ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA ELEKTROTECHNICKÁ

Katedra aplikované elektroniky a telekomunikací

DIPLOMOVÁ PRÁCE

Sada vzorových příkladů pro STM32

Bc. Jan Špika

Plzeň 2014

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2013/2014**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Jan ŠPIKA**
Osobní číslo:	**E12N0056P**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronika a aplikovaná informatika**
Název tématu:	**Sada vzorových příkladů pro STM32**
Zadávající katedra:	**Katedra aplikované elektroniky a telekomunikací**

Zásady pro vypracování:

Vytvořte sadu vzorových příkladů pro vývojový kit s procesorem řady STM32. Řešení musí obsahovat minimálně příklady využití:

1. SD karty
2. USB rozhrani
3. Rozhraní Ethernet

Vytvořené příklady důkladně zdokumentujte.

Rozsah grafických prací: **podle doporučení vedoucího**

Rozsah pracovní zprávy: **30 - 40 stran**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

Student si vhodnou literaturu vyhledá v dostupných pramenech podle doporučení vedoucího práce.

Vedoucí diplomové práce: **Ing. Kamil Kosturik, Ph.D.**
Katedra aplikované elektroniky a telekomunikací

Datum zadání diplomové práce: **14. října 2013**

Termín odevzdání diplomové práce: **12. května 2014**

Doc. Ing. Jiří Hammerbauer, Ph.D.
děkan

L.S.

Doc. Dr. Ing. Vjačeslav Georgiev
vedoucí katedry

V Plzni dne 14. října 2013

Anotace

Špika Jan - Sada vzorových příkladů pro STM32. Katedra aplikované elektroniky a telekomunikací, Západočeská univerzita v Plzni – Fakulta elektrotechnická, 2014, 50 s., vedoucí: Ing. Kamil Kosturik, Ph.D.

Diplomová práce obsahuje vzorové příklady na využití SD karty, USB, ethernetu a I2C na platformě stm32, konkrétně STM32F107VC. V příkladu na USB je ukázán běh mikrokontroléru v USB host módu a přenos dat mezi SD kartou a flash diskem připojeným přes USB. Aplikace s ethernetem obsahuje komunikaci s mikrokontrolérem prostřednictvím ethernetu a webové aplikace. Program s využitím I2C implementuje komunikaci se senzorem BMP085.

Klíčová slova

SD karta, USB, Ethernet, I2C, STM32

Abstract - Set of source codes for STM32

Špika Jan. Set of source codes for STM32. Department of applied electronics and telecommunications, University of West Bohemia in Pilsen – Faculty of electrical engineering, 2014, 50 p., head: Ing. Kamil Kosturik, Ph.D.

The master thesis contains a set of source codes for MCBSTM32C eval board and stm3210c-eval board using microcontroller STM32F107VC. The source codes are done for SD card, USB, Ethernet and I2C. The program for the SD card shows how to work with this peripheral. The USB example demonstrates using USB in the host mode and transporting data from the SD card to the flash disc connected to the board via USB. The ethernet application is based on lwIP stack and shows how to control the board´s peripherals over the web application and how to retrieve information from that. The last I2C application clarifies manipulation with BMP085 sensor via I2C.

Key words

SD card, USB, Ethernet, I2C, STM32

Prohlášení

Předkládám tímto k posouzení a obhajobě diplomovou práci, zpracovanou na závěr studia na Fakultě elektrotechnické Západočeské univerzity v Plzni.

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této diplomové práce, je legální.

V Plzni dne 5.5.2014 Jméno příjmení

……

Poděkování

Tímto bych chtěl poděkovat všem, kteří mě během studia na vysoké škole podporovali, a to hlavně rodině.

Dále bych chtěl poděkovat lidem, kteří mi svými radami, připomínkami a hlavně svou ochotou pomohli při tvorbě této diplomové práce. Těmi byli zejména Ing. Kamil Kosturik, Ph.D., Ing. Petr Krist, Ph.D., Ing. Ondřej Hála a Ing. Milan Brych.

Poděkování též patří Katedře aplikované elektroniky a telekomunikací za zapůjčení vývojových kitů.

Obsah

OBSAH **8**
SEZNAM SYMBOLŮ A ZKRATEK **10**
SEZNAM OBRÁZKŮ **11**
SEZNAM TABULEK **11**
ÚVOD **12**
1 HARDWARE **13**
1.1 PLATFORMA STM32 13
1.1.1 STM32F105/107 ..... *14*
1.2 POUŽITÉ VÝVOJOVÉ KITY 16
1.2.1 MCBSTM32C *16*
1.2.2 STM3210C-eval *18*
1.3 ULINK-ME 19
2 PŘÍKLAD NA VYUŽITÍ SD KARTY **20**
2.1 HARDWAROVÉ ŘEŠENÍ SD KONEKTORŮ 20
2.2 KOMUNIKACE SD KARTY S PROCESOREM 21
2.3 FATFS - GENERIC FILE SYSTEM 21
2.4 DISK I/O 21
2.5 APLIKACE PRÁCE S SD KARTOU 22
3 USB **23**
3.1 ZÁKLADNÍ VLASTNOSTI USB 23
3.1.1 USB rychlosti přenosu *23*
3.1.2 USB typy přenosů *24*
3.1.3 USB typy paketů *24*
3.1.4 USB deskriptory *25*
3.2 USB ON-THE-GO HOST AND DEVICE LIBRARY 28
3.3 USB OTG LOW LEVEL DRIVER 28
3.4 STM32_USB_HOST_LIBRARY 28
3.4.1 Stavový automat USB host knihovny *29*
3.4.2 Enumerace zařízení *30*
3.4.3 USB host - Mass storage class *31*
3.4.4 USB host - core *31*
3.5 APLIKACE S USB HOST MÓDEM 31

3.5.1 Úpravy MSC USB host příkladu *32*
3.5.2 Běh programu *32*
3.6 APLIKACE S USB DEVICE MÓDEM 33
4 ETHERNET 34
4.1 LWIP STACK 34
4.2 WEBOVÁ APLIKACE A SERVER 35
4.2.1 AJAX cross domain *36*
4.2.2 Sledování chodu komunikace *36*
4.3 PROGRAM MIKROKONTROLÉRU 37
5 APLIKACE I2C 39
5.1 I2C KOMUNIKACE 39
5.2 SENZOR BMP085 42
5.2.1 Parametry senzoru *42*
5.2.2 Zapojení senzoru *43*
5.3 VÝPOČETNÍ CYKLUS 44
5.4 UKÁZKOVÝ PROGRAM 45
5.4.1 I2C_WriteReg() *45*
5.4.2 I2C_ReadReg() *46*
6 IMPLEMENTACE PŘÍKLADŮ 47
ZÁVĚR 48
POUŽITÁ LITERATURA 49

Seznam symbolů a zkratek

ARP	Address Resolution Protocol
CAN	Controller Area Network
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
GPIO	General Purpose Input/Output
I²C	Inter-Integrated Circuit
I²S	Integrated Interchip Sound
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LwIP	lightweight IP
MII	Media Independent Interface
PPP	Point-to-Point Protocol
PWM	Pulse-width modulation
RMII	Reduced Media Independent Interface
SD	Secure Digital
SDHC	Secure Digital High Capacity
SNMP	Simple Network Management Protocol
SPI	Serial Peripheral Interface
SRAM	Static random-access memory
STM	STMicroelectronics
SWD	Serial wire debug
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
USART	Universal asynchronous receiver/transmitter
USB	Universal Serial Bus

Seznam obrázků

Obr. 1.1: Blokový diagram vnitřního uspořádání STM32f105/107, [2, s. 12] 15
Obr. 1.2: MCBSTM32C [4] 17
Obr. 1.3: Blokový diagram stm3210c-eval, [5, s. 7] 18
Obr. 1.4: ULINK-ME [6] 19
Obr. 2.1: SD konektor stm3210c-eval [5, s. 47] 20
Obr. 2.2: SD konektor MCBSTM32C [7] 20
Obr. 3.1: Stavový automat USB host knihovny [12, s.77] 29
Obr. 3.2: Enumerace zařízení [12, s.78] 30
Obr. 3.3: USB host aplikace na stm3210c-eval 33
Obr. 4.1: Ukázka webové aplikace 36
Obr. 4.2: Vzhled aplikace na MCBSTM32C 38
Obr. 5.1: I2C sběrnice 39
Obr. 5.2: Časové průběhy na I2C sběrnici [15] 41
Obr. 5.3: Zapojení BMP085 43
Obr. 5.4: Výpočetní cyklus u BMP085[16] 44
Obr. 5.5: I2C aplikace s BMP085 46

Seznam tabulek

Tab. 1.1: Rozdělení sérií stm32[1] 13
Tab. 1.2: Základní vlastnosti STMF105/107 [2] 14
Tab. 3.1: Deskriptor zařízení 25
Tab. 3.2: Konfigurační deskriptor 26
Tab. 3.3: Deskriptor rozhraní 27
Tab. 3.4: Deskriptor koncového bodu 27
Tab. 3.5: Řetězcový deskriptor 28
Tab. 5.1: Základní parametry BMP085 42
Tab. 5.2: Mezní parametry BMP085 43

Úvod

Cílem diplomové práce je vytvořit sadu vzorových příkladů pro platformu STM32, neboť vývojové kity na této platformě jsou používány při výuce, zejména v magisterském studiu na Fakultě elektrotechnické v Plzni. Dalším důvodem je rozšířenost této platformy mezi studenty a i možnost zapůjčení různých vývojových kitů s STM32 mikrokontroléry na katedře aplikované elektroniky a telekomunikací.

Ethernet, SD karta a USB byly vybrány z důvodu největšího potenciálního využití studenty při tvorbě semestrálních prací. Dále byl přidán příklad na využití I²C sběrnice, jelikož ji pro komunikaci využívá velké množství senzorů.

Během magisterského studia jsem se osobně přesvědčil, jak je aplikace i základních periférií, bez dřívějších zkušeností a použití vyššího programovacího jazyka, časově náročná. Proto si příklady nekladou za cíl podrobné vysvětlení dané periférie, ale měly by studentům poskytnout nejjednodušší, a tedy i časově nejefektivnější cestu k možné aplikaci daných periférií bez nutnosti jejich podrobného studia.

1 Hardware

Jak již bylo uvedeno v úvodu, diplomová práce se zaměřuje na platformu stm32. Konkrétně byly využity kity stm3210c-eval a MCBSTM32C s mikrokontrolérem STM32F107. Jako programátor byl použit ULINK-ME. Vlasti jednotlivých zařízení budou uvedeny v následujících podkapitolách.

1.1 Platforma STM32

Platforma STM32 32bitových mikrokontrolérů je založena na ARM Cortex™-M procesorech. Podrobné rozdělení sérií a jejich vlastností je možné vidět v následující tabulce.

Tab. 1.1: Rozdělení sérií stm32[1]

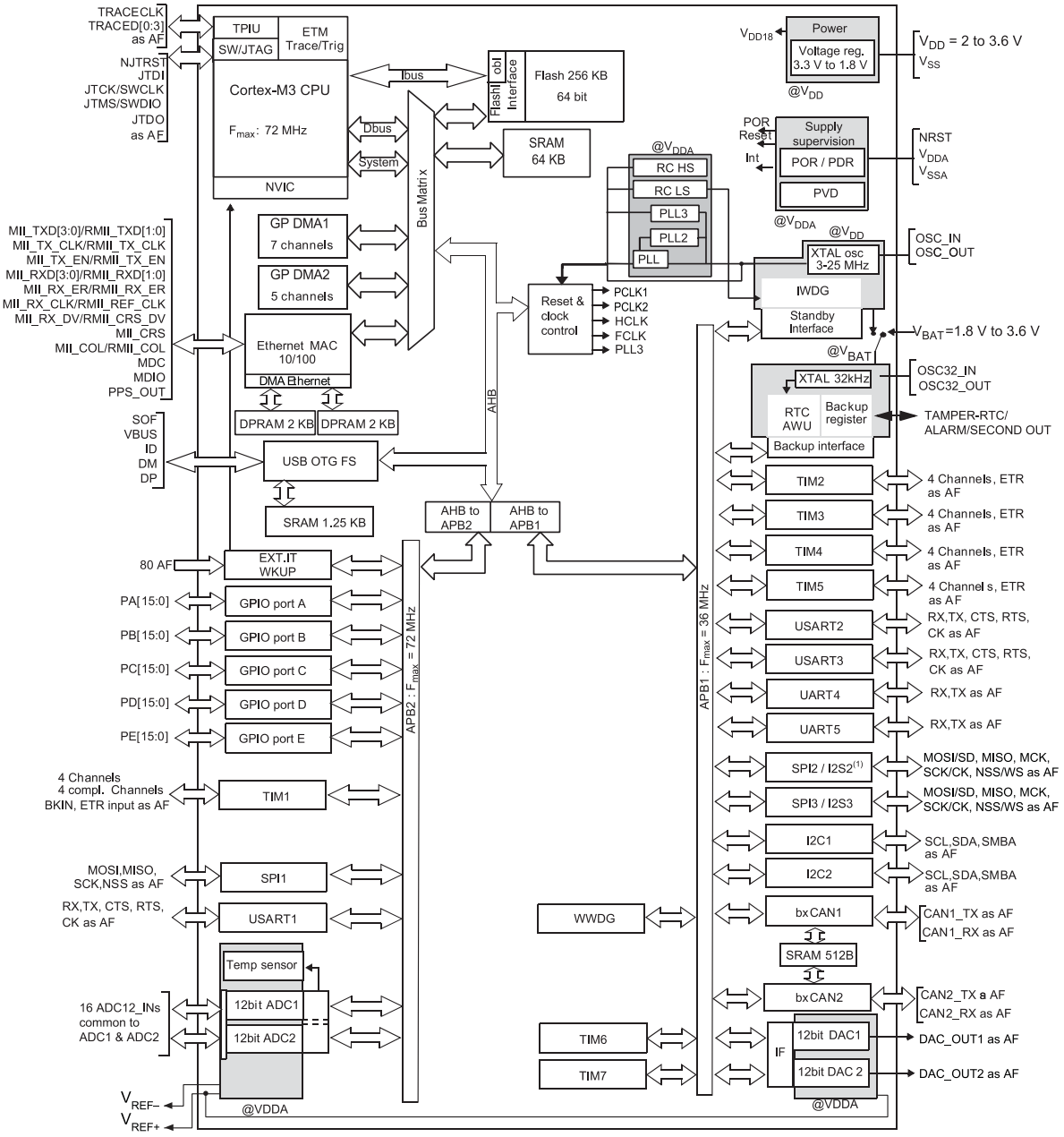
STM32 F4 series - High performance with DSP (STM32F401/405/415/407/417/427/437/429/439)								
Up to 180 MHz Cortex-M4 DSP/FPU	Up to 2-Mbyte Flash	Up to 256-Kbyte SRAM	2x USB 2.0 OTG FS/HS	1x 12-bit A/D timer	2x CAN 2.0B	SDIO 2x I²S audio Camera IF	Ethernet IEEE 1588	LCD-TFT SDRAM I/F
STM32 F3 series - Mixed-signal with DSP (STM32F301/302/303/373/x8)								
72 MHz Cortex-M4 with DSP and FPU	Up to 512-Kbyte Flash	Up to 64-Kbyte SRAM CCM-SRAM	USB 2.0 FS	3x 16-bit A/D timer (144 MHz)	CAN 2.0B	Up to 7x comparator 4x 12-bit DAC 4x PGA	HDMI CEC	3x 16-bit ∑Δ ADC
STM32 F2 series - High performance (STM32F205/215/207/217)								
120 MHz Cortex-M3 CPU	Up to 1-Mbyte Flash	Up to 128-Kbyte SRAM	2x USB 2.0 OTG FS/HS	1x 12-bit A/D timer	2x CAN 2.0B	SDIO 2x I²S audio Camera IF	Ethernet IEEE 1588	Crypto
STM32 F1 series - Mainstream - 5 product lines (STM32F100/101/102/103 and 105/107)								
Up to 72 MHz Cortex-M3 CPU	Up to 1-Mbyte Flash	Up to 96-Kbyte SRAM	USB 2.0 OTG FS	1x 12-bit A/D timer	Up to 2x CAN 2.0B	SDIO 2x I²S audio	Ethernet IEEE 1588	
STM32 F0 series - Entry-level (STM32F030/x1/x2/x8)								
48 MHz Cortex-M0 CPU	Up to 128-Kbyte Flash	Up to 16-Kbyte SRAM 20-byte backup data	USB clock free	USB 2.0 FS Crystal less	CAN 2.0B	DAC Comparator	CEC	
STM32 L1 series - Ultra-low-power (STM32L100/151/152/162)								
32 MHz Cortex-M3 CPU	Up to 512-Kbyte Flash	Up to 80-Kbyte SRAM	Up to 16-Kbyte EEPROM	USB 2.0 FS device	LCD 8x40 4x44	Op-amps comparator	BCR MSI VScal	AES 128-bit
STM32 L0 series - Ultra-low-power (STM32L0x1/x2/x3)								
32 MHz Cortex-M0+ CPU	Up to 64-Kbyte Flash	Up to 8-Kbyte SRAM	Up to 2-Kbyte EEPROM	USB 2.0 FS Crystal less	LCD 8x28 4x32	True RNG	BCR MSI VScal	AES 128-bit

1.1.1 STM32F105/107

V kapitole budou pouze výčtově vypsány nejdůležitější vlastnosti této série mikrokontrolérů. Detailní informace lze nalézt v [2], [3].

Tab. 1.2: Základní vlastnosti STMF105/107 [2]

Procesor	ARM® 32-bit Cortex®-M3 CPU
	72 MHz
Pamětí	64 až 256 Kbytes Flash paměť
	64 Kbytes SRAM
Hodiny	3 až 25 MHz krystalový oscilátor
	Vnitřní 8 MHz z výroby upravené RC
	Vnitřní 40 kHz RC s kalibrací
	32 kHz oscilátor pro RTC s kalibrací
Low power módy	Sleep, Stop a Standby
	VBAT zdroj pro RTC a backup registy
A/D	2 x 12b převodník (16 kanálů)
D/A	2 x 12b převodník
DMA	12 kanálů
Debug módy	SWD & JTAG
	Cortex®-M3 Embedded Trace Macrocell™
I/O	80
Časovače	10 s možností přemapování pinů
	4 16-bitové časovače
	2 x watch dog časovače
	SysTick časovač
	2 x 16-bit základní časovač pro řízení DAC
	1 x 16-bit motor control PWM časovač s dead-time generací a emergency stop
Komunikační rozhraní	14 s možností přemapování pinů
	2 x I2C
	5 x USART
	3 x SPI, 2 s I2S
	2 x CAN
	USB 2.0 OTG
	10/100 Ethernet



Obr. 1.1: Blokový diagram vnitřního uspořádání STM32f105/107, [2, s. 12]

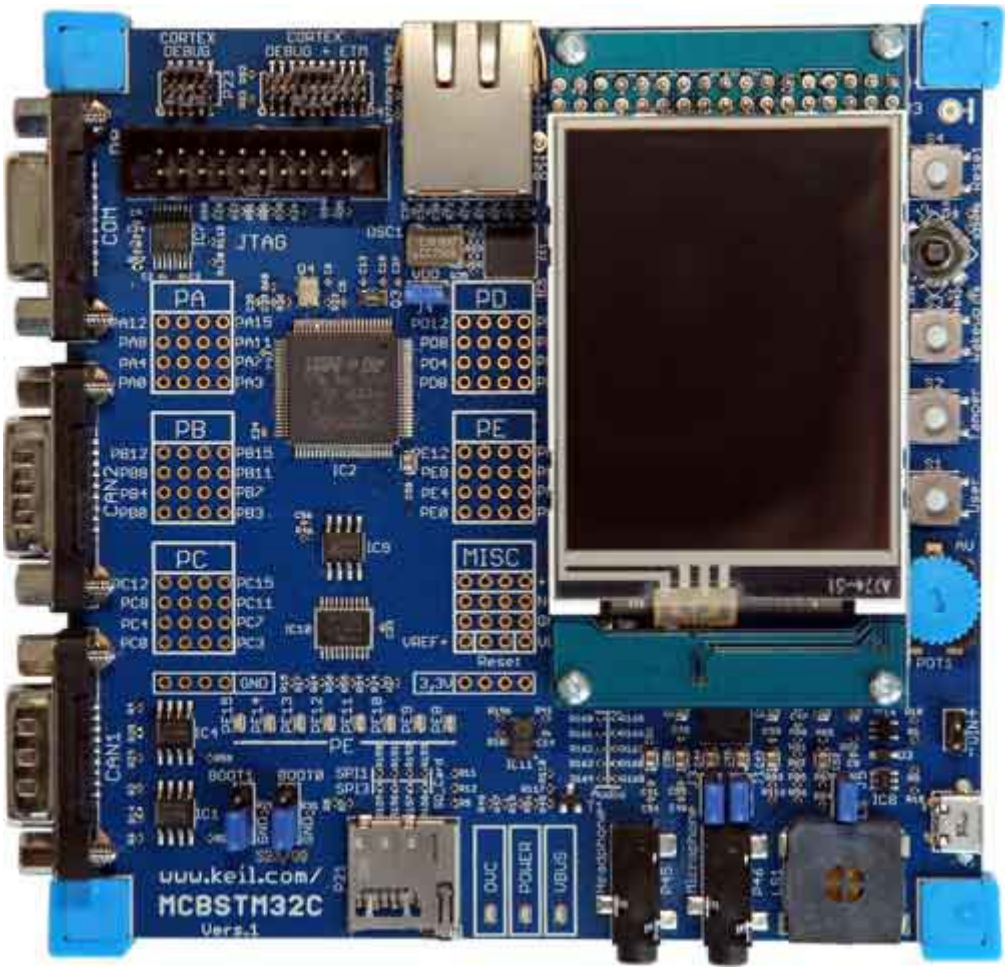
1.2 Použité vývojové kity

Pro zhotovení této práce byly použity dva vývojové kity. První je stm3210c-eval board od STM a druhý MCBSTM32C od ARMu. Hlavní důvod pro použití těchto desek byla jejich dostupnost na fakultě. MCBSTM32C je používán při výuce předmětů na fakultě elektrotechnické a stm3210c-eval si je možné zapůjčit při tvorbě semestrálních prací. Důležitou vlastností obou desek je, že pracují s mikrokontrolérem STM32F107VC.

1.2.1 MCBSTM32C

MCBSTM32C je vývojový kit od společnosti ARM založený na mikrokontroléru STM32F107VC. Jeho hlavními parametry jsou:

- 72MHz STM32F107VC ARM Cortex™-M3 procesor
- Paměť na čipu: 256KB Flash & 64KB RAM
- Externí paměť: 8KB I2C Flash
- Dotykový barevný QVGA TFT LCD displej
- 10/100 Ethernet Port
- USB 2.0 Full Speed - USB, USB-OTG, & USB Host
- 2 x CAN Interface
- Serial/UART Port
- MicroSD Card Interface
- 5-poziční Joystick a push-tlačítka
- 3-osý senzor pohybu/Akcelerometer
- Analog Voltage Control pro ADC vstup
- Audio CODEC s Line-In/Out a reproduktorem
- 80 GPIO pinů
- Debugovací Interface konektory:
    - 20-pin JTAG (0.1" Connector)
    - 10-pin Cortex debug (0.05" Connector)
    - 20-pin Cortex debug + ETM Trace (0.05" Connector)

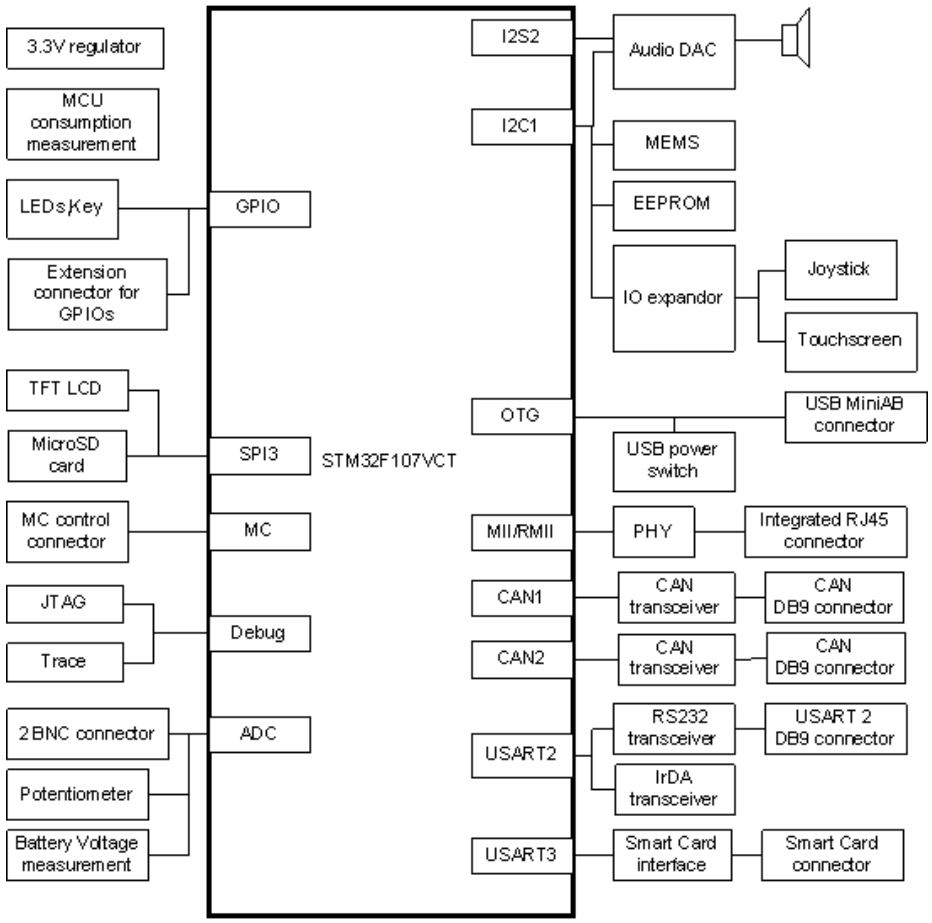


Obr. 1.2: MCBSTM32C [4]

Napájení je možné prostřednictvím mikro USB konektoru nebo VIN konektoru připojeného na napájení 5 až 5.5V DC. V případě využívání USB v režimu USB host nebo USB OTG je nutné kit napájet přes VIN konektor napětím 5.5V. [4]

1.2.2 STM3210C-eval

Stm3210c-eval je kit od společnosti STM. Veškeré vlastnosti lze vyčíst z blokového diagramu desky.



Obr. 1.3: Blokový diagram stm3210c-eval, [5, s. 7]

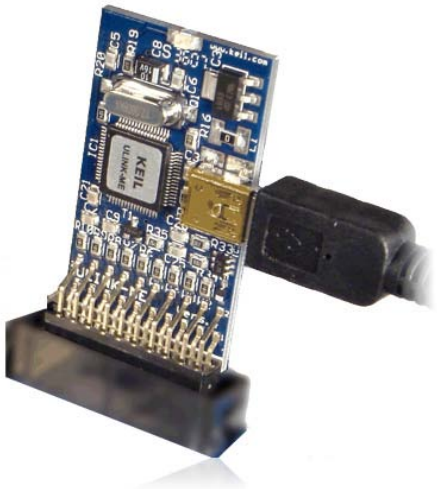
Možnosti napájení: Power jack, USB konektor nebo daughterboard

1.3 ULINK-ME

Ulink-me je debugovací adaptér od společnosti ARM.

Použití:

- tzv. On-chip Debugging
- Flash Memory Programovaní



Obr. 1.4: ULINK-ME [6]

Parametry:

- JTAG rychlost do 10MHz
- SWD pro zařízení s ARM Cortex-M
- SWV pro Cortex-M3 (M4)
- Podporované Keil µVision IDE & Debugger
  USB napájení
- Plug-and-play instalace
- 10-pin (0.05") - Cortex Debug Connector
- 20-pin (0.10") - ARM Standard JTAG Connector

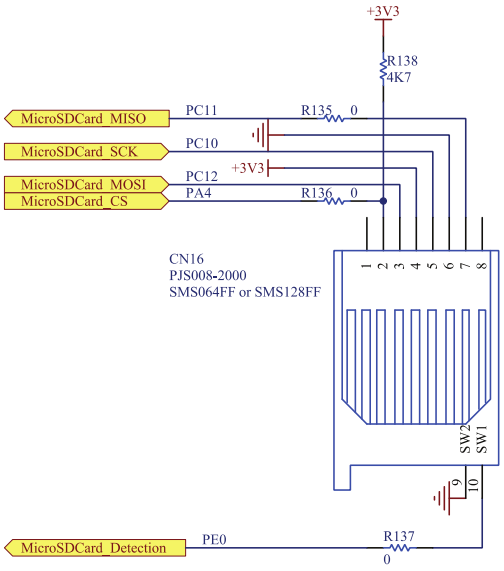
[6]

2 Příklad na využití SD karty

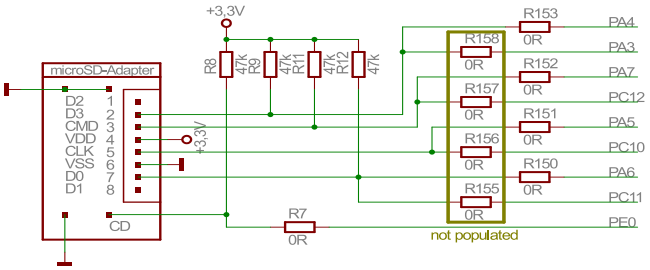
V této kapitole bude popsán způsob komunikace mikrokontroléru s SD kartou a veškeré nutné kroky k provedení této komunikace.

2.1 Hardwarové řešení SD konektorů

Jak lze vidět ze schémat zapojení, jsou řešení u jednotlivých desek rozdílná. U MCBSTM32C jsou pull-up rezistory zapojeny přímo u konektoru, a navíc si zde programátor musí dávat pozor na neosazené cesty u jednotlivých vývodů. U stm3210c-eval boardu je přes vnější pull-up rezistor připojen pouze MicroSDCard_CS pin a pro připojení k čipu je zde použita jen jedna cesta.



Obr. 2.1: SD konektor stm3210c-eval [5, s. 47]



Obr. 2.2: SD konektor MCBSTM32C [7]

2.2 Komunikace SD karty s procesorem

Komunikace s SD kartou probíhá přes rozhraní SPI. Podrobné informace k nastavení SPI lze nalézt v [3]. Pro oba kity lze použít knihovnu `stm32_eval_spi_sd.c`, kde jsou funkce pro práci s SD kartou a která dále využívá `stm3210c_eval.c`, v němž lze najít funkce pro nastavení SPI periférií. MCBSTM32C využívá pro komunikaci SPI1 a stm3210c-eval SPI3. Proto tyto knihovny lze použít pro obě desky, ale v případě MCBSTM32C je nutné provést změnu SPI a jeho pinů v `stm3210c_eval.h`.

2.3 FatFs - generic File system

FatTs je volně použitelný file systém pro malé embedded systémy, který je napsán v souladu s ANSI C a je kompletně oddělen od Disk I/O vrstvy. Díky tomu není závislý na hardwaru. Veškeré informace o tomto file systému lze najít v [8]. Mezi nejdůležitější patří aplikační informace, tedy informace o systémové organizaci, limitech systému atd., fórum a v neposlední řadě možnost stáhnout si různé verze systému a jejich ukázkové aplikace, což velmi ulehčuje implementaci.

Pro samotné programování jsou nejdůležitější limitující faktory systému a seznam funkcí, které může programátor použít, s kompletním popisem jejich vlastností. Nastavení file systému se provádí v souboru `ffconf.h` a ovlivňuje funkčnost jak jednotlivých funkcí, tak celého systému. Při používání funkcí je tedy vždy nutno zkontrolovat požadavky na nastavení tohoto souboru.

2.4 Disk I/O

Tato low-level vrstva není s file systémem dodávána. Na internetu lze stáhnout prázdnou kostru `diskio.c` a dále záleží na vývojáři, jak ji upraví pro potřeby konkrétní aplikace. Pro aplikace s STM32 lze využít zdrojové soubory pro low-level vrstvy z ukázkových příkladů poskytovaných přímo STM a dle potřeby je kombinovat.

2.5 Aplikace práce s SD kartou

Tato aplikace má za úkol ukázat nejjednodušším možným způsobem práci s SD kartou, to jest v konkrétním případě nastavení komunikace s SD kartou přes SPI, detekovaní SD karty, namountovaní, vytvoření složky, odmountování a zrušení SPI komunikace.

Na začátku programu je volána funkce `SD_Init()`, která nastaví SD_SPI, to jest povolí hodiny, v nutnosti přemapování přemapuje SPI piny, nastaví piny MISO, MOSI, SCK, CS a detect pin a provede samotné nastavení SPI komunikace. Podrobné informace o SPI a jeho nastavení lze nalézt v [3]. Je zde nutné zmínit, že veškeré funkce z `stm32_eval_spi_sd.c` mají návratové hodnoty signalizující stav provedení funkce. V našem případě je uživatel informován o bezchybném provedení na LCD displeji a v případě chyby je vypsán její kód, jehož znění lze nalézt v `stm32_eval_spi_sd.h`. Dalším krokem je zjištění detekce SD karty, což je provedeno funkcí `SD_Detect()`, která kontroluje stav na SD detect pinu.

Nyní je již možné využívat funkce file systému, které opět mají návratové hodnoty a v případě chyby lze najít daný kód v knihovně `ff.h`. Jako první je nutné zavolat funkci f_mount(), kterou přiřadíme pracovní oblast file systému. Dále je pro ukázku vytvořena složka a provedeno odmountování a deinicializace SD SPI komunikace. Všechny použité funkce pro práci s FatFs lze najít v `ff.c` a, jak již bylo zmíněno, low level funkce, které jsou těmito funkcemi využívány, v `diskio.c`.

3 USB

USB je využíváno v mnoha typech aplikací. Proto STM poskytuje širokou podporu v podobě STM32Fxx USB On-The-Go Host and Device knihovny, kde lze nalézt příklady, ať už pro USB-host, USB-device, tak USB-host-device. První příklad na USB pouze upravuje příklad na USB-device mass storage pro MCBSTM32C. Druhý příklad ukazuje přenos dat mezi SD kartou a flash diskem připojeným přes USB.

3.1 Základní vlastnosti USB

V následující kapitole budou popsány nejdůležitější vlastnosti USB.

3.1.1 USB rychlosti přenosu

USB dle typu podporuje několik rychlostí přenosu:

- USB 1.1
  - Low Speed 1,5Mbit/s
  - Full Speed 12Mbit/s
- USB 2.0
  - High Speed 480Mbit/s
- USB 3.0
  - Super Speed 5Gbit/s
- USB 3.1
  - Super Speed 10Gbit/s

[9]

Výše zmiňovanou USB On-The-Go Host and Device knihovnou jsou podporované Full Speed a High speed přenosy. Typ rychlosti přenosu je softwarovou záležitostí, nikoliv hardwarovou.

3.1.2 USB typy přenosů

USB má celkem čtyři typy přenosů, každý z nich je určen rozdílnými vlastnostmi komunikačního přenosu, to jest: formát dat, směr komunikačního toku, velikost paketu, přístup na sběrnici, zpoždění, sekvence požadavku o data, řešení chyb. USB definuje čtyři typy přenosu:

- Hromadný (bulk) - Přenos určený pro velký objem dat. Je zde zajištěno doručení dat, ale nikoliv doba doručení.
- Izochronní (isochronous) - V tomto typu přenosu je zajištěna plynulost přenosu dat o definované rychlosti. V případě selhání doručení se neprovádí opětovné vyslání.
- Přenos při přerušení (interrupt) - Určen pro zařízení, která příležitostně posílají nebo přijímají data. Tento přenos má zaručenou dobu obsluhy a v případě selhání je zde opětovný pokus o doručení.
- Řídící přenos (control) - Používaný pro řízení, konfiguraci a získávání statusů USB zařízení.

[10]

3.1.3 USB typy paketů

USB komunikace obsahuje čtyři druhy paketů :

- Token paket - definuje typ přenosu na USB
- Datový paket - obsahuje data přenosu
- Handshake paket - tzv. potvrzovací paket
- Preamble - určen k přepínaní mezi full a hihg speed

[10]

3.1.4 USB deskriptory

Vlastnosti USB zařízení jsou uloženy v tzv. deskriptorech. Ty mají přesně definovanou strukturu a jsou využívány při enumeraci.

- Deskriptor zařízení - udává vlastnosti celého zařízení. Z tohoto důvodu má každé zařízení pouze jeden tento deskriptor.

Tab. 3.1: Deskriptor zařízení

Offset	Název	Velikost	Hodnota	Popis
0	bLength	1	number	velikost deskriptoru
1	bDescriptorType	1	constant	typ deskriptoru, device (0x01)
2	bcdUSB	2	BCD	verze USB, se kterou je zařízení kompatibilní
4	bDeviceClass	1	class	kód třídy zařízení
5	bDeviceSubClass	1	subClass	kód podtřídy
6	bDeviceProtocol	1	protocol	kód protokolu
7	bMaxPacketSize	1	number	maximální velikost paketu pro endpoint 0
8	idVendor	2	ID	vendor ID
10	idProduct	2	ID	product ID
12	bcdDevice	2	BCD	číslo verze zařízení
14	iManufacturer	1	index	index string deskriptoru popisující výrobce
15	iProduct	1	index	index string deskriptoru popisující výrobek
16	iSerialNumber	1	index	index string deskriptoru popisující sériové číslo
17	bNumConfigurations	1	integer	počet možných konfigurací

- konfigurační deskriptor - udává popis konfiguračního nastavení USB zařízení, které může mít těchto nastavení několik.

Tab. 3.2: Konfigurační deskriptor

Offset	Název	Velikost	Hodnota	Popis
0	bLength	1	number	velikost deskriptoru
1	bDescriptorType	1	constant	typ deskriptoru, configuration (0x02)
2	wTotalLength	2	number	délka všech deskriptorů této konfigurace. Obsahuje délku konfiguračního deskriptoru, deskriptoru rozhraní a deskriptorů koncových bodů.
4	bNumInterfaces	1	number	počet rozhraní
5	bConfigurationValue	1	number	hodnota používaná jako argument k výběru této konfigurace
6	iConfiguration	1	index	index string deskriptoru popisující toto nastavení
7	bmAttributes	1	bitmap	D7 rezervováno, nastaveno do 1 D6 napájení ze sběrnice/vlastní napájení D5 vzdálené probuzení D4..0 rezervováno, nastaveno do 0
8	bMaxPower	1	mA	maximální odběr ze sběrnice. Jednotkou jsou 2 mA.

- Deskriptor rozhraní - popisuje nastavení daného rozhraní.

Tab. 3.3: Deskriptor rozhraní

Offset	Název	Velikost	Hodnota	Popis
0	bLength	1	number	velikost deskriptoru
1	bDescriptorType	1	constant	typ deskriptoru, interface (0x04)
2	bNumInterfaces	1	number	číslo rozhraní
3	bAlternateSetting	1	number	hodnota použitá k výběru alternativního nastavení
4	bNumEndpoints	1	number	počet koncových bodů (bez koncového bodu nula) tohoto rozhraní; je-li hodnota nulová, používá se pouze koncový bod nula
5	bInterfaceClass	1	class	kód třídy
6	bInterfaceSubClass	1	subClass	kód podtřídy
7	bInterfaceProtocol	1	protocol	kód protokolu
8	iInterface	1	index	index string deskriptoru popisující toto rozhraní

- Deskriptor koncového bodu - informace z tohoto deskriptoru jsou používány hostitelem pro určení šířky pásma.

Tab. 3.4: Deskriptor koncového bodu

Offset	Název	Velikost	Hodnota	Popis
0	bLength	1	number	velikost deskriptoru
1	bDescriptorType	1	constant	typ deskriptoru, endpoint (0x05)
2	bEndpointAddress	1	endpoint	adresa koncového bodu bity 0..3b číslo koncového bodu bity 4..6b rezervováno, nastaveno do 0 bity 7 směr přenosu 0 = Out, 1 = In
3	bmAttributes	1	bitmap	typ a vlastnosti přenosu
4	wMaxPacketSize	2	number	maximální velikost paketu
6	bInterval	1	number	interval pro polling

- Řetězcový deskriptor - není povinný. Jestliže zařízení nepodporuje řetězcové deskriptory, všechny odkazy na ně uvnitř deskriptorů zařízení, konfiguračních a rozhraní, musí být nastaveny na null.

Tab. 3.5: Řetězcový deskriptor

Offset	Název	Velikost	Hodnota	Popis
0	bLength	1	number	velikost deskriptoru
1	bDescriptorType	1	constant	typ deskriptoru, string (0x03)
2	bString	n	unicode	řetězec v UNICODE

[11]

3.2 USB On-The-Go Host and Device library

Tato knihovna je určena pro mikrokontroléry řad STM32F105xx, STM32F107xx, STM32F2xx a STM32F4xx. Jejím cílem je poskytnout ukázkové aplikace pro full speed a high speed přenosy všech typů. V následujícím textu budou popsány součásti používané v aplikaci s USB host módem.

3.3 USB OTG low level driver

Low level soubory knihovny se dělí do tří kategorií. Obecné, host a device. Usb_core.c /.h a usb_bsp.c /.h jsou obecné a zbylé dva soubory patří k host módu.

- usb_core.c /.h - soubor obsahující hardwarově abstraktní vrstvu a komunikační operace
- usb_bsp.c /.h - soubor obsahuje low level inicializaci

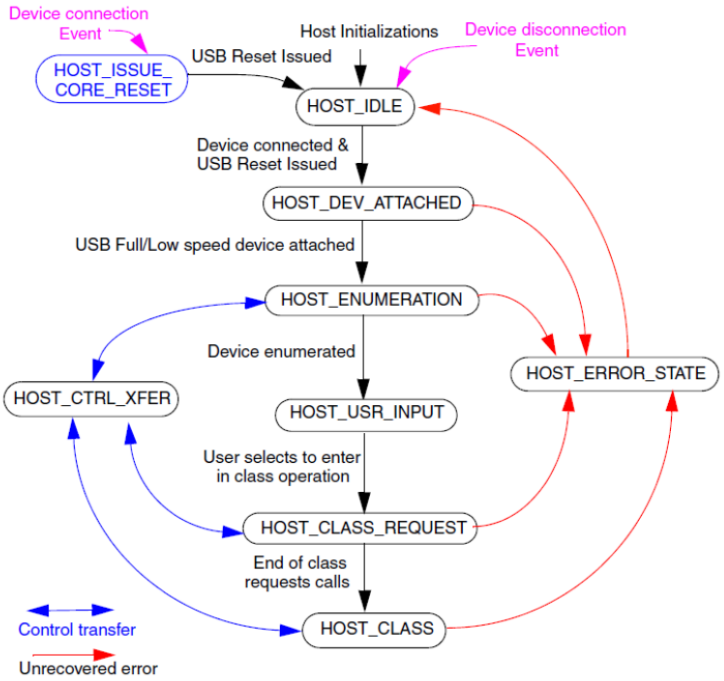
- usb_hcd.c /.h - soubor obsahující host interface vrstvu, používanou knihovnou pro přístup k jádru
- usb_hcd_int.c /.h - soubor obsahující interuptní procesy pro host mode

3.4 STM32_USB_HOST_Library

Tato knihovna je rozdělena na dvě podsložky. První, class, obsahuje soubory vztahující se k implementaci používané třídy, v našem případě MSC. Druhá, core, obsahuje USB Host library procesy definované v USB specifikaci rev. 2.0.

3.4.1 Stavový automat USB host knihovny

Běh USB host knihovny je řízen dle stavového automatu, který lze vidět na obr. 3.1.



Obr. 3.1: Stavový automat USB host knihovny [12, s.77]

Jádro může nabývat devíti stavů:

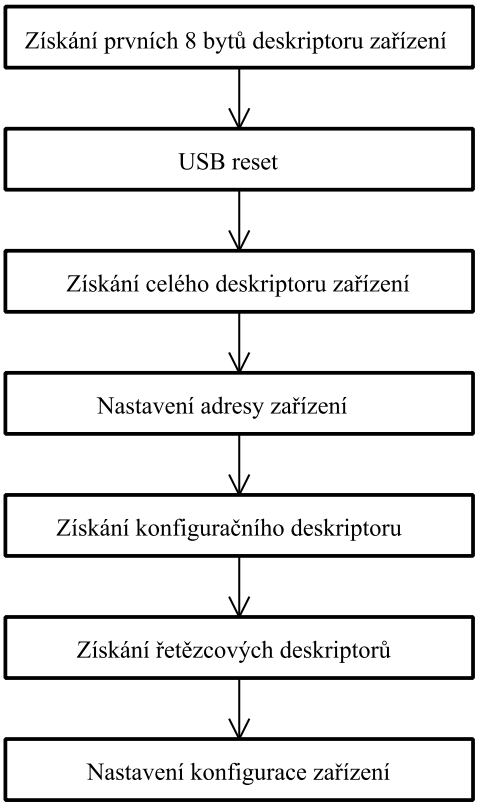
- HOST_IDLE - do tohoto stavu se jádro dostane po inicializaci host a následně čeká na připojení zařízení. Přejde do něj taktéž, jestliže je indikováno odpojení zařízení, a nebo když se objeví nepokrytá chyba.
- HOST_ISSUE_CORE_RESET - přechod do tohoto stavu nastane v okamžiku, kdy je připojeno zařízení za účelem vytvoření resetu USB sběrnice.
- HOST_DEV_ATTACHED - jádro přejde do tohoto stavu, když je připojeno zařízení. Jestliže je zařízení detekováno, stavový automat přejde do stavu enumerace.
- HOST_ENUMERATION - v tomto stavu probíhá enumerace, která bude podrobněji popsána dále.
- HOST_USR_INPUT - tento stav následuje po enumeraci a zahrnuje čekání na uživatelský vstup, aby se zahájil provoz třídy USB.
- HOST_CLASS_REQUEST - v tomto stavu přebírá řízení ovladač třídy a je zavolán

stavový automat požadavků třídy, aby vyřešil všechny původní požadavky třídy. Po dokončení všech požadavků se přechází do stavu HOST_CLASS.

- HOST_CLASS - v tomto stavu je zavolán stavový automat třídy pro operace dané třídy.
- HOST_CTRL_XFER - do tohoto stavu se přechází, kdykoliv je nutno provést kontrolní přenos.
- HOST_ERROR_STATE - do tohoto stavu se přechází, kdykoliv se vyskytne nepokrytá chyba ve stavovém automatu knihovny.

3.4.2 Enumerace zařízení

Po rozpoznání zařízení následuje enumerace. Postup je ukázán v následujícím obrázku.



Obr. 3.2: Enumerace zařízení [12, s.78]

Stavový automat enumerace je implementován ve funkci `USBH_HandleEnum()`. Ta je volána stavovým automatem jádra a využívá funkce z `usbh_stdreq.c`. Na konci enumerace je volán uživatelský callback, aby umožnil uživateli využít informace z deskriptorů.

3.4.3 USB host - Mass storage class

Tato podsložka je tvořena soubory sloužícími k implementaci MSC ovladače podporujícího BOT (Bulk-Only Transport) a SCSI příkazy. Tvoří ji následující soubory:

- usbh_msc_core.c /.h - implementace stavového automatu jádra MSC
- usbh_msc_bot.c /.h - implementace BOT
- usbh_msc_scsi.c /.h - SCSI implementace
- usbh_msc_fatfs.c /.h - funkce pro komunikace s FatFs

3.4.4 USB host - core

Všeobecný význam souborů v této složce byl popsán výše. Nyní bude uveden konkrétnější popis jednotlivých souborů.

- usbh_core.c /.h - soubory obsahující funkce pro řízení USB komunikace a stavového automatu
- usbh_stdreq.c /.h - soubory obsahující implementaci žádostí z kapitoly 9 USB specifikace
- usbh_hcs.c /.h - soubory obsluhující alokaci host kanálu a spouštěcí procesy
- usbh_ioreq.c /.h - soubory obsluhující generování USB transakcí
- usbh_conf.h - soubory obsahující konfiguraci interface čísla zařízení a velikosti paketu

[12]

3.5 Aplikace s USB host módem

Tato aplikace je postavena na MSC USB host příkladu z STM32F105/7XX and STM32Fxx USB On-The-Go Host and Device Library V2.0.0 / 22-July-2011. Cílem je ukázat další možné využití módu USB host, a to přenos dat z SD karty do flash disku připojeného prostřednictvím USB. Tento příklad je k dispozici pouze pro stm3210c-eval, jelikož kit MCBSTM32C by musel mít externí napájení.

Z činnosti programu zůstal výpis základních informací o flash disku po připojení k stm3210c-eval. Dále je uživatel tázán, jestli si přeje přehrát předem daný dokument, a jestliže akci potvrdí, proběhne přenos souboru.

3.5.1 Úpravy MSC USB host příkladu

Hlavní úprava programu spočívala v implementaci low level vrstvy pro komunikaci s SD kartou a přechodu z FAT file system module R0.07e na FAT file system module R0.10a.

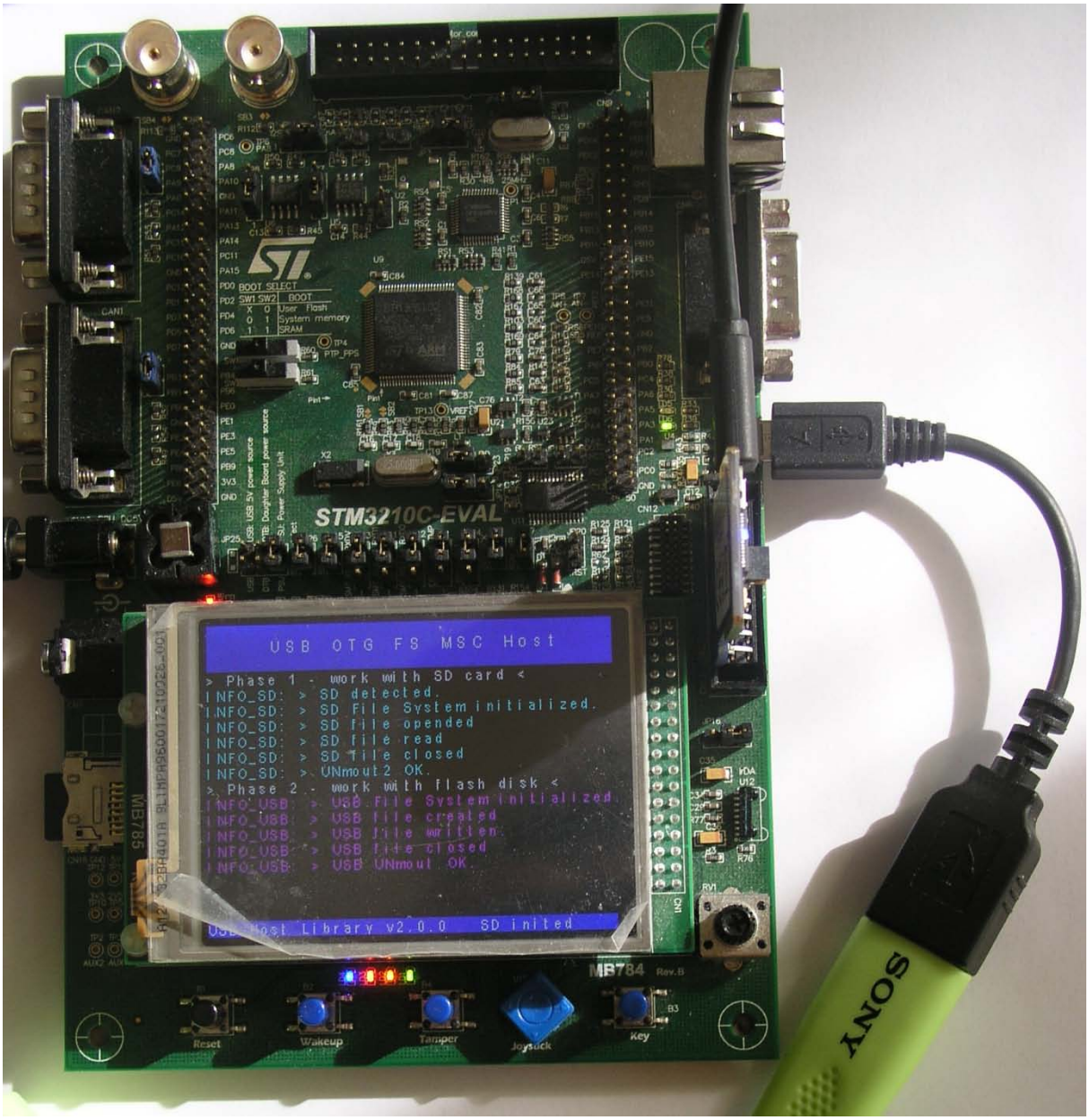
Low level vrstva byla implementována do `usbh_msc_fatfs.c`, kde se pomocí funkce `switch()` a vstupní proměnné drv řeší ve funkcích obsluha pro SD nebo USB. Pro změnu file systému byly aktualizovány soubory `ff.h`, `ff.c` a `ffconf.h`.

3.5.2 Běh programu

V `main.c` se nacházejí dvě základní funkce programu. `USBH_Init()` slouží k nastavení USB host knihovny a hardwaru. Druhou funkcí je periodicky volaná funkce `USBH_Process()`, ve které je implementován stavový automat jádra.

Jelikož využíváme MSC, je po provedení inicializace třídy MSC stavovým automatem volána funkce `USBH_USR_MSC_Application()`. V ní je zprostředkováno veškeré uživatelské chování funkce. Část programu řešící kopírování dat probíhá ve dvou fázích. V první se kontroluje detekce SD karty. Poté je vytvořen file systém, otevřen soubor na SD kartě v režimu pro čtení pomocí funkce `f_open()` a funkcí `f_read(&fileSD, buffer, sizeof buffer, &br)` je přečten, obsah je uložen do `buffer` a počet přečtených bytů je uložen do proměnné `br`. Následuje uzavření souboru a odmountování file systému.

V druhé fázi je vytvořen file systém pro USB a je vytvořen soubor pro zápis na flash disk. Funkcí `f_write(&file, buffer, br, &bw)` je zapsán obsah `buffer` do souboru, velikost zapisovaných dat je určená proměnnou `br`. Následuje opět uzavření souboru a odmountování file systému. Průběh funkcí je pro kontrolu logován na LCD displej. Na displej je pro kontrolu vypsán i stav inicializace SD karty, která probíhá ve funkci `USBH_USR_Init()`.



Obr. 3.3: USB host aplikace na stm3210c-eval

3.6 Aplikace s USB device módem

Pro tento mód byl opět vybrán ukázkový příklad z třídy MSC od STM pro stm3210c-eval a byl překonfigurován pro MCBSTM3210C. Příklad ukazuje nastavení desky, která se po připojení k PC tváří jako flash disk. Nastavení jiného názvu zařízení po připojení k PC je možné změnou deskriptorů v souboru `usb_desc.c`.

4 Ethernet

Tato kapitola má za cíl ukázat implementaci ethernetu na stm3210c-eval nebo MCBSTM32C. STM pro své aplikace používá tři ethernetové stacky. NicheLite, uIP stack nebo lwIP stack. Aplikace obsahuje komunikaci mikrokontroléru prostřednictvím lwIP stacku s webserverem. Program na STM32 je založen na STM32F107 LwIP demonstration package V1.0.0 - 11/20/2009 a webový server je naprogramován pomocí jquery a jquery-ui.

4.1 lwIP stack

LwIP (lightweight IP) je opensource stack a je následovníkem uIP stacku. Původně byl vyvinut Adamsem Duklesem a nyní je jeho vývoj umístěn na Savannahu, kde se na něm může podílet kdokoliv. Jeho hlavními přednostmi jsou implementační nezávislost a malá velikost, přes kterou poskytuje plný rozsah TCP, a proto je vhodný pro embedded systémy.

Vlastnosti a protokoly v lwIP stacku:

- IP (Internet Protocol)
- ICMP (Internet Control Message Protocol)
- IGMP (Internet Group Management Protocol)
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- DNS (Domain names resolver)
- SNMP (Simple Network Management Protocol)
- DHCP (Dynamic Host Configuration Protocol)
- AUTOIP
- PPP (Point-to-Point Protocol)
- ARP (Address Resolution Protocol)

[13]

4.2 Webová aplikace a server

Nejpoužívanější způsob, jak udělat webovou aplikaci pro komunikaci s vývojovými deskami, je udělat webové stránky, přeložit je do hexa kódu a uložit je buď do paměti mikrokontroléru nebo např. na SD kartu. Při komunikaci a obnově dat by se pak pouze měnily námi požadované položky. Pro tuto aplikaci byla webová aplikace uložena mimo kit.

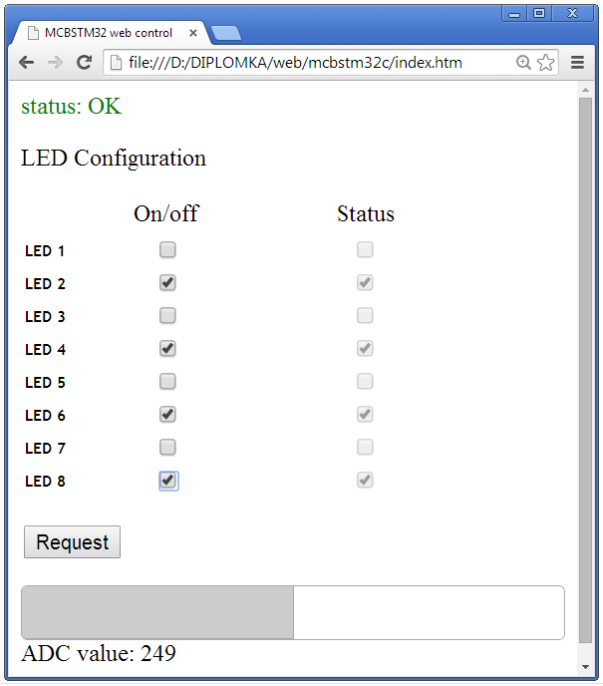
Webové stránky poskytují několik možností. V prvé řadě je na nich řídící sloupec check boxů, prostřednictvím kterých lze ovládat led diody. Vedle je tzv. informační sloupec check boxů, který ukazuje jejich stav. Jestli je check box zaškrtnut, je led dioda aktivní a naopak. Progressbar graficky znázorňuje hodnotu ADC, která je pod ním navíc číselně vypisována. Dále je sledován stav komunikace mezi serverem a deskou, který je zobrazován v levém horním rohu aplikace.

Webový server se skládá ze tří funkcí, které řídí chování celé webové aplikace.

Funkce `Main()` obsahuje řídící algoritmy pro volání obslužných funkcí. Jestliže dojde k zaškrtnutí kontrolních check boxů ve webové aplikaci, je volána funkce `SendCmd()`. Dále je zde provedeno nastavení progressbaru, to jest jeho maximální hodnoty a hodnoty po inicializaci, a čítače, který každou sekundu vyvolá funkci zjišťující stav led diod a ADC, tím je zajištěno automatické získávání dat pro aplikaci po této době.

Funkce `SendCmd()` je volána pokaždé, když je zaškrtnut jakýkoliv řídící check box na webové stránce. Na začátku funkce je vytvořena proměnná pro každý check box, ve které bude stav daného check boxu, a je defalutně nastavena na nulovou hodnotu. V další části programu se provede kontrola zaškrtnutí všech check boxů a v případě zaškrtnutí se odpovídající proměnná nastaví do jedničky. Získané informace se odešlou ve tvaru `"http://" + boardIP + "/method=get", { "led1": led1, "led2": led2, "led3": led3, "led4": led4, "led5": led5, "led6": led6, "led7": led7, "led8": led8 })`". Stav odeslaní je logován pomocí `consol.log()`.

Funkcí `SendRequest()` se odešle "`http://" + boardIP + "/STM32F107AD", "le`"". Jestliže běžící aplikace na mikrokontroléru rozpozná správný příkaz, pošle zpět data v JSON formátu. Přijatá data jsou logována a následuje vyčtení stavu led diod z obdržených dat a jeho zobrazení v informačních check boxech. Stav a hodnota ADC se ukáže v progressbaru. V této funkci je navíc kontrolován stav komunikace.



Obr. 4.1: Ukázka webové aplikace

4.2.1 AJAX cross domain

Jak již bylo zmíněno, webový server je umístěn mimo vývojový kit. To s sebou přináší problém v podobě AJAX cross domain. Pro vyřešení tohoto problému je nutné použít např. Google Chrome a spouštět jej s vypnutím webového zabezpečení. To jest kliknout pravým tlačítkem na spouštěcí ikonu Google Chromu, dát Properties a v kolonce Target přidat na konec *--disable-web-security*.

4.2.2 Sledování chodu komunikace

Pro sledování komunikace byl použit program Wireshark, který umožňuje rozsáhlé možnosti v sledování síťové komunikace. Jednoduchost použití tohoto programu odpadá v případě, kdy komunikace neprochází přes síťovou kartu, tedy například jestliže si uživatel naprogramuje chování serveru a zkouší činnost webové aplikace. Pro ladění aplikace byl zprvu použit Nástroj vývojáře ve FireFoxu. Po zjištění problému s ajax cross domain byl nakonec zvolen Google Chrome v režimu omezení zabezpečení a pro sledování komunikace použity nástroje pro vývojáře.

4.3 Program mikrokontroléru

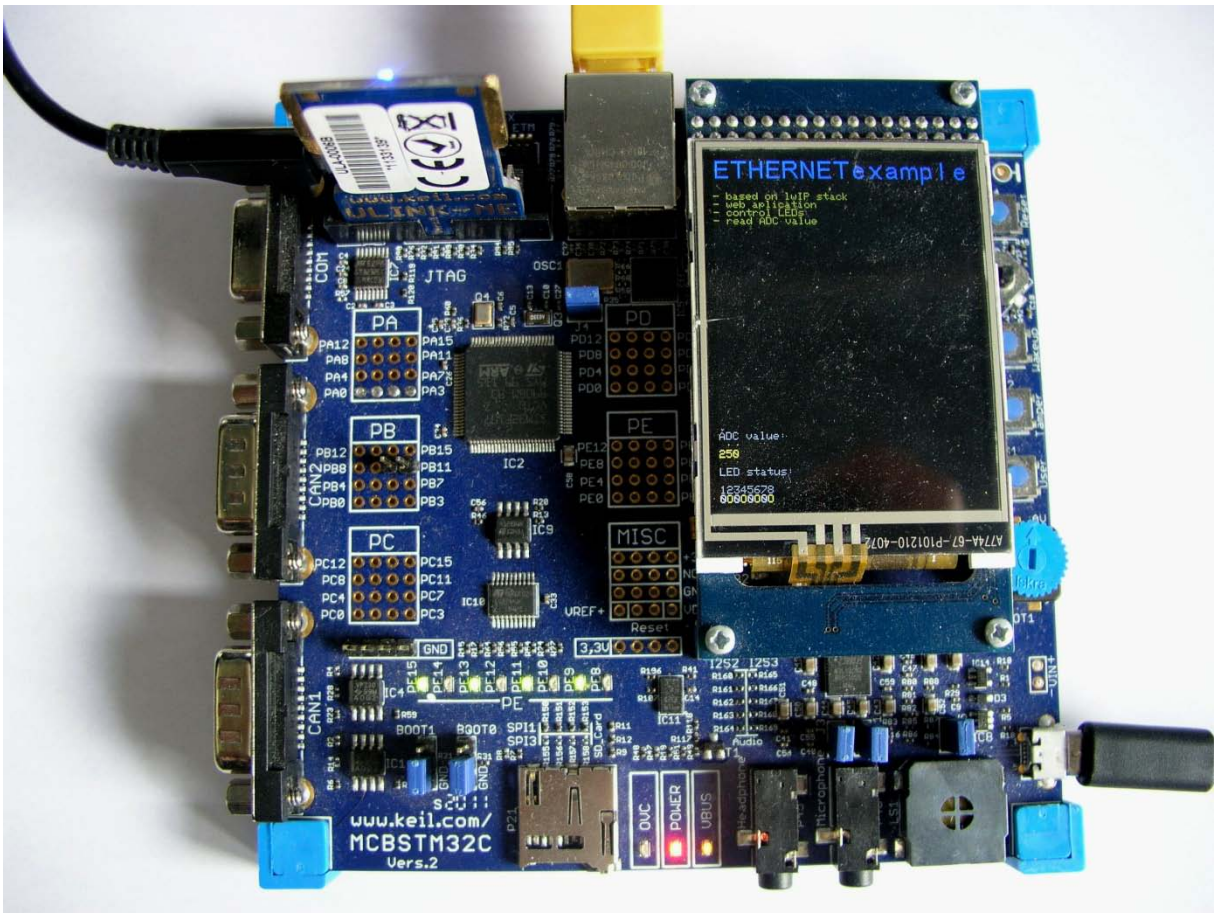
Program běžící na mikrokontroléru vychází z výše zmiňovaného STM32F107 LwIP demonstration package. Na začátku programu je nutné v souboru `stm32f107.c` definovat, jestli se bude používat RMII nebo MII. Inicializace probíhá ve třech základních krocích. Nejprve je volána funkce `System_Setup()`, prostřednictvím které se nastaví hodiny, ethernet, GPIO, NVIC, ADC a dle vývojového kitu LCD a led diody. Tím je provedeno nastavení samotných kitů. Dále je provedeno nastavení lwIP funkcí `LwIP_Init()`. V posledním kroku se nastaví TCP voláním funkce `httpd_init()`. V té se nejprve vytvoří nová TCP struktura. Dále se provede nastavení IP adres, se kterými se bude pracovat, a číslo lokálního portu. Nakonec se nastaví TCP do listening módu a nastaví se funkce, která proběhne po vytvoření TCP spojení.

Chování aplikace je dáno funkcí `http_recv()`, která se nachází v `httpd.c`. V ní se po provedení kontroly správného příjmu a nenulových dat dělí program dle obsahu doručených dat. Ty se pomocí funkce `strncmp()` porovnávají s "`GET /STM32F107AD`" a "`GET /method=get`".

Jestliže je obdržen řetězec `GET /STM32F107AD`, provede se vyčtení hodnoty z ADC. Ta je následně převedena na řetězec a vypsána na LCD displej. Po vynulování pole, prostřednictvím kterého se odesílá odpověď serveru, je do něj zapsána hlavička odpovědí ve tvaru `JSON: HTTP/1.1 GET\r\nContent-type: application/json\r\n\r\n`. V cyklu for je funkcí `STM_EVAL_LEDStatus()` kontrolován stav GPIO, na kterých jsou led diody, a tento stav je přidáván do odpovědi pomocí funkce `strcat()` v JSON formě a ještě zobrazován na displeji. Po zkontrolování všech led diod je následně do odpovědi ještě přidán stav ADC a celková odpověď je poslána funkcí `send_data()`.

V případě obdržení "`GET /method=get`" se postupně kontrolují další části doručené zprávy, kde jsou informace z webové aplikace o zaškrtnutí check boxů, a dle těchto informací je řízen stav led diod na desce.

Na následujícím obrázku lze vidět vzhled aplikace na vývojovém kitu dle nastavení ve webové aplikaci, obr. 4.1.



Obr. 4.2: Vzhled aplikace na MCBSTM32C

5 Aplikace I2C

Tento typ sběrnice je velmi často používaný v oblasti senzorů a akčních členů. Hlavní výhodou je jednoduchost a snadná implementace. Ve vzorovém příkladu je ukázána komunikace prostřednictvím I2C mezi stm3210c-eval a BMP085.

5.1 I2C komunikace

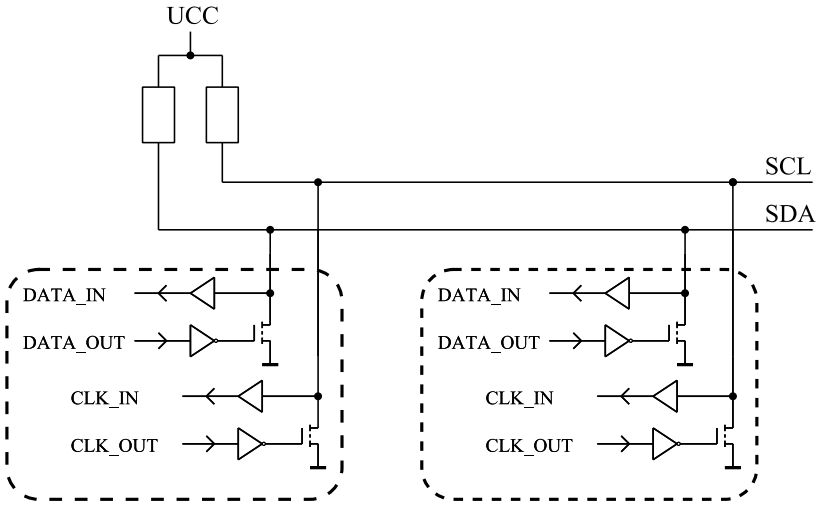
I2C je standardizovaná sběrnice využívající 2 vodiče, datový SDA a hodinový SCL.

Základní charakteristika:

- adresa stanic 7-bitová nebo 10-bitová
- počet stanic omezen celkovou kapacitní zátěží sběrnice (do 400pF)
- možná kombinace TTL a CMOS obvodů
- frekvence hodinových impulsů dána použitým módem (100kHz, 400kHz, 3,4MHz)

[14]

I2C využívá budiče s otevřeným kolektorem, a to umožňuje nedestruktivní konflikty na sběrnici. Kolektory jsou připojeny k Ucc přes pull-up rezistory. V klidovém stavu jsou vodiče v takzvaném recesivním stavu, kdy je na nich log. 1.



Obr. 5.1: I2C sběrnice

Před započetím I2C komunikace každá stanice kontroluje, jestli je sběrnice v klidovém stavu. Každá stanice při vysílání porovnává vyslaná data se stavem sběrnice. Při konfliktu mezi těmito daty, tedy v situaci, kdy na sběrnici vstoupila jiná stanice, nastává arbitráž. Arbitráž se provádí dle obsahu adresového pole. Jestliže jsou pole totožná, je oslovována stejná stanice a nelze konflikt rozpoznat. V případě, že stanice vysílá H a na sběrnici zjistí stav L, odstupuje.

7-bitová adresa

MSB LSB

A_6	A_5	A_4	A_3	A_2	A_1	A_0	R/W

10-bitová adresa

MSB LSB

1	1	1	1	0	A_9	A_8	R/W

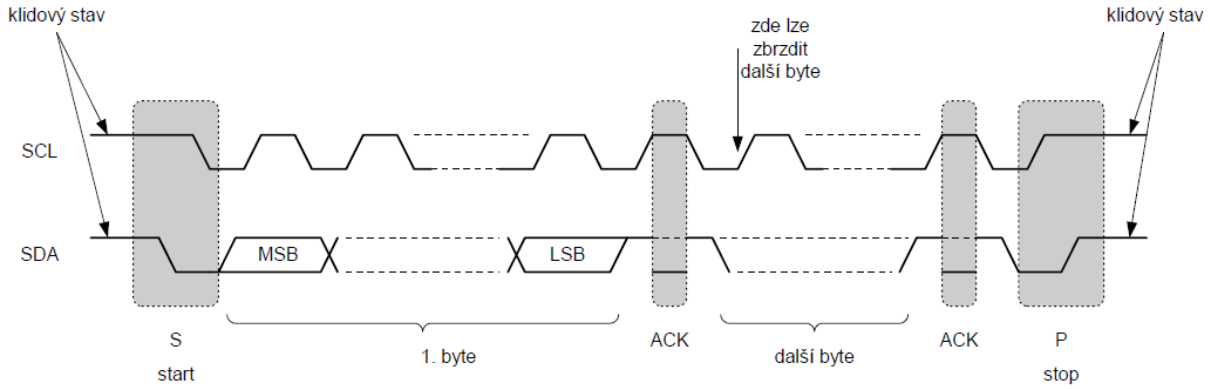
MSB LSB

A_7	A_6	A_5	A_4	A_3	A_2	A_1

R/W=0, přenos master -> slave

R/W=1, přenos slave -> master

Začátek vysílání je zahájen start sekvencí, kdy nejdříve přejde do dominantního stavu SDA a následně SCL. Poté je poslána adresa. Tvar adresy, tedy prvního, případně prvních dvou bytů, byl popsán výše. Přijímací stanice vyšle vždy po přijetí jednoho bytu dat ACK, a tím je zařízena kontrola příjmu celého bytu. Následuje vyslání dat a ukončovací sekvence. Důležitou vlastností slavů je možnost pozdržet hodinový signál, vnutit na SCL stav L, a to v případě, kdy příjemce nestačí zpracovávat data. Grafické znázornění průběhu lze vidět na následujícím obrázku.



Obr. 5.2: Časové průběhy na I2C sběrnici [15]

V [3, s.723] lze najít konkrétní sekvence pro vysílání a příjem dat v různých módech provozu. V dalším popisu budou popsány dvě nejdůležitější sekvence.

7-bitová adr., master vysílání dat

S		Address	A			Data1	A	Data2	A		.....	DataN	A		P
	EV5			EV6	EV8_1	EV8		EV8		EV8				EV8_2	

S= Start, P= Stop, A= Acknowledge

EV5: SB=1, vynulováno přečtením SR1 reg. následované zapsáním adresy do DR reg.

EV6: ADDR=1, vynulováno přečtením SR1 reg. následované přečtením SR2

EV8_1: TxE=1, shift a data reg. jsou prázdné, DATA1 zapsané v DR

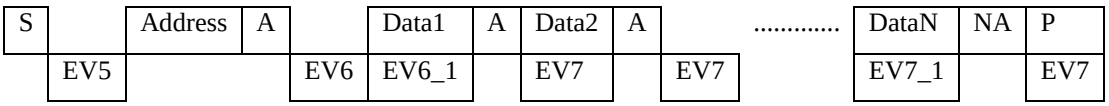
EV8: TxE=1, shift reg. není prázdný, data reg. je prázdný, vynulování zapsáním do DR reg.

EV8_2: TxE=1, BTF=1, programová STOP žádost, TxE a BTF jsou vynulovány hardwarově STOP podmínkou

EV5, EV6, EV8_1, EV8_2 pozdrží hodiny do ukončení příslušné softwarové sekvence

EV8 softwarová sekvence musí být ukončena před koncem přenosu aktuálního bytu. V případě, že tuto podmínku nelze splnit, je doporučeno použít BTF místo TxE na úkor zpomalení komunikace

7-bitová adr., master příjem dat



S= Start, P= Stop, A= Acknowledge, NA= Non-acknowledge

EV6_1: Pouze pro přenos 1 bytu. Zakázání ACK a STOP podmínka jsou generovány po EV6, to jest po vynulování ADDR

EV7: RxNE=1, vynulování přečtením DR reg.

EV7_1: RxNE=1, vynulování přečtením DR reg., ACK=0, a STOP žádost

5.2 Senzor BMP085

BMP085 je digitální senzor tlaku od firmy Bosch. Dále je s ním možné měřit teplotu, a tedy výpočtem ze získaných dat určit i nadmořskou výšku. Obsahuje piezoelektrický senzor, AD převodník, řídící jednotku s EEPROM a I2C interface. V EEPROM jsou uložena kompenzační data vždy pro konkrétní senzor, a to o celkové velikosti 176b, pro výpočet skutečných hodnot z nekompenzovaných dat získaných ze senzoru.

5.2.1 Parametry senzoru

V následujících tabulkách jsou uvedeny základní a maximální parametry senzoru. Zbylé parametry, jako např. hodnoty různých veličin pro jednotlivé módy senzoru, lze najít v datasheetu.

Tab. 5.1: Základní parametry BMP085

Parametr	Symbol	Stav	Min.	Typ.	Max.	Jednotka
Provozní teplota	Ta		-40		85	°C
Napájecí napětí	Vdd		1.8		3.6	V
Rozlišení výstupních dat		tlak		0.01		hPa
		teplota		0.1		°C

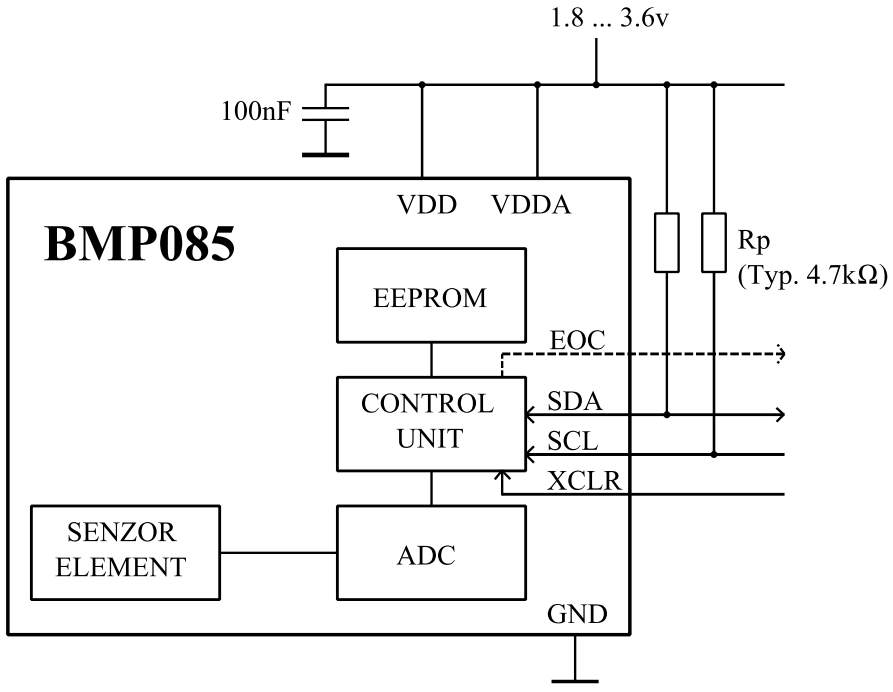
Tab. 5.2: Mezní parametry BMP085

Parametr	Podmínky	Min.	Max.	Jednotka
Skladovací teplota		-40	85	°C
Max. napájecí napětí	všechny piny	-0.3	4.25	V
ESD zatížitelnost	HBM, R=1.5kΩ, C=100pf		± 2	kV
Max. tlak			10 000	hPa

[16]

5.2.2 Zapojení senzoru

Typické zapojení senzoru lze vidět na následujícím obrázku. Nezbytné piny pro funkci jsou SCL, SDA a napájecí piny.



Obr. 5.3: Zapojení BMP085

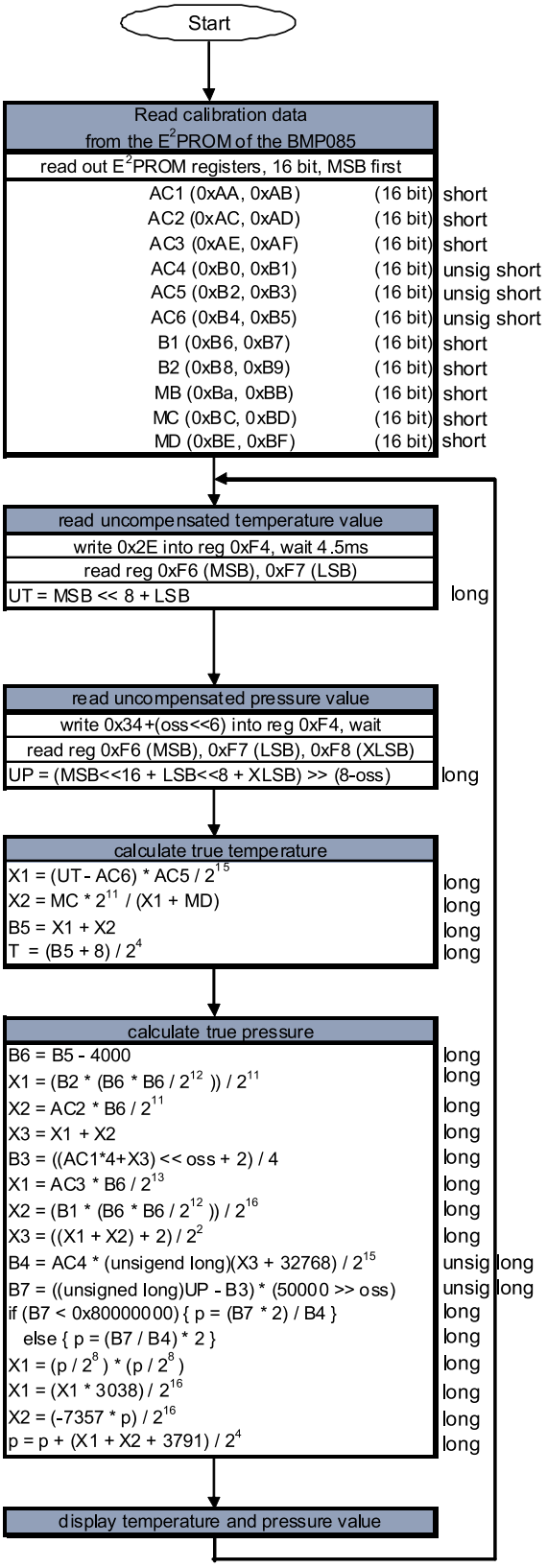
5.3 Výpočetní cyklus

Při získávání hodnot je nutné dodržet přesný postup výpočtu. Diagram ukazující algoritmus výpočetního programu je na Obr. 5.4.

Před samotným začátkem výpočetního cyklu je nutno získat již zmíněná kalibrační data z EEPROM. Ta je rozdělena do 11 slov o 16b. Rozložení a názvy jednotlivých slov a bytů lze opět vidět na Obr. 5.4.

Čekání před čtením UP je závislé na aplikačním módu senzoru (ultra low power, standart, high resolution, ultra high resolution), vlastnosti daných módů jsou uvedeny v datasheetu.

Po provedení výpočetního algoritmu je již možno zobrazit T jako skutečnou teplotu, p jako skutečný tlak a dále případně z těchto hodnot ještě vypočítat nadmořskou výšku.[16]



Obr. 5.4: Výpočetní cyklus u BMP085[16]

5.4 Ukázkový program

Pro tento program jsou využívány standardní knihovny pro stm32. Dále byl vytvořen soubor *BMP085.c*, kde lze nalézt veškeré potřebné funkce pro inicializaci komunikace, komunikaci a vlastní výpočet teploty a tlaku.

V první fázi programu je provedena inicializace LCD displeje. Dále je volána funkce `BMP085_Init()`, v níž se pomocí `I2C_LowLevel_Init()` provede povolení hodin do IC2 periférií a nastaví se piny `I2C_SCL_PIN` a `I2C_SDA_PIN`. Následně jsou registry funkcí `I2C_DeInit()` nastaveny do jejich defalutních hodnot, je nastavena vlastní struktura I2C, povoleno přerušení od I2C a povoleno samotné I2C.

V následujících funkcích jsou získány vlastní hodnoty ze senzoru a vypočítány skutečné hodnoty. Během těchto kroků je využito funkcí `I2C_WriteReg()` a `I2C_ReadReg()`, které budou nyní podrobně vysvětleny, jelikož pro bezchybný běh I2C komunikace je potřebný přesný sled instrukcí.

5.4.1 I2C_WriteReg()

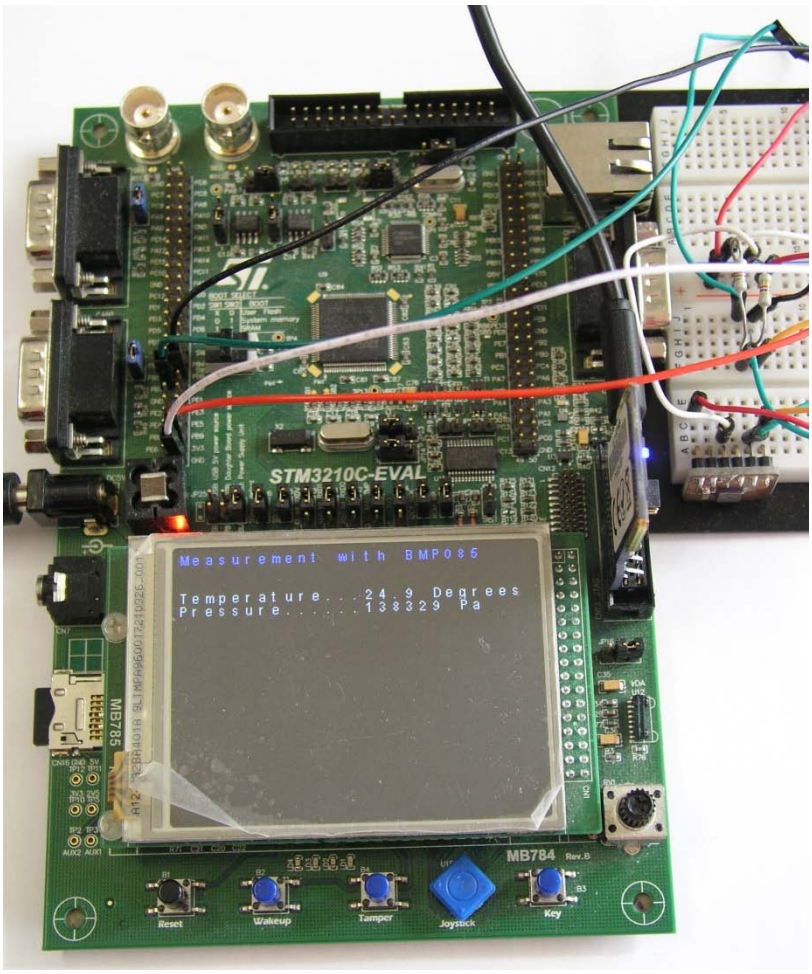
V našem případě má tato funkce pouze jeden vstupní parametr, a to je hodnota, kterou chceme v průběhu funkce poslat. Na začátku funkce je tedy vytvořen buffer, do kterého je vstupní hodnota uložena, a následně testován I2C_FLAG_BUSY. Jestliže je busy_flag neaktivní, je nastaveno DMA a generována startovací podmínka I2C komunikace. Ta je ověřena kontrolou I2C_FLAG_SB. Po poslání adresy, tedy 0xEE pro zápis, se kontroluje EV6 a jsou poslána data. Následuje kontrola TXE FLagu. Vlastní přenos je uskutečněn pomocí DMA, proto je nutné povolit žádost o DMA přenos daného I2C a povolit DMA TX kanál. Ukončení DMA přenosu signalizuje I2C_DMA_TX_TCFLAG. Před generováním STOP podmínky je ještě nutné zkontrolovat stav I2C_FLAG_BTF. Následně je zakázán DMA TX kanál, žádosti o DMA od I2C a vynulován I2C_DMA_TX_TCFLAG.

5.4.2 I2C_ReadReg()

Na začátku funkce se inicializuje přijímací buffer a proměnné funkce. V první části funkce, po kontrole I2C_FLAG_BUSY, se nakonfiguruje DMA a povoluje automatická generace DMA NACK. Po generaci I2C startovací podmínky a testování I2C_FLAG_SB je vyslána adresa 0xEE, tedy pro zápis. Následuje kontrola ADDR flagu, vyslání 0xF6 a je kontrolován TXE Flag.

Následuje opět startovací podmínka, kontrola I2C_FLAG_SB a poslání adresy 0xEF, tedy pro čtení. Po kontrole ADDR Flagu je povolena žádost o DMA, povolen DMA RX kanál a čeká se na ukončení DMA přenosu. Po ukončení je poslána STOP podmínka a je zakázán DMA RX kanál a žádost o DMA. Na závěr je vynulován I2C_DMA_RX_TCFLAG.

Na konci se funkce dělí dle toho, jestli je využívána pro získání UP nebo UT. To se zjistí dle velikosti vstupní proměnné funkce - BufferSize.



Obr. 5.5: I2C aplikace s BMP085

6 Implementace příkladů

Všechny příklady jsou k dispozici na přiloženém CD, kde jsou dle typu vytvořeny složky SD, USB, Ethernet, I2C a složka Materiály, kde lze nalézt všechny potřebné datasheety. Nejjednodušší způsob implementace spočívá v přidání potřebných částí kódu do již hotových projektů. Druhý, náročnější, spočívá v převzetí potřebných částí kódů a jejich následné implementaci. Projekty mají nastavené relativní cesty inkludovaných složek. Jestliže se tedy změní pozice k projektovému souboru, je třeba změnit i tyto cesty. V uVision to tedy znamená udělat změnu v Project - Options for Target - C/C++ - Include paths. V této záložce, konkrétně v Prepocessor symbols - Define, lze udělat i globální definice pro projekt. Dále je vhodné v Project - Options for Target -Utilities - Settings - Flash Donwload - Download Function zaškrtnout Reset and run.

Závěr

V diplomové práci byly dle zadání zhotoveny ukázky příkladů na využití SD karty, USB, Ethernetu a I2C.

Příklad s SD kartou ukazuje minimální potřebné kroky k funkčnosti této periférie. To znamená nastavení SPI na vývojovém kitu, přes které probíhá komunikace, práci s file systémem a low level vrstvou. SD karta je následně využita i v USB příkladech, kde lze vidět další možné využití a implementaci ve složitějších projektech.

Příklady na USB obsahují práci s USB periférií v host a device módu. Oba příklady jsou zaměřeny na třídu MSC. V device módu je pouze předělán příklad od STM pro funkčnost na MCBSTM32C. Příklad na host mód je udělán pro stm3210c-eval z důvodu nutného externího napájení pro MCBSTM32C v tomto režimu a obsahuje ukázku přenosu dat mezi SD kartou a flash diskem připojeným k desce prostřednictvím USB. Ten spočívá ve vytvoření file systému pro SD kartu, přenosu dat do bufferu, zrušení SD file systému, vytvoření file systému pro USB a přenosu dat na flash disk.

Aplikace s Ethernetem je postavena na lwIP stacku a ukazuje komunikaci mezi MCBSTM32C a webovým serverem. Prostřednictvím webové aplikace lze ovládat led diody a vyčítat data z ADC a status led diod. V aplikaci na desce je vytvořena TCP struktura, která čeká na příchozí TCP spojení na všech IP adresách a portu 80. Jestliže jsou obdržena data v požadovaném formátu, provede se buď obsluha led diod nebo vyčtení jejich statusu a hodnoty ADC a tyto hodnoty jsou následně poslány v JSON formátu zpět do webové aplikace, kde jsou zobrazeny. Webová aplikace se na tato data dotazuje v pravidelných intervalech v závislosti na nastavení časovače.

I2C aplikace ukazuje komunikaci se senzorem BMP085 prostřednictvím I2C. Jsou zde tedy názorně vidět potřebné kroky k nastavení I2C a postup při komunikaci.

Použitá literatura

[1] Flstm32. STMICROELECTRONICS. [online]. February 2014 [cit. 2014-04-22]. Dostupné z: http://www.st.com/web/en/resource/sales_and_marketing/promotional_material/flyer/flstm32.pdf

[2] CD00220364. STMICROELECTRONICS. [online]. [cit. 2014-04-22]. Dostupné z: http://www.st.com/web/en/resource/technical/document/datasheet/CD00220364.pdf

[3] CD00171190. STMICROELECTRONICS. [online]. [cit. 2014-04-22]. Dostupné z: http://www.st.com/web/en/resource/technical/document/reference_manual/CD00171190.pdf

[4] Mcbstm32c. ARM LTD AND ARM GERMANY GMBH. [online]. [cit. 2014-04-22]. Dostupné z: http://www.keil.com/mcbstm32c/

[5] CD00212441: UM0600 User manual. STMICROELECTRONICS. [online]. [cit. 2014-04-22]. Dostupné z: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/CD00212441.pdf

[6] ULINK-ME Debug Adapter. ARM LTD AND ARM GERMANY GMBH. [online]. [cit. 2014-04-22]. Dostupné z: http://www.arm.com/products/tools/debug-adapters/ulink/ulink-me.php

[7] Mcbstm32c base board schematics. ARM LTD AND ARM GERMANY GMBH. [online]. [cit. 2014-04-22]. Dostupné z: http://www.keil.com/mcbstm32c/mcbstm32c-base-board-schematics.pdf

[8] FatFs - Generic FAT File System Module. [online]. [cit. 2014-04-22]. Dostupné z: http://elm-chan.org/fsw/ff/00index_e.html

[9] Universal Serial Bus. USB-IF. [online]. [cit. 2014-04-22]. Dostupné z: http://www.usb.org/home

[10] Universal Serial Bus Specification Revision 2.0. [online]. [cit. 2014-04-22]. Dostupné z: http://www.usb.org/developers/docs/usb20_docs/

[11] Beyondlogic: USB Descriptors. [online]. [cit. 2014-04-22]. Dostupné z: http://www.beyondlogic.org/usbnutshell/usb5.shtml

[12] CD00289278: UM1021. STMICROELECTRONICS. [online]. 08-Mar-2012 [cit. 2014-04-22]. Dostupné z: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/CD00289278.pdf

[13] LwIP Wiki. [online]. [cit. 2014-04-22]. Dostupné z: http://lwip.wikia.com/wiki/LwIP_Wiki

[14] UM10204: I 2 C-bus specificatio n and user manual. [online]. 4 April 2014 [cit. 2014-04-22]. Dostupné z: http://www.nxp.com/documents/user_manual/UM10204.pdf

[15] 9 - Doplněk IIC: Sériová sběrnice IIC. *KAE/MPP - Mikroprocesory a počítače* [online]. [cit. 2014-04-22]. Dostupné z: https://courseware.zcu.cz/

[16] BMP085 Data sheet. [online]. 15 Oct 2009 [cit. 2014-04-22]. Dostupné z: https://www.sparkfun.com/datasheets/Components/General/BST-BMP085-DS000-05.pdf