UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

DIPLOMOVÁ PRÁCE

2011 Bc. Jiří Paar

UNIVERZITA PARDUBICE

FAKULTA ELEKROTECHNIKY A INFORMATIKY

Zařízení pro detekci pohybu osob a vozidel

Bc. Jiří Paar

Diplomová práce

2011

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2010/2011**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Jiří Paar**
Osobní číslo:	**I09404**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Komunikační a řídicí technologie**
Název tématu:	**Zařízení pro detekci pohybu osob a vozidel**
Zadávající katedra:	**Katedra elektrotechniky**

Zásady pro vypracování:

Cílem práce je vytvořit systém autonomních senzorů detekce osob a vozidel na křižovatce. Senzory využívají mikrovlnné senzory pracující na principu Dopplerova jevu. Signál z tohoto senzoru vyhodnocuje mikrokontrolér LM3S811 s jádrem Cortex od firmy Texas Instruments. Senzory detekce vozidel umožňují měření rychlosti. Rychlost vozidel je určena na základě rychlé Fourierovy transformace, která je vypočtena z ovzorkovaného signálu mikrovlnného senzoru. Senzory detekce osob umožňují pouze detekci osob. Jednotlivé senzory komunikují s centrálním prvkem bezdrátovou technologií na vzdálenost až 300m. Úkolem centrálního prvku je přijatá data vyhodnotit a na jejich základě případně vyslat povel pro elektronické dopravní značky. Senzory musejí umožňovat napájení ze solárního panelu nebo z dopravního zařízení. Osnova: - popis signálu z Dopplerova senzoru, - popis systému, - návrh HW detektorů, - popis algoritmů, - měření a testy.

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

[1] **Stellaris ARM Cortex-M3 [online]. 2010 [cit. 2010-10-21]. Dostupné z WWW:**
[url: http://focus.ti.com/docs/prod/folders/print/lm3s811.html].
[2] **ARM The Architecture For The Digital World [online]. Version 2.0. 2010-09-12 [cit. 2011-07-29]. CMSIS - Cortex Microcontroller Software Interface Standard. Dostupné z WWW: [url: http://www.arm.com/products/processors/cortex-m/cortex-microcontroller-software-interface-standard.php].**
[3] **IQRF [online]. 2010 [cit. 2010-10-21]. Dostupné z WWW: [url: http://www.iqrf.org/weben/index.php].**
[4] **SKOLNIK, Merrill I. Introduction to radar systems. 2. Singapore : McGraw-Hill Book Co., 1981. 581 s. ISBN 0-07-057909-1.**

Vedoucí diplomové práce: **Ing. Richard Capalini, CSc.**
Steinel Technik Pardubice

Datum zadání diplomové práce: **12. listopadu 2010**

Termín odevzdání diplomové práce: **29. srpna 2011**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Zdeněk Němec, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2011

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezentačním zpřístupněním své práce v Univerzitní knihovně Univerzity Pardubice.

V Pardubicích dne 29. 8. 2011

Jiří Paar.

Poděkování:

Úvodem bych rád poděkoval vedoucímu práce panu ing. Richardovi Capalini CSc. za odborné rady a vedení během práce. Také velice děkuji společnosti Steinel Technik, která mi umožnila realizaci a následné testování vyvinutého zařízení. Poděkování také patří všem, kdo mě v práci pomáhali a podporovali.

SOUHRN

Práce se zabývá návrhem a realizací senzorových jednotek pro detekci chodců a vozidel v blízkosti dopravní křižovatky. Práce se zaměřuje na popis jednotlivých částí jednotek a jejich návrhem zapojení. Důraz je také kladen na odvození důležitých vztahů nutných pro samotný návrh. Každá jednotka je řízena vybraným mikrokontrolérem, který zpracovává signál z mikrovlnného senzoru a následně předává vyhodnocenou informaci bezdrátovému modulu, který ji vysílá do centrální jednotky křižovatky.

KLÍČOVÁ SLOVA

Inteligentní křižovatka, Dopplerův jev, mikrokontrolér, bezdrátová komunikace, IQRF.

TITLE

Equipment for detecting movement of persons and vehicles

ABSTRACT

The work deals with design and implementation of sensor units to detect pedestrian and vehicle traffic near the intersection. The work focuses on descriptions of the units involved and their design. Emphasis is also placed on the derivation of important relationships required for the proposal. Each unit is controlled by the chosen microcontroller that processes the signal from the microwave sensor and then passes the information assessed wireless module, which it sends to the central unit junction.

KEYWORDS

intelligent intersection, Doppler, microcontroller, wireless communication, IQRF.

Obsah

Seznam zkratek:

A/D	Analog-Digital Converter	Analogově-číslicový převodník
CMSIS	Cortex Microcontroller Software Interface Standart	Knihovna zprostředkující přístup k periferním obvodům mikrokontrolérů s jádry Cortex
CRC	Cyclic Redundancy Check	Cyklický redundantní kód
CW	Continuous Wave	Spojitá vlna
DFT	Discrete Fourier Transform	Distrétní Fourierova transformace
DIF	Decimation In Frequency	Decimace ve frekvenci
FIFO	First In First Out	Typ zásobníku, data jsou čtena ve stejném pořadí v jakém byla zapsána
FFT	Fast Fourier Transformation	Rychlá Fourierova transformace
FSK	Frequency Shift Keying	Frekvenční modulace
FT	Frourier Transformation	Fourierova transformace
GPTM	General Purpose Timer Module	Časovač pro všeobecné použití
IIC, I^2C	Inter Integrated Circuit	Sériová sběrnice pro periferie
JTAG	Joint Test Action Group	Standardizovaný protokol pro testování desek plošného spoje a pro programování.
PLL	Phase Locked Loop	Jednotka fázového závěsu pro vytvoření dané výstupní frekvence
PWM	Pulse Width Modulation	Pulsně šířková modulace
SPI	Seriál Peripherial Interface	Sériová sběrnice pro periferie
TF	Transfer Function	Funkce programu MATLAB pro vytvoření systému s daným přenosem

TVS	Transient Voltage Suppresion	Rychlá dioda pro ochranu proti přepětí
UART	Universal Asynchronous Receiver Transmitter	Periferie umožňující příjem i vysílání asynchronních sériových dat
WP	Write Protect	Vývod pro ochrana proti zápisu sériové paměti EEPROM

Seznam obrázků:

Seznam tabulek:

Úvod

Cílem této práce je navrhnout senzorové jednotky pro detekci osob na přechodu pro chodce a pro detekci vozidel s měřením rychlosti. Tyto jednotky budou umístěny na tzv. inteligentní křižovatce, kde budou kontrolovat a umožňovat řízení okolního provozu křižovatky. Získané údaje budou bezdrátově přenášeny do centrální jednotky, která na základě těchto informací bude řídit provoz křižovatky. Centrální jednotka bude kromě příjmu informací ze senzorových jednotek schopna vysílat bezdrátově informace pro dopravní značení.

Každá senzorová jednotka má být zcela autonomní. Proto bude napájena z baterie, která bude dobíjena ze solárního panelu. Informace o pohybu okolních objektů, tedy osob nebo vozidel, bude získávána pomocí mikrovlnného senzoru pracujícího na principu Dopplerova jevu. Bezdrátová komunikace má být zaručena na vzdálenost až 300 m.

Začátek textu je věnován popisu základních principů funkce inteligentní křižovatky. Další část je věnována odvození vztahů pro výstupní signál dopplerova senzoru. Další kapitoly se zaměří na popis použitých obvodů a odvození vztahů nutných pro správný návrh obvodů. Následující kapitoly jsou věnovány podrobnému popisu obvodové realizace jednotlivých částí senzorové jednotky. Další část popisuje digitální zpracování signálu ze senzoru pro jednotlivé typy senzorových jednotek včetně popisu bezdrátové komunikace. Poslední kapitola je zaměřena na popis provedených měření.

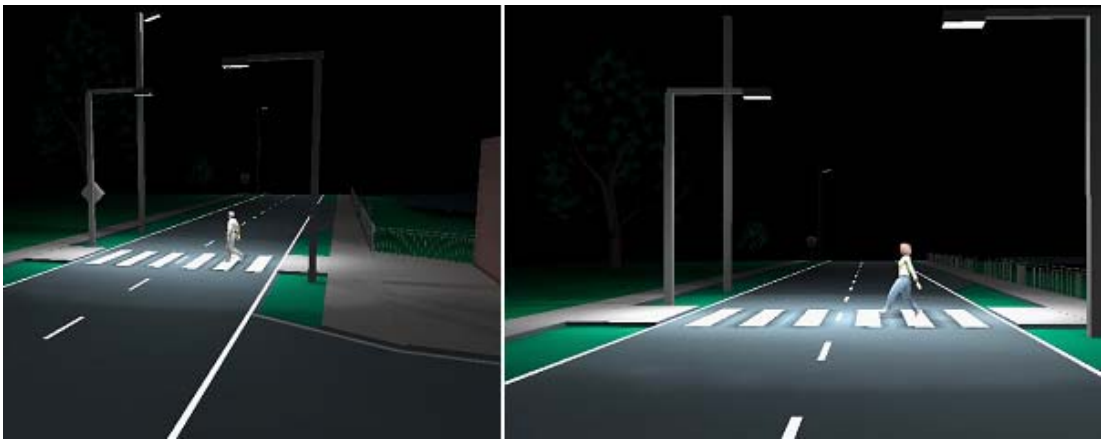
1. Rozbor práce

Systémy inteligentních křižovatek začaly nabývat na důležitosti s rostoucím počtem vozidel a tím i zvyšováním hustoty provozu. Největší uplatnění nacházejí na nejkritičtějších místech, kde se setkávají dva odlišné typy provozu. Jedním druhem je automobilová doprava a druhým typem jsou chodci.

Zabezpečení přechodů pro chodce je důležitým opatřením pro ochranu občanů. Zabezpečení přechodů se rozděluje podle množství bezpečnostních prvků umístěných v blízkosti přechodu nebo přímo na přechodu. Podrobné informace můžeme najít na [1].

1.1 Přehled stávajícího zabezpečení přechodů pro chodce

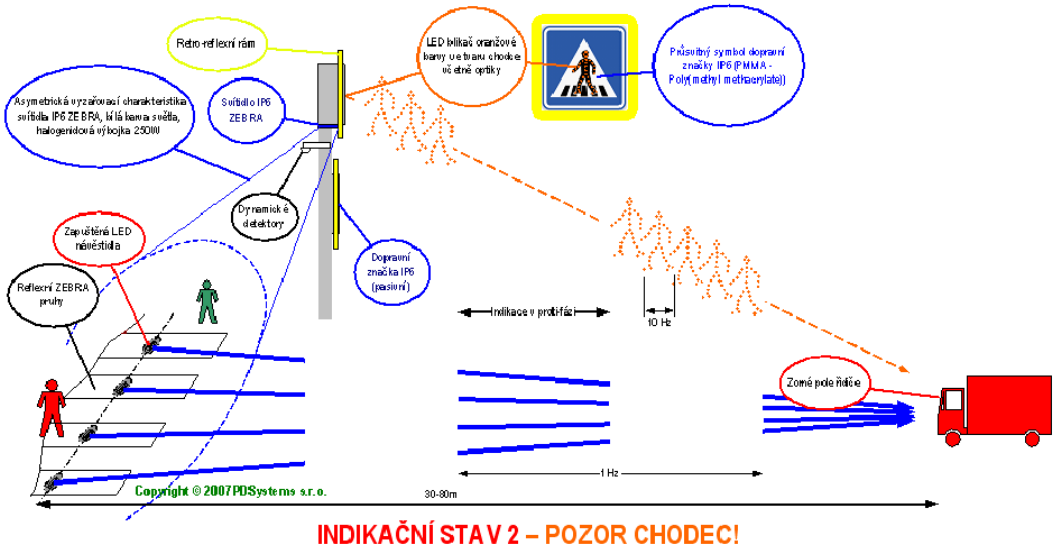
Jednoduchým opatřením pro zvýšení bezpečnosti na přechodu je pasivní systém, který je založen na rčení „vidět a být viděn“. Toto opatření spočívá ve vhodném osvětlení přechodu, viz následující obrázek:



Obrázek 1.1 – Osvětlení přechodu pro chodce [1]

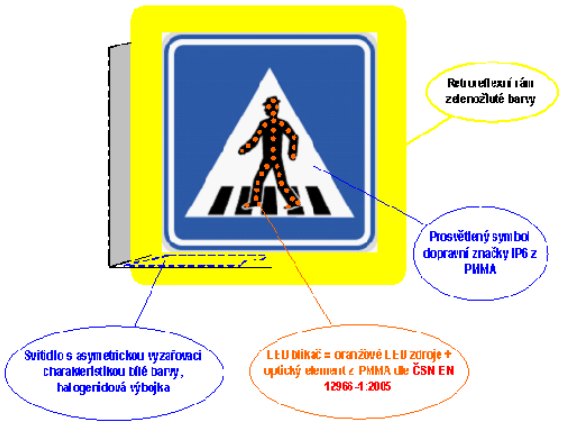
Další zvýšení bezpečnosti přinášejí obousměrné LED diody, které jsou umístěny uprostřed samotného přechodu pro chodce. Pro dopravně-inženýrský pohled je nejvhodnější oranžová barva. Pro nebezpečné přechody označené červenou barvou je vhodné použít také červenou barvu pro LED diody. Tato opatření se označují jako 1. generace.

2. generace navazuje na 1. generaci a rozšiřuje ji o aktivní upozornění na výskyt chodce, samozřejmostí je přisvícení přechodu. Systém detekuje chodce bez jakéhokoli zpoždění pro včasné upozornění řidiče. Informace o vstupu chodce na přechod je známa ještě dříve, než chodec na přechod fyzicky vstoupí (detekční zóna zasahuje i mimo přechod) a je detekován po celou dobu jeho výskytu na přechodu. Řidiči je informace o přítomnosti chodce předána pomocí LED blikačů umístěných ve svislé dopravní značce IP6 ZEBRA (viz obrázek 1.2):



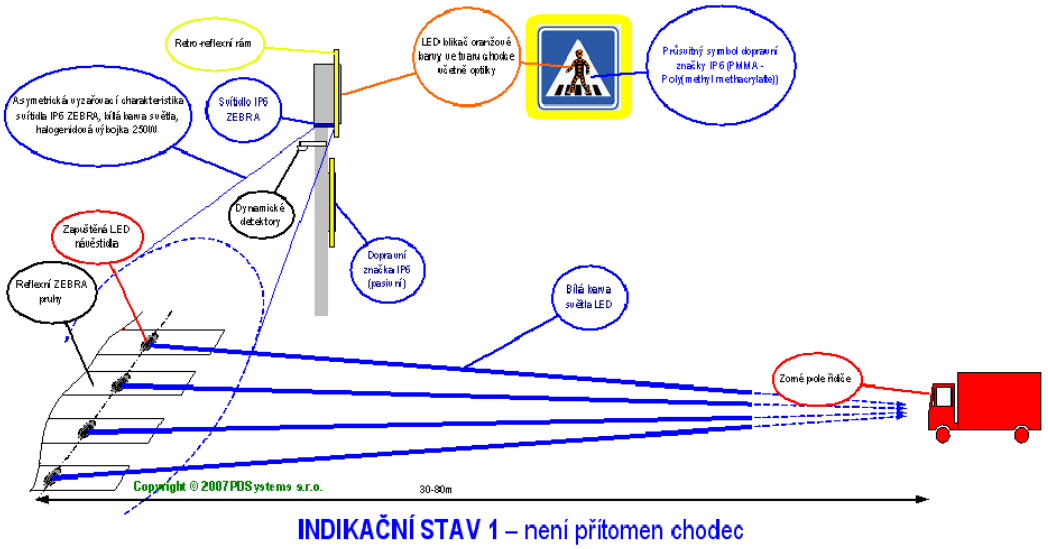
Obrázek 1.2 – Situace detekování chodce na přechodu 2. generace [1]

Detekování chodce na přechodu 2. generace způsobí blikání LED diod zapuštěných v samotném přechodu s frekvencí 1 Hz a blikáním LED diod s frekvencí 10 Hz ve značce IP6 ZEBRA umístěné nad přechodem. Bližší pohled na značku IP6 ZEBRA ukazuje obrázek 1.3:



Obrázek 1.3 – Značka IP6 ZEBRA [1]

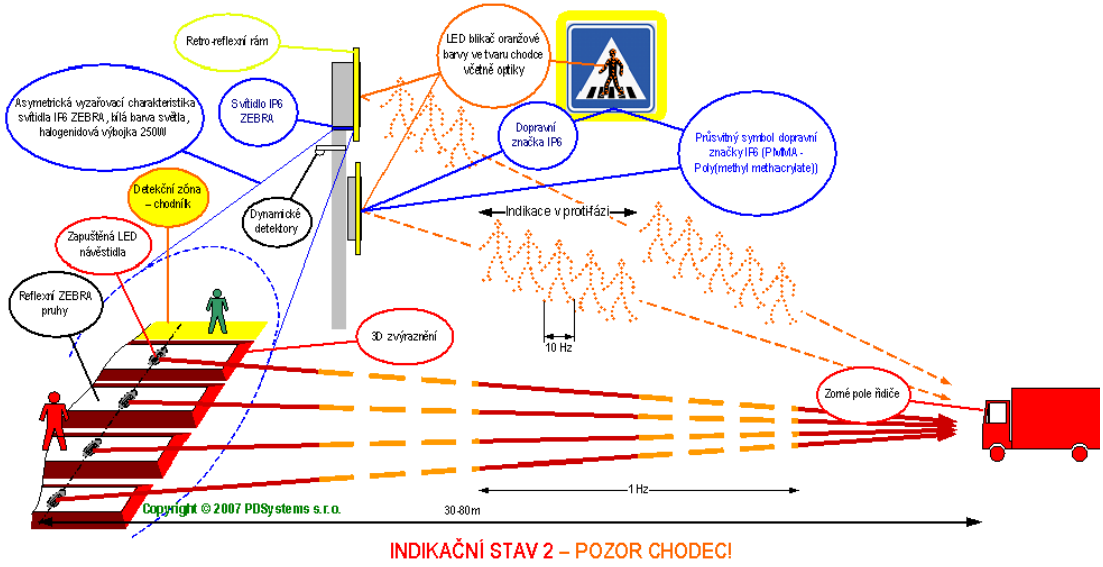
Situaci, kdy není zjištěn chodec na přechodu, vystihuje tento obrázek:



Obrázek 1.4 – Situace nepřítomnosti chodce na přechodu 2. generace [1]

V situaci, kdy není detekován chodec, neblikají žádné LED diody, pouze svítí.

3. generace opět navazuje na všechny předešlé generace. Tato generace upravuje způsob upozornění řidiče automobilu na detekci chodce na přechodu. Indikace stále využívá LED diod zapuštěných v přechodu pro chodce, ale nyní je použit nový typ LED diod, které jsou umístěny i mimo vozovku (značka IP6 ZEBRA). Stávající svítidla byla nahrazena novými obousměrnými, vícebarevnými LED návěstidly. Používají se barvy bílá, oranžová a červená. Bližší pohled na provedení signalizace přítomnosti chodce poskytuje obrázek 1.5:



Obrázek 1.5 – Situace detekování chodce na přechodu 3. generace [1]

Přítomnost chodce na přechodu 3. generace je signalizována blikáním LED diod umístěných v přechodu pro chodce střídáním barev červená a oranžová s celkovou frekvencí 1 Hz. Také blikají svislé dopravní značky IP6 ZEBRA (umístěné před přechodem) střídavě s frekvencí 10 Hz. Tato generace zabezpečení přechodů zavádí další bezpečnostní prvek v podobě 3D zvýraznění přechodu pro chodce. 3D zvýraznění spočívá ve správném použití geometrických prvků a barev pro vytvoření optického klamu v podobě třírozměrného přechodu, které nevyžaduje žádné stavební úpravy.

Obrázek 1.6 – 3D zvýraznění přechodu pro chodce [1]

Pro zvýšení bezpečnosti na přechodech pro chodce se používá také úprava nazvána ROCBINDA. Tato úprava stejně jako 3D zvýraznění nevyžaduje stavební úpravy. Přispění k bezpečnosti spočívá v použití bezpečného pásu červené barvy umístěného v obvyklé délce 20m před přechodem. Červená barva upozorňuje řidiče na nebezpečný úsek, drsný povrch navíc přispívá ke dřívějšímu zastavení vozidla. Na mokré vozovce se dosahuje až 33% zkrácení brzdné dráhy.

Obrázek 1.7 – Konstrukční provedení zvýraznění ROCBINDA [1]

1.2 Popis navrhovaného zabezpečení přechodu pro chodce

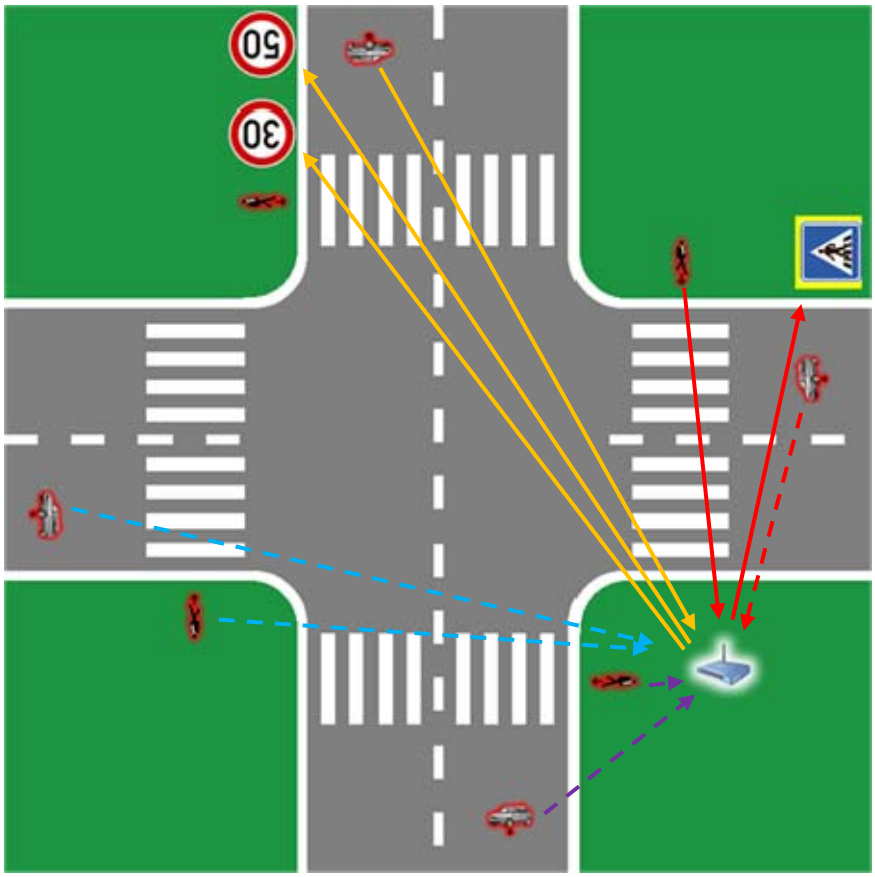
Navrhovaný systém, který má být výsledkem této práce, je zcela založen na aktivních prvcích a doplňuje stávající zabezpečení přechodů pro chodce o kontrolu přítomnosti vozidel v blízkosti křižovatek, resp. přechodů pro chodce. Nový systém, tedy kromě přítomnosti chodců v blízkosti přechodů pro chodce, vyhodnocuje také přítomnost vozidel pomocí dvojice typů samostatných senzorových jednotek.

Jednotka určená pro detekci chodců na přechodu vyhodnocuje pouze přítomnost chodce na přechodu, zatímco jednotka pro detekci vozidel je schopna měřit rychlost projíždějícího vozidla. Získané informace budou bezdrátově přenášeny do centrální jednotky, která je umístěna u řídící jednotky křižovatky. Řídící jednotka křižovatky následně tyto informace bude schopna zpracovat a na jejich základě bude řídit provoz křižovatkou. Bude tedy možné rychle projíždějící vozidlo zpomalit, nebo dokonce zastavit. Z množství projíždějících vozidel bude možné zjistit, který z jízdních směrů je více vytížen a upravit propustnost jízdních pruhů. Aby nebylo nutné zasahovat do stavebního provedení dané křižovatky, bude každá ze senzorových jednotek napájena z vlastní baterie, která bude dobíjena ze solárního panelu. Dalším uvažovaným zdrojem napájení je stávající napájení, např. z dopravního značení.

Centrální jednotka bude schopna kromě vyhodnocování přijatých informací také zasílat bezdrátově povely pro pevné nebo proměnlivé dopravní značení. Toto dopravní značení slouží pro předávání informací projíždějícím vozidlům.

Pro bezdrátovou komunikaci je použito bezlicenční pásmo 868 MHz. V tomto pásmu může komunikovat každý, např. řízení topení v domě apod. Proto musí být bezdrátová komunikace dostatečně zabezpečena. Pro omezení rušení je možné pro každý jízdní směr použít jiný komunikační kanál, tyto kanály bude možné nastavovat. Z tohoto důvodu bude centrální jednotka osazena několika bezdrátovými přijímači/vysílači, každý nastaven na jiném kanálu. Senzorové jednotky je vhodné osadit směrovými anténami pro zvětšení dosahu vysílaného signálu a omezení rušení pouze pro daný směr k centrální jednotce. Centrální jednotka naopak musí být vybavena všesměrovými anténami, aby byla schopna přijímat signál ze všech směrů, pro případ použití jednoho komunikačního kanálu pro všechny jízdní směry. V případě použití více komunikačních kanálů je možné centrální jednotku osadit několika směrovými anténami.

Pro lepší vysvětlení fungování nového systému ukažme situaci detekování chodce na přechodu a detekování rychle jedoucího vozidla:



Obrázek 1.8 – Situace detekování chodce a vozidla pro navrhovaný systém

Uvažovaná křižovatka je osazena čtyřmi detektory osob (umístěnými u přechodů) a čtyřmi detektory vozidel (umístěnými nad jednotlivými jízdními pruhy), u každého přechodu je umístěna značka IP6 ZEBRA (zobrazena pouze jedna) a u každého jízdního pruhu jsou např. umístěny proměnlivé značky pro omezení rychlosti na 30 a 50 $km \cdot h^{-1}$ (zobrazeny opět pouze u jednoho jízdního pruhu). Jednotlivé vysílací kanály jsou označeny různými barvami, probíhající komunikace mezi senzorovými jednotkami a centrální jednotkou jsou znázorněny plnou čarou, možné komunikace jsou zobrazeny čárkovanou čarou.

Zaznamená-li detektor vozidel přijíždějící automobil (horní jízdní pruh), odešle informaci o rychlosti automobilu do centrální jednotky. Ta tuto rychlost vyhodnotí a na základě velikosti rychlosti buď odešle, nebo neodešle příkaz pro proměnlivé dopravní značení ve směru automobilu, které se rozsvítí a bude automobilu přikazovat snížení rychlosti.

Pro detekování chodce na přechodu je situace velmi podobná s 2. Generací, resp. s 3. generací zabezpečení přechodů. Bude-li na přechodu detekován chodec,

zpráva o přítomnosti chodce bude odeslána do centrální jednotky, která odešle příkaz pro rozsvícení (např. dopravní značky IP6 ZEBRA) pro daný jízdní pruh, do kterého chodec vešel. Bude-li navíc ve stejném jízdním pruhu zjištěno vozidlo, bude kromě upozornění na přítomnost chodce nuceno snížit rychlost vozidla pomocí proměnného dopravního značení.

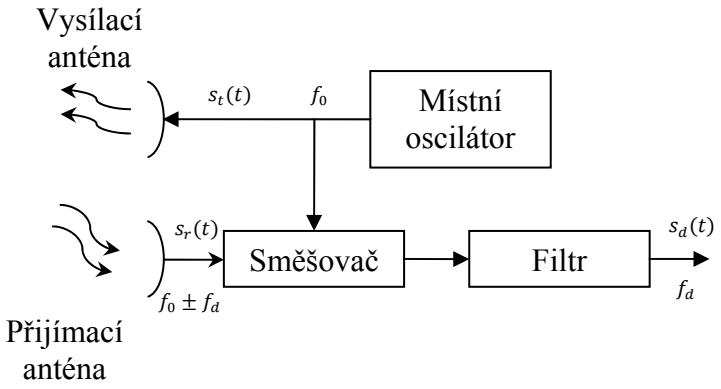
Mikrovlnné senzory jednotek pro detekci osob budou orientovány tak, aby svazek antény senzoru byl široký, aby bylo možné zachycení osoby ve větším okolí přechodu pro chodce. Jednotky budou umístěny v těsné blízkosti přechodu pro chodce. V případě jednotek pro detekci vozidel bude svazek úzký, aby nedocházelo k ovlivňování senzoru vozidly projíždějícími v sousedním pruhu. Orientace ve směru úzkého svazku přináší i vyšší dosah senzoru. Tyto jednotky mohou být umístěny ve vzdálenosti až 300 m od křižovatky.

Každé zařízení umístěné na křižovatce má svoji vlastní adresu a patří do určité bezdrátové sítě určené identifikátorem sítě. Bývá zvykem, že centrální prvek sítě má adresu rovnu nule, proto není ani zde použita jiná adresa. Všechny senzorové jednotky odesílají bezdrátové pakety právě zařízením s adresou nula, tedy centrální jednotce křižovatky.

2 Princip mikrovlnného senzoru

Mikrovlnný senzor, který je důležitou součástí každé senzorové jednotky, pracuje na principu Dopplerova jevu. Princip Dopplerova jevu a celé odvození rovnic pro určení tvaru přijatého a výstupního signálu senzoru jsou popsány v [2] a [3], odkud byly i převzaty.

Mikrovlnné senzory pracující na principu Dopplerova jevu vysílají spojitou vlnu, a proto jsou označovány jako CW (Continuous Wave). Vnitřní provedení takovéhoto senzoru je ukázáno na obrázku 2.1:



Obrázek 2.1 – Blokové schéma mikrovlnného senzoru [3]

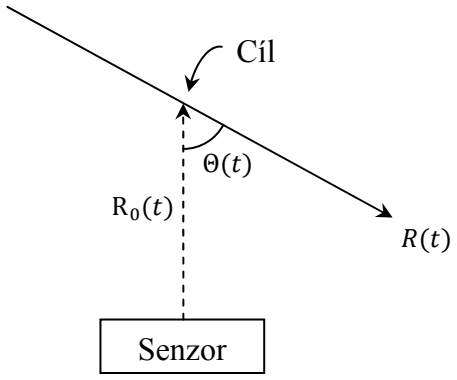
Senzor CW neustále vysílá signál $s_t(t)$ o frekvenci f_0 a vlnové délce λ_0 a přijímá signál $s_r(t)$ o frekvenci $f_0 \pm f_d$ (ukázáno bude později). Tento přijatý signál je přiveden na směšovač, kde je směšován se signálem místního oscilátoru o frekvenci f_0. Směšováním jsou signály sčítány na prvku s nelineární voltampérovou charakteristikou, např. dioda. Tímto procesem vznikají všechny součtové a rozdílové frekvence s původní mikrovlnnou složkou, která je následně odfiltrována.

2.1 Dopplerův jev

Dopplerův jev je znám zejména v akustice. Lze si ho představit tak, že pozorovatel, který přijímá akustický signál z přibližujícího zdroje konstantní akustické vlny, slyší tento kmitočet vyšší, než ve skutečnosti je, a naopak při vzdalování zdroje přijímá frekvenci nižší. Stejného efektu docílíme i v případě, kdy se zdroj nepohybuje a pohybuje se pozorovatel. Důležitým aspektem je tedy relativní

vzájemný pohyb pozorovatele a zdroje. Podobný jev je znám i u elektromagnetických vln.

Pozici mikrovlnného senzoru a cíle znázorňuje následující obrázek:



Obrázek 2.2 – Prostorové znázornění senzoru a cíle [3]

Cíl (objekt, který chceme zachytit) se pohybuje po trajektorii $R(t)$, senzor přijímá pouze radiální směr, který lze určit pomocí úhlu $\Theta(t)$ svíraného mezi trajektorií cíle a spojnicí senzoru s cílem, cíl je vzdálen od senzoru ve vzdálenosti $R_0(t)$. Pro pohybující cíl jsou všechny tyto veličiny funkcemi času t.

Pro vzdálenost R_0 cíle od senzoru lze určit celkový počet vlnových délek N_λ:

$$N_\lambda = \frac{R_0}{\lambda_0} \tag{2.1}$$

R_0 je vzdálenost cíle od senzoru [m],

λ_0 je vlnová délka vysílaného signálu [m].

Vzdálenost R_0 ovšem urazí signál celkem dvakrát, od senzoru k cíly a zpět. Fázový posun pro tuto vzdálenost lze určit vztahem:

$$\Phi = 4\pi \cdot \frac{R_0}{\lambda_0} \tag{2.2}$$

Úhlovou dopplerovu frekvenci lze odvodit jako změnu fáze v čase:

$$\omega_d = \frac{d\Phi}{dt} = \frac{4\pi}{\lambda_0} \cdot \frac{dR_0}{dt} = \frac{4\pi \cdot v_r}{\lambda_0} \ \ [rad.s^{-1}] \tag{2.3}$$

v_r je radiální rychlost cíle [m·s^{-1}],

λ_0 je vlnová délka vysílaného signálu senzorem [m].

Dopplerovu frekvenci lze vyjádřit ze vztahu (2.3) také takto:

$$f_d = \frac{2 \cdot v_r}{\lambda_0} = \frac{2 \cdot f_0 \cdot v_r}{c} \; [Hz] \tag{2.4}$$

f_0 je frekvence vysílaného signálu senzorem [Hz],

v_r je radiální rychlost [m·s^{-1}]

c je rychlost světla [m·s^{-1}].

Radiální rychlost v_r stejně jako radiální vzdálenost cíle od senzoru vyjadřuje složku rychlosti směřující přímo k senzoru:

$$v_r = v \cdot \cos \Theta \tag{2.5}$$

v je skutečná rychlost cíle [m·s^{-1}],

Θ je aktuální úhel mezi trajektorií cíle a spojnicí senzoru s cílem [rad.].

2.2 Výstupní signál

Výstupní vysílaný signál $s_t(t)$ Dopplerovského CW senzoru je roven tomuto tvaru:

$$s_t(t) = A_t \cdot \cos(2\pi \cdot f_0 \cdot t) \tag{2.6}$$

A_t je amplituda signálu [V],

f_0 je frekvence signálu [Hz],

t představuje čas [s].

Označíme-li vzdálenost cíle od senzoru R_0, potom zpoždění τ, které urazí signál od senzoru k cíli a zpět je:

$$\tau = \frac{2 \cdot R_0}{c} \; [s] \tag{2.7}$$

Pro pohybující se cíl bude zpoždění τ funkcí času a přijímaný signál lze vyjádřit jako zpožděný vyslaný signál ve tvaru:

$$s_t\big(t - T(t)\big) \tag{2.8}$$

$T(t)$ je zpoždění v čase t. Pro které platí:

$$T(t) = \frac{2}{c} \cdot R_0\left(t - \frac{T(t)}{2}\right)\ [s] \tag{2.9}$$

$R_0(t)$ je vzdálenost cíle od senzoru v čase t [m].

Pro další odvození je vhodné rozvinout rovnici pro $T(t)$ v Taylorovu řadu v okolí bodu $T(\tau) = \tau$.

Taylorův rozvoj pro funkci $f(x)$ v okolí bodu a lze zapsat následujícím vztahem:

$$\begin{aligned} f(x) &= f(a) + \frac{f'(a)}{1!} \cdot (x-a) + \frac{f''(a)}{2!} \cdot (x-a)^2 + \cdots = \\ &= \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!} \cdot (x-a)^k \end{aligned} \tag{2.10}$$

Pro uvažovanou funkci $T(t)$ je Taylorův rozvoj tento:

$$T(t) = \tau + T'(\tau) \cdot (t-\tau) + \frac{T''(\tau)}{2!} \cdot (t-\tau)^2 + \cdots \tag{2.11}$$

Vztahy pro derivace funkce $T(t)$ obsahují derivace funkce $R(t)$, pro které platí, že $R'(t)$ odpovídá radiální rychlosti $v_r(t)$ a $R''(t)$ odpovídá radiálnímu zrychlení $a_r(t)$. Při uvažování těchto souvislostí dostáváme tyto vztahy pro derivace $T(t)$:

$$T'(\tau) = \frac{\frac{2}{c} \cdot v_r\left(\frac{\tau}{2}\right)}{1 + \frac{v_r\left(\frac{\tau}{2}\right)}{c}} = \frac{2 \cdot v_r\left(\frac{\tau}{2}\right)}{c + v_r\left(\frac{\tau}{2}\right)} \tag{2.12}$$

$$T''(\tau) = \frac{\frac{2}{c} \cdot a_r\left(\frac{\tau}{2}\right)}{\left(1 + \frac{v_r\left(\frac{\tau}{2}\right)}{c}\right)^3} = \frac{2 \cdot a_r\left(\frac{\tau}{2}\right) . c^2}{\left(c + v_r\left(\frac{\tau}{2}\right)\right)^3} \tag{2.13}$$

Dosazením Taylorova rozvoje do rovnice pro přijímaný signál $s_r(t)$ dostaneme tento výraz:

$$
\begin{aligned}
s_r(t) &= A_r \cdot s_t\big(t - T(t)\big) = \\
&= A_r \cdot s_t\left(t - \tau - T'(\tau_r) \cdot (t-\tau) - \frac{T''(\tau)}{2!} \cdot (t-\tau)^2 - \cdots\right) = \\
&= A_r \cdot s_t\left((t-\tau) \cdot \left(1 - \frac{2 \cdot v_r\left(\frac{\tau}{2}\right)}{c + v_r\left(\frac{\tau}{2}\right)} - \frac{2 \cdot a_r\left(\frac{\tau}{2}\right) \cdot c^2}{\left(c + v_r\left(\frac{\tau}{2}\right)\right)^3} \cdot \frac{(t-\tau)}{2!} - \cdots\right)\right)
\end{aligned}
\tag{2.14}
$$

Pro případ rovnoměrného pohybu cíle je rychlost cíle konstantní $v_r(\tau) = v_r$ a zrychlení je nulové $a_r(\tau) = 0$. Budeme-li uvažovat tento typ pohybu, vztah pro přijímaný signál $s_r(t)$ přejde do tohoto tvaru:

$$
\begin{aligned}
s_r(t) &= A_r \cdot \cos\left(2\pi \cdot f_0 \cdot \left(1 \pm \frac{2 \cdot v}{c + v}\right) \cdot (t - \tau)\right) = \\
&= A_r \cdot \cos\left(2\pi \cdot \left(f_0 \cdot t \pm \frac{2 \cdot f_0 . v}{c + v} - f_0 \cdot \tau \cdot \pm \frac{2 \cdot f_0 \cdot v}{c + v} \cdot \tau\right)\right)
\end{aligned}
\tag{2.15}
$$

Výraz pro $s_r(t)$ lze dále zjednodušit. Ve výrazu ve jmenovateli, který obsahuje proměnné c a v, lze zanedbat proměnnou v, protože je několikrát menší než rychlost světla c. Dosazením výrazů pro zpoždění τ z rovnice (2.7) a dopplerovy frekvence f_d z rovnice (2.4) dostáváme výsledný vztah pro přijímaný signál $s_r(t)$:

$$
s_r(t) = A_r \cdot \cos\left(2\pi \cdot \left(f_0 \cdot t \pm f_d \cdot t - f_0 \cdot \frac{2 \cdot R_0}{c} \pm f_d \cdot \frac{2 \cdot R_0}{c}\right)\right) \tag{2.16}
$$

Průchodem směšovače a následným filtrem se přijímaný signál $s_r(t)$ změní na výstupní signál $s_d(t)$ senzoru:

$$
s_d(t) = A_d \cdot \cos\left(2\pi \cdot \left(\pm f_d \cdot t - f_0 \cdot \frac{2 \cdot R_0}{c} \pm f_d \cdot \frac{2 \cdot R_0}{c}\right)\right) \tag{2.17}
$$

A_d je amplituda výstupního signálu.

Vztah pro výstupní signál $s_d(t)$ lze zjednodušit. Funkce $\cos(x)$ je sudou funkcí, pro kterou platí $\cos(x) = \cos(-x)$. Uvažováním tohoto vztahu dostáváme výsledný vztah pro výstupní signál:

$$s_d(t) = A_d \cdot \cos\left(2\pi \cdot \left(f_d \cdot t + \frac{2 \cdot R_0}{c} \cdot (f_0 \pm f_d)\right)\right) = \\ = A_d \cdot \cos\left(2\pi \cdot f_d \cdot t + \frac{4\pi \cdot R_0}{c} \cdot (f_0 \pm f_d)\right) \tag{2.18}$$

Odvozené vztahy neobsahují systémové zpoždění, které bývá konstantní, a lze ho odstranit kalibrací.

Porovnáme-li výsledný výraz pro výstupní signál $s_d(t)$ s obecným tvarem signálu $s(t)$, jak ukazuje tento vztah,

$$s(t) = A \cdot \cos(2\pi \cdot f + \varphi) \tag{2.19}$$

A je amplituda signálu,

f je frekvence,

φ je fáze signálu.

zjistíme, že výstupní frekvence je rovna dopplerově frekvenci, která je přímo úměrná rychlosti pohybu cíle. Tento fakt má zásadní vliv pro další zpracování signálu.

2.3 Výpočet mezních frekvencí senzorových jednotek

Z předchozího textu vyplývá, že mikrovlnný senzor pracující na principu Dopplerova jevu vysílá kontinuální signál a přijímá signál obsahující po směšování frekvenci úměrnou rychlosti pohybujícího se objektu. Aby bylo možné od sebe odlišit jednotlivé objekty (např. osoby a vozidla), lze to provést třemi způsoby. Prvním způsobem je takové umístění senzorů, aby senzor snímal pouze ty správné typy objektů, druhým způsobem je frekvenční omezení výstupního signálu ze senzoru. Osoby se pohybují menší rychlostí než vozidla, ovšem pro nízké frekvence, které odpovídají nízkým rychlostem, které budou schopny zpracovávat oba senzory, mohou pomalý objekt zachytit oba typy senzorových jednotek. Proto by se mohlo stát, že např. detektor osob bude snímat i pomalu se pohybující vozidlo apod. Frekvenční omezení je nutné provést v každém případě, aby se ve výstupním signálu neobjevovaly frekvence odpovídající objektům, jež zpracovávat nechceme. Třetím způsobem je detailní zpracování signálu z mikrovlnného senzoru, kde se kromě

frekvence zpracovává i fáze, aby se zjistil i směr pohybujícího se objektu. Tento způsob je ovšem velmi náročný a pro dobré určení by musely být přítomny alespoň dva senzory.

Pro oddělení typů pohybujících se objektů v senzorových jednotkách bude použito prvního a druhého způsobu, tedy způsobem rozmístění senzorů (senzory pro detekci vozidel budou umístěny nad jízdními pruhy ve větší vzdálenosti od křižovatky a senzory pro detekci osob budou umístěny u přechodů pro chodce) a způsobem frekvenčního omezení. Frekvenčního omezení výstupního signálu bude docíleno na zesilovači analogového signálu zařazeného za mikrovlnným senzorem. Velikost dopplerovy frekvence (frekvence výstupního signálu) byla popsána v rovnici (2.4). Malou úpravou této rovnice získáme výraz pro rychlost v, minimální frekvenci $f_{d_{min}}$ pro minimální rychlost v_{min} a maximální frekvenci $f_{d_{max}}$ pro maximální rychlost v_{max} pohybujícího se objektu:

$$v = \frac{f_d \cdot \lambda_0}{2} \; [m \cdot s^{-1}] \tag{2.20}$$

$$f_{d_{min}} = \frac{2 \cdot v_{min}}{\lambda_0} \; [Hz] \tag{2.21}$$

$$f_{d_{max}} = \frac{2 \cdot v_{max}}{\lambda_0} \; [Hz] \tag{2.22}$$

f_d je dopplerova frekvence výstupního signálu senzoru [Hz],
v_{min} je minimální rychlost objektu [m·s^{-1}],
v_{max} je maximální rychlost objektu [m·s^{-1}],
λ_0 je vlnová délka vysílaného signálu senzorem [m].

Frekvenční omezení pro oddělení typů objektů nese jedno velké úskalí, mikrovlnný senzor umí zpracovávat pouze radiální rychlost, viz rovnice (2.5). Tato rovnice znamená, že pokud se bude objekt blížit k senzoru kolmo, pro senzor se bude jevit rychlost objektu jako snižující, pro úhel 90° bude rychlost dokonce nulová. Proto by senzor detekce osob umístěný kolmo k vozovce snadno detekoval projíždějící vozidlo jako chodce. Pro správné rozlišení chodce od vozidla musí být provedeno další vyhodnocení, např. pro projíždějící vozidlo bude platit krátký časový okamžik detekce než pro procházejícího chodce. Další možnost se naskýtá ve

skutečnosti, že přijíždějící automobil bude nejprve zachycen detektorem vozidel a až poté bude detekován jako chodec na přechodu, kterým je ve skutečnosti projíždějící vozidlo. Obě tyto skutečnosti znamenají, že konečné rozhodnutí o výskytu chodce na přechodu se přesouvá na centrální jednotku, která z doby výskytu nebo z projíždějícího vozidla rozhodne, zda se na přechodu vyskytuje chodec nebo ne.

Minimální frekvence bude u obou typů senzorů (senzor pro detekci osob a pro detekci vozidel) co možná nejmenší, aby se zpracovávaly i pomalé pohyby. Maximální frekvence budou voleny tak, aby odpovídali maximální rychlosti daného uvažovaného objektu. Pro detektor pohybu osob zvolíme takovou rychlost chůze, aby byly detekovány i rychle se pohybující osoby, např. přeběhnutí silnice apod. Průměrná rychlost chůze člověka se pohybuje okolo 3 $km \cdot h^{-1}$, zvolíme rychlost o něco vyšší – 7 $km \cdot h^{-1}$, této rychlosti odpovídá rychlost zhruba 2 $m \cdot s^{-1}$.

Maximální rychlost pro detektor pohybu vozidel bude odvozena od maximální povolené rychlosti, která je v České republice dána legislativou. V obci je maximální přípustná rychlost 50 $km \cdot h^{-1}$ a lze ji místním značením navýšit o 30 $km \cdot h^{-1}$ na 80 $km \cdot h^{-1}$. Maximální rychlost na rychlostních silnicích (dálnice a rychlostní silnice) je stanovena na hodnotu 130 $km \cdot h^{-1}$. Bohužel řidiči na silnicích povolené rychlosti často nedodržují, proto bude senzor detekce vozidel snímat vozidla pohybující se až rychlostí 150 $km \cdot h^{-1}$. Tato hodnota by měla zaručit detekci každého vozidla. Rychlosti 150 $km \cdot h^{-1}$ odpovídá rychlosti zhruba 42 $m \cdot s^{-1}$. Dané frekvenci odpovídá maximální rychlost vypočtená z rovnice (2.22).

Maximální frekvence detektorů shrnuje následující tabulka (frekvence senzoru je 24,2 GHz, viz dále):

Tabulka 2.1 – Přehled maximálních frekvencí pro detektory

Typ detektoru	**Maximální rychlost**	**Maximální frekvence**
Detektor osob	2 $m \cdot s^{-1}$	322,6667 Hz
Detektor vozidel	42 $m \cdot s^{-1}$	6776 Hz (7000 Hz)

Maximální frekvence pro detektor pohybu osob je přesně určena na hodnotu 322 Hz a není vhodné ji navyšovat, aby nezačaly být zpracovávány i jiné objekty, např. pomalá vozidla. Naopak maximální frekvenci pro detekci vozidel je možné

navýšit na hodnotu 7000 Hz. Tím budou detekovány i objekty pohybující se rychleji než rychlostí 150 km·h^{-1}.

2.4 Detekce cíle

Každý signál je zatížen šumem. Proto je důležité umět rozhodnout, zda signál obsahuje signál určený pohybem cíle nebo pouze šum. Šum se uvažuje bílý, tedy s normálním rozdělením pravděpodobnosti a nulovou střední hodnotou a konstantním frekvenčním spektrem pro všechny frekvence.

Signál cíle je detekován překročením prahu detekce označeným U_D [V]. Pro práh umístěný nízko má senzor vysokou citlivost, ovšem může se stát, že náhodný šum práh překročí a tím detekuje cíl, který nebyl ve skutečnosti přítomen, proto se definuje tzv. pravděpodobnost falešného poplachu. Pravděpodobnost falešného poplachu P_n se vypočte podle vztahu (2.23):

$$P_n = e^{-\frac{U_D^2}{2 \cdot N_e}} \tag{2.23}$$

P_n je pravděpodobnost falešného poplachu,

U_D je hodnota prahu [V],

N_e je výkon šumu [V^2].

Pravděpodobnost falešného poplachu lze také určit vztahem (2.24),

$$P_n = \frac{\sum_{k=1}^{N} t_k}{\sum_{k=1}^{N} T_k} \tag{2.24}$$

$\sum_{k=1}^{N} t_k$ jsou veškeré časové okamžiky překročení prahu bez přítomnosti cíle [s],

$\sum_{k=1}^{N} T_k$ jsou veškeré časové okamžiky [s].

Výkon šumu, který je reprezentován jednotlivými vzorky je možné vypočítat následujícím vztahem:

$$N_e = \frac{1}{N} \cdot \sum_{k=1}^{N} U_{N_i}^2 \quad [V^2] \tag{2.25}$$

N je počet vzorků šumu [-],

U_{N_i} je hodnota i-tého vzorku šumu [V].

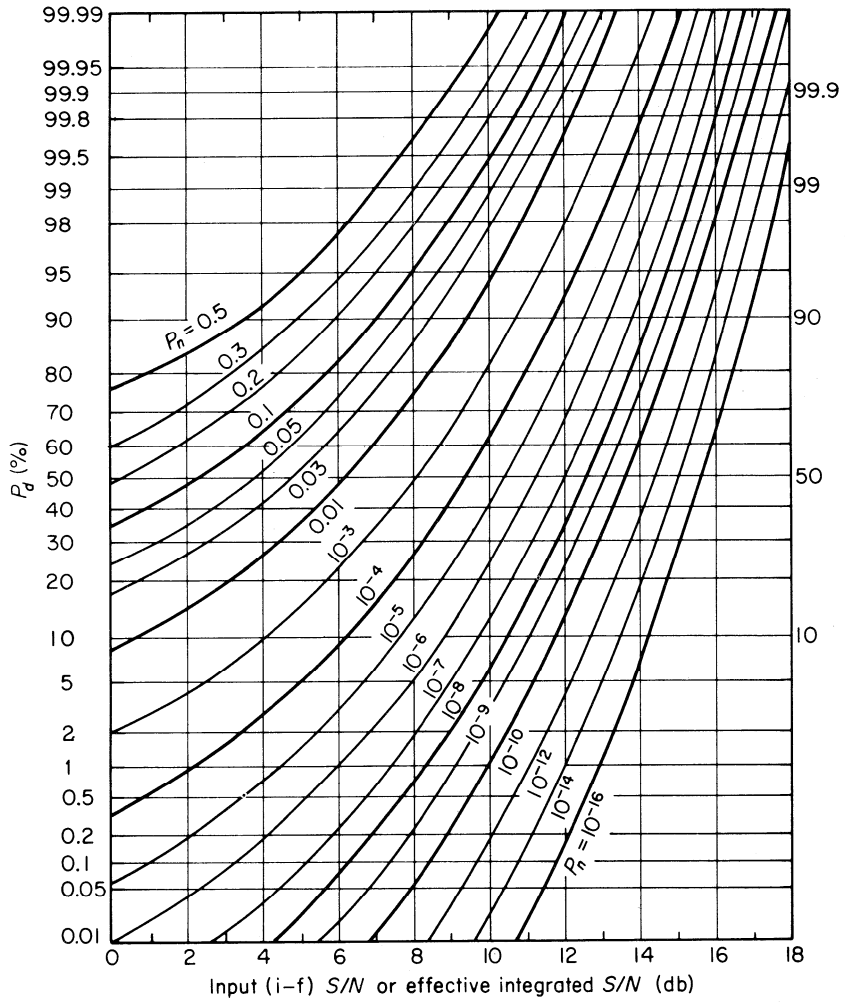
Protože je výstup ze zesilovače (viz kapitola 5) stejnosměrně posunut na polovinu napájecího napětí, je nutné od hodnoty vzorku odečíst střední hodnotu.

$$N_e = \frac{1}{N} \cdot \sum_{k=1}^{N} \left(U_{N_i} - U_E\right)^2 \quad [V^2] \tag{2.26}$$

U_E je střední hodnota signálu [V].

V případě přítomnosti stejnosměrné složky, která byla pro výpočet výkonu šumu odečtena, je nutné po výpočtu hodnoty prahu U_D tuto střední hodnotu opět přičíst.

Bude-li připuštěna celková doba překročení prahu detekce 50 ms za dobu 24 hodin provozu, bude pravděpodobnost falešného poplachu podle vztahu (2.24) rovna $P_n = 0{,}6 \cdot 10^{-6}$. Ze známé hodnoty pravděpodobnosti falešného poplachu je možné stanovit práh detekce (rovnice (2.23)) a z grafu z [21] (obrázek 2.3) lze stanovit poměr signálu k šumu S/N.



Obrázek 2.3 – Určení poměru signálu k šumu S/N [21]

Pro pravděpodobnost detekce cíle $P_d = 99\%$ a zmíněné pravděpodobnosti falešného poplachu $P_n = 0{,}6 \cdot 10^{-6}$ vychází poměr S/N přibližně $14{,}5dB$.

3 Výpočet frekvenčního spektra signálu

Vztah pro výstupní signál z mikrovlnného dopplerova senzoru (2.18) ukazuje, že frekvence signálu je přímo úměrná rychlosti pohybujícího se objektu (viz rovnice (2.4)). Proto je vhodné pro některé výpočty použít frekvenční spektrum.

3.1 Fourierova transformace

Pro výpočet frekvenčního spektra spojitého neperiodického signálu slouží Fourierova transformace určená vztahem (3.1):

$$S(\omega) = \int_{-\infty}^{+\infty} s(t) \cdot e^{-j \cdot \omega \cdot t} \cdot dt \tag{3.1}$$

$S(\omega)$ je frekvenční spektrum,

$s(t)$ je signál, ze kterého se vypočte frekvenční spektrum [V],

ω je úhlový kmitočet [rad·s^{-1}],

t je jednotka času [s].

Výsledné spektrum je spojité a je komplexní v rozmezí frekvencí $\omega \epsilon (-\infty, +\infty)$. S komplexním číslem se hůře pracuje než s reálným číslem, proto se určuje amplitudové $|S(\omega)|$ a lze vyjádřit i fázové spektrum $\varphi(\omega)$:

$$|S(\omega)| = \left| \int_{-\infty}^{+\infty} s(t) \cdot e^{-j \cdot \omega \cdot t} \cdot dt \right| \tag{3.2}$$

$$\varphi(\omega) = \arg\big(S(\omega)\big) \tag{3.3}$$

3.2 Diskrétní Fourierova transformace

Protože pro digitální zpracování analogového signálu není možné zpracovávat přímo analogový signál, ale pouze diskrétní hodnoty vzorků analogového signálu, byla odvozena diskrétní Fourierova transformace DFT:

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot W_N^{nk} \tag{3.4}$$

$X(k)$ je frekvenční spektrum,

n je pořadí vzorků,

$x(n)$ jsou vzorky vstupního signálu,

N je počet zpracovávaných vzorků,

W_N^{nk} je komplexní exponenciála.

$$W_N^{nk} = e^{\frac{-j \cdot 2\pi \cdot n \cdot k}{N}} \tag{3.5}$$

Výsledné frekvenční spektrum je opět komplexní a není spojité, ale je reprezentováno jednotlivými frekvenčními čarami. Pořadí frekvenčních čar je řazeno tak, že nejprve se vyskytují kladné frekvence od $\langle 0, \frac{f_{vz}}{2} \rangle$ [Hz], kde f_{vz} je velikost vzorkovací frekvence v [Hz], následují záporné frekvence $\langle -\frac{f_{vz}}{2}, 0 \rangle$ [Hz] (frekvence $-\frac{f_{vz}}{2}$ představuje jednu hodnotu). Spektrum lze tedy rozdělit na dvě poloviny, na kladné a záporné frekvence, kladné frekvence jsou o jednu spektrální čáru delší (stejnosměrná složka) oproti záporným frekvencím. Až na tento rozdíl nabývají obě poloviny stejných hodnot pro reálný signál, kde je spektrum symetrické kolem vertikální nulové osy. Výpočet diskrétní Fourierovy transformace je velice početně náročný, je nutné provést řadu komplexních násobení a součtů. Proto byla odvozena rychlá Fourierova transformace (FFT).

3.3 Rychlá Fourierova transformace

Rychlá Fourierova transformace je přímo určena pro digitální zpracování signálů. Pro výpočet výsledného frekvenčního spektra je využito rozdělení výpočtu diskrétní Fourierovy transformace na dílčí menší Fourierovy transformace.

Pro výpočet frekvenčního spektra bude použit algoritmus DIF (Decimation In Frequency) Radix-4. Tento algoritmus má menší vliv na výsledný šum, který je zaváděn při výpočtech. Při výpočtu je použit zlomkový formát čísel. Při sčítání nebo odečítání čísel ve zlomkovém formátu nedochází ke kvantovacímu šumu, k němu dochází pouze při násobení. Při násobení je část výsledného čísla oříznuta, čímž

dochází ke vzniku kvantovacího šumu. Označení Radix-4 určuje, že jsou najednou zpracovány čtyři vzorky vstupního signálu. Popis algoritmu lze nalézt v [22]. Nejprve se frekvenční spektrum rozdělí na čtvrtiny:

$$
\begin{aligned}
X(k) = & \\
& = \sum_{n=0}^{\frac{N}{4}-1} x(n) \cdot W_N^{nk} + \sum_{\frac{N}{4}}^{\frac{2N}{4}-1} x(n) \cdot W_N^{nk} + \sum_{\frac{2N}{4}}^{\frac{3N}{4}-1} x(n) \cdot W_N^{nk} + \\
& + \sum_{\frac{3N}{4}}^{N-1} x(n) \cdot W_N^{nk} = \\
& = \sum_{n=0}^{\frac{N}{4}-1} x(n) \cdot W_N^{nk} + \sum_{n=0}^{\frac{N}{4}-1} x\left(n + \frac{N}{4}\right) \cdot W_N^{\left(n+\frac{N}{4}\right)k} + \\
& + \sum_{n=0}^{\frac{N}{4}-1} x\left(n + \frac{N}{2}\right) \cdot W_N^{\left(n+\frac{N}{2}\right)k} + \sum_{n=0}^{\frac{N}{4}-1} x\left(n + \frac{3N}{4}\right) \cdot W_N^{\left(n+\frac{3N}{4}\right)k} = \\
& = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + x\left(n + \frac{N}{4}\right) \cdot W_N^{\left(n+\frac{N}{4}\right)k} + x\left(n + \frac{N}{2}\right) \cdot W_N^{\left(n+\frac{N}{2}\right)k} + \right. \\
& \left. + x\left(n + \frac{3N}{4}\right) \cdot W_N^{\left(n+\frac{N}{2}\right)k} \right] \cdot W_N^{nk}
\end{aligned}
\tag{3.6}
$$

Výraz lze dále zjednodušit zavedením následujících výrazů:

$$W_N^{\frac{N}{4}k} = \left[\cos\left(\frac{\pi}{2}\right) - j \cdot \sin\left(\frac{\pi}{2}\right)\right]^k = (-j)^k \tag{3.7}$$

$$W_N^{\frac{N}{2}k} = [\cos(\pi) - j \cdot \sin(\pi)]^k = (-1)^k \tag{3.8}$$

$$W_N^{\frac{3N}{4}k} = \left[\cos\left(\frac{3\pi}{2}\right) - j \cdot \sin\left(\frac{3\pi}{2}\right)\right]^k = j^k \tag{3.9}$$

Vztah pro rychlou Fourierovu transformaci přejde do tvaru:

$$X(k) = \\ = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + (-j)^k \cdot x\left(n + \frac{N}{4}\right) + (-1)^k \cdot \right. \\ \left. \cdot x\left(n + \frac{N}{2}\right) + (j)^k \cdot x\left(n + \frac{3N}{4}\right) \cdot W_N^{nk} \right] \cdot W_N^{nk} \tag{3.10}$$

Výsledný vztah pro frekvenční spektrum se následně rozdělí na jednotlivé frekvenční čáry, pro algoritmus Radix-2 se rozdělují na sudé a liché:

$$X(4k) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) + x\left(n + \frac{3N}{4}\right) \right] \cdot W_{\frac{N}{4}}^{nk}$$

$$X(4k+1) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) - j \cdot x\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) + j \cdot x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^n \cdot W_{\frac{N}{4}}^{nk}$$

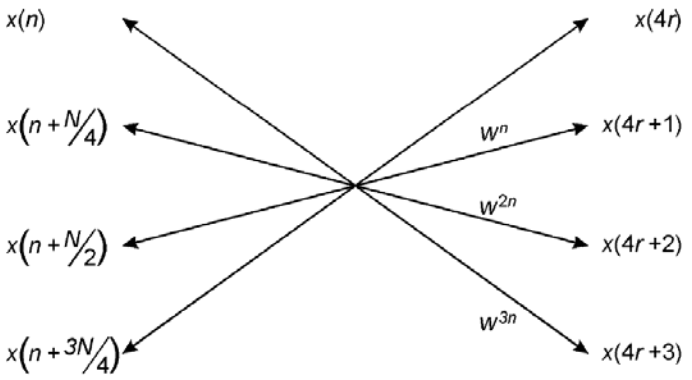
$$X(4k+2) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) - x\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) - x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{2n} \cdot W_{\frac{N}{4}}^{nk} \tag{3.11}$$

$$X(4k+3) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + j \cdot x\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) - j \cdot x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{3n} \cdot W_{\frac{N}{4}}^{nk}$$

pro $k = 0$ do $\frac{N}{4} - 1$

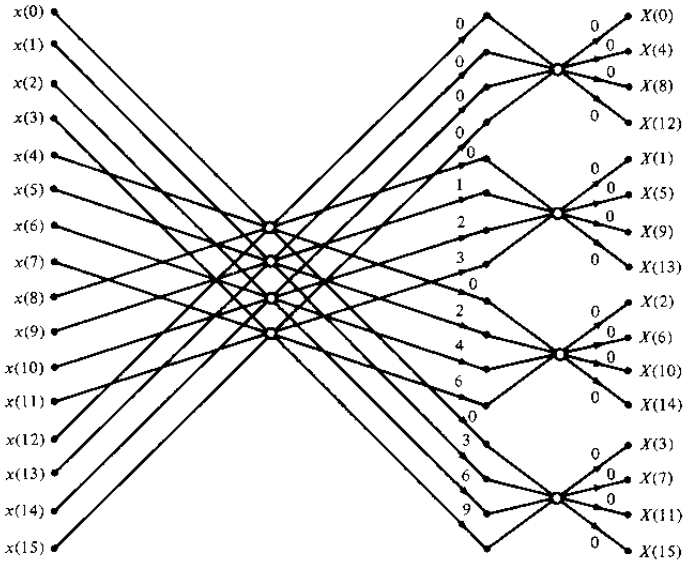
$X(4k)$, $X(4k+1)$, $X(4k+2)$ a $X(k+3)$ jsou dílčí diskrétní Fourierovy transformace o počtu $\frac{N}{4}$ vzorků.

Rychlá Fourierova transformace se graficky zakresluje pomocí tzv. motýlků. Motýlek pro výše odvozenou rychlou Fourierovu transformaci je ukázán na následujícím obrázku:



Obrázek 3.1 – Grafické znázornění FFT DIF Radix-4 [22]

Příklad výpočtu rychlé Fourierovy transformace algoritmem DIF Radix-4 ze 16 vzorků signálu ukazuje následující obrázek:



Obrázek 3.2 – Grafická ukázka výpočtu FFT DIF Radix-4 ze 16 vzorků signálu [23]

Obrázek ukazuje, že vstupní posloupnost vzorků $x(n)$ je seřazena sestupně, zatímco výstupní posloupnost frekvenčních čar $X(k)$ je řazena reverzně.

3.4 Reverzní adresování

Výsledná posloupnost spektrálních čar algoritmu rychlé Fourierovy transformace DIF je seřazena reverzně. Reverzní adresování je ukázáno na následujícím obrázku:

Normální adresování		Reverzní adresování	
Binární číslo	Dekadické číslo	Binární číslo	Dekadické číslo
000	0	000	0
001	1	100	4
010	2	010	2
011	3	110	6
100	4	001	1
101	5	101	5
110	6	011	3
111	7	111	7

Obrázek 3.3 – Ukázka reverzního adresování pro FFT DIF pro 8 vzorků

Reverzní adresování jsou pouze čtené hodnoty binárních bitů v obráceném pořadí, neboli normální adresování je čteno zleva doprava, pro reverzní adresování jsou bity čteny zprava doleva.

3.5 Výpočet amplitudového frekvenčního spektra

Výpočet amplitudového spektra je ukázán rovnicí (3.2). Absolutní hodnota komplexního čísla je určena následujícím vztahem:

$$|C| = \sqrt{\mathrm{Re}(C)^2 + \mathrm{Im}(C)^2} \tag{3.12}$$

C je obecné komplexní číslo,

$\mathrm{Re}(C)$ je operátor pro reálnou část komplexního čísla,

$\mathrm{Im}(C)$ je operátor pro imaginární část komplexního čísla.

Výpočet odmocniny je pro většinu mikrokontrolérů náročný, protože neexistuje instrukce umožňující tento výpočet. Existují sice algoritmy pro určení hodnoty odmocniny, ale tyto algoritmy trvají delší dobu. Z těchto důvodů se pro výpočet amplitudového spektra spíše používá výpočet výkonového spektra $|X(k)|^2$. Tímto algoritmem odpadá nutnost použití odmocniny. Hodnoty výkonového spektra se pochopitelně liší od hodnot amplitudového spektra, ovšem mocnina je přímo

úměrné původním hodnotám, závislost sice není lineární, ale to nemá na potřebné výpočty vliv.

4 Napájecí zdroje

Elektronické obvody je nutné napájet vhodným napájecím zdrojem. Dnes se používají dva základní typy napájecích zdrojů. Prvním a nejstarším typem jsou lineární zdroje, které využívají pro regulaci výstupního napětí aktivních prvků, např. tranzistorů nebo zenerových diod apod. Druhým a modernějším typem jsou spínané zdroje dosahující vyšší účinnosti v porovnání s lineárními zdroji. Tyto zdroje využívají pro transformaci energie ze vstupu na výstup indukčnost.

Pro výběr vhodného napájecího zdroje je nutné specifikovat požadavky, které musí zdroj splňovat:

- vstupní napětí zdroje se musí pohybovat kolem napětí 3,6 V, které odpovídá nominální hodnotě Li-onové baterie, ze které bude senzorová jednotka napájena,
- minimální vstupní napětí by mělo být co nejmenší, aby byla zajištěna správná funkce i při ne zcela nabité baterii,
- vysoká účinnost zdroje,
- schopnost zdroje vytvořit požadované výstupní napětí při libovolné hodnotě vstupního napětí zdroje.

Li-onová baterie je vybrána pro její výhody, mezi které patří vysoká kapacita, malé samovybíjení, nemají paměťový efekt.

Pro splnění požadavku vysoké účinnosti a schopnosti zvýšení nebo snížení výstupního napětí oproti hodnotě vstupního napětí byla určena volba pro výběr z některých typů spínaných zdrojů (DC/DC měničů).

4.1 Principy DC/DC měničů

DC/DC měniče (spínané zdroje) jsou takové zdroje, které mají vstupní i výstupní napětí stejnosměrné. V porovnání s klasickými lineárními zdroji se odlišují ve způsobu transformace vstupního napětí na výstupní. Spínané zdroje jsou v dnešní době velmi používanými zdroji, protože se vyznačují několika podstatnými výhodami, mezi které zejména patří vysoká účinnost uplatňující se zejména při

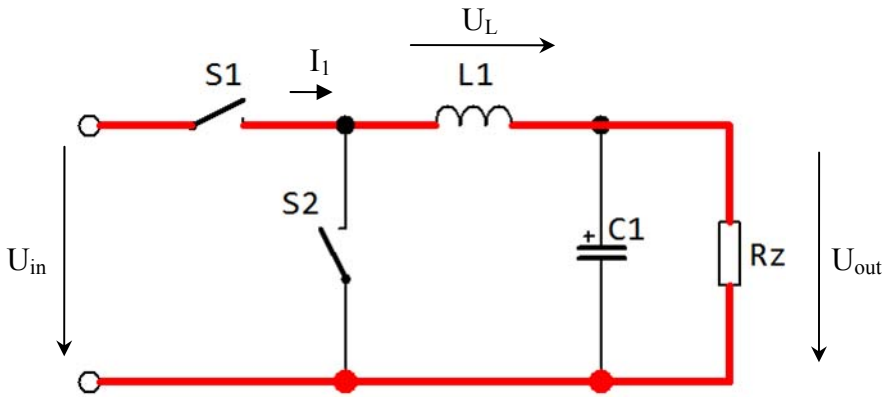
napájení z baterií. Vyšší účinnosti se dosahuje použitím vyšší spínací frekvence, pro kterou musejí být navrženy použité součástky. Dnes prodávané spínané zdroje v podobě integrovaného obvodu potřebují pro svoji činnost minimální počet okolních součástek. Velikost spínané frekvence souvisí s další výhodou DC/DC měničů, zvlnění výstupního napětí lze lépe filtrovat.

DC/DC měniče se rozlišují podle velikosti výstupního napětí v porovnání se vstupním napětím. Proto existují měniče snižující, které mají výstupní napětí vždy nižší než napětí vstupní, měniče zvyšující, kde je naopak výstupní napětí vyšší než vstupní a měniče inverzní, u kterých má výstupní napětí opačnou polarizaci než napětí vstupní. Podrobné informace o DC/DC měničích lze nalézt v [4].

4.1.1 Snižující měnič

První typ měniče je snižující, označovaný jako BUCK. U tohoto typu měniče je výstupní napětí vždy nižší než napětí vstupní. U všech typů měničů se pro transformaci vstupního napětí na napětí výstupní používá indukčnost. Dalším důležitým prvkem je spínač, pomocí kterého se indukčnost připíná ke vstupnímu zdroji apod.

Popis snižujícího měniče lze rozdělit do dvou částí. První fáze je znázorněna na obrázku 4.1:



Obrázek 4.1 – Snižujícího měniče – první fáze

V této fázi je sepnut spínač S1. Proud ze vstupního zdroje U_{in} prochází přes indukčnost a dále do zátěže Rz. Bude-li spínač sepnut po dobu t_1, lze napsat následující vztahy:

$$U_L = L \cdot \frac{dI_1}{dt} \; [V] \tag{4.1}$$

$$I_1 = \frac{1}{L} \cdot \int_0^{t_1} (U_{in} - U_{out}) \cdot dt = \frac{(U_{in} - U_{out}) \cdot t_1}{L} \; [A] \tag{4.2}$$

L je indukčnost cívky [H],

U_L je napětí na cívce [V],

I_1 je proud procházející obvodem [A],

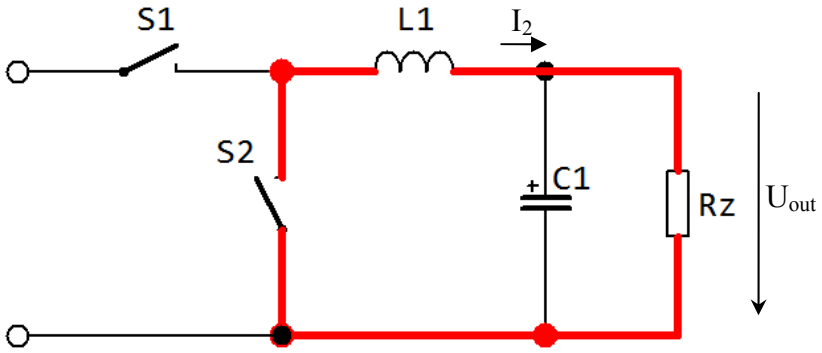
U_{in} je napětí vstupního zdroje [V],

U_{out} je výstupní napětí zdroje [V],

t_1 je doba sepnutí spínače S1 [s].

Předchozí vztahy přepokládají konstantní napětí na vstupu i na výstupu měniče, proto je napětí U_L na cívce také konstantní, proud obvodem lineárně narůstá podle vztahu (4.2).

V další fázi je spínač S1 rozepnut a sepnut je spínač S2. Tuto situaci vystihuje následující obrázek:



Obrázek 4.2 – Snižujícího měniče – druhá fáze

Cívka se snaží udržet proud protékající v předchozí fázi, ale v tomto případě je cívka zdrojem, proto bude proud cívkou klesat:

$$I_2 = \frac{1}{L} \cdot \int_0^{t_2} U_{out} \cdot dt = \frac{U_{out} \cdot t_2}{L} \; [A] \tag{4.3}$$

t_2 je doba sepnutí spínače S2 [s].

Za předpokladu celkového využití energie nasbírané cívkou v předchozí fázi lze napsat následující podmínku:

$$I_1 - I_2 = 0 \tag{4.4}$$

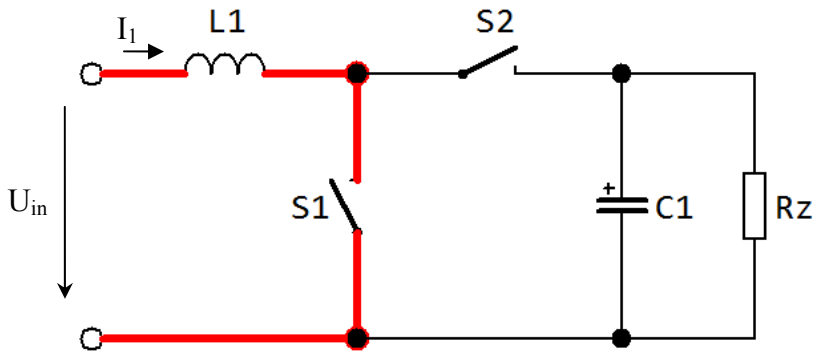
$$\frac{(U_{in} - U_{out}) \cdot t_1}{L} - \frac{U_{out} \cdot t_2}{L} = 0 \tag{4.5}$$

$$U_{out} = U_{in} \cdot \frac{t_1}{t_1 + t_2} \; [V] \tag{4.6}$$

Z rovnice (4.6) je zřejmé, že výstupní napětí bude vždy nižší než napětí vstupní. Součet časů t_1 a t_2 můžeme nazvat periodou spínání. Budeme-li dobu t_1 prodlužovat při zachování periody spínání, bude výstupní napětí vyšší a naopak snižování této doby bude způsobovat snižování výstupního napětí.

4.1.2 Zvyšující měnič

Dalším typem měniče je zvyšující měnič (označovaný také jako BOOST), u kterého je výstupní napětí vyšší, než je napětí vstupní. Popis činnosti je dobré opět rozdělit na dvě fáze jako u předchozího typu měniče:



Obrázek 4.3 – Zvyšujícího měniče – první fáze

První fáze zvyšujícího měniče je znázorněna na obrázku 4.3, kde je sepnut spínač S1 a indukčnost L1 je připojena přímo ke vstupnímu zdroji napětí. Bude-li spínač sepnut po dobu t_1, bude proud I_1 roven:

$$I_1 = \frac{1}{L} \cdot \int_0^{t_1} U_{in} \cdot dt = \frac{U_{in} \cdot t_1}{L} \; [A] \tag{4.7}$$

I_1 je proud procházející cívkou [A],

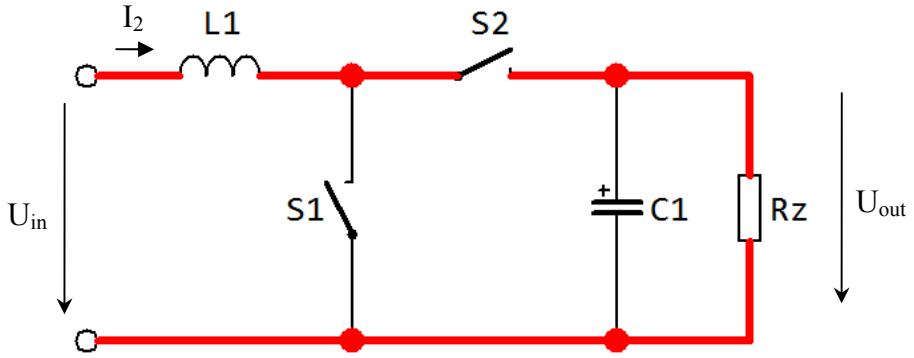
L je indukčnost cívky [H],

U_{in} je velikost vstupního napětí [V],

t_1 je doba sepnutí spínače S1 [s].

I v tomto případě je cívka připojena na konstantní zdroj napětí, který je tvořen vstupním napětím U_{in}, proto proud I_1 lineárně narůstá podle vztahu (4.7).

V druhé fázi je spínač S1 rozepnut a sepnut je spínač S2 po dobu t_2. Cívka má opět snahu udržet původní hodnotu proudu i jeho směr. Cívka v této fázi tvoří zdroj proudu, protože je v sérii se zdrojem vstupního napětí, výstupní napětí je vyšší. Situaci vystihuje následující obrázek:



Obrázek 4.4 – Zvyšujícího měniče – druhá fáze

Proud cívkou nyní klesá, za předpokladu konstantního vstupního i výstupního napětí, tedy i konstantního napětí na cívce platí následující vztah:

$$I_2 = \frac{1}{L} \cdot \int_0^{t_2} (U_{out} - U_{in}) \cdot dt = \frac{(U_{out} - U_{in}).\, t_2}{L}\ [A] \tag{4.8}$$

U_{out} je velikost výstupního napětí [V],

t_2 je doba sepnutí spínače S2 [s].

Předchozí rovnice platí za předpokladu, že je výstupní napětí vyšší, než je napětí vstupní, tato podmínka vyplývá z požadavku na funkci měniče. Bude-li veškerá energie cívky dodaná v předchozí fázi spotřebována, bude platit tato rovnost:

$$I_1 - I_2 = 0 \tag{4.9}$$

$$\frac{U_{in} \cdot t_1}{L} = \frac{(U_{out} - U_{in}) \cdot t_2}{L} = 0 \tag{4.10}$$

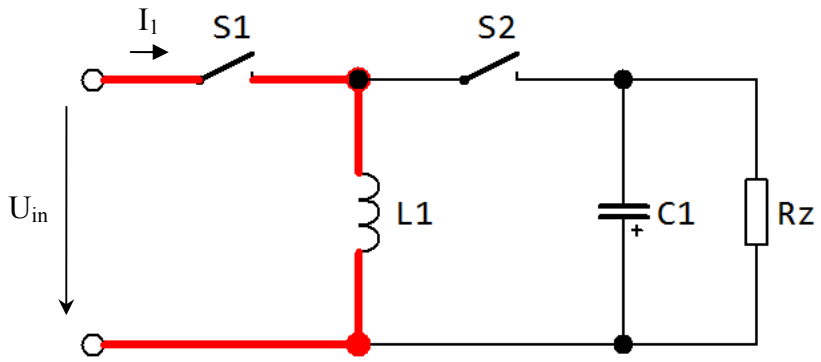
$$U_{out} = U_{in} \cdot \frac{t_1 + t_2}{t_2}\ [V] \tag{4.11}$$

Výsledný vztah (4.11) pro výstupní napětí U_{out} ukazuje, že výstupní napětí bude vždy vyšší než napětí vstupní. Bude-li perioda spínání konstantní a bude se zkracovat doba sepnutí t_2 spínače S2, výstupní napětí bude narůstat. Naopak při prodlužování doby t_2 výstupní napětí klesá.

4.1.3 Invertující měnič

Posledním typem měniče je invertující měnič, který umožňuje nastavit hodnotu výstupního napětí větší nebo menší oproti hodnotě vstupního proudu, ovšem s opačnou polaritou napětí. Tento měnič bývá také označován jako BUCK-BOOST.

Popis měniče je možné provést ve dvou fázích. První je znázorněna na obrázku 4.5, v této fázi je sepnut spínač S1 po dobu t_1 a indukčnost je přímo připojena ke vstupnímu napájecímu zdroji:



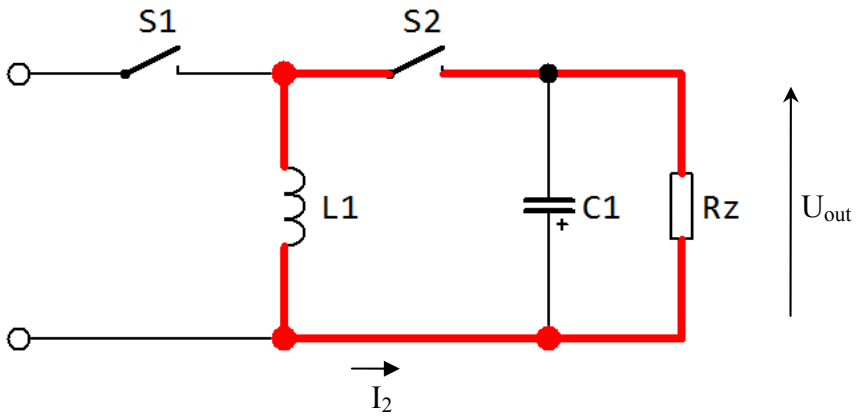
Obrázek 4.5 – Invertující měnič – první fáze

Vztah pro proud I_1 je stejný s první fází zvyšujícího měniče:

$$I_1 = \frac{U_{in} \cdot t_1}{L} \; [A] \tag{4.12}$$

I_1 je proud cívkou [A],
U_{in} je vstupní napětí [V],
L je indukčnost cívky [H],
t_1 je doba sepnutí spínače S1 [s].

Ve druhé fázi, která je naznačena na obrázku 4.6, je spínač S1 rozepnut a sepnut je spínač S2 po dobu t_2, zátěží prochází proud opačným směrem, proto je i výstupní napětí orientováno opačně.



Obrázek 4.6 – Invertující měnič – druhá fáze

Pro uvažované konstantní výstupní napětí proud I_2 lineárně klesá podle vztahu (4.13):

$$I_2 = \frac{1}{L} \cdot \int_0^{t_2} -U_{out} \cdot dt = -\frac{U_{out} \cdot t_2}{L} \tag{4.13}$$

I_2 je proud procházející zátěží [A],

L je indukčnost cívky [H],

U_{out} je velikost výstupního napětí [V],

t_2 je doba sepnutí spínače S2 [s].

Pro hodnotu výstupního proudu potom platí:

$$\frac{U_{in} \cdot t_1}{L} + \frac{U_{out} \cdot t_2}{L} = 0 \tag{4.14}$$

$$U_{out} = -U_{in} \cdot \frac{t_1}{t_2} \tag{4.15}$$

U_{in} je velikost vstupního napětí [V].

Hodnotu výstupního napětí lze nastavit správným poměrem časů t_1 a t_2.

Předchozí text měl ukázat, že typ měniče je závislý na způsobu spínání indukčnosti. Použité vztahy předpokládaly, že všechna napětí v obvodu jsou konstantní, to ovšem není zcela pravda, protože narůstajícím proudem cívky narůstá i výstupní napětí (zátěží protéká vyšší proud a napětí je také vyšší), není tedy konstantní. Výstupní kondenzátor C1 má za úkol filtrovat výstupní napětí a také udržovat výstupní napětí konstantní, čímž napomáhá ke splnění požadavků na výstupní napětí.

5 Zesilovač mikrovlnného senzoru

Analogový signál z mikrovlnného senzoru nabývá velice malých hodnot, proto je nutné signál pro následující digitální zpracování vhodně zesílit. Pro zesílení analogového signálu se používá zesilovač, který obvykle využívá operační zesilovače. Zesilovač má kromě zesílení další důležitý úkol spočívající ve frekvenčním omezení signálu, zejména z toho důvodu, aby bylo možné rozlišit detekované objekty, a také proto, že následujícím stupněm zpracování signálu je digitální zpracování. Pro digitální zpracování musí být dodrženo Nyquistovo vzorkovací kritérium, které říká, že vzorkovací frekvence musí být minimálně dvakrát větší než největší frekvence vzorkovaného analogového signálu, signál musí mít známé frekvenční omezení.

Zesilovač realizující zároveň funkci filtru je tvořen pomocí operačních zesilovačů. Celý filtr je realizovaný jako filtr typu pásmová propust, skládá se z filtrů typu dolní a horní propust.

Na operační zesilovač jsou kladeny následující požadavky:

- schopnost napájení z jediného zdroje (úspora jednoho napájecího zdroje),
- napájení prostřednictvím napětí 5 V (stejným napětím je napájen mikrovlnný senzor),
- velká šířka pásma (zesilovač dosahuje vysokého zesílení),
- minimální proudový odběr (napájení z baterií).

Kladeným požadavkům nejlépe odpovídá operační zesilovač od společnosti Analog Devices AD8648ARZ, jehož celý popis lze najít v [6]. Výhodou tohoto zesilovače je, že v jediném pouzdře integrovaného obvodu jsou přítomny celkem čtyři operační zesilovače. Další výhodou je vlastnost Rail-to-Rail, tedy výstup operačního zesilovače se co nejvíce přibližuje hodnotě napájecího napětí.

Parametry operačního zesilovače AD8648ARZ shrnuje tabulka 5.1:

Tabulka 5.1 – Souhrn parametrů operačního zesilovače AD8648ARZ [6]

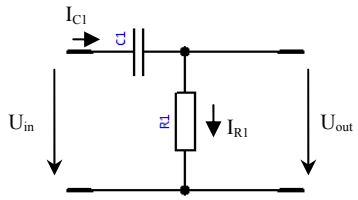
Parametr	**Minimální hodnota**	**Typická hodnota**	**Maximální hodnota**	**Jednotka**
Napájecí napětí	2,7		5,5	V
Šířka pásma		24		MHz
Proudový odběr na zesilovač		1,5	2	mA
Maximální výstupní napětí (I_{OUT} = 1mA)	4,98	4,99		V
Minimální výstupní napětí (I_{OUT} = 1mA)		8,4	20	mV

Operační zesilovače obvykle požadují napájení pomocí kladného a záporného napětí, to ovšem vyžaduje jeden napájecí zdroj navíc. Tento požadavek lze obejít, je možné použít operační zesilovač umožňující napájení pouze z jednoho zdroje, dále pro rozkmit výstupního signálu v celém rozsahu napájecího napětí je nutné na všechny vstupy operačního zesilovače přivést hodnotu rovnu polovičnímu napájecímu napětí, toto napětí je následně přítomno i na výstupu operačního zesilovače a představuje tzv. virtuální zem.

V následujících podkapitolách bude rozebrána funkce jednotlivých typů filtrů s odvozením vztahů pro mezní kmitočty a vztahů pro výpočet součástek určujících velikost mezních kmitočtů.

5.1 Filtr typu horní propust

Filtry typu horní propust nejsou ve výsledném filtru tvořeny pomocí operačního zesilovače, ale pouze pasivními součástkami – kondenzátorem a rezistorem. Pohled na filtr typu horní propust ukazuje obrázek 5.1:



Obrázek 5.1 – Určení přenosu filtru typu horní propust

Rozbor funkce tohoto filtru lze popsat takto: na nízkých frekvencích představuje kondenzátor vysokou impedanci a na výstupu filtru se nemůže objevit téměř žádný signál. S rostoucí frekvencí se impedance kondenzátoru snižuje, výstupní napětí filtru v závislosti na frekvenci vstupního signálu roste.

Analýzou toho filtru lze určit přesně mezní kmitočet. Přenos $H_{FH}(j\omega)$ tohoto filtru lze popsat následujícími rovnicemi:

$$U_{out} = U_{in} \cdot \frac{R1}{R1 + \frac{1}{j \cdot \omega \cdot C1}} \tag{5.1}$$

$$U_{out} = U_{in} \cdot \frac{j \cdot \omega \cdot C1 \cdot R1}{j \cdot \omega \cdot C1 \cdot R1 + 1} \tag{5.2}$$

$$H_{FH}(j\omega) = \frac{U_{out}}{U_{in}} = \frac{R1}{R1 + \frac{1}{j \cdot \omega \cdot C1}} = \frac{(j\omega) \cdot C1 \cdot R1}{(j\omega) \cdot C1 \cdot R1 + 1} \tag{5.3}$$

U_{out} je výstupní napětí filtru [V],
U_{in} je vstupní napětí filtru [V],
$R1$ je hodnota rezistoru [Ω],
$C1$ je hodnota kondenzátoru [F],
ω je frekvence vstupního signálu [rad·s^{-1}],
H_{FH} je přenos filtru.

Přenos je funkcí proměnné $j\omega$ a představuje racionální lomenou funkci, neboli podíl dvou polynomů proměnné $j\omega$ (čitatel je polynom nultého řádu).

Následně nás bude zajímat hodnota mezního kmitočtu ω_{0_H} tohoto typu filtru. Mezní kmitočet lze definovat jako kmitočet, na kterém poklesne amplitudový přenos filtru na hodnotu maxima amplitudového přenosu $|H_{FH}(j\omega)|_{max}$ vydělenou určitou

hodnotu k. Např. pro $k = 2$ bude mezní kmitočet představovat pokles na polovinu napětí oproti maximální hodnotě. Pro určení mezního kmitočtu musíme vyjádřit amplitudový přenos:

$$|H_{FH}(j\omega)| = \frac{R1}{\sqrt{R1^2 + \left(\frac{1}{\omega \cdot C1}\right)^2}} = \frac{\omega \cdot C1 \cdot R1}{\sqrt{\omega^2 \cdot (C1 \cdot R1)^2 + 1}} \tag{5.4}$$

$|H_{FH}|$ je amplitudový přenos filtru typu horní propust.

Dále musíme určit maximální hodnotu amplitudového přenosu $|H_{FH}(j\omega)|_{max}$. Amplitudový přenos filtru typu horní propust nabývá svého maxima pro $\omega \to \infty$, proto maximální hodnotu amplitudového přenosu vyjádříme následovně:

$$|H_{FH}(j\omega)|_{max} = \lim_{\omega \to \infty} |H_{FH}(j\omega)| =$$

$$= \lim_{\omega \to \infty} \frac{R1}{\sqrt{R1^2 + \frac{1}{\omega^2 \cdot C1^2}}} = 1 \tag{5.5}$$

Pro určení této hodnoty přenosu je výhodné vyjít z první části rovnice (5.4). Výsledek rovnice (5.5) ukazuje, že pro vysoké frekvence je výstupní napětí filtru rovno vstupnímu, tento fakt lze určit i ze základního předpokladu funkce filtru, kondenzátor pro vysoké frekvence představuje tak nízkou impedanci, kterou lze požadovat za zkrat, tedy výstupní napětí je rovno vstupnímu.

Následně se musí maximální hodnota amplitudového přenosu $|H_{FH}(j\omega)|_{max}$ snížit o určitý pokles vyjádřený konstantou k. Poté najdeme kmitočet, na kterém se rovná amplitudový přenos $|H_{FH}(j\omega)|$ sníženému maximálnímu přenosu $|H_{FH}(j\omega)|_{max}$ o hodnotu k:

$$|H_{FH}(j\omega)|_k = \frac{1}{k} \cdot |H_{FH}(j\omega)|_{max} = \frac{1}{k} \tag{5.6}$$

$$|H_{FH}(j\omega)| = |H_{FH}(j\omega)|_k \tag{5.7}$$

$$\frac{\omega \cdot C1 \cdot R1}{\sqrt{\omega^2 \cdot (C1 \cdot R1)^2 + 1}} = \frac{1}{k} \tag{5.8}$$

$$\frac{\omega^2 \cdot (C1 \cdot R1)^2}{\omega^2 \cdot (C1 \cdot R1)^2 + 1} = \frac{1}{k^2} \tag{5.9}$$

Výsledný vztah pro mezní kmitočet lze určit z předcházející rovnice (5.9):

$$\omega_{0_H} = \sqrt{\frac{1}{(C1 \cdot R1)^2 \cdot (k^2 - 1)}} \ [rad \cdot s^{-1}] \tag{5.10}$$

ω_{0_H} přestavuje mezní kmitočet filtru typu horní propust [rad·s^{-1}].

Hodnota rezistoru $R1$ bývá ovlivněna okolním zapojením, proto můžeme považovat jeho hodnotu za známou. Hodnotu kondenzátoru $C1$ lze podobně jako mezní kmitočet určit z rovnice (5.9):

$$C1 = \sqrt{\frac{1}{(k^2 - 1) \cdot \left(\omega_{0_H} \cdot R1\right)^2}} \tag{5.11}$$

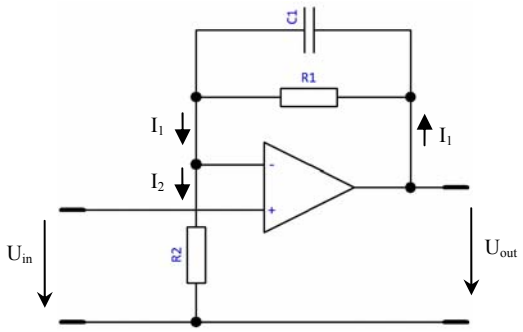
Předchozí vztahy platí pouze pro případ, že filtr není zatížen žádným proudem.

5.2 Filtr typu dolní propust

Filtry typu dolní propust již využívají operační zesilovače. Tyto zesilovače mají ve zpětné vazbě umístěný kondenzátor, který svoji frekvenčně závislou impedancí ovlivňuje zesílení zesilovače. V závislosti zapojení operačního zesilovače určujeme neinvertující a invertující zesilovač.

5.2.1 Neinvertující zesilovač – filtr typu dolní propust

Typické zapojení neinvertujícího zesilovače ve funkci filtru typu dolní propust ukazuje obrázek 5.2:



Obrázek 5.2 – Filtr typu dolní propust s operačním zesilovačem – neinvertující

Protože se kondenzátor $C1$ nachází ve zpětné vazbě operačního zesilovače, jeho impedance přímo ovlivňuje zesílení zesilovače. Na nízkých frekvencích je impedance kondenzátoru vysoká, proto se v paralelním zapojení s rezistorem $R1$ příliš neuplatňuje a zesílení je určeno pouze rezistory $R1$ a $R2$. Při zvyšování frekvence se impedance kondenzátoru snižuje a již se jeho impedance začíná uplatňovat na celkovém zesílení zesilovače. Pro vysoké frekvence se zesílení vlivem kondenzátoru snižuje, a tedy vysoké frekvence jsou méně zesilovány.

Následující vztahy při použití operačních zesilovačů předpokládají ideální operační zesilovač s těmito vlastnostmi:

- nulovou výstupní impedanci,
- nekonečnou vstupní impedanci,
- nulové napětí mezi invertujícím a neinvertujícím vstupem.

Proto se na invertujícím vstupu nachází stejné napětí jako na neinvertujícím vstupu. Následující vztahy uvažují všechny předchozí předpoklady:

$$I_2 = \frac{U_{in}}{R2} \tag{5.12}$$

$$I_1 = \frac{U_{out} - U_{in}}{\frac{R1 \cdot \frac{1}{j \cdot \omega \cdot C1}}{R1 + \frac{1}{j \cdot \omega \cdot C1}}} = \frac{U_{out} - U_{in}}{\frac{R1}{j \cdot \omega \cdot C1 \cdot R1 + 1}} \tag{5.13}$$

$$I_1 = I_2 \tag{5.14}$$

I_2 je proud rezistorem $R2$ [A],

U_{in} je vstupní napětí [V],

U_{out} je výstupní napětí filtru [V],

ω je kmitočet [rad·s^{-1}].

Dosazením rovnic (5.12) a (5.13) do rovnice (5.14) lze určit přenos filtru typu dolní propust $H_{FD}(j\omega)$:

$$\frac{\frac{U_{out} - U_{in}}{R1}}{j \cdot \omega \cdot C1 \cdot R1 + 1} = \frac{U_{in}}{R2} \tag{5.15}$$

$$H_{FD}(j\omega) = \frac{U_{out}}{U_{in}} = \frac{(j\omega) \cdot C1 \cdot R1 \cdot R2 + R1 + R2}{(j\omega) \cdot C1 \cdot R1 \cdot R2 + R2} \tag{5.16}$$

Stejně jako v případě filtru typu horní propust je nutné pro určení mezní frekvence vyjádřit amplitudový přenos $|H_{FD}(j\omega)|$:

$$|H_{FD}(j\omega)| = \sqrt{\frac{\omega^2 \cdot (C1 \cdot R1 \cdot R2)^2 + (R1 + R2)^2}{\omega^2 \cdot (C1 \cdot R1 \cdot R2)^2 + R2^2}} \tag{5.17}$$

Filtr typu dolní propust dosahuje maximálního amplitudového přenosu $|H_{FD}(j\omega)|_{max}$ pro nulovou frekvenci, proto stačí dosadit za $\omega = 0$ do rovnice (5.17):

$$|H_{FD}(j\omega)|_{max} = \frac{R1 + R2}{R2} \tag{5.18}$$

Vyjádření mezního kmitočtu ω_{0_D} provedeme obdobně jako u předchozího typu filtru. Proto musí platit následující vztahy (snížený maximální amplitudový přenos o hodnotu k označíme $|H_{FD}(j\omega)|_k$):

$$|H_{FD}(j\omega)|_k = \frac{1}{k} \cdot |H_{FD}(0)| = \frac{1}{k} \cdot \frac{R1 + R2}{R2} \tag{5.19}$$

$$|H_{FD}(j\omega)| = |H_{FD}(j\omega)|_k \tag{5.20}$$

$$\sqrt{\frac{\omega^2 \cdot (C1 \cdot R1 \cdot R2)^2 + (R1 + R2)^2}{\omega^2 \cdot (C1 \cdot R1 \cdot R2)^2 + R2^2}} = \frac{1}{k} \cdot \frac{R1 + R2}{R2} \tag{5.21}$$

$$\frac{\omega^2 \cdot (C1 \cdot R1 \cdot R2)^2 + (R1 + R2)^2}{\omega^2 \cdot (C1 \cdot R1 \cdot R2)^2 + R2^2} = \frac{1}{k^2} \cdot \frac{(R1 + R2)^2}{R2^2} \tag{5.22}$$

$$(R1 + R2)^2 \cdot R2^2 \cdot (k^2 - 1) =$$
$$= \omega^2 \cdot (R1 + R2)^2 \cdot (C1 \cdot R1 \cdot R2)^2 - \omega^2 . k^2 . R2^2 \cdot (C1 \cdot R1 \cdot R2)^2 \quad (5.23)$$

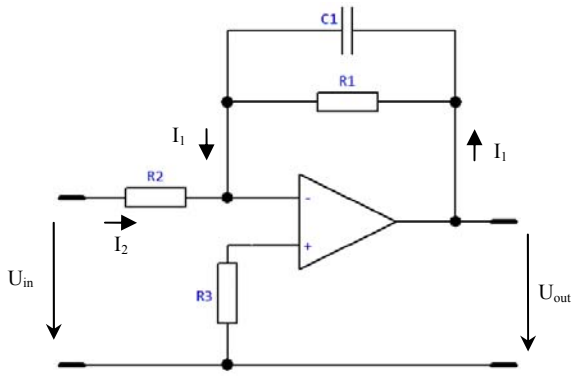
$$\omega_{0_D} = \sqrt{\frac{(R1 + R2)^2 \cdot R2^2 \cdot (k^2 - 1)}{(R1 + R2)^2 \cdot (C1 \cdot R1 \cdot R2)^2 - k^2 \cdot R2^2 \cdot (C1 \cdot R1 \cdot R2)^2}} \; [rad.s^{-1}] \quad (5.24)$$

Hodnotu kondenzátoru $C1$, která určuje velikost mezního kmitočtu ω_{0_D}, lze vyjádřit z rovnice (5.22):

$$C1 =$$
$$= \sqrt{\frac{(R1 + R2)^2 \cdot R2^2 \cdot (k^2 - 1)}{\omega^2 \cdot (R1 + R2)^2 \cdot (R1 \cdot R2)^2 - \omega^2 \cdot k^2 \cdot R2^2 \cdot (R1 \cdot R2)^2}} \; [F] \quad (5.25)$$

5.2.2 Intertující zesilovač – filtr typu dolní propust

Invertující zesilovač je velice podobný neinvertujícímu. Rozdíl je v přivedení vstupního signálu, u neinvertujícího zesilovače je signál převeden na neinvertující vstup, u invertujícího zesilovač je signál přiveden na invertující vstup. Schéma zapojení invertujícího zesilovače jako filtru typu dolní propust poskytuje obrázek 5.3:



Obrázek 5.3 – Filtr typu dolní propust s operačním zesilovačem – invertující

Kondenzátor $C1$ má opět vliv na zesílení pro různé frekvence. Vyšší frekvence jsou díky nízké impedanci kondenzátoru ve zpětné vazbě zesilovače méně zesilovány.

Přenos $H_{FD_{inv}}(j\omega)$ tohoto filtru lze za předpokladu ideálního operačního zesilovače vyjádřit pomocí následujících rovnic:

$$I_1 = \frac{U_{out}}{\frac{R1 \cdot \frac{1}{j \cdot \omega \cdot C1}}{R1 + \frac{1}{j \cdot \omega \cdot C1}}} = \frac{U_{out}}{\frac{R1}{j \cdot \omega \cdot C1 \cdot R1 + 1}} \tag{5.26}$$

$$I_2 = \frac{U_{in}}{R2} \tag{5.27}$$

$$I_2 = -I_1 \tag{5.28}$$

I_1 je proud z výstupu zesilovače do zpětné vazby [A],

I_2 je vstupní proud ze vstupu [A],

U_{out} je výstupní napětí [V],

U_{in} je vstupní napětí [V],

ω je kmitočet [rad·s^{-1}].

Do neinvertujícího vstupu operačního zesilovače díky nekonečné vstupní impedanci neteče žádný proud, proto je na invertujícím vstupu nulové napětí. Do invertujícího vstupu také neteče žádný proud, proto se proud ze vstupního zdroje rovná zápornému proudu z výstupu zesilovače.

Přenos filtru $H_{FD_{inv}}(j\omega)$ se rovná následujícímu vztahu:

$$H_{FD_{inv}}(j\omega) = \frac{U_{out}}{U_{in}} = -\frac{R1}{(j\omega) \cdot C1 \cdot R1 \cdot R2 + R2} \tag{5.29}$$

Ve výrazu pro přenos filtru se nevyskytuje hodnota rezistoru $R3$. Tento rezistor má za úkol minimalizovat vliv proudové nesymetrie reálného operačního zesilovače. Hodnota rezistoru $R3$ se určí jako paralelní kombinace rezistorů $R1$ a $R2$:

$$R3 = \frac{R1 \cdot R2}{R1 + R2} \tag{5.30}$$

Pro určení mezního kmitočtu je potřeba určit amplitudový přenos $\left|H_{FD_{inv}}(j\omega)\right|$ filtru:

$$\left|H_{FD_{inv}}(j\omega)\right| = \frac{R1}{\sqrt{(\omega \cdot C1 \cdot R1 \cdot R2)^2 + R2^2}} \tag{5.31}$$

Mezní kmitočet $\omega_{0_{FD_{inv}}}$ filtru určíme stejně jako v předchozích případech. Maximální hodnotu amplitudového přenosu označíme $\left|H_{FD_{inv}}(j\omega)\right|_{max}$, frekvenční přenos této hodnoty nabývá pro $\omega = 0$. Snížený amplitudový přenos o hodnotu k označíme $\left|H_{FD_{inv}}(j\omega)\right|_k$:

$$\left|H_{FD_{inv}}(j\omega)\right|_{max} = \frac{R1}{R2} \tag{5.32}$$

$$\left|H_{FD_{inv}}(j\omega)\right|_k = \frac{1}{k} \cdot \left|H_{FD_{inv}}(j\omega)\right|_{max} = \frac{1}{k} \cdot \frac{R1}{R2} \tag{5.33}$$

$$\left|H_{FD_{inv}}(j\omega)\right| = \left|H_{FD_{inv}}(j\omega)\right|_k \tag{5.34}$$

$$\frac{R1}{\sqrt{(\omega \cdot C1 \cdot R1 \cdot R2)^2 + R2^2}} = \frac{1}{k} \cdot \frac{R1}{R2} \tag{5.35}$$

$$\frac{R1^2}{(\omega \cdot C1 \cdot R1 \cdot R2)^2 + R2^2} = \frac{1}{k^2} \cdot \frac{R1^2}{R2^2} \tag{5.36}$$

$$R1^2 \cdot k^2 \cdot R2^2 = R1^2 \cdot [(\omega \cdot C1 \cdot R1 \cdot R2)^2 + R2^2] \tag{5.37}$$

$$\omega_{0_{FD_{inv}}} = \sqrt{\frac{(k^2 - 1)}{(R1 \cdot C1)^2}} \ [rad \cdot s^{-1}] \tag{5.38}$$

Hodnotu kondenzátoru C1 pro danou mezní frekvenci lze vyjádřit z rovnice (5.36):

$$C1 = \sqrt{\frac{(k^2 - 1)}{\left(R1 \cdot \omega_{0_{FD_{inv}}}\right)^2}} \ [F] \tag{5.39}$$

6 Obvodový popis senzorové jednotky

Každá senzorová jednotka se skládá ze stejných dílčích celků, bez rozdílu, zda se jedná o senzorovou jednotku určenou pro detekci osob či určenou pro detekci vozidel, pouze se liší hodnotami některých součástek a softwarovým vybavením.

Tyto celky jsou následující:

1. napájecí zdroj pro mikrovlnný senzor a zesilovač a napájecí zdroj pro mikroprocesorovou část jednotky,
2. mikrovlnný senzor pracující na principu Dopplerova jevu,
3. zesilovač zesilující signál mikrovlnného senzoru,
4. mikroprocesorová jednotka tvořena mikrokontrolérem LM3S811,
5. paměť EEPROM pro uložení konfigurace senzorové jednotky,
6. bezdrátový modul pro přenos informace do centrální jednotky.

Senzorová jednotka obsahuje dva zdroje, jeden je určený pro napájení analogové části skládající se z mikrovlnného senzoru a zesilovače. Druhý je určen pro digitální část obsahující mikrokontrolér, paměť EEPROM a bezdrátový modul IQRF. Použitý mikrovlnný senzor (popsaný v příloze I) se jmenovitou hodnotou napájení 5 V a zesilovač (viz kapitola 5) schopný také 5V napájení mají společný zdroj. Mikrokontrolér LM3811 (popsaný v příloze H), sériová paměť EEPROM 24LC04B (viz příloha F) a bezdrátový modul IQRF TR-53B (popsaný v příloze G) jsou napájeny ze zdroje s napětím 3,3 V.

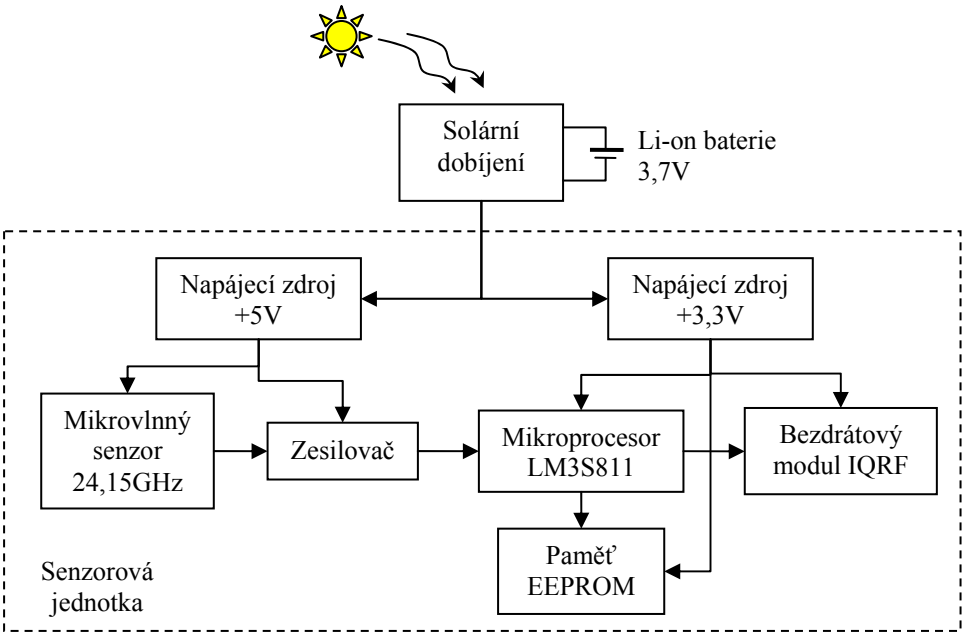
Mikrokontrolér není opatřen žádným typem paměti, který by umožňovat zápis vlastních dat, sice se dají nalézt zdrojové kódy umožňující zápis do programové paměti flash, ale tento způsob může být nebezpečný, mohlo by dojít k přepisu zdrojového kódu mikrokontroléru. Z tohoto důvodu jsou senzorové jednotky vybaveny sériovou pamětí EEPROM 24LC04B sloužící pro ukládání dat, které se nesmí po odpojení napájení vymazat a lze je měnit. Mezi tato data patří např. nastavení bezdrátové komunikace nebo nastavení pro analogové zpracování signálu senzoru. Výsledkem je, že zdrojový kód mikrokontroléru je pro daný typ jednotky neměnný a stačí pouze změnit obsah paměti EEPROM a bude možné změnit např.

adresu jednotky v rámci bezdrátové sítě, nebo hodnoty zpracování analogového signálu.

Bezdrátový modul zprostředkovává odesílání bezdrátových zpráv ze senzorové jednotky do centrální jednotky křižovatky. Pro snadný vývoj a vysoký bezdrátový dosah byly vybrány IQRF moduly společnosti Microrics.

Celé schéma je obsaženo v příloze A pro jednotku detekce osob a v příloze B pro jednotku detekce vozidel.

Zapojení těchto bloků ukazuje toto blokové schéma:

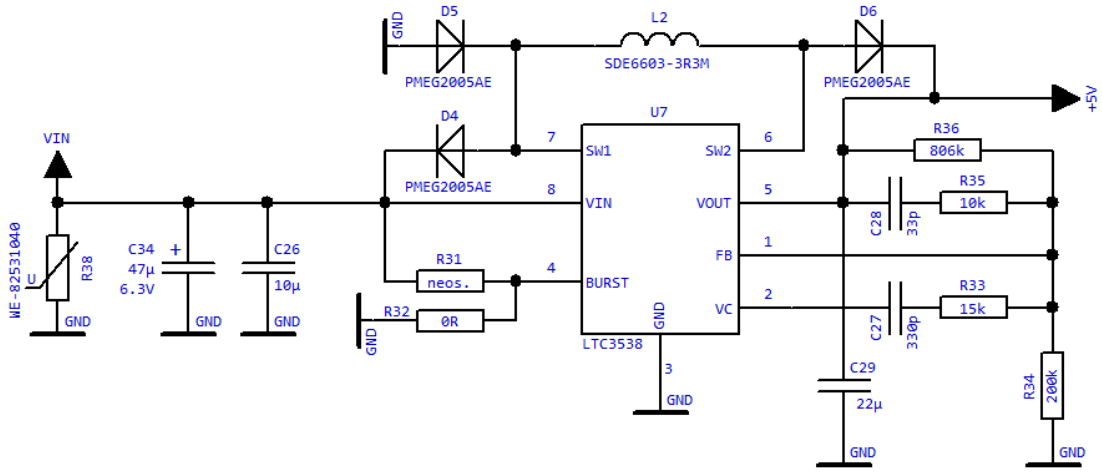


Obrázek 6.1 – Blokové schéma senzorové jednotky

6.1 Napájecí zdroj analogové části senzorové jednotky

Analogová část detektoru pohybu osob nebo detektoru vozidel je tvořena mikrovlnným senzorem a analogovým zesilovačem zesilujícím signál mikrovlnného senzoru. Protože je jmenovitá hodnota napětí mikrovlnného senzoru 5 V, je i zesilovač napájen napětím 5 V. Zesilovač by bylo možné napájet i napětím shodným s digitální částí, bez nutnosti napěťového přizpůsobení výstupního napětí zesilovače, při vývoji bylo ovšem zjištěno, že takovéto oddělení analogové a digitální části je vhodné kvůli snížení rušení signálu mikrovlnného senzoru.

Napájecí zdroje jsou tvořeny DC/DC měničem společnosti Linear Technology LTC3538 (příloha E).



Obrázek 6.2 – Schéma napájecího zdroje pro analogovou část detektoru

Schéma napájecího zdroje analogové části je převzato z dokumentace k obvodu LTC3538 [5]. Oproti dokumentaci je zapojení rozšířeno o tři ochranné diody označené D4, D5 a D6. Tyto diody omezují případné napěťové špičky vznikající na indukčnosti a chrání tak obvod (zejména vnitřní spínací tranzistory) proti zničení vyšším napětím nebo záporným napětím. Toto ochranné opatření je převzato z vývojového kitu [20] vyvinutého společností Linear Technology. Vstupní napětí označené VIN, které je společné i pro napájecí zdroj digitální části, je chráněno proti přepětí varistorem s označením R38 a dále je toto napětí filtrováno pomocí kondenzátoru C34 s kapacitou 47 µF, tento kondenzátor je také společný pro oba napájecí zdroje. Varistor je vyráběn v SMD pouzdře velikosti 1206 společností Würth Elektronik a je určený pro 5V aplikace. Do této kategorie spadá i toto zařízení, protože maximální vstupní napětí obvodu LTC3538 je 5,5 V.

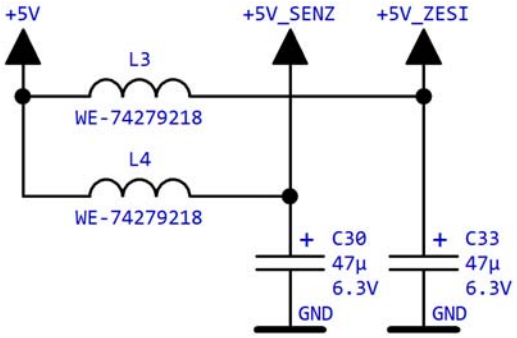
Kondenzátor C27 s rezistorem R33 nastavují zpětnou vazbu chybového zesilovače, spolu s kondenzátorem C28 a rezistory R35, R36 a R34 určují přenos chybového zesilovače. Rezistory R36 a R34 zároveň nastavují hodnotu výstupního napětí podle vztahu (E.1):

$$V_{OUT} = 1 \cdot \left(1 + \frac{R36}{R34}\right) = \left(1 + \frac{806 \cdot 10^3}{200 \cdot 10^3}\right) = 5{,}03\ [V] \qquad (6.1.)$$

Kondenzátor C26 s kapacitou 10 µF tvoří vstupní kapacitu, kondenzátor C29 s kapacitou 22 µF představuje výstupní kapacitu. Spínaná cívka L2 musí být dimenzována na minimální proud 0,5 A (viz rovnice (E.3) a (E.4) při uvažování horších případů, aby proud cívkou vycházel větší) a s co nejmenším stejnosměrným odporem. Proto byla vybrána cívka SDE6603-3R3M s indukčností 3,3 µH, maximální proud 2 A a se stejnosměrným odporem 0,08 Ω.

Výrobce obvodu LTC3538 doporučuje pro snížení zvlnění výstupního napětí použít režim PWM. Z tohoto důvodu je vývod BURST přiveden na nulový bod vstupního zdroje pomocí rezistoru R32 s nulovou hodnotou odporu. Pro testovací účely je možné připojením nulového rezistoru R31 a odstraněním rezistoru R32 použít režim BURST. Protože se výstupní proud zdroje pohybuje v desítkách mA, dosahuje obvod vysoké účinnosti i v režimu PWM (viz obrázek E.3).

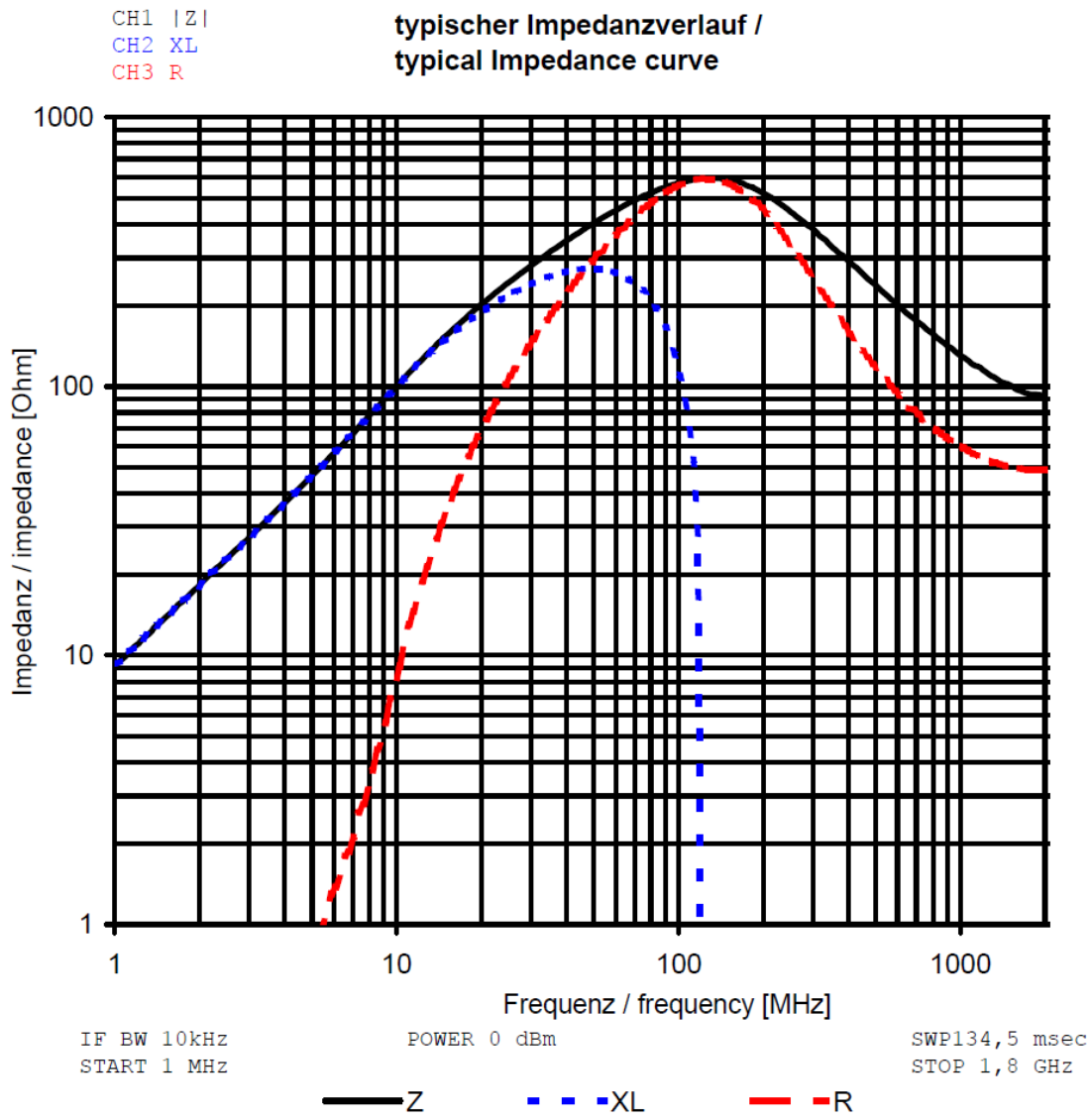
Testování prototypového zapojení detektorů ukázalo problémy s přítomnosti rušícího signálu v analogovém signálu na výstupu zesilovače. Bližším zkoumáním byla zjištěna nutnost oddělení zdroje od zátěže pomocí tzv. feritové perly, která představuje určitou indukčnost a klade vyšším frekvencím vyšší impedanci a tím zabraňuje pronikání vysokých frekvencí do dalších obvodů. Tyto frekvence mohou např. vznikat při větších rázových odběrech některého obvodu (např. mikroprocesoru nebo bezdrátového modulu nebo i samotného zesilovače a mikrovlnného senzoru). Aby se lépe těmto rušením zabránilo, jsou i jednotlivé obvody jednoho napájecího zdroje odděleny také těmito indukčnostmi, proto je oddělen mikrovlnný senzor od zesilovače a mikroprocesor s pamětí EEPROM od bezdrátového modulu (viz dále). Pro zajištění stejnosměrného napětí je nutné za indukčnost zařadit kondenzátor s větší kapacitou, který s indukčností tvoří odrušovací filtr. Toto opatření zabrání situacím, kdy by obvod potřebovat větší proudový odběr, této prudké změně se ovšem snaží indukčnost zabránit, ale kondenzátor má naopak snahu tuto změnu vykompenzovat dodáním energie. Indukčnost tedy zabraňuje šíření rušivého signálu do okolních obvodů, ale i z okolních obvodů, a kondenzátor dodává energii nutnou pro vykompenzování prudkých změn proudového zatížení způsobující kolísání napětí. Typické zapojení tohoto typu obvodu ukazuje následující obrázek:



Obrázek 6.3 – Zapojení odrušovacího filtru pro analogovou část detektoru

Základní napájecí napětí označené +5V z výstupu obvodu LTC3538 je rozděleno pomocí indukčností L3 a L4 na dvě napájecí větve s napětím +5V_SENZ pro mikrovlnný senzor a napětím +5_ZESI pro zesilovač. Obě tyto větve jsou opatřeny kondenzátory s kapacitou 47 µF, které mají mít co nejmenší ztrátový odpor, aby co nejlépe vyrovnávaly napěťové změny.

Feritové perly označené L3 a L4 byly vybrány od společnosti Würth Elektronik s typovým označením 74279218. Tyto perly výrobce přímo doporučuje pro účely zabránění rušení. Stejnosměrný odpor této součástky, vyráběné v SMD pouzdře o velikosti 1206, je 0,1 Ω, maximální proud 2 A, pro kmitočet 100 MHz dosahuje impedance hodnoty 600 Ω, pro kmitočet 700 MHz hodnoty 700 Ω. Závislost impedance těchto perel na frekvenci ukazuje následující obrázek:

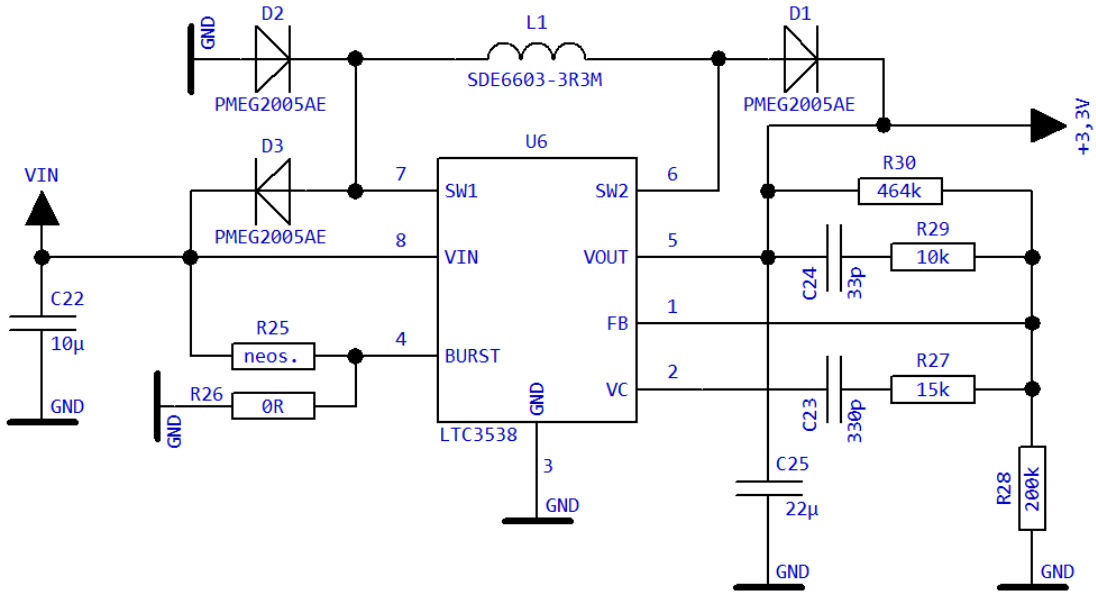


Obrázek 6.4 – Průběhy impedancí feritové perly 74279218 v závislosti na kmitočtu [26]

Obrázek 6.4 ukazuje průběhy všech typů impedancí feritové perly 74279218, tedy výsledné impedance Z, imaginární část impedance XL a reálnou část impedance R. Z průběhu celkové impedance Z vyplývá, že pro vyšší frekvence je impedance značně vyšší, tedy klade vyšší odpor a lépe zabraňuje šíření rušení do obvodů. Od kmitočtu kolem 100 MHz začíná impedance klesat, tím se mohou rušivé signály o vyšších frekvencích snáze dostávat přes indukčnost. Společnost Würth Elektronik nabízí i odrušovací feritové perly s impedancí, kterým se od žádné frekvence impedance nesnižuje. Nevýhodou těchto perel je ovšem nižší impedance na nízkých frekvencích.

6.2 Napájecí zdroj digitální části senzorové jednotky

Digitální část detektoru pohybu osob nebo detektoru vozidel je tvořena mikroprocesorem LM3S811, sériovou pamětí EEPROM 24LC04B a bezdrátovým modulem IQRF. Protože nominální hodnota napájecího napětí mikroprocesoru je 3,3 V a tato hodnota je také v rozmezí dovoleného napájecího napětí pro bezdrátový modul i paměť EEPROM, je celá digitální část detektoru napájena napětím 3,3 V. Návrh napájecího napětí vychází opět z doporučení výrobce obvodu LTC3538, tento návrh je téměř totožný s napájecím zdrojem pro analogovou část, pouze se liší v hodnotách rezistorů, kterými se určuje hodnota výstupního napětí.



Obrázek 6.5 – Schéma napájecího zdroje pro digitální část detektoru

Funkce jednotlivých součástek jsou shodné se součástkami použitými v napájecím zdroji analogové části. Hodnoty součástek jsou také totožné s hodnotami pro napájecí zdroj analogové části, pouze hodnota rezistoru R30 se liší. Tímto rezistorem je nastavena velikost výstupního napětí V_{OUT} podle vztahu (E.1):

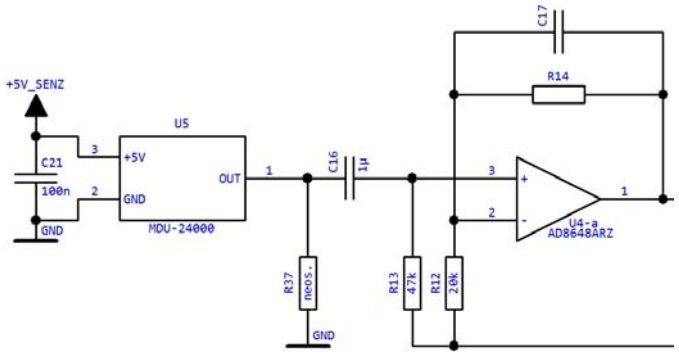
$$V_{OUT} = 1 \cdot \left(1 + \frac{R30}{R28}\right) = \left(1 + \frac{464 \cdot 10^3}{200 \cdot 10^3}\right) = 3{,}32\ [V] \qquad (6.2)$$

Stejně jako v případě napájecího zdroje analogové části jsou i zde jednotlivé obvody odděleny feritovými perlami (indukčnostmi) pro zabránění pronikání rušení prostřednictvím napájení z okolních obvodů, ale i pronikání rušení do okolních obvodů.

Výstupní napětí označené +3,3V z obvodu LTC3538 je rozděleno na dvě samostatné napájecí větve, které jsou následně opatřeny filtračními kondenzátory s kapacitou 47 µF. Napájecí větev +3,3V_LM3S slouží pro napájení mikroprocesoru a paměti EEPROM a větev +3,3V_IQRF napájí bezdrátový modul. Jako feritové perly jsou opět použity indukčnosti s označením výrobce 74279218.

6.3 Připojení mikrovlnného senzoru k zesilovači

Mikrovlnný senzor je napájen z větve označené jako +5V_SENZ se jmenovitou hodnotou 5 V. Toto napětí je blokováno blokovacím kondenzátorem o hodnotě 100 nF. Výstup senzoru je stejnosměrně oddělen kondenzátorem C16 se vstupem zesilovače. Rezistor R37, který není na desce plošného spoje osazen, je určen pro ladící účely k nastavení výstupní impedance senzoru resp. k nastavení vstupní impedance zesilovače.

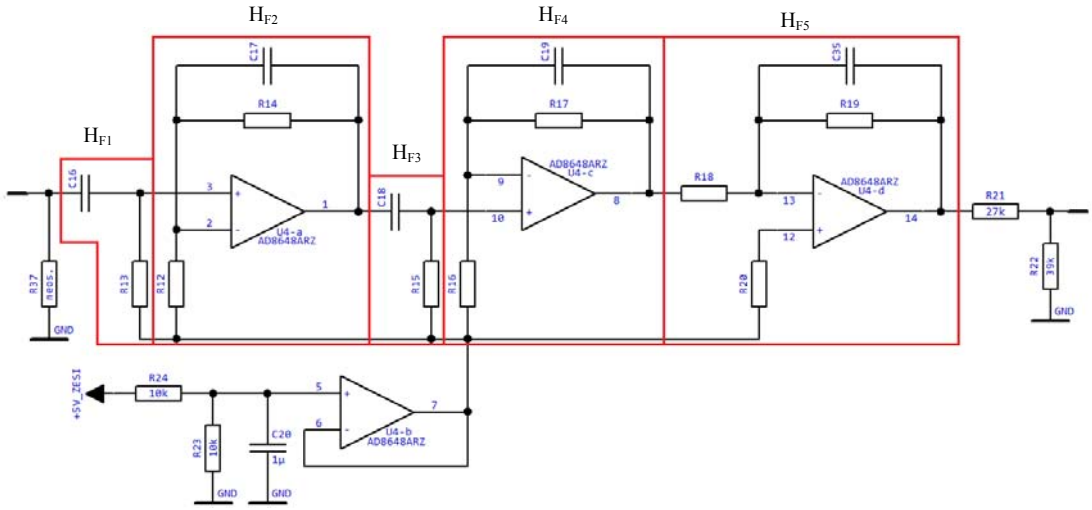


Obrázek 6.6 – Schéma připojení mikrovlnného senzoru k zesilovači

6.4 Návrh zesilovače signálu mikrovlnného senzoru

Zesilovač signálu mikrovlnného senzoru je nejvíce odlišná část pro detektor pohybu osob a detektoru vozidel. Zesilovač pro tyto dvě senzorové jednotky je nastaven na jiné mezní frekvence. Dolní mezní frekvence jsou nastaveny tak, aby senzorové jednotky zpracovávaly různé objekty, jednotka detekce vozidel má dolní mezní kmitočet roven hornímu meznímu kmitočtu jednotky detekce osob. Horní mezní kmitočet je odvozen od maximální rychlosti zpracovávaného objektu (osoba nebo vozidlo).

Zesilovač tedy plní funkci frekvenčního filtru, který se skládá z několika dílčích filtrů typu dolní a horní propust. Schéma filtru ukazuje obrázek 6.7:



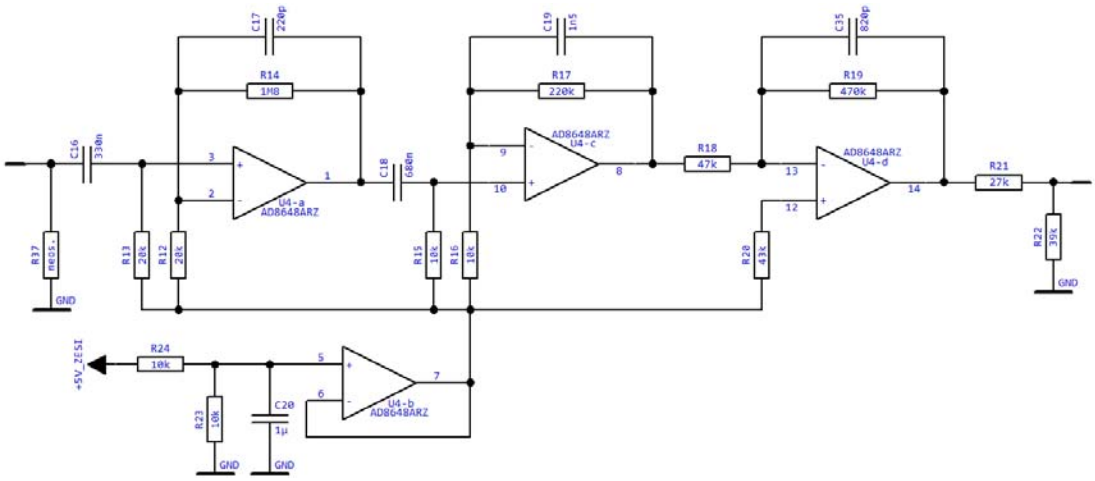
Obrázek 6.7 – Obecné schéma zesilovače

Vstup zesilovače je na rezistoru R37, výstup se nachází na rezistoru R33. Operační zesilovače jsou napájené z jediného zdroje, aby byl možný rozkmit výstupního signálu operačního zesilovače co největší, musejí být vstupy zesilovače umístěny na hodnotu rovnu polovičnímu napájecímu napětí. Toto napětí umožňuje stejný rozkmit pro kladné i záporné hodnoty. Pro tento účel je využit operační zesilovač označený U4-b zapojený jako neinvertující zesilovač s jednotkovým zesílením. Na neinvertující vstup je připojeno pomocí rezistorového děliče R23 a R24 poloviční napájecí napětí z napájecí větve +5V_ZESI určené pro napájení zesilovače. Napájecí napětí zesilovače je blokováno kondenzátorem s kapacitou 100 nF (na obrázku není zobrazen), tento kondenzátor má za úkol kompenzovat prudké změny napájecího napětí.

Jednotlivé dílčí filtry jsou označeny svými přenosy H_{F1} až H_{F5}. Filtry označené jako H_{F1} a H_{F3} jsou typu horní propust, filtry označené H_{F2}, H_{F4} jsou neinvertující filtry typu dolní propust a filtr H_{F5} je invertující filtr typu dolní propust. Všechny tyto typy filtrů byly popsány v kapitole 5.

6.4.1 Návrh hodnot součástek pro senzorovou jednotku detekce osob

Hodnoty rezistorů zesilovače určují výsledné zesílení zesilovače, které je poměrně vysoké. Velikost zesílení byla volena nahodile, aby byly splněny požadavky pro nastavení hodnoty prahu detekce a pro co největší dosah mikrovlnného senzoru. Hodnoty kondenzátorů určují mezní frekvence. Mezní frekvence jsou nastaveny na hodnoty $f_{D_1} = 25$ Hz a $f_{H_1} = 322$ Hz. Horní mezní kmitočet je stanoven maximální rychlostí chůze člověka uvažované 7 km·h^{-1}. Hodnota dolního mezního kmitočtu je volena z toho důvodu, že výsledný signál ze zesilovače je výrazně méně zašuměn. Frekvence 25 Hz odpovídá rychlosti přibližně 15 cm·s^{-1}, tato rychlost je dostatečně nízká pro zachycení pomalých chodců. Výsledné schéma zesilovače je ukázáno na následujícím obrázku:



Obrázek 6.8 – Schéma zesilovače pro detektor osob

Celkový zesilovač se skládá z několika dvojbranů, každý má své vlastní šumové číslo F_x (x označuje pořadí dvojbranu) sloužící k hodnocení šumových vlastností dvojbranů. Prostřednictvím šumového čísla lze určit míru šumu M:

$$M = \frac{F - 1}{1 - \frac{1}{A_{PA}}} \tag{6.3}$$

M je míra šumu [-],

F je šumové číslo [-],

A_{PA} je výkonové zesílení [-].

V [27] najdeme, že je nutné na prvním místě zařadit dvojbran s nejmenší mírou šumu. Všechny operační zesilovače jsou složeny ze stejných typů součástek, proto lze považovat šumové číslo pro všechny dílčí zesilovače za stejné. Ze vztahu (6.3) vyplývá, že nejmenší míru šumu bude mít zesilovač s největším zesílením při stejných šumových číslech ostatních zesilovačů. Proto je celkové zesílení zesilovače rozloženo na jednotlivé dílčí zesilovače a je postupně stupňováno s klesající tendencí, první zesilovač má největší zesílení, další má menší a poslední má nejmenší. Tímto způsobem se eliminuje zesílení vlastního šumu jednotlivých zesilovačů.

Filtry typu horní propust, určující dolní mezní kmitočet, jsou ve výsledném filtru celkem dva, oba filtry plní zároveň funkci oddělení stejnosměrných složek, hlavně pro první stupeň oddělující mikrovlnný senzor a zesilovač. Pro uvažovaný pokles o 6 dB (pokles napětí na polovinu) na mezním kmitočtu, pro dva filtry spadá na každý filtr pokles o 3 dB, proto má konstanta k, pomocí níž byl odvozen vztah pro mezní kmitočet (viz rovnice (5.10)), hodnotu určenou vztahem (6.5), vztah (6.4) ukazuje přepočet hodnoty napětí na dB:

$$20 \cdot \log V = D \ [dB] \tag{6.4}$$

V je poměr napětí [-],

D je přepočtená hodnota na [dB].

$$k_D = 10^{\frac{3}{20}} = 1{,}4125 \tag{6.5.}$$

Hodnota rezistorů R13 a R15, určující spolu s kondenzátory C16 a C18 dolní mezní kmitočet, je vypočtena z rovnice (5.30) pro proudovou kompenzaci vstupů operačního zesilovače. Tímto vztahem je určena i hodnota rezistoru R20:

$$R13 = \frac{R14 \cdot R12}{R14 + R12} = \frac{1{,}8 \cdot 10^6 \cdot 20 \cdot 10^3}{1{,}8 \cdot 10^6 + 20 \cdot 10^3} = 19{,}78\ [k\Omega] \tag{6.6}$$

$$R15 = \frac{R17 \cdot R16}{R17 + R16} = \frac{220 \cdot 10^3 \cdot 10 \cdot 10^3}{220 \cdot 10^3 + 10 \cdot 10^3} = 9{,}57\ [k\Omega] \tag{6.7}$$

$$R20 = \frac{R19 \cdot R18}{R19 + R18} = \frac{470 \cdot 10^3 \cdot 47 \cdot 10^3}{470 \cdot 10^3 + 47 \cdot 10^3} = 42{,}73\ [k\Omega] \tag{6.8}$$

Hodnoty rezistorů nejsou vyráběny, je nutné vybrat nejbližší vyráběné hodnoty: 20kΩ, 10kΩ a 43kΩ.

Rezistory R21 a R22 přizpůsobují výstupní napěťovou úroveň zesilovače pro 3V referenční napětí A/D převodníku mikrokontroléru LM3S811.

Filtry typu dolní propust jsou ve filtru přítomny tři, proto na každý filtr připadá pokles 2 dB pro dosažení poklesu 6 dB pro horní mezní kmitočet, proto hodnota k_H nabývá pro tento typ filtru následující hodnoty:

$$k_H = 10^{\frac{2}{20}} = 1{,}2589 \tag{6.8}$$

Hodnoty rezistorů jsou známé, hodnoty kondenzátorů určíme podle patřičných vztahů podle typu filtru, C16 podle vztahu (5.11), C17 podle vztahu (5.25), C18 podle vztahu (5.11), C19 podle vztahu (5.25) a C35 podle vztahu (5.39). Vypočtené hodnoty kondenzátorů nejsou vyráběny, proto je nutné vybrat co nejbližší hodnoty z vyráběných hodnot kondenzátorů:

$$C17 = \sqrt{\frac{1}{(k_D^2 - 1) \cdot \left(2\pi \cdot f_{D_1} \cdot R13\right)^2}} = 319{,}08 \sim 330\ [nF] \tag{6.9}$$

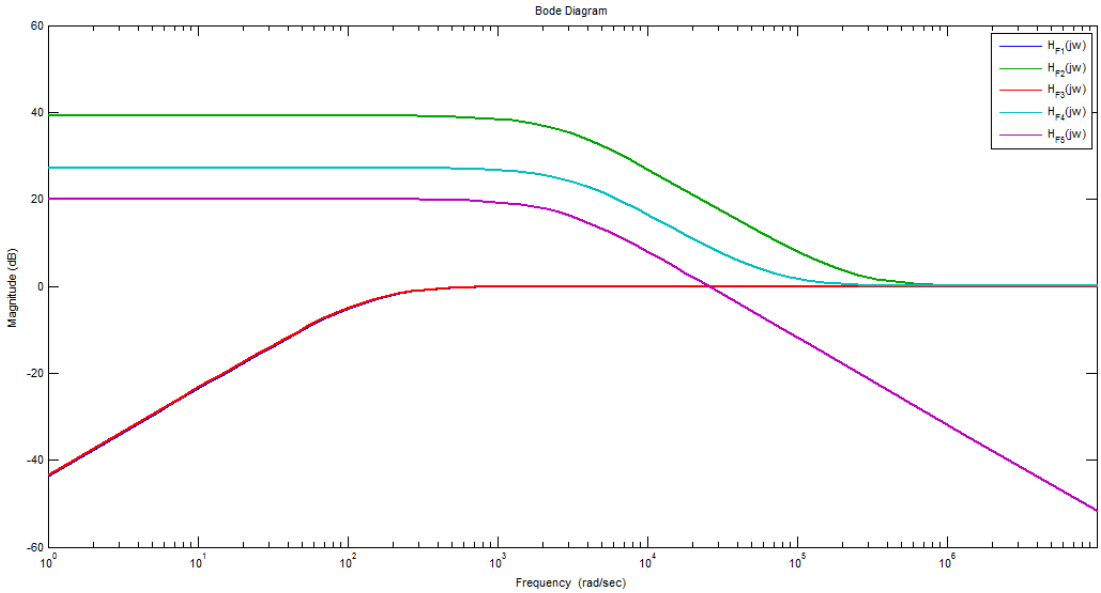
$$C17 = \sqrt{\frac{(R14 + R12)^2 \cdot R12^2 \cdot (k_H^2 - 1)}{\left(2\pi \cdot f_{H_1}\right)^2 \cdot (R14 + R12)^2 \cdot (R14 \cdot R12)^2 - \left(2\pi \cdot f_{H_1}\right)^2 \cdot k_H^2 \cdot R12^2 \cdot (R14 \cdot R12)^2}} = \\ = 210{,}0138 \sim 220\ [pF] \tag{6.10}$$

$$C18 = \sqrt{\frac{1}{(k_D^2 - 1) \cdot \left(2\pi \cdot f_{D_1} \cdot R15\right)^2}} = 638{,}1672 \sim 680\ [nF] \tag{6.11}$$

$$C19 = \sqrt{\frac{(R17 + R16)^2 \cdot R16^2 \cdot (k_D^2 - 1)}{\left(2\pi \cdot f_{H_1}\right)^2 \cdot (R17 + R16)^2 \cdot (R17 \cdot R16)^2 - \left(2\pi \cdot f_{H_1}\right)^2 \cdot k_D^2 \cdot R16^2 \cdot (R17 \cdot R16)^2}} = \\ = 1{,}7207 \sim 1{,}5\ [nF] \tag{6.12}$$

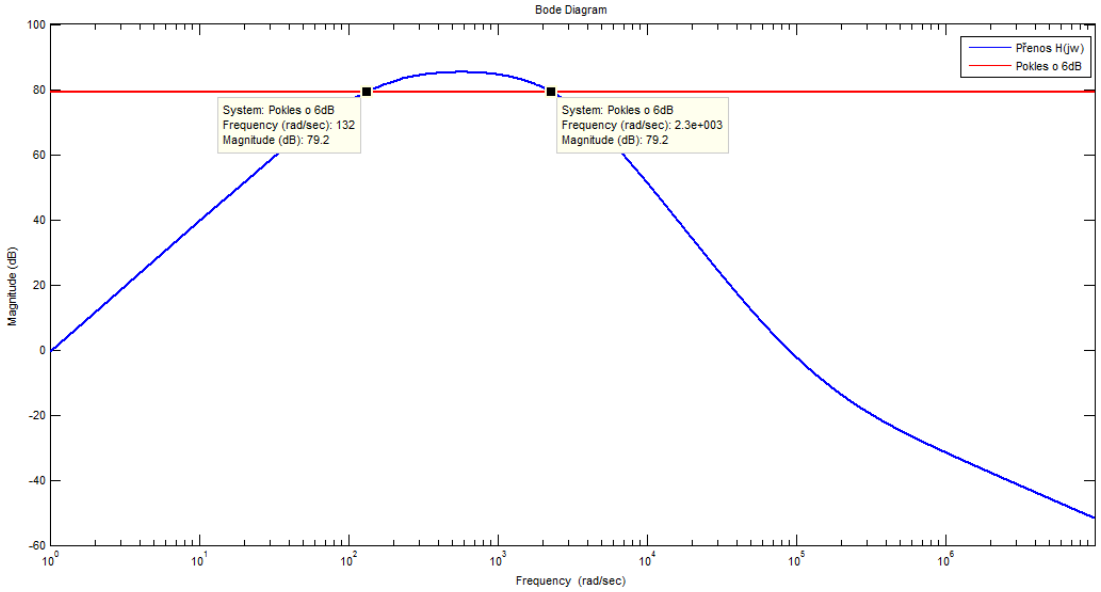
$$C35 = \sqrt{\frac{(k_D^2 - 1)}{\left(R19 \cdot 2\pi \cdot f_{H_1}\right)^2}} = 804{,}23 \sim 820\ [pF] \tag{6.13}$$

Budeme-li uvažovat přenos jako polynom proměnné $j\omega$, lze použít funkci *tf* (Transfer Function) programu MATLAB pro vytvoření přenosu Laplaceovy transformace (pro frekvenční přenos platí substituce $s = j\omega$) a pro zobrazení amplitudových frekvenčních přenosů dílčích filtrů použijeme funkci *bodemag*:



Obrázek 6.9 – Průběhy amplitudových přenosů dílčích filtrů zesilovače pro detektor osob

Celkový amplitudový přenos zesilovače $H(j\omega)$, resp. filtru, získáme jako součet amplitudových přenosů dílčích filtrů přepočtených do dB:



Obrázek 6.10 – Amplitudový frekvenční přenos celkové filtru detektoru osob

Obrázek 6.10 ukazuje amplitudový frekvenční přenos celého filtru, zároveň je zobrazen pokles o 6 dB oproti maximální hodnotě přenosu. Průsečíky těchto dvou křivek představují mezní kmitočty shrnuté v tabulce 6.1:

Tabulka 6.1 – Přehled očekávaných a vypočtených mezních kmitočtů senzoru osob

	Dolní mezní kmitočet	Horní mezní kmitočet
Teoretické kmitočty	25 [Hz]	322,6667 [Hz]
Výsledné kmitočty	132 [rad·s^{-1}]	2,3·10^{3} [rad·s^{-1}]
	21,009 [Hz]	366,056 [Hz]
Relativní frekvenční odchylka	-15,97 %	13,45 %
Rychlost objektu	0,1308 [m·s^{-1}]	2,2689 [m·s^{-1}]

Odečtené hodnoty mezních frekvencí se od požadovaných (teoretických) odlišují o několik procent. Chyba je způsobena zaokrouhlováním hodnot kondenzátorů, ale také mohou být způsobeny chybným odečtením nebo omezeným frekvenčním krokem při vykreslování amplitudového přenosu. Mezní kmitočet vychází nižší než požadovaný, tato chyba nemá negativní vliv, protože budou zachyceny i pomaleji se pohybující objekty. Zato horní mezní kmitočet vychází vyšší, chyba již může mít negativní důsledky v podobě detekce rychlejšího objektu, než který chceme detekovat. Chyba může být také způsobena úzkým frekvenčním pásmem, kde průběh nenabude očekávané maximální hodnoty a při poklesu 6 dB od nižší hodnoty způsobí rozšíření frekvenčního pásma. Tato alternativa je nejvíce pravděpodobná, protože stejnosměrné zesílení má být rovno hodnotě 20930, ale z průběhu přenosu byla odečtena hodnota 85,27 dB, která odpovídá zesílení 18339.

6.4.2 Návrh hodnot součástek pro senzorovou jednotku detekce vozidel

Zesilovač pro detektor projíždějících vozidel je navržen zcela identicky jako pro detektor osob. Liší se pouze v hodnotách kondenzátorů určujících horní mezní kmitočet. Dolní mezní kmitočet je nastaven na horní mezní kmitočet pro detektor osob, jednotka detekce vozidel nebude schopna detekovat chodce, $f_{D_2} = 322$ Hz, hodnota horního mezního kmitočtu určuje maximální rychlost zachyceného objektu, bylo odvozeno, že tato frekvence nabývá hodnoty $f_{H_2} = 7000$ Hz. Hodnoty součástek získáme pomocí stejných vztahů jako pro předešle popsaný typ senzorové jednotky.

$$C17 = \sqrt{\frac{1}{(k_D^2 - 1) \cdot \left(2\pi \cdot f_{D_1} \cdot R13\right)^2}} = 24{,}7736 \sim 27\ [nF] \tag{6.14}$$

$$C17 = \sqrt{\frac{(R14 + R12)^2 \cdot R12^2 \cdot (k_H^2 - 1)}{\left(2\pi \cdot f_{H_1}\right)^2 \cdot (R14 + R12)^2 \cdot (R14 \cdot R12)^2 - \left(2\pi \cdot f_{H_1}\right)^2 \cdot k_H^2 \cdot R12^2 \cdot (R14 \cdot R12)^2}} = \tag{6.15}$$

$$= 9{,}6606 \sim 10\ [pF]$$

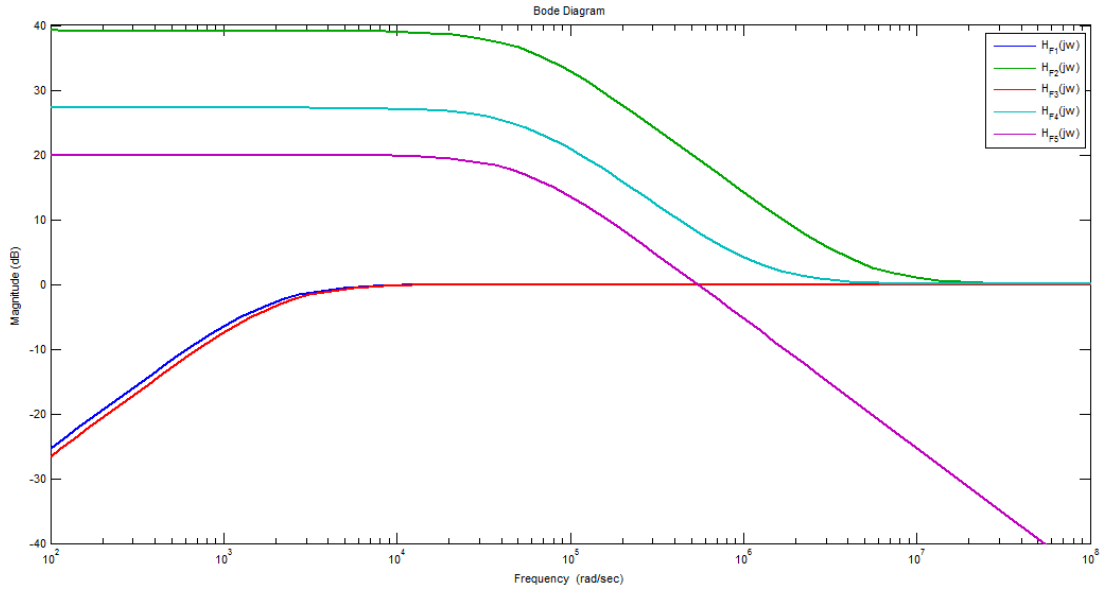
$$C18 = \sqrt{\frac{1}{(k_D^2 - 1) \cdot \left(2\pi \cdot f_{D_1} \cdot R15\right)^2}} = 49{,}5471 \sim 47\ [nF] \tag{6.16}$$

$$C19 = \sqrt{\frac{(R17 + R16)^2 \cdot R16^2 \cdot (k_D^2 - 1)}{\left(2\pi \cdot f_{H_1}\right)^2 \cdot (R17 + R16)^2 \cdot (R17 \cdot R16)^2 - \left(2\pi \cdot f_{H_1}\right)^2 \cdot k_D^2 \cdot R16^2 \cdot (R17 \cdot R16)^2}} = \tag{6.17}$$

$$= 79{,}1527 \sim 82\ [pF]$$

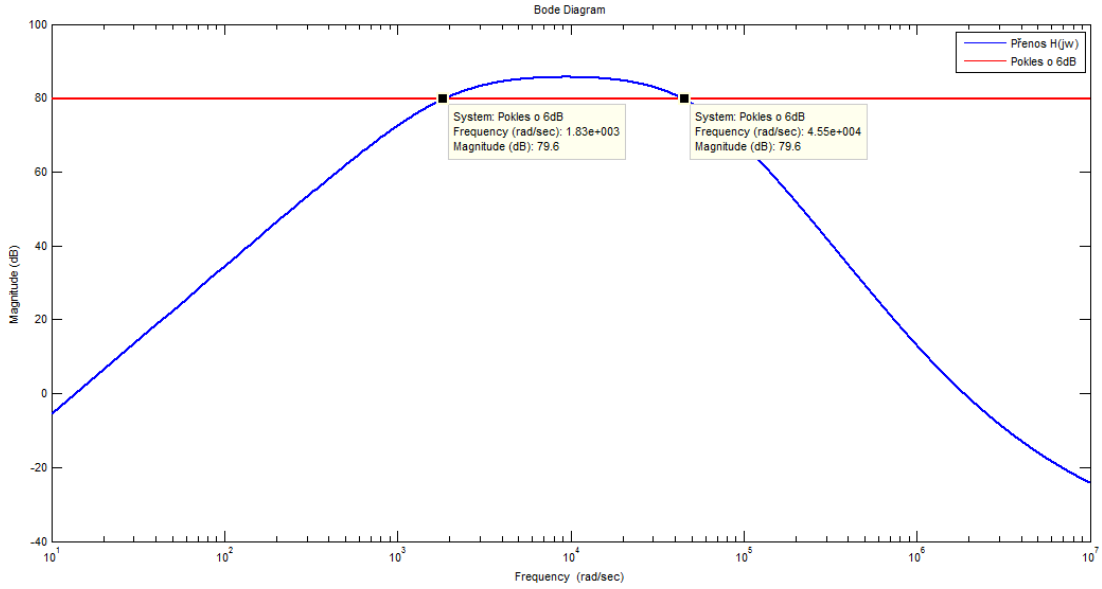
$$C35 = \sqrt{\frac{(k_D^2 - 1)}{\left(R19 \cdot 2\pi \cdot f_{H_1}\right)^2}} = 36{,}9946 \sim 39\ [pF] \tag{6.18}$$

Jsou-li známé všechny hodnoty součástek, můžeme opět vynést dílčí amplitudově frekvenční přenosy filtrů:



Obrázek 6.11 – Průběhy amplitudových přenosů dílčích filtrů zesilovače pro detektor vozidel

Celkový amplitudový přenos zesilovače pro detektor vozidel ukazuje následující obrázek 6.12:



Obrázek 6.12 – Amplitudový frekvenční přenos celkové filtru detektoru vozidel

Srovnání teoretických a odečtených hodnot mezních kmitočtů shrnuje následující tabulka 6.2.

Tabulka 6.2 – Přehled očekávaných a vypočtených mezních kmitočtů senzoru vozidel

	Dolní mezní kmitočet	Horní mezní kmitočet
Teoretické kmitočty	322 [Hz]	7000 [Hz]
Výsledné kmitočty	$1{,}83 \cdot 10^3$ [rad·s^{-1}]	$4{,}55 \cdot 10^4$ [rad·s^{-1}]
	291,25 [Hz]	7241,5 [Hz]
Relativní frekvenční odchylka	-9,6%	3,45%
Rychlost objektu	1,8053 [m·s^{-1}]	44,8853 [m·s^{-1}]

Pro detektor vozidel je chyba mezi teoretickými a odečtenými hodnotami mezních frekvencí menší, na rozdíl od detektoru osob. Horní mezní frekvence vyšla o něco vyšší, což má za následek detekování rychlejších vozidel, tato chyba nemá negativní charakter. Naopak dolní mezní kmitočet vychází poněkud menší, tato chyba může mít za následek detekci rychle se pohybujících se chodců, ale také budou lépe detekována pomalu se pohybující vozidla.

6.5 Zapojení mikrokontroléru LM3S811

Zapojení mikrokontroléru je identické pro oba typy senzorových jednotek. Mikrokontrolér je zapojen podle doporučení popsaných v příloze H.2. Hodinovou frekvenci tvoří krystalový oscilátor s hodnotou 6 MHz. Výstup zesilovače je připojen na analogový vstup ADC0 mikrokontroléru. Programovací a ladící rozhraní JTAG využívající vývody TCK, TDO, TDI a TMS je přivedeno na samostatný konektor pro programování, vývody jsou zároveň chráněny proti přepětí pomocí ochrany TVS diodami od společnosti Würth Elektronik s typovým označením 82402304 určené pro ochranu 5V zařízení (mikrokontrolér umožňuje zpracovávat 5V napěťové úrovně) s kapacitou kanálu 13 pF (obvod obsahuje celkem čtyři ochranné diody – kanály). Vývody rozhraní JTAG jsou blokovány přes rezistory s hodnotou 100 kΩ k napájecímu napětí. Stejná přepěťová ochrana je použita pro ochranu vývodů (RX0, TX0, RX1 a TX1 určených pro ladící účely). Ostatní ladící vývody IO0 a IO1 jsou chráněny pomocí dvoukanálové ochrany vyráběné firmou Würth Elektronik s označením 824022 s kapacitou kanálu 12 pF.

6.6 Zapojení paměti EEPROM a bezdrátového modulu IQRF

Paměť EEPROM je přímo připojena k I^2C rozhraní mikrokotroléru LM3S811, oba datové vodiče jsou připojeny přes pull-up rezistory R5 a R4 k napájecímu napětí. Paměť má povolený zápis uzemněním vývodu WP.

Bezdrátový modul je připojen k SSI periferii mikrokontroléru. Napájení pro bezdrátový modul je kvůli programování přerušeno pomocí propojovacího konektoru (součást programovacího konektoru), za běžného provozu je tento konektor propojen, při programování se rozpojí a přivede se napájecí napětí pro IQRF modul z programátoru. Během programování musí být držen mikrokontrolér v resetovacím stavu.

6.7 Návrh desky plošného spoje

Senzorová jednotka je realizována na dvouvrstvé desce plošného spoje. Většina spojů a součástek je umístěna na jedné straně plošného spoje, druhou stranu z velké části zabírá zemní plocha, je zde umístěn i mikrovlnný senzor. Umístění

mikrovlnného senzoru bylo voleno z důvodu prostorového i mechanického, ale zejména z funkčního hlediska, zemnící plocha zabraňuje pronikání rušení z obvodů umístěných na druhé straně desky, než je umístěn mikrovlnný senzor.

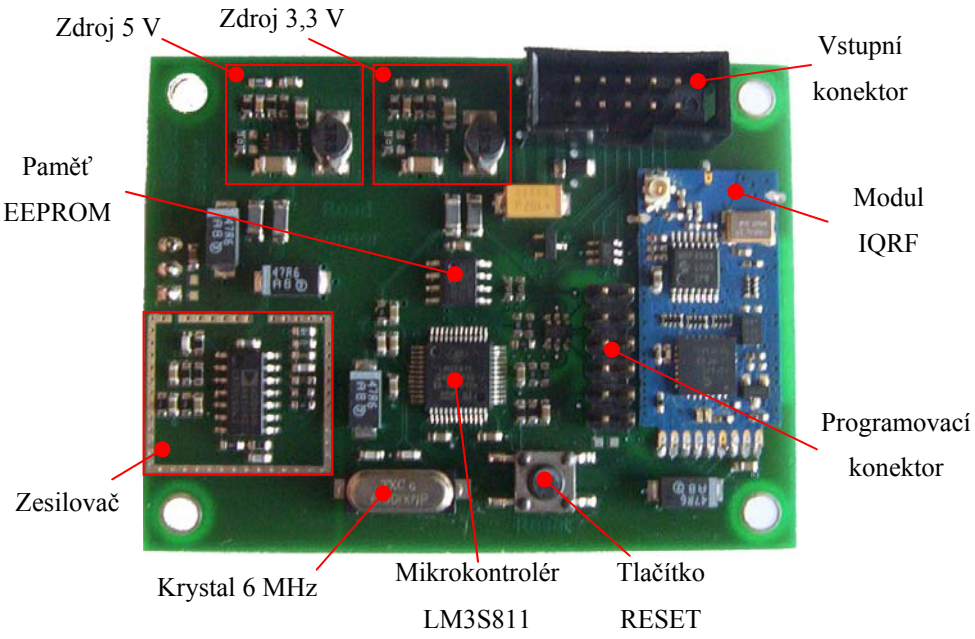
Při návrhu bylo nutné dodržet několik pravidel. Spoje musely být co možná nejkratší, hlavně v případě napájecích zdrojů a analogového zesilovače. U napájecích zdrojů je nejvíce citlivým místem zpětná vazba chybového zesilovače, ta by neměla vést kolem žádné části, kde dochází k rychlým změnám, v opačném případě by mohlo docházet k naindukování rušivého signálu do zpětné vazby spínaného zdroje a tím k nežádoucímu zvlnění výstupního napětí zdroje. Spoje od indukčnosti spínaného zdroje je vhodné vést co možná nejširší, cívka má malý stejnosměrný odpor a slabé přívodní vodiče by mohly tento odpor zvyšovat. Kolem každého napájecího zdroje je rozlita zemnící plocha sloužící jako chlazení pouzdra integrovaného obvodu LTC3835.

Spoje určené pro napájení jsou vedeny širšími spoji, aby při rychlých proudových odběrech nevznikaly na vodičích úbytky napětí. Vstupní napětí přivedené k senzorové jednotce je vedeno nejsilnějšími spoji k varistoru R38, z něhož se prokovenými otvory přivede minusový pól zdroje na zemnící plochu na druhé straně plošného spoje. Tyto prokovené díry tvoří bod, ze kterého se rozdělí země pro jednotlivé části senzorové jednotky, tedy pro analogovou a digitální část. Kladný pól vstupního napájecího zdroje je také přímo přiveden na varistor, dále na filtrační kondenzátor C34, z něhož je opět rozdělen na dvě části pro jednotlivé napájecí zdroje. Toto opatření má vliv na zlepšení proudových a tedy i napěťových poměrů napájecích zdrojů. Každý napájecí zdroj bude odebírat proud ze společného bodu a nebudou se navzájem ovlivňovat, případné napěťové změny bude kompenzovat kondenzátor C34 s vyšší kapacitou.

Přepěťové ochrany rozhraní JTAG a vývodů mikrokontroléru LM3S811 vyvedené na výstupní konektor J1 jsou umístěny mezi daný konektor a mikroktrolér. Spoje vedené z mikrokontroléru jsou co možná nejkratší, výjimku tvoří spoje mezi mikrokotrolérem a bezdrátovým modulem IQRF, které nebylo možné vést jinou cestou.

Všechny napájecí vývody obvodů jsou blokovány keramickými kondenzátory s kapacitou 100 nF umístěnými co nejblíže napájecím vývodům. Kondenzátory umístěné za feritovými perlami jsou umístěny v blízkosti daného obvodu.

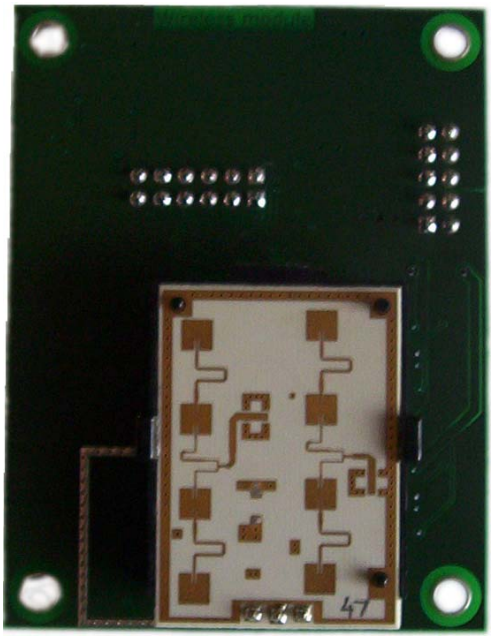
Následující obrázek ukazuje osazenou desku plošného spoje senzorové jednotky stranu součástek s vyznačenými nejdůležitějšími obvody:



Obrázek 6.13 – Strana součástek s popisy důležitých obvodů

Kolem analogového zesilovače je umístěn zemnící pásek určený pro přiletování krycího stínícího plechu pro zabránění vnikání rušení z okolí do zesilovače. U výše osazené desky nebylo počítáno s ochrannými diodami obvodů LTC3538, proto na desce nejsou přítomny. Návrh desky plošného spoje obsažený v příloze C již tyto diody obsahuje.

Obrázek 6.14 ukazuje stranu spojů senzorové jednotky:



Obrázek 6.14 – Strana spojů senzorové jednotky

Proudová spotřeba obou senzorových jednotek je i přes všechny snahy o její snížení poměrně značná. V případě senzorové jednotky detekce osob se spotřeba pohybuje kolem 95 mA, u detekce vozidel je spotřeba kolem 105 mA. Tyto vysoké proudy jsou zejména způsobeny velkým odběrem mikrovlnného senzoru a také tím, že napětí pro mikrovlnný senzor je vytvořeno zvyšujícím měničem, tedy při uvažovaném měniči se 100 % účinností bude vstupní proud vyšší než výstupní, vstupní a výstupní výkon bude stejný. Použitý měnič ovšem není dokonalý a tak je vstupní proud ještě o něco vyšší.

7 Programová realizace senzorových jednotek

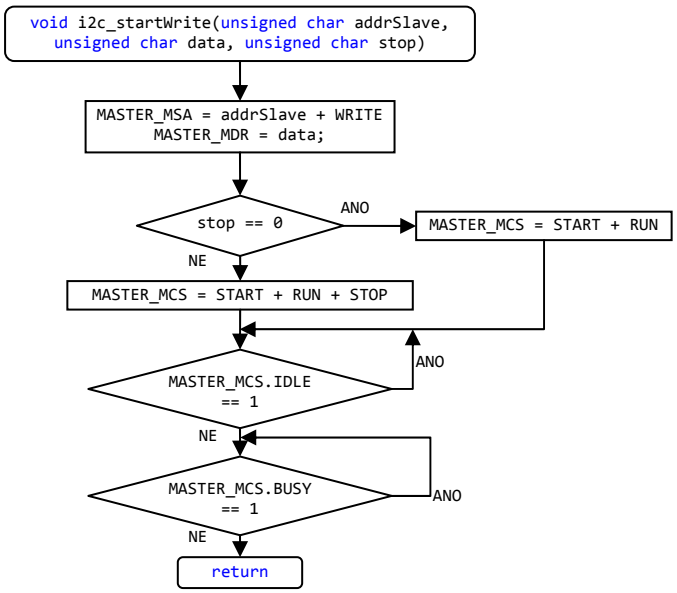
Senzorové jednotky vykonávají svoji činnost na základě programu uloženého v mikrokontroléru LM3S811, v bezdrátovém modulu IQRF je přítomen také program vykonávající příkazy zaslané mikrokontrolérem a na jejich základě provede modul danou činnost – buď odešle požadovaná data, provede nastavení bezdrátové komunikace, nebo přejde do režimu spánku za účelem snížení spotřeby. Výpis zdrojových kódů je uveden v příloze D.

Některé programové činnosti jsou shodné pro oba typy senzorových jednotek, např. program IQRF modulů, komunikace po sběrnici I^2C, komunikace s IQRF modulem. Největší rozdíl nastává ve zpracování analogového signálu z mikrovlnného senzoru zesíleného zesilovačem. Program pro mikrokontrolér byl vytvořen ve vývojovém prostředí µVision společnosti KeilARM v jazyce C, program pro bezdrátový modul IQRF je napsán také v jazyce C. V programu mikrokontroléru je použit pro přístup k periferiím přímý zápis do registrů, zejména kvůli rychlosti zpracování, výjimku tvoří povolení přerušení pomocí funkcí *IntMasterEnable* a *IntEnable* využívající knihovnu StellarisWare. Podrobný popis používaných periferií mikrokontroléru můžeme najít v příloze H.

7.1 Komunikace po sběrnici I^2C

Komunikace se sběrnicí I^2C mikrokontroléru LM3S811 je realizována odděleně v samostatných zdrojových souborech *i2c.c* a *i2c.h*, to umožňuje snadné přenášení zdrojového kódu mezi projekty obou senzorových jednotek. Pro jednotlivé senzorové jednotky je nutné nastavit různou hodinovou frekvenci I^2C sběrnice z důvodu odlišných hodinových frekvencí. Popis použité paměti EEPROM a sběrnice I^2C nalezneme v příloze F.

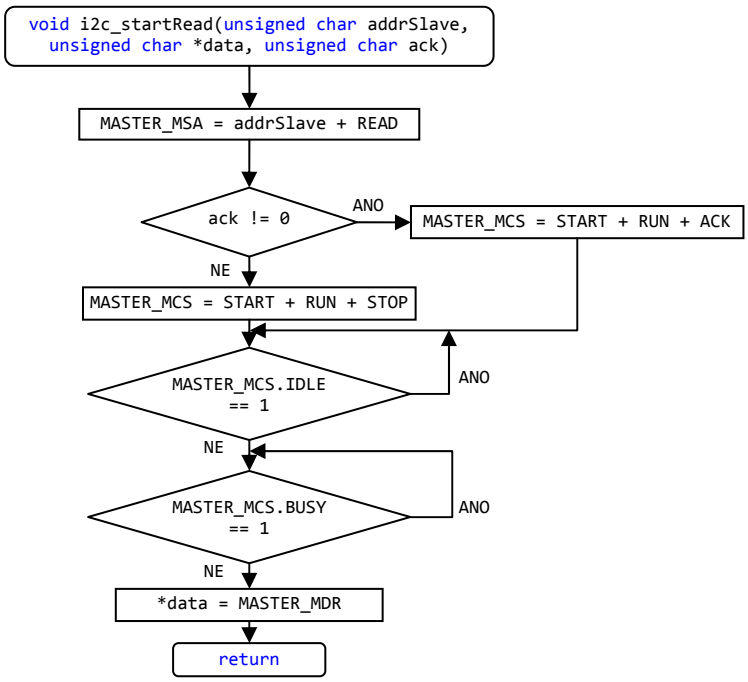
Sběrnice I^2C je inicializována funkcí *i2c_init.* Provede se povolení periferie I^2C v režimu MASTER a nastaví se velikost hodinové frekvence přenášení dat na sběrnici. Pro zahájení přenosu dat slouží dvě funkce podle typu přenosu dat. První funkcí je *i2c_startWrite* pro zaházejí operace zápisu do obvodu SLAVE, zde do paměti EEPROM 24LC04B, viz následující vývojový diagram:



Obrázek 7.1 – Vývojový diagram funkce „i2c_startWrite“

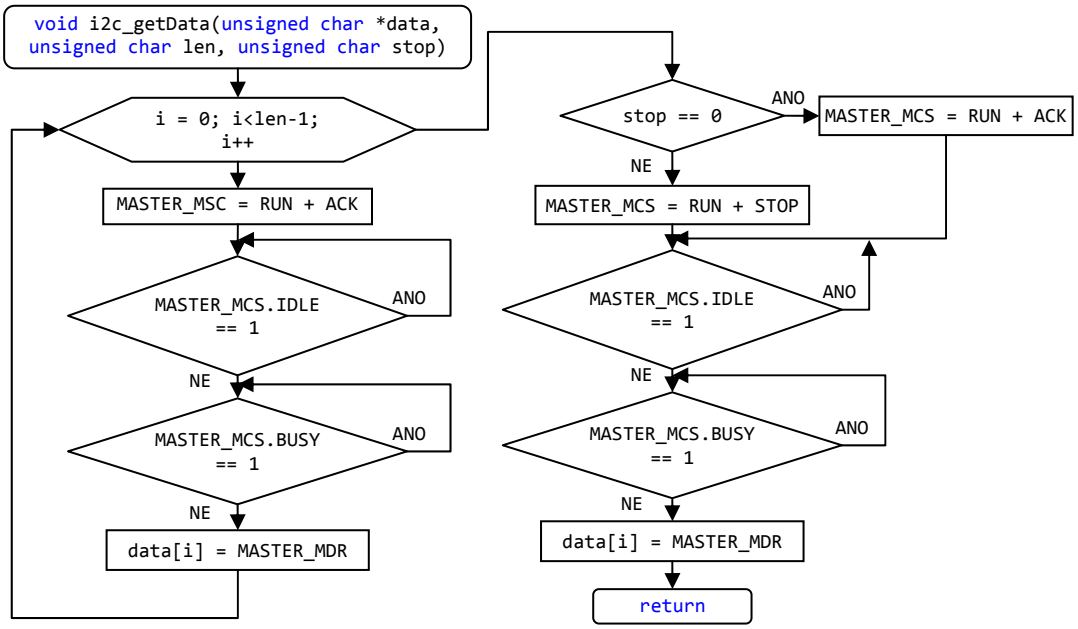
Prvním parametrem funkce je adresa obvodu SLAVE, druhý parametr obsahuje data vyslaná za adresou, v případě paměti EEPROM je to adresa, na kterou se má provést zápis, poslední parametr funkce určuje, jestli má být za daty vyslána událost STOP. Funkce vyplní registr pro adresu SLAVE, zapíše data a na základě posledního parametru zahájí komunikaci. Následně počká, až bude mikrokontrolér komunikovat a poté na dokončení komunikace. Toto je nutné udělat po každé operaci.

Druhá funkce *i2c_startRead* zahajuje komunikaci pro čtení z obvodu SLAVE. První parametr funkce je opět adresa obvodu SLAVE, druhým parametrem je adresa v paměti mikrokontroléru, na kterou se uloží přijatá data, poslední parametr určuje, jestli přijatá data budou potvrzena nebo bude vyslána událost STOP. Funkce opět nastaví adresu zařízení SLAVE, na základě posledního parametru provede danou operaci a počká na přijetí dat, která uloží na adresu obsaženou ve druhém parametru funkce. Funkci lze popsat následujícím vývojovým diagramem (viz obrázek 7.2):



Obrázek 7.2 – Vývojový diagram funkce „i2c_startRead“

Pro příjem dat slouží funkce *i2c_getData*. První parametr určuje adresu zásobníku, kam se mají data uložit, druhý parametr obsahuje počet přijatých dat a poslední určuje, jestli bude komunikace ukončena událostí STOP. Před voláním funkce je nutné obvod SLAVE inicializovat funkcí *i2c_startRead*. Funkce na začátku přijme všechna data kromě posledního bajtu, data budou potvrzována. Následně se podle posledního parametru buď poslední data potvrdí, nebo se odešle událost STOP, po ukončení komunikace se poslední bajt uloží do zásobníku, viz následující obrázek:

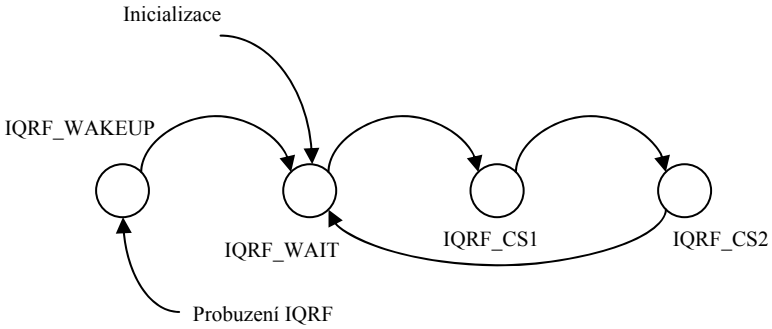


Obrázek 7.3 – Vývojový diagram funkce „i2c_getData“

7.2 Komunikace s modulem IQRF

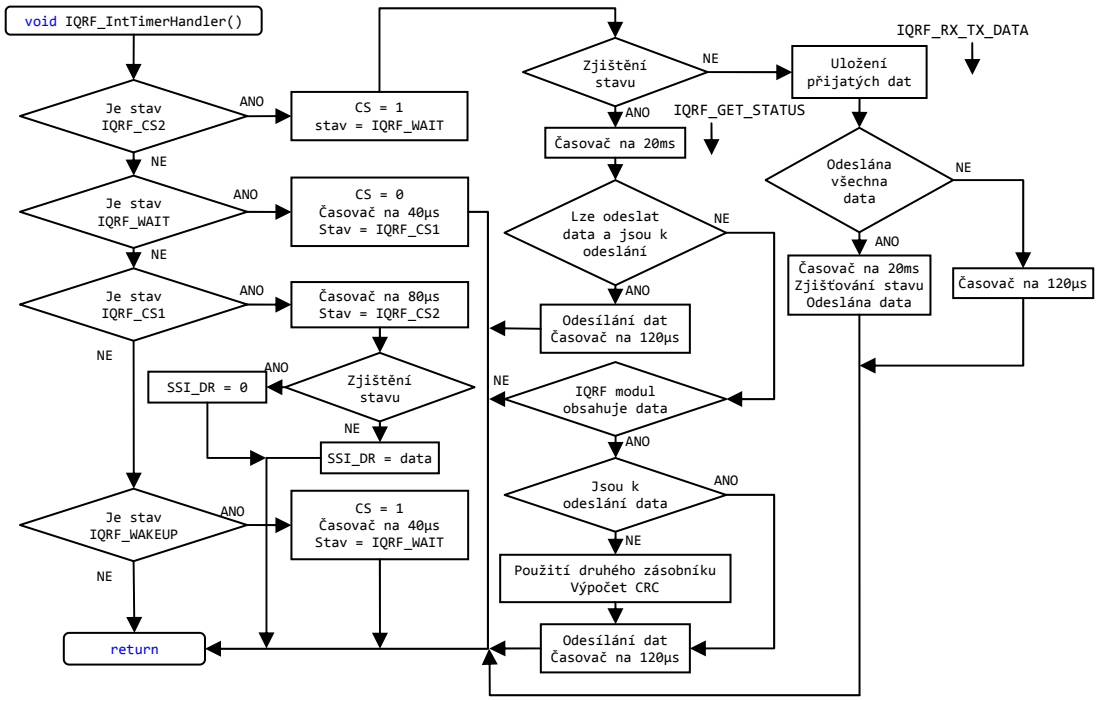
Komunikace s bezdrátovým modulem IQRF je realizována pomocí SPI sběrnice zastoupená periferií SSI mikrokontroléru LM3S811. Popis SPI komunikace a časování komunikace lze najít v příloze G. Celá komunikace je realizována na pozadí hlavního programu mikrokontroléru pomocí přerušení časovače TIMER2A *IQRF_IntTimerHandler* z toho důvodu, aby nebyl spotřebováván hlavní strojový čas mikrokotroléru. Časovač pracuje vždy v režimu jednoho spuštění, musí být vždy znovu spuštěn. Zdrojové kódy komunikace jsou stejně jako pro I^2C sběrnici odděleny ve zdrojových souborech *iqrf.c* a *iqrf.h*. Pro jednotlivé typy senzorových jednotek je nutné zdrojové kódy pro komunikaci s IQRF modulem trochu upravit z důvodu jiných hodnot hodinových frekvencí mikrokontrolérů.

Časování komunikace je realizováno v podobě stavového automatu provádějícího operace podle diagramu uvedeného na obrázku 7.4:



Obrázek 7.4 – Stavový diagram časování komunikace s IQRF

Stavový automat má celkem čtyři stavy: IQRF_WAKEUP, IQRF_WAIT, IQRF_CS1 a IQRF_CS2. Výchozí stav je IQRF_WAIT, v tomto stavu je časovač nastaven na 20 ms v případě čekání na nový datový přenos, nebo na 120 μs v době čekání mezi odesláním dalšího bajtu dat. Po uplynutí doby stavu IQRF_WAIT přejde časovač do stavu IQRF_CS1, nastaví časovač na 40 μs a vynuluje vývod SS pro zahájení komunikace s IQRF modulem. Ve stavu IQRF_CS1 se časovač nastaví na dobu 80 μs, odešle určený bajt dat a přejde do stavu IQRF_CS2. Tato doba zahrnuje dobu nutnou pro odeslání dat a také dobu po odeslání dat do nastavení vývodu SS. Periferie SSI mikrokontroléru neumožňuje vyvolání přerušení po odeslání jednoho bajtu dat, ale přerušení se vyvolává neustále, než se vysílací zásobník periferie SSI nenaplní, a proto není možné určit okamžik odeslání pouze jednoho bajtu dat. Po uplynutí doby 80 μs se nastaví vývod SS do hodnoty 1, zpracují se přijatá data, na jejich základě se nastaví časovač na 20 ms nebo na 120 μs a přejde se do stavu IQRF_WAIT. Časové intervaly jsou o něco prodlouženy oproti minimálním hodnotám výrobce IQRF modulu. Obsluhu přerušení popisuje následující obrázek:

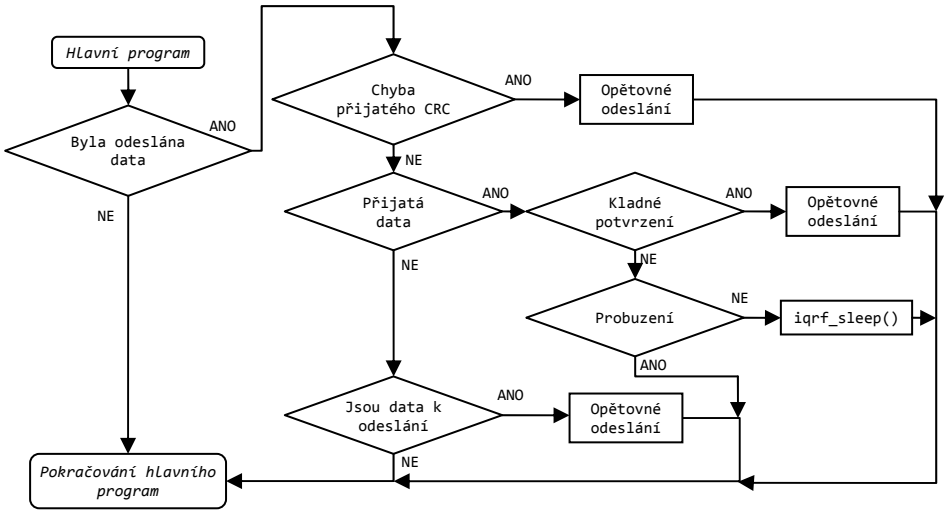


Obrázek 7.5 – Vývojový diagram zpracování přerušení časovače TIMER2A

Přenos dat je realizován ve dvou režimech, prvním režimem (IQRF_GET_STATUS) je zjištění stavu SPI IQRF modulu, tento režim je výchozí. Na základě přijatého stavu se rozhodne, jestli se budou přenášet data z IQRF modulu nebo pokud jsou připravena data k odeslání do IQRF modulu, na základě těchto informací se může přenos dat přepnout do režimu příjmu/odeslání dat (IQRF_RX_TX_DATA), oba režimy jsou vyznačeny v předešlém vývojovém diagramu. Na začátku přechodu do režimu příjmu dat se provede kontrola, jestli jsou připravena data k odeslání, pokud nejsou, naplní se záložní zásobník pro odesílaní dat (musejí být odeslána nějaká data, hlavně musí být správný kontrolní součet CRC), časovač se nastaví na dobu 120 µs. Po odeslání, resp. příjmu všech dat se nastaví příznak dokončeného datového přenosu, dojde k přepnutí do režimu kontroly stavu SPI IQRF modulu (IQRF_GET_STATUS) a časovač se nastaví na hodnotu 20 ms.

V hlavní smyčce programu se provede kontrola příznaku dokončeného přenosu dat pro IQRF modul. Provede se kontrola přijatého kontrolního součtu. Pokud se přijala nějaká data z IQRF modulu, budou zpracována. V případě, že je odpověď z IQRF modulu neadekvátní nebo kontrolní součet není správný, odešlou se data do

IQRF modulu znovu. V případě neodeslání nebo nepřijetí potvrzení, budou data znovu odelána centrální jednotce. Podrobnější pohled poskytuje obrázek 7.6:



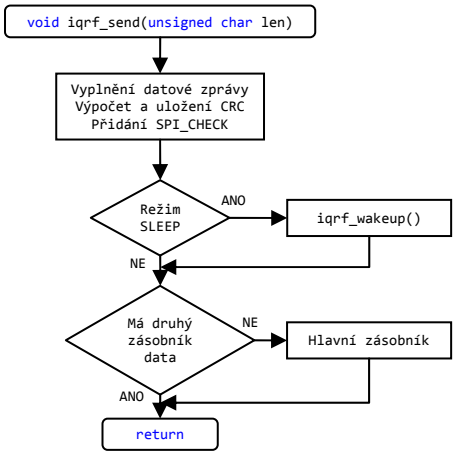
Obrázek 7.6 – Vývojový digram zpracování dat IQRF modulu v hlavní smyčce programu

Posledním stavem časovače je stav IQRF_WAKEUP, do které přejde po probuzení IQRF modulu z hlavní smyčky programu mikrokontroléru. Probuzení IQRF modulu se provede funkcí *iqrf_wakeUp*(), funkce vynuluje vývod SS (tím se provede fyzické probuzení IQRF modulu), časovač se nastaví na 10 µs (doba, po kterou bude držen vývod SS v hodnotě nula), spustí se časovač (v režimu spánku není časovač spouštěn).

Periferie SSI je inicializována funkcí *iqrf_init*, funkce rovněž inicializuje a spustí časovač TIMER2A, načte konfigurační nastavení IQRF modulu z paměti EEPROM (adresa 0) a odešle je. Pro odesílání dat jsou použity dva datové zásobníky, jeden slouží jako hlavní pro odesílání dat, druhý je záložní pro případ odeslání neplatných dat při příjmu dat z IQRF modulu. Každý zásobník má vlastní příznak určující platnost dat v zásobníku. IQRF modul se uvede do režimu spánku s minimální spotřebou zavoláním funkce *iqrf_sleep*. Funkce *iqrf_checkCrcRx* provede kontrolu kontrolního součtu přijatého z IQRF modulu. Funkce vrací hodnotu 1 v případě správného kontrolního součtu, hodnotu 0 v případě špatného kontrolního součtu.

Pro odeslání dat do IQRF modulu slouží funkce *iqrf_send* nebo *iqrf_reSend*, která pouze opětovně odešle data z hlavního odesílacího zásobníku. Funkce *iqrf_send* provede naplnění datového pole SPI_CMD a PTYPE nutných pro správné

přijetí dat IQRF modulem, vypočte kontrolní součet a uloží ho do vysílacího zásobníku, za kontrolním součtem je umístěn příkaz SPI_CHECK pro zjištění správnosti přijatého datového přenosu IQRF modulu a nastaví se příznak platnosti dat v hlavním zásobníku, data jsou pak již odesílána. Pokud se IQRF modul nachází v režimu spánku, bude probuzen a následně budou odeslána požadovaná data. Data zasílaná IQRF modulu se musejí do zásobníku před zavoláním funkce *iqrf_send* naplnit, k tomuto účelu slouží proměnná *iqrfTxBuf*, která je ukazatelem na data v hlavním odesílacím zásobníku. Diagram funkce iqrf_send ukazuje následující obrázek 7.7:

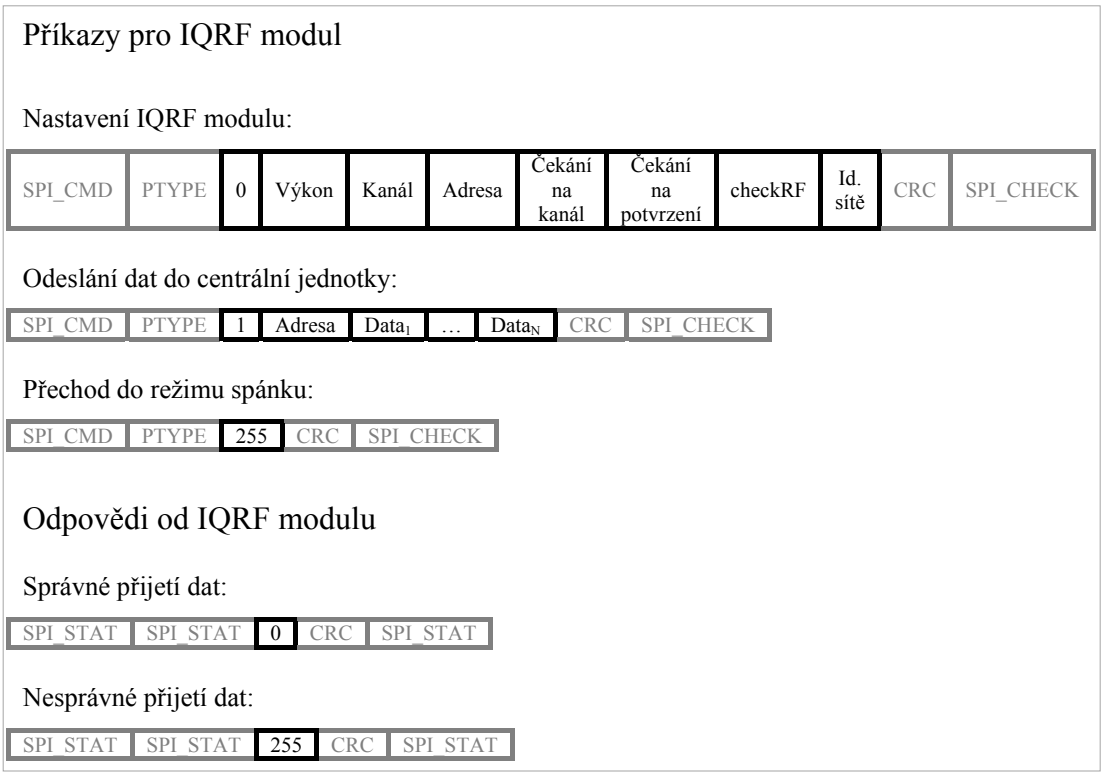


Obrázek 7.7 – Vývojový diagram funkce „iqrf_send“

Datové přenosy probíhající mezi mikrokontrolérem a IQRF modulem jsou popsány v následující podkapitole.

7.2.1 Popis formátu datových přenosů IQRF modulu

Mikrokontrolér LM3S811 zasílá bezdrátovému IQRF modulu příkazy a IQRF modul na základě těchto dat mikrokontroléru zasílá odpověď. Mikrokontrolér může zaslat hodnoty nastavení IQRF modulu, data k odeslání nebo příkaz pro vstup do režimu spánku. Všechny datové přenosy jsou ukázány na obrázku 7.8:



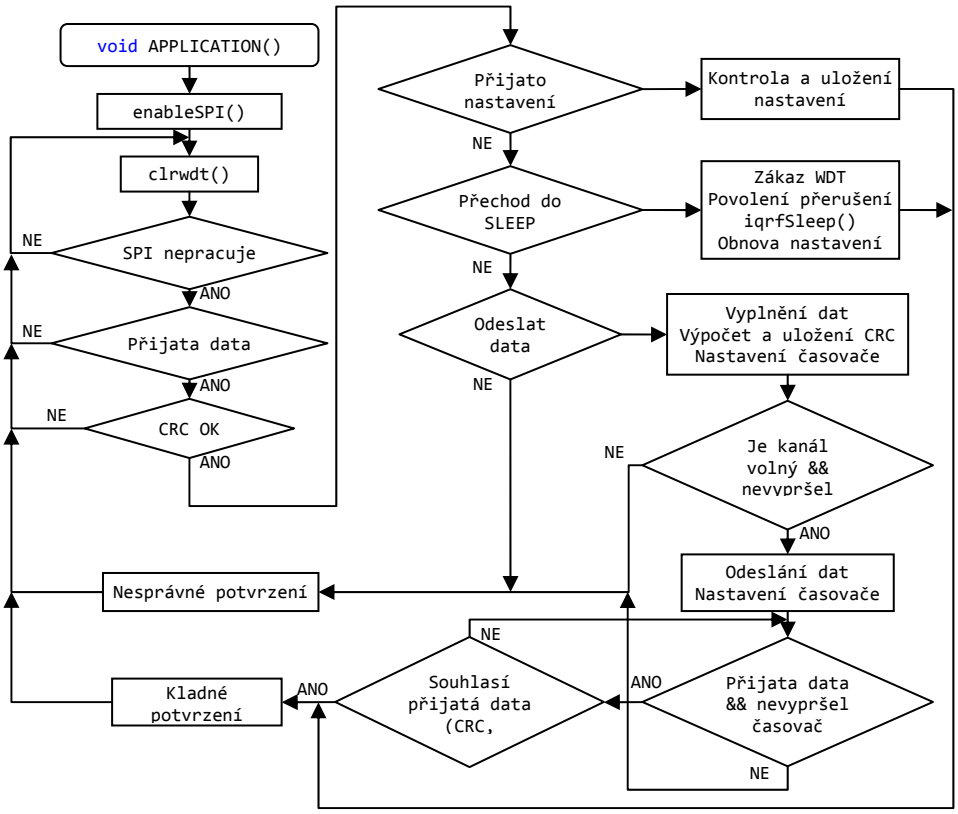
Obrázek 7.8 – Formáty datových přenosů pro IQRF modul

Přenášená data jsou vyznačena černou barvou, šedivá barva označuje nadbytečná data nutná pro správné přijetí dat IQRF modulem. Každý datový přenos je ukončen zjištěním SPI stavu IQRF modulu za účelem vyhodnocení správnosti příjmu dat (IQRF modul odešle hodnotu 3Fh.). V datovém přenosu pro nastavení IQRF modulu začínají data hodnotou 0 a následují data pro nastavení vysílacího výkonu, nastavení kanálu modulu, adresa modulu v rámci sítě, doba čekání na uvolnění kanálu v 10 ms, doba čekání na příjem potvrzení v 10 ms a identifikátor sítě. Příkaz pro bezdrátové odeslání dat začíná hodnotou 1, následuje adresa zařízení, kterému mají být data odeslána, následují data pro odeslání. Modul přejde do režimu spánku příkazem s číslem 255.

Na základě přijatých dat IQRF modul data potvrdí. Pokud byla data přijata správně, IQRF modul odpoví hodnotou 0, pokud data nebyla přijata správně, IQRF odešle hodnotu 255. Např. pokud se nepodaří odeslat data do centrální jednotky z důvodu zarušení kanálu nebo pokud nebylo přijato potvrzení, IQRF modul odpoví hodnotou 255, v případě správného odeslání dat IQRF odešle hodnotu 0. Po návratu z režimu spánku vždy IQRF modul odpoví hodnotou 0.

7.3 Programové vybavení IQRF modulu

Program IQRF modulu je napsán v jazyce C a využívá funkce operačního systému. Uživatelský program je napsán ve funkci *APPLICATION* v souboru *IQRF_main.c*. Na začátku funkce jsou deklarace proměnných nutných v dalších částech kódu. Hlavní program IQRF modulu je znázorněn na obrázku 7.9:



Obrázek 7.9 – Vývojový diagram hlavního programu IQRF modulu

V hlavní smyčce programu IQRF modulu je nejprve nutné nulovat Watchdog časovač nastavený na 4,1 s (výchozí hodnota). Následuje kontrola příjmu dat prostřednictvím SPI sběrnice. V případě příjmu dat po SPI sběrnici se provede kontrola správnosti přijatého kontrolního součtu CRC, v případě nesprávné hodnoty SPI sběrnice bude resetována. Je-li CRC kód v pořádku, je rozhodnuto o tom, která data byla přijata (nastavení, odeslání dat nebo režim spánku). V případě nastavení IQRF modulu jsou volány funkce operačního systému pro nastavení parametrů bezdrátové komunikace (vysílací výkon a kanál), hodnoty jsou zkontrolovány a v případě neplatné hodnoty je nastavena nejbližší platná hodnota. Ostatní parametry jsou uloženy do proměnných a použity v jiných částech kódu. Jsou-li data přijata

správně, bude uložena hodnota 0 do proměnné *bufferCOM* a odeslána po SPI sběrnici (data budou reálně odeslána, až si je mikrokontrolér vyžádá).

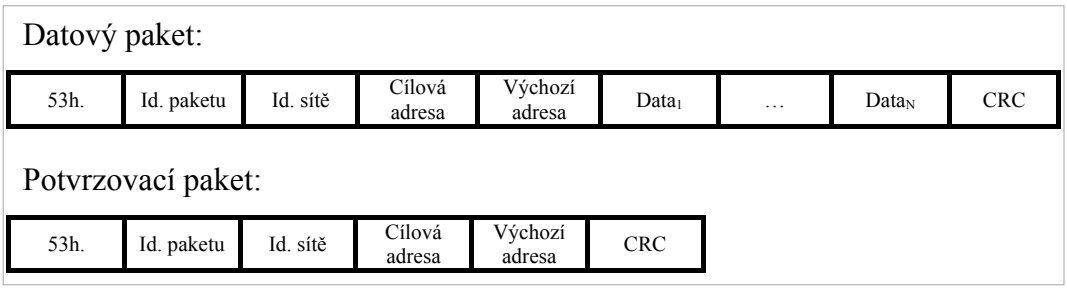
Pokud byla přijata data pro odeslání do centrální jednotky, naplní se datové pole *bufferRF* správnými daty (viz následující podkapitola), nastaví se časovač na dobu přijatou v nastavení IQRF modulu pro čekání na uvolnění kanálu. Během této doby je volána funkce *checkRF* s parametrem obdrženým také v nastavení IQRF modulu. Je-li detekováno malé zarušení kanálu (neprobíhá komunikace), data budou odeslána zavoláním funkce *RFTXpacket*, v opačném případě je odeslána odpověď 255 do mikrokontroléru (data nebyla odeslána). Po odeslání dat se nastaví časovač na dobu čekání na potvrzení (nastavení IQRF modulu), po tuto dobu se provádí čekání na příjem dat. Po přijetí dat jsou data zkontrolována, a pokud jsou správná (adresy a kontrolní součet jsou v pořádku), odpověď pro mikrokontrolér bude 0. Pokud nebyla data do stanovené doby přijata nebo nebyla správná, IQRF odpoví hodnotou 255 mikrokontroléru. Data sice byla odeslána, ale nebyla obdržena odpověď od přijímací strany.

Obdrží-li IQRF modul příkaz pro vstup do režimu spánku, je zakázáno přerušení, zastaven časovač Watchdog a povoleno přerušení od změny na portu B (přerušení změnou vývodu SS) mikrokntroléru PIC16F886 IQRF modulu a zavolána funkce *iqrfSleep*, po probuzení jsou všechny parametry vráceny do původních hodnot a je odeslána odpověď 0 do mikrokontroléru LM3S811.

Pokud IQRF modul obdržel neplatná data, odpoví hodnotou 255.

7.3.1 Formát bezdrátových paketů zasílaných IQRF modulem

Bezdrátové pakety mezi jednotlivými IQRF moduly jsou dvojího druhu: prvním je odeslání dat (datový paket) a druhým je potvrzení dat (potvrzovací paket). Odesílána data jsou IQRF modulem automaticky odeslána v paketu pro síť typu point to point popsaném v příloze G.2. Formáty bezdrátových paketů jsou zobrazeny na obrázku 7.10:



Obrázek 7.10 – Formát bezdrátových paketů senzorových jednotek

Každý paket je zahájen hodnotou *53h*. Následuje *identifikátor paketu*, pro datový paket je nejvyšší bit roven 0, pro potvrzovací paket je roven 1, zbývající část identifikátoru je rovna s datovým i potvrzovacím paketem (určuje typ odeslaného, resp. přijatého paketu). *Cílová adresa* představuje pro datový paket adresu zařízení, kterému se mají data doručit (přijatá z mikrokontroléru spolu s daty k odeslání), pro potvrzovací paket představuje adresu zařízení, kterému jsou data určena (musí být shodná s adresou odeslanou v nastavení IQRF modulu). *Výchozí adresa* pro datový paket je rovna adresa odesílajícímu zařízení, pro potvrzovací paket je rovna adresa zařízení vysílající potvrzení. Kontrolní součet *CRC* je roven xorovaným hodnotám odesílaných/přijímaných dat s hodnotou 12h. *Identifikátor paketu* je vždy po odeslání dat o jedničku navýšen, proto je jeho hodnota vždy jiná.

V adresách je obsažena i informace o typu senzorové jednotky. Pro detektor osob je nejvyšší bit roven hodnotě 0, pro detektor vozidel je roven hodnotě 1. Tímto způsoben bude centrální jednotka schopna rozpoznat senzorovou jednotku, od které přijala data, a tedy i vyhodnotit smysluplnost přijatých dat, např. senzorová jednotka detekce osob bude vysílat pouze omezený datový rozsah, naopak detektor osob bude vysílat širší rozsah platných číslic.

Formát bezdrátových paketů je volen zcela nahodile, proto je minimální pravděpodobnost, že by ve stejné lokalitě byly umístěny dva různé systémy se zcela stejným formátem dat. Síť je vždy identifikována svým identifikátorem, ale také vysílacím/přijímacím kanálem.

7.4 Popis zdrojového kódu mikrokontroléru detektoru osob

Zdrojový kód mikrokontroléru LM3S811 je odlišný pro oba typy senzorových jednotek ve zpracování analogového signálu z mikrovlnného senzoru zesíleného zesilovačem. Protože je zpracování signálu velice odlišné, je odlišná i inicializace zdrojového kódu mikrokontroléru.

Pro senzorovou jednotku detektoru osob je nejprve nastaven zdroj hodinového kmitočtu mikrokontroléru na hodnotu 6 MHz, není použita násobička kmitočtu PLL a hodinový kmitočet je roven externímu krystalovému oscilátoru. Tato nízká frekvence je volena z důvodu menší spotřeby jednotky. Následně je přiveden hodinový kmitočet k použitým periferiím mikrokontroléru. Provede se inicializace používaných vstupně/výstupních portů. Nastaví se časovače TIMER0A na hodnotu 1ms (odpovídá vzorkovací frekvenci 1 kHz), časovač se bude periodicky spouštět a bude spouštět analogový převodník. Analogově-digitální převodník využívá pouze vzorkovací sekci 3 nakonfigurovanou na spouštění od časovače a provádí vzorkování analogového vstupu ADC0. Inicializuje se I^2C sběrnice a komunikace s IQRF modulem, povolí se globální přerušení a přerušení pro analogovou vzorkovací sekci 3 a pro časovač TIMER2A provádějící časovaní komunikace s IQRF modulem. Pro správnou funkci komunikace s IQRF modulem je nutné počkat na odeslání nastavení IQRF modulu. Následuje načtení dat nutných pro digitální zpracování vzorků mikrovlnného senzoru z paměti EEPROM (adresa 10h.). V poslední části inicializace je nastaven ochranný Watchdog časovač na hodnotu 4 s a je spuštěn s možností vyvolání resetu mikrokontroléru.

V hlavní smyčce programu se provádí resetování Watchdog časovače a je provedena kontrola odeslaných dat IQRF modulu a případně jsou zpracována, kontroluje se také detekování cíle a stav odeslání informace o detekci cíle, viz následující text.

7.4.1 Popis zasílaných zpráv o detekování chodce

Bezdrátová komunikace mezi senzorovou jednotkou a centrální jednotkou křižovatky nesmí být velmi častá, aby se zbytečně nezahlcoval komunikační kanál (možné rušení komunikace ostatních zařízení – volné komunikační pásmo). Proto je

při detekování cíle odeslána zpráva o přítomnosti chodce do centrální jednotky a při ztrátě cíle je odeslána zpráva o nepřítomnosti chodce.

Z tohoto důvodu je v hlavní smyčce programu mikrokontroléru kontrována proměnná obsahující informaci o přítomnosti cíle a stav odeslání informace o cíli. V případě, že je detekován cíl a nebyla odeslána zpráva o přítomnosti cíle, je zpráva odeslána. Naopak pokud cíl není detekován a byla odeslána zpráva o přítomnosti cíle, je odeslána zpráva o nepřítomnosti cíle.

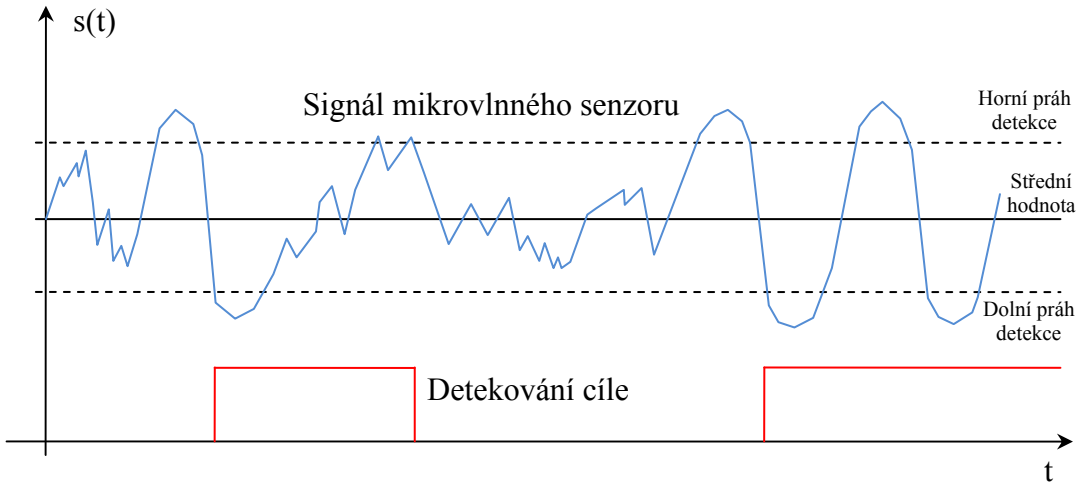
Všechny bezdrátové informace jsou odesílány na adresu 0, kterou má v rámci sítě pouze centrální jednotka. Je-li cíl detekován, je odeslána v datové části komunikačního paketu hodnota 1, v případě nedetekování cíle je odeslána hodnota 0.

7.4.2 Zpracování analogového signálu detektoru osob

Analogově-digitální převodník je spouštěn časovačem TIMER0A nastaveným na dobu časování 1ms, odpovídající frekvenci 1kHz, která je minimálně dvakrát vyšší než je hodnota horního mezního kmitočtu analogového filtru (366,056 Hz), je tedy dodrženo Nyquistovo vzorkovací kritérium.

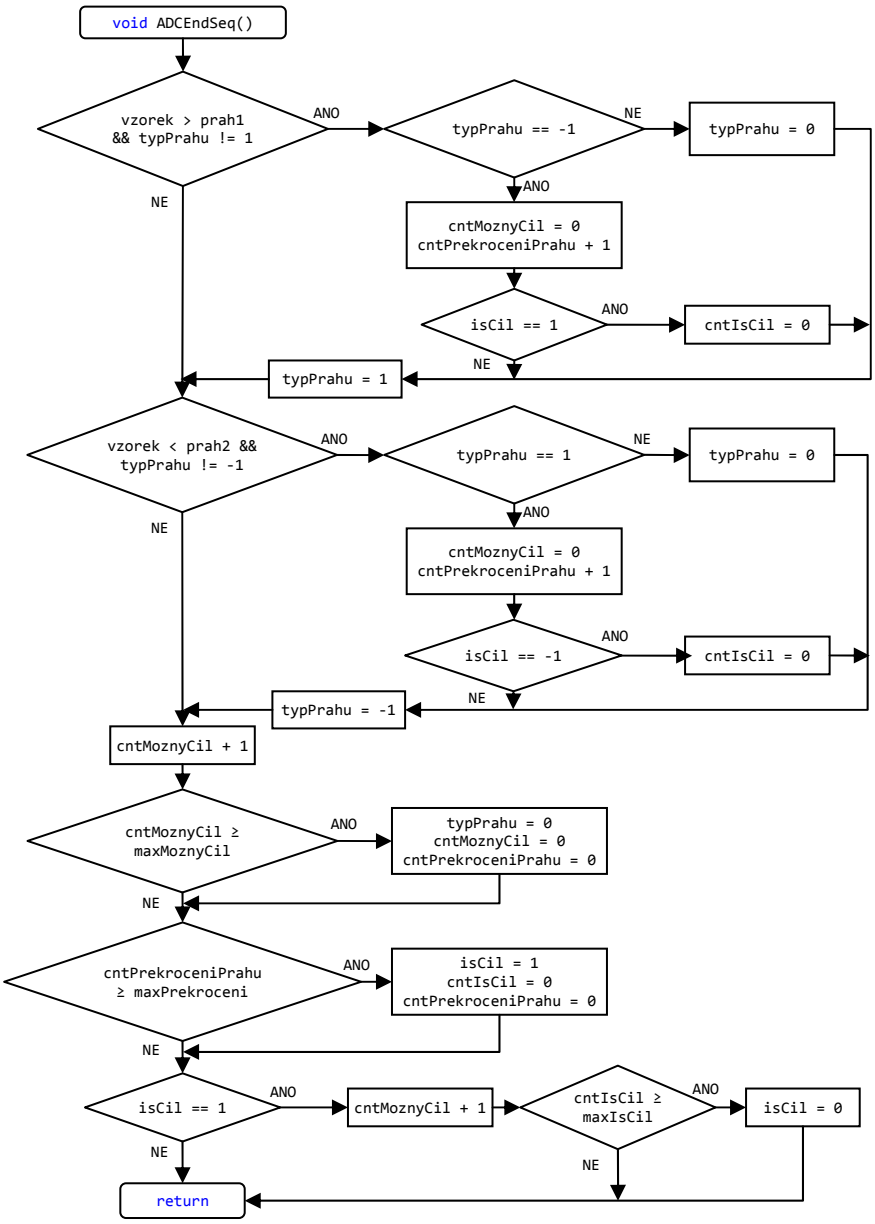
Analogový signál je porovnáván s hodnotou prahu získanou z hodnot vzorků šumu, tedy signálu bez přítomnosti cíle. Z těchto vzorků byl následně spočítán výkon šumu podle rovnice (2.26) a z rovnice (2.23) byla vypočtena hodnota prahu detekce. Při výpočtu prahu byla uvažována pravděpodobnost falešného poplachu $P_n = 0{,}6 \cdot 10^{-6}$. Přičtením hodnoty prahu ke střední hodnotě a odečtením hodnoty prahu od střední hodnoty signálu získáme dva prahy detekce (horní a dolní).

Bohužel i při uvažované nízké pravděpodobnosti detekce falešného poplachu docházelo často k náhodnému překročení prahu i bez přítomnosti cíle, pravděpodobnost falešného poplachu byla řádově vyšší než předpokládaná. Proto pro zvýšení spolehlivosti detekce chodce bylo zavedeno opatření v podobě kontroly překročení hodnoty horního a dolního prahu detekce. Pro detekování cíle musí docházet ke střídání překračování hodnoty detekce prahů. Pokud nejsou prahy překračovány střídavě, není cíl detekován. Zároveň je zaveden časový limit, do kterého musí dojít k překročení druhého typu prahu od posledního překročení prahu. Situaci detekování cíle a ztráty detekce cíle ukazuje následující obrázek:



Obrázek 7.11 – Ukázka detekování chodce detektorem osob

Mikrokontrolér si v paměti uchovává poslední typ překročení prahu. Dojde-li k překročení např. horního a poté dolního prahu (do určité doby), bude detekován cíl, pokud nedojde k překročení stejného prahu dvakrát za sebou, dojde ke ztrátě detekce cíle. Při detekování cíle je nulován časovač, pokud nedojde po stanovenou dobu k překročení opačného prahu, bude cíl ztracen a bude zaznamenán typ překročení prahu (žádný práh). Proměnná *typPrahu* obsahuje informaci o naposledy překročeném typu prahu, hodnota 1 znamená horní práh, -1 dolní práh a hodnota 0 znamená žádný práh (začátek střídání prahů). Podrobný popis zpracování analogového signálu poskytuje obrázek 7.12:



Obrázek 7.12 – Diagram zpracování analogového signálu detektoru osob

Zpracování signálu se provádí v přerušení třetí vzorkovací sekce A/D převodníku. Vždy se provádí kontrola aktuálního vzorku s hodnotou prahu a s typem posledně překročeného prahu, pokud je detekováno první překročení prahu, provede se kontrola předešlého prahu, pokud byl překročen opačný práh, je přičtena hodnota 1 k čítači překročení prahu a vynulován časovač možného cíle. Tento časovač slouží pro časování doby nutné pro překročení opačného prahu. Pokud nedojde k překročení opačného typu prahu do nastavené doby, je nastaven typ prahu na žádný. V případě, že už byl detekován cíl (proměnná *isCil* je rovna hodnotě 1), je nulován časovač výskytu cíle.

Cíl je detekován určitým počtem různě se střídajících typů prahů hned za sebou, proměnná *cntPrekroceniPrahu* je rovna určité hodnotě. Detekcí cíle dojde k nastavení proměnné *isCil*, vynulování časovače *cntIsCil* a proměnné *cntPrekroceniPrahu* prahu (pro nulování časovače *cntIsCil* je nutná opětovná detekce překračování prahů).

V každém průběhu přerušení A/D převodníku je navyšována hodnota časovače *cntMoznyCil*, při dosažení maximální hodnoty je časovač vynulován, nastaven žádný typ posledně překročeného prahu a vynulován časovač *cntPrekroceniPrahu*. Časovač *cntMoznyCil* slouží jako ochrana opožděných překračování prahů, prahy musí být překračovány před dosažením maximální hodnoty časovače.

Je-li detekován cíl (proměnná *isCil* je rovna 1), je navyšována hodnota časovače *cntIsCil*, pokud tento časovač dosáhne své maximální hodnoty, je cíl ztracen. Tento časovač má za úkol minimalizovat rychlé detekování a ztrátu cíle při nerovnoměrném výskytu chodce, tímto způsobem se také minimalizuje bezdrátová komunikace s centrální jednotkou.

Hodnoty prahů, maximální hodnoty čítačů a časovačů jsou načteny během inicializace mikrokontroléru z paměti EEPROM. Časovače jsou představovány pouze paměťovými místy, která jsou navyšována v přesných časových okamžicích v přerušení A/D převodníku.

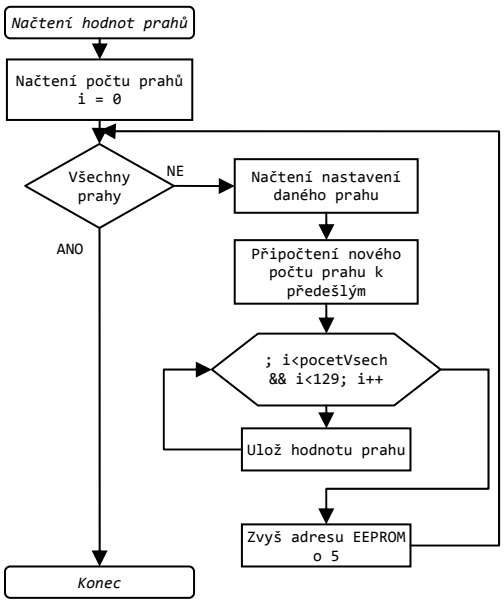
7.5 Popis zdrojového kódu mikrokontroléru detektoru vozidel

Na začátku inicializace chodu mikrokontroléru je nutné nastavit hodinovou frekvenci, pro senzorovou jednotku detekce vozidel je tato frekvence nastavena na hodnotu 12,5 MHz. Frekvence je vytvořena z externího krystalového oscilátoru s hodnotou 6 MHz, pro vytvoření hodinové frekvence 12,5 MHz je použita násobička kmitočtu PLL, následně vydělená maximálně možnou hodnotou. Tato vyšší frekvence je nutná pro správnou funkci A/D převodníku a má za následek větší proudovou spotřebu celé jednotky.

Po nastavení hodinové frekvence je povolena hodinová frekvence pro používané periferní obvody. Následuje inicializace vstupně-výstupních portů,

inicializace časovače TIMER0A pro časování spouštění A/D převodníku, frekvence spouštění je přibližně nastavena na 15 kHz, tato hodnota odpovídá více než dvojnásobku maximální frekvence analogového signálu (přibližně 7300 Hz). Následuje nastavení A/D převodníku stejně jako v případě senzorové jednotky detekce osob. Je nutné nastavit periferii I^2C a SSI, načíst a odeslat nastavení bezdrátové komunikace do IQRF modulu. Povolí se nutná přerušení a načte se nastavení časování zpracování analogového signálu. Všechny tyto operace jsou obdobné s jednotkou detekce osob.

Na rozdíl od senzorové jednotky detekce osob je nutné provést načtení více prahů (viz podkapitola 7.5.1). Prahy představují vektor dat, kde na různých místech jsou jiné hodnoty prahů, vypočtené hodnoty z analogového signálu pak stačí s tímto vektorem porovnat pomocí indexů. Tento způsob je mnohem výpočetně rychlejší než např. porovnávání jednotlivých hodnot. Hodnoty prahů jsou v paměti EEPROM uloženy tak, že na adrese 20h. je uložen počet všech prahů, následují jednotlivé prahy, které představují počet hodnot daného prahu ve vektoru prahů a hodnota prahu. Informace o počtu a hodnotě prahu zabírá v paměti celkem pět bajtů, počet představuje jeden bajt, hodnota prahu je 32bit. číslo, zabírá tedy čtyři bajty. Postupným načtením nastavení všech hodnot prahů a jejich postupným ukládáním do vektoru prahů získáme kompletní realizaci prahů. Načtení prahů ukazuje následující diagram:

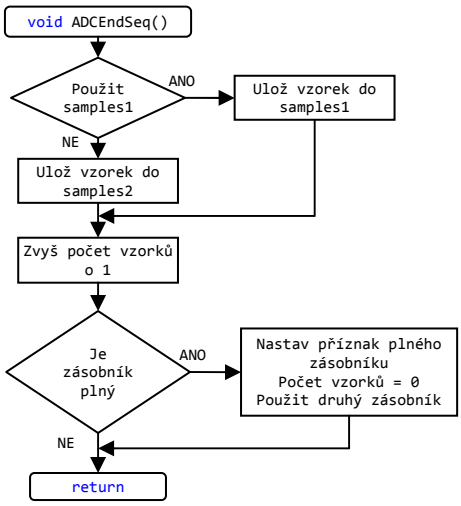


Obrázek 7.13 – Vývojový diagram načítání prahů detektoru vozidel

Další odlišností v inicializaci chodu mikrokontroléru oproti jednotce detekce osob je v nastavení zpracování rychlé Fourierovy transformace FFT. Pro výpočet FFT je použita knihovna CMSIS popsaná v [28]. Tato knihovna je obdobou knihovny StellarisWare umožňující také přístup k periferiím mikrokontroléru. Od verze 2.0 je součásti knihovny CMSIS také DSP knihovna nabízející právě výpočet FFT, výpočet výkonového spektra apod. Inicializace FFT je realizována pomocí funkce *arm_cfft_radix4_init_g31*. Prvním parametrem je ukazatel na strukturu typu *arm_cfft_radix4_instance_q31*, druhý představuje počet zpracovávaných vzorků, třetí určuje typ transformace (0 = dopředná, 1 = zpětná) a poslední parametr určuje seřazení výstupní posloupnosti dat (0 = reverzní, 1 = normální).

Posledním krokem inicializace je čekání na odeslání nastavovacích dat pro IQRF modul. Pro jednotku detekce vozidel není použit hlídací časovač Watchdog, z toho důvodu, že na rozdíl od aplikační poznámky výrobce by měl Watchdog časovač pracovat i při použití násobičky kmitočtu PLL, časovač byl schopen provést reset mikrokontroléru, ale z resetovacího stavu se již nevrátil do funkčního. Z hlediska jeho funkce nebylo použití nutné.

V přerušení A/D převodníku *ADCEndSeq* se provádí ukládání aktuálního vzorku analogového signálu. Zde jsou použity dva datové zásobníky (*samples1* a *samples2*) pro ukládání vzorků signálu. Oba tyto zásobníky se navzájem přepínají, do jednoho jsou ukládány vzorky a druhý je mezitím v hlavní smyčce programu zpracováván a naopak. Z důvodu, že funkce pro výpočet FFT předpokládá vstupní formát 1.31 (viz dále), je hodnota vzorku z A/D převodníku posunuta o 21 bitů doleva, tím se budou data pohybovat v rozmezí 0 až 1 ve formátu 1.31. Popis ukládání vzorků ukazuje následující diagram:



Obrázek 7.14 – Vývojový diagram ukládání vzorků detektoru vozidel

V hlavní smyčce programu se stejně jako v jednotce detekce osob provádí kontrola stavu komunikace s IQRF modulem, na základě různých stavů jsou data zpracována, viz popis v předchozím textu.

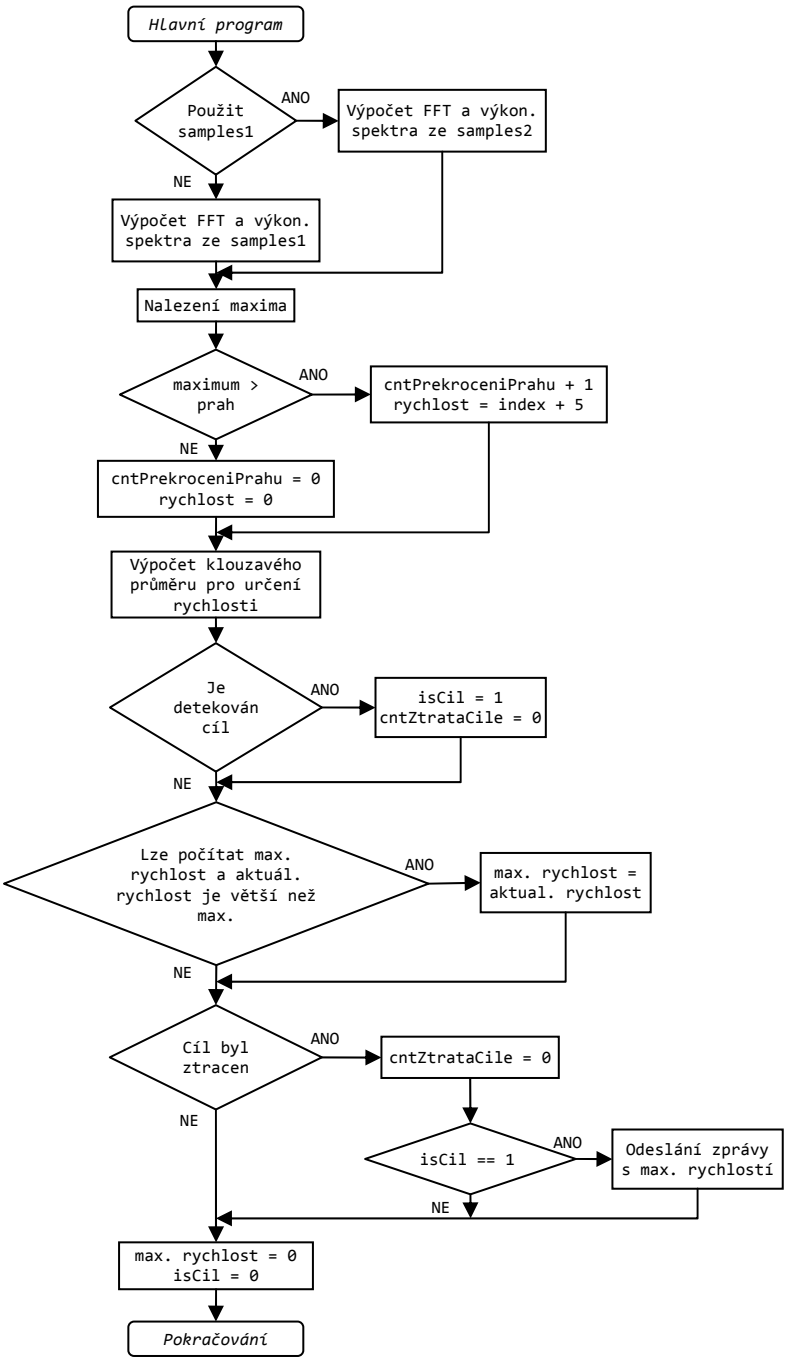
7.5.1 Zpracování analogového signálu detektoru vozidel

Protože frekvence výstupního signálu z mikrovlnného senzoru je úměrná rychlosti pohybujícího se objektu, viz rovnice (2.18) a (2.4). Výpočtem frekvenčního spektra obdržíme přehled o přítomných frekvencích v signálu a o jejich velikosti. Pohybující se objekt bude v signálu představovat velkou hodnotu dané frekvence, proto se ve vypočteném frekvenčním spektru nalezne maximum odpovídající frekvenci, resp. rychlosti, pohybu objektu. Pro určení maximální hodnoty je nutné provést výpočet výkonového spektra, viz kapitola 3.5. Frekvenční spektrum je vypočítáváno z 256 vzorků signálu. Z vypočteného frekvenčního spektra stačí zpracovávat pouze kladné frekvence (záporné mají stejné hodnoty pro reálný signál), dostáváme 129 hodnot frekvenčního spektra, pro uvažovanou maximální rychlost kolem 150 $km \cdot h^{-1}$, odpovídá každá frekvenční čára přibližně 1 $km \cdot h^{-1}$.

V hlavní smyčce programu se provádí kontrola příznaku plného zásobníků vzorků vstupního signálu. Je-li zásobník plný, provede se výpočet FFT vzorků v daném zásobníku. Zde je smysl zásobníků oproti přerušení A/D převodníku opačný, probíhá ukládání např. do zásobníku *samples1*, také se zpracovávají data v zásobníku *samples2* apod. K výpočtu FFT se používá funkce *arm_cfft_radix4_q31*

z knihovny CMSIS, prvním parametrem funkce je ukazatel na stejnou datovou strukturu, která byla použita při inicializaci FFT, druhý parametr představuje ukazatel na vzorky signálu. Funkce využívá výpočet popsaný v kapitole 3.3. Vstupním formátem dat je zlomkový formát 1.31 (nejvyšší bit představuje znaménko, další bity jsou s vahami 2^{-1}, 2^{-2}, …), výstupní formát je také zlomkový ve formátu 9.23. Funkce vrací vypočtené spektrum ve stejném zásobníku představující zároveň zásobník vzorků signálu, vzorky jsou přepsány. Protože je pro výpočet FFT vstupní signál považovaný za komplexní (má reálnou a imaginární složku), je velikost zásobníku dvakrát větší, než je počet zpracovávaných vzorků. Liché indexy zásobníku představují reálnou složku a sudé imaginární. Protože jsou vzorky vždy během výpočtu FFT přepisovány, je nutné pro reálný signál vždy nulovat imaginární složku, to je zajištěno již při ukládání vzorků v přerušení A/D převodníku.

Dalším krokem je výpočet výkonového spektra prostřednictvím funkce *arm_cmplx_mag_squared_q31*, první parametr funkce představuje frekvenční spektrum, druhý je ukazatel na výstupní pole dat, poslední parametr je počet komplexních vzorků. Vstupní formát je opět zlomkový ve formátu 1.31 a výstupní ve formátu 3.29. Zlomkový formát je pouze myšlený (je to pouze číslo v paměti, které je nějak reprezentováno), proto nevadí použití různých zlomkových formátů za sebou, pouze má výsledné číslo jinou hodnotu. Pro úsporu času výpočtu je vypočteno výkonové spektrum pouze pro prvních 129 vzorků, tedy pro kladné frekvence. Následuje nalezení maximální hodnoty ve výkonovém spektru, prvních pět spektrálních čar je vynecháno, zde se vyskytují pomalu se pohybující objekty, např. chodci. Pro nalezení maximální hodnoty je použita funkce *arm_max_q31* z knihovny CMSIS, první parametr je ukazatel na pole, kde se maximum hledá, druhým parametrem je počet dat, třetím je ukazatel na paměťové místo, kam se uloží hodnota maxima, posledním parametrem je ukazatel, kam se uloží pozice nalezeného maxima. Popis celého analogového zpracování ukazuje následující obrázek 7.15:

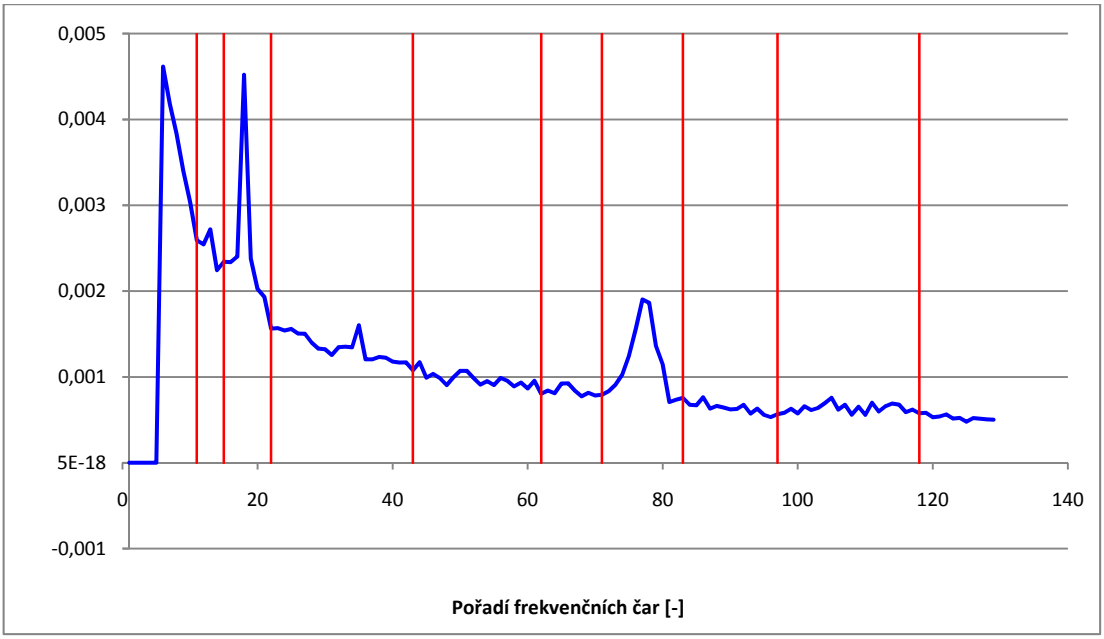


Obrázek 7.15 – Vývojový diagram zpracování signálu detektoru vozidel

Pro určení přítomnosti cíle je nutné frekvenční spektrum vhodně prahovat. Prahování se provede podobně jako u detekce osob, zde se ovšem nebude prahovat signál, ale frekvenční spektrum. Protože šum nabývá různých frekvencí, je nutné celé spektrum vhodně rozdělit, z těchto částí se následně vypočte průměr a výkon podle vztahu (2.26), následně se z vztahu (2.23) určí práh detekce pro pravděpodobnost falešného poplachu $P_n = 0{,}6 \cdot 10^{-6}$. Pro rozdělené části tak vypočteme různé

hodnoty prahů, tyto prahy tvoří vektor prahů, který je v inicializační fázi mikrokontroléru naplněn.

Frekvenční spektrum bylo rozděleno na jednotlivé části tak, že se provedlo průměrování několika frekvenčních spekter, tím byly zjištěny nejvíce zastoupené šumové složky. Použité rozdělení je ukázáno na následujícím obrázku 7.16:



Obrázek 7.16 – Ukázka rozdělení frekvenčního spektra pro určení prahů detekce

Po nalezení maxima se hodnota maxima porovná s prahem odpovídající pozici nalezeného maxima ve vektoru prahů. Pokud je maximum nad prahem, je navýšen čítač počtu překročení prahu *cntPrekroceniPrahu* o hodnotu 1 a je uložena hodnota polohy maxima do pomocného kruhového zásobníku. Pokud nebylo maximum nad prahem, je přidána hodnota 0 do kruhového zásobníku a je vynulován časovač *cntPrekroceniPrahu*. Následně se z hodnot v kruhovém zásobníku vypočte průměr, tento zásobník obsahuje pouze čtyři poslední hodnoty, proto vypočtený průměr odpovídá klouzavému průměru.

Protože maximální hodnota výkonového spektra mnohdy náhodně překračuje hodnotu prahu a je tak falešně detekován cíl, je nutné stejně jako v případě detekce osob provést hlubší zpracování. Zpracování probíhá formou kontroly četnosti překračování prahu. Proto se pro uznání přítomnosti cíle kontroluje hodnota čítače *cntPrekroceniPrahu*, pokud je hodnota vyšší než načtená z paměti EEPROM

(hodnota záleží na tom, zda byl již detekován cíl nebo ne), je nastaven příznak přítomnosti cíle *isCil* a je nulován čítač ztráty cíle *cntZtrataCile*.

Výsledná rychlost je určena jako maximální hodnota z průměrovaných hodnot. Rychlost se přímo v jednotce neurčuje, pouze se vyhodnocuje poloha maxima ve frekvenčním spektru, ale poloha je přímo úměrná rychlosti objektu. Pro určování maximální rychlosti musí být splněna podmínka, že čítač *cntPrekroceniPrahu* bude mít určitou hodnotu. Tím se eliminuje možnost přítomnosti náhodného rychlého objektu, resp. šumu, vysoká hodnota by v průměrované hodnotě mohla nabývat vyšší hodnoty, než která odpovídá rychlosti objektu. Předpokládá se krátký výskyt těchto šumů. Zároveň nesmí být hodnota čítače *cntPrekroceniPrahu* pro výpočet maximální hodnoty rychlosti moc vysoká, aby nedošlo k vynechání opravdové maximální hodnoty pohybujícího se objektu.

Posledním krokem je kontrola ztráty cíle. Cíl je ztracen, pokud čítač *cntZtrataCile* dosáhl v závislosti na přítomnosti cíle určité hodnoty, čítač *cntZtrataCile* je inkrementován v každém průběhu výpočtu. Je-li detekována ztráta cíle, je nulován čítač *cntZtrataCile*. Pokud byl dříve detekován cíl, je odeslána informace o projetém vozidle do centrální jednotky křižovatky, maximální rychlost je nulována a je též nulován příznak přítomnosti cíle *isCil*.

Problém může nastat v situaci, kdy těsně za sebou pojedou dvě a více vozidel. Tuto situaci by mohla jednotka vyhodnotit jako přítomnost jednoho vozidla a došlo by ke změření rychlosti pouze jednoho vozidla. Tento případ nebyl v rámci testování ověřen.

Časová náročnost použitých algoritmů musí být menší než 17 ms (doba nutná pro převedení 256 vzorků analogového signálu při dané vzorkovací frekvenci), výpočty frekvenčního spektra, výkonového spektra a nalezení maximální hodnoty trvají přibližně 9 ms, v této době nejsou zahrnuty časy nutné pro ukládání vzorků v přerušení A/D převodníku a časy při komunikaci s IQRF modulem. Všechny časy v přerušeních trvají pouze nezbytně krátkou dobu. Proto veškeré zpracování časově netrvá déle než 17 ms a lze zpracovávat všechny analogové vzorky.

7.5.2 Popis zasílání zpráv o průjezdu vozidla

Z důvodu, že je nutné během zpracování maximálních hodnot výkonového frekvenčního spektra, resp. filtrovaných maximálních hodnot, hledat maximální filtrovanou hodnotu, musí být informace o přítomnosti cíle, tedy vozidla, odeslána až po jeho projetí, není jasné, kde se maximum bude nacházet. Z tohoto důvodu jsou senzorové jednotky detekce vozidel umístěny v dostatečné vzdálenosti od křižovatky. Proto je informace o rychlosti vozidla odeslána až v okamžiku, kdy byla detekována ztráta cíle.

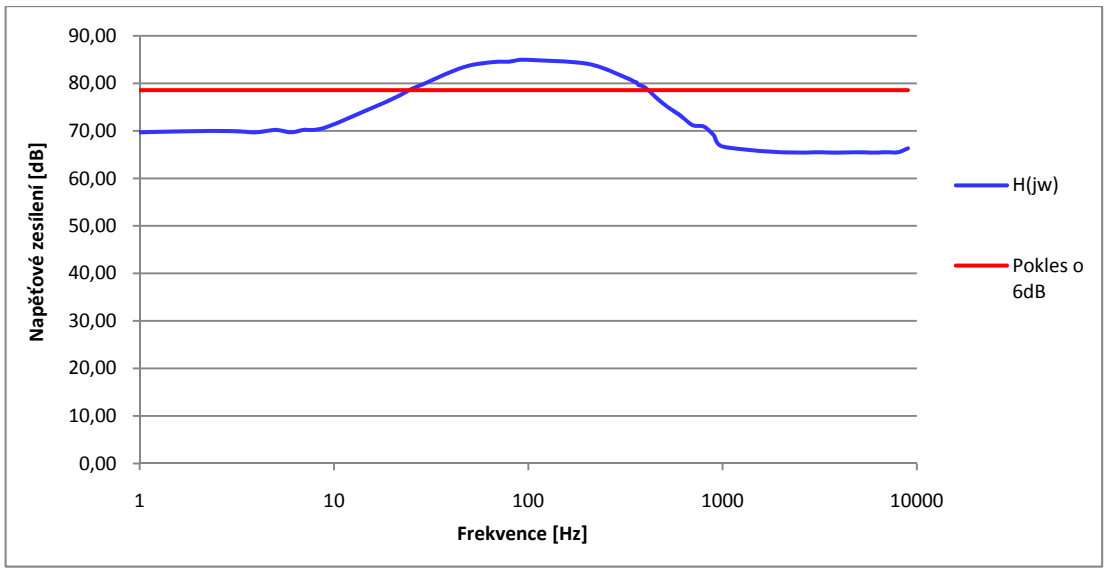
Pokud došlo ke ztrátě cíle a byl předtím cíl detekován, je naplněn datový zásobník pro odeslání do IQRF modulu. První bajt obsahuje příkaz pro odeslání dat do centrální jednotky, druhý obsahuje adresu příjemce (hodnota 0 – pouze centrální jednotka), do posledního bajtu je uložena maximální filtrovaná hodnota poloh maxim výkonových spekter. Protože se maxima hledají pouze v kladných frekvencích, maximální možné rychlosti odpovídá číslo 129 (pořadí poslední frekvenční čáry), je možné tuto informaci uložit do jediného bajtu. Senzorová jednotka neodesílá přímo rychlost projetého vozidla, ale pouze číslo úměrné rychlosti vozidla. Pro určení rychlosti vozidla by bylo nutné použít desetinná čísla v plovoucí řádové čárce, tyto výpočty ovšem nejsou na použitém mikrokontroléru možné. Centrální jednotka toto přijaté číslo porovná s čísly odpovídajícím různým rychlostem a bude schopna určit skutečnou rychlost vozidla.

8 Testování a měření senzorových jednotek

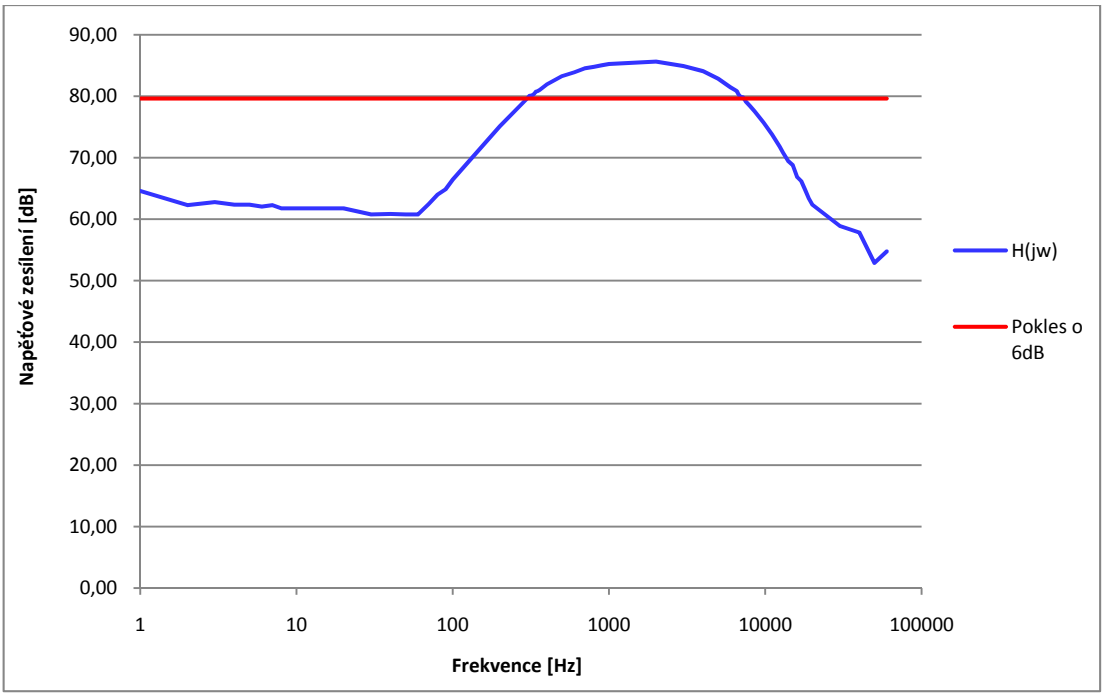
Pro ověření správné funkce navržené senzorové jednotky je nutné provést její testování a pro ověření dosahu mikrovlnného senzoru je nutné provést měření senzorové jednotky. Měření senzorové jednotky je odlišné dle jejího typu. Součástí měření senzorových jednotek bylo také měření dosahu bezdrátové komunikace IQRF modulů.

8.1 Ověření funkce analogového zesilovače

Na navrženém analogovém zesilovači bylo provedeno měření za účelem ověření navrženého frekvenčního pásma, které zesilovač zesiluje. Na vstup zesilovače byl přiveden sinusový signál z generátoru, snížený pomocí rezistorového děliče na přijatelnou hodnotu, aby nedocházelo k limitaci výstupního signálu. Na výstup zesilovače byl připojen osciloskop, připojený také na výstup generátoru. Naměřené efektivní hodnoty vstupního i výstupního napětí pro různé frekvence byly vynášeny do tabulky. Vstupní hodnota napětí byla přepočtena na opravdovou hodnotu vstupního napětí zesilovače ze znalosti rezistorového děliče. Naměřené průběhy pro oba typy zesilovače jsou následující:



Obrázek 8.1 – Naměřené napěťové zesílení detektoru osob



Obrázek 8.2 – Naměřené napěťové zesílení detektoru vozidel

Provedeme-li porovnání s očekávanými hodnotami mezních kmitočtů shrnutých v tabulce 6.1 s naměřenými, získáme toto srovnání:

Tabulka 8.1 – Srovnání naměřených a předpokládaných mezních kmitočtů detektoru osob

	Dolní mezní kmitočet	Horní mezní kmitočet
Kmitočty z programu MATLAB	21,009 [Hz]	366,056 [Hz]
Naměřené kmitočty	~ 25 [Hz]	~ 420 [Hz]

Srovnáním zjistíme, že horní mezní kmitočet je již poměrně značně vysoký oproti předpokládané hodnotě. Hodnota dolního mezního kmitočtu je již více blízká požadované hodnotě. Chyby mohou být způsobeny jednak značně omezeným frekvenčním krokem při měření a také výrobní tolerancí hodnot součástek filtru. Porovnáme-li napěťové zesílení, kdy předpokládané zesílení bylo rovno 85,27 dB a naměřené zesílení rovno 84,57 dB, zjistíme, že zesílení jsou téměř shodná.

Pro detektor vozidel je srovnání mezních frekvencí následující:

Tabulka 8.2 – Srovnání naměřených a předpokládaných mezních kmitočtů detektoru vozidel

	Dolní mezní kmitočet	Horní mezní kmitočet
Kmitočty z programu MATLAB	291,25 [Hz]	7241,5 [Hz]
Naměřené kmitočty	~ 300 [Hz]	~ 7250 [Hz]

Naměřené a vypočtené hodnoty mezních kmitočtů jsou pro detektor vozidel velice blízké. Naměřené hodnoty jsou vlivem značně omezeného frekvenčního kroku pouze přibližné, tím lze naměřené hodnoty považovat za totožné s předpokládanými. Naměřené zesílení zesilovače je rovno hodnotě 85,63 dB, předpokládané zesílení je rovno 85,66 dB, zesílení jsou stejná.

Předpokládané a naměřené frekvenční charakteristiky se nejvíce liší v pásmech před a za navrhovaným frekvenčním pásmem. Tyto charakteristiky jsou způsobeny přítomností šumu. Hodnota šumu je značně nižší u detektoru vozidel, zde je dolní mezní kmitočet umístěn poměrně vysoko, tím se odfiltrují frekvenční složky, které způsobují nejvíce šumu.

8.2 Měření dosahu bezdrátové komunikace IQRF modulů

Pro ověření dosahu bezdrátové komunikace je nutné vybavit IQRF modul vhodnou anténou. Z důvodu, že na dopravní křižovatce je vhodné osadit senzorové jednotky směrovými anténami, byla anténa vybírána právě jako směrová.

Pro používané komunikační pásmo 868 MHz je vhodná anténa BD 910A vyráběná společností RCD Radiokomunikace, jejíž vlastnosti jsou uvedeny v příloze J.

Dosah bezdrátové komunikace byl měřen tak, že byl u vysílací antény umístěn IQRF modul naprogramovaný jako vysílač vysílající periodicky datovou zprávu, u přijímací antény byl umístěn také IQRF modul přijímající zprávu. Pokud přijímač obdržel zprávu, vyhodnotil ji a pokud byla zpráva v pořádku, odeslal potvrzení, které vysílač přijal. Tím bylo možné u vysílače zpětně určit, zda byla odeslaná zpráva správně přijata přijímačem.

S dosahem vysílání a příjmu IQRF modulu byla zároveň měřena chybovost přijatých, resp. odeslaných zpráv. Vysílač průběžně vyhodnocoval počet odeslaných a potvrzených zpráv. Poměrem těchto hodnot byla zjištěna chybovost datových přenosů. Na vzdálenost až 700 m byla dosažena úspěšnost doručených zpráv až kolem 99 %. Měření probíhalo v průmyslovém areálu za běžného dopravního provozu, který bude v reálné křižovatce mnohem vyšší. Dosah bezdrátové komunikace byl zkoušen i na mnohem větší vzdálenost, vyslaná a potvrzená datová zpráva byla zachycena až na vzdálenost kolem 4,5 km. Toto měření neobsahovalo zjištění chybovosti přenosu, ale mělo pouze zjistit dosah bezdrátové komunikace.

8.3 Měření senzorové jednotky detekce pohybu osob

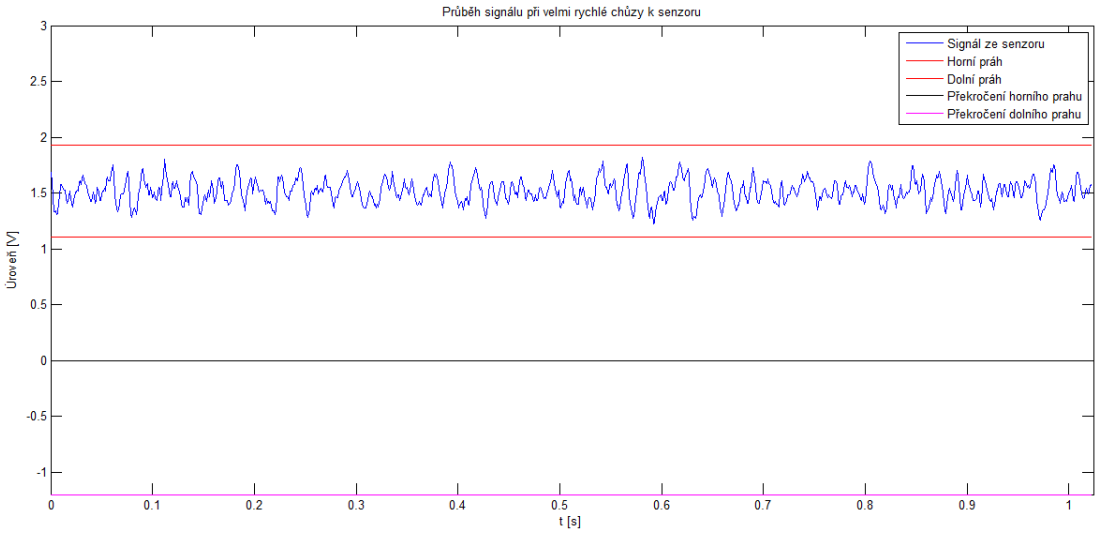
Měření jednotky pro detekci osob na přechodu pro chodce bylo prováděno v laboratorních prostorech. Senzorová jednotka byla umístěna ve výšce 2,28 m pod úhlem přibližně 10 °, mikrovlnný senzor byl orientován vertikálně, tedy svazek antény byl široký.

Měření probíhalo na chodbě o šířce 1,96 m. Objektem byla dospělá osoba o výšce 180 cm. Senzorová jednotka posílala navzorkovaná data do počítače, kde byla ukládána a následně zobrazována. Instalaci senzorové jednotky ukazuje následující obrázek:

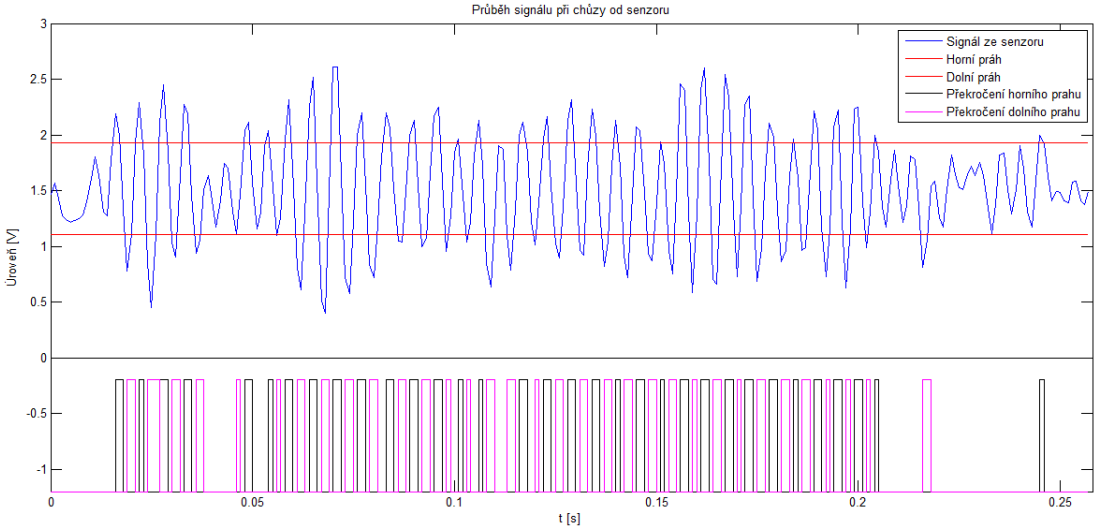
Obrázek 8.3 – Instalace senzorové jednotky detekce osob

U senzorové jednotky je také instalována anténa BD 910A pro účely testování bezdrátové komunikace.

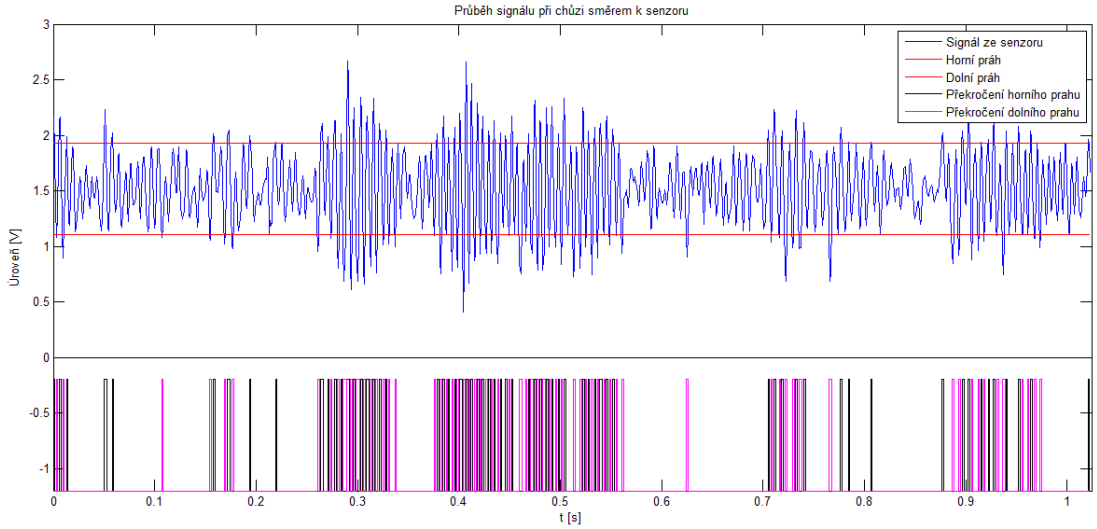
Naměřené průběhy jsou zobrazeny na následujících obrázcích:



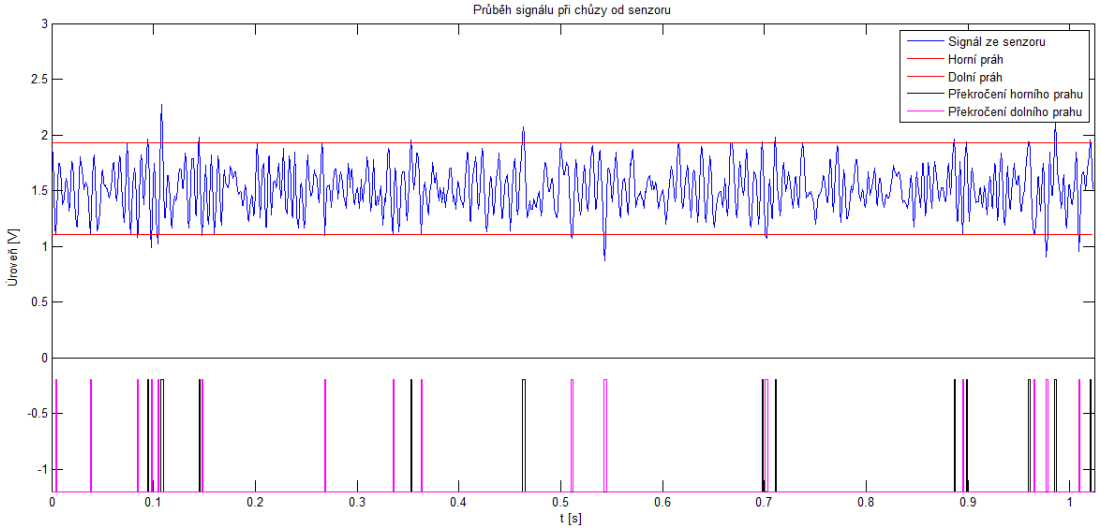
Obrázek 8.4 – Průběh signálu z analogového zesilovače bez přítomnosti cíle



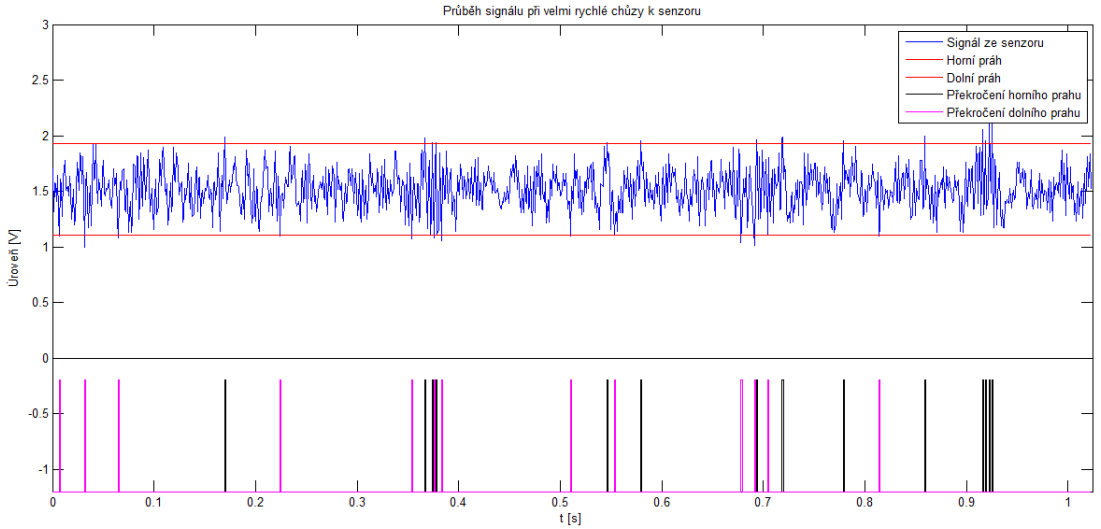
Obrázek 8.5 – Průběh signálu z analogového zesilovače při chůzi směrem od senzoru ve vzdálenosti 5,5 m od senzorové jednotky



Obrázek 8.6 – Průběh signálu z analogového zesilovače při chůzi směrem od senzoru ve vzdálenosti 8 m od senzorové jednotky



Obrázek 8.7 – Průběh signálu z analogového zesilovače při chůzi od senzoru ve vzdálenosti 15 m od senzorové jednotky



Obrázek 8.8 – Průběh signálu z analogového zesilovače při chůzi k senzoru ve vzdálenosti 5,5 m od senzorové jednotky při velmi rychlé chůzi

Každý obrázek obsahuje průběh navzorkovaného signálu z analogového zesilovače, který byl v počítači zpětně převeden z digitálního čísla na hodnotu napětí. Součástí obrázků jsou také oba prahy detekce, horní i dolní. Ve spodní části obrázků jsou vyneseny okamžiky překročení horního i dolního prahu detekce.

První z naměřených průběhů ukazuje situaci, kdy není detekován žádný cíl, průběh ukazuje pouze šum. Další obrázek ukazuje situaci, kdy osoba směřovala směrem od senzorové jednotky normální chůzí a byla od senzorové jednotky vzdálena 5,5 m. Na obrázku je zřetelně patrné střídavé překračování jednotlivých

prahů detekce. Následující obrázek ukazuje průběh ze senzoru při chůzi směrem k senzoru ve vzdálenosti 8 m, která odpovídá přibližně šířce přechodu pro chodce na jednoproudé vozovce. Předposlední obrázek ukazuje situaci, kdy byla detekovaná osoba vzdálená od senzorové jednotky 15,5 m. Tato vzdálenost je již limitující, signál prahy detekce překračují již velice málo, přesto je možné osobu detekovat. Poslední obrázek ukazuje situaci, kdy se osoba pohybovala velmi rychlou chůzí k senzorové jednotce. Signál stejně jako v předešlém případě překračuje prahy detekce jen velice málo, ale stále bylo možné určit přítomnost cíle. Z frekvenční analýzy posledního signálu bylo zjištěno, že rychlost cíle již byla na mezi schopnosti rozpoznání objektu. U signálů, které byly na mezi detekce cíle, již není patrné pravidelné střídání překračování prahů detekce, což může mít za následek, že cíl nebude nedetekován.

Z naměřených dat je možné určit, že senzorová jednotka je schopna detekovat chodce až na vzdálenost 15 m. Bude-li uvažovaný přechod pro chodce široký 8 m (dva jízdní pruhy), bude chodec spolehlivě detekován (viz obrázek 8.6).

8.4 Měření senzorové jednotky detekce vozidel

Měření senzorové jednotky detekce pohybu vozidel probíhalo ve venkovních prostorách. Jednotka byla umístěna na konstrukci umožňující projíždění vozidla pod jednotkou ve výšce 2,5 m nad zemí. Sklon senzorové jednotky byl volen 10 ° nebo 20 °. Mikrovlnný senzor, tedy i celá jednotka, byl orientován horizontálně, svazek senzoru byl úzký. Mikrokontrolér senzorové jednotky pouze převáděl analogové vzorky a následně je posílal do počítače, kde byly ukládány a následně zpracovány.

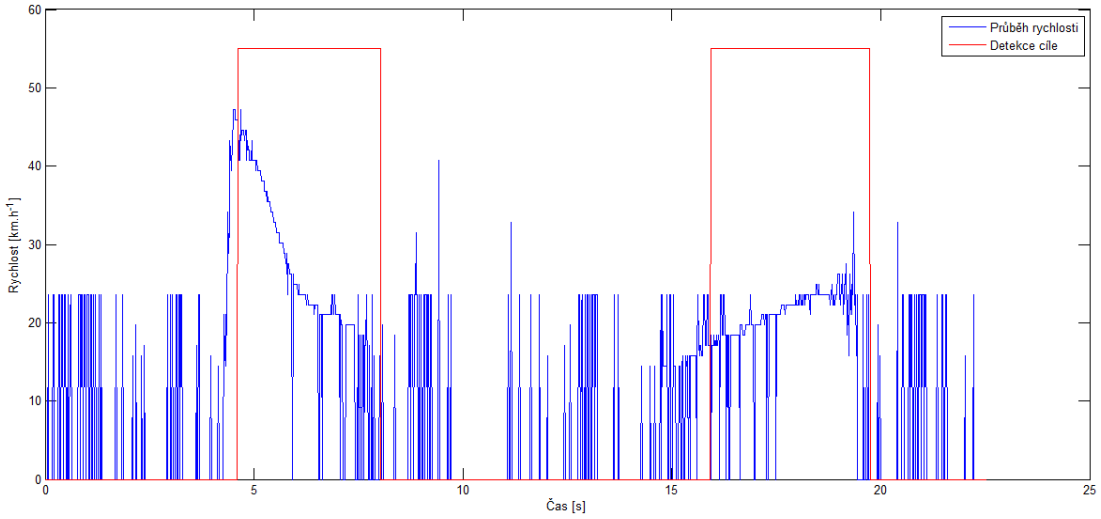
Situaci umístění jednotky vystihuje následující obrázek 8.9:

Obrázek 8.9 – Situace umístění senzorové jednotky detekce vozidel při měření

Bližší pohled na umístění senzorové jednotky detekce vozidel ukazuje následující obrázek:

Obrázek 8.10 – Bližší pohled na umístění senzorové jednotky

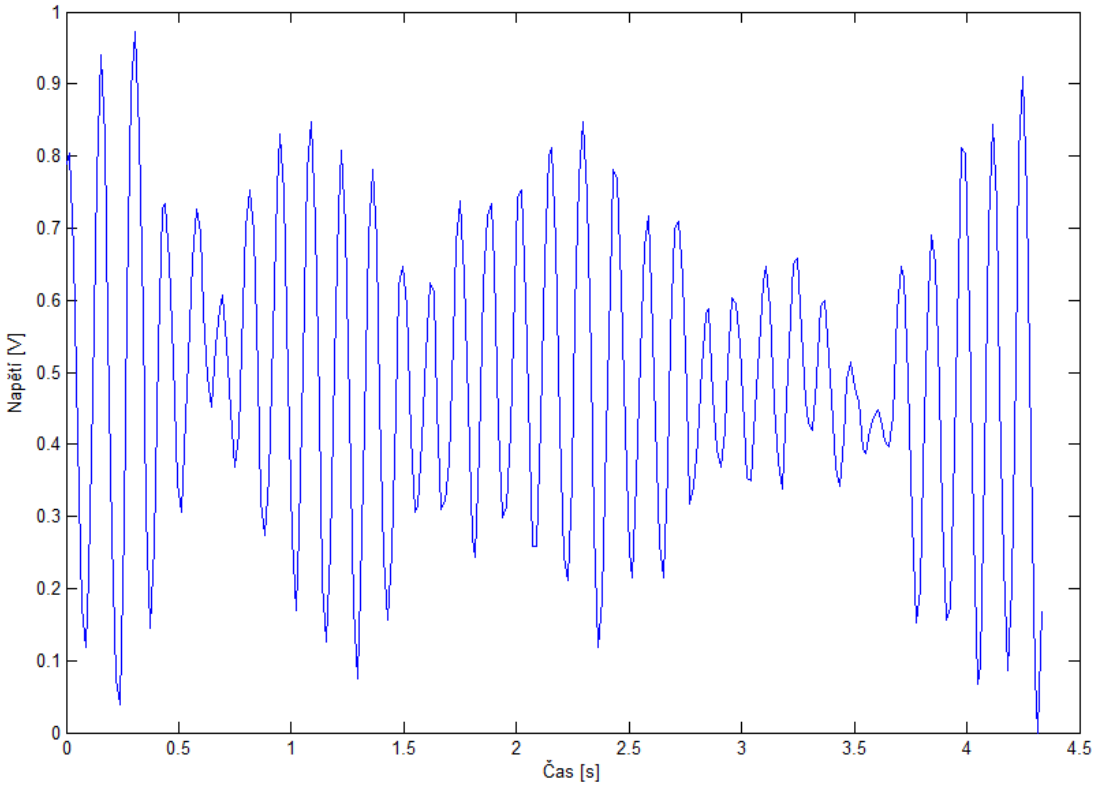
Naměřené vzorky signálu mikrovlnného senzoru byly v počítači rozděleny po 256 vzorcích (stejný počet, který je zpracováván mikrokontrolérem), následně byla vypočtena frekvenční spektra, vypočteny hodnoty prahů detekce (viz kapitola 7.5.1). S vypočtenými hodnotami prahů byla spektra vypočtena znovu, tentokrát byla ve spektrech hledána maxima a provádělo se porovnání s hodnotou daného prahu. Při překročení prahu byla z polohy maxima vypočtena rychlost pohybu objektu. Tímto způsobem obdržíme např. tento průběh rychlostí s vyznačenými okamžiky detekování vozidla podle algoritmu popsaným v 7.5.1:



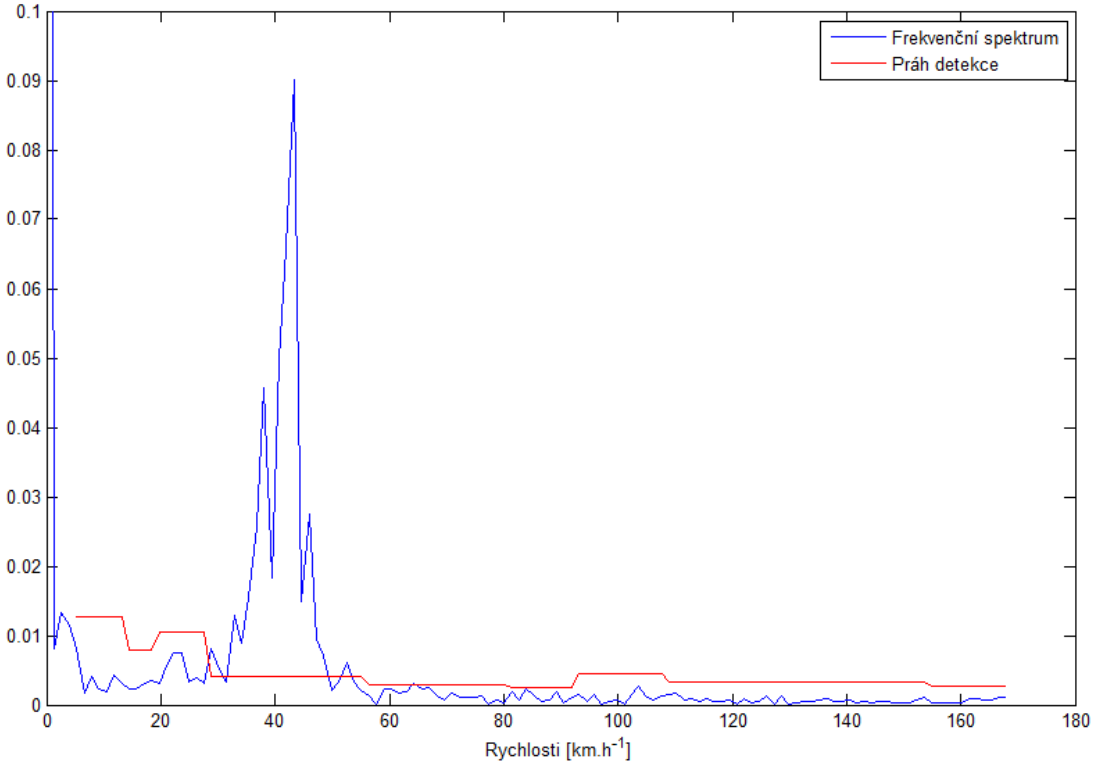
Obrázek 8.11 – Průběh naměřených rychlostí s vyznačenými okamžiky detekce vozidla s náklonem 20 °

Obrázek 8.11 ukazuje naměřené rychlosti, v průběhu rychlostí je vidět řadu falešných cílů, které se vyskytují nahodile a jen po velice krátkou dobu. V průběhu jsou také vidět jednolité průběhy představující přítomnost vozidla, tyto okamžiky jsou vyznačeny červenou barvou. Během tohoto konkrétního měření bylo vozidlo detekováno celkem dvakrát. První detekci představuje vozidlo odjíždějící od senzorové jednotky, byla zachycována zadní část vozidla, druhá detekce představuje přibližující se vozidlo k jednotce, byla zachycována přední část vozidla. V průběhu rychlostí je také patrné postupné snižování nebo zvyšování rychlosti podle změny úhlu, a tím změny radiální rychlosti, vozidla vůči senzorové jednotce.

Následující obrázky ukazují vybraný průběh signálu s jeho frekvenčním spektrem:

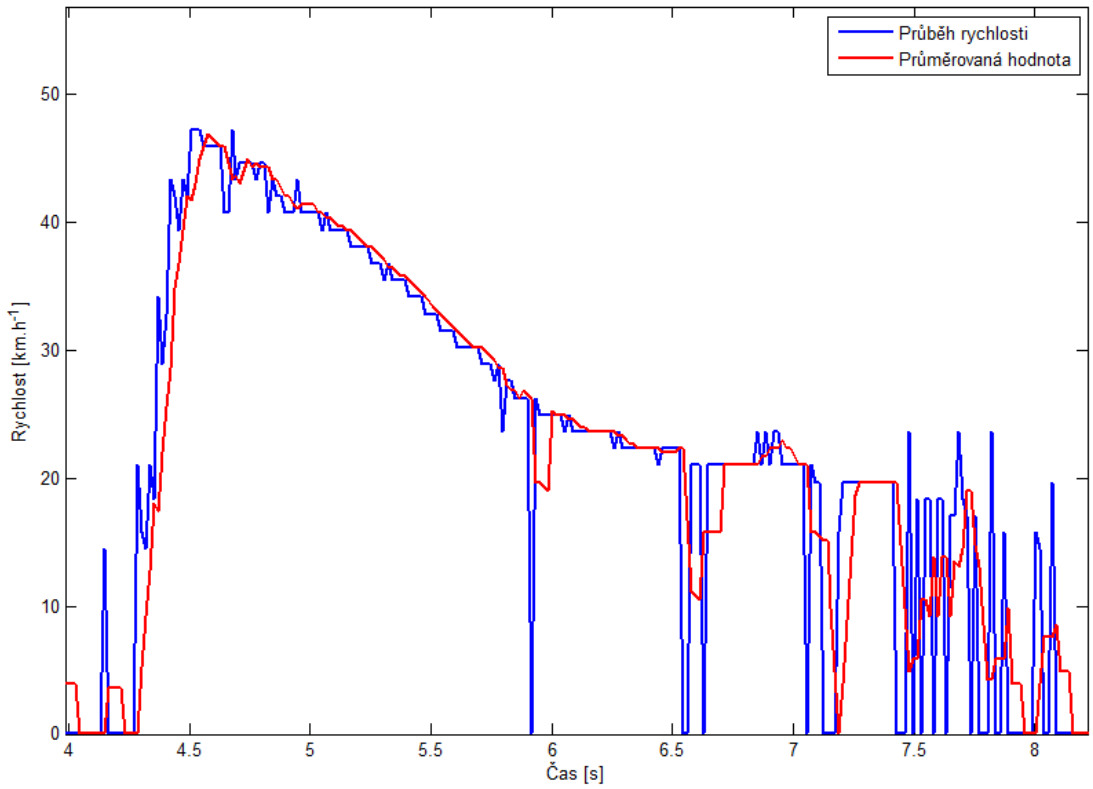


Obrázek 8.12 – Vybraný průběh signálu při detekci vozidla

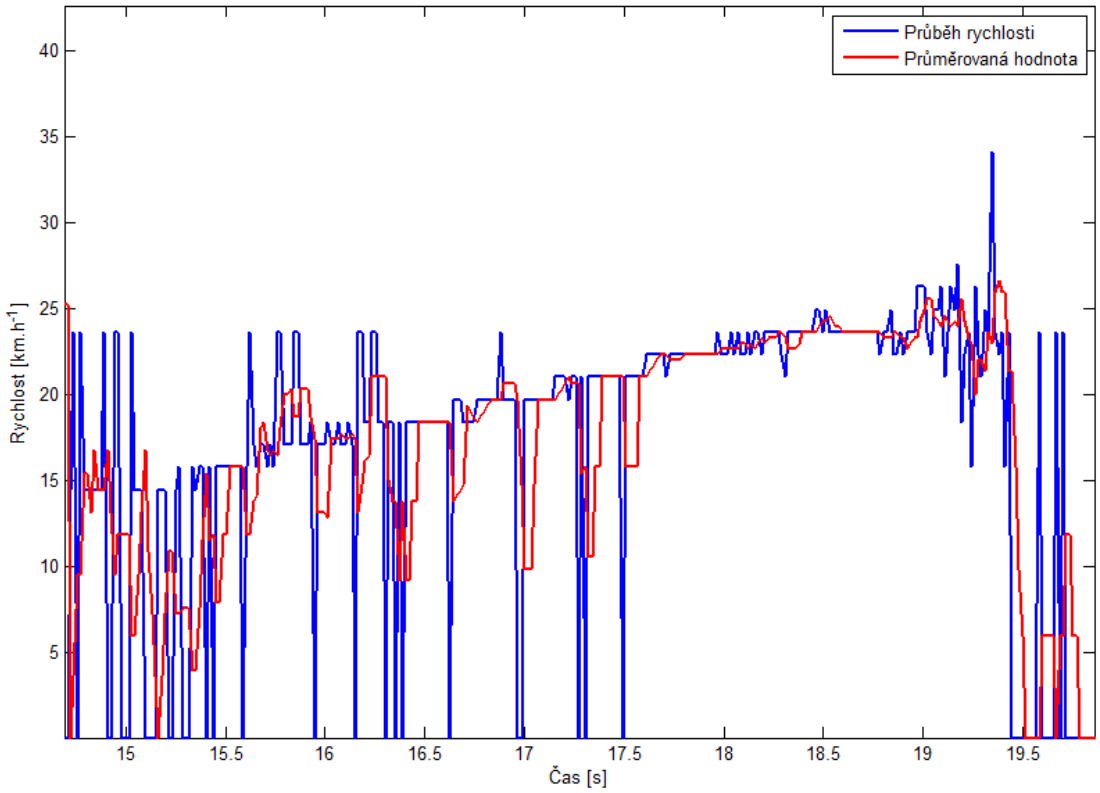


Obrázek 8.13 – Frekvenční spektrum vybraného signálu

Provedeme-li bližší pohled na detekování vozidla a provedeme-li průměrování rychlostí pomocí klouzavého průměru ze čtyř hodnot, obdržíme následující průběhy:



Obrázek 8.14 – Průběh naměřených a průměrovaných rychlostí – jízda od jednotky



Obrázek 8.15 – Průběh naměřených a průměrovaných rychlostí – jízda k jednotce

Detailní pohledy na okamžiky detekování cíle ukazují nutnost určitého zpracování, rychlosti nejsou jednoznačné. Jako vhodné zpracování se ukázalo průměrování vždy ze čtyř hodnot rychlostí. Toto průměrování se nazývá klouzavé. Ze známé rychlosti vozidla a z průměrovaných změřených rychlostí se ukázalo vhodné hledání maximální hodnoty průměrovaných rychlostí. Na základě naměřených rychlostí a jejich vyhodnocení byl navržen algoritmus popsaný v kapitole 7.5.1.

9 Závěr

V dnešní době existuje řada opatření a zařízení pro zvýšení bezpečnosti chodců na přechodech. Podle výše zabezpečení se rozdělují do různých generací. Všechna dnešní zabezpečení neuvažují přítomnost vozidel blížících se k přechodu, pouze upozorňují na přítomnost chodce, nebo napomáhají ke zviditelnění chodce na přechodu. Navrhovaný systém bere v úvahu i přítomnost vozidel. Na základě jejich přítomnosti a přítomnosti chodce zprostředkuje možnost zpomalení vozidla. Umožňuje také měření rychlosti projíždějícího vozidla a podle maximální přípustné rychlosti v oblasti, je možné vozidlo zpomalit nebo podle dalšího chování řidiče vozu i vozidlo zastavit.

Pro návrh senzorových jednotek detekce chodců nebo vozidel bylo nutné nastudovat řadu technických materiálů, např. dokumentaci k mikrokontroléru, operační systém IQRF modulů apod. Zároveň bylo nutné obvodově upravovat navrženou desku plošného spoje, aby se minimalizoval výstupní šum zesilovače na co nejmenší hodnotu. Při technickém návrhu a následné programové realizaci bylo nutné přihlédnout ke snížení proudové spotřeby jednotek. Pro správné vyhodnocení cíle (přítomnosti chodce nebo vozidla) bylo nutné určit patřičné prahy detekce a vyřešit náhodné překračování prahů díky přítomnosti šumu. Z důvodu, že se bezdrátová komunikace provádí v pásmu, pro které není nutná licence, je nutné datové zprávy omezit na minimum, aby nedocházelo k rušení jiných účastníků.

S navrženými senzorovými jednotkami byla provedena řada testování a měření. Při měření dosahu bezdrátové komunikace byla zjištěna možnost komunikace až do vzdálenosti 4,5 km, požadovaný dosah 300 m je tedy zaručen a je možné senzorové jednotky umístit i do větší vzdálenosti od centrální jednotky křižovatky. Při vzdálenosti 700 m byla naměřena chybovost datových zpráv pouhé 1 %. V případě senzorové jednotky detekce osob bylo nutné k podmínce překročení prahu detekce přidat další podmínku v podobě za sebou se střídajících obou typů prahů detekce (horní a dolní práh). Detekce chodce byla v laboratorních podmínkách měřena až na vzdálenost 15 m, tato vzdálenost je ale již hraniční. Pro senzorovou jednotku detekce vozidel bylo nutné spočítat řadu dílčích prahů detekce podle umístění šumu ve frekvenčním spektru. Pro určení rychlosti vozidla a tím i jeho detekci byla zjištěna

nutnost počítání překročení prahu detekce, pokud počet překročení prahu za sebou dosáhl určité hodnoty, bylo rozhodnuto o přítomnosti cíle, dále bylo nutné získané rychlosti filtrovat pomocí klouzavého průměru. Ve filtrovaných hodnotách bylo hledáno maximum, které bylo po ztrátě detekce vozidla odesláno do centrální jednotky křižovatky.

Všechny jednotky byly testovány pouze v laboratorních podmínkách. Jednotka detekce vozidel byla sice měřena ve venkovních prostorách, ale nebyla testována za plného provozu na křižovatce. Při reálném provozu by mohl u jednotky detekce vozidel nastat problém s více jedoucími vozidly za sebou, tato vozidla by se mohla jevit jako jediné vozidlo. Naopak celý navržený systém přispívá ke snížení četnosti srážek chodců s vozidly.

10 Seznam použité literatury

1. *www.bezpecneprechody.cz* [online]. 2011 [cit. 2011-08-07]. Bezpečné přechody pro chodce. Dostupné z WWW: <http://www.bezpecneprechody.cz/>.
2. SKOLNIK, Merril I. *Spravočnik po radiolokacii*. Moskva : Sovetskoe radio, 1976. 527 s.
3. POJMON, Leoš. *Zpracování signálů soustavy dopplerovských senzorů*. Pardubice, 2000. 100 s. Diplomová práce. Univerzita Pardubice, Dopravní fakulta Jana Pernera.
4. KREJČIŘÍK, Alexandr. *Napájecí zdroje I.*. Praha : Nakladatelství BEN - technická literatura, 1996. 351 s. ISBN 80-86056-02-3.
5. *www.linear.com* [online]. 2011 [cit. 2011-08-06]. LTC3538 - 800mA Synchronous Buck-Boost DC/DC Converter. Dostupné z WWW: < http://cds.linear.com/docs/Datasheet/3538fb.pdf >.
6. *www.analog.com* [online]. 2010 [cit. 2011-08-06]. AD8646: 24MHz Rail-to-Rail Quad Amplifier. Dostupné z WWW: <http://www.analog.com/en/all-operational-amplifiers-op-amps/operational-amplifiers-op-amps/ad8648/products/product.html>.
7. *www.iqrf.org* [online]. 2011 [cit. 2011-08-06]. IQRF - Simple way to smarter wireless solutions. Dostupné z WWW: <http://www.iqrf.org/weben/index.php>.
8. *www.iqrf.org* [online]. 2011 [cit. 2011-08-06]. TR-53B. Dostupné z WWW: <http://www.iqrf.org/weben/index.php?sekce=products&id=tr-53b&ot=transceivers&ot2=tr-53b>.
9. *www.iqrf.org* [online]. 2011 [cit. 2011-08-06]. TR-53B Transceiver Module. Dostupné z WWW: <http://www.iqrf.org/weben/downloads.php?id=163>.
10. *www.iqrf.org* [online]. 2011 [cit. 2011-08-06]. IQRF Operating system. Dostupné z WWW: <http://www.iqrf.org/weben/index.php?sekce=support&id=os>.
11. *www.iqrf.org* [online]. 2010 [cit. 2011-08-06]. SPI Implementation in IQRF TR modules. Dostupné z WWW:

<http://www.iqrf.org/weben/edit_support/downloads.php?s=MNSPI_101223.pdf>.

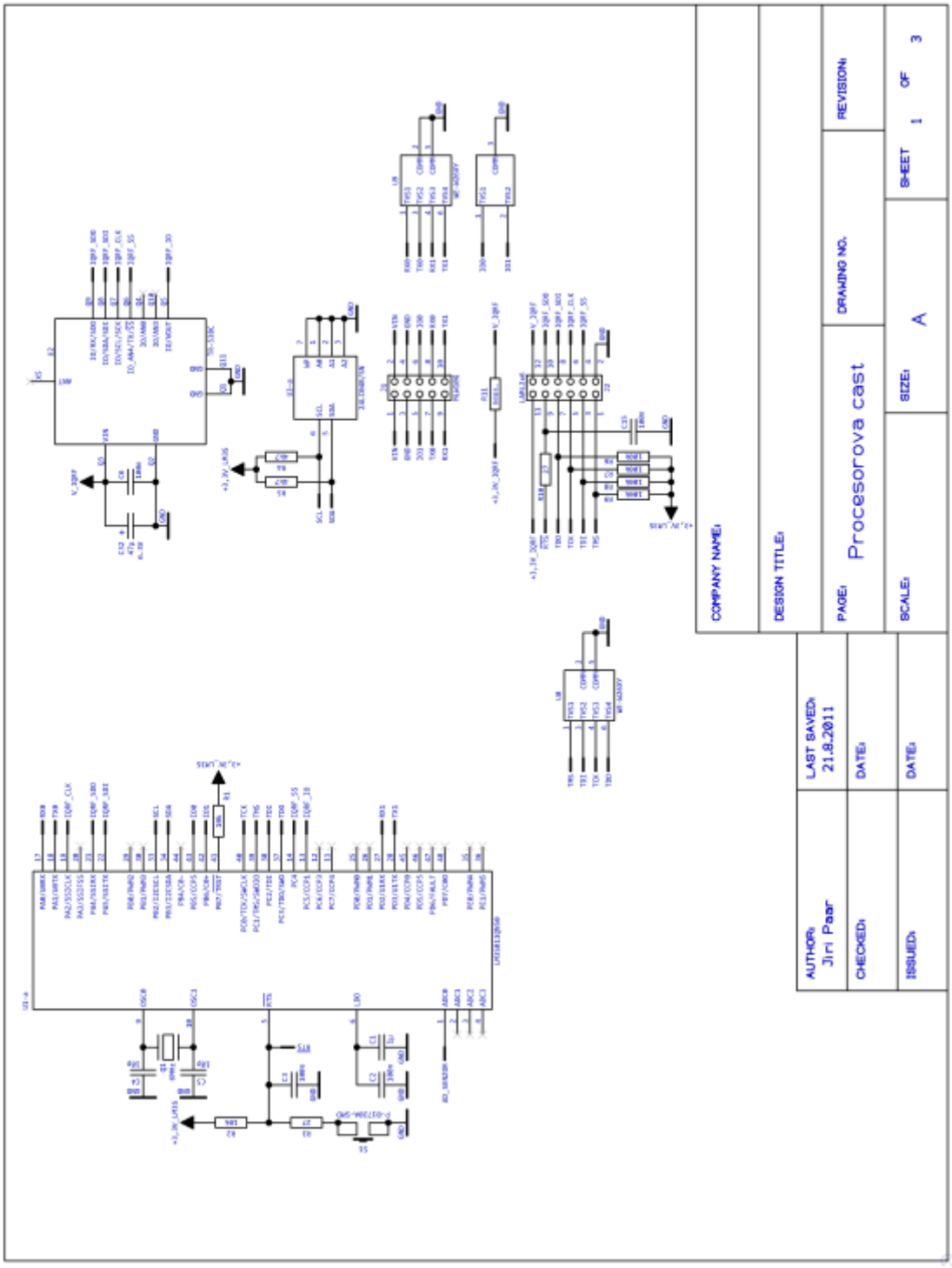
12. *www.iqrf.org* [online]. 2011 [cit. 2011-08-06]. CK-USB-02 - IQRF programmer and debugging kit. Dostupné z WWW: <http://www.iqrf.org/weben/index.php?sekce=products&id=ck-usb-02&ot=development-tools&ot2=development-kits>.
13. *www.iqrf.org* [online]. 2011 [cit. 2011-08-06]. CK-USB-04 - IQRF programmer and debugging kit. Dostupné z WWW: <http://www.iqrf.org/weben/index.php?sekce=products&id=ck-usb-04&ot=development-tools&ot2=development-kits>.
14. *www.iqmesh.org* [online]. 2011 [cit. 2011-08-06]. Detailed IQMESH protocol description and its specifications are expected end of 2011. Dostupné z WWW: <http://www.iqmesh.org/iqmesh/>.
15. *www.ti.com* [online]. 2011-02-08 [cit. 2011-08-07]. Stellaris LM3S811 Microcontroller. Dostupné z WWW: <http://focus.ti.com/docs/prod/folders/print/lm3s811.html>.
16. *www.ti.com* [online]. 2010-07-16 [cit. 2011-08-07]. Stellaris LM3S811 RevC2 Errata. Dostupné z WWW: <http://focus.ti.com/lit/er/spmz090f/spmz090f.pdf>.
17. *www.microchip.com* [online]. 2009 [cit. 2011-08-07]. 24AA04/24LC04B. Dostupné z WWW: <http://ww1.microchip.com/downloads/en/DeviceDoc/21708K.pdf>.
18. *Pandatron.cz* [online]. 2010-02-10 [cit. 2011-08-07]. Úvod do architektury Cortex-M3 - díl. 1. Dostupné z WWW: <http://pandatron.cz/?1252&uvod_do_architektury_cortex-m3_-_dil._1>.
19. *Pandatron.cz* [online]. 2010-02-26 [cit. 2011-08-07]. Úvod do architektury Cortex-M3 - díl. 2. Dostupné z WWW: <http://pandatron.cz/?1281&uvod_do_architektury_cortex-m3_-_dil._2>.
20. *www.linear.com* [online]. 2007 [cit. 2011-08-11]. Quick Start Guide for Demonstration Circuit 1095. Dostupné z WWW: <http://cds.linear.com/docs/Demo%20Board%20Manual/dc1095A.pdf>.
21. BARTON, David K. *Radar System Analysis*. New Jersey : Prentice-Hall, 1965. xv, 608s.

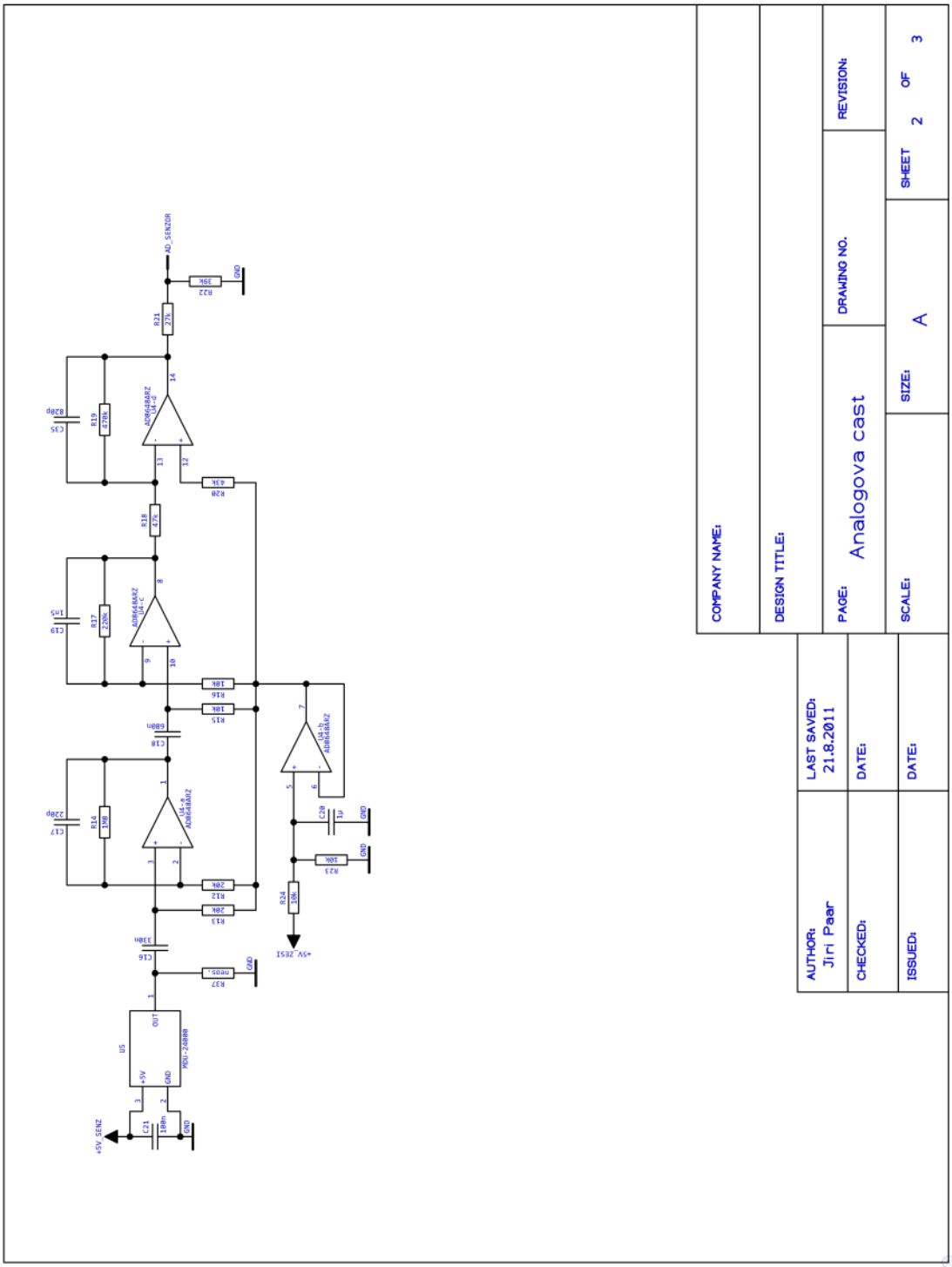
22. *www.ti.com* [online]. 1998 [cit. 2011-08-14]. Implementing the Radix-4 Decimation in Frequency (DIF) Fast Fourier Transform (FFT) Algorithm Using a TMS320C80 DSP. Dostupné z WWW: <http://www.ti.com/lit/an/spra152/spra152.pdf>.
23. *www.cmlab.csie.ntu.edu.tw* [online]. 2005 [cit. 2011-08-14]. Fast Fourier Transform (FFT). Dostupné z WWW: <http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/transform/fft.html>.
24. *www.rcd.cz* [online]. 2010 [cit. 2011-08-20]. Směrová anténa BD910 BD910A. Dostupné z WWW: <http://www.rcd.cz/cz/pdf/bd910-cz.pdf>.
25. *www.microwave-solutions.com* [online]. 2011 [cit. 2011-08-21]. K-Band Doppler Motion Detector Units Model Numbers MDU2400/2410. Dostupné z WWW: <http://docs.microwave-solutions.com/createPdf.php?id=MDU2400>.
26. *www.we-online.com* [online]. 2005-12-16 [cit. 2011-08-21]. Spezifikation für Freigabe / specification for release. Dostupné z WWW: <http://katalog.we-online.de/kataloge/eisos/media/pdf/74279218.pdf>.
27. EICHLER, CSc., Prof. Ing. Josef; ŽALUD, CSc., Ing. Václav. *Radioelektronická zařízení I*. Praha : Vydavatelství ČVUT, 1977. 340 s.
28. *www.arm.com* [online]. 2011 [cit. 2011-08-27]. CMSIS - Cortex Microcontroller Software Interface Standard. Dostupné z WWW: <http://www.arm.com/products/processors/cortex-m/cortex-microcontroller-software-interface-standard.php>.

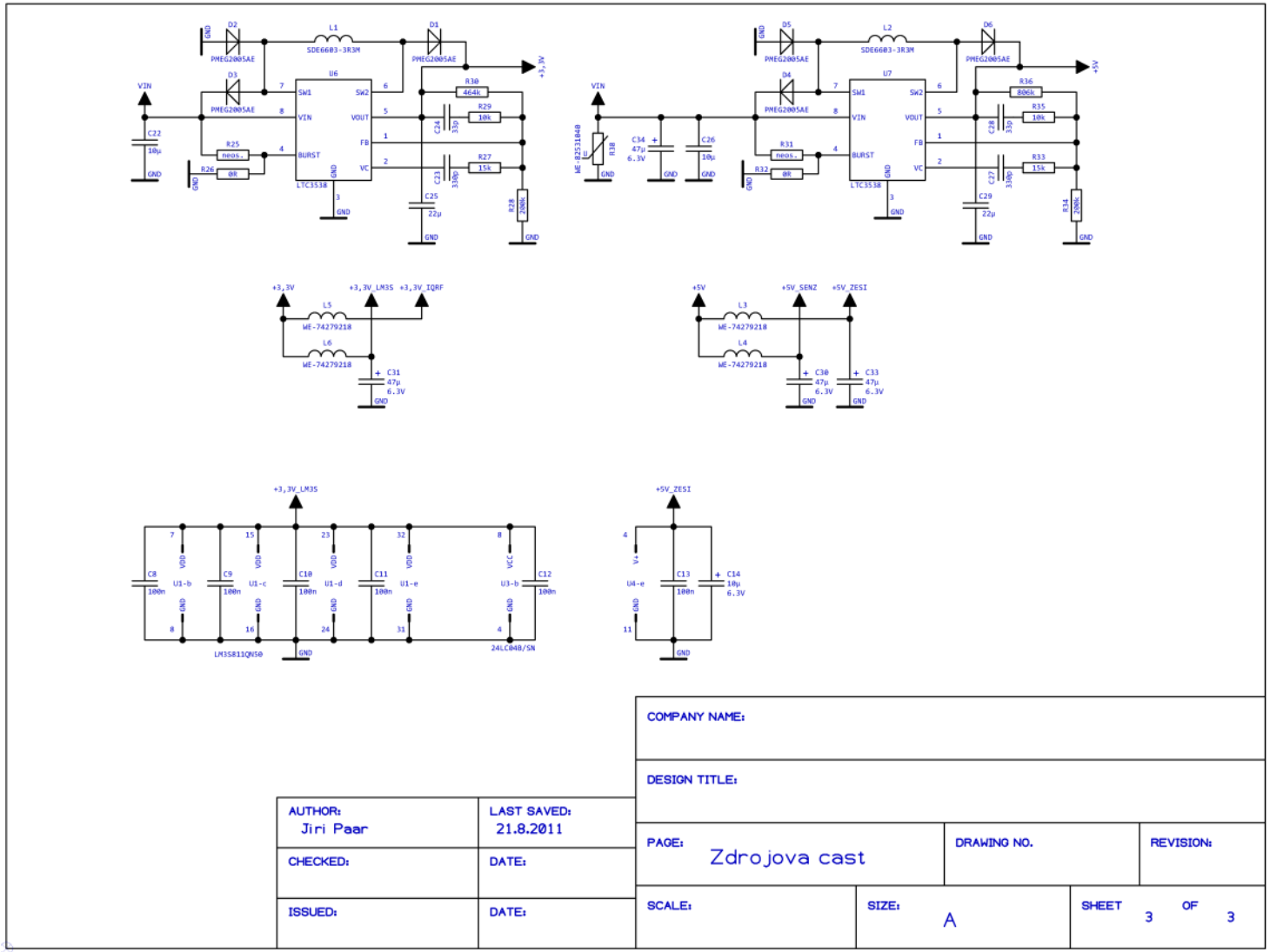
Příloha

Příloha A. Schéma senzorové jednotky detekce osob

Příloha A obsahuje kompletní schéma zapojení senzorové jednotky detekce osob.

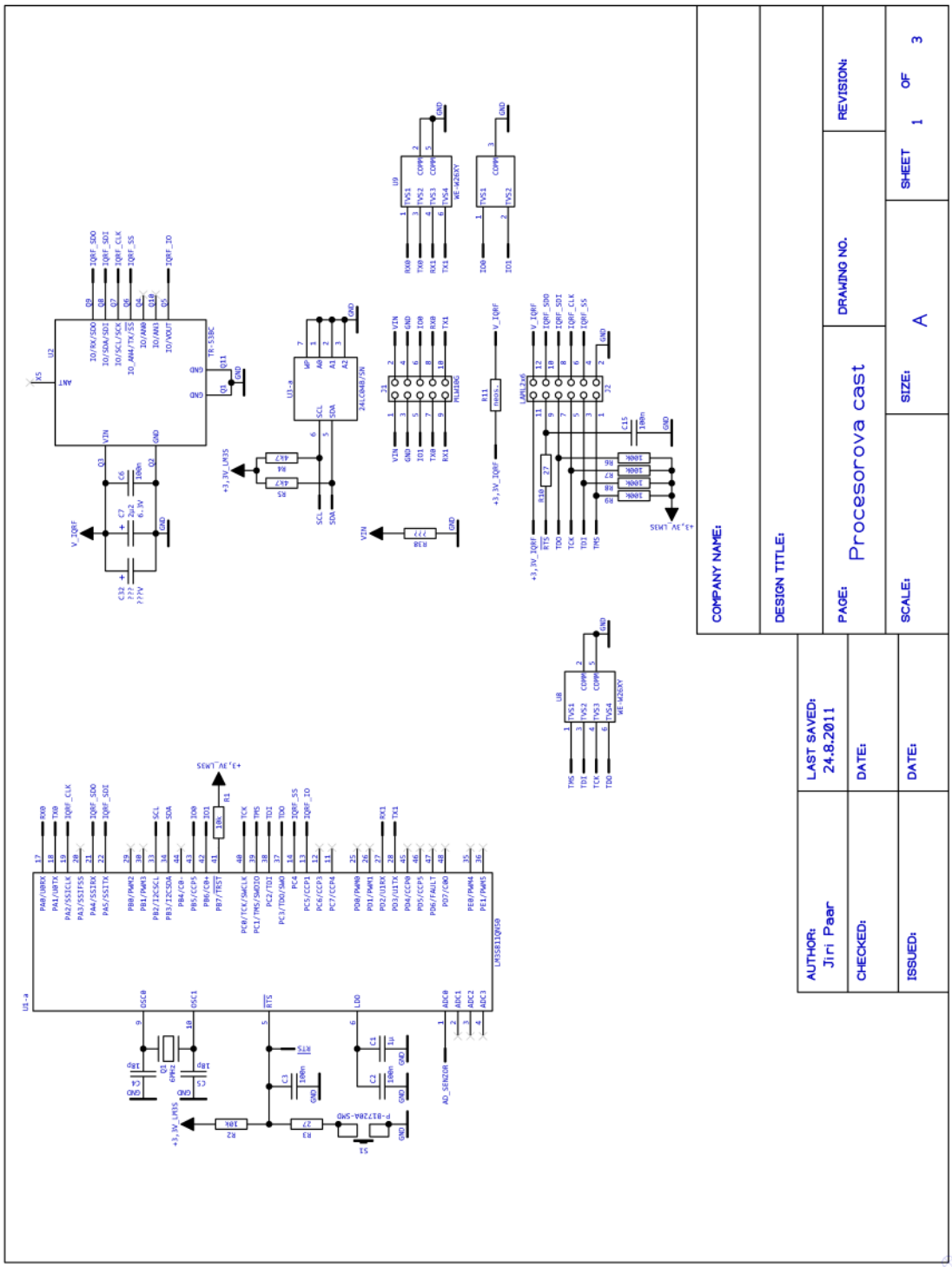

COMPANY NAME:
DESIGN TITLE:
PAGE: Procesorova cast
DRAWING NO.
REVISION:
SIZE: A
SHEET 1 OF 3
SCALE:
AUTHOR: Jiri Pasr
LAST SAVED: 21.8.2011
CHECKED:
DATE:
ISSUED:
DATE:

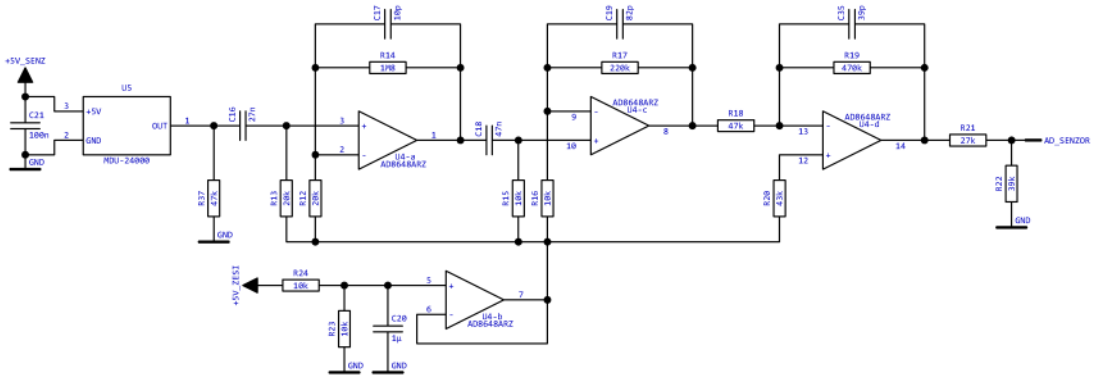

COMPANY NAME:
DESIGN TITLE:
PAGE: Analogova cast
DRAWING NO.
REVISION:
SIZE: A
SHEET 2 OF 3
SCALE:
AUTHOR: Jiri Paar
LAST SAVED: 21.8.2011
CHECKED:
DATE:
ISSUED:
DATE:
AD_SENZOR
+5V_SENZ
+9V_ZESI
GND
OUT
+5V
MDU-24000


+3,3V_LM3S
+5V_ZESI
LM3S811QN50
24LC04B/SN
COMPANY NAME:
DESIGN TITLE:
AUTHOR: Jiri Paar
LAST SAVED: 21.8.2011
CHECKED:
DATE:
ISSUED:
DATE:
PAGE: Zdrojova cast
DRAWING NO.
REVISION:
SCALE:
SIZE: A
SHEET 3 OF 3

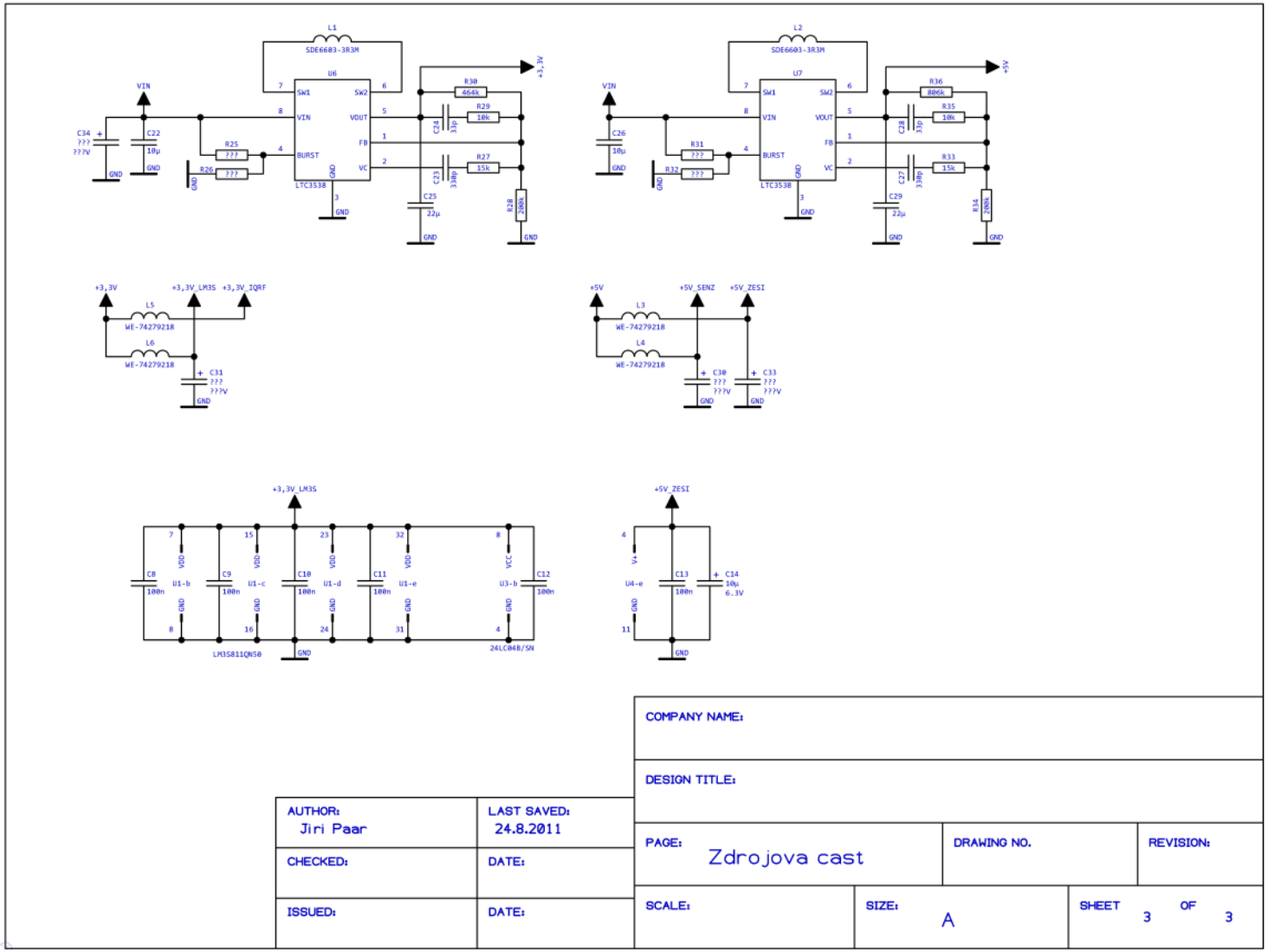
Příloha B. Schéma senzorové jednotky detekce vozidel

Příloha B obsahuje kompletní schéma zapojení senzorové jednotky detekce vozidel.





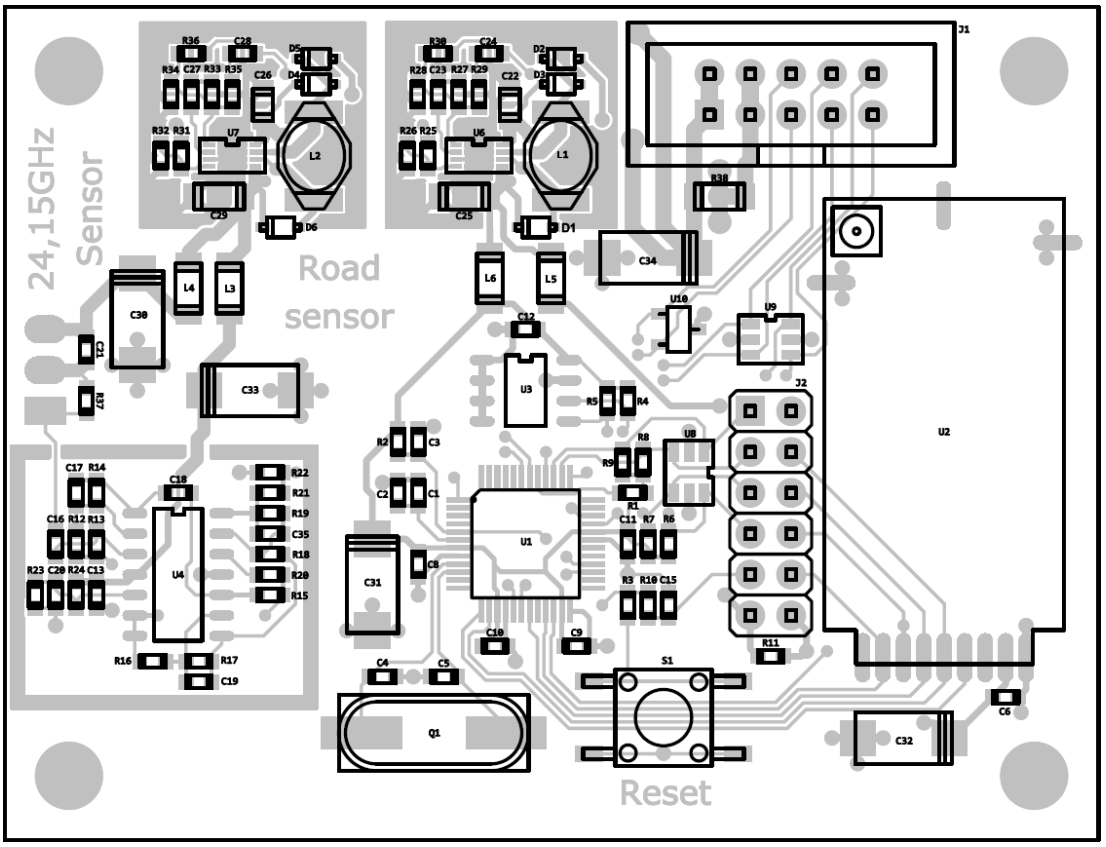
COMPANY NAME:				
DESIGN TITLE:				
AUTHOR: Jiri Paar	LAST SAVED: 24.8.2011	PAGE: Analogova cast	DRAWING NO.	REVISION:
CHECKED:	DATE:			
ISSUED:	DATE:	SCALE:	SIZE: A	SHEET 2 OF 3

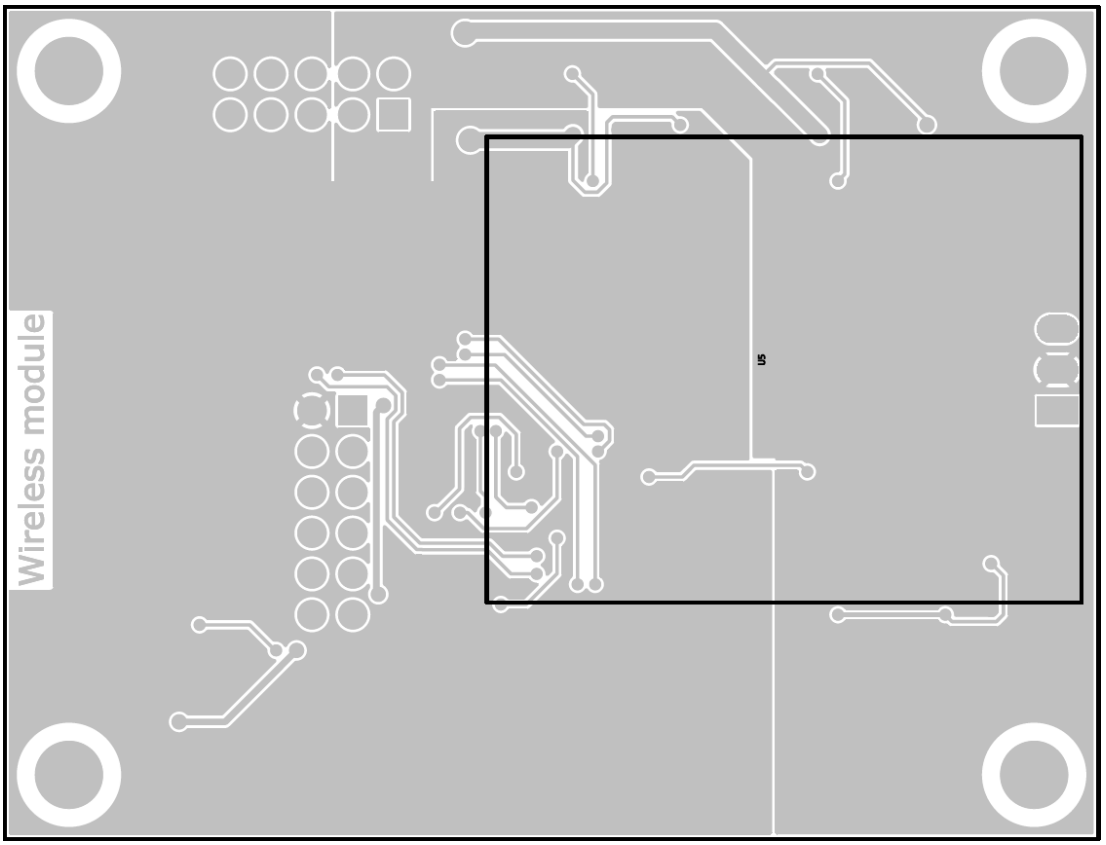


COMPANY NAME:	
DESIGN TITLE:	
AUTHOR: Jiri Paar	LAST SAVED: 24.8.2011
CHECKED:	DATE:
ISSUED:	DATE:
PAGE: Zdrojova cast	DRAWING NO. / REVISION:
SCALE:	SIZE: A / SHEET 3 OF 3

Příloha C. Návrh desky plošného spoje

Příloha C obsahuje obrázky návrhu desky plošného spoje senzorových jednotek. První obrázek ukazuje osazení strany součástek, druhý osazení strany spojů.




Wireless module

Příloha D. Zdrojové kódy

Příloha D obsahuje jednotlivé zdrojové kódy pro IQRF modul i pro mikrokontrolér LM3S811 pro oba typy senzorových jednotek.

Zdrojový kód IQRF modulu se nachází v souboru **IQRF_main.c**.

IQRF_main.c

```
#include "includes/template-basic.h"

unsigned char GetCRC(); // Funkce pro výpočet kontrolního součtu

void APPLICATION()
{
    // Pomocné proměnné
    unsigned char delayWaitTX, delayWaitACK, address, checkRFvalue, idNetwork, enTX,
                                packetID = 0, crc, crcRF, destAddr, temp;

    PORTC.2 = 0; // Vývod Q5 na hodnotu 0
    TRISC.2 = 0; // Vývod Q5 jako výstup
    enableSPI(); // Povolení systémové SPI sběrnice

    while (1) // Hlavní smyčka programu
    {
    main2:
        clrwdt(); // Reset Watchdog časovače
        if (getStatusSPI() == 0) // SPI sběrnice není zaneprázdněná
        {
            if (_SPIRX) // Jsou přijata data po SPI sběrnici
            {
                if (_SPICRCok == 0) // Poslední kontrolní součet nebyl přijat správně
                {
                    startSPI(0); // Reset SPI
                    goto main2; // Na začátek programu
                }
                else
                {
                    param1--; // Počet přijatých bajtů o 1 méně, řídící znak
                    // Je přijato nastavení IQRF modulu
                    if (bufferCOM[0] == 0x00 && param1 == 7)
                    {
                        // Nastavení vysílacího výkonu
                        if (bufferCOM[1] > 7)
                            setTXpower(7); // Přijatá hodnota je vyšší než povolená
                        else
                            setTXpower(bufferCOM[1]); // Nastavení přijaté hodnoty

                        // Nastavení vysílacího/přijímacího kanálu
                        if (bufferCOM[2] > 61)
                            setRFchannel(61); // Přijatá hodnota je vyšší než povolená
                        else
                            setRFchannel(bufferCOM[2]); // Nastavení přijaté hodnoty

                        address = bufferCOM[3]; // Uložení adresy IQRF modulu
                        delayWaitTX = bufferCOM[4]; // Uložení doby čekání na uvolnění kanálu
                        delayWaitACK = bufferCOM[5]; // Uložení doby čekání na potvrzení

                        // Nastavení míry zarušení
                        if (bufferCOM[6] > 64)
                            checkRFvalue = 64; // Přijatá hodnota je vyšší než povolená
                        else
                            checkRFvalue = bufferCOM[6]; // Uložení přijaté hodnoty

                        idNetwork = bufferCOM[7]; // Uložení identifikátoru sítě
```

```
                    bufferCOM[0] = 0; // Uložení správného potvrzení
                }
                // Je požedavek na odeslání a jsou k odeslání nějaká data
                else if (bufferCOM[0] == 0x01 && param1 > 1)
                {
                    PORTC.2 = 1; // Výstup Q5 = 1 - budou se odesílat data
                    destAddr = bufferCOM[1]; // Uložení adresy cílového zařízení
                    // Naplnění hlavičky datové zprávy
                    bufferRF[0] = 0x53;
                    bufferRF[1] = packetID & 0x7F;
                    bufferRF[2] = idNetwork;
                    bufferRF[3] = destAddr;
                    bufferRF[4] = address;
                    // Zkopírování dat z bufferCOM do bufferRF
                    copyMemoryBlock(bufferCOM+2, bufferRF+5, param1-1);
                    DLEN = param1+5; // Celkový počet dat k odeslání
                    crc = GetCRC(); // Výpočet kontrolního součtu
                    writeToRAM(bufferRF+DLEN-1,crc); // Uložení CRC do bufferRF
                    packetID++; // Zvýšení hodnoty identifikátoru paketu

                    startDelay(delayWaitTX); // Nastavení časovače na uvolnění kanálu
                    enTX = 0; // Vysílání není povoleno
                    do
                    {
                        // Kanál není zarušen - nikdo nevysílá
                        if (checkRF(checkRFvalue) == 0)
                        {
                            enTX = 1; // Povolení vysílání
                            break; // Konec čekání
                        }
                    } while(isDelay()); // Čekej na vynulování časovače
                    // Kanál byl zarušen
                    if (enTX == 0)
                    {
                        bufferCOM[0] = 0xFF; // Uložení špatného potvrzení
                        goto SendSPI; // Skok na odeslání po SPI
                    }
                    PIN = 0; // Nesíťový paket
                    RFTXpacket(); // Odeslání dat

                    toutRF = delayWaitACK; // Nastavení timeoutu pro příjem dat
                    startDelay(delayWaitACK); // Spuštění časovače pro potvrzení
                    do
                    {
                        if (RFRXpacket()) // Je přijat nějaký datový paket
                        {
                            temp = packetID-1; // Výpočet odeslaného id. paketu
                            crc = GetCRC(); // Výpočet přijatého CRC
                            crcRF = readFromRAM(bufferRF+DLEN-1); // Načtení přijatého
                                                                  // CRC
                            temp |= 0x80; // Výpočet id. paketu pro potvrzeovací paket

                            // Kontrola správnosti přijatého paketu, adresy, kontrolní
                                                          // součty apod.
                            if (bufferRF[0] == 0x53 && bufferRF[1] == temp &&
bufferRF[2] == idNetwork && bufferRF[3] == address && bufferRF[4] == destAddr &&
      crcRF == crc)
                            {
                                bufferCOM[0] = 0; // Paket byl potvrzen, uložení
                                                  // správného potvrzení
                            }
                            else
                                bufferCOM[0] = 0xFF; // Paket nebyl potvrzen, uložení
                                                  // nesprávného potvrzení
                        }
                        else
                            bufferCOM[0] = 0xFF; // Paket nebyl přijat, uložení
                                                 // nesprávného potvrzení
                    } while(isDelay()); // Čekej celou dobu na potvrzení
                }
                // Požadavek na přechod do režimu SLEEP
                else if (bufferCOM[0] == 0xFF)
                {
                    startSPI(0); // Reset SPI
```

```
                    waitDelay(5); // Čekej 5ms - klid na SPI sběrnici
                    GIE = 0; // Zákaz globálního přerušení
                    RBIE = 1; // Povolení přerušení PORTB
                    SWDTEN = 0; // Zákaz Watchdog
                    iqrfSleep(); // Přechod do SLEEP
                    RBIF = 0; // Vynulování přerušení od PORTB
                    RBIE = 0; // Zákaz přerušení PORTB
                    GIE = 1; // Povolení globálního přerušení
                    clrwdt(); // Reset Watchdog
                    SWDTEN = 1; // Povolení Watchdog
                    bufferCOM[0] = 0; // Uložení správného potvrzení
                }
                else // Chybný požadavek
                    bufferCOM[0] = 0xFF; // Uložení chybného potvrzení

                SendSPI: // Odeslání potvrzení po SPI
                    PORTC.2 = 0; // Vynulování výstupu Q5 - již se neodesílají žádná data
                    startSPI(1); // Odeslání potvrzení
            }
        }
    }
}

// Výpočet CRC pro data v bufferRF o délce DLEN-1
unsigned char GetCRC()
{
    unsigned char i, crc = 0; // Pomocné proměnné

    for (i = 0; i < DLEN-1;i++) // Přes všechna data kromě posledního (CRC)
        crc ^= readFromRAM(bufferRF+i); // Započtení hodnoty v bufferRF

    return crc ^ 0x12; // Dopočítání CRC a návrat
}
```

Součástí každého kódu senzorové jednotky je hlavičkový soubor **board.h** obsahující definice jednotlivých vývodů mikrokontroléru, makra pro inicializaci portů a makra pro manipulaci s jednotlivými vývody.

board.h

```
#ifndef BOARD_H
#define BOARD_H

// Definice vývodů mikrokontroléru, inicializace portů a manipulace s vývody

//Piny PORTB
#define PIN_IO0 5
#define PIN_IO1 6

// Piny PORTC
#define PIN_IQRF_CS 3
#define PIN_IQRF_IO 5

// Inicializace PORTA
// Vývody U0RX, U0TX, SSICLK, SSIRX a SSITX jsou přižazeny ke své periferii
#define InitPortA() (GPIO_PORTA_AFSEL_R = (1 << 0) | (1 << 1) | (1 << 2) | (1 << 4) |
                                          (1 << 5))

// Inicializace PORTB
// Vývody IO0 a IO1 jako výstupy
// Vývody IO0 a IO1 jsou digitální
// Vývody I2CSCL a I2CSDA jsou pro I2C sběrnici
// Výstupy do hodnoty 0
#define InitPortB() GPIO_PORTB_DIR_R = (1 << PIN_IO0) | (1 << PIN_IO1), \
    GPIO_PORTB_DEN_R |= (1 << PIN_IO0) | (1 << PIN_IO1), \
        GPIO_PORTB_AFSEL_R = (1 << 2) | (1 << 3), \
             GPIO_PORTB_DATA_R = 0
```

```
// Inicializace PORTC
// Vývod IQRF_CS jsou výstup
// Vývody IQRF_CS a IQRF_IO jsou výstupy
// Vývody IQRF_CS a IQRF_IO jsou obecné vstupně-výstupní vývody
// Výstup IQRF_CS do 1, ostatní jsou 0
#define InitPortC() GPIO_PORTC_DIR_R |= (1 << PIN_IQRF_CS) | (1 << PIN_IQRF_IO), \
    GPIO_PORTC_DEN_R |= (1 << PIN_IQRF_CS) | (1 << PIN_IQRF_IO), \
        GPIO_PORTC_AFSEL_R &= ~((1 << PIN_IQRF_CS) | (1 << PIN_IQRF_IO)), \
            GPIO_PORTC_DATA_R = (1 << PIN_IQRF_CS)

// Manipulace s vývodem IQRF_CS
#define ClrIQRFCs() GPIO_PORTC_DATA_R &= ~(1 << PIN_IQRF_CS) // Vynulování
#define SetIQRFCs() GPIO_PORTC_DATA_R |= (1 << PIN_IQRF_CS) // Nastavení
#define ToggleIQRFCs() GPIO_PORTC_DATA_R ^= (1 << PIN_IQRF_CS) // Změna

// Manipulace s vývodem IO0
#define ClrI0() GPIO_PORTB_DATA_R &= ~(1 << PIN_IO0) // Vynulování
#define SetI0() GPIO_PORTB_DATA_R |= (1 << PIN_IO0) // Nastavení
#define ToggleI0() GPIO_PORTB_DATA_R ^= (1 << PIN_IO0) // Změna

// Manipulace s vývodem IO1
#define ClrI1() GPIO_PORTB_DATA_R &= ~(1 << PIN_IO1) // Vynulování
#define SetI1() GPIO_PORTB_DATA_R |= (1 << PIN_IO1) // Nastavení
#define ToggleI1() GPIO_PORTB_DATA_R ^= (1 << PIN_IO1) // Změna

// Zjištění hodnoty vývodu IQRF_IO
#define IsIQRFIo() (GPIO_PORTC_DATA_R & (1 << PIN_IQRF_IO))

#endif
```

Dalším společným zdrojovým souborem senzorové jednotky je zdrojový kód pro manipulaci s I^2C sběrnicí uložený v souborech **i2c.h** a **i2c.c**. Pro různé typy senzorových jednotek je nutné nastavit různé hodinové kmitočty sběrnice (v kódu jsou opatřeny komentáři).

i2c.h

```
#ifndef I2C_H
#define I2C_H

#define ADDR_EEPROM 0xA0 // Adresa zařízení SLAVE (EEPROM)
#define I2C_READ 1 // Nejnižší bit adresy pro čtení
#define I2C_WRITE 0 // Nejnižší bit adresy pro zápis

extern void i2c_init(void); // Funkce provede inicializaci periferie I2C

// Zahájení zápisu do paměti EEPROM
// Parametry - addrSlave - adresa zařízení
//           - data - adresa zápisu do EEPROM (za adresou)
//           - stop - určuje jestli za daty bude odeslána událost STOP
extern void i2c_startWrite(unsigned char addrSlave, unsigned char data, unsigned char stop);

// Zahájení čtení z paměti EEPROM
// Parametry - addrSlave - adresa zařízení
//           - data - adresa, na kterou se uloží přijatá data za adresou
//           - ack - = 1 poslední data budou potvrzena, = 0 data nebudou potvrzena a bude
              // odesláno STOP
extern void i2c_startRead(unsigned char addrSlave, unsigned char *data, unsigned char ack);

// Čtení dalších dat z EEPROM
// Parametry - data - adresa počátku zásobníku, do kterého se budou ukládat data
//           - len - počet přečtených dat
//           - stop - určuje jestli za daty bude odeslána událost STOP nebo ACK
extern void i2c_getData(unsigned char *data, unsigned char len, unsigned char stop);

#endif
```

i2c.c

```
#include "inc/lm3s811.h" // Definice registrů a jejich bitů mikrokontroléru
#include "board.h" // Definice zapojení vývodů na desce plošného spoje a manipulace s nimi
#include "i2c.h" // Hlavičkový soubor pro I2C

// Inicializace I2C sběrnice
void i2c_init()
{
	I2C0_MASTER_MCR_R = I2C_MCR_MFE; // Povolení I2C v režimu MASTER
	I2C0_MASTER_MTPR_R = 1; // Nastavení frekvence =312,5kHz - detektor osob, =150kHz -
				//detektor vozidel
}

// Začátek pro zápis dat po I2C
void i2c_startWrite(unsigned char addrSlave, unsigned char data, unsigned char stop)
{
	I2C0_MASTER_MSA_R = addrSlave + I2C_WRITE; // Nastavení adresy SLAVE
	I2C0_MASTER_MDR_R = data; // Uložení dat následujících po adrese
	if (stop == 0)
		I2C0_MASTER_MCS_R = I2C_MCS_START | I2C_MCS_RUN; // Odeslíná adresy + data
	else
		I2C0_MASTER_MCS_R = I2C_MCS_START | I2C_MCS_RUN | I2C_MCS_STOP; // Odeslání
							// adresy + data + STOP

	while ((I2C0_MASTER_MCS_R & I2C_MCS_IDLE)); // Čekání na provoz I2C
	while ((I2C0_MASTER_MCS_R & I2C_MCS_BUSY)); // Čekání na uvolnění I2C
}

// Začátek pro čtení dat z I2C
void i2c_startRead(unsigned char addrSlave, unsigned char *data, unsigned char ack)
{
	I2C0_MASTER_MSA_R = addrSlave + I2C_READ; // Nastavení adresy SLAVE
	if (ack != 0)
		I2C0_MASTER_MCS_R = I2C_MCS_START | I2C_MCS_RUN | I2C_MCS_ACK; // Odeslání
							// adresy + potvrzení přijatých dat
	else
		I2C0_MASTER_MCS_R = I2C_MCS_START | I2C_MCS_RUN | I2C_MCS_STOP; // Odeslání
						// adresy + nepotvrzení dat , tedy STOP
	while ((I2C0_MASTER_MCS_R & I2C_MCS_IDLE)); // Čekání na provoz I2C
	while ((I2C0_MASTER_MCS_R & I2C_MCS_BUSY)); // Čekání na uvolnění I2C
	*data = (unsigned char)I2C0_MASTER_MDR_R; // Vložení přijatých dat jako návratovou
							// hodnotu funkce
}

// Získání dat po I2C sběrnici
void i2c_getData(unsigned char *data, unsigned char len, unsigned char stop)
{
	unsigned char i;
	for (i = 0; i < len-1; i++) // Přenos všech dat kromě posledního
	{
		I2C0_MASTER_MCS_R = I2C_MCS_RUN | I2C_MCS_ACK; // Příjem dat + potvrzení
		while ((I2C0_MASTER_MCS_R & I2C_MCS_IDLE)); // Čekání na provoz I2C
		while ((I2C0_MASTER_MCS_R & I2C_MCS_BUSY)); // Čekání na uvolnění I2C
		data[i] = (unsigned char)I2C0_MASTER_MDR_R; // Uložení přijatých dat
	}


	if (stop == 0)
		I2C0_MASTER_MCS_R = I2C_MCS_RUN | I2C_MCS_ACK; // Příjem dat + potvrzení
	else
		I2C0_MASTER_MCS_R = I2C_MCS_RUN | I2C_MCS_STOP; // Příjem + nepotvrzení, tedy
							// STOP
	while ((I2C0_MASTER_MCS_R & I2C_MCS_IDLE)); // Čekání na provoz I2C
	while ((I2C0_MASTER_MCS_R & I2C_MCS_BUSY)); // Čekání na uvolnění I2C
	data[i] = (unsigned char)I2C0_MASTER_MDR_R; // Uložení posledního přijatého bajtu
}
```

Pro komunikaci s IQRF modulem slouží zdrojové soubory **iqrf.h** a **iqrf.c**. Protože mikrokontrolér každé senzorové jednotky pracuje na jiném hodinovém

kmitočtu, je nutné jiné nastavení hodinové frekvence periferie SSI a doby čekání jednotlivých stavů časovače TIMER2A, proto jsou jednotlivé hodnoty zakomentovány a opatřeny vhodným komentářem.

iqrf.h

```
#ifndef IQRF_H
#define IQRF_H

// Strunktura vysílacího zásobníku
typedef struct
{
	unsigned char spiCmd; // Příkaz pro IQRF
	unsigned char ptype; // Směr a počet bajtů
	unsigned char data[64]; // Data
	unsigned char len; // Délka dat k odeslání
	unsigned char hasData; // Příznak platných dat
} stIqrfTx;

// Proměnné pro hlavní vysílací zásobník a ukazatel na aktuální zásobník
extern stIqrfTx iqrfTx, *pIqrfTx;

// Struktura přijímacího zásobníku
typedef struct
{
	unsigned char spiStat1; // Stav IQRF
	unsigned char spiStat2; // Stav IQRF
	unsigned char data[64]; // Přijatá data
} stIqrfRx;

// Proměnná přijímacího zásobníku
extern stIqrfRx iqrfRx;

// Struktura stavu IQRF
typedef struct
{
	unsigned txDone : 1; // Data byla přijata (odeslána)
	unsigned workSleep : 1; // Přechod do režimu Sleep
	unsigned sleep : 1; // Režim Sleep
	unsigned wakeup : 1; // Provádí se probuzení IQRF
} bfIqrfState;

// Proměnná stabu IQRF
extern bfIqrfState iqrfState;

// Ukazatel na data v hlavním vysílacím zásobníku
extern volatile unsigned char *iqrfTxBuf;

// Inicializace periferie SSI, načtení dat z EEPROM (nutná předchozí inicializace),
// odeslání nastavení z EEPROM IQRF modulu
extern void iqrf_init(void);

// Odešle data z iqrfTxBuf o délce len
extern void iqrf_send(unsigned char len);
// Provede kontrolu přijatého CRC, vrací 1 při shodě
extern unsigned char iqrf_checkCrcRx(void);
// Znovu odeslání dat IQRF modulu
extern void iqrf_reSend(void);
// Odeslání příkazu na přechod do Sleep režimu
extern void iqrf_sleep(void);
// Provede probuzení IQRF modulu
extern void iqrf_wakeUp(void);

#endif
```

iqrf.c

```
#include "inc/lm3s811.h"
#include "board.h"
#include "iqrf.h"
#include "i2c.h"

// Časové konstanty pro detektor osob
/*#define WAIT_IQRF_20ms 12000
#define WAIT_IQRF_120us 8
#define WAIT_IQRF_40us 3
#define WAIT_IQRF_10us 1*/

// Časové konstanty pro detektor vozidel
#define WAIT_IQRF_20ms 25000
#define WAIT_IQRF_120us 15
#define WAIT_IQRF_40us 5
#define WAIT_IQRF_10us 2

// Vysílací buffery, iqrfTxTemp slouží pro prázdný přenos dat při přenosu z IQRF,
// *pIqrfTx je pointer na právě odesílaný buffer
stIqrfTx iqrfTx, iqrfTxTemp, *pIqrfTx = &iqrfTx;

// Přijímací buffer
stIqrfRx iqrfRx;

// Inicializace ukazatele na vysílací data
volatile unsigned char *iqrfTxBuf = iqrfTx.data;
static unsigned char iqrfStatus, crc, i; // Pomocné proměnné

unsigned char indexIqrf; // Index pro vysílací data

// Stav časovače pro IQRF
typedef enum
{
    IQRF_WAIT, // Čekání na nový datový přenos
    IQRF_CS1, // CS = 0, čekání 40us, a pak přenos dat
    IQRF_CS2, // Konec přenosu dat, CS = 1
    IQRF_WAKEUP // Probuzení IQRF
} IQRF_STATE_SPI;
IQRF_STATE_SPI iqrfStateSpi = IQRF_WAIT;

typedef enum
{
    IQRF_GET_STATUS, // Kontrola stavu IQRF
    IQRF_RX_TX_DATA // Odeslání nebo přímej dat pro IQRF
} IQRF_STATE_DATA;

IQRF_STATE_DATA iqrfStateData = IQRF_GET_STATUS; // Inicializace

bfIqrfState iqrfState = {0}; // Vynulování příznaků stavu IQRF

// Inicializace IQRF, povolení časovače pro řízení přenosu, povolení SSI a načtení a odeslání
// z EEPROM do IQRF
void iqrf_init()
{
    // Povolení časovače 2A
    TIMER2_CFG_R = TIMER_CFG_16_BIT; // 16bitový časovač
    TIMER2_TAMR_R = TIMER_TAMR_TAMR_1_SHOT; // Časovač pouze na jedno spuštění
    TIMER2_TAPR_R = 9; // Předdělička 10x
    TIMER2_TAILR_R = WAIT_IQRF_20ms; // Doba běhu časovače 20ms
    TIMER2_IMR_R = TIMER_IMR_TATOIM; // Povolení přerušení od časovače A
    TIMER2_CTL_R = TIMER_CTL_TAEN; // Povolení časovače 2A

    // Nastavení SSI
    // Nastavení hodinové frekvence sbarnice na 200kHz (bity SCR = 0)
    // SSI0_CPSR_R = 50; // - detektor osob
    // Nastavení hodinové frekvence sbarnice na 208,333kHz (bity SCR = 0) - detektor vozidel
    SSI0_CPSR_R = 80; // - detektor osob
    SSI0_CR0_R = SSI_CR0_DSS_8; // Přenos bude mít 8 bitů
    SSI0_CR1_R = SSI_CR1_SSE; // Povolení SSI
```

```
    i2c_startWrite(ADDR_EEPROM, 0x00, 0); // Zahájení komunikace I2C, odeslání adresy EEPROM,
                                          // adresa pro čtení je 0, bez STOP události
    i2c_startRead(ADDR_EEPROM, (unsigned char *)&iqrfTxBuf[1], 1); // Začátek čtení dat, data
                                          // do iqrfTxBuf[1], potvrzení dat
    i2c_getData((unsigned char *)&iqrfTxBuf[2], 5, 1); // Přečtení zbylých 5 bajtů dat, konec
                                          // je s událostí STOP

    iqrfTxBuf[0] = 0x00; // První bajt je roven 0 - nastavení modulu

    iqrf_send(7); // Odeslání 7 bajtů do IQRF
}

// Odeslání dat o délce len do IQRF
void iqrf_send(unsigned char len)
{
    register unsigned char i;
    iqrfTx.spiCmd = 0xF0; // Příkaz pro přenos dat
    iqrfTx.ptype = 0x80 + len; // Zápis dat do IQRF
    iqrfTx.len = len+4; // Celková délka přenášených dat, včetně CRC apod.

    indexIqrf = 0; // Nulový index pro přenos dat

    crc = 0x5F ^ iqrfTx.spiCmd ^ iqrfTx.ptype; // Začátek výpočtu CRC

    for (i = 0; i < len; i++)
    {
        crc ^= iqrfTx.data[i]; // Výpočet zbývajících částí CRC
    }
    iqrfTx.data[len] = crc; // Uložení CRC
    iqrfTx.data[len+1] = 0; // Za přenosem je umístěno zjištění stavu IQRF

    iqrfTx.hasData = 1; // Příznak, že jsou připravena data k odeslání

    if (iqrfState.sleep == 1)
    {
        iqrf_wakeUp(); // Je-li IQRF ve stavu spánku, dojde k probuzení
    }

    if (iqrfTxTemp.hasData == 0) // Pokud záložní buffer vysílacích dat nemá data
        pIqrfTx = &iqrfTx; // Bude odesílán buffer požadovaných dat
}

// Opětovné odeslání dat
void iqrf_reSend()
{
    pIqrfTx = &iqrfTx; // Bude se odesílat požadovaný buffer
    TIMER2_TAILR_R = WAIT_IQRF_120us; // Časovač na dobu nutnou mezi dvěma přenosy
    indexIqrf = 0; // Nulový index
}

// Uvedení IQRF do režimu spánku
void iqrf_sleep()
{
    iqrfTxBuf[0] = 0xFF; // Příkaz pro SLEEP
    iqrf_send(1); // Odeslání příkazu
    iqrfState.workSleep = 1; // Příkaz prováděného spánku
}

// Probuzení IQRF ze spánku
void iqrf_wakeUp()
{
    ClrIQRFCs(); // CS = 0 - probuzení IQRF
    iqrfState.sleep = 0; // Není režim spánku
    iqrfState.wakeup = 1; // Je v procesu probouzení

    TIMER2_TAILR_R = WAIT_IQRF_10us; // Časovač na 10us
    TIMER2_CTL_R = TIMER_CTL_TAEN; // Povolení časovače
    iqrfStateSpi = IQRF_WAKEUP; // Stav SPI je probuzení
}

// Kontrola přijatého CRC
unsigned char iqrf_checkCrcRx()
{
    register unsigned char i;
```

```
    crc = pIqrfTx->ptype  ^ 0x5F; // Začátek výpočtu CRC

    for (i = 0; i < pIqrfTx->len - 4; i++)
    {
        crc ^= iqrfRx.data[i]; // Započítání aktuální hodnoty pro CRC
    }

    if (crc == iqrfRx.data[pIqrfTx->len-4])
        return 1; // CRC je rovnají
    else
        return 0; // CRC se nerovnají
}

// Přerušení pro časovač 2A
void IQRF_IntTimerHandler()
{
    TIMER2_ICR_R = TIMER_ICR_TATOCINT; // Vynulování příznaku přerušení
    switch (iqrfStateSpi) // Stav SPI sběrnice
    {
        case IQRF_WAKEUP: // Probuzení IQRF
            SetIQRFCs(); // CS = 1
            TIMER2_TAILR_R = WAIT_IQRF_40us; // Časovač na 40us
            iqrfStateSpi = IQRF_WAIT; // Čekání na nový datový přenos
            TIMER2_CTL_R = TIMER_CTL_TAEN; // Povolení časovače
            break;
        case IQRF_WAIT: // Čekání na nový datový přenos
            ClrIQRFCs(); // CS = 0
            TIMER2_TAILR_R = WAIT_IQRF_40us; // Čekej 40us
            iqrfStateSpi = IQRF_CS1; // Stav pro zahájení přenosu dat
            TIMER2_CTL_R = TIMER_CTL_TAEN; // Povolení časovače
            break;
        case IQRF_CS1: // Zahájení přenosu dat
            iqrfStateSpi = IQRF_CS2; // Stav ukončující přenos
            if (iqrfStateData == IQRF_GET_STATUS) // Má se zjistit stav IQRF
                SSI0_DR_R = 0x00;
            else
                SSI0_DR_R = ((unsigned char *)pIqrfTx)[indexIqrf]; // Odesílají se data do
                                                                   // IQRF
            TIMER2_TAILR_R = 2*WAIT_IQRF_40us; // Čekání na odeslání + doba nutná za odesla
                                               // nými daty než se CS = 1
            TIMER2_CTL_R = TIMER_CTL_TAEN; // Povolení časovače
            break;
        case IQRF_CS2: // Ukončení přenosu dat
            SetIQRFCs(); // CS = 1
            iqrfStateSpi = IQRF_WAIT; // Stav čekání na nový přenos dat
            if (iqrfStateData == IQRF_GET_STATUS) // Zjišťoval se stav IQRF
            {
                iqrfStatus = SSI0_DR_R; // Vyzvednutí přijatých dat
                TIMER2_TAILR_R = WAIT_IQRF_20ms; // Čekej 20ms - výchozí hodnota

                if (iqrfStatus == 0x80 && pIqrfTx->hasData != 0) // IQRF může příjmout data a
                                                                 // data jsou připravena
                {
                    iqrfStateData = IQRF_RX_TX_DATA; // Stav pro odesílání dat
                    TIMER2_TAILR_R = WAIT_IQRF_120us; // Čekej 120us - doba mezi jednotlivými
                                                      // přenosy
                }
                else if (iqrfStatus > 0x40 && iqrfStatus <= 0x69) // Modul má k odeslání
                                                                  // nějaká data
                {
                    if (pIqrfTx->hasData == 0) // Pokud aktuální vysílací buffer nemá data je
                                               // nutné nějaká data připravit, musí se odeslat
                    {
                        pIqrfTx = &iqrfTxTemp; // Bude se odesílat záložní buffer
                        iqrfStatus -= 0x40; // Zjištění počtu bajtů k přečtení

                        // Naplnění hlavičky dat
                        pIqrfTx->spiCmd = 0xF0;
                        pIqrfTx->ptype = 0x00 + (iqrfStatus);
                        crc = 0x5F ^ pIqrfTx->spiCmd ^ pIqrfTx->ptype; // Začátek výpočtu CRC
                        pIqrfTx->len = iqrfStatus + 4; // Celková délka odesílaných dat

                        for (i = 0; i < iqrfStatus; i++)
                        {
```

```
                        crc ^= pIqrfTx->data[i]; // Výpočet CRC
                    }
                    pIqrfTx->data[iqrfStatus] = crc; // Uložení CRC
                    pIqrfTx->data[iqrfStatus+1] = 0; // Za daty bude kontrola IQRF
                    indexIqrf = 0; // Nulový index pro odesílání dat
                }
                iqrfStateData = IQRF_RX_TX_DATA; // Budou se odelílat data
                TIMER2_TAILR_R = WAIT_IQRF_120us; // Čekej 120us
            }
        }
        else // Režim odesílání dat
        {
            *(&iqrfRx.spiStat1+indexIqrf) = SSI0_DR_R; // Vyzvednutí přijatých dat
            indexIqrf++; // zvýšení indexu pro ukládání dat

            if (pIqrfTx->len == indexIqrf) // Odeslána všechna data
            {
                TIMER2_TAILR_R = WAIT_IQRF_20ms; // Čekej 20ms pro další přenos
                iqrfStateData = IQRF_GET_STATUS; // Příště se bude zjišťovat stav

                iqrfState.txDone = 1; // Data byla odeslána
                pIqrfTx->hasData = 0; // Nejsou žádná data k odeslání
            }
            else
                TIMER2_TAILR_R = WAIT_IQRF_120us; // Nejsou odeslána všechna data, čekej
                                                  // 120us
        }

        TIMER2_CTL_R = TIMER_CTL_TAEN; // Povolení časovače
        break;
    }
}
```

Následující výpis programu představuje hlavní program mikrokontroléru LM3S811 pro zpracování v senzorové jednotce detekce osob:

Osoby_main.c

```
#include "inc/lm3s811.h"
#include "board.h"
#include "iqrf.h"
// Soubory knihovny StellarisWare
#include "inc/hw_ints.h"
#include "inc/hw_types.h"
#include "inc/hw_memmap.h"
#include "driverlib/interrupt.h"
#include "i2c.h"

// Datová struktura s nastavením zpracování signálu
typedef struct
{
    unsigned short prahH; // Horní práh
    unsigned short prahD; // Dolní práh
    unsigned short maxMoznyCil; // Maximální hodnota časovače cntMoznyCil
    unsigned short maxIsCil; // Maximální hodnota časovače cntIsCil
    unsigned char maxPrekroceniPrahu; // Hodnota počtu překročení prahů pro detekci cíle
} stSetupAnalog;

stSetupAnalog setupAnalog;

volatile unsigned int cntPrekroceniPrahu = 0, vzorek, cntIsCil = 0, cntMocnyCil = 0;
volatile unsigned char isCil = 0, odeslanCil = 0;
volatile signed char typPrahu = 0;

int main()
{
    // Nastavení hodinové frekvence
    SYSCTL_RCC_R = SYSCTL_RCC_OSCSRC_MAIN | SYSCTL_RCC_XTAL_6MHZ | SYSCTL_RCC_BYPASS |
            SYSCTL_RCC_OEN | SYSCTL_RCC_PWRDN;
```

```
// Povolení hodinové frekvence pro použité periferie
SYSCTL_RCGC2_R = SYSCTL_RCGC2_GPIOC | SYSCTL_RCGC2_GPIOB | SYSCTL_RCGC2_GPIOA;
SYSCTL_RCGC1_R = SYSCTL_RCGC1_TIMER0 | SYSCTL_RCGC1_SSI0 | SYSCTL_RCGC1_TIMER2 |
                                 SYSCTL_RCGC1_I2C0;
SYSCTL_RCGC0_R = SYSCTL_RCGC0_ADC | SYSCTL_RCGC0_ADCSPD500K | SYSCTL_RCGC0_WDT;

// Inicializace portů
InitPortA();
InitPortB();
InitPortC();

// Povolení časovače TIMER0A - pro A/D
TIMER0_CFG_R = TIMER_CFG_16_BIT; // 16bit. časovač
TIMER0_TAMR_R = TIMER_TAMR_TAMR_PERIOD; // Periodicky spouštěn
TIMER0_TAILR_R = 8000; // 1ms
// Povolení časovače jako spouštění pro A/D
TIMER0_CTL_R = TIMER_CTL_TAEVENT_POS | TIMER_CTL_TAOTE | TIMER_CTL_TAEN;

// Nastavení A/D převodníku
ADC_SSPRI_R = ADC_SSPRI_SS3_1ST; // Sekce 3 - 1 priorita
ADC_ACTSS_R = ADC_ACTSS_ASEN3; // Povolení 3 sekce
ADC_EMUX_R = ADC_EMUX_EM3_TIMER; // Spouštění časovačem
ADC_SSMUX3_R = 0; // Vstup ADC0
ADC_SSCTL3_R = ADC_SSCTL3_END0 | ADC_SSCTL3_IE0; // povolení přerušení a ukončení převodu
ADC_IM_R = ADC_IM_MASK3; // Povolení přerušení pro 3 sekci
ADC_PSSI_R = ADC_PSSI_SS3; // Inicializace 3 sekce

i2c_init(); // Nastavení I2C sběrnice
iqrf_init(); // Inicializace SSI periferie - načtení a odeslání konfigurace IQRF modulu

IntMasterEnable(); // Globální povolení přerušení
IntEnable(INT_ADC3); // Povolení přerušení pro A/D
IntEnable(INT_TIMER2A); // Povolení přerušení pro TIMER2A - časování komunikace s IQRF

// Načtení konfigurace pro zpracování analogových vzorků z paměti EEPROM
i2c_startWrite(ADDR_EEPROM, 0x10, 0);
i2c_startRead(ADDR_EEPROM, (unsigned char *)&setupAnalog.prahH, 1);
i2c_getData(((unsigned char *)(&setupAnalog.prahH))+1, 8, 1);

while (iqrfState.txDone == 0); // Čekání na odeslání nastavení IQRF modulu

while(1) // Hlavní smyčka programu
{
    if (iqrfState.txDone == 1)
    {
        iqrfState.txDone = 0;
        if (!iqrf_checkCrcRx())
            iqrf_send(iqrfTx.len-4);
        else if (iqrfRx.spiStat1 > 0x40 && iqrfRx.spiStat1 <= 0x69) // Byla přijatá
                                                    nějaká data z IQRF
        {
            if (iqrfRx.data[0] != 0x00) // Neplatné potvrzení
            {
                iqrf_send(iqrfTx.len-4); // Znovu odeslání
            }
            else if (iqrfState.wakeup == 0) // Odpověď je sprvávná a neprobíhá probuzení
            {
                iqrf_sleep(); // Přechod do režimu SLEEP
            }

            iqrfState.wakeup = 0; // Neprobíhá probuzení
        }
        else
        {
            if (iqrfTx.hasData != 0) // Hlavní zásobník má data
            {
                iqrf_reSend(); // Opětovné odeslání dat
            }
        }
    }

    // Je detekován cíl a nebyla odeslána informace o přítomnosti chodce
    if (isCil == 1 && odeslanCil == 0)
    {
```

```
            // Data k odeslání
            iqrfTxBuf[0] = 0x01; // Odeslání dat
            iqrfTxBuf[1] = 0; // Adresa příjemce
            iqrfTxBuf[2] = 1; // Data - je chodec
            iqrf_send(3); // Odeslání dat

            odeslanCil = 1; // Informace o přítomnosti chodce byla odeslána
        }
        // Cíl není detekován a byla odeslána informace o přítomnosti chodce
        else if (isCil == 0 && odeslanCil == 1)
        {
            // Data k odeslání
            iqrfTxBuf[0] = 0x01; // Příkaz - odeslání dat
            iqrfTxBuf[1] = 0; // Adresa příjemce
            iqrfTxBuf[2] = 0; // Data - chodec není
            iqrf_send(3); // Odeslání dat

            odeslanCil = 0; // Informace o nepřítomnosti chodce byla odeslána
        }
    }
    return 0;
}

// Zpracování analogového signálu
void ADCEndSeq()
{
    vzorek = ADC_SSFIFO3_R; // Vyzvednutí vzorku signálu

    // Je vzorek vyšší než horní práh a je detekováno první překročení
    if (vzorek > setupAnalog.prahH && typPrahu != 1)
    {
        // Byl posledním překročeným prahem dolní práh
        if (typPrahu == -1)
        {
            cntMocnyCil = 0; // Nulování časovače překračování prahů
            cntPrekroceniPrahu++; // Navýšena hodnota překročeného opačného prahu
            if (isCil) // Je již detekován cíl
                cntIsCil = 0; // Nulování časovače výskytu cíle
        }
        else
        {
            typPrahu = 0; // Prahy nebyly střídány - žádný typ prahu
        }

        typPrahu = 1; // Byl překročen horní práh
    }

    // Je vzorek nižší než dolní práh a je detekováno první překročení
    if (vzorek < setupAnalog.prahD && typPrahu != -1)
    {
        // Byl posledním překročeným prahem horní práh
        if (typPrahu == 1)
        {
            cntMocnyCil = 0; // Nulování časovače překračování prahů
            cntPrekroceniPrahu++; // Navýšena hodnota překročeného opačného prahu
            if (isCil) // Je již detekován cíl
                cntIsCil = 0; // Nulování časovače výskytu cíle
        }
        else
        {
            typPrahu = 0; // Prahy nebyly střídány - žádný typ prahu
        }

        typPrahu = -1; // Byl překročen dolní práh
    }

    cntMocnyCil++; // Navýšení časovače
    // Dosáhl časovač maximální hodnoty
    if (cntMocnyCil >= setupAnalog.maxMoznyCil)
    {
        typPrahu = 0; // žádný práh nebyl překročen
        cntMocnyCil = 0; // Nulování časovače
        cntPrekroceniPrahu = 0; // Nulování počtu překročených prahů
    }
```

```
    // Dosáh počet střídajících se překročení prahů hodnotu pro detekci cíle
    if (cntPrekroceniPrahu >= setupAnalog.maxPrekroceniPrahu)
    {
        isCil = 1; // Cíl je detekován
        cntIsCil = 0; // Nulování časovače přítomnosti cíle
        cntPrekroceniPrahu = 0; // Nulování počtu překročených prahů
    }

    // Je detekován cíl - navýšení hodnoty časovače a kontrola dosažení maximální hodnoty
    if (isCil == 1 && ++cntIsCil >= setupAnalog.maxIsCil)
        isCil = 0; // Časovač přetekl - cíl není detekován
}
```

Soubor **Vozidla_main.c** obsahuje kompletní výpis programu mikrokontroléru jednotky detekce vozidel:

Vozidla_main.c

```
#include "inc/lm3s811.h"
#include "board.h"
#include "iqrf.h"
#include "i2c.h"
// Soubory knihovny StellarisWare
#include "inc/hw_ints.h"
#include "inc/hw_types.h"
#include "inc/hw_memmap.h"
#include "driverlib/interrupt.h"

#define ARM_MATH_CM3 // Typ procesoru pro knihovnu CMSIS
#include "include/arm_math.h" // Soubor knihovny CMSIS

// Používané proměnné
signed int samples1[512], samples2[512];
signed int mag[129], res;
unsigned int index;
unsigned int indexSamples = 0;
unsigned char isNewSamples = 0, s1 = 1;

unsigned int prah[129];

unsigned int klPrumer[4], aktualniRychlost, maximalniRychlost, cntPrekroceniPrahu,
                                cntZtrataCile;
unsigned char indexKlPrumer = 0;
unsigned char isCil = 0;

// Datová struktura pro nastavení časování analogového signálu
typedef struct
{
    unsigned char prekroceniNoCil; // Počet překročení prahu bez výskytu cíle
    unsigned char prekroceniIsCil; // Počet překročení prahu při výskytu cíle
    unsigned char minPrekroceniMaxRych; // Počet překročení prahu pro výpočet rychlosti
    unsigned char maxZtrataCileNoCil; // Maximální hodnota pro ztrátu cíle při nepřítomnosti
                                                            // cíle
    unsigned char maxZtrataCileIsCil; // Maximální hodnota pro ztrátu cíle při přítomnosti
                                                            // cíle
} stSetupAnalog;

volatile stSetupAnalog setupAnalog;

// Datová struktura pro nastavení jednoho prahu
typedef struct
{
    unsigned char pocet; // Počet výskytů prahu
    unsigned int hodnota; // Hodnota prahu
} stSetupPrah;

volatile stSetupPrah setupPrah;

volatile unsigned int i,pocetPrahu, adrPrahu, zacatekPrahu;

int main()
```

```
{
    arm_cfft_radix4_instance_q31 S;

    // Nastavení zdroje hodinové frekvence
    SYSCTL_RCC_R = SYSCTL_RCC_SYSDIV_16 | SYSCTL_RCC_OSCSRC_MAIN | SYSCTL_RCC_XTAL_6MHZ |
                                SYSCTL_RCC_USESYSDIV;

    // Povolení všech použitých periferií
    SYSCTL_RCGC2_R = SYSCTL_RCGC2_GPIOC | SYSCTL_RCGC2_GPIOB | SYSCTL_RCGC2_GPIOA;
    SYSCTL_RCGC1_R = SYSCTL_RCGC1_TIMER0 | SYSCTL_RCGC1_SSI0 | SYSCTL_RCGC1_TIMER2 |
                SYSCTL_RCGC1_I2C0;
    SYSCTL_RCGC0_R = SYSCTL_RCGC0_ADC | SYSCTL_RCGC0_ADCSPD500K;

    // Inicializace portů
    InitPortA();
    InitPortB();
    InitPortC();

    // Nastavení časovače pro spouštění A/D převodníku
    TIMER0_CFG_R = TIMER_CFG_16_BIT; // Časovač je 16bitový
    TIMER0_TAMR_R = TIMER_TAMR_TAMR_PERIOD; // Spouštěn periodicky
    TIMER0_TAILR_R = 830; // Doba pro čítání ~15kHz
    // Povolení časovače, spouštění A/D převodníku
    TIMER0_CTL_R = TIMER_CTL_TAEVENT_POS | TIMER_CTL_TAOTE | TIMER_CTL_TAEN;

    // A/D převodník
    ADC_SSPRI_R = ADC_SSPRI_SS3_1ST; // Nejvyšší priorita pro Sequencer 3
    ADC_ACTSS_R = ADC_ACTSS_ASEN3; // Povolení Sequenceru 3
    ADC_EMUX_R = ADC_EMUX_EM3_TIMER; // Spouštění časovačem
    ADC_SSMUX3_R = 0; // Vstup ADC0
    ADC_SSCTL3_R = ADC_SSCTL3_END0 | ADC_SSCTL3_IE0; // Povolení přerušení a ukončení
                                                                    // sekvence
    ADC_IM_R = ADC_IM_MASK3; // Povolení přerušení pro Sequencer 3

    // Inicializace FFT, 256 vzorků, dopředná transformace, reverzní adresování
    arm_cfft_radix4_init_q31(&S, 256, 0, 1);

    i2c_init(); // Inicializace I2C sběrnice
    iqrf_init(); // Inicializace IQRF modulu, načtení a odeslání konfiguračního nastavení

    IntMasterEnable(); // Povolení globálního přerušení
    IntEnable(INT_ADC3); // Povolení přerušení pro Sequecer 3
    IntEnable(INT_TIMER2A); // Povolení pro Timer2A - časování komunikace s IQRF

    // Načtení konfigurace pro zpracování analogových vzorků z paměti EEPROM
    i2c_startWrite(ADDR_EEPROM, 0x10, 0);
    i2c_startRead(ADDR_EEPROM, (unsigned char *)&setupAnalog.prekroceniNoCil, 1);
    i2c_getData(((unsigned char *)(&setupAnalog.prekroceniNoCil))+1, 4, 1);

    // Zjištění počtu prahů
    i2c_startWrite(ADDR_EEPROM, 0x20, 0);
    i2c_startRead(ADDR_EEPROM, (unsigned char *)&pocetPrahu, 0);

    // Načtení hodnot prahů
    i = 0;
    adrPrahu = 0x21; // Adresa nastavení prvního prahu
    zacatekPrahu = 0; // Počet načtených prahů
    while (pocetPrahu != 0) // Zpracovat všechny prahy
    {
        // Načtení nastavení prahu
        i2c_startWrite(ADDR_EEPROM, adrPrahu, 0);
        i2c_startRead(ADDR_EEPROM, (unsigned char *)&setupPrah.pocet, 1);
        i2c_getData(((unsigned char *)(&setupPrah.hodnota)), 4, 1);

        // Přičtení počtu hodnot prahu k předešlým
        zacatekPrahu += setupPrah.pocet;
        // Uložení počtů prahu
        for(;i<zacatekPrahu && i<129; i++)
            prah[i] = setupPrah.hodnota;

        adrPrahu += 5; // Navýšení adresy EEPROM
    }

    while (iqrfState.txDone == 0); // Čekání na odeslání dat pro IQRF
```

```
while (1) // Hlavní smyčka programu
{
    if (iqrfState.txDone == 1) // Byla odeslána data pro IQRF
    {
        iqrfState.txDone = 0;
        if (!iqrf_checkCrcRx()) // Kontrola CRC přijatých dat
            iqrf_send(iqrfTx.len-4); // Pokud je CRC špatné, odešlou se data znovu
        else if (iqrfRx.spiStat1 > 0x40 && iqrfRx.spiStat1 <= 0x69) // Byla přijatá
                                                                    // nějaká data z IQRF
        {
            if (iqrfRx.data[0] != 0x00) // Pokud IQRF odpovědělo cokoli kromě 0 - chyba
            {
                iqrf_send(iqrfTx.len-4); // Odeslání dat znovu
            }
            else if (iqrfState.wakeup == 0) // Pokud je odpověď správná a neprobíhá
                                            // probuzení
            {
                iqrf_sleep(); // Režim SLEEP
            }

            iqrfState.wakeup = 0; // Neprobíhá probuzení IQRF modulu
        }
        else
        {
            if (iqrfTx.hasData != 0) // V hlavním zásobníku jsou data
            {
                iqrf_reSend(); // Opětovné odeslání dat
            }
        }
    }

    if (isNewSamples != 0) // Jsou nové vzorky analog. signálu
    {
        isNewSamples = 0; // Nulování příznaku nových vzorků
        if (s1) // Podle zásobníku
        {
            arm_cfft_radix4_q31(&S, samples2); // Výpočet FFT ze zásobníku 2
            arm_cmplx_mag_squared_q31(samples2, mag, 129); // Výpočet výk. spektra
        }
        else
        {
            arm_cfft_radix4_q31(&S, samples1); // Výpočet FFT ze zásobníku 1
            arm_cmplx_mag_squared_q31(samples1, mag, 129); // Výpočet výk. spektra4
        }
        arm_max_q31(&mag[5], 124, &res, &index); // Nalezení maxima

        // Je maximální hodnota nad daným prahem detekce
        if (res > prah[index])
        {
            cntPrekroceniPrahu++; // Navýšení počtu překročeného prahu
            // Uložení rychlosti do zásobníku pro průměr
            klPrumer[indexKlPrumer] = index+5;
        }
        else // Není překročen práh
        {
            cntPrekroceniPrahu = 0; // Nulování počtu překročeného prahu
            klPrumer[indexKlPrumer] = 0; // Není žádná rychlost
        }
        // Navýšení ukazatele na prázdné slovo v kruhovém zásobníku
        if (++indexKlPrumer == 4)
            indexKlPrumer = 0; // Ukazatel je moc veliký - přesun na začátek

        // Výpočet aktuální rychlosti - průměr
        aktualniRychlost = (klPrumer[0] + klPrumer[1] + klPrumer[2] + klPrumer[3]) >> 2;

        // Je dosaženo požadovaného počtu překročení prahu
        if (isCil == 0 && cntPrekroceniPrahu >= setupAnalog.prekroceniNoCil || isCil == 1
                                    && cntPrekroceniPrahu >= setupAnalog.prekroceniIsCil)
        {
            isCil = 1; // Cíl je detekován
            cntZtrataCile = 0; // Nulování čítače ztráty cíle
        }
```

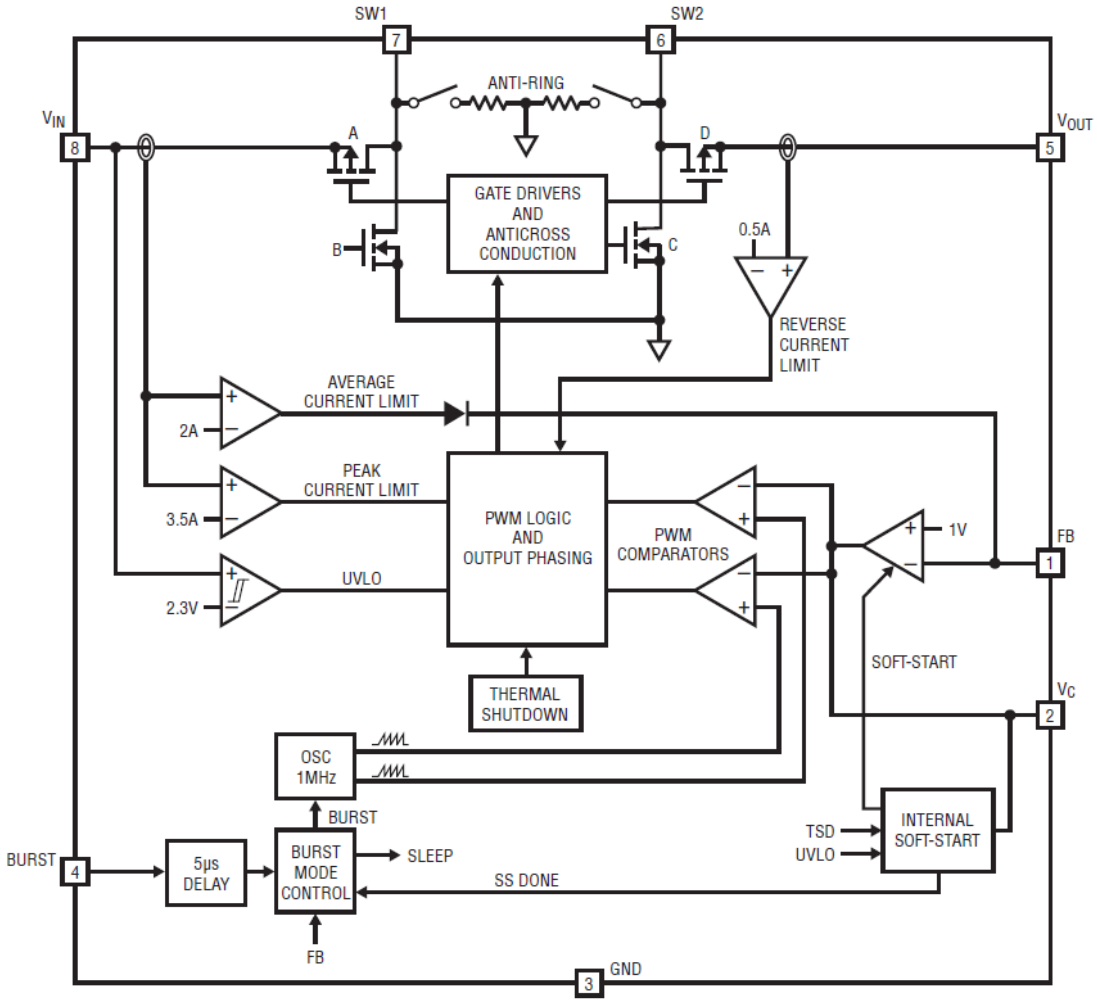
```
            // Je potencionální cíl detekován a je aktuální rychlost menší než maximální
            if (cntPrekroceniPrahu >= setupAnalog.minPrekroceniMaxRych
                                                && aktualniRychlost > maximalniRychlost)
                maximalniRychlost = aktualniRychlost; // Uložení nové maximální hodnoty

            cntZtrataCile++; // Navýšení časovače ztráty cíle
            // Je časovač moc veliký
            if (isCil == 1 && cntZtrataCile >= setupAnalog.maxZtrataCileIsCil || isCil == 0
                                        && cntZtrataCile >= setupAnalog.maxZtrataCileNoCil)
            {
                cntZtrataCile = 0; // Nulování časovače
                if (isCil == 1) // Byl detekován cíl
                {
                    // Odeslání informace o vozidlu
                    iqrfTxBuf[0] = 1; // Příkaz pro přenos dat
                    iqrfTxBuf[1] = 0; // Adresa příjemce
                    iqrfTxBuf[2] = maximalniRychlost; // Rychlost vozidla
                    iqrf_send(3); // Odeslání dat
                }
                maximalniRychlost = 0; // Nulování maximální rychlosti
                isCil = 0; // Cíl již není detekován
            }
        }
    }
    return 0;
}

// Přerušení A/D převodníku
void ADCEndSeq()
{
    // Podle aktuálního zásobníku vzorků
    if (s1)
    {
        // Uložení dat do zásobníku
        samples1[indexSamples++] = ((ADC_SSFIFO3_R & 0x3FF) << 21); // Reálná část
                                                                   // (formát 1.31)
        samples1[indexSamples++] = 0; // Imaginární část
    }
    else
    {
        samples2[indexSamples++] = ((ADC_SSFIFO3_R & 0x3FF) << 21); // Reálná část
                                                                   // (formát 1.31)
        samples2[indexSamples++] = 0; // Imaginární část
    }

    // Jsou uloženy všechny vzorky
    if (indexSamples == 512)
    {
        indexSamples = 0; // Na začátek nového zásobníku
        s1 ^= 1; // Přepnutí zásobníků
        isNewSamples = 1; // Příznak nových vzorků
    }
}
```

Příloha E. Popis DC/DC měniče LTC3538

DC/DC měnič LTC3538 je synchronní BUCK-BOOST měnič se zatěžovacím proudem až 800 mA, výrobcem je společnost Linear Technology. Popis celého obvodu lze nalézt v [5]. Obvod LTC3538 je určen pro zařízení typu MP3 přehrávače, digitální kamery, zařízení PDA, GPS přijímače apod. Rozsah vstupního napětí je od 2,4 V do 5,5 V, rozsah výstupního napětí je od 1,8 V do 5,25 V, frekvence spínání je 1 MHz, rozměry pouzdra obvodu jsou malé (2 mm x 3 mm). Obvod umožňuje funkci jak snižujícího, tak zvyšujícího měniče s vysokou účinností a nízkým šumem.

Tento měnič umožňuje automatickou rekonfiguraci zapojení indukčnosti, která umožňuje základní funkci měniče (viz kapitola 4). Rekonfigurace probíhá na základě hodnoty vstupního napětí a hodnotě požadovaného výstupního napětí. Obvod umožňuje dva režimy funkce. Prvním z nich je režim PWM, tedy režim s konstantní frekvencí, druhým je režim BURST, v tomto režimu je frekvence spínání proměnlivá, závisí na proudu zátěže.

Pro bližší představu o funkci tohoto obvodu je dobré představit vnitřní blokové zapojení:



Obrázek E.1 – Vnitřní zapojení obvodu LTC3538 [5]

Spínaná indukčnost je připojena mezi vývody označené SW1 a SW2. Tyto vývody jsou umístěny mezi čtveřici spínacích MOFSET tranzistorů označených A, B, C a D. V závislosti na aktuálně používaném typu měniče dojde k rekonfiguraci spínacích tranzistorů. Spínací tranzistory si lze přestavit jako spínače použité v popisu principu měničů. Ve čtveřici tranzistorů jsou dva tranzistory typu N se jmenovitou impedancí 0,17 Ω a dva tranzistory typu P se jmenovitou impedancí 0,2 Ω.

Obsluhu čtveřice spínacích tranzistorů A, B, C a D zajišťuje blok nazvaný GATE DRIVERS AND ANTICROSS CONDUCTION zaručující, že při spínání tranzistorů nedojde ke vzniku nežádoucích situací, např. nesepnou dva tranzistory najednou a nedojde tak ke vzniku zkratu napájení apod. Tento blok přijímá povely

centrálního bloku nazvaného PWM LOGIC AND OUTPUT PHASING, do kterého směřují veškeré informace (výstupy komparátorů) a který tyto informace vyhodnocuje a na jejich základě řídí chod celého obvodu.

Napěťovou úrovní na vývodu BURST se určuje režim, ve kterém bude obvod pracovat (PWM nebo BURST). Pokud je tento vývod uzemněn, pracuje obvod v režimu PWM a napěťová úroveň nad hodnotou 1,4 V uvede obvod do režimu BURST.

Pro ochranu obvodu slouží trojice „proudových" komparátorů kontrolujících vstupní proud do obvodu z vývodu V_{IN}. První komparátor kontroluje průměrnou hodnotu proudu, pokud hodnota proudu překročí 2 A, výstup komparátoru bude překlopen do kladné hodnoty. Výstup komparátoru je přiveden na vývod FB kontrolující velikost výstupního napětí, vysoká hodnota napětí na tomto vývodu způsobí snížení napětí na výstupu obvodu a tím i snížení vstupního proudu. Pro správnou funkci tohoto komparátoru musí být paralelní impedance připojená k vývodu FB větší než 100 kΩ . Další „proudový" komparátor kontroluje špičkovou hodnotu vstupního proudu na hodnotu 3,5 A, výstup komparátoru je přiveden do bloku označeného PWM LOGIC AND OUTPUT PHASING. Tento blok řídí spínání tranzistorů, při zjištění nadproudové špičkové hodnoty bude ovlivněno spínání tranzistorů za účelem snížení tohoto proudu, rozepnutím tranzistoru A na dobu 50 ns. Posledním třetím „proudovým" komparátorem je komparátor kontrolující hodnotu výstupního proudu z vývodu V_{OUT}. Komparátor kontroluje hodnotu výstupního proudu oproti referenční hodnotě 0,5 A, při překročení této hodnoty proudu dojde k rozpojení tranzistoru D.

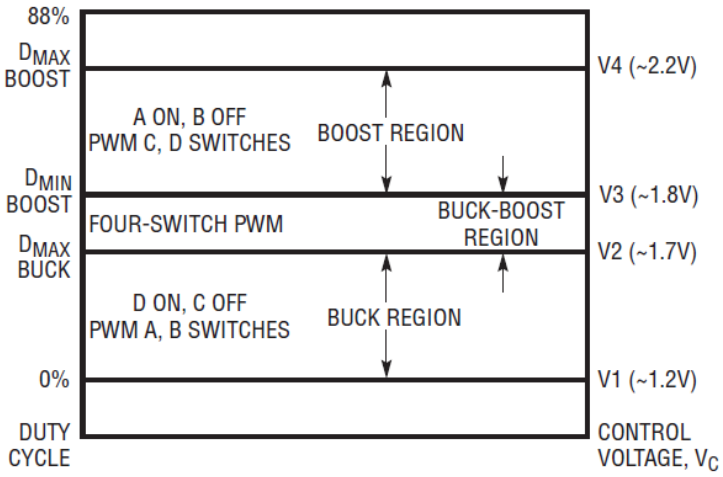
Ke vstupnímu vývodu V_{IN} je připojen ještě jeden komparátor kontrolující hodnotu napětí na tomto vývodu oproti napětí 2,3 V. Výstup komparátoru určuje moment připojení obvodu ke vstupnímu napětí. Po připojení vstupního napětí (hodnota vstupního napětí bude vyšší než 2,3 V) je spuštěn pomalý rozběh obvodu (Soft-Start), tento rozběh trvá po dobu 1,5 ms. Po tuto dobu je obvod v režimu PWM bez ohledu na hodnotu vývodu BURST.

Vývod FB slouží jako vstup pro zpětnou informaci o hodnotě výstupního napětí obvodu. Hodnota napětí na tomto vývodu je přivedena na invertující vstup

chybového zesilovače. Tento zesilovač má za úkol vyhodnotit velikost chyby na vývodu FB oproti referenční hodnotě 1 V. Pro lepší nastavení chybového zesilovače je výstup zesilovače vyveden na vývod označený V_C. Tímto vývodem spolu s vývodem FB lze nastavit zesílení chybového zesilovače, ale i frekvenční pásmo, ve kterém bude zesilovač pracovat. Výstup chybového zesilovače je vnitřně dále přiveden na dvojici PWM komparátorů (PWM COMPARATORS), které výstup chybového zesilovače porovnávají s interním oscilátorem pracujícím na frekvenci 1 MHz. Vývod V_C kromě výstupu chybového zesilovače plní další funkci, pokud na tento vývod bude externě připojeno napětí menší než 0,25 V, obvod se uvede to režimu SLEEP, obvod bude odebírat ze vstupního zdroje minimální proud, a spínací tranzistory budou odpojeny.

Vývod označený GND je spojen se zemí. Posledním vývodem, který na blokovém schématu obvodu není znázorněn, je termální ploška sloužící pro odvod tepla z obvodu. Termální ploška musí být také spojena se zemí a s co největší plochou pro co nejlepší odvod tepla.

Typ měniče (zvyšující nebo snižující) se automaticky vybírá, to způsobuje změnu poměru času sepnutí a odpojení indukčnosti (DUTY CYCLE) tuto situaci vystihuje následující obrázek:

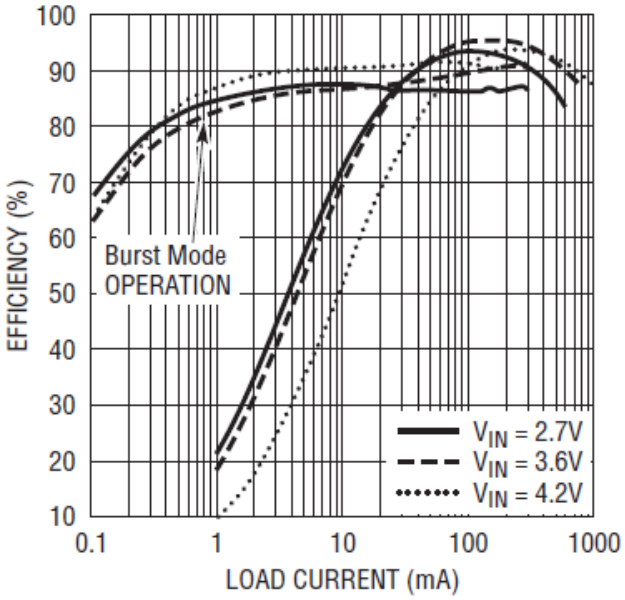


Obrázek E.2 – Přehled poměru připnutí/odepnutí indukčnosti s funkcí jednotlivých spínacích tranzistorů a velikost napětí na výstupu V_C [5]

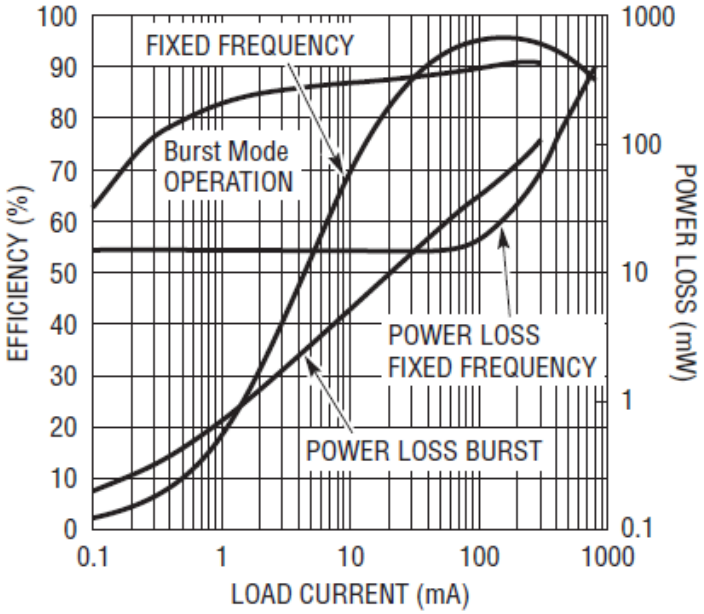
Při funkci snižujícího měniče (BUCK REGION), kdy je $V_{IN} > V_{OUT}$ jsou tranzistory C a D vždy rozpojeny. Až výstup V_C dosáhne hodnoty napětí V1 (~1,2 V)

spínací tranzistor A začne spínat, v okamžicích, kdy je tranzistor A rozepnut, spíná tranzistor B. V případě, že se vstupní napětí rovná výstupnímu $V_{IN} \sim V_{OUT}$, obvod se nachází v oblasti BUCK-BOOST REGION. Do této oblasti obvod přejde, pokud se na výstupu V_C objeví napětí větší než V2 (~1,7 V). V této pracovní oblasti pracují všechny čtyři tranzistory. Poslední pracovní oblastí je oblast zvyšujícího měniče (BOOST REGION), kde je výstupní napětí vyšší než napětí vstupní $V_{IN} < V_{OUT}$. Překročením napětí na výstupu V_C nad hodnotu V3 (~1,8 V) budou tranzistory A a B vyřazeny z činnosti a budou pracovat pouze tranzistory C a D.

Obvod LTC3538 se vyznačuje poměrně vysokou účinností dosahující za určitých okolností hodnoty až kolem 95 %, přesnější pohled poskytují následující obrázky:



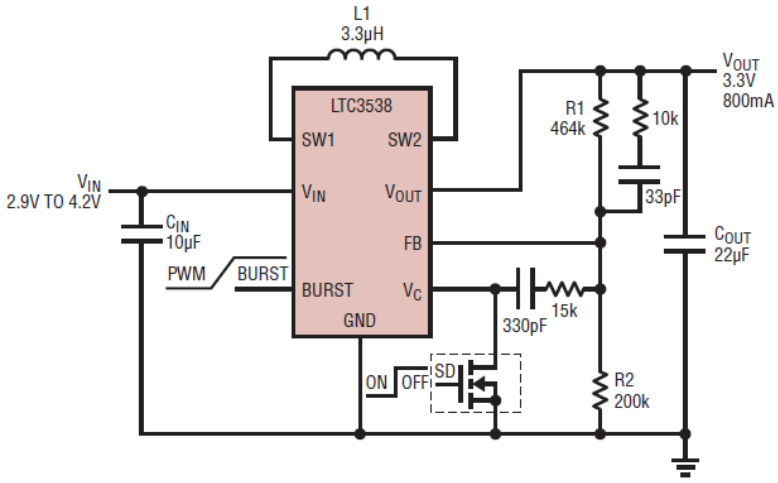
Obrázek E.3 – Závislost účinnosti na výstupním proudu pro vstupní napětí z Li-onové baterie a výstupní napětí 3,3 V [5]



Obrázek E.4 – Závislost účinnosti a ztrátového výkonu na výstupním proudu [5]

Z obrázku E.3 je zřejmé, že velikost účinnosti je závislá na zvoleném režimu obvodu (PWM nebo BURST režim). Pro nižší proudy dosahuje vyšší účinnosti režim BURST, ale pro vyšší proudy naopak vyšší účinnosti dosahuje režim PWM. Také je patrná závislost účinnosti na velikosti vstupního napětí. Na obrázku E.4 jsou znázorněny průběhy účinnosti a ztrátového výkonu v závislosti na proudu zátěží. V režimu BURST ztrátový výkon téměř lineárně narůstá, zatímco pro režim PWM je ztrátový výkon pro proudy od 0,1 mA do 50 mA konstantní s hodnotou kolem 15 mW, pro vyšší proudy ovšem ztrátový výkon začne poměrně rychle růst.

Typické zapojení obvodu LTC3538, která uvádí výrobce, je na následujícím obrázku:



Obrázek E.5 – Ukázka zapojení obvodu LTC3538 [5]

Hodnota vstupního kondenzátoru označeného C_{IN} má mít minimální hodnotu 4,7 μF, výrobce doporučuje keramické dielektrikum tohoto kondenzátoru. Výstupní kondenzátor C_{OUT} má mít co nejmenší ztrátový odpor označený ESR. Hodnoty rezistorů R1 a R2 určují velikost výstupního napětí podle následujícího vztahu:

$$V_{OUT} = 1 \cdot \left(1 + \frac{R1}{R2}\right) [V] \tag{E.1}$$

Pozn.: výraz v závorce je násoben hodnotou 1, protože referenční napětí na vstupu FB má mít hodnotu 1 V.

Rezistory s hodnotami 15 kΩ a 10 kΩ spolu s kondenzátory s hodnotami 330 pF a 33 pF určují zpětnou vazbu chybového zesilovače z výstupu V_C a vstupu FB.

Proud I_L, na který má být dimenzována cívka zapojená mezi vývody SW1 a SW2, je závislý na aktuální konfiguraci obvodu, jestli se jedná o zvyšující nebo snižující typ měniče, proto pro proud cívkou musí platit následující podmínky:

$$I_L = I_{OUT} + \frac{\Delta I_{L,P-P}}{2} [A] \tag{E.2}$$

$$\Delta I_{L,P-P,BUCK} = \frac{V_{OUT} \cdot \frac{V_{IN} - V_{OUT}}{V_{IN}}}{f \cdot L} [A] \tag{E.3}$$

$$\Delta I_{L,P-P,BOOST} = \frac{V_{OUT} - V_{IN}}{f \cdot L} [A] \tag{E.4}$$

V_{OUT} je výstupní napětí [V],

V_{IN} je vstupní napětí [V],

$\Delta I_{L,P-P}$ je zvlnění proudu cívkou [A],

I_{OUT} je nominální hodnota výstupního proudu [A],

f je frekvence spínání [Hz] (typicky 1 MHz),

L je hodnota indukčnosti [H].

Stejnosměrný odpor cívky by měl být co možná nejmenší, aby se zamezilo ztrátám.

Pro daný výstupní kondenzátor C_{OUT} je možné určit velikost zvlnění výstupního napětí. Velikost výstupního zvlnění je opět závislá na aktuálním typu měniče (zvyšující nebo snižující):

$$\Delta V_{P-P,BUCK} = \frac{(V_{IN} - V_{OUT}) \cdot V_{OUT}}{8 \cdot L \cdot V_{IN} \cdot C_{OUT} \cdot f^2} \; [V] \tag{E.5}$$

$$\Delta V_{P-P,BOOST} = \frac{I_{OUT} \cdot (V_{OUT} - V_{IN})}{C_{OUT} \cdot V_{OUT} \cdot f} \; [V] \tag{E.6}$$

ΔV_{P-P} je velikost výstupní zvlnění [V],

L je indukčnost cívky [H],

V_{IN} je hodnota vstupního napětí [V],

V_{OUT} je hodnota výstupního napětí [V],

C_{OUT} je výstupní kapacita [F],

f je velikost spínací frekvence [Hz],

I_{OUT} je nominální výstupní proud [A].

Potřebné parametry DC/DC měniče LTC3538 shrnuje tabulka E.1:

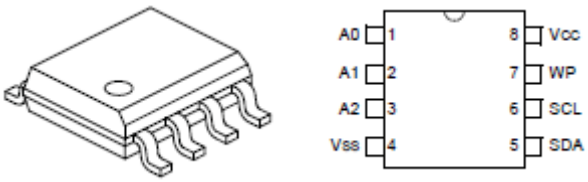
Tabulka E.1 – Přehled parametrů obvodu LTC3538 [5]

Parametr	Minimální hodnota	Typická hodnota	Maximální hodnota	Jednotka
Vstupní napětí V_{IN}	2,4		5,5	V
Referenční hodnota na pinu FB	0,98	1	1,02	V
Proud do vstupu FB		1	50	nA
Maximální vstupní prou I_{IN}	1,4	2		A
Frekvence oscilátoru	0,8	1	1,2	MHz
Čas náběhu po zapnutí		1,5		ms
Hodnota vstupu BURST pro režim BURST	1,4			V
Hodnota vstupu BURST pro režim PWM			0,4	V
Hodnota vývodu V_C pro vypnutí obvodu			0,25	V

Příloha F. Sériová paměť EEPROM

Paměť EEPROM slouží pro uložení dat, které se nesmí po odpojení napájení vymazat. Paměť je vhodná pro uložení častěji měněných dat, kvůli kterým není vhodné měnit zdrojový kód mikrokontroléru.

Pro ukládání dat byla vybrána paměť 24LC04B. Tato paměť využívá sériové rozhraní I^2C, její kapacita je 4 kb, tedy 512 B. Tato kapacita pro účely ukládání bohatě postačuje. Výrobcem paměti je společnost Microchip, celý popis paměti lze nalézt v [17]. Obvod se vyrábí v osmi vývodovém pouzdru SOIC:



Obrázek F.1 – Ukázka pouzdra obvodu 24LC04B a zapojení vývodů [17]

Tabulka F.1 – Vývody paměti 24LC04B [17]

Název vývodu	Číslo vývodu	Popis
A0	1	Není využit
A1	2	Není využit
A2	3	Není využit
V_{SS}	4	Připojení nulového napětí (GND)
SDA	5	Vstup/výstup pro datové informace
SCL	6	Vstup pro hodinový signál
WP	7	Povolení/zakázání zápisu do paměti
V_{CC}	8	Napájení napětí

Přehled důležitých parametrů obvod popisuje následující tabulka:

Tabulka F.2 – Přehled parametrů obvodu 24LC04B [17]

Název parametru	Minimální hodnota	Typická hodnota	Maximální hodnota	Jednotka
Napájecí napětí V_{CC}	2,5		5,5	V
Úroveň logické hodnoty 1 pro vstupy	$0,7 \cdot V_{CC}$			V
Úroveň logické hodnoty 0 pro vstupy			$0,3 \cdot V_{CC}$	V
Úroveň logické hodnoty 0 pro výstupy			0,4	V
Proud při operaci zápisu		0,1	3	mA
Proud při operaci čtení		0,05	1	mA
Proud v klidovém stavu		0,01	1	μA
Velikost hodinové frekvence			400	kHz
Doba zápisu			5	ms
Počet zápisů	1M			cykly

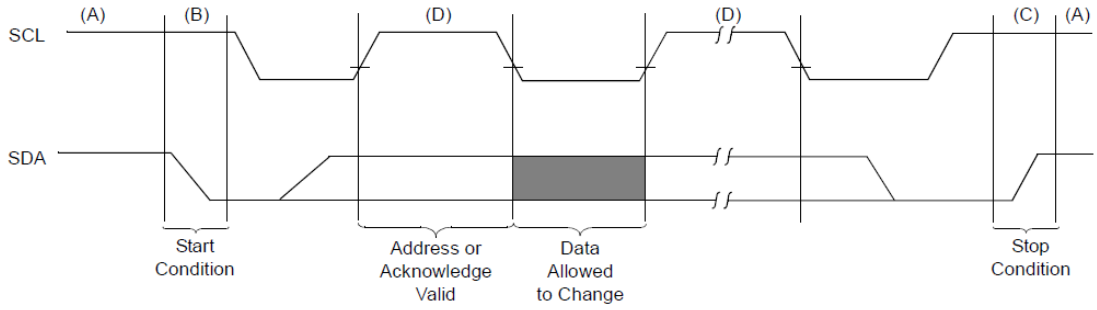
Příloha F.1 Sériová sběrnice I^2C

Sběrnice I^2C označovaná také jako IIC (Inter-Integrated Circuit) byla vyvinuta společností Philips. Převážně slouží pro připojení periferních obvodů s rychlostmi

přenosu 100 kHz, 400 kHz nebo 1 MHz. Sběrnice používá pro obousměrnou komunikaci dva vodiče, hodinový vodič je označen SCL a datový SDA. Všechny obvody, které jsou připojeny ke sběrnici I^2C, jsou vybaveny výstupy s otevřeným kolektorem na komunikačních vodičích. Toto opatření zabraňuje vzniku zkratu při nesprávných napěťových úrovních na sběrnici. Výstupy s otevřenými kolektory vyžadují tzv. pull-up rezistory připojené mezi datový vývod a napájecí napětí. Pull-up rezistory zajišťují přivedení kladné napěťové úrovně na vývod, o přivedení nulového napětí se stará vnitřní tranzistor obvodu.

Na sběrnici se mohou vyskytnout dva typy obvodů. Prvním a nejdůležitějším typem obvodu je tzv. MASTER, řídí veškerou probíhající komunikaci na sběrnici, generuje hodinový signál. Druhým typem je tzv. SLAVE přijímající hodinový signál od obvodu MASTER a na základě datových zpráv buď přijímá, nebo vysílá data po sběrnici. Dnešní obvody jsou vybaveny pomocnými obvody umožňujícími připojení i více než jednoho obvodu typu MASTER.

Sběrnice I^2C definuje několik událostí, které musejí být pro správné provedení komunikace vykonány. Každý přenos je zahájen událostí START generovanou obvodem MASTER, následuje odeslání adresy obvodu, se kterým se bude komunikovat, součástí je informace, jestli se bude do obvodu SLAVE zapisovat nebo číst, poté následují potvrzovaná data, poslední událostí je událost STOP ukončující komunikaci.



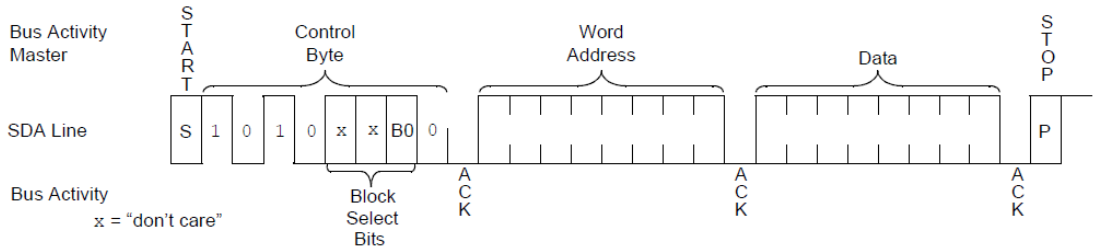
Obrázek F.2 – Detail událostí na sběrnici I^2C [17]

Událost START je signalizována jako změna z logické hodnoty 1 do hodnoty 0 na datovém vodiči SDA, hodinový vodič zůstává v hodnotě 1. Událost STOP je signalizována podobně, hodinový vodič je v hodnotě 1, ale datový vodič se změní

z hodnoty 0 na hodnotu 1. Platnost dat je určena hodnotou 1 na vodiči SCL, pokud je vodič SCL v hodnotě 0, data se mohou měnit.

Příloha F.2 Ukázky komunikace s pamětí 24LC04B

Obvod paměti EEPROM je typu SLAVE, adresa obvodu je pevně dána výrobcem. Paměť umožňuje jak čtení uložených dat, tak i zápis nových dat. Obrázek F.3 ukazuje situaci zápisu jednoho bajtu dat do paměti.



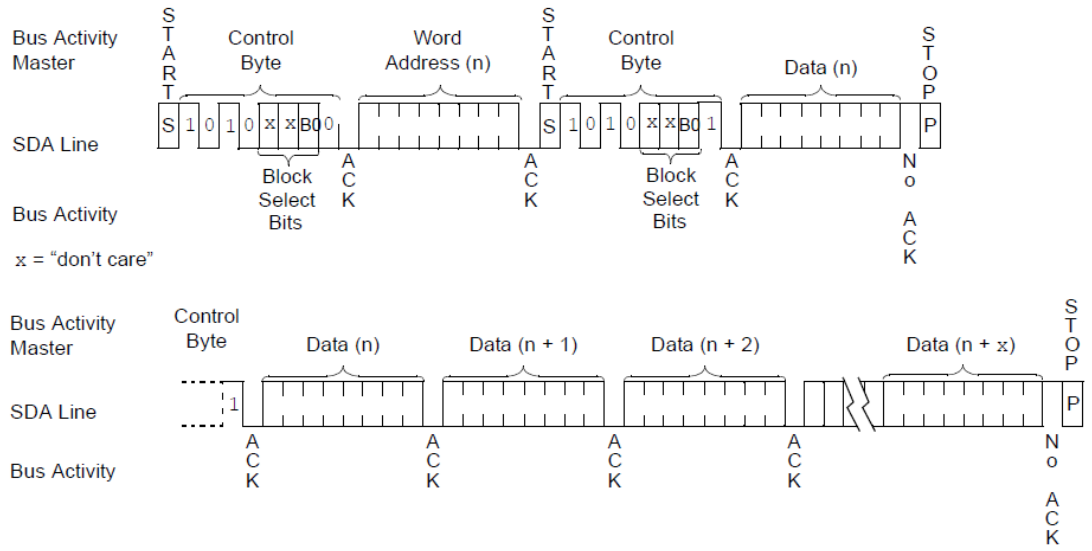
Obrázek F.3 – Události na sběrnici I^2C [17]

Vždy po odeslání osmi datových bitů následuje s devátým hodinovým pulsem potvrzení od obvodu, který data přijal, přivedením logické hodnoty 0 na datový vodič SDA. Logická hodnota 1 znamená, že data byla poškozena, nebo že už nebudou odeslána žádná data, po nepotvrzení dat musí ihned následovat událost STOP. Řídící slovo, které následuje za událostí START, obsahuje adresu obvodu ($1010xxB_0R_W$), pokud je obvod s touto adresou připojen ke sběrnici, potvrdí příjem adresy. Protože obvod umožňuje zápis nebo čtení až 512 B, tedy adresa má délku 9 bitů, je nejvyšší bit adresy umístěn do řídícího slova (bit B_0). Poslední bit R_W řídícího slova určuje směr přenosu dat, pro hodnotu 0 jsou data do paměti zapisována. Za kontrolním slovem následují zbylé bity adresy, na kterou se provede zápis, nebo ze které se bude číst. Po odeslání řídícího slova při zápisu vysílá obvod MASTER data, která jsou obvodem SLAVE potvrzována. Do paměti je možné zapsat jeden nebo až 16 bajtů (velikost vnitřního zásobníku) dat najednou. Při ukládání více dat najednou je nutné dodržet, aby rozsah zapisovaných adres byl celistvým násobkem velikosti vnitřního zásobníku.

Paměť je také vybavena ochranou proti zápisu. Pokud bude na vývod označený WP (Write Protect) přivedeno napájecí napětí (logická hodnota 1), zápis do paměti

bude zakázán. Naopak uzemnění vývodu WP (logická hodnota 0) způsobí povolení zápisu.

Čtení z paměti se musí provést ve dvou krocích. Prvním krokem je zápis adresy, ze které se budou data číst, poslední bit kontrolního slova bude roven 0. Následně se provede druhý krok začínající znovu událostí START, zápisem adresy obvodu (stejná jako v prvním kroku), poslední bit kontrolního slova bude ovšem roven 1 (čtení z paměti), následovat budou data posílaná z paměti, přijatá data jsou potvrzována obvodem MASTER. Počet čtených bajtů není omezen. Komunikaci pro čtení jednoho a více bajtů ukazuje obrázek F.4:



Obrázek F.4 – Čtení z paměti 24LC04B [17]

Za posledním přečteným bajtem vždy MASTER neodešle potvrzení příjmu dat. Tím dává obvodu SLAVE (paměti) najevo, že komunikace bude v co nejbližší době ukončena.

Příloha G. Bezdrátový komunikační modul

Pro odesílání bezdrátových zpráv slouží komunikační modul. Tento modul musí pracovat v pásmu, pro které není nutná licence. Nejznámější pásma používaná pro komunikaci jsou na frekvencích 434 MHz, 868 MHz a 2,4 GHz. Frekvence 434 MHz se již v dnešní době přestává používat, dnes nejvíce využívaným volným pásmem je 868 MHz.

Pro komunikaci s centrální jednotkou byly vybrány moduly IQRF od společnosti Microrisc. Tyto moduly se vyznačují rychlostí vývoje bezdrátové komunikace, obsahují totiž vlastní operační systém zprostředkující rozhraní mezi uživatelskou aplikací uloženou v bezdrátovém modulu a komunikačním obvodem umístěným na desce bezdrátového modulu. Nespornou výhodou bezdrátových obvodů IQRF je také velký dosah vysílaného signálu. Bezdrátové moduly jsou velikostně konstruovány tak, aby byla možná instalace do držáčku pro SIM kartu mobilního telefonu. Celou bezdrátovou platformu IQRF lze nalézt na [7]. Platforma IQRF umožňuje vytvoření několika typů bezdrátových sítí. Lze vytvořit spojení typu „point to point“, tedy „každý s každým“, nebo lze vytvořit bezdrátovou síť IQMESH s jedním centrálním prvkem.

Společnost Microrisc nabízí ke svým modulům IQRF širokou základnu vývojových prostředků nebo již hotových výrobků. Všechny produkty jsou založeny na mikrokontrolérech PIC společnosti Microchip. Lze si zakoupit programátor pro IQRF moduly, malé moduly pro testování s vlastní baterií apod. až po zařízení s dotykovým LCD displejem, který má osazen bezdrátový IQRF modul, SD kartu, apod., nebo s ethernetový rozhraním.

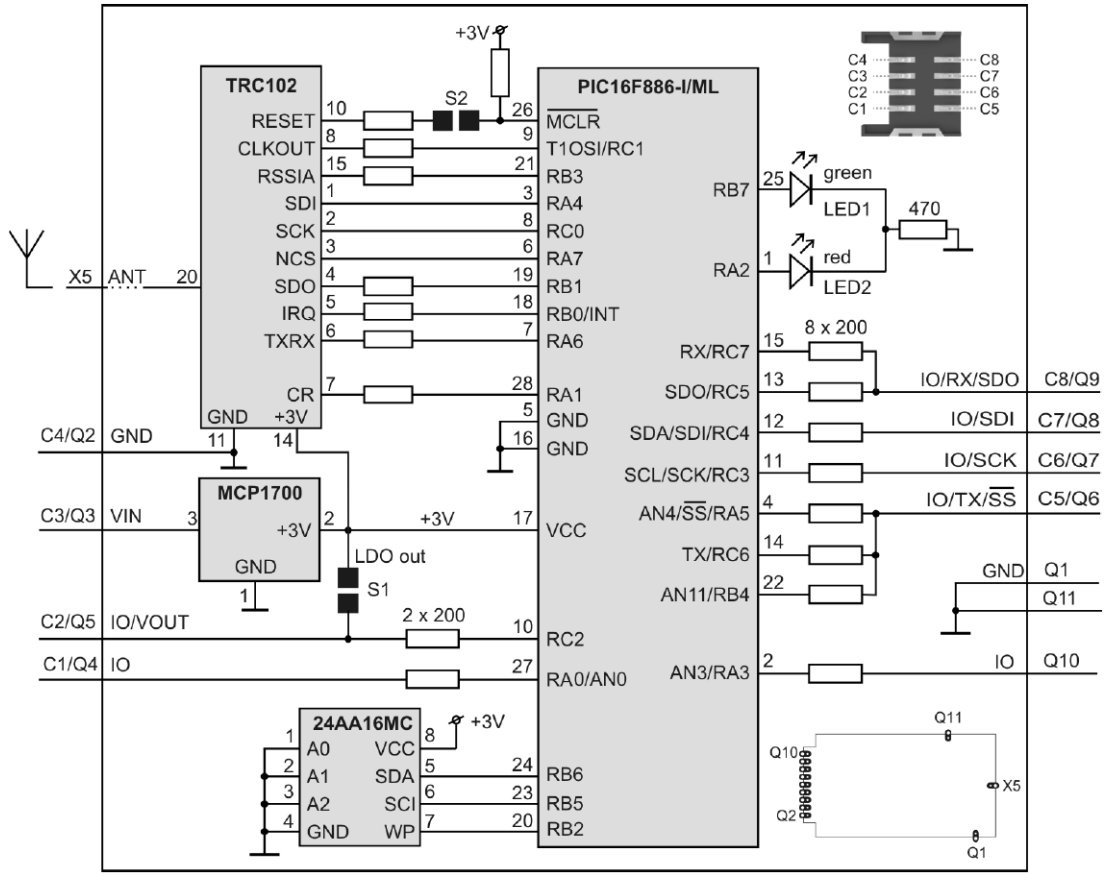
Příloha G.1 Bezdrátový komunikační modul TR-53B

Komunikační bezdrátový modul TR-53B, který je z nové řady IQRF modulů, obsahuje nový operační systém verze v3.00 s řadou nových vylepšení, např. zvětšení adresového prostoru na 65000 prvků v bezdrátové síti, inteligentní tvorbou topologie sítě (dříve musela být topologie sítě předem známa a zadávaná v uživatelské aplikaci), apod. Další výraznou změnou je umožnění montáže přímo na desku plošného spoje, tím se omezí možnost oxidace kontaktů v držáku SIM karty a tím dojde ke zvýšení bezpečnosti provozu. Veškeré informace lze najít na [8].

Obrázek G.1 – Ukázka bezdrátového modulu TR-53B [8]

Komunikační modul má vyvedenu řadu vývodů, které jsou rozmístěny po obvodu (lze jimi zapájet desku modulu na desku plošného spoje), obsahující 7 vstupně/výstupních portů nebo 3 A/D vstupy. Modul je schopen komunikovat na frekvencích 868 MHz (pro EU) nebo 916 MHz (pro USA). Spotřeba se pohybuje okolo 17 µA až 1 mA, v režimu spánku pouhé 2 µA, při příjmu je spotřeba kolem 13 mA a při vysílání 14 mA až 24 mA v závislosti na vysílacím výkonu. Napájecí napětí je v rozsahu 3,1 V až 5,3 V. Rychlost přenosu lze nastavit z těchto možností 1,2 kb/s, 19,2 kb/s, 57,6 kb/s, 86,2 kb/s, doporučovaná rychlost je 19,2 kb/s.

Funkce modulu je řízená mikrokontrolérem PIC16F886, bezdrátovou část modulu obsluhuje obvod TRC102 společnosti RFM, obvod využívá FSK modulaci pro vysílání dat. Mikrokontrolér obsahuje celý operační systém, zároveň je možné do mikrokontroléru naprogramovat vlastní aplikaci, bez které by modul nepracoval. Vlastní aplikace využívá funkce operačního systému pro docílení výsledné bezdrátové aplikace. Operační systém do jisté míry umožňuje ovládat periferie mikrokontroléru, např. ovládání vstupně/výstupních portů, A/D převodník apod. Modul je také osazen dvěma LED diodami, které lze využít pro signalizaci různých stavů. Na modulu je přítomna také paměť EEPROM, umožňující ukládání uživatelských dat, pokud je ovšem modul použit jako centrální prvek sítě IQMESH, je paměť využita operačním systém pro uložení adres jednotlivých uzlů sítě.



Obrázek G.2 – Blokové schéma bezdrátového modulu TR-53B [9]

Příloha G.2 Popis paketu platformy IQRF

Popis celkové komunikace společnost Microrisc nezveřejňuje. Jediné, co lze o struktuře bezdrátové komunikace nalézt, je podoba komunikačního paketu, který modul vysílá a přijímá. Existují celkem dvě podoby tohoto paketu. Prvním typem je tzv. nesíťový paket, tento paket nemá žádného adresáta ani příjemce (síť „point to point“). Druhým typem je síťový paket, ten je moduly vysílán, pokud jsou zařazeny do některé sítě a komunikují v rámci této sítě. Protože senzorové jednotky budou mezi sebou a centrální jednotkou komunikovat prostřednictvím nesíťových paketů, bude detailně popsán pouze nesíťový paket. Podrobný popis paketů lze nalézt v [14].

PIN	*DLEN*	*CRCH*	*DATA*	*CRCD*	*CRCS*

Obrázek G.3 – Detail nesíťového paketu

- *PIN* – obsahuje informace o paketu, význam má pouze nejvyšší bit (roven 0)
- *DLEN* – délka pole *DATA*, povolený rozsah 1 až 64bajtů

- *CRCH* – kontrolní součet pro *PIN* a *DLEN*
- *DATA* – data obsažená v paketu
- *CRCD* – kontrolní součet pro pole *DATA*
- *CRCS* – kontrolní součet celého paketu

Pro síťový paket je pouze přidáno další pole obsahující informace o síti, adresu sítě, adresu cílového a zdrojového zařízení paketu apod. Pro síťový paket je nejvyšší bit pole *PIN* roven 1.

Pole *PIN* a *DLEN* jsou reprezentovány v operačním systému paměťovými místy se stejnými názvy umožňujícími zápis nebo čtení. Proto před odesláním paketu je nutné tato pole správně nastavit a po přijetí paketu tato pole obsahují přijaté informace. Pole *DATA* jsou v operačním systému reprezentováno jako pole s názvem *bufferRF* (bližší informace budou popsány v následující části textu).

Příloha G.3 Operační systém platformy IQRF

Popis operačního systému lze najít na [10]. Operační systém, který je součástí mikrokontroléru PIC16F886 bezdrátového modulu IQRF, v současné verzi v3.00, nabízí veškeré funkce pro ovládání bezdrátové části modulu. Součástí systému jsou některé funkce běžící na pozadí všech činností. Mezi tyto funkce např. patří funkce pro měření času, zpoždění, nebo pro komunikaci po SPI sběrnici s řídícím mikrokontrolérem senzorové jednotky. Sběrnice SPI je přímo podporována operačním systémem se zvláštním komunikačním protokolem, při programování uživatelské aplikace se využívá také tato sběrnice. Systémovou sběrnici SPI lze vyřadit a naprogramovat si vlastní komunikaci, např. UART nebo I^2C. Mezi podporované funkce patří např. i funkce pro ovládání signalizačních LED diod bezdrátového modulu.

Operační systém nedovoluje používání pointerů, tedy přímého přístupu do libovolné paměti mikrokontroléru (např. procházení pole dat v cyklu, konstanta jako index do pole je povolena, ale proměnný index není přípustný), proto si operační systém při programování uživatelské aplikace kontroluje používané instrukce a objeví-li použití registrů umožňující přímý přístup do paměti pomocí pointerů, dané instrukce vynechá, což může způsobit nefunkčnost navržené aplikace. Pro možnost

zápisu dat do paměti jsou připraveny funkce kontrolující paměťový rozsah, do kterého se provádí zápis nebo čtení, pokud bude rozsah správný, zápis nebo čtení bude provedeno.

Uživatelskou aplikaci, bez které IQRF modul nevykonává žádnou činnost, lze do modulu naprogramovat pomocí systémové sběrnice SPI prostřednictvím programátoru CK-USB-02 [12], CK-USB-04 [13] nebo od verze v3.00 operačního systému lze modul naprogramovat i bezdrátově. Uživatelská aplikace resp. funkce musí být pojmenována jako *APPLICATION()*. Společnost Microrisc poskytuje řadu softwarových vývojových prostředků. Nejdůležitějším je vývojové prostředí IQRF IDE zprostředkující programování a ladění. Pokud IQRF modul umožňuje komunikaci prostřednictvím SPI sběrnice, IQRF IDE tuto komunikaci pomůže zprostředkovat. Společně s prostředím IQRF IDE lze na stránkách výrobce najít ukázkové aplikace. Uživatelskou aplikaci lze vyvíjet ve vyšším programovacím jazyce C, pro překlad navržené aplikace lze využít překladač CC5X. Společnost Microrisc tento překladač přímo doporučuje a lze ho stáhnout ze stránek platformy IQRF, bohužel se jedná pouze o neplnou verzi překladače nedovolující některé základní syntaxe jazyka C, např. inicializace globálních proměnných. Do uživatelské aplikace musí být vložen hlavičkový soubor *template-basic.h* (součást ukázkových příkladů), tento soubor zajišťuje překlad kódu spouštějícího uživatelskou aplikaci, dále vkládá další soubor *IQRF-functions-???.h* (kde *???* je nahrazen typem modulu např. *52B* – pro moduly TR-52B a TR-53B). Tento soubor obsahuje deklarace funkcí operačního systému s adresami, kde jsou v paměti mikrokontroléru uloženy.

Operační systém poskytuje přístup k několika systémovým proměnným např. již zmíněné proměnné *PIN*, *DLEN* apod. Přehled důležitých systémových proměnných poskytuje následující přehled:

- *PIN* – informace o vysílaném/přijatém paketu (viz výše),
- *DLEN* – délka datové části paketu (viz výše),
- *bufferRF* – pole dat určené pro bezdrátovou komunikaci,
- *bufferCOM* – pole dat pro komunikaci po SPI sběrnici,
- *param1* – návratová proměnná některých funkcí,
- *toutRF* – nastavení doby čekání na příjem nového paketu.

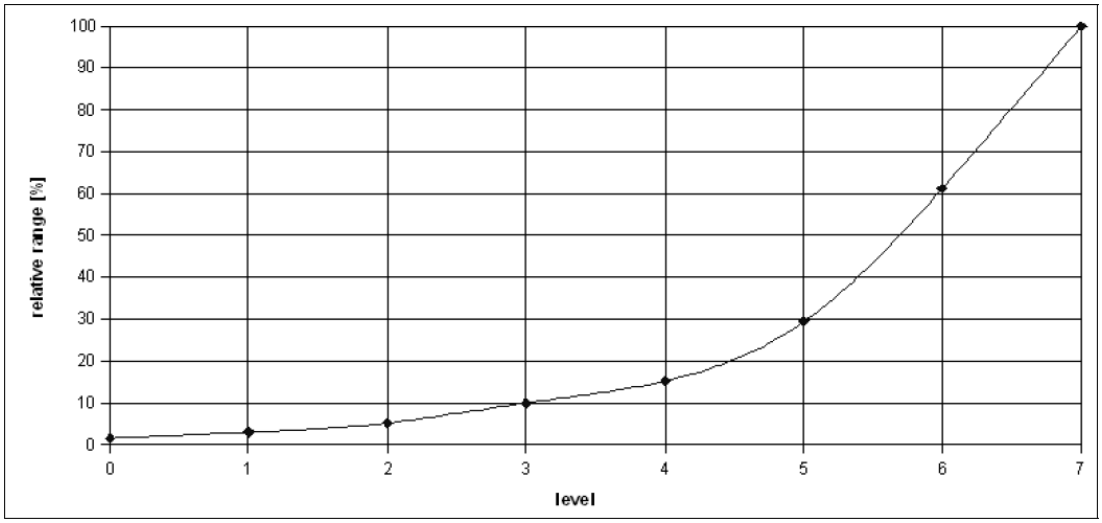
Důležité funkce operačního systému shrnuje tabulka G.1:

Tabulka G.1 – Přehled důležitých funkcí operačního systému platformy IQRF [10]

Název funkce	**Popis funkce**
enableSPI()	Povolení systémové komunikace po SPI sběrnici
startSPI(length)	Zahájí odesílání dat z *bufferCOM* o celkové velikosti *length*
bit getStatusSPI()	Vrací aktuální stav SPI sběrnice, sběrnice je zaneprázdněná nebo není. Proměnná *_SPIRX* je nastavena pokud byla přijata nová data, *_SPICRCok* je nastavena v případě správně přijatého CRC, proměnná *param1* obsahuje počet přijatých bajtů.
clrwdt()	Vynulování Watchdog časovače
void setTXpower(level)	Nastavení vysílacího výkonu viz obrázek 38.
setRFchannel(channel)	Nastavení kanálu, na kterém bude modul komunikovat
bit checkRF(level)	Provede kontrolu zarušení, resp. kontrolu komunikace jiného zařízení, na nastaveném kanálu s určitou mírou signálu (parametr *level*)
void RFTXpacket()	Odešle připravená data v poli *bufferRF*
bit RFRXpacket()	Přijetí nového paketu. Proměnná *toutRF* určuje dobu (násobky 10 ms) čekání na příjem nového paketu.
uns8 readFromRAM(addr)	Čtení z dané adresy *addr* z paměti RAM, lze použít např. pro proměnný index pole.
void writeToRAM(addr, value)	Umožní zápis na adresu *addr* hodnotu *value*, náhrada např. za proměnný index pole
copyMemoryBlock(from, to, length)	Provede kopírování z adresy *from* na adresu *to* s délkou dat *length*
startDelay(ticks)	Spustí časovač na pozadí, s dobou trvání *ticks*·10 ms

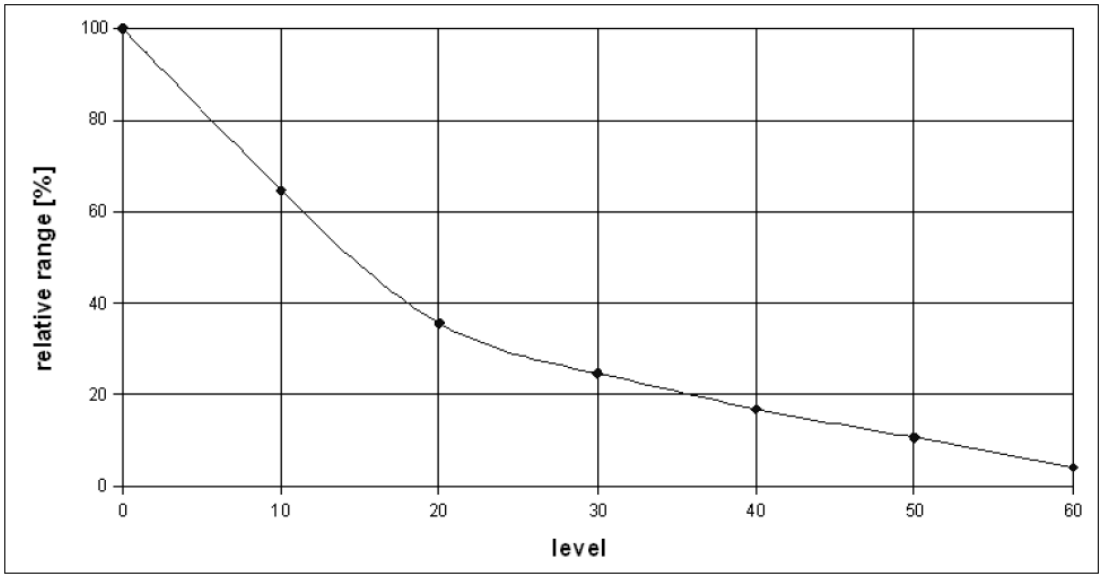
bit isDelay()	Vrací informaci o stavu časovače nastaveného funkcí *startDelay*, pokud časovač stále běží, vrací hodnotu 1
waitDelay(ticks)	Zastaví běh hlavního programu na dobu *ticks*·10 ms
iqrfSleep()	Uvede modul do režimu spánku - minimální odběr celého modulu.

Parametry funkcí *setTXpower* a *checkRF* nejsou snadno určitelné. Jejich hodnoty nemají lineární závislost např. mezi hodnotou parametru a vysílacím výkonem. Proto lze v popisu k danému IQRF modulu nalézt průběhy těchto závislostí:



Obrázek G.4 – Závislost parametru funkce setTXpower [9]

Předcházející obrázek ukazuje závislost vysílacího výkonu IQRF modulu na hodnotě parametru funkce *setTXpower*.



Obrázek G.5 – Závislost parametru funkce checkRF [9]

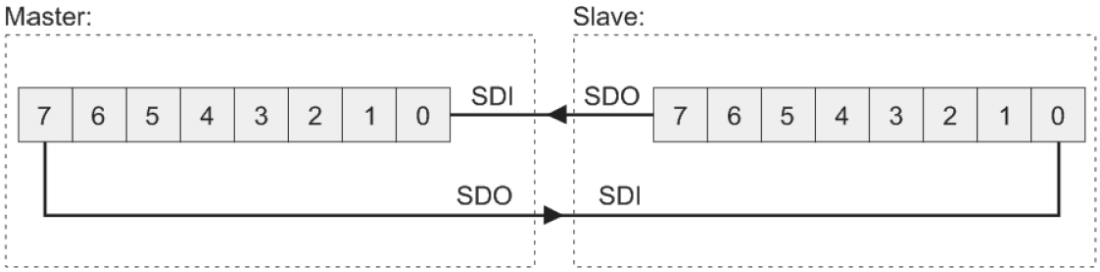
Obrázek G.5 ukazuje závislost parametru funkce *checkRF* na hodnotě okolního zarušení.

Příloha G.4 SPI sběrnice

Sběrnice SPI (Seriál Peripheral Interface) je obousměrná synchronní sériová sběrnice určená ke komunikaci řídícího obvodu, např. mikrokontroléru a periferního obvodu, např. A/D převodník apod. Na sběrnici mohou být podobně jako u sběrnice I^2C umístěny dva typy obvodu, MASTER a SLAVE. Obvod typu MASTER může být na sběrnici přítomen pouze jeden. Obvody mezi sebou komunikují pomocí čtveřice vodičů, vodič SCK představuje hodinový signál, vodič SDO jsou výstupní data, vodič SDI jsou vstupní data a vodič SS (Slave Select) nebo CS (Chip Select) vybírá obvod typu SLAVE, se kterým bude prováděna komunikace. Obvod typu MASTER během komunikace generuje hodinový signál, nastavuje výstupní data a vybírá obvod SLAVE.

Před zahájením komunikace vybere MASTER obvod SLAVE správným nastavením daného vodiče SS, k obvodu MASTER může být připojeno několik obvodů SLAVE, obvykle má vodič SS invertující smysl, tedy obvod se vybere vynulováním vodiče SS. Poté začne generovat hodinový signál a před danou hranou hodinového signálu musí nastavit platná data na vodiči SDO (data mohou být platná jak na sestupnou nebo vzestupnou hranu hodinového signálu), obvod SLAVE musí

před hranou hodinového signálu nastavit platná data na vodiči SDI. Vnitřní datové bloky obvodů MASTER i SLAVE fungují jako posuvné registry, viz obrázek G.6. Obvody MASTER i SLAVE mohou nastavovat data od nejvyššího významného bitu MSB nebo od nejnižšího významného bitu LSB, délka datového slova může být libovolně dlouhá. Obě strany datového řetězce (MASTER i SLAVE) musejí být nastaveny identicky.

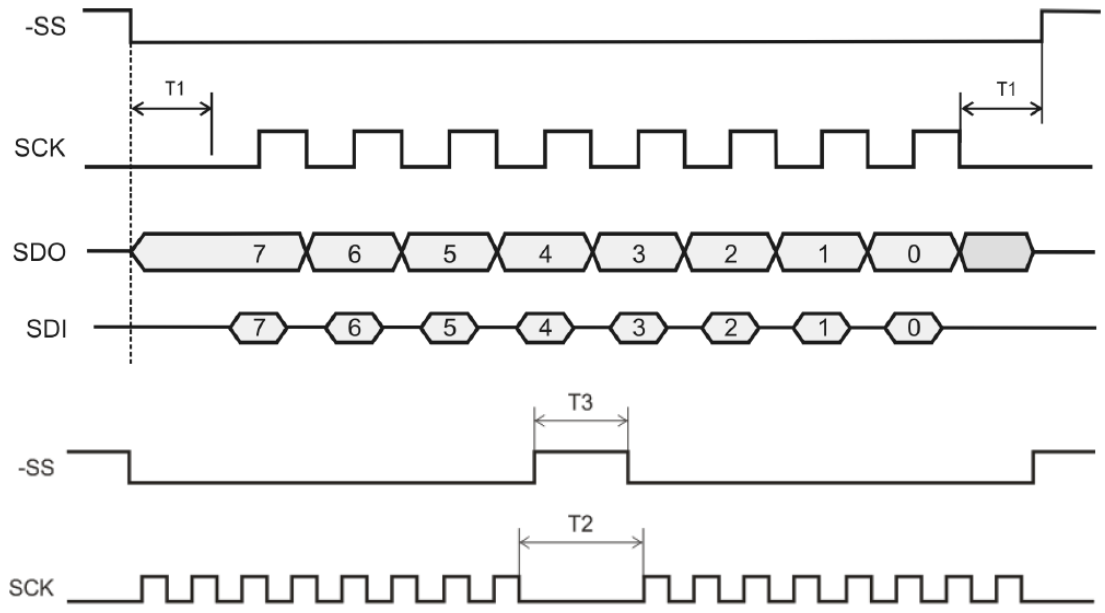


Obrázek G.6 – Datové pochody na SPI sběrnici [10]

Obrázek G.6 ukazuje situaci, kdy jsou data vysílána od nejvyššího významného bitu MSB, délka slova je osm bitů. Obvod MASTER nastaví datový bit na vodič SDO, po hodinovém signálu se zapíše hodnota z vývodu SDI obvodu MASTER na pozici nejnižšího bitu LSB, obvod SLAVE zapíše hodnotu ze svého datového vodiče SDI také na pozici nejnižšího bitu LSB, poté se oba paměťové bloky (jak v obvodu MASTER tak i SLAVE) o jeden bit posunou doleva. Po osmi hodinových impulsech se tímto způsobem odešlou všechny datové bity a data z obvodu SLAVE se přenesou do obvodu MASTER a naopak. Data se tedy přenášejí obousměrně najednou.

Příloha G.5 Systémová SPI sběrnice platformy IQRF

Operační systém modulů IQRF podporuje komunikaci prostřednictvím SPI sběrnice popsané v předchozím textu. Komunikace i časování sběrnice je přesně definováno. Data jsou platná pro náběžnou hranu hodinového signálu, obvod SLAVE je vybrán vynulování vodiče SS, data jsou vysílána od bitu MSB, délka slova je osm bitů. Časování sběrnice popisuje následující obrázek:



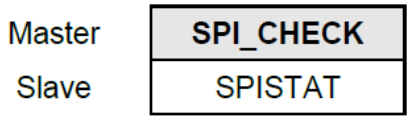
Obrázek G.7 – Časování SPI sběrnice platformy IQRF [11]

Pro správnou komunikaci mezi řídícím mikrokontrolérem senzorové jednotky a IQRF modulem musejí být dodrženy časy mezi vybráním obvodu (SS = 0) a generováním hodinového signálu, mezi jednotlivými datovými slovy nebo frekvenci hodinového signálu. Všechny tyto parametry shrnuje tabulka G.2:

Tabulka G.2 – Přehled parametrů časování SPI sběrnice platformy IQRF [11]

Označení	Popis	Hodnota	Tolerance
SCK	Frekvence hodinového signálu	250 kHz	max.
T1	Doba mezi změnou SS a generováním hodinového signálu	10 µs	min.
T2	Doba mezi datovými slovy	100 µs	min.
T3	Doba mezi ukončením a zahájením komunikace (změny vodiče SS)	20 µs	min.

Komunikační protokol SPI sběrnice definuje dva typy komunikace. Prvním je zjištění stavu IQRF modulu, druhým je přenos dat. Zjištění stavu je provedeno odesláním jednoho bajtu, který se označuje SPI_CHECK (hodnota 00h. – h. představuje hexadecimální soustavu), modul IQRF vždy na začátku komunikace odesílá svůj stav označovaný jako SPISTAT, SPI_CHECK pouze informuje IQRF o tom, že se nebudou odesílat data. Tuto situaci vystihuje následující obrázek a tabulka s přehledem stavů IQRF modulu.



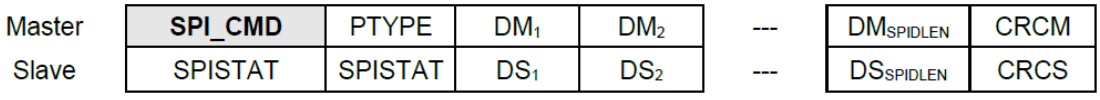
Obrázek G.8 – Komunikační přenos pro zjištění stavu IQRF modulu [11]

Tabulka G.3 – Přehled stavů SPI sběrnice platformy IQRF [11]

Hodnota SPISTAT (hex. soustava)	**Popis**
00h.	SPI není aktivní (funkce *disableSPI*)
07h.	SPI je pozastaveno (funkce *stopSPI*)
3Fh.	SPI není připraveno, poslední přijatá data nebyla doposud zpracována (kontrolní součet CRCM byl v pořádku)
3Eh.	SPI není připraveno, poslední přijatá data nebyla doposud zpracována (kontrolní součet CRCM nebyl v pořádku)
41h. až 40h.+N_{max}	SPI je připraveno a jsou připravena data k odeslání z IQRF modulu
80h.	SPI je připraveno (komunikační mód)
81h.	SPI je připraveno (programovací mód)
82h.	SPI je připraveno (ladící mód)
FFh.	SPI není aktivní (hardwarová chyba)

Na základě obdrženého stavu SPI sběrnice, resp. stavu zpracování dat IQRF modulu lze usoudit schopnost přijímání nebo odesílání dat IQRF modulem. Zároveň lze vyhodnotit správnost přijetí posledních dat pomocí kontrolního součtu popsaného v následujícím textu. Bude-li IQRF modul schopen přijímat data (vrací hodnotu SPISTAT 80h.), lze mu data odeslat, nebo pokud bude obsahovat nějaká data k odeslání (hodnota SPISTAT 40h. až 40h.+N_{max} – N_{max} je maximálně roven 41), lze prostřednictvím jednoho datového přenosu data odeslat i data přijmout.

Pro odeslání nebo příjem dat prostřednictvím SPI sběrnice je struktura datového přenosu jiná. Místo odeslání SPI_CHECK je odesláno SPI_CMD (hodnota F0h.), následuje PTYPE, za kterým jsou zařazena data, na konci je kontrolní součet. Komunikační přenos je znázorněn na obrázku G.9.



Obrázek G.9 – Komunikační přenos pro odeslání dat pro IQRF modul [11]

Pole PTYPE má následující strukturu:

b7	b6	b5	b4	b3	b2	b1	b0
CTYPE		SPIDLEN					

Obrázek G.10 – Struktura pole PTYPE [11]

Pole SPIDLEN určuje počet přenášený datových bajtů (maximální hodnota je 41), CTYPE určuje směr přenášených dat. Pro CTYPE = 00 se data přenášejí z IQRF modulu do řídícího mikrokontroléru (pole *bufferCOM* operačního systému IQRF nebude změněn), CTYPE = 10 určuje přenos do IQRF modulu nebo z a do IQRF najednou (pole *bufferCOM* bude změněno).

Komunikační přenosy jsou zakončeny kontrolními součty, které se vypočítají podle následujícího postupu. Kontrolní součty slouží pro zjištění, zda byla data odeslána nebo přijata v pořádku:

CRCM SPI_CMD xor PTYPE xor DM_1 xor DM_2 … xor $DM_{SPIDLEN}$ xor 5Fh.

CRCS PTYPE xor DS_1 xor DS_2 … xor $DS_{SPIDLEN}$ xor 5Fh.

Příloha H. Popis mikrokontrolér LM3S811

Mikrokontrolér LM3S811 patří do rodiny Stellaris společnosti Texas Instruments. Tento mikrokontrolér je vybaven novým typem jádra Cortex-M3, který je založen na jádře ARMv7, popis jádra Cortex-M3 lze nalézt v [18] a [19]. Jádro je určeno pro náročnější automatizační a průmyslové řídící systémy s přihlédnutím pro vestavěné aplikace. Jádro podporuje instrukční sadu Trumb používánou i u jader ARMv7, nově podporuje také instrukční sadu Trumb-2 umožňující zvýšení výkonu na 1 MHz frekvence procesoru až o 70 %, navíc se dosahuje zvýšení výkonu o 35 % než u shodných procesorů. Procesory dosahují s použitím instrukční sady Trumb-2 zpracování počtu instrukcí až 1,25 MIPS / MHz.

Jádro Cortex-M3 bylo také vyvinuto s cílem zajistit rychlé vytvoření cílové aplikace bez použití kódu symbolických adres – assembleru. Zpracování instrukcí se provádí pomocí tří stupňovité pipeline skládající se z části fetch (vyzvednutí instrukce z paměti), decode (dekódování instrukce) a z části execute (vykonání instrukce).

Mikrokrontrolér LM3S811 dokáže pracovat na hodinové frekvenci až 50 MHz, pro vytvoření vhodné hodinové frekvence lze využít vnitřní násobičku kmitočtu PLL, nominální hodnota napájecího napětí je 3,3 V, obsahuje řadu periferních obvodů, např. I^2C, SSI, A/D převodník, Watchdog, čtyři časovače, PWM, UART, … Velikost paměti programu flash je 64 kB, datová paměť RAM má velikost 8 kB. Veškeré informace lze najít v [15].

Pro usnadnění vytvoření cílové aplikace poskytuje společnost Texas Instruments programátorskou knihovnu StellarisWare obsahující dílčí knihovny - Peripheral Driver Library usnadňující přístup k periferiím, Graphics Library usnadňující zobrazení grafických prvků na displejích a USB Library pro přístup k USB sběrnici. K jednotlivým periferiím lze pochopitelně přistupovat i přímo prostřednictvím patřičných registrů. Tento způsob je z pohledu vykonávání strojového kódu rychlejší, jedná se pouze o instrukce čtení nebo zápisu, zatímco v případě použití knihoven dojde k nastavení periferie za delší dobu, protože každá funkce je volána jako podprogram a provádí kontrolu zadaných parametrů.

Pro mikrokontroléry vybavené jádrem Cortex-M3 existuje řada vývojových prostředků např. IAR Embedded Workbench od společnosti IAR Systems, Code Composer Studio od společnosti Texas Instruments nebo µVision od společnosti KEIL. Tato prostředí jsou většinou volně dostupná s jistými omezeními, ve většině případů se jedná o omezení velikosti zdrojového kódu na 32 kB. Mikrokontroléry s jádry ARM, tedy i s jádry Cortex-M3, se dají programovat prostřednictvím rozhraní JTAG, proto všechna předcházející vývojová prostředí umožňují programování pomocí tohoto rozhraní. Přes rozhraní JTAG je možný i tzv. On-Chip Debugging, tedy ladění programu přímo v mikrokontroléru.

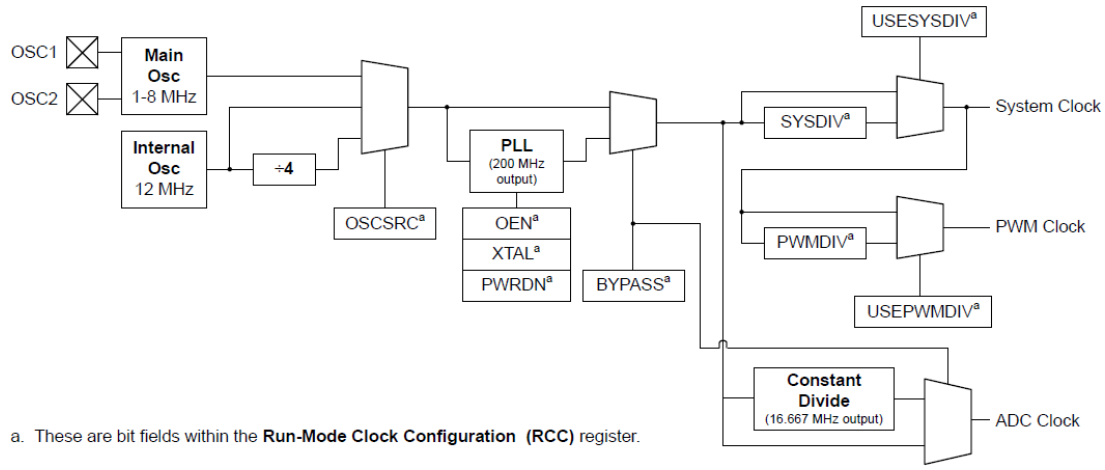
Následující podkapitoly budou věnovány periferiím použitými pro realizaci senzorové jednotky.

Příloha H.1 Nastavení hodinového kmitočtu

Mikrokontrolér LM3S811 disponuje širokou možností nastavení vhodného kmitočtu mikrokontroléru. Každou periferii je nutné před použitím povolit a připojit k hodinovému signálu, připojení nebo odpojení hodinového signálu má vliv na

výslednou proudovou spotřebu mikrokontroléru. K povolení hodinového signálu pro danou periferii slouží registry RCGC0 (Run Mode Clock Gating Control Register 0) až RCGC2, viz dokumentace k mikrokontroléru.

Nastavení zdroje hodinového kmitočtu se provádí prostřednictvím registru RCC (Run-Mode Clock Configuration). Možnosti nastavení zdroje hodinového kmitočtu popisuje obrázek H.1:



Obrázek H.1 – Blokové schéma nastavení zdroje hodinového kmitočtu [15]

Bity OSCSRC vybírají zdroj hodinového kmitočtu, lze vybrat mezi třemi zdroji, jedním je hlavní krystalový oscilátor, dalšími zdroji je interní zdroj 12 MHz a 3 MHz (12 MHz vydělených 4). Vnitřní násobička kmitočtu PLL vytváří frekvenci o hodnotě 200 MHz, aby bylo možné přesné nastavení kmitočtu, je pomocí bitů XTAL nastavena hodnota odpovídající vstupní frekvenci násobičky (krystalového oscilátoru), proto není možno připojit libovolnou hodnotu krystalového oscilátoru. Bit OEN povoluje násobičku kmitočtu a bit BYPASS vybírá zdroj hodinového signálu pro další obvody, lze vybrat násobičku kmitočtu nebo přímý zdroj vybraný bity OSCSRC. Pro periferní obvody a pro samotné jádro mikrokontroléru představuje zdroj hodinového signálu vodič označený jako System Clock. Bit USESYSDIV vybírá zdroj hodin buď z výstupu vybraného bitem BYPASS, nebo z výstupu děličky SYSDIV umožňující nastavení menší frekvence.

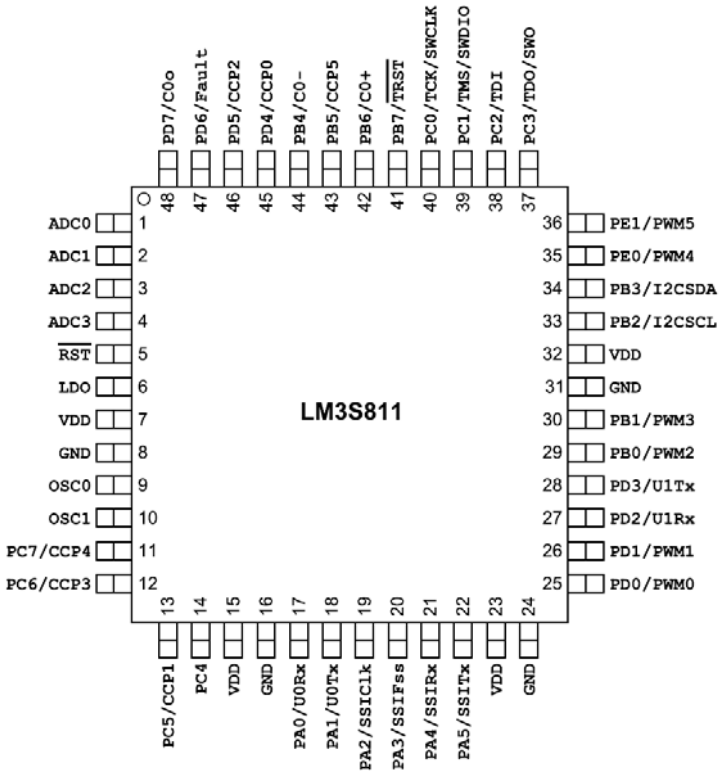
Zdroj hodinového signálu pro A/D převodník je vybírán také bitem BYPASS. Zdroj může být tvořen frekvencí 16,667 MHz vydělenou z 200 MHz násobičky PLL,

tento zdroj je vybrán, je-li vybrána násobička kmitočtu jako zdroj hodinového kmitočtu. Druhý zdroj je tvořen výstupem vybraným přímo bitem BYPASS.

Příloha H.2 Vstupně výstupní porty

Každý mikrokontrolér je vybaven vstupně-výstupními porty představujícími rozhraní mezi mikrokontrolérem a okolními obvody. Mikrokontrolér LM3S811 je vybaven celkem pěti porty (port A až E) s celkovým počtem 32 vstupně-výstupních vývodů.

Na některé vývody jsou vyvedeny vývody vnějších periferií, např. pro sběrnici I^2C, UART nebo JTAG. Vstupy pro A/D převodník jsou vyvedeny na čtyři oddělené vývody ADC0 až ADC3. Zapojení pouzdra mikrokontroléru LM3S811 spolu s popsanými vývody ukazuje obrázek H.2:

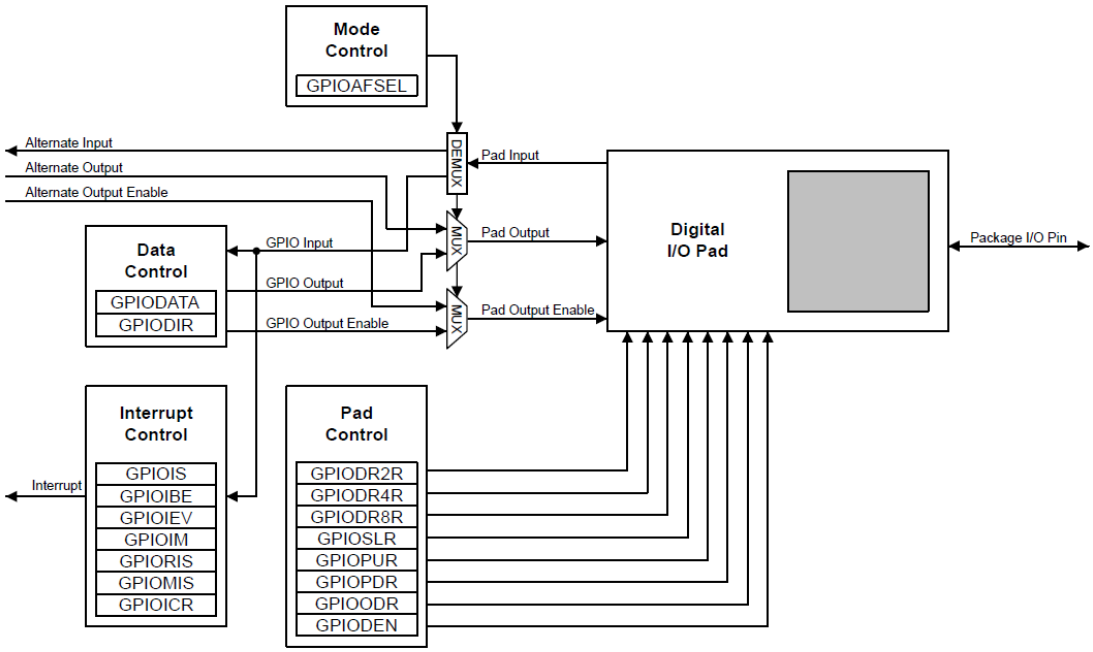


Obrázek H.2 – Umístění vstupně-výstupních portů na pouzdře obvodu LM3S811 [15]

Na vývod označený LDO je vyvedeno napětí z interního stabilizátoru napětí. Toto napětí je použito pro napájení samotného jádra mikrokontroléra, některé typy mikrokontrolérů z rodiny Stellaris mají vyvedeno napájecí napětí pro jádro na samostatné vývody, které musejí být spojeny s výstupem LDO. Tento mikrokontrolér

má napájecí napětí pro jádro přivedeno interně. Vývod pouze slouží pro filtraci napětí pomocí kondenzátoru. Výrobce doporučuje k vývodu LDO připojit kapacity s hodnotami 1 μF a 100 nF. Vývody OSC0 a OSC1 slouží pro připojení externího krystalového oscilátoru. Tyto vývody mají být připojeny kromě k oscilátoru také přes kondenzátory s kapacitou 18 pF na zem. Vývod označený #RTS (znak # označuje opačný smysl hodnoty vývodu, vývod je aktivní v logické hodnotě nula) slouží jako reset mikrokontroléru. Vývod PB7/#TRST se doporučuje přivést přes rezistor na napájecí napětí.

Mikrokontrolér LM3S811 je vybaven širokými možnostmi nastavení vstupně-výstupních portů. Lze nastavit směr portu (registr GPIODIR), buď bude konfigurován jako vstup nebo jako výstup, lze nastavit způsob přerušení na změnu hodnoty vývodu (pouze jako vstup). Přerušení lze nastavit na změnu vývodu, na vzestupnou nebo sestupnou hranu nebo na obě hrany, nebo na úroveň (registry GPIOIS, GPIOIBE, GPIOIEV), každý vstupně-výstupní port má k dispozici vlastní vektor přerušení. Hodnoty výstupních vývodů portu určuje registr GPIODATA. Pro vývod, který lze nastavit jako vstupně-výstupní nebo jako vývod určité periferie, určuje registr GPIOAFSEL to, ke které periferii bude vývod připojen (vstupně-výstupní port nebo speciální periferie). Pro jednotlivé vývody lze nastavit maximální proud z vývodu nakonfigurovaného jako výstup prostřednictvím registrů GPIODR2R (pro proud 2 mA), GPIODR4R (pro proud 4 mA) a GPIODR8R (pro proud 8 mA). Vývody lze také konfigurovat jako výstupy s otevřeným kolektorem pomocí registru GPIOODR. Vstupní piny lze opatřit vnitřními pull-up rezistory pomocí registru GPIOPUR nebo vnitřními pull-dowm rezistory pomocí registru GPIOPDR. Pro analogové vstupy A/D převodníku (u tohoto mikrokontroléru nevyužito) nebo u analogového komparátoru, se funkce vývodu (analogový nebo digitální) nastavuje registrem GPIODEN. Blokové schéma vstupně-výstupního portu ukazuje následující obrázek:



Obrázek H.3 – Blokové schéma vstupně-výstupního portu [15]

Důležité parametry vstupně-výstupních portů shrnuje tabulka H.1:

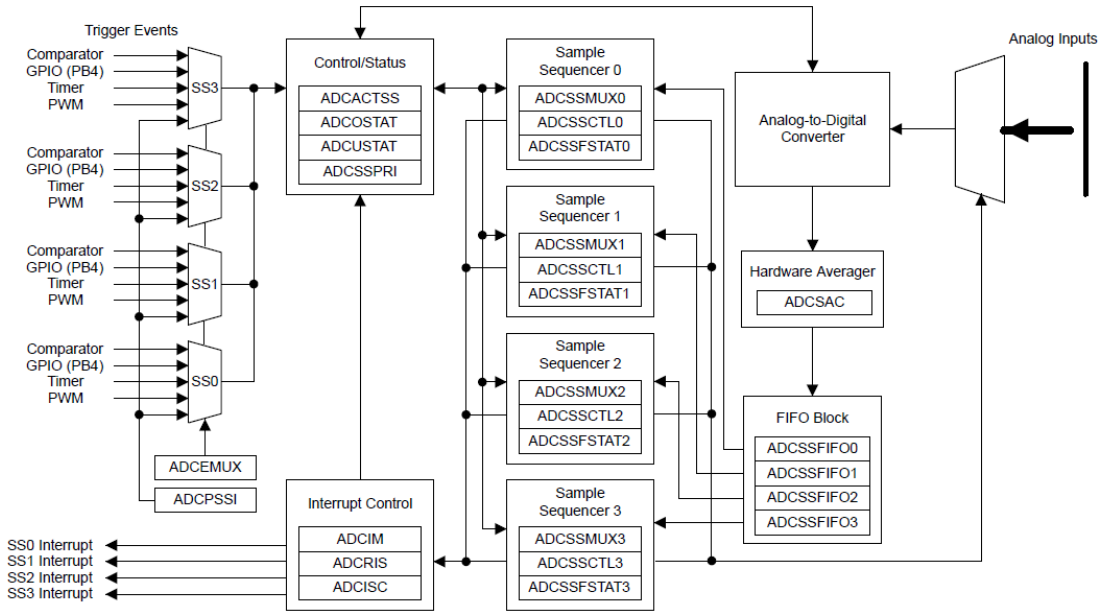
Tabulka H.1 – Přehled parametrů vstupně-výstupních portů [15]

Parametr	Popis	Minimální hodnota	Typická hodnota	Maximální hodnota	Jednotka
V_{IH}	Vstupní hodnota pro logickou hodnotu 1	2,0		5,0	V
V_{IL}	Vstupní hodnota pro logickou hodnotu 0	-0,3		1,3	V
V_{OH}	Výstupní hodnota pro logickou hodnotu 1	2,4			V
V_{OL}	Výstupní hodnota pro logickou hodnotu 0			0,4	V

Příloha H.3 Analogově-digitální převodník

Jednou z periferií mikrokontroléru LM3S811 je analogově-digitální převodník (A/D převodník) převádějící analogovou veličinu (napětí) na binární číslo. Převodník je schopen zpracovávat čtyři analogové vstupy označené ADC0 až ADC3, nebo interní teplotní senzor s rychlostí až 500 ks/s. Přesnost převodníku je 10 bitů. Referenční napětí pro A/D převodník je tvořeno vnitřní referencí o hodnotě 3 V,

převodník je tedy schopen zpracovávat napětí v rozmezí 0 V až 3 V. Blokové schéma A/D převodníku je znázorněno na obrázku H.4:



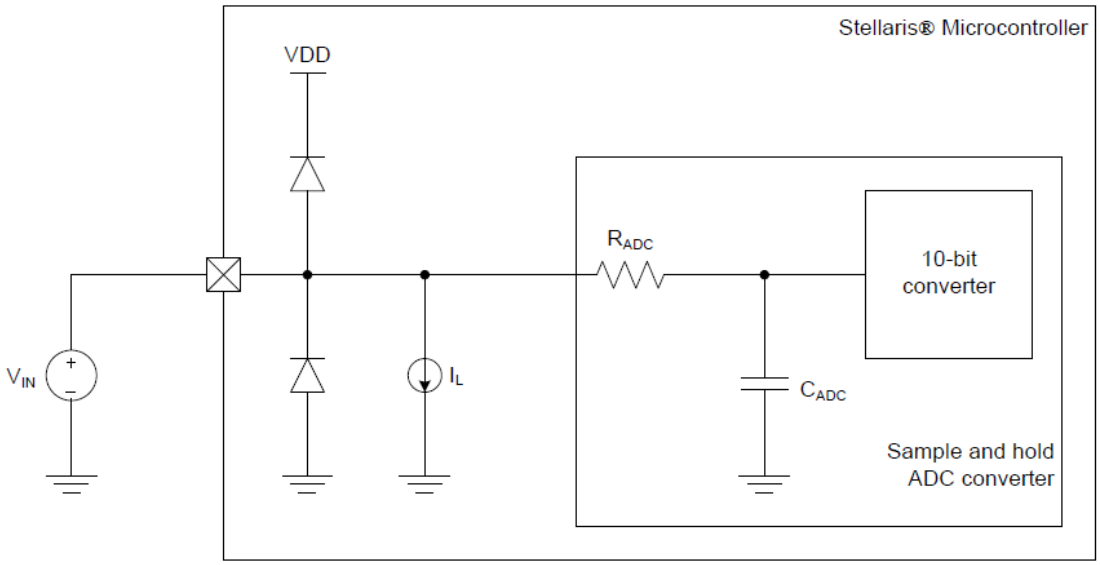
Obrázek H.4 – Blokové schéma A/D převodníku [15]

A/D převodník je tvořen čtyřmi vzorkovacími sekcemi (Sample Sequencer). Každou sekci lze naprogramovat samostatně pomocí registrů ADCSSMUX, ADCSSCTL a ADCSSFSTAT, mezi možnosti dané sekce patří i nastavení výběru teplotního senzoru nebo analogového vstupu, vyvolá-li dokončení převodu daného vstupu přerušení, ukončí-li převod vstupu převod celé sekce nebo bude-li vstup zpracováván diferenciálně s jiným analogovým vstupem. Převodník umožňuje hardwarové průměrování vzorků (Hardware Averager) pomocí registru ADCSAC. Vzorkovací sekce je schopna zpracovávat všechny čtyři analogové vstupy. První sekce označená Sample Sequencer 0 umožňuje zpracovat osm vzorků za sebou, druhá a třetí sekce zpracovává čtyři vzorky a čtvrtá sekce jeden vzorek, počet vzorků odpovídá velikosti paměťového místa FIFO Block pro danou sekci. Paměťová místa jsou tvořena jako paměť typu FIFO (First In First Out – data jsou čtena ve stejném pořadí, v jakém byla uložena), pro kontrolu stavu paměti slouží registry ADCOSTAT – kontrola přetečení zásobníku a ADCUSTAT – kontrola podtečení zásobníku. Převedené digitální číslo poskytují registry ADCSSFIFO0 až ADCSSFIFO3.

Prostřednictvím registru ADCEMUX se vybírá způsob spouštění vzorkovacích sekcí. Spuštění lze provést pomocí analogového komparátoru, externím vývodem

vstupně-výstupního portu, časovačem, generovanou pulsně šířkovou modulací a softwarově (manuálně). Po spouštěcí události daná sekce začne postupně převádět naprogramované vstupy jeden po druhém. Bude-li nakonfigurováno přerušení pro danou sekci (každá sekce má vlastní vektor přerušení), bude vyvoláno přerušení.

Blokové schéma jednoho analogového vstupu poskytuje následující obrázek:



Obrázek H.5 – Blokové schéma analogového vstupu [15]

Důležité parametry shrnuje následující tabulka H.2:

Tabulka H.2 – Přehled důležitých parametrů A/D převodníku [15]

Parametr	Popis	Minimální hodnota	Typická hodnota	Maximální hodnota	Jednotka
N	Přesnost		10		bitů
f_{ADC}	Hodinová frekvence při použití PLL	15	16,667	18	MHz
t_{LT}	Zpoždění spuštění převodu		2		hod. cykly
I_L	Ztrátový proud			±3,0	μA
R_{ADC}	Sériový odpor			10	kΩ
C_{ADC}	Kapacita pro udržení hodnoty napětí	0,9	1,0	1,1	pF
E_L	Chyba nelinearity			±1	LSB

Příloha H.4 Časovače

Mikrokontrolér LM3S811 je vybaven celkem čtyřmi časovači, tři jsou tvořeny normálními časovači, označované jako GPTM (General-Purpose Timer), čtvrtý časovač představuje Watchdog, který zabraňuje nechtěnému zacyklení programu mikrokontroléru.

Příloha H.4.1 Časovače GPTM

Časovače GPTM lze nakonfigurovat jako čítač/časovač, tzn., že při dosažení určité hodnoty časovače bude nastaven příznak umožňující vyvolání přerušení, pomocí něhož se signalizuje uplynutí určité, nastavené doby. Další možností konfigurace je generátor pulsně šířkové modulace PWM, ta se generuje porovnáváním nastavené hodnoty šířky pulsu s hodnotou časovače, pokud je hodnota časovače menší, výstup je v hodnotě 0, naopak je výstup v hodnotě 1. Poslední možností konfigurace je určení počtu změny hodnoty na určitém vstupu mikrokontroléru, nebo měření délky logické hodnoty na vstupu mikrokontroléru.

Každý ze třech časovačů přestavuje ve skutečnosti časovače dva, označované jako A a B, např. TIMER0A nebo TIMER2B apod. Tyto dílčí časovače jsou

16 bitové, je možná konfigurace jako jeden časovač s 32 bity. V této konfiguraci celý časovač zastupuje časovač A, hodnota čítání se zapisuje do registru patřícímu časovači A atd. Každý časovač je realizován jako odčítací s předděličkou, hodnota zapsaná do registru GPTMTAILR (GPTM TimerA Interval Load) nebo GPTMTBILR (GPTM TimerB Interval Load) se přepíše do registrů GPTMTAR (GPTM Timer A) nebo GPTMTBR (GPTM Timer B) představující aktuální hodnotu časovače, od této hodnoty se po každém pokynu od předděličky odečte číslo 1. Až bude některý z registrů GPTMTAR nebo GPTMTBR roven nule, bude nastaven bit pro signalizaci přerušení a bude-li přerušení povoleno, dojde k vyvolání přerušení. Každý časovač má dva vektory přerušení pro své dílčí časovače A a B. Pokud dojde k vynulování hodnoty časovače, do registru se opět načte hodnota z registru GPTMTAILR nebo GPTMTBILR a dochází opět k odečítání této hodnoty.

Každý časovač může být nastaven jako periodický, po vynulování hodnoty časovače začne další odčítací cyklus, nebo jako jednou spustitelný, v tomto režimu dojde při vynulování hodnoty časovače k zakázání časovače a další cyklus již není spuštěn. Pro nastavení chování časovače slouží registry GPTMCFG, GPTMTAMR, GPTMTBMR a GPTMCTL. Každý z časovačů lze nastavit jako spouštěcí pro A/D převodník, toto nastavení se provede nastavením bitu TAOTE nebo TBOTE v registru GPTMCTL. Spouštěcí podmínky od všech časovačů jsou mezi sebou orovány, tedy každý z časovačů může spustit A/D převod. Z tohoto důvodu se doporučuje nastavení pouze jednoho časovače jako spouštěcího pro A/D převodník. *Pozn.:* V aplikační poznámce výrobce [16] lze nalézt dodatek k používání časovače ke spouštění A/D převodníku. Pro čtvrtou vzorkovací sekci se musí použít pouze 16 bitový časovač, v opačném případě není zaručena správná funkce spouštění A/D převodníku.

Příloha H.4.2 Watchdog časovač

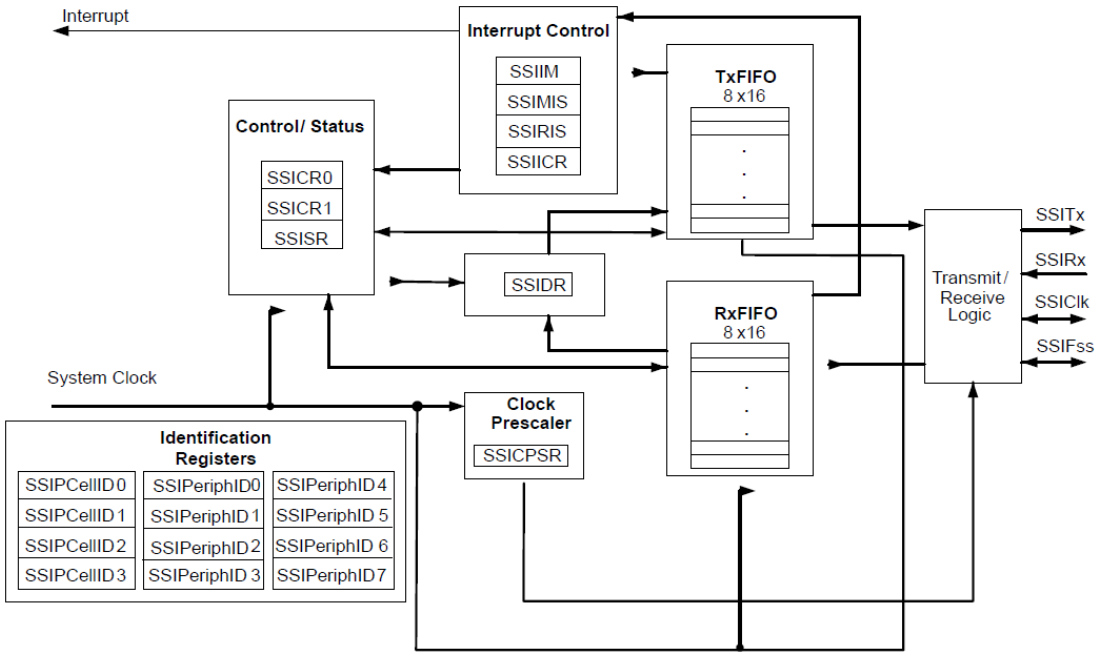
Časovač Watchdog pracuje na podobném principu jako časovač GPTM. Na rozdíl od ostatních časovačů může vynulování Watchdog časovače způsobit reset mikrokontroléru. Proto se tento časovač používá k zabránění zacyklení programu mikrokontroléru. Pokud by z nějakých neuvažovaných podmínek došlo k nadměrnému zpoždění v některé části programu, Watchdog časovač by nebyl ve

správný okamžik resetován a došlo by k jeho vynulování, čímž by došlo k resetu programu. Časovač je tedy nutné neustále resetovat.

Ovládání Watchdog časovače je velice jednoduché. Stačí nastavit dobu trvání, než se časovač vynuluje, zápisem do registru WDTLOAD (doba se přímo určuje z hodinové frekvence mikrokontroléru, bez předděličky), časovač povolit pomocí registru WDTCTL a nastavit možnost resetování mikrokotroléru. Dále se musí znovu nastavovat hodnota registru WDTLOAD, zápisem hodnoty do tohoto registru se znovu nastaví hodnota časovače (registr WDTVALUE), od kterého se odčítá hodnota 1. Vynulování registru WDTVALUE způsobí reset mikrokontroléru.

Příloha H.5 SSI – Synchronní sériové rozhraní

Periferie označovaná SSI (Synchronous Serial Interface) je obdobou SPI sběrnice. Umožňuje synchronní sériovou komunikaci pomocí čtveřice datových vodičů označených SSITx (výstupní datový vodič, pro SPI označovaný SDO), SSIRx (vstupní datový vodič, pro SPI označovaný SDI), SSIClk (hodinový signál, pro SPI označovaný SCK) a SSIFss (výběr zařízení, pro SPI označovaný SS). Je možné použít periferii jako obvod typu MASTER nebo SLAVE. Blokové schéma SSI periferie ukazuje obrázek H.6:



Obrázek H.6 – Blokové schéma periferie SSI [15]

Periferie SSI obsahuje dva datové zásobníky představující paměť typu FIFO s velikostí 16 bitů datového slova, do každého zásobníku lze uložit osm datových slov. Pro ukládání nebo čtení dat ze zásobníků slouží jediný registr SSIDR. Periferii SSI lze konfigurovat a řídit pomocí registrů SSICR0, SSICR1 a SSICR. Velikost hodinové frekvence je odvozena z velikosti systémové frekvence a je určena hodnotou registru SSICPSR podle následujícího vztahu:

$$SSIClk = \frac{FSysClk}{SSICPSR \cdot (1 + SCR)} \ [Hz] \qquad \text{(H.1)}$$

$SSIClk$ je hodnota hodinové frekvence přenosu dat periferie SSI [Hz],

$FSysClk$ je hodinová frekvence mikrokontroléru [Hz],

$SSICPSR$ je hodnota registru předděličky v rozsahu 2 až 254,

SCR je hodnota bitů v registru SSICR0, v rozsahu 0 až 255.

Pro obvod typu MASTER musí být hodinová frekvence přenosu dat dvakrát pomalejší, než je hodinová frekvence mikrokontroléru. Pro obvodu typu SLAVE musí být přenos dat dvanáctkrát pomalejší.

SSI umožňuje hardwarovou podporu řady typů sběrnic. Mezi podporované sběrnice patří:

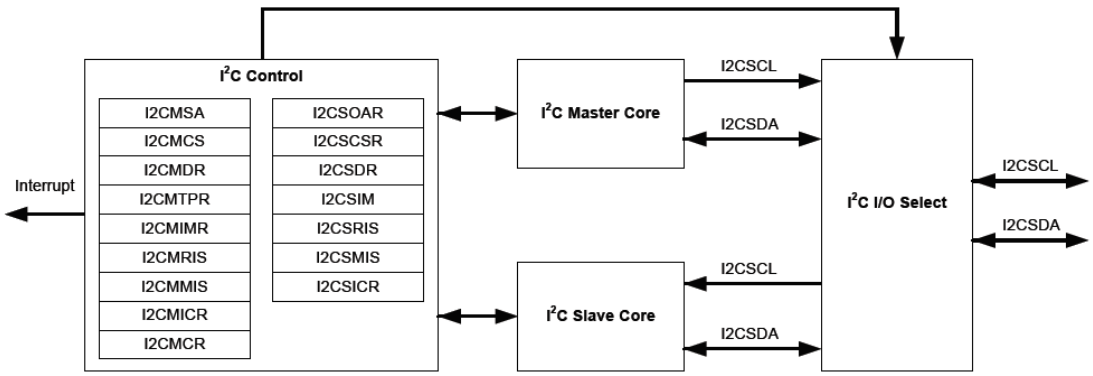
- synchronní sériový formát společnosti Texas Instruments,
- formát SPI,
- formát MICROWIRE.

Režim SPI, který je použit pro komunikaci s bezdrátovým IQRF modulem, umožňuje funkci totožnou s dříve popsanou sběrnicí SPI (viz příloha G). Periferii lze nakonfigurovat tak, jestli se jedná o typ obvodu MASTER nebo SLAVE, na jakou hranu hodinového signálu se budou data přenášet, v jaké hodnotě hodinové signálu se výstupní data mohou měnit. Velikost datového slova je nastavitelná v rozsahu 4 až 16 bitů.

Příloha H.6 Sběrnice I^2C

Pro použití sběrnice I^2C je mikrokontrolér LM3S811 vybaven přímo periferií podporující tuto sběrnici. Při použití této periferie je možné naprogramovat obvod

typu MASTER i SLAVE. Blokové schéma I²C periferie ukazuje následující obrázek H.7:



Obrázek H.7 – Blokové schéma periferie I²C [15]

Registry jsou rozděleny podle typu výsledného obvodu (MASTER nebo SLAVE), registry jsou namapovány na stejné adresy, ale význam jejich bitů je odlišný, pro čtení a zápis se význam bitů může také lišit. Mikrokontrolér bude používán jako obvod typu MASTER pro sériovou paměť EEPROM, proto bude popsáno pouze chování pro obvod typu MASTER.

Komunikace je řízená registrem I2CMCS, na základě nastavení určité kombinace bitů lze zajistit přechod do jednoho z některých stavů, ze stavu nečinnosti (IDLE), do stavu vysílání (Transmit) nebo příjmu (Receive). Adresa zařízení SLAVE se zapisuje do registru I2CMSA včetně bitu určujícího směr následné komunikace. Před samotným zahájením komunikace je nutné zapsat do registru I2CMDR hodnotu dat, která se mají vyslat po adrese zařízení SLAVE, po těchto krocích se může zahájit komunikace vysláním události START, zapsaná data se automaticky odešlou a obvod přejde do vnitřního stavu vysílání nebo příjmu podle hodnoty nejnižšího bitu adresy. Zahájení komunikace se uskutečňuje pomocí registru I2CMCS obsahujícího i stavové bity (pouze při čtení) pro kontrolu stavu sběrnice, např. byla všechna data odeslána apod. Hodnota bitu DATACK tohoto registru určuje, jestli se mají přijatá data potvrdit nebo ne. *Pozn.:* Při práci s touto periferií bylo zjištěno, že pro správné určení ukončení některé operace (bit BUSY) na sběrnici je nutné nejprve počkat na „zaneprázdnění“ sběrnice (vynulování bitu IDLE).

Rychlost datového přenosu je možné nastavit registrem I2CMTPR určujícího velikost hodinové frekvence. Tuto frekvenci lze vypočítat následujícím vztahem:

$$SCL\_PERIOD = \\ = 2 \cdot (1 + TIMER\_PRD\,) \cdot (SCL\_LP + SCL\_HP\,) \cdot \\ \cdot CLK\_PRD \;\; [s] \tag{H.2}$$

SCL_PERIOD je hodnota periody hodinové frekvence [s],

$TIMER\_PRD$ je hodnota registru I2CMTPR v rozsahu 1 až 127,

SCL_LP je doba trvání logické 0 na hodinovém vodiči SCL (hodnota 6),

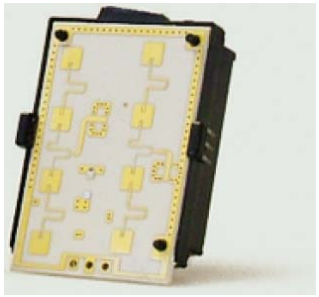
SCL_HP je doba trvání logické 1 na hodinovém vodiči SCL (hodnota 4),

CLK_PRD je perioda trvání hodinového signálu mikrokontroléru [s].

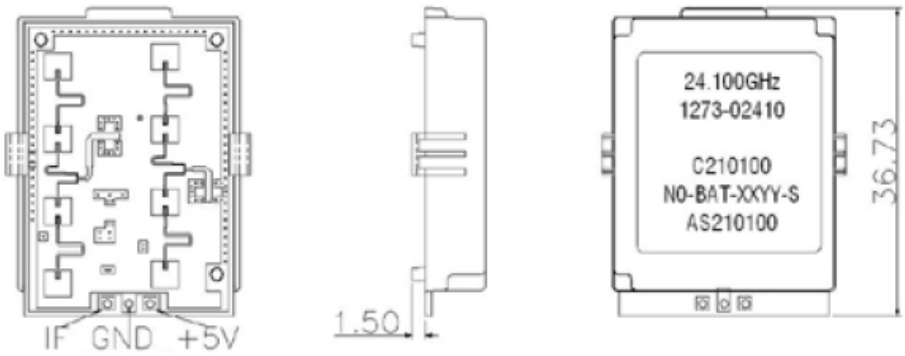
Příloha I. Mikrovlnný senzor

Tento senzor má za úkol přijímat z okolního prostoru signál, který na základě Dopplerova jevu vyhodnocuje, a výsledný signál lze sledovat na výstupu senzoru. Princip tohoto senzoru byl vysvětlen v kapitole 2.

Na základě provedených měření s různými typy senzorů od různých výrobců byl vybrán senzor MDU2410 od společnosti Microwave Solutions. Tento senzor pracuje na kmitočtu 24,2 GHz. Konstrukční provedení tohoto senzoru je zobrazeno na obrázku I.1 a I.2:



Obrázek I.1 – Konstrukční provedení mikrovlnného senzoru MDU2410 [25]



Obrázek I.2 – Rozměry a označení vývodů mikrovlnného senzoru MDU2410 [25]

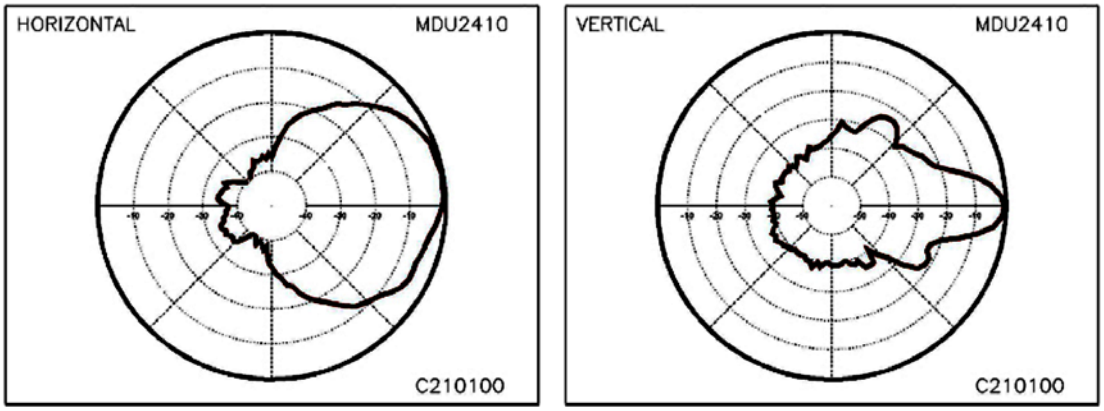
Rozměry mikrovlnného senzoru jsou poměrně malé. Připojení senzoru do obvodu je snadné prostřednictvím tří vývodů, dva jsou tvořeny napájením senzoru, poslední třetí vývod je výstup analogového signálu ze senzoru. Mezi další výhody senzoru patří nízká cena, vysoká citlivost, součástí senzoru je patch anténa.

Základní parametry mikrovlnného senzoru shrnuje následující tabulka:

Tabulka I.1 – Přehled základních parametrů mikrovlnného senzoru MDU2410 [25]

Parametr	**Hodnota**
Frekvence	24,2 GHz
Přesnost frekvence	10 MHz
Provozní teplota	-10 °C až +55 °C
Minimální výstupní výkon	7 dBm EIRP
Zisk antény	8 dBi
Napájecí napětí	+5 V ±0,25 V
Odebíraný proud	50 mA (max. 60 mA)

Diagramy antény senzoru jsou zobrazeny na následujícím obrázku:



Obrázek I.3 – Charakteristika antény mikrovlnného senzoru [25]

Charakteristiky antény jsou vynášeny ve dvou rovinách vzhledem k pozici senzoru na obrázku I.2. Je-li senzor umístěn vertikálně (kratší hrana senzoru je vodorovně), je charakteristika antény široká (obrázek I.3 vlevo), senzor zabírá široké okolí. Pokud bude senzor umístěn horizontálně (delší hrana senzoru je vodorovně), je charakteristika antény úzká (obrázek I.3 vpravo), senzor zabírá pouze úzké okolí.

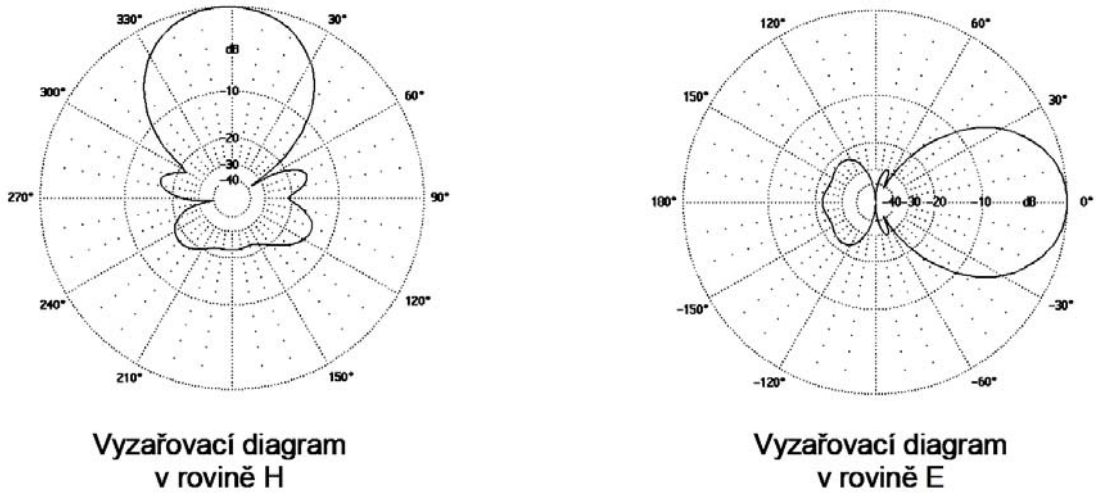
Výstupní analogový signál mikrovlnného senzoru nabývá velmi malých hodnot, které neumožňují přímé digitální zpracování. Proto je nutné za mikrovlnný senzor umístit zesilovač analogového signálu.

Příloha J. Popis antény BD 910A

Směrová anténa je vhodná pro datové přenosy v komunikačním pásmu 868 MHz. Tato anténa dosahuje zisku 11 dBi. Parametry antény jsou shrnuty v tabulce J.1 a vyzařovací diagramy jsou ukázány na obrázku J.1:

Tabulka J.1 – Přehled parametrů antény BD 910A [24]

Popis parametru	Hodnota	Jednotka
Kmitočtový rozsah	845 – 894	MHz
Zisk	11	dBi
Vyzařovací úhel v H rovině	50 – 62	°
Vyzařovací úhel v E rovině	46 – 53	°
Předozadní poměr	17 – 22	dB
Polarizace	Vertikální	
Impedance	50	Ω
PSV	< 1,8	
Maximální vstupní výkon	40	W
Rozměry d × v	600 × 180	mm



Obrázek J.1 – Vyzařovací diagramy antény BD 910A [24]

Příloha K. CD

K diplomové práci je přiloženo CD obsahující text práce v digitální podobě, zdrojové kódy, apod. Struktura složek na CD je následující:

- *Senzorová jednotka* – složka obsahuje podklady senzorové jednotky rozdělené do následujících složek:
  - *Návrh desky plošného spoje* – složka s obrázky rozmístění součástek na desce plošného spoje senzorové jednotky (měřítko není 1:1).
  - *Schéma* – ve složce jsou uložena schémata pro obě senzorové jednotky.
- *Texty* – soubory PaarJ_DetekcePohybu_RC_2011.docx a PaarJ_DetekcePohybu_RC_2011.pdf s textem této práce v digitální podobě.
- *Zdrojové kódy* – složka obsahující zdrojové kódy senzorových jednotek i IQRF modulu, rozdělené do následujících složek:
  - *Detektor osob* – složka se zdrojovým kódem jednotky detekce osob obsahující soubor Osoby_main.c.
  - *Detektor vozidel* – složka obsahující soubor Vozidla_main.c se zdrojovým kódem pro jednotku detekce vozidel.
  - *IQRF* – obsahuje soubor IQRF_main.c se zdrojovým kódem IQRF modulu.
  - *Společné části* – obsahuje zdrojové kódy společné pro obě senzorové jednotky. Zdrojové soubory jsou následující: board.h, i2c.c, i2c.h, iqrf.c a iqrf.h.