УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Проектирование вычислительных систем»

**Лабораторная работа №1**

Вариант 15

Студент

*Крюков А. Ю.*

*Патутин В. М.*

*P34101*

Преподаватель

*Пинкевич В. Ю.*

Санкт-Петербург, 2022 г.

Задание лабораторной работы

Разработать и реализовать драйверы управления светодиодными индикаторами и чтения состояния кнопки стенда SDK-1.1М (индикаторы и кнопка расположены на боковой панели стенда). Функции и другие компоненты драйверов должны быть универсальными, т. е. пригодными для использования в любом из вариантов задания и не должны содержать прикладной логики программы. Функции драйверов должны быть неблокирующими, то есть не должны содержать задержек на определенное время с использованием активного ожидания (функция HAL\_Delay() и собственные варианты аналогичной реализации), а также активного ожидания событий в циклах. Написать программу с использованием разработанных драйверов в соответствии с вариантом задания.

Вариант задания

Реализовать «кодовый замок». После ввода единственно верной последовательности из не менее чем восьми коротких и длинных нажатий должен загореться зелѐный светодиод, обозначающий «открытие» замка. Светодиод горит некоторое время, потом гаснет, и система вновь переходит в «режим ввода». Каждый неправильно введѐнный элемент последовательности должен сопровождаться миганием красного светодиода и сбросом в «начало», каждый правильный – миганием жѐлтого. После трѐх неправильных вводов начинает мигать красный светодиод, и через некоторое время возвращается в «режим ввода». Если код не введен до конца за некоторое ограниченное время, происходит сброс в «начало».

Исходный код

Функция отвечающая за проверку правильности ввода нажатий:

void checkSequency(const bool \* sequency, bool isLongPress, int\* iterator, uint32\_t ms, uint32\_t\* last\_key\_pressed\_time, uint32\_t\* yellow\_off\_time, uint32\_t\* red\_off\_time, uint32\_t\* wrong\_tries, uint32\_t\* last\_wrong\_time){

(\*last\_key\_pressed\_time) = ms;

if (sequency[\*iterator] == isLongPress) {

(\*iterator)++;

(\*yellow\_off\_time) = ms + 300;

} else {

(\*iterator) = 0;

(\*red\_off\_time) = ms + 300;

(\*wrong\_tries)++;

(\*last\_wrong\_time) = ms;

}

}

Основная программа:

int main(void)

{

HAL\_Init();

SystemClock\_Config();

MX\_GPIO\_Init();

uint8\_t short\_state = 0;

uint8\_t long\_state = 0;

uint32\_t time\_key1 = 0;

uint32\_t opened\_time = 0;

uint32\_t last\_key\_pressed\_time = 0;

uint32\_t yellow\_off\_time = 0;

uint32\_t red\_off\_time = 0;

uint32\_t wrong\_tries = 0;

uint32\_t last\_wrong\_time = 0;

bool opened = false;

int iterator = 0;

const bool sequency[SEQUENCY\_LENGTH] = { false, false, false, true, true, true, false, false, false};

bool green\_state = false;

bool yellow\_state = false;

bool red\_state = false;

bool old\_green\_state = false;

bool old\_yellow\_state = false;

bool old\_red\_state = false;

while (1)

{

uint32\_t ms = HAL\_GetTick();

if (green\_state != old\_green\_state) {

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_13, green\_state ? GPIO\_PIN\_SET : GPIO\_PIN\_RESET);

old\_green\_state = green\_state;

}

if (yellow\_state != old\_yellow\_state) {

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_15, yellow\_state ? GPIO\_PIN\_SET : GPIO\_PIN\_RESET);

old\_yellow\_state = yellow\_state;

}

if (red\_state != old\_red\_state) {

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_14, red\_state ? GPIO\_PIN\_SET : GPIO\_PIN\_RESET);

old\_red\_state = red\_state;

}

// todo приколы с переполнением

if (wrong\_tries == 3 && (ms - last\_wrong\_time) < 5000 ){ // мигать красным светодиодом

red\_state = ((ms - last\_wrong\_time) / 200) % 2 == 0;

continue;

}

if (wrong\_tries >= 3){

wrong\_tries = 0;

}

red\_state = false;

// todo приколы с переполнением

green\_state = opened && (ms - opened\_time) < 10000; // зажечь зеленый светодиод

if (green\_state){

red\_state = false;

yellow\_state = false;

continue;

}

opened = false;

// yellow and red leds off by time

yellow\_state = ms > yellow\_off\_time;

red\_state = ms > red\_off\_time;

if (ms - last\_key\_pressed\_time > 5000 && iterator > 0 ){

iterator = 0;

red\_off\_time = ms + 800;

continue;

}

uint8\_t key1\_state = HAL\_GPIO\_ReadPin(GPIOC, GPIO\_PIN\_15);

if(key1\_state == 0 && !short\_state && (ms - time\_key1) > 50)

{

short\_state = 1;

long\_state = 0;

time\_key1 = ms;

}

else if(key1\_state == 0 && !long\_state && (ms - time\_key1) > 1000)

{

long\_state = 1;

checkSequency(sequency, true, &iterator, ms, &last\_key\_pressed\_time, &yellow\_off\_time, &red\_off\_time, &wrong\_tries, &last\_wrong\_time);

}

else if(key1\_state == 1 && short\_state && (ms - time\_key1) > 50)

{

short\_state = 0;

time\_key1 = ms;

if(!long\_state)

{

checkSequency(sequency, false, &iterator, ms, &last\_key\_pressed\_time, &yellow\_off\_time, &red\_off\_time, &wrong\_tries, &last\_wrong\_time);

}

}

if (iterator >= SEQUENCY\_LENGTH){

opened\_time = ms;

iterator = 0;

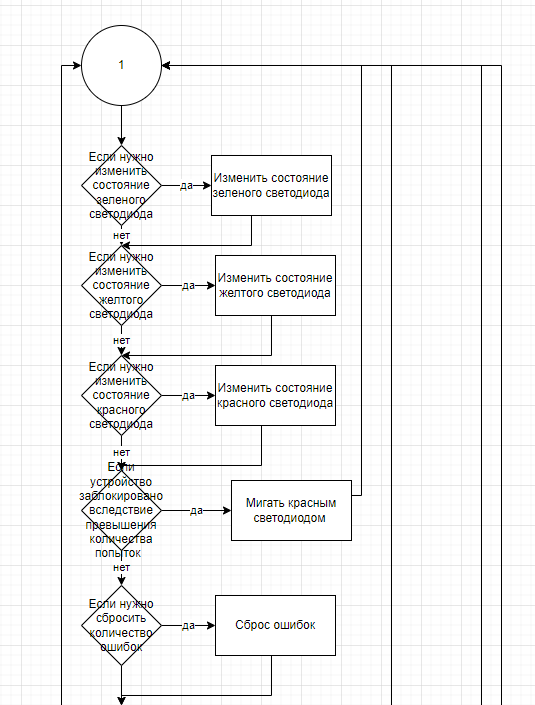
opened = true;

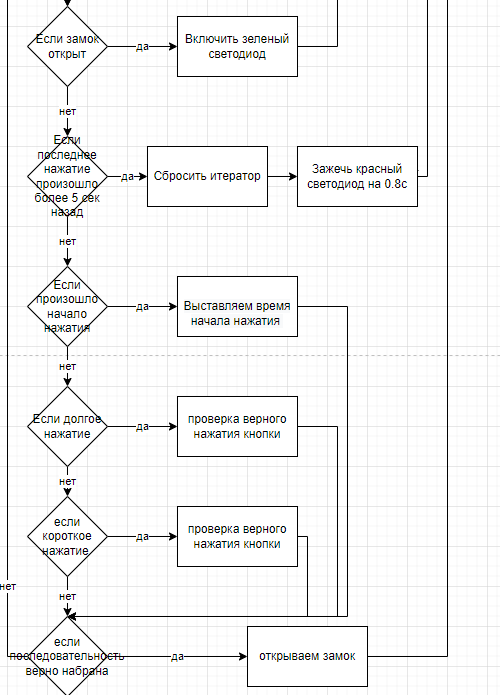
}

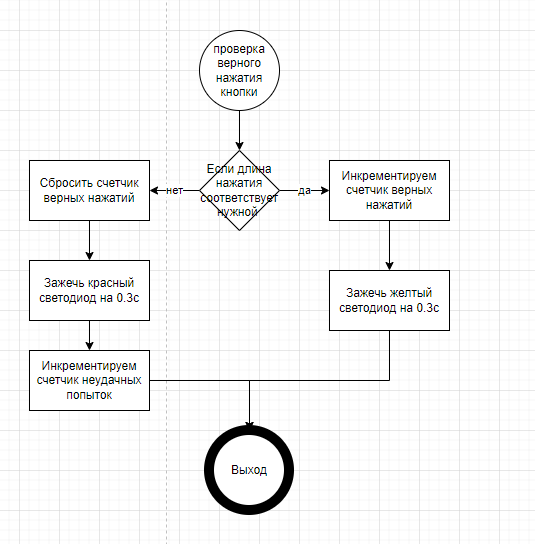
}

}

Блок схема







Вывод

Во время выполнения лабораторной работы мы получили базовые знания об устройстве стенда SDK 1.1M и изучили работу с интерфейсами ввода-вывода в микроконтроллерах, разработали собственную программу для управления световыми индикаторами в которой продемонстрировали полученные знания.