SLOVENSKÁ TECHNICKÁ UNIVERZITA

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

VNORENÉ SYSTÉMY

Programátorská dokumentácia

Elektronická hracia kocka

Bc. Jerguš Frajt

Bc. Jozef Goga

Bc. Jaroslav Grejták

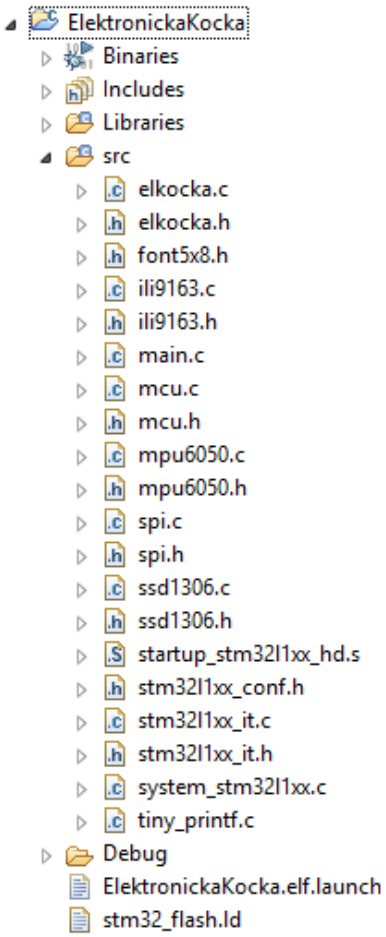
Bc. Zuzana Žemličková

ZS 2016/2017

Obsah

1. Súborová štruktúra

Štruktúra projektu vo vývojovom prostredí *Atollic True Studio* je znázornená na obrázku (Obr. 1.1).



Obr. 1.1: *Súborová štruktúra projektu*

Súbory:

- *ili9163.c* - obsahuje funkcie na prácu s SPI Nokia displejom 128x128 bodov
- *ili9163.h* - hlavičkový súbor funkcií pre komunikáciu s displejom
- *mcu.c* - obsahuje pomocné funkcie pre prácu s perifériami
- *mcu.h* - hlavičkový súbor pre prácu s perifériami
- *spi.c* - obsahuje funkcie na prácu s SPI zbernicou
- *spi.h* - hlavičkový súbor pre prácu s SPI zbernicou

- *ssd1306.c* - obsahuje pomocné funkcie pre zobrazenie znakov na displeji
- *ssd1306.h* - hlavičkový súbor na zobrazenie znakov na displeji

nie sú súčasťou tejto dokumentácie, pretože sme ich neupravovali. Tieto súbory sme dostali ako podklady pre projekt. Súbory generované vývojovým prostredím, ktoré sú súčasťou štandardnej periférnej knižnice a štandardné knižnice v jazyku C použité v tomto projekte nespadajú do tejto dokumentácie. Nami vytvorené súbory sú:

- *elkocka.c* - obsahuje pomocné funkcie pre projekt
- *elkocka.h* - hlavičkový súbor pomocných funkcií
- *main.c* - základný programový súbor projektu
- *mpu6050.c* - obsahuje funkcie na prácu s akcelerometrom MPU6050
- *mpu6050.h* - hlavičkový súbor pre prácu s akcelerometrom MPU6050

2. Detailný popis funkcií

2.1 main.c

```
/*
Hlavná slučka programu. Inicializuje všetky periférie použité v
    projekte: SPI zbernicu, I2C zbernicu, UART zbernicu, AD
    prevodník a GPIO. Zobrazí na displeji základnú obrazovku,
    inicializuje akcelerometer a v cykle načítava aktuálne hodnoty a
     zobrazuje ich na displeji.
 ------------------------
 VSTUPY:
 void
 ------------------------
 NAVRATOVÁ HODNOTA:
 int - číslo chyby
 ------------------------
*/
int main(void);
```

```
/*
Prerušenie od MPU6050 pre DMA prístup. Načítava aktuálne hodnoty z
    akcelerometra do pamäte pomocou DMA.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void DMA1_Channel7_IRQHandler(void);
```

2.2 mpu6050.c

```
/*
Funkcia na načítanie aktuálnych hodnôt z akcelerometra a uloženie
    do štruktúry MPU6050_t.
------------------------
VSTUPY:
int16_t* data - smerník na 16-bitové pole dát obsahujúce dáta z
    akcelerometra
```

```
MPU6050_t* Sensor - smerník na štruktúru akcelerometra, do ktorej
    sa uložia aktuálne dáta
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void MPU6050_readAcc(int16_t* data, MPU6050_t* Sensor);
```

```
/*
Funkcia na inicializáciu akcelerometra MPU6050. Otestuje, či
    komunikuje po I2C zbernici so správnym zariadením. Prebudí
    zariadenie zo sleep režimu a nakonfiguruje rozsah akcelerometra
    a gyroskopu podľa zadaných parametrov.
------------------------
VSTUPY:
MPU6050_t* Data - smerník na štruktúru akcelerometra, do ktorej sa
    uložia aktuálne nastavenia
MPU6050_Zariadenie_t DeviceNumber - číslo zariadenia v závislosti
    od logickej hodnoty AD0 pinu
MPU6050_Akcelerometer_t citlivostA - citlivosť akcelerometra
MPU6050_Gyroskop_t citlivostG - citlivosť gyroskopu
------------------------
NAVRATOVÁ HODNOTA:
char - číslo chyby, 0 ak je všetko v poriadku
------------------------
*/
char initMPU6050(MPU6050_t* Data, MPU6050_Zariadenie_t DeviceNumber
    , MPU6050_Akcelerometer_t citlivostA, MPU6050_Gyroskop_t
    citlivostG);
```

2.3 elkocka.c

```
/*
Funkcia na inicializáciu DMA prenosu pre I2C zbernicu.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void I2C1_initDMA(void);
```

```
/*
```

```
Funkcia na získanie adresy akcelerometra.
------------------------
VSTUPY:
uint8_t - adresa zariadenia MPU6050 pripojeného na zbernicu I2C
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
uint8_t I2C1_getDeviceAddress(void);
```

```
/*
Funkcia na vymazanie adresy komunikujúceho akcelerometra.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void I2C1_clearDeviceAddress(void);
```

```
/*
Funkcia na získanie hodnoty zásobníka podľa žiadaného indexu.
------------------------
VSTUPY:
int - žiadaný index zásobníka, ktorého hodnotu chceme načítať
------------------------
NAVRATOVÁ HODNOTA:
int32_t - aktuálna hodnota prijatého zásobníka podľa žiadaného
    indexu
------------------------
*/
int32_t I2C1_getRxBuffer(int index);
```

```
/*
Funkcia na získanie adresy čítaného registra.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
int8_t - adresa čítaného registra MPU6050 akcelerometra
------------------------
*/
```

```
uint8_t I2C1_getReadRegister(void);
```

```
/*
Funkcia na vymazanie adresy čítaného registra akcelerometra.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void I2C1_clearReadRegister(void);
```

```
/*
Funkcia na vygenerovanie skutočného náhodného čísla. Spustí AD
   prevodník pri najvyššej rýchlosti a načítava vnútornú teplotu
   procesora v cykle. Pomocou vygenerovania CRC kódu a získanej
   hodnoty z AD prevodníka vygeneruje 32-bitové náhodné číslo,
   ktoré normalizuje do rozsahu 1-6.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
uint8_t - skutočné háhodné číslo z rozsahu 1-6
------------------------
*/
uint8_t getTrueRandomNumber(void);
```

```
/*
Inicializácia GPIO periférie pre tlačidlo, ktoré spúšťa generovanie
    náhodného čísla.
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void initButton(void);
```

```
/*
Inicializácia ADC prevodníka pre načítavanie vnútornej teploty
   procesora pri najvyššej rýchlosti.
```

```
-----------------------
VSTUPY:
void
-----------------------
NAVRATOVÁ HODNOTA:
void
-----------------------
*/
void initADC(void);
```

```
/*
Inicializácia I2C zbernice .
-----------------------
VSTUPY:
void
-----------------------
NAVRATOVÁ HODNOTA:
void
-----------------------
*/
void initI2C1(void);
```

```
/*
Odošle zadaný počet bytov pomocou I2C zbernice do akcelerometra.
-----------------------
VSTUPY:
uint8_t slaveAddr - adresa komunikujúceho zariadenia
uint8_t pBuffer[] - dáta, ktoré chceme odoslať
uint8_t length - počet bytov, ktoré chcem odoslať
uint8_t writeAddr - adresa zápisového registra
-----------------------
NAVRATOVÁ HODNOTA:
void
-----------------------
*/
void I2C1_BytesWrite(uint8_t slaveAddr, uint8_t pBuffer[], uint8_t
   length,uint8_t writeAddr);
```

```
/*
Načítanie údajov pomocou I2C zbernice cez DMA prenos.
-----------------------
VSTUPY:
uint8_t slaveAddr - adresa komunikujúceho zariadenia
uint8_t readAddr - adresa čítaného registra
uint8_t numberBytesReceive - počet bytov, ktoré chceme prijať
-----------------------
```

```
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void I2C1_DMA_Read(uint8_t slaveAddr, uint8_t readAddr,uint8_t
    numberBytesReceive);
```

```
/*
Inicializácia USART zbernice .
------------------------
VSTUPY:
void
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void initUSART2(void);
```

```
/*
Vráti stranu kocky na základe aktuálnych hodnôt akcelerometra.
------------------------
VSTUPY:
MPU6050_t* Sensor - smerník na štruktúru akcelerometra, ktorá
    obsahuje aktuálne hodnoty
------------------------
NAVRATOVÁ HODNOTA:
char - strana kocky na základe hodnôt z akcelerometra z rozsahu 1-6
------------------------
*/
char diceSide(MPU6050_t* Sensor);
```

```
/*
Odošle string po sériovej zbernici UART.
------------------------
VSTUPY:
char *s - smerník na pole znakov, ktoré chceme odoslať
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void sendUSART2(char *s);
```

```
/*
Funkcia na zmazanie displeja zvolenou farbou.
```

```
------------------------
VSTUPY:
uint16_t colour - farba displeja, ktorou chceme zmazať displej
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void clearDisplay(uint16_t colour);
```

```
/*
Funkcia na zvykreslenie vyplneného kruhu zvolených parametrov na
    displeji.
------------------------
VSTUPY:
int16_t xCentre - x-ová pozícia stredu kruhu
int16_t yCentre - y-ová pozícia stredu kruhu
int16_t radius - polomer kruhu v pixeloch
uint16_t colour - farba kruhu
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void lcdFilledCircle(int16_t xCentre, int16_t yCentre, int16_t
    radius, uint16_t colour);
```

```
/*
Funkcia na zvykreslenie vyplneného kruhu zvolených parametrov na
    displeji.
------------------------
VSTUPY:
int16_t x0 - x-ová pozícia mriežky
int16_t y0 - y-ová pozícia mriežky
uint16_t cislo - číslo z rozsahu 1-6 na zobrazenie kocky
int16_t radius - polomer kruhu v pixeloch
uint16_t colour1 - farba guličiek
uint16_t colour2 - farba pozadia
------------------------
NAVRATOVÁ HODNOTA:
void
------------------------
*/
void lcdMriezka3x3(int16_t x0, int16_t y0, uint16_t cislo, uint16_t
    colour1, uint16_t colour2);
```