Lampička s STM8

Technická dokumentace Masopust Patrik

3.A

Křelov 16.06.2024

**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:**

BTN3

BTN1

RGB

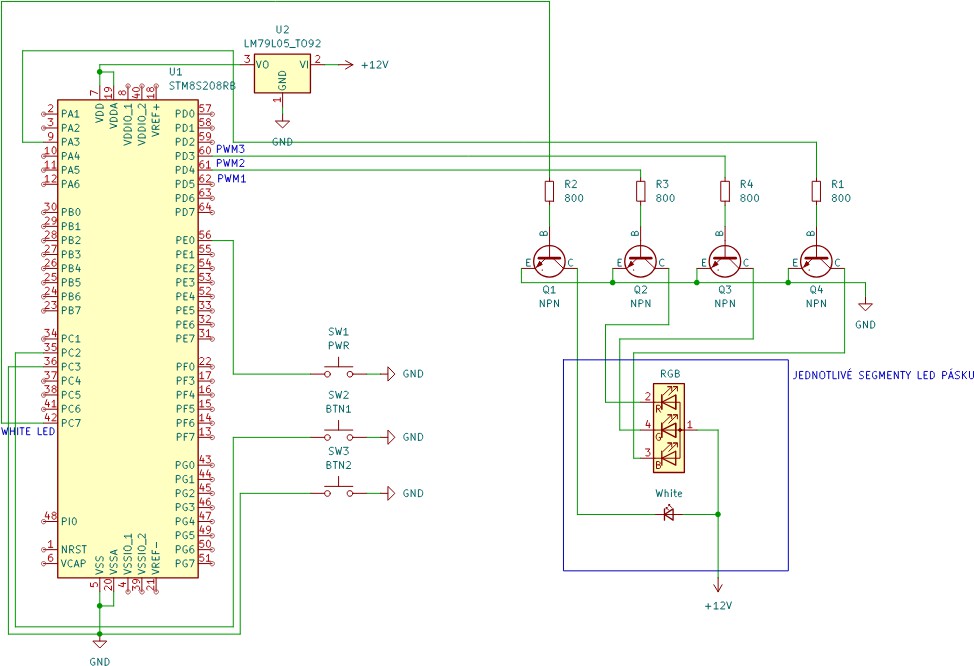
LED

pásek

BTN2

Řídící jednotka

STM8

**Schéma zapojení:**

**Vývojový diagram:**

Start

inicializace

Ne

Mod 1

Mod 0

Ano

Nastavení na RGB pásek

Mode=1

Mode=0

Kontrola jaký btn

Btnp\_flag =?

mode(0)

Vypne se off()

Hl. smyčka while (1)

**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 střídy

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 střídy 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 řídící jednotky, tří 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.