MIT

ZÁVĚREČNÝ PROJEKT – Ukazatel zařazené rychlosti na motocyklu

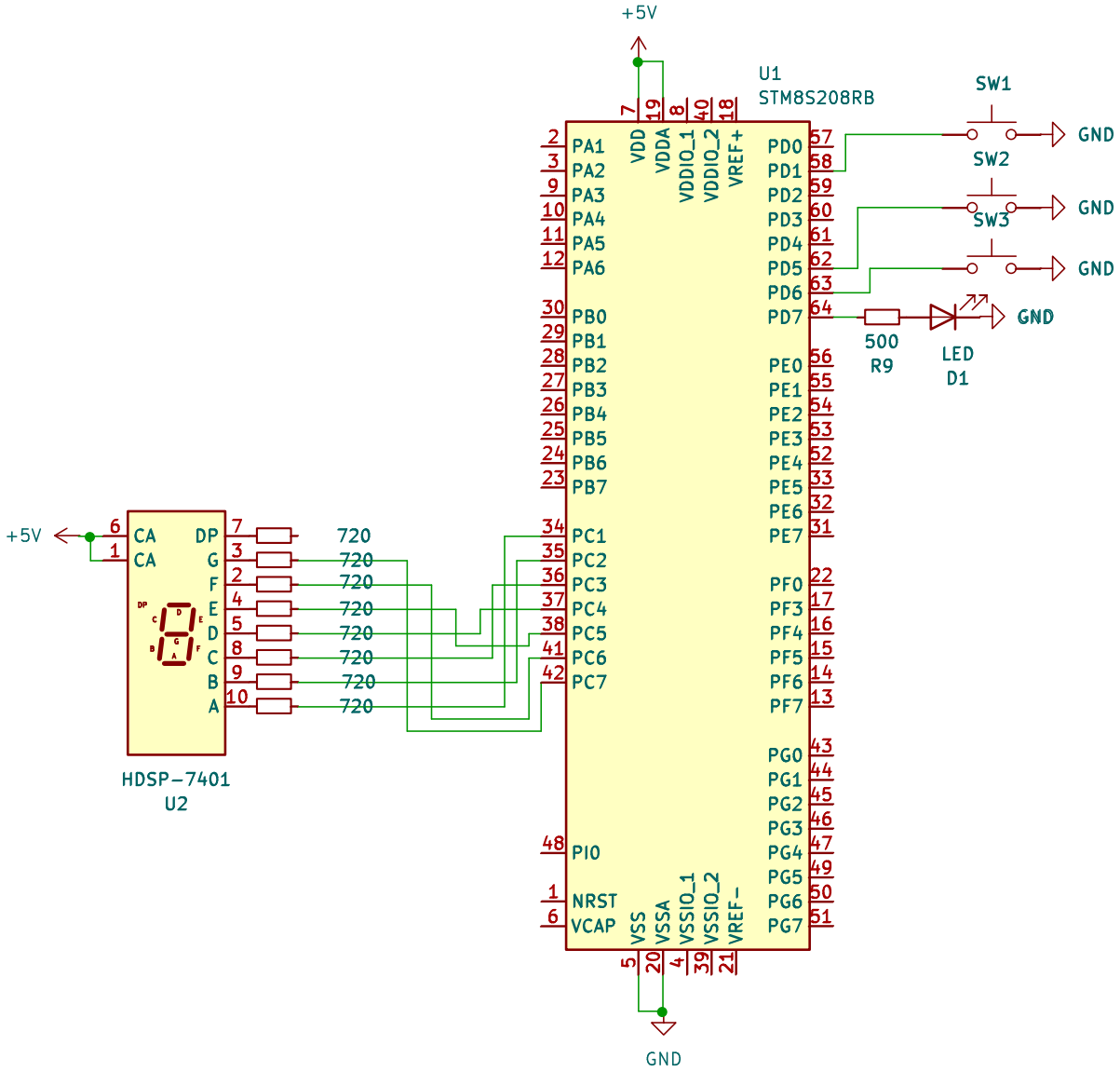
Hlavní myšlenka:

Ukazatel zařazené rychlosti s čidlem na řadící páce se vstupem neutrálu. Zobrazeno na sedmisegmentovém displeji.

Použité součástky:

- 2 čidla
- STM8
- sedmisegmentový display
- rezistory

Schéma KiCad:



Jméno: Jan Piskovský	Třída: 3.A	List 2/5

Popis činnosti programu:

Tento program umožňuje uživateli přepínat a zobrazovat převodové stupně řazení na motocyklu. Pomocí tlačítek, které slouží jako snímače, zobrazuje aktuální převodový stupeň na sedmisegmentovém displeji. Horní tlačítko převodové stupně zvyšuje a dolní tlačítko je snižuje. Tlačítka přepíná imitace řadící páky vyrobená ze dřeva. Pokud je stisknuto tlačítko do polohy "NEUTRAL", rozsvítí se zelená LED dioda jako imitace kontrolky neutrálu.

Zdrojový kód:

```
#include <main.h> // Zahrnuje hlavní hlavičkový soubor projektu
#include <utils.h> // Zahrnuje užitečné funkce a definice
#include <stm/stm8s.h> // Zahrnuje specifické definice a funkce pro STM8S mikrokontroléry
#include <stdbool.h> // Zahrnuje definice pro práci s booleovskými hodnotami
#include <stdio.h> // Zahrnuje standardní vstupní a výstupní funkce

// Definice portů a pinů
#define SEGMENT_PORT GPIOC // Port pro sedmisegmentový displej
#define INPUT_PORT GPIOD // Port pro vstupní piny
#define DOWN_PIN GPIO_PIN_1 // Pin pro tlačítko "DOWN"
#define UP_PIN GPIO_PIN_5 // Pin pro tlačítko "UP"
#define NEUTRAL_PIN GPIO_PIN_6 // Pin pro tlačítko "NEUTRAL"
#define NEUTRAL_LED_PIN GPIO_PIN_7 // Pin pro LED indikující neutrální stav
#define NEUTRAL_LED_PORT GPIOD // Port pro LED

// Hodnoty pro zobrazení čísel na sedmisegmentovém displeji
uint8_t seg_vals[10] = {0b01000000,0b11110010,0b01001000,0b10100000, 0b00110010,
0b00100100, 0b00000100, 0b11110000,0b00000000, 0b00100000};

void rx_action(void) // Funkce pro zpracování přijatých dat (bude se kompilovat pouze s touto
definicí události)
{
   char c = UART1_ReceiveData8(); // Přečte přijatý znak z UART1
}

// Obsluha externího přerušení
INTERRUPT_HANDLER(EXTI_PORTD_IRQHandler, 6) {
}

int main(void)
{
        // Inicializace hodinového systému na plnou rychlost
        CLK_HSIPrescalerConfig(CLK_PRESCALER_HSIDIV1); //Set CLK

        // Inicializace pinů pro sedmisegmentový displej
        GPIO_Init(SEGMENT_PORT, GPIO_PIN_0, GPIO_MODE_OUT_PP_LOW_SLOW);
        GPIO_Init(SEGMENT_PORT, GPIO_PIN_1, GPIO_MODE_OUT_PP_LOW_SLOW);
        GPIO_Init(SEGMENT_PORT, GPIO_PIN_2, GPIO_MODE_OUT_PP_LOW_SLOW);
        GPIO_Init(SEGMENT_PORT, GPIO_PIN_3, GPIO_MODE_OUT_PP_LOW_SLOW);
```

```
GPIO_Init(SEGMENT_PORT, GPIO_PIN_4, GPIO_MODE_OUT_PP_LOW_SLOW);
GPIO_Init(SEGMENT_PORT, GPIO_PIN_5, GPIO_MODE_OUT_PP_LOW_SLOW);
GPIO_Init(SEGMENT_PORT, GPIO_PIN_6,  GPIO_MODE_OUT_PP_LOW_SLOW);
GPIO_Init(SEGMENT_PORT, GPIO_PIN_7, GPIO_MODE_OUT_PP_LOW_SLOW);

// Inicializace pinu pro LED indikující neutrál
GPIO_Init(NEUTRAL_LED_PORT, NEUTRAL_LED_PIN,
GPIO_MODE_OUT_PP_LOW_SLOW);

// Inicializace vstupních pinů pro tlačítka
GPIO_Init(INPUT_PORT, DOWN_PIN, GPIO_MODE_IN_PU_NO_IT);
GPIO_Init(INPUT_PORT, UP_PIN, GPIO_MODE_IN_PU_NO_IT);
GPIO_Init(INPUT_PORT, NEUTRAL_PIN, GPIO_MODE_IN_PU_NO_IT);



// Nastavení počáteční hodnoty na sedmisegmentovém displeji
SEGMENT_PORT->ODR = seg_vals[0];

// Inicializace funkcí pro měření času a UART1
init_milis();
init_uart1();

// Proměnné pro uložení časových razítek a stavů tlačítek
uint16_t last_time_stamp = 0;
uint8_t old_down_state = 0;
uint8_t old_up_state = 0;
int8_t gear = 1; // NEUTRAL
int i = 0;
while(1) {
    // Kontrola stavu tlačítka "DOWN"
    if(GPIO_ReadInputPin(INPUT_PORT, DOWN_PIN) == 0){
        if(old_down_state == 0) gear++;
        old_down_state = 1;
    }else old_down_state = 0;

    // Kontrola stavu tlačítka "UP"
    if(GPIO_ReadInputPin(INPUT_PORT, UP_PIN) == 0){
        if(old_up state == 0) gear--;
        old_up_state = 1;
    }else old_up_state = 0;

    // Kontrola stavu tlačítka "NEUTRAL"
    if(GPIO_ReadInputPin(INPUT_PORT, NEUTRAL_PIN) == 0) {
        gear = 1;
        GPIO_WriteHigh(NEUTRAL_LED_PORT, NEUTRAL_LED_PIN);
    }else GPIO_WriteLow(NEUTRAL_LED_PORT, NEUTRAL_LED_PIN);

    // Omezování hodnoty proměnné "gear" na rozsah 0-6
    gear = gear < 0 ? 0 : gear > 6 ? 6 : gear;

    // Nastavení hodnoty na sedmisegmentovém displeji podle proměnné "gear"
    SEGMENT_PORT->ODR = seg_vals[gear];

    // Zpoždění
    delay(5);
}
```

```
}
/*------------------------------  Assert ----------------------------------*/
#include "stm/__assert__.h" // Zahrnuje hlavičkový soubor pro aserci
```