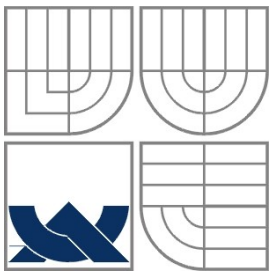


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

NÁVRH A REALIZACE MODULŮ PRO OVĚŘOVÁNÍ FUNKČNOSTI HW PERIFERIÍ MOBILNÍHO ROBOTU

THE PERIPHERAL TESTING MODULE DESIGN FOR MOBILE ROBOT

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE **BC. PETR MAŠEK**
AUTHOR

VEDOUCÍ PRÁCE **ING. STANISLAV VĚCHET, PH.D.**
SUPERVISOR

BRNO 2012

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2011/2012

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Petr Mašek
který/která studuje v **magisterském navazujícím studijním programu**
obor: **Aplikovaná informatika a řízení (3902T001)**

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

Návrh a realizace modulů pro ověřování funkčnosti HW periferií mobilního robotu

v anglickém jazyce:

The peripheral testing module design for mobile robot.

Stručná charakteristika problematiky úkolu:

Hlavním cílem práce je návrh a realizace modulů pro testování funkčnosti periferií mobilního robotu. Navržené moduly musí být schopny detekovat nefunkční periferie a podporovat logování chybových stavů. Důraz je kladen na kompletní test vybrané periferie a odhalení možných nefunkčních prvků.

Cíle diplomové práce:

1) Seznamte se s možnými periferiemi mobilních robotů.

2) Navrhněte testovací moduly pro vybrané periferie.

3) Navržené řešení oživte.

5) Proveďte experimentální ověření funkce realizovaných modulů na mobilním robotu.

Seznam odborné literatury:

[1] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series). Intelligent robotics and autonomous agents. The MIT Press, August 2005.

[2] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. Principles of Robot Motion: Theory, Algorithms, and Implementations (Intelligent Robotics and Autonomous Agents). The MIT Press, June 2005.

Vedoucí diplomové práce: Ing. Stanislav Věchet, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2011/2012.

V Brně, dne 12.12.2011

L.S.

Ing. Jan Roupec, Ph.D. Ředitel ústavu	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c. Děkan fakulty

ABSTRAKT

Tato diplomová práce se zabývá návrhem a realizací diagnostických modulů pro odhalení vadné periferie v mobilním robotu. Jeden z testovacích modulů je určen pro diagnostiku ultrazvukových sonarů. Druhý simuluje chování těchto sonarů, díky čemuž je schopen zjistit, na jakém rozhraní je nutné hledat chybu.

ABSTRACT

This diploma thesis deals with design and realization of testing modules for detecting a defective peripheral in a mobile robot. One of the modules is intended for the diagnosis of ultra sonic range finder. The second one simulates the behaviour of these ultra sonic range finders, making it able to determine interface on which it is necessary to look for an error.

KLÍČOVÁ SLOVA

Testovací modul, periferie mobilního robotu, rozhraní, ultrazvukové sonary, simulace, TWI, RS-485

KEYWORDS

Testing module, peripheral of mobile robot, interface, ultra sonic range finder, simulation, TWI, RS-485

PROHLÁŠENÍ O ORIGINALITĚ

Prohlašuji, že jsem uvedenou práci vypracoval samostatně pod vedením Ing. Stanislava Věcheta, Ph.D. Veškeré zdroje a odborná literatura využité při zpracování této práce jsou řádně citovány a uvedeny v seznamu použité literatury.

Bc. Petr Mašek

BIBLIOGRAFICKÁ CITACE

MAŠEK, P. *Návrh a realizace modulů pro ověřování funkčnosti HW periferií mobilního robotu.*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2012. 61 s. Vedoucí diplomové práce Ing. Stanislav Věchet, Ph.D..

PODĚKOVÁNÍ

Touto cestou bych chtěl poděkovat Ing. Stanislavu Věchetovi, Ph.D. za odborné vedení, cenné rady a čas věnovaný autorovi při tvorbě této práce. Dále chci poděkovat Ing. Janu Hrbáčkovi, specialistovi z firmy Bender Robotics s.r.o., za jeho ochotu, trpělivost a předání odborných zkušeností při vývoji testovacích modulů.

Obsah:

Zadání závěrečné práce **3**
Abstrakt **5**
Prohlášení o originalitě **7**
1 Úvod **11**
2 Seznámení s problematikou **13**
2.1 Robot Advee 13
2.2 Periferie 13
2.2.1 Ultrazvukové sonary 14
2.2.2 Infračervené senzory 16
2.2.3 Kamery 19
2.2.4 Laserový skener 19
2.2.5 GPS 20
2.3 Rozhraní pro periferie 21
2.3.1 USART 21
2.3.2 TWI 22
2.3.3 RS-485 24
3 Návrh a realizace **25**
3.1 Testovací modul pro ultrazvukové sonary 26
3.1.1 Volba součástek 26
3.1.2 Návrh schématu 27
3.1.3 Návrh a realizace DPS 29
3.1.4 Program pro mikrokontroler 31
3.1.5 Výsledná funkčnost 36
3.2 Testovací modul pro zařízení na zpracování dat z ultrazvukových sonarů 37
3.2.1 Volba součástek 38
3.2.2 Návrh schématu 39
3.2.3 Návrh a realizace DPS 40
3.2.4 Program pro mikrokontroler 42
3.2.5 Výsledná funkčnost 46
4 Funkce a použití testovacích modulů **49**
5 Závěr **53**
Seznam použité literatury **55**
Seznam příloh **57**

1 ÚVOD

Žádný autonomní mobilní robot by neměl svojí činností ohrozit člověka, popřípadě poškozovat cizí majetek. Proto je pro autonomní mobilní roboty velmi důležité, aby měly přehled o prostředí, ve kterém se pohybují. Hlavně je pro ně důležitý přehled o svém blízkém okolí. Jestliže robot není schopen vyhnout se překážce, případně před překážkou zastavit, může být pro své okolí nebezpečný. Zvláště jestliže se jedná o robota dostatečně robustního na to, aby dokázal svou nesprávnou činností ublížit člověku.

Pro orientaci v prostředí, detekci překážek, nebo rozpoznávání tvarů se v mobilní robotice používají různé druhy senzorických modulů. Správná a spolehlivá funkčnost těchto senzorů je nepostradatelná. Avšak stejně jako každé jiné elektronické zařízení, ani tato nejsou bezporuchová. Na trhu lze najít velké množství výrobců každého druhu modulu a tím pádem ještě více různých druhů provedení. Z tohoto důvodu neexistuje žádný univerzální návod, jak odhalit vadný senzor.

Použité senzorické moduly pracují jako periferie, zajišťující základní interakci robotu s okolím. Daný senzor vždy komunikuje se svým nadřízeným zařízením prostřednictvím určitého druhu rozhraní. Nadřízeným zařízením může být například řídící jednotka robotu, která data z těchto senzorů vyhodnocuje. Pro detekci okolních překážek je většinou potřeba mít na robotu rozmístěných více senzorických modulů, kde každý hlídá určitý vyhrazený prostor, což si při pohledu shora na robota lze představit jako kruhovou výseč. Z důvodu použití více periferií stejného druhu na mobilním robotu se často pro taková řešení uplatňují datové sběrnice. Využití sběrnice znesnadňuje detekci poruchy jednotlivých senzorických modulů v rámci sběrnice.

Z důvodu problematické detekce vadných senzorických modulů je hlavním cílem této práce návrh a realizace testovacích modulů periferií. Testovací moduly jsou primárně určeny pro autonomní mobilní robot Advee od firmy Bender Robotics, která úzce spolupracuje s VUT FSI. Robot Advee bude představen v kapitole 2.1.

Před návrhem diagnostických zařízení bude nutné se podrobně seznámit s periferiemi, které jsou v mobilním robotu implementovány. Tato problematika bude podrobněji rozebrána v kapitole 2.2. Následně se v kapitole 2.3 seznámíme s rozhraními, pomocí kterých mohou tyto periferie komunikovat s jinými zařízeními.

Diagnostická zařízení budou navržena pro periferie s nejvyšší poruchovostí. Pomocí testovacích modulů bude možno efektivně v případě poruchy diagnostikovat vadnou periferii. Testovací modul z kapitoly 3.1 je mimo jiné určen pro modifikaci firmwaru tak, aby nová periferie plnila původní funkci. Pomocí diagnostického zařízení z kapitoly 3.2 bude možné kontrolovat zpracovávaná data danou periferií. V kapitole 4 budou následně provedeny experimenty s testovacími moduly.

2 SEZNÁMENÍ S PROBLEMATIKOU

Roboty, které se dnes běžně používají, lze rozdělit do několika kategorií. Pro tuto práci je však důležité posoudit je z hlediska poruch a rychlosti servisních zásahů. Například pro tzv. „hobbyroboty“ není určitě podstatné, aby byly rychle opraveny. Tyto roboty jsou většinou konstrukčně jednoduché a jejich výroba není komplikovaná. Oproti tomu u komerčních robotů, jejichž výroba je technologicky i konstrukčně komplikovaná, je důležité, aby servisní zásah trval co nejkratší dobu. Například u robotů, které jsou ve výrobních halách, je už velice důležité, aby odstraňování poruch trvalo co nejkratší dobu, protože každá porucha, při které se robot musí zastavit, stojí výrobní závod nemalé peníze, poněvadž za dobu, po kterou robot stál, mohl vyrobit nebo zpracovat velké množství výrobků.

Z výše uvedených důvodů bych robota Adveeho zařadil do kategorie těch robotů, které potřebují v případě poruchy rychlou diagnostiku a servisní zásah. Jedná se sice o prezentačního robota, který však nesprávnou činností může kvůli své stokilogramové váze ohrozit zdraví lidí popřípadě poškodit majetek.

S robotem a jeho hlavními periferiemi se seznámíme v následujících podkapitolách. V dalších podkapitolách rozeberu rozhraní, přes která se periferie připojují k robotu.

2.1 Robot Advee

Advee je robot sloužící hlavně k reklamním účelům. Je schopen se sám bez ovládání pohybovat, rozeznávat lidské tváře a používat hlasovou interakci s člověkem, na požádání je pak schopen vytisknout leták.[1]

Tento robot je zajímavým doprovodným doplňkem na VIP akcích, plesech, kongresech, konferencích, reprezentačních a propagačních akcích, veletrzích nebo v rámci reklamní kampaně.[1]

V robotu Adveem jsou použity ultrazvukové sonary a infračervené senzory pro detekci překážek a pohyb v nevyzpytatelném prostředí. Také zde najdeme kameru, servomotory, dotykový display, mikrofon, reproduktory a tiskárnu pro tisk letáků. K „indoor“ navigaci slouží speciální vysílače, které komunikují s robotem určitým signálem, díky němuž rozpozná, kde se v prostředí nachází.

Obr. 1: Vzhled robotu Advee. [1]

2.2 Periferie

V mobilní robotice se využívá množství různých periferií, jako je například kamera, nebo rozhraní pro vstup, jako je například klávesnice. U autonomních robotů se ale především požívají periferie pro detekci překážek, což jsou třeba ultrazvuky a infračervené senzory, popřípadě nákladnější laserové scannery. Podle použití robota může k jeho navigaci sloužit

GPS modul, kompas nebo akcelerometr. Pro pohyb robotu se většinou kvůli snadnému řízení pomocí mikrokontrolerů používají servomotory.

Všechny signály, ať už z infračervených čidel, ultrazvuků nebo servomotorů, musí být nějak zpracovány. K tomu mohou sloužit podřízené řídící jednotky, které budou přijímat a zpracovávat data z ultrazvuků a dále je posílat hlavní řídící jednotce. Podobně si můžeme představit i řídící jednotku pro řízení serv, která jen přijímá povely od hlavní řídící jednotky, a o řízení servomotorů se stará tato podřízená. Tím se dá hlavní řídící jednotce možnost ušetřit procesorový čas a může se věnovat hlavně zpracování navigačních algoritmů. Proto jsou tyto podřízené řídící jednotky označovány také jako periferie.

2.2.1 Ultrazvukové sonary

Ultrazvukové sonary jsou senzory určené k měření vzdálenosti k překážce, popř. k detekci překážky. Ultrazvukové sonary jsou založeny na měření doby mezi vysláním akustického impulsu a okamžikem přijetí odraženého signálu od překážky – echa. Je-li vyhodnocována pouze přítomnost překážky a jestliže uživatele nezajímá vzdálenost k ní, je vyhodnocováno pouze přijetí a nepřijetí echa. [2]

U běžných sonarů, sloužících např. pro měření výšek hladiny, měření vzdáleností ve stavebnictví, zemědělství, lesnictví, robotice nebo automatizaci a pro hlídání prostoru, se používá ultrazvukový signál nejčastěji o kmitočtu kolem 40 kHz. [3]

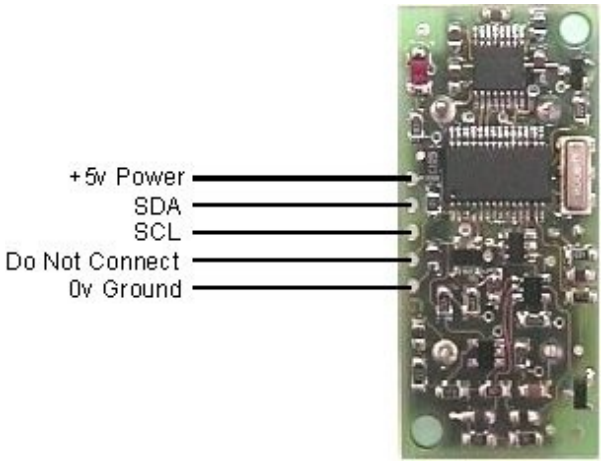
Obr. 2: Vzhled ultrazvukového sonaru SRF08. [4]

Existuje mnoho různých ultrazvukových sonarů s různými vlastnostmi a použitými rozhraními. V robotu, pro který budou testovací moduly určeny, se používají ultrazvukové sonary SRF08 (Obr. 2) od výrobce Devantech, následující text proto bude zaměřen hlavně na vlastnosti tohoto typu ultrazvukového sonaru SRF08.

Ultrazvukový sonar SRF08 má několik nastavitelných módů. Dokáže například vyhodnotit vícenásobné odrazy, čímž je možné vyhodnotit vzdálenosti k více překážkám. Díky tomu je možné u mobilního robota vybaveného tímto ultrazvukovým sonarem měřit vzdálenosti i skrz otevřené dveře. Dále v módu ANN (Artificial Neural Network) poskytuje naměřené hodnoty již dopředu zpracované pro vyhodnocení pomocí neuronové sítě. O zpracování dat do vhodné formy se stará mikrokontroler PIC16F872 umístěný na ultrazvukovém sonaru. Pomocí ultrazvukového sonaru lze měřit intenzitu osvětlení díky zabudovanému senzoru. Měření osvětlení pak probíhá při každém měření vzdálenosti. [3]

Ultrazvukový sonar SRF08 komunikuje po I2C (TWI) sběrnici (bude popsána v kapitole 2.3.2), která je u mikrokontrolerů velice populární. Rozsah měření vzdálenosti sonaru je od 3 cm až do 11 m. Sonar je stavěný na napětí 5V, při měření odebírá ze zdroje proud 15 mA. Při čekání na další příkaz je schopen přepnout se do úsporného režimu, ve kterém odebírá ze zdroje pouze 3 mA [2]. U modulu je možné softwarově měnit adresu, pod kterou se hlásí na sběrnici TWI. Defaultně je z výroby nastavena adresa E0. Dále lze nastavit adresy E2, E4, E6, E8, EA, EC, EE, F0, F2, F4, F6, F8, FA, FC, FE. Lze tedy využívat současně šestnáct adres. Pokud bychom chtěli zahájit měření ze všech adres, stačí

zavolat adresu 0, což je univerzální adresa na sběrnici TWI. Která adresa je na ultrazvuku navolena, lze zjistit podle počtu zablikání LED, jak je uvedeno v Tab. 1, je to však poměrně nepraktické a zdlouhavé. Může se stát, že se při počítání blikání spleteme, pak by na sběrnici byly připojeny dva sonary se stejnou adresou, a tak by byla narušena komunikace.



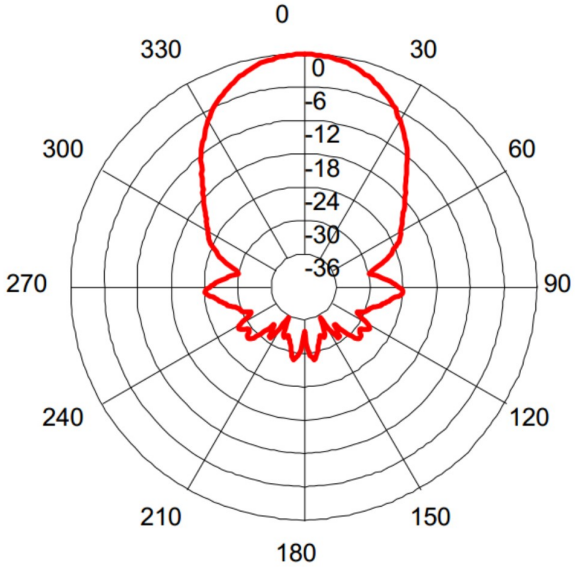
Obr. 3: Zapojení ultrazvukového sonaru SRF08. [5]

Na Obr. 3 lze vidět zapojení ultrazvukového sonaru SRF08. Vývody SDA a SCL slouží na komunikaci po TWI, vývody +5 V Power a 0 V Ground slouží pro připojení napájení, poslední vývod Do Not Connect slouží pouze výrobci k nahrání programu do mikrokontroleru ultrazvukového sonaru a nesmí se připojovat.

Adresa		Dlouhé bliknutí	Krátké bliknutí
Hexadecimálně	Desítkově		
E0	224	1	1
E2	226	1	2
E4	228	1	3
E6	230	1	4
E8	232	1	5
EA	234	1	6
EC	236	1	7
EE	238	1	8
F0	240	1	9
F2	242	1	10
F4	244	1	11
F6	246	1	12
F8	248	1	13
FA	250	1	14
FC	252	1	15
FE	254	1	16

Tab. 1: Jak lze poznat adresu u ultrazvukového sonaru SRF08. [5]

Pro veškeré měření, změny rozsahů, změny měřících modů, samozřejmě i změnu adresy je potřeba poslat přesně daný příkaz, popřípadě sled příkazů. To bude popsáno v kapitole 3.1.4, protože budou potřeba při programování jednoho z testovacích modulů.



Obr. 4: Vyzařovací diagram ultrazvukového sonaru SRF08. [5]

Na Obr. 4 lze vidět vyzařovací diagram ultrazvukového sonaru SRF08. Úhel, který ultrazvukový sonar dokáže zabírat, je až 55°. K tomuto největšímu záběru dochází při odchylce -6 dB akustického tlaku.

Ultravukové sonary mají oproti infrasenzorům, které budou popsány v následující kapitole, výhodu v mnohem větší šířce zabíraného prostoru. Díky tomu lze pomocí jednoho ultrazvukového sonaru zabírat stejný prostor jako bychom zabírali pomocí čtyř infračervených senzorů. Další výhodou je schopnost měřit ve výhodnějších rozsazích, a to v případě SRF08 od 3 cm až do 11 m. Nevýhodou je možnost špatně změřené vzdálenosti. Bude-li překážka vzhledem ke směru vysílaného echa pod příliš ostrým úhlem, dojde k odrazu signálu jiným směrem. V tomto případě nedojde ke změření vzdálenosti. Jestliže by se však signál dále odrazil a vrátil by se zpět k ultrazvukovému sonaru, došlo by k vyhodnocení, které by obsahovalo špatné údaje o vzdálenosti. S podobným neduhem se setkáme i u infrasenzorů.

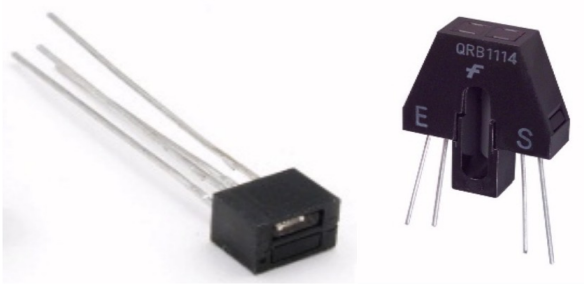
Veškeré technické informace obsaženy v této kapitole byly převzaty od výrobce z datasheetu k SRF08 zde [5].

2.2.2 Infračervené senzory

Dalším zařízením, které se hojně používá k určování vzdálenosti případně detekci překážek, jsou infračervené senzory. Lze je ale také použít pro sledování čáry, kterou například robot detekuje a podle které jezdí. Pro oba typy detekce se používají rozdílné infrasenzory s odlišnými parametry.

Pokud chceme, aby robot sledoval čáru, využívá se rozdílné odrazivosti světla od tmavého nebo světlého povrchu. Osvítíme-li plochu, na které se střídá černá a bílá, a snímáme-li odražené světlo senzorem, dostaneme na výstupu tohoto senzoru rozdílnou velikost elektrického signálu při snímání odrazu vysílaného světla od černé nebo bílé plochy. Je samozřejmé, že úroveň tohoto signálu bude značně ovlivněna okolním osvětlením a celý senzor je tedy nutné zaclonit před dopadajícím parazitním světlem, například uzavřením do neprůhledné schránky. Pro omezení této závislosti na rušení okolním světlem se

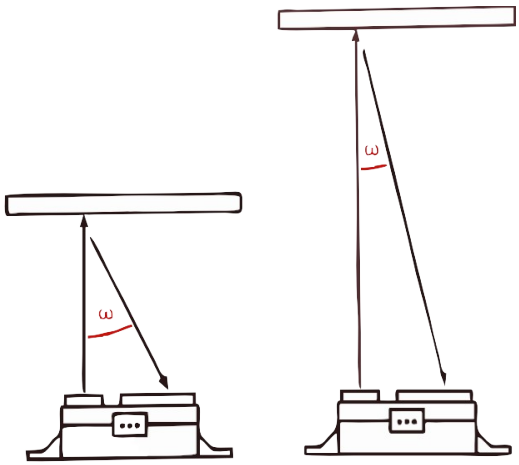
v běžných světelných senzorech používá LED vyzařující v infračervené části spektra, pro lidské oko neviditelné. Přijímací fototranzistor má naopak předřazen filtr, který viditelné světlo nepropustí. Elektrický signál je pak dále nutné zpracovat, to se většinou provádí v mikrokontroleru, výhodnější je ale na výstup senzoru ještě připojit vyhodnocovací obvod, který přiřadí černé a bílé barvě jednoznačnou logickou úroveň L nebo H. Předejde se tak rušivým vlivům. [6]



Obr. 5: Odrazové infrasenzory, vlevo QRD1114 pro zapájení do plošného spoje, vpravo QRB1114 určený k připevnění ke konstrukci šroubkem M3. [4]

Další možností, jak už bylo výše řečeno, je využití infrasenzoru pro určování vzdálenosti. Používá se podobného principu jako u senzorů pro sledování čáry. Máme zde také vysílací infračervenou LED a přijímač, který toto infračervené světlo zachytává. Je tu však velký rozdíl v tom, jak toto vyhodnocení probíhá.

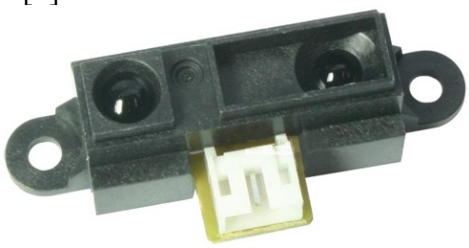
Čidlo měří úhlovou odchylku odraženého paprsku. Obsahuje výkonnou vysílací infračervenou LED a několik přijímacích prvků, optiku a vyhodnocovací elektroniku. Vysílací infračervená LED zobrazí na předmětu intenzivně osvětlený bod, přijímací optika promítne tento bod na řadu přijímacích prvků. Nejintenzivněji osvětlený prvek z řady pak udává úhel, pod kterým je vidět odražený paprsek. Na Obr. 6 lze tento princip vidět společně s tím, že čím dále překážka bude, tím menší bude úhel, pod kterým se paprsek odráží. Aby senzor nemusel obsahovat velké množství jednotlivých přijímačů, vyhodnocuje se také intenzita signálu. Vestavěná elektronika pak vygeneruje příslušné výstupní napětí odpovídající vzdálenosti překážky. [7]



Obr. 6 Princip měření dálkoměrného infrasenzoru. [7]

Dálkoměrné infrasenzory se připojují k mikrokontroleru buď k digitálním vstupům mikrokontroleru, nebo na piny analogově digitálního převodníku. Zde pak dochází k vyhodnocení výšky napětí, tím pádem i vzdálenosti od překážky. To, které hodnoty napětí odpovídají jaké vzdálenosti, se vždy dozvíme v katalogovém listu k danému čidlu. Tyto hodnoty je pak nutné zanést i do programu mikrokontroleru.

Přepočet napětí na odpovídající vzdálenosti je možný několika způsoby. Můžeme použít tabulku hodnot, kde ke každé hodnotě napětí vyhledáme vzdálenost. Tento způsob je rychlý, ale náročný na zapamatování konstant. Další možností je použít jenom některé hodnoty z tabulky a zbylé dopočítat lineární extrapolací. Tento způsob zmenší velikost tabulky, ale pro mikrokontroler komplikuje výpočet. Třetí, asi nejpoužívanější možností, je aproximace racionální funkcí. [7]



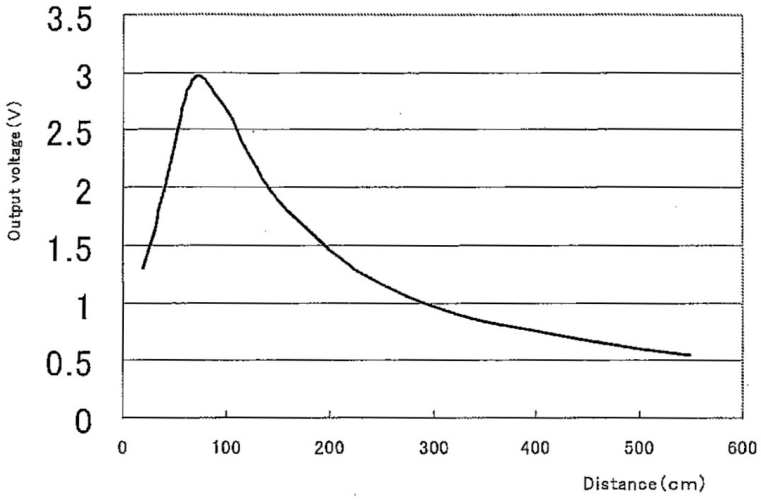
Obr. 7: Infračervený dálkoměr GP2D120. [10]

Podle aplikace, pro kterou bude robot určen, a podle prostředí, ve kterém se bude robot pohybovat, je třeba vybrat vhodný odrazový infrasenzor. V Tab. 2 lze vidět srovnání čtyř druhů odrazových infrasenzorů. Je zde patrné, že když budeme chtít například detekovat člověka, který se náhle postaví do cesty robotu, musíme si vybrat senzor, jenž detekuje vzdálenosti například od 4 cm. Musíme se ale u něj spokojit s tím, že maximální měřitelná vzdálenost bude 30 cm jako je to například v katalogovém listu odrazového infrasenzoru GP2D120, který najdeme zde [8] a jenž je na Obr. 7. Pokud na druhou stranu budeme chtít vědět o překážkách, které jsou před robotem ve vzdálenosti větší než 30 cm, můžeme vzít například odrazový infrasenzor GP2Y0A700, jenž má katalogový list zde [9]. Ten detekuje vzdálenosti v rozmezí 100 - 500 cm, ale už nedetekuje překážky, které jsou blíže přímo před robotem. Důvod lze vidět v grafu na Obr. 8. Je zde zřejmé, že kdybychom chtěli měřit tímto infrasenzorem vzdálenosti menší než 100 cm, hodnota napětí odpovídající vzdálenosti pod 100 cm by zároveň odpovídala vzdálenosti nad 100 cm. Tím by bylo měření velmi nejasné. Proto se senzor používá v daném rozsahu.

Odrazový infrasenzor	Měřitelná vzdálenost v cm	
	Minimální	Maximální
GP2D120	4	30
GP2Y0A21	10	80
GP2Y0A02	20	150
GP2Y0A700	100	500

Tab. 2: Srovnání infračervených odrazových senzorů.

Jestliže použijeme robota v prostředí, kde jsou po stěnách lesklé nebo dokonce skleněné prvky, jako jsou třeba skleněné dveře, popřípadě výlohy atp., bude docházet k velkým chybám v určování vzdálenosti. Lesklé povrchy budou odrážet vyslaný paprsek mimo detektory na čidlu, popřípadě pod špatným úhlem a tím pádem bude docházet k velkým nepřesnostem při měření. Infračervené dálkoměrné senzory jsou oproti ultrazvukový sonarům cenově méně nákladné. Z důvodů výše zmiňovaných vlastností je však vhodné použít kombinaci těchto čidel, popřípadě každému dát jiný úkol. Ultrazvukové sonary například mohou sloužit pro detekci okolních překážek i určování vzdáleností od překážek. Infračervené senzory pak mohou být použity na podvozku robotu pro detekci terénu, aby robot nesjel ze schodů.



Obr. 8: Graf závislosti vzdálenosti a výstupního napětí odrazového infrasenzoru GP2Y0A700. [9]

2.2.3 Kamery

Kamery se často používají ve výrobě pro sledování zmetkovosti. Pomocí kamer dnes již dokážeme určovat, jestli jsou výrobky ve správné rozměrové toleranci či jestli jsou všechny vrtané díry ve správné vzdálenosti od sebe.

Pro nás je však důležité, jaké mají kamery využití u mobilních robotů. Jak ve vnitřním, tak venkovním prostředí se dnes často používají kamery, díky kterým může být robot schopen rozeznávat celé objekty, obličeje a popřípadě hledat nejvhodnější cestu. Kamery jsou dnes cenově srovnatelné s již zmíněnými prvky pro detekci překážek, jako jsou ultrazvukové sonary či infrasenzory, a stejně jako ony je kamera velmi lehká.

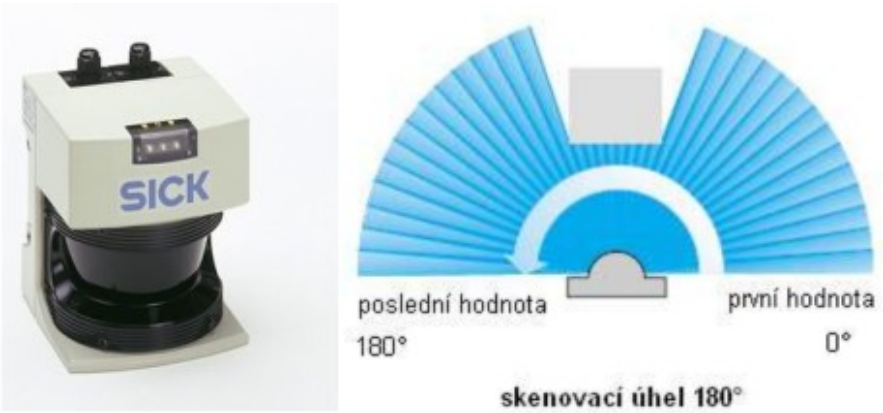
Použití kamery s sebou samozřejmě oproti ostatním senzorům přináší mnohé nevýhody, jako je nutnost zpracovat velké množství dat. Další nevýhodou je vliv okolního osvětlení a také nemožnost přímo měřit vzdálenosti. Zmíněné nevýhody lze však vyřešit softwarově. Jakmile budeme mít obrázky z kamery v počítači, můžeme v nich rozpoznávat objekty, lze manipulovat s jejich kontrastem či jasem, ale také je skládat za účelem získání třírozměrné podoby. Pro zpracování obrazu lze použít starší počítač, který bude mít dostatečný výkon. S takovýmto řešením se můžeme dostat na podobnou cenu jako bychom dali za několik ultrazvukových sonarů a řídící jednotku na zpracování dat. Existují kamery připojitelné přes rozhraní I2C (TWI), kterým je vybavena většina mikrokontrolerů, je však potřeba brát v úvahu, že takové řešení není vhodné k zvládání složitějších algoritmů pro zpracování obrazu. Díky kameře připojené k počítači lze rozpoznávat výrazné objekty podle barvy, můžeme určovat jejich polohu nejen v obraze samotném, ale i v okolním prostoru. [11]

Zkombinujeme-li kameru s dalšími senzory, dokážeme se tak například nejen zastavit před náhlou překážkou, ale dokonce poznat, že se jedná o člověka, a oslovit ho, jak je to podobně řešeno v robotu Adveem.

2.2.4 Laserový skener

Dalším velice zajímavým, ale bohužel dosti cenově náročným detektorem, je laserový skener. Ten je v mobilní robotice vhodný jak pro venkovní, tak vnitřní použití. Má podobný princip jako infračervené dálkoměrné senzory pouze s tím rozdílem, že zde se jedná o laserový infračervený impulz.

Vysílač laserového měřícího skeneru emituje laserový impulz. Ten je vysílán a pokud narazí na nějakou překážku, odrazí se od ní. Odražený paprsek se vrací zpět a je přijat přijímačem. Doba letu paprsku od vyslání až po přijetí je přímo úměrná vzdálenosti skeneru od dané překážky. Aby se parsek odrazil, musí stejně jako u infračervených dálkoměrů mít překážka určitou odrazivost a samozřejmě musí být v dosahu skeneru. [12]



Obr. 9 Laserový skener SICK LMS291-S05 vpravo, princip měření laserovým skenerem vpravo. [12], [13]

Pomocí vnitřního rotačního zrcátka je laserový impulz vychýlen a umožňuje tak skenovat okolí, jak je vidět na Obr. 9. Z posloupnosti odražených impulzů se určí obrys překážky. Laserový skener má tedy vždy nějaký zorný úhel, který je schopen zabírat. Například u laserového skeneru LMS291-S05 od firmy SICK, který je vidět také na Obr. 9, je úhel zorného pole 180°. Úhlové rozlišení je pak možné nastavit na 0,2°; 0,5° nebo 1°, přičemž rozmezí měření je od 0 m do 80 m. Tím se řadí v porovnání s předchozími čidly k nejlepším, avšak jak už jsem zmiňoval, jedná se o poměrně drahé zařízení. Toto čidlo lze připojit přes rozhraní RS-232, popřípadě RS-422. [12],[13]

2.2.5 GPS

Global Positioning system, zkráceně GPS, je určen pro použití venku a je dnes součástí téměř každého mobilního telefonu, případně jako příslušenství v automobilu. Tento systém se však dá využít i v mobilní robotice pro navigaci robota na určité místo.

Jedná se, jak už trochu z názvu vyplývá, o družicový poziční systém, který provozuje ministerstvo obrany USA od roku 1994. Sestává se z 32 družic, pozemního řídícího střediska a uživatelských přijímačů. Družice jsou umístěny na šesti oběžných drahách kolem Země. Sytém GPS umožňuje určit polohu přijímače kdekoliv na zeměkouli. Poloha je dána zeměpisnou šířkou, délkou a nadmořskou výškou s přesností několika metrů.[14]

Přesnost, s jakou lze pomocí GPS určit polohu, závisí dle [15] na schopnostech měření doby letu signálu, ale hlavně na množství viditelných satelitů a jejich geometrické konfiguraci. Představíme-li si 2D případ, byla by výsledná pozice průnikem tří kružnic (známe jejich středy - pozice satelitů a relativní rozdíly poloměrů). Naše měření jsou však nepřesná, a tak místo kružnic počítáme průsečíky tří mezikruží. Výsledný průnik má pak nenulový obsah a jeho velikost záleží jak na chybě jednotlivých měření, tak na vzájemné poloze satelitů. Existují však metody, jak tyto chyby minimalizovat. [15]

Při použití na mobilním robotu nám bude stačit GPS přijímač. Pro tyto účely jsou k dostání speciální přijímače, které nejsou vybaveny displejem ani klávesnicí (říká se jim GPS moduly) [14]. Jsou však vždy vybaveny nějakým komunikačním rozhraním (základní rozhraní budou popsána v kapitole 2.3), pomocí kterého je možno data zpracovávat v mikrokontroleru, popřípadě v počítači. Jak GPS přijímač ovládat, popřípadě jaké příkazy

mu předávat udává vždy výrobce daného typu GPS modulu a je potřeba buď vždy prostudovat datasheet, nebo má většina výrobců na svých stránkách hotové knihovny, se kterými je pak použití GPS modulu velmi jednoduché.

2.3 Rozhraní pro periferie

V předchozí kapitole 2.2 jsem rozebral nejpoužívanější periferie pro navigaci a detekci překážek u mobilního robotu. Každé periferii je ale potřeba nějakým způsobem dávat povely, aby započalo měření, a pak příslušná data od periferie získat. To se děje pomocí komunikačního rozhraní. Některé rozhraní nebo periferie funguje jako sběrnice, a lze tedy na něj připojit větší množství zařízení. Jiné lze zase pomocí snadno dostupných modulů využít na bezdrátové přenosy apod. V této kapitole budou rozebrány periferie, které se v robotice často používají a které jsou i použity v robotu Adveem.

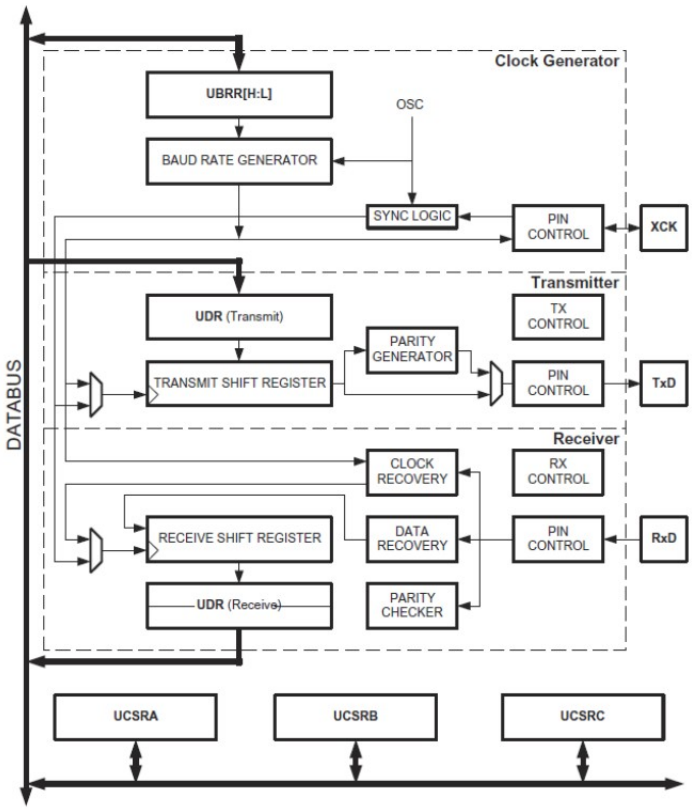
2.3.1 USART

Universal Synchronous and Asynchronous Receiver Transmitter (zkráceně USART) v překladu znamená univerzální synchronní a asynchronní sériový přijímač a vysílač, někdy se též označuje jako SCI, Serial Communications Interface, což v překladu znamená rozhraní pro sériovou komunikaci. Jedná se o jedno z nejpoužívanějších rozhraní u mikrokontrolerů schopné obousměrné komunikace. Je možné je využívat na komunikaci s jiným mikrokontrolerem, a také, jak bude popsáno níže, je možné po menší úpravě pomocí USART komunikovat přímo s počítačem.

USART lze používat v několika režimech, a to v asynchronním (plný duplex), který se často užívá při komunikaci s jinými mikrokontrolery a může zde probíhat příjem i odesílání dat zároveň. V asynchronním režimu není vysílán hodinový signál. Další dva režimy jsou synchronní–master a synchronní–slave (oba poloviční duplex). Ty se používají například pro komunikaci se sériovou EEPROM. V režimu Master mikrokontroler vysílá hodinový signál sám a naopak v režimu Slave je hodinový signál přijímán. Ani u jednoho nedochází ve stejný čas k vysílání a k příjmu. Při posílání dat je příjem zakázán a obráceně. Vždy jsou používány dva různé piny označené zpravidla jako TxD a RxD, jeden pro vysílání a druhý pro příjem dat. Při komunikaci dvou mikrokontrolerů je třeba, aby měly nastavenu stejnou přenosovou rychlost. Mikrokontroler, který přijímá data, se na začátku pokusí synchronizovat svoji vnitřně generovanou přenosovou rychlost s přicházejícími asynchronními rámci dat. Synchronizační proces je opakován při každé detekci start bitu. V synchronním režimu je zapotřebí přenášet hodinový signál. Pokud mikrokontroler generuje hodinový signál vnitřně a posílá ho na svůj výstupní pin, pracuje v režimu Master. Pokud sám hodinový signál přijímá, nachází se v režimu Slave. [16]

Na Obr. 10 je možné vidět blokové schéma modulu USART, konkrétně mikrokontroleru ATmega8. Nalezneme zde tři hlavní části, a to vysílací část označenou jako Transmitter, dále část pro příjem dat označenou jak Receiver a poslední část pro generování hodinového signálu označenou jako Clock Generator. Jak je patrné z předchozího odstavce, tuto poslední část ne vždy využijeme, protože záleží na režimu, který budeme u USART využívat. Z blokového schématu je patrné, že některé bloky označené tučným písmem jsou pak přístupné při programování mikrokontroleru. To, k čemu se tyto jednotlivé bloky využívají, je vždy možné se dočíst v Datasheetu k mikrokontroleru, v tomto případě k ATmega8, zde [17]. Dále je na obrázku znázorněna datová sběrnice označena jako DATABUS, která je s jednotlivými částmi propojena a také propojuje řídící registry (UCSRA, UCSRB a UCSRC), ty jsou s jednotlivými bloky sdíleny. Generátor hodinového signálu obsahuje synchronizační logiku pro příjem externích hodin při synchronní komunikaci v režimu Slave a také blok označený jako BAUD RATE GENERATOR, pro odvození

přenosové rychlosti. Pin XCK vycházející z části Clock Generator je používán pouze při synchronní komunikaci v režimu Slave. Pin TxD vychází z vysílací části, která se skládá ze vstupního bufferu označeného jako UDR, sériového posuvného registru označeného TRANSMIT SHIFT REGISTER, generátoru parity (PARITY GENERATOR) a kontrolní logiky pro zvládnutí různých formátů posílaných rámců. RxD, tedy pin vycházející z přijímací části, která obsahuje blok CLOCK RECOVERY, se stará o obnovu hodin a blok DATA RECOVERY zajišťuje obnovu dat. Oba tyto bloky jsou používány při asynchronním příjmu dat. Mimo jiné tato část pro příjem dat obsahuje blok pro kontrolu parity označený jako PARITY CHECKER, kontrolní logiku, sériový posuvný registr (RECEIVE SHIFT REGISTER) a dvouúrovňový přijímací buffer označený jako UDR. [16],[17]



Obr. 10 Blokové schéma modulu USART pro ATmega8. [17]

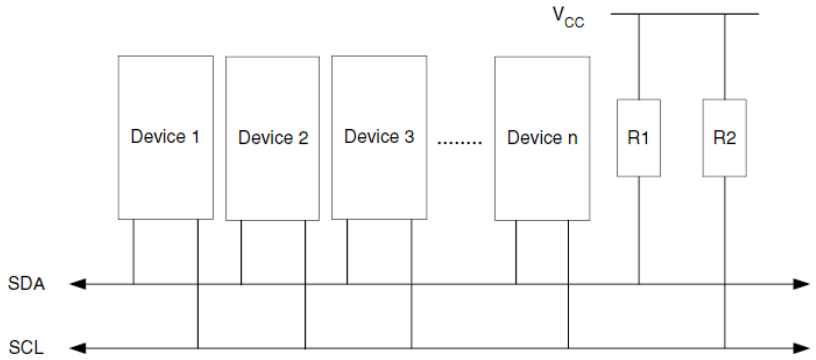
Rozhraní USART lze použít i pro komunikaci s PC přes RS-232. Je ale nutné brát v úvahu, že každé z těchto rozhraní má jiné napěťové úrovně pro logickou jedničku a logickou nulu. U USART je to 0 až 0,8 V pro logickou nulu a 2 až 5 V pro logickou jedničku, kdežto u RS-232 jsou tato napětí dosti odlišná, podle použitého zařízení mohou nabývat hodnot ±5 V, ±10 V, ±12 V nebo dokonce ±15 V. U všech platí záporná hodnota napětí pro logickou nulu a kladná pro logickou jedničku. Proto je potřeba použít převodník jako například MAX232. Díky této možnosti převodu se rozhraní USART používá nejčastěji pro komunikaci s počítačem. Stejným způsobem, pomocí převodníku, lze rozhraní USART použít na komunikaci pomocí RS-485, které bude popsáno v kapitole 2.3.3. [16]

2.3.2 TWI

Pokud budeme chtít komunikovat pomocí jednoho rozhraní s více zařízeními, bude pro nás nejvýhodnější použít nějaký typ sběrnice a právě tyto požadavky splňuje rozhraní TWI. Můžeme se také setkat s označením I2C, jež smějí používat pouze výrobci, kteří mají licenci od firmy Philips, jež na toto rozhraní drží patent. Ostatní výrobci, jako například firma

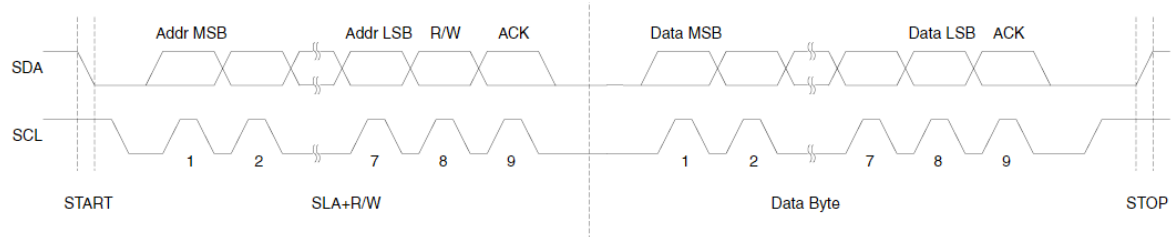
Atmel, používají označení TWI (Two-wire Serial Interface, v překladu dvoudrátové sériové rozhraní), které je s I2C kompatibilní. Díky této sběrnici můžeme obousměrně komunikovat až se 127 zařízeními typu Slave, a to díky tomu, že je zde zavedena 7 bitová adresace už v samotném protokolu. Nelze však současně komunikovat obousměrně. [18]

Jak už vyplývá z názvu rozhraní, k propojení zařízení se používají dvě linky. Jedna je datová a označuje se jako SDA. Druhá linka se nazývá SCL, jedná se o linku, po které běží hodinový signál neboli synchronizační linku, pomocí níž se určuje rychlost komunikace, jež může být např. dle datasheetu k ATmega8, který nalezneme zde [17], až 400 kHz. Jak jednotlivá zařízení připojit ke sběrnici, lze vidět na Obr. 11. Obě linky jsou napojeny přes dva pull-up rezistory k napájení, což zajišťuje v klidovém stavu na obou linkách logickou jedničku. Jednotlivá zařízení mohou ze sběrnice číst aktuální stav. [18]



Obr. 11: Způsob zapojení zařízení k TWI rozhraní. [18]

Průběh komunikace po TWI je vidět na Obr. 12. Komunikaci vždy zahajuje zařízení typu Master. Pokud chce Master začít vysílat, musí nejdříve vyslat tzv. START condition, kdy se na SDA změní stav z logické 1 na 0 a SCL přitom zůstává na 1. Od tohoto okamžiku dochází k posílání jednotlivých datových bitů. Všechna ostatní zařízení typu Slave zatím sledují průběh na obou linkách. Platnost bitu zaručuje SCL nastavené na logickou 1, takže k veškerým změnám na SDA dochází, když je SCL na logické 0. Výjimku v tomto tvoří tzv. START a STOP condition, kdy dochází ke změnám pouze na SDA a SCL zůstává na logické 1. Každý rámec má 8 datových bitů a jeden potvrzovací bit nazvaný ACK. První bit vyslaný Masterem je adresa, jak už bylo výše řečeno, skládající se ze 7 bitů. Poslední bit vyslaný Masterem určuje, zda chce ze zařízení přijímat data nebo je do něj odesílat. Adresované zařízení se pak podle toho přepne do čtecího či odesílacího módu. Pokud Slave data přijímá, potvrzuje každý rámec, pokud přijímá Master, potvrzení je na něm. Ačkoliv Master řídí časování na SCL, Slave může komunikaci zpomalit. Jestliže nastaví SCL na nulu, nemůže Master vysílat a čeká. [18]



Obr. 12: Příklad průběhu komunikace po TWI. [18]

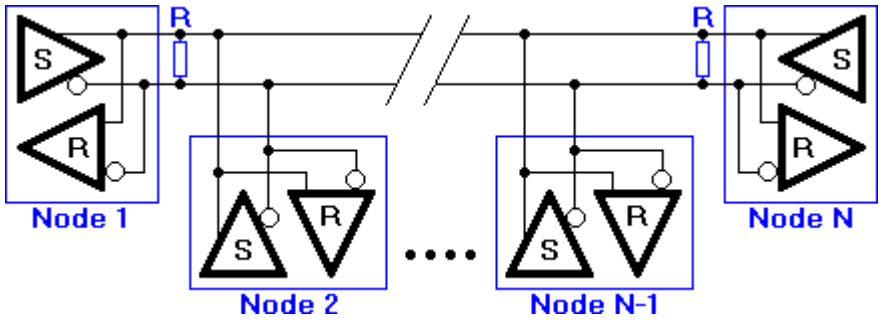
Pomocí TWI lze, jak už bylo řečeno, komunikovat s jinými mikrokontrolery, mimo to také lze pomocí tohoto rozhraní ovládat speciální typy LCD panelů, přistupovat k pamětem typu EEPROM, nebo dokonce ovládat serva. Kromě výše popsaného zapojení, kde vystupoval pouze jeden Master, lze zapojit ke sběrnici více zařízení téhož typu, jen je

potřeba dávat pozor, aby se v komunikaci nepřekrývala, pak by docházelo k chybovým stavům. Pokud bychom chtěli takovéhoto zapojení využít, je potřeba pečlivě prostudovat datasheet k mikrokontroleru, který budeme používat.

2.3.3 RS-485

Zde se jedná o rozhraní používané hlavně v průmyslu. Používá se však i v robotu Adveem a bude použito v praktické části této práce. Podobně jako v předchozí kapitole se jedná o sběrnici, po které lze obousměrně komunikovat až mezi 32 zařízeními.

Datová komunikace může probíhat sériově po dvouvodičovém stíněném kabelu, tzv. kroucené dvoulince. Jednotlivé logické stavy jsou pak reprezentovány rozdílným napětím mezi vodiči. Jestliže bychom jeden vodič označili jako A, druhý jako B a při měření napěťového rozdílu mezi vodiči bychom naměřili napětí menší než -0,3 V, jednalo by se o logickou jedničku. Logickou nulu pak reprezentuje napěťový rozdíl mezi vodiči větší než +0,3 V. Správný vysílač by měl na výstupu generovat úrovně +2 V a -2 V a přijímač by měl být schopen rozlišit úrovně 200 mV a -200 mV jako platný signál. Důvodem, proč na výstupu vysílače a na vstupu přijímače mohou být tak rozdílná napětí odpovídající stejnému stavu, je to, že je tato sběrnice mimo jiné určená k přenosu dat na větší vzdálenosti. Samotný přenos probíhá v sedmi nebo osmibitových rámcích, s jedním startbitem a jedním či více stopbity. Startbit představuje logickou nulu. Naopak stopbit nebo neaktivní stav je logickou jedničkou. [19]



Obr. 13: Způsob připojení zařízení ke sběrnici RS-485. [20]

Na Obr. 13 nalezneme způsob připojení zařízení na sběrnici. Jak je mimo jiné vidět, každé zařízení může obsahovat zesilovač. Na koncích vedení je umístěn rezistor, který snižuje rušení, jehož hodnota odpovídá impedanci vedení. Je zřejmé, že nelze najednou po jednom páru přenášet data oběma směry, je proto vyžadováno řízení přístupu na sběrnici, což se řeší softwarově. Druhou možností je využití čtyřvodičové verze RS-485 neboli dvou kroucených dvoulinek, to nabízí obousměrnou komunikaci a není tedy nutné zabývat se řízením směru komunikace. Více využívaná je však dvouvodičová verze. Většina RS-485 systémů používá podobně, jako tomu bylo u rozhraní TWI, Master/Slave architekturu, tedy každá Slave jednotka má svoji adresu a odpovídá pouze na příkazy adresované jí. Tyto příkazy rozesílá Master, který periodicky obesílá všechny jednotky. [19],[21]

3 NÁVRH A REALIZACE

Testovací moduly, které je potřeba realizovat, jsou určeny pro mobilního robota Adveeho vyvinutého firmou Bender Robotics, jehož jsme si představili v kapitole 2.1. Po konzultaci se specialistou z této firmy jsem se dozvěděl, že nejčastější problémy jsou s ultrazvukovými sonary popsanými v kapitole 2.2.1. Bylo proto domluveno, že jeden z testovacích modulů by měl být schopen rozeznat vadný ultrazvukový sonar, popřípadě co nejvíce ulehčit jeho výměnu za nový. Protože se v robotu používá více sonarů, které se rozlišují svými adresami na rozhraní TWI, které bylo popsáno v kapitole 2.3.2, bude potřeba novému ultrazvukovému sonaru přiřadit jeho unikátní adresu. Testovací modul pro ultrazvukové sonary, by měl umět:

- rozeznat stávající adresu ultrazvukového sonaru
- možnost zvolit novou adresu a zapsat ji
- provádět test funkčnosti ultrazvukového sonaru

Druhým zařízením, se kterým bývají problémy, je řídící jednotka na zpracování dat z ultrazvuků. Ta využívá pro sběr dat z ultrazvuků již zmíněné rozhraní TWI a pak tato data předává po rozhraní RS-485, popsané v kapitole 2.3.3, jiné nadřízené řídící jednotce, proto budeme i tuto řídící jednotku na zpracování dat považovat za periferii. Tyto jednotky na zpracování dat z ultrazvuků jsou v robotu použity dvě. Každá z nich zpracovává data z osmi ultrazvuků. Jedna z ultrazvuků umístěných vpředu na robotu a druhá vzadu. Úkolem také bude tyto dvě jednotky mezi sebou rozlišit. Testovací modul jednotek na zpracování dat z ultrazvukových sonarů by měl umět:

- kontrolovat komunikaci probíhající po rozhraní TWI
- simulovat osm ultrazvukových sonarů připojených na rozhraní TWI
- kontrolovat probíhající komunikaci po rozhraní RS-485
- kontrolovat data přijatá po RS-485
- rozlišovat jednotky pro sběr dat z předních či zadních ultrazvukových sonarů

Bude tedy potřeba navrhnout a realizovat dva testovací moduly. Každý bude řízený mikrokontrolerem a do každého bude potřeba navrhnout program, který bude zmíněné periferie testovat. Ještě předtím bude ale potřeba navrhnout zapojení a zvolit vhodné součástky, kterými budeme chybové stavy detekovat a prezentovat. Poté, až bude hotov návrh schématu, se musí pro každý modul navrhnout deska plošných spojů (dále jen DPS). Ty by měly být navrženy tak, aby byla jejich funkce pochopitelná a moduly byly snadno ovladatelné. Jak pro návrh schématu, tak pro návrh DPS bude použit program Eagle, který je dostupný zde [22] a je možné ho pro studijní účely používat zdarma s menšími omezeními, která mi však při návrhu nevadila. Mikrokontrolery byly zvoleny od firmy Atmel, proto pro vývoj softwaru pro ně bude použito AVR studio volně distribuované touto firmou, které je dostupné zde [23]. Po nahrání softwaru do mikrokontrolerů už bude zbývat jen ověření funkčnosti testovacích modulů, a to jak na funkčních, tak nefunkčních periferiích, aby se ověřilo, že jsou moduly schopny detekovat chybové stavy.

V následujících kapitolách proto podrobněji rozeberu celý postup vývoje testovacích modulů od výběru vhodných součástek až po otestování jejich funkčnosti.

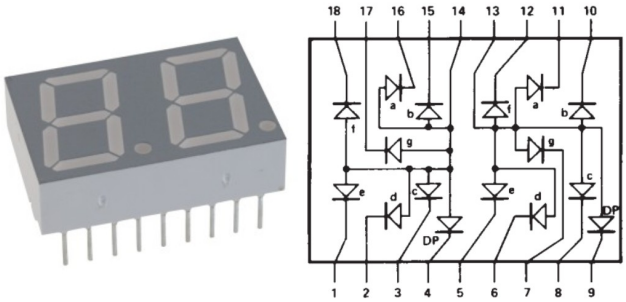
3.1 Testovací modul pro ultrazvukové sonary

V robotu Adveem se používají ultrazvukové sonary SRF08, které byly podrobněji rozebrány v kapitole 2.2.1. Víme, že ultrazvuk lze označit jednou z šestnácti adres. V robotu se však počítá s využitím maximálně patnácti z nich, protože defaultní adresa E0 se používat nebude. Úkolem je, aby testovací modul uměl přečíst adresu, pod kterou se ultrazvuk hlásí, a nějak nám ji interpretovat. V případě, že bychom chtěli použít jinou adresu, než kterou ultrazvuk zrovna disponuje, bylo by dobré, abychom si mohli zvolit jednu z patnácti zbývajících adres a tuto do ultrazvukového sonaru zapsali. Ultrazvukové sonary SRF08 komunikují po rozhraní TWI a pomocí něho se i adresy zapisují.

Ověřit, že ultrazvukový sonar funguje, by se dalo tak, že bychom spustili sekvenci na měření. Naměřené hodnoty bychom nějak interpretovali a podle toho by uživatel modulu mohl poznat, jestli ultrazvuk měří správně. Protože se může zdát, že je ultrazvukový sonar funkční, ale ve skutečnosti tomu tak není. Například případ, kdy při připojení napájení zabliká s LED podle Tab. 1 a při sběru naměřených dat také problikne, se naoko tváří, že je všechno tak, jak má být. Kdybychom se ale podívali blíže, jaká data ultrazvuk posílá, zjistili bychom, že posílá buď stále stejnou hodnotu, i když by se měla měnit, nebo nulovou hodnotu, což řídící jednotka robota vyhodnotí jako překážku před ultrazvukem nebo okolo něho. Proto je důležité pomocí modulu také ověřit hodnoty, které ultrazvuk odesílá.

3.1.1 Volba součástek

Z předešlých kapitol už víme, že budeme potřebovat něco, čím budeme zobrazovat naměřené hodnoty z ultrazvukového sonaru, mimo to také budeme z ultrazvuku číst adresu, kterou má přiřazenou, a dále budeme i zobrazovat adresu, kterou si navolíme a následně do ultrazvuku zapíšeme. K tomu by nám mohl stačit segmentový display. Zvolil jsem proto display HDSP-K121, jenž je možné vidět na Obr. 14. společně s jeho vnitřním zapojením, které pro nás bude důležité při návrhu schématu.



Obr. 14: Vpravo segmentový display HDSP-K121, vlevo jeho vnitřní zapojení. [24]

Dále budeme potřebovat něco, čím si budeme volit adresu, jež budeme zapisovat. Zde najdeme spoustu možností, které můžeme použít, například jediné tlačítko, kterým budeme stále dokola přepínat jednu z patnácti adres a až dojdeme na konec, začneme volit adresy zase od začátku. To je pouze jedna z možností. Jako vhodnější se však jeví ta, kde by šla adresa navolit více prvky. Máme patnáct adres, které budeme chtít navolit. Při použití čtyř spínacích prvků bude mít každý z nich dva stavy, a to zapnuto nebo vypnuto. Na těchto čtyřech prvcích lze díky tomu navolit 2^4 kombinací, což je celkem 16 kombinací, takové množství je určitě dostačující a poslední šestnáctou kombinaci si navíc můžeme nechat na nějaký jiný mód modulu. Pro takové řešení se zdá nejvýhodnější čtyřpólový spínač DIP 04, který lze vidět na Obr. 15. Ten bude pro požadovanou funkci dostačující. Na něm lze totiž, jak už bylo řečeno, navolit šestnáct možných kombinací a z nichž patnáct použijeme na volbu adresy, kterou budeme chtít do ultrazvukového sonaru zapsat.

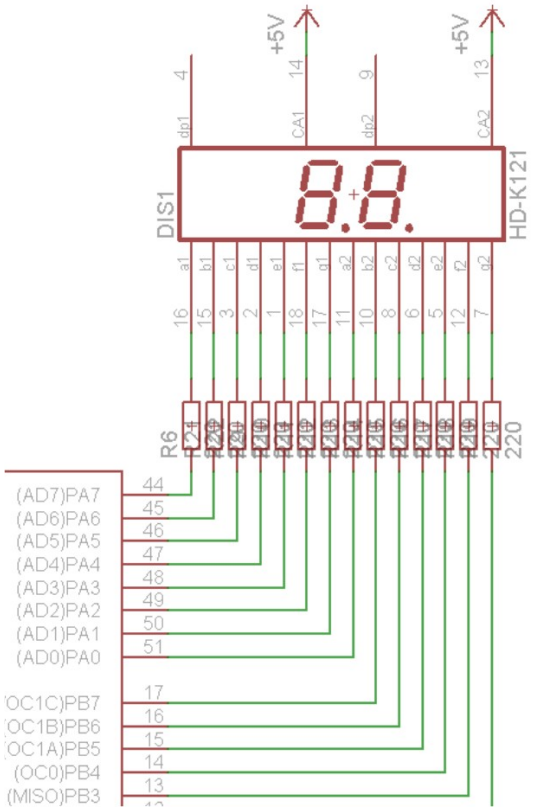
Obr. 15: Čtyřpólový spínač DIP 04 PIANO BLUE. [25]

Mimo výše zmíněné součástky budeme potřebovat mikrospínač, kterým potvrdíme, že chceme adresu zapsat, konektor pro připojení ultrazvukového sonaru k modulu, posuvný spínač pro zapnutí modulu, konektor pro připojení napájení a programovací konektor. V neposlední řadě ještě musíme zvolit vhodný mikrokontroler, ke kterému budou veškeré součástky připojeny. Pro segmentový display, jenž má možnost zobrazit dvě čísla nebo písmena, budeme potřebovat sedm výstupů pro každou číslici/písmeno, tzn. dohromady čtrnáct výstupů pro display. Dále čtyři vstupy pro spínač DIP 04 a samozřejmě ještě jeden vstup pro mikrospínač. Budeme tedy potřebovat mikrokontroler, který bude mít dohromady alespoň 19 vstupů/výstupů a bude mít možnost použití rozhraní TWI. Z těchto důvodů se jeví jako nejlepší použít mikrokontroler ATmega128, který všechny požadavky více než dostatečně splňuje.

Nyní máme všechny důležité součástky, které pro požadovanou funkci modulu budeme potřebovat, a můžeme se pustit do návrhu zapojení. Samozřejmě není to kompletní výčet součástek, jedná se pouze o ty nedůležitější. Jsou potřeba ještě další pro zapojení různých podpůrných obvodů, aby nedošlo ke zničení mikrokontroleru. Proto budou ještě potřeba rezistory, popřípadě kondenzátory pro dodržení správného zapojení.

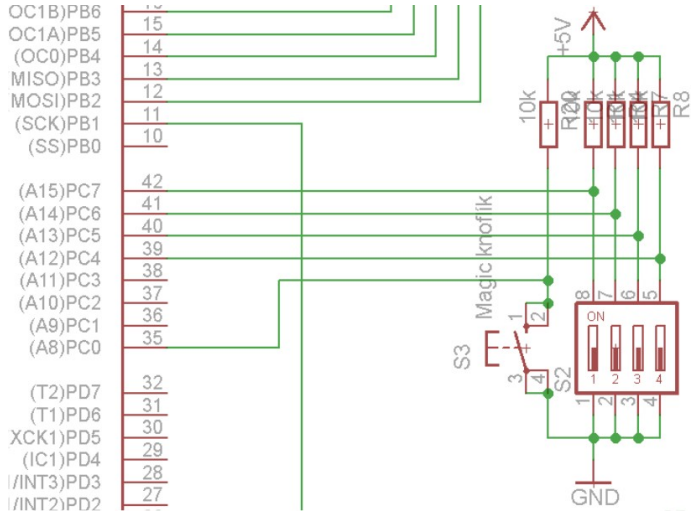
3.1.2 Návrh schématu

Nyní již máme představu o tom, jaké součástky budeme potřebovat k tomuto testovacímu modulu. Jak již byl zmíněno v předchozí kapitole, u mikrospínačů, spínačů a LED diod je potřeba použít správné zapojení. Tato doporučená zapojení můžeme najít například zde [26]. Segmentový display, jak bylo vidět na Obr. 14, se uvnitř skládá z LED diod. Každá ze čtrnácti LED diod má v displayi samostatný vývod na katodu a anoda je připojená společně pro každý ze dvou znaků zvlášť. Jestliže jsme se seznámili s předešlým, můžeme display připojit v návrhu schématu k volným pinům mikrokontroleru tak, jak je vidět na Obr. 16. Mezi katodu každé LED diody v displayi a piny mikrokontroleru je ještě zapojen rezistor. Zvolil jsem velikost 220 Ω. Je možné zvolit i větší, čímž se však sníží svítivost displaye. Lze také vidět, že dva z vývodů displaye nejsou připojeny, což je úmyslné, protože slouží ke zobrazení teček u znaků a to nepotřebujeme.



Obr. 16: Zapojení displaye HDSP-K121 k mikrokontroleru.

Další již zmiňovanou součástkou, kterou jsme zvolili v předešlé kapitole, je čtyřpólový spínač DIP 04 a mikrospínač. Zde také dodržíme doporučená zapojení a v návrhu schématu tedy dojdeme k zapojení, které je vidět na Obr. 17. V tomto zapojení je nutné také použít rezistory, tentokrát je použita hodnota 10 kΩ. Jedna strana mikrospínače a DIP 04 jsou připojeny k zemi (GND), druhá strana k mikrokontroleru a paralelně přes rezistor na napájení +5 V.



Obr. 17: Zapojení mikrospínače a čtyřpólového spínače DIP04.

Poslední důležitou částí, kterou budeme potřebovat, je zapojení konektoru pro rozhraní TWI. Jedná se o dvouvodičové rozhraní, proto na konektor přivedeme signál ze dvou pinů mikrokontroleru, a to z pinů označených jako SDA a SCL, byla o nich řeč v kapitole 2.3.2. Konektorem budeme chtít však ultrazvukový sonar i napájet, abychom to nemuseli dělat

zvlášť. Proto je konektor čtyřkolíkový a kromě výše zmíněných dvou pinů je na něj ještě připojeno napájení.

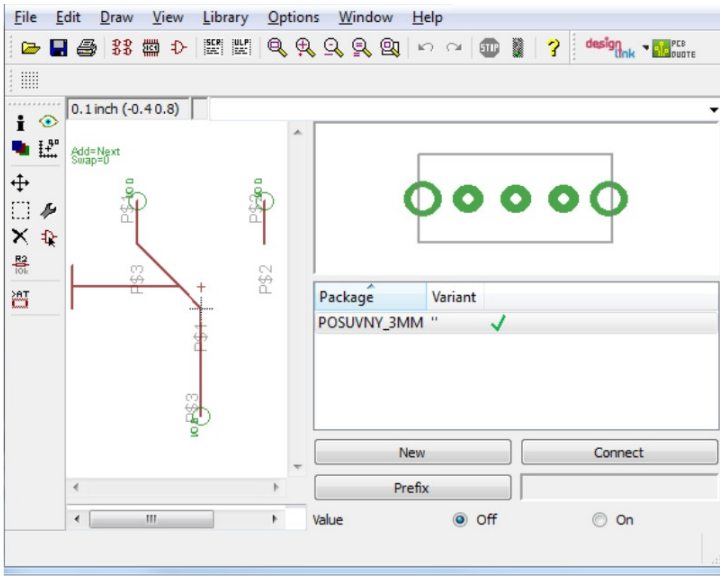
Nyní již máme zapojené nejdůležitější součástky pro funkce testovacího modulu. Zbylé důležité vývody mikrokontroleru už zapojíme dle doporučených zapojení, které můžeme nalézt například na stránkách výrobce mikrokontroleru zde [27]. Jakmile máme celé schéma navržené, můžeme se pustit do návrhu DPS. Výsledné schéma zapojení je k dispozici v přílohách této práce.

3.1.3 Návrh a realizace DPS

Návrhový systém Eagle již v sobě obsahuje poměrně rozsáhlou knihovnu součástek. Přesto můžeme narazit na součástky, které tato knihovna neobsahuje. Při návrhu tohoto konkrétního modulu jsem se setkal pouze s jednou takovou součástkou. Jednalo se o posuvný spínač.

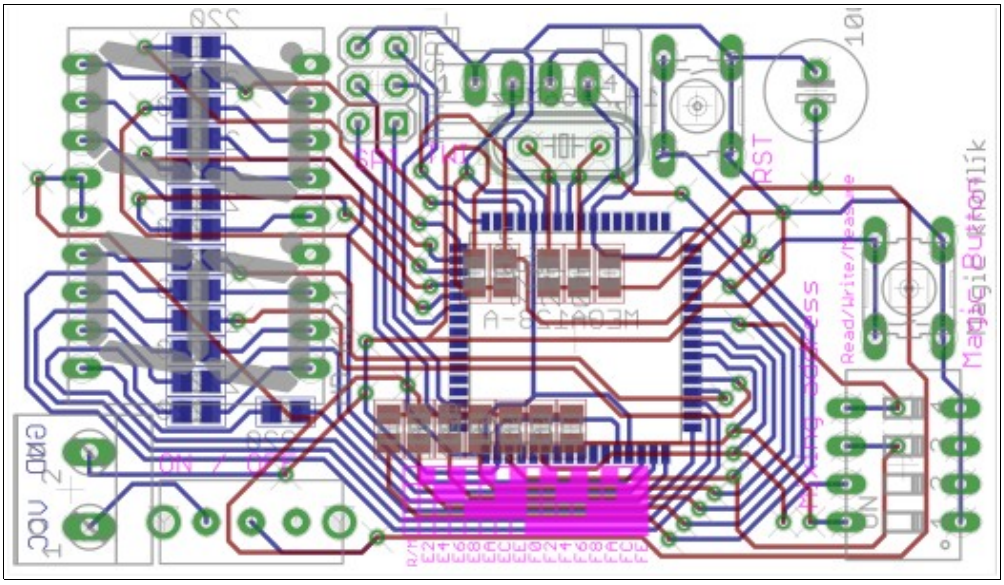
Vznikla tedy potřeba navrhnout součástku, která v knihovnách programu Eagle chybí. K tomu stačí buď v katalogu výrobce najít přesné rozměry součástky a zjistit velikosti děr, které budou potřeba pro součástku vyvrtat, nebo rozměry součástky přímo na ní naměřit. V programu Eagle si pak jen vytvoříme vlastní knihovnu, ve které pro danou součástku vytvoříme schématickou značku a překreslíme její pouzdro včetně pájecích plošek. Když jsou tyto úkony hotové, stačí už jen propojit schématickou značku s pouzdrem, tzn. vývody na návrhu pouzdra propojíme s vývody na návrhu schématické značky. Na Obr. 18 lze vidět okno programu Eagle pro návrh součástky a jejího pouzdra. V této fázi je již vše navrhnuto a pájecí plošky jsou propojeny se schématickou značkou.

Jakmile máme součástku navrženou, stačí už ji jen použít, připojit ve schématu a následně se nám zobrazí i v návrhovém okně pro DPS.



Obr. 18: Návrh součástky chybějící v knihovnách Eagle.

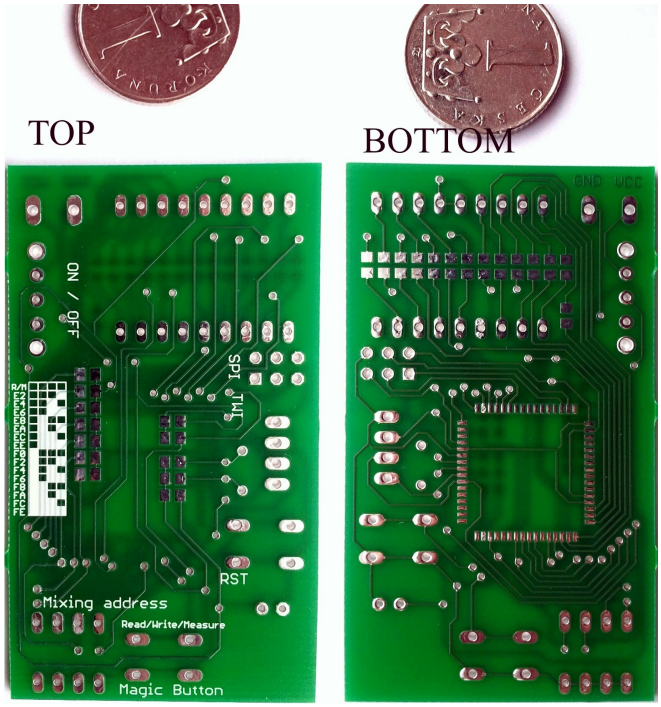
Nyní už jen stačí rozmístit součástky po desce tak, aby nám to vyhovovalo. Konektory je vhodné dávat ke krajům, aby byly snadno přístupné a nepřekážely si mezi sebou. Při návrhu byla požita oboustranná deska, tzn. že se návrh DPS kreslí ve dvou vrstvách, díky čemuž se docílí mnohem menší velikosti DPS. Mimoto jsou v návrhu použity SMD součástky, čímž se dosáhne také mnohem menší velikosti. Jednotlivé strany jsou pak propojeny pomocí prokovů. Je vhodné mikrokontroler umístit na stranu spodní, označované z angličtiny jako „Bottom“, a vrtané, popřípadě zobrazovací součástky na stranu horní, označovanou jako „Top“.



Obr. 19 Navržená DPS pro testovací modul ultrazvukových sonarů SRF08 v měřítku 2:1.

Propojíme veškeré cesty a můžeme ještě přidat popisky s k důležitým součástkám, jakou jsou tlačítka a konektory. Doplnil jsem ještě přímo na plošný spoj tabulku, podle které se pak bude dát řídit při volbě adresy, kterou budeme chtít zapsat do ultrazvukového sonaru.

V přílohách této práce najdete DPS ze strany Bottom a Top zvlášť bez zobrazených součástek. Na Obr. 19 můžeme vidět již hotový návrh DPS. Modře jsou vyznačeny cesty na straně Bottom a červeně na straně Top.

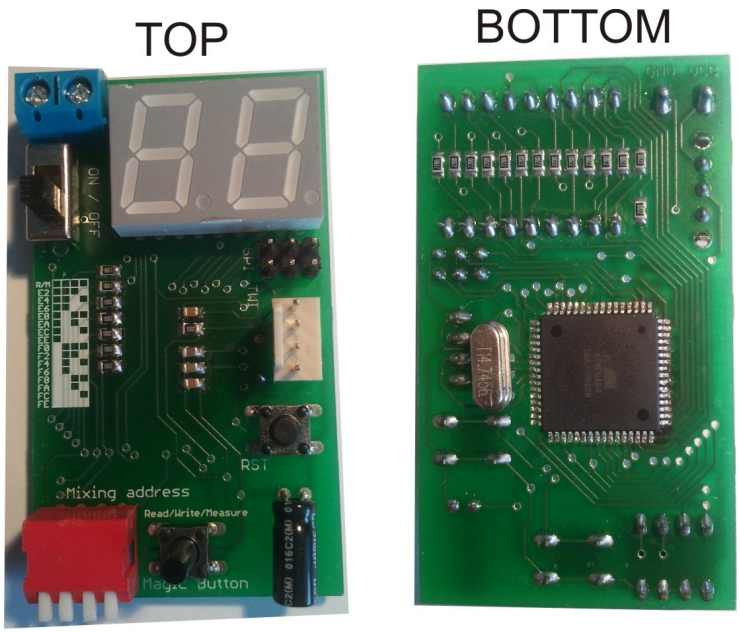


Obr. 20: DPS testovacího modulu pro ultrazvukové sonary SRF08 před osazením, vpravo ze strany Top a vlevo ze strany Bottom.

Nyní již lze DPS vyrobit. První prototypy byly vyráběny fotocestou. Jakmile bylo na prototypech ověřeno, že jsou funkční, nechal jsem vyrobit výslednou desku i s popisky u firmy Pragoboard. Tato deska ještě před osazením součástkami je vidět na Obr. 20, kde můžeme mimo jiné spatřit, že kromě popisků pro konektory a tlačítka je zde i zmiňovaná

tabulka, podle které se pak bude postupovat při volbě adresy určené k zapsání.

Poté je potřeba připájet veškeré součástky na své místo, oživit obvod a jestliže je vše v pořádku může se začít s návrhem programu pro mikrokontroler. Osazenou DPS je možné vidět na Obr. 21 .



Obr. 21: Osazená DPS testovacího modulu ultrazvukových sonarů SRF08, vpravo ze strany Top a vlevo ze strany Bottom.

3.1.4 Program pro mikrokontroler

S ultrazvukovým sonarem budeme komunikovat po rozhraní TWI. Jestliže si spojíme dohromady informace z kapitol 2.2.1 a 2.3.2, víme, že ultrazvukové sonary se budou chovat po rozhraní TWI jako Slave. Abychom k nim mohli přistupovat, kdy budeme chtít, musíme se chovat jako tzv. Master. To znamená, že to budeme my, kdo bude zahajovat a ukončovat komunikaci. Potřebujeme tedy pro mikrokontroler jakési ovladače, říká se jim knihovny, pro rozhraní TWI. Můžeme si je buď napsat sami, nebo využít možností internetu, kde najdeme volně dostupná již hotová řešení. Zvolil jsem velice oblíbené knihovny, které lze najít například zde [28]. V těchto knihovnách je již vše potřebné pro komunikaci naprogramováno. Obsahují i dostatečné komentáře, jak dané knihovny použít. Knihovny tedy tzv. „nainkludujeme“ do našeho projektu a můžeme dále programovat.

Program bude navržen v jazyce C ve vývojovém prostředí AVR Studio dostupném zde [23], jehož instalační soubor v sobě již obsahuje kompilátor GCC. Kompilátorem se vytvoří ze zdrojového kódu tzv. „makefile“, který se ve své podobě nahraje do mikrokontroleru zapojeného v obvodu. Pro nahrání „makefilu“ do mikrokontroleru slouží program AVRdude, jenž se také nainstaluje jako součást AVR studia. Do mikrokontroleru se „makefile“ nahraje sériovým programováním přes rozhraní SPI pomocí programátoru připojeného k počítači. V tomto případě bylo použito programátoru USBasp dostupného v laboratořích Ústavu automatizace a informatiky.

Víme, že budeme potřebovat pomocí mikrokontroleru ovládat display. Nejdříve se musí nadefinovat příslušné piny, popřípadě celé porty, na kterých je display připojen jako výstupní. Poté si vytvoříme funkci, která bude mít dvě vstupní proměnné, abychom pak, když funkci zavoláme, do ní mohli zapsat dvě čísla nebo písmena a ta se nám zobrazí na displayi.

Výsledná funkce pak vypadá podobně jako ve zkrácené ukázce na Obr. 22.

```
void show(char left, char right)          //funkce s 2 vst. prom.
{
        switch(left)         //přepínání pro levý znak
        {
                case '0':     //když bude za prom left dosazena 0
                PORTA = 0b00000011;          //zobrazí se i na dispalayi
                break;
                case '1':
                PORTA = 0b10011111;
                break;
                //takto jsou nadefinovány i ostatní znaky jako: 2,3,F,...
        }

        switch(right)
        {
                //podobně jako u levého znaku
        }
}
```

Obr. 22: Ukázka zdrojového kódu pro ovládání displaye.

Jakmile máme nadefinovány veškeré znaky pro display, můžeme přistoupit k čtyřpólovému přepínači. U něj naopak nadefinujeme piny, na které je připojen jako vstupní. Jak už bylo zmíněno, budeme z něj chtít snímat 16 stavů. 15 z nich budeme mít pro nastavení adres a šestnáctý stav si necháme pro speciální mód, ve kterém budeme číst aktuální adresu z ultrazvuku, popřípadě spustíme sekvenci měření, abychom otestovali ultrazvuk. V Tab. 3 je vidět, jak budeme chtít volit jednotlivé adresy. Tab. 3 je skoro shodná se seznamem adres v Tab. 1. Chybí zde jen adresa E0, která, jak bylo vysvětleno v kapitole 2.2.1, je defaultní a nebudeme ji využívat. Máme již představu o tom, jak budeme adresy přepínat. Program uděláme tak, že když přepneme nějaký přepínač, hned vyhledáme, jakému stavu z tabulky odpovídá a zobrazíme ho na displayi. Bylo by zároveň dobré danou adresu, kterou navolíme, ukládat do globální proměnné. Jakmile stiskneme tlačítko pro potvrzení, budeme chtít rovnou adresu zapsat.

```
void Adress()
{
        if(!(S1)&&!(S2)&&!(S3)&&(S4))       //stav odpovídající prvnímu řádku Tab. 3
        {
                new_adress = 0xE2;    //adresu zároveň uložíme do glob. prom
                show('E', '2');              //funkce pro zobrazení na display
        }
        //podobně bychom nadefinovali všechny zbývající adresy (stavy)
}
```

Obr. 23: Ukázka řešení zobrazení adresy dle stavu DIP 04 a její uložení.

Kód na sledování stavu na čtyřpólovém přepínači a zobrazení adresy odpovídající stavu dle Tab. 3 pak vypadá, jak je vidět na Obr. 23.

ID	Číslo přepínače na DIP 04			
	1	2	3	4
E2	0	0	0	1
E4	0	0	1	0
E6	0	0	1	1
E8	0	1	0	0
EA	0	1	0	1
EC	0	1	1	0
EE	0	1	1	1
F0	1	0	0	0
F2	1	0	0	1
F4	1	0	1	0
F6	1	0	1	1
F8	1	1	0	0
FA	1	1	0	1
FC	1	1	1	0
FE	1	1	1	1

Tab. 3: Přehled stavů čtyřpólového přepínače DIP 04 pro volbu adresy.

Když máme zobrazování na displayi z navolených stavů hotové, musíme tyto hodnoty zapsat do ultrazvukového sonaru. V předchozí ukázce kódu jsme si navolenou adresu ukládali jako globální proměnnou a toho nyní využijeme při uložení adresy do ultrazvukového sonaru. V datasheetu výrobce ultrazvukového sonaru SRF08 [5] se dočteme, že je potřeba po rozhraní TWI, abychom změnili adresu, poslat sekvenci čtyř byte do registru 0, kde poslední byte obsahuje novu adresu. Dle zmíněného datasheetu [5] můžeme změnit adresu z defaultní, tedy 0xE0 třeba na adresu 0xF2 (v kódu se hexadecimální konstanty zapisují tímto způsobem, na začátku se píše 0x a hned následuje daná hodnota), poslali bychom do registru 0 sekvenci 0xA0, 0xAA, 0xA5, 0xF2. První tři byty jsou pevně dány výrobcem a poslední upravujeme dle toho, jakou adresu chceme zapsat. Problém však nastává, jestliže celou tuto sekvenci je potřeba poslat po rozhraní TWI na původní adresu. Může se stát že, nebudeme vědět, jakou adresou zrovna ultrazvuk disponuje, a budeme ji muset zjistit. Předpokládá se totiž, že známe adresu původní. Ta u nového ultrazvukového sonaru SRF08 bude vždycky E0. Ale ne vždy se stane, že budeme potřebovat změnit adresu u nového ultrazvukového sonaru. Je velice pravděpodobné, že budeme muset změnit adresu ultrazvuku, který již byl někdy použitý.

Z výše uvedeného plyne, že řešení, které bude vytvořeno, musí být univerzální. Uživatel, který bude modul používat, se nebude starat o zjišťování adresy. Musíme, navrhnout funkci, pomocí které zjistíme adresu ultrazvukového sonaru. Řešení je ve výsledku poměrně jednoduché. Víme dle Tab. 1, že sonar může mít pouze jednu z šestnácti adres. Veškeré adresy, které může ultrazvukový sonar mít, nadefinujeme a vložíme do pole. K testovacímu modulu budeme mít vždy připojený pouze jeden ultrazvukový sonar. Můžeme zkusit projít pole a zahájit komunikaci s každou z adres, kterou máme nyní uloženou v poli. Pro komunikaci použijeme již zmiňovanou knihovnu [28], se kterou se pokusíme na každou adresu požádat o zahájení komunikace. Když nedojde k odpovědi, zkusíme další adresu.

Jakmile se nám na jednu z nich dostane odezvy, víme, že jsme našli naši adresu, kterou budeme chtít přepsat. Jak je tento problém vyřešen v programu, lze vidět na Obr. 24.

```
void i2c_find_id()
{
	for (int i = 0; i < SRF08_COUNT; i++)	//zkusíme projít celé pole adres
	{
		if (!i2c_start(srf08[i] + I2C_WRITE))//jakmile dostaneme odpověď
		{
			current_adress = srf08[i];	//máme i adresu
		}
	}
}
```

Obr. 24: Ukázka kódu, pomocí kterého lze zjistit aktuální adresu SRF08.

Na ukázce kódu je mimo jiné vidět, že si adresu rovnou uložíme do globální proměnné. Je to z toho důvodu, že se ji pak budeme chystat použít v jiných funkcích. Nyní již máme vše potřebné k tomu, abychom mohli adresu přepsat. Bude to probíhat tak, že uživatel modulu si bude na čtyřpólovém přepínači volit stav podle toho, jakou adresu bude chtít zapsat, a zároveň se mu podle navoleného stavu bude jemu odpovídající adresa ukazovat na displayi. Jakmile ji bude mít navolenou, bude se jen čekat na stisk potvrzovacího tlačítka, čímž se nejdřív spustí funkce na zjištění aktuální adresy, jež se uloží do globální proměnné a následně se spustí již zmíněná posloupnost čtyř bytů na získanou adresu, a tak dojde k jejímu přepsání adresy.

Druhou možností, kde funkci pro zjištění adresy uplatníme, je, když budeme chtít z ultrazvukového sonaru adresu jen přečíst a nepřepisovat ji. K tomu nám může sloužit volný šestnáctý stav na čtyřpóĺovém přepínači DIP 04, kdy jsou všechny kolíbky na přepínači vypnuté, tedy stav 0-0-0-0. Tento stav využijeme pro čtení aktuální adresy a její ukázaní na displayi. Po stisknutí tlačítka v tomto stavu je zavolána funkce na zjištění adresy a získaná adresa je zobrazena. K tomu je použita velice jednoduchá funkce, kde se jen podle zjištěné adresy zavolá již zmíněná funkce pro zobrazení znaků (Obr. 22) a na displayi se ukáže získaná adresa. Takto se uživatel například může ujistit, že při zapsání adresy vše proběhlo v pořádku, popřípadě může jen informativně zjišťovat, jestli všechny ultrazvukové sonary v robotu mají správnou adresu.

Poslední, neméně důležitou funkcí, kterou budeme po testovacím modulu chtít, je, aby ověřil funkčnost ultrazvukového sonaru. To, že ultrazvukovému sonaru lze změnit adresu, ještě neznamená, že je funkční. Nejlepším způsobem, jak bychom mohli funkci ověřit, je zahájit sekvenci měření a pokusit se změřit vzdálenost k nějaké překážce, například ke zdi. Jenže poslední volný stav na čtyřpólovém přepínači již byl použit na čtení adresy. To ale vůbec nevadí. Máme k dispozici potvrzovací tlačítko. Vytvoříme si v programu počítadlo, které bude počítat stisknutí tlačítka v případě, že bude čtyřpólový přepínač ve stavu 0-0-0-0. Pokud bude počítadlo na nule, budeme čekat na stisknutí tlačítka od uživatele. Jestliže bude potvrzovací tlačítko stisknuto, přičteme k počítadlu o jednu vyšší hodnotu a to bude znamenat, že můžeme spustit naši funkci na na vyhledání (Obr. 24) a zobrazení aktuální adresy ultrazvukového sonaru na displayi. Jestliže stiskneme tlačítko znovu a opět přičteme jedničku, bude mít na počítadle hodnotu dva. To bude znamenat, že se spustí sekvence pro měření. Lze ji vidět na Obr. 25. Dle již uváděného datasheetu výrobce ultrazvukového sonaru SRF08 [5] se pro zahájení sekvence měření musí na danou adresu do registru 0 zaslat příkaz s desítkovou hodnotou 81 pro zahájení měření v centimetrech, jak je na Obr. 25 v řádku okomentovaném jako „zahájení měření v cm“. Pro získání hodnot z ultrazvuků je potřeba přečíst dvě hodnoty, a to z vyššího a nižšího bytu , viz Obr. 25, kde oba byty ukládáme do proměnné x. Následně tuto hodnotu ukážeme na displayi a poté ji vynulujeme.

Mezi jednotlivými měřeními je ještě v kódu zařazeno zpoždění 500 ms z toho důvodu, že mezi jednotlivými měřeními by měla být doba minimálně 65 ms, a aby se hodnoty na displayi neměnily příliš rychle a dalo se okem zaznamenat, zda měření probíhá v pořádku.

```
void measure()
{

	i2c_transmit(current_adress, 0, 81); //zahájení měření v cm
	for (int i = 0; i < 50; i++) _delay_ms(10); //zpoždění
	int x = i2c_receive(current_adress,2) << 8;//uložení hodnoty vyšší byte
	x += i2c_receive(current_adress,3);//uložení hodnoty nižší byte
	show_range(x); //zobrazení výsledku na display
	x = 0; //nulování
}
```

Obr. 25: Zdrojový kód pro spuštění sekvence měření ultrazvukového sonaru SRF08 a zobrazení hodnoty na displayi.

Hodnotu získanou z měření pomocí funkce z Obr. 25 je ještě nutné převést na vhodnou formu, abychom pro zobrazení na displayi mohli použít funkci z Obr. 22. Tato funkce pro zobrazení naměřené hodnoty na displayi je na Obr. 26. Nejdříve naměřenou hodnotu převedeme na celé číslo. Máme pouze dvoumístný display a získané hodnoty jsou v centimetrech. To znamená, že nejvyšší hodnota, kterou zobrazíme, bude 99 cm. Jakmile bude naměřená hodnota vyšší, zobrazíme na displayi písmena „EE“, což bude znamenat, že je naměřená hodnota mimo zobrazitelný rozsah. Pro hodnoty menší než 100 cm nejdříve získáme číslo, které budeme chtít zobrazit na levém displayi. To získáme tak, že naměřenou hodnotu vydělíme desítkou. Číslo, které budeme chtít zobrazit na pravém displayi, získáme tak, že na naměřenou hodnotu použijeme operátor modulo 10. Získáme zbytek po dělení deseti. To je hodnota, kterou chceme zobrazit na pravém displayi. Obě hodnoty, nyní už každou zvlášť, už jen převedeme na tvar pro zobrazení na displayi.

```
void show_range(unsigned char rangee)
{
	int range = (int)rangee;
	char left;
	char right;
	if(range >= 100){show('E', 'E');} //když je naměřená hodnota vyšší než 100cm
	if(range < 100) //když je naměřená hodnota nižší než 100cm
	{
		int leftDisp;
		int rightDisp;
		char left;
		char right;
		leftDisp = range / 10;
		rightDisp = range % 10;

		if (leftDisp == 0){left = '0';}
		if (leftDisp == 1){left = '1';}
		//stejně je nadefinováno uložení dalších čísel pro levý segment

		if (rightDisp == 0){right = '0';}
		if (rightDisp == 1){right = '1';}
		//stejně je nadefinováno uložení dalších čísel pro pravý segment

		show(left, right);
	}
}
```

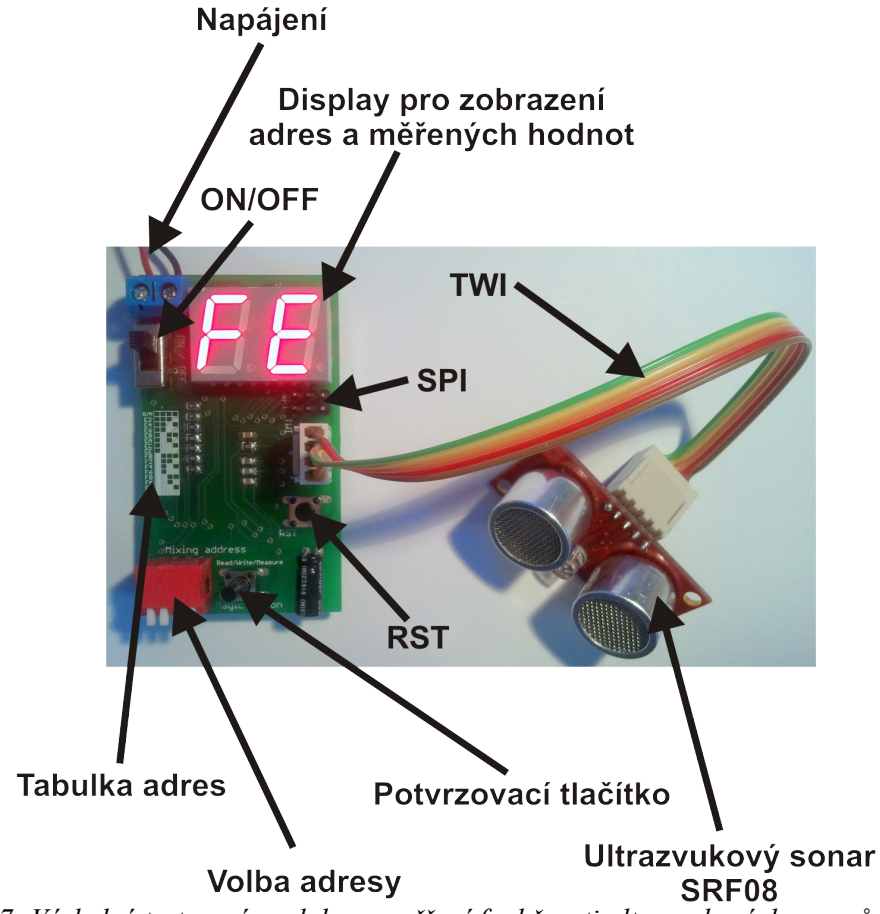
Obr. 26: Funkce převádějící naměřenou hodnotu pro zobrazení na displayi.

Nyní již máme veškeré důležité funkce pro testovací modul hotové, stačí je jen správně použít v hlavní funkci „main“ v programu mikrokontroleru. Následně nahrajeme program do mikrokontroleru a ověříme jeho funkčnost. Samozřejmě samostatné testování funkčnosti probíhalo již při vývoji programu. Každou funkci bylo potřeba po jejím napsání do mikrokontroleru nahrát a ověřit, protože kdybychom napsali program najednou, hledala by se případná chyba velmi těžko, protože ve vývojovém prostředí se případné ladění a krokování programu provádí velice obtížně.

3.1.5 Výsledná funkčnost

Program je v mikrokontroleru již nahraný a můžeme ověřit jeho funkčnost. Testovací modul připojíme k napájení. K TWI konektoru připojíme ultrazvukový sonar. Na čtyřpólovém přepínači DIP 04 si navolíme jednu z patnácti adres, kterou budeme chtít ultrazvukovému sonaru přiřadit. Jestliže zmáčkneme potvrzovací tlačítko, adresa se zapíše.

Na Obr. 27 je možné vidět popis testovacího modulu pro ověření funkčnosti ultrazvukových sonarů. Jestliže je čtyřpólový přepínač DIP 04 v poloze 0-0-0-0, tzn. že jsou všechny jeho prvky ve vypnuté poloze, čeká testovací modul na stisknutí potvrzovacího tlačítka. Při prvním stisknutí ukáže adresu, kterou aktuálně ultrazvukový sonar disponuje, při druhém stisknutí se začne s měřením. Hodnoty na displayi se ukazují v cm až do hodnoty 99 cm. Stačí tedy ultrazvukový sonar namířit přibližně jeden metr od stěny a postupně se k ní s ultrazvukovým sonarem přibližovat. Jestliže se na displayi testovacího modulu naměřené hodnoty postupně snižují, máme ověřeno, že je ultrazvukový sonar funkční a můžeme ho použít v robotu. Popřípadě při hledání nějaké závady můžeme tímto rychle vyloučit, že problém není v ultrazvukovém sonaru, a je potřeba hledat chybu jinde.

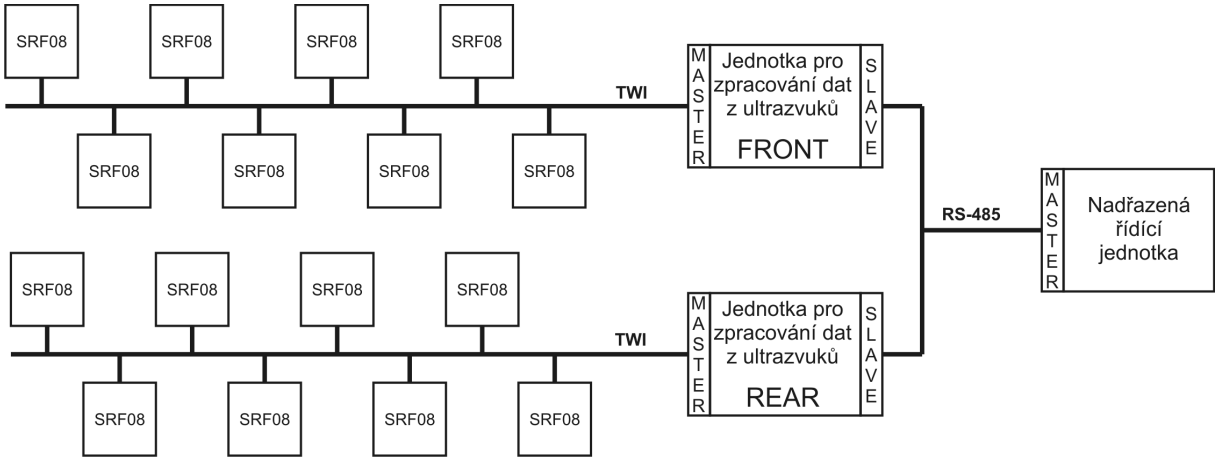


Obr. 27: Výsledný testovací modul pro ověření funkčnosti ultrazvukových sonarů SRF08.

3.2 Testovací modul pro zařízení na zpracování dat z ultrazvukových sonarů

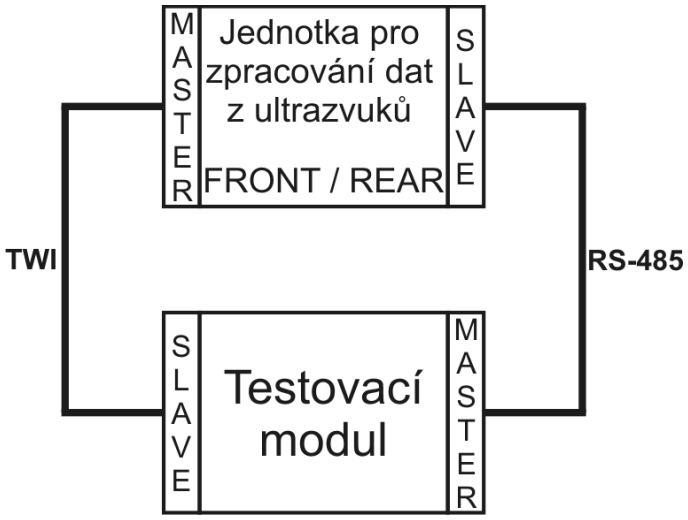
Dalším zařízením, pro které bude potřeba navrhnout testovací modul, je jednotka pro zpracování dat z ultrazvuků. Tato jednotka po rozhraní TWI sbírá data z osmi ultrazvukových sonarů SRF08 připojených k tomuto rozhraní. Tato data pak předává po rozhraní RS-485 (viz kapitola 2.3.3) do své nadřazené řídící jednotky.

Blokové schéma tohoto zapojení v robotu Avdeem lze vidět na Obr. 28. Jednotky na zpracování dat z ultrazvuků jsou v robotu Adveem dvě. Označují se jako Front a Rear. Front sbírá data z ultrazvukových sonarů umístěných vpředu na robotu a Rear provádí sběr dat z ultrazvukových sonarů umístěných v zadní části robotu. Obě jednotky na zpracování dat ze senzorů jsou pak, přes rozhraní RS-485 spojeny s nadřazenou řídící jednotkou, která tato data dále zpracovává a vytváří reakce na případné překážky.



Obr. 28: Blokové schéma zapojení jednotek na zpracování dat z ultrazvuků k ultrazvukový sonarům a k nadřazené řídící jednotce.

Požadavkem je ověřit funkčnost těchto jednotek. To znamená, že po rozhraní TWI se bude simulovat chování osmi ultrazvukových sonarů a zároveň po rozhraní RS-485 si budeme o tato data žádat. Budeme při tom provádět kontrolu, zda si jednotka řekla o data ze všech ultrazvukových sonarů. Po rozhraní RS-485 naopak budeme kontrolovat, jestli se data při přenosu nějakým způsobem nepozměnila, tzn. budeme kontrolovat, zda jednotka dostala data z osmi ultrazvukových sonarů a předává je všechna v neporušené podobě.



Obr. 29: Blokové schéma požadovaného zapojení testovacího modulu k jednotce pro zpracování dat z ultrazvukových sonarů.

Z výše uvedeného plyne, že budeme postupovat velmi podobně jako u předchozího testovacího modulu. Nyní se však nebudeme na rozhraní TWI chovat jako Master, ale jako Slave, protože Master je nyní jednotka, kterou budeme testovat. Budeme se chovat jako osm zařízení typu Slave. Na rozhraní RS485 se však musíme chovat jako Master. Budeme to my, kdo si bude žádat o data. Jak už bylo zmíněno, u veškeré komunikace bude probíhat kontrola. Jakmile by něco v rámci komunikace nebylo v pořádku, budeme tuto skutečnost signalizovat na modulu. Dále by bylo velice užitečné, kdybychom dokázali signalizovat, o jakou jednotku se jedná. To znamená, že budeme signalizovat, jestli se jedná o jednotku Front nebo Rear. Blokové schéma požadovaného zapojení lze vidět na Obr. 29.

Nyní již máme představu o tom, jaké funkce budeme po modulu požadovat a můžeme začít s návrhem a realizací testovacího modulu pro zařízení na zpracování dat z ultrazvukových sonarů.

3.2.1 Volba součástek

U tohoto modulu oproti předešlému nebudeme potřebovat žádná tlačítka. Veškeré funkce začnou probíhat ihned po připojení periferie.

Víme, že budeme potřebovat zprovoznit rozhraní RS-485. Z kapitoly 2.3.1 vyplývá, že je možné rozhraní USART převést pomocí speciálního obvodu na rozhraní RS-485. Pro tento převod byl zvolen obvod s označením SN75176A, který tuto funkci zajišťuje. Veškeré informace o správném zapojení tohoto převodníku lze najít v jeho datasheet zde [29].

Dále je potřeba vyřešit volbu mikrokontroleru. Nyní nám bude stačit mikrokontroler s menším počtem vstupů/výstupů. Pro signalizaci pomocí LED nám bude stačit 6 výstupů. Dále víme, že budeme samozřejmě potřebovat mikrokontroler s rozhraním TWI a USART.

Jako vhodná volba se tedy jeví například Atmega8, ke kterému najdeme datasheet zde [17]. Problém však nastane, když se zamyslíme nad tím, jak pomocí tohoto mikrokontroleru simulovat 8 ultrazvuků. Jestliže použijeme totiž tento mikrokontroler a bude mít nastavenou určitou adresu, pod kterou bude reagovat na požadavky od Mastera, nebudeme už vědět o ostatních, o které si Master bude žádat. Když bude nastavena určitá adresa v mikrokontroleru testovacího modulu, bude dle specifikací k rozhraní mikrokontroleru (dostupné zde [17]), mikrokontroler nastavený jako Slave reagovat pouze na požadavky jemu adresované. Na jakékoliv ostatní požadavky nereaguje. Jedinou výjimkou je tzv. „General call“, který zajišťuje, když Master odešle na adresu 0x00, že na něj musí odpovědět všechna připojená Slave zařízení. Náš problém to ale neřeší, protože mikrokontroler, který je v jednotce na zapracování dat z ultrazvuků, toto volání nepoužívá. Žádá si totiž v cyklu o každou adresu zvlášť.

Jestliže si tedy jednotka pro zpracování dat z ultrazvuků žádá o jednotlivé adresy ultrazvukových sonarů postupně v cyklu, zdálo by se vhodným řešením udělat to úplně stejně, měnit také postupně adresy ve stejném pořadí, v jakém si o ně Master žádá. Kdybychom však toto řešení zkusili použít, zjistíme, že nám nebude fungovat. Jestliže by fungovalo, musela by to být hodně velká náhoda, protože se nám cykly budou míjet. Když budeme zrovna na testovacím modulu mít nějakou z adres, museli bychom mít hodně velké štěstí, aby to byla zrovna ta, o kterou si Master žádá.

Řešením tohoto problému je použití mikrokontroleru Atmega48, který dle svého datasheet dostupného zde [30], má téměř všechny vlastnosti stejné jako zmiňovaný mikrokontroler ATmega8. Má však jednu velice zásadní vlastnost navíc. Tou je registr pro „vymaskování“ adres TWAMR (TWI address mask register), pomocí kterého docílíme toho, že budeme zároveň zahrnovat větší spektrum adres než jenom jednu. Běžně mají mikrokontrolery od firmy Atmel pro nastavení adresy k dispozici jen registr TWAR, do

kterého se nastaví jedna adresa, pod kterou pak mikrokontroler reaguje, jestliže si to Master vyžádá. Pomocí zmíněného dalšího registru TWAMR lze tzv. „vymaskovat“ i další adresy, na jejichž zavolání bude mikrokontroler reagovat.

Ostatní součástky budeme volit podobně jako u předchozího testovacího modulu. Konektory byly zvoleny tak, aby odpovídaly konektorům, které jsou použity na jednotce pro zpracování dat z ultrazvuků. Podobně jako u předchozího testovacího modulu budeme i zde potřebovat další součástky, hlavně jako jsou rezistory a kondenzátory, abychom dodrželi jejich doporučená zapojení.

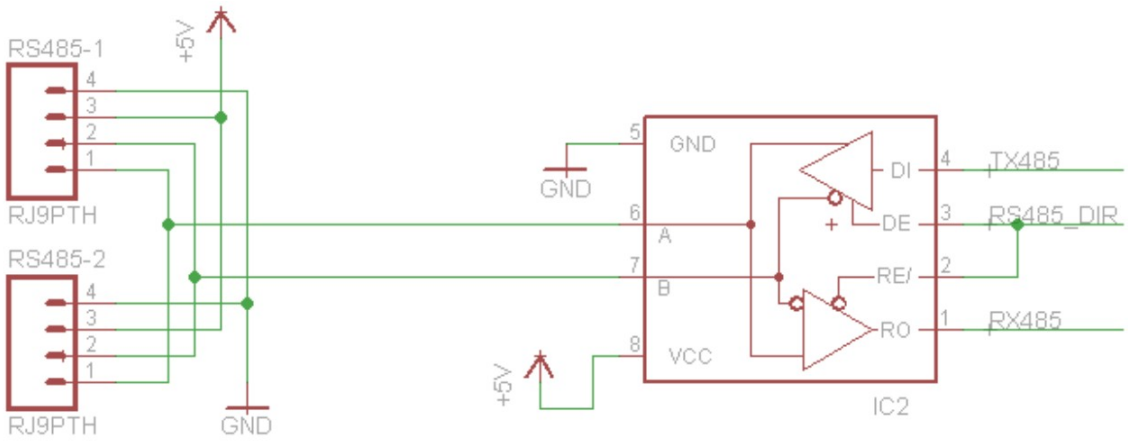
Nyní již máme veškeré důležité komponenty zvoleny a můžeme začít s návrhem schématu.

3.2.2 Návrh schématu

Jakmile máme utvořenou představu o tom, co všechno budeme chtít na testovacím modulu mít, můžeme začít navrhovat schéma. Většina zapojení je velice podobná těm u předchozího testovacího modulu. LED diody jsou zapojeny podobně, jako byl zapojený segmentový display, tzn. že mezi pin mikrokontroleru a katodu LED diody je vždy zapojený rezistor a anoda LED diody je pak připojena ke kladnému pólu napájení.

V návrhu je celkově šest LED diod. Dvě budou sloužit pro rozlišení, jestli se jedná o jednotku pro zpracování dat z předních nebo zadních ultrazvukových sonarů. Další dvě (červená a zelená) budou signalizovat, zda na rozhraní TWI probíhá vše v pořádku. Pokud ano, rozsvítí se zelená LED dioda, a pokud ne, rozsvítí se červená LED dioda. Stejným způsobem bude signalizována funkčnost rozhraní RS-485 zbylými dvěma LED diodami. Pokud nedojde ke komunikaci, hodnoty dat nebudou stejné nebo budou data přicházet nějakým způsobem porušená, rozsvítí se také červená LED dioda. Jestliže bude vše v pořádku, rozsvítí se zelená.

Hlavním rozdílem v zapojení obvodu je přítomnost rozhraní RS485. Rozhraní USART je převedeno na RS-485. Samozřejmě je komunikace obousměrná, takže obvod SN75176A převádí i komunikaci z RS-485 na USART. Z kapitoly 2.3.3 víme, že komunikace po rozhraní RS-485 probíhá po dvoudrátovém vedení. Jedno vedení je označované jako A, druhé jako B. Stejně označené vstupy/výstupy je možné vidět i v zapojení na obrázku Obr. 30. Jsou vyvedeny na konektory, na které je mimo jiné přivedeno i napájení. Pak budeme moci připojeným konektorem rovnou testovací modul i napájet, případně pokud budeme napájet testovací modul jiným konektorem, můžeme rovnou napájet testovanou jednotku na zpracování dat z ultrazvukových sonarů. Na Obr. 30 je dále obvod SN75176A připojen k mikrokontroleru. Vývod označený jako DI je připojen na mikrokontroler na pin TxD a na RxD je připojen vývod RO. Máme tedy připojenou TxD vysílací část a RxD přijímací část rozhraní USART. Dále je vstup/výstup DE a RE připojen na pin PD2 mikrokontroleru. DE a RE je takto zapojen pro přepínání směru komunikace. Poslední, co je z obrázku zřejmé, je, že obvod SN75176A je ještě připojen k napájení.

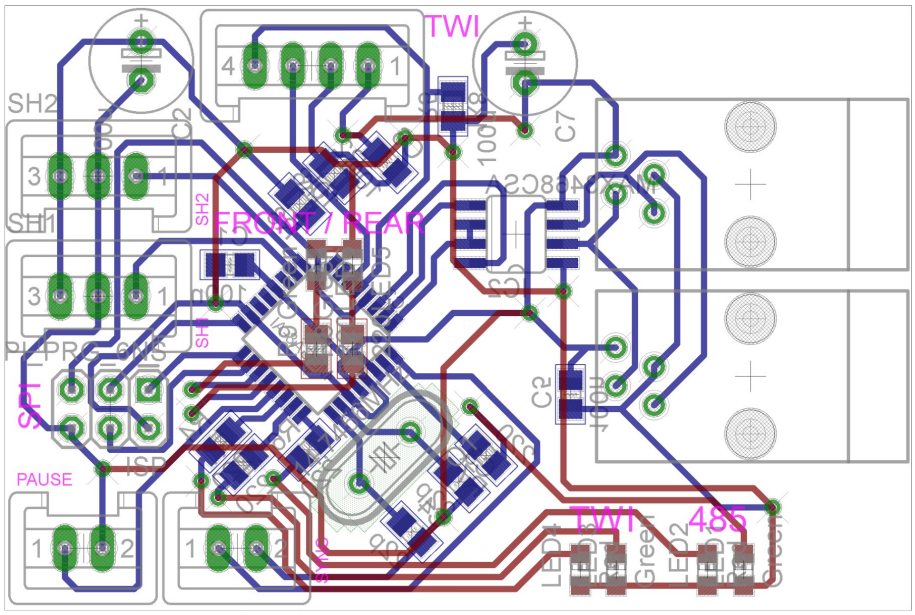


Obr. 30: Zapojení obvodu SN75176Ak mikrokontroleru.

Nyní už jen zapojíme piny SDA a SCL ke konektoru pro rozhraní TWI a také na něj vyvedeme napájení. Kompletní schéma zapojení je možné najít v přílohách této práce.

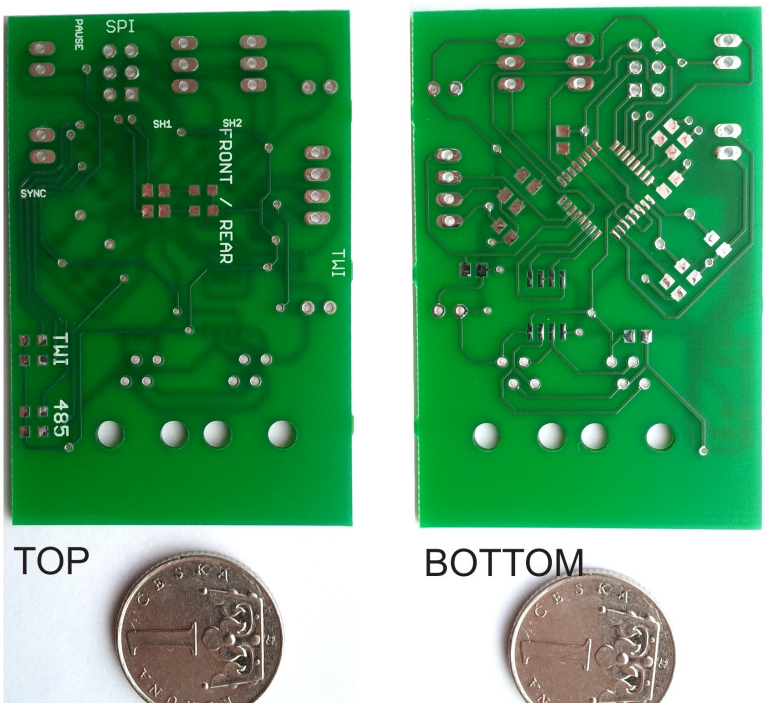
3.2.3 Návrh a realizace DPS

Stejně jako u předešlého testovacího modulu budou i zde kvůli minimalizaci velikosti použity převážně SMD součástky. DPS bude také oboustranná, čímž se samozřejmě také dosáhne menší velikosti.



Obr. 31: Navržená DPS pro testovací modul jednotek na zpracování dat z ultrazvukových sonarů v měřítku 2:1.

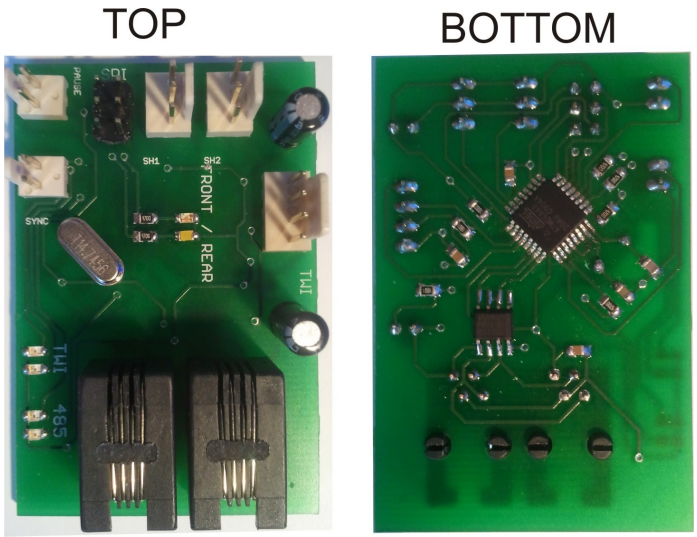
Mikrokontroler bude opět na straně Bottom. Na straně Top budou signalizační LED diody a vrtané konektory rozmístěné po krajích desky. Výsledný návrh z programu Eagle lze vidět na Obr. 31. V přílohách této práce lze nalézt návrhy DPS ze strany Bottom a Top zvlášť bez zobrazených součástek.



Obr. 32: DPS testovacího modulu pro jednotku na zpracování dat z ultrazvukových sonarů před osazením, vpravo ze strany Top a vlevo ze strany Bottom.

Máme již vše připraveno pro osazení DPS součástkami. Všechny součástky umístíme na své místo a následně obvod oživíme. Na Obr. 33 je vidět DPS osazená součástkami.

Nyní je již návrh hotov a je potřeba desku vyrobit. Deska byla vyrobena ručně fotocestou a jakmile byla ověřena funkčnost tohoto prototypu a vyladěny všechny případné chyby, byla stejně jako u předchozího testovacího modulu zadána výroba DPS firmě Pragoboard. Tyto DPS ve stavu před osazením součástkami je možné vidět na Obr. 32. Jedním z hlavních důvodů, proč byly DPS po ověření jejich funkčnosti zadány na profesionální výrobu, je, že tyto DPS mají mnohem vetší životnost (nedochází ke korodování cest) a také jsou mnohem odolnější.



Obr. 33: Osazená DPS testovacího modulu jednotek pro zpracování dat z ultrazvuků, vpravo ze strany Top a vlevo ze strany Bottom.

3.2.4 Program pro mikrokontroler

Podobně jako u předchozího testovacího modulu i zde budeme potřebovat knihovnu pro rozhraní TWI. Nyní je však poměrně velký rozdíl v tom, že bude potřeba se chovat na tomto rozhraní jako několik zařízení typu Slave. Nejdříve jsem se proto pokusil upravit knihovny, které jsem získal zde [31]. Hlavní úprava spočívala v přidání možnosti zápisu do registru TWAMR, díky němuž má být možné „vymaskovat“ adresy, na které bude mikrokontroler reagovat, a tím se chovat jako více zařízení typu Slave.

Po odzkoušení tohoto řešení však bylo však zjištěno, že nereaguje na volání více adres. Reagoval pouze na jednu, a to zapsanou v registru TWAR, kam se při běžném použití knihovny zapisuje jediná adresa, na kterou má mikrokontroler reagovat. Dále byly odzkoušeny ještě další knihovny, které na první pohled vypadaly, že jsou psány trochu jiným stylem, ale nakonec i u nich bylo zjištěno, že nejsou pro toto použití vhodné.

Řešení tohoto problému jsem nakonec našel zde [32]. Jedná se o knihovnu, jejíž hlavní rozdíl oproti předchozím je v tom, že se v ní už dopředu počítá s použitím registru TWAMR. S touto knihovnou už mikrokontroler byl schopen reagovat na volání všech Slave zařízení (tedy ultrazvukových sonarů), o které si volal Master.

Nastavení registru masky adres a registru adresy bylo provedeno, jak je vidět v Tab. 4. K hodnotám zapsaným v těchto registrech dojdeme tak, že nejdříve zjistíme, jaké bity mají adresy společné. To lze zjistit tak, že na všechny adresy, které se používají v jednotkách pro zpracování dat z ultrazvukových sonarů, použijeme logickou funkci AND. Tím získáme hodnotu, která je v Tab. 4 vidět pro nastavení registru TWAR. Je vidět že všechny adresy mají společné tři nejvyšší bity. Stejným postupem i zjistíme v jakých bitech se adresy liší, pouze nyní použijeme logickou funkci OR. Tím získáme hodnotu, kterou zapíšeme do registru TWAMR. Díky takto nastaveným registrům se nám mikrokontroler bude hlásit na volání všech adres, které se používají v jednotkách na zpracování dat z ultrazvukových sonarů.

Popis	Hexadecimálně	Binárně
Nastavení TWAR	E0	1110 0000
Nastavení TWAMR	1E	0001 1110

Tab. 4: Nastavení registru adresy a registru masky adres.

Nyní je ještě potřeba si připravit knihovnu pro rozhraní RS-485. Víme, že toto rozhraní budeme používat prostřednictvím rozhraní USART, kterým disponuje mikrokontroler. Knihovnu je však pro tuto aplikaci potřeba upravit. Takto upravenou knihovnu jsem získal od specialisty z firmy Bender Robotics [33]. Jedná se o upravenou knihovnu od [28], kde hlavní úprava spočívá v přepínání směru komunikace pomocí pinu PD2 mikrokontroleru připojeného k obvodu SN75176A.

Nyní máme vše potřebné pro komunikaci s periferií jak přes rozhraní TWI, tak přes rozhraní RS-485. Knihovny je jen potřeba do projektu „nainkludovat“ a poté můžeme při psaní programu využívat jejich funkce.

Na rozhraní TWI jsme schopni odpovídat na jakoukoliv adresu, o kterou si jednotka na zpracování dat z ultrazvuků zažádá. Je však ještě potřeba vyřešit, abychom měli přehled, o kterou z adres bylo zažádáno. Nyní bychom totiž byly schopni pouze odpovědět a poslat nějaká data, když si o ně Master zažádá. Nevíme však pro jakou adresu tento požadavek přišel. Víme jen, že to může být jedna z „vymaskovaných“ adres. Řešení tohoto problému jsem našel zde [34], kde je popsáno, že v určité chvíli při vykonávání rutin, které obstarává knihovna k rozhraní TWI, je v určité chvíli registru TWDR uložena adresa Slave zařízení,

které bylo osloveno od Mastera. Toho využijeme tak, že vždy když se v rutině dojde na toto místo, si adresu z registru TWDR uložíme do globální proměnné.

Jestliže můžeme číst adresy, o které si žádá jednotka pro zpracování dat z ultrazvukových sonarů a víme, že Rear si žádá postupně o data z 8 adres v přesně daném pořadí a Front také, pouze o jiné. Nadefinujeme si proto tyto adresy jako konstanty a uložíme je do dvou polí. Jedno pole je pro Rear a druhé pro Front. Budeme chtít adresy, o které si Master žádá uložit do pole a pak toto porovnat s polem, kde máme uloženy adresy Rearu, popřípadě Frontu. Tak budeme moci signalizovat, o kterou z jednotek pro zpracování dat z ultrazvukových sonarů se jedná. Na Obr. 34 je vidět funkce, kterou voláme z místa, kde je v registru TWDR uložena adresa volaná Masterem. Nejdříve probíhá kontrola, jestli se volaná adresa liší od předchozí volané. To je z důvodu, aby se některá adresa neuložila dvakrát nebo dokonce vícekrát za sebou. Pokud se stará a nová adresa liší, nastaví se příznak změny na hodnotu 1 a adresa se zapíše do globální proměnné. Jestliže ke změně nedošlo, přepíše se příznak změny na hodnotu 0.

V dalším kroku na Obr. 34, se kontroluje pomocí funkce „isItBegin“, zda adresy uložené v globálních proměnných „newAddress“ a „oldAdress“ se shodují s prvními dvěma adresami, o které si má žádat Rear nebo Front. Jestliže je tato podmínka splněna a zároveň má příznak změny hodnotu 1, uloží se tyto první dvě adresy do pole. Zároveň je do globální proměnné „startedIt“ zapsána hodnota 1. Ta slouží jako příznak pro další krok. Dále se v tomto kroku vynuluje příznak změny, protože došlo k uložení adres do pole.

```
void saveAdressToArray(char addr)
{
	if (addr != oldAddress) //kontrola jestli se nová adresa liší od staré
	{
		newAddress = addr;
		indexOfChange = 1; //pokud se liší je index změny 1
	}
	if (addr == oldAddress)
	{
		indexOfChange = 0; //pokud se neliší je index změny 0
	}
	if ((indexOfChange == 1)&&(isItBegin() == 1)) //začátek Rearu nebo Frontu
	{
		addressArray[0] =  oldAddress; //uložení první adresy na první pozici
		addressArray[1] = newAddress; //uložení druhé adresy na druhou pozici
		startedIt = 1; //příznak začátku ukládání adres
		indexOfChange = 0;
	}
	if((startedIt == 1)&&(indexOfChange == 1)) //pokud už začalo a proběhla změna
	{
		for (int i = 2; i <= 7; i++)
		{
			if (addressArray[i] == 0) //vyhledání poslední volné pozice
			{
				addressArray[i] = newAddress; //uložení nové adresy
				indexOfChange = 0;
				break;
			}
		}
	}
	oldAddress = newAddress;
}
```

Obr. 34: Funkce pro uložení adres, o které si žádá od jednotky na zpracování dat z ultrazvuků.

V posledním kroku této funkce pro uložení adres se kontroluje, zda již proběhl předchozí krok pomocí zmiňovaného příznaku „startedIt“. Pokud ano a zároveň došlo

ke změně adresy, najde se poslední volná pozice v poli a uloží se do ní nová adresa. Jakmile je uložena, nastaví se příznak změny zase na nulu, protože nová adresa je již uložena v poli.

Po proběhnutí všech výše zmiňovaných kontrol, případně uložení adres, je možné prohlásit novou adresu za starou a poté čekat až dojde ke změně.

V dalších funkcích provedeme kontrolu, zda je již pole naplněno adresami. Pokud je již plné, porovnáme každou z adres v tomto poli s adresami, o které si má žádat Front a Rear, uloženými také v polích. Na Obr. 35 je možné vidět tuto funkci, která se stará o rozsvícení příslušné LED diody, podle toho, zda se jedná o Rear nebo Front. Toto vyhodnocení probíhá ve funkci nazvané jako „addressMatch“. Jestliže dané vyhodnocení nevyhovuje uloženým adresám pro jednotku Rear ani Front, rozsvítí se červená dioda indikující chybu v rozhraní TWI.

Knihovny, které jsou použity pro rozhraní TWI [32], poskytují mnoho kontrol chybových stavů, které mohou nastat při komunikaci po tomto rozhraní, například pokud se zpráva neodeslala celá. Tohoto využijeme při indikaci chybových stavů. Do připravených nežádoucích stavů umístíme signalizaci chyby na rozhraní TWI. Jestliže komunikace po rozhraní TWI probíhá v pořádku a jednotka pro zpracování dat z ultrazvuků si žádá neustále dokola o data ze všech osmi ultrazvukových sonarů, rozsvítíme pro rozhraní TWI zelenou LED diodu.

```
void magicFunction()
{
	char match = addressMatch();

	if (match==1)
	{
		FRONT_LED_ON; //indikace že se jedná o Front
		REAR_LED_OFF;
		adress485 = RS485_FRONT_ADDRESS; //RS-485 adresa pro Front
		tim1Start();
		sei();
	}
	if (match==2)
	{
		FRONT_LED_OFF;
		REAR_LED_ON; //indikace že se jedná o Rear
		adress485 = RS485_REAR_ADDRESS; //RS-485 adresa pro Rear
		tim1Start();
		sei();
	}
	indexOfChange = 0;
}
```

Obr. 35: Funkce pro indikaci druhu připojené periferie pro zpracování dat z ultrazvukových sonarů.

Stejně jako knihovny pro rozhraní TWI poskytují prostředky pro kontrolu chybových stavů, poskytují tuto možnost i knihovny pro rozhraní RS-485 [33]. Proto na všechna místa, kde v kódu při provádění rutin pro komunikaci dojde k chybovému stavu, umístíme indikaci chyby na rozhraní RS-485.

V kódu na Obr. 35 je vidět, že se zapisuje i adresa pro rozhraní RS-485. Důvodem je potřeba se chovat na sběrnici RS-485 jako zařízení typu Master. Budeme tedy podřízené jednotky oslovovat jejich adresou. Ve funkci na Obr. 35 proto přiřadíme adresu odpovídající příslušné jednotce pro zpracování dat z ultrazvukových sonarů typu Front nebo Rear a pro jejich rozpoznání.

Při ověřování funkčnosti testovacího modulu bylo zjištěno, že testovací modul si žádal o data z jednotek pro zpracování dat z ultrazvukových sonarů dříve, než je měl všechna

zpracovaná. Z toho důvodu je testovací modul po rozhraní RS-485 přijímal neúplná. Proto je ve funkci na Obr. 35 pro uložení adresy pro rozhraní RS-485 zvolena funkce „tim1start“ a následně povoleno přerušení. Pomocí funkce „tim1start“ se spustí časovač, který běží nezávisle na běhu programu, protože se jedná o vnitřní funkci mikrokontroleru. To znamená, že program běží dál, dokud běží časovač. To, jak časovač nastavit podle vlastních požadavků, se vždy dozvíme v datasheetu výrobce daného mikrokontroleru. V našem případě pro mikrokontroler Atmega48 je datasheet k dispozici zde [30]. Na Obr. 36 lze vidět nastavení časovače použitého v programu pro testovací modul. Časovač je nastaven tak, že čítá nahoru a až dojde na nastavené hodnoty, může se vyvolat přerušení. Zde je dělička nastavena tak, aby k přerušení došlo za 200 ms.

```
void tim1Start(void)
{
	TCCR1A |= (1<<WGM11); //nastavení registrů čítače
	TCCR1B |= (1<<WGM13) | (1<<WGM12); //nastavení registrů čítače
	TCCR1B |= (1<<CS11) | (1<<CS10); //delicka nastavena na 64
	ICR1 = 46080-1;//nastavitelná hodnota;
	TIMSK1 |= 1<<TOIE1; //povolí přerušení od čítače
}
```

Obr. 36: Nastavení čítače pro zpoždění komunikace po RS-485.

Po přetečení časovače dojde k vyvolání přerušení, tzn. zavolá se příslušný vektor přerušení (obsluha přerušení), viz Obr. 37. Do obsluhy přerušení se nesmí dávat žádné složitější funkce. Z tohoto důvodu je zde jen nastavován příznak, který se pak kontroluje, zda se změnil.

```
ISR(TIMER1_OVF_vect)
{
	flag485 = 1;
	TCCR1B &= ~((1<<CS11) | (1<<CS10));
}
```

Obr. 37: Obsluha přerušení.

Kontrola, jestli již došlo k přerušení, je vidět na Obr. 38. Tato podmínka je v kódu umístěna přímo v těle hlavního cyklu „while“, kde jsou samozřejmě umístěny i veškeré ostatní již zmiňované funkce. Po provedení přerušení pokračuje program tam, kde skončil před tím, než bylo vyvoláno přerušení. Jakmile dojde k této podmínce, vynuluje příznak, že došlo k přerušení a poté již zahájí komunikaci po rozhraní RS-485. Když celá komunikace proběhne, může dojít k vynulování pole adres, které byly uloženy pomocí funkce z Obr. 34.

```
if (flag485 == 1)
{
	flag485 = 0; //vynulování příznaku
	RS485comunication(); //zahájení komunikace po RS-485
	zeroArray(); //vynulování pole uložených adres
}
```

Obr. 38: Zahájení komunikace po RS-485.

Funkce pro komunikaci po rozhraní RS-485 je poměrně složitá, proto je na Obr. 39 možno vidět její nejdůležitější část, kde dochází k ověření přijatých dat. Nejdříve se kontroluje, jestli adresa z pole uložených adres souhlasí s příslušnou pozicí přijatých dat po RS-485. Data jsou po rozhraní TWI zasílána tak, aby se nyní dalo snadno ověřit, zda nedochází při komunikaci k jejich zkreslování . K hodnotám dat zasílaných po TWI je zaslána adresa s přičtenou hodnotou o jedna vyšší, než je hodnota adresy. Jakmile je celá

podmínka splněna, rozsvítí se signalizace, že je na rozhraní RS-485 vše v pořádku, tzn. zelená LED dioda. Pokud by některá data nevyhovovala podmínce, rozsvítí se červená LED dioda, která signalizuje, že na rozhraní RS-485 došlo k chybě.

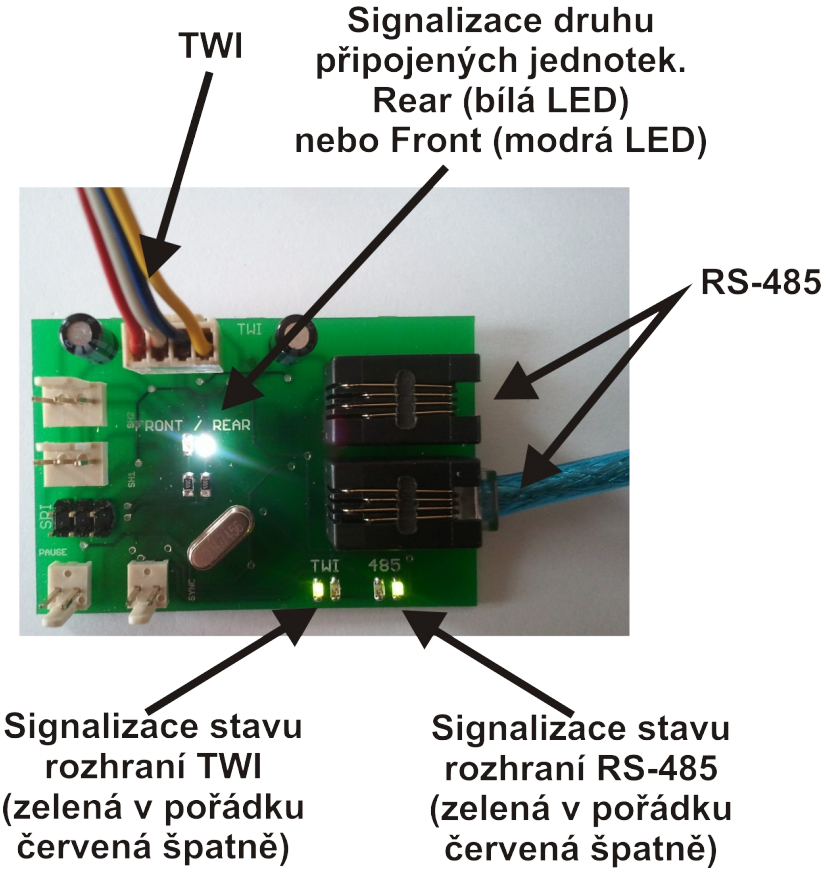
```
for (i = 0; i < data[0]; i++)
{
	if ((addressArray[i] == data[i*3+1])
		&&((addressArray[i] + 2) == data[i*3+2])
		&&((addressArray[i] + 3) == data[i*3+3]))
			//ověření adres a příchozích dat
	{
		RS485good();
	}
	else
	{
		RS485wrong();
		break;
	}
}
```

Obr. 39: Ověření dat zaslaných po RS-485.

3.2.5 Výsledná funkčnost

Výsledný program nahrajeme do mikrokontroleru a ověříme jeho funkčnost. Stejně jako u předchozího testovacího modulu i zde bylo potřeba program ověřovat a ladit již během vývoje programu.

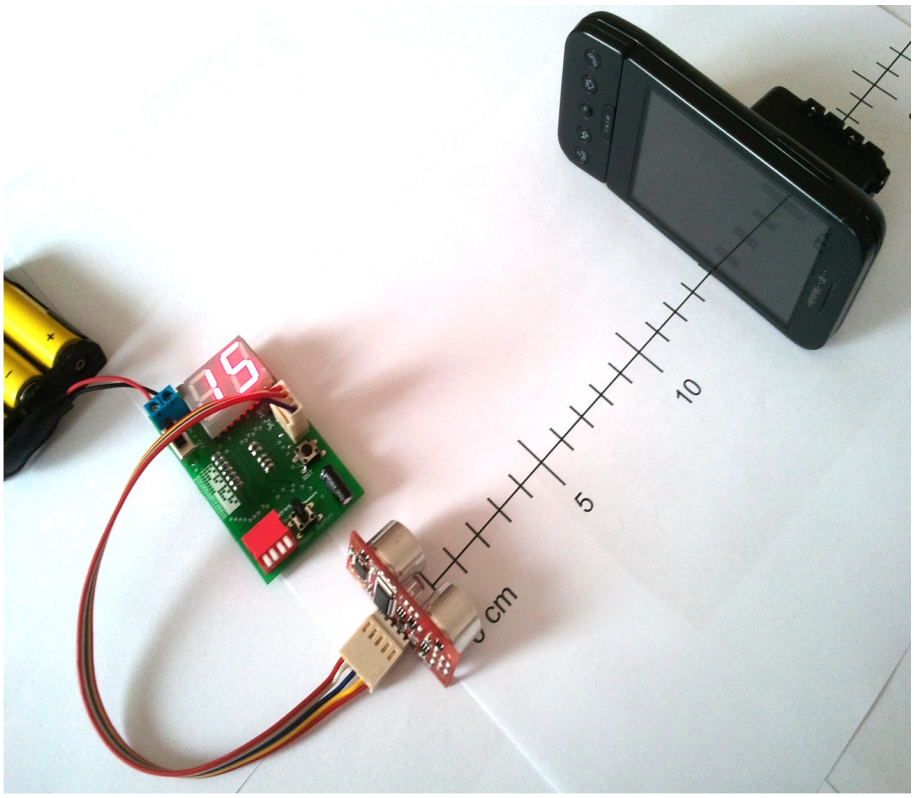
Na Obr. 40 lze vidět popis testovacího modulu jednotek na zpracování dat z ultrazvuků. Testovací modul je schopný rozeznávat, jestli se jedná o jednotku pro zpracování dat z ultrazvukových sonarů typu Rear nebo Front. Dokáže také signalizovat chyby při komunikaci po rozhraní TWI. V případě, že nastane jakákoliv chyba při komunikaci po tomto rozhraní, rozsvítí se červená LED dioda. Červená LED dioda se rozsvítí i v případě, že jednotka pro zpracování dat z ultrazvuků vynechá některou z adres a nebo si případně zažádá o jinou, než která má vyhovovat seznamu adres, o které si žádají Rear a Front. V případě, že veškerá komunikace po tomto rozhraní probíhá v pořádku, rozsvítí se zelená LED dioda. Stejně je signalizovaný i stav rozhraní RS-485. Jestliže na něm dochází k chybám nebo nejsou přijata stejná data, která byla zaslána po rozhraní TWI, rozsvítí se červená LED dioda. Jestliže veškerá komunikace po tomto rozhraní i přijatá data jsou v pořádku, rozsvítí se zelená LED dioda. Jestliže svítí obě zelené LED diody u obou rozhraní, je testovací modul v pořádku.



Obr. 40: Popis testovacího modulu pro jednotky na zpracování dat z ultrazvukových sonarů.

4 FUNKCE A POUŽITÍ TESTOVACÍCH MODULŮ

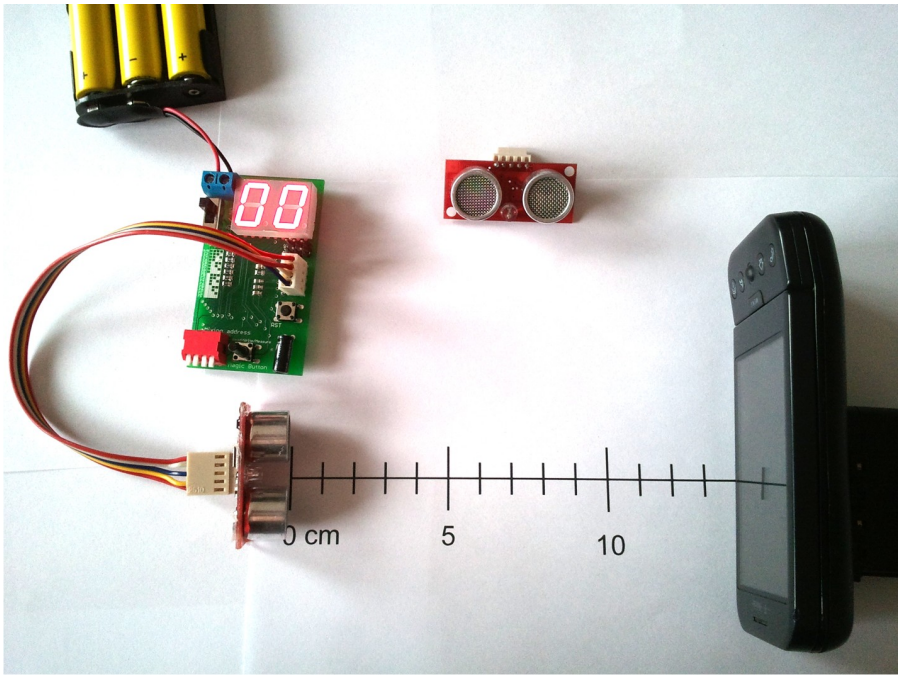
Pomocí testovacího modulu pro ověření funkčnosti ultrazvukových sonarů lze snadno rozpoznat vadný ultrazvukový sonar typu SRF08. Na Obr. 41 je možné vidět část průběhu testování ultrazvukového sonaru SRF08. Do tohoto ultrazvukového sonaru šlo zapsat jednu z navolených adres na testovacím modulu připojeném k ultrazvukovému sonaru. Následně ve vypnutém stavu všech spínačů na DIP04 se po prvním stisku tlačítka zobrazila adresa, která byla předtím zapsána. Tím je potvrzeno, že z ultrazvukového sonaru lze číst a zapisovat adresy. Po dalším stisku potvrzovacího tlačítka se spustila funkce měření. Před ultrazvukový sonar byla umístěna překážka, se kterou bylo pohybováno a hodnoty na displayi se zmenšovaly při přibližování překážky k ultrazvukovému sonaru a zvětšovaly až do hodnoty 99 cm. Následně se na displayi objevily znaky „EE“, což znamená, že hodnota je mimo zobrazitelný rozsah displaye. Hodnoty zobrazované na displayi velmi přesně korespondovaly se skutečnou vzdáleností. Jak je vidět na Obr. 41, překážka od ultrazvukového sonaru je vzdálena 15 cm a na displayi se ukazuje také hodnota „15“. Díky výše zmíněným testům bylo možné tento ultrazvukový sonar prohlásit za plně funkční.



Obr. 41: Test ultrazvukového sonaru SRF08, který je funkční.

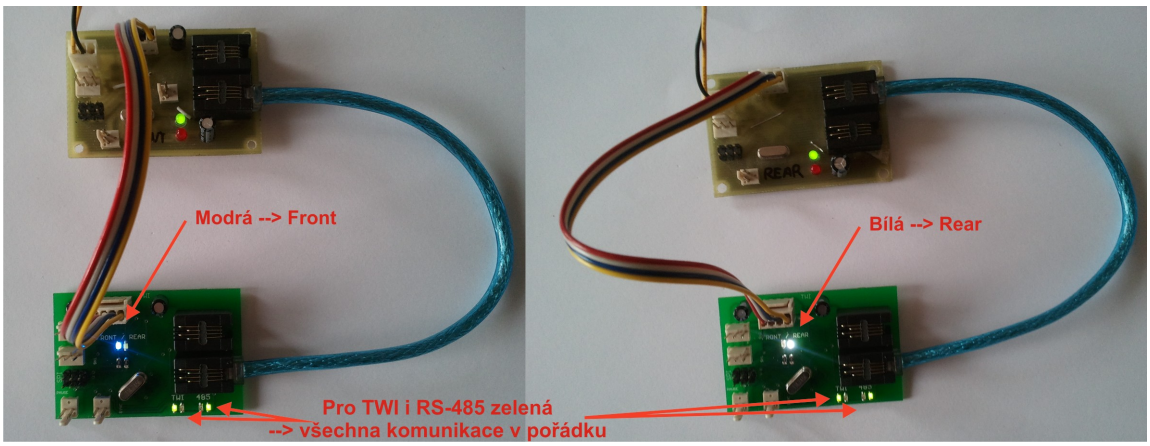
Při realizaci testovacího modulu pro ověření funkčnosti ultrazvukových sonarů jsem dostal k dispozici i jeden z ultrazvukových sonarů, který se v robotu Adveem porouchal. Část jeho testování je možné vidět na Obr. 42. Při testování tohoto ultrazvukového sonaru bylo zjištěno, že je možné do něj zapsat novou adresu. Také bylo možné následně adresu přečíst. Do tohoto ultrazvukového sonaru tedy bylo možné stejně jako u předchozího zapisovat a číst adresy. Po spuštění sekvence měření však byla odhalena testovacím modulem jeho chyba. Při pohybování s překážkou se na displayi ukazovala stále stejná hodnota, ať byla překážka jakkoliv daleko nebo blízko. Hlavní problém je, že hodnota, kterou ultrazvukový sonar posílá, je nulová. To může robotu znemožnit pohyb, protože bude stále ve svém okolí vyhodnocovat

překážku. Oproti předchozímu ověření funkčnosti ultrazvukového sonaru z Obr. 41 je možné tento připojený k testovacímu modulu z Obr. 42 prohlásit za nefunkční.



Obr. 42: Test ultrazvukového sonaru SRF08, který je nefunkční.

Nyní je potřeba ověřit funkčnost testovacího modulu pro jednotky na zpracování dat z ultrazvukových sonarů. Na Obr. 43 lze vidět ověření funkčnosti obou jednotek na zpracování dat z ultrazvukových sonarů, a to jak typu Rear, tak Front. Obě si po rozhraní TWI žádala o data z osmi ultrazvukových sonarů, která mají přidělená. Díky tomu došlo u obou k rozeznání, o jaký typ se jedná, viz Obr. 43. Dále je vidět, že jak pro Front, tak pro Rear svítí zelené LED diody na signalizaci obou periferií. To znamená, že na rozhraní TWI ani na rozhraní RS-485 nedošlo k žádnému chybovému stavu.

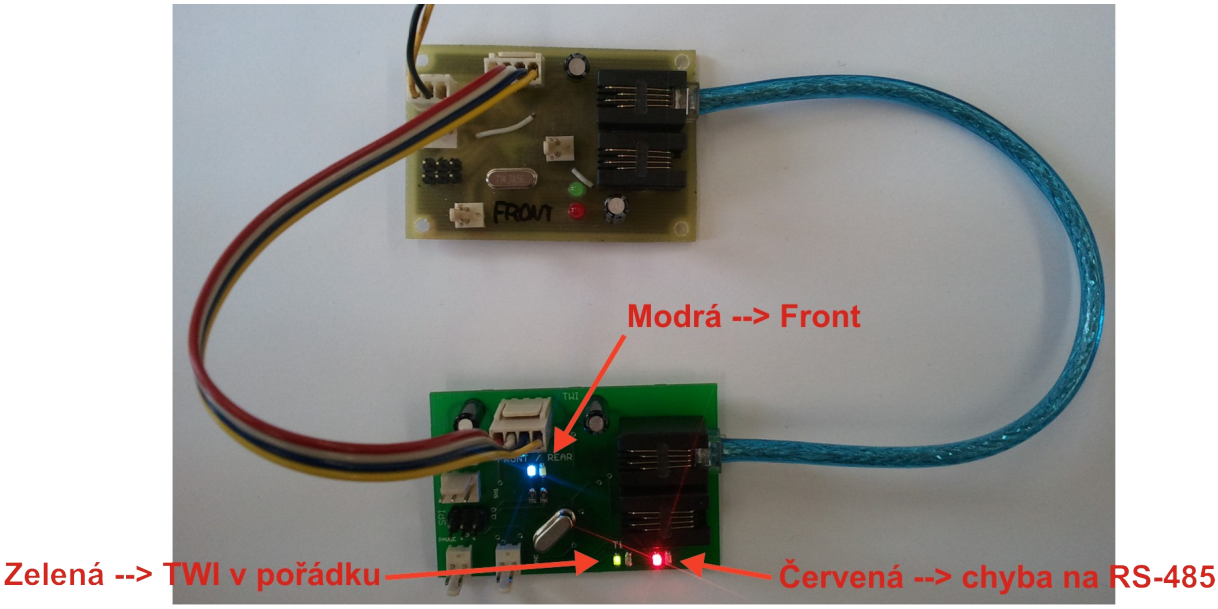


Obr. 43: Test jednotek na zpracování dat z ultrazvukových sonarů (vlevo jednotka Front, vpravo jednotka Rear).

Za spolupráce specialisty z firmy Bender Robotics, bylo nasimulováno několik možných chybových stavů, které mohou nastat na jednotkách pro zpracování dat z ultrazvukových sonarů.

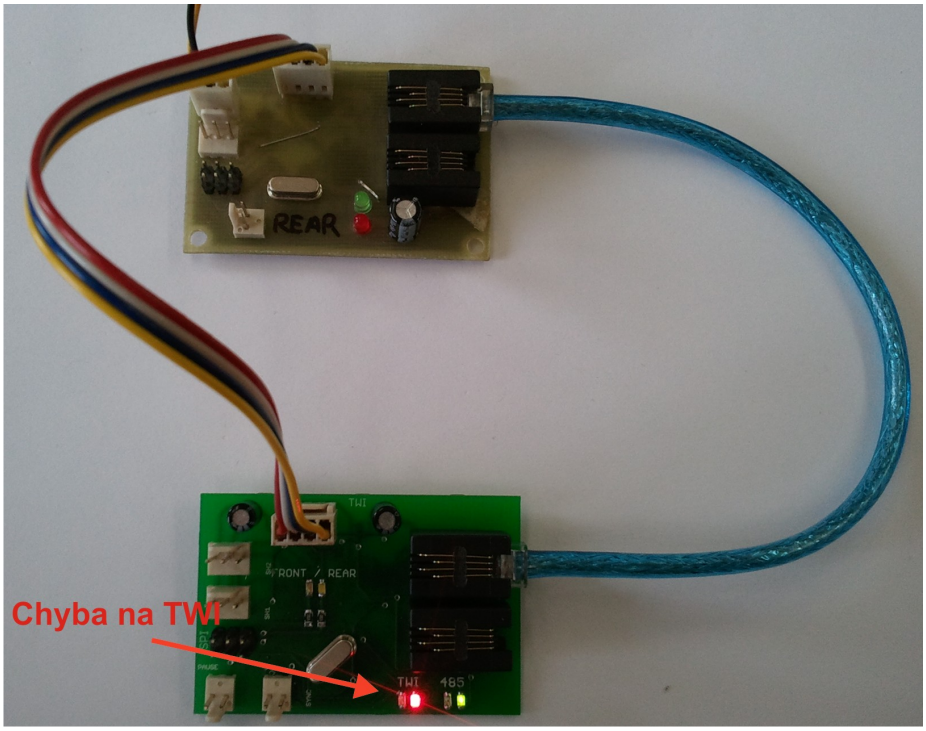
Nejdříve byl nasimulován stav, kdy má na rozhraní TWI probíhat vše správně a jednotka pro zpracování dat z ultrazvukových sonarů si žádá o data ze všech osmi ultrazvukových sonarů. Pro rozhraní RS-485 je následně posílá zkreslená. Reakci testovacího

modulu na tento stav lze vidět na Obr. 44. Tento chybový stav byl testovacím modulem odhalen a signalizován.



Obr. 44: Ověření funkčnosti detekování chybových stavů na rozhraní RS-485.

Dále byla nasimulována situace, kdy jednotka na zpracování dat z ultrazvukových sonarů vynechala žádost o data z jednoho z ultrazvukových sonarů. Detekci této chyby lze vidět na Obr. 45.



Obr. 45: Oěření funkčnosti detekování chybových stavů na rozhraní TWI

Jestliže byla vynechána jedna z adres, ze které má jednotka pro zpracování dat z ultrazvukový sonarů sbírat data, zobrazí se chyba na rozhraní TWI, jak je vidět na Obr. 45. Mimo jiné si lze všimnout, že se nerozsvítila signalizace pro rozlišení jednotky Front nebo Rear. Je to z toho důvodu, že seznam adres není úplný, a v takovém případě neodpovídá ani jednomu seznamu adres pro Front nebo Rear.

5 ZÁVĚR

Jedním z cílů diplomové práce bylo seznámit se s periferiemi používanými v mobilní robotice. Základní rozdělení periferií pro mobilní robotiku je v kapitole 2.2. Každá periferie disponuje rozhraním, pomocí kterého probíhá komunikace mezi periferií a jeho nadřízeným zařízením případně více zařízeními. Rozhraní pro periferie, důležitá pro tuto práci, jsou popsána v kapitole 2.3. Na základě získaných znalostí bylo možné navrhnout a realizovat testovací moduly.

Pro oba testovací moduly bylo zapotřebí na základě požadované funkce nejdříve zvolit vhodné součástky a navrhnout zapojení s mikrokontrolerem. Následně byla navrhnuta deska plošných spojů pro tato zapojení. Poslední důležitou částí bylo navrhnout program pro mikrokontroler.

V kapitole 3.1 je popsán postup vývoje testovacího modulu ultrazvukových sonarů SRF08. S těmito ultrazvukovými sonary jsme se blíže seznámili v kapitole 2.2.1. Tento ultrazvukový sonar komunikuje po rozhraní TWI, které bylo představeno v kapitole 2.3.2.

V robotu, pro kterého je testovací modul ultrazvukových sonarů určen, se používá pro detekci překážek většího množství ultrazvukových sonarů. Každý ultrazvukový sonar se na rozhraní TWI rozlišuje svou unikátní adresou. Adresu je třeba po TWI nejdříve nastavit místo adresy z výroby. Na rozhraní TWI se nesmí vyskytnout dva sonary se stejnou adresou. Aby případná výměna vadného ultrazvukového sonaru probíhala co možná nejjednodušeji, je možné si na testovacím modulu navolit adresu, kterou přiřadíme ultrazvukovému sonaru. Testovacím modulem lze adresu i přečíst, čímž se ověří správnost nastavené adresy. Pro ověření funkčnosti ultrazvukového sonaru lze spustit sekvenci měření. Aktuální měřená hodnota se pak ukazuje na displayi. V kapitole 4 byl proveden experiment, že takto lze snadno odhalit vadný ultrazvukový sonar. Testovací modul pro ultrazvukové sonary umožňuje:

- navolení jedné z patnácti adres a zapsání této adresy do ultrazvukového sonaru
- čtení adresy z ultrazvukového sonaru
- ověření funkčnosti ultrazvukového sonaru
- zobrazování všech funkcí za pomocí displaye

Dále bylo potřeba navrhnout a realizovat testovací modul pro jednotku na zpracování dat z ultrazvukových sonarů. Jednotka sbírá data z ultrazvukových sonarů prostřednictvím zmiňovaného rozhraní TWI. Data pak dále předává své řídící jednotce prostřednictvím rozhraní RS-485 popsaném v kapitole 2.3.3. Jednotky jsou v robotu dvě. Jedna se stará o sběr dat z přední části robotu a druhá ze zadní části. Na testovacím modulu je možné signalizovat, jestli komunikace s touto periferií probíhá v pořádku. V kapitole 4 bylo experimenty provedeno ověření funkčnosti odhalení vadné periferie. Testovací modul pro jednotku na zpracování dat z ultrazvukových sonarů umožňuje:

- rozlišení jednotek pro zpracování dat z ultrazvukových sonarů umístěných na přední část robotu nebo zadní části
- odhalení chybových stavů při komunikaci po rozhraní RS-485
- odhalení chybových stavů při komunikaci po rozhraní TWI
- kontrolu zpracovaných dat
- kontrolu, zda dochází k požadavkům o data ze všech ultrazvukových sonarů

Oba testovací moduly jsou plně funkční a umožňují odhalení vadné periferie. Testovací modul ultrazvukových sonarů navíc umožňuje snadnou a rychlou výměnu případného vadného ultrazvukového sonaru.

Hlavním přínosem této práce jsou testovací moduly schopné odhalit vadnou periferii. Veškeré cíle této práce jsou splněny. Testovací moduly mohou být používány pro řešení nenadálých situací.

SEZNAM POUŽITÉ LITERATURY

[1]Bender Robotics s.r.o. [online] 20. února 2012 [cit. 2012-4-15]. Dostupné z http://www.advee.eu

[2]Ultrazvukové sonary. *Automa: časopis pro automatizační techniku*. Praha: FCC Public, 2004, roč. 10, č. 05. ISSN 1210-9592. Dostupné z: http://www.odbornecasopisy.cz/index.php?id_document=32330

[3]Výhled trhu s ultrazvukovými hladinoměry a indukčními průtokoměry. *Automa: časopis pro automatizační techniku*. Praha: FCC Public, 2003, roč. 9, č. 05. ISSN 1210-9592. Dostupné z: http://www.odbornecasopisy.cz/index.php?id_document=28809

[4]Robotika - senzory pro roboty ultrazvukové, magnetické aj. SNAIL INSTRUMENTS. [online]. 2012 [cit. 2012-04-15]. Dostupné z: http://www.snailinstruments.com/cze/robotics/sensors.php

[5]DEVANTECH LTD. SRF08 Ultra sonic range finder: Technical Specification [online]. [cit. 2012-04-15]. Dostupné z: http://www.robot-electronics.co.uk/htm/srf08tech.shtml

[6]ROTTA, Jiří. Černá nebo bílá?. *Robot revue: magazín ze světa robotiky*. Praha: RCR, 2009, roč. 1, start, s. 5. ISSN 1804-056x.

[7]HANZAL, Josef. GP2D120 – infračervené dálkoměrné čidlo. *Robot revue: magazín ze světa robotiky*. Praha: RCR, 2009, roč. 1, start, s. 7. ISSN 1804-056x.

[8]SHARP. GP2D120: General Purpose Type Ditance Measuring Sensors [online]. 4 s. [cit. 2012-04-15]. Dostupné z: http://sharp-world.com/products/device/lineup/data/pdf/datasheet/gp2d120_e.pdf

[9]SHARP. GP2Y0A700K0F: Specification [online]. August 26, 2005, 10 s. [cit. 2012-04-15]. Dostupné z: http://www.snailinstruments.com/docs/gp2y0a700.pdf

[10]Sharp GP2D120 IR Range Sensor - 4 to 30 cm. ROBOTSHOP. [online]. 2012 [cit. 2012-04-15]. Dostupné z: http://www.robotshop.com/sharp-gp2d120-ir-range-sensor-4-30-cm.html

[11]KRAJNÍK, Tomáš, SZÜCSOVÁ, Hana, SOVA, Jan, VONÁSEK, Vojtěch. Jak to vidí roboti. *Robot revue: magazín ze světa robotiky*. Praha: RCR, 2010, roč. 2, č. 3, s. 37-39. ISSN 1804-056x.

[12]UHER, O. *3D Laserový dálkoměr vertikálně polohovatelný*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2008. 37 s. Vedoucí bakalářské práce Ing. Tomáš Marada, Ph.D.

[13]Laser measurement technology: LMS2xx / LMS291 / Outdoor / Mid Range. SICK. Sensor Intelligence [online]. 2012 [cit. 2012-04-16]. Dostupné z: https://www.mysick.com/eCat.aspx?go=FinderSearch&Cat=Row&At=Fa&Cult=English&FamilyID=344&Category=Produktfinder&Selections=34284,34258

[14]HANZAL, Josef. GPS. *Robot revue: magazín ze světa robotiky*. Praha: RCR, 2010, roč. 2, č. 3, s. 14-15. ISSN 1804-056x.

[15]DLOUHÝ, Martin. GPS: systém pro globální lokalizaci. In: *Robotika.cz* [online]. 2006-06-2 [cit. 2012-04-17]. Dostupné z: http://robotika.cz/guide/gps/cs

[16]USART. In: *Technical issues and other things: Anything what I find interesting* [online]. 2012 [cit. 2012-04-18]. Dostupné z: http://www.techsite.ic.cz/?page_id=86

[17]ATmega8, ATmega8L: 8-bit with 8KBytes In-System Programmable Flash. ATMEL. [online]. Rev. 2486Z. 2002, 302 s., 02/11 [cit. 2012-04-18]. Dostupné z: http://www.atmel.com/Images/doc2486.pdf

[18]DLOUHÝ, Martin. Komunikace: Předávání informací mezi čipy. In: *Robotika.cz* [online]. 2006-02-28 [cit. 2012-04-18]. Dostupné z: http://robotika.cz/guide/comm/cs

[19]VOJÁČEK, Antonín. Základní informace o RS-485 a RS-422 pro každého. [online]. 14.Červenec 2007 [cit. 2012-05-08]. Dostupné z: http://automatizace.hw.cz/zakladni-informace-o-rs-485-rs-422-pro-kazdeho
[20]TIŠNOVSKÝ, Pavel. Sběrnice RS-422, RS-423 a RS-48. In: *ROOT.CZ*: Informace nejen ze světa Linuxu [online]. 18. 12. 2008 [cit. 2012-04-19]. Dostupné z: http://www.root.cz/clanky/sbernice-rs-422-rs-423-a-rs-485/
[21]STANĚK, Jan a Jan ŘEHÁK. RS 485 & 422. In: *Hw.cz* [online]. 15. Leden 1998 [cit. 2012-04-19]. Dostupné z: http://www.hw.cz/teorie-a-praxe/dokumentace/rs-485-422.html
[22]CADSOFT COMPUTER GMBH. Eagle Online: České stránky editoru plošných spojů EAGLE [online]. 2003, 12.12.2011 [cit. 2012-04-20]. Dostupné z: http://www.eagle.cz/
[23]Atmel AVR Studio 5.1. ATMEL. [online]. 2012, 02/2012 [cit. 2012-04-20]. Dostupné z: http://www.atmel.com/tools/ATMELAVRSTUDIO.aspx
[24]HDSP-K121. GM ELECTRONIC. [online]. 2012 [cit. 2012-04-25]. Dostupné z: http://www.gme.cz/sedmisegmentove-led-displeje-do-020mm/hdsp-k121-p512-079/#dokumentace
[25]DIP 04 PIANO BLUE. GM ELECTRONIC. [online]. 2012 [cit. 2012-04-25]. Dostupné z: http://www.gme.cz/sedmisegmentove-led-displeje-do-020mm/hdsp-k121-p512-079/#dokumentace
[26]BABČANÍK, Jan. Začínáme s mikroprocesory Atmel AVR. In: *Hw.cz* [online]. 1. Prosinec 2006 [cit. 2012-04-25]. Dostupné z: http://www.hw.cz/teorie-a-praxe/zaciname-s-mikroprocesory-atmel-avr.html
[27]ATMEL. Corporate Headquarters [online]. 2012 [cit. 2012-04-25]. Dostupné z: http://www.atmel.com/about/contact/default.aspx?contactType=Online%20Directory
[28]FLEURY, Petr. Petr Fleury Online: AVR Software [online]. 2006, 31.10.2010 [cit. 2012-04-26]. Dostupné z: http://homepage.hispeed.ch/peterfleury/avr-software.html
[29]SN75176A: Differential Bus Transceiver. TEXAS INSTRUMENTS. [online]. Dallas, Texas, 1984, May 1995 [cit. 2012-05-02]. Dostupné z: http://www.gme.cz/dokumentace/959/959-001/dsh.959-001.1.pdf
[30]ATmega48/V, ATmega88/V, ATmega168/V: 8-bit Atmel Microcontroller with 4/8/16K Bytes In-System Programmable Flash. ATMEL. [online]. Rev. 2545T-04/11. 2004, 04/11 [cit. 2012-05-02]. Dostupné z: http://www.atmel.com/Images/doc2545.pdf
[31]JUNGHANS, Martin. Library - I2C/TWI Slave. In: *Junghans: Elektronik Projekte* [online]. 2004, 19.2.2010 [cit. 2012-05-04]. Dostupné z: http://www.jtronics.de/avr-projekte/library-i2c-twi-slave.html
[32]TWI Multi Slave. In: *LTWA. Beijingcode* [online]. October 15, 2004 [cit. 2012-05-04]. Dostupné z: http://beijingcode.org/projects/quadcopter/browser/i2c/slave/multi_slave?rev=03572d826df20800c9e20ea7a288b2ee1ea34ff6
[33]HRBÁČEK, J. *Návrh a realizace regulátoru otáček stejnosměrného motoru pro mobilní robot.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2008. 49 s. Vedoucí bakalářské práce Ing. Jiří Krejsa, Ph.D.
[34]Multiple I2C slave address issue. In: *AVR Freaks: The Nr.1 AVR Comunity* [online]. Aug 20, 2008 [cit. 2012-05-04]. Dostupné z: http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&p=480030

SEZNAM PŘÍLOH

Příloha 1: Schéma testovacího modulu ultrazvukových sonarů SRF08

Příloha 2: Plošný spoj testovacího modulu ultrazvukových sonarů SRF08, strana Top v měřítku 2:1

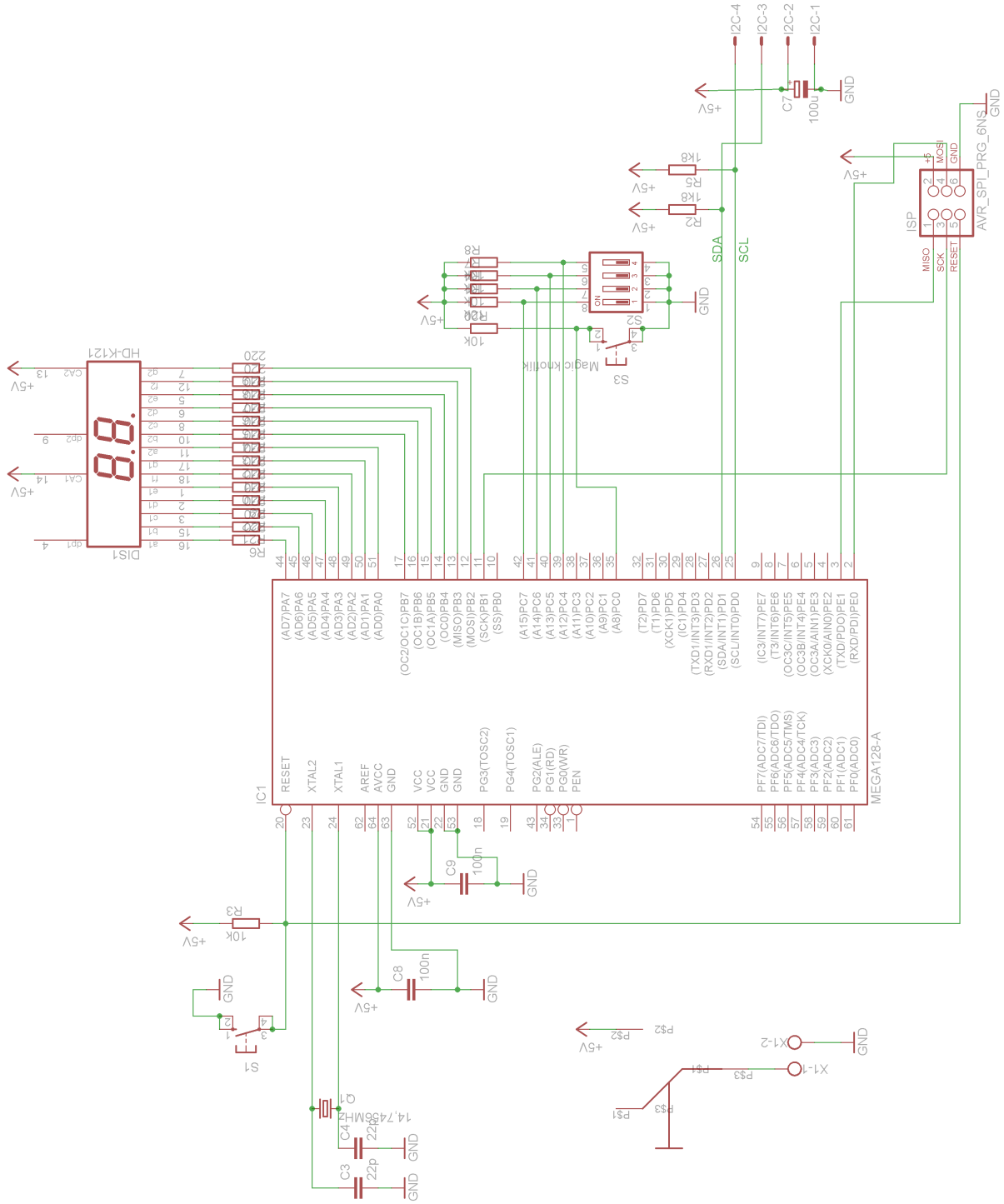
Příloha 3: Plošný spoj testovacího modulu ultrazvukových sonarů SRF08, strana Bottom v měřítku 2:1

Příloha 4: Schéma testovacího modulu jednotek na zpracování dat z ultrazvukových sonarů

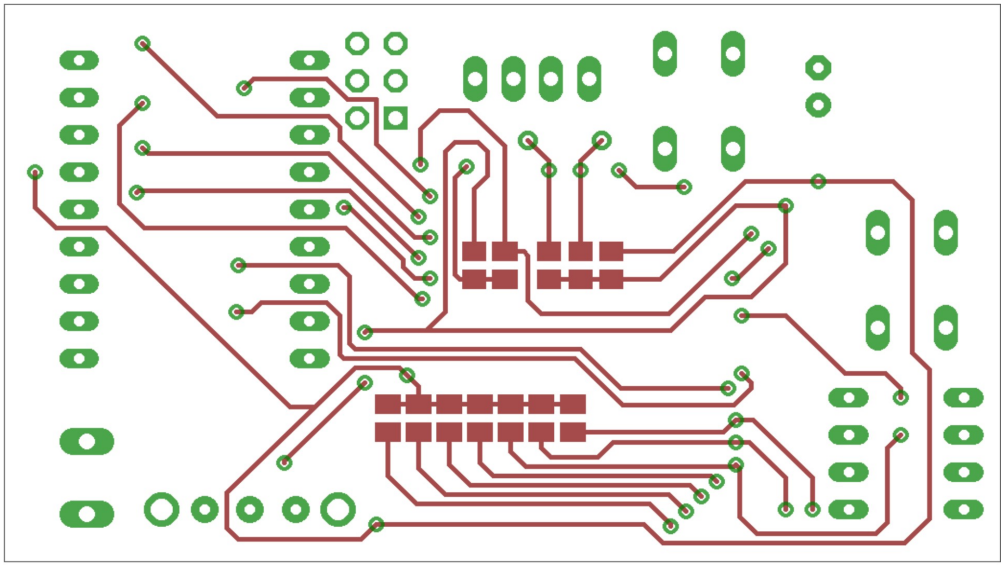
Příloha 5: Plošný spoj testovacího modulu jednotek na zpracování dat z ultrazvukových sonarů, strana Top v měřítku 2:1

Příloha 6: Plošný spoj testovacího modulu jednotek na zpracování dat z ultrazvukových sonarů, strana Top v měřítku 2:1

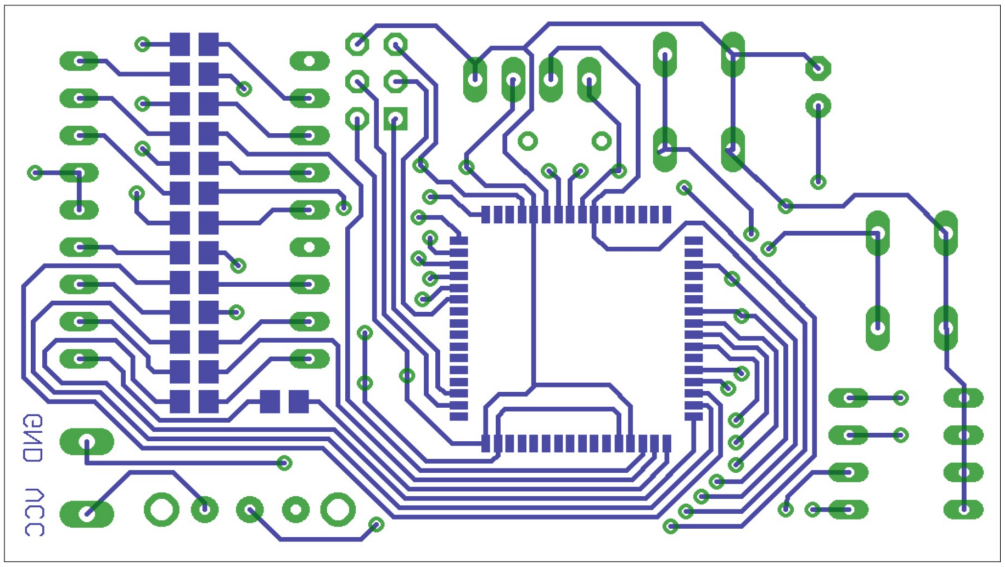
Příloha 1: *Schéma testovacího modulu ultrazvukových sonarů SRF08*



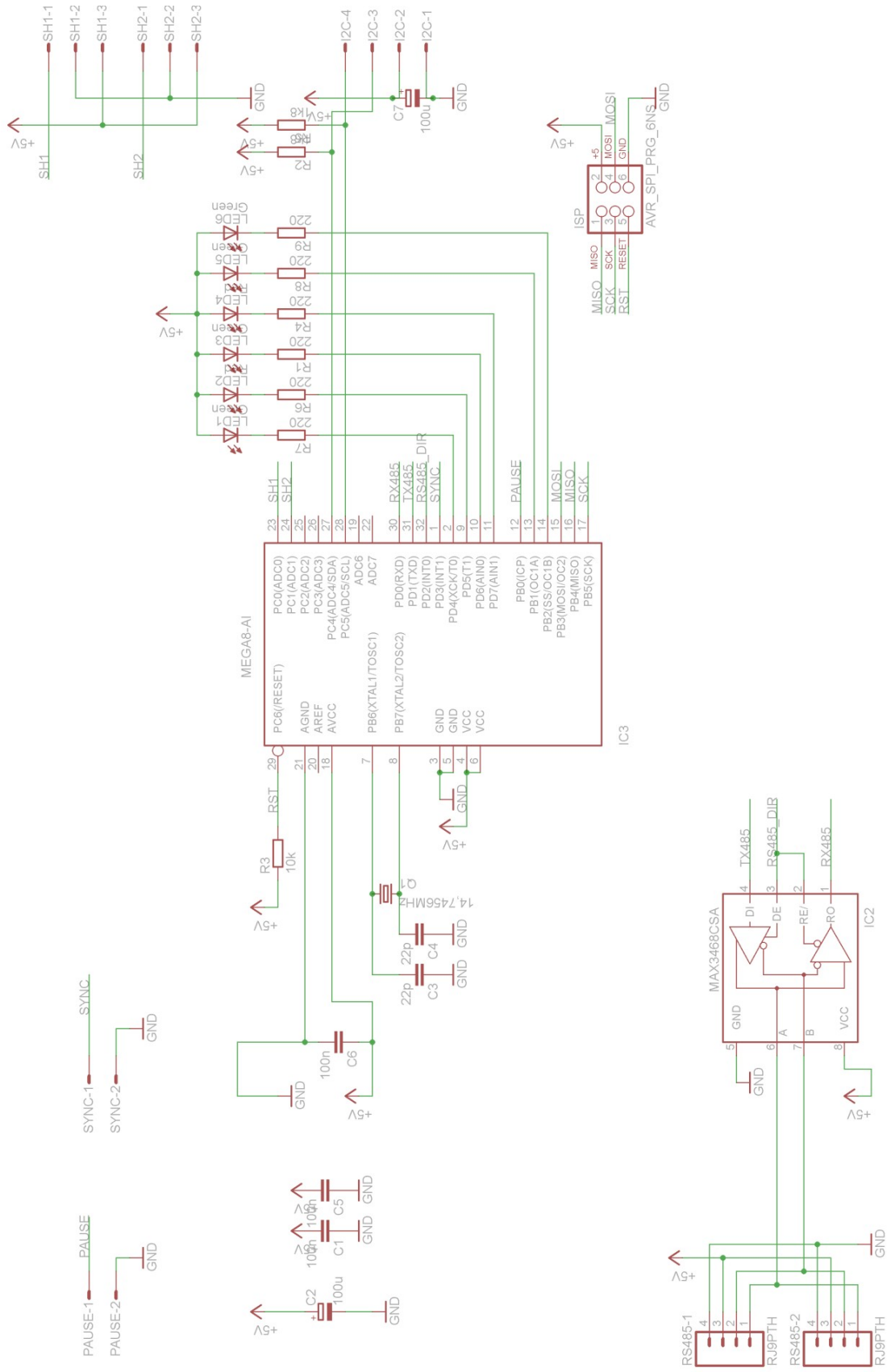
Příloha 2: *Plošný spoj testovacího modulu ultrazvukových sonarů SRF08, strana Top v měřítku 3:1*



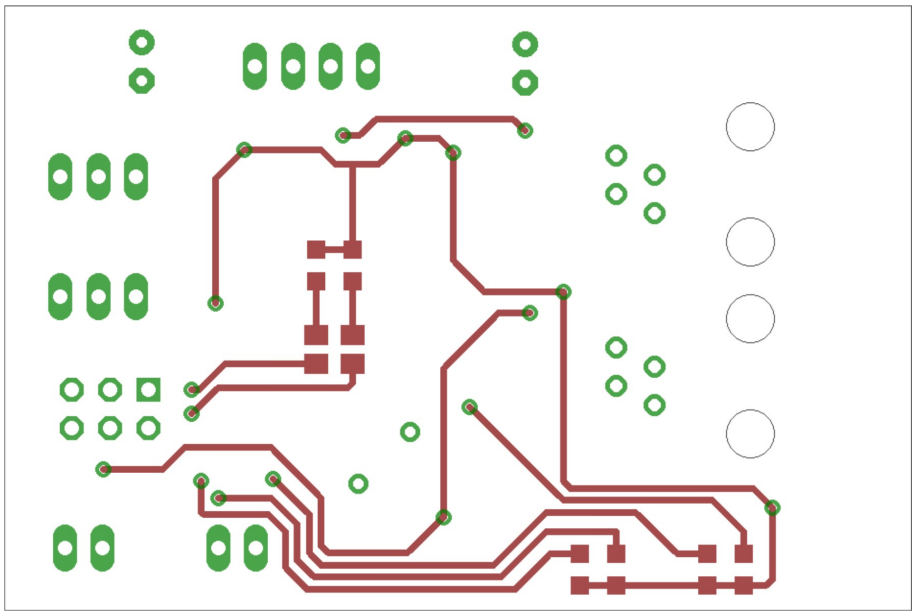
Příloha 3: *Plošný spoj testovacího modulu ultrazvukových sonarů SRF08, strana Bottom v měřítku 2:1*



Příloha 4: *Schéma testovacího modulu jednotek na zpracování dat z ultrazvukových sonarů*



Příloha 5: *Plošný spoj testovacího modulu jednotek na zpracování dat z ultrazvukových sonarů, strana Top v měřítku 2:1*



Příloha 6: *Plošný spoj testovacího modulu jednotek na zpracování dat z ultrazvukových sonarů, strana Bottom v měřítku 2:1*