#include <stDHT.h> //библиотека для работы с датчиком серии DHT

#include <Wire.h> //библиотека для работы с интерфейсов I2C

#include <LiquidCrystal\_I2C.h> //библиотека подключения дисплея через I2C

#include <TroykaMQ.h> //библиотека для работы с датчиками серии MQ

#include <EEPROM.h> //библиотека для работы с памятью

//номер вывода на плате для подключения кнопки перехода в меню

#define MENU\_BUTTON\_PIN 2

//номер вывода на плате для подключения синего светодиода

#define BLUE\_LED\_PIN 4

//номер вывода на плате для подключения желтого светодиода

#define YELLOW\_LED\_PIN 5

//номер вывода на плате для подключения красного светодиода

#define RED\_LED\_PIN 6

//номер вывода на плате для подключения зуммера

#define BUZZER\_PIN 7

//номер вывода на плате для подключения датчика температуры и влажности

#define DHT22\_SENSOR\_PIN 8

//номер вывода на плате для подключения реле

#define RELAY\_PIN 9

//номер вывода на плате для подключения резистивной клавиатуры

#define ANALOG\_KEYBOARD\_PIN A0

//номер вывода на плате для подключения датчика углекислого газа

#define MQ135\_SENSOR\_PIN A1

//номер вывода на плате для подключения датчика метана

#define MQ4\_SENSOR\_PIN A2

//время прогрева датчиков газа, равное 60с

#define WARMING\_UP\_TIME 60000

//временной интервал для вывода прогресса прогревания, равный 3с

#define WARMING\_UP\_INTERVAL 3000

//временной интервал для опроса датчиков, равный 2с

#define SENSOR\_POLLING\_INTERVAL 2000

//допустимое отклонение от значения АЦП при анализе нажатой кнопки

#define ERROR 20

//количество аналоговых кнопок

#define ANALOG\_BUTTONS\_NUM 4

//объявляем объект библиотеки, указывая параметры дисплея

LiquidCrystal\_I2C display(0x27, 20,4);

//объявляем  объект библиотеки, указывая модель датчика температуры

DHT sensorDHT22(DHT22);

//объявляем  объект библиотеки, указывая пин подключения к arduino

MQ4 sensorMQ4(MQ4\_SENSOR\_PIN);

//объявляем  объект библиотеки, указывая пин подключения к arduino

MQ135 sensorMQ135(MQ135\_SENSOR\_PIN);

//переменная-флаг входа в меню

volatile int menuFlag = 0;

//переменная, запоминающая время (в мс) с момента запуска системы

unsigned long time = 0;

//переменная, храненящая температуру воздуха

int temperature = 0;

//переменная, храненящая влажность воздуха

int humidity = 0;

//переменначя, храненящая концентрации CH4 в ppm

unsigned long concentrationCH4 = 0;

//переменначя, храненящая концентрации CO2 в ppm

unsigned long concentrationCO2 = 0;

//максимальное допустимое значение температуры в градусах Цельсия

int maxTemperature = 25;

//максимальное допустимое значение влажности воздуха в процентах

int maxHumidity = 60;

//максимальное допустимое значение CH4 в ppm

unsigned long maxCH4Concentration = 4000;

//максимальное допустимое значение CO2 в ppm

unsigned long maxCO2Concentration = 2000;

//режим запускаемой сигнализации

byte signalingFlags = 0;

//значения АЦП для при нажатии на 1 из 4 кнопок

int values[4] = {254, 498, 774, 1023};

//Обработчик 0-ого прерывания. Изменяет значение флага меню

int changeMenuFlag(){

  menuFlag = (menuFlag == 1) ? 0 : 1;

}

//опрделение номера кнопки по значению АЦП

int buttonNumber(int value){

  int i = 0; //переменная цикла, прохода по массиву допустимых АЦП

  //цикл прохода по массиву возможных значений АЦП

  for(; i < ANALOG\_BUTTONS\_NUM; i++){

    //проверка значения из массива АЦП с учетом погрешности

    if((value <= (values[i] + ERROR)) && (value >= (values[i] - ERROR))){

      return i + 1; //вернуть номер кнопки

    }

  }

  return -1; //нет нажатых кнопок, вернуть -1

}

//функция проверки нажатия кнопок на клавиатуре

int analogKeyBoard(){

//счетчик для определения дастоверности нажатия кнопки

  static int count;

//старое значение аналоговой клавиатуры

  static int oldKeyValue;

//для хранения нажатой клавиши на протяжении 10 проходов

  static int innerKeyValue;

//получить текущую нажатую кнопку

  int actualKeyValue = buttonNumber(analogRead(ANALOG\_KEYBOARD\_PIN));

  //проверяем действильно ли нажато кнопка путем 10 проходов по алгоритму

  if(innerKeyValue != actualKeyValue){

    count = 0; //обнуление счетчика(все снчала)

//обновить текущее значения полученное с обработчика клавитуры

    innerKeyValue = actualKeyValue;

  }

  else{

    count++; //увеличить счетчик проходов

  }

  //кнопка нажата и старое значение не совпадает с актуальным

  if((count >= 10) && (oldKeyValue != actualKeyValue)){

    oldKeyValue = actualKeyValue; //запоминаем новое значение

  }

  return oldKeyValue;

}

//фцнкция прогрева и калибровки датчиков

void  gasSensorsCalibration(){

  unsigned long progressTime = 0; //для получения промежутков по 3с в цикле

  int position = 0; //указатель позиции на дисплее (степень прогрева)

//подать напряжение на синий светодиод (начало прогрева датчиков)

  digitalWrite(BLUE\_LED\_PIN, HIGH);

  display.clear(); //очистить дисплей от символов

  display.setCursor(1,0); //установить курсор на позицию (0, 0)

  display.print("WARMING UP SENSORS"); //вывести надпись "warming up sensors"

  display.setCursor(6, 2); //установить курсор на позицию (6, 2)

  display.print("PROGRESS"); //вывести надпись "progress"

  time = millis(); //получить текущеее время работы

  //цикл задержки на 60с, время прогрева датчиков газа

  while((millis() - time) < WARMING\_UP\_TIME){

    //каждые три секунды обновлять поле progress

    if((millis() - progressTime) > WARMING\_UP\_INTERVAL){

      progressTime = millis(); //получить текущеее время работы

//установить курсор на позицию (position, 3)

      display.setCursor(position++, 3);

      display.print("#"); //вывести символ '#'

    }

  }

  sensorMQ4.calibrate(); //выполнить калибровку датчика MQ4 на чистом воздухе

//выполнить калибровку датчика MQ135 на чистом воздухе

  sensorMQ135.calibrate();

  digitalWrite(BLUE\_LED\_PIN, LOW); //снять напряжения с синего светодиода

  display.clear(); //очистить дисплей от символов

}

//функция чтения показаний с датчиков

void readDataFromSensors(){

//снять значение температуры с датчика

  temperature = sensorDHT22.readTemperature(DHT22\_SENSOR\_PIN);

//снять значение влажности с датчика

  humidity = sensorDHT22.readHumidity(DHT22\_SENSOR\_PIN);

//снять значение концентрации метана

  concentrationCH4 = sensorMQ4.readMethane();

//снять значение концентрации углекислого газа

  concentrationCO2 = sensorMQ135.readCO2();

}

//вывод информации на дисплей

void printInfo(unsigned long valueCH4, unsigned long valueCO2, int valueTemp,

int valueHum, int linePointer){

//если не в меню, то вывод с нулевой позиции, иначе с первой

  int startPosition = (linePointer == -1) ? 0 : 1;

  display.clear(); //очистить дисплей

//установить курсор на позицию (startPosition, 0)

  display.setCursor(startPosition, 0);

  display.print("CH4: "); //вывести название параметра

  display.print(valueCH4); //вывести значениие

  display.print("PPM"); //вывести велечину измерения

//установить курсор на позицию (startPosition, 1)

  display.setCursor(startPosition, 1);

  display.print("CO2: "); //вывести название параметра

  display.print(valueCO2); //вывести значениие

  display.print("PPM"); //вывести велечину измерения

//установить курсор на позицию (startPosition, 2)

  display.setCursor(startPosition, 2);

  display.print("TEMPERATURE: "); //вывести название параметра

  display.print(valueTemp); //вывести значениие

  display.print("C"); //вывести велечину измерения

//установить курсор на позицию (startPosition, 3)

  display.setCursor(startPosition, 3);

  display.print("HUMIDITY: "); //вывести название параметра

  display.print(valueHum); //вывести значениие

  display.print("%"); //вывести велечину измерения

//если в режиме меню, то выводить указаетль '>' и '<'

  if(linePointer != -1){

//установить курсор на позицию (0, linePointer)

    display.setCursor(0, linePointer);

    display.print(">"); //вывести указатель меню '>'

//установить курсор на позицию (19, linePointer)

    display.setCursor(19, linePointer);

    display.print("<"); //вывести указатель меню '<'

  }

}

//функция анализа считанных с датчиков данных

void dataAnalysis(){

  signalingFlags = 0; //сбрасывание всех флагов

  //проверка на превышение CH4 и CO2

  if((concentrationCH4 > maxCH4Concentration) ||

(concentrationCO2 > maxCO2Concentration)){

    signalingFlags = signalingFlags | 1; //установить флаг 1

  }

  //проверка на превышения температуры и влажности

  if((temperature > maxTemperature) || (humidity > maxHumidity)){

    signalingFlags = signalingFlags | (1 << 1); //установить флаг 2

  }

  digitalWrite(RED\_LED\_PIN, LOW); //снять питание с красного светодиода

  digitalWrite(YELLOW\_LED\_PIN, LOW); //снять питание с желтого светодиода

  digitalWrite(RELAY\_PIN, LOW); //закрыть реле

  noTone(BUZZER\_PIN); //отключить пъезодинамик

  //определение типа сигнализирования по установленным флагам

  switch(signalingFlags){

    case 1: //превышают только газовые параметры

//подать напряжение на красный светодиод

      digitalWrite(RED\_LED\_PIN, HIGH);

      digitalWrite(RELAY\_PIN, HIGH); //открыть реле

      tone(BUZZER\_PIN, 1500); //включить пъезодинамик

      break;

    case 2: //превышает только температура или влажность

//подать напряжение на желтый светодиод

      digitalWrite(YELLOW\_LED\_PIN, HIGH);

      tone(BUZZER\_PIN, 1700); //включить пъезодинамик

      break;

    case 3: //превышает температура или влажность, а также газовые параметры

//подать напряжение на красный светодиод

      digitalWrite(RED\_LED\_PIN, HIGH);

//подать напряжение на желтый светодиод

      digitalWrite(YELLOW\_LED\_PIN, HIGH);

      digitalWrite(RELAY\_PIN, HIGH); //открыть реле

      tone(BUZZER\_PIN, 1900); //включить пъезодинамик

      break;

  }

}

//основное меню

void menuFunc(){

  int menuPointer = 0; //указатель на пункт меню

  int currentKeyValue = 0; //значение нажатой кнопки

  int flag = 0; //флаг для определения момента отпускания кнопки

  //вывести меню с указателеме '> ... <' на дисплей

  printInfo(maxCH4Concentration, maxCO2Concentration, maxTemperature,

maxHumidity, menuPointer);

  while(menuFlag == 1){ //если была нажата кнопка входа в меню

    currentKeyValue = analogKeyBoard(); //чтение значения с клавиатуры

    if((currentKeyValue != -1) && (flag == 0)){ //обработчик нажатия кнопки

      flag = 1; //кнопка нажата

      if(currentKeyValue == 1){ //если нажата 1 кнопка

        menuPointer++; //переход к следующему пункту меню

        //когда дошло до конца меню сброс в начало

        menuPointer = (menuPointer == 4) ? 0 : menuPointer;

      }

      else if(currentKeyValue == 2){ //если нажата 2 кнопка

        menuPointer--; //переход к предыдущему пункту меню

        //когда дошло до начала меню сброс в конец

        menuPointer = (menuPointer == -1) ? 3 : menuPointer;

      }

      else if(currentKeyValue == 3){ //если нажата 3 кнопка

//в зависимости от пункта меню уменьшаем максимальное значение

        switch(menuPointer){

          case 0: //menuPointer указывает на максимальную концентрацию CH4

            maxCH4Concentration -= 25;

            break;

          case 1: //menuPointer указывает на максимальную концентрацию CO2

            maxCO2Concentration -= 25;

            break;

          case 2: //menuPointer указывает на максимальную температуру

            maxTemperature -= 1;

            break;

          case 3: //menuPointer указывает на максимальную влажность

            maxHumidity -= 1;

            break;

        }

      }

      else if(currentKeyValue == 4){ //если нажата 4 кнопка

//в зависимости от пункта меню увеличиваем максимальное значение

         switch(menuPointer){

          case 0:  //menuPointer указывает на максимальную концентрацию CH4

            maxCH4Concentration += 25;

            break;

          case 1: //menuPointer указывает на максимальную концентрацию CO2

            maxCO2Concentration += 25;

            break;

          case 2: //menuPointer указывает на максимальную температуру

            maxTemperature += 1;

            break;

          case 3: //menuPointer указывает на максимальную влажность

            maxHumidity += 1;

            break;

        }

      }

    }

//обработчик отпускания кнопки

    if((currentKeyValue == -1) && (flag == 1)){

      flag = 0; //кнопки не нажаты

      //обновить меню на дисплее

      printInfo(maxCH4Concentration, maxCO2Concentration, maxTemperature,

maxHumidity, menuPointer);

    }

  }

}

//чтения данных из eeprom (инициализация максимальных значений)

void initVariablesFromEEPROM(){

//считать данные в maxCH4Concentration из памяти

  EEPROM.get(0, maxCH4Concentration);

//считать данные в maxCO2Concentration из памяти

  EEPROM.get(4, maxCO2Concentration);

//считать данные в maxTemperature из памяти

  EEPROM.get(8, maxTemperature);

//считать данные в maxHumidity из памяти

  EEPROM.get(10, maxHumidity);

}

//запись данных в eeprom

void saveVariablesIntoEEPROM(){

//записать данные из maxCH4Concentration в память

  EEPROM.put(0, maxCH4Concentration);

//записать данные из maxCO2Concentration в память

  EEPROM.put(4, maxCO2Concentration);

//записать данные из maxTemperature в память

  EEPROM.put(8, maxTemperature);

//записать данные из maxHumidity в память

  EEPROM.put(10, maxHumidity);

}

void setup() { //задание начальных параметров

//установить режим вывода на прием сигнала

  pinMode(MENU\_BUTTON\_PIN, INPUT);

//установить режим вывода на выдачу сигнала

  pinMode(BLUE\_LED\_PIN, OUTPUT);

//установить режим вывода на выдачу сигнала

  pinMode(YELLOW\_LED\_PIN, OUTPUT);

//установить режим вывода на выдачу сигнала

  pinMode(RED\_LED\_PIN, OUTPUT);

//установить режим вывода на выдачу сигнала

  pinMode(BUZZER\_PIN, OUTPUT);

//установить режим вывода на прием сигнала

  pinMode(DHT22\_SENSOR\_PIN, INPUT);

//установить режим вывода на выдачу сигнала

  pinMode(RELAY\_PIN, OUTPUT);

//связываем прерывание с обработчиком

  attachInterrupt(0, changeMenuFlag, RISING);

//инициализация дисплея

  display.init();

//включить подсветку дисплея

  display.backlight();

//инициализировать максимльные значения

  initVariablesFromEEPROM();

//прогрев и калибровка датчиков

  gasSensorsCalibration();

}

void loop() { //основной цикл программы

  //если прошло 2 секунды опрос датчиков

  if((millis() - time) > SENSOR\_POLLING\_INTERVAL){

    time = millis(); //обновить время работы контроллера

    readDataFromSensors(); //получить данные с датчиков

    //вывести информацию на дисплей с обновленными параметрами

    printInfo(concentrationCH4, concentrationCO2, temperature, humidity, -1);

    dataAnalysis(); //анализ полученных данных

  }

  if(menuFlag == 1){ //зайти в меню

    menuFunc();

    saveVariablesIntoEEPROM(); //запимсать значения в память

  }

}