

**SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY**

**SEMESTRÁLNY PROJEKT - RC VYSIELAČ
(Programátorský manuál)**

Študijný program:

Robotika a kybernetika

Bratislava 2016

Bc. Jakub Vydra

Bc. Tomáš Kolek

Bc. Anna Pružinská

Bc. Branislav Dluhoš

Obsah

1 HW a SW prostriedky	3
1.1 Schéma zariadenia.....	3
2 Popis funkčných častí	4
2.1 Páky na ovládanie kanálov	4
2.2 Troj-polohový prepínač	4
2.3 Štyri tlačidlá	4
2.4 Display	4
2.5 SPI	4
2.6 Výstupný signál.....	4
3 Funkcie.....	5
3.1 MAIN.....	5
3.2 TEST 1.	5
3.2.1 Inicializácia.....	6
3.2.2 Pohyb Menu	8
3.2.2 Zhrnutie	9
3.3 SUBMENU.....	9
3.3.1 Normalizuj.....	9
3.3.2 FloatToString	9
3.3.3 FloatToStringReverz.....	Chyba! Záložka nie je definovaná.
3.3.4 Otvor_info	10
3.3.5 Otvor Revers.....	10
3.3.6 Otvor Expo	10
3.3.7 Otvor mix	10
3.3.8 EPA.....	10
4 Link na GitHub	Chyba! Záložka nie je definovaná.

1 HW a SW prostriedky

Hardvér

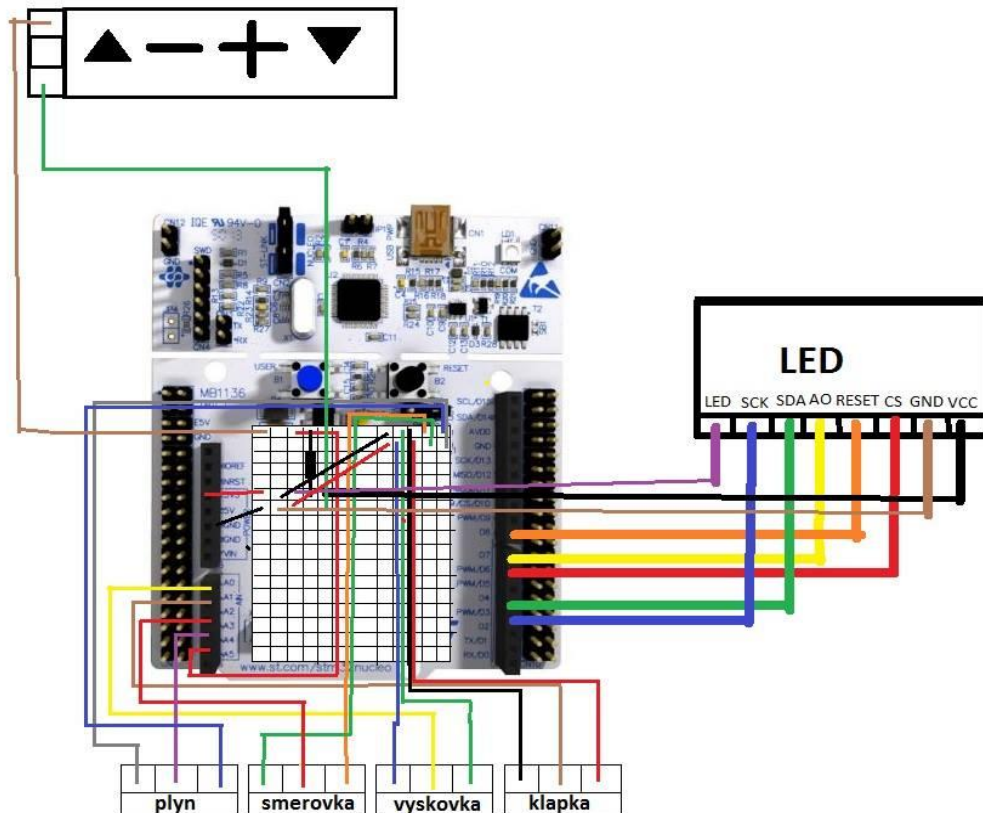
1. STM32L152RE
2. 4x potenciometer na ovládanie
3. 2x 3-polohové prepínače
4. 8k2 rezistor
5. 4 tlačidlá, display
6. Vodiče

Softvér

1. Atollic Studio
2. GitHub

1.1 Schéma zariadenia

Na obrázku číslo 1 môžeme vidieť schematický znázornený RC vysielateľ, ktorý obsahuje všetky časti potrebné na realizáciu nášho projektu.



Obrázok 1 Schéma

2 Popis funkčných častí

Základ bude tvoriť doska STM32L152RE. K doske budú pripojené potenciometre, ktorými sa bude simulovať ovládanie jednotlivých kanálov RC vysielačky. Takisto pomocou kontaktného poľa budú pripojené na dosku aj 4 tlačidlá, 2x 3-polohové prepínače a display.

2.1 Páky na ovládanie kanálov

Štyri kanály zabezpečujú ovládanie motora, výškovky, smerovky a klapiek. Každý kanál okrem ovládania motora sa jeho „knypel“ vráti do počiatočnej polohy (0, 0).

2.2 Troj-polohový prepínač

Ďalší kanál slúži na ovládanie podvozku, kde pre jednotlivé pozície prepínača sú zadefinované stavy podvozku ako vysunutý, medzi poloha a zasunutý.

2.3 Štyri tlačidlá

Tieto tlačidlá slúžia na pohyb v menu ovládača. Tlačidlá reprezentujú pohyb šípky v menu hore, dole, „do“ menu (enter) a „z“ menu (back).

2.4 Display

Display slúži na zobrazovanie informácií a menu, ktoré je popísané vyššie.

2.5 SPI

Komunikácia je zabezpečená pomocou SPI rozhrania. SPI je štvorvodičová synchrónna sériová zbernica pracujúca v móde full duplex slúžiaca na prepojenie periférií s mikropočítačmi. Zariadenia komunikujú spôsobom master/slave.

2.6 Výstupný signál

Výstupom aplikácie sú hodnoty posielané na jednotlivé kanály v ich normalizovaných rozsahoch.

3 Funkcie

Náš semestrálny projekt sa skladá z viacerých funkcií, ktoré budú popísané v jednotlivých častiach.

3.1 MAIN

V "maine" sa nachádza samostatná iniciálizácia častí projektu(obr.2), bez ktorých by nebolo možné vykonať všetky funkcie, ktoré sú potrebné. Najprv sme inicializovali zbernicu SPI, DMA,GPIO, úvodné menu, ADC prevodník a nakoniec NVIC - radič prerušení. V "maine" sa nachádza aj funkcia, ktorá slúži na kontrolu zvolenej položky menu, ktorú si zvolí používateľ . Podľa tejto kontroly sa následne spúšťajú jednotlivé submenu.

```
int main(void)
{
    initSPI2(); //inicializujem zbernicu SPI
    dma_init(); //inicializujem DMA
    initGPIO(); //Inicializujem GPIO
    initCD_Pin(); //
    initCS_Pin();
    initRES_Pin();
    initMenu(); //inicializujem uvodne menu
    adc_init(); //inicializujem ADC
    nvic_init(); //inicializujem NVIC
    while (1)
    {
        pohybMenu(klavesnica); //funkcia
        //Delay(50);
    }
}
```

Obrázok 2 Kód main-u

3.2 TEST 1.

Je to modul, ktorý slúži na prácu s displejom. Obsahuje viacero funkcií, ktoré budú rozpísané v jednotlivých bodoch.

3.2.1 Inicializácia

Táto časť je veľmi dôležitá, pretože tu sme inicializovali všetky súčasti projektu, bez ktorých by nebola možná realizácia.

Inicializácia menu

Slúži na inicializáciu menu, kde sa zobrazí hlavné menu a vypíšu sa jednotlivé položky.

Inicializácia GPIO

V tejto časti sme inicializovali periférie GPIO. V našom projekte sme používali tri porty GPIOA,GPIOB,GPIOC. Pre všetky porty sme zapli hodiny. Nainicializovali sme pin 1 na port C, kde sme jeho režim zvolili ako analógový a nepotrebovali sme ani žiaden pull up alebo pull down. (obr.č.3)

```
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOA, ENABLE);  
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOB, ENABLE);  
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOC, ENABLE);  
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1 ;  
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;  
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL ;  
GPIO_Init(GPIOC, &GPIO_InitStructure);
```

Obrázok 3 Inicializácia GPIOC

Na port A sme inicializovali piny 0, 1,4. Taktiež sme zvolili také nastavenia ako pri prvom príklade. Na port B sme nainicializovali pin 0 s rovnakými nastaveniami.

Inicializácia Direct Memory Access (DMA)

V tejto časti sme inicializovali DMA s potrebnými nastaveniami. Nastavovali sme to na kanál 1, ktorý môžeme vidieť na obrázku č. 4.

```

dma_init(){ //inicializujem DMA1
    DMA_InitTypeDef      DMA_InitStruct;
    RCC_AHBPeriphClockCmd(RCC_AHBPeriph_DMA1, ENABLE);
    /* DMA1 Stream0 channel0 configuration **** */
    DMA_InitStruct.DMA_PeripheralBaseAddr = (uint32_t)&ADC1->DR; //ADC1's data register
    DMA_InitStruct.DMA_MemoryBaseAddr = (uint32_t)&ADC1ConvertedValue[0];
    DMA_InitStruct.DMA_DIR = DMA_DIR_PeripheralSRC;
    DMA_InitStruct.DMA_BufferSize = 5;
    DMA_InitStruct.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
    DMA_InitStruct.DMA_MemoryInc = DMA_MemoryInc_Enable;
    DMA_InitStruct.DMA_PeripheralDataSize = DMA_PeripheralDataSize_HalfWord;
    DMA_InitStruct.DMA_MemoryDataSize = DMA_MemoryDataSize_HalfWord; //Stores 10 bits
    DMA_InitStruct.DMA_Mode = DMA_Mode_Circular;
    DMA_InitStruct.DMA_Priority = DMA_Priority_High;
    DMA_Init(DMA1_Channel1, &DMA_InitStruct);
    DMA_Cmd(DMA1_Channel1, ENABLE); //Enable the DMA1 - Channel1
}

```

Obrázok 4 DMA

Inicializácia ADC prevodníka

Inicializácia ADC prevodníka je zobrazená na obrázku 5. Môžeme vidieť ako sme nastavili jeho jednotlivé súčasti. Rozlíšenie sme zvolili 12 b, konverzný mód sme zvolili continuous. Výsledok z ADC prevodníka sme zvolili tak, aby sa zobrazilo napravo. Taktiež sme tu nastavovali kanály, kde sme čas vzorkovania zvolili 9 cyklov.

```

//while(RCC_GetFlagStatus(RCC_FLAG_HSRDY) == RESET);
RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);
ADC_StructInit(&ADC_InitStructure);
ADC_InitStructure.ADC_Resolution = ADC_Resolution_12b;
ADC_InitStructure.ADC_ScanConvMode = ENABLE; //The scan is configured in multiple mode
ADC_InitStructure.ADC_ContinuousConvMode = ENABLE;
ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_None;
ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
ADC_InitStructure.ADC_NbrOfConversion = 5;
ADC_InitStructure.ADC_ExternalTrigConv = 0;
ADC_Init(ADC1, &ADC_InitStructure);
ADC_RegularChannelConfig(ADC1, ADC_Channel_0, 1, ADC_SampleTime_9Cycles); //PC0
ADC_RegularChannelConfig(ADC1, ADC_Channel_1, 2, ADC_SampleTime_9Cycles);
ADC_RegularChannelConfig(ADC1, ADC_Channel_4, 3, ADC_SampleTime_9Cycles);
ADC_RegularChannelConfig(ADC1, ADC_Channel_8, 4, ADC_SampleTime_9Cycles);
ADC_RegularChannelConfig(ADC1, ADC_Channel_11, 5, ADC_SampleTime_9Cycles);
//...

```

Obrázok 5 ADC

Inicializácia NVIC

Ako posledný krok je inicializácia radiča prerušení pre ADC prevodník. Nastavenie môžeme vidieť na obrázku 6.

```

void nvic_init(){                                     // inicializacia preruseni
    //NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);
    NVIC_InitTypeDef NVIC_InitStructure;
    //NVIC_InitStructure.NVIC_IRQChannel = ADC1_IRQn;
    NVIC_InitStructure.NVIC_IRQChannel = DMA1_Channel1_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
    DMA_ITConfig(DMA1_Channel1, DMA_IT_TC, ENABLE);
    DMA_Cmd(DMA1_Channel1, ENABLE);
    ADC_DMACmd(ADC1, ENABLE);

```

Obrázok 6 NVIC

3.2.2 Pohyb Menu

Táto funkcia slúži na pohyb v menu a na výber operácie, ktorá sa má vykonať na danom zariadení. Na základe nasledúcich podmienok (obr. č. 7) sa zavolá príslušná funkcia implementujúca danú operáciu.

```

else if ((klavesnica >= 2800) && (klavesnica <= 2940)){ //vstup do submenu
    if ((aktualneA == 0 && defaultA == 12) || aktualneA == 12){
        otvorInfo();
    }
    else if (aktualneA == 32){
        otvorRevers();
    }
    else if (aktualneA == 52){
        otvorExpo();
    }
    else if (aktualneA == 72){
        otvorMix();
    }
    else if (aktualneA == 92){
        otvorEPA();
    }
}

```

Obrázok 7 Operácie

Pohyb menu je implementovaný prostredníctvom funkcií PosunSipkyDole a PosunSipkyHore.

3.2.2 Zhrnutie

Na začiatku je potrebná inicializácia všetkých pinov, ADC prevodníka, NVIC - radiča prerušení a DMA. Po inicializácii nasleduje funkcia, ktorá vytvára menu. Defaultné nastavenie je že šípka sa nachádza na prvej položke v menu. Podľa stlačenej klávesy sa šípka prekreslí na novú pozíciu. Pri stlačení plusu sa nám na klávesnici otvorí submenu. Opustenie submenu sa vykona stlačením mínusu. Na to aby bol premazaný displej slúži funkcia `lcdClearDisplay`. Na vykreslenie šípky je funkcia `lcdPinyTrojuholník`. Funkcia pohyb menu sleduje aké tlačidlo bolo stlačené. Je veľmi dôležité poznamenať, že pracujeme s hodnotami z ADC prevodníka, ktorú získame prostredníctvom DMA a je postupne ukladaná do poľa. Pole sa naplní vždy pri prerušení.

3.3 SUBMENU

V tejto časti sa budeme venovať jednotlivým funkciám, ktoré má podľa zadania náš RC vysielateľ vykonávať.

3.3.1 Normalizuj

Táto funkcia slúži na normalizáciu hodnôt, ktoré dostaneme z ADC prevodníka. V praxi to znamená, že dané hodnoty interpolujeme do intervalu $<-1,1>$

```
float normalizuj(float hodnota, float hodnotaMIN, float hodnotaMAX, bool hodnotaREV, bool hodnotaEXP, float hodnotaTRIM) {
    NORM = ((hodnota - hodnotaMIN) / (hodnotaMAX - hodnotaMIN)) * (1 - (-1)) + (-1);
    if (hodnotaREV == true) {
        NORM = NORM * (-1);
    }
    if (hodnotaEXP == true) {
        if (NORM < 0) {
            NORM = exp2(NORM * (-1)) - 1;
            NORM = NORM * (-1);
        }
        else {
            NORM = exp2(NORM) - 1;
            NORM = NORM * (-1);
        }
    }
    NORM = NORM + hodnotaTRIM;
    return NORM;
}

float normalizuj(float hodnota, float hodnotaMIN, float hodnotaMAX) { //normalizovanie hodnot od -1 do 1
    NORM = (((hodnota - hodnotaMIN) / (hodnotaMAX - hodnotaMIN)) * (1 - (-1))) + (-1);
    return NORM;
}
```

Obrázok 8 Normalizácia

3.3.2 FloatToInt

Táto funkcia slúži na vypísanie formátu desatinného čísla v správnom tvare. Vstupom je float číslo a výstupom je int. Vracia sa hodnota desatinného čísla pred desatinnou čiarkou.

3.3.3 FloatToInt2

Táto funkcia slúži na vypísanie formátu desatinného čísla v správnom tvare. Vstupom je float číslo a výstupom je int. Vracia sa hodnota desatinného čísla za desatinnou čiarkou.

3.3.4 Otvor_info

Táto funkcia slúži na spustenie informácií v menu. Keď užívateľ stlačí tlačidlo Info zobrazia sa hodnoty z ovládačov. Tieto jednotlivé hodnoty reprezentujú klapku, výškovku, plyn a smerovku. Nemôžeme zabudnúť, že jednotlivé hodnoty sú normalizované.

3.3.5 Otvor Revers

Hlavná časť tejto funkcie je nastavenie reverzného ovládania výškoviek, smerovky, plynu alebo klapiek. Jednotlivé kanály sa dajú zapnúť ON/OFF, pomocou bool premennej. Samotný výpočet zohľadňujúci reverz je vo funkcii normalizuj.

3.3.6 Otvor Expo

Funkcia expo slúži na exponenciálne ovládanie kanálov. Jednotlivé kanály sa dajú zapnúť ON/OFF, pomocou bool premennej. Samotný výpočet zohľadňujúci expo je vo funkcii normalizuj.

3.3.7 Otvor mix

Táto funkcia primárne slúži na prepínanie medzi klasickým ovládaním a mix ovládaním. Dá sa nastaviť váhy 50 na 50, 25 na 75, 75 na 25, alebo môže mix ostať vypnutý. Tu sú samostatné hodnoty, ktoré sa ukladajú aby ich potom v Info mohla funkcia vypísať.

3.3.8 TRIM

Trimovanie je zabezpečené pomocou funkcie otvorEPA. Po otvorení sa dostávame do ďalšieho submenu, kde zvolíme kanál a po zvolení kanálu vieme zvlášť navoliť hodnoty pre jednotlivé kanály. Hodnota trimu sa zohľadňuje vo funkcii normalizuj.

<https://github.com/tomaskolek/test11/tree/master/src>