Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО».

Факультет программной инженерии и компьютерной техники

# Мастер класс №1
Вариант №1

Выполнил
Путинцев Данил Денисович
Группа P3307
ИСУ: 409425
Проверил(а)
Преподаватель:
Гончаров Алексей Андреевчи

Санкт-Петербург 2025 год

## Текст задания

Разработать систему управления доступом с использованием 4-разрядного семисегментного индикатора, мембранной клавиатуры и светодиодов для имитации процесса ввода кода из 4-ёх цифр.

### Задачи:

1. Реализовать динамическую индикацию для вывода символов на 4-разрядный семисегментный индикатор.
2. Организовать опрос мембранной клавиатуры 3x4.
3. Реализовать конечный автомат состояний для управления меню системы.
4. Реализовать обработку ввода и проверку PIN-кода.
5. Использовать светодиоды для индикации состояния системы (доступ разрешен/запрещен, режим настройки).

## Ссылка на реализацию

https://wokwi.com/projects/442600401165747201

## Изменения в коде

**Подключил еще 2 светодиода:**

```
void initGPIO() {
  // Включаем тактирование GPIOA и GPIOB
  RCC->AHBENR |= RCC_AHBENR_GPIOAEN | RCC_AHBENR_GPIOBEN;

  GPIOA->MODER = (GPIOA->MODER & ~(3 << 10 | 3 << 18 | 3 << 30))
              | (1 << 10) | (1 << 18) | (1 << 30);

  GPIOA->OTYPER &= ~((1 << 5) | (1 << 9) | (1 << 15));
  GPIOA->OSPEEDR |= (1 << 10) | (1 << 18) | (1 << 30);
}
```

**Переписал логику в main:**

```
int main(void) {
```

```
  initGPIO();
  initUSART2();
  initSysTick();
  initKeyboard();
  tm1637_init();

  printf("Hello, %s!\n", "Wokwi Simulation");

  while (1) {
    if (code > 0) {
        if (counter == code) {
            GPIOA->ODR = (GPIOA->ODR & ~(1 << 15)) | (1 << 9);
        }
        else {
            GPIOA->ODR = (GPIOA->ODR & ~(1 << 9)) | (1 << 15);
        }
    }
    if (has_code) {
      GPIOA->ODR = (GPIOA->ODR & ~((1 << 9) | (1 << 15))) | (1 << 5);
    }
    else {
      GPIOA->ODR &= ~(1 << 5);
    }
    tm1637_update();
    scanKeyboard();
  }

  return 0;
}
```

**Завел новые переменные для логики:**

```
has_code = 0;
code = 0;
buffer_symbol = '\0';
```

**Добавил новых функций в tm1637.c:**

```
void tm1637_update(void) {
  if ((tickCount - last_display_update) >= 1000) {
    tm1637_delete_last_digit();
    tm1637_clear_display();
    tm1637_display_number(counter);
    tm1637_input_spec_symbol();
    tm1637_input_digit();
  }
}

void tm1637_input_spec_symbol(void) {
  if (buffer_symbol == '*') {
    if (lastKey == '*') {
      has_code = 1;
      buffer_symbol = '\0';
      lastKey = '\0';
      code = 0;
    }
    // else if (lastKey == '#') {
    //   has_code = 0;
    //   buffer_symbol = '\0';
    // }
  }
  else if (buffer_symbol == '#') {
```

```
    if (lastKey == '#') {
      has_code = 0;
      code = counter;
      buffer_symbol = '\0';
      lastKey = '\0';
      counter = 0;
    }
  }
  else if (lastKey == '*' || lastKey == '#') {
    buffer_symbol = lastKey;
    lastKey = '\0';
  }
  else {
    buffer_symbol = '\0';
  }

}

//Функция для ввода символов на 4-х разрядный индикатор
void tm1637_input_digit(void) {
  if (lastKey != '\0' && lastKey != '*' && lastKey != '#' && counter < 1000) {
    int new_digit = lastKey - '0';
    counter *= 10;
    counter += new_digit;
    tm1637_display_number(counter);
    lastKey = '\0';
  }
}

//При нажатии на * удалить последний символ
```

```
void tm1637_delete_last_digit(void) {
  if (lastKey == '*' && counter > 0 && has_code == 0) {
    if (counter < 10) {
      counter = 0;
    }
    else {
      counter /= 10;
    }
    tm1637_display_number(counter);
    lastKey = '\0';
  }
}


void tm1637_clear_display(void) {
    if (lastKey == '#' && has_code == 0) {
      counter = 0;
      tm1637_display_number(counter);
      lastKey = '\0';
    }
}
```

# Инструкция эксплуатации

## Режим настройки

Для входа в режим настройки необходимо нажать два раза подряд на кнопку «*»

Во время режима настройки будет гореть led1 зеленым цветом.

Во время режима настройки необходимо ввести код, который устройство должно запомнить. По сути задается код доступа.

Код доступа должен быть меньше 10000 и больше 1

Чтобы сохранить код доступа, нажмите два раза подряд на кнопку «#». На индикаторе сбросится код, а led1 перестанет гаснуть. Тем самым мы вышли из режима настройки

### Рабочий режим

Если не один светодиод не горит — значит код не установлен. Перейдите в режим настройки для задания кода

После выхода из режима настройки будет гореть всегда гореть один из светодиодов.

Код не может быть меньше или равен 0

Если код правильный, то будет гореть зеленый светодиод

Если код неверный, то будет гореть красный светодиод

При нажатии на клавишу «*» стирается последний символ

При нажатии на клавишу «#» стираются все символы

Чтобы перейти в режим настройки на индикаторе должен быть 0

## Выводы

Во время выполнения данной лабораторной работы я вспомнил язык программирования C, познакомился с ключевым словом volatile, узнал чем embedded C отличается от desktop C и научился писать код, когда цикл в функции main() никогда не заканчивается