

# Lampička s STM8

Technická dokumentace

Masopust Patrik

3.A

## **Původní zadání:**

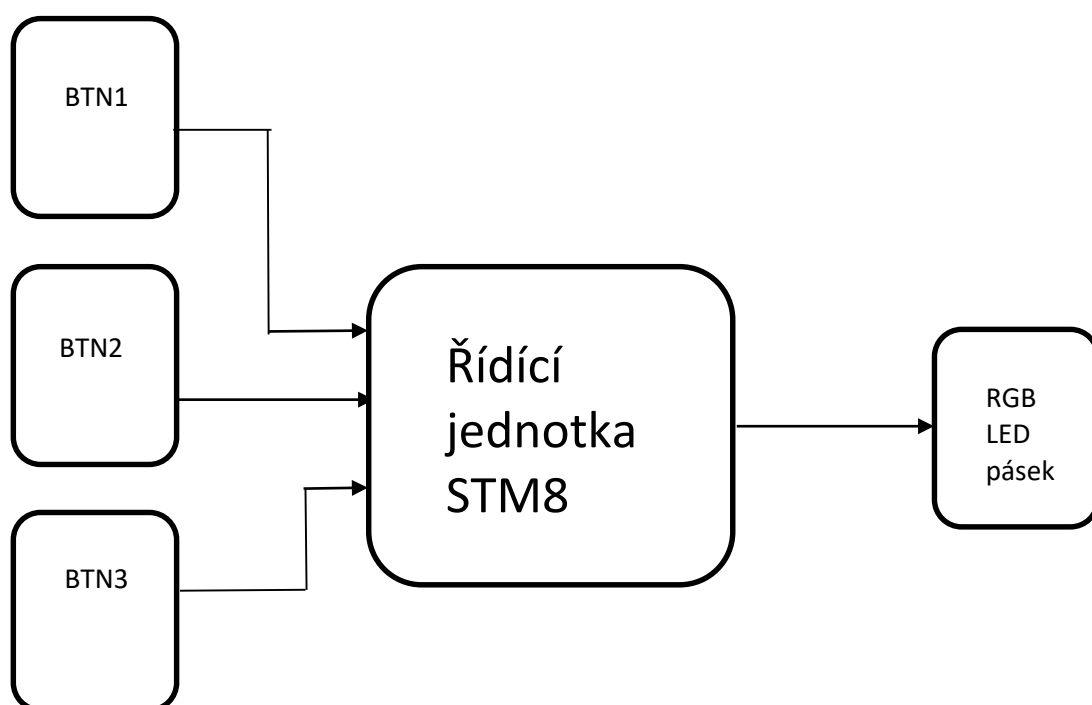
# **Lampička s STM-8**

## **Ročníkový projekt do MIT – 3 ročník**

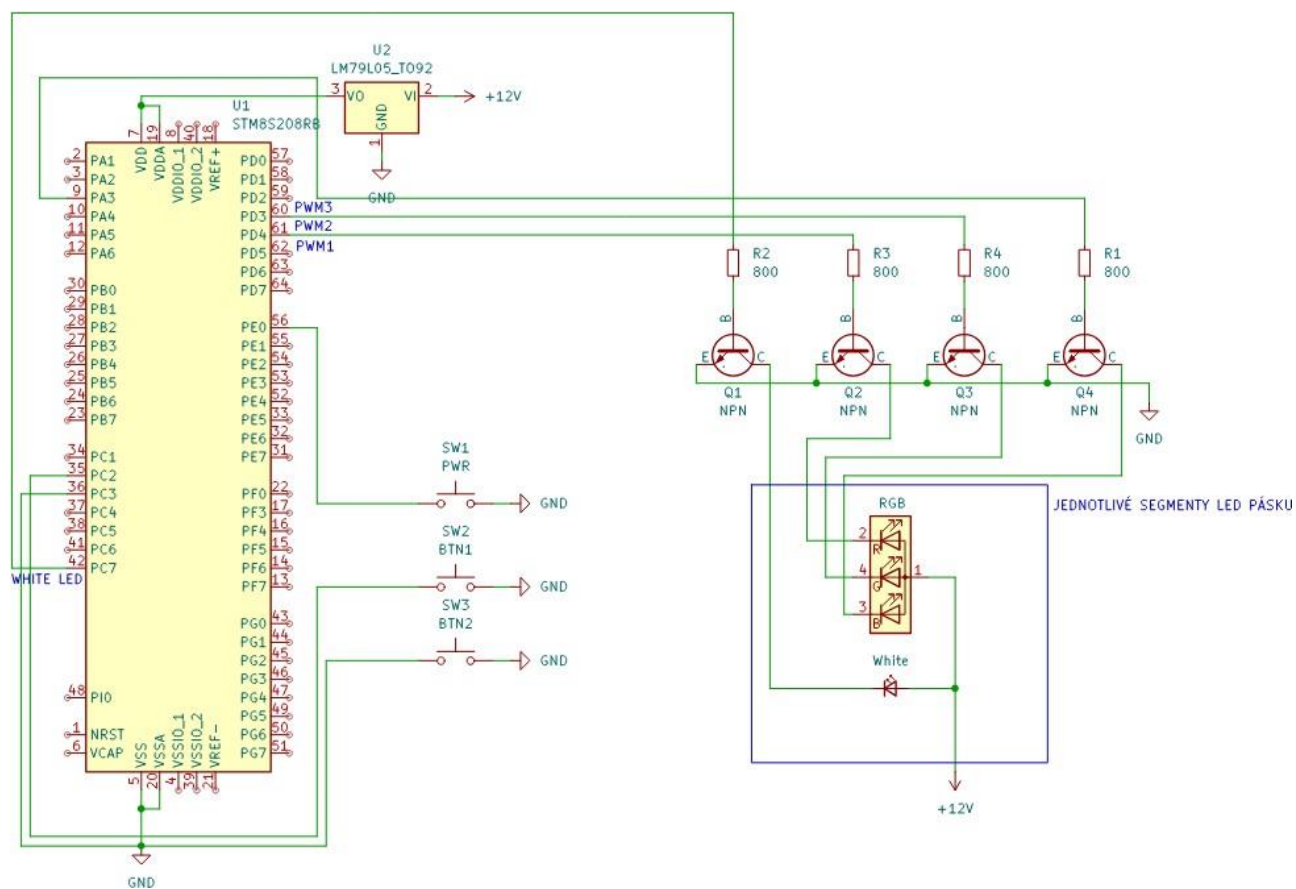
Vnukla se mi myšlenka vytvořit lampičku za použití STM8 NUCLEO jako mozku celé lampičky. Samotnou lampičku, respektive její kostru, si vytisknu na 3D tiskárně, pravděpodobně z materiálu PLA, který je hezčí a více se v něm ztratí nedokonalosti tiskárny. O svícení v lampičce se postará LED pásek, který propojím a bude ovládán mikroprocesorem. Ovládání bude prováděno pomocí tří tlačítek. První tlačítko bude zapínat samotnou lampičku (ON/OFF button), druhé tlačítko bude zapínat svícení bílou barvou, protože většinou chceme svítit bíle (např. při čtení) a třetí tlačítko bude přepínat barevné módy, které budou svítit v pozadí (červená, modrá, fialová, zelená,...). Lampičku budu napájet pravděpodobně ze sítě.

Kód pro mikroprocesor napíši v jazyce C za použití toolchainu vytvořeného Panem učitelem Nožkou. Plánuji využít znalosti o PWM a Timeru nabyté v hodinách mikroprocesorové techniky.

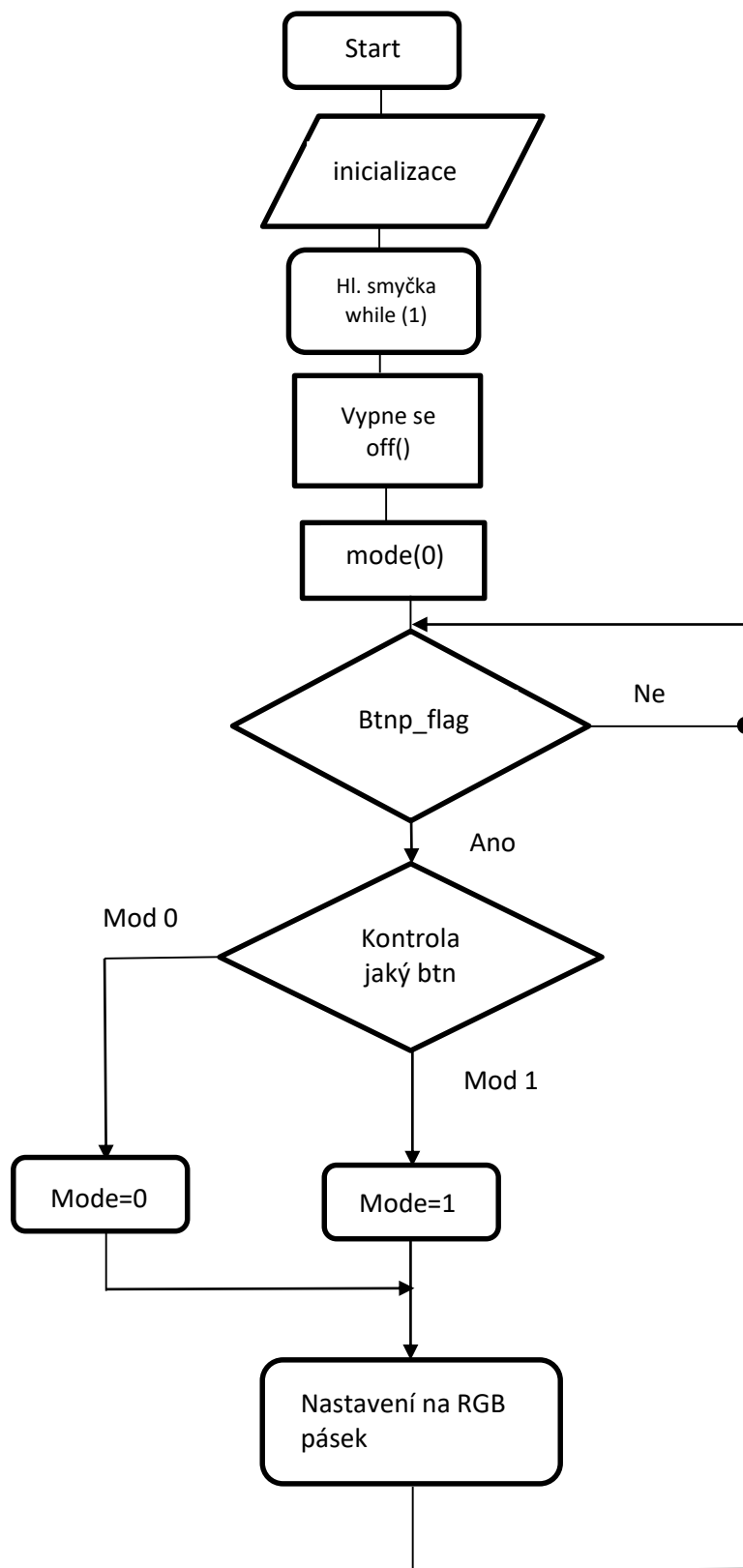
## Blokové schéma:



## Schéma zapojení:



## Vývojový diagram:



## **Poznámky k vývojovému diagramu:**

- Diagram začíná inicializací systému (init funkce), poté se přechází do hlavní smyčky (while (1)).
- V hlavní smyčce se nejprve vypne systém (off funkce) a nastaví se výchozí režim (mode = 0).
- Pokud je btnp\_flag nastaven, zapne se systém (pwr\_state = 1) a vnitřní smyčka (while (pwr\_state)) se spustí.
- Ve vnitřní smyčce se nejprve systém zapne (on funkce) a kontrolují se stavy tlačítek (btnp\_flag, btn1\_flag, btn2\_flag).
- Podle režimu (mode) se spouští různé příkazy v switch struktuře.
- Režimy 0 až 3 zahrnují různé akce pro nastavení barev a střídý PWM.
- Po ukončení vnitřní smyčky (while (pwr\_state)) se vrátí do hlavní smyčky (while (1)).

## **Popis kódu:**

### **Záhlaví a knihovny:**

```
#include <stdbool.h>
#include <stm8s.h>
#include "main.h"
#include "milis.h"
#include "delay.h"
```

### **Definice pinů a portů:**

```
#define BTNP_PORT GPIOE #define
BTNP_PIN GPIO_PIN_0 #define
BTN1_PORT GPIOC
```

```
#define BTN1_PIN GPIO_PIN_2 #define  
BTN2_PORT GPIOC #define BTN2_PIN  
GPIO_PIN_3 #define WHITE_PORT  
GPIOC #define WHITE_PIN GPIO_PIN_7
```

## **Globální proměnné:**

„volatile“ označuje, že proměnné mohou být změněny v přerušení.

```
volatile bool btnp_flag = 0;  
volatile bool btn1_flag = 0;  
volatile bool btn2_flag = 0;
```

## **Inicializační funkce:**

Inicializuje časovač TIM2 pro PWM (Pulse Width Modulation).

```
void init_tim2(void) {}
```

Inicializuje piny pro tlačítka.

```
void init_btns(void) {}
```

Hlavní inicializační funkce, která volá jednotlivé inicializační funkce.

```
void init(void) {  
    CLK_HSIPrescalerConfig(CLK_PRESCALER_HSIDIV1); init_milis();  
    init_tim2();  
    init_btns();  
    GPIO_Init(GPIOC, GPIO_PIN_1, GPIO_MODE_OUT_PP_LOW_FAST);  
    GPIO_Init(WHITE_PORT, WHITE_PIN, GPIO_MODE_OUT_PP_LOW_SLOW);  
}
```

## **Funkce pro ovládání PWM:**

Nastavují střídu PWM pro jednotlivé kanály.

```
void set_duty_pwm1(uint8_t duty) {  
    TIM2_SetCompare1((uint16_t)duty * 10);  
}
```

```
void set_duty_pwm2(uint8_t duty) {  
    TIM2_SetCompare2((uint16_t)duty * 10);  
}
```

```
void set_duty_pwm3(uint8_t duty) {  
    TIM2_SetCompare3((uint16_t)duty * 10);  
}
```

Povolují nebo zakazují jednotlivé PWM kanály.

```
void enable_pwm1(bool state) {  
    TIM2_CCxCmd(TIM2_CHANNEL_1, state ? ENABLE : DISABLE);  
}
```

```
void enable_pwm2(bool state) {  
    TIM2_CCxCmd(TIM2_CHANNEL_2, state ? ENABLE : DISABLE);  
}
```

```
void enable_pwm3(bool state) {  
    TIM2_CCxCmd(TIM2_CHANNEL_3, state ? ENABLE : DISABLE);  
}
```

## **Funkce pro převod barvy na PWM:**

Převádí hexadecimální barvu na střídu PWM.

```
void color_hex_to_duty_cycle(uint32_t color, uint8_t* duty_red, uint8_t* duty_green, uint8_t*  
duty_blue) {  
  
    uint8_t red = (color >> 16) & 0xFF; uint8_t  
green = (color >> 8) & 0xFF;
```



```
uint8_t blue = color & 0xFF;

*duty_red = (uint8_t)((red / 255.0) * 100);

*duty_green = (uint8_t)((green / 255.0) * 100);

*duty_blue = (uint8_t)((blue / 255.0) * 100);

}
```

### **Funkce pro nastavení barvy:**

Nastavuje barvu pomocí PWM kanálů.

```
void set_color(uint32_t color) {
    uint8_t duty_red, duty_green, duty_blue;

    color_hex_to_duty_cycle(color, &duty_red, &duty_green, &duty_blue);

    set_duty_pwm1(duty_red);

    set_duty_pwm2(duty_green);

    set_duty_pwm3(duty_blue);

}
```

### **Funkce pro zapnutí a vypnutí:**

Zapínají a vypínají všechny PWM kanály a LED.

```
void off(void) { enable_all_pwm(false);
    GPIO_WriteLow(WHITE_PORT, WHITE_PIN);
}

void on(void) { enable_all_pwm(true);
}
```

## Hlavní funkce main :

Hlavní smyčka programu, která kontroluje stavy tlačítek a podle toho mění módy a barvy LED.

```
int main(void) {  
    init();  
  
    uint8_t mode;  
    bool pwr_state = 0;  
  
    while(1) {  
        off();  
  
        mode = 0;  
  
        if (btnp_flag) { pwr_state =  
            1;  
            btnp_flag = 0;  
        }  
  
        while (pwr_state) { on();  
            if (btnp_flag) {  
                pwr_state = 0;  
                btnp_flag = 0;  
                mode = 0;  
            }  
  
            if (btn1_flag) {  
                mode++;  
                if (mode > 3) mode = 2;  
                btn1_flag = 0;  
            }  
        }  
    }  
}
```

```
if(btn2_flag){  
    mode = 0;  
    btn2_flag = 0;  
}  
  
switch (mode) {  
    case 0:  
        set_duty_pwm1(0);  
        set_duty_pwm2(0);  
        set_duty_pwm3(0);  
        GPIO_WriteHigh(WHITE_PORT, WHITE_PIN);  
        break;  
    case 1:  
        set_duty_pwm1(100);  
        set_duty_pwm2(100);  
        set_duty_pwm3(100);  
        GPIO_WriteHigh(WHITE_PORT, WHITE_PIN);  
        break;  
    case 2:  
        GPIO_WriteLow(WHITE_PORT, WHITE_PIN);  
        for (uint8_t i = 0; i < 255; i++) {  
            set_color(merge_to_hex(i, 0, 254 - i)); delay_ms(10);  
            if (btn1_flag || btn2_flag || btnp_flag) break;  
        }  
        for (uint8_t i = 0; i < 255; i++) {
```

```

        set_color(merge_to_hex(254 - i, 0, i)); delay_ms(10);

        if (btn1_flag || btn2_flag || btnp_flag) break;
    }

    break;

case 3:
    GPIO_WriteLow(WHITE_PORT, WHITE_PIN);

    for (uint8_t i = 0; i < 255; i++) {
        set_color(merge_to_hex(0, i, 254 - i)); delay_ms(10);

        if (btn1_flag || btn2_flag || btnp_flag) break;
    }

    for (uint8_t i = 0; i < 255; i++) {
        set_color(merge_to_hex(0, 254 - i, i)); delay_ms(10);

        if (btn1_flag || btn2_flag || btnp_flag) break;
    }

    break;
}

}

}

}

```

## Rutina přerušení:

Přerušení pro tlačítka na portu C.

```
INTERRUPT_HANDLER(EXTI_PORTC_IRQHandler, 5) { if  
    (PUSH(BTN1)) {  
        btn1_flag = 1;  
    } else if (PUSH(BTN2)) { btn2_flag = 1;  
    }  
}
```

Přerušení pro tlačítko na portu E.

```
INTERRUPT_HANDLER(EXTI_PORTE_IRQHandler, 7) {  
    btnp_flag = 1;  
}
```

## **Závěr:**

Projekt „Lampička s STM8“ splnil veškeré cíle a požadavky stanovené v úvodním zadání. Hlavní myšlenkou bylo vytvořit plně funkční lampičku, která by nabízela klasické bílé osvětlení a také různé barevné režimy pomocí RGB LED pásku. K dosažení tohoto cíle bylo využito vývojové desky STM8 NUCLEO jako základní řídicí jednotky, tři tlačítek které byly umístěny na kostru lampičky, kterou jsem vytiskl na 3D tiskárně s využitím materiálu PLA, který byl zvolen pro jeho vhodné estetické vlastnosti a snadnou zpracovatelnost.

Po hardwarové stránce bylo navrženo a realizováno stabilní zapojení, které zajistilo spolehlivou funkčnost všech komponent. Díky pokročilé práci s PWM (Pulse Width Modulation) bylo možné přesně řídit intenzitu a barvy světla. Ovládací logika byla naprogramována v jazyce C s využitím nástrojů dostupných pro STM8.

Výsledkem projektu je intuitivní ovládání lampičky prostřednictvím tří tlačítek: zapnutí/vypnutí, aktivace bílého světla a přepínání mezi barevnými režimy.

I přesto, že projekt v současné podobě splňuje definované cíle, je zde prostor pro jeho další rozvoj. Možnosti zahrnují například přidání bezdrátového ovládání, automatizované stmívání na základě senzorů okolního světla, nebo rozšíření o další režimy barevných efektů. Další potenciál skýtá vylepšení designu konstrukce pro zajištění větší flexibility a vyššího estetického dojmu.