#include <iarduino\_RTC.h> // Библиотека часов

#include <avr/wdt.h> // Библиотека сна

#include <avr/sleep.h>

#include <avr/eeprom.h> // Библиотека для EEPROM

#include "GyverButton.h"

#define BTN\_PIN\_HOUR 4 // кнопка для часов

#define BTN\_PIN\_MINUTE 3 // кнопка для минут

#define FOUR\_SECOND 1<<12

#define ONE\_SECOND 1<<10

#define SET\_TIME 1 // Если нужно установить время при компиляции (берет время из системы)

#if SET\_TIME

// Определяем системное время: // Время загрузки скетча.

const char\* strM = "JanFebMarAprMayJunJulAugSepOctNovDec"; // Определяем массив всех вариантов текстового представления текущего месяца.

const char\* sysT = \_\_TIME\_\_; // Получаем время компиляции скетча в формате "SS:MM:HH".

const char\* sysD = \_\_DATE\_\_; // Получаем дату компиляции скетча в формате "MMM:DD:YYYY", где МММ - текстовое представление текущего месяца, например: Jul.

// Парсим полученные значения sysT и sysD в массив i: // Определяем массив «i» из 6 элементов типа int, содержащий следующие значения: секунды, минуты, часы, день, месяц и год компиляции скетча.

const int i[6] {(sysT[6] - 48) \* 10 + (sysT[7] - 48), (sysT[3] - 48) \* 10 + (sysT[4] - 48), (sysT[0] - 48) \* 10 + (sysT[1] - 48), (sysD[4] - 48) \* 10 + (sysD[5] - 48), ((int)memmem(strM, 36, sysD, 3) + 3 - (int)&strM[0]) / 3, (sysD[9] - 48) \* 10 + (sysD[10] - 48)};

#endif

const int leds\_hour[] = {12, 11, 10, 9};

const int leds\_minute[] = {8, 7, 6, 14, 15, 16};

const size\_t num\_leds\_hour = sizeof(leds\_hour) / sizeof(leds\_hour[0]);

const size\_t num\_leds\_minute = sizeof(leds\_minute) / sizeof(leds\_minute[0]);

byte hour, minute;

iarduino\_RTC watch(RTC\_DS3231);

GButton hour\_button(BTN\_PIN\_HOUR);

GButton min\_button(BTN\_PIN\_MINUTE);

void setup() {

watch.begin(); // Инициируем RTC модуль

Serial.begin(9600);

#if SET\_TIME

if(!eeprom\_read\_byte(0)){

watch.settime(i[0], i[1], i[2], i[3], i[4], i[5]); // Устанавливаем время в модуль: i[0] сек, i[1] мин, i[2] час, i[3] день, i[4] месяц, i[5] год, без указания дня недели.

eeprom\_write\_byte(0, 1);

}

#endif

for (size\_t i = 0; i < num\_leds\_hour; i++)

pinMode(leds\_hour[i], OUTPUT);

for (size\_t i = 0; i < num\_leds\_minute; i++)

pinMode(leds\_minute[i], OUTPUT);

}

void loop() {

hour\_button.tick();

min\_button.tick();

Serial.println(FOUR\_SECOND);

if (min\_button.isStep()) { // удержание

long long time\_now = millis();

bool is\_button\_presed = false;

minute = 0;

while (millis() - time\_now < FOUR\_SECOND) {

if (is\_button\_presed) {

time\_now = millis();

while (millis() - time\_now < FOUR\_SECOND) {

min\_button.tick();

if (min\_button.isPress() || min\_button.isStep()) {

if (minute > 60)

minute = 0;

minute++;

time\_now = millis();

}

for (size\_t i = 0; i < num\_leds\_minute; i++)

digitalWrite(leds\_minute[num\_leds\_minute - i - 1], bitRead(minute, i));

}

for (size\_t i = 0; i < num\_leds\_minute; i++)

digitalWrite(leds\_minute[i], LOW);

delay(100);

for (size\_t i = 0; i < num\_leds\_minute; i++)

digitalWrite(leds\_minute[i], HIGH);

delay(100);

for (size\_t i = 0; i < num\_leds\_minute; i++)

digitalWrite(leds\_minute[i], LOW);

watch.settime(0, minute);

}

for (size\_t i = 0; i < num\_leds\_minute && !is\_button\_presed; i++) {

min\_button.tick();

if (min\_button.isSingle())

is\_button\_presed = true;

digitalWrite(leds\_minute[i], HIGH);

delay(50);

}

for (size\_t i = 0; i < num\_leds\_minute && !is\_button\_presed; i++) {

min\_button.tick();

if (min\_button.isSingle())

is\_button\_presed = true;

digitalWrite(leds\_minute[i], LOW);

delay(50);

}

}

}

if (hour\_button.isStep()) { // удержание

long long time\_now = millis();

bool is\_button\_presed = false;

hour = 0;

while (millis() - time\_now < FOUR\_SECOND) {

if (is\_button\_presed) {

time\_now = millis();

while (millis() - time\_now < FOUR\_SECOND) {

hour\_button.tick();

if (hour\_button.isPress() || hour\_button.isStep()) {

if (hour > 12)

hour = 0;

hour++;

time\_now = millis();

}

for (size\_t i = 0; i < num\_leds\_hour; i++)

digitalWrite(leds\_hour[num\_leds\_hour - i - 1], bitRead(hour, i));

}

for (size\_t i = 0; i < num\_leds\_hour; i++)

digitalWrite(leds\_hour[i], LOW);

delay(100);

for (size\_t i = 0; i < num\_leds\_hour; i++)

digitalWrite(leds\_hour[i], HIGH);

delay(100);

for (size\_t i = 0; i < num\_leds\_hour; i++)

digitalWrite(leds\_minute[i], LOW);

watch.settime(0, watch.gettime("i"), hour);

}

for (size\_t i = 0; i < num\_leds\_hour && !is\_button\_presed; i++) {

hour\_button.tick();

if (hour\_button.isSingle())

is\_button\_presed = true;

digitalWrite(leds\_hour[i], HIGH);

delay(50);

}

for (size\_t i = 0; i < num\_leds\_hour && !is\_button\_presed; i++) {

hour\_button.tick();

if (hour\_button.isSingle())

is\_button\_presed = true;

digitalWrite(leds\_hour[i], LOW);

delay(50);

}

}

}

// if (minute != String(watch.gettime("i")).toInt()) {

hour = String(watch.gettime("h")).toInt();

minute = String(watch.gettime("i")).toInt();

for (size\_t i = 0; i < num\_leds\_hour; i++)

digitalWrite(leds\_hour[num\_leds\_hour - i - 1], bitRead(hour, i)); // Установка пинов по времени для часов

for (size\_t i = 0; i < num\_leds\_minute; i++)

digitalWrite(leds\_minute[num\_leds\_minute - i - 1], bitRead(minute, i)); // Установка пинов по времени для минут

// }

// wdt\_enable(WDTO\_1S); // Сон 1 секунду

// WDTCSR |= (1 << WDIE);

// set\_sleep\_mode(SLEEP\_MODE\_PWR\_DOWN);

// sleep\_mode();

}

ISR (WDT\_vect) {

wdt\_disable();