#include <SoftwareSerial.h> // Библиотека програмной реализации обмена по UART-протоколу

const byte button = 11; //

const byte handBrakeIn = 9; // ручной тормоз

const byte neutralGearIn = 6; // брелок

const byte startSIn = 4; // сигнал автозапуска от сигнализации

const byte beeperOut = 10; // сирена

const byte powerOut = 17; // зажигание

const byte starterOut = 2; // стартер

const byte fanOut = 10; // бензонасос

const byte autostartModeOut = 8 ; // LED const byte signalOut = 9; //

const byte maxTryesCount = 3; // максимальное число попыток запуска

const byte powerDelayTime = 30; // задержка включения питания (перерыв между попытками), секунд

const byte starterDelayTime = 5; // задержка включения стартера, секунд

const byte maxStartTime = 12; // время работы стартера, секунд

const byte fanDelayTime = 10; // задержка включения вентилятора отопителя, секунд

const word maxRunTime = 900; // время работы двигателя после запуска, секунд (600 секунд - 10 минут, 1200 секунд - 20 минут)

volatile bool engineRuning = false; // состояние двигателя, выключен / работает

bool needStart = false; // наличие запроса на запуск двигателя

bool autostartMode = false; // режим автозапуска, нет / да

bool startMode = false; // режим попытки запуска двигателя, нет / да

bool powerOn = false; // подали питание

volatile bool starterOn = false; // включили стартер

volatile bool fanOn = false; // проверка бензонасоса

unsigned long timeStart = 0; // время запуска двигателя

unsigned long tryStartTime = 0; // время начала очередной попытки запуска двигателя

byte tryesCount = 0; // число попыток запуска

byte tryCurrent = 0; // текущая попытка запуска

word tahoSignal = 0; // сигнал тахометра

byte blinkCount = 0;

bool ledOn = false;

word blinkLength = 500; // длительность сигнала светодиода

word speedSignal = 0;

void setup() {

pinMode(handBrakeIn, INPUT);

pinMode(neutralGearIn, INPUT);

pinMode(speedIn, INPUT);

pinMode(tahoIn, INPUT);

pinMode(startSIn, INPUT);

pinMode(immoOut, OUTPUT);

pinMode(powerOut, OUTPUT);

pinMode(beeperOut, OUTPUT);

pinMode(starterOut, OUTPUT);

pinMode(fanOut, OUTPUT);

pinMode(autostartModeOut, OUTPUT);

pinMode(signalOut, OUTPUT);

digitalWrite(immoOut, LOW);

digitalWrite(powerOut, LOW);

digitalWrite(beeperOut, LOW);

digitalWrite(starterOut, LOW);

digitalWrite(fanOut, LOW);

digitalWrite(autostartModeOut, LOW);

digitalWrite(signalOut, LOW);

delay(2000); // задержка 10 секунд после включения питания

void loop() {

unsigned long timeFromStart = 0;

unsigned long currentMillis = millis(); // текущее время

needStart = false;

// получаем состояние двигателя

tahoSignal = pulseIn(tahoIn, HIGH, 100000);

engineRuning = (tahoSignal > 0 && tahoSignal < 18000);

speedSignal = digitalRead(speedIn);

//Serial.println(speedSignal);

if (engineRuning) {

if (timeStart == 0) {

timeStart = currentMillis; // определяем момент запуска двигателя

} else {

timeFromStart = (currentMillis - timeStart) / 1000; // время работы двигателя в секундах

// машина заведена, прошло время задержки для включения дополнительного оборудования, включаем его, если ещё не включено

if (!fanOn && timeFromStart > fanDelayTime) {

digitalWrite(fanOut, HIGH);

fanOn = true;

}

}

} else {

if (fanOn) { // если двигатель заглушился, а доп оборудование было включено - выключаем его.

digitalWrite(fanOut, LOW);

fanOn = false;

}

timeStart = 0;

}

// режим автозапуска

if (autostartMode) {

// получаем сигнал с ручника и датчика нейтрали,

// а так же проверяем время прошедшее с момента запуска двигателя

// если что-то не так - глушим всё

if (

digitalRead(startSIn) == 1 || // сигнал отключения

digitalRead(handBrakeIn) != handBrakeSignal || // ручник

digitalRead(neutralGearIn) != neutralGearSignal || // нейтралка

tryesCount > maxTryesCount || // число попыток запуска

timeFromStart > maxRunTime // время работы

) {

autostartOff();

delay (500);

if (digitalRead(handBrakeIn) != handBrakeSignal) {

//sendMessage("Ошибка: Сигнал ручника! Автозапуск отключен.");

sendMessage("Handbrake", true);

blinkAndBeep(3, 300);

}

if (digitalRead(neutralGearIn) != neutralGearSignal) {

//sendMessage("Ошибка: Сигнал нейтрали! Автозапуск отключен.");

sendMessage("Gearbox", true);

blinkAndBeep(4, 300);

}

if (tryesCount > maxTryesCount) {

//sendMessage("Ошибка: исчерпано число попыток. Автозапуск отключен.");

sendMessage("Max tryes off", true);

blinkAndBeep(5, 300);

}

if (timeFromStart > maxRunTime) {

//sendMessage("Успешное завершение автозапуска. Время работы: ");

sendMessage("Completed OK", false);

blinkAndBeep(1, 1000);

}

return;

}

if (startMode) {

// запуск двигателя

if (tryCurrent != tryesCount) { // новая попытка

tryStartTime = currentMillis; // время начала очередной попытки

tryesCount = tryCurrent;

digitalWrite(starterOut, LOW);

digitalWrite(immoOut, LOW);

digitalWrite(beeperOut, LOW);

digitalWrite(signalOut, LOW);

digitalWrite(fanOut, LOW);

digitalWrite(powerOut, LOW);

powerOn = false;

starterOn = false;

fanOn = false;

// если это первый запуск - включаем зажигание, чтобы аккумулятор начал работать

if (tryCurrent == 1) {

digitalWrite(immoOut, HIGH);

delay(300);

digitalWrite(powerOut, HIGH);

}

} else {

// сколько секунд прошло с начала процедуры

unsigned int timePassed = (currentMillis - tryStartTime) / 1000;

// задержка автозапуска, включаем питание

if (!powerOn && timePassed > powerDelayTime) {

digitalWrite(powerOut, LOW);

delay(500);

digitalWrite(immoOut, HIGH);

delay(300);

digitalWrite(powerOut, HIGH);

delay(500);

digitalWrite(signalOut, HIGH);

powerOn = true;

}

// задержка, включаем стартер если двигатель не запущен

if (!engineRuning && !starterOn && timePassed > powerDelayTime + starterDelayTime) { //

digitalWrite(starterOut, HIGH);

starterOn = true;

}

// завелась, двигатель работает уже 2 секунды, выключаем стартер, выходим из режима запуска

if (engineRuning && starterOn && timeFromStart > 0) {

digitalWrite(starterOut, LOW);

starterOn = false;

startMode = false;

sendMessage("Engine running", false);

}

// не завелась, а время вышло - выключаем стартер, запускаем новую попытку

if (!engineRuning && starterOn && timePassed > powerDelayTime + starterDelayTime + maxStartTime) { //

digitalWrite(starterOut, LOW);

starterOn = false;

tryCurrent = tryCurrent + 1;

}

}

// конец автозапук двигателя

}

} else {

// получаем наличие сигналов на запуск двигателя

if (digitalRead(startSIn) == 1 ) {

needStart = true;

// } else {

// needStart = false;

}

if (!engineRuning) { // если двигатель не заведён

if (needStart) { // и есть сигнал на его запуск

autostartMode = true; // переходим в режим запуска двигателя

startMode = true; // и пробуем запустить двигатель

tryesCount = 0;

tryCurrent = 1;

timeStart = 0;

digitalWrite(autostartModeOut, HIGH);

sendMessage("AutoStart", false);

delay(1000); // подождём, чтобы исчез сигнал, на случай, чтобы не повторять попытку неудачного запуска

// вешаем прерывание на датчик скорости

attachInterrupt(digitalPinToInterrupt(speedIn), speedChange, CHANGE);

}

}

}

if (blinkCount > 0) {

makeSignal(currentMillis);

}

}

void blinkAndBeep(int count, int signalLength){

blinkCount = count;

if (signalLength == 0) {

blinkLength = 500;

} else {

blinkLength = signalLength;

}

ledOn = false;

digitalWrite(beeperOut, LOW);

digitalWrite(autostartModeOut, LOW);

}

void makeSignal(unsigned long currentMillis){

int signalFull = blinkLength \* 2;

if (ledOn) {

if (currentMillis % signalFull < blinkLength) {

digitalWrite(autostartModeOut, LOW);

digitalWrite(beeperOut, LOW);

ledOn = false;

blinkCount--;

}

} else {

if (currentMillis % signalFull > blinkLength) {

digitalWrite(autostartModeOut, HIGH);

digitalWrite(beeperOut, HIGH);

ledOn = true;

}

}

}

void speedChange(){

if (starterOn || engineRuning) {

autostartOff();

//sendMessage("Ошибка: Началось движение! Автозапуск отключен:");

sendMessage("Car moving", true);

blinkAndBeep(2, 300);

}

}

void autostartOff(){

digitalWrite(starterOut, LOW);

digitalWrite(signalOut, LOW);

digitalWrite(immoOut, LOW);

digitalWrite(powerOut, LOW);

digitalWrite(beeperOut, LOW);

digitalWrite(autostartModeOut, LOW);

timeStart = 0;

tryCurrent = 0;

autostartMode = false;

startMode = false;

powerOn = false;

starterOn = false;

ledOn = false;

detachInterrupt(digitalPinToInterrupt(speedIn));

}

}