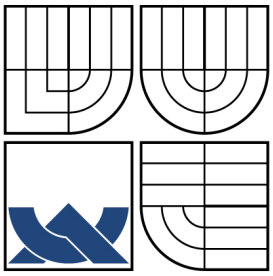
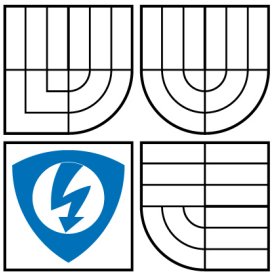


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ŘÍZENÍ HUDEBNÍ ELEKTRONIKY POMOCÍ SNÍMAČŮ NEELEKTRICKÝCH VELIČIN S VYUŽITÍM PROTOKOLU MIDI

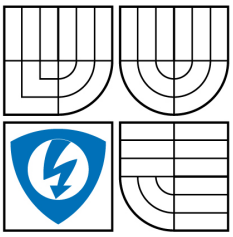
CONTROL OF MUSIC EQUIPMENT WITH NON-ELECTRIC SENSORS USING MIDI PROTOCOL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. MILOŠ VLACH
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ SCHIMMEL, Ph.D.
SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Miloš Vlach ***ID:*** 89262
Ročník: 2 ***Akademický rok:*** 2008/2009

NÁZEV TÉMATU:

Řízení hudební elektroniky pomocí snímačů neelektrických veličin s využitím protokolu MIDI

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a realizujte zařízení pro řízení hudební elektroniky pomocí snímačů neelektrických veličin s využitím protokolu MIDI. Zařízení bude umožňovat připojení několika snímačů více než jednoho typu, např. snímače tlaku a vzdálenosti. Uživatel bude moci nastavovat přiřazení čísel MIDI kontrolerů konkrétním snímačům pomocí zpráv typu System Exclusive protokolu MIDI ve standardním formátu. Pro realizaci využijte vývojový modul Arduino Diecimila s mikroprocesorem Atmel ATmega168.

DOPORUČENÁ LITERATURA:

[1] FORRÓ, D. Svět MIDI. Praha : GRADA publishing, 1997. ISBN 80-7169-415-6.
[2] BANZI,M. Getting started with Arduino [online]. Dostupný z WWW: <http://www.arduino.cc/en/uploads/Booklet/Arduino_booklet02.pdf>.
[3] Atmel Corporation : ATmega48/88/168 Datasheet [online]. c2007. Last updated 09/2007. Dostupný z WWW: <http://www.atmel.com/dyn/resources/prod_documents/doc2545.pdf>.

Termín zadání: 9.2.2009 ***Termín odevzdání:*** 26.5.2009

Vedoucí práce: Ing. Jiří Schimmel, Ph.D.

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práve třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Abstrakt

Cílem této diplomové práce je navrhnout a sestrojit zařízení pro ovládání hudební elektroniky pomocí protokolu MIDI, kdy bude využito snímačů ne-elektrických veličin jakožto netradičních ovládacích prvků a modulu Arduino Diecimila, jako prvku řídícího. Tato práce se komplexně zabývá daným problémem od návrhu, až po samotnou realizaci. V průběhu práce je navrženo řešení pro dva různé typy snímačů, optický snímač vzdálenosti a odporový snímač váhy. Pro tyto snímače jsou navrženy přizpůsobovací obvody, včetně rozhraní, které jsou dále realizovány a osazeny na desce plošných spojů. Dále je vytvořen obslužný program, který signál ze snímačů převádí na datový tok MIDI zpráv, které odesílá přes sériové USB rozhraní do osobního počítače, jakožto ovládaného zařízení. Chod programu a vlastnosti jednotlivých MIDI prvků je možné řídit pomocí zpráv pod-protokolu realizovaných za využití zpráv MIDI SysEx.

Klíčová slova: MIDI, Arduino, snímače, SysEx, ovládání hudby, pěnový snímač síly, oprický snímač vzdálenosti.

Abstract

The goal of this Master's thesis is to create a device that is able to control MIDI musical electronics, via use of non-electrical sensors as un-usual control elements and module Arduino Diecimila used as the processing element. This thesis is dealing with this given problem from a complex point of view, from design to actual realisation of the device. In the process of the thesis the solution for an optical proximity sensor and for resistance weight sensor is delivered. For both of them the adjustments circuitry is designed including interfaces, and further on it is constructed on PCB board. Then, control software, which processes the input sensor signal into a stream of MIDI data, which is sent to the computer as a slave device, is developed. The runtime of the program and the behaviour of MIDI elements can be controlled with a sub-protocol based on MIDI SysEx messages.

Keywords: MIDI, Arduino, sensory, SysEx, music control, foam force sensor, optical proximity sensor.

Bibliografická citace

VLACH, M. *Řízení hudební elektroniky pomocí snímačů neelektrických veličin s využitím protokolu MIDI*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 63 stran, 9 příloh . Vedoucí diplomové práce Ing. Jiří Schimmel, Ph.D.

Poděkování

V úvodu práce bych rád poděkoval panu Ing. Jiřímu Schimmelovi PhD. za jeho odborné vedení průběhu práce. Dále bych rád poděkoval panu Ing. Petrovi Lungovi ze společnosti Ploma s.r.o. za materiální podporu v rámci této diplomové práce.

V Brně dne 26.5.2009 .....

Prohlášení o pùvodnosti práce v tomto znění:

Prohlašuji, že jsem svou diplomovou práci na téma „Řízení hudební elektroniky pomocí snímačů neelektrických veličin s využitím protokolu MIDI“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce s použitím odborné literatury a dalších informaèních zdrojù, které jsou všechny uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušilautorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení §11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 26.5.2009 .....

OBSAH

Úvod 7

Cíle této práce 8

Dělení práce 9

1. Teoretický rozbor 10

1.1. Snímání vlastností a akcí osob 11

1.1.1. Snímání vzdálenosti od hranic prostoru nebo objektů 11

1.1.2. Snímání aktivity osob 11

1.1.3. Snímání pohybu, koncentrace a rozmístění v prostoru 12

1.1.4. Snímače neelektrických veličin 12

1.1.5. Snímače z hlediska měřící veličiny 13

1.1.6. Snímače z hlediska měřené veličiny 14

1.2. Řídící jednotka 17

1.2.1. Moduly Arduino 17

1.2.2. Modul Arduino Diecimila 19

1.3. Řízení digitální hudební elektroniky pomocí MIDI 21

1.3.1. Struktura protokolu MIDI 23

1.3.2. Kanálové MIDI zprávy 24

1.3.3. Běžné systémové zprávy 27

1.3.4. Systémové zprávy v reálném čase 30

2. Realizace řešení 32

2.1. Snímání pohybů vystupujícího 32

2.1.1. Snímání vzdálenosti vystupujícího 33

2.1.2. Snímání Váhy vystupujícího 37

2.2. Připojení snímačů k modulu Arduino 45

2.2.1. Rozhraní přizpůsobovacího modulu 47

2.2.2. Propojení modulu Arduino s osobním počítačem 49

2.2.3. Použití sériového rozhraní procesoru ATmega168 50

2.3. Obslužný program mikroprocesoru 53

2.3.1. Čtení hodnot ze snímačů a vysílání MIDI zpráv 55

2.3.2. Příjem SysEx zpráv 59

2.3.3. Protokol pro přijímání SysEx zpráv 59

3. Závěr 62

4. Použitá literatura 64

5. Seznam příloh 65

ÚVOD

V dnešní době výkonných procesorů se digitální zpracování signálů prosazuje do všech oborů a činností člověka. Jedním z oborů, kde se tento trend začal projevovat jako jeden z prvních, je hudební technika, kde dal možnost vzniknout mnoha novým hudebním stylům, které ve výsledku kompletně změnili kulturu a chování moderní civilizace.

Jednou z věcí, které tento trend odstartovaly, je právě protokol MIDI (Musical Instrument Digital Interface, digitální rozhraní hudebních nástrojů), který se stal jedním z hlavních důvodů masivního a rychlého rozšíření digitálního zpracování zvuku. Svým vznikem umožnil snadné ovládání digitálních nástrojů, jako jsou například syntetizátory a samplery, pomocí jednoduchého a již zažitého ovládacího rozhraní – klávesových ovladačů. Nicméně využití tohoto protokolu nezůstalo pouze u hudebního umění, ale za necelého čtvrt století se tento protokol rozšířil i do jiných oblastí, jako jsou „vizuální umění“ a divadlo, a svojí jednoduchostí a univerzálností poskytuje obrovskou podporu při živých vystoupeních.

Postupem času se ovládací zařízení z původních klávesových ovladačů rozrostla o rodinu zařízení založených tlačítcích a otočných a tažných potenciometrech. Zde ale vývoj MIDI kontrolérů téměř ustal. Jedinou komerční výjimkou je společnost Korg, která již řadu let experimentuje s přítlačnými kontroléry. Jiné výjimky jsou vesměs experimentální.

Dalším faktem je, že vývoj MIDI kontrolérů stále směřuje k jednotlivým nástrojům, ve smyslu že kontrolér je vždy jeden „objekt“ obsluhován jedním člověkem. Oproti tomu, pokud se podíváme na trendy, které panují v novém tisíciletí v takzvaných nových médiích, dají se shrnout slovy: kolaborace, zapojení a spolutvorba. Tento trend se již projevuje v různých uměleckých oborech, kdy například v sochařství, spojeném s moderním designem a automatizací, vznikají interaktivní, automatizované sochy a instalace, které interagují s návštěvníky a jejich vzhled a tvar je poté dán právě těmito interakcemi. A přesně toto je cesta, která, sic zatím neprozkoumaná, může nabídnout obrovské možnosti a perspektivy v oblasti hudebních a audiovizuálních vystoupení.

Cíle této práce

Cílem této diplomové práce je navrhnout a sestrojit zařízení pro ovládání digitálních hudebních zařízení, které bude založeno na alternativních ovládacích prvcích. Alternativními ovládacími prvky jsou myšleny jiné prvky, než byly popsány výše, a které jsou navrženy pro jiné typy interakcí než pouze pomocí prstů, jak to umožňují běžně dostupné ovládací prvky jako klávesnice, myš nebo běžně dostupné ovladače využívající otočných nebo posuvných potenciometrů. Tyto ovládací prvky by měly být použitelné jak při ovládání jednou osobou, tak při ovládání více osobami. Zároveň je snaha, aby toto zařízení bylo co nejflexibilnější a umožňovalo použití v různých aplikacích a snadné rozšíření do budoucna.

Společnou vlastností běžných ovládacích zařízení je, že jsou většinou ovládány pouze jednou osobou, která je navíc odsunuta mimo hlavní dění na scéně, díky “nízké atraktivitě” svojí činnosti. Toto ve výsledku staví pomyslnou zeď mezi ovládaným zařízením, jeho hudebním či vizuálním výstupem a osobami vystupujícími na scéně a klade zvýšené nároky na koordinaci mezi vystupujícími a osobou ovládající zařízení.

Tato práce má tedy poskytnout technologický základ pro technologii, která bude tuto pomyslnou zeď bořit a poskytne možnost samotným vystupujícím ovládat hudební zařízení, a to bez újmy na choreografii vystoupení. “Ovládající akce” musí být včlenitelné do výsledné choreografie, nebo nesmí být rozeznatelné, že nejsou součástí choreografie a tím přenechat řízení kompletně v režii vystupujícího. Což poskytne vystoupení více flexibility a nechá vystupujícímu více prostoru pro improvizaci.

Všechny tyto předpoklady kladou podstatné nároky na způsoby interakce mezi vystupujícími a ovládacím zařízením. Všechny interakce by měly být založené na co nejpřirozenějších lidských pohybech, akcích a vlastnostech, které nebudou kolidovat s choreografií vystoupení, ba na opak ji podporovat a umocňovat.

Dalším krokem jak rozšířit perspektivu tohoto projektu, je snaha začlenit do procesu ovládání více osob. Jednak toto mohou být osoby na jevišti a jednak to mohou být osoby v publiku, čímž bude umožněno publiku umožněno se částečně, nebo úplně podíle na vystoupení, kterého se účastní. V tuto chvíli je vhodné připomenout, že cílem projektu je vytvořit technologický základ pro takovéto vystoupení, nikoliv koncept vystoupení samotného.

Dělení práce

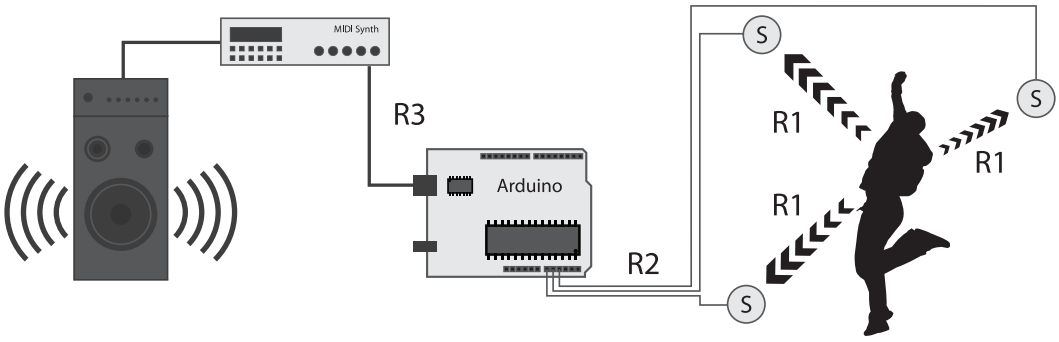
První část práce je teoretická a zabývá se rozborem současných trendů a technologií, které budou nebo by mohly být využity při realizaci projektu. Jejím cílem je seznámit čtenáře s rozebíranými technologiemi v dostatečném rozsahu pro pochopení detailů a souvislostí popsaných v části realizace projektu. První podkapitola této části rozebírá možné typy senzorů neelektrických veličin, které lze použít pro detekci interakcí, jejich vlastnosti, výhody a nevýhody. Druhá podkapitola poskytuje vhled do technologií používaných v oboru digitálních hudebních nástrojů, hlavně pak protokolu MIDI. Třetí podkapitola je věnována modulu Arduino, který bude sloužit pro zpracování signálů ze snímačů a jejich následný přenos do ovládaného zařízení.

Druhá část práce je popis samotné realizace projektu. V této části je rozebráno jedno konkrétní řešení a je popsáno včetně všech důležitých detailů. První podkapitola se věnuje výběru vhodných typů snímačů, druhá jejich přizpůsobení a připojení k modulu Arduino, třetí řeší připojení modulu k řízeným zařízením a čtvrtá se zabývá obslužným programem mikroprocesoru, který čte data ze snímačů a převádí je na MIDI zprávy, které poté vysílá do ovládaného zařízení.

V závěru jsou rozebrány výsledky práce a problémy, které při práci vyvstaly , a zároveň je nastíněno jejich řešení.

1. TEORETICKÝ ROZBOR

Systém, který se snažíme vytvořit, je vhodné nejprve rozebrat, rozdělit ho na jednotlivé stupně a popsat rozhraní mezi nimi. Toto ilustruje obrázek 1. Prvním rozhraním je interakce mezi člověkem a snímačem. Snímač převede určitou vlastnost nebo charakteristiku interagující osoby na některou elektrickou veličinu. Tato je přes rozhraní R2 postoupena dalšímu stupni, kterým je mikrokontrolér, který tuto elektrickou veličinu převádí na datový tok zpráv. Tento datový tok je přes rozhraní R3 přenesen do ovládaného zařízení, kde je interpretován a na jeho základě spuštěn odpovídající vzorek hudby. Pro zachování koherence posupuje text této práce stejně, od vzniku interakcí mezi člověkem a senzorem, k jeho přenesení do podřízeného zařízení.



Obr 1. Znázornění jednotlivých částí projektu

1.1. Snímání vlastností a akcí osob

Tato kapitola se snaží rozvést diskuzi nad veličinami, které je možné v rámci projektu snímat, nad jejich relevancí, typy snímačů kterými je lze měřit, jejich charakteristikami a principy měření. Pokud uvažujeme nad umístěním projektu, lze předpokládat, že se bude jednat o omezený prostor, v němž se budou snímané osoby pohybovat pouze ve dvou rozměrech. Co lze tedy snímat?

» sílu, kterou působí na podlahu prostoru nebo objekty v prostoru
» vzdálenost od hranic prostoru nebo objektů umístěných v prostoru,
» pohyb po osách prostoru
» rozmístění a koncentrace osob v prostoru
» jejich aktivitu

Je zřejmé, že některé z vyčtených veličin vyžadují složitější počítačové zpracování a proto z tohoto důvodu budou z projektu vynechány. Nicméně bude zajímavé alespoň zde rozebrat možnosti, jaké jsou k dispozici pro jejich snímání.

Pravděpodobně jediné možné řešení tohoto snímání je pomocí snímačů sil umístěných v podlaze nebo v daných objektech. Hlavním problémem při realizaci tohoto případu je volba vhodných snímačů. To ovšem z velké míry záleží na konkrétní aplikaci.

1.1.1. Snímání vzdálenosti od hranic prostoru nebo objektů

Pro řešení toho případu by se dalo využít jak vhodných snímačů vzdálenosti, tak i zpracování obrazu, osobně se ale domývám, že řešení pomocí snímačů je efektivnější. Nicméně záleží na požadavcích aplikace a hlavně na počtu potřebných snímačů.

1.1.2. Snímání aktivity osob

Jak vlastně změřit aktivitu osob? Pravděpodobně neexistuje žádná přímá metoda měření aktivity, dá se ale najít způsob měření sekundárních projevů. Například pokud uvažujeme umístění takovéhoto systému do uzavřeného prostoru, lze aktivitu měřit přeneseně pomocí měření teploty v prostoru. Pokud stoupá aktivita, lidská

těla produkují více tepla a v jistých případech mohou být rozdíly teplot až v řádu desítek stupňů. Jiným možným přístupem by mohlo být měření aktivity pomocí vibrací v prostoru. Toto by se dalo navíc rozšířit o snímání pomocí sítě senzorů, kdy by byla měřena aktivita v jednotlivých částech prostoru.

1.1.3. Snímání pohybu, koncentrace a rozmístění v prostoru

Pro snímání pohybu osob v prostoru se nabízejí dvě principielně odlišné metody – pomocí snímačů umístěných v podlaze, nebo pomocí zpracování obrazu. Při použití snímačů v podlaze se dá toto měření spojit s měřením váhy a využít stejné technologie, při použití tohoto řešení by dále muselo být rozšířeno zpracování dat o programové řešení určování pozice, pravděpodobně pomocí matice snímačů a jejího vyhodnocování.

Při použití zpracování obrazu, by se jednalo o čistě softwarové řešení, které by nevyhovovalo zadání práce. Nicméně je zajímavé, že většinu rozebíraných konceptů by bylo možné nahradit měřením s využitím zpracování obrazu. Velmi zajímavým by bylo například řešení pomocí infračervené kamery, které by umožnilo měřit jak pohyb, koncentraci a rozmístění v prostoru, tak i aktivitu v jednotlivých částech.

1.1.4. Snímače neelektrických veličin

Snímač neboli senzor (z lat. sensus) je, v tomto kontextu, označení prvku sloužícímu k převodu neelektrické veličiny na veličinu elektrickou (obecně je to převod měřené veličiny na měronosnou). Měřená veličina není na měronosnou převedena rovnou, ale pomocí třetí, pomocné, dalo by se říci měřící, veličiny. V případě převodu neelektrické veličiny na veličinu elektrickou, je měronosnou veličinou většinou napětí nebo proud (muže být i například frekvence, ale ta jako taková je již vlastností napětí nebo proudu). Měřená, měřící i měronosná veličina hrají zásadní roli při určování typu snímače.

Jako měřící veličiny se většinou využívají vlastnosti prvků zapojených v elektrickém obvodu, které jsou proměnné v závislosti na aspektech, kterými na ně neelektrická veličina působí. Tímto jsou posléze ovlivněny elektrické parametry obvodu, do něhož je prvek zapojen. Při tomto je využito vlastností materiálů, z nichž jsou obvodo-

vé prvky vyrobeny, a jevů, jež se k těmto materiálům váží.

1.1.5. Snímače z hlediska měřící veličiny

i ***Poznámka:***
Cílem tohoto oddílu je projít a zrekapitulovat typy snímačů z hlediska měřící veličiny, neboť k těmto pojmům bude referováno dále v textu.

Odporové snímače
Odporové snímače jsou založeny na změně odporu prvku v elektrickém obvodu. Tato změna může být buď skoková, nebo plynulá. Skokové změny se využívá kontaktních snímačů a většinou dochází ke skoku z hodnoty blížící se nule na hodnotu blížící se nekonečnu (například vypínač). U snímačů s postupnou změnou odporu se využívá fyzikálních vlastností a vlastností materiálu, z něhož je prvek vyroben. Při změně těchto vlastností dochází ke změně odporu.

Indukční snímače
To této skupiny spadají snímače využívající jak změny indukčnosti cívky, tak i změny výstupního napětí v závislosti na snímané veličině. Pro to se většinou využívá zasouvání jádra do cívky nebo přibližování a oddalování dvou cívek a využití transformátorového efektu.

Kapacitní snímače
Tyto snímače převádějí měřenou veličinu na měronosnou pomocí změny kapacity prvku (kapacitoru). Změny kapacity se dá dosáhnout: změnou vzdálenosti mezi deskami, změnou plochy desek, nebo změnou vlastností dielektrika. U prvního způsobu měřená veličina působí na jednu, nebo obě desky čímž se mění jejich vzdálenost, ve druhém případě působí posunutí desek. Ve třetím je zasouváno mezi desky jiné dielektrikum, nebo dochází ke změně jeho permitivity například stlačením, zvlhnutím nebo ohřevem.

Termoelektrické snímače
Termoelektrické snímače využívají Seebeckova jevu. Ten v praxi říká, že na styku dvou kovů s různou teplotou může vzniknout rozdíl potenciálů. Proto mají tyto snímače dva „konce“, jeden je měřící („teplý“) a druhý srovnávací („studený“).

Piezoelektrické snímače

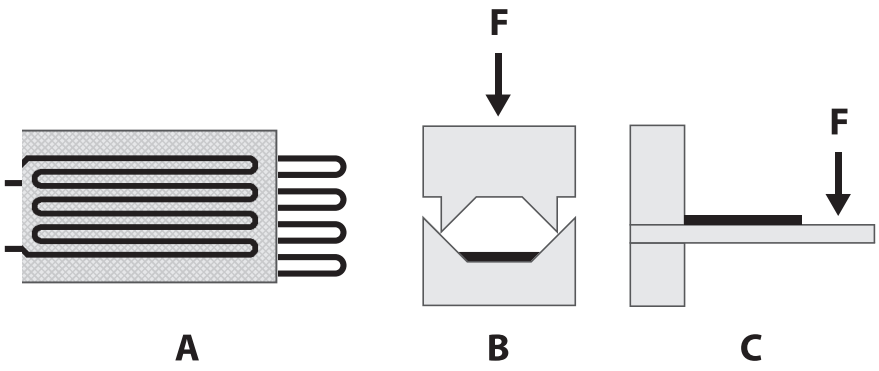
Využívají piezoelektrického jevu, kdy při mechanické deformaci dochází v materiálu k vnitřní polarizaci, čímž na povrchu vznikají zdánlivé elektrické náboje, které následně vážou, nebo uvolňují, skutečné náboje v přiložených elektrodách.

1.1.6. Snímače z hlediska měřené veličiny

Snímače síly

Při měření síly je třeba zhodnotit tři klíčové aspekty: rozsah sil a přesnost snímače/nejmenší měřitelná jednotka a jeho velikost. V tomto projektu jsou snímače použity pro snímání váhy osob, proto je potřebný rozsah snímače přibližně 0-100 Kg a pro měření bude postačující přesnost minimálně cca. 500g.

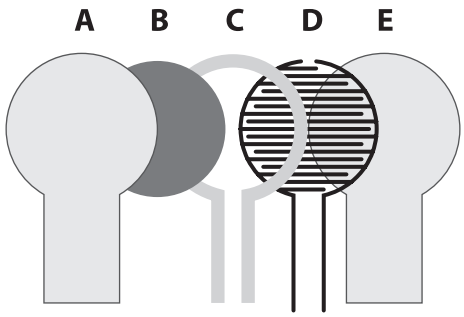
Pro měření váhy se v největší míře využívá tenzometrických snímačů. Tento typ snímačů se dále dělí na kovové a polovodičové. U kovových tenzometrů se využívá efektu změny odporu při změně délky vodiče způsobené vnější silou - tahem. Oproti tomu u polovodičových se využívá efektu piezorezistence, kdy se při stlačení mění koncentrace nosičů v polovodiči, a tudíž se mění jeho odpor. Tenzometrické snímače se vyznačují velkou přesností a rozsahem. Princip funkce tenzometru je znázorněn na obrázku.



Obr 2. Tenzometrický snímač

A - tenzometr; B,C - dva příklady konstrukce snímače

Jiným řešením jsou páskové snímače typu FSR (Force Sensing Resistor). Tyto snímače jsou také založeny na piezorezistentní vlastnosti polovodičů, mají ovšem odlišnou konstrukci než snímače tenzometrické. Tyto snímače jsou navíc dostupné jako elementární prvky, čímž značně klesá jejich pořizovací cena. Konstrukce snímače je znázorněna na obrázku 3 .



Obr 3. FSR Snímač

A,E - flexibilní krycí vrstva; B - piezoresistentní polovodič; C - izolant; D - elektrody

V praxi je B natištěno na A a D na E.

Jak je vidět na obrázku, FSR snímač se skládá ze tří vrstev. První vrstvou je dvojice elektrod vzájemně proložených, druhou je oddělující izolant, který zaručuje nekonečný odpor v klidovém stavu, a třetí vrstvou je polovodič který při stlačení mění svůj odpor. Z tohoto je patrné, že FSR rezistory „obráceně“ oproti tenzometrickým, kdy FSR polovodič stlačují a snižují tím jeho odpor, kdežto tenzometry polovodič natahují.

Sílu lze měřit i pomocí změny kapacity a indukce. V případě snímače založeného na změně kapacity se mění tloušťka dielektrika mezi elektrodami a tím je ovlivněna jeho kapacita. V případě indukčního snímače lze jak měnit vzdálenost mezi cívkami i lze zasouvat jádro dovnitř cívky. Nicméně tyto metody nejsou příliš využívané.

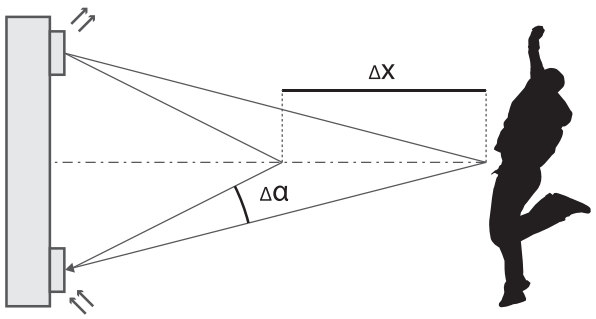
Snímače vzdálenosti

Při snímání vzdálenosti opět záleží na obdobných aspektech jako při snímání síly: rozsah vzdálenosti a přesnost. Snímače vzdálenosti se dají dělit na dotykové a bezdotykové, vzhledem k povaze projektu jsou dále rozebrány pouze snímače bezdo-

tykové. Pro měření malých vzdáleností lze použít snímače indukční nebo laserové, nicméně cílem projektu je měřit vzdálenost osob, je tedy postačující rozlišení v řádu centimetrů až desítek centimetrů, s dosahem jednotek až desítek metrů záleží na konkrétní aplikaci. Pro takovéto rozsahy se běžně využívají snímače ultrazvukové, nebo optické.

První metodou je měření času mezi vysláním a příjmem vlny. Pro odlišení od pozadí je většinou vysílán určitý kód. Tato metoda je běžnější u měření pomocí ultrazvukových vln, nicméně existují i optické či laserové varianty.

Druhou metodou je metoda triangulační. Při této metodě je počítán úhel dopadu paprsku do přijímače. Princip této metody je ilustrován na obrázku 4. Tato metoda je běžná u optických nebo laserových snímačů.



Obr 4. Princip triangulační metody

Δx - změna vzdálenosti objektu; Δα - změna úhlu, pod kterým dopadá odražený paprsek

Snímače teploty

Pro snímání teploty se v praxi používají nejvíce dvě technologie: měření pomocí termočlánku nebo měření pomocí odporového snímače. Měření pomocí termočlánku je popsáno výše v odstavci termoelektrických snímačů.

Odporové snímače využívají jevu, kdy některé kovové nebo polovodičové materiály mění svůj odpor v závislosti na teplotě. V případě kovových snímačů se nejčastěji se používá platinových, niklových nebo měděných vynutí. Z oblasti polovodičů se nejvíce využívá termistorů nebo křemíkových čidel. Všechny typy snímačů jsou běžně

dostupné v integrované podobě v různých variantách rozsahů, výstupních signálů a rozlišení.

1.2. Řídící jednotka

V minulé kapitole byly rozebrány různé typy snímačů, jakožto prvního převodního stupně systému, který převádí lidské interakce na elektrický signál. Tato kapitola je zaměřena na technologie druhého stupně, kdy je zapotřebí převést elektrické signály na digitální datový tok směřující k ovládanému zařízení. Tímto převodovým stupněm je mikrokontrolér, který snímá analogové signály na svých vstupech a převádí je na zprávy, které vysílá po sériovém rozhraní do další části systému.

Rozebírat rodiny mikrokontrolérů jednu po druhé by bylo vzhledem k jejich počtu poněkud nesmyslné a navíc je použitá technologie jasně specifikována v zadání. Z tohoto důvodu je tato kapitola koncipována jako přehled vlastností, které poskytují mikroprocesor ATmega168 platforma Arduino.

1.2.1. Moduly Arduino

Arduino je volně dostupná platforma pro elektronické modelování založená na flexibilním a jednoduše použitelném hardwaru a softwaru. Tato platforma je primárně určena pro použití při vývoji a tvorbě interaktivních objektů a prostředí.

Důvodem proč je zde Arduino referováno jako platforma je že se nejedná pouze o modul s mikroprocesorem, ale obsahuje i vlastní vývojové prostředí, které využívá vlastní, vyšší abstrakci jazyka C++ a integruje určité funkce, čímž velmi ulehčuje práci programátora. Hlubší informace, než obsahuje tato kapitola, jsou k nalezení v Banzi[3]

Moduly Arduino jsou levné, robustní vstupně výstupní moduly, založené na mikroprocesorech od společnosti Atmel. Jsou dostupné v poměrně mnoha variacích lišících se v použitém mikroprocesoru a v integrovaných periféríích. Většina modulů má určitým způsobem již integrované kompletní sériové rozhraní, čímž velmi ulehčují programování a jsou programovatelné bez použití dalších programovacích zařízení. Rozlišují se verze s rozhraním RS232, USB, BlueTooth, ale i jinými. Nicméně

vzhledem k tomu že jsou všechna postaveny na mikroprocesorech Atmel, všechny typy jsou programovatelné přes ICSP(In-Circuit Seriál Programming).

Arduino NG, Diecimila a Duemilanove

Verze modulu s USB rozhraním založené na mikroprocesoru ATmega168, podporuje automatické restartování modulu při nahrávání zdrojového kódu a má nadproudovou ochranu pro USB rozhraní. Jednotlivé ze tří uvedených jsou generační následovníci a liší se v detailech

Arduino Mini

Obdoba předchozích uvedených verzí modulů, co se týká vybavení, ale je navržený pro aplikace, při nichž je klíčová úspora místa. Z tohoto důvodu nemá modul integrováno jiné sériové rozhraní než UART, který poskytuje ATmega168. Nicméně je dostupná i verze s USB rozhraním.

Arduino LilyPad

Modul navržený pro aplikace na textiliích, je proto nejmenší z řady modulů Arduino a může být na textilii jednoduše přišit (v průměru měří 50mm). Opět nemá integrováno sériové rozhraní.

Arduino BT

Tato verze odstraňuje potřebu kabelů připojených ke kontroléru, neboť je vybavena rozhraním BlueTooth a modul může být programován a komunikovat s počítačem nebo jiným zařízením BlueTooth do vzdálenosti 10 m v interiérech a 100 m ve volném prostranství.

Arduino Nano

Další z kompaktních modulů, odobné provedení jako Arduino Mini má, ale navíc USB konektor.

Arduino Mega

Nejnovější přírůstek do rodiny modulů Arduino. Tento modul je oproti ostatním o poznání robustnější. Je založen na procesoru ATmega1280, celkem nabízí 54 digitálních a 16 analogových pinů, 128 KB FLASH a 8 KB SRAM paměti.

1.2.2. Modul Arduino Diecimila

Diecimaila znamená v Italštině 10000 a tento typ byl takto pojmenován díky skutečnosti, že již bylo vyrobeno přes 10000 modulů. Arduino Diecimila je modul založený na mikroprocesoru Atmega168 a je nejnovějším ze série modulů s USB rozhraním.

Modul má 14 digitálních V/V (vstupně výstupních) pinů, z nichž 6 může být použito jako výstupy PWM (Puse Width Modulation, impulsní šířková modulace) – na digitální výstupy je pomocí PWM zapisována analogová hodnota (PWM vlna). Dále obsahuje 6 analogových vstupů, 16 MHz krystalový oscilátor, USB rozhraní napájecí konektor, ISCP (In-Circuit Serial Programming) rozhraní a tlačítko pro resetování mikroprocesoru.

Paměť
Mikroprocesor ATmega168 má tři typy paměti:

» FLASH: pro uchovávání kódu programu
» SRAM: statická paměť pro uchovávání a práci s proměnnými
» EEPROM: elektronicky mazatelná semipermanentní paměť

Pouze FLASH a EEPROM jsou trvalé paměti, ve kterých data zůstanou i po odpojení napájení.

Vstupy a výstupy
Každý ze 14 V/V digitálních pinů, může být použit buď jako vstup nebo jako výstup. Každý pracuje na napětí 5 V s maximálním vstupním nebo výstupním proudem 40 mA a má vnitřní pull-up rezistor s rozmezím 20 – 50 kΩ. Piny 3, 5, 6, 9, 10 a 11 mohou poskytnout PWM výstup (viz níže). Pokud je cokoliv připojeno k pinům 0 a 1, tak dochází k rušení komunikace přes rozhraní USB, zablokování možnosti nahrání zdrojového kódu nebo jiné sériové komunikace.

Arduino Diecimila má 6 analogových vstupů, každý z nich s 10 bitovým rozlišením (1024 hodnot). Standardně tyto vstupy kvantují signál mezi nulovým potenciálem (0V) a 5 V, nicméně je možné posunout horní hranici při využití pinu AREF a vhodného přizpůsobovacího kódu, ne však přes hranici 5 V.

Komunikace

Modul Arduino Diecimila nabízí několik možností pro komunikaci s počítačem, jiným modulem Arduino nebo jinými mikrokontroléry. Procesor Atmega168 poskytuje UART TTL (5V) sériovou komunikaci, která je vyvedena na digitálních pinech 0 (RX) a 1 (TX). Integrovaný mikročip FTDI FT232RL zajišťuje propojení komunikačního rozhraní s rozhraním USB a ovladač FTDI, který je součástí vývojového prostředí Arduino, poskytuje virtuální sériovou bránu pro software na počítači.

Procesor Atmega168 také podporuje sběrnici I2C [5] a synchronní komunikační rozhraní SPI (Synchronous Peripheral Interface) [5]. Vývojové prostředí Arduino obsahuje knihovnu pro jazyk Wire, která zjednodušuje použití I2C sběrnice.

Napájení

Modul Arduino Diecimila může být napájen jak z rozhraní USB, tak z externího zdroje. Zdroj napájení lze vybrat pomocí propojky PWR_SEL na desce mikrokontroléru. Pro výběr napájení z USB se propojka umístí na dva piny blíže USB konektoru, pro napájení z externího zdroje na dva piny blíže ke konektoru napájení. Při použití externího zdroje by se napětí jím poskytované mělo pohybovat v rozmezí 6 – 12 V. Použit může být jak adaptér, tak baterie s tím, že při použití baterie se doporučuje ji připojit na piny Gnd a Vin.

Programování

Modul Arduino Diecimila je programovatelný pomocí vývojového prostředí Arduino. Použitý mikroprocesor Atmega168 má již předpřipravený podprogram pro zavedení startovacího kódu, který umožňuje nahrát zdrojový kód do mikroprocesoru bez nutnosti použití externího zařízení pro programování. Komunikace při nahrávání zdrojového kódu probíhá podle protokolu STK500. Podprogram pro zavedení startovacího kódu je možné přemostit a programovat tak procesor Atmega168 pomocí rozhraní ISCP.

Vývojové prostředí Arduino

Vývojové prostředí Arduino je jednoduché IDE (Integrated Development Environment), umožňující uživatelům jednoduchou tvorbu zdrojového kódu v jazyce Arduino a jeho následující nahrání do paměti mikrokontroléru. Prostředí je vyvinuté v jazyce Java a založené na volně dostupnéhm softwaru jako je Processing, avr-gcc a jiných. Je dostupné pro OS Windows, OS Linux a Mac OS X.

Mikroprocesor	Atmega168
Pracovní napětí	5V
Napájecí napětí	6-12 V
Digitální V/V piny	14 (6 s PWM výstupem)
Vstupní analogové piny	6
Maximální proud protékající V/V pinem	40 mA
Paměť typu Flash	16 KB
Paměť typu SRAM	1 KB
Paměť typu EEPROM	512 B
Frekvence oscilátoru	16 Mhz
USB nadproudová ochrana	500mA

Tab. 1. Vlastnosti modulu Arduino Diecimila

1.3. Řízení digitální hudební elektroniky pomocí MIDI

V předchozích byly teoreticky rozebrány snímače, které lze použít pro snímání interakcí člověka, a byly probrány technické specifikace modulu, který bude převádět signály ze snímačů na zprávy, kterými bude ovládáno podřízené zařízení. Poslední chybějící částí teorie je protokol podle, kterého bude probíhat přenos zpráv. A přesně toto je cílem této kapitoly.

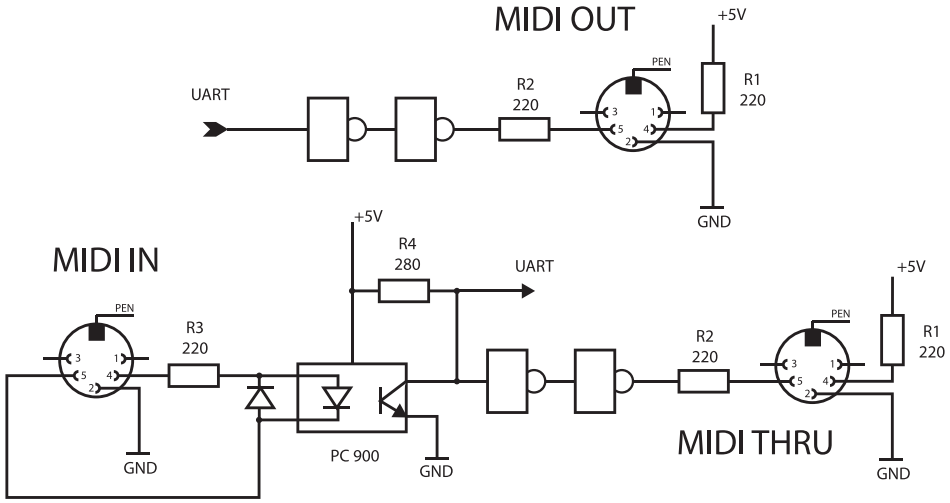
Jak je dáno zadáním této práce, tato kapitola bude zaměřena na protokol MIDI. Pro řízení digitální hudební elektroniky samozřejmě existuje více možností, nicméně protokol MIDI byl jedním z prvních a doposud je zdaleka nejrozšířenější. Je pravděpodobné, že časem bude překonán, tato doba zatím ale není v dohledu. Kdesi za pomyslným obzorem se rýsují nové perspektivní projekty, jako například protokol Open Sound Control, nicméně než dosáhnou stejného rozšíření a popularity jako protokol MIDI, uplyne možná i několik desetiletí.

MIDI je standardizovaný protokol umožňující různým elektronickým hudebním nástrojům, počítačům a jiným zařízením vzájemnou komunikaci, synchronizaci a řízení. Počátky jeho vzniku se datují do období mezi rokem 1981, kdy Dave Smith,

označovaný též jako otec MIDI, podal návrh organizaci Audio Engineering Society o zařazení specifikace rozhraní mezi její standardy, a rokem 1983, kdy byla vydána specifikace MIDI 1.0. Protokol MIDI patří mezi nejvýznamnější protokoly hudební techniky a je na něm pozoruhodné, že se od svého vzniku zachoval v téměř nezměněné podobě.

Protokol MIDI jako takový pracuje na principu „řídící - podřízený", kdy řídící zařízení vysílá řídící signály zařízením. Řídící zařízení je často označováno jako MIDI kontrolér. MIDI specifikace obsahuje jak popis komunikačního protokolu, tak popis fyzického rozhraní.

Fyzické rozhraní MIDI je simplexní asynchronní sériová komunikační linka s přenosovou rychlostí 31,25 kBaudů, založená na 5mA proudové smyčce, kdy stavu logické nuly odpovídá stav, při kterém protéká smyčkou proud. Využívá se jeden start bit, osm datových bitů a jeden stop bit. Standardním konektorem rozhraní MIDI je 5 pinový konektor DIN a na zařízení pracujícím podle standardu MIDI většinou najdeme tři tyto konektory – IN (vstupní), OUT (výstupní), THRU (průchod dat – jsou na něj kopírována data ze vstupu). Pro odstranění zemních smyček mezi přístroji je vstup od samotného přístroje oddělen pomocí optoizolátoru s reakčním časem menším než 2ms (Sharp PC-900).



Obr 5. Zapojení rozhraní MIDI

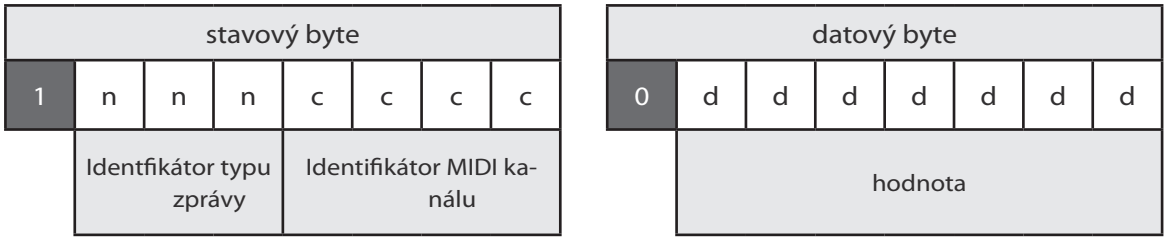
Pro připojení rozhraní MIDI k osobnímu počítači se často využívá „gameportu", který je integrován na základové desce nebo zvukové kartě a který má na své piny vyvedeny UART (Universal Asynchronous Receiver/Transmitter, univerzální asynchronní přijímač/vysílač) signály, nebo portů USB (Universal Seriál Bus, univerzální sériová sběrnice), které taktéž podporují komunikaci pomocí UART.

1.3.1. Struktura protokolu MIDI

Ačkoliv se jedná o „Digitální rozhraní pro hudební nástroje", nejsou promocí MIDI přenášeny hudební signály jako takové, ale jsou přenášeny pouze zprávy s informací o nové události nebo o změně parametrů – tyto zprávy se nazývají MIDI zprávy. Každá zpráva se skládá z řetězce bytů. Nejběžnější jsou 3 bytové zprávy, ale jsou definovány i zprávy 1 a 2 bytové a jeden typ zpráv se dokonce může skládat z nekonečného počtu bytů. Nicméně to co mají všechny typy zpráv společné je první byte zprávy, který se nazývá „stavový byte" a jako jediný má nastavený nejvýznamnější bit, čímž je zajištěna synchronizace zpráv – žádná jiný byte nemá tento byt nastaven, a proto lze rozeznat, kde zpráva začíná. Vzhledem k tomuto jsou stavové byty pouze v rozsahu 0x80 – 0xFF a datové byty pouze v rozsahu 0x00 - 0x7F (0x je označení číslic v 16 číselné soustavě používané v jazyce C).

Stavové byty v rozsahu 0x80 – 0xEF uvozují zprávy, které mohou být vysílány na jakémkoliv z 16 dostupných kanálů a které spadají do takzvané „Voice Category" (kanálové MIDI zprávy). Jak bylo popsáno výše, stavový byte se skládá ze dvou šestnáctkových číslic neboli osmi bitů. Pro určení, o jaký typ zprávy se jedná, je zapotřebí tento byte rozdělit na dvě čtyřbitové části, tzv. nibbly - dolní nibble (bity 0 – 3) a horní nibble (bity 4 – 7). Horní nibble určuje o jaký typ zprávy se jedná a dolní určuje na jakém kanále k události dochází. V kategorii kanálových MIDI zpráv rozlišujeme tyto typy zpráv, kde *n* označuje číslo kanálu:

- 8*n* Note off (nota vypnuta)
- 9*n* Note on (nota zapnuta)
- A*n* AfterTouch (tlak na klávesu)
- B*n* Control Change (změna kontroleru)
- C*n* Program Change (změna programu)
- D*n* Channel Pressure ()
- E*n* Pitch Wheel ()



Obr 6. Struktura MIDI zprávy

i ***Poznámka:***
MIDI definuje 16 kanálů, které jsou v protokolu značeny jako 0-A čili 0-15, nicméně všechna zařízení včetně počítačového softwaru číslují kanály 1-16. Takže pokud řízené zařízení obdrží stavový byte o hodnotě 0x90 (=10010000 binárně), tedy horní hibble 0x9 (=1001 binárně) a a dolní 0x0 (=0000 binárně), je nutné si uvědomit, že se jedná o zprávu „nota zapnuta“ na kanále 1.

Zprávy se stavovým bytem 0xF0 – 0xFF nejsou vysílány po žádném specifickém kanálu a všechna zřetězená zařízení je „slyší“ a mohou se rozhodnout na ně reagovat. Tyto zprávy se dále dělí na zprávy „System Common“ (běžné systémové zprávy) a zprávy „System Realtime“ (systémové zprávy v reálném čase). Navíc některé zprávy z těchto rozsahů nejsou používány a jsou rezervované pro budoucí použití.

1.3.2. Kanálové MIDI zprávy

Vzhledem k faktu, že MIDI slouží k digitální reprezentaci událostí, ke kterým dochází při hře na běžný hudební nástroj, je jeho součástí specifikace reprezentací MIDI not. Těchto not je v MIDI 128 a jejich seznam je uveden v příloze 1. Kromě not se jako druhého hlavního ovládacího prvku používají takzvané kontroléry. Mimo těchto dvou jsou do kanálových zpráv řazeny i další typy zpráv, které přímo nesouvisí s „hraním“ hudby, ale spíše s nastavením nástroje.

Note Off
Zpráva určuje, která z hrajících not má přestat hrát. V některých případech je vhodné nastavit rychlost přechodu mezi stavy hrající a vypnutá (používá se při napodobování technik hry na jiné než klávesové nástroje).

	Význam bytu	Rozsah
	Status byte	0x80 – 0x8F
	Číslo noty	0x00 – 0x7F
	Rychost přeběhu	0x00 – 0x7F

i ***Poznámka:***

Kromě „Note Off“ umožňuje MIDI vypnou notu i pomocí zprávy „Note On“ s nulovou rychlostí přeběhu. Nicméně tento způsob není implementován u všech zařízení. Detailnější rozbor lze nalézt v [1].

Note On

Zpráva určuje, která nota má začít hrát a opět je zde definována určitá rychlost přechodu mezi stavy vypnutá a hrající. Jak bylo zjištěno, rychlost stisku klávesy je přímo úměrná dynamice hry, která je důležitá zejména u klávesových nástrojů.

	Význam bytu	Rozsah
	Status byte	0x90 – 0x9F
	Číslo noty	0x00 – 0x7F
	Rychost přeběhu	0x00 – 0x7F

AfterTouch

Stejně jako „původní“ klávesové nástroje, kde zvuk hraných not liší podle tlaku na klávesy, tak i MIDI má specifikován tlak na klávesy právě hrajících not.

	Význam bytu	Rozsah
	Status byte	0xA0 – 0xAF
	Číslo noty	0x00 – 0x7F
	Hodnota tlaku na klávesu	0x00 – 0x7F

Control change

Změní hodnotu kontroléru. Kontroléry jsou modifikátory, pomocí nichž lze modifikovat parametry jednotlivých zvuků nebo celého hudebního výstupu. Běžnými ovládacími prvky kontrolérů jsou např. tahové a otočné potenciometry a spínače.

	Význam bytu	Rozsah
	Status byte	0xB0 – 0xBF

	Význam bytu	Rozsah
	Číslo kontroléru	0x00 – 0x7F
	Nová hodnota na niž je kontrolér naszaven	0x00 – 0x7F

Kontrolérů existuje několik druhů. Užití některých je definováno v MIDI specifikaci, užití jiných je ponecháno na výrobci. Nejběžnější dělení je uváděno na:

» Spojité
» Spínače
» Inkrementační/dekrementační
» Povely

Běžný rozsah MIDI zpráv je 0-127, nicméně tento rozsah je podstatně nepřesný. Proto byla zavedena dvojice zpráv MSB a LSB zpráv o změně spojitých kontrolérů (označovány také jako hrubé a jemné nastavení kontrolérů). MSB i LSB zpráva mají odlišné číslo kontroléru: například pro kontrolér Modulation Wheel je MSB – 1 a LBS - 33. Pokud zařízení přijme MSB zprávu, kontrolér nastaví nejprve kontrolér na hrubou hodnotu a LSB na 0x00 a po přijmutí LSB zprávy hodnotu upřesní (Výjimkou je kontrolér Bank Change který čeká i na přijetí LSB zprávy a až poté nastaví hodnotu kontroléru). Takto je dáno pro rozlišení hodnot spojitých kontrolérů 14 bitů, čili 16384 hodnot.

Podskupinou spojitých kontrolérů jsou tzv. jednobytové kontroléry (70-95). Jsou spojité, nicméně rozsah jejich hodnot, je pouze 0 – 127, proto nevyužívají dělení na MSB a LSB zprávy. Do této skupiny patří převážně kontroléry nastavující barvu zvuku (70 - 74) a efektové kontroléry (91 - 95).

Speciální skupinou kontrolérů jsou registrovaná a neregistrovaná čísla parametrů (Registered Parameter Number RPN a Non-Registered Parameter Number NRPN). Tyto kontroléry slouží k výběru parametru, který je potom nastaven pomocí kontrolérů Data Entry, Increment Data Button nebo Decrement Data Button. Vzhledem k tomu, že tyto kontroléry využívají MSB/LSB zprávy, je počet použitelných kontrolérů obrovsky rozšířen. Čísla kontrolérů jsou RPN MSB/LSB 98/99 a NRPN MSB/LSB 100/101. Využití NRPN je ponecháno na výrobci a využití RPN je specifikováno v [1].

Spínače nabývají pouze hodnot „vypnuto“ a „zapnuto“ a jejich rozsah je 64 – 69

(0x3F 0x45) Inkrementační a dekrementační kontroléry jsou dva - Data Button Increment (96) a Data Button Decrement (97). Slouží k postupnému zvyšování či snižování parametrů určených pomocí RPN nebo NRPN. Poslední skupinou jsou povely (120 – 127), sloužící k nastavení parametrů přijímače.

Program change

Změní program (rejstřík). Podřízená zařízení obsahují programy, které lze pomocí těchto MIDI zpráv přepínat.

	Význam bytu	Rozsah
	Status byte	0xC0 – 0xCF
	Číslo programu na, který se má stávající změnit	0x00 – 0x7F

Channel Pressure

Tento typ zpráv je obdobou zpráv AfterTouch s tím rozdílem, že jsou aplikovány na celý kanál, tudíž změna tlaku se projeví na všech hrajících notách.

	Význam bytu	Rozsah
	Status byte	0xD0 – 0xDF
	Hodnota tlaku aplikovaná na všechny klávesy	0x00 – 0x7F

Pitch Wheel

Určuje relativní změnu výšky tónu, tzv. ohýbání tónu (pitch bender).

	Význam bytu	Rozsah
	Status byte	0xE0 – 0xEF
	Relativní změna výšky tónu	0x00 – 0x7F

1.3.3. Běžné systémové zprávy

System Exclusive (SysEx)

Tento typ zpráv slouží k přenosu velkého množství dat směrem do kontroléru. Obsah dat je libovolný a jejich velikost není specifikována. Důležité ovšem je, že všechny

SysEx zprávy začínají status bytem 0xF0 a končí status bytem 0xF7. Více o tomto typu zpráv v dalším textu.

SysEx zprávy jsou jeden typ systémových zpráv, které nejsou vázány jen na určitý kanál, ale jsou určeny pro celý systém. Nicméně jak již slovo „exclusive“ napovídá, zprávy tohoto typu jsou odlišné od ostatních systémových zpráv a stejně tak jsou odlišné reakce v různých zařízeních.

V principu je využití SysEx zpráv ponecháno plně na výrobci. Běžně bývají tyto zprávy pro přenos různorodých dat, například parametrů pro podřízená zařízení, dat k samplerům a přenos souborů.

Stejně tak jako jejich využití je na výrobci ponechán i jejich formát. Jsou však jistá základní pravidla, která musí být dodržena. V MIDI řetězci jako takovém může být zapojeno velké množství různých zařízení různých výrobců (což je také jeden ze základních cílů MIDI), je proto nezbytné, aby zpráva dosáhla pouze k zařízením, která ji budou správně interpretovat a budou mít technické vybavení na její zpracování.

Jak již bylo zmíněno dříve základem rozpoznání SysEx zprávy stavový bytů určujících počátek a konec zprávy - 0xF0 = začátek a 0xF7 = konec.

Vzhledem k tomu, že formáty SysEx zpráv se liší s výrobcem je prvním datovým bytem tzv. identifikátor výrobce (manufacturers ID). Základní rozdělění těchto identifikátorů je mezi Americké, Evropské, Japonské a jiné výrobce a identifikátory pro speciální použití. Jsou přidělovány MMA (MIDI Manufacturers Association) a JMSC (Japanese Midi Standard Committee).

Význam bytu	Rozsah
Status byte	0xF0
ID výrobce	0x00 – 0x7F
ID zařízení	0x00 – 0x7F
Sub-ID #1 - volitelné	0x00 – 0x7F
Sub-ID #2 - volitelné	0x00 – 0x7F
Data - libovolný počet	0x00 – 0x7F
Ukončení SysEx	0xF7

Dále je již formát SysEx zprávy ponechaný na výrobci a mohou následovat užitečná data. Je však běžné, že výrobce vyrábí více druhů MIDI zařízení s různým technickým vybavením, která nemusí být schopna zpracovat zprávu určenou pro jiná zařízení téhož výrobce. Je proto běžné že následuje ID zařízení a další identifikátory.

Speciálními identifikátory výrobce jsou identifikátory v rozmezí 0x7D – 0x7F. 0x7D je identifikátor určený pro nekomerční využití např. ve výzkumu a na školách a nesmí být použit v komerčních zařízeních uvedených na trh. 0x7E a 0x7F jsou využity pro univerzální MIDI zprávy a jejich použití je pevně stanoveno MIDI standardem [1].

MTC Quarter Frame

Některé „master"zařízení využívají tuto zprávu pro synchronizaci svých „slave" zařízení pomocí synchronizačního kódu MIDI Time Code.

	Význam bytu	Rozsah
	Status byte	0xF1
	Časová známka	0x00 – 0x7F

Song Position Pointer

Zpráva k nastavení posunu v rámci skladby. Pozice je určena pomocí 14-ti bitové hodnoty složené z obou bytů. Určuje posunutí momentálně přehrávané skladby

	Význam bytu	Rozsah
	Status byte	0xF2
	Hodnota posunutí	0x00 – 0x7F
	Hodnota posunutí	0x00 – 0x7F

Song Select

Určuje kterou píseň má sekvencer hrát. (speciální zařízení pro záznam a přehrávání MIDI událostí)

	Význam bytu	Rozsah
	Status byte	0xF3
	Číslo písně	0x00 – 0x7F

Tune Request
Po přijetí této zprávy,provede zařízení ladění.

Význam bytu	Rozsah
Status byte	0xF6

1.3.4. Systémové zprávy v reálném čase

Jak je patrné z názvu , jsou tyto zprávy vysílané v reálném čase. Většinou jsou to zprávy vztahující se k časování systému a přehrávání. Vzhledem ke své povaze neobsahují žádná data.

MIDI Clock
Synchronizační zpráva, která je vysílána periodicky, v závislosti na tempu řídícího zařízení.

Význam bytu	Rozsah
Status byte	0xF8

MIDI Tick
Synchronizační zpráva, která je vysílána periodicky v intervalech 10 ms

Význam bytu	Rozsah
Status byte	0xF9

MIDI Start
Zpráva kontrolující přehrávání sekvenceru, přiměje jej začít přehrávat skladbu od jejího začátku.

Význam bytu	Rozsah
Status byte	0xFA

MIDI Stop
Zastaví přehrávání skladby.

	Význam bytu	Rozsah
	Status byte	0xFC

MIDI Continue

Obnoví přehrávání předtím zastavené skladby ze stejného místa.

	Význam bytu	Rozsah
	Status byte	0xFB

Active Sensing

Zařízení vysílá tuto zprávu, pokud za posledních 300 ms nevysílalo jinou MIDI zprávu. Tím oznamuje ostatním zařízením v MIDI systému, že spojení je stále aktivní.

	Význam bytu	Rozsah
	Status byte	0xFE

System Reset

Po obdržení této zprávy by se zařízení mělo resetovat do základního nastavení.

	Význam bytu	Rozsah
	Status byte	0xFF

2. REALIZACE ŘEŠENÍ

V této kapitole je rozebrána realizace zvoleného řešení. Toto řešení není samozřejmě jediné možné, naopak z povahy zadání je tento projekt relativně flexibilní a záleží a výsledné řešení záleží na konkrétní aplikaci. Nicméně snahou je, aby i zvolené řešení bylo co nejvíce flexibilní a jednoduše rozšiřitelné v případě že si to budou budoucí aplikace vyžadovat.

i

Poznámka:

Vzhledem k faktu, že toto je studentský a z valné části samofinancovaný projekt a jedním z hlavních limitujících faktorů je finanční nákladnost jednotlivých komponent, je často dána přednost technologiím levnějším před technologiemi sofistikovanými. Obdobným případem je i dostupnost jednotlivých komponent. V takovýchto případech je snahou na toto čtenáře upozornit a uvést i postup při řešení pomocí sofistikovanějších nebo jakkoliv nedostupných technologií.

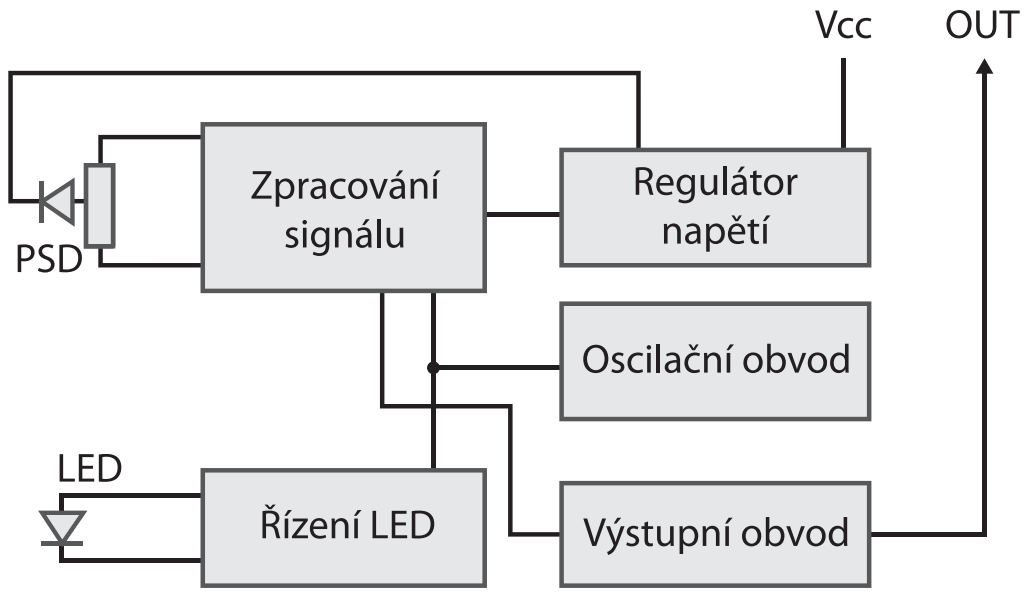
2.1. Snímání pohybů vystupujícího

V kapitole 2 bylo diskutováno nad možnostmi snímání akcí vystupujícího, které charakteristiky jsou z tohoto hlediska relevantní a jaké jsou možnosti jejich měření. Cílem této práce je implementovat řízení digitální hudebních zařízení pomocí snímačů neelektrických veličin dvou typů. V případě tohoto projektu byla jako nejrelevantnější zvolena měření vzdálenosti vystupujícího od objektů umístěných na jevišti a váhy vystupujícího působící silou na podlahu jeviště nebo objekty na něm umístěné. Toto jsou předpoklady, ze kterých následující práce vychází.

2.1.1. Snímání vzdálenosti vystupujícího

Pravděpodobně nejvhodnějšími technologiemi pro snímání vzdálenosti mezi vystupujícím a objekty umístěnými na jevišti jsou měření vzdálenosti pomocí ultrazvukového snímače nebo snímače optického (principy obou technologií jsou popsány v kapitole 2). Jako optimální pro aplikaci v tomto projektu byl po zvážení nároků, možností a vlastností zvolen snímač GP2Y0A21YK0F - optický snímač od společnosti SHARP založený na triangulační metodě. Tato konkrétní varianta snímače je s analogovým výstupním signálem snímající v rozmezí vzdáleností 10-80cm, nicméně společnost SHARP nabízí varianty jak s analogovým tak s digitálním výstupem pro rozsahy měření do 500cm.

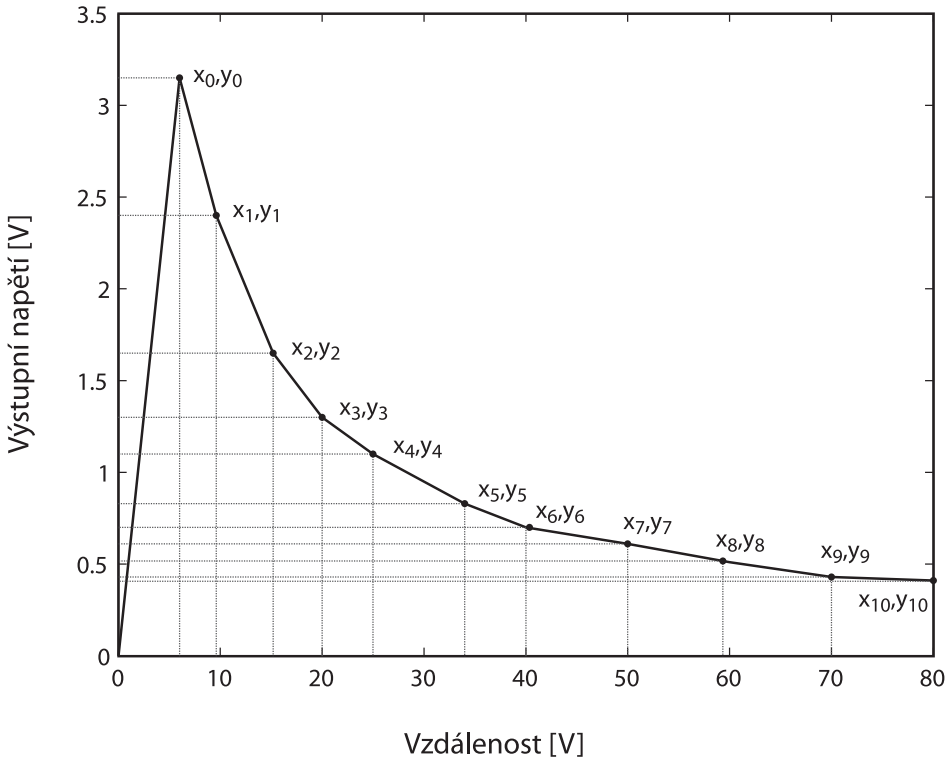
Optický snímač vzdálenosti SHARP GP2Y0A21YK0F [7]
SHARP GP2Y0A21YK0F je optický snímač vzdálenosti s analogovým výstupem, skládající se z detektoru vzdálenosti (position sensitive dectector - PSD), infračervené diody (infrared emitting diode - IRED) a obvodů pro zpravování signálu. Princip tohoto snímače je založen na triangulační metodě, pro poskytuje stabilní výsledky pro objekty s různou odrazivostí, v různých prostředích a za různých teplot. Výstupem tohoto snímače je napětí přímo úměrné vzdálenosti.



Obr 7. Vnitřní zapojení snímače GP2Y0A21YK0F

Měřící rozsah	10 -80 cm
Výstupní napětí při vzdálenosti 80cm	0,4 V
Rozdíl výstupního napětí (80 - 10 cm)	1,9 V
Napájecí napětí	4,5 - 5,5 V
Odběr proudu	30 - 40mA

Tab. 2. Vlastnosti optického snímače GP2Y0A21YK0F



Obr 8. Průběh výstupního napětí snímače GP2Y0A21YK0F

Jak je vidět z grafu průběhu [7], výstupní napětí senzoru je značně nelineární. V měřícím rozsahu by se funkce výstupního napětí *F(x)* dala považovat za funkci exponenciální, kdy k největší m změnám dochází v posledních cca. 20-ti centimetrech. To v důsledku znamená, že pokud budeme měřit přímo, bude se citlivost snímače měnit v závislosti na vzdálenosti. Toto není nutně na závadu a záleží na požadavcích konkrétní aplikace, bude ale dobré zauvažovat nad možnostmi linearizace.

Při linearizaci výstupu ze snímače, lze obecně volit mezi linearizaci analogovou a digitální. Oba dva způsoby zakládají na stejné myšlence, nicméně každý volí jinou cestou. Základní myšlenkou, společnou pro obě metody, je linearizace pomocí inverzní funkce. V případě analogového způsobu, spočívá linearizace v přivedení nelineárního výstupu snímače na vstup obvodu s inverzní charakteristikou průběhu. Nejtěžším úkolem je v tomto případě návrh obvodu, který má přesně inverzní charakteristiku, která je v čase dostatečně stálá. Analogová linearizace se z tohoto důvodu dnes používá už jen velmi zřídka. [8]

Linearizace digitální je postavena na stejné myšlence, pouze není zapotřebí vřadit za snímač blok s inverzní charakteristikou, nýbrž jsou vzorkovaná a kvantizovaná data použita jako argumenty funkce inverzní k průběhu charakteristiky snímače. Obor hodnot inverzní funkce má poté lineární průběh. Nicméně inverzní funkce může, a často také je, výpočetně náročná, proto je často výhodnější tuto funkci nahradit takzvanou „linearizační tabulkou“ (LUT - Look-Up Table). Princip vyhledávací tabulky spočívá v přímém přiřazení hodnoty v závislosti na argumentu. Toto je zejména vhodné pokud známe počet a konkrétní stavy, kterých mohou argumenty nabývat. Přesně což je případ tohoto projektu, kdy víme, že procesor ATmega168 je vybavena 10 bitovým A/D převodníkem a pomocí grafu průběhu od výrobce je možné dostatečně přesně přiřadit všech 1024 hodnot. Jak tedy linearizační tabulku vytvořit?

Graf průběhu výstupního napětí na vzdálenosti, je po částech lineární (pravděpodobně pouze graf, nikoliv průběh samotný), proto stačí tedy odečíst hodnotu napětí a vzdálenost v bodech zlomu na intervalu 10-80 cm. Druhou alternativou je provést samostatné měření průběhu nicméně v práci byla použita první.

Nejprve je nutné spočítat reprezentaci signálu na vstupech mikrokontroléru. Ačkoliv je horní hranice vstupního napětí nastavitelná můžeme dopředu počítat, že po přizpůsobení se bude signál na vstupu mikrokontroléru pohybovat v rozmezí 0 – 5 V. Hodnoty napětí proto nejprve posuneme, aby hodnota výstupního napětí v 80 cm byla rovna 0, a výsledný interval transformujeme na interval 0 – 1024, což je výstupní obor hodnot z A/D převodníku.

$$y'(n) = \frac{(1023 \times (y(n) - y_n))}{y_0} \tag{2.1}$$

Interval 10 – 80 cm se poté rozdělí na 1024 bodů, kdy je jeden dílek minimální měřitelná vzdálenost:

$$x'(a) = \frac{(80-10)}{1023 \times a};\ a \in < 0, 1023 >$$

(2.2)

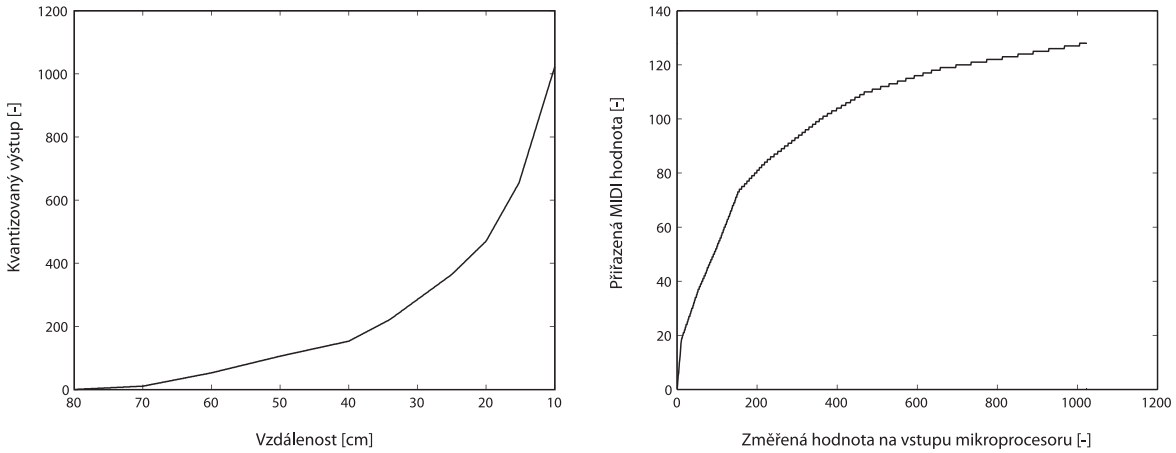
Obor hodnot je poté interpolován ve všech 1024 bodech:

$$y''(a) = y'_{n-} + (x''(n) - x_{n-}) \times \frac{y'_{n+} - y'_{n-}}{x_{n+} - y_{n-}}$$

(2.3)

kde x_{n-}, $y'n$- jsou souřadnice nejbližšího nižšího známého prvku a x_{n+},y'_{n+} nejbližšího vyššího prvku

Tímto je vytvořena kompletní rekonstrukce charakteristiky signálu na vstupu mikrokontroléru. Inverzní funkci poté získáme velmi jednoduše a to zaměněním definičního oboru a oboru hodnot



Obr 9. Porovnání hdonot na výstupu snímače a hodnot přiřazovaných vyhledávací tabulkou

Samotnou LUT je vhodné realizovat jako pole kde index prvku je hodnota z definičního oboru a obsah z jejího oboru hodnot. Toto vyžaduje, přeskládání prvků, eliminaci prvků, které mají duplicitní hodnotu v definičním oboru a následnou interpolaci hodnot. Další možnou úpravou je transformace nového oboru hodnot do rozsahu 0 – 127, tedy rozsahu hodnot, jež mohou nabývat MIDI zprávy, čímž je docíleno přímého přiřazení. Součástí práce je na přiloženém disku m-file, což je skript programu Matlab, který tuto tabulku generuje. Výsledek linearizačního procesu je zobrazen na obrázku 9 a je také jedním z výstupů skriptu.

2.1.2. Snímání Váhy vystupujícího

Pravděpodobně nejvhodnějšími pro měření váhy v aplikacích, ve kterých bude tento projekt nasazen, jsou snímače založené na změně kapacity nebo rezistivity. Ač obě technologie by byly schopny splnit daný cíl téměř stejně byla zvolena cesta snímačů rezistivních.

Nejvhodnější by bylo v tomto projektu použít snímače FSR (Force Sensing Resistor,). Tato technologie je pro danou aplikaci téměř ideální co se týká fyzikálních i elektrických parametrů. Bohužel nevýhodou je maloobchodní nedostupnost v České Republice. Nicméně technologie, na níž jsou založeny FSR snímače, je jednoduchá vhodná pro použití. Při použití vhodných materiálů, lze docílit obdobných obdobného efektu, jako je piezoresistence polovodičů.

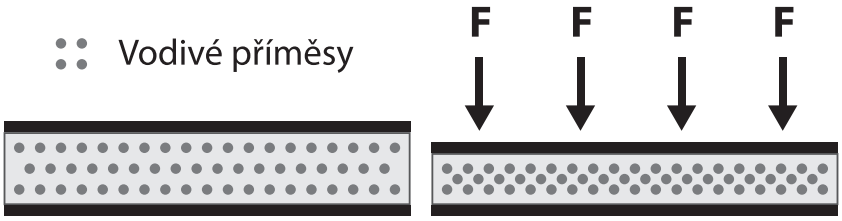
Snímače váhy vlastní výroby

FSR snímače jsou založeny na piezorezistentní vlastnosti určitých polovodičů, kdy při zvýšení tlaku na materiál klesá hodnota odporu materiálu. U FSR snímače je tento jev zpoůsoben změnou koncentrace vodivých částic které byly do polovodiče dotovány.

i ***Poznámka:***

Závěry, prezentovány v tomto textu, byly vyvozeny experimentálně z testovaných vzorků. Je velmi pravděpodobné, že při rozsáhlejším testování by mohly být objeveny jiné materiály a vlastnosti, než zde popsané. Všechna měření popsaná v tomto oddílu byla provedena za improvizovaných podmínek a v proto výsledky pravděpodobně trpí chybou z nepřesnosti měření.

Od objevení polymerů se stále více technologií směřuje právě tímto směrem, proto je oblast polymerů intuitivně prvním místem, kde takovýto materiál hledat. Pokud hledáme materiál, který mění svůj odpor, hledáme zároveň materiál, který mění svoji vodivost, tudíž je v normálních podmínkách alespoň částečně vodivý. Polymery samy o sobě vodivé nejsou, nicméně spousta jich má v elektrotechnice uplatnění (vodivé) a to díky, většinou uhlíkovým, příměsím. V normálním stavu se dá předpokládat, že materiál je dotován rovnoměrně a má stabilní objemovou vodivost. Pokud chceme jeho vodivost změnit, musíme změnit vodivost mezi jednotlivými vodivými částicemi v jeho vnitřní struktuře. A přesně k tomuto dochází při působení externí síly a stlačením daného materiálu (Obr. 10). Hlavním předpokladem hledaného materiálu je jeho relativní flexibilita, aby umožnil dostatečné přiblížení vodivých částic pro v dostatečném rozsahu měřitelnou změnu vodivosti.



Obr 10. Ilustrace změny hustoty vodivých částic v elektrovodivé pěně při stlačení

Běžně dostupných materiálů splňující tyto podmínky je na trhu celá řada, většinou se jedná o různé vodivé pryže nebo pěny.

Běžné použití pryží jsou vodivá těsnění, gumové elektrody, a jiné. Pro uplatnění v takovýchto aplikacích je nutná vodivost co největší, a proto jsou hojně dotované. Navíc je struktura většiny pryží relativně hustá. Díky tomu jsou změny vodivosti pryže relativně malé a špatně se měří.

i

Poznámka:

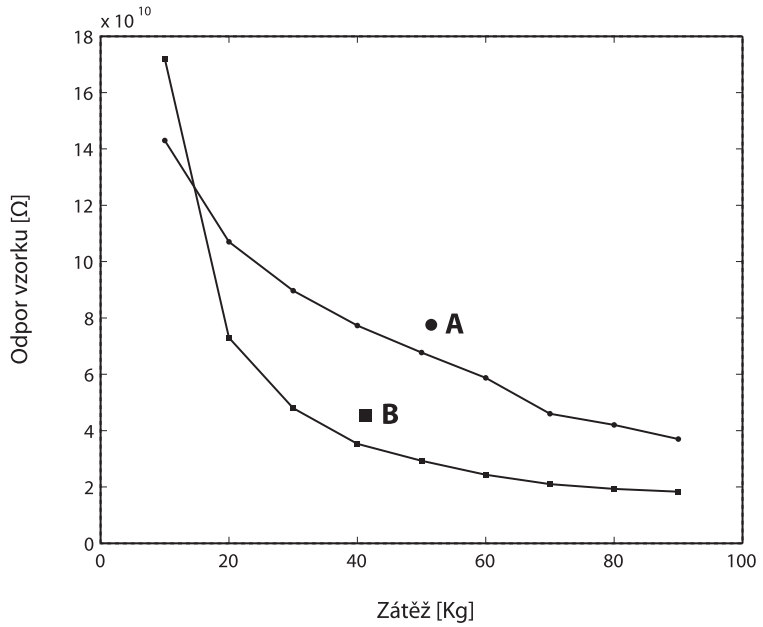
Pokud by se našla, nebo vyrobila vhodně dotovaná pryž, bylo by její použití více než vítané, neboť díky pevné struktuře by bylo možné připravit oproti pěně relativně tenký vzorek, čímž by se snížila celková tloušťka výsledného snímače. Navíc se dá předpokládat menší paměť a větší stabilita vzorků.

Jinak je tomu u vodivých pěn. Tyto pěny jsou většinou vyráběny z polyuretanu nebo polyetylenu a jejich nejběžnější použití je jako antistatických pěn používaných při balení integrovaných obvodů, jejich běžná rezistivita je uváděna 10e5-10e6 ohm, což je relativně vhodné pro měření, a tak lze pomocí těchto pěn vyrobit popisovaný snímač váhy. Tyto pěny se vyrábějí v různých variantách šířek a tvrdostí, proto je zapotřebí vybrat vhodnou variantu a experimentálně ověřit zda se hodí pro danou aplikaci. V tomto projektu byla použita pěna 900 QCM tloušťky 5mm a střední tvrdosti.

! ***Pozor:***

Pro podobné aplikace se užívá i disipativních pěn, které mají ovšem rezisitivitu až okolo 10e9 Ω, což je pro měření velmi těžce použitelné. Typicky jsou disipativní pěny růžové a vodivé tmavě či světle šedé.

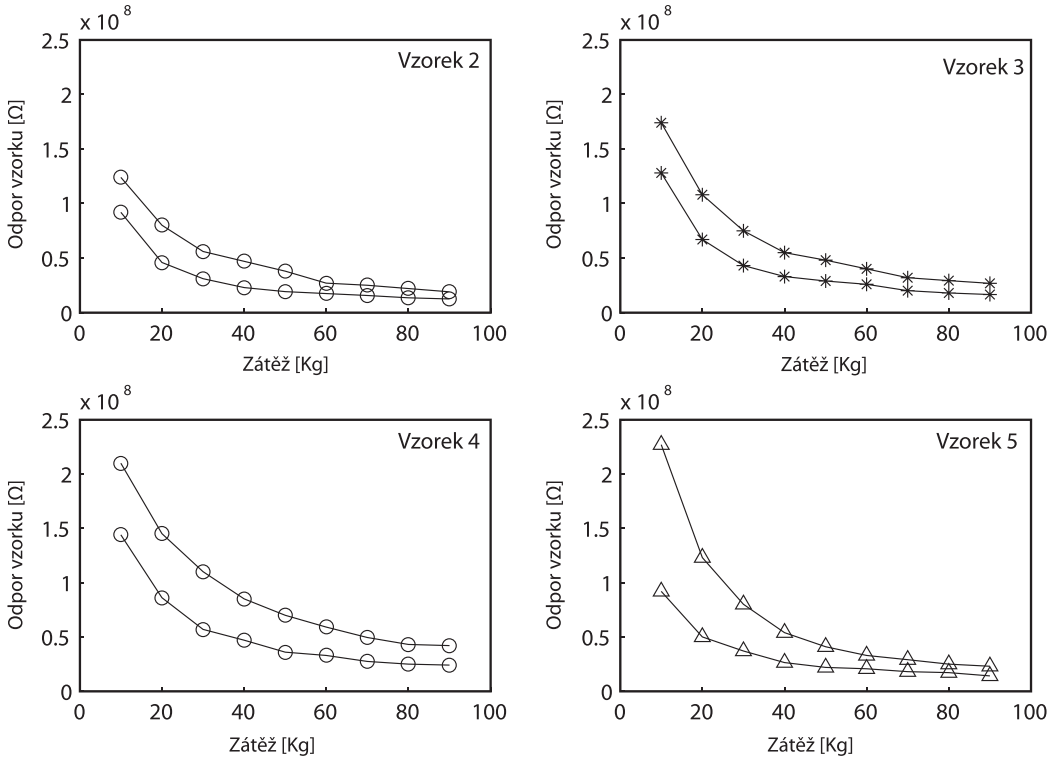
Pokud už máme vhodný materiál s vhodnou vodivostní charakteristikou, dalším krokem je zamyslet se nad elektrodami, kterými bude odpor materiálu měřen. K dispozici je celá řada materiálů a konkrétní výběr opět záleží na konkrétní aplikaci. Ve všech aplikacích je ovšem zapotřebí zohlednit několik následujících faktorů: velikost výsledného senzoru, jeho maximální zatížení a rozložení zatížení na ploše senzoru.



Obr 11. Porovnání charakteristik odporu v závislosti na zatížení pro snímače velikosti A - 4cm x 4cm a B - 8cm x 8cm

Při vývoji senzorů pro tento projekt je jako elektrod použito cuprextitových destiček, které se běžně používají pro výrobu plošných spojů. Cuprextit je materiál ze skelné tkaniny, tvrzený fenolformaldehydovou pryskyřicí (jinak znám také jako pertinax), s nanesenou tenkou vrstvou mědi. Měď poskytuje skvělé vlastnosti jako vodivý materiál elektrod a pretinax dostatečně pevný materiál jako oporu, čím se dosáhne rovnoměrného zatížení senzoru.

Dalším krokem je určení vhodné velikosti snímačů a pro toto je vhodné experimentálně ověřit charakteristiky odporu v závislosti na zatížení pro snímače různé velikosti. Jak ukazuje Obr. 10, snímač 4cm x 4cm vykazuje oproti snímači 8cm x 8cm výrazně lineárnější charakteristiku a proto byl vybrán jako vhodný rozměr pro konstrukci snímačů v tomto projektu. Nicméně snímač a) byl používán déle a pro více testů, oproti tomu snímač b) byl vyroben nový pro tento experiment. Z tohoto by se dal vyvodit předpoklad, že charakteristika rezistivity materiálu se mění v závislosti na opotřebení. Tento předpoklad byl záhy potvrzen experimentálně.



Obr 12. Porovnání charakteristik odporu snímačů velikosti 4cm x 4cm před a po vystavení intenzivnímu zatěžování v rozmezí 10 - 90 Kg po dobu cca. 40minut.

Z Obr.12 a je patrné, že ke změně charakteristiky opravdu dochází. Při větším opotřebení snímače se zdá charakteristika konverguje do obdobného průběhu (vzorek 2 a 5). Dále je vidět, že k hlavnímu zlomu charakteristiky dochází v oblasti okolo 30 Kg a k menšímu v oblasti o kolo 60 Kg. Dalo by se tedy doporučit použití tohoto snímače pro zatížení v oblastech 10-30, 30-60 a 60-90 Kg.

i

Poznámka:

Tyto grafy ukazují, poněkud neblahý jev - paměť materiálu. Tento efekt se projevuje až při delším zatížení. Z experimentů vyplívá domněnka že existuje jakési limitní charakteristika k níž snímače konvergují. Snímače které jsou více opotřebeny konvergují rychleji. Hodnoty z měření jsou dostupné na přiloženém

Nyní máme připraven převodník váhy na odpor, dalším krokem je rozšířit tento převodník o další stupeň, kterým změnu odporu převedeme na veličinu, kterou jsme schopni měřit pomocí mikroprocesoru. Těmito veličinami jsou většinou proud nebo napětí, v případě modulu Arduino a procesoru ATMega168, který je vybaven 6 analogovými vstupy, zvolíme napětí. K tomuto se nám nabízejí dvě možnosti:

» a) měření pomocí odporového děliče napětí
» b) měření pomocí Wheatstoneova můstku.

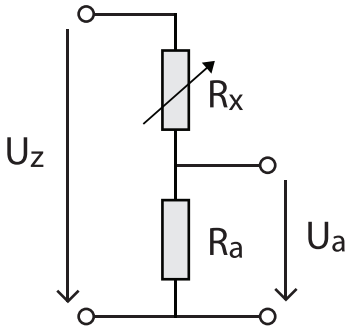
Měření pomocí odporového děliče napětí

Měření pomocí odporového děliče napětí je v podstatě nejjednodušší měření v elektrotechnice a vychází ze dvou elementárních zákonů elektrotechniky:

a) 2. Kirchoffova zákona:
„Součet úbytků napětí na spotřebičích se v uzavřené části obvodu (smyčce) rovná součtu elektromotorických napětí zdrojů v této části obvodu.“

b) Ohmova zákona:
„Napětí na prvku je přímo úměrné procházejícímu proudu.“

$$U = R \cdot I \tag{2.4}$$



Obr 13. Měření odporu snímače pomocí odporového děliče napětí

Napětí na celé větvi je Uz pro celkový odpor platí:

$$R = R_x + R_a \tag{2.5}$$

čili proud protékající celou větví je:

$$I = \frac{U_z}{R} = \frac{U_z}{(R_x + R_a)} \tag{2.6}$$

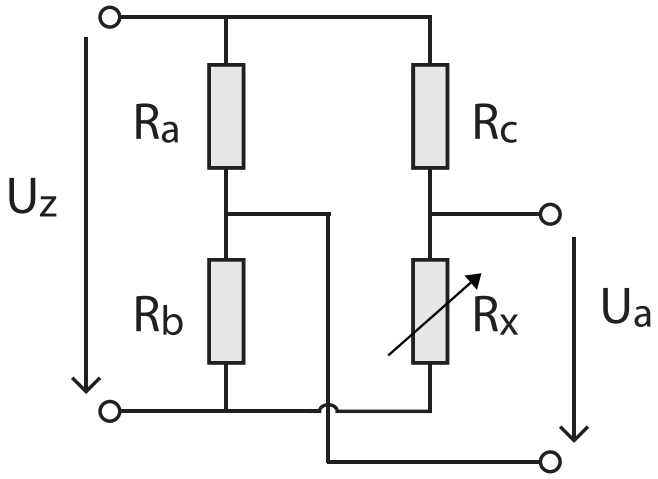
kde Uz je napětí zdroje, Rx je odpor snímače a Ra odpor odporu na němž měříme. Vzhledem k tomu, že odporem Ra protéká proud I, Je úbytek napětí na Ra roven:

$$U_a = \frac{R_a}{I} = U_z \cdot \frac{R_a}{R_a + R_x} \tag{2.7}$$

což je vztah pro napětí na výstupu děliče v závislosti na odporu snímače.

Měření pomocí Wheatstoneova můstku

Ve svojí podstatě se jedná o zdokonalenou metodu měření pomocí odporového dělič, kdy jsou místo jednoho děliče použity děliče dva a na výstupu se měří rozdílové napětí mezi těmito děliči



Obr 14. Měření odporu snímače pomocí Wheatstoneova můstku

Pro výpočet napětí můžeme použít vztah 2.7, pro napětí na děliči, který aplikujeme na každou větev a dostaneme vztahy:

$$I = \frac{U_z}{R} = \frac{U_z}{R_x + R_a} \tag{2.8}$$

$$U_x = U_z \cdot \frac{R_x}{R_c + R_x} \tag{2.9}$$

Poté pomocí principu superpozice dostaneme vztah pro rozdíl mezi napětími na jednotlivých děličích:

$$U_{b-x} = U_b - U_x = U_z \cdot \left(\frac{R_b}{R_a + R_b} - \frac{R_x}{R_c + R_x} \right) \tag{2.10}$$

Kterým opět získáme vztah pro napětí na výstupu snímače závislé na odporu. Oproti obvodu samotného děliče má toto zapojení výhodu, že pokud můstek vyvážíme, tzn. $R_a=R_b=R_c=R_x$ je výstupní napětí rovno nule.

Oba dva obvody, jak dělič, tak Wheatstonův můstek, jsou ovšem náchylné na zatížení v dalším stupni, kdy uvedené vztahy přestanou platit. Proto je nutné obvod oddělit od dalšího stupně, nejlépe pomocí operačního zesilovače.

Při použití Wheatstonova můstku, lze pomocí vhodné volby odporů $R_a=R_b=R_c$ můstek vyvážit na určitou hodnotu a pokud odpor R_x stoupne nad hodnotu, na kterou je můstek vyvážen, bude U_{B-X} záporné, naopak pokud R_x klesne pod tuto hodnotu, bude U_{b-x} kladné a bude stoupat s klesajícím odporem R_x. Jak nepopsáno v předchozím odstavci, musíme obvod nějak oddělit od dalších stupňů operačním zesilovačem. Tímto můžeme při vhodném zapojení nastavit určitý práh odporu R_x, pod nímž bude výstup ze snímače roven nule (respektive šumu v operačním zesilovači). Z tohoto hlediska by bylo použití této metody preferováno, na druhou stranu zapojení s napěťovým děličem je flexibilnější a na takto ošetřený vstup je možné připojit i snímače s napěťovým výstupem. Z tohoto důvodu byla pro konstrukci snímačů dána přednost variantě s odporovým děličem.

i ***Poznámka:***

Varianta s odporovým děličem je také doporučována, výrobcem FSR snímačů.

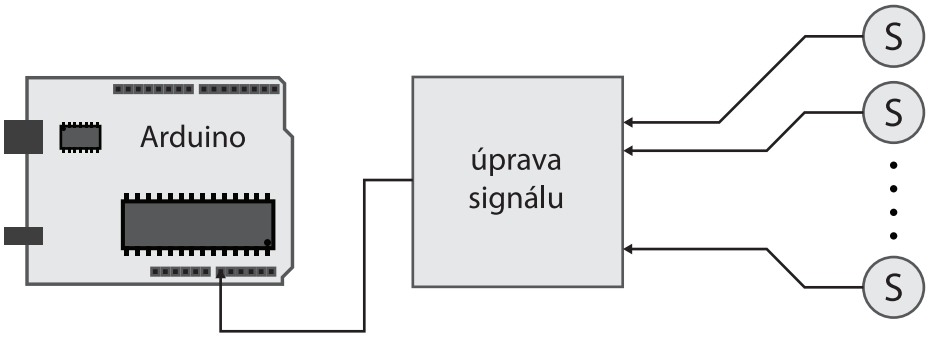
Při použití metody odporového děliče je zapotřebí zvážit hodnotu odporu, na němž měříme výstupní napětí. Při výběru hodnot je důležité, aby hodnota napětí na výstupu byla při minimálním zatížení co nejnižší, ale nadrbou stranu při maximálním zatížení co nejvyšší, takže bude dobře měřitelná. Jako ideální se ukázala hodnota 10 MΩ. Z grafu na obrázku je patrné že odpor při 10 kg je přibližně 145MΩ a pro zatížení 90 kg přibližně 35 MΩ. Při dosazení těchto hodnot spolu s napájecím napětím 12 V do vztahu 2.7 dostaneme výstupní parametry snímače:

Měřící rozsah	10 - 90 Kg
Výstupní napětí při zatížení 90 Kg	0,05 V
Rozdíl výstupního napětí (80 - 10 cm)	2,62 V
Napájecí napětí	12 V
Odběr proudu	54 µA

Tab. 3. Vlastnosti odporového snímače

2.2. Připojení snímačů k modulu Arduino

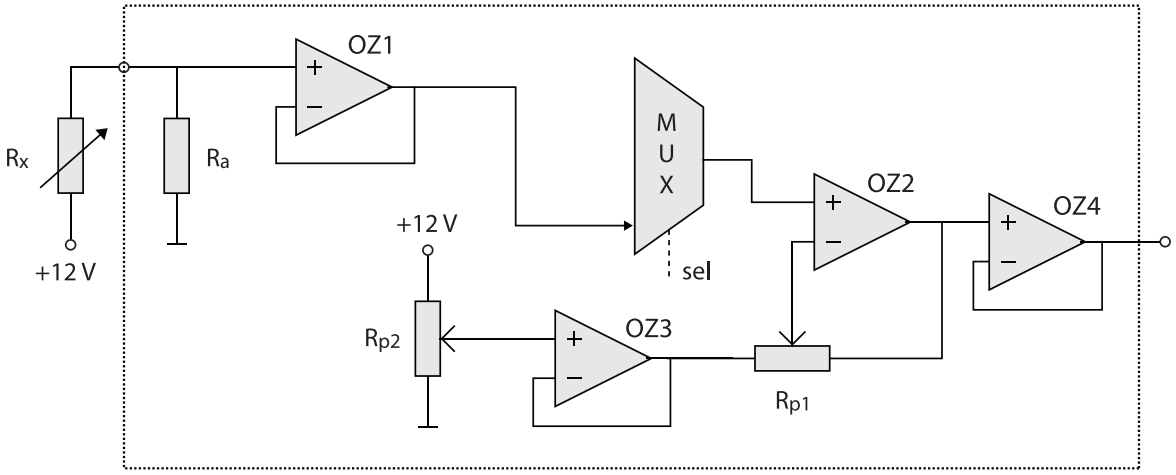
Modul Arduino a mikroprocesor ATMega168 má 6 analogových vstupů/pinů se standardním rozsahem 0-5V. Vzhledem k faktu, že každý typ snímače má jiný rozsah, je nutné toto nějak ošetřit. Jednou z možností by mohlo být upravení horní hranice rozsahu pomocí pinu AREF [3,4], Toto řešení je ovšem značně neflexibilní. Zároveň využití pouze 6 vstupů, které nám Arduino v základu nabízí, značně podhodnocuje možnosti, které nám nabízí protokol MIDI. Z těchto důvodů byla zvolena varianta, kdy bude před modul vřazen obvod, který upraví vstupní analogový signál a zároveň pomocí multiplexerů rozšíří vstupní potenciál modulu Arduino.



Obr 15. Signálová cesta od snímačů k procesoru

Tímto způsobem bude výsledný produkt modulární a jeho jednotlivé části nahraditelné v závislosti na požadavcích aktuální aplikace. Je nutno podotknout, že obvody pro zpracování signálu jsou přizpůsobeny na jednotlivé druhy snímačů. Niméně lze obvody přizpůsobit aby podporovaly snímačů více. Na realizovaný modul je možné připojit i jiné odporové snímače s přibližně stejným rozsahem nebo snímače s napěťovým výstupem, ale už ne snímače například kapacitní, pro ty je nutné vyvinout a vytvořit obvod jiný.

Při tvorbě přizpůsobovacích obvodů je integrován multiplexer pro zvětšení počtu vstupů, obvod pro oddělení obvodů snímače od zatížení vyšších stupňů, obvod pro zesílení signálu snímačů na hladinu 5V při maximálním zatížení, a obvod pro posun stejnosměrného napětí, které vzniká na měřeném odporu v děliči.



Obr 16. Zapojení přizpůsobovacího obvodu

Začněme popis s předpokladem, že se jedná o přizpůsobovací obvod pro zapojení se snímači váhy. Nicméně později dokážeme, že je tento obvod využitelný i pro snímače vzdálenosti a jiné snímače s napěťovým výstupem. V zapojení podle Obr.16 je odpor R_a druhou polovinou odporového děliče z Obr.12.

Pro oddělení děliče od zatížení z dalších částí obvodu souží OZ 1 v zapojení sledovače napětí. V ideálním případě má zesilovač v tomto zapojení zesílení $A = 1$. Nicméně vzhledem k tomu že pracujeme s odpory přesahujícími 100MΩ, je zapotřebí zvolit zesilovač co s vstupním odporem alespoň v řádu desítek TΩ.

Dalším stupněm je multiplexer, v rámci projektu byl použit šestnácti kanálový analogový multiplexer/demultiplexer typu 4067. Tento rozšiřuje vstupní kapacitu zařízení. Je možné použít i multiplexery s jiným počtem kanálů, to je ovšem, jak si později ukážeme, zapotřebí ošetřit v programu mikroprocesoru. Výsledkem je, že při potřebě stačí vyměnit přizpůsobovací modul za jiný s vhodným počtem kanálů a tím upravit počet vstupů podle potřeby.

Výstupní zesilovač OZ 2 slouží pro zesílení signálu do požadované hodnoty (v tomto případě 5 V). Zesílení je nastaveno pomocí potenciometru R_{p1}, kterým lze nastavit zesílení v rozmezí od 1 do maximální hodnoty napětí při které začne zesilovač saturovat. Běžně by měl být potenciometr R_{p1} připojen na nulový potenciál, pro

odstranění dříve zmíněného prahu napětí vytvořeného děličem, je ovšem zapojen zesilovač OZ 3, který pomocí potenciometru R_{p2} vytváří „virtuální zem“. Zapojení OZ 3 je opět sledovač napětí, pro oddělení potenciometrů R_{p1} a R_{p2}, aby se vzájemně neovlivňovaly. Na konec je vřazen OZ 4 který poskytuje napěťový výstup a odděluje obvod od dalších stupňů.

Díky tomuto zapojení jsme schopni nastavit na výstupu modulu rozmezí 0 – 5 V, v závislosti na připojených snímačích. Toto zapojení jako celek zároveň umožňuje použití snímačů s napěťovým výstupem, kdy tyto snímače připojíme na vstup, čili na odpor R_a, který je zároveň v paralelní kombinaci s vstupním odporem neinvertujícího vstupu OZ 1 vstupním odporem modulu. Běžnou vlastností obvodů s napěťovým výstupem je nízký výstupní odpor, proto nebude tento vstup zatěžovat a napětí na odporu bude rovno výstupnímu napětí snímače. Tento modul je tedy využitelný pro všechny odporové snímače s odporem v mezích 10MΩ – 100MΩ (pro Vcc 12 V) a snímače s napěťovým výstupem v mezích 0 – 5V. Omezení pro odporové snímače lze navíc odstranit použitím vhodných dekád nebo trimrů na místo odporů R_a.

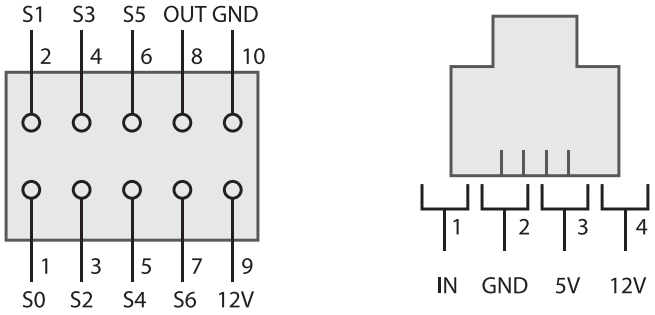
2.2.1. Rozhraní přizpůsobovacího modulu

Opět i zde je kladen hlavní důraz na flexibilitu a dobře či špatně navržené rozhraní hraje v tomto případě hlavní roli.

Rozhraní mezi přizpůsobovacím modulem a snímači by mělo být schopné a připravené pro připojení více typů snímačů. Z tohoto důvodu je vhodné na rozhraní vyvést jak měřící vstup, tak i napájecí napětí a zem. Problémem zůstává, že každý typ snímače má rozdílné napájecí napětí. Faktem ale je, že většina snímačů má napájecí napětí v okolí 5 V nebo 12 V, proto jsou na rozhraní vyvedeny obě tyto napětí a rozhraní je tedy čtyřvodičové. Fyzicky je toto rozhraní realizováno konektorem RJ-14 a zapojení vodičů na rozhraní ukazuje následující obrázek.

Při návrhu rozhraní mezi přizpůsobovacím modulem a modulem Arduino je hlavním aspektem, který je třeba zohlednit, počet vodičů mezi oběma moduly. Proto je vhodné si nyní shrnout, které signály je potřeba přenášet. Hlavním je určitě analogový signál z měřených snímačů, dalšími jsou signály pro řízení multiplexerů, napájecí napětí, a zem. Kromě signálů pro řízení multiplexerů je na každý signál zapotřebí jeden vodič. Ačkoliv je v základní verzi použit 16-ti kanálový multiplexer, na

jehož ovládání by stačili 4 řídící signály, je zde počítáno s potenciálním rozšířením v budoucnu. Protokol MIDI, který v tomto projektu využíváme, povoluje až 128 not a kontrolérů na kanál, je zajímavou vizí použít toto číslo jako limit pro počet snímačů na kanál, čímž bychom eventuelně dosáhli možnosti obsadit kompletní kanál pomocí dvou analogových vstupů modulu Arduino (noty a kontroléry). Pro ovládání 128 kanálového multiplexeru potřebujeme 7 řídících signálů, což po sečtení dává 10 signálů, které je zapotřebí přenést mezi přizpůsobovacím modulem a modulem Arduino. Pro fyzické rozhraní je v tomto případě vhodné použít 10-ti vodičového plochého kabelu a header konektorů MLW10A ‚jakožto vidlic, a PFL10, jakožto zásuvek.



Obr 17. Zapojení konektorů na rozhraní mudulu přizpůsobovacích obvodů
A - konektor rozhraní s modulem Arduino: S0-S6 signály pro ovládání multiplexeru, S0 LSB;
B - Rozhraní se snímači

Na straně modulu Arduino je rozhraní realizováno stejným konektorem. Je proto zapotřebí někde 6 těchto konektorů umístit. Dále je zapotřebí umístit před samotné analogové vstupní piny „pull-down“ rezistory, aby i v případě že na vstup nebude připojeno žádné další zařízení, byl vstup připojen na zem a „neplaval“. Tyto odpory i konektory rozhraní jsou umístěny na externí desce plošných spojů („štítu“), která se umisťuje nad samotný modul Arduino a je k němu připojena pomocí oboustranných dutinkových lišt. Tímto jsou také opět vyvedeny všechny piny modulu a je možné jich použít pro další rozšíření. Kompletní zapojení „štítu“ je v příloze 3.

Modul přizpůsobovacích obvodů i všechny jeho periferie jsou napájeny z modulu Arduino, kde je vyvedeno napájecí napětí na pinu Vin. Na modulu přizpůsobova-

cích obvodů je toto napájení regulováno na 5V pomocí obvodu 7805 s maximálním výstupním proudem 1A. Je proto nutné, aby celkový proud odebíraný ze všech rozhraní snímačů nepřekročil tuto hodnotu. Napájení modulu Arduino je provedeno pomocí adaptéru s výstupním napětím 12 V.

2.2.2. Propojení modulu Arduino s osobním počítačem

Propojení modulu s počítačem je důležitou částí projektu. Jednak je možné modul Arduino programovat pomocí sériového rozhraní a není tedy nutné používat externí paralelní nebo sériový programátor (nicméně možnost programování přes ICSP je možná). Za druhé, ačkoliv MIDI protokol původně nebyl navržen pro komunikaci s počítačem, v dnešní době spíše převládá přístup, kdy se pomocí MIDI ovádacích prvků ovládá software na počítači místo hardwarových synetizátorů a jiných MIDI zařízení. A přesně toto je případ tohoto projektu.

Při návrhu možných řešení, je nutné vyjít z dostupných zdrojů které jsou k dispozici na modulu Arduino a na běžném osobním počítači nebo notebooku.

Jak je již zmíněno výše, modul Arduino obsahuje 14 digitálních programovatelných vstupně/výstupních pinů s tím, že piny 0 a 1 jsou též používány jako vysílací a přijímací pro komunikaci po sériovém rozhraní. Dále má modul integrován výstup pro rozhraní USB spolu s potřebným integrovaným obvodem pro realizaci obousměrného sériového spojení.

Na druhé straně, běžný osobní počítač je standardně vybaven konektorem USB nebo konektorem Cannon9 s rozhraním RS232 připojeným ke „gameportu“ na zvukové kartě. Běžně je součástí systému ovladač zařízení kompatibilní se standardem MPU-401. K dispozici jsou i ovladače jiné, vydané různými výrobci MIDI zařízení sdruženými v organizaci MMA (MIDI Manufacturers Association).

Je patrné, že existuje více způsobů jak realizovat spojení mezi modulem a počítačem a proto budou tyto způsoby v následujícím oddílu jeden po druhém představeny. Nicméně předem už je dáno, že spojení bude realizováno pomocí USB rozhraní, neboť je toto v dnešní době nejvíce podporováno.

2.2.3. Použití sériového rozhraní procesoru ATmega168

Při použití sériového rozhraní je práce programátora modulu značně zjednodušena, neboť lze použít metod sériového rozhraní *Serial.begin()* a *Serial.print(),* implementovaných v IDE Arduino. V tomto případě se programátor nemusí zabývat samotným převodem dat do registrů brány UART ani časováním.

Pro časování stačí pouze použít funkci *Serial.begin()*, při inicializaci ve funkci *setup()*, se vstupním parametrem 31250, čímž definujeme výstupní rychlost dat na 31250 baudů (přenosová rychlost MIDI). Vzhledem k tomu, že v jednom znaku přenášíme jeden bit, je přepočet na bity 1:1. Výsledná rychlost tedy je 31250 b/s.

Pro odeslání připravených dat po sériovém rozhraní v příslušném místě hlavní smyčky *loop()* stačí pouze zavolat metodu *Serial.print()* s odesílanými daty poskytnutými jako vstupní parametr (y).

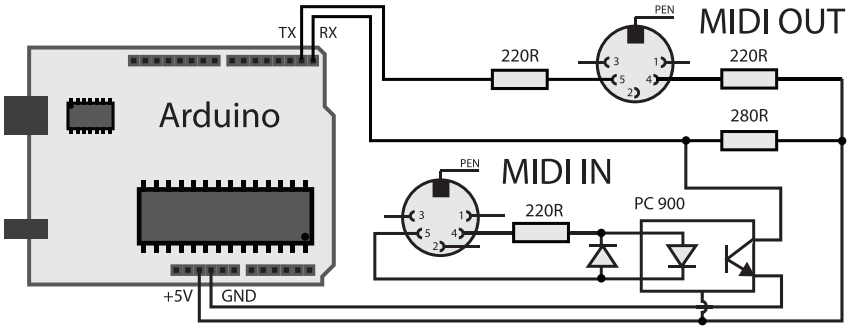
V tuto chvíli jsou data odesílána po sériovém rozhraní a jsou možné dva způsoby, jak a přes jaké rozhraní je přenést do počítače.

Použití pinů RX, TX modulu Arduino

Na obrázku 18 je naznačeno schéma zapojení při realizaci MIDI vstupu a výstupu pomocí DIN konektorů a RX, TX sériových portů procesoru ATmega168. Zapojení vychází ze specifikace MIDI, ale není s ní úplně shodné. Oproti MIDI specifikaci chybí na výstupu invertující hradlo a tranzistor, nicméně funkčnost je zachována. Maximální proud protékající smyčkou je 5mA, V/V piny mají mezní hodnotu 40 mA a pin je schopen odebrat dostatek proudu sám.

Samotné propojení je realizováno pomocí kabelu DIN (vidlice) – Cannon9, který je běžně používán připřipojení MIDI zařízení k osobnímu počítači. Na osobních počítačích bývá MIDI rozhraní zprostředkováno zvukovou kartou, kdy má suková karta pomocí „gameportu“ vyvedeny UART signály. Na toto rozhraní bývá navázán standardní obladač praující v režimu kompatibility s MPU - 401.

Toto řešení má svoje pro i proti. Pro je určitě fakt, že pomocí zařízení s tímto rozhraním lze ovládat i starší zařízení, které mají rozhraní podle originální MIDI specifikace. Proti je, že dnešní notebooky již nemají zvukovou kartu s gameportem.



Obr 18. Modul Arduino s MIDI vstupem a výstupem realizovaným pomocí DIN konektorů a sériových RX, TX

i

Poznámka:

I pokud nemá počítač gameport na zvukové kartě existuje řešení. Na trhu jsou běžně dostupné konvertory MIDI – USB které obsahují jeden a více MIDI vstupů a USB výstup pro připojení k osobnímu počítači. Toto řešení je ovšem poněkud drahé neboť cena takovýchto konvertorů se pohybuje v rámci tisíc korun.

Využití USB rozhraní na modulu Arduino

Pro komunikaci s osobním počítačem lze též využít USB rozhraní integrované na modulu Arduino. Programování modulu je v obou případech stejné, neboť piny RX a TX na mikroprocesoru ATmega jsou připojeny jak k pinům 0 a 1 na modulu, tak na vstup čipu FTDI FT232BM, který zajišťuje sériovou komunikaci pomocí rozhraní USB.

Součástí vývojového prostředí je též ovladač FTDI nebo lze nejaktuálnější verzi stáhnout z Internetových stránek společnosti FTDI (www.ftdichip.com). Tyto ovladače slouží pro vytvoření virtuální sériové brány (většinou brána 0x232) a po spuštění přenosu ze strany modulu jsou data dostupná na této bráně.

Zde ovšem nastává první problém a to, že ačkoliv data již v počítači jsou, počítač nemá „tušení“, že se jedná o data v MIDI formátu určené pro nějakou specifickou aplikaci. Aby byla data ve formátu MIDI pro počítač rozpoznatelná, je zapotřebí aby data přicházela z MIDI rozhraní, které je v počítači reprezentováno ovladačem.

Jedním z řešení je použití standardního ovladače kompatibilního s rozhraním MPU-401 pracujícím na bráně s adresou 0x330 a nějakým způsobem data přicházející na sériovou bránu přemostit na bránu tohoto ovladače. Přemostit tyto dvě brány lze pomocí jednoduché aplikace. Takovouto aplikací je například volně dostupná aplikace S2MIDI dostupná na www.mameteam.net.

Jiným řešením je například použití ovladače Serial->MIDI od společnosti Roland, dostupného na jejích Internetových stránkách (www.roland.com). Tento ovladač v podstatě zapouzdří vybranou sériovou bránu a vytvoří v systému počítače nové rozhraní MIDI, přes které jsou data vysílaná modulem Arduino dostupná cílové aplikaci.

Nyní by se zdálo, že vše je již nastaveno a nic nebrání přenosu dat z modulu k cílové aplikaci. Opak je bohužel pravdou a ještě jeden problém technického charakteru zbývá nevyřešen. Tento problém je spjat s faktem, že u obou způsobů využíváme sériové brány a systém Windows standardně povoluje provoz sériové brány pouze v několika pevně definovaných rychlostech. Bohužel žádná z nich není 31250 kBaudů, což je přenosová rychlost definována standardem MIDI, se kterou ovladače MIDI rozhraní načítají data z příslušných bran. Díky tomu dochází k rozpadu synchronizace.

Řešením tohoto problému je provédst modifikaci ovladače FTDI. Tato modifikace je popsána v [4] a spočívá v editaci souboru *ftdiport.inf* před instalací ovladače. Výsledkem je, že systém Windows považuje ovladač za ovladač pracující na jím povolené rychlosti, nicméně ovladač jako takový pracuje na rychlosti jiné. Je dobré takto modifikovat rychlost co nejbližší rychlosti požadované. V tomto případě je změněna rychlost 38400 b/s na 31250 b/s.

Při instalaci ovladačů je zapotřebí nejprve provést tuto modifikaci FTDI ovladače, a teprve poté modifikovaný ovladač nainstalovat. Po instalaci ovladače FTDI a připojení modulu Arduino ověříme na jakém COM portu je modul připojen. Poté nainstalujeme ovladač Serial->MIDI od společnosti Roland, kdy v průběhu instalace vybereme příslušný COM port. Při použití aplikace S2MIDI vybereme příslušný vstupní COM port a příslušný výstupní MIDI port a stiskneme tlačítko start, kterým přemostíme vybrané porty. Pro kontrolu lze použít aplikaci s funkcí MIDI analyzátoru. Touto je například MIDI-OX který je ze svých stránek (www.midiox.com) dostupný pro osobní potřebu zdarma.

Použití běžných pinů pro realizaci výstupního rozhraní
Tato možnost je podle svého fyzického provedení téměř stejná jako při použití konektoru DIN připojeného na pin 1 modulu, který je v podstatě sériovým výstupem z mikroprocesoru Atmega168, s tím rozdílem, že nyní lze konektor připojit na kterýkoliv pin, neboť nebudeme využívat funkce sériového rozhraní.

V tomto případě je nutné zabezpečit osobně jak převod dat do formátu definovaným UART tak i jejich časování a synchronizaci. Obtížnost tohoto není nějak nestandardní, nicméně vzhledem k tomu, že jistou již takovéto funkce v jazyce Arduino implementovány jedná se spíše o zbytečnost. Navíc to posouvá zaměření celého projektu spíše na problematiku protokolu UART než MIDI.

2.3. Obslužný program mikroprocesoru

Při tvorbě obslužného programu je nejprve důležité si uvědomit funkce programu. Hlavní funkcí je zpracovávat data ze snímačů na analogových vstupech, převádět je do formátu specifikovaného protokolem MIDI a tyto potom příslušnou rychlostí vysílat po sériovém rozhraní do podřízených zařízení. Vzhledem k tomu, že počet vstupů je rozšířen pomocí multiplexerů, je další funkcí programu řízení těchto multiplexerů. Nakonec pro rozšíření možností a flexibility výsledného zařízení je implementována možnost nastavování parametrů jednotlivých snímačů pomocí zpráv typu SysEx.

i ***Poznámka:***
Následující text obsahuje pouze povrchní popis funkcí programu. V příloze C. je k nahlédnutí dokumentace k programu, v příloze B vývojové diagramy důležitých funkcí a metod a zdrojový kód programu je umístěn na přiloženém disku

Na rozdíl od běžného programu v jazyce C/C++, kód v prostředí Arduino neobsahuje funkci *main()*, nýbrž je tato funkce rozdělena na funkci *setup()* a funkci *loop()*. Funkce *setup()* je inicializační a proběhne automaticky pár vteřin po spuštění (zapojení napájení). Funkce *loop()* funkci *main()* ve svojí podstatě zastupuje a běží v nekonečné smyčce. Konkrétní obsah těchto dvou funkcí, tak jak je definován ve zdrojovém kódu programu, je zobrazen v ukázce kódu 1.

O mnoho důležitější pro chod programu jsou ovšem pole struktur analogInput typu

aInput a sensorInput typu sInput. Tyto struktury obsahují proměnné, které určují chod programu, charakteristiku elementů připojených na snímačové vstupy a jim přiřazené MIDI prvky. Navíc jsou tyto proměnné nastavitelné pomocí MIDI SysEx zpráv, jak bude vysvětleno dále. Struktura obou struktur je zobrazena na ukázce kódu 3 o několik stránek dále.

Jak bylo zmíněno výše, funkce *setup()* je inicializační. Nejprve tato funkce pomocí metody sériového rozhraní *Serial.begin()* nastaví přenosovou rychost která bude na sériovém rozhraní použita. V tomto případě je rychlost 31250 kBaudů, tedy rychlost specifikována v MIDI. Dále pomocí funkcí *pinMode()*, integrovaných v prostředí Arduino, inicializuje vybrané piny jako výstupy. V tomto případě se jedná o digitální piny, které slouží k řízení multiplexerů a byly definovány dříve v kódu. Posledním krokem je inicializace struktur analogInput a sensorInput. Toto je provedeno dvojitým cyklem.

Vnější cyklus prochází přes všechny analogové vstupy. Nejprve každému přiřadí počet multiplexovaných kanálů, dále ho nastaví na neaktivní (je nuté ho poté aktivovat pomocí příslušné SysEx zprávy) a spustí vnitřní cyklus procházející přes všechny multiplexované vstupy.

Vnitřní cyklus nastaví každému vstupu snímače tyto vlastnosti: číslo - je přiřazeno číslo podle multiplexovaného kanálu; hodnotu - je nastavena na 0; kanál - výchozí stav je, že každý analogový vstup vysílá na vlasním kanálu; mód - pomocí hodnoty *true* je nastaven do módu noty; linearizaci - určuje zda je na vstupní hodnotu aplikována linearizace; aktivitu vstupu - opět je ve výchozím stavu neaktivní; a nakonec zda vstup momentálně hraje -přirozeně je výchozí hodnotou *false*.

i

Poznámka:

Proměnné *linearization* a *mode* jsou definovány jako typ boolean z důvodu úspory místa. Všechny struktury typu aInput i sInput je momentálně nutné uchovávat v interní SRAM paměti procesoru ATmega168 o velikosti 1 KB, která je sdílana všemi proměnnými použitými v programu. Velikost jedné struktury aInput je 9b (8+1) a sInput 24 b (8+8+8+1+1+1+1), dohromady 448 b (6x9+16x24), dále je 128 b využito jako buffer sériového portu. Po sečtení zbívá 424 b pro ostatní výpočty programu. Překročení hranice 50% bývá z praxe uváděno jako riskantní, je proto nutné šetřit každý byte. Nicméně v současném stavu je program stabilní.

```
void setup(){
   Serial.begin(31250);
   pinMode(S[0], OUTPUT);
   pinMode(S[1], OUTPUT);
   pinMode(S[2], OUTPUT);
   pinMode(S[3], OUTPUT);
   pinMode(S[4], OUTPUT);
   pinMode(S[5], OUTPUT);
   pinMode(S[6], OUTPUT);
   for(int i = 0;i < 6;i++){
      analogInput[i].mux = DEFAULT_MUX;
      analogInput[i].enabled = true;
      for(int e = 0;e < analogInput[i].mux;e++){
         sensorInput[i][e].number = e;
         sensorInput[i][e].value = 0;
         sensorInput[i][e].channel = i;
         sensorInput[i][e].mode = true;
         sensorInput[i][e].linearization = true;
         sensorInput[i][e].enabled = true;
         sensorInput[i][e].playing = false;
      }
   }
}

void loop(){
   readData();
   readInputs();
}
```

Ukázka kódu 1. Funkce *setup()* a *loop()*

Funkce *loop()* obsahuje funkci *readData()*, obsluhující příjem SysEx zpráv, a read *Inputs()*, snímající hodnoty ze vstupů a odesílající MIDI kanálové zprávy. Tyto budou rozebrány dále.

2.3.1. Čtení hodnot ze snímačů a vysílání MIDI zpráv

Zabývejme se nejprve zpracováním dat ze snímačů a jejich odesláním pomocí MIDI protokolu. Samotná funkce *readInputs()*, je velmi jednoduchá. Jedná se opět o dvo-

jitou smyčku, která obsluhuje jednotlivé analogové a multiplexované vstupy, a v případě splněných podmínek odesílá příslušnou MIDI zprávu. Její funkce je patrná z vývojového diagramu, který je k dispozici v příloze B.1:

```
void readInputs(){
   for(int i=0;i<1;i++){
      if(analogInput[i].enabled == true){
         for(int e=0;e<analogInput[i].mux;e++){
            if(sensorInput[i][e].enabled == true){
               setControls(e);
               byte currentVelocity = sensorInput[i][e].setValue(analogRead(i));
               if(currentVelocity != 0){
                  sensorInput[i][e].sendMsg(currentVelocity);
               }
            }
         }
      }
   }
}
```

Ukázka kódu 2. Funkce *readInputs()*

Struktury analogInput zastupují jednotlivé analogové vstupy mikrokontroléru a obsahují informace o daném vstupu, konkrétně zda je vstup využíván (aktivní/enabled) a o velikosti multiplexeru, který je na daném vstupu připojen.

Oproti tomu, jak již název napovídá, struktury sensorInput zastupují jednotlivé multiplexované vstupy pro snímače. Každá z těch struktur obsahuje informace o MIDI element (notě nebo kontroléru): číslo elementu, hodnotu, která se mění v závislosti na hodnotě získané ze snímače, kanál na kterém element vysílá, a dále pak informace k jeho řízení: zda je element aktivní, zda je použita linearizace, a v jakém módu element pracuje. Tyto parametry je možno nastavit pomocí MIDI SysEx zpráv, jak bude popsáno v příštím oddílu.

Kromě nastavitelných parametrů obsahují struktury obslužné metody. Jsou to metody *setMux()*, ve strukturách typu aInput, a metody *setNumber(),setChannel(),se-*

tValue() a *sendMsg()* ve strukturách typu sInput. Metody *setMux()*, *setNumber()* a *setChannel()* pouze ukládají číselné hodnoty do proměnných ve struktuře a hlídají, zda číselná hodnota nepřekračuje povolený rozsah.

Metoda *setValue()*, oproti tomu, hodnotu pouze neukládá, ale převádí hodnotu sejmutou ze snímače na hodnotu v rozsahu protokolu MIDI. Zároveň pokud je na příslušném elementu nastavena linearizace je hodnota linearizována přiřazením hodnoty z vyhledávací tabulky _opti_linTable.

```
struct sInput{
   byte number;
   byte value;
   byte channel;
   boolean mode;
   boolean linearization;
   boolean enabled;
   boolean playing;
//------------------
   boolean setNumber(byte num){};
   boolean setChannel(byte chan){};
   int setValue(int val){};
   void sendMsg(int vel){};
} sensorInput[16]

struct aInput {
   byte mux;
   boolean enabled;
//------------------
   boolean setMux(byte mx){};
} analogInput[6];
```

Ukázka kódu 3. Struktury sInput a aInput

Metoda *sendMsg()* odesílá MIDI zprávu s hodnotami příslušného elementu, kdy záleží, zda je element v módu noty nebo kontroléru a v jakém stavu se element nachází. V případě, že je element v módu noty a jeho hodnota se změní z nuly na hodnotu vyšší (je stisknuta klávesa), což je indikováno stavem „element nehraje ale hodnota je vyšší než 0“, je vyslána zpráva NoteOn. V případě, že nota hraje a její hodnota je větší než nula je odeslána zpráva Aftertouch, značící přítlak na notu. A v případě, že

nota hraje, ale její hodnota klesne na nulu, je odeslána zpráva NoteOff. Navíc díky podmínce ve funkci readInputs, jsou zprávy odesílány, pouze pokud dojde ke změně rychlosti přeběhu. Vývojový diagram funkce *sendMsg()* je k dispozici v příloze B.2..

```
void readData(){
   while(Serial.available() > 0){
      incomingByte = Serial.read();
      if(ExpectData == true){
         if(incomingByte == 0xF7){
            ExpectData = false;
            SysExEnd = true;
         }else{
            SysExData[SysExDataEndPtr] = incomingByte;
            SysExDataEndPtr++;
         }
      }
      if(ExpectID == true){
         if(incomingByte == deviceID){
            ExpectData = true;
         }
      ExpectID = false;
      }
      if(ExpectManu == true){
         if(incomingByte == manuID){
            ExpectID = true;
         }
         ExpectManu = false;
      }
      if(incomingByte == 0xF0){
         ExpectManu = true;
         SysExDataEndPtr = 0;
      }
      if(SysExEnd == true){
         evaluateSysEx();
         SysExEnd = false;
      }
   }
}
```

Ukázka kódu 4. Funkce *readData()*

2.3.2. Příjem SysEx zpráv

Nyní prozkoumejme funkci *readData()* obstarávající příjem SysEx zpráv po sériovém rozhraní. Funkce nejprve kontroluje, zda jsou nějaká data ke čtení na sériovém vstupu, pokud nejsou, nic se neděje a program pokračuje dále v běhu. Pokud na sériovém vstupu jsou data, program vstoupí do smyčky, která probíhá tak dlouho doku nejsou přečtena všechna. V každém kroku smyčky je načten jeden byte dat, který je poté porovnáván. S čím je porovnáván, závisí na nastavených příznacích

Pokud žádný příznak nastaven není, je příchozí byte porovnáván s hodnotou 0xF0, což je dle MIDI specifikace hodnota označující začátek SysEx zprávy. Jakmile je přijat byte označující začátek zprávy SysEx je nastaven příznak začátku a příznak očekávání ID zařízení. V případě, že následující byte nemá hodnotu odpovídající 0x7D (Tato hodnota je dána MIDI specifikací pro experimentální a studijní zařízení.), je příjem zprávy přerušen a všechny příznaky nastaveny na výchozí hodnotu. Pokud je ID správné a tím je ověřeno, že zpráva je pro toto zařízení, je příznak očekávání ID zrušen a nastaven příznak očekávání dat. Při nastaveném příznaku očekávání dat je přijímaný byte porovnáván s hodnotou 0xF7 označující konec SysEx zprávy, pokud není podmínce vyhověno je byte uložen do bufferu pro příchozí zprávu. Jakmile je přijat konec SysEx zprávy, jsou všechna data v bufferu vyhodnocena dle níže popsaného protokolu a všechny příznaky uvedeny do původního stavu. Funkce je opět vysvětlena pomocí vývojového diagramu, který je k nalezení v příloze B.3..

2.3.3. Protokol pro přijímání SysEx zpráv

Při návrhu protokolu je hlavním aspektem, typ zpráv které budou pomocí protokolu přenášeny, kolik budou mít tyto zprávy parametrů a jakých budou tyto nabývat hodnot. V případě tohoto projektu jsou zprávy vztaženy k jednotlivým analogovým, nebo multiplexovaným vstupům. Tento „podprotokol“ je součástí protokolu MIDI, a proto nesmí jednotlivé prvky odporovat jeho specifikaci.

Pro analogové vstupy jsou přenášeny tyto zprávy:

» aktivovat daný analogový vstup
» deaktivovat daný analogový vstup
» nastavit počet kanálů multiplexeru na daném vstupu

Pro senzorové vstupy tyto:

» aktivovat daný senzorový vstup
» deaktivovat daný senzorový vstup
» nastavit číslo MIDI elementu
» nastavit kanál, na kterém MIDI element vysílá
» nastavit mód MIDI elementu
» nastavit linearizaci elementu

Nejjednodušším postupem je nastavit jednotlivým typům zpráv unikátní identifikátor, podle něhož budou příchozí zprávy třízeny a na základě tohoto vyhodnoceny parametry. Následující tabulka ukazuje strukturu protokolu s uvedenými rozsahy jednotlivých parametrů.

0xF0	0x7D	0x00	typ zprávy	analog ID	sensor ID	param. 3	0xF7
aktivuj analogový vstup			0x00	0x00 - 0x05	-	-	-
deaktivuj analogový vstup			0x01	0x00 - 0x05	-	-	-
nastav počet mux. na kanálu			0x02	0x00 - 0x05	0x00 - 0x7F	-	-
aktivuj senzorový vstup			0x03	0x00 - 0x05	0x00 - 0x7F	-	-
deaktivuj sensorový vsttup			0x04	0x00 - 0x05	0x00 - 0x7F	-	-
nastav číslo prvku			0x05	0x00 - 0x05	0x00 - 0x7F	0x00 - 0x7F	-
nastav kanál			0x06	0x00 - 0x05	0x00 - 0x7F	0x00 - 0x0F	-
nastav mód			0x07	0x00 - 0x05	0x00 - 0x7F	0x00 - 0x01	-
nastav linearizaci			0x08	0x00 - 0x05	0x00 - 0x7F	0x00 - 0x01	-

Tab. 4. Struktura pod-protokolu příjmu SysEx zpráv

Možností, jak přenášet zprávy, by bylo poslat identifikátor typu a každý z parametrů jako samostatný byte. Toto řešení je ovšem poněkud neefektivní, vzhledem k množství přenášených zbytečných informací, a je vhodné tento proces nějak zefektivnit. Nyní je také vhodné si uvědomit omezení, které klade na přenášené zprávy protokol MIDI: zpráva je přenášena na pozici data bytů a proto všechny byty zprávy musí respektovat požedavky – nejvýznačnější bit musí být roven logické nule.

Ze struktury protokolu je zřetelné, že je zapotřebí přenášet celkem 9 typů zpráv, pro jejich rozlišení jsou tedy zapotřebí 4 bity (kdy ještě zbývá 6 pozic na potenciální rozšíření v budoucnu). Zároveň pokud se podíváme na strukturu protokolu, všimneme si, že všechny typy zpráv mají společný první parametr, čímž je číslo analogového vstupu, kterého se zpráva týká. Vzhledem k tomu, že mikrokontrolér má pouze 6 vstupů, má tento parametr velmi malý rozsah – konkrétně 3 bity. Je proto vhodné, spojit identifikátor zprávy a identifikátor analogového vstupu do jednoho bytu. Celkem tedy jsou SysEx zprávy maximálně 7 bytové.

Dalším místem, kde je přenášena zbytečná informace, jsou zprávy nastavující linearizaci a mód elementu. Čtvrté parametry těchto zpráv, nabývají pouze dvou hodnot, ale přenášeny jsou v samostatném bytu. Toto je zanecháno z důvodu plánovaných rozšíření v budoucnu, kdy bude projekt pravděpodobně rozšířen i o jiné snímače a bude potřeba rozšířit počet linearizací.

3. ZÁVĚR

Cílem práce bylo navrhnout a zrealizovat způsob řízení hudební MIDI elektroniky pomocí snímačů, které nejsou běžně k tomuto účelu využívány. Výsledkem této práce je funkční prototyp a jeho návrh, který byl rozebrán v kapitole 2. Z celkového pohledu je tato práce velmi komplexní a řeší daný problém přes všechny úrovně návrhu - od návrhu snímače, přes řešení obvodů, až po program ve výsledném zařízení. V průběhu práce byly zjištěny některé nepříznivé vlastnosti a parametry v navrhovaném zařízení, které, vzhledem k nedostatku času, nemohly být odstraněny. Pro elegantní odstranění některých by navíc musela být vyměněna technologie, která je obsažena v zadání práce. Je tedy možné, že by po jejich odstranění práce zadání nesplňovala. Je ale vhodné tyto nedostatky nyní rozebrat a nastínit jejich řešení.

Prvním z nich je menší počet multiplexovaných kanálů na jednom analogovém vstupu, než byla původní vize. Tento problém je dán velikostí SRAM paměti v procesoru ATmega168. Tato je pouze 1 KB, je sdílena všemi proměnnými, se kterými se v průběhu chodu programu pracuje, a tudíž se do ní nevejde dostatečný počet struktur využitých pro řízení. V současném stavu je velikost jedné struktury 24 b a její zmenšení, pro dosažení funkčnosti 128 kanálového multiplexu na 6-ti vstupech, je dle mého pohledu nereálné. Jednoduchým řešením je tedy výměna procesoru nebo modulu za jiný, například modul Arduino Mega a procesor ATmega1280, který má již paměti 8 KB. Vzhledem k tomu, že jedním z hlavních kritérií projektu je flexibilita, stačí pouze modul vyměnit a v programu změnit několik proměnných a bude dosaženo kýžené funkčnosti. Díky připojení modulovému návrhu obvodů pro zpra-

cování a jejich připojení pomocí „štítu“ stačí doslova modul Arduino Diecimila za Arduino Mega vyměnit a pomocí vhodné SysEx zprávy zvětšit počet multiplexovaných kanálů. Nicméně je zapotřebí vytvořit nový modul pro zpracování obsahující multiplexer s větším počtem kanálů. Tento multiplexer by měl být také rychlejší a stabilnější než který byl použit ve vytvořeném prototypu.

Dalším momentálním nedostatkem je ovládací program, který by zařízení pomocí SysEx zpráv nastavil. Tento program v rámci práce implementován nebyl a zatím je tedy nutné veškerá nastavení provádět „ručně“ hexa kódem. Ideálním prostředím pro vývoj takového programu je prostředí Processing, které je sesterským projektem k projektu Arduino.

Posledním nedostatkem jsou navržené snímače váhy. Tato část není součástí zadání a nejedná se tedy o závažnou závadu zařízení, stačí pouze použít jiné snímače, například již zmíněné FSR snímače od společnosti Interlink Technicss. Nicméně popis těchto snímačů je součástí práce a je proto vhodné zde tento nedostatek uvést. Problémem je značná nestabilita těchto snímačů. Toto je dáno materiálem. Volba pěny, která byla pro jejich výrobu použita, se ukázala jako nevhodná, protože tento materiál má poměrně velkou paměť. Snímač se tedy po zatížení nevrátí do původního stavu. Stav, do něhož se snímač navrací, je závislý na celkovém opotřebení. Pokud jsou snímače nově vyrobené, mohou být jejich charakteristiky odporu v závislosti na zatížení značně odlišné. Na druhou stranu se ale u tohoto projevuje určitá hranice opotřebení, po jejímž překročení se začnou snímače rychle konvergovat do stavu s podobnou charakteristikou. Nicméně stále se jistá paměť uplatňuje. Proto by bylo o mnoho vhodnější použít jako materiál vhodně dotovanou a tuhou pryž, u které byl projev paměti minimální. Ačkoliv nejsou současné snímače ideální, jsou použitelné v aplikacích, kdy není vyžadována přesnost, například interaktivní hry pro děti a jiné aplikace zábavního charakteru.

Ačkoliv je výsledkem práce prototyp, který jak je výše popsáno, trpí jistými nedostatky, nemyslím si, že by tytobyly známkou neúspěchu, naopak bych práci charakterizoval jako úspěšnou a velmi inspirativní a tyto nedostatky pouze jako výsledky práce které ukazují směr kterým pokračovat. Tato diplomová práce začala jako osobní projekt, vývoj zařízení po skončení práce neustává a výše zmíněné nedostatky budou vylepšeny v dalších verzích zařízení.

4. POUŽITÁ LITERATURA

[1] MIDI Manufacturers Association, *The Complete MIDI 1.0 Detailed Specification*. Tech. Rep., MMA, 1996, www.midi.org.

[2] Schimmel, J. *Komunikační rozhraní MIDI*. *Electrorevue*. [online] 2002/69 - 20.12.2002. www.elektrorevue.cz

[3] BANZI,M. *Getting started with Arduino* [online]. www.arduino.cc

[4] Future Technology Devices International Ltd. *AN232B-05 Configuring FT232R, FT2232 and FT232B Baud Rates*. 2006. Dostupný z www.ftdi.com

[5] Atmel Corporation : *ATmega48/88/168 Datasheet* [online]. 2007. Last updated 09/2007. Dostupný z www.atmel.com

[6] Vrba, K. Lattenberg, I. Matějíček, L. : *Analogová technika* Skriptum FEKT VUT

[7] Sharp Corp. *GP2Y0A21YK0F datasheet*. Sheet No.: E4-A00201EN .2006. ©SHARP Corporation

[8] Beneš, P. : *Inteligentní snímače*. FEKT VUT UAMT.

5. SEZNAM PŘÍLOH

A. Navržené obvody 66
A.1. Seznam použitých součástek 66
A.2. Zapojení přizpůsobovacího modulu 67
A.3. Zapojení rozšiřovacího štítu 68
A.4. Rozložení prvků na DPS 69

B. Vývojové diagramy 70
B.1. Vývojový diagram funkce *readInputs()* *70*
B.2. Vývojový diagram funkce *sendMsg()* *71*
B.3. Vývojový diagram funkce *readData()* *72*

C. Dokumentace programu 73
C.1. Struktura sInput 73
C.2. Struktura aInput 75
C.3. Ostatní 76

A. Navržené obvody

A.1. Seznam použitých součástek

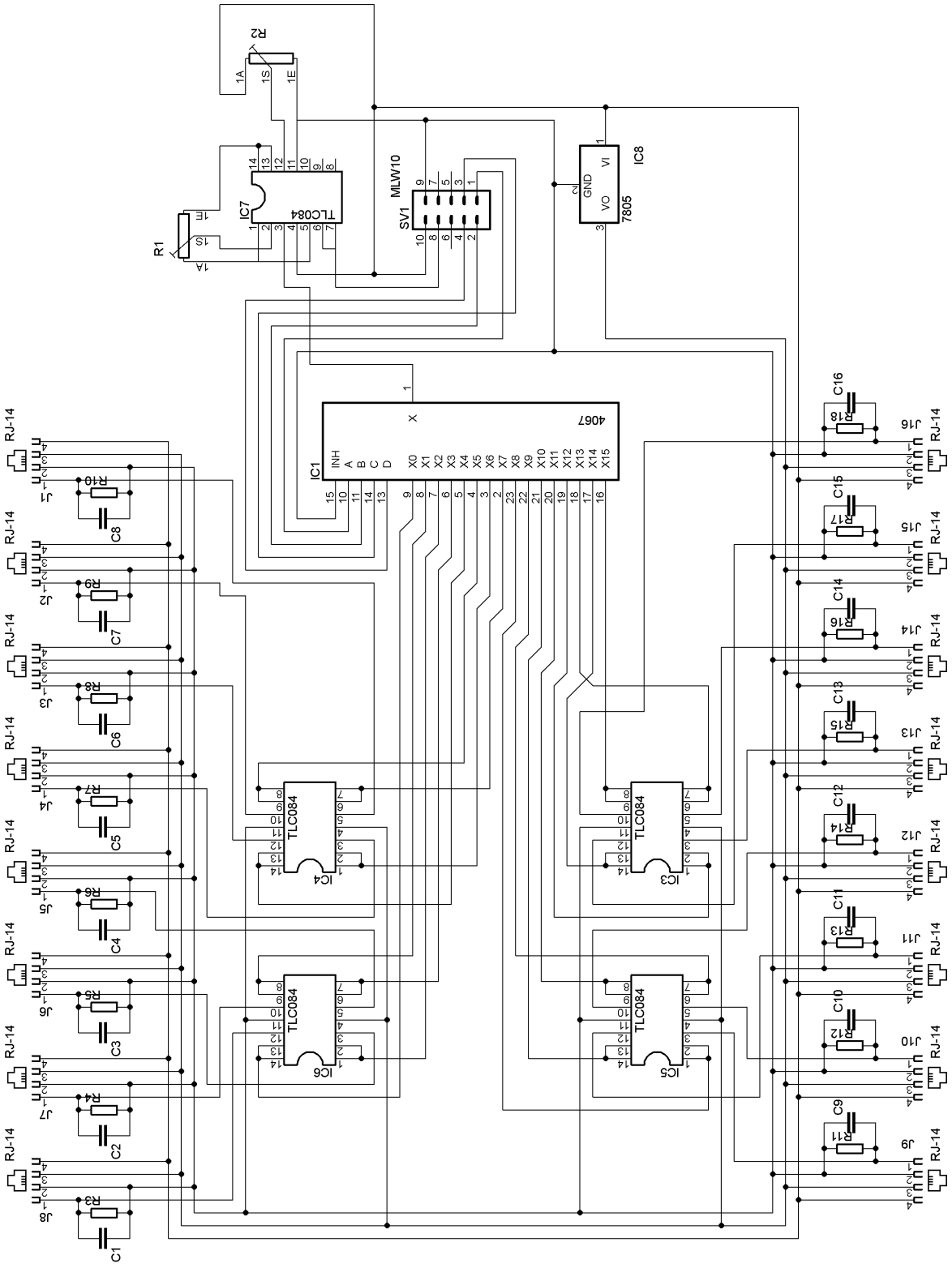
Snímače váhy:
Elektrovodivá pěna QCM 900
Cuprexitové desky plošných spojů

Snímače vzdálenosti:
SHARP GP2Y0A21YK0F

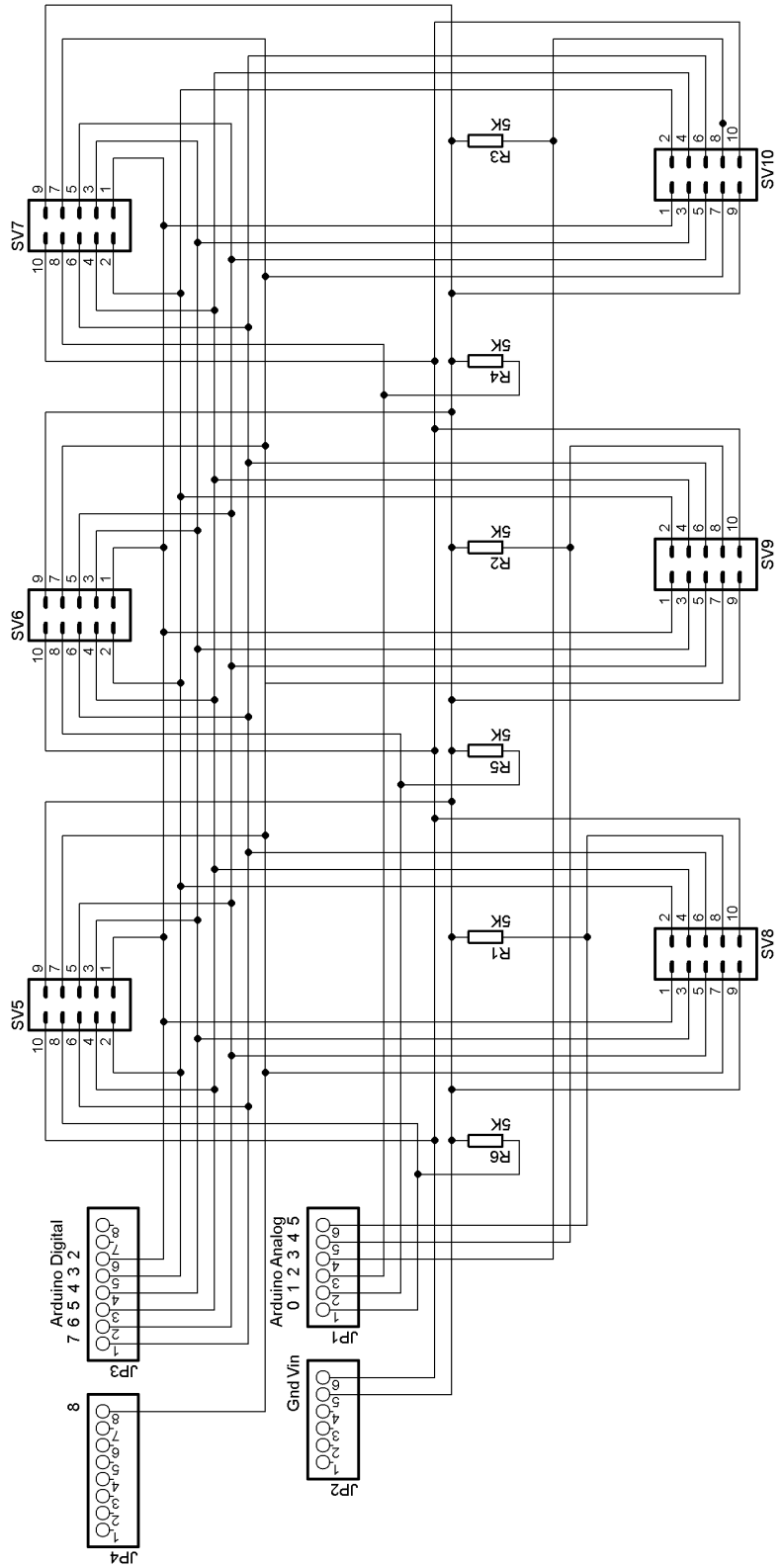
Modul přizpůsobovacích obvodů:
Konektor RJ-14 16x
Konektor MLW10 1x
Čtřnásobný operační zesilovač TLC 084 5x
Multiplexer HCF 4067 1x
Odpor 10 MΩ 16x
Kondenzátor 100 nF 16x
Regulátor napětí 7805 1x
Potenciometr PC1221NK001 2x
Deska plošného spoje

Rozšiřující štít na modul Arduino:
Konektor MLW10 6x
Oboustraná header dutinka 22x
Odpor 5 kΩ6x
Deska plošného spoje

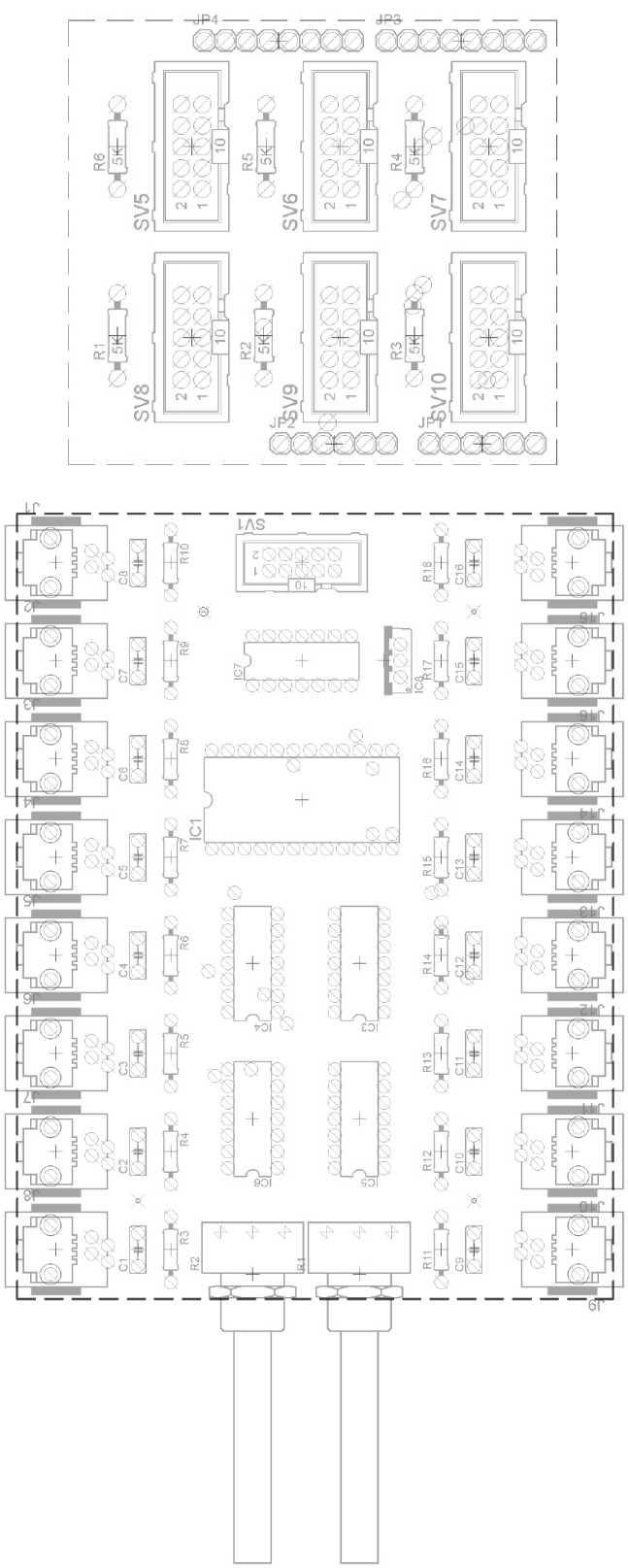
A.2. Zapojení přizpůsobovacího modulu



A.3. Zapojení rozšiřovacího štítu

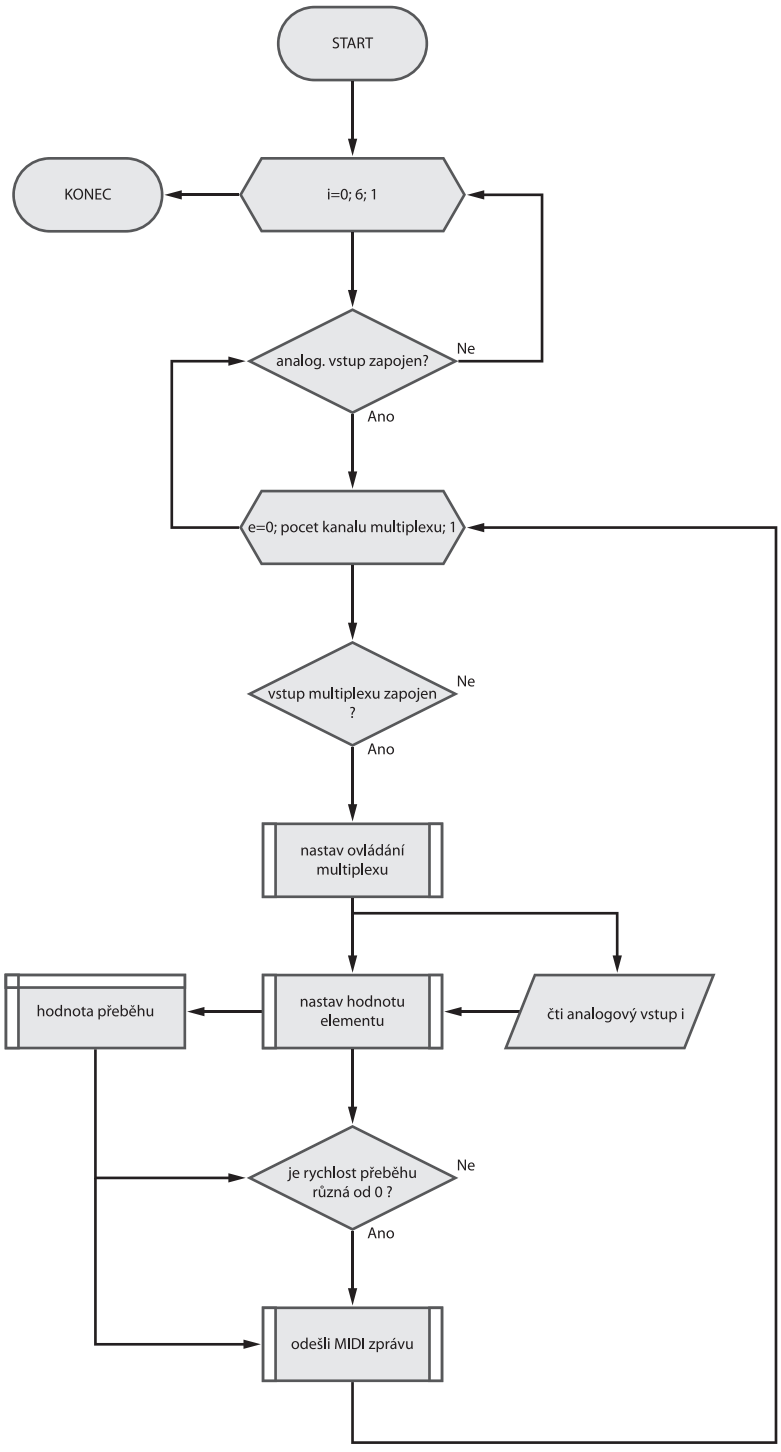


A.4. Rozložení prvků na DPS

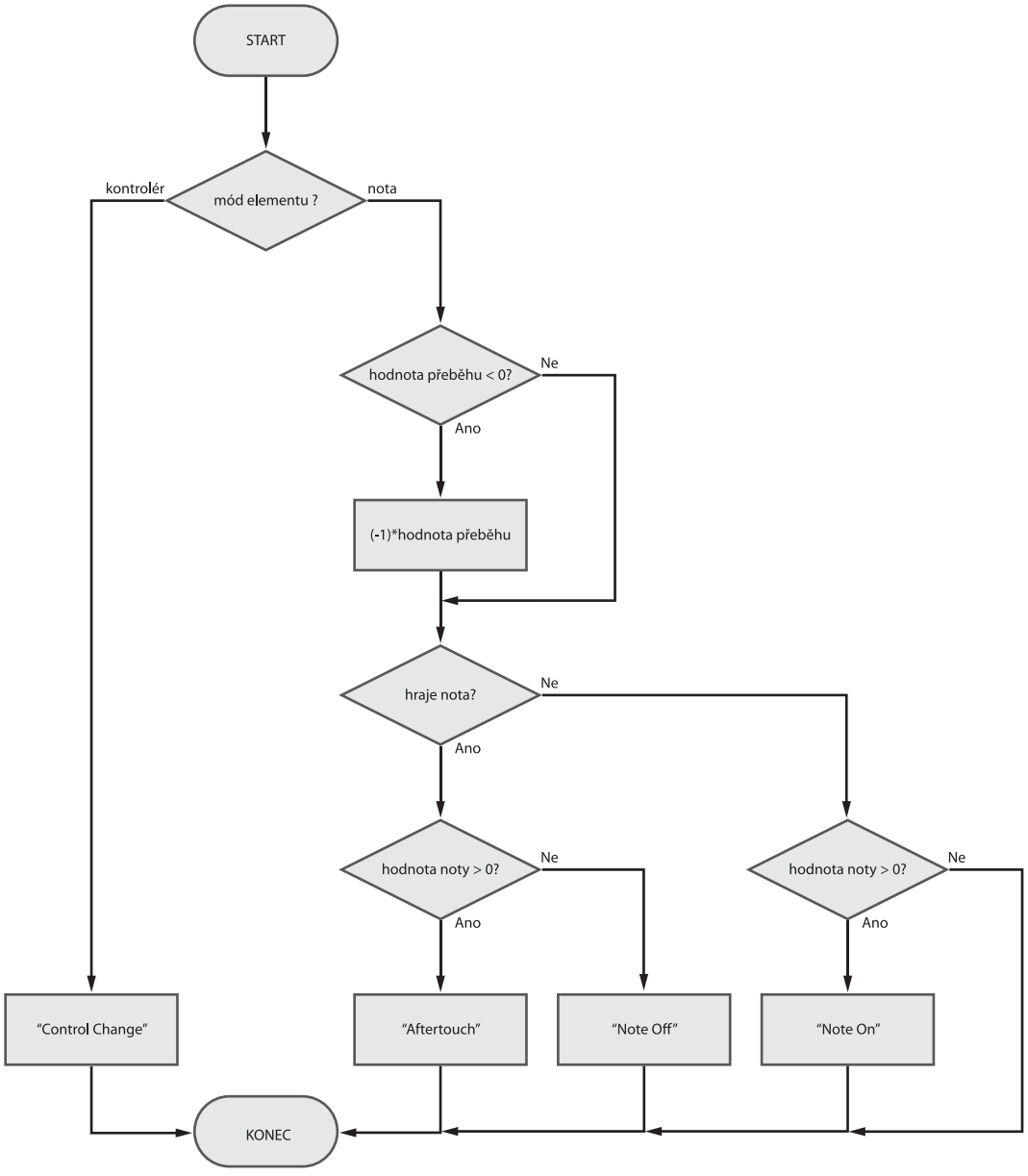


B. Vývojové diagramy

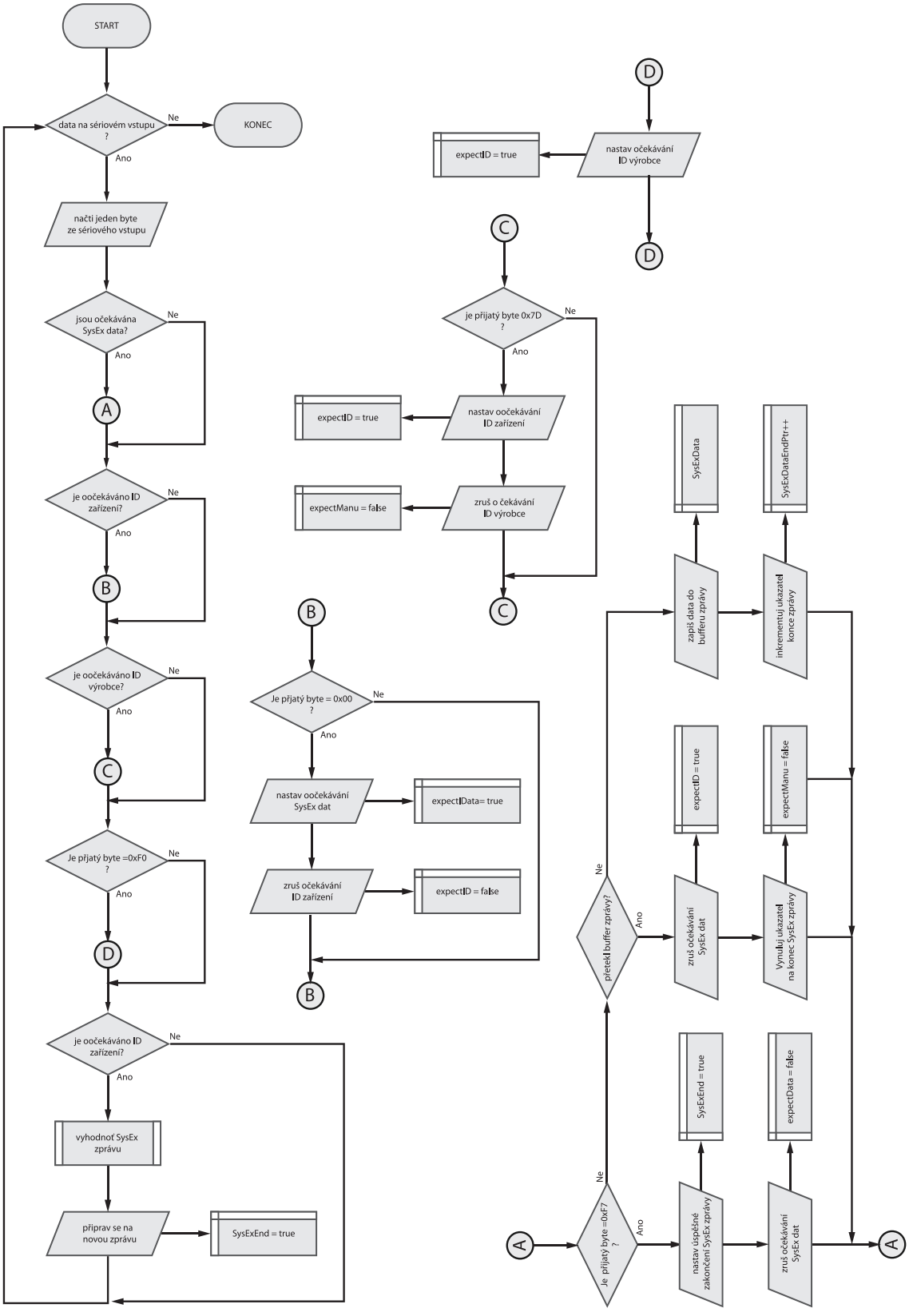
B.1. Vývojový diagram funkce *readInputs()*



B.2. Vývojový diagram funkce *sendMsg()*



B.3. Vývojový diagram funkce *readData()*



C. Dokumentace programu

C.1. Struktura sInput

Veřejné metody	
boolean	setNumber (byte numb)
boolean	setChannel (byte chan)
int	setValue (int val)
void	sendMsg (int velocity)
Datové položky	
byte	number
byte	value
byte	channel
boolean	mode
boolean	linearization
boolean	enabled
boolean	playing

Detailní popis

Jde o prakdicky nejdůležitější strukturu v celém programu a reprezentuje jednotlivé MIDI elementy (noty nebo controlery). Obsahuje proměnné určující stav a vlastnosti elementu a několik obslužných funkcí. Ihned po definici je vytvořeno pole těchto struktur v maximálním možném rozsahu.

Dokumentace k metodám

void sInput::sendMsg(int velocity)

Odesílá MIDI zprávu v závyslosti na stavu a vlastnostech elementu. Pokud je element v módu noty, jsou rozlišeny a ošetřeny tři stavy: nota nehraje a je sepnuta, nota hraje a je sepnuta, nota hraje a není sepnuta. V případě, že nota nehraje a je sepnuta je odeslána MIDI zpráva NoteOn s příslušným číslem a rychlostí přeběhu na příslušném kanálu. V příladě, že nota hraje a je sepnuta je odeslána zpráva Aftertouch značící přítlak na klávesu. A v případě, že nota hraje a není sepnuta je nota vypnuta zprávou NoteOff. Pokud je element v módu controleru je odeslána jeho hodnota.

boolean sInput::setChannel (byte chan)

Přiřazuje číslo kanálu na němž element vysílá. Dle MIDI specifikace musí být číslo maximálně 0x7F(127), v případě, že je číslo vyšší, není přiřazeno a funkce skončí s návratovou hodnotou false.

boolean sInput::setNumber (byte numb)

Přiřazuje číslo elementu. Dle MIDI specifikace musí být číslo maximálně 0x7F(128), proto v případě že je číslo vyšší, není přiřazeno a funkce skončí s návratovou hodnotou false.

boolean sInput::setValue (int val)

Přiřazuje hodnotu elementu a zároveň vypočte rychlost přeběhu, kterou po svém dokončení navrátí. Hodnota je přiřazena v závislosti zda je vstup linearizován, pokud je, hodnota je vybrána z linearizační tabulky, pokud není, je vstupní hodnota vydělena 8 čímž je dosaženo rozsahu dle MIDI specifikace (1024/8 = 128).

Dokumentace k položkám

byte sInput::channel

Proměnná určující na jakém kanálu element vysílá.

boolean sInput::enabled

Flag signalizující zda je element použit.

boolean sInput::linearization

Proměná určující zda je pro tento element použita linearizace. Momentálně dvoustavová při použití více typů linearizací možno rozšířit. Linearizace je použita v příladě hodnoty true.

boolean sInput::mode

Proměná určující mód v němž element pracuje. Momentálně typu bolean z důvodu úspory paměti. Element je v režimu noty v případě hodnoty true a v režimu controleru v případě hodnoty false.

byte sInput::number

Proměná určující číslo elementu. V závislosti na nastavení proměnné mode jde buď o číslo noty nebo controleru.

boolean sInput::playing

Flag signalizující zda element vysílá/hraje.

byte sInput::value

Proměnná určující hodnotu elementu. Popřevedení jsou zde ukládány hodnoty sejmuté ze snímačů

C.2. Struktura aInput

Veřejné metody	
boolean	setMux (byte mx)
Datové položky	
byte	mux
boolean	enabled

Detailní popis

Podpůrná struktura nastavující parametry analogových vstupů mikroprocesoru. Id-hed po definici je vytvoeno pole 6-ti těchto struktur.

Dokumentace k metodám

boolean aInput::setMux(byte mx)

Nastavuje počet kanálů multiplexeru na vstupu. Opět do rozsahu 0x7F(127), pokud je hodnota vyšší funkce vrací hodnotu false.

Dokumentace k položkám

boolean aInput::enabled

Flag signalizující zda je vstup použit.

byte aInput::mux

Proměnná určující počet kanálů multiplexeru na danném vstupu.

C.3. Ostatní

Datové struktury	
struct	sInput
struct	aInput
Definice maker	
#define	DEFAULT_ID 0x7D
#define	BUFFER_LENGHT 3
#define	DEFAULT_MUX 16
Funkce	
void	setup ()
void	loop ()
void	readInputs ()
void	readData ()
void	evaluateSysEx ()
void	setControls (int i)
void	sendMIDI (byte status, byte data1, byte data2)
Proměnné	
byte	deviceID = DEFAULT_ID
byte	incomingByte = 0
byte	SysExData [BUFFER_LENGHT]
int	SysExDataEndPtr = 0
boolean	ExpectManu = false
boolean	ExpectID = false
boolean	ExpectData = false
boolean	SysExEnd = false
int	S [] = {2,3,4,5,6,7,8}
PROGMEM prog_uchar	_opti_linTable [1024]
struct sInput	sensorInput [6][16]
struct aInput	analogInput [6]

Dokumentace k metodám

void setup()

Inicializační funkce programu. Jsou inicializovány výstupní piny pro řízení multiplexerů a výše vytvořená pole struktur sensorInput a analogInput.

void loop()

Hlavní funkce programu, probíhá v nekonečné smyčce.

void evaluateSysEx()

Vyhodnocuje přijatou SysEx zprávu podle navrženého protokolu. Nejprve kontroluje typ zprávy a poté podle obsažených dat nastavuje příslušné parametry.

void readData()

Čte data přicházející po sériovém rozhraní. Jediná očekávaná data jsou přicházející SysEx zprávy se správným ID zařízení. Ostatní jsou zahozena

void readInput()

Čte data ze snímačů a v případě splnění parametrů odesílá MIDI zprávu. Snímače prochazí pomocí dvojité smyčky - nejprve kontroluje zda je analogový vstup aktivní, poté zda jsou aktivní jednotlivé vstupy snímačů. Poté nastaví ovládací signály na příslušnou úroveň a přečte hodnotu ze snímače a pokud je rozdílná oproti předešlé pošle MIDI zprávu

void sendMIDI (byte status, byte data1, byte data2)

Ze tří vstupních argumentů sestaví MIDI zprávu a tu odešle po seériovém rozhraní.

void setControls (int i)

Podle vstupního argumentu zapisuje hodnoty na řídící piny multiplexerů.

Dokumentace k proměnným

PROGMEM prog_uchar _opti_linTable[1024]

Look-up tabulka sloužící pro linearizaci hodnot přijatých z optických snímač

struct aInput analogInput[6]

Podpůrná struktura nastavující parametry analogových vstupů mikroprocesoru. Idhed po definici je vytvoeno pole 6-ti těchto struktur.

byte deviceID = DEFAULT_ID

Identifikace zařízení.

boolean ExpectData = false

Flag ukazující zda je očekáván příjem dat

boolean ExpectID = false

Flag ukazující zda je očekávána identifikace zařízení

byte incomingByte = 0

Proměnná pro uložení přijímaného bajtu

int S[] = {2,3,4,5,6,7,8}

Pole přiřazující fyzické výstupní piny pro ovládání multiplexerů

struct sInput sensorInput[6][16]

Struktura sInput.

Jde o prakdicky nejdůležitější strukturu v celém programu a reprezentuje jednotlivé MIDI elementy (noty nebo controlery). Obsahuje proměnné určující stav a vlastnosti elementu a několik obslužných funkcí. Ihned po definici je vytvořeno pole těchto struktur v maximálním možném rozsahu.

byte SysExData[BUFFER_LENGHT]

Buffer pro přijímaná SysEx data. Délka podle BUFFER_LENGHT

int SysExDataEndPtr = 0

Ukazatel na konec bufferu

boolean SysExEnd = false

Flag ukazující zda byla SysEx zpráva zakončena