VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

FACULTY OF MECHANICAL ENGINEERING

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

DETEKCE SMĚRU OTÁČENÍ PALIVOVÉ PUMPY

THE DETECTION OF THE ROTATION DIRECTION OF A FUEL PUMP

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — MICHAL MATĚJÁSKO

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. ROBERT GREPL, Ph.D.

SUPERVISOR

BRNO 2013

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav mechaniky těles, mechatroniky a biomechaniky
Akademický rok: 2012/2013

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Michal Matějásko

který/která studuje v **bakalářském studijním programu**

obor: **Mechatronika (3906R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Detekce směru otáčení palivové pumpy

v anglickém jazyce:

The detection of the rotation direction of a fuel pump

Stručná charakteristika problematiky úkolu:

Práce se zabývá návrhem, výrobou a zprovozněním zařízení pro detekci směru otáčení DC motoru použitého jako palivové pumpy v nádržovém modulu. Směr otáčení motoru bude zjišťován na základě změny magnetického pole při jeho pohybu. Detekce směru otáčení bude použita ve výrobě při testování možné záměny napájecích kabelů motorku. Práce je realizována ve spolupráci s průmyslovým partnerem.

Cíle bakalářské práce:

1) Zjistěte jaké jsou na trhu dostupné hall sensory a popište případně experimentálně ověřte jejich vlastnosti. Na základě této rešerše vyberte nejvhodnější sensor.
2) Navrhněte DPS pro vyhodnocení signálů z pole hall sensorů (3x3 sensory). Použijte mikrokontrolér PIC, nejlépe s podporou automatického generování C kódu ze Simulinku v Kerhuel toolboxu.
3) Navrhněte algoritmus kalibrace zařízení a vyhodnocení směru otáčení. Experimentálně ověřte funkčnost na nádržových modulech různých typů.

Seznam odborné literatury:

[1] Valášek, M.: Mechatronika, Vydavatelství ČVUT 1995
[2] Mann, B.: C pro mikrokontroléry, Nakladatelství BEN, 2003
[3] Herout, P.: Učebnice jazyka C

Vedoucí bakalářské práce: doc. Ing. Robert Grepl, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2012/2013.

V Brně, dne 19.11.2012

L.S.

_____	_____
prof. Ing. Jindřich Petruška, CSc.	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c.
Ředitel ústavu	Děkan fakulty

Abstrakt

Tato práce se zabývá návrhem a zprovozněním zařízení pro detekci směru otáčení DC motoru použitého jako palivová pumpa v nádržovém modulu. Směr otáčení motoru je zjišťován na základě změny magnetického pole při jeho pohybu. Detekce směru otáčení bude použita ve výrobě při testování možné záměny napájecích kabelů motoru. Práce je realizována ve spolupráci s průmyslovým partnerem.

Abstract

This thesis describes the design and commissioning of a device for detection of the rotation direction of a DC fuel pump. The rotation direction is calculated on the basis of change of magnetic field during its movement. Detection of the rotation direction will be applied in manufacture at possible supply cables confusion testing. Thesis is carried out in cooperation with industrial partner.

Klíčová slova

detekce směru otáčení, DC pumpa, magnetická indukce, MAG3110, I^2C

Keywords

detection of the rotation direction, DC pump, magnetic flux density, MAG3110, I^2C

Bibliografická citace práce

MATĚJÁSKO, M. *Detekce směru otáčení palivové pumpy*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2013. 41 s. Vedoucí bakalářské práce doc. Ing. Robert Grepl, Ph.D..

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval sám na základě svých znalostí, snahy, dovedností, rad a pokynů vedoucího bakalářské práce. Uvedl jsem veškeré použité podklady a literaturu.

V Brně dne: …… ……...……..

Michal Matějásko

Poděkování

Chtěl bych poděkovat vedoucímu bakalářské práce doc. Ing. Robertu Greplovi, PhD. za nabídku zajímavého tématu práce. Dále pak Ing. Josefu Vejlupkovi za jeho cenné rady, ochotu a pozornost, kterou mi věnoval. Děkuji také své rodině za důvěru, kterou ve mne měla a podporu během studia.

OBSAH

1. Úvod

V dnešní době se stále více zvyšuje důraz na snižování výrobních nákladů. Jedním z mnoha způsobů, jak ztrátám zabránit u průmyslových podniků, je využívat kvalitní nástroje pro kontrolu výrobků jdoucích z produkce. Úspory vylepšením kontroly kvality jsou pak citelné zejména u technologicky náročných produktů.

V automobilovém průmyslu, kde důraz na chybovost produkce míří k *1 ppm* (parts per milion) jsou tyto nástroje naprostou nezbytností. Je-li neodhalená, chybná palivová pumpa již namontována v automobilu, mohou být náklady na nápravu velmi vysoké. Právě na tuto oblast se naše práce zaměřuje.

Cílem této práce je sestrojit zařízení pro určení směru otáčení palivových pump, které bude využito v průmyslové praxi. Zařízení bude součástí několika nástrojů výstupní kontroly. Návrh zařízení je přizpůsoben tak, aby bylo možno zařízení synchronizovat s dalšími měřicími přístroji v případě potřeby provozovatele.

K řešení problému lze přistoupit několika možnými způsoby, jako např. v [1]. Zvolili jsme metodu snímání změny magnetické indukce okolo palivové pumpy při průchodu elektrického proudu pumpou. K tomu jsme využili senzory Hallova napětí propojené s řídicí elektronikou tak, abychom byli schopni určit směr otáčení různých druhů stejnosměrných pump a signalizovat jej obsluze.

Nejdříve bylo potřeba vybrat dostatečně citlivý a na trhu dostupný senzor. Senzory jsme následně umístili na desku plošných spojů (dále jen DPS) do pole 3x3, což nám umožní při správném nastavení vyhodnotit směr u více rozměrových typů stejnosměrných palivových pump. Bylo potřeba také navrhnout DPS s mikrokontroléry (dále jen μC) pro zajištění komunikace a vyhodnocování dat ze senzorů. K návrhu DPS byl použit CadSoft EAGLE PCB Design Software.

Abychom zajistili správnou funkčnost hardwaru, bylo pro něj potřeba vytvořit software. Využili jsme prostředí Simulink s jeho nadstavbou Stateflow, což je uživatelsky velmi intuitivní a přehledné prostředí pro programování stavových zařízení. Ze Simulinku a další rozšiřující knihovny Kerhuel Toolbox jsme poté snadno vygenerovali aplikovatelný kód.

Vzhledem k tomu, že jsme již od počátku zamýšleli programovat μC s využitím Kerhuel Toolboxu, bylo potřeba vybrat takové μC, které jím jsou podporovány. Software obsluhuje komunikaci se senzory po sériové sběrnici I^2C, zajišťuje správný postup měření, vyhodnocuje data, určuje, kterým směrem se pumpa otáčí a signalizuje výsledky měření.

Práce je realizována ve spolupráci s průmyslovým partnerem **Robert Bosch, spol. s.r.o**.

2. Rešerše

Ještě než bylo možno přistoupit k jakékoliv realizaci, bylo potřeba ověřit proveditelnost. Základním problémem byla otázka, zdali je magnetická indukce okolo pumpy při průchodu proudu dostatečně veliká na to, abychom ji byli schopni měřit na trhu dostupnými Hallovými senzory, a to zejména s přihlédnutím k faktu, že tělo pumpy je kovové a bude zapouzdřeno v plastovém palivovém modulu.

Z toho plyne fakt, že bude potřeba měřit v určité minimální vzdálenosti. Bylo by však vhodné, aby kvalita měření nebyla zajištěna pouze v určitém místě, ale aby byl stanoven nějaký rozsah vzdáleností, ve kterém budeme schopni garantovat věrohodnost naměřených veličin.

Z triviální úvahy ohledně velikosti magnetické indukce okolo vodiče s proudem, kdy u 12V pump, které nám byly k práci dodány, a na které má být výsledné zařízení kalibrováno a jimiž protéká při otáčení proud až 2A, jsme po dosazení do vzorce obdrželi, pro vzdálenost 20cm, následující hodnotu magnetické indukci.

$$B = \frac{\mu \cdot I}{2 \cdot \pi \cdot d} = \frac{\mu_r \cdot \mu_0 \cdot I}{2 \cdot \pi \cdot d} = \frac{1 \cdot 4 \cdot \pi \cdot 10^{-7} \cdot 2}{2 \cdot \pi \cdot 0{,}2} = \boldsymbol{2mT} \quad (1)$$

Na základě této úvahy si můžeme velmi přibližně stanovit oblast, v jakých řádech by měly námi hledané senzory mít rozsah.

2.1 Hallův jev

Hallův jev je fyzikální fenomén, při kterém dochází k posunu vodivostních elektronů působením magnetické síly. V [2] je definován takto:

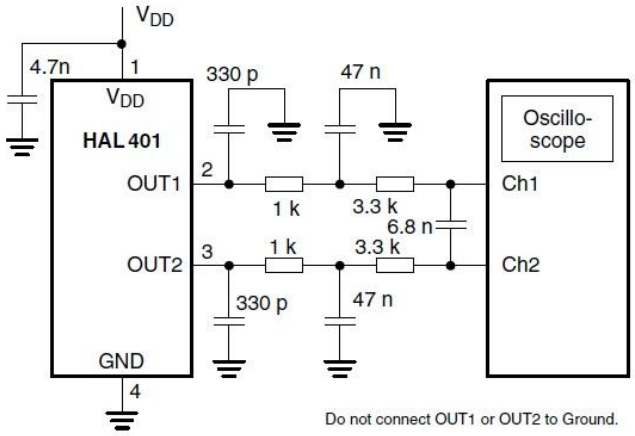
„Hallův jev je posun vodivostních elektronů ve vodiči, kterým prochází elektrický proud, působením magnetické síly a následně vznik příčného elektrického pole ve směru kolmém k vektoru magnetické indukce a ke směru elektrického proudu. Mezi protilehlými stranami vodiče vznikne rozdíl potenciálů, tzv. ***Hallovo napětí****.“*

Tento jev byl pojmenován po svém objeviteli, jímž byl Edwin Hall.

2.2 Senzor HAL401

Na základě výše uvedeného empirického odhadu, byl mezi cenově dostupnými senzory vybrán senzor společnosti Micronas HAL401. Jedná se o lineární senzor Hallova napětí s diferenciálním napěťovým výstupem. Výstupní napětí je přímo úměrné magnetické indukci. Uvedený měřitelný rozsah je -50mT - +50mT.

Dle následujícího zapojení z datasheetu jsme senzor zapojili.[3]



Obr. 1: Zapojení senzoru HAL401

Při následném spouštění pump a měření okolní magnetické indukce se výstupní diferenciální napětí oproti očekávání neměnilo. Byli jsme schopni naměřit pouze mírný šum. Rozdíl na výstupu byl patrný, až když jsme senzor přiblížili malému permanentnímu magnetu. Bylo tedy jasné, že bude zapotřebí citlivější senzor.

2.3 Senzor MAG3110

Jako další jsme vybrali senzor *MAG3110* výrobce *Freescale Semiconductor*. Jedná se o malý tříosý senzor magnetické indukce s nízkou spotřebou.

Přehled základních vlastností [4]:

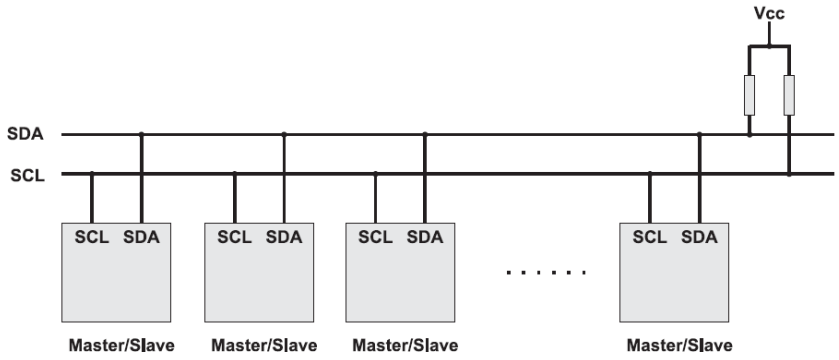
- Napájecí napětí (VDD) 1,95V až 3,6V
- Napájecí napětí rozhraní (VDDIO) 1,62V až VDD
- Velmi malé pouzdro 2mm x 2mm x 0,85mm
- Měřicí rozsah -1000μT až +1000μT
- Citlivost 0,1μT
- Rychlost snímání až 80Hz
- Výstupní rozhraní I^2C s rychlostí až 400kHz
- Několik integrovaných funkcí

Na rozdíl od předchozího senzoru je tento schopen měřit ve třech osách a s vysokou citlivostí. Byl vybrán také díky přijatelné ceně, která se v maloobchodech pohybuje okolo 50 – 60 Kč. Jak se později ukázalo, je jeho provedení také mnohem vhodnější pro námi zamýšlenou aplikaci.

Tento senzor využívá výstupního sériového rozhraní I^2C *(více v 2.4)*, díky kterému nemůžeme získat výstupy tak jednoduše jako u předešlého senzoru. Bylo tedy potřeba navrhnout testovací jednotku s μC *(kap. 3.1)*, který by senzor obsluhoval.

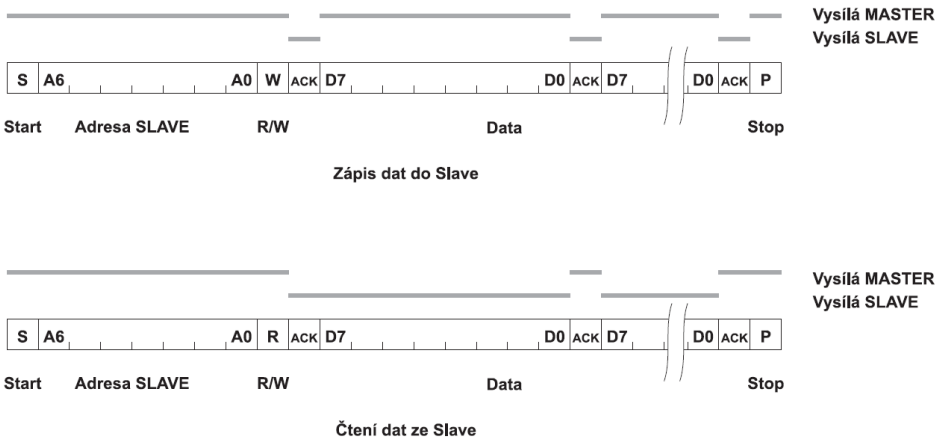
2.4 I²C sběrnice

I²C je sběrnice typu multimaster vyvinutá společností Philips. Jedná se o sběrnici, která umožňuje připojení několika slave a master zařízení. Obsahuje pouze dva vodiče, Serial Data Line (SDA) a Serical Clock Line (SCL). [5]



Obr. 2: Koncepce I²C

Každé zařízení připojené na I²C má svou 7 – 10 bitovou adresu pomocí níž k němu master zařízení přistupuje. Po zachycení signálu Start obvody porovnávají svou adresu s adresou vysílanou. Nalezne-li zařízení shodu, potvrdí bitem přijetí adresy a poté může být komunikace zahájena.



Obr. 3: Schéma komunikace dvou zařízení po I²C

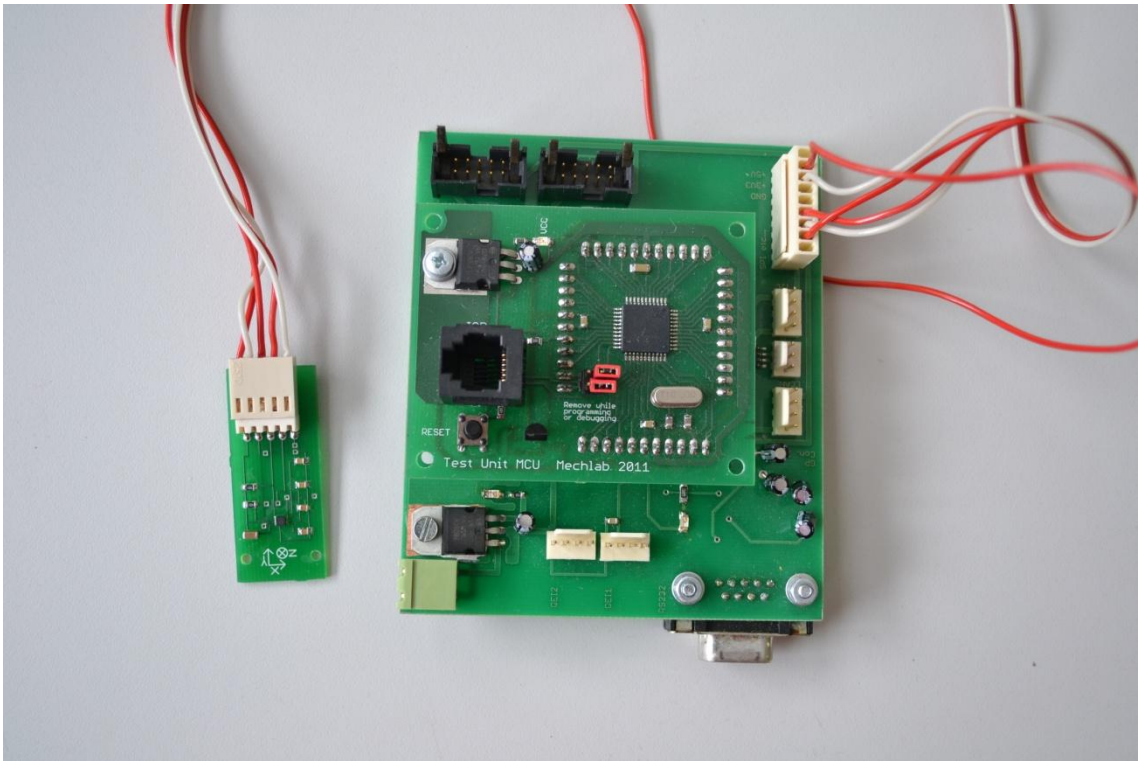
Nevýhodou je, že současně může pouze jeden master komunikovat s jedním slave zařízením. Dalším omezujícím faktorem je rychlost komunikace, standardně 100kHz nebo 400kHz, nejnověji však mohou dosahovat i rychlosti až 5MHz.

U aplikací, kde není vysoká rychlost natolik důležitá, je toto jednoduché rozhraní velmi dobře využitelné. Mírnou překážkou však může být, tak jako tomu je u naší aplikace, chceme-li připojit několik slave zařízení se stejnou adresou. V tom případě je potřeba volit komplikovanější řešení.

3. Vlastní realizace

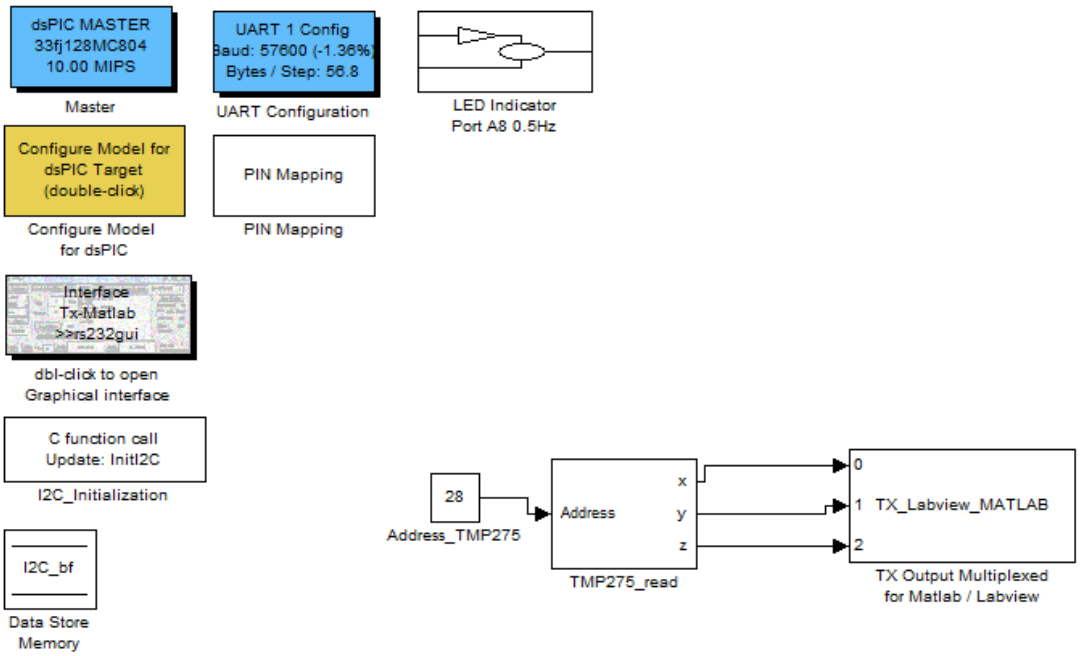
3.1 Testovací jednotka

Jako první bylo sestrojeno testovací zařízení s jedním senzorem MAG3110, které jsme připojili k již hotové platformě s µC, prostřednictvím které jsme byli schopni navázat komunikaci se senzorem. Naměřená data jsme posílali po UART do PC, kde jsme je mohli vykreslovat do grafu.



Obr. 4: Destička se senzorem (vlevo) připojená k platformě s µC

Vytvořili jsme model v Simulinku s využitím sady bloků z Kerhuel Toolboxu (více v kapitole 3.3.1), které nám umožňuji jednoduše vytvořit komunikaci mezi jednotlivými zařízeními. Program slouží ke čtení dat přes I^2C ze senzoru pomocí µC, do kterého jej nahrajeme. Ten je následně posílá po UART do PC, kde je vykreslujeme.



Obr. 5: Model v Simulinku pro čtení dat ze senzoru

Model obsahuje bloky *Master, UART Configuration, TX Output Multiplexed for Matlab/Labview* a *Graphical Interface*, které jsou součástí Kerhuel Toolboxu. O něm více v kapitole 3.3.1.

Dalšími bloky jsou:

„I2C_Initialization“
Volá funkci napsanou v C, která nastaví µC pro komunikaci po I^2C, tedy v našem případě se senzorem Hallova napětí.

„Data Store Memory“
Slouží k ukládání přicházejících po I^2C.

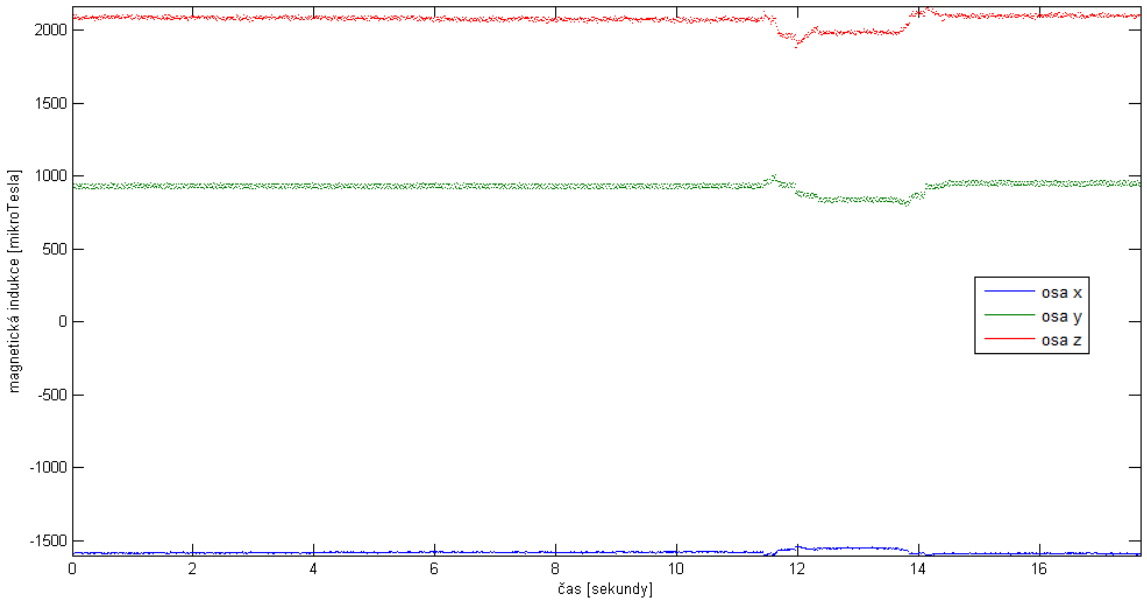
„TMP275_read“
Jedná se o subsystém, který obsahuje sadu bloků typu „*C function call“* obsluhujících příjem dat po I^2C ze senzoru a jejich následné interpretace do hodnot v jednotlivých osách.

Je potřeba zmínit, že pro komunikaci po I^2C (tedy bločky s voláním funkcí v jazyce C v našem modelu) jsme využili výsledků bakalářské práce Bc. Matěje Šimurdy [6].

Jakmile byl program hotov, mohli jsme započít měření. Výsledky byly pozitivní. Díky dostatečné citlivosti senzoru MAG3110 jsme byli schopni zaznamenat změnu magnetické indukce v okolí palivového modulu s pumpou ve všech třech měřených osách a to až do vzdálenosti 0,3m od modulu. Jelikož se jedná o vysoce citlivý senzor

reagující také na magnetické pole Země, bylo potřeba zajistit při měření jeho stabilitu upevněním do stojanu, protože změna náklonu nebo polohy senzoru způsobila výchylku hodnot magnetické indukce ve všech osách.

Při měření směru otáčení palivové pumpy jsme obdrželi hodnoty zobrazené v obr. 6.



Obr. 6: Data ze senzoru pro jednotlivé osy

Z obrázku je patrné, že hodnoty měřené magnetické indukce se pohybují stabilně v úzkém rozsahu hodnot, okolo určité průměrné hodnoty. Při otáčení pumpou (první zub na křivce naměřených dat) následovalo ustálení hodnot okolo odlišné průměrné hodnoty vlivem magnetizace součásti po průchodu proudu. Při otočení pumpou opačným směrem (druhý zub) se hodnoty opět vychýlily. Podstatným zjištěním je, že při průchodu proudu pumpou byly hodnoty vždy poblíž určité hodnoty a ta se lišila podle směru otáčení.

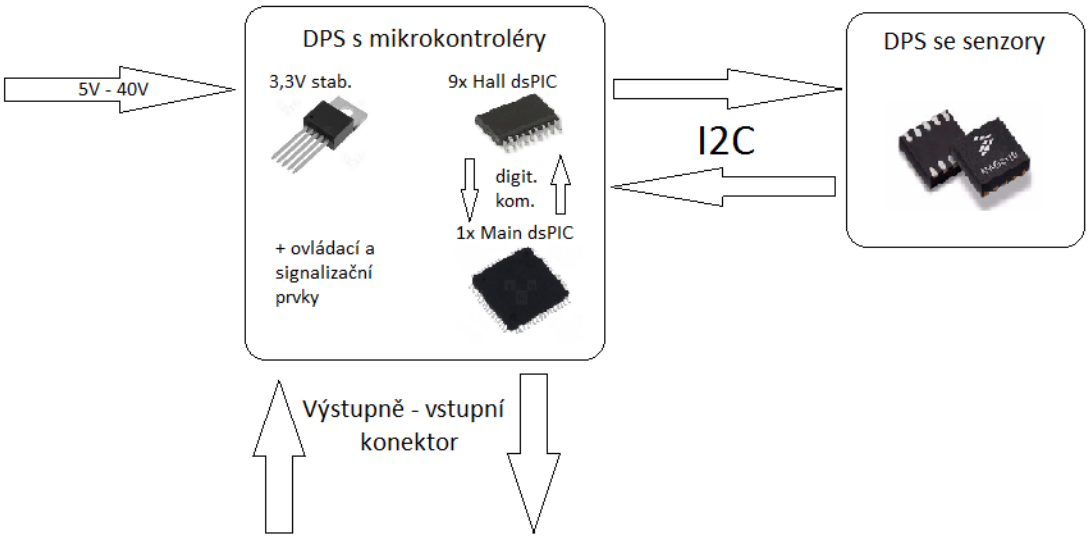
Z měření plynou dvě zjištění. První, že použitý senzor je vhodný k měření požadovaných veličin s dostatečnou přesností a můžeme jej tedy využít pro naše zařízení. Druhé, že budeme moci ze změny magnetické indukce určit, kterým směrem se pumpa otáčí.

Na základě provedené rešerše jsme zjistili, že je možné řešit zadaný úkol pomocí na trhu dostupných senzorů magnetické indukce. Jako vhodný senzor jsme vybrali MAG3110 výrobce *Freescale Semiconductor*, se kterým jsme na testovací jednotce dosáhli uspokojivých výsledků

3.2 Elektronika navrženého zařízení

Elektronika výsledného zařízení je složena ze dvou hlavních částí. Těmi jsou DPS se senzory Hallova napětí a DPS s µC, LED diodami a uživatelskými prvky. Tyto dvě části zařízení je zamýšleno umístit nad sebou. Spojení mezi DPS zajišťuje čtyřiatřiceti pinový konektor. Na přední straně zařízení se nachází devět LED diod sloužící k signalizaci výsledku vyhodnocení směru otáčení.

Řízení zařízení stejně jako komunikaci se senzory a vyhodnocování směru otáčení zajišťuje deset µC dsPIC od firmy Microchip.



Obr. 7: Blokové schéma zařízení

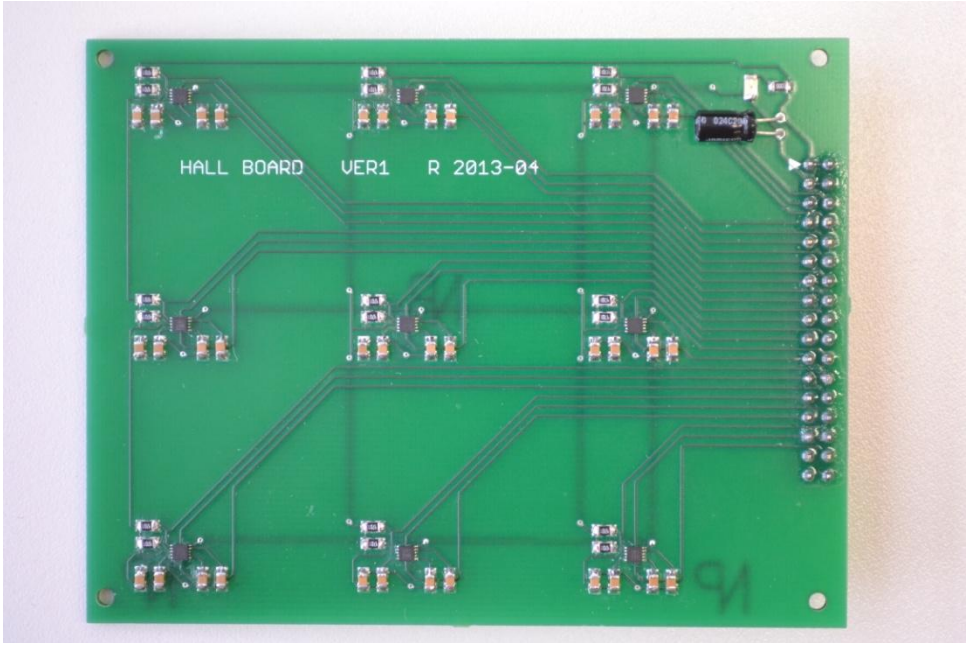
Napěťový vstup jsme navrhli takový, aby umožňoval připojení napájení více způsoby. V budoucnu můžeme jako zdroj využít baterii, avšak během vývoje jsme využili napájení ze sítě pomocí malého síťového transformátoru.

Vzhledem k požadavku, aby byl vyhodnocován stav od jednotlivých senzorů samostatně, je vhodné rozmístit signalizační LED diody tak, aby bylo obsluze patrné, od kterého senzoru přišel jaký výsledek.

K návrhu elektronických obvodů byl využit CAD program Eagle. Nezbytnými zdroji informací byly specifikace obou µC [7] a [8].

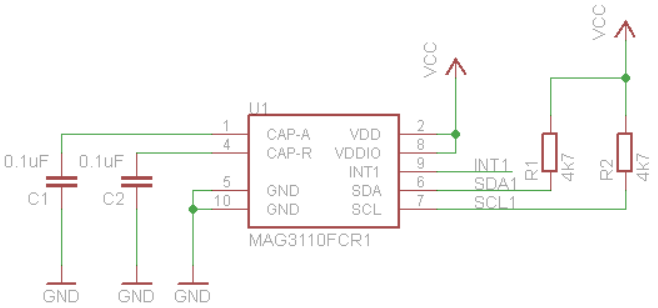
3.2.1 DPS s Hallovými senzory

Vzhledem k množství součástek, které bude naše zařízení obsahovat, jsme se rozhodli jeho senzorickou část umístit na zvláštní DPS. Další z důvodů přispívající k tomuto rozhodnutí je fakt, že elektromagnetické děje probíhající v okolních elektronických součástkách by mohly negativně ovlivnit měření.



Obr. 8: DPS se senzory

Senzory Hallova napětí jsou rozmístěny v poli 3x3 a vzdáleny od sebe navzájem 3cm. U senzorů jsou rozmístěny čtyři kondenzátory, dva filtrační a dva pro potřeby senzoru samotného (viz. Obr. 9) a pull-up rezistory na sběrnici I^2C. Více o I^2C sběrnici v kapitole 2.4. Všechny signály jsou svedeny do konektoru tvořícího spojení s druhou DPS. Ke konektoru je umístěn elektrolytický filtrační kondenzátor, který by měl pohltit jakékoliv nežádoucí napětí naindukované na kabeláži nebo cestičkách vedení. Připojena je také smd LED dioda značící přívod elektrické energie na desku.



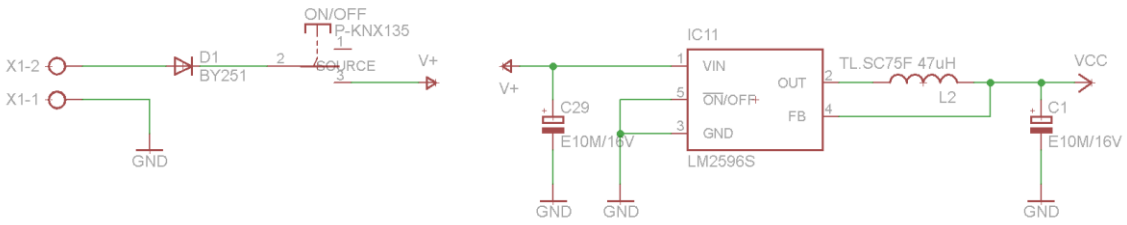
Obr. 9: Zapojení senzoru Hallova napětí

3.2.2 DPS s mikrokontroléry

Na druhé DPS se nachází všechny ostatní potřebné součásti celého našeho zařízení. Podstatnou část tvoří 10 µC, jež jsou mozkem zařízení. Devět (Hall µC) k obsluze senzorů a jeden hlavní (Main µC), zajišťující zejména spouštění měření a komunikaci s vnějším světem. Dále pak LED diody signalizující stavy, ve kterých se zařízení nachází a stav vyhodnocení směru otáčení, atp.

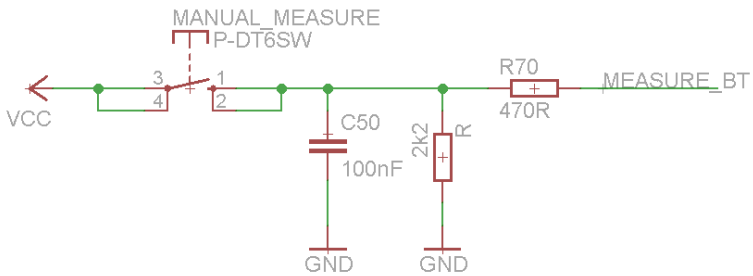
Napájení je na DPS přivedeno konektorem typu AK500/2-H. Ten nám umožňuje snadno připojit zdroj elektrické energie, ať už bateriový, či jiný. Na napájecí napětí je připojena dioda sloužící jako ochrana proti přepólování, a dále páčkový spínač sloužící k zapnutí zařízení.

Přivedené napětí je ještě potřeba stabilizovat na námi požadovanou úroveň 3,3V vhodnou pro všechny součástky, které využijeme. K tomu slouží snižující napěťový stabilizátor LM2596-3.3 [9] schopný dodávat až 3A při velké šíři vstupního napětí, a to od 4,75V do 40V. Na vstup a výstup stabilizátoru jsme přidali elektrolytické kondenzátory k filtraci průběhu napájení.



Obr. 10: Schéma zapojení napájení na DPS

Na zařízení je umístěno tlačítko na manuální spuštění měření (měření lze rovněž spouštět softwarově). Nezbytností jsou konektory pro programování µC, konektor propojující obě DPS a výstupní konektor.

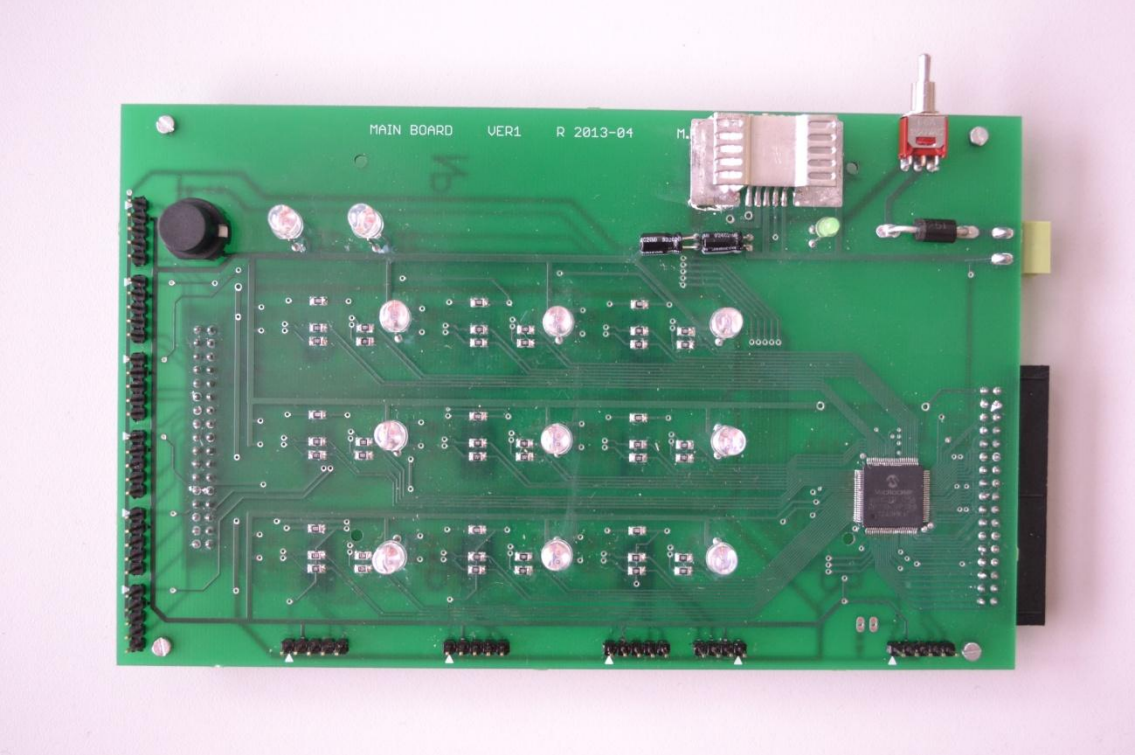


Obr. 11: Schéma zapojení tlačítka pro manuální spoušť měření

Jak již bylo zmíněno v úvodu, jsou na DPS s µC rozmístěny také LED diody signalizující pracovní stavy zařízení a výsledky měření. Kvůli úspoře místa jsme zvolili dvoubarevné diody se společnou anodou. Připojili jsme je k µC tak, že nožička µC slouží jako drain[1], což jej nebude natolik zatěžovat. Nutí nás to však změnit logiku spínání diody, kdy pro sepnutí bude nutno na příslušný pin zapsat logickou nulu namísto jedničky.

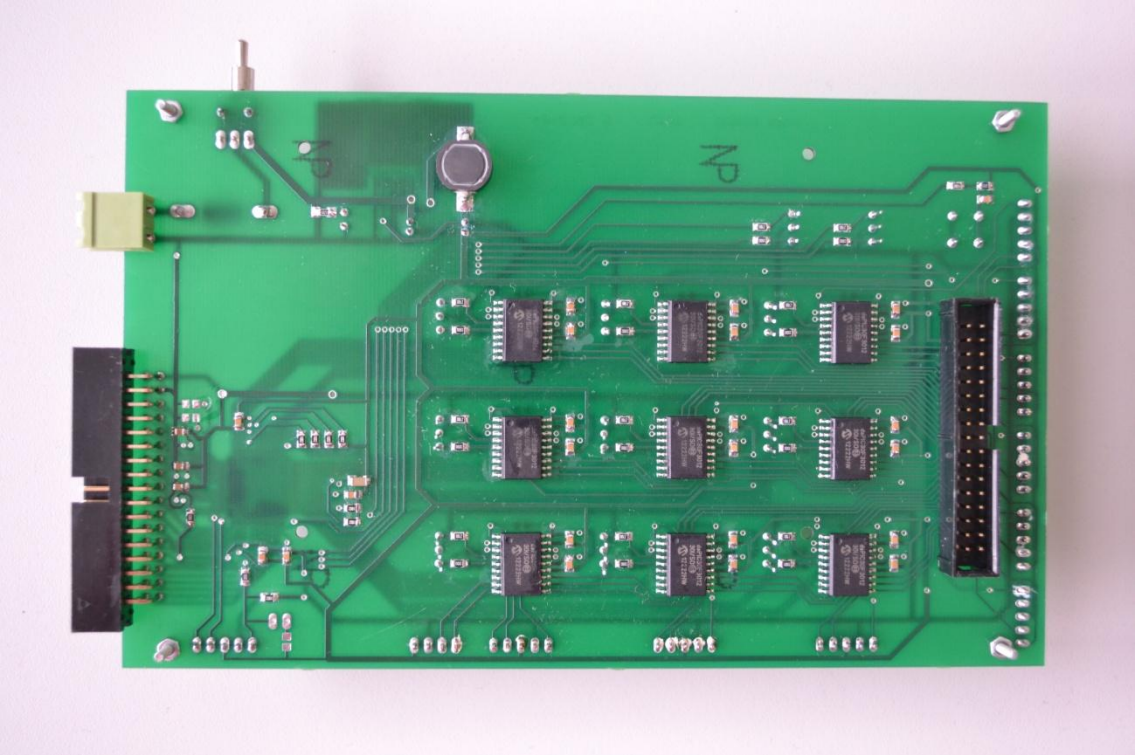
Diody jsou rozmístěny do mřížky 3x3 stejně jako senzory. Otáčí-li se pumpa požadovaným směrem a je tedy správně zapojena, rozsvítí se zelená barva. V opačném případě se na LED diodě rozsvítí červená. Vyhodnocení z každého senzoru je nezávislé na ostatních. Každá jedna LED dioda odpovídá jednomu senzoru na druhé straně zařízení a to tak, že součástky patřící k sobě leží nad sebou.

Zbývající dvě LED diody signalizují blikáním stavy, ve kterých se zařízení právě nachází. Různé způsoby blikání uživateli napovídají, co se v zařízení děje - zdali je v pohotovosti, měří, nebo jestli se vyskytla chyba apod.



Obr. 12: Přední strana DPS

[1] drain – proud teče dovnitř



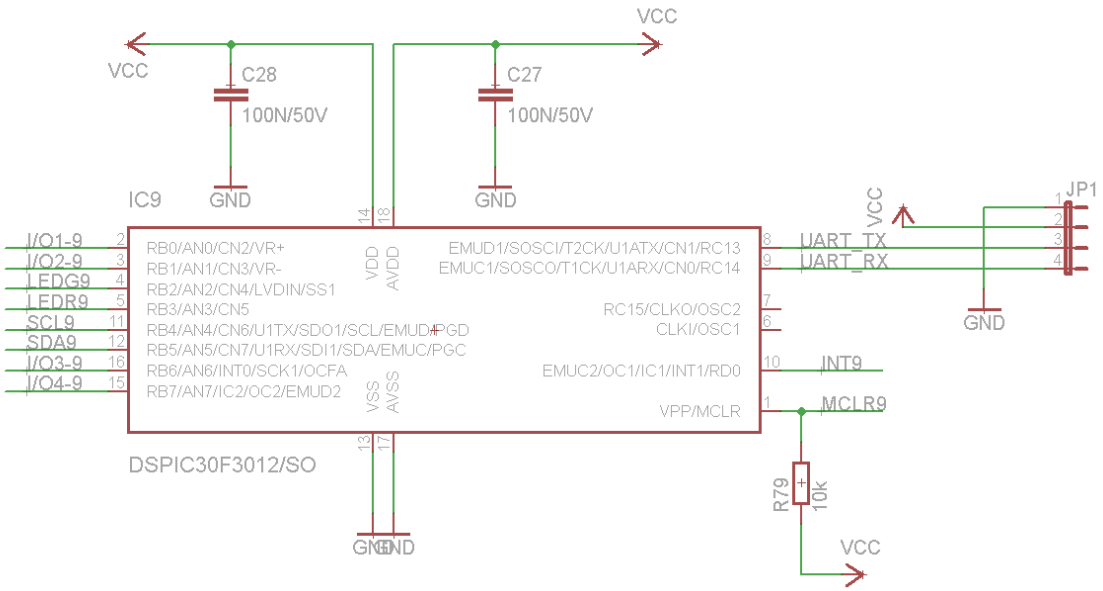
Obr. 13: Zadní strana DPS

3.2.3 Hall mikrokontrolér

Abychom zajistili čtení dat a jejich vyhodnocování ode všech devíti senzorů najednou, použijeme devět µC, z nichž každý jeden obsluhuje právě jeden senzor nezávisle na ostatních. Nezávislé vyhodnocení dat ze senzorů bylo v požadavcích zadavatele.

Na základně požadavků jsme vybrali µC dsPIC30F3012. Jedná se o µC s šestnácti bitovou vylepšenou Harvardskou architekturou a podporou ze strany Kerhuel Toolboxu, což je pro nás stěžejní vlastnost, jelikož uvažujeme programování generováním kódu ze Simulinku.

Důležitá je podpora I^2C rozhraní, po kterém potřebujeme komunikovat se senzory. Deklarovaných 24kB flash paměti by mělo tvořit dostatečný prostor pro zamýšlený program. Vzhledem k tomu, že není potřeba zajistit zcela přesné a rychlé časování, budou interní oscilátory s frekvencí 7,37MHz nebo 512kHz poskytovat dostatečný výpočetní výkon s příslibem nízké spotřeby.



Obr. 14: Zapojení Hall µC na DPS

Tyto µC mají za úkol zejména obsluhu senzorů a signalizaci výsledků pomocí připojených LED diod. Komunikují také s hlavním µC pomocí čtyř digitálních pinů. Přehled funkcí na jednotlivých pinech přináší tabulka na další straně.

Číslo pinu	Označení signálu	Funkce
1	MCLR	Programovací pin, vymazání paměti
2	I/O1	Digitální vstup/výstup
3	I/O2	Digitální vstup/výstup
4	LEDG	Signál k LED diodě - zelená barva
5	LEDR	Signál k LED diodě - červená barva
6	-	nevyužitý pin
7	-	nevyužitý pin
8	UART_TX	UART, vysílací pin (pouze µC č. 9)
9	UART_RX	UART, přijímací pin (pouze µC č. 9)
10	INT	Interrupt signál k senzorům
11	SCL	Serial Clock (I^2C)
12	SDA	Serial Data (I^2C)
13	GND	Napájení, zem
14	VCC	Napájení, napětí
15	I/O4	Digitální vstup/výstup
16	I/O3	Digitální vstup/výstup
17	GND	Napájení, zem
18	VCC	Napájení, napětí

Tabulka 1: Signály na pinech Hall µC

Jak je patrné z tabulky 1 a schématu zapojení (obr. 14), vyvedli jsme z µC s označením devět signály pro UART na konektor. Toto nám umožní odladit program µC ve fázi programování a kalibrace a podrobně sledovat dění v něm, což je velmi důležité. Nemůžeme předpokládat, že vše bude fungovat dle našich představ hned napoprvé. Bez možnosti bližšího nahlédnutí do dění uvnitř bychom chybu hledali velice obtížně.

3.2.4 Hlavní mikrokontrolér

Nejvýznamnějším kritériem pro výběr hlavního µC byla potřeba obsluhovat všechny Hall µC a mít dostatečný počet pinů na výstupní konektor ze zařízení. Neméně důležitou potřebou byla podpora ze strany Kerhuel Toolboxu. Volba padla na sto pinový dsPIC33FJ256GP710. Tento µC poskytuje dostatečnou rezervu pinů i výpočetní výkon pro možná rozšíření.

K µC je přivedeno všech 9x4 digitálních pinů od Hall µC, na jejichž cestičce jsou umístěny 1k2 rezistory, aby byla jasně definována logická hodnota a nedocházelo k neočekávaným stavům. Připojeno je také tlačítko pro manuální spuštění měření a dvě LED diody. Na konektor ven ze zařízení je vyvedeno 30pinů. Zbývající nevyužité piny jsou přes 10k odpor připojeny k zemi, softwarově nastaveny jako výstupy a uvedeny do nízkého logického stavu.

3.3 Software navrženého zařízení

Velkou částí práce je také software. Bylo by možné jej psát v jazyce C nebo přímo v jazyce symbolických adres (assembler), což by bylo zdlouhavé. Díky moderním vývojovým prostředkům jako je Simulink jsme schopni návrh urychlit a uživatelsky zpříjemnit. Nakonec jsme se však psaní kódu v C nevyhnuli.

K návrhu obslužných programů μC jsme se rozhodli použít generování C kódu z prostředí Simulink. Díky nástroji Kerhuel Toolbox a jeho podpoře pro μC firmy Microchip, jsme schopni snadno aplikovat kód se všemi potřebnými nastaveními bez dalších zásahů. V Simulinku integrované grafické prostředí Stateflow navíc snadno a intuitivně umožňuje programovat zařízení, která fungují stavově, což je náš případ.

3.3.1 Kerhuel Toolbox

Jedná se o rozšiřující knihovnu pro Simulink, speciálně zaměřenou na podporu μC firmy Microchip. Obsahuje v sobě již předpřipravené funkce (bločky) s možnostmi nastavení cílového μC. Není proto potřeba podrobně studovat jeho registry a příkazy, čímž se vývoj značně urychlí. Toolbox obsahuje tyto typy bločků:

- **Konfigurační** – obsahují nastavení parametrů μC a kompilátoru
- **Komunikační** – nastavení rozhraní (UART, I^2C, SPI, CAN)
- **I/O** – vstupně/výstupní periferie (např. digitální porty, převodníky ADC,PWM, apod.)
- **Ostatní** – volání C funkcí, softwarový reset, atd.

Mezi nejdůležitější bloky patří:

Master
Musí být obsažen v každém modelu určeném ke generování kódu pro již zmíněné μC. Obsahuje množství nastavení jako nastavení oscilátoru, časovače, přímé zadání počtu instrukcí, které se mají provést za jednu vteřinu, a dalších podle typu použitého μC.

Digital Output Write
Umožňuje nastavit digitální piny jako výstupní a na ty zapisovat data typu *boolean*[2]. V nastavení bloku se vybere požadovaný port a zadá se číslo pinu daného portu, který takto chceme nastavit. Takto lze jedním bločkem nastavit i více pinů. Jejich čísla se pak zadávají ve vektorovém tvaru.

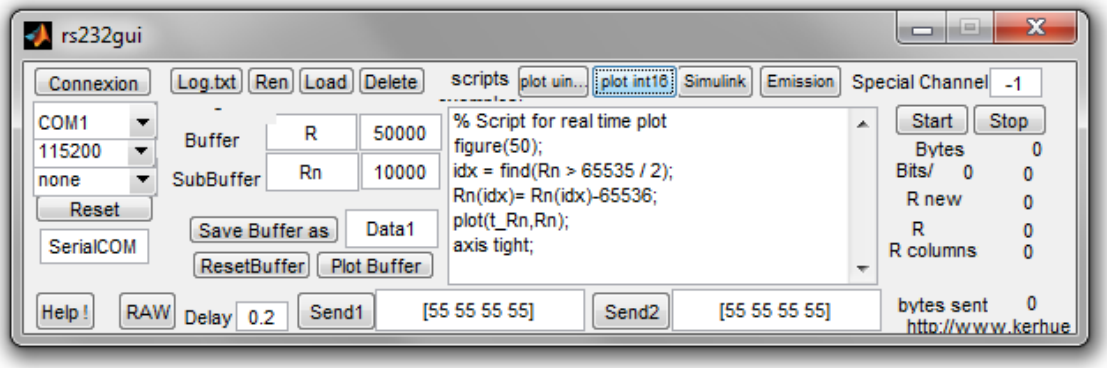
Digital Input
Obdobně jako u výše uvedeného s tím rozdílem, že tento bloček nastaví pin jako vstupní a přečte z něj logickou hodnotu.

[2] boolean – datový typ reprezentován hodnotami **true** nebo **false**

UART configuration
Nastavuje parametry UARTu. Zejména přenosovou rychlost. Rovnou také spočítá, kolik Bytů je při dané rychlosti schopen za jednu vteřinu přenést.

TX Output Multiplexed for Matlab/Labview
Slouží k propojení µC s prostředím Simulink nebo Labview skrze sériový port. Lze nastavit až 16 kanálů pro příjem dat, a to buď osmi, nebo šestnácti bitových. Tento bloček je propojen s následujícím bločkem.

Graphical interface
Uživatelské rozhraní propojené s bločkem „*TX Output Multiplexed for Matlab/Labview*“. Před zahájením komunikace je potřeba nastavit rychlost na stejnou jako v bločku „UART configuration“ a vybrat správný sériový port. Po kliknutí na tlačítko „*Connexion“* se naváže spojení a po stisku tlačítka „Start“ v pravé části okna začne přijímat data. Přijatá data lze pak snadno vykreslovat pomocí přednastavených skriptů, volajících funkci *plot* z Matlabu. Odsud lze na UART data také posílat.



Obr. 15: Otevřené Graphical interface

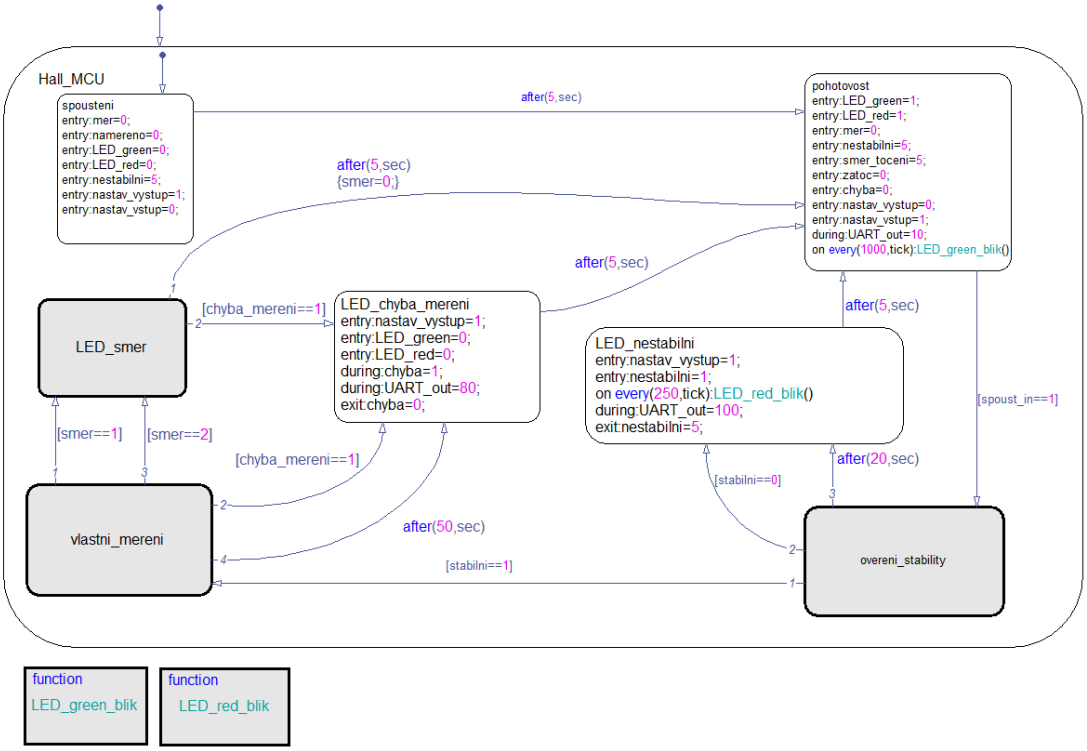
C function call
Tento bloček volá funkci v jazyce C při generování kódu ze Simulinku. Do bločku se zadá jméno funkce, kterou chceme zavolat a můžeme nastavit její funkčnost. Důležité je mít danou funkci v pracovním adresáři. Jméno C souboru, ve kterém je obsažena, musí být zahrnuto v menu *Configuration Parameters → Code Generation → Custom Code*.

3.3.2 Program Hall µC

Jak již bylo řečeno, pro řízení procesů v µC využijeme stavového řízení, které naprogramujeme s pomocí knihovny Stateflow v Simulinku. Program v těchto µC bude obsluhovat komunikaci se senzorem, přijímat a vyhodnocovat data, signalizovat výsledek vyhodnocení a potřebné signály zasílat na hlavní µC pomocí kombinací čtyř digitálnch pinů.

Obrázek celého modelu programu v Simulinku je součástí přílohy na straně 33.

Obsah hlavní řídící části programu, tedy bločku *Chart* je na následujícím obrázku.



Obr. 16: Stateflow diagram řídící chod programu Hall µC

Na obrázku vidíme, jaké stavy se v programu Hall µC objevují, co je jejich obsahem, jaké jsou podmínky přechodů do jiných částí programu a které akce se kdy provádějí. V dolní části jsou umístěny Stateflow funkce zajišťující blikání LED.

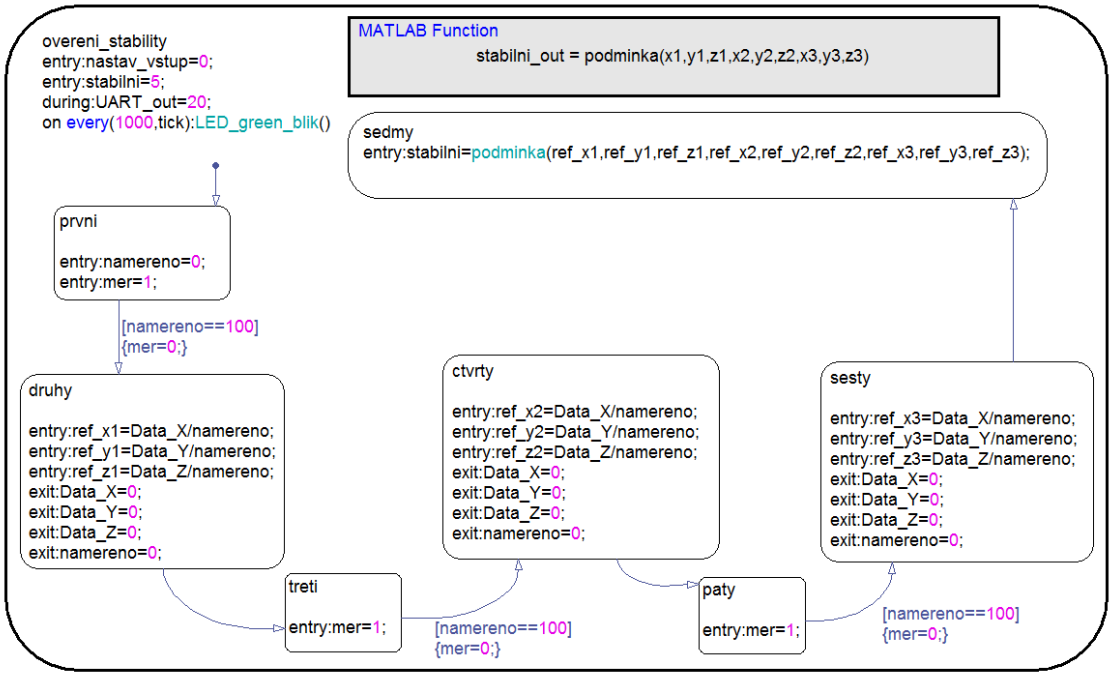
Ihned po spuštění se program uvede do úvodního stavu, jenž je poznačen ukazatelem s tečkou v levé horní části diagramu. V tomto případě je stav pojmenován „*spousteni*“. Tento stav je zde umístěn proto, aby zařízení mělo čas uvést se do provozu. Šipky mezi jednotlivými grafickými bloky značí směry přechodu mezi stavy. Na těchto přechodech můžeme definovat podmínky, při jejichž splnění se přechod má vykonat, ale také akce, které se vykonají ihned po splnění podmínky ještě před přechodem do jiného stavu.

Příkaz *entry* obecně provede požadovanou akci vždy při vstupu do stavu. Příkaz *during* vykonává zadanou akci každý časový krok. Pro potřeby ladění programu jsme nastavili, aby v každém stavu, ve kterém se program nachází, posílal na UART jiné číslo. To nám umožní podrobně sledovat děj uvnitř.

Ve stavu *„spousteni“* proběhne nastavení některých proměnných. Po nastavených 5 vteřinách přejde program do stavu *„pohotovost“*. Tento stav slouží jako výchozí funkční stav zařízení, v němž čeká na signál pro zahájení měření. Aby tento signál mohl obdržet, je potřeba nastavit digitální piny pro komunikaci s Main µC jako vstupy. Nastavení pinů provedeme zapsáním hodnoty 1 do proměnné *nastav_vstup*, která spustí subystém[3] jenž nastavení provede. Více o fungování komunikace pomocí digitálních pinů v kapitole 3.3.5.

Přijde-li od hlavního µC signál k zahájení měření, tj. na vstupu se objeví kombinace pinů potvrzující tento signál, zapíše se do proměnné *„spoust_in“* jednička. Tím je splněna podmínka přechodu do stavu *„overeni_stability“*.

Stav *„ověřeni_stability“*, jak název napovídá, slouží k ověření, zdali se zařízení, respektive pole senzorů, nachází v klidové poloze. Klidová poloha je pro zajištění správnosti měření nezbytností a je ji vždy nutno ověřit. Stav *„overeni_stability“* vypadá v našem programu následovně:

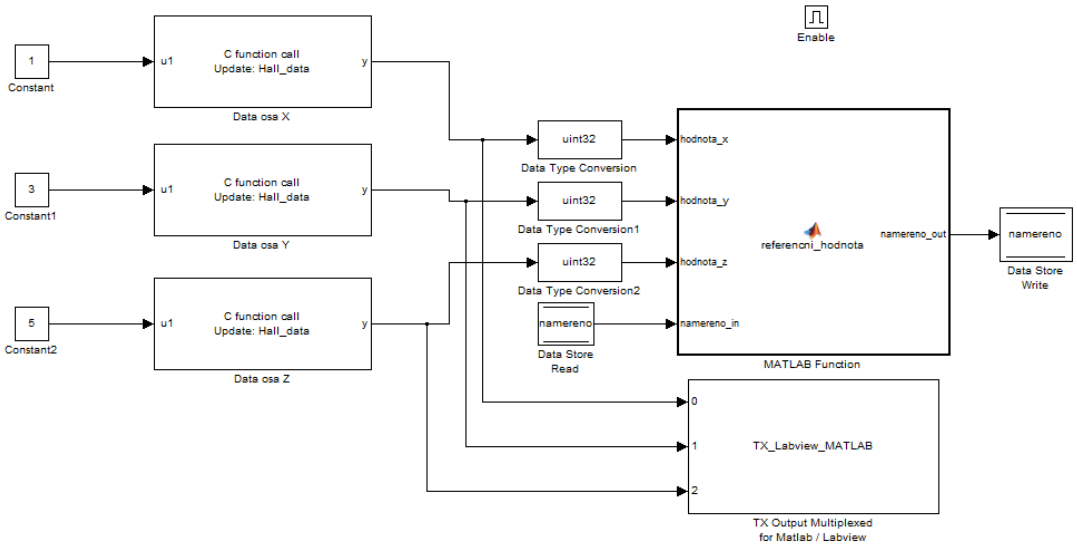


Obr. 17: Stav „overeni_stability“

V tomto stavu vstoupí program do podstavu jménem *„prvni“*. V něm se nastaví lokální proměnná *namereno* do hodnoty nula. Do této proměnné se zapisuje počet měření ze senzorů. Jako další v pořadí se zapíše do proměnné *mer* hodnota 1, což způsobí, že se v modelu [str. 33] aktivuje subsystém a µC započne přijímat data ze senzorů.

[3] subsystém – dílčí systém, podsystém

Není potřeba, aby čtení dat ze senzorů probíhalo neustále, a proto jsme se rozhodli umístit funkce k obsluze příjmu dat do zvlášť spouštěné programové sekce, v našem případě reprezentované bločkem *Enabled Subsystem*. Takto můžeme příjem dat realizovat pouze ve chvílích, kdy je potřeba. Zmiňovaný subsystém vypadá takto:



Obr. 18: Enabled Subsystem – příjem dat ze senzorů

Uvnitř najdeme bločky *C function call* jež volají kód obsluhy komunikace po I^2C napsaný v jazyce C *(více v kapitole 3.3.3)*. Z nich dostáváme 16-bitovou hodnotu magnetické indukce z jednotlivých os. Konstanty na vstupu do těchto bločků říkají proceduře, jaký registr má obsloužit. Tyto hodnoty, poté pomocí *Data Type Conversion* bločků převádíme na hodnoty 32-bitové, a to z důvodu toho, že následně chceme sečíst 100 hodnot, a potřebujeme zajistit, aby se nám výsledné číslo vešlo do jedné proměnné. Ačkoliv používáme 16-bitový µC, lze použít i 32-bitovou hodnotou čísla.

V navazujícím bločku *MATLAB function*, pomocí následujícího jednoduchého kódu v syntaxi Matlabu, tato data sčítáme a zapisujeme do globálních proměnných pro jednotlivé osy a také počítáme počet provedených měření.

ZAČÁTEK KÓDU:

```
function namereno_out =
referencni_hodnota(hodnota_x,hodnota_y,hodnota_z,namereno_in)
%#codegen

global Data_X                   % deklarace globální proměnné
global Data_Y                   % deklarace globální proměnné
global Data_Z                   % deklarace globální proměnné

Data_X = Data_X + hodnota_x;    % sčítání dat a zapisování do proměnné
Data_Y = Data_Y + hodnota_y;    % sčítání dat a zapisování do proměnné
Data_Z = Data_Z + hodnota_z;    % sčítání dat a zapisování do proměnné

namereno_out = namereno_in + 1; % počítadlo provedených měření
end
```

KONEC KÓDU

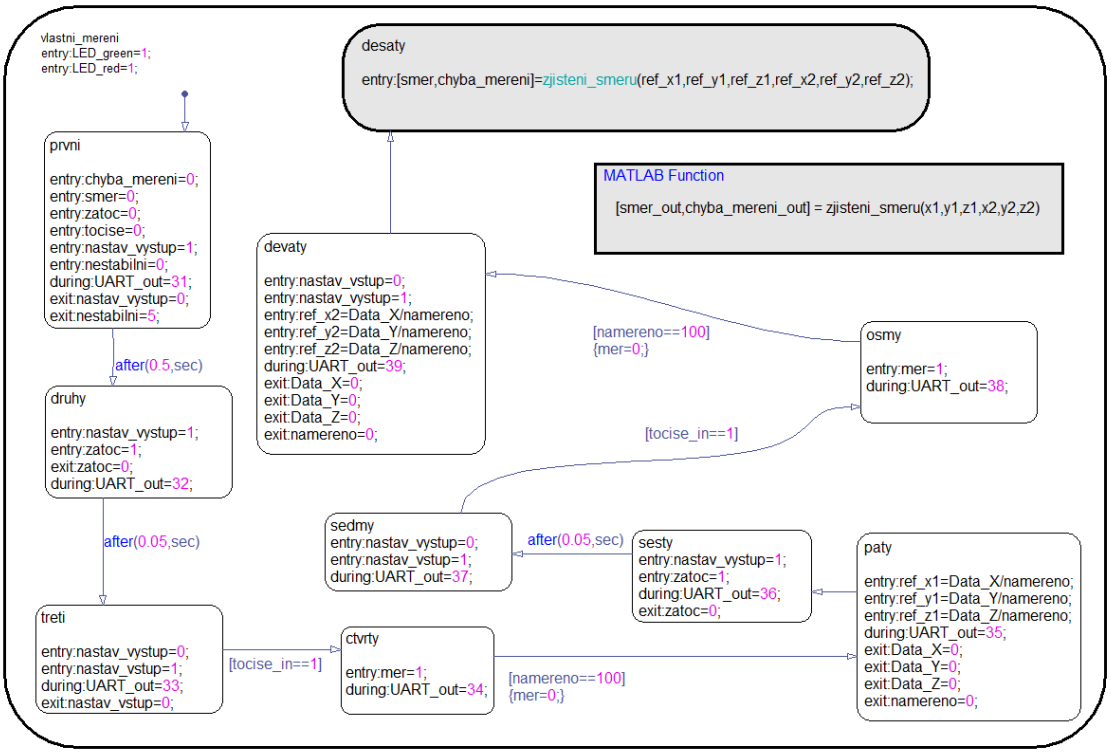
Zpátky ve stavu *„overeni_stability“*, po dosažení proměnné *namereno* na hodnotu 100, se do proměnné *mer* zapíše 0, čímž se ukončí příjem dat ze senzorů a program přejde do stavu jménem *„druhy“*. Zde se spočte průměrná hodnota magnetické indukce v jednotlivých osách, zapíše do příslušné proměnné a na výstupu z bločku se použité globální proměnné vynulují.

Celý tento proces proběhne celkem třikrát, čímž dostaneme tři referenční hodnoty, které následně budeme mezi sebou porovnávat ve stavu jménem *„sedmy“*, který volá další funkci *podminka* psanou v syntaxi Matlabu. Měření těchto tří referenčních hodnot provádíme, abychom zjistili, že je zařízení v klidové poloze. Jak lze vidět na obrázku 6 v kapitole 3.4, i když jsou senzory v klidové poloze, data ze senzorů zaznamenávají jistý šum. Právě z tohoto faktu vyplynula potřeba počítat průměrnou hodnotu. Celý tento proces zabere přibližně tři sekundy.

Na výstupu ze stavu *„overeni_stability“* dostáváme na základě vyhodnocení rozdílů mezi referenčními hodnotami z funkce *podminka* buď, že je zařízení v klidové poloze (*stabilni=1),* nebo je v nestabilní poloze *(stabilni=0)*. Pro případ, že by nastala neočekávaná situace a nebyli jsme schopni určit stabilitu (např. kdyby došlo k odpojení senzorů, senzor by přestal pracovat apod.), je přidán ještě jeden přechod, který zajistí, že po 20s µC zahlásí chybu.

Nastane-li případ, že zařízení je nestabilní, přejde program do stavu *„nestabilita“*. Zde zapíše do proměnné *nestabilni* jedničku, čímž vyšle signál hlavního µC. Zároveň pomocí LED diody signalizuje tento stav a po čase přejde zpět do pohotovostního režimu.

Je-li zařízení stabilní, program přejde do stavu *„vlastni_mereni“*, kde proběhne fáze zjišťování směru otáčení palivové pumpy. Stav je napsán podobným stylem jako předchozí a vypadá takto:



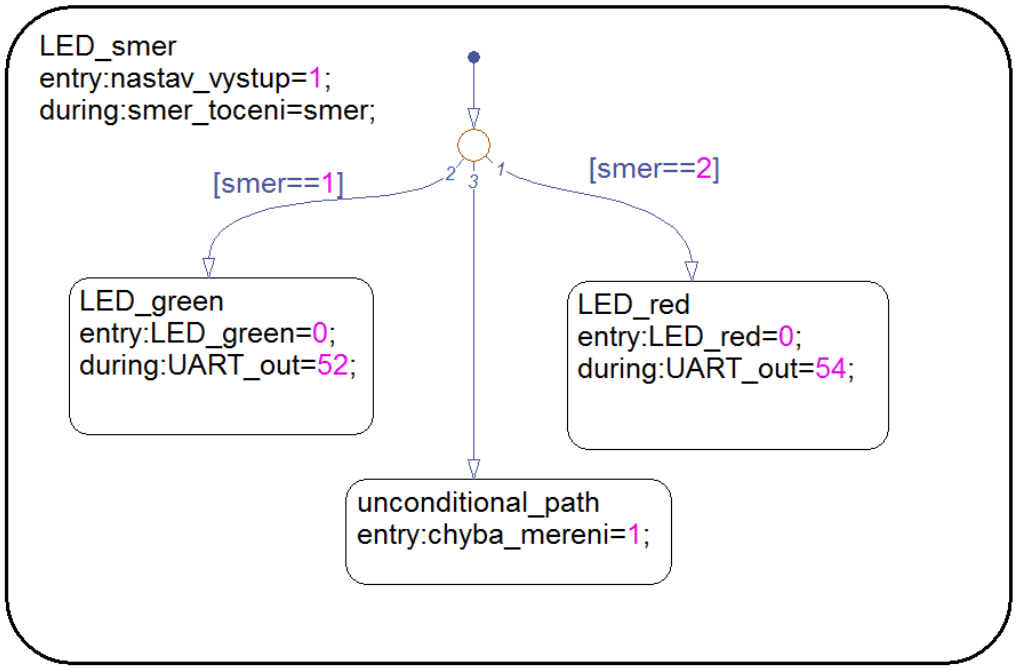
Obr. 19: Stav „vlastni_mereni“

Při vstupu do tohoto stavu se nastaví dále používané proměnné a proměnná *nastav_vystup*, pomocí níž zapneme subsystém pro nastavení digitálních pinů jako výstupy. Program vyšle signál *nestabilni=0* čímž řekne hlavnímu µC, že zařízení je stabilní a nyní chce provést měření. Na výstupu ze stavu tuto hodnotu změníme na takovou, pro které není na digitálních pinech definována kombinace. Po stanoveném čase program přejde do dalšího stavu.

Ve stavu *„druhy“* program vyšle signál k zatočení pumpu. Po 50ms přejde do stavu *„treti“*, nastaví piny jako vstupy a čeká na signál o tom, že se pumpa otáčí. Jakmile jej obdrží, přejde do dalšího stavu a spustí měření. Po naměření sta hodnot přechází do dalšího stavu, kde z těchto dat počítá průměrnou hodnotu a zapisuje do příslušných proměnných. Celý tento proces se poté opakuje, ovšem podruhé se pumpa otáčí směrem opačným.

Na konci tohoto procesu program zavolá funkci *zjisteni_smeru* a pošle do ní naměřená data. Tato funkce na základě nastavení data vyhodnotí a určí směr otáčení.

Byl tedy zjištěn směr otáčení, program přejde do dalšího stavu *„LED_smer“*, který slouží k signalizaci výsledků pomocí LED diody. Signalizace směru pomocí LED pak ve Stateflow vypadá následovně:



Obr.20: Stav signalizace výsledků pomocí LED

V tomto stavu se pouze rozhodne na základě hodnoty v proměnné *smer*, která je výstupní proměnnou z funkce *zjisteni_smeru* ve stavu *„vlastni_mereni“*, která barva na LED diodě se rozsvítí. Je připojen i stav bez podmínky (stavy s podmínkou mají však vyšší prioritu), který by v případě neočekávané hodnoty v proměnné zaznamenal chybu. Když proběhne signalizace zjištěného směru, program přejde po pěti vteřinách zpět do stavu *„pohotovost“*, jak lze vidět na obrázku 16. Během přechodu je vynulována hodnota proměnné *smer*.

3.3.3 I²C komunikace

Oproti původnímu očekávání, že budeme moci při realizaci využít již použité řešení pro komunikaci po I²C, které jsme úspěšně aplikovali na testovací jednotce, jsme byli nuceni celou komunikaci napsat v jazyce C znova od začátku.

Z nám nejasných příčin jsme nebyli schopni komunikaci zprovoznit a po mnoha hodinách pokusů jsme se rozhodli k časově náročnému programování (v případě problematiky neznalého programátora) této komunikace, které rovněž znamenalo podrobnou studii datasheetu senzorů [4] a μC [7].

Zde je uvedena část programu zodpovědná za samotný příjem dat ze senzorů. Celý kód včetně inicializace je přiložen na CD.

ZAČÁTEK ZKRÁCENÉHO KÓDU:

```
extern unsigned int Hall_data(char u1){

    Init_2_I2C();               // Secondary inicialization call

    unsigned int I2C_bf1;       // Variables declaration
    unsigned int I2C_bf2;
    unsigned int I2C_bf;

    IdleI2C();                  // Wait for I2C idle

    StartI2C();                 // Start I2C
    while(I2CCONbits.SEN);      // Wait till Start sequence is completed
    IFS0bits.MI2CIF = 0;        // Clear interrupt flag

    MasterWriteI2C(0x1C);       // Slave adress for write
    while(I2CSTATbits.TBF);     // 8 clock cycles
    IFS0bits.MI2CIF = 0;        // Clear interrupt flag
    while(I2CSTATbits.ACKSTAT); // Wait for acknowledge

    IdleI2C();

    MasterWriteI2C(u1);         // Register adress write
    while(I2CSTATbits.TBF);     // 8 clock cycles
    IFS0bits.MI2CIF = 0;        // Clear interrupt flag
    while(I2CSTATbits.ACKSTAT); // Wait for acknowledge

    IdleI2C();

    RestartI2C();               // Restart I2C
    while(I2CCONbits.RSEN);     // Wait till Start sequence is completed
    IFS0bits.MI2CIF = 0;        // Clear interrupt flag

    MasterWriteI2C(0x1D);       // Slave adress for read
    while(I2CSTATbits.TBF);     // 8 clock cycles
    IFS0bits.MI2CIF = 0;        // Clear interrupt flag
    while(I2CSTATbits.ACKSTAT); // Wait for acknowledge

    IdleI2C();                  // Wait for I2C idle

    I2C_bf1 = MasterReadI2C();  // Read first 8 bits of register
    AckI2C();                   // Send acknowledge status
```

KÓD POKRAČUJE NA NÁSLEDUJÍCÍ STRANĚ

POKRAČOVÁNÍ KÓDU

```
    IdleI2C();                  // Wait for I2C idle

    I2C_bf2 = MasterReadI2C();  // Read second 8 bits of register
    NotAckI2C();                // Send notacknowledge status
    StopI2C();                  // Stop I2C
    while(I2CCONbits.PEN);      // Wait till stop sequence is completed

    I2C_bf = (I2C_bf1 << 8);    // Bit shift
    I2C_bf = (I2C_bf | I2C_bf2);// Complete bits

    return (I2C_bf);            // Return value
}
```

KONEC ZKRÁCENÉHO KÓDU

Tuto funkci voláme pomocí bločku *C function call* v subsystému zajišťujícím čtení dat ze senzorů. Na začátku funkce se spustí sekundární inicializace, kterou je potřeba provést při každém volání této funkce. Následuje deklarace lokálních proměnných a ověření, je-li sběrnice nečinná. Funkce *StartI2C* poté sběrnici spustí a můžeme zahájit komunikaci.

Nejprve pomocí příkazu *MasterWriteI2C* pošleme po sběrnici adresu zařízení pro zápis (adresy zařízení pro zápis do něj a čtení z něj se liší, obvykle o hodnotu 1) a počkáme na signál, že toto zařízení adresu přijalo. Následně zapíšeme adresu registru, ke kterému chceme přistupovat, z hodnoty vstupující do bločku volající tuto funkci v Simulink modelu, a opět počkáme na potvrzení.

Důležitou roli zde hraje příkaz *IdleI2C*. Po sekvencích, kdy posíláme data po sběrnici, je vždy potřeba počkat, až bude sběrnice volná, jinak nebude komunikace fungovat. Při programování skladby komunikace jsme vycházeli z datasheetu senzorů [4], kde je problematika popsána. Využívali jsme zároveň některých zjednodušení, která mají v sobě senzory již zabudované, jako je např. automatická inkrementace ukazatele registru při čtení dat.

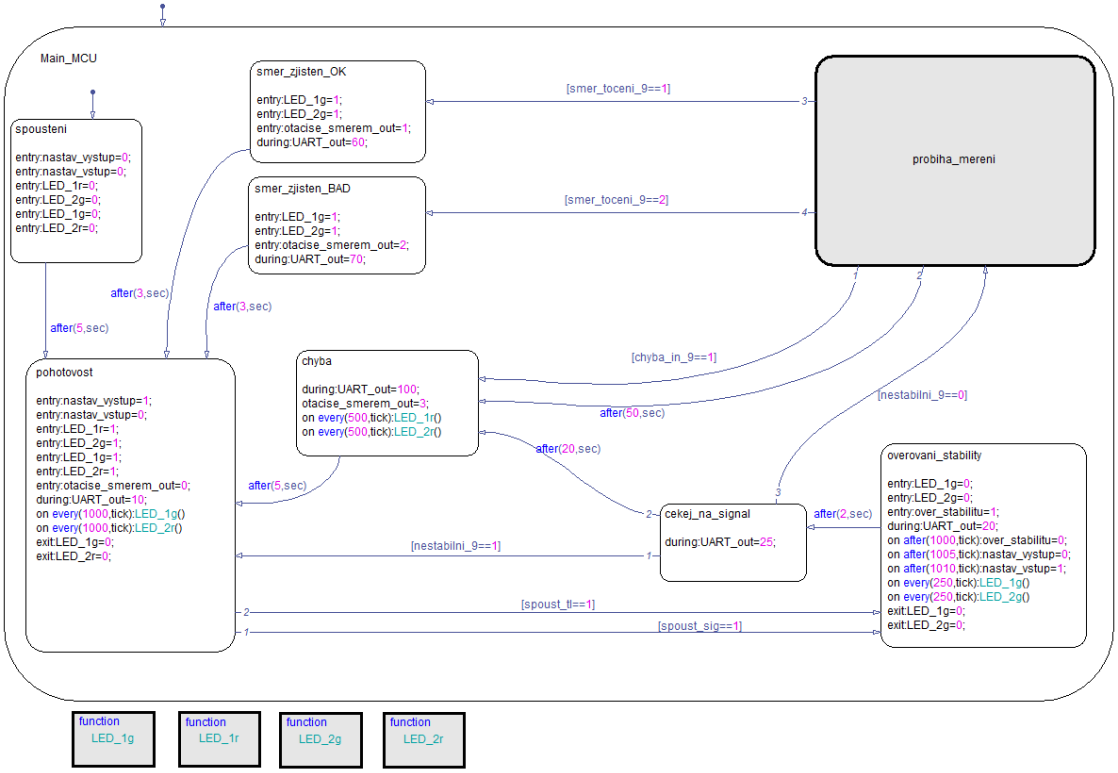
Po proběhnutí zápisu adresy registru je nutno sběrnici restartovat pomocí *RestartI2C*. Nyní zapíšeme adresu zařízení pro čtení z něj , počkáme na proběhnutí procedury a můžeme provést čtení příkazem *MasterReadI2C*. Následně zašleme signál *AckI2C*, jímž potvrdíme příjem. Díky přednastavení v senzoru automaticky senzor posune ukazatel registru o jedna výše, což nám umožní okamžitě dalším příkazem pro čtení obdržet jeho hodnotu. Toto činíme proto, jelikož jsou v senzoru hodnoty pro jednu osu rozděleny ve dvou za sebou jdoucích registrech.

Signálem *NotAckI2C* sdělíme senzoru, že již nechceme, aby posouval ukazatel registru a pomocí *StopI2C* komunikaci ukončíme.

Nyní již stačí složit obě hodnoty dohromady a odeslat na výstup funkce, čili vložit do příkazu *return* námi složenou lokální proměnnou.

3.3.4 Program Main µC

Tento program obstarává zejména synchronizaci všech devíti Hall µC a komunikaci ven ze zařízení. Obsluhuje rovněž dvě LED diody signalizující stav, ve kterém se zařízení nachází. Program ve Stateflow je obdobný tomu pro µC v předchozí kapitole. Pro přehlednost jsou na obrázku 21 některé části, zejména podmínky přechodů mezi stavy, upraveny pouze pro komunikaci s jedním Hall µC. Popisována bude však funkčnost celková. Program vypadá následovně:



Obr. 21: Program Main µC ve Stateflow

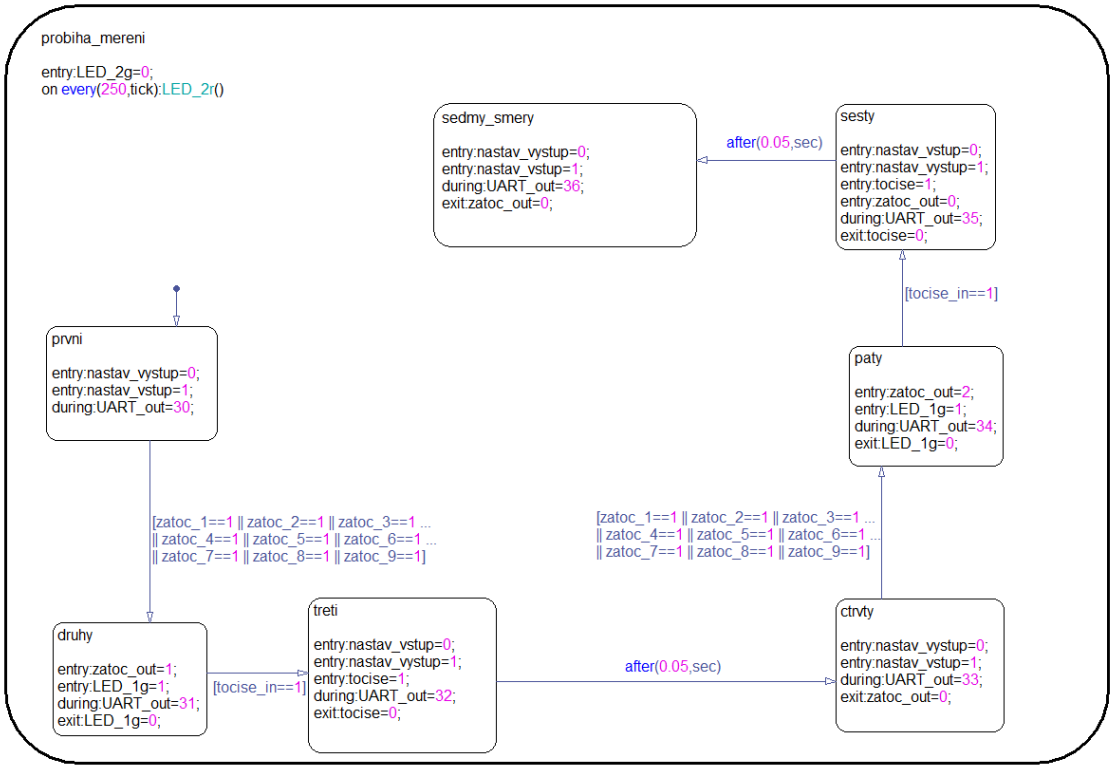
Po zapnutí zařízení se program nachází ve stavu *„spousteni“*, kde proběhne nastavení některých proměnných. Proměnné *nastav_vystup* a *nastav_vstup* jsou zodpovědné za spouštění částí programu nastavující digitální piny ke komunikaci s Hall µC jako výstupy nebo vstupy. Ty jsou v modelu v Simulinku reprezentovány Enabled subsystémy. Obrázek modelu je v příloze na straně 34. Obsahují-li tyto proměnné 0, jsou tyto bloky neaktivní. Naopak se aktivují, když zapíšeme do dané proměnné 1.

Po pěti vteřinách se program převede do stavu *„pohotovost“*. Piny ke komunikaci s Hall µC se nastaví jako výstupy a začnou blikat dvě LED diody. Logika LED diod je zde rovněž inverzní, tedy je-li v příslušné proměnné 1, dioda nesvítí a naopak. Pro potřebu odlazení je zde taktéž proměnná *UART_out* do níž zapisujeme čísla tak, abychom na UART výstupu zjistili, ve kterém stavu se zařízení právě nachází.

Nyní tedy zařízení čeká na obdržení signálu ke spuštění měření. Ten může přijít buď signálem přes konektor, nebo od tlačítka.

Jakmile přijde signál k zahájení měření, program přejde do stavu *„overovani_stability“*. Vyšle signál k zahájení měření ke všem devíti µC pomocí zapsání do proměnné *over_stabilitu* a změní číslo proměnné jdoucí na UART. Následně se po jedné sekundě (při nastavení frekvence výpočtů 1kHz) vypne posílání tohoto signálu, poté se změní nastavení digitálních pinů z výstupu na vstup, rozblikají se stanoveným způsobem LED diody a po dvou sekundách program přejde do dalšího stavu.

Ve stavu *„cekej_na_signal“* program čeká, až se na digitálních pinech objeví příslušná kombinace, tedy, že se v proměnné *nestabilni* objeví buďto 0 nebo 1. Když přijde signál, že zařízení je nestabilní, přejde program zpět do stavu pohotovost. Je-li zařízení stabilní, program vstoupí do stavu *„probiha_mereni“*, který vypadá následovně:



Obr. 22: Stav „probiha_mereni“

Tento stav zajišťuje správnou synchronizaci dějů při měření. Je potřeba, aby Hall µC měřily přesně ve chvíli, kdy bude pumpou procházet proud. Nejdříve program nastaví piny jako vstupy a počká na signál k zatočení pumpou. Při jeho obdržení přejde do dalšího stavu, vyšle signál na výstupní konektor k zatočení pumpou směrem, ve kterém očekáváme, že by měla být správně zapojena. Poté se čeká, než se na příslušném pinu na konektoru objeví signál o tom, že se pumpa otáčí.

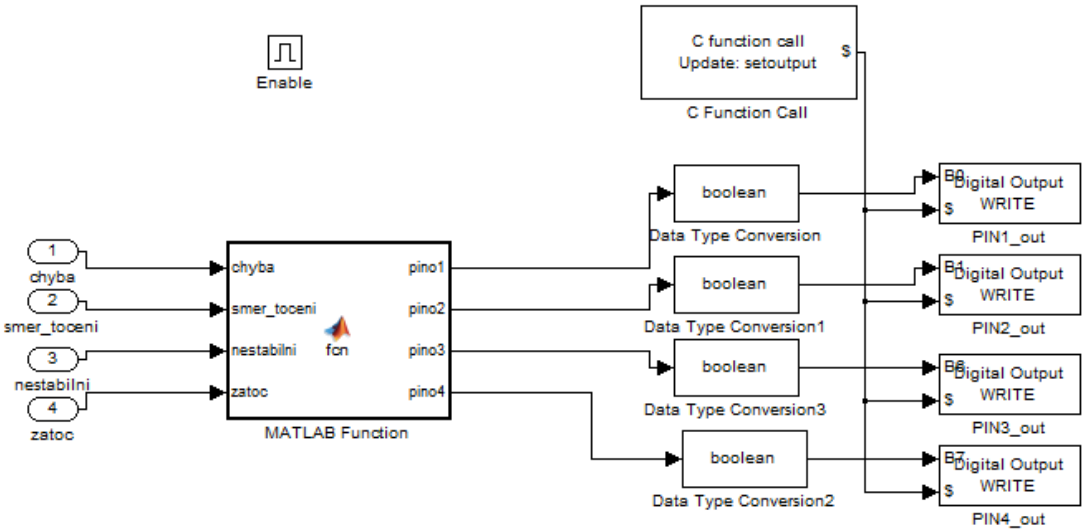
Jakmile jej obdrží, opět změní nastavení pinů k Hall µC na výstup, zašle signál o otáčející se pumpě a za 50ms přejde do dalšího stavu. Zde se opakuje to, co ve stavech předchozích, akorát s tím rozdílem, že program pošle signál o zatočení pumpou ve směru opačném. Jakmile program dojde do stavu posledního, tak čeká na výsledky měření.

Na obr. 21 lze vidět, že jsou zde dva stavy, do kterých může program přejít. Buďto je výsledkem měření, že se pumpa otáčí očekávaným směrem a program přejde do stavu *„směr_zjisten_OK“* anebo se otáčí směrem opačným a program tudíž přejde do stavu *„směr_zjisten_BAD“*. Pomocí LED diod signalizuje, že se v těchto stavech nachází, a na výstupní konektor pošle informaci o zjištěném stavu. Po několika vteřinách se program navrací do stavu pohotovosti a zařízení je připraveno provést další měření.

3.3.5 Komunikace mezi µC pomocí digitálních pinů

Komunikace mezi Main µC a Hall µC probíhá za použití digitálních signálů. K tomuto řešení jsme přistoupili, protože stačí posílat jen několik druhů signálů. Na každém Hall µC slouží k této komunikaci 4 piny. Abychom však byli schopni rozpoznat nebo definovat digitální reprezentaci proměnných, které jsou posílány, bylo potřeba vytvořit jednoduchou funkci, která to bude zajišťovat.

Jak již bylo zmíněno, tyto digitální piny slouží jednak jako vstupy a také jako výstupy. Přepínat mezi nimi nám umožňuje bloček *Enabled subsystem* v Simulink modelu. Ten může vypadat následovně:



Obr. 23: Enabled subsystem - nastavení dig.pinů jako výstupy

Na vstup do tohoto subsystému přicházejí proměnné z *Chart* bločku v modelu. Ty vstupují do vnořené funkce, která na základě hodnoty v proměnných přiděluje 1 či 0 na jednotlivé piny. Tyto signály reprezentované číslem je ještě potřeba pomocí bločku *Data Type Conversion* převést na signály typu *boolean*.

Ještě než mohou být odeslány na výstupní pin, je zavolána funkce *setoutput* psaná v jazyce C. V ní se nastavuje *TRIS* registr příslušných pinů. Aby se funkce provedla dříve, než se piny nastaví jako výstupní, zajišťuje možnost *Blok ordering* v menu jednotlivých bloků.

Všechny programy a funkce jsou přiloženy na CD.

4. Popis fungování celého zařízení

Sepnutím páčkového spínače přivedeme napětí do obvodů zařízení. Po pěti vteřinách se zařízení uvede do stavu pohotovosti. Zařízení je připraveno k zahájení měření a čeká na pokyn. Ten lze vyvolat stisknutím tlačítka, nebo přivedením signálu na příslušný pin konektoru ze zařízení.

Po spuštění měření probíhá ověřování stability zařízení. Je důležité, aby zařízení bylo po celou dobu měření ve stabilní poloze vzhledem k Zemi. Jinak nelze zajistit správnost měřených výsledků. Je-li zařízení nestabilní, resp. je-li nestabilní magnetické pole okolo něj, zahlásí chybu signalizací pomocí LED diod a po chvíli se vrátí do stavu pohotovosti.

Je-li zařízení stabilní, pokračuje v procesu měření a vyšle signál na zatočení pumpou směrem, který očekáváme, že je správný. Poté čeká na potvrzující signál, že se pumpa točí a provede měření. Důležité je, aby se pumpa točila po celou dobu snímání, tedy přibližně jednu sekundu. Jakmile již nebude potřeba pumpou točit, signál na konektoru ze zařízení se změní.

Následně se provede to stejné při otáčení pumpou druhým směrem. Nyní má zařízení naměřená data, vyhodnotí je a pomocí LED diod signalizuje obsluze výsledky. Výsledky jsou také zaslány na výstupní konektor. Po chvíli se opět navrátí do stavu pohotovosti a je připraveno provést další měření.

Kdyby se měření nějakým způsobem přerušilo, pokazilo, nebo došlo k příliš velké časové prodlevě, zařízení samo po stanoveném čase přejde do výchozího stavu. V programu jsou vestavěny podmínky pro maximální čas, ve kterém by měly stavy proběhnout. Neproběhnou-li do stanovené doby, zařízení nahlásí chybu a vrátí se do pohotovosti.

5. Závěr

Cílem práce bylo navrhnout, sestrojit a naprogramovat zařízení pro detekci směru otáčení palivové pumpy, což jsme úspěšně splnili. Na základě rešerše jsme vybrali vhodný senzor magnetické indukce a experimentálně ověřili jeho vlastnosti. Ukázalo se, že pro naši aplikaci je potřeba senzor vysoce citlivý. Dokonce tak, že zachytává magnetické pole Země. Z toho vyplynula potřeba zajištění stabilní polohy senzoru i pumpy při měření.

Dle zadání byla navrhnuta DPS s polem senzorů 3x3 a DPS s μC pro vyhodnocení signálů ze senzorů. Jako vhodné jsme zvolili μC dsPIC firmy Microchip. Abychom mohli současně číst data ze všech senzorů a nezávisle je vyhodnocovat, přistoupili jsme k řešení pomocí devíti μC, jenž každý jeden slouží ke čtení dat právě z jednoho senzoru. K těmto μC jsou připojeny LED diody signalizující výsledky měření. Připojen je dále jeden hlavní μC, který synchronizuje chod všech devíti μC a předává potřebné signály na výstup ze zařízení.

Byl rovněž vytvořen program zajišťující správnou funkčnost zařízení. K jeho vytvoření bylo využito automatické generování kódu z prostředí Simulink a jeho doplňující knihovny Kerhuel Toolbox. Při realizaci práce se ukázala být problémem komunikace po sběrnici I^2C mezi senzorem a μC. Oproti předpokladu, že využijeme již hotového řešení problému, bylo potřeba celou komunikaci v jazyce C znova naprogramovat. Řešení zabralo velké množství času.

Zařízení je schopno rozpoznat, zdali se nachází ve stabilní poloze, měřit změnu magnetické indukce okolo palivové pumpy, vyhodnocovat směr jejího otáčení, signalizovat výsledky a stavy zařízení, a předávat signály dál.

Zařízení je nyní kalibrováno na využití jednoho senzoru z připojeného pole. Pro využití všech devíti senzorů je potřeba upravit rozhodovací části programu vhodně pro každý senzor zvlášť. Toto je způsobeno faktem, že magnetická indukce a její změna při průchodu proudu je na pozici každého senzoru značně rozdílná.

6. Použité zdroje a literatura

[1] Grepl, R.; Vejlupek, J.; Matejasko, M.; Zouhar, F., "Sensorless Detection of DC Motor Rotation Direction for Automotive Fuel Pump Fault Diagnosis", Recent Advances in Circuits, Communications and Signal Processing, 2013

[2] TARÁBEK, Pavol. *Odmaturuj! z fyziky*. Vydání druhé. Brno: Didaktis, 2006, s. 125. ISBN 80-86285-39-1.

[3] HAL®401, *Linear Hall-Effect Sensor IC*. Micronas GmbH. Data Sheet, Datum revize 8.12.2008.
Dostupné z:
http://www.micronas.com/sites/default/files/downloads/files/HAL401_Linear_Hall_Effect_Sensor_IC_%28DSH000018002EN%29.pdf.

[4] MAG3110, *Xtrinsic MAG3110 Three-Axis, Digital Magnetometer*. Data Sheet, rev. 9.1, 10/2012. Dostupné z:
http://www.freescale.com/files/sensors/doc/data_sheet/ MAG3110.pdf

[5] DUDÁČEK, K., *Sériová rozhraní SPI, Microwire, I2C a CAN*. [online]. 2013-5-4.
Dostupné z: http://home.zcu.cz/~dudacek/NMS/Seriova_rozhrani.pdf

[6] ŠIMURDA, M. *Návrh a realizace doplňkového sensorického systému pro experimentální vozidlo*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2010. 33 s. Vedoucí Ing. Robert Grepl, Ph.D.

[7] MICROCHIP. ds*PIC30F2011/2012/3012/3013, High-Performance, 16-bit Digital Signal Controllers*.[online]. Dostupné z:
http://ww1.microchip.com/downloads/en/DeviceDoc/70139G.pdf

[8] MICROCHIP. ds*PIC33FJXXXGPX06/X08/X10, High-Performance, 16-Bit Digital Signal Controllers*.[online]. Dostupné z:
http://ww1.microchip.com/downloads/en/DeviceDoc/70286C.pdf

[9] TEXAS INSTRUMENTS. *LM2596 SIMPLE SWITCHER® Power Converter 150 kHz 3A Step-Down Voltage Regulator* [online]. Dostupné z:
http://www.ti.com/lit/ds/symlink/lm2596.pdf

7. Seznam obrázků a tabulek

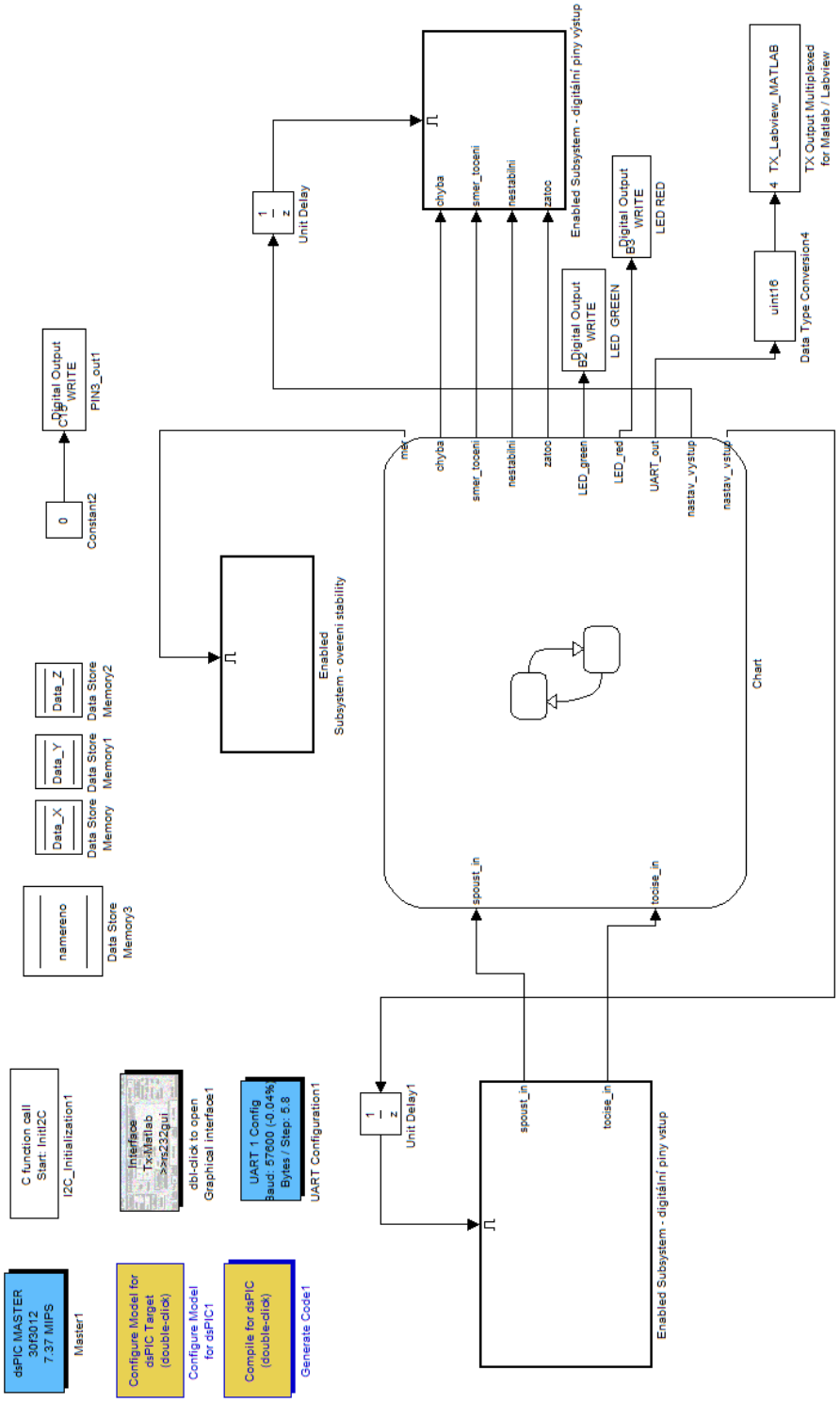
8. Použité zkratky a symboly

µC	*microcontroller*	mikrokontrolér
DPS		deska plošných spojů
UART	*Universal Asynchronous Receiver and Transmitter*	univerzální asynchronní vysílač a přijímač
I^2C	*Inter-Integrated Circuit*	sériová komunikační sběrnice
I/O	*Input/Output*	vstup/výstup
LED	*light emitting diode*	svítivá dioda
Hz	*Hertz*	Hertz – jednotka frekvence
PC	*personal komputer*	osobní počítač

9. Seznam příloh

10. Přílohy

Model programu Hall µC s UART v Simulinku



Model programu Main µC s UART v Simulinku