FW balík SlotCar v2.0 (MPC-SPS) - programátorský manuál

V 1.1, 1.10.2020

1. Obecné:

FW balík byl vytvořen pro snadnější programování studentského autonomního autíčka v laboratorním cvičení předmětu MPC-SPS a obsahuje několik modulů pro ovládání jednotlivých periferií. Snahou bylo umístit každý modul do separátního souboru (kombinace zdrojového *.c souboru a hlavičkového *.h souboru, vždy se stejným jménem). Soubory pro periferii, které mají být použity, je nutno přilinkovat k projektu (nahráním do adresáře s projektem) – je doporučeno vytvořit separátní adresáře „`include`“ (pro *.h soubory) a „`source`“ (pro *.c soubory). Daná dvojice se potom v hlavním souboru projektu (obvykle `main.c`) připojí syntaxí

```
#include "include/SPI.h"
```

Aktuálně vytvořené soubory (drivery):

- I2C.c + I2C.h - soubory pro práci s I2C sběrnicí
- SPI.c + SPI.h - soubory pro práci s SPI sběrnicí
- ADC.c + ADC.h - soubory pro práci s AD převodníkem
- LED.c + LED.h - soubory pro ovládání LED diod
- motor.c + motor.h - soubory pro ovládání H můstku – 2BC

2. Popis jednotlivých modulů

Na začátku hlavní funkce projektu `int main(void){}` je nutno:

- Zastavit watchdog (pokud se nebude využívat)
  `WDTCTL = WDTPW | WDTHOLD;  // stop watchdog timer`
- Provést inicializaci hodinového systému procesoru
  `initClockTo16MHz();`
- Povolit (pokud se využije) globální přerušení
  `_BIS_SR(GIE);`
- Inicializovat použité moduly, např.:
  `LED_init();`

I2C	
`void I2C_init(uint8_t slave_addr)`	Inicializuje I2C periferii (na portu UCSI B0 – P3.2 a P3.1) na rychlosti cca 400 kHz
Parametry	`slave_adr` – adresa slave zařízení, s nímž se bude komunikovat.
Návratové hodnoty	Žádné
`void I2C_read_byte(uint8_t r_adr, uint8_t count)`	Přečte ze slave zařízení určený počet bytů z určené adresy.
Parametry	`r_adr` – počáteční adresa ve slave zařízení, od níž se má vyčítat

	count – počet byte, které se mají vyčítat. Pozn. Je povoleno vyčítat max. počet určený makrem I2C_RX_BUFFER_SIZE definovaným v main.c
Návratové hodnoty	Žádné, ale funkce zapisuje vyčtené I2C hodnoty do globálního pole RX_buffer[], které je definováno v main.c.

Pozn.	Pro správou funkci je třeba mít povolené globální přerušení a definovány následující proměnné:
	uint8_t RX_buffer[I2C_RX_BUFFER_SIZE] = {0};
	uint8_t RXByteCtr = 0;
	uint8_t ReceiveIndex = 0;

void I2C_write_byte(uint8_t r_adr, uint8_t data)	Zapíše do slave zařízení na danou adresu jeden byte.
Parametry	r_adr – počáteční adresa ve slave zařízení, na kterou se má zapisovat data – data, která se mají zapsat na adresu r_adr
Návratové hodnoty	Žádné
Poznámky:	

1. Každé I2C zařízení má svou unikátní adresu.
2. Některá I2C zařízení mohou mít i pin(y) pro dodefinování adresy.
3. Soubory s adresami jednotlivých registrů pro některé sensory jsou rovněž součástí projektu (např. ADXL343.h)

Příklad použití funkcí:

```
I2C_init(ADXL343_I2C_RD);     // initialize I2C interface for slave with
                                 ADXL343_I2C_RD address
I2C_write_byte(0x01, 0x02);  // data 0x02 will be written at address 0x01
I2C_read_byte(0x00, 5);      // 5 bytes will be read from address 0x00
```

SPI	
void SPI_init(void)	Inicializuje SPI rozhraní (na portu UCSI B0 – P3.3 a P3.0) na rychlosti 4 MHz
Parametry	žádné
Návratové hodnoty	žádné
uint8_t SPI_read_byte (uint8_t addr)	Přečte 1 byte na dané adrese v zařízení připojeném na SPI sběrnici.
Parametry	addr – adresa registru, jehož hodnota se má přečíst
Návratové hodnoty	Přečtená data ze zařízení z dané adresy
void SPI_write_byte(uint8_t addr, uint8_t data)	Zapíše 1 byte dat do SPI zařízení na registr z danou adresou.
Parametry	addr – registr, na jehož adresu se má zapsat data – data, která se mají zapsat
Návratové hodnoty	žádné
Poznámky	

1. Soubory s adresami jednotlivých registrů pro některé sensory jsou rovněž součástí projektu (např. ADXL343.h).

Příklad použití funkcí:

```
SPIData = SPI_read_byte(WHO_AM_I | L3GD20H_READ); // reads the byte from
WHO_AM_I address (defined in L3GD20H.h file)
SPI_write_byte(CTRL1 | L3GD20H_WRITE, 0xFF);  // writes 0xFF to the register
with address CTRL1 (defined in L3GD20H.h file)
```

ADC	
`void ADC_init(void)`	Inicializuje AD převodník s následujícími parametry: - Vzorkovací frekvence ~20 kHz - Nastavení AVCC jako reference - SW start - Povolení kanálů A3 – A7 - Mód single sequence, vyvolání přerušení po převodu posledního kanálu.
Parametry	žádné
Návratové hodnoty	Žádné, ale zapisuje do globální uint16 proměnné `results[5]`, kam jsou postupně uloženy následující hodnoty: `results[0]` – proud motorem `results[1]` – vstupní napětí z kolejí – viz schema `results[2]` – vstup ADC7 `results[3]` – vstup ADC6 `results[4]` – vstup ADC5
Poznámky	

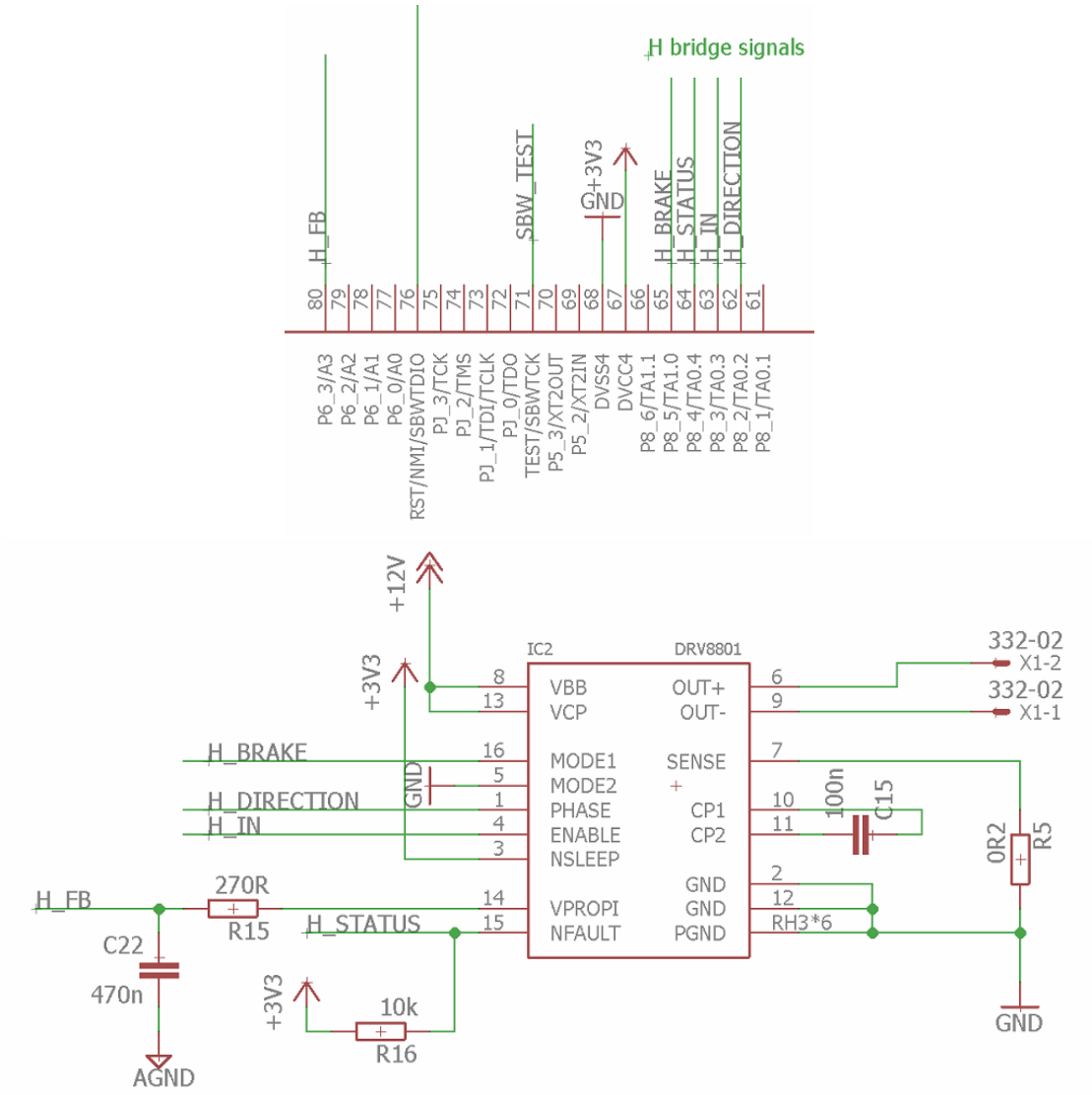
1. ADC převodník se spouští jednorázově příkazem
`ADC12CTL0 |= ADC12SC;  // Start conv - software trigger`
Po dokončení převodu (v obsluze přerušení) se převodník znovu spustí.
2. Pro správnou funkci je potřeba mít povoleno globální přerušení.	

LED	
`void LED_init(void)`	Inicializuje GPIO porty s připojenými LED diodami.
Parametry	žádné
Návratové hodnoty	Žádné
Poznámky	
LED diody se ovládají pomocí maker definovaných v hlavičkovém souboru LED.h. Makra jsou pro všechny LED obdobná, dle následujícího klíče: **`LED_xy_Z();`** x … F (pro přední LED), R (pro zadní LED) y … L (pro levou LED), R (pro pravou LED) Z … ON (pro rozsvícení LED), OFF (pro zhasnutí LED). Např. příkaz **`LED_FL_ON();`** rozsvítí levou přední LED.	

Ovládání motoru pomocí H můstku:

Zdrojový i hlavičkový soubor je prázdný – nutno doprogramovat vlastní funkčnost:

1) Nastavit typ daných pinů, které ovládají H můstek (viz schéma zapojení)



2) Nastavováním log. úrovní na jednotlivých pinech řídit směr a rychlost otáčení motoru – detaily viz datasheet DRV8801:

Table 1. Control Logic Table[1]

PINS							OPERATION
PHASE	ENABLE	MODE 1	MODE 2	nSLEEP	OUT+	OUT-	
1	1	X	X	1	H	L	Forward
0	1	X	X	1	L	H	Reverse
X	0	1	0	1	L	L	Brake (slow decay)
X	0	1	1	1	H	H	Brake (slow decay)
1	0	0	X	1	L	H	Fast-decay synchronous rectification[2]
0	0	0	X	1	H	L	Fast-decay synchronous rectification[2]
X	X	X	X	0	Z	Z	Sleep mode

(1) X = Don't care, Z = high impedance
(2) To prevent reversal of current during fast-decay synchronous rectification, outputs go to the high-impedance state as the current approaches 0 A.