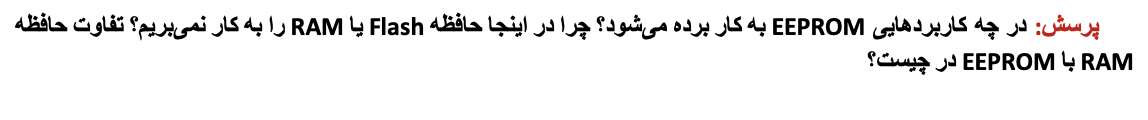
**به نام خدا**

پیش گزارش آزمایش هفتم آزمایشگاه ریزپردازنده

امیرپارسا سلمان خواه

۹۸۳۱۰۳۴



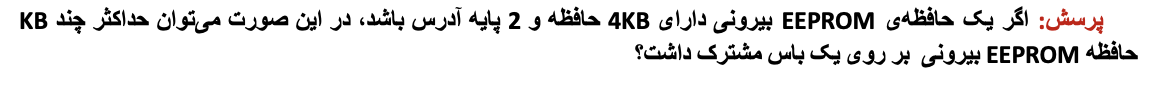
از eeprom برای نگه داری داده هایی که به طور خیلی سریعی تغییر نمی کنند استفاده می شود. به عنوان مثال پارامتر های مورد نیاز برای کانفیگ کردن یک دستگاه در یک eeprom ذخیره می شوند تا در صورت خاموش شدن و روشن شدن دستگاه، این پارامتر ها را بتوان بازیابی کرد. همچنین در bios کامپیوتر ها و برای بالا اوردن کامپیوتر، اطلاعاتی نظیر آدرس سیستم عامل در eeprom نگه داری می شود تا برنامه ی bootstrap بتواند آن را پیدا و اجرا کند.

از حافظه ی فلش معمولا زمانی استفاده می شود که به حجم وسیع تری از داده ها نیاز است. در واقع حافظه ی فلش خود یک نوع eeprom است اما با این تفاوت که اطلاعات را نه به شکل بایت بایت بلکه به شکل بلاک بلاک منتقل می کند و این باعث می شود که سریع تر باشد. همچنین می توان بار ها روی حافظه های فلش نوشت. این در حالی است که در eeprom ها تعداد محدودی عملیات نوشتن را می توان انجام داد.

حافظه ی ram یک حافظه ی volatile است و با قطع شدن برق، اطلاعات آن از بین می رود. اما eeprom یک حافظه ی non volatile است و با قطع برق اطلاعات آن از بین نمی رود.

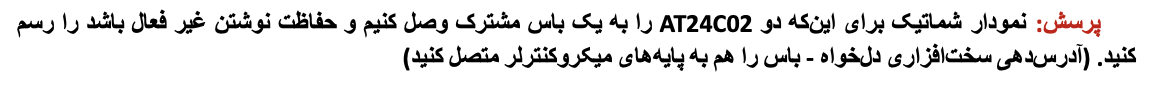


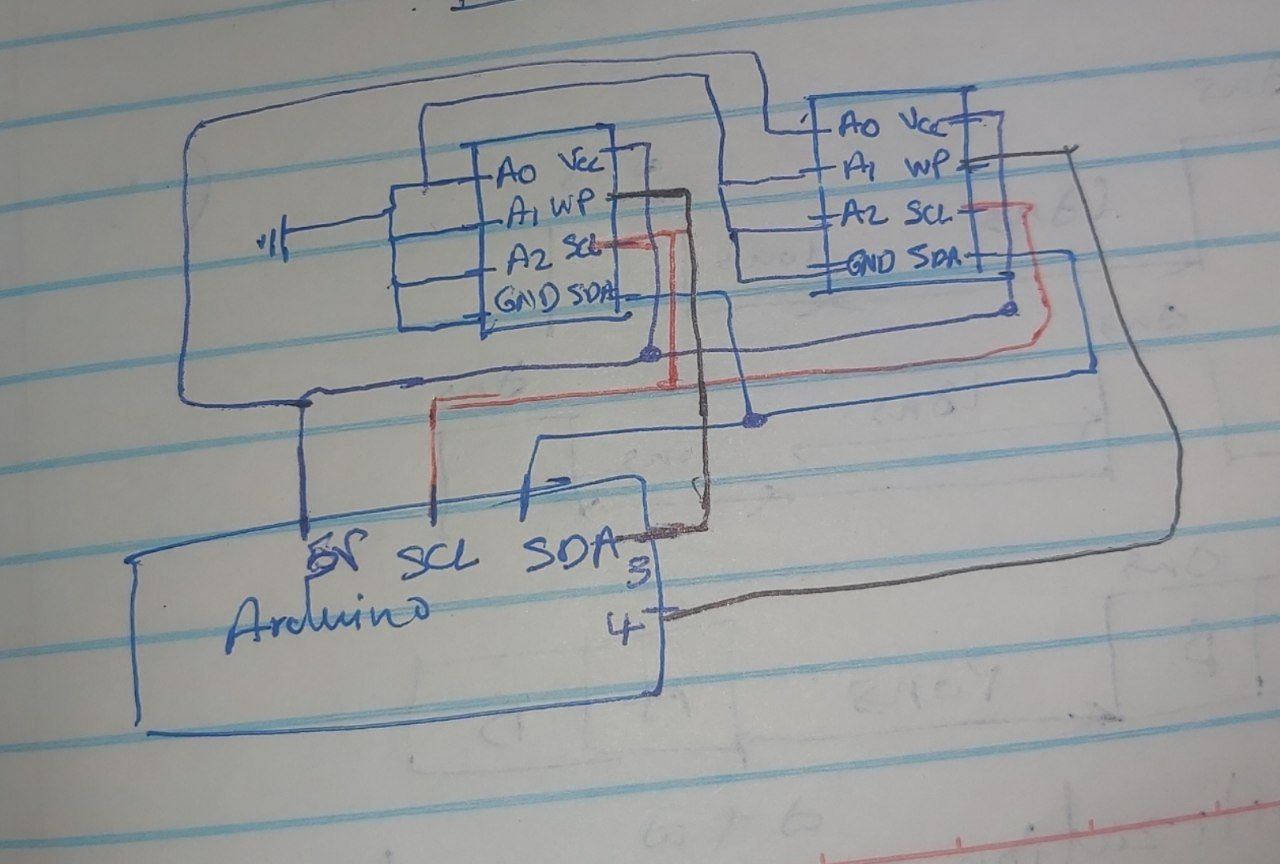
باید ابتدا کل بلاک خوانده شود و سپس بخشی از آن که می خواهیم تغییر کند را تغییر دهیم و سپس دوباره آن را بر روی حافظه بنویسیم.

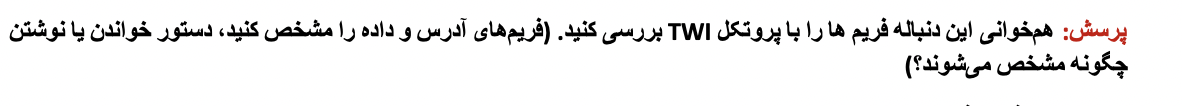


از آن جا که ۲ پایه ی آدرس داریم، پس می توانیم ۴ تا از این حافظه را روی یک باس داشته باشیم.

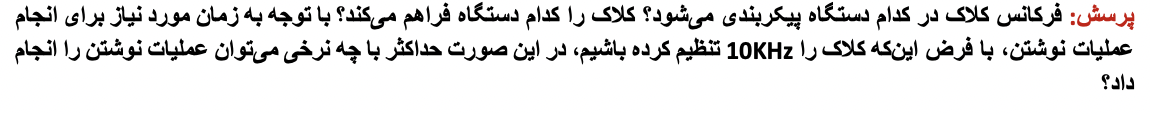
بنابراین ۴ تا ۴ کیلوبایت یعنی ۱۶ کیلوبایت حافظه خواهیم داشت.



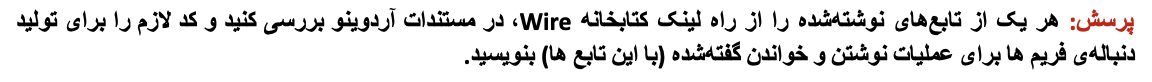




دنباله ی فریم ها کاملا با پروتکل I2C مطابق است. به این شکل که ابتدا یک بیت شروع فرستاده می شود و بعد از آن آدرس slave و بیت خواندن یا نوشتن فرستاده می شود. سپس ack از slave دریافت می شود و بعد از آن ارسال دیتا شروع می شود و یک بایت یک بایت دیتا فرستاده می شود و ack دریافت می شود. در اینجا دیتا ها آدرس خانه ی حافظه و مقدار مورد نظر برای نوشتن است. در پایان نیز بیت پایانی فرستاده می شود.



فرکانس کلاک در master که در اینجا همان آردویینو است تامین می شود.

برای ارسال هر بسته تقریبا به ۴۰ بیت نیاز است و بعد از آن به ۵ میلی ثانیه زمان برای نوشتن دیتا روی eeprom نیاز است. ارسال ۴۰ بیت دیتا نیازمند ۴۰ کلاک است و چون هر کلاک ۰.۱ میلی ثانیه است، ۴ میلی ثانیه برای ارسال بسته طول می کشد. بنابراین با گذشت ۹ میلی ثانیه می توان یک عملیات نوشتن را به شکل کامل انجام داد و سراغ عملیات بعدی رفت. 

begin()

اینترفیس I2C را راه اندازی می کند.  
setClock()

کلاک را تنظیم می کند.  
beginTransmission()

با دریافت آدرس، انتقال دیتا را آغاز می کند.

write()

بسته های دیتا را ارسال می کند.  
endTransmission()

ارسال اطلاعات را به إتمام می رساند.

requestFrom()

با دریافت آدرس از slave درخواست اطلاعات می کند.  
available()

در صورتی که دیتایی دریافت شده باشد، مقدار ۱ منطقی را بر می گرداند.  
read()

دیتای دریافت شده را بر می گرداند.

برای ارسال و دریافت اطلاعات می توان از دو تابع زیر استفاده کرد:

void writeI2CByte(byte data\_addr, byte data){

digitalWrite(WRITE\_PROTECT, LOW);

Wire.beginTransmission(ADDR);

Wire.write(data\_addr);

Wire.write(data);

Wire.endTransmission();

digitalWrite(WRITE\_PROTECT, HIGH);

delay(5);

}

byte readI2CByte(byte data\_addr){

byte data = NULL;

Wire.beginTransmission(ADDR);

Wire.write(data\_addr);

Wire.endTransmission();

Wire.requestFrom(ADDR, 1);

delay(5);

if(Wire.available()){

data = Wire.read();

}

return data;

}

کد ماشین لباس شویی:

#include <MsTimer2.h>

#include <Keypad.h>

#include <LiquidCrystal\_I2C.h>

#include <Wire.h>

const byte ROWS = 4;

const byte COLS = 4;

char keys[ROWS][COLS] = {

{'1','2','3','A'},

{'4','5','6','B'},

{'7','8','9','C'},

{'\*','0','#','D'}

};

#define KEYPAD1 6

#define KEYPAD2 13

#define KEYPAD3 12

#define KEYPAD4 11

#define KEYPAD5 10

#define KEYPAD6 9

#define KEYPAD7 8

#define KEYPAD8 7

#define WRITE\_PROTECT 2

#define ADDR 0x57

byte rowPins[ROWS] = {KEYPAD1, KEYPAD2, KEYPAD3, KEYPAD4}; //connect to the row pinouts of the keypad

byte colPins[COLS] = {KEYPAD5, KEYPAD6, KEYPAD7, KEYPAD8}; //connect to the column pinouts of the keypad

LiquidCrystal\_I2C lcd(0x27, 16, 2);

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

byte const INIT\_STATE = 0;

byte const PREWASH\_STATE = 1;

byte const DETERGENT\_STATE =2;

byte const WATER\_STATE = 3;

byte const DRYING\_STATE = 4;

byte const CONFIG\_MENU\_STATE = 5;

byte const CONFIG\_ENTER\_TIME\_STATE = 6;

byte const FINISH\_STATE = 7;

byte readI2CByte(byte data\_addr);

byte prewash\_time = 10;

byte detergent\_time = 10;

byte water\_time = 10;

byte drying\_time = 10;

int input = 0;

byte state = INIT\_STATE;

long start\_time = 0;

byte remaining\_time = 0;

byte editing = 0;

byte paused = 0;

void setup() {

// put your setup code here, to run once:

Serial.begin(9600);

Wire.begin();

lcd.begin(); // Init LCD

lcd.display();

pinMode(WRITE\_PROTECT, OUTPUT);

// writeI2CByte(0, 10);

// writeI2CByte(1, 10);

// writeI2CByte(2, 10);

// writeI2CByte(3, 10);

// writeI2CByte(4, 0);

// writeI2CByte(5, 10);

// writeI2CByte(6, 0);

prewash\_time = readI2CByte(0);

detergent\_time = readI2CByte(1);

water\_time = readI2CByte(2);

drying\_time = readI2CByte(3);

state = readI2CByte(4);

remaining\_time = readI2CByte(5);

paused = readI2CByte(6);

lcd.clear();

switch (state){

case INIT\_STATE:

print\_at\_center("1) Start", 0);

print\_at\_center("2) Config", 1);

break;

case PREWASH\_STATE:

print\_at\_center("Prewash State", 0);

print\_at\_center(String(remaining\_time), 1);

break;

case DETERGENT\_STATE:

print\_at\_center("Detergent State", 0);

print\_at\_center(String(remaining\_time), 1);

break;

case WATER\_STATE:

print\_at\_center("Water State", 0);

print\_at\_center(String(remaining\_time), 1);

break;

case DRYING\_STATE:

print\_at\_center("Drying State", 0);

print\_at\_center(String(remaining\_time), 1);

break;

case FINISH\_STATE:

print\_at\_center("Done!", 0);

break;

}

start\_time = millis();

}

void loop() {

// put your main code here, to run repeatedly:

char key = keypad.getKey();

switch (state){

case INIT\_STATE:

if (key == '1'){

state = PREWASH\_STATE;

writeI2CByte(4, state);

remaining\_time = prewash\_time;

start\_time = millis();

}

else if (key == '2'){

state = CONFIG\_MENU\_STATE;

lcd.clear();

print\_at\_center("1)PREWASH 2)DET", 0);

print\_at\_center("3)WATER 4)DRY", 1);

}

break;

case PREWASH\_STATE:

if (paused == 0 && key == NO\_KEY && remaining\_time != remaining\_time - (millis() - start\_time) / 1000){

start\_time = millis();

remaining\_time -= 1;

lcd.clear();

print\_at\_center("Prewash State", 0);

print\_at\_center(String(remaining\_time), 1);

writeI2CByte(5, remaining\_time);

if (remaining\_time == 0){

state = DETERGENT\_STATE;

writeI2CByte(4, state);

remaining\_time = detergent\_time;

start\_time = millis();

}

}

else if (key == 'A'){

paused = 1 - paused;

writeI2CByte(6, paused);

if (paused == 0){

start\_time = millis();

}

}

break;

case DETERGENT\_STATE:

if (paused == 0 && key == NO\_KEY && remaining\_time != remaining\_time - (millis() - start\_time) / 1000){

start\_time = millis();

remaining\_time -= 1;

writeI2CByte(5, remaining\_time);

lcd.clear();

print\_at\_center("Detergent State", 0);

print\_at\_center(String(remaining\_time), 1);

if (remaining\_time == 0){

state = WATER\_STATE;

writeI2CByte(4, state);

remaining\_time = water\_time;

start\_time = millis();

}

}

else if (key == 'A'){

paused = 1 - paused;

writeI2CByte(6, paused);

if (paused == 0){

start\_time = millis();

}

}

break;

case WATER\_STATE:

if (paused == 0 && key == NO\_KEY && remaining\_time != remaining\_time - (millis() - start\_time) / 1000){

start\_time = millis();

remaining\_time -= 1;

writeI2CByte(5, remaining\_time);

lcd.clear();

print\_at\_center("Water State", 0);

print\_at\_center(String(remaining\_time), 1);

if (remaining\_time == 0){

state = DRYING\_STATE;

writeI2CByte(4, state);

remaining\_time = drying\_time;

start\_time = millis();

}

}

else if (key == 'A'){

paused = 1 - paused;

writeI2CByte(6, paused);

if (paused == 0){

start\_time = millis();

}

}

break;

case DRYING\_STATE:

if (paused == 0 && key == NO\_KEY && remaining\_time != remaining\_time - (millis() - start\_time) / 1000){

start\_time = millis();

remaining\_time -= 1;

writeI2CByte(5, remaining\_time);

lcd.clear();

print\_at\_center("Drying State", 0);

print\_at\_center(String(remaining\_time), 1);

if (remaining\_time == 0){

state = FINISH\_STATE;

writeI2CByte(4, state);

lcd.clear();

print\_at\_center("Done!", 0);

}

}

else if (key == 'A'){

paused = 1 - paused;

writeI2CByte(6, paused);

if (paused == 0){

start\_time = millis();

}

}

break;

case FINISH\_STATE:

if (key != NO\_KEY){

state = INIT\_STATE;

writeI2CByte(4, state);

lcd.clear();

print\_at\_center("1) Start", 0);

print\_at\_center("2) Config", 1);

break;

}

case CONFIG\_MENU\_STATE:

if (key == '1'){

lcd.clear();

print\_at\_center("Set Prewash time:",0);

print\_at\_center(String(prewash\_time),1);

editing = 1;

}

else if (key == '2'){

lcd.clear();

print\_at\_center("Set Det time:",0);

print\_at\_center(String(detergent\_time),1);

editing = 2;

}

else if (key == '3'){

lcd.clear();

print\_at\_center("Set Water time:",0);

print\_at\_center(String(water\_time),1);

editing = 3;

}

else if (key == '4'){

lcd.clear();

print\_at\_center("Set Drying time:",0);

print\_at\_center(String(drying\_time),1);

editing = 4;

}

if (key >= '1' && key <= '4'){

state = CONFIG\_ENTER\_TIME\_STATE;

}

break;

case CONFIG\_ENTER\_TIME\_STATE:

if (key >= '0' && key <= '9'){

input = input \* 10 + key - '0';

lcd.clear();

print\_at\_center(String(input),0);

}

else if (key == 'D'){

switch(editing){

case 1:

prewash\_time = input;

writeI2CByte(0, input);

break;

case 2:

detergent\_time = input;

writeI2CByte(1, input);

break;

case 3:

water\_time = input;

writeI2CByte(2, input);

break;

case 4:

drying\_time = input;

writeI2CByte(3, input);

break;

}

input = 0;

state = INIT\_STATE;

lcd.clear();

print\_at\_center("1) Start", 0);

print\_at\_center("2) Config", 1);

}

break;

default:

break;

}

}

void print\_at\_center(String message, byte row){

int col = (16 - message.length()) / 2;

lcd.setCursor(col, row);

lcd.print(message);

}

void writeI2CByte(byte data\_addr, byte data){

digitalWrite(WRITE\_PROTECT, LOW);

Wire.beginTransmission(ADDR);

Wire.write(data\_addr);

Wire.write(data);

Wire.endTransmission();

digitalWrite(WRITE\_PROTECT, HIGH);

delay(5);

}

byte readI2CByte(byte data\_addr){

byte data = NULL;

Wire.beginTransmission(ADDR);

Wire.write(data\_addr);

Wire.endTransmission();

Wire.requestFrom(ADDR, 1); //retrieve 1 returned byte

delay(5);

if(Wire.available()){

data = Wire.read();

}

return data;

}