VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

MEMS AKCELEROMETR S MIKROKONTROLÉREM

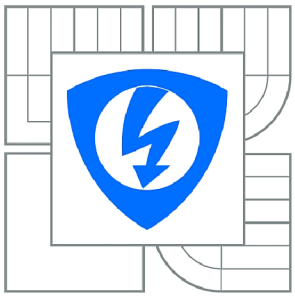
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JAN VROBEL
AUTHOR

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

MEMS AKCELEROMETR S MIKROKONTROLÉREM

MICROCONTROLLER BASED SYSTEM EQUIPPED BY MEMS ACCELEROMETER

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

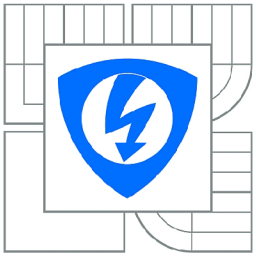
AUTOR PRÁCE — JAN VROBEL

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. ZDENĚK BRADÁČ, Ph.D.

SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student:	Jan Vrobel	***ID:***	125185
Ročník:	3	***Akademický rok:***	2012/2013

NÁZEV TÉMATU:

MEMS akcelerometr s mikrokontrolérem

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte koncepci mikrokontrolérového systému s MEMS akcelerometrem. Systém navrhněte jako miniaturní systém vybavený mikrokontrolérem a nezbytnými rozhraními. Navrhněte elektroniku, realizujte DPS, osaďte a oživte. Navrhněte demonstrační úlohu pro jeho využití. Vybavte programovým vybavením, otestujte a předvěďte funkčnost.

DOPORUČENÁ LITERATURA:

Pavel Herout: Učebnice jazyka C, KOPP, 2004, IV. přepracované vydání, ISBN 80-7232-220-6
Dle pokynů vedoucího práce.

Termín zadání:	11.2.2013	***Termín odevzdání:***	27.5.2013

Vedoucí práce: doc. Ing. Zdeněk Bradáč, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Práce se zabývá realizací systému pro měření zrychlení postaveném na MEMS senzoru a mikrokontroléru. V jednotlivých částech je popsán průběh realizace od prvotního koncepčního návrhu přes návrh elektroniky, volbu jednotlivých součástek až po realizaci desky plošných spojů. Zařízení je navrženo jako miniaturní baterií napájený přístroj s bezdrátovým datovým rozhraním. Dále je popsána realizace firmware demonstrující funkčnost celého systému a aplikace pro PC pro komunikaci se zařízením.

KLÍČOVÁ SLOVA

měření zrychlení, krokoměr, přenosný systém, MEMS akcelerometr, mikrokontrolér, ADXL345, SPI, AVR, Bluetooth, FRAM

ABSTRACT

The object of this thesis is the implementation of the measurement system based on MEMS acceleration sensor and a microcontroller. In the individual chapters of the thesis is described the process of implementation from the initial conceptual design to electronics design, selection of components, to the implementation of the PCB. The device is designed as a miniature battery-powered device with wireless data interface. There is also described the implementation of the firmware demonstrating the functionality of the system and the PC application to communicate with the device.

KEYWORDS

acceleration measurements, pedometer, portable system, MEMS accelerometer, microcontroller, ADXL345, SPI, AVR, Bluetooth, FRAM

VROBEL, Jan *MEMS akcelerometr s mikrokontrolérem*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2013. 68 s. Vedoucí práce byl doc. Ing. Zdeněk Bradáč, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „MEMS akcelerometr s mikrokontrolérem“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu doc. Ing. Zdeňkovi Bradáčovi Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy při práci na projektu, rodině za velikou trpělivost, firmě Schiebel s.r.o. za podporu při studiu a doc. Ing. Adamovi Heroutovi Ph.D. za motivaci, podporu a odborné rady.

Brno . . . . . .

. . . . . .

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Dnešní moderní technologie umožňují vyrobit univerzální akcelerometr (přístroj pro měření zrychlení) kapesní velikosti. Takové zařízení by bylo možno využít k široké paletě měřicích úkolů. Od jednoduchého měření zrychlení auta/motocyklu/letadla, přes komplikovanější úlohy typu vyhodnocení vibrací, až po úlohy komplexnější, kdy je vyhodnocován průběh změn směru a velikosti zrychlení, např. aplikace typu krokoměr, monitor aktivity nebo detektor pádu člověka na zem.

Cílem bakalářské práce je navrhnout hardware a software měřicího přístroje, který bude schopen měřit a vyhodnocovat zrychlení ve třech osách. Přístroj bude osazen senzorem zrychlení, mikrokontrolérem, pamětí pro uložení dat, displejem pro zobrazení měřených dat a obsluhu, a bezdrátovým rozhraním pro přenos měřených dat. Dále zde budou přítomny napájecí obvody a další pomocné obvody zajišťující funkci celého zařízení.

Při návrhu obvodového schématu zmíním některá specifika použitých komponent a výpočty hodnot důležitých součástek. V rámci návrhu desky plošných spojů (DPS) zmíním podstatné požadavky na návrh DPS, zejména specifika spínaných zdrojů. Popíši způsob osazovaní DPS a průběh oživování jednotlivých funkčních celků.

Návrh software bude rozdělen do dvou částí. Jednak bude vytvořen software senzoru samotného, dále nazývaný firmware, a také bude vytvořena jednoduchá demonstrační aplikace pro PC/Windows.

Také zmíním průběh testování jednotlivých částí hardware a sběr dat pro finální nastavení parametrů senzoru.

1 REŠERŠE

Při přípravě a realizaci tohoto projektu jsem narazil na několik podobných produktů.

Jsou to jednak sportovní doplňky tzv. wearable sensors. Například Nike+ senzor http://http://nikeplus.nike.com/plus/, umísťovaný na botu a obsahující akcelerometr bezdrátově propojený se zařízením typu Apple iPod. Společnost Nike, vyrábí taktéž náramek Nike+ FUELBAND, což je monitor a záznamník celodenní aktivity postavený na MEMS akcelerometru. Firma Adidas nabízí produkty řady miCoach, http://micoach.adidas.com/ s podobnými funkcemi jako konkurenční Nike. Firma Philips nabízí také podobný monitor denní aktivity s názvem DirctLife, http://www.directlife.philips.com/. Všechno jsou to produkty určené pro sledování aktivity, neumožňující jiné využití.

Velmi zajímavým produktem je Osobní pohybový senzor české firmy PRINCIP, obsahující 3D akcelerometr a 3D gyroskop. Aktuální verze jejich senzoru však podporuje pouze 50 měření za vteřinu, což je pro některé aplikace nedostatečné, firma však pracuje na další verzi senzoru, kde slibuje výrazné zlepšení parametrů. Firma taktéž nabízí senzor s exportem surových dat pro další analýzu.

Další typ systémů tvoří speciální mobilní telefony, např. Emporia http://www.emporia.cz i v provedení náramkových hodinek ProSafe http://www.prosafe.penzista.net/, navržené již s přihlédnutím k jejich možnému použití jako detektor pádu. Tyto přístroje jsou obvykle vybaveny různými SOS tlačítky, detektory odepnutí pásku a podobně. Mnohdy je k těmto zařízením nabízena nadstandardní služba připojení k pultu centrální ochrany s dohledem 24 hodin denně např. http://www.stavebnictvi3000.cz/clanky/osobni-alarm-nove-generace/, http://www.kde.je/. Tyto přístroje však nabízí pouze detekci pádu, případně dalších událostí a mají tedy omezené použití, nelze je použít jako univerzální měřiče zrychlení.

Dalším typem systémů k měření zrychlení jsou různé průmyslové systémy, sestávající ze senzoru zrychlení(obvykle piezoelement) a vyhodnocovací jednotky, např. čidla vibrací společnosti AURA a.s.http://www.auranet.cz/index.php?m1=3&m2=16&lang=1, Brüel & Kjær http://www.bksv.com/Products/transducers/vibration/accelerometers.aspx. Propojení mezi senzorem a jednotkou je realizováno metalickými vodiči. Tyto senzory jsou používány k měření zrychlení(vibrací) na stacionárních měřicích pracovištích, např. na vibračních stolech používaných k ověřování seismické odolnosti, nebo jsou vestavěny, např. v obráběcích strojích pro detekci vibrací. Nedokáží měřit statická zrychlení, jejich princip je založen na piezoelektrickém jevu, kdy během deformace piezoelektrického krystalu můžeme na jeho výstupních elektrodách naměřit stejnosměrné napětí úměrné rychlosti deformace.

2 NÁVRH KONCEPCE SYSTÉMU

Mezi existujícími produkty jsem nenalezl takový, který by měl ve svých možnostech měřit zrychlení s vysokou vzorkovací frekvencí a který by nabízel možnost záznamu takto rychlého měření. Z toho vyplynuly některé z požadavků na návrh. Při návrhu koncepce systému postaveném na MEMS akcelerometru s mikrokontrolérem jsem si stanovil následující cíle:

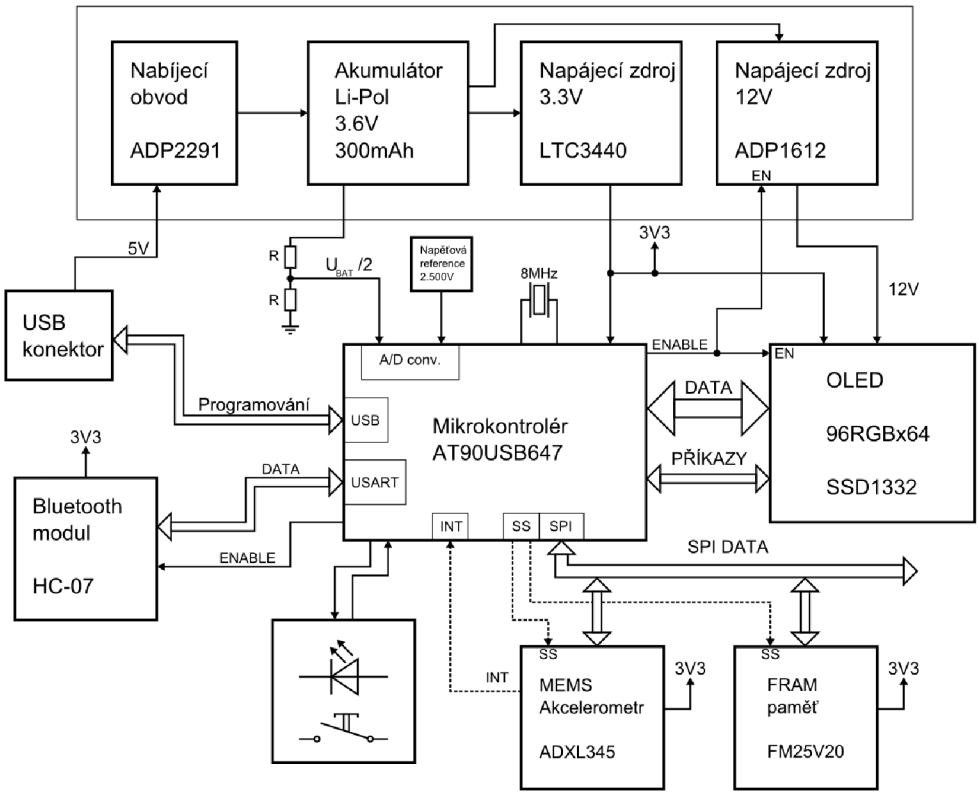
- přenosné zařízení kompaktních rozměrů s univerzálním použitím
- široký rozsah měřených zrychlení
- možnost zaznamenat měření zrychlení s četností alespoň 500 vzorků/s
- schopnost měřit statické i dynamické zrychlení
- přítomnost zobrazovacího displeje
- přenos dat ze zařízení bezdrátově
- nízká spotřeba, měření napětí baterie se signalizací vybití
- záznam měření do *nevolatilní* paměti

Celkové rozměry prototypu by měly být menší než běžný mobilní telefon. Tento požadavek souvisí s přenosností a možností používání zařízení v režimu krokoměru, nebo osobního senzoru denní aktivity. Pro ostatní měřicí úkoly nejsou rozměry zařízení až tak podstatné.

Na obrázku 2.1 je blokové schéma měřicího systému. Data z akcelerometru bude zpracovávat 8-bitový mikrokontrolér řady AVR, který bude obsluhovat paměť pro ukládání výsledků a pomocných dat, Bluetooth rozhraní, OLED displej (zobrazení měření a stavů) a další ovládací prvky. Napájecí část tvoří tři obvody a Li-Pol akumulátor.

2.1 MEMS Akcelerometr

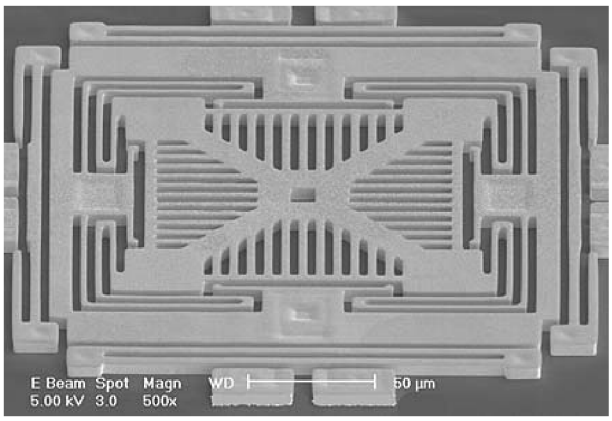
MEMS (MicroElectroMechanical Systems) je název pro produkty i pro samotnou technologii používající techniky známé z výroby integrovaných obvodů k výrobě mechanických, elektromechanických a elektronických součástek i funkčních celků integrovaných společně na jednom křemíkovém čipu [Wikipedie][1]. MEMS technologií se vyrábějí převážně senzory mechanického pohybu (akcelerometry, gyroskopy, mikrofony, senzory tlaku...), které přesně definovaným způsobem převádějí mechanickou veličinu či energii na elektrický signál, ale lze touto technologií vyrábět i akční členy (mikromotory, mikročerpadla) a mechanické převody s rozměry komponent v řádu mikrometrů až desetin milimetrů. Velikost celého MEMS obvodu je potom v řádu desetin až několika jednotek milimetrů. U senzorů se nejčastěji převod mezi mechanickou a elektrickou veličinou realizuje elektrostaticky - po-



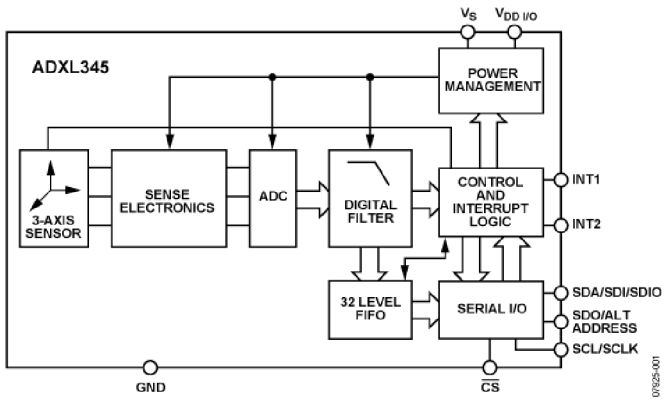
Obr. 2.1: Blokové schéma systému

mocí miniaturních proměnných kondenzátorů, jejichž jedna elektroda je pevná a druhá je připevněna na detekční část senzoru. Touto detekční částí je v případě akcelerometru seismická hmotnost, uložená na pružinách. Obvykle je na jednom čipu integrována mikromechanická i elektronická (vyhodnocovací) část daného zařízení, pak je takové zařízení nazýváno termínem „System-On-Chip“. MEMS obvody jsou obvykle zapouzdřeny do podoby klasických SMD součástek, obvykle do miniaturních pouzder QFN,LGA a podobných. Jedno z možných provedení MEMS struktury akcelerometru je na obr. 2.2

Výrobou MEMS akcelerometrů se zabývá více společností. Např. *STMicroelectronics*, *Analog Devices*, *InvenSense*, a jiné. Po prozkoumání vlastností a dostupnosti několika vytipovaných obvodů jsem si vybral akcelerometrický senzor *Analog Devices* **ADXL345** [2], který uvnitř pouzdra obsahuje tříosý akcelerometr, vyhodnocovací a pomocnou elektroniku s digitálním rozhraním **SPI** (Serial Peripheral Interface) pro připojení k mikrokontroléru. Blokové schéma obvodu je na obrázku 2.3 . Má rozsah měření od $\pm 2g$ až do $\pm 16g$. Spotřeba tohoto senzoru se při napájecím napětí 3.3V pohybuje od max. 180µA při rychlosti vzorkování 3200 Hz až po 40µA při 1 Hz a ve standby režimu. Tento obvod jsem si vybral jednak pro jeho vhodné parametry,



Obr. 2.2: MEMS tříosý akcelerometr - struktura

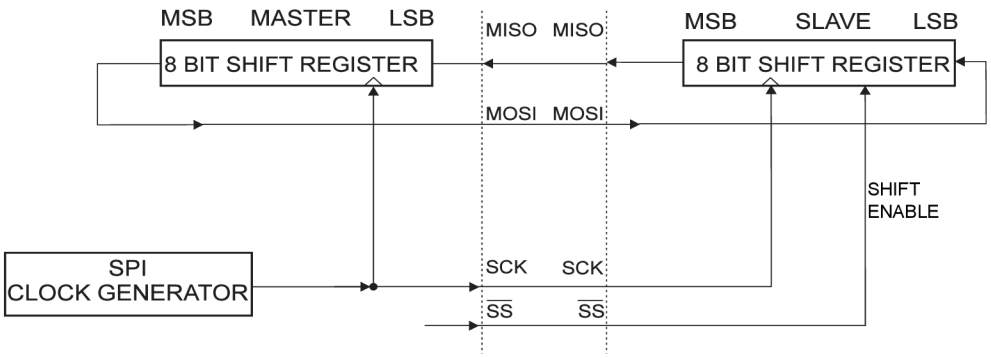


Obr. 2.3: ADXL345 - blokové schéma

dále pro jeho dostupnost jako samostatný obvod v LGA pouzdře(„bezvývodové“ pouzdro 3x5x1 mm, 14 vývodů v rastru 0.8 mm), a také díky možnosti zakoupit na Internetu modul osazený tímto akcelerometrem a klasickými drátovými vývody v rastru 2.54mm. Díky tomu je možno při vývoji zařízení použít klasických technologií pro ruční osazování plošných spojů. K mikrokontroléru je ADXL345 připojen přes sběrnici **SPI**.

2.1.1 Sběrnice SPI

Sběrnice SPI je vysokorychlostní plně duplexní osmibitová nebo šestnáctibitová sériová sběrnice používající v základním provedení čtyři signálové vodiče obr. 2.4. Jsou to vodiče **SCK**, **MISO**, **MOSI** a **SS**. Maximální hodinový kmitočet sběrnice je 70 MHz, počet zařízení není omezen, každé zařízení musí mít přiveden zvláštní vodič pro signál SS, ostatní vodiče jsou společné. Komunikace na sběrnici probíhá



Obr. 2.4: Schéma propojení Master-Slave na sběrnici SPI

tak, že Master(může být na sběrnici pouze jeden, řídí provoz na sběrnici) nejprve nastaví vybraný vodič SS na úroveň LOW, tím aktivuje konkrétní Slave přijímač. Poté postupně synchronně s hodinovým signálem vystavuje na sběrnici data ze svého posuvného registru(bufferu pro SPI). Zároveň Slave (přijímač), který má vstup SS v úrovni LOW tato data přijímá do svého posuvného registru. Synchronně s příjmem dat Slave vystavuje do zpětné SPI linky data ze svého posuvného registru. Tato data zároveň přijímá Master. Během jednoho cyklu tedy proběhne výměna obsahu obou registrů (duplex) mezi Master a Slave zařízeními.

2.2 Mikrokontrolér

Jako hlavní řídící obvod jsem zvolil mikrokontrolér od společnosti *Atmel* z řady AVR, 8-bitový **AT90USB647** [3]. Tento mikrokontrolér nabízí dostatečný výpočetní výkon a všechny potřebné periferní obvody. V zapojení jsem použil následující funkční části: USB rozhraní - pro nahrávání firmware - není potřeba používat programátor, A/D převodník pro měření napětí baterie, USART rozhraní pro Bluetooth připojení, paralelní rozhraní pro připojení OLED displeje a SPI rozhraní pro komunikaci s akcelerometrem a přídavnou FRAM pamětí pro ukládání většího množství dat. Dále jsem využil univerzálních vstupů/výstupů pro připojení ovládacích tlačítek, přepínačů a signalizační RGB LED. Tyto komponenty budou osazeny především z vývojových důvodů. Dále firmware akcelerometru využívá vnitřní obvody mikrokontroléru, jako jsou časovače, paměť RAM atd.

Mikrokontrolér obsahuje 64kBytů Flash EEPROM paměti pro program, 4kByty operační paměti a bude provozován na frekvenci 8MHz. Při této frekvenci nabízí výkon 8MIPS v RISCové architektuře.

Proudová spotřeba mikrokontroléru v aktivním režimu je při napájení 3.3 V asi 5.5 mA, v režimu Power-down kolem 33 µA.

2.3 Napájecí část, OLED, FRAM, spotřeba, uživatelské rozhraní

Napájecí část tvoří trojice integrovaných obvodů, kde ADP2291 zajišťuje správné nabíjení Li-Pol akumulátoru, LTC3440 je Buck-Boost DC/DC měnič napětí s výstupem 3.3 V pro napájení digitální části zařízení a ADP1612 DC/DC měnič 12 V poskytuje napájecí napětí pro zobrazovací OLED displej. Všechny integrované obvody jsou vybrány s ohledem na možnost ručního pájení, v bezvývodových pouzdrech se dnes vyrábějí obvody s lepšími parametry i funkcemi.

2.3.1 Nabíjecí obvod s ADP2291

Obvod *Analog Devices* ADP2291 [4] je monolitický lineární nabíjecí obvod navržený pro nabíjení jednoho Li-Pol článku. Algoritmus nabíjení Li-Pol článků je poměrně jednoduchý, nicméně je nutno nepřekročit maximální hodnoty v jednotlivých fázích nabíjecího cyklu. Vybitý článek se nabíjí konstantním proudem (hodnotu udává výrobce článku, obvykle max. 1C, kde C označuje kapacitu článku v Ah), po dosažení napětí 4.2V (kolem 70% kapacity je nabito) nabíjení přechází do režimu konstantní napětí, kdy postupně klesá nabíjecí proud až k nulové hodnotě. ADP2291 detekuje nabíjecí proud a při poklesu na 1/10 nominální hodnoty automaticky ukončí nabíjení.

Pokud je článek Li-Pol hluboce vybitý (pod napětí 2.8 V), je nutno jej nabíjet nízkým proudem, jinak by hrozilo jeho zničení, proto ADP2291 zahajuje nabíjení takového článku proudem o velikosti 1/10 proudu nominálního.

Nabíjení článku je indikováno LED diodou připojenou na výstupní pin CHG. K nastavení nominálního nabíjecího proudu se používají externí nastavovací rezistory, v datasheetu [4] jsou uvedeny jejich konkrétní hodnoty podle velikosti nabíjecího proudu.

2.3.2 Hlavní napájecí zdroj LTC3440

Jako hlavní napájecí zdroj v zapojení jsem vytipoval obvod *Linear Technology* LTC3440 [5]. Jedná se o Buck-Boost DC/DC měnič napětí, tedy měnič, jenž je schopen pracovat jak v režimu snižujícím, tak zvyšujícím napětí. S připojenou externí tlumivkou 10 μH je schopen dodávat proud až 600 mA. Při napětí nižším než 2.5 V se přepne automaticky do Shutdown režimu s velmi nízkou spotřebou (max. 1 μA), a tím chrání připojený Li-Pol článek proti přílišnému vybití. Při návrhu desky plošných spojů je nutné zohlednit doporučení výrobce pro rozmístění kritických součástek, především výkonové tlumivky, výstupního kondenzátoru a součástek

zpětné vazby. Dále je potřeba dostatečně odstínit tlumivku pro omezení vf elektromagnetického rušení.

Vlastní spotřeba tohoto obvodu je v závislosti na aktuálním režimu maximálně 1 mA.

2.3.3 Pomocný napájecí zdroj 12 V ADP1612

Pro napájení budičů OLED displeje je použit obvod *Analog Devices* ADP1612 [6]. Jedná se o Step-Up (zvyšující) DC-DC měnič napětí. Výstupní napětí obvodu je pomocí externích rezistorů nastaveno na 12 V. Zapínání a vypínání tohoto regulátoru lze ovládat externím signálem do vstupu EN, což je v zařízení využito pro režimy nízké spotřeby. Pro rozmístění součástek a stínění vf rušení je vhodné dodržet pokyny výrobce, které jsou uvedeny v datasheetu [6].

Vlastní spotřeba tohoto obvodu je maximálně 4 mA, v režimu nízké spotřeby pak 2 μA.

2.3.4 OLED panel

Jako zobrazovací displej jsem zvolil miniaturní OLED panel s rozlišením 96RGB x 64 bodů. OLED displej jsem volil z důvodu jeho vyšší odolnosti proti otřesům, ve srovnání s LCD technologií. Zvolený displej má úhlopříčku 1.04". Je řízen řadičem SSD1332 [7], který poskytuje možnost připojení přes paralelní nebo sériové SPI rozhraní. V zapojení je použito paralelního připojení, protože na sériové sběrnici bude komunikovat akcelerometr a přídavná FRAM paměť.

Spotřeba OLED displeje je v závislosti na aktuálním provozním režimu od 5 μA ve Sleep módu do přibližně 40 mA při rozsvícení všech segmentů displeje.

2.3.5 FRAM

Pro ukládání naměřených dat a konfigurace je v zařízení použita FRAM paměť Ramtron FM25V20 [8]. Jedná se o sériovou 2Mb F-RAM paměť pracující na feroelektrickém principu. Tato paměť je nevolatilní, nepotřebuje pro uchování informace stálé napájení. Přitom je podstatně rychlejší, než běžné paměti typu Flash/EEPROM. Je tedy použitelná jednak pro trvalé ukládání dat nezávislé na napájení, ale také jako paměť RAM. Má vysoký počet povolených zápisových cyklů, výrobce v současné době zaručuje 10^{14} zápisových a čtecích cyklů. Vnitřně tato paměť pracuje na fyzikálním principu změny polarizace atomů feroelektrického materiálu, které tvoří samostatné paměťové buňky.

Spotřeba paměti se pohybuje od max. 3 μA ve Sleep režimu do asi 1 mA v aktivním režimu při frekvenci sběrnice 4 MHz.

2.3.6 Obvody rozhraní

Hlavním prvkem rozhraní k okolním přístrojům (PC, příp. mobilní telefon apod.) je Bluetooth modul HC-07 [9], výrobce Guangdong, Čína, http://www.wavesen.com/probig.asp?id=17. Tento modul podporuje Bluetooth ve verzi 2.0. Pracuje na frekvenci 2.4 GHz s max. rychlostí přenosu 1.4 Mbit/s. Doporučená maximální přenosová rychlost pro spojení s PC je 115,200 baudů. Dosah signálu je na volném prostranství asi 10m. Bluetooth modul je k mikrokontroléru připojen pomocí integrovaného USART rozhraní.

Spotřeba Bluetooth modulu je při aktivním přenosu dat 35 mA, v aktivním režimu bez přenosu dat asi 8 mA a v Sleep režimu 40 μA.

Rozhraní USB integrované do mikrokontroléru bude využíváno pouze pro účely nahrávání firmware.

2.3.7 Spotřeba

Vzhledem k tomu, že systém bude napájen z akumulátoru, vypočítal jsem odhadovanou spotřebu všech funkčních bloků a z toho plynoucí odhadovanou dobu provozu na akumulátor v různých režimech činnosti. Výpočet je proveden pro článek s nominální kapacitou 300 mAh. Proudové spotřeby vycházejí z katalogových údajů a v praxi se mohou lišit.

Tab. 2.1: Spotřeba proudu jednotlivých funkčních bloků

Funkční blok	Aktivní režim [mA]	Sleep režim [μA]
ADXL345	0.18	40
AT90USB647	5.5	33
LTC3440	0.6	25
ADP1612	4	2
Dělič napětí	0.021	21
OLED	40	5
FRAM	1	3
Bluetooth	8(35)	4
Celkem	59(87)	133
bez OLED a BT	7	133

Celková spotřeba zařízení je tedy v režimu maximální aktivity všech funkčních částí přibližně 87 mA. Z toho vyplývá výdrž na plně nabitý akumulátor 300 mAh asi 3.5 hodiny. Pro měření zrychlení je tento čas vyhovující, obvykle měřené děje nebudou nikdy probíhat tak dlouhý čas. V případě použití měřiče jako krokoměru, či detektoru pádu je ovšem tato doba provozu zcela nedostatečná. Ovšem v tomto případě nebude nikdy nutno trvale přenášet data - tedy lze vypnout Bluetooth rozhraní až do chvíle potřeby přenést data nebo do detekce některé významné události(pád, dlouhá doba nečinnosti, nízké napětí akumulátoru apod.), podobně lze odpojit displej(včetně napájecího měniče). Tím se spotřeba sníží na přibližně 7 mA, z čehož vyplývá doba provozu asi 43 hodin. Dále lze prodloužit tuto dobu přepnutím většiny součástek do úsporného režimu v případě, kdy nebude delší dobu zaznamenán pohyb, přepnutí zpět do aktivního režimu zajistí akcelerometr pomocí vyvolání externího přerušení. Lze též zvýšit kapacitu akumulátoru. Tímto způsobem by bylo možno prodloužit provozní dobu řádově na jeden týden i více, dle míry používání. Maximální teoretická doba mezi nabíjecími cykly akumulátoru, pokud by zařízení bylo trvale v úsporném režimu(spotřeba 133 μA) je přibližně 94 dnů.

3 NÁVRH OBVODOVÉHO SCHÉMATU

3.1 Napájecí část

Schéma zapojení (s. 27) vychází z blokového schématu obr. 2.1.

3.1.1 Nabíjecí obvod

Napájecí napětí 5V z USB konektoru je přes Schottkyho ochrannou diodu přivedeno na vstup nabíjecího obvodu s ADP2291 (schéma s. 27-A-2,3). Zapojení nabíjecího obvodu vychází z katalogového zapojení [4]. Na s. 13 datasheetu [4] je uvedeno doporučení pro volbu externího tranzistoru Q1. Parametry tohoto tranzistoru by měly odpovídat rovnicím (5), (6) a (7). V našem případě vycházíme ze vstupních údajů:
$V_{ADAPTER} = 5\ V, I_{MAX} = 250\ mA, V_{PROTECT} = 0.32\ V, V_{ADJ} = 1.87\ V, V_{RS} = 75\ mV$ Vypočteme požadavky na vlastnosti tranzistoru Q1:

$$\beta_{MIN} = \frac{I_{MAX}}{I_B} = \frac{250mA}{40mA} = 6.25 \tag{3.1}$$

$$\begin{aligned} V_{CE(SAT)} &= V_{ADAPTER} - V_{PROTECT} - V_{RS} - V_{BAT} \\ &= 5\ V - 0.32\ V - 0.075\ V - 4.2\ V \\ &= 0.405\ V \end{aligned} \tag{3.2}$$

$$\begin{aligned} P_{DISS} &= I_{MAX} \times (V_{ADAPTER} - V_{PROTECT} - V_{RS} - V_{BAT}) \\ &= 0.25A \times 5\ V - 0.32\ V - 0.075\ V - 2.8\ V \\ &= 0.45\ W \end{aligned} \tag{3.3}$$

Na základě vypočtených parametrů tranzistoru jsem vybral vhodný typ FMMT718 firmy ZETEX. Zvolený nabíjecí proud 250 mAh bude vybitý akumulátor o kapacitě 300 mAh nabíjet přibližně 72 minut.

3.1.2 Hlavní napájecí zdroj 3.3 V

je tvořen Buck-Boost DC/DC měničem LTC3440 [5] (schéma s. 27-A-4,5). Tento obvod transformuje napětí akumulátoru (2.8 až 4.2 V) na stabilní napájecí napětí 3.3 V. Toto napětí je poté rozvedeno do všech aktivních funkčních bloků. Zapojení zdroje vychází z doporučeného schématu zapojení výrobce(s.1, [5]). Vzhledem

k nízké předpokládané spotřebě proudu všech komponent napájených 3.3 V zdrojem jsem použil tzv. Burst módu (s.9, [5]) měniče, kdy je vnitřní řídicí elektronika spínaného zdroje během každého cyklu přenosu energie z cívky do výstupního kondenzátoru přepnuta do sleep módu. V Burst módu je vlastní spotřeba obvodu LTC3440 pouze 25 μA.

3.1.3 Pomocný napájecí zdroj 12 V

je postaven na Step-Up DC/DC měniči ADP1612 [6] (schéma s. 27-A-7,8). Poskytuje napájecí napětí pro napájení driverů OLED displeje. Zapínání a vypínání tohoto měniče je ovládáno mikrokontrolérem. Měnič je provozován na frekvenci 650 kHz (vyšší účinnost než na 1.3 MHz, menší vf rušení, s. 11, [6]). Výstupní napětí je nastaveno odporovým děličem ve zpětné vazbě. Odpor rezistoru R13 volíme 10 kΩ, odpor rezistoru R14 v děliči je definován vztahem(s. 13, rov.(2), [6]):

$$R_14 = R_{13} \times (\frac{V_{OUT} - 1.235}{1.235}) = 87.2\ k\Omega \tag{3.4}$$

Nejbližší vyráběnou hodnotou rezistoru je 86.6 kΩ. Volba cívky, vstupního a výstupního kondenzátoru se řídí doporučením výrobce, pro stabilní funkci dle katalogového zapojení obr. 44, s. 17, [6] je doporučeno L2=10 μH, C8=C11=10 μF, 16V.

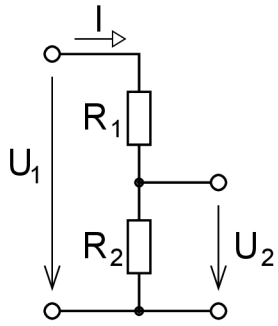
3.2 Digitální část

3.2.1 Mikrokontrolér

Digitální část zapojení tvoří řídicí obvod IC1 - mikrokontrolér AT90USB647 [3] (schéma s. 27-C,D-2,3) a k němu připojené komponenty. Mikrokontrolér je napájen napětím 3.3 V z hlavního zdroje (3.1.2). Taktován je na frekvenci 8 MHz externím krystalovým rezonátorem Y1. Resetovací vstup mikrokontroléru je zapojen dle standardního schématu. Kondenzátor C16 spolu s rezistorem R18 (schéma s. 27-B-1) zajistí dostatečně dlouhé trvání nízké napěťové úrovně na RESET vstupu během připojení napájecího napětí pro spolehlivý start mikrokontroléru. Na vstup AREF je přivedeno referenční napětí pro A/D převodník (měření napětí akumulátoru viz 3.2.1). Vstup AVCC - napájení A/D převodníku je chráněn proti vnějšímu rušení filtrační tlumivkou L3 a kondenzátorem C12.

Měření napětí akumulátoru

Pro měření napětí akumulátoru je použit A/D převodník mikrokontroléru (1. kanál, pin PF0) připojený k akumulátoru přes odporový dělič napětí obr. 3.1 . Pro správ-



Obr. 3.1: Odporový dělič napětí

nou funkci A/D převodníku je nutno zajistit, aby napětí na vstupu A/D převodníku bylo vždy nižší, než napětí na referenčním vstupu A/D převodníku. Toto napětí poskytuje referenční napěťový zdroj LM4120-2.5 a je rovno 2.5 V v širokém rozsahu provozních podmínek. Li-Pol akumulátor má minimální povolené provozní napětí 2.5 V, maximální nabíjecí napětí je 4.2 V. Pro měření je tedy optimální použít dělič 1/2 (schéma s. 27-E-1)vhodně navržený tak, aby zbytečně nespotřebovával energii z akumulátoru (tedy co největší odpor použitých rezistorů) a zároveň byl dostatečně „tvrdý“ vzhledem k proudu, který spotřebovává měřicí vstup A/D převodníku (tedy relativně malý odpor použitých rezistorů). Z datasheetu pro mikrokontrolér lze vyčíst, že analogové vstupy mají vstupní impe-danci typicky $R_i = 100$ MΩ. Vypočteme tedy zatížení akumulátoru děličem slože-ným ze dvou sériově zapojených rezistorů $R_1 = R_2 = 100$ kΩ:

$$I = \frac{U_1}{R_1 + R_2} = \frac{4.2 \text{ V}}{200 \text{ k}\Omega} = 21 \text{ μA} \tag{3.5}$$

Toto zatížení je relativně malé, v případě použití akumulátoru s kapacitou 300mAh by tento proud teoreticky vybil akumulátor po 600 dnech. Vypočteme tedy chybu metody: pro plně nabitý akumulátor bude na nezatíženém děliči napětí:

$$U_{2nez} = U_1 \frac{R_2}{R_1 + R_2} = U_{bat} \frac{100 \text{ k}\Omega}{200 \text{ k}\Omega} = 2.1 \ V \tag{3.6}$$

Při zatížení děliče měřicím vstupem bude napětí na výstupu děliče vnitřní odpor R_i A/D převodníku připojen paralelně k odporu R_2 děliče, potom tedy bude

$$U_{2zat} = U_1 \frac{R_2 \| R_i}{R_1 + R_2 \| R_i} = U_{bat} \frac{99.9 \text{ k}\Omega}{199.9 \text{ k}\Omega} = 2.099 \ V \tag{3.7}$$

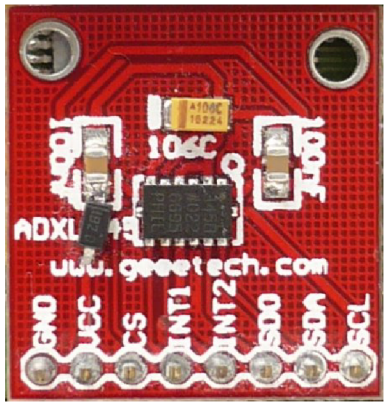
Absolutní chyba metody tedy činí

$$\Delta_m = U_{2nez} - U_{2zat} = 0.01 \ V \tag{3.8}$$

což je hodnota zanedbatelná. Navržený odporový dělič napětí je tedy pro daný účel vyhovující.

3.2.2 MEMS Akcelerometr

V prototypu bude z důvodu komplikovaného ručního osazování LGA pouzdra použit modul s již osazeným akcelerometrem a blokovacími kondenzátory napájení. Na rozhraní modulu (obr. 3.2) jsou vyvedeny všechny signály pro napájení i komunikaci. Modul je připojen k mikrokontroléru SPI sběrnicí (obr. 2.4) (schéma s. 27-C-5).



Obr. 3.2: Modul s osazeným akcelerometrem ADXL345

ADXL345 [2] pracuje vždy v režimu Slave. Vstup SDA(MOSI) slouží pro přenos dat(příkazů) do akcelerometru, výstup SDO slouží pro přenos dat do mikrokontroléru. SCL je taktovací (hodinový) vstup. $\overline{\text{CS}}$ vstup slouží k výběru Slave zařízení na sběrnici SPI (v úrovni LOW). ADXL345 však tento vstup používá také pro zapnutí I2C módu (v úrovni HIGH). Potom v případě, kdy je na sběrnici více SPI zařízení, může nastat situace, že data na sběrnici budou obvodem ADXL345 interpretována jako platný příkaz I2C. Poté by hrozilo nebezpečí, že ADXL345 bude odesílat data na sběrnici a způsobí tím kolizi dat. Proto je v zapojení použito logické hradlo OR, které v případě, že je vstup $\overline{\text{CS}}$ v úrovni HIGH, zablokuje linku proti vstupu nežádoucích dat do akcelerometru.

Výstupy INT1 a INT2 jsou použity pro přenos informace o některé ze specifických událostí, které ADXL345 rozeznává. Tyto signály zpracovává přerušovací systém mikrokontroléru. Signál přerušení INT1 je použit pro probuzení mikrokontroléru z režimu spánku.

3.2.3 Obvody rozhraní

K USART rozhraní mikrokontroléru je křížově (RX/TX) připojen bezdrátový komunikační Bluetooth modul HC-07 [9] (schéma s. 27-E-5). Zapínání a vypínání tohoto modulu je ovládáno výstupem mikrokontroléru PD4. Přepínání režimu (AT mode, Comm mode) je ovládáno výstupem mikrokontroléru PB7. Konfigurace komunikace

je nastavena softwarově pomocí AT příkazů. Aktivitu na bezdrátovém spoji indikuje LED připojená k BT modulu.

Na univerzální I/O piny (konfigurovány budou jako vstupní) PA1, PA2 a PE1, PE0 jsou připojena tlačítka S1,S2 (schéma s. 27-D-4) a 2 přepínače S4 pro ovládání (s. 27-E-4), na I/O piny (výstupy) PD5, PD6, PD7 je připojena RGB LED (3LED1) pro signalizaci stavu (s. 27-D-5).

USB vstup je zapojen dle datasheetu [3] a používán je pouze k nahrávání firmware (s. 27-A-1).

Ve schématu je zakreslen i konektor připojený k JTAG rozhraní mikrokontroléru (s. 27-D-1), umožňující nejen nahrávání firmware, ale především odladování a odstraňování chyb. Tento konektor byl u prototypu připojen vodiči přímo na odpovídající vývody mikrokontroléru.

3.2.4 OLED

Použitý displej obsahuje integrovaný řadič SSD1332 [7] (schéma s. 27-C-7). Zapojení displeje vychází z datasheetu [7]. Přívody napájení displeje je nutno opatřit blokovacími kondenzátory na vstupu všech napájecích napětí. Na výstup IREF je nutno připojit rezistor (R11), jímž bude procházet referenční proud, který určuje proudovou referenci pro budiče segmentů displeje. Doporučený referenční proud je $I_{REF} = 10 \pm 2$ µA. Pro výpočet rezistoru platí následující vztah:

$$R_{REF} = (V_{CC} - V_{SS})/I_{REF} = (12\ \text{V} - 3\ \text{V})/10\ \mu\text{A} = 900\ \text{k}\Omega \tag{3.9}$$

Pro rezistor R11 tedy použijeme nejbližší vyráběnou hodnotu 909 kΩ.

Logické úrovně na vstupech BS1 a BS2 určují konfiguraci rozhraní displeje. Kombinace [0 1] nastaví rozhraní do módu 6800 paralelní, kombinace [1 1] do módu 8080 paralelní a [0 0] do sériového módu. Připojení OLED k mikrokontroléru je realizováno pomocí osmi datových (D0 až D7) a pěti pomocných signálových linek:

- Nastavením vstupu $\overline{\text{CS}}$ na úroveň LOW se povolí komunikace mezi displejem a mikrokontrolérem.
- Vstup $\overline{\text{RESET}}$ slouží pro inicializaci displeje po zapnutí.
- Vstup D/C přepíná řadič mezi režimem DATA a COMMAND. Určuje, zda na linkách D0 až D7 je ovládací příkaz pro řadič, nebo data k zobrazení
- vstup R/W($\overline{\text{WR}}$) spouští zápis dat do paměti displeje. Aktivní je v úrovni LOW
- vstup E($\overline{\text{RD}}$) spouští čtení dat z paměti displeje. Aktivní je v úrovni LOW

3.2.5 FRAM

Přídavná FRAM paměť FM25V20 [8] (schéma s. 27-C-4) je připojena SPI sběrnicí, obr. 2.4, přímo na SPI sběrnici mikrokontroléru. Vstup $\overline{\text{W}}$ slouží pro ochranu proti zápisu do Write Status registru, v zařízení jej nebude využito, proto je připojen na napájecí napětí, tedy vyřazen z funkce. Vstup $\overline{\text{HOLD}}$ slouží k pozastavení právě probíhající operace. Pokud je tento vstup používán, změny logické úrovně musí probíhat když je vstup CLK v úrovni LOW.

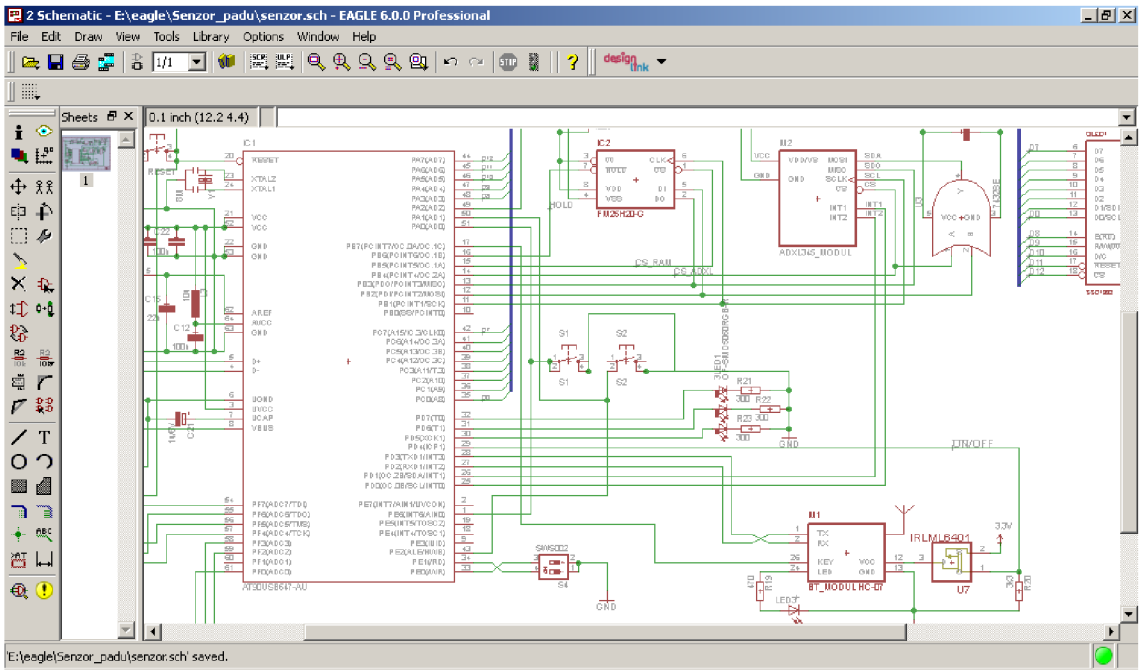
Připojení napájení je opatřeno blokovacím kondenzátorem pro omezení rušivých vlivů při rychlých změnách napájecího proudu.

3.2.6 Uživatelské rozhraní

V prototypu je použito pouze jednoduché uživatelské rozhraní sestávající ze dvou tlačítek, dvojitého DIP přepínače a RGB LED. Popis ovládání je uveden v kapitole 7 - Firmware akcelerometru. Bohužel se nepodařilo během vývoje zprovoznit OLED panel, ačkoliv vynaložené úsilí bylo nemalé. OLED panel byl zkoušen v režimu rozhraní 8080, v režimu 6800, byly zkoušeny různé módy (LSB/MSB first), časování, rychlosti, metody inicializace, vše bohužel bez výsledku. Zakoupeny a vyzkoušeny byly dva kusy displejů z různých výrobních sérií, důvod nefunkčnosti však nebyl ani při dlouhodobém úsilí nalezen. Nicméně zvolená velikost displeje by pro praktické použití nejspíš nebyla zcela vyhovující. Při vývoji zařízení by však přítomnost displeje byla značným přínosem.

3.3 Kreslení schéma v EAGLE

Schéma zapojení je vytvořeno v grafickém editoru EAGLE (Easily Applicable Graphical Layout Editor). Tento grafický editor je navržen pro kreslení elektrotechnických schémat a desek plošných spojů. Návrh desky plošných spojů vychází z el. schématu a použitých pouzder součástek, je interně provázaný se schématem a automaticky je aktualizován při změnách schématu. Editor schémat je možno plně ovládat myší, volbou jednotlivých příkazů (funkcí) z menu a lišt funkčních tlačítek, nebo lze také příkazy zadávat z klávesnice do příkazového řádku. Objekty (součástky, symboly, texty...) se na pracovní plochu umisťují pomocí myši. V režimu kresby schématu postupně přidáváme součástky příkazem ADD z knihoven součástek. Poté ve schématu jednotlivé vývody součástek pomocí spojů NET nebo BUS vzájemně propojíme. Obvykle mají součástky nějakou vlastnost, např. velikost odporu u rezistoru, kterou je potřeba měnit - Edit Value. Také lze měnit pojmenování součástky



Obr. 3.3: EAGLE v režimu kreslení schémat

ve schématu, obvykle jde o číslování součástek podle příslušnosti ke konkrétním funkčním blokům apod.

Pokud danou součástku nemáme v knihovně, lze se poohlédnout po Internetu, zda již někdo knihovnu s danou součástkou vytvořil, nebo lze v editoru knihoven danou součástku vytvořit. Součástka (Device) se skládá ze dvou částí - Symbol (značka) a Package (pouzdro). V definici Device je potom určeno, který vývod Symbolu je připojen na konkrétní vývod Device. Definice Symbolu obsahuje grafickou podobu schématické značky součástky, rozmístění a popis vývodů součástky, definici názvu a hodnoty součástky a další textové informace, např. výrobce, dodavatel, provedení součástky atd... V definici Package je nakresleno v reálných rozměrech pouzdro dané součástky (resp. jeho obrys) a rozmístění, tvar a forma (drátové, smd...) vývodů. Pro součástku lze definovat více pouzder, díky tomu lze v definici Device vytvořit více provedení téže součástky.

Pro kreslení schémat existují nepsaná pravidla a doporučení. Z důvodu přehlednosti a čitelnosti se všechny čáry spojů kreslí horizontálně a svisle, šikmo pouze výjimečně, nebo pokud daná čára má zvýraznit nějakou konkrétní skutečnost, např. překřížení signálových vodičů. Křížení spojů se snažíme co nejvíce vyhnout, podobně by žádný spoj neměl vést přes jakýkoliv textový popisek či symbol. Při kreslení schémat (i symbolů součástek) se používá funkce mřížky v grafickém editoru. Standardně se používá krok mřížky 0.1". Je to z toho důvodu, že všechny symboly součástek jsou

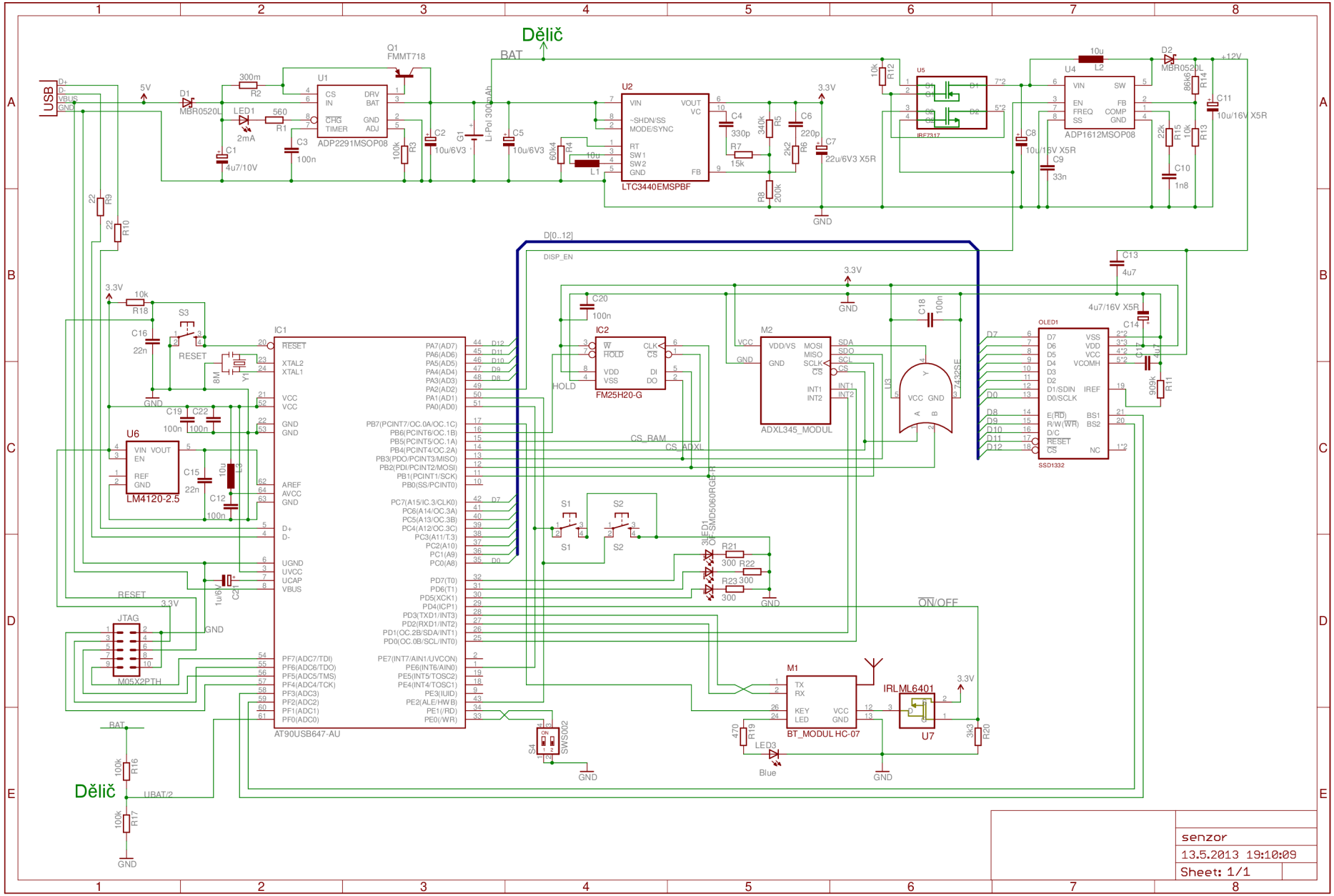
obvykle nakresleny v tomto rastru a při použití jiného rastru by bylo komplikované, až téměř nerealizovatelné propojení jednotlivých součástek.

Po nakreslení celého schématu lze pomocí nástroje Electrical Rule Check prověřit správnost zapojení na určité základní úrovni. Např. zda je ke všem napájecím pinům integrovaných obvodů přivedeno napětí, zda nemáme uzemněný některý ze zdrojů, zda se na výstupní pin součástky nesnažíme „vnutit“ napájecí napětí apod.

Editor schémat nabízí samozřejmě mnohem širší paletu funkcí a příkazů, než bylo zde zmíněno, cílem byl pouze letmý náhled na funkce systému a základní způsob práce. Zájemce o další studium lze odkázat na www stránky výrobce [10].

Pokud jsme s podobou schématu spokojeni, můžeme přejít k tvorbě návrhu desky plošných spojů, viz kap. 4.3.

Na následující straně je kompletní schéma navrženého akcelerometru.


Dělič
BAT
USB
5V
D1
MBR0520L
LED1
2mA
R1
560
R2
300m
C1
4u7/10V
C2
10u/6V3
C3
100n
U1
ADP2291MSOP08
Q1
FMMT718
G1
Li-Pol 300mAh
C5
10u/6V3
R3
100k
R4
60k4
L1
10u
U2
LTC3440EMSPBF
C4
330p
R5
340k
R6
2k2
C6
220p
R7
15k
C7
22u/6V3 X5R
R8
200k
3.3V
GND
U5
IRF7317
U4
ADP1612MSOP08
L2
10u
D2
MBR0520L
R12
10k
R14
86k6
R15
22k
R13
10k
C8
10u/15V X5R
C9
33n
C10
1n8
C11
10u/16V X5R
+12V
R9
22
R10
22
D[0..12]
DISP_EN
IC1
AT90USB647-AU
RESET
XTAL2
XTAL1
Y1
8M
C16
22n
C19
100n
C22
100n
U6
LM4120-2.5
C15
22n
L3
10u
C12
100n
C21
1u/6V
S3
R18
10k
JTAG
M05X2PTH
IC2
FM25H20-G
HOLD
C20
100n
M2
ADXL345_MODUL
CS_RAM
CS_ADXL
C18
100n
U3
74325F
C13
4u7
C14
4u7/16V X5R
C17
4u7
R11
909k
OLED1
SSD1332
S1
S2
3LED1
OK-SMD5060RGB-PR
R21
300
R22
300
R23
300
ON/OFF
M1
BT_MODUL HC-07
IRLML6401
U7
R19
470
LED3
Blue
R20
3k3
S4
SWS002
R16
100k
R17
100k
UBAT/2
Dělič
senzor
13.5.2013 19:10:09
Sheet: 1/1

4 NÁVRH DESKY PLOŠNÝCH SPOJŮ

Návrh desky plošných spojů spočívá ve zvolení vhodného rozložení součástek a správném rozmístění a dimenzování a navržení spojů. Desku jsem navrhl jako oboustrannou, s prokovenými otvory, o rozměrech 63x45 mm. Základní pravidla pro návrh obrazce plošných spojů určuje jednak povaha samotného obvodu, jednak použitá technologie výroby:

- spoje mají být co nejkratší a nejpřímější, 90° zlomy doplnit o úsek pod úhlem 45°
- silové spoje (napájecí) co nejširší, zejména zemnicí spoj GND, min. 0.032”
- signálové spoje minimální šíře 0.012”
- izolační vzdálenost mezi spoji min. 0.25 mm
- prokovené otvory pro přechod mezi horní a spodní vrstvou min. 0.6 mm
- vodiče jedné sběrnice by měly mít přibližně stejnou délku a trasu
- pro přehlednost je vhodné spoje vést rovnoběžně s obvodem desky, příp. pod úhlem 45°

4.1 Napájecí část

Součástky nabíjecího obvodu nemají zvláštní požadavky pro umístění, jedná se o lineární obvod, pouze je vhodné dodržet co nejkratší a širší silové spoje, a součástky umístit do rozumné vzdálenosti od řídicího obvodu ADP2291. Součástky hlavního napájecího zdroje 3.3 V je nutné umístit dle doporučení výrobce, jedná se o spínaný zdroj. Zejména cívku L1, vstupní a výstupní kondenzátory C5 a C7 a napěťový dělič zpětné vazby složený z rezistorů R5 a R8. Zdroj pracuje na frekvenci 300kHz až 2MHz, pro omezení vf rušení je nutné vytvořit v jeho okolí stínění. To je možno provést ponecháním měděné vrstvy na DPS v okolí kritických součástek, zejména cívky L1 a silových vodičů. Ponechaná plocha se propojí se zemnicím vodičem. Podobné nároky má na rozmístění součástek, provedení vodičů a stínění také pomocný napájecí zdroj 12 V.

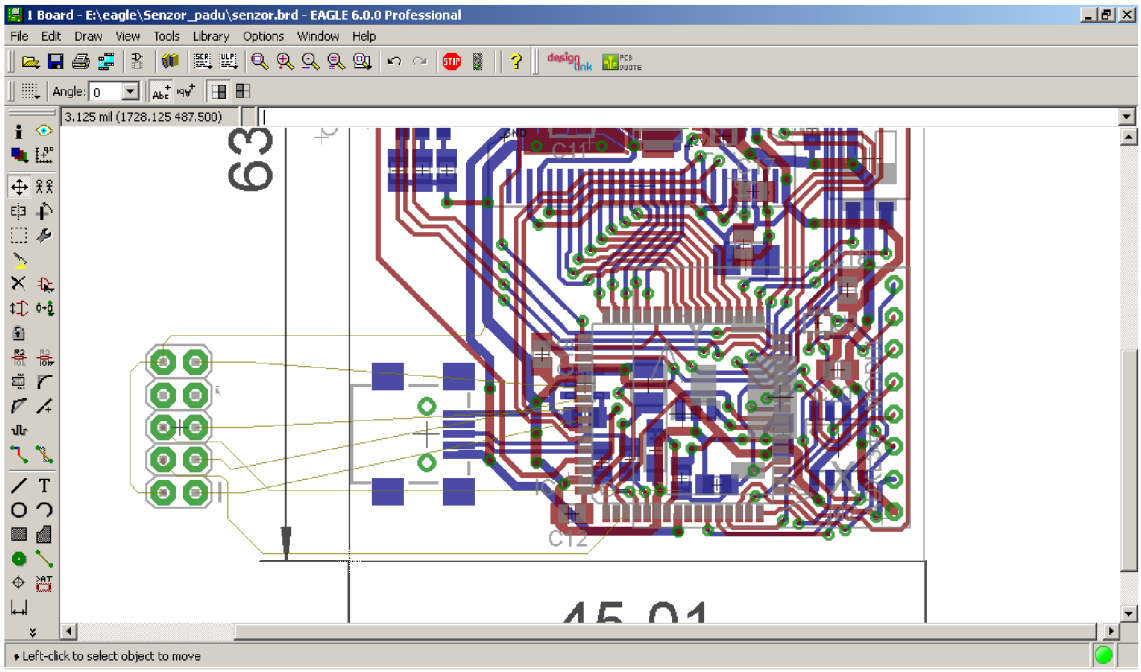
4.2 Digitální část

Digitální část systému nemá na rozmístění součástek a způsob vedení vodičů nějaké zvláštní požadavky. Blokovací kondenzátory u jednotlivých integrovaných obvodů je nutno umístit co nejblíže jejich pouzdrům. Podobně filtrační tlumivku L3 je vhodné umístit co nejblíže k napájecímu přívodu A/D převodníku. U vodičů SPI sběrnice je vhodné dodržet jejich srovnatelnou délku a souběžné vedení na DPS.

Součástky ovládací části (LED, displej, tlačítka a přepínače) jsem umístil na stejnou stranu desky pro pohodlnou dostupnost všech ovládacích prvků.

4.3 Návrh desky v EAGLE

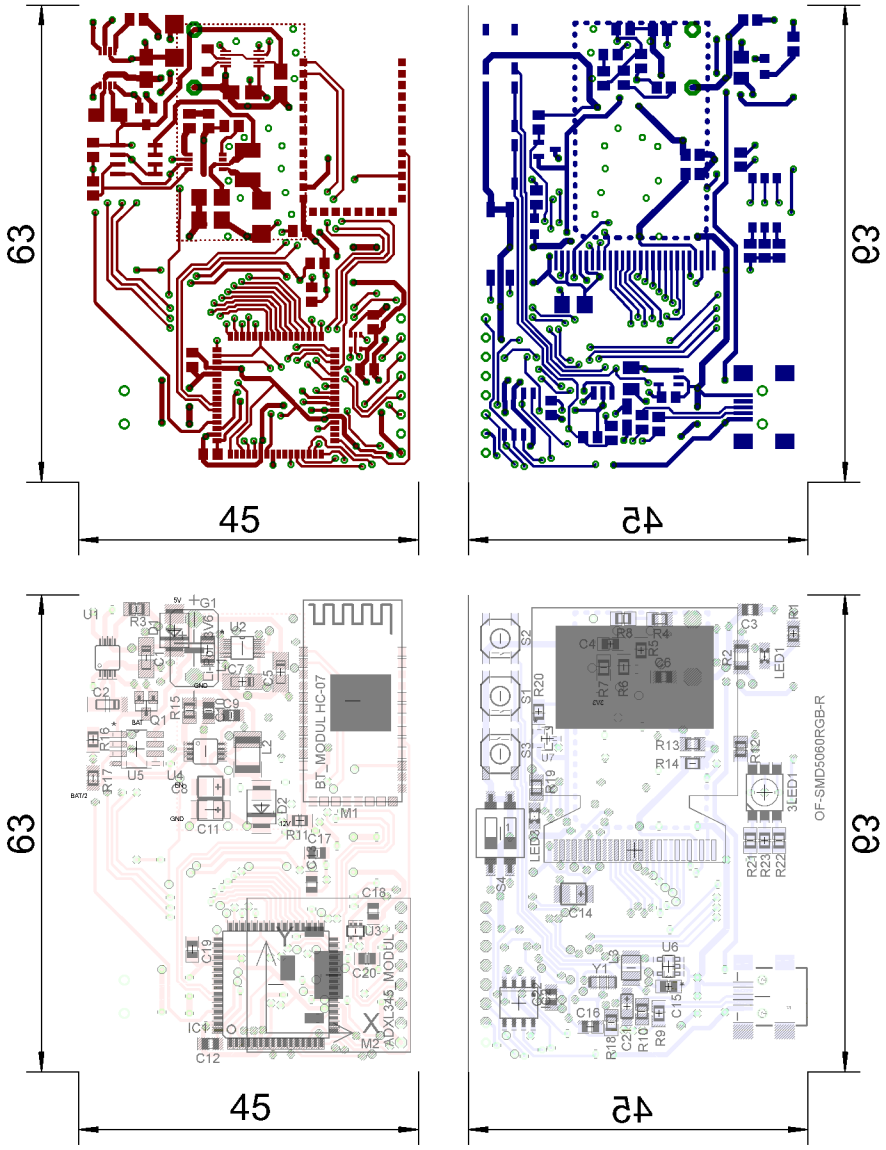
Návrh obrazce plošného spoje vychází ze schématu zapojení. V prostředí editoru schémat EAGLE zvolíme příkaz Switch to board a tím se přepneme do režimu kreslení DPS. Při vytváření DPS ze schématu EAGLE nastaví nějakou výchozí velikost DPS (lze ji později upravit) a všechny součástky (obrysy pouzder s vývody) „rozloží“ do levého dolního rohu editačního okna. Vývody součástek jsou propojeny podle schématu „vzdušnými“ spoji (Airwires). Po rozmístění součástek na správné pozice a na správnou stranu desky (horní/spodní) pomocí funkce Route ručně převádíme spoje Airwires na konkrétní vodivé cesty. Po převedení části nebo všech spojů lze provést kontrolu Design Rules Check, která prověří správnost a bezchybnost návrhu DPS dle definovaných pravidel (izolační vzdálenosti, přesahy pouzder součástek, počet zbývajících Airwires atd.).



Obr. 4.1: EAGLE v režimu kreslení DPS

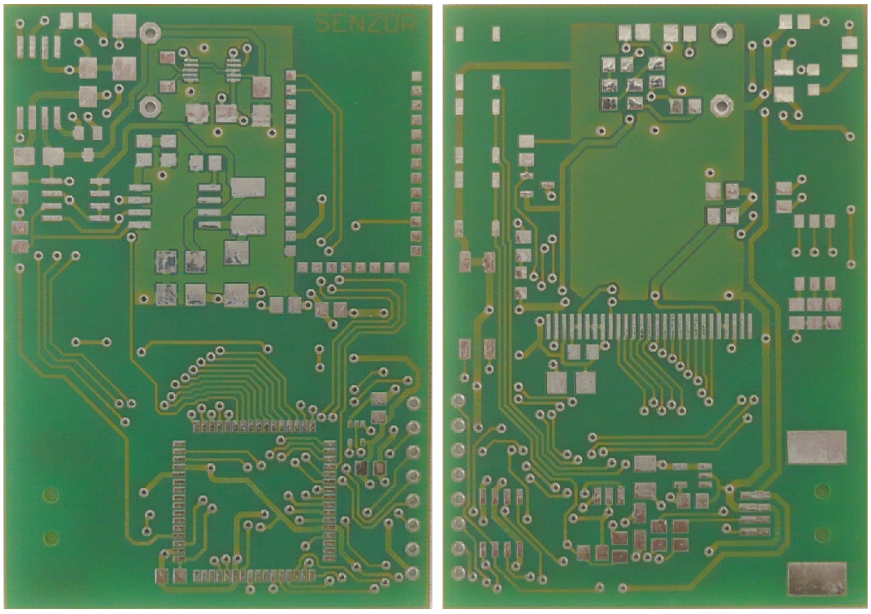
EAGLE obsahuje CAM procesor, který generuje data pro výrobu DPS, lze exportovat další výstupy v podobě obrázku DPS, výtisku na tiskárně, seznam součástek apod.

Na obr. 4.2 je obrazec plošných spojů tak, jak byl během vývoje upraven do finální podoby.



Obr. 4.2: Deska plošných spojů a rozmístění komponent

Na obr. 4.3 je fotografie prototypové verze desky plošných spojů. Deska byla vyrobena ve firmě AJ Technology, s.r.o. včetně nepájivé masky, s prokovenými otvory a pocínovanými pájecími body.



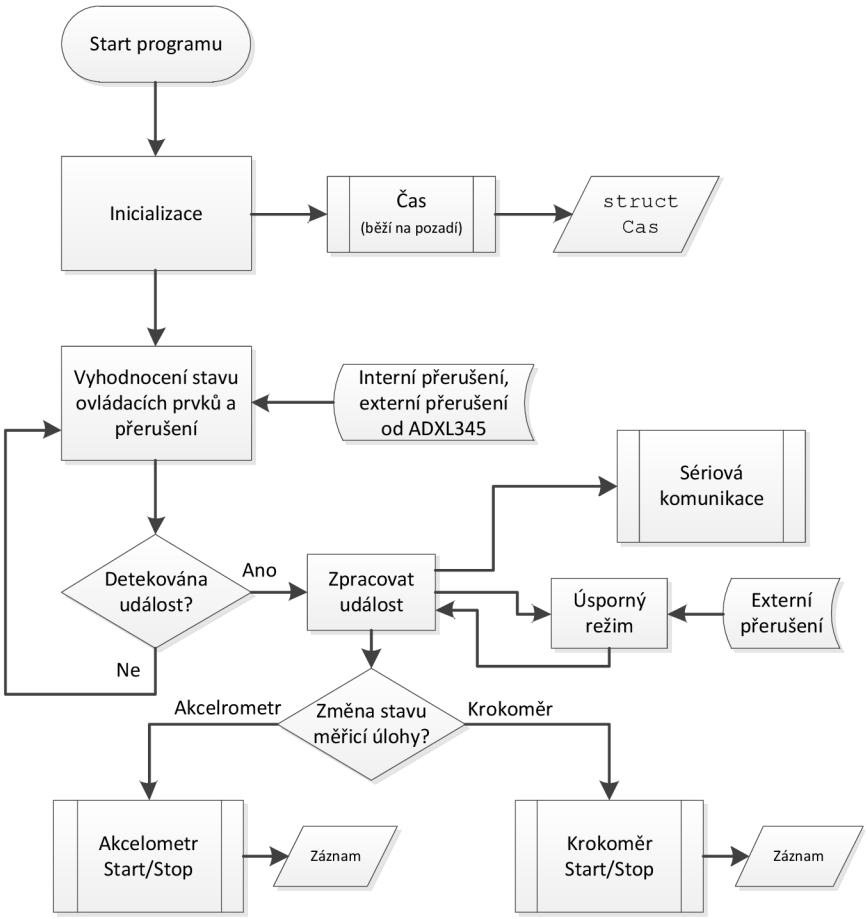
Obr. 4.3: Fotografie prototypové verze DPS

5 DEMONSTRAČNÍ SW VYBAVENÍ

Demonstrační software bude sestávat ze dvou částí. První část bude firmware samotného měřicího zařízení. Druhá část bude software pro PC ke komunikaci s modulem a získávání informací o zaznamenaných měřeních.

Firmware bude založen na klasickém modelu úvodní inicializace a nekonečné smyčky, kdy bude při každém průchodu smyčky testován stav ovládacích prvků a příznaků přerušení, a dle detekované akce či změny se spustí provádění konkrétní funkce. Přerušovací systém procesoru nebude použit k provádění složitějších operací, v obslužné rutině přerušení se provedou jen jednoduché a krátkodobé operace (změna času, nastavení příznaku v globální proměnné, restart časovače), funkce obsluhy události budou volány až v rámci hlavní smyčky dle nastavených příznaků.

Akcelerometr bude vybaven firmwarem pro dvě měřicí úlohy.



Obr. 5.1: Diagram hlavních funkcí firmware

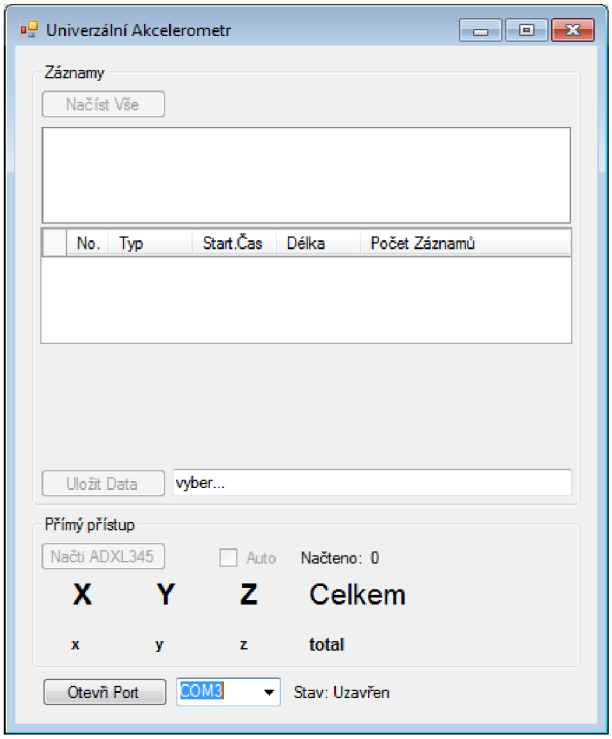
První úloha - Akcelerometr, bude měření zrychlení s přepínatelným rozsahem a

vzorkovacím kmitočtem. Tento základní mód činnosti bude zaznamenávat hodnoty zrychlení ve všech osách ve volitelných časových intervalech. Po ukončení měření bude možné naměřená data přenést do PC k uložení a následnému zpracování či vyhodnocení.

Druhá úloha bude Krokoměr - mikrokontrolér bude filtrovat a vyhodnocovat data z akcelerometru při chůzi/běhu a zaznamenávat počet kroků.

Dále bude ve firmware využito integrovaných funkcí samotného senzoru, na Aktivitu/Inaktivitu bude navázáno automatické přepínání do úsporného režimu, oživení zpět do aktivního režimu bude provedeno dvojitým klepnutím (Double Tap) na zařízení. Na pozadí poběží hodiny „reálného“ času, senzor bude reagovat na požadavky zvenčí přes Bluetooth rozhraní, reagovat na vstupy uživatele pomocí tlačítek a informovat o stavu pomocí RGB LED.

Vzhledem k tomu, že se v prototypu bohužel nepodařilo zprovoznit displej, všechny operace vyžadující složitější uživatelské rozhraní budou prováděny v demonstrační aplikaci pro Windows.



Obr. 5.2: Návrh aplikace pro Windows

6 REALIZACE HARDWARE SENZORU

6.1 Nákup součástek, náklady

Součástky a jednotlivé moduly pro celé zařízení byly postupně nakoupeny přes internet v několika e-shopech, některé byly objednány přímo u výrobců. Pasivní prvky (rezistory, kondenzátory, tlumivky) a základní polovodiče (LED, diody, tranzistory) jsem nakoupil u evropské společnosti TME, http://www.tme.eu. Tato firma se zabývá prodejem široké palety elektronických součástek i nářadí a dalších doplňků pro elektroniku. Kromě dost velkých minimálních množství pasivních součástek (obvykle min. 100 ks) bych tuto firmu doporučil jako spolehlivého a rychlého dodavatele. Naproti tomu česká firma s dosti podobným názvem se mi vůbec neosvědčila, dodávka několika málo komponent, které byly označeny v internetovém obchodě „skladem“ zabrala po urgencích více než tři týdny.

Tab. 6.1: Náklady na výrobu prototypu

Komponent	Cena [Kč]	Komponent	Cena [Kč]
Modul ADXL345	230	SMD rezistory	5
AT90USB128	260	SMD kondenzátory	25
FRAM FM25H20	630	Tlumivky	18
Bluetooth	160	Aku Li-Pol 300 mAh	165
ADP2291	70	Tlačítka, DIP spínač	65
LTC3440	78	Konektor USB	10
ADP1612	38	DPS	150
OLED panel	160	Ostatní náklady	30
LED	10		
Tranzistory, diody	40		
Celkem aktivní komp.	1676	Celkem pasiv. komp.	468
Celkem			**2144**

Mikrokontrolér jsem poptal u firmy Atmel jako vzorek, dorazil během tří dnů z Filipín. Podobně integrované obvody zdrojů jsem poptal přímo u Analog Devices, resp. Linear Technology jako vzorky, opět jsem je během tří/čtyř dnů měl na stole, obvykle ze zámoří. FRAM paměť jsem taktéž objednal jako vzorek od výrobce Ramtron Inc., se stejně rychlou odezvou. Bluetooth modul, modul akcelerometru a OLED jsem objednal u malé české firmy VILANTERIO GROUP s.r.o., zabývající

se prodejem speciálních součástek pro vývoj elektroniky (různé moduly senzorů, procesorů, Arduino, součástky pro malé roboty apod.). Aktuální nabídku firmy lze nalézt na aukčním serveru http://www.Aukro.cz pod názvem prodejce „saxik1“. V tabulce 6.1 uvádím přibližné náklady na pořízení prototypu v případě nákupu všech součástek, nikoliv pro případ získání několika kusů vzorků od výrobce zdarma.

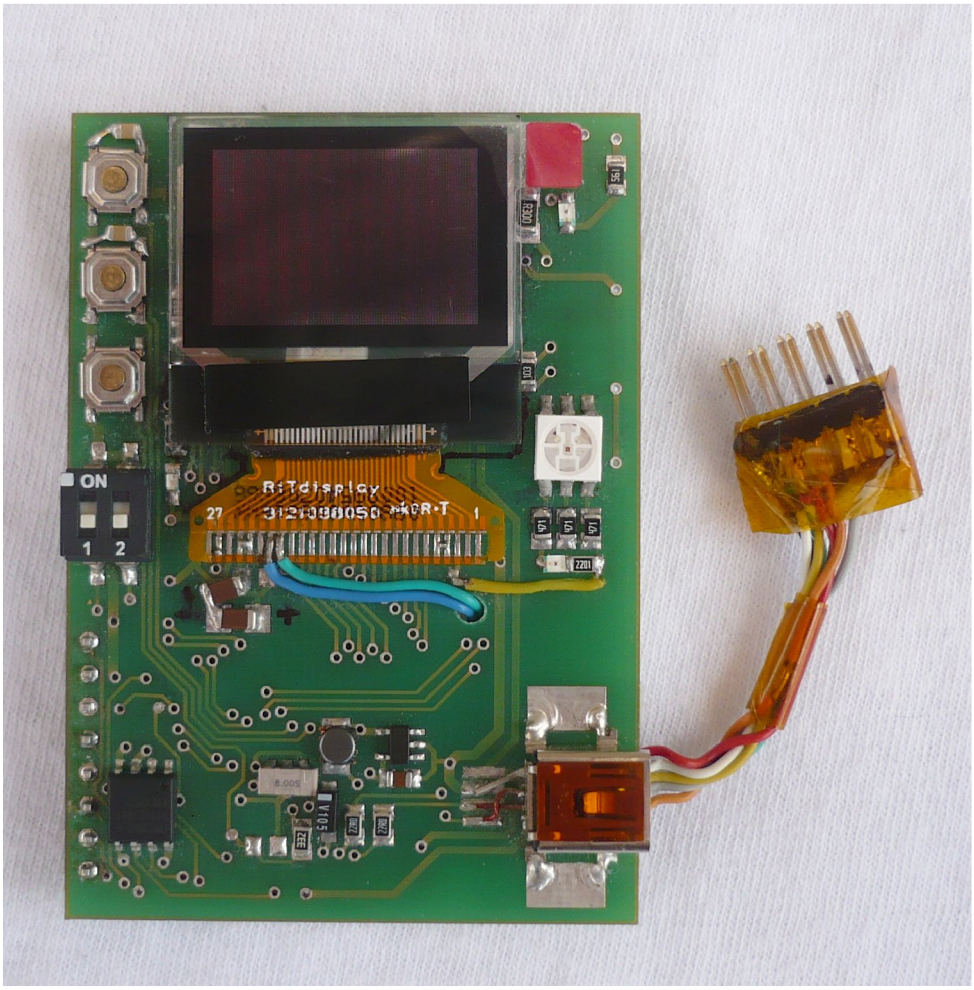
Celková cena prototypu je celkem vysoká, předpokládal jsem nižší náklady. Cestou k úsporám by bylo použití levnějšího, ale dostatečně výkonného mikrokontroléru např. vývodově kompatibilní AT90USB646 stojí kolem 160 Kč, dále by bylo možné použít standardní levnou FLASH paměť, cena kolem 20 Kč, ušetřit lze i na senzoru, samotný obvod ADXL345 v LGA pouzdru stojí kolem 130 Kč. Všechny tyto změny by měly za následek větší či menší omezení možností zařízení, nicméně pro většinu potenciálních aplikací by byly parametry stále dostačující.

6.2 Osazení DPS součástkami

Všechny součástky a moduly jsem na desku připájel ruční mikropáječkou. Postupoval jsem dle schématu, kdy jsem postupně osazoval a oživoval jednotlivé části zapojení. Jako první jsem osadil a otestoval modul nabíjení Li-Pol článku. Po otestování, změření nabíjecího proudu a dokončení jednoho nabíjecího cyklu jsem pokračoval dále spínanými zdroji. Tak jako nabíjecí obvod oba pracovaly na první zapojení, jen zdroj 3.3V vykazoval napětí 3.1V. Provedl jsem korekci změnou rezistoru R5 na 390kΩ (z pův. 340kΩ) ve zpětnovazebném děliči U2, poté již napětí na výstupu mělo předepsaných 3.3V. Výstupní napětí zdroje pro budiče OLED bylo v požadované velikosti 12V ihned po prvním zapnutí. Po oživení zdrojů jsem na desku připájel mikrokontrolér, FRAM, a všechny ostatní komponenty. Po osazení všech součástek, viz obr. 6.1,6.2 a důkladné kontrole jsem začal s realizací firmware.

Po implementaci základních obslužných funkcí hardware jsem orientačně změřil proudovou spotřebu v různých režimech systému. Tabulka 6.2 uvádí naměřené hodnoty a odpovídající teoretickou dobu provozu na baterii.

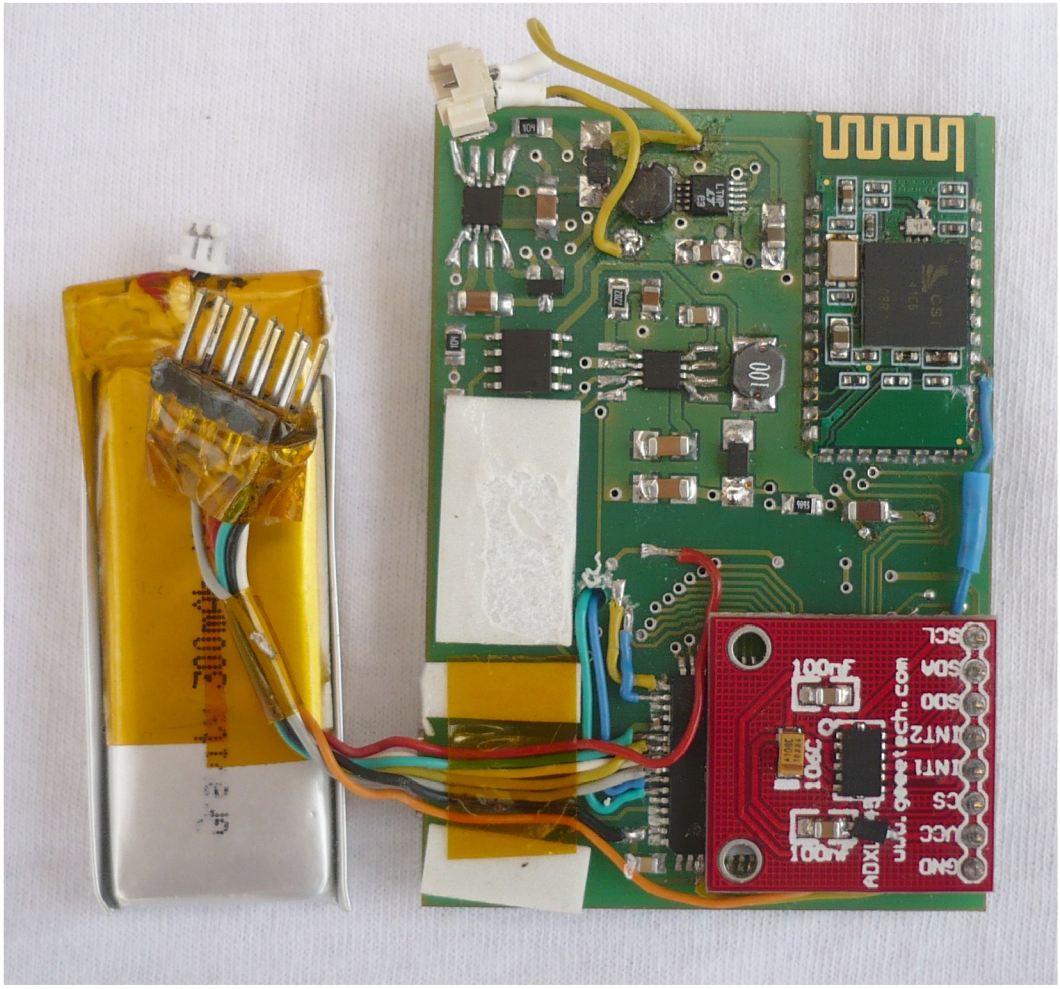
Proud odebíraný z napájecího Li-Pol článku jsem měřil pro orientační stanovení provozní doby. K měření jsem použil multimetr FK Technics model FK8250 nastavený na rozsah 200 mA, u něhož výrobce udává přesnost ± 1,2% z naměřené hodnoty ± 2 digit. Změřená spotřeba zařízení v režimu měření je o 9 mA větší, než byl předpoklad. Přesto je doba provozu v režimu měření pro dané účely dostatečně dlouhá. Také ve sleep režimu má zařízení řádově větší spotřebu, než bylo odhadováno, což je způsobeno především tím, že mikrokontrolér není přepnut do režimu s nejnižší spotřebou, taktéž senzor je i ve Sleep režimu zařízení stále v aktivním měřicím módu.



Obr. 6.1: Osazená a oživená DPS - přední strana

Tab. 6.2: Proudová spotřeba v různých režimech činnosti

Režim činnosti	Spotřeba [mA]	Doba provozu [hod]
Čekání na událost	16	18,8
Měření	16	18,8
Zapnutý Bluetooth - bez spojení	95	3,2
Bluetooth - spojení navázáno	55	5,5
Sleep režim	2,5	120



Obr. 6.2: Osazená a oživená DPS - zadní strana

6.2.1 Chyby v návrhu DPS

Při realizaci prototypu byly nalezeny chyby, jichž jsem se dopustil během návrhu zapojení a návrhu desky plošných spojů. Z toho vyplývají některé „nelogičnosti“ v návrhu. Např. oblasti, které zabírají napájecí zdroj pro displej a nabíjecí obvod jsou jen řídce osazené součástkami, je to z důvodu chybného(většího) pouzdra použitého u návrhu prototypové desky. Signály pro nastavení displeje byly dotaženy v prototypu pomocí metalických vodičů, pro účely snahy uvést displej do chodu. Spoj vedoucí od Bluetooth modulu okrajem desky k mikrokontroléru taktéž nebyl v prototypu plánován, datasheet neuváděl funkci tohoto vývodu a až experimentem bylo zjištěno, že je nutno pro možnost komunikace tento spoj použít. USB konektor jsem nakonec použil jiný, menší, a v prototypu jsem ho připojil krátkými drátovými propojkami. Zanesl jsem tedy změny v zapojení i do obrazce desky plošných spojů tak, aby se co nejvíce podobal prototypu, a zároveň aby odpovídal reálnému zapo-

jení. Vzhledem k tomu, že množství změn u prototypového kusu nebylo nějak vysoké jsem se rozhodl pouze desku upravit, nikoliv ji celou překreslit znovu. Během vývoje firmware se ukázalo nutností mít vyveden konektor pro ladění aplikace přímo v procesoru. Původně jsem plánoval firmware nahrávat do mikrokontroléru přes USB rozhraní a k ladění používat simulátor obvodu v Atmel Studiu. Ale zjistil jsem, že Atmel Studio simulátor zvoleného mikrokontroléru neobsahuje. Chvíli jsem používal k simulaci simulátor mikrokontroléru ATmega 128, který má podobnou architekturu, jako AT90USB1287. Nicméně se po čase ukázalo, že bude nutné použít jiný přístup. Zapůjčil jsem si tedy od vedoucího práce doc. Ing. Zdeňka Bradáče Ph.D. programátor a debugger AVR JTAGICE3, a na desku připojil JTAG konektor pomocí metalických vodičů.

6.2.2 3D tisk obalu

Při vývoji firmware vyvstala potřeba umístit zařízení do vhodného obalu pro testovací měření a komfortnější obsluhu. Vytvořil jsem proto ve 3D software Rhinoceros model osazené DPS, a na míru navrhl vhodný obal. Poté jsem tento obal vytiskl na 3D tiskárně ZCorp ZPrinter 650. Tiskárna ZPrinter 650 tiskne do sádrokompozitního prášku do postupně nanášených jednotlivých vrstev o tloušťce 0.1 mm pomocí CMYK barevných pojiv. Po vytvrzení pojiva je model vyjmut z tiskárny a odstraněn přebytečný prášek. Poté je vhodné vytvořený díl napustit infiltrantem pro zvýšení mechanické pevnosti. Nejčastěji je k tomu používáno nízkoviskózní kyanoakrylátové (tzv. vteřinové) lepidlo. 3D tisk prototypového obalu trval přibližně 45 minut, po technologické 1,5 hod. přestávce pro vytvrzení pojiva bylo možno model vyjmout, a dokončit jej začištěním a infiltrací kyanoakrylátem. Vytištěn byl ve firmě Schiebel s.r.o., které bych tímto rád poděkoval za podporu. Fotografie hotového prototypu měřiče a obalu je na obr. 6.3.

Obr. 6.3: Prototyp měřiče v obalu zhotoveném na 3D tiskárně

7 FIRMWARE AKCELEROMETRU

Jak bylo předestřeno v kapitole 5, firmware pracuje v nekonečné smyčce v níž se vyhodnocuje změna stavu ovládacích tlačítek či příznak přerušení. Hlavní smyčce předchází inicializační rutina celého akcelerometru.

V následujících oddílech bude popsán průběh algoritmu hlavní smyčky a postupně rozvedeny detaily jednotlivých akcí navázaných na konkrétní události. Celý firmware jsem napsal v jazyce C ve vývojovém prostředí výrobce mikrokontroléru Atmel Studio v.6.0. Hlavní modul celého projektu se jmenuje *Akcelerometr.c*. Do tohoto hlavního modulu jsou připojeny jednotlivé moduly obsahující definice funkcí a proměnných dané konkrétní části hardware nebo firmware a definice maker preprocesoru.

V hlavním modulu jsou deklarovány dvě globální proměnné, obecně jsem se však používání globálních proměnných snažil vyhnout. Pro komunikaci mezi rutinami přerušení a ostatními funkcemi programu je však použití globální proměnné jediným rozumným způsobem přenosu informací.

První globální proměnná `volatile RTC Cas;` nese po celou dobu chodu programu interní čas, je to strukturní proměnná typu RTC `typedef struct { uint16_t msec; uint8_t sec, min, hour, day;} RTC;`. Tato proměnná je každé 2 ms pomocí časovače `TIMER0` aktualizována ve funkci obsluhy přerušení `ISR(TIMER0_OVF_vect);`. Počáteční inicializace proměnné `Cas` je provedena funkcí `RTC_Init` definované v modulu *time.c(.h)*, v tomto modulu jsou definovány také všechny další časové funkce. Funkce `void RTC_Init(void);` načte poslední známý časový údaj z paměti FRAM, který je zde ukládán vždy při ukončení měření i dalších významnějších událostech a nastaví a spustí časovač `TIMER0`. Hodnota proměnné `Cas` je používána pouze pro časové odlišení jednotlivých záznamů. Časový údaj neslouží jako skutečné hodiny reálného času, je používán především pro výpočet délky doby spuštěného měření. V úsporném režimu se totiž funkce časovače `TIMER0` zastaví. Funkce `void RTC_Calculate(volatile RTC *Result, volatile RTC *StartTime, volatile RTC *EndTime);` vypočte časový rozdíl mezi dvěma zadanými hodnotami proměnných typu `RTC;`.

Druhá globální proměnná `volatile uint8_t ISR_Vector;` je použita ke komunikaci mezi přerušovacím systémem mikrokontroléru a hlavním programem. V obslužných funkcích vyvolaných na základě přerušení se provádí pouze nastavení příznaků v této globální proměnné a případně nenáročné operace, jako např. resetování časovače. Veškeré komplikovanější operace vyžadující delší výpočetní čas, případně volání dalších funkcí se provádí až voláním z hlavního programu. Toto schéma jsem zvolil z důvodu paměťové a časové náročnosti funkčního volání během přerušení, a také proto, že v době, kdy procesor zpracovává jedno přerušení nemůže reago-

vat na další přerušení, a v případě odeznění žádosti o další přerušení v době, kdy je procesor v režimu obsluhy přerušení, by se tato žádost vůbec nezaznamenala. Proměnná `ISR_Vector` je pomocí maker preprocesoru definovaných v hlavičkovém souboru *hw_defs.h* nastavována či testována po jednotlivých bitech. Je zde tedy 8 možných sledovaných příznaků přerušení, v aktuální verzi firmware je využito prvních 6 bitů. Bity 0 a 1 jsou nastaveny externími přerušeními od čidla ADXL345, bit 2 nastavuje rutina přerušení od časovače při obecném měření zrychlení, bit 3 nastavuje rutina přerušení od časovače krokoměru, bit 4 nastavuje rutina přerušení USARTu a konečně bit 5 nastavuje rutina přerušení spuštěná ukončením převodu napětí A/D převodníkem.

Hlavní funkce `int main(void)` začíná deklarací a definicí proměnných pro obsluhu jednotlivých úloh. Jedná se o proměnné ke sledování aktivity přepínačů a tlačítek, proměnné pro vyhodnocení délky stisknutí tlačítek, proměnné pro ukládání informací o aktuálně spuštěné úloze a další pomocné proměnné.

7.1 Inicializace hardwaru

Vlastní program měřiče začíná inicializací hardware funkcí `HW_Init();`. Tato funkce deklarovaná v hlavičkovém souboru *hw_defs.h* nastaví jednotlivé porty mikrokontroléru jako vstupní či výstupní, zapne pull up rezistory na příslušných pinech, nastaví vstupy externích přerušení od čidla ADXL345, vypne TWI rozhraní a blikne RGB diodami pro identifikaci zapnutí, resp. resetu. V témže hlavičkovém souboru jsou nadefinována makra zpřehledňující a zjednodušující práci s jednotlivými zařízeními, připojenými přímo na konkrétní piny mikrokontroléru. Tak např. použití makra definovaného `#define BT_POWER BT_PORT &= ~BT_PWR` zapne Bluetooth modul (nastaví příslušný výstupní pin na log. `0`. Komplementární definice k `BT_POWER` je `#define BT_NOPOWER BT_PORT |= BT_PWR`, nastavuje log. `1`. V tomto makru jsou použity další dvě definice a to `#define BT_PORT PORTD` a `#define BT_PWR (1<<PD4)`. `PORTD` a `PD4` dále odkazují do hlavičkového souboru s definicemi I/O rozhraní konkrétního mikrokontroléru. Obdobným způsobem jsou nadefinována makra pro obsluhu ostatních hw částí. V tomto hlavičkovém souboru jsou také nadefinovány konstanty označující jednotlivé implementované měřicí úlohy a dále makra k manipulaci a testování pomocné globální proměnné příznaků přerušení `ISR_Vector`. Dále jsou v tomto hlavičkovém souboru definovány společné datové typy a některé funkce blízké hardwaru. Detaily těchto funkcí a datových typů budou popsány dále v příslušných částech textu.

Po inicializaci hardwaru mikrokontroléru je spuštěna inicializace SPI rozhraní, hodinový kmitočet je nastaven na 1/4 základního kmitočtu oscilátoru, tedy na

2MHz. SPI rozhraní obsluhuje jediná funkce `uint8_t spi( uint8_t data)`, která uloží jeden znak do registru SPI rozhraní SPDR a čeká, až mikrokontrolér odešle pomocí SPI řadiče data připojenému Slave zařízení a zároveň přijme vysílaná data zpět do registru SPDR, poté jsou tato data (znak) vrácena jako výsledek funkce.

Dále následuje inicializace hodin „reálného“ času. Při inicializaci hodin je načten poslední známý čas uložený v paměti FRAM od adresy 0x06. Poté je inicializováno USART rozhraní, nastavena rychlost a další parametry sériové linky.

Jako poslední je inicializován senzor ADXL345. Po počátečním nastavení rozsahu, citlivosti, rychlosti vzorkování a nastavení externích přerušení je přepnut do měřicího módu.

7.1.1 Uživatelské rozhraní

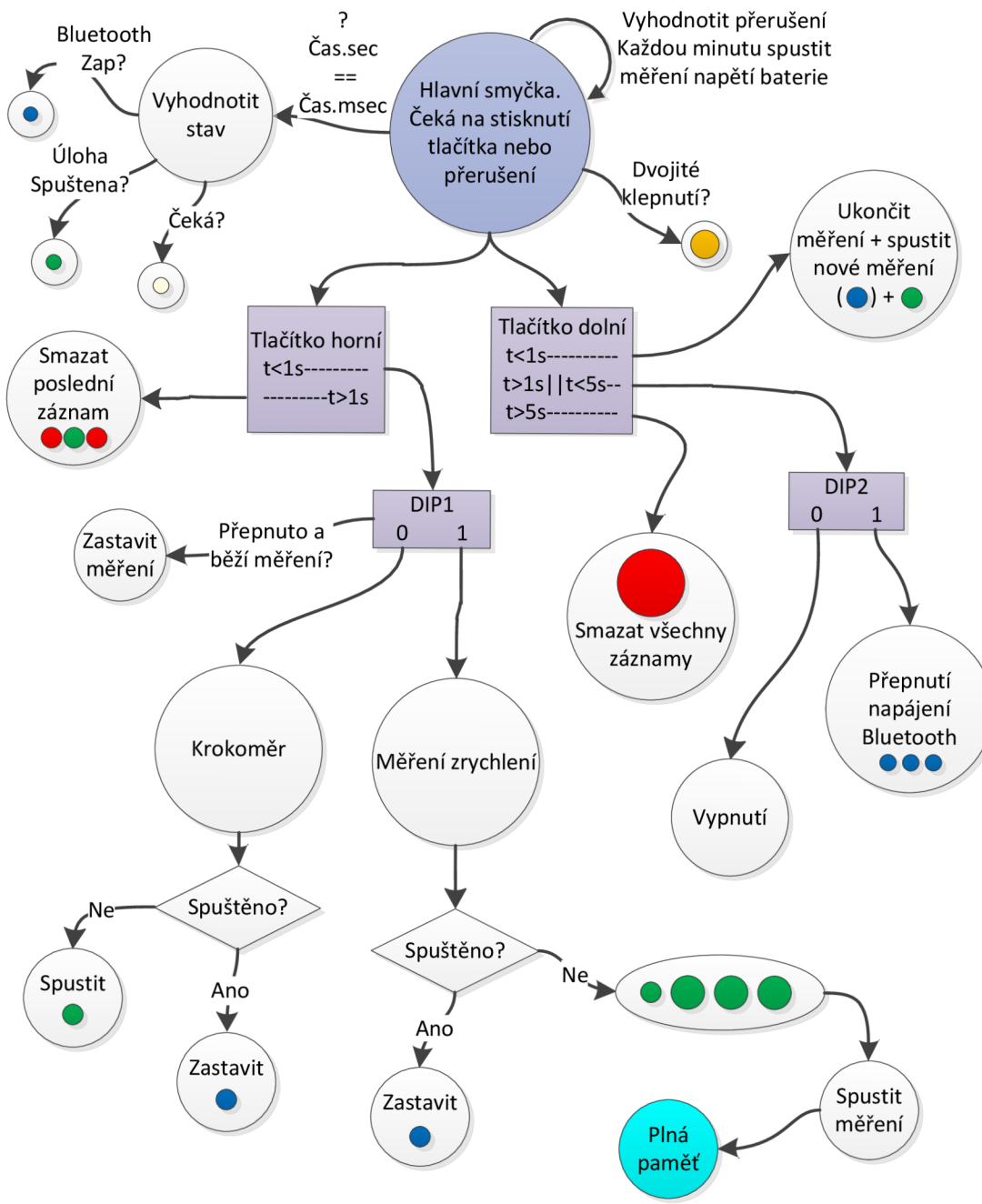
Po skončení inicializace následuje zjištění konfigurace ovládacích DIP přepínačů. Přepínač DIP1 určuje režim měření - tedy tlačítky ovládanou úlohu. Přepínač DIP2 určuje, zda se bude dlouhým stiskem spodního tlačítka vypínat celý měřič, nebo pouze přepínat napájení Bluetooth modulu. Po načtení stavu konfiguračních přepínačů je spuštěna hlavní smyčka programu.

Na začátku hlavní smyčky je zjištěn stav proměnné `uint8_t ISR_Vector=0;` a případně je spuštěna funkce navázaná na daný příznak přerušení. Po zjištění, zda je aktuální sekunda hodin reálného času shodná s aktuální milisekundou (což nastává přibližně každé dvě sekundy) je provedeno informační bliknutí. Toto bliknutí jednak signalizuje, zda je měřič v aktivním režimu, a také konkrétně v jakém stavu se aktuálně nachází. Pokud je aktivní jedna z měřicích úloh a je tedy prováděno měření, je provedeno zelené bliknutí. Pokud je zapnutý Bluetooth a neměří se, blikání je modré. A pokud není ani měření, ani Bluetooth aktivní, blikání je bílé(R+G+B).

Přibližně každou minutu je spuštěno měření napětí baterie, v případě příliš nízkého napětí dojde k automatickému vypnutí měřiče.

Při každém průchodu hlavní smyčkou je zjištěn stav konfiguračních přepínačů, při změně nastavení přepínače měřené úlohy je v případě spuštěné úlohy tato zastavena. Po zjištění konfigurace je prověřen stav ovládacích tlačítek. Pokud je zjištěno, že bylo některé z nich stisknuto, je zaznamenán čas stisknutí, a po uvolnění tlačítka je spočten časový interval stisknutí. V návaznosti na délku stisknutí a stav měřiče je poté provedena příslušná akce. Na obr. 7.1 je schéma zpracování vstupů od ovládacích tlačítek a přepínačů. Je zde také znázorněno, jakým způsobem reaguje měřič na daný povel od tlačítek, příp. vnitřních událostí. Velké barevné kolečko znamená dlouhé bliknutí příslušnou barvou, menší kolečko odpovídá kratšímu bliknutí.

U horního ovládacího tlačítka jsou rozeznávány dva časové intervaly. Dlouhý stisk tlačítka na více než 1 sekundu smaže poslední záznam, toto je potvrzeno



Obr. 7.1: Funkce uživatelského rozhraní

bliknutím informační LED v sekvenci červená-zelená-červená. Krátký stisk (do 1 s) spustí nebo ukončí měřicí úlohu. Spuštění úlohy je ohlášeno zeleným bliknutím. Zastavení úlohy potvrzuje modré bliknutí. To, která úloha (krokoměr/měření zrychlení) je ovládána je určeno přepínačem DIP1.

U spodního tlačítka jsou rozeznávány tři časové intervaly. Stisknutí na dobu delší než 5 sekund spustí funkci pro smazání všech záznamů. Stisk mezi 1 až 5 sekundami vypne celý měřič, nebo přepne napájení Bluetooth modulu, dle nastavení přepínače DIP2. Krátké stisknutí do jedné sekundy zastaví aktuální měření a ihned spustí měření nové.

7.1.2 Implementace funkcí hardware

Čidlo ADXL345

Modul firmware *ADXL345.h(.c)* obsahuje funkce a potřebné definice k obsluze senzoru. Senzor má poměrně široké možnosti nastavení. Při počáteční inicializaci funkcí `void ADXL_Init(void);` je nastavena přenosová rychlost na 1600 Hz, rozsah měření zrychlení ±4g, velikost změny zrychlení, jež je považována za aktivitu na 0,625g, změny pod touto úrovní jsou považovány za inaktivitu. V případě inaktivity delší než nastavených 60 sekund je vyvoláno externí přerušení, které je softwarovým nastavením připojeno v vývodu INT1. Dále jsou nastaveny parametry pro detekci dvojitého klepnutí (Double Tap), při jeho detekci bude vyvoláno externí přerušení na vývodu INT2. Po tomto nastavení a přečtení (resetu) paměti přerušení je akcelerometr přepnut do režimu měření.

Po přepnutí senzoru do měřicího módu je možno kdykoliv na vyžádání číst přes SPI data, např. měřené velikosti zrychlení. Toto čtení zajišťuje funkce `uint8_t ADXL_Read(uint8_t addr)`, která čte prostřednictvím SPI byte z adresy `addr` registru ADXL345. Data měřeného zrychlení jsou uložena v šesti registrech (6 uint_8 hodnot) následujících za sebou od adresy `0x32`(konstanta `DATAX0` v *ADXL345.h*). Čtení těchto registrů provádí funkce `void ADXL_ReadAll(Accel_data *obj)`. Tato funkce zavolá lokální funkci `static uint8_t ADXL_Read_Data(uint8_t addr,uint8_t num, uint8_t *data)`, která čte ze senzoru celkem `6` bytů a ukládá je do ukazatelem určené proměnné `*data`. Po načtení naměřených hodnot z registrů senzoru do pomocného pole šesti bytů jsou načtená data převedena (př. hodnota v ose X) `obj->X=((int16_t)data[1]<<8)|(int16_t)data[0];` a korekcí přepočtena `obj->X=((obj->X-OFFSET_X)*0x0100)/SENS_X;` na 16-bitové číslo pro uložení do struktury typu `Accel_data` a poté zapsána na adresu určenou ukazatelem při volání funkce. Uživatelsky definovaný typ `Accel_data` má následující definici (v souboru *hw_defs.h*):

```
typedef struct {
int16_t X, Y, Z;
}Accel_data;
```

V hlavičkovém souboru *ADXL345.h* jsou dále definovány konstanty adres jednotlivých registrů, konstanty pro korekci citlivosti a offsetu senzoru. Protože je senzor napájen napětím 3.3 V, a ne nominálním 2.5V, musí být provedena korekce odchylky měření, způsobené jinými elektrostatickými silami působícími na mikrostrukturu čidla, než jsou nominální. Dále jsou v tomto hlavičkovém souboru deklarovány kromě výše popsaných funkcí pro čtení i ostatní funkce pro obsluhu senzoru. Funkce `void ADXL_Write(uint8_t addr, uint8_t data);` zapíše data do senzoru na adresu `addr`. Funkce je použita jednak ke konfiguraci senzoru, ale také pro přepnutí do sleep módu. Poslední funkcí obsaženou v tomto modulu je funkce pro nalezení kalibračních korekcí `void ADXL_Find_Offsets(void);`, která je navržena pro jednorázové zjištění kalibračních korekcí během vývoje zařízení ve spolupráci s debuggerem Atmel Studia.

Ukládání dat do FRAM, paměťová mapa

Funkce pro obsluhu FRAM paměti jsou sdruženy v modulu *fram.c(.h)*. Zde jsou definovány jednak funkce pro přímé čtení a přímý zápis do paměti, tak funkce vyšší úrovně realizující jednoduchý „souborový systém“, a funkce pomocné.

Způsob ukládání jednotlivých záznamů jsem pro tyto účely navrhl jako jednoduchý souborový systém, viz tab. 7.1. Na úplném začátku paměťového prostoru (prvních 64B) jsou uloženy konfigurační data měřiče, především je ale zde uložen počet záznamů uložených v paměti. Dále následuje alokační tabulka pro 256 záznamů, zde se ukládají informace o typu a umístění dat jednotlivých záznamů, v pořadí za sebou tak, jak byly postupně ukládány. Následující datová část paměti již obsahuje data jednotlivých záznamů lineárně za sebou.

Funkce pro zápis do paměti FRAM:

```
void FRAM_Write_Data(const uint32_t addr, const uint16_t size, \
                     const void *src);
void FRAM_Write(const uint32_t addr, const uint8_t data);
void FRAM_WriteL(uint32_t addr, const uint32_t data);

void FRAM_WriteFileInfo(const uint8_t Num, const uint8_t Type, \
                        const uint32_t Size);
void FRAM_WriteFileHeader(const uint8_t Num, const void *FileHeader,\
                          const uint32_t HeaderSize);
```

Tab. 7.1: Rozvržení paměti FRAM na jednotlivé části

Adresa	**Uložená data**	**Velikost [B]**
0x00-0x3f	**Konfigurace měřiče**	64
0x00	Počet uložených záznamů	1
0x01-0x05	adresa konce posledního záznamu	4
0x06-0x0b	čas poslední události	6
0x0c-0x0f	nastavení časovače měření zrychlení	4
⋮		
0x40-0x840	**Alokační tabulka záznamů**	2k(256*8)
0x40 - 0x43	adresa konce záznamu č.0	4
0x44 - 0x48	typ záznamu 1B, další info...	4
	0x01 - krokoměření, 0x02 - měření zrychlení	
0x48 - 0x4b	adresa konce záznamu č.1	4
0x4c - 0x4f	typ záznamu 1B, další info...	4
⋮		
0x840-0x3ffff	**data jednotlivých záznamů**	253k
	(příklad - krokoměr, akcelerometr)	
0x00840-0x00845	čas začátku 1. záznamu (krokoměření)	6
0x00846-0x0084b	čas konce prvního záznamu	6
0x0084c-0x0084f	počet kroků	4
0x00850-0x00851	max rychlost kroků/hod	2
0x00852-0x00853	avg rychlost kroků/hod,	2
0x00854-0x00859	čas začátku 2. záznamu (měření zrychlení)	6
0x0085a-0x0085f	čas konce druhého záznamu	6
0x0085a-0x0085f	čas konce druhého záznamu	6
0x00860-0x00863	adresa začátku dat z měření	4
0x00864-0x00867	nastavení časovače měření zrychlení	4
0x00868-0x00873	min a max hodnoty změřeného zrychlení	12
0x00874-0xXXX	jednotlivé „šestibyty“ záznamu měření zrychlení	dle délky záznamu
⋮		
0x3ffff	<Konec paměti>	

Obecná funkce pro zápis dat `FRAM_Write_Data` provede zápis bloku dat z paměti mikrokontroléru od adresy dané ukazatelem `*src` o velikosti `size` do paměti na adresu `addr`. Tuto základní funkci využívají ostatní funkce zápisu do paměti FRAM.

`FRAM_Write` zapíše jeden byte a `FRAM_WriteL` zapíše čtyři byty(long int). Funkce vyšší úrovně zajišťují zápis dat při ukončení úlohy měření. `FRAM_WriteFileInfo` zapisuje informace do alokační tabulky, `FRAM_WriteFileHeader` zapíše hlavičku záznamu. V případě úlohy Krokoměr jsou veškeré informace o měření součástí hlavičky. V případě měření zrychlení však hlavička obsahuje jen základní informace (min/max hodnoty, časy apod.) a počáteční adresu dat měření. Data měření zrychlení jsou uložena v paměti za hlavičkou záznamu.

Funkce pro čtení dat z FRAM:

```
uint8_t FRAM_Read(const uint32_t addr);
uint32_t FRAM_ReadL(const uint32_t addr);
void FRAM_Read_Data(const uint32_t addr, uint16_t size, void *dest);
void FRAM_ReadFileInfo(const uint8_t Num, uint8_t *Type, uint32_t \
                       *Size, uint32_t *StartAddress);
```

Tyto funkce pro čtení jsou komplementární k příslušným funkcím pro zápis dat. `FRAM_Read` slouží k načtení jednoho byte z paměti, načtený byte vrací jako návratovou hodnotu. Podobně návratovou hodnotu typu `uint32_t` (unsigned long int) vrací funkce `FRAM_ReadL`. Obecná funkce pro čtení bloku dat o zadané velikosti se jmenuje `FRAM_Read_Data`. Funkce `FRAM_ReadFileInfo` je určena k získání základních informací o konkrétním záznamu specifikovaného parametrem `Num`, vrací typ uloženého záznamu, jeho velikost a počáteční adresu uložení v FRAM paměti. Se znalostí těchto údajů lze načíst z FRAM kompletní záznam funkcí `FRAM_Read_Data()`.

Dále jsou v modulu *fram.c(.h)* definovány pomocné funkce. Funkce `void FRAM_Clear(void);` přepíše prvních 0x840 bytů paměti hodnotou 0, a uloží do paměti aktuální čas a základní nastavení časovače pro vzorkování měření zrychlení. `void FRAM_Sleep(void);` a `void FRAM_WakeUp(void);` slouží pro přechod do úsporného režimu a návrat zpět, `uint8_t FRAM_Read_Status(void);` vrací informace o aktuálním stavu paměti, především informaci o zamknutí některé její části proti zápisu. `uint32_t FRAM_Free(void);` vrací počet volných bytů FRAM paměti.

Měření napětí baterie

Napětí baterie je přes dělič 1/2 přivedeno na vstup ADC0 mikrokontroléru. Každou minutu (`if (Cas.min==Cas.sec)`) je z hlavní smyčky spuštěn převod napětí A/D převodníkem s příslušným nastavením (vstup ADC0, vnější referenční napětí, povolení přerušení). Po dokončení měření je vyvolána funkce přerušení `ISR(ADC_vect)`. Vzhledem k tomu, že měření napětí není nutné vyhodnotit v reálném čase je zde

pouze nastaven příznak `ISR_ADC` v globální proměnné `ISR_Vector`. Při dalším průchodu hlavní smyčkou je tento příznak vyhodnocen a je spuštěna obslužná funkce k vyhodnocení měření `void ISR_Adc(void)`. V ní je načteno měřené napětí z registru procesoru ADCW, a provedeno porovnání na zvolené hodnoty. Měřené napětí je převedeno na šestnáctibitové číslo ADCW dle vztahu

$$ADC = \frac{U_I N \cdot 1024}{U_R EF} \tag{7.1}$$

Hodnotu kritického napětí napájecí baterie jsem zvolil 2.80V, což odpovídá hodnotě ADCW=0x23d. Uvedené napětí je obvykle uváděno jako nejnižší dovolené napětí článku Li-Pol. Při dosažení tohoto napětí je proveden měřiče do úsporného režimu, a jeho nové zapnutí je možné pouze tlačítkem Reset. V době před úplným vypnutím ale měřič upozorňuje na nízké napětí (pod 3.2V) bliknutím červenou LED po každém změření napětí.

Bluetooth modul

Zpracování příchozích dat ze sériové linky přemostěné Bluetooth rozhraním probíhá dle schématu obr. 7.2 a je spuštěno příznakem přerušení `ISR_USART_RX`.

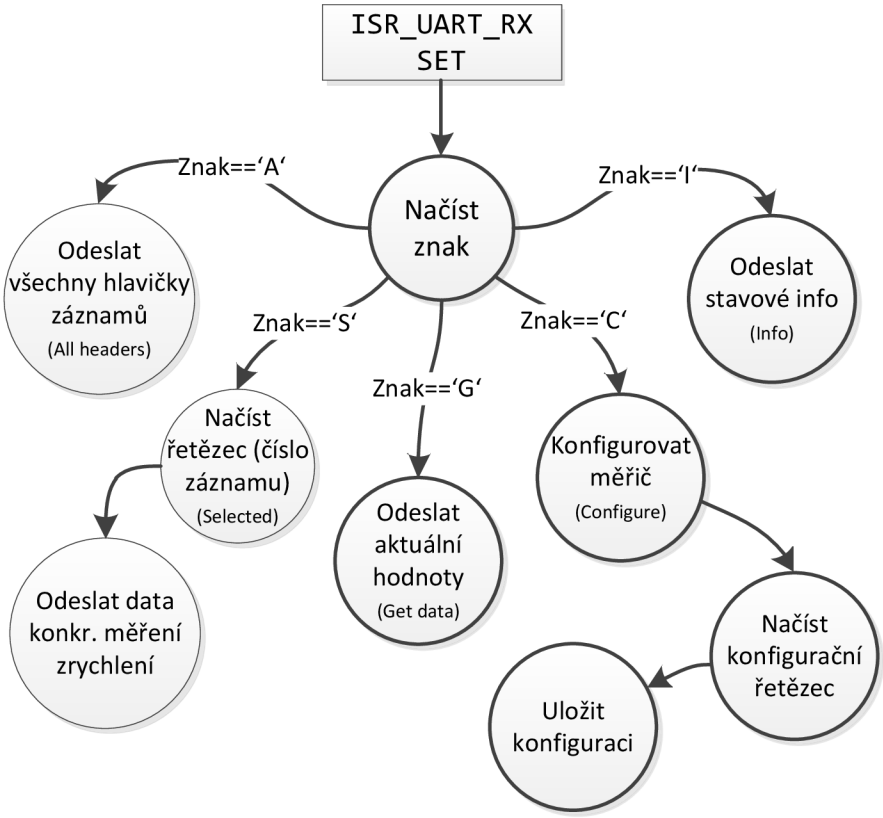
Pro komunikaci mezi měřičem a PC jsem navrhl jednoduchý protokol pracující v ASCII režimu. Díky tomu lze načítat data a měnit konfiguraci i běžným terminálem komunikujícím po sériové lince.

V případě, kdy se na vstupu USART rozhraní objeví jakákoliv příchozí data, funkce obsluhy přerušení nastaví příslušný příznak v proměnné `ISR_Vector`, který je vyhodnocen na začátku každého průchodu hlavní smyčkou a v návaznosti na něj je spuštěna obslužná funkce USARTu `void ISR_Usart_Receive(void)`. Tato funkce načte první znak ze vstupního bufferu, a pokud odpovídá některému ze známých příkazů, provede příslušnou akci dle schématu 7.2.

Komunikace směrem k PC je založena na podobném principu jednoduchého ASCII protokolu. Jsou zde zavedeny pakety s unifikovaným formátem, každý paket začíná znaky `%ACC`, následují jednotlivé údaje oddělené středníkem, a končí koncem řádku (sekvencí `\r\n`). V tab. 7.2 a 7.3 je uveden seznam všech implementovaných USART příkazů a odpovídajících odezev.

7.2 Testovací měření

Po implementaci uživatelského rozhraní, komunikace se senzorem, systému ukládání dat a komunikačního protokolu jsem vytvořil jednoduchou měřicí úlohu (postupně přetvořenou do úlohy Akcelerometr, popsanou dále), pomocí které jsem experimentálně nasbíral několik záznamů testovacích dat. Jednak jsem zaznamenal chůzi a



Obr. 7.2: Komunikace po sériové lince

Tab. 7.2: Příkazy pro čtení dat z měřiče

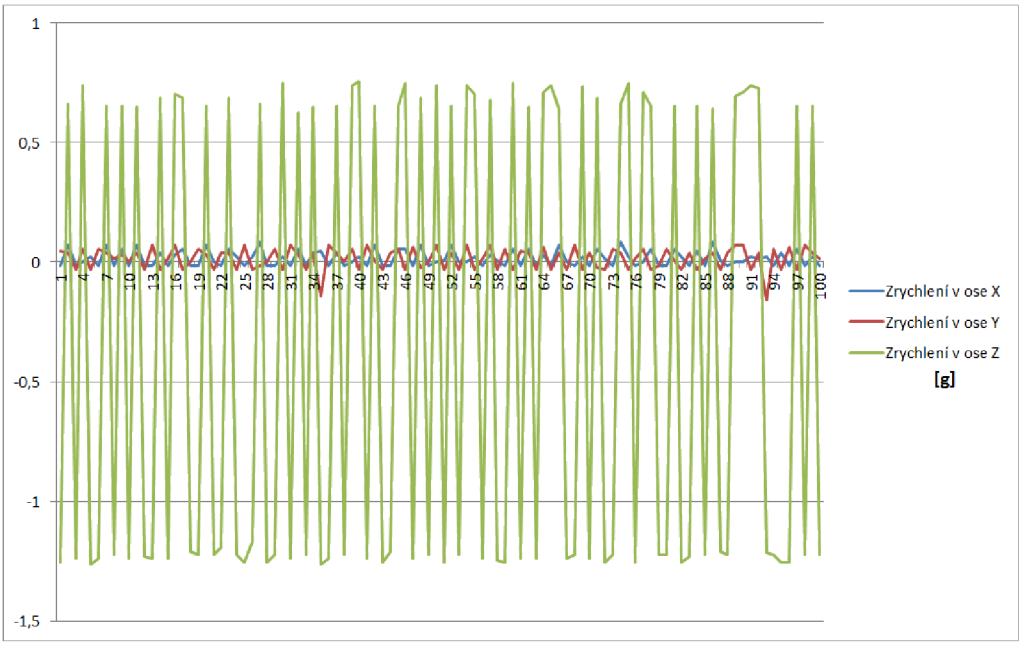
Příkaz	**Popis, příklad paketu**	**Význam**
'G'	Načtení aktuálních dat	
	%ACC;AKT;-123;-4;-2;	Aktuální měřená data; X; Y; Z
'I'	Načtení stavového infa	
	%ACC;STA;AT;10;16;58;31;402;Cnt;	Stavové info; Aktuální čas;
	37;Free;3;100;43;ACS;0;3;251;35;	Počet záznamů; Volná paměť;
		Nastavení časovače měř. zrychl.
'A'	Načtení všech hlaviček záznamů	
	%ACC;SCH;ST;50;10;6;9;328;ET;	Měření kroků; Počáteční čas;
	50;10;6;28;332;Num;2;Avg;0;Max;0;	Konc. čas; Počet kroků; Prům.;
		a Maximální rychlost
	%ACC;AMH;ST;50;10;6;37;334;ET;	Měření zrychlení;Počáteční čas;
	50;10;6;56;240;SR;3,fb1e;MxX;79;	Koc. čas; Nastavení vzorkování;
	MnX;-135;MxY;80;MnY;-116;MxZ;	Min a Max naměřené hodnoty;
	230;MnZ;-101;Sz;0,765;	Počet jednotlivých vzorků
	⋮	
	%ACC;HDR;END;	Ukončovací sekvence
'S5'	Načtení dat záznamu č.5	
	%ACC;AMD;START;	Úvodní paket,
	%ACC;12;7;-85;	počáteční statické zrychlení
	%ACC;1;-1;-43;	odchylka od počátku X; Y; Z
	%ACC;1;-1;-43;	odchylka od počátku
	⋮	⋮
	%ACC;1;2;-42;	odchylka od počátku
	%ACC;AMD;END;	Koncový paket

běh, a také jízdu automobilem. Během prvních záznamů jsem zjistil, že signál senzoru obsahuje velmi silné rušení, viz obr. 7.3. Toto rušení dosahovalo relativní odchylky až přibližně ±0.9g, i když byl senzor v klidu. Vzhledem k tomu, že od úlohy obecného měření zrychlení jsem očekával lepší výsledky, bylo nutno najít příčinu. Po prostudování datasheetu senzoru [2] a proměření zdroje 3.3V osciloskopem jsem došel k závěru, že toto rušení je způsobeno právě tímto spínaným zdrojem. Velikost kolísání napětí tohoto zdroje byla naměřena ve velikosti 0.08V. Jako nejjednodušší a účinné řešení se ukázalo zařazení Schottkyho diody do přívodu napájení senzoru. Za touto diodou je na desce modulu připojen blokovací kondenzátor, který dostatečně

Tab. 7.3: Příkazy pro konfiguraci měřiče

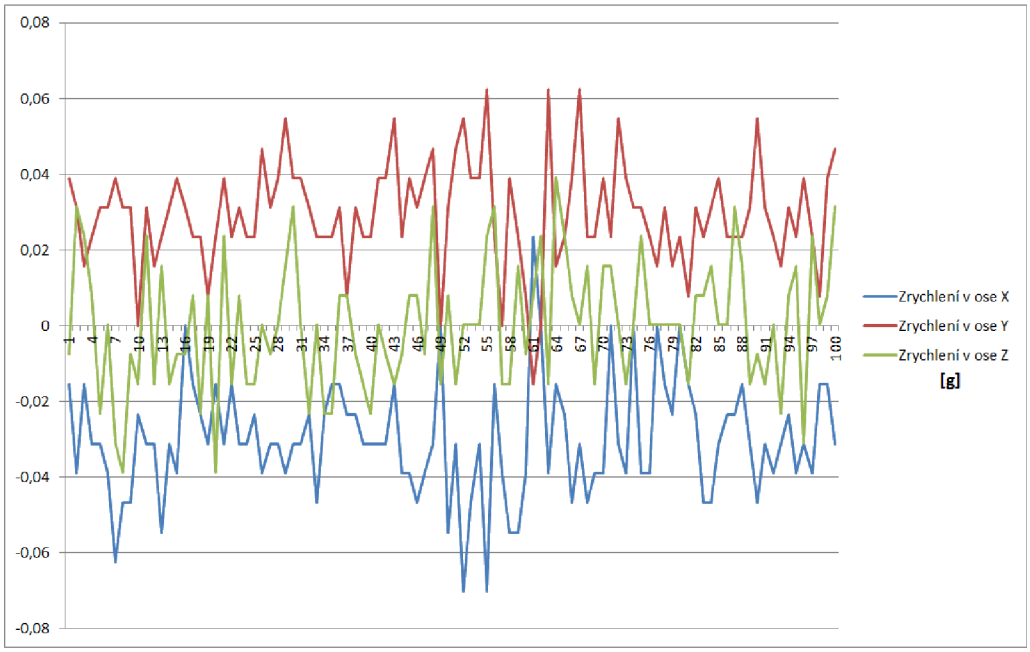
Příkaz	**Popis, formát zprávy**	**Význam**
'CT'	Nastavení hodin RTC	
	CT;10;15;25;50;	Nastavení dne;hod.; min.; sek.
'CS'	Nastavení časovače akcelerometru	
	CS;3;253;143;	hodnoty jednotl. registrů TIMERu
'CC'	Smazání posledních n záznamů	
	CC;n;	n dosadit počet mazaných zázn.

stabilizuje napájecí napětí senzoru. Na fotografii modulu ADXL345 3.2 je tato dioda viditelná, částečně zakrývá potisk ADXL345 na destičce senzoru. Po této jednoduché úpravě se snížil šum na úroveň kolem ±0.03g, záznam šumu v klidovém stavu je na obr.7.4.



Obr. 7.3: Šum senzoru v klidu před úpravou napájení

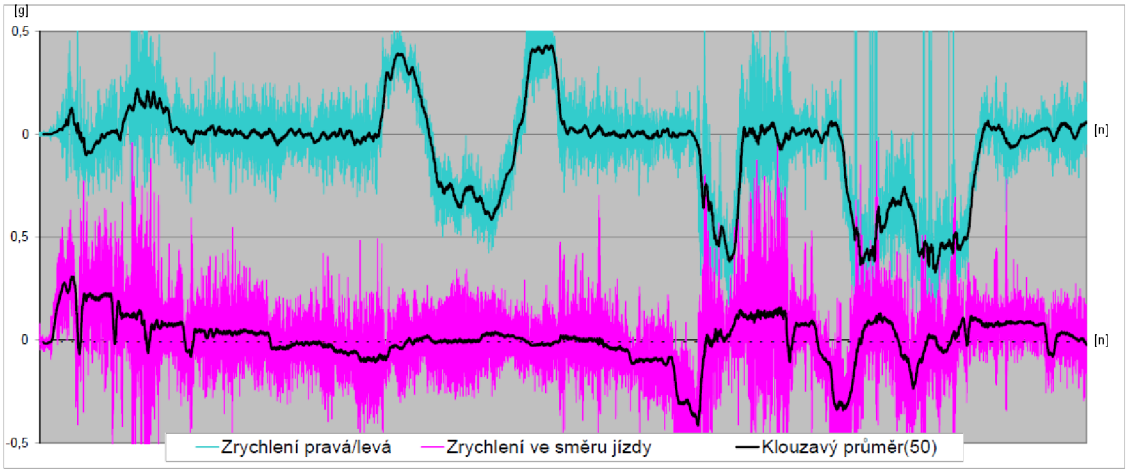
Pro obecné měření zrychlení je potřeba zajistit vhodné nepružné spojení měřené soustavy a měřiče. Při testovacím měření v automobilu jsem měřič uložil na tlumicí podložku na palubní desku. Měřič jsem napolohoval osou Z svisle, a osou X



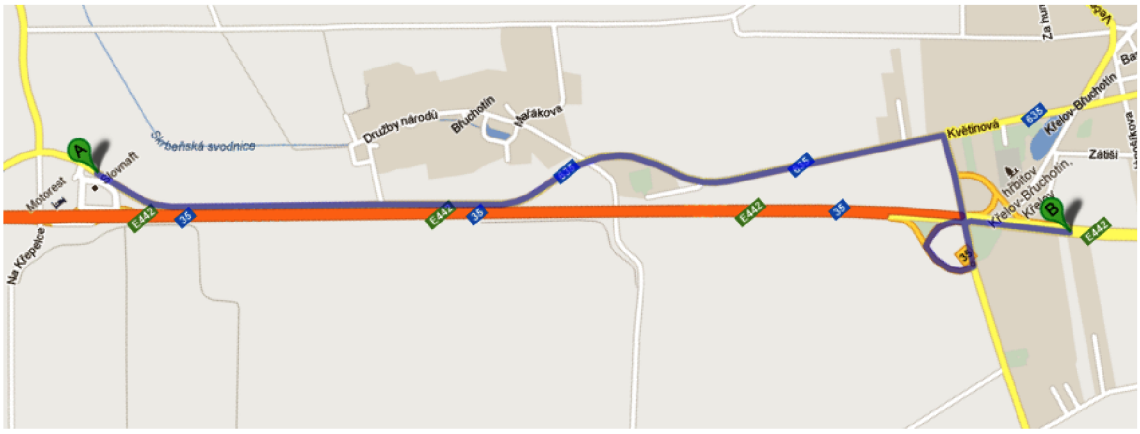
Obr. 7.4: Šum senzoru v klidu po úpravě napájení

rovnoběžně s podélnou osou automobilu. Použité umístění měřiče však nebylo nejvhodnější, palubní deska je pružná a při jízdě značně rezonuje a kmitá, což vnáší do měření šum. Přesto však naměřená data poskytují poměrně jasný obrázek o průběhu jízdy. Pro toto měření by bylo vhodné, vzhledem k pomalému a plynulému průběhu změn sledovaného zrychlení, upevnit měřič na podložku s kvalitním tlumením vibrací. Naměřená data zanesená do grafu a doplněná o plovoucí průměr posledních 50 hodnot jsou zobrazena na obr. 7.5. Na dalším obrázku 7.6 je mapka zobrazující projetou trasu. Celková délka trasy z bodu A do bodu B je 3,1 km, doba záznamu 138,83 vteřin, použitá vzorkovací perioda byla 10 ms (tzn. celkový počet vzorků byl 13883). V grafu je zobrazeno jednak zrychlení ve směru jízdy, na počátku trasy je identifikovatelné řazení jednotlivých rychlostních stupňů (spodní křivka), a také je z horní křivky možné odečíst hodnoty dostředivého zrychlení působící na senzor při průjezdu zatáčkami.

Pro úlohu krokoměru bylo stěžejní nalézt optimální algoritmus detekce kroků. Po naměření několika vzorových záznamů a zobrazení jejich grafů v programu Excel jsem usoudil, že vhodný způsob detekce kroků by mohl být založen na vyhledávání průsečíků filtrovaného měřeného signálu a střední hodnoty tohoto signálu. Pokud bude počítáno s vektorovým součtem zrychlení působícího ve všech třech osách, odstraní se tím závislost na polohování měřiče. Během testování jsem zjistil, že vhodné umístění měřiče pro daný způsob detekce je na takovém místě, kde jsou



Obr. 7.5: Graf měření zrychlení automobilu

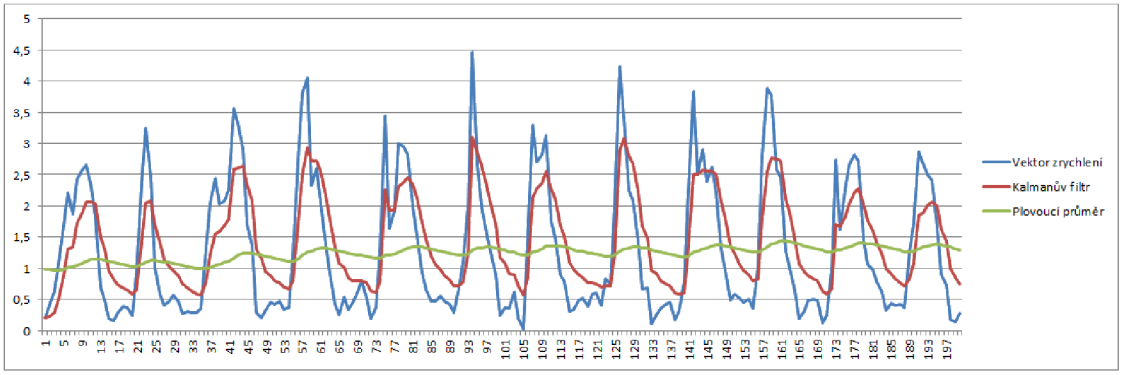


Obr. 7.6: Mapka trasy (modře) měření zrychlení automobilu

změny polohy přístroje během pohybu co možná nejplynulejší. Tedy např. na opasku, v kapse bundy, připevněný na popruhu na noze, nebo na ruce. Všechny tyto pozice poskytovaly při testování správné výsledky. Naproti tomu umístění měřiče volně do kapsy kalhot na stehně se ukázalo jako naprosto nevhodné, při chůzi docházelo k náhodným, i vícečetným odrazům měřiče od svalů končetiny což vedlo k chybným výsledkům měření.

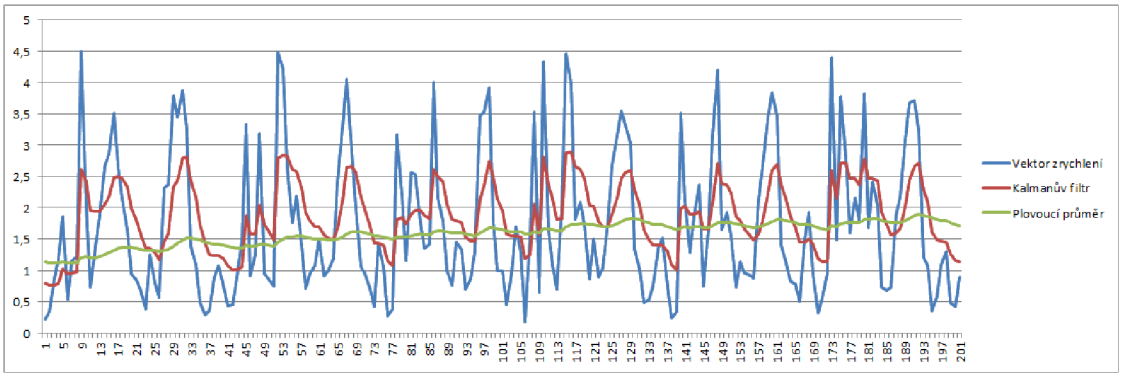
Při hledání vhodného algoritmu pro filtraci měřeného signálu jsem narazil na zmínku o Kalmanově filtraci. Po prostudování principu tohoto filtru jsem jej otestoval na naměřených datech s různým nastavením parametrů. Použil jsem algoritmus z webu leteckých modelářů RCEO [12] k filtrování signálu z čidla, pro detekci průsečíků je ještě dále počítán plovoucí průměr s exponenciálním zapomínáním nastavený na velmi dlouhé průměrovací okno, způsob výpočtu jsem použil ze stránek robotika.cz [11].

Experimentoval jsem také s nastavením vzorkovací frekvence, nakonec jsem zvolil periodu vzorkování 2 ms. Mikrokontrolér bez problémů zvládá všechny potřebné výpočty i při této rychlosti vzorkování, a kvalita signálu je pro danou úlohu naprosto vyhovující. Při dlouhé periodě vzorkování (testoval jsem 20ms) byl vzorkovaný signál i po filtraci za určitých podmínek až nepoužitelný. Při běžné chůzi, kdy jsou změny měřeného zrychlení pomalé a plynulé, bylo vzorkování vzorkování 20 ms periodou vyhovující, viz obr 7.7, při běhu však již docházelo k náhodným překmitům, viz obr 7.8, a v důsledku malého množství dat k analýze nemohla ani filtrace Kalmanovým filtrem zajistit kvalitní detekci kroků.



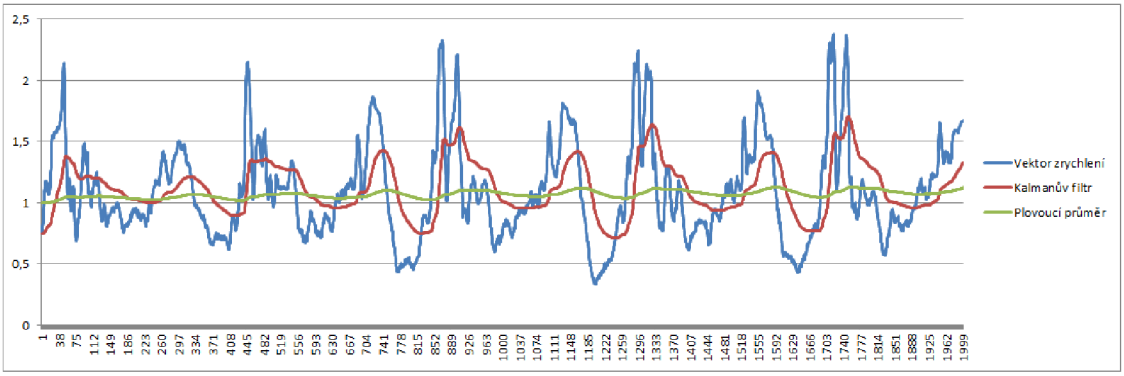
Obr. 7.7: Graf záznamu krokoměru se vzorkováním 20ms - chůze

Pro vývojové účely jsem nejprve filtrační algoritmy realizoval na PC, data pro měření jsem po nasnímání a uložení do FRAM paměti přenesl do PC a analýzy prováděl na vzorcích dat o známém počtu kroků iteračně. Postupně jsem zkoušel různé parametry nastavení filtrů a porovnáním výsledků algoritmu se skutečným

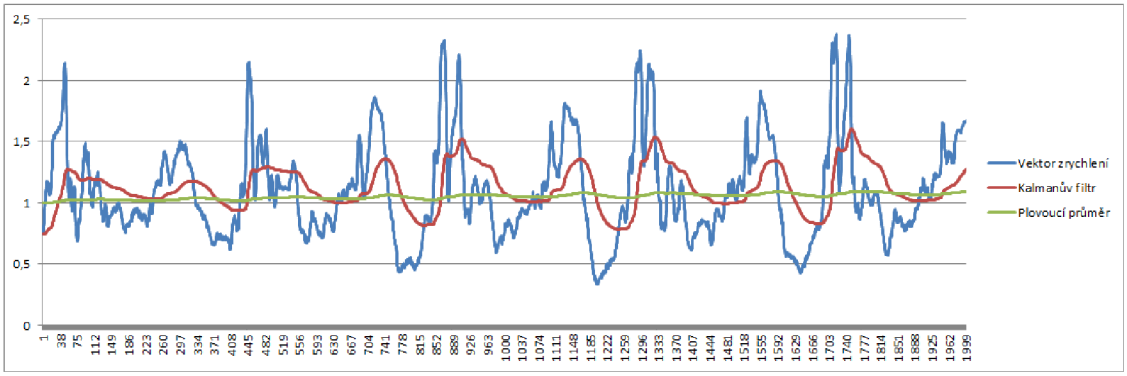


Obr. 7.8: Graf záznamu krokoměru se vzorkováním 20ms - běh

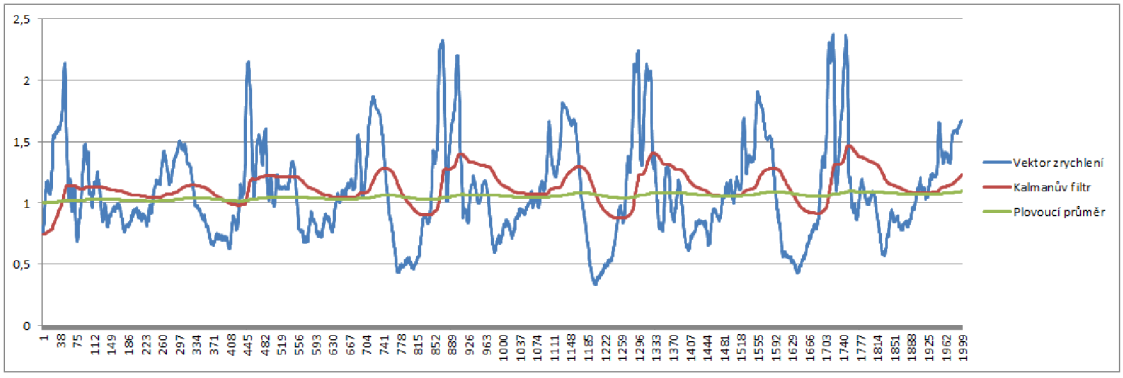
počtem kroků jsem zvolil vhodné parametry, jež dávaly dostatečně přesné výsledky (stanovil jsem si cílovou odchylku do 5%). Poté jsem nasnímaná data doplnil o vypočtené hodnoty a v programu MS Excel jsem si vygeneroval jednotlivé průběhy měřených a vypočtených signálů. Tímto jsem zkontroloval, že algoritmus skutečně počítá kroky, nikoliv že správný počet kroků z algoritmu vyšel jen náhodou. Pro ilustraci uvádím grafy pro tři různá nastavení parametrů filtrace, nejpřesnějších výsledků spočtených kroků dosahuje algoritmus při nastavených parametrech dle grafu7.10, tedy filtrační faktor Kalmanova filtru 30, průměrovací okno pro plovoucí průměr o velikosti 1000 vzorků.



Obr. 7.9: Graf krokoměru, filtrační faktor 20, průměrovací okno 500 vzorků, perioda vzorkování 2ms



Obr. 7.10: Graf krokoměru, filtrační faktor 30, průměrovací okno 1000 vzorků, perioda vzorkování 2ms



Obr. 7.11: Graf krokoměru, filtrační faktor 50, průměrovací okno 1000 vzorků, perioda vzorkování 2ms

7.3 Měřicí moduly firmware

Firmware obsahuje dvě měřicí úlohy. Funkce pro obecné měření zrychlení jsou definovány v modulu *accmeter.h(.c)*. Funkce krokoměru jsou definovány v modulu *stepcounter.c(.h)*. Filtrace signálu je obecně použitelná ve více úlohách, proto jsou filtrační funkce definovány v samostatném modulu *filtry.c(.h)*. Hlavní funkce obou měřicích úloh jsou spouštěny v pravidelných intervalech vnitřním časovačem mikrokontroléru.

7.3.1 Accmeter - obecné měření zrychlení

Při aktivním měřicím módu pro obecné měření zrychlení (DIP1=1) je spuštění úlohy akcelerometru předřazena časová prodleva provázená blikáním zelené LED, viz schéma uživatelského rozhraní na obr. 7.1. Tato prodleva je zařazena pro možnost odeznění přechodového děje mezi zmáčknutím ovládacího tlačítka a tím způsobených případných vibrací celého zařízení.

Měření využívá integrovaný časovač mikrokontroléru TIMER1. Při aktivaci úlohy je z paměti FRAM načteno nastavení parametrů tohoto časovače, a také je z posledního uloženého záznamu zjištěna adresa pro ukládání dat. Tato adresa je uložena do lokální statické proměnné `FRAM_addr`, a dále je tato proměnná aktualizována při každém zápisu aktuální hodnoty měření do paměti. Po nastavení adresy pro ukládání dat je inicializována statická strukturní proměnná `AccMeter` typu `AccelMeterData` pro ukládání informací o měření. Strukturní typ AccelMeterData je definován v modulu *hw_defs.h*:

```
typedef struct acccdata
{
RTC StartTime;                  //počáteční čas měření
RTC EndTime;                    //koncový čas měření
uint32_t DataLocation;          //adresa uložení prvních dat měření
uint32_t SampleRate;            //použité nastavení časovače
int16_t MaxAccX, MinAccX;       // Min a Max měřené hodnoty zrychlení
int16_t MaxAccY, MinAccY;
int16_t MaxAccZ, MinAccZ;
} AccelMeterData;
```

Poté je nastaven časovač na načtené hodnoty, povoleno přerušení od časovače TIMER1 a nastaven globální příznak povolení přerušení. Tím je dokončena inicializace měření a řízení je předáno do hlavní smyčky.

Jakmile dojde k přetečení časovače, je spuštěna rutina obsluhy přerušení, ta nastaví příslušný příznak v globální proměnné `ISR_Vector` a při vyhodnocení příznaku v hlavní smyčce je spuštěna funkce pro záznam aktuální hodnoty zrychlení `void ISR_Accmeter(void)`. Tato funkce při prvním spuštění (tedy na počátku měření) změří a vyhodnotí aktuální statické zrychlení. V této chvíli by měl být měřič v klidu.

Po zjištění statického zrychlení je načtena funkcí `ADXL_ReadAll(&Data);` aktuální hodnota zrychlení, hodnota je porovnána s aktuálními hodnotami maxima a minima měřeného zrychlení, které jsou případně aktualizovány. Od aktuální měřené hodnoty je poté odečteno statické zrychlení. Poté je uložena jako šestice bytů do paměti FRAM na adresu `FRAM_addr`. Měřené hodnoty zrychlení nejsou nijak filtrovány, z důvodu možného odfiltrování užitečných dat. Je zvýšena hodnota lokální statické proměnné `FRAM_addr` a poté je funkce ukončena.

V případě dosažení konce paměti FRAM je rozsvícena zelená a modrá LED pro indikaci plné paměti. Je uložen čas konce záznamu a deaktivován časovač spouštějící měření. V tomto stavu zařízení čeká na potvrzení ukončení měření tlačítkem.

Maximální počet zaznamenaných vzorků při prázdné paměti je 43 333, což při vzorkovací periodě 2 ms obsáhne dobu záznamu 86 sekund. Tato perioda je tedy vhodná pro záznam krátkých rychlých dějů. Naproti tomu při použití vzorkovací periody 100 ms lze uložit až 72 minut záznamu.

Ukončení měření je vyvoláno stisknutím ovládacího tlačítka, zápis informací o měření zajišťuje funkce `void StopAccMeter(void)`. V této funkci je deaktivováno přerušení při přetečení časovače a vytvořen v paměti nový záznam. Jednak je zapsán údaj do alokační tabulky, poté je proměnná `Accmeter` zapsána do paměti FRAM jakožto hlavička celého záznamu měření, a nakonec je čítač záznamů uložený v FRAM na adrese 0x0 zvýšen o 1.

7.3.2 Stepcounter - krokoměr

Měření je spuštěno stisknutím tlačítka při aktivovaném režimu krokoměru (DIP1=0). V hlavičkovém souboru *stepcounter.h* jsou definovány konstanty použité při filtraci signálu a konstanty pro snížení citlivosti detekce kroků jak v čase, tak v amplitudě. Tato necitlivost je přidána do algoritmu pro zamezení falešných vyhodnocení kroků při zastavení chůze či běhu, kdy jsou změny zrychlení víceméně náhodné, v malé amplitudě a mnohdy ve velmi krátkých časových intervalech za sebou.

Funkce spuštění krokoměru `void StartStepCounter(void)` inicializuje lokální statickou proměnnou `SCounter` typu `StepCounterData` na počáteční nulové hodnoty a uloží do ní aktuální čas. Poté nastaví a spustí časovač TIMER2, který na

konfigurován pro periodu přetečení 2 ms, a vrátí řízení hlavnímu programu. Datový typ `StepCounterData` je definován v modulu *hw_defs.h*:

```
typedef struct scdata
{
RTC StartTime;              //počáteční čas měření
RTC EndTime;                //koncový čas měření
uint32_t NumSteps;          //počet kroků
uint16_t MaxSpeed,AvgSpeed; //max. a min rychlost (neimplementováno)
} StepCounterData;
```

Z hlavního programu je již popsaným mechanismem spouštěna hlavní funkce krokoměru `void ISR_StepCounter(void)`. V této funkci jsou definovány dva logické přepínače `Nahoru` a `Dolu` používané k uložení předcházející vzájemné polohy. V této funkci proběhne výpočet vektorového součtu jednotlivých složek působícího zrychlení, výpočet plovoucího průměru a filtrace měřené hodnoty Kalmanovým filtrem. Vzájemná poloha těchto dvou hodnot je získána jejich odečtením. V případě, kdy rozdíl filtrované hodnoty a plovoucího průměru je:

- větší než necitlivost v amplitudě,
- a zároveň předchozí vzájemná poloha měla záporné znaménko (`Dolu==1`),
- a zároveň je počet načtených hodnot od posledního zaznamenaného kroku větší, než necitlivost v čase

je zvětšena proměnná udávající počet kroků o 1 (`SCounter.NumSteps++`)a pomocné logické proměnné `Nahoru` a `Dolu`, jsou nastaveny na hodnoty `Nahoru=1` a `Dolu=0`. Je tedy zaznamenána rostoucí tendence filtrovaného signálu oproti střední hodnotě. Až tehdy, kdy hodnota filtrovaného signálu opět poklesne pod hodnotu plovoucího průměru, provede se změna hodnot logických proměnných `Nahoru=0` a `Dolu=1`.

Inicializace proměnných používaných v této funkci proběhne před vlastním měřením, logické proměnné `Nahoru` a `Dolu` jsou obě nastaveny na `1`, Kalmanův filtr je inicializován hodnotou aktuálně změřené velikosti zrychlení a počáteční hodnota plovoucího průměru je nastavena na `1`(g).

7.4 Projekt Univerzální akcelerometr

Během prací na tomto projektu nebyly některé funkce implementovány nejvhodněji, bylo by vhodné je přepracovat, bohužel to již v rámci vymezeného času nebylo možné. Do budoucna plánuji přepracovat hodiny reálného času taky, aby byl skutečně reálný, jako nejvhodnější se jeví použití speciálního RTC obvodu. Dále by bylo výhodné přesunout kalibraci měřiče do PC. Mnohé z funkcí používaných k zápisu do paměti a ke komunikaci jsou potenciálně nebezpečné, při chybném

zadání velikosti ukládaných dat hrozí přepsání jiných údajů, kontrola správnosti dat je na programátorovi. Bylo by tedy vhodné tyto kontroly do funkcí doplnit. Do algoritmu krokoměru plánuji přidat algoritmy pro měření maximální a průměrné frekvence kroků. Také by bylo vhodné implementovat lepší signalizaci vybití baterie, aktuálně zařízení sice výstražně blikne každou minutu, to však pravděpodobně nebude v praxi dostatečné. Také bude nutné najít přesnou příčinu kolísání napájecího napětí zdroje 3,3V, nejpravděpodobněji bude problém v použitém Burst módu, kdy je obvod zdroje automaticky vypnut po dosažení nastaveného napětí a po poklesu tohoto napětí je opětovně nastartován. Tuto hypotézu nebylo možné ověřit z důvodu nutnosti přerušit spoj na DPS vedoucí přímo pod pouzdrem tohoto obvodu. Obával jsem se, že by při pokusu o jeho odpájení mohlo dojít k jeho nevratnému poškození, příp. poškození DPS.

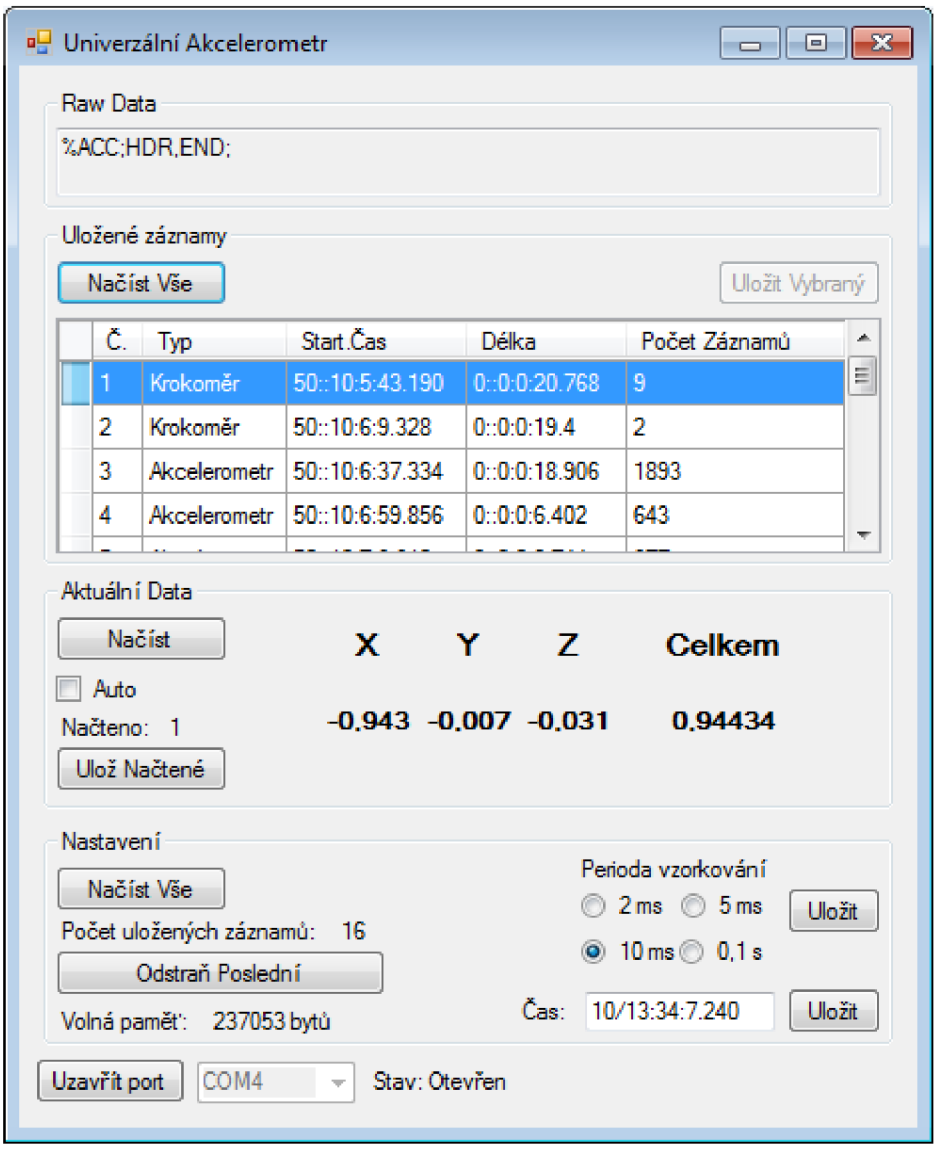
V úloze měření zrychlení by bylo možné experimentovat s použitím Kalmanova filtru k odstranění šumu, pravděpodobně by aplikace tohoto filtru v jednotlivých složkách měřeného zrychlení zvýšila kvalitu výstupu.

Také se nabízí přidat další měřicí úlohu, např. pádového detektoru, senzor ADXL345 obsahuje přímou podporu detekce volného pádu, bylo by tedy možné této funkce využít a rozšířit ji o záznam průběhu a komplexnější analýzu detekovaného pádu. Toto by bylo možné využít ve spojení s mobilním telefonem jako automatický hlásič pádu pro nemocné nebo hůře pohyblivé osoby.

Další možné využití by byl monitor aktivity, který by průběžně integroval naměřené změny zrychlení pro odhad uživatelem vydané energie.

Takto by vznikl Univerzální akcelerometr, široce použitelné zařízení pro různé druhy měřicích a detekčních úkolů.

8 DEMONSTRAČNÍ APLIKACE



Obr. 8.1: Okno komunikačního software pro Windows

Na obrázku 8.1 je okno ovládacího a komunikačního software pro Windows. Tato aplikace byla vytvořena ve Visual Studiu 2012, napsána je v jazyce C#. Okno je rozděleno do několika funkčních skupin. Pro připojení počítače k měřiči je nutné mít nainstalován Bluetooth adaptér, umožňující funkci virtuálního sériového portu. Před prvním připojením měřiče je nutno nejprve zařízení tzv. spárovat. To znamená, že počítač, který vystupuje v roli řídicího zařízení, vyhledá dostupná zařízení a poté je provedeno párování. Jde o autentizační proces, kdy je do zařízení poslán párovací kód, zařízení tento kód potvrdí, obě zařízení si vzájemně uloží identifikační údaje

partnera, a poté je navázáno spojení. Po přerušení a novém navázání komunikace již není nutné zařízení znovu párovat. Po spárování měřiče s PC je vytvořen virtuální sériový port, viditelný ve správci zařízení, mezi zařízeními Bluetooth se po spárování objeví zařízení s názvem HC-07. Ve vlastnostech lze zobrazit, který port je s tímto zařízením propojen.

Na spodním okraji okna vytvořené aplikace je tlačítko pro otevření vybraného portu. Po úspěšném otevření portu se aktivují ostatní ovládací prvky aplikace.

Úplně nahoře ve skupině ovládacích prvků *Raw Data* je textový box, v němž je zobrazen poslední načtený řetězec z měřiče.

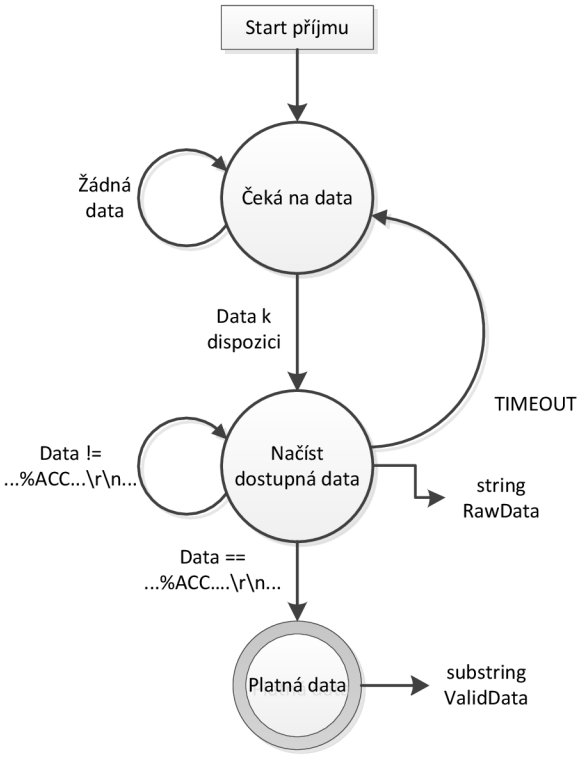
Pod ním je skupina *Uložené Záznamy*, kde jsou seskupeny ovládací prvky pro načítání obsahu paměti FRAM. Tlačítkem *Načíst Vše* je vyslán přes sériovou linku do měřiče příkaz `'A'`, měřič odpoví odesláním všech hlaviček záznamů, tak jak jsou uloženy v FRAM. V tabulce jsou poté zobrazeny některé z načtených údajů, ve sloupci *Počet Záznamů* je v případě, kdy jde o záznam z krokoměru, počet zaznamenaných kroků. V případě záznamu typu *Akcelerometr* je zde uveden počet jednotlivých sejmutých vzorků. Po kliknutí na záznam typu *Akcelerometr* lze tlačítkem *Uložit Vybraný* uložit v textovém CSV formátu data zaznamenaná při měření. Tato data lze dále zpracovávat, např. programem MS Excel do grafů, apod..

Uprostřed okna aplikace je skupina ovládacích prvků *Aktuální Data*, kde lze provést načtení aktuální hodnoty zrychlení působící na měřič. Zatržením políčka *Auto* lze aktivovat automatické načítání. Hodnoty jsou průběžně ukládány do paměti počítače a později je lze uložit tlačítkem *Uložit Načtené*. Vzhledem k použité komunikační rychlosti (38400 Bd) je maximální dosažitelná vzorkovací frekvence 20 vzorků.s^{-1}.

Ve spodní části okna aplikace je skupina ovládacích prvků *Nastavení*, kde lze kliknutím na tlačítko *Načíst Vše* zjistit aktuální nastavení měřiče, odstranit poslední záznam, změnit nastavení periody vzorkování pro úlohu měření zrychlení a nastavit čas vnitřních hodin měřiče. Také je zde zobrazena volná kapacita paměti FRAM.

Jádrem celé aplikace je sériová komunikace s měřičem, většina ostatních implementovaných funkcí slouží k obsluze uživatelského rozhraní, ke konverzi mezi různými datovými typy a k ukládání dat. Vzhledem k tomu, že senzor vysílá data po sériové lince na požádání, zvolil jsem pro komunikaci schéma, kdy po vyslání příkazu je port čten ve smyčce a čtení je přerušeno existencí validních dat nebo uplynutím nastaveného časového intervalu. Na obr. 8.2 je schéma stavového automatu obsluhujícího čtení dat ze sériového portu.

Po načtení validních dat, která vždy začínají sekvencí %ACC, a končí `\r\n` jsou tato data rozdělena na jednotlivé části oddělené znakem `';'` a po konverzi jsou uložena do lokální proměnné pro další zpracování, nebo jsou zobrazena na příslušném místě v okně programu.



Obr. 8.2: Stavový automat obsluhy sériového portu

9 ZÁVĚR

V práci je popsána realizace univerzálního mikrokontrolérem řízeného systému k měření a vyhodnocení zrychlení. Navrhl a realizoval jsem hardware měřiče, jenž je vybaven tříosým MEMS senzorem zrychlení, základním uživatelským rozhraním, pamětí pro záznam měření a bezdrátovým komunikačním modulem.

Pro elektroniku jsem navrhl v 3D CAD systému ochranný obal pro komfortnější obsluhu a možnost bezpečného testování. Obal jsem poté vytiskl na 3D tiskárně.

Zařízení jsem poté vybavil základním firmware pro obecné měření zrychlení, který jsem využil při hledání vhodného algoritmu pro počítání kroků - krokoměru. Po nasbírání několika vzorových měření s různou vzorkovací frekvencí jsem navrhl detekční algoritmus, jenž s určitou necitlivostí detekuje průsečíky křivky grafu filtrovaného signálu měřeného zrychlení a střední hodnoty tohoto signálu. Experimentálně jsem stanovil parametry pro filtraci i necitlivost tak, aby docházelo ke správné detekci za různých podmínek (běh, chůze, schody atd.).

Poté jsem obě měřicí úlohy integroval společně do finální verze firmware. Zařízení tedy může pracovat v režimu krokoměru, kdy výstupem z měření je počet vykonaných kroků. V režimu Akcelerometru zařízení vzorkuje výstup čidla v nastavené periodě a surové údaje ukládá do vestavěné paměti FRAM.

Vzhledem k nešťastné volbě zobrazovacího OLED panelu je uživatelské rozhraní pro zpracování výsledků přesunuto do aplikace pro Windows. Vytvořil jsem základní demonstrační úlohu pro načítání dat přímo ze senzoru i pro načítání uložených záznamů. Záznamy měření v režimu Akcelerometru je možné exportovat v CSV souboru pro další zpracování. Z prostředí aplikace lze také provést základní konfiguraci zařízení.

Práce provedené na projetu však považuji stále za začátek cesty k univerzálnímu zařízení, jež by v budoucnu mělo obsahovat nejen dvě již integrované úlohy. Je potřeba najít příčinu kolísajícího napájecího zdroje, také bude potřeba vyřešit problém displeje. Ve firmware je možnost rozšířit funkce stávajících úloh, např. přidání uživatelsky konfigurovatelné filtrace surových dat Akcelerometru, krokoměr by bylo zajímavé rozšířit o funkce pádového detektoru a monitoru celodenní aktivity. V případě rozšíření hardware o další senzory, např. gyroskop a barometr, by bylo možné se ubírat směrem k inerciální navigaci např. pro roboty.

Práce na tomto projektu byla velmi zajímavá, postupně se dařilo plnit jednotlivé body zadání a přitom jsem se naučil mnoho nového a zajímavého. Technologie MEMS a FRAM byly pro mne úplnou novinku, s mikrokontrolérem řady AVR jsem se nepotkal poprvé, ale tentokrát byl záběr projektu řádově širší. Dle mého názoru se podařilo naplnit jednotlivé body zadání, přesto budu pokračovat ve zdokonalování a rozšiřování funkcí zařízení, tak aby bylo reálně použitelné a přinášelo užitek.

LITERATURA

[1] Wikipedia, The Free Encyklopedia Wikipedia. *Microelectromechanical systems* [online]. [cit.2012-12-08] Dostupné z: http://en.wikipedia.org/wiki/Microelectromechanical_systems

[2] Analog Devices. *ADXL345:3-Axis, ±2 g/±4 g/±8 g/±16 g Digital Accelerometer*. [datasheet]. Norwood, U.S.A.: One Technology Way, ©2009-2011, Rev. C. Dostupné z: http://www.analog.com

[3] Atmel. *AT90USB647(646,1286,1287): 8-bit Atmel Microcontroller with 64/128Kbytes of ISP Flash and USB Controller*. [datasheet]. San Jose, U.S.A., Atmel Corporation, ©2012, v. 09/12. Dostupné z: http://www.atmel.com

[4] Analog Devices. *ADP2291: Compact, 1.5A Linear Charger for Single-Cell Li+ Battery*. [datasheet]. Norwood, U.S.A., One Technology Way, ©2005, Rev. A. Dostupné z: http://www.analog.com

[5] Linear Technology. *LTC3440: Micropower Synchronous Buck-Boost DC/DC Converter*. [datasheet]. Milpitas, U.S.A., Linear Technology Corporation, ©2007, Rev. B. Dostupné z: http://www.linear.com

[6] Analog Devices. *ADP1612: 650 kHz/1.3 Mhz Step-Up PWM DC-to-DC Switching Converters*. [datasheet]. Norwood, U.S.A., One Technology Way, ©2009-2012, Rev. D. Dostupné z: http://www.analog.com

[7] SOLOMON SYSTECH. *SSD1332: 96RGB x 64 Dot Matrix OLED/PLED Segment/Common Driver with Controller*. [datasheet]. Hong Kong, Solomon Systech Limited, ©2005, Rev. 1.6. Dostupné z: http://www.solomon-systech.com

[8] RAMTRON. *FM25V20:2Mb Serial 3V F-RAM Memory*. [datasheet]. Colorado Springs, U.S.A., Ramtron International Corporation , August 2012, Rev. 3.0. Dostupné z: http://www.ramtron.com

[9] EBAY. *HC-07, Slave mode Bluetooth UART RS232 serial Converter Module* [online]. [cit.2012-12-06] Dostupné z: http://www.ebay.com/itm/HC-07-Bluetooth-UART...

[10] CadSoft Computer GmbH and CadSoft Inc. [online]. http://www.cadsoftusa.com

[11] robotika.cz [online]. http://robotika.cz/guide/filtering/en

[12] RCEO - KLUZÁKY S ELEKTRICKÝM MOTOREM [online]. http://www.rceo.eu/Elektronika

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

AVR Označení 8bitových mikropočítačů firmy Atmel

DC/DC Direct Current, stejnosměrný proud, měniče DC/DC transformují vstupní stejnosměrné napětí na jinou hodnotu napětí s vysokou účinností

DPS Deska plošných spojů

EEPROM Electrically Erasable Programmable Read Only Memory - elektricky mazatelná a programovatelná permanentní paměť

FRAM Ferroelectric RAM (random-access memory) - feroelektrická operační paměť, „pamatuje si“ i po vypnutí napájení

I2C Inter-Integrated Circuit - nízkorychlostní multimaster sériová sběrnice

LED Light Emition Diode - světlo emitující dioda

LGA Land Grid Array - typ pouzdra integrovaných obvodů s vývody pouze na spodní straně, nevyčnívají z pouzdra

Li-Pol Lithium-Polymer, lithiové akumulátory s polymerním dielektrikem

MEMS Micro-Electro-Mechanical Systems

MIPS Million Instruction Per Second - milion instrukcí za sekundu

MISO Master In, Slave Out - zařízení v režim Master tímto pinem data přijímá, v režimu Slave vysílá

MOSI Master Out, Slave In - zařízení v režim Master tímto pinem data vysílá, v režimu Slave přijímá

OLED Organic Light Emition Diode - organická světlo emitující dioda

QFN Quad Flat No Leads, viz LGA

RISC Reduced Instruction Set Computing - procesory s redukovanou instrukční sadou

SCK Serial Clock - hodinový signál

SPI Serial Peripheral Interface - vysokorychlostní sériová sběrnice

SS Slave Select - pin sloužící k přepnutí zařízení do režimu Slave, tedy příjmu

USART Universal Synchronous / Asynchronous Receiver and Transmitter - sériové rozhraní

USB Universal Serial Bus - sériové rozhraní používané v PC

SEZNAM PŘÍLOH