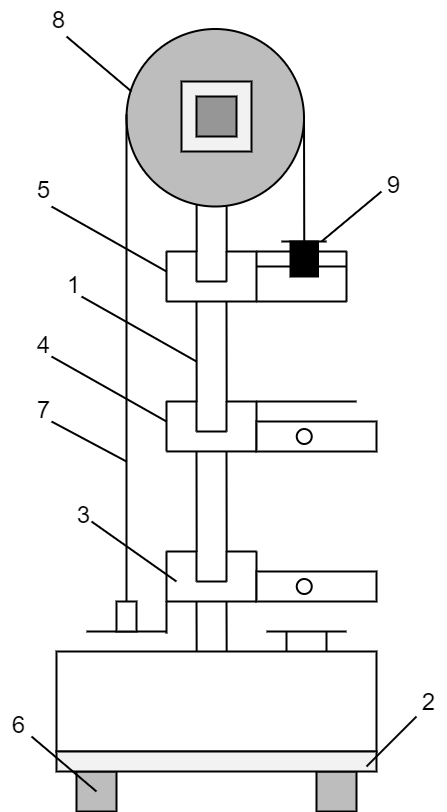
**ПРИЛОЖЕНИЕ 1**



Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 1.

Структурная схема лабораторной установки “Машина Атвуда”

Лит.

Листов

1

Кафедра общей физики

Реценз.

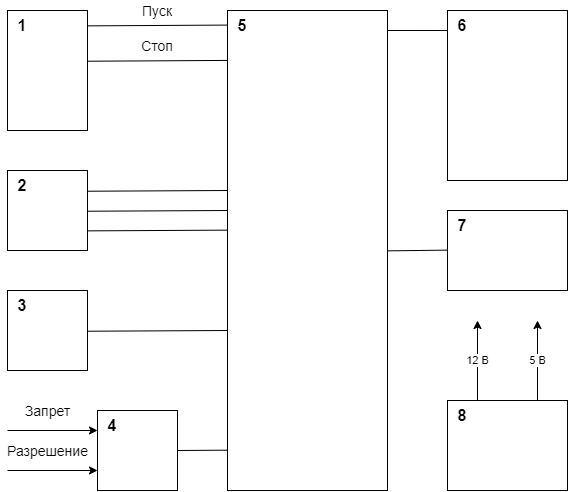
Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 2**



Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 2.

Структурная схема модернизированной установки

”

Лит.

Листов

1

Кафедра общей физики

Реценз.

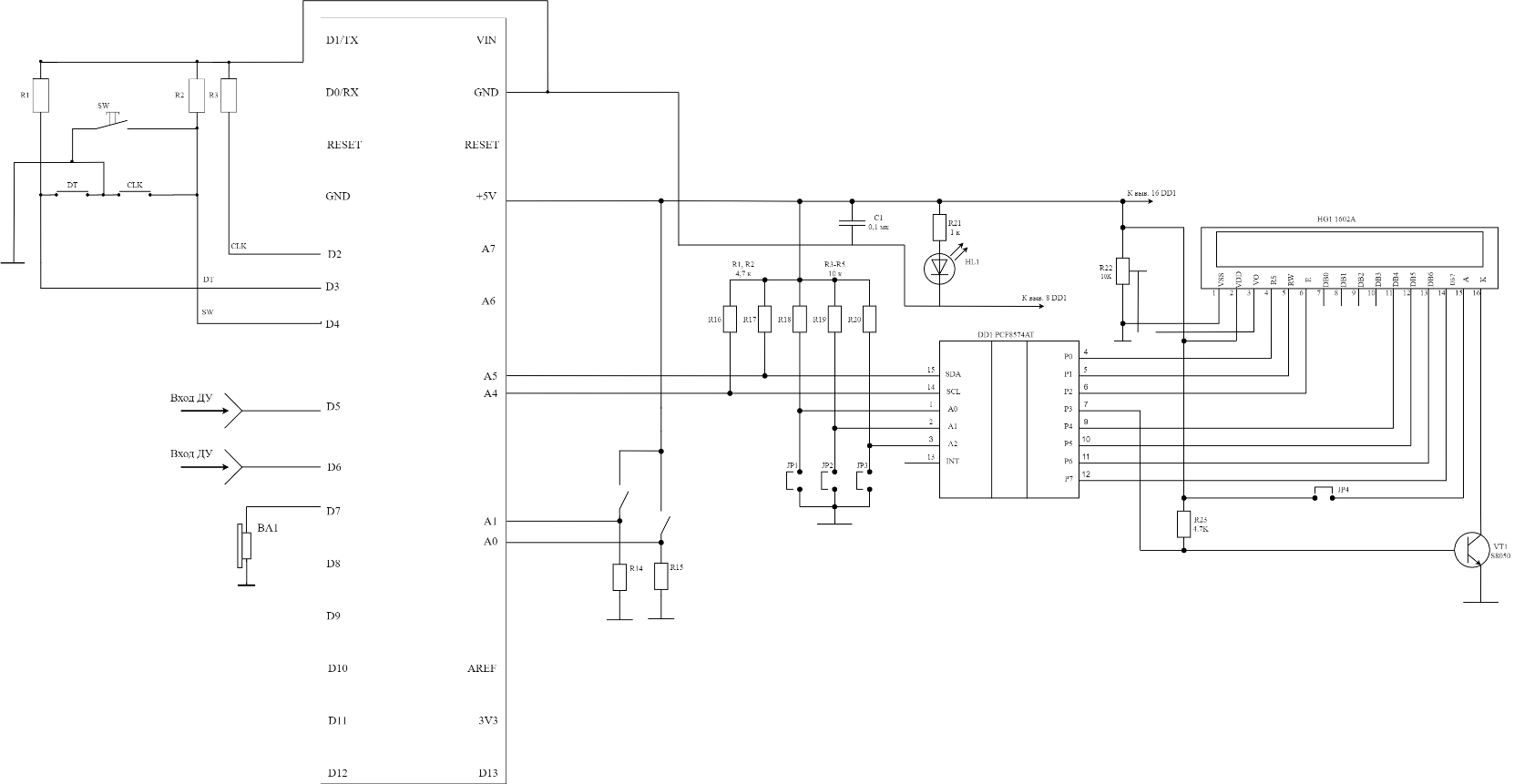
Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 3**



Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 3.

Функциональная схема модернизированного устройства ”

Лит.

Листов

1

Кафедра общей физики

Реценз.

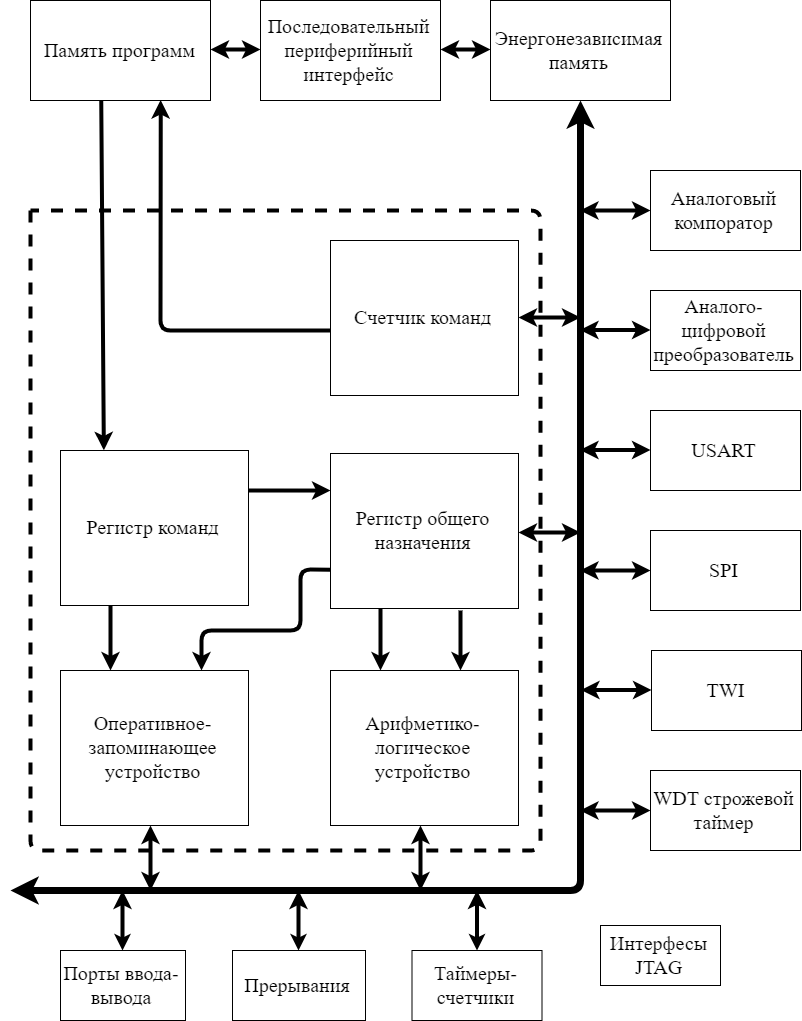
Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 4**



Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 4.

Функциональная схема блока микроконтроллера”

Лит.

Листов

1

Кафедра общей физики

Реценз.

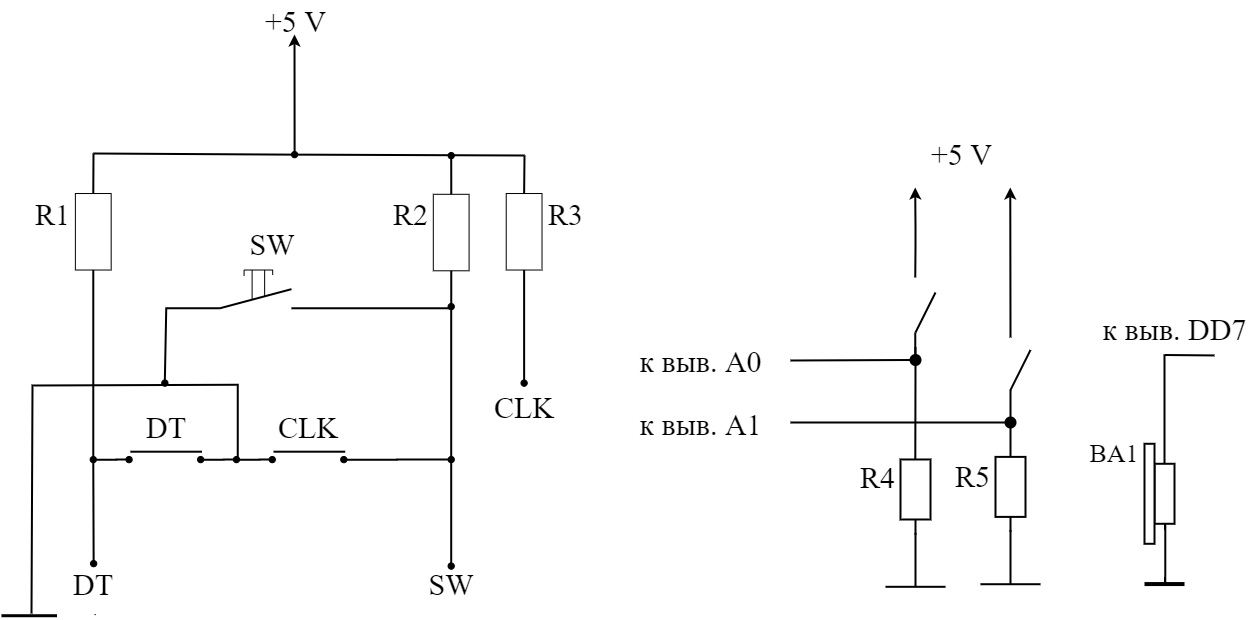
Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 5**



Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 5.

Принципиальная схема блока управления и задач, кнопок и звукового излучателя

”

Лит.

Листов

1

Кафедра общей физики

Реценз.

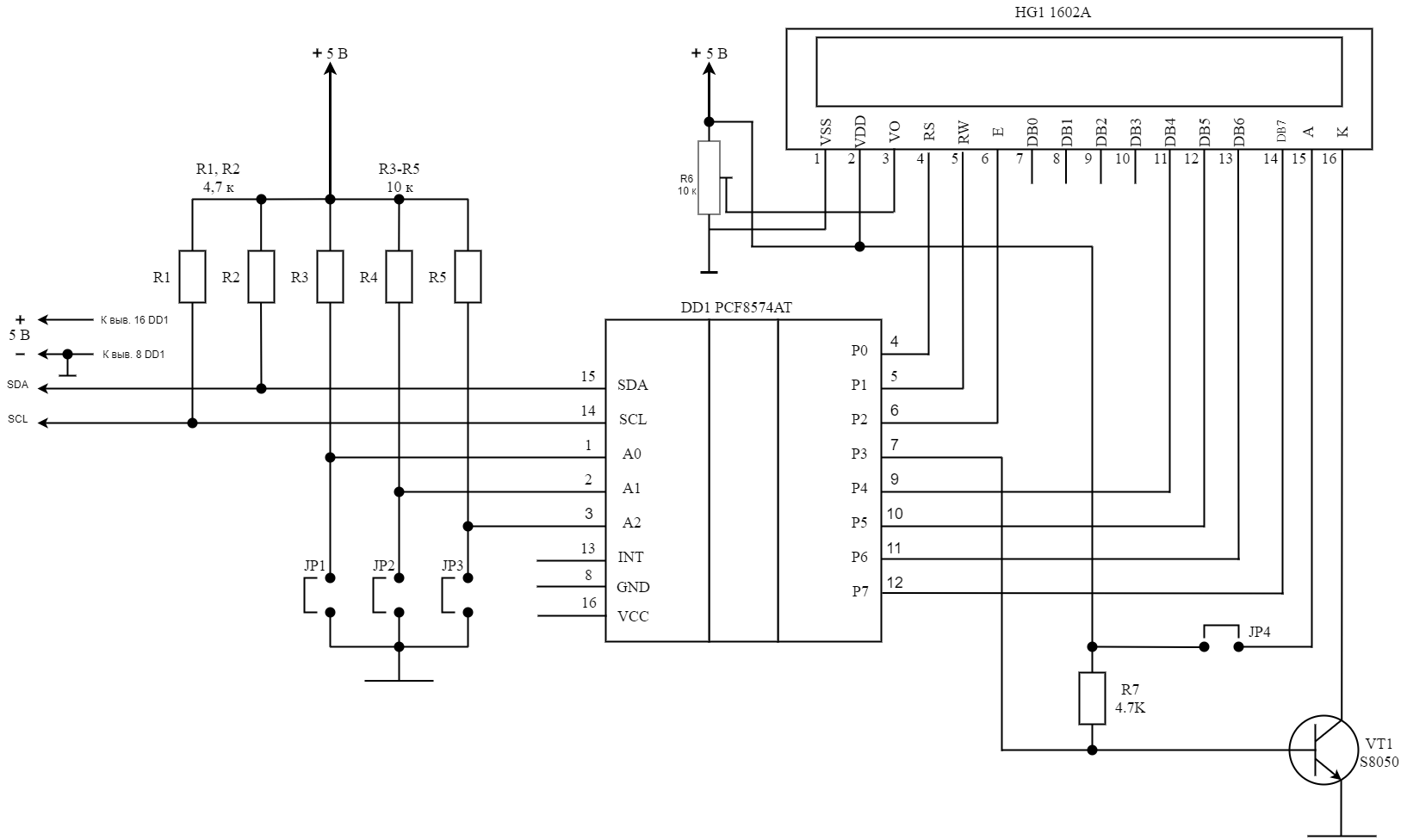
Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 6**



Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 6.

Принципиальная схема блока индикации

Лит.

Листов

1

Кафедра общей физики

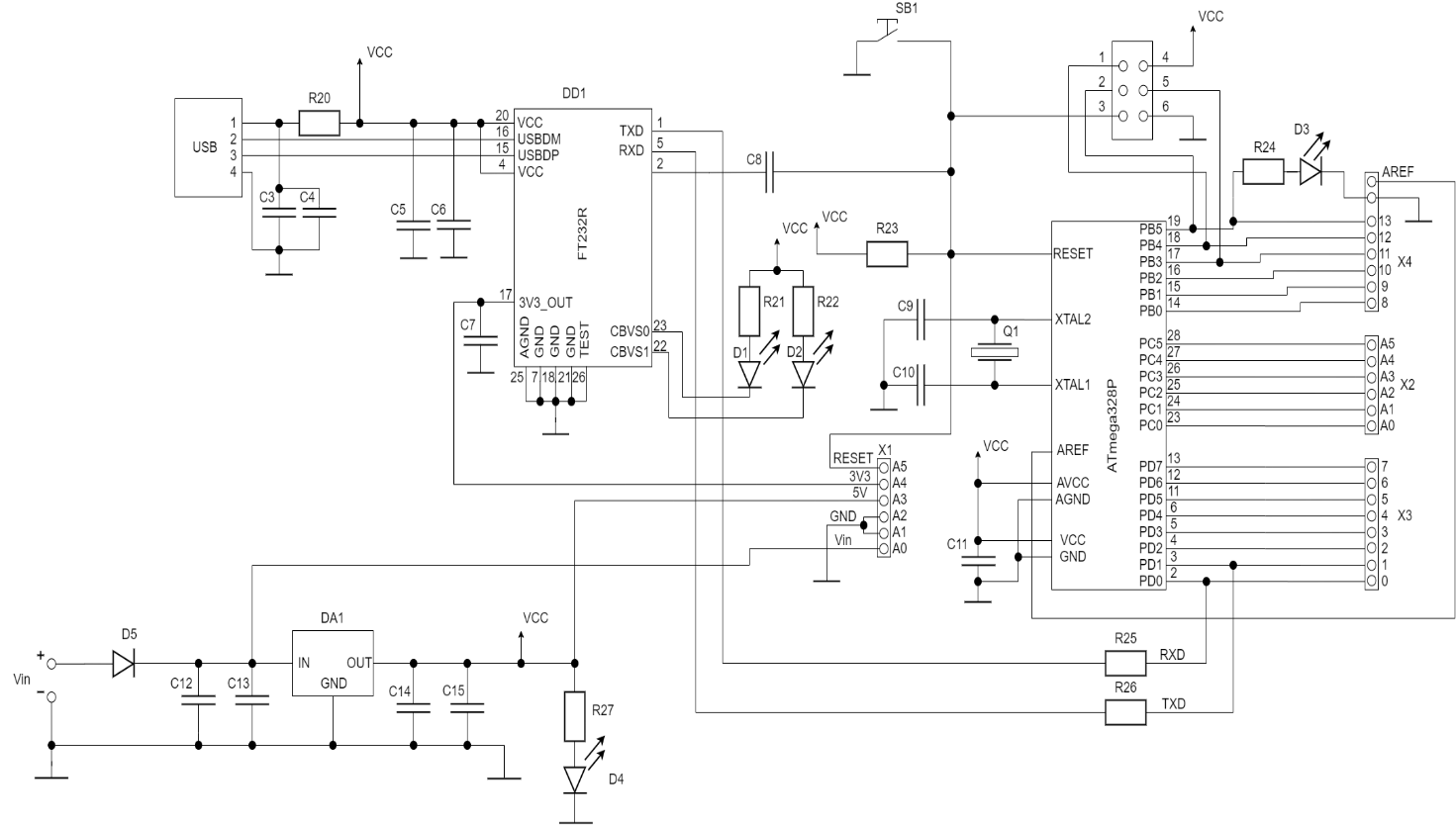
Реценз.

Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 7**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 7.

Принципиальная схема блока микроконтроллера

Лит.

Листов

1

Кафедра общей физики

Реценз.

Масса

Масштаб

1

1 : 1

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

ГрГУ 0000001ЭЗ

Разраб.

Ильюков И.А.

Провер.

Маскевич А.А.

Т. Контр.

Н. Контр.

Утверд.

*Разработанное устройство*

Схема 8.

Блок схема программного кода

Лит.

Листов

1

Кафедра общей физики

Реценз.

Масса

Масштаб

1

1 : 1

**ПРИЛОЖЕНИЕ 8**

#include <LCD\_1602\_RUS.h>

LCD\_1602\_RUS lcd(0x27, 16, 2);

#define pin\_CLK 4

#define pin\_DT 2

#define pin\_Btn 9

const int pin\_Sound\_Signal = 3;

unsigned long CurrentTime, LastTime;

enum eEncoderState{

eNone,

eLeft,

eRight,

eButton

};

uint8\_t EncoderA, EncoderB, EncoderAPrev;

int8\_t counter = 0;

bool ButtonPrev;

eEncoderState GetEncoderState(){

eEncoderState Result = eNone;

CurrentTime = millis();

if (CurrentTime - LastTime >= 5){

LastTime = CurrentTime;

if (digitalRead(pin\_Btn) == LOW){

if (ButtonPrev){

Result = eButton;

ButtonPrev = 0;

}

} else {

ButtonPrev = 1;

EncoderA = digitalRead(pin\_DT);

EncoderB = digitalRead(pin\_CLK);

if ((!EncoderA) && (EncoderAPrev)){

if (EncoderB){

Result = eRight;

} else {

Result = eLeft;

}

}

EncoderAPrev = EncoderA

}

}

return Result;

}

void setup(){

Wire.begin();

lcd.init();

lcd.backlight();

pinMode(pin\_DT, INPUT);

pinMode(pin\_CLK, INPUT);

pinMode(pin\_Btn, INPUT\_PULLUP);

pinMode(pin\_Sound\_Signal, OUTPUT);

Serial.begin(9600);

counter = 0;

}

bool isNeedPreView = true;

bool isEncoderButtonPressed = false;

void loop(){

if (isNeedPreView){

prtintTitle();

printOption();

isNeedPreView = false;

}

if (isEncoderButtonPressed){

goToOption();

isNeedPreView = true;

}

switch (GetEncoderState()){

case eNone:

return;

case eButton:

isEncoderButtonPressed = true;

break;

case eLeft:

counter--;

prtintTitle();

printOption();

isEncoderButtonPressed = false;

break;

case eRight:

counter++;

prtintTitle();

printOption();

isEncoderButtonPressed = false;

break;

}

delay(100);

}

void encoder(){

switch (GetEncoderState()){

case eButton:

isEncoderButtonPressed = true;

break;

case eLeft:

counter++;

isEncoderButtonPressed = false;

break;

case eRight:

counter--;

isEncoderButtonPressed = false;

break;

case eNone:

isEncoderButtonPressed = false;

return;

}

}

void prtintTitle(){

if (counter > 1 || counter < 0){

counter = 0;

}

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L"PEЖИМ");

}

void printOption(){

lcd.setCursor(0, 1);

if(counter < 0 || counter > 1){

counter = 0;

}

switch (counter){

case 0:

lcd.print(L"0. СЕКУНДОМЕР");

break;

case 1:

lcd.print(L"1. ТАЙМЕР");

break;

}

}

void goToOption(){

switch (counter){

case 0:

stopwatch();

exitFromOption();

break;

case 1:

timer();

exitFromOption();

break;

}

}

void exitFromOption(){

counter = 1;

encoder();

delay(500);

}

void stopwatch(){

const byte analogPinForStartButtonOfStopWatch = 0;

const byte analogPinForStopButtonOfStopWatch = 1;

long timeOfStopWatch = 0;

long currentTime = 0;

bool isNeedPreView = true;

do {

if (isNeedPreView) {

printPreViewForStopWatch();

isNeedPreView = false;

}

if (isAnalogButtonPressed(analogPinForStartButtonOfStopWatch)) {

timeOfStopWatch = micros();

while (!isAnalogButtonPressed(analogPinForStopButtonOfStopWatch)) {

currentTime = micros() - timeOfStopWatch;

printTimeForStopWatch(currentTime);

delay(100);

}

printResultTimeForStopWatch(currentTime);

delay(2500);

isNeedPreView = true;

}

encoder();

} while (!isEncoderButtonPressed);

}

void printTimeForStopWatch(long timeToPrint){

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L"ВРЕМЯ ");

long second = timeToPrint / 1000000;

long millisecond = timeToPrint - second \* 1000000;

millisecond = millisecond / 100;

lcd.print(String(second));

lcd.print(".");

lcd.print(String(millisecond));

lcd.setCursor(0, 1);

lcd.print(L" секунд ");

}

void printPreViewForStopWatch() {

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L" ДЛЯ НАЧАЛА ");

lcd.setCursor(0, 1);

lcd.print(L" НАЖМИТЕ СТАРТ ");

}

void printResultTimeForStopWatch(long timeToPrint){

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L" РЕЗУЛЬТАТ: ");

long second = timeToPrint / 1000000;

long millisecond = timeToPrint - second \* 1000000;

millisecond = millisecond / 100;

lcd.setCursor(0, 1);

lcd.print(String(second));

lcd.print(".");

lcd.print(String(millisecond));

lcd.print(L" сек.");

}

void timer() {

const int countOfSound = 2;

const int delayBetweenSound = 500;

byte analogPinForStartButtonOfTimer = 0;

double stepForTimer = 0.5;

double setTimeForTimer = 0;

double currentTime = 0;

double needTimeForTimer = 0;

bool isNeedPreView = true;

counter = 0;

int last = counter;

bool isNeedPrint = true;

do {

if (last != counter || isNeedPreView){

counter = counter <= 0 ? 0 : counter;

last = counter;

setTimeForTimer = last \* stepForTimer;

printPreViewForTimer(setTimeForTimer);

isNeedPreView = false;

}

if (isAnalogButtonPressed(analogPinForStartButtonOfTimer)){

needTimeForTimer = micros() + ( setTimeForTimer \* 1000000 );

while (needTimeForTimer > micros()) {

delay(100);

currentTime = needTimeForTimer - micros();

printRemaimingTimeForTimer(currentTime);

}

printWhenTimeEndsForTimer();

soundSignal(numberSound, delayBetweenSound);

delay(1000);

counter = 0;

isNeedPreView = true;

}

encoder();

} while (!isEncoderButtonPressed);

}

void printPreViewForTimer(double setTimeToPrint){

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L"УСТАНОВИТЕ ВРЕМЯ");

lcd.setCursor(0, 1);

lcd.print(String(setTimeToPrint));

lcd.print(L" сек.");

}

void printRemaimingTimeForTimer(double timerTimeToPrint){

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L"ВРЕМЯ ");

long second = timeToPrint / 1000000;

long millisecond = timeToPrint - second \* 1000000;

millisecond = millisecond / 100;

lcd.print(String(second));

lcd.print(".");

lcd.print(String(millisecond));

lcd.setCursor(0, 1);

lcd.print(L" сек.");

}

void printWhenTimeEndsForTimer(){

lcd.setCursor(0, 0);

lcd.clear();

lcd.print(L" СТОП! ");

}

bool isAnalogButtonPressed(byte analogPinNumber){

return analogRead(analogPinNumber) < 300;

}

bool isPinHigh(byte pinNumber){

return digitalRead(pinNumber) == HIGH;

}

void soundSignal(int numbersOfSound ,int timeBetweenSonds){

const int highLevelOfSignal = 100;

const int lowLevelOfSignal = 0;

for (int j = 0; j < numbersOfSound; j++){

analogWrite(pin\_Sound\_Signal, highLevelOfSignal);

delay(timeBetweenSonds);

analogWrite(pin\_Sound\_Signal, lowLevelOfSignal);

delay(timeBetweenSonds);

}

}

**ПРИЛОЖЕНИЕ 9**



