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-1.1M (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;
```

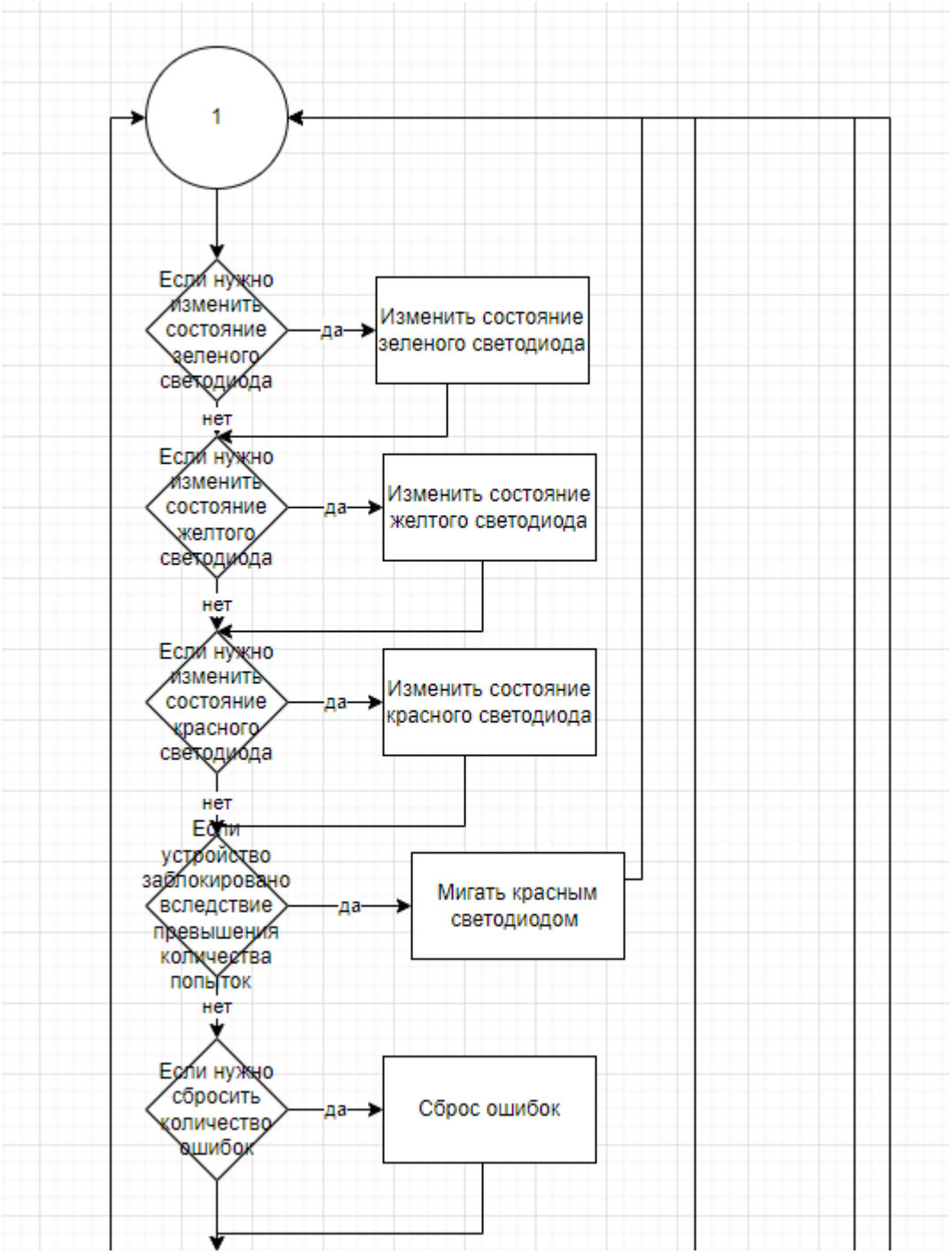
```
  uint32_t last_key_pressed_time = 0;
  uint32_t yellow_off_time = 0;
  uint32_t red_off_time = 0;



  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();
```

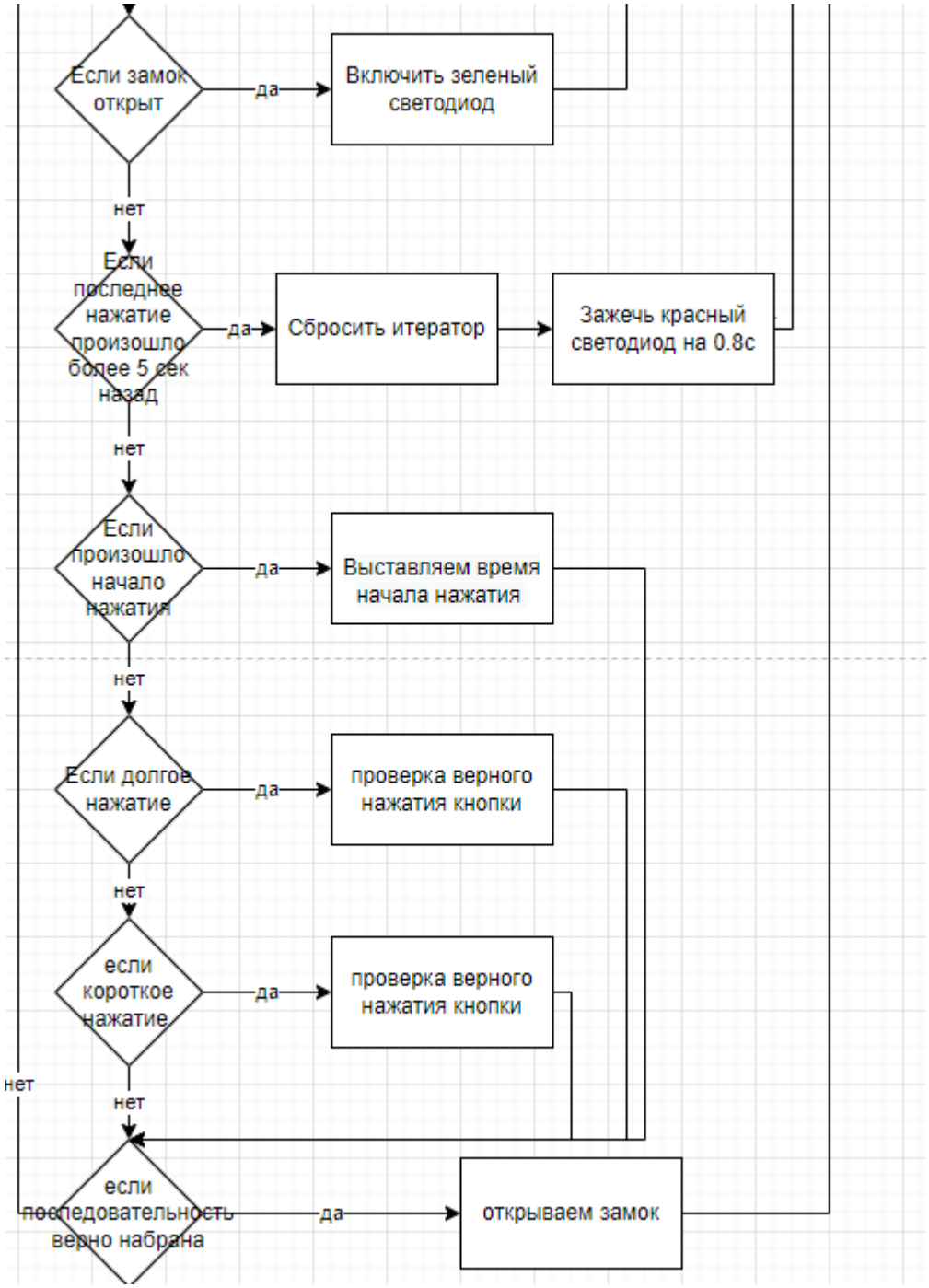
```
    if (green_state != old_green_state) {

        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;
```

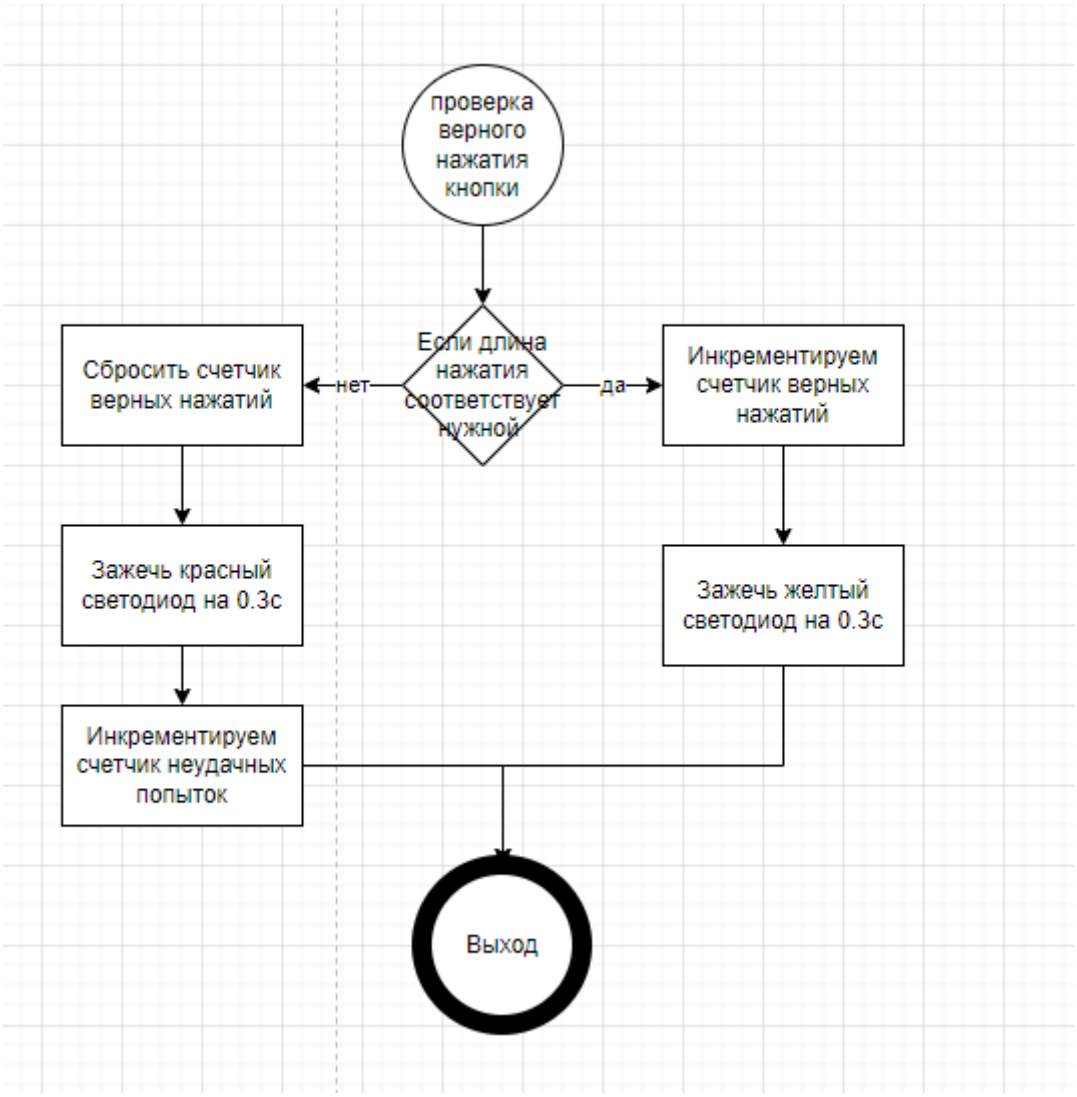
```
// todo приколы с переполнением
green_state = opened && (ms - opened_time) < 10000; // зажечь зеленый светодиод
	if (green_state){
		red_state = false;
		yellow_state = false;
		continue;
	}
opened = false;

// 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


1
Если нужно изменить состояние зеленого светодиода
да
Изменить состояние зеленого светодиода
нет
Если нужно изменить состояние желтого светодиода
да
Изменить состояние желтого светодиода
нет
Если нужно изменить состояние красного светодиода
да
Изменить состояние красного светодиода
нет
Если устройство заблокировано вследствие превышения количества попыток
да
Мигать красным светодиодом
нет
Если нужно сбросить количество ошибок
да
Сброс ошибок


Если замок открыт
да
Включить зеленый светодиод
нет
Если последнее нажатие произошло более 5 сек назад
да
Сбросить итератор
Зажечь красный светодиод на 0.8с
нет
Если произошло начало нажатия
да
Выставляем время начала нажатия
нет
Если долгое нажатие
да
проверка верного нажатия кнопки
нет
если короткое нажатие
да
проверка верного нажатия кнопки
нет
нет
если последовательность верно набрана
да
открываем замок



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.