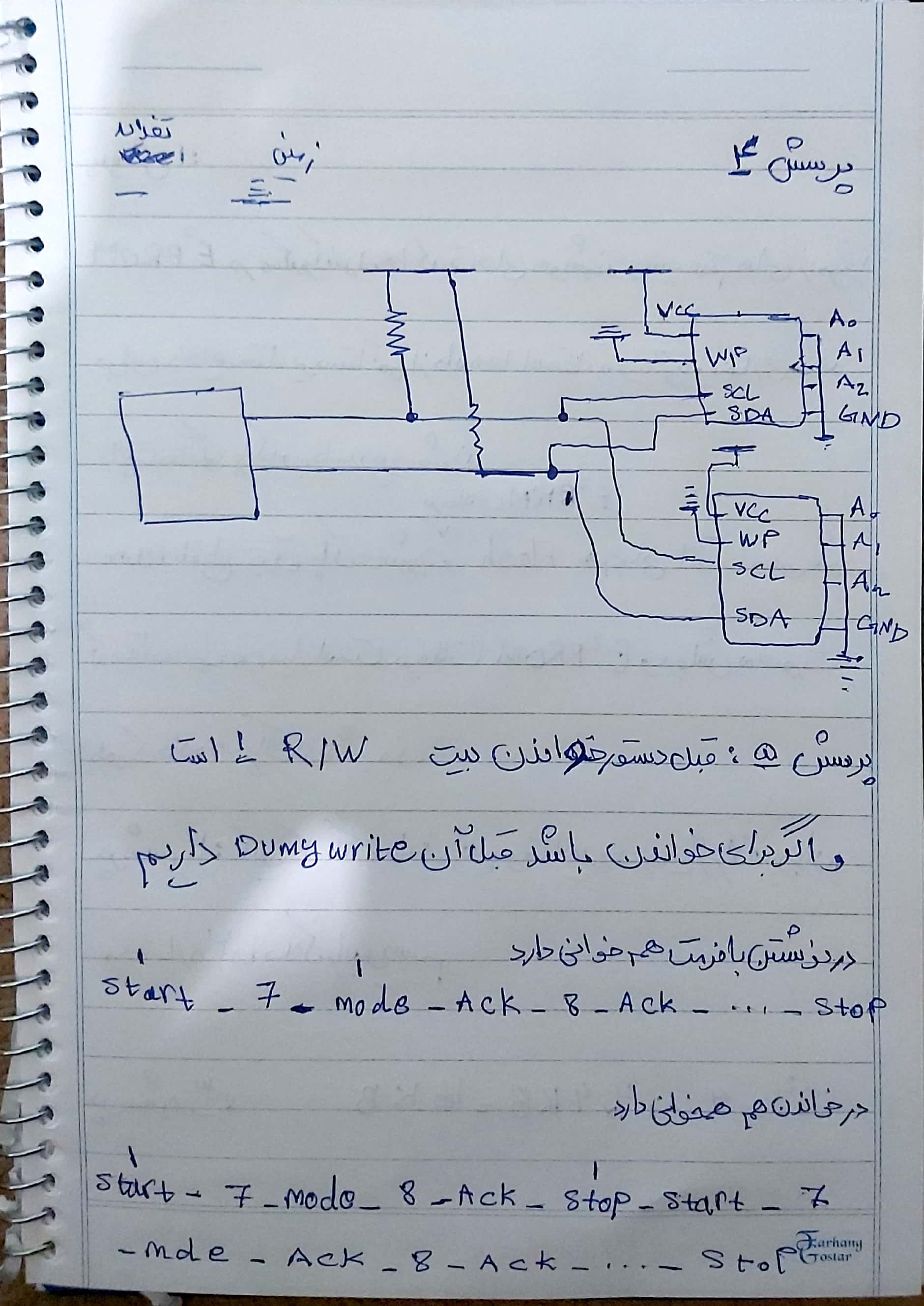
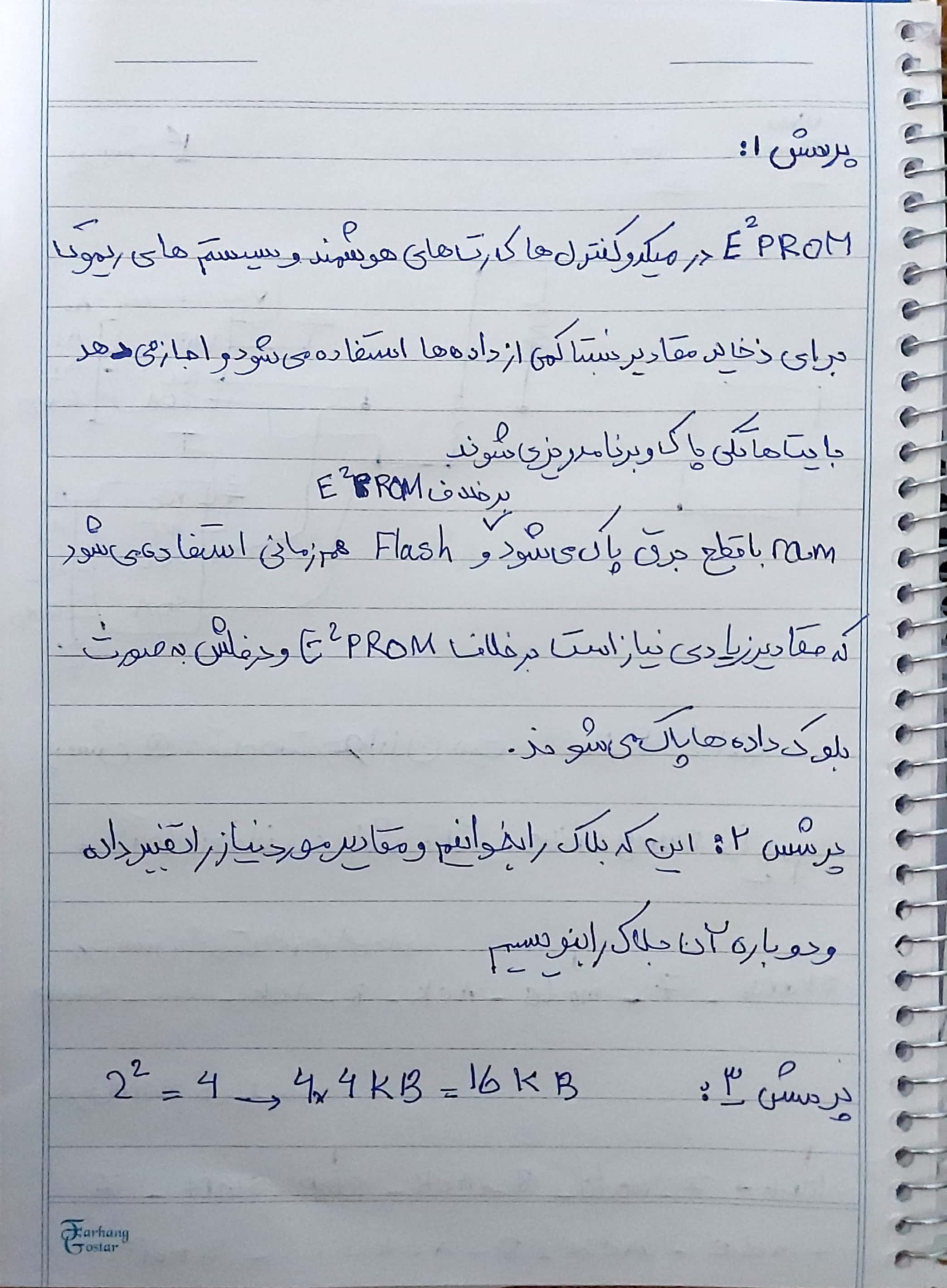
به نام خداوند مهربان

آزمایش 7

امیررضارجبی 9831126





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);

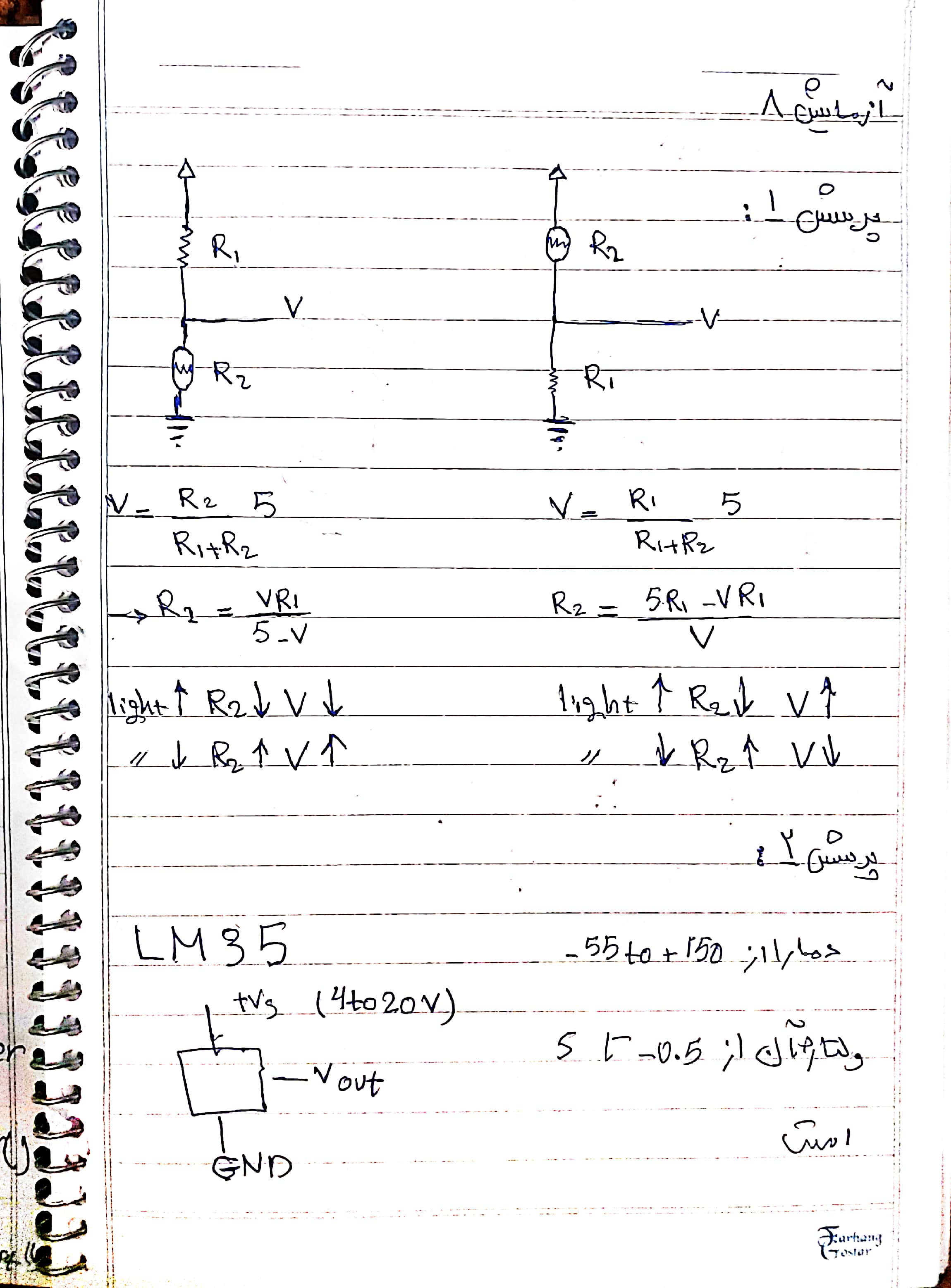
for (int i = 0; i < \_size; i++) {

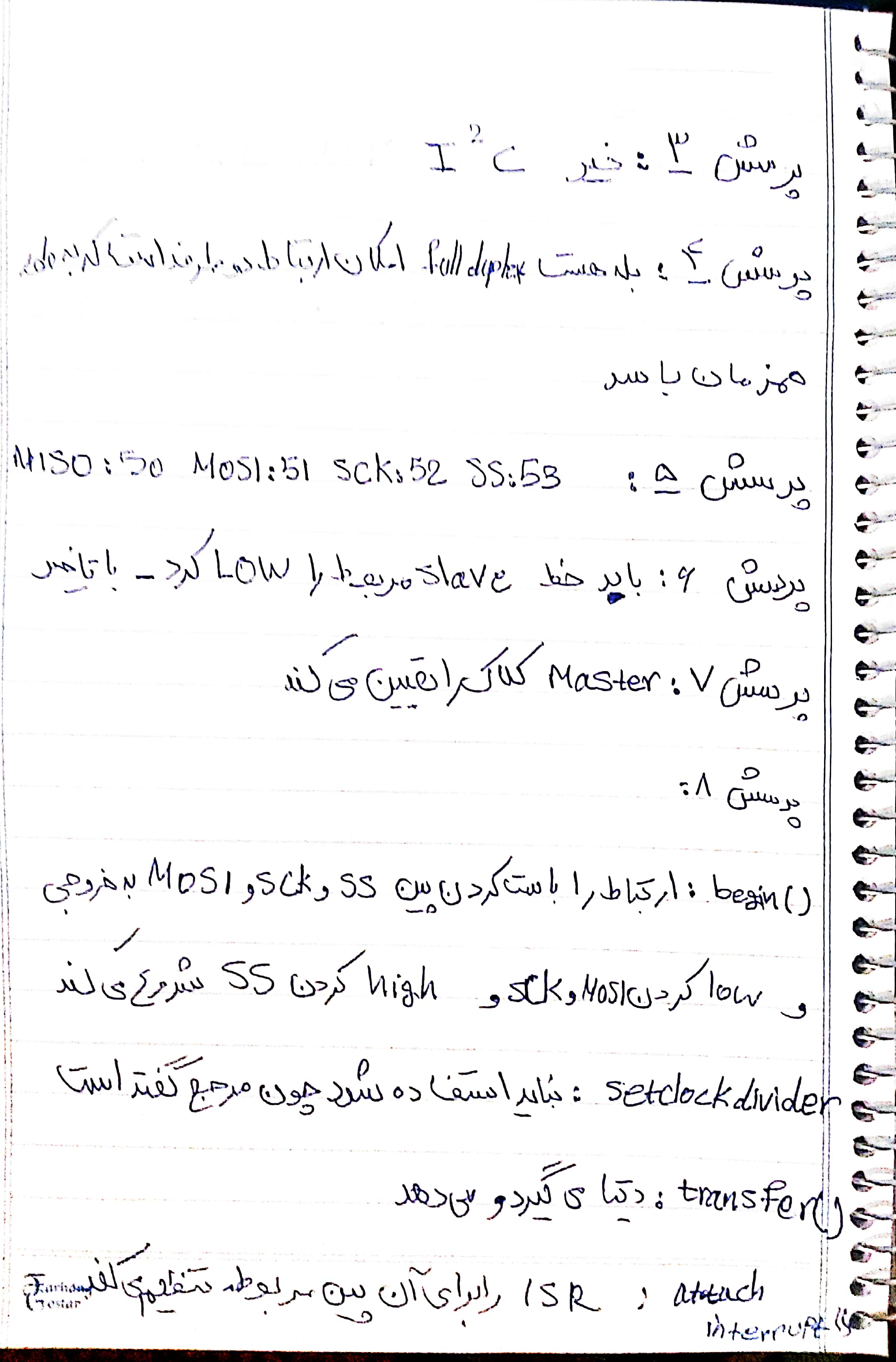
data[i] = Wire.read();

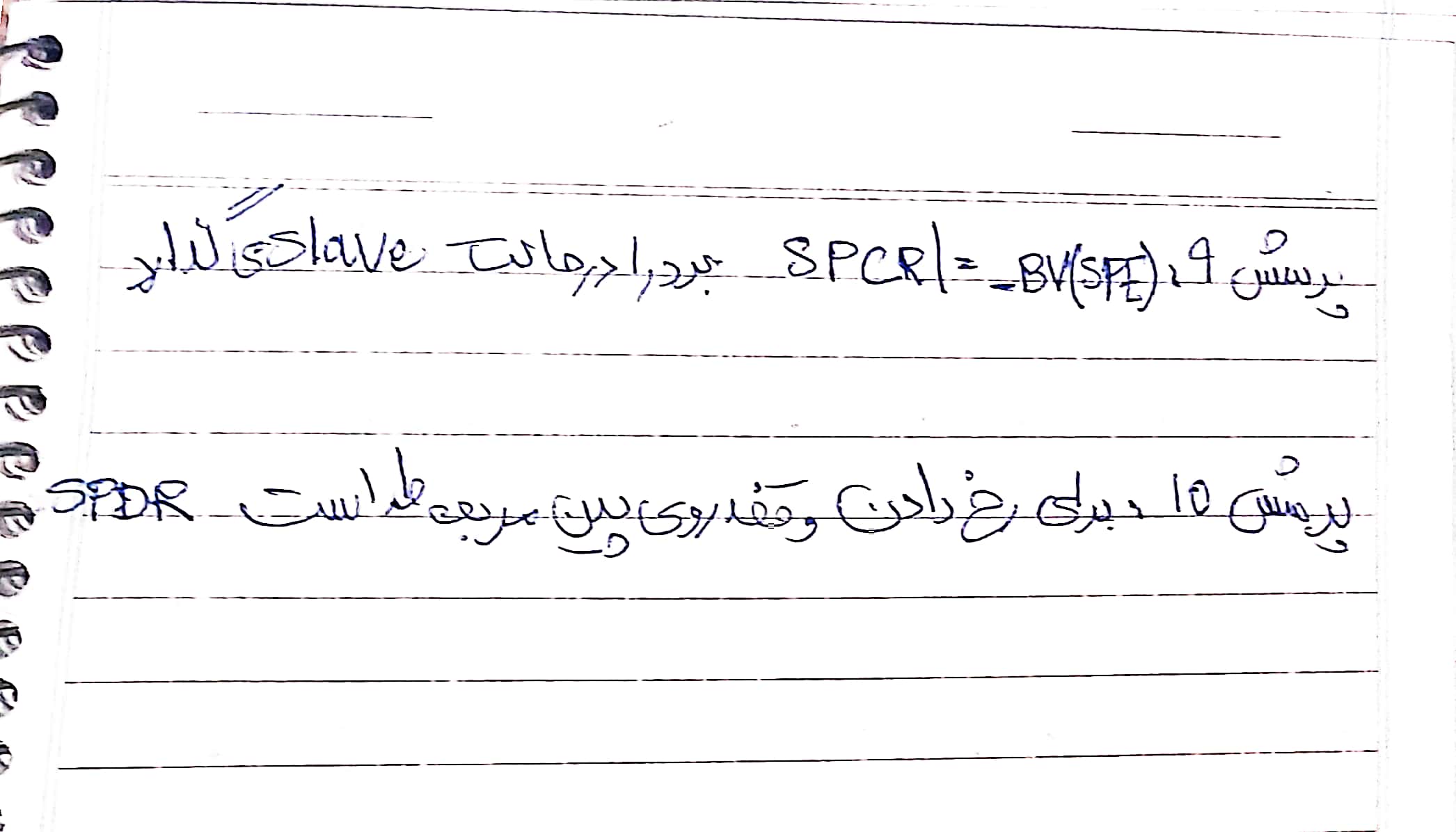
}

}

آزمایش 8







کد قسمت 1 و 3 به درستی اجرا می شود در حین ارایه کد اجرا قسمت 5 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.

1: 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 3: 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 5: 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;

}