VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

MĚŘENÍ A VIZUALIZACE DAT ZE SENZORU VZDÁLENOSTI V REÁLNÉM ČASE ARDUINO A XBEE KOMUNIKACI V PROSTŘEDÍ MATLAB

MEASURING AND VISUALIZING DISTANCE SENSOR DATA IN REAL-TIME VIA ARDUINO AND XBEE WIRELESS COMMUNICATION TO MATLAB

BAKALÁŘSKÁ PRÁCE

BACHELOR´S THESIS

AUTOR PRÁCE — MAREK ZAPLETAL
AUTHOR

VEDOUCÍ PRÁCE — ENG. MOURAD KARAKHALIL
SUPERVISOR

BRNO 2014

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2013/2014

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Marek Zapletal

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

obor: **Strojní inženýrství (2301R016)**

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

Měření a vizualizace dat ze senzoru vzdálenosti v reálném čase Arduino a XBee komunikaci v prostředí MATLAB

v anglickém jazyce:

Measuring and Visualizing Distance Sensor Data in Real-Time via Arduino and Xbee wireless communication to Matlab

Stručná charakteristika problematiky úkolu:

Práce se bude zabývat návrhem měření vzdálenosti a vizualizací v reálném čase pomocí
-senzoru pro měření vzdálenosti
-bezdrátové komunikace XBEE
-vývojové desky Arduino
-vizualizace v prostředí MATLAB.

Cíle bakalářské práce:

1. Zapojení okruhu dle schématu. (Arduino + senzor HC-S04 + moduly Xbee)
2. Měření vzdálenosti pomocí senzoru.
3. Odeslání dat přes Xbee k zpracování do PC.
4. Zpracování dat pomocí programu MATLAB.
5. Návrh praktického využití bakalářské práce.

Seznam odborné literatury:

1. OXER, Jonathan. Practical Arduino: cool projects for open source hardware. Berkeley, CA: Apress, c2009. ISBN 978-1-4302-2477-8.

2. FALUDI, Robert. Building wireless sensor networks. 1st ed. Beijing: O´Reilly, 2010. ISBN 978-0-596-80773-3.

3. MONK, Simon. 30 Arduino projects for the evil genius. New York: McGraw-Hill, c2010, xiii, 191 p. ISBN 00-717-4133-X.

4. BANZI, Massimo. Getting started with Arduino. 2nd ed. Farnham: O'Reilly, 2011. ISBN 978-144-9309-879.

5. KNIGHT, Andrew. Basics of MATLAB and beyond. Boca Raton, FL: Chapman, c2000, 202 p. ISBN 08-493-2039-9.

Vedoucí bakalářské práce: Mourad Karakhalil

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2013/2014.

V Brně, dne 25.11.2013

L.S.

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

Abstrakt

Práce se zabývá tvorbou měřícího obvodu, který se skládá z ultrazvukového senzoru, programovatelné desky Arduino, v němž bezdrátový přenos dat do počítače s programem MATLAB zajišťují moduly Xbee.

V teoretické části byly popsány jednotlivé komponenty obvodu včetně nutného programového vybavení, praktická část byla věnována vlastní realizaci obvodu z daných součástek a současně tvorbě programů. Obvod byl fyzicky sestaven a získaná data ze senzoru vzdálenosti byla v reálném čase přenesena a vykreslována v počítači vybaveném programem MATLAB, kde jsou jednoduchým filtrem dat zkorigována a filtrovaná data jsou vynesena do grafů závislosti vzdálenosti na čase.

Tato práce je jedním z možných prvních kroků k tvorbě inteligentních strojů či robotů obecně z hlediska mapování prostoru anebo jejich navádění a orientace v něm.

Abstract

Work describes making of measuring system consisting of ultrasonic sensor, programmable board Arduino, where the wireless communication is provided by Xbee modules.

Theoretical part depicts the single components including the programmes for making the code necessary for proper function of the system. On the other hand, realization and creation of programmes is the centre of practical part of the work. System was physically made and the obtained data from distance sensor were in real-time transferred and depicted in MATLAB programme, where the data are filtered and the graphs of elapsed time- distance are drawn.

This work is one of the first possible steps in making intelligent devices or robots from the general aspect of mapping, navigating and orienting in a surrounding area.

Klíčová slova

Arduino, ultrazvuk, měření vzdálenosti, MATLAB, Xbee

Key words

Arduino, ultrasound, distance measuring, MATLAB, Xbee

Prohlášení o originalitě

Prohlašuji, že bakalářskou práci jsem vypracoval samostatně pod vedením Eng. Mourada Karakhalila a za použití uvedených zdrojů.

V Brně dne 20.5.2014 Marek Zapletal

Bibliografická citace

ZAPLETAL, M. *Měření a vizualizace dat ze senzoru vzdálenosti v reálném čase Arduino a XBee komunikaci v prostředí MATLAB*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2014. XY s. Vedoucí bakalářské práce Mourad Karakhalil.

Poděkování

Chtěl bych poděkovat vedoucímu bakalářské práce Eng. Mouradu Karakhalilovi za jeho pomoc při vypracování a za jeho ochotu a vstřícnost dalece překračující jeho povinnosti. Dále bych chtěl poděkovat rodině, přátelům a všem těm, kteří mě podporují.

Obsah

Úvod

Od nepaměti je lidské pokolení hnáno vpřed touhou ulehčit si práci. V pravěku začal člověk užívat první kamenné nástroje, aby si snáze získal potravu, později za dob průmyslové revoluce začal ve velkém používat stroje, aby zefektivnil a zkvalitnil výrobu.

Nyní se průmyslová výroba více a více spoléhá na práci těchto strojů, ale ani ty nezůstaly nezměněny a prošly vývojem. V dnešní době se více než kdy dřív ve výrobním procesu prosazují stroje s prvky inteligentního chování, které jsou schopny pružně reagovat na podněty vnějšího okolí. K této interakci by nedošlo bez použití různých typů senzorů, které umožňují, toto okolí vnímat. Právě kvalitní „vnímání“ je základem dobré a spolehlivé práce celého zařízení a přesné vyhodnocení podnětu je alfou a omegou při návrhu automaticky pracujících systémů.

Zaměření práce jasně vyplývá ze všech těchto skutečností. Skvělým příkladem je návrh a realizace obvodu s využitím programovatelné desky Arduino a ultrazvukového senzoru k měření vzdálenosti s využitím bezdrátového spojení za pomocí rádiových modulů Xbee tak, aby bylo dosaženo co největší použitelnosti a flexibility zejména v uspořádání měřící obvod - výpočetní středisko (počítač). Zpracování získaných dat je prováděno v prostředí programu MATLAB.

Teoretická část

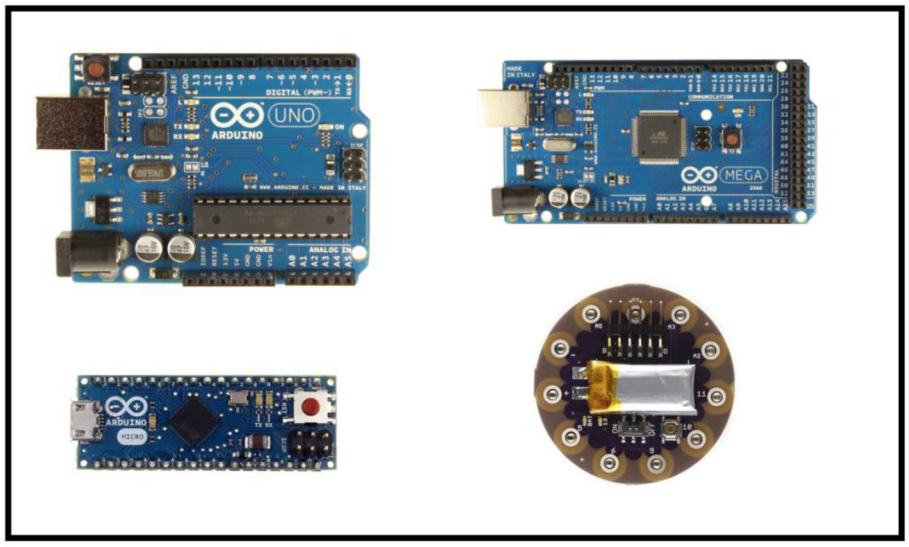
1 Arduino

1.1 Představení projektu Arduino

Arduino je open-source (výsledky prací jednotlivých uživatelů jsou sdílené a volně dostupné) elektronická programovatelná platforma založená na jednoduchém vstup/výstup systému, která zahrnuje programovací prostředí vycházející z prostředí Processing. [1, 2, 3, 4,5]

Nejdůležitější částí desky Arduino je mikrokontrolér značky Atmel, obvykle z řady ATmega. Začátek úspěšného projektu se dá datovat od roku 2004, kdy při jeho vzniku stáli zakladatelé Hernando Barragan, Massimo Banzi, David Cuartielles a nemnoho dalších.[1, 5]

Deska je primárně určena pro netechnickou veřejnost, která je jejím prostřednictvím schopna vytvářet interaktivní objekty, aniž by nutně musela mít detailní znalosti z oboru elektrotechniky či programování. Deska je již sestavená a připravena pro okamžité použití bez nutnosti cokoliv pájet. Na obrázku 1 je několik základních provedení desek Arduino. [1, 5]

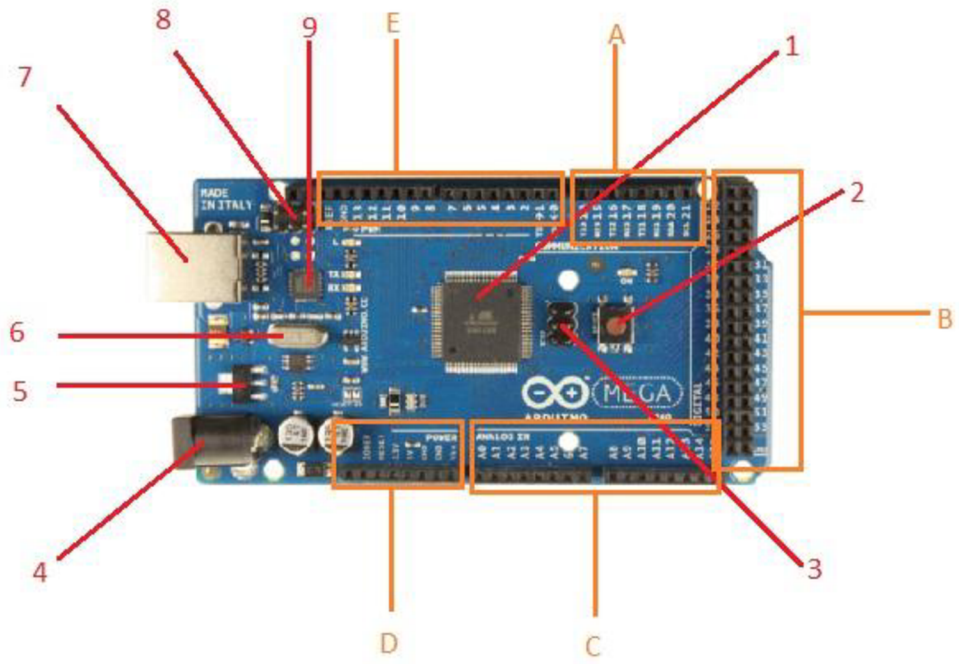


Obr. 1: Základní typy desek Arduino:
a) Uno b) Mega c) Micro
d) LilyPad (kruhovitý tvar) [6-upraveno].

1.2 Hardware desky

1.2.1 Zevrubný popis vnějších částí

Základní popis nejdůležitějších komponent desky MEGA je na obrázku 2. Detailní popis není předmětem této práce, avšak je snadno dohádatelný online, či v odborných publikacích. Tzv. pinout a podobné zapozicování, jako na obrázku 2, se nachází v příloze A.



Obr. 2: Umístění komponent na desce a jejich zapozicování.) [6-upraveno].

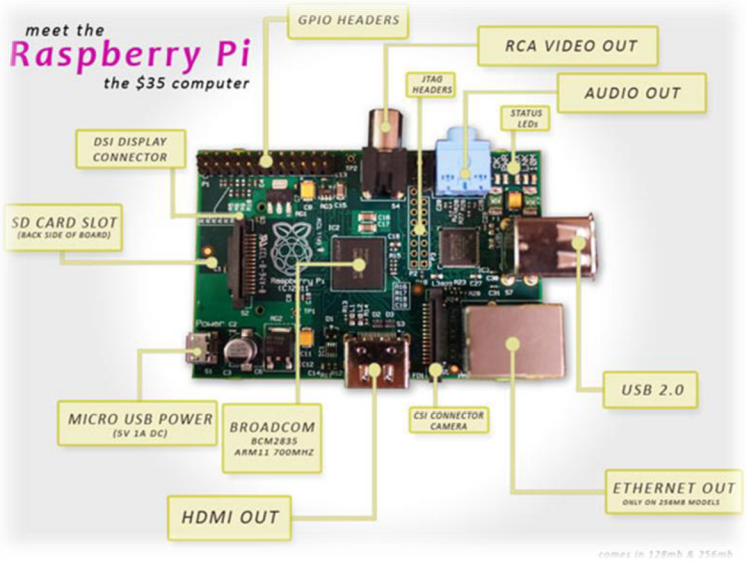
Vysvětlivky k obrázku 2:
1 - Mikrokontrolér
2 - Tlačítko RESET
3 - ICSP komunikace mikrokontroléru
4 - Napájecí jack
5 - Regulátor napětí
6 - 16 MHz krystal
7 - USB jack
8 - ICSP komunikace čipu zajišťující přenos dat
9 - Komunikační čip
A - Komunikační piny
B - Digitální piny
C - Analogové vstupní piny
D - Napájecí (power) piny
E - Digitální piny umožňující regulaci délky pulsů (PWM)

1.2.2 Mikroprocesor, mikropočítač, mikrokontrolér

Středobodem desky, jak již bylo zmíněno, je mikrokontrolér výrobce Atmel. Slovo mikrokontrolér je často zaměňováno s výrazy jako mikroprocesor a mikropočítač (microcomputer), ačkoliv se významově liší.

Mikroprocesor - je procesor (CPU) z jediného čipu. Dříve byly procesory složeny z více integrovaných čipů. Průkopníkem v této oblasti je společnost Intel se svým procesorem 4004, jehož všechny části byly vloženy na jediný čip, čímž se stal prvním mikroprocesorem.

Mikropočítač - spojením mikroprocesoru, periferních vstupních a výstupních zařízení a paměti vznikne mikropočítač. Na obrázku 3 je populární mikropočítač Raspberry Pi. [7]



Obr. 3: Mikropočítač Raspberry Pi s popisem. [8]

Mikrokontrolér - vložením všech komponent mikropočítače na jediný čip získáme mikrokontrolér. S prvním mikrokontrolérem přišla americká společnost Texas Instruments. Jednalo se o řadu TMS1000, jehož zástupce je na obrázku 3. [7]

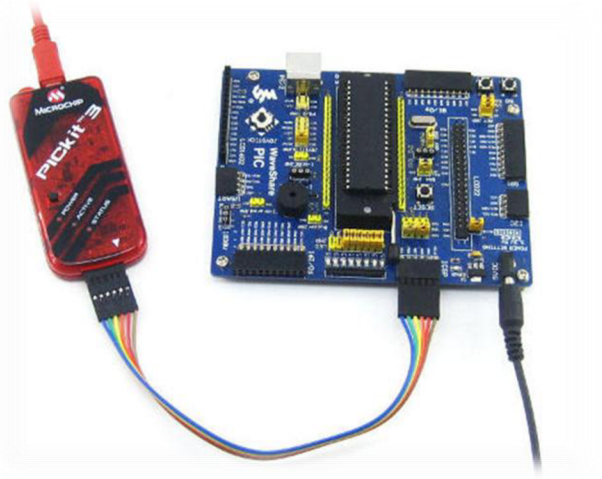
Obr. 4: Mikrokontrolér řady TMS 1000. [9]

1.2.3 Tlačítko RESET

Tlačítko RESET slouží k restartování, celého programu nahraného do paměti desky. Je velkým pomocníkem při vytváření a nového programu.

1.2.4 ICSP

In circuit serial programming značí programování součástky, která již byla umístěna do obvodu. Důvodem použití tohoto druhu programování je daleko snazší přístup k čipu, který nemusí být vyjmut a poté následně pájen. Programuje se pomocí programátoru právě přes ICSP konektor (obr. 5). [10]



Obr. 5: ICSP programování za použití programátoru. [10]

1.2.5 Napájecí jack

Obvyklým způsobem je deska napájena přes spojení USB avšak zejména v případech zařízení, která se nějakým způsobem pohybují nebo mají odesílat data na vzdálenost, která nedovoluje spojení s počítačem, je třeba použít jiný zdroj energie. Tím může být např. 9 V baterie, která se umístí do svorek a spojení těchto svorek s baterií a deskou je realizováno právě přes napájecí jack. Další možností napájení je použití AC – DC adaptéru.

Použitelné rozmezí napětí, při kterém může deska pracovat (v případě desky MEGA) je 6- 20 V. Pokud je ale dodávané napětí nižší než 7 V, může se stát, že na 5 V pinu bude méně než 5 V, což činí desku nestabilní a nemusela by tedy pracovat správně. Na druhou stranu, použití většího napětí než 12V může mít za následek přehřátí regulátoru napětí, a tím pádem spálení v podstatě celé desky. Doporučená hodnota na webu výrobce je tedy 7 - 12 V. [6]

1.2.6 16 MHz krystal (krystalový oscilátor)

Jedná se o elektronickou součástku využívající piezoelektrického jevu. Piezoelektrický jev objevili v roce 1880 bratři Curieové. Po stlačení piezokrystalu je na jeho povrchu možno pozorovat vznik elektrického náboje. O rok později byl objeven i jev opačný- přivedením náboje se krystal jeho působením deformuje. [11]

16 MHz krystal (existují různá provedení) generuje velmi stabilní kmity o přesných frekvencích a používá se jako zařízení pro udržování času (tzv. time tracking) a to nejen u desky Arduino. [12]

1.2.7 USB konektor

Universal serial bus, zkráceně USB značí vysokorychlostní sériovou sběrnici, která desce poskytuje přímé spojení s počítačem, a tedy umožňuje touto cestou nahrát program, kterým se bude řídit fungování desky. Toto spojení je oboustranné - deska může také odesílat data zpět do počítače k případnému zpracování a vyhodnocení. Skrze USB je, jak již bylo řečeno dříve, je ve většině případů poskytováno potřebné napájecí napětí. [13]

Existuje několik tvarově i velikostně odlišných typů USB, jak je ukázáno na obrázku 6. V případě Arduino je obvykle použit typ B, počítače mají standardně typ A, tento problém je vyřešen užitím kabelu s odlišnými typy koncovek.

Obr. 6: Typy USB koncovek. [14]

1.2.8 Piny

Piny, označené na desce v obrázku 2, jsou místem, kde dochází ke spojení desky a externího zařízení např. senzoru nebo led diody. Toto spojení je realizováno zasunutím samce drátu do samice na desce. Pro snadnější práci a přehlednost jsou piny na desce očíslovány.

Pinů existuje více druhů a to:

a) analogové a digitální (I/O)

c) komunikační

d) power piny

a) Analogové a digitální piny

Skvělým příkladem práce s analogovým signálem je užití obvodu se světelným senzorem (např. za použití fotorezistoru). Vlastnosti senzoru se mají pokud možno plynule měnit např. zvyšováním či snižováním odporu v závislosti na množství dopadajícího světla. To se projeví spojitou změnou napětí, což představuje analogový signál. Pro vstup analogových signálů se užívají analogové piny.

Na druhou stranu, snadnější práce je s digitálními signály. Většina mikrokontrolérů operuje s napětím 0 – 5 V. Digitální signál nabývá pouze dvou hodnot a to logická 0 nebo 1 (např. led dioda nesvítí/svítí), v závislosti na napěťové hodnotě signálu, přičemž hodnoty se v určitém pásmu překrývají, což může způsobit nesprávnou funkci systému.

Společným znakem analogových i digitálních pinů je možnost změny jejich konfigurace na vstup či výstup dle libosti uživatele.[6]

b) Komunikační piny

Tyto piny jsou použity, jak již název napovídá ke komunikaci prostřednictvím dat. Jejich účel je zřejmý a jedna z forem přenosu dat bude popsána v dalších bodech této práce.

c) Power piny

Napájecí piny poskytují požadované napětí v standardizovaných hodnotách 5 V; 3,3 V a GROUND neboli země což odpovídá 0 V.[6]

1.2.9 Shieldy

Rozšíření desky se v případě Arduino nazývá shield. Tyto shieldy bývají obvykle zaměřeny na konkrétní oblast. Příkladem může být např. Ethernet shield, jestliže je třeba pracovat s internetovým připojením. Je-li třeba pracovat s různými typy motorů, ať už jde o krokový motor nebo servo motor a jiné. Kvůli otevřenosti projektu jsou k dispozici množství různých typů shieldů nerůznějšího určení.

Díky shieldům se rozšiřuje spektrum oblastí, kde se tato deska dá použít. V mnoha případech alespoň usnadňuje práci obzvláště v použití se softwarovými knihovnami. Open source licence nabízí komukoliv vytvořit vlastní shield, jenž je pak stává volně dostupným.

Na obrázku 7 je příklad Ethernet shieldu osazeného skrze piny na Arduino – což je standardní způsob spojení shieldu s deskou. [6]

Obr. 7: Ethernet shield. [15]

1.2.10 Paměti

Využití paměti jako prostoru k nahrávání či ukládání dat je esenciální pro každé programovatelné zařízení. Na desce Arduino je použito 3 základních typů pamětí [6] :

1. ROM
2. RAM
3. Flash

a) Paměť ROM

ROM (read-only memory) je nevolativní (nezávislá na napájení) vestavěná elektronická paměť, jejíž obsah je dán již při výrobě. Její data se tedy, vzhledem k nezávislosti na napájení, uchovávají i při vypnutí zařízení.

V této paměti je uložen základní program zařízení. Obdobou je například BIOS u klasických počítačů, který ovládá počítač od zapnutí do doby, než se spustí vlastní operační systém, který je nahrán na pevném disku, či jiném úložném zařízení. [13]

b) Paměť EEPROM

Tato poněkud krkolomná zkratka znamená electrically erasable programmable read only memory, volně přeloženo elektricky přemazávatelná programovatelná paměť ROM. Čipy paměti jsou vybaveny ovladačem, který umožňuje změnu dat uložených v paměti. To je potřeba při aktualizacích firmware, což je právě to, co se ukládá do paměti ROM (např. BIOS, firmware mobilních telefonů atd.). Dříve bylo u pamětí typu ROM nutné fyzicky vyměnit čip, jestliže jsme potřebovali změnit firmware. Tento problém řeší právě EEPROM.

Kapacita této paměti je závislá na typu desky. Například deska UNO má k dispozici 1KB, zatímco MEGA disponuje pamětí 4x větší.[13, 6]

c) Paměť RAM

Dalším nezbytným typem paměti je paměť RAM neboli random-access memory. RAM je na rozdíl od ROM závislá na napájení a její obsah vymizí po vypnutí zařízení. Tato paměť se laicky dá popsat jako místo, kde se aktuálně pracuje s programem uloženým na úložišti. Existují dva základní typy paměti RAM, které se samozřejmě dále větví. [13, 6, 16]

d) Dynamická paměť RAM

V okamžiku kdy do dynamické paměti jsou odeslána data, započne proces ztrácení těchto dat. Aby paměť fungovala správně, začne tzv. refresh dat. To je způsobeno tím, že kondenzátory, které formují buňku paměti, se vybíjejí, a tím nemohou nadále uchovávat hodnotu v sobě uloženou. Rozhodujícím parametrem u dynamických pamětí je tedy doba s jakou ztrácí kondenzátory svoji hodnotu (popř. obnovují data - provádí refresh). Čím menší je tato doba, tím je paměť lepší, přičemž doba označovaná v anglicky psané literatuře jako refresh rate je řádově v nanosekundách. [13, 16]

e) Statická paměť RAM

Statická paměť RAM se liší od dynamické svým provedením. Buňka paměti je tvořena kombinacemi tranzistorů. Právě svojí stavbou udržuje statická paměť své hodnoty

tak dlouho, dokud je do paměťového modulu dodávána energie. Tento typ paměti, kterým disponují i desky Arduino, je mnohem rychlejší, avšak to je vykoupeno vyšší cenou. [13, 16]

Na stránkách výrobce Arduino je dále připomenut problém s nedostatečnou pamětí SRAM. Model UNO má k dispozici 2 KB SRAM. Pro běžný sketch (název pro program v souvislosti s Arduino) je to kapacita dostatečná, avšak při rozsáhlejších pracích, zejména při použití velkého množství dat typu char, není problém paměť vyčerpat. Příkladem může být příkaz:

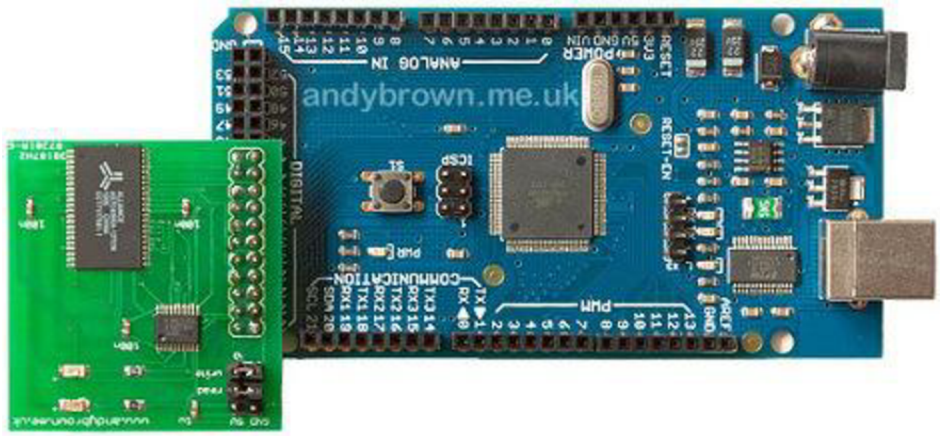
```
char message[] = „Faculty of mechanical engineering.“;
```

Tato zpráva obsahuje 33 znaků, každý zabere 1 byte a navíc ještě je tento objem navýšen o jeden byte pro tzv. null terminator ‚\0‘, který ozančuje konec stringu neboli textového řetězce. Celkem tato zpráva zabere 34 bytů, což je značná porce vzhledem k tomu, že sdělení není nikterak obsáhlé a kapacita paměti modelu UNO je 2000 bytů. Z tohoto důvodu je třeba s daty typu char nakládat rozumně a pokud je to možné, vyhýbat se dlouhým textovým sdělením, které by mohly zbytečně zatěžovat statickou paměť.

Jestliže by byla překročena kapacita paměti, mohou se objevit neočekávané potíže- program nemusí fungovat vůbec nebo může fungovat chybně, přestože se objeví hlášení, že sketch byl nahrán úspěšně. K rozpoznání takového problému je doporučeno zkrátit obsahy řetězců, či upravit sketch tak, aby zátěž na paměť byla menší a v případě bezproblémového chodu je pravděpodobné, že u původního kódu byla překročena kapacita paměti SRAM. [6]

Úprava sketche může být následující [6] :

1. V případě spolupráce s programem na počítači, přesunout část operací na tento program.
2. Zrevidovat použité datové typy. Typ int zabírá 2 byty, oproti tomu typ byte zabírá pouze jeden, avšak může z podstaty věci uložit menší škálu hodnot. Vždy je nejlepší používat, co nejmenší datový typ, který je pro danou hodnotu (či hodnoty) dostatečný.
3. Pokud textové řetězce nejsou modifikovány během chodu programu, je možné je uložit na paměť flash.
4. Použití rozšiřujícího modulu paměti SRAM (obr. 8).



Obr. 8: Modul SRAM na desce MEGA. [17]

f) Paměť FLASH

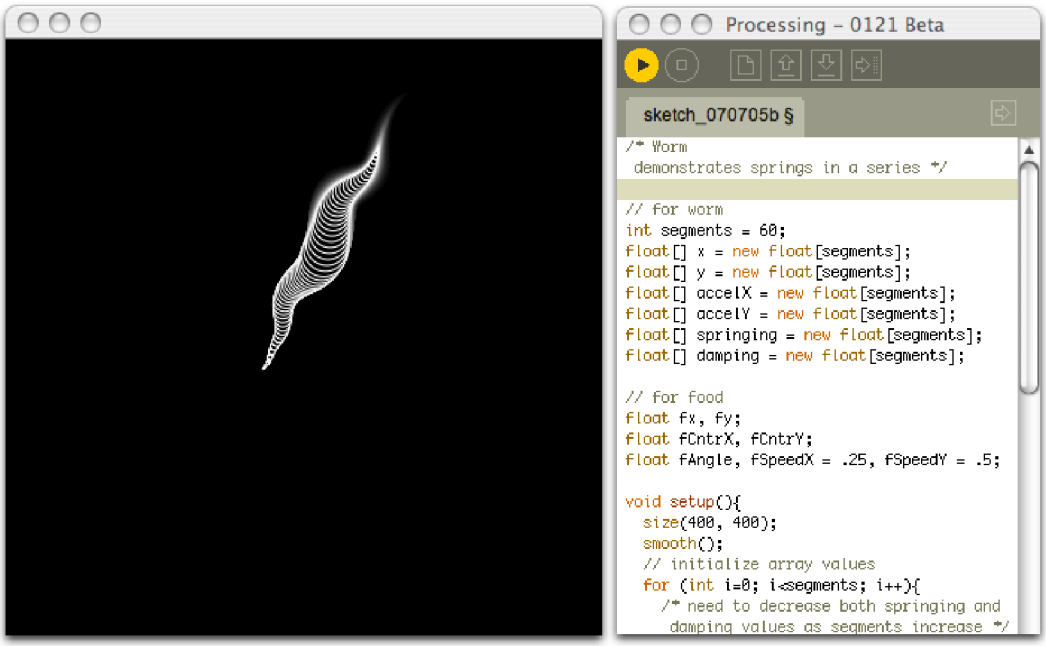
Paměť je určena k uložení sketch. Jak již bylo řečeno dříve, tato data nemohou být v průběhu programem upravována. Jestliže je potřeba úprava (např. různé mat. operace) musí být data nejprve zkopírována do paměti SRAM. Vlastností paměti flash je, že je nevolativní, tedy data jsou trvale uložena i po odpojení zdroje. Jako nevýhoda se jeví omezená životnost paměti, která se pohybuje okolo 100 K cyklů, avšak tento počet je více než dostatečný pro standardní účely. Kdybychom nahráli denně 10 programů, paměť by vystačila na 27 let takového nahrávání než by došlo k jejímu zničení. [6,16]

1.3 Software desky

Prostředí desky Arduino je nezbytnou součástí práce s touto programovatelnou platformou. Arduino IDE samotné je napsané v jazyce Java a svým založením vychází z prostředí jazyka Processing. Program Arduino je možno použít u operačních systémů Windows, Mac i Linux. [1, 6, 16]

1.3.1 Processing

Hlavní cílovou skupinou, pro kterou bylo vyvinutí tohoto jazyka zamýšleno, jsou umělci, vynálezci, studenti a návrháři. Výhodou použití Processing je jeho orientace na grafický výstup, díky čemuž si získal značné oblíbenosti. Na obrázku 8 je prostředí jazyka Processing a ukázka výsledku práce v něm. [19]



Obr. 9: Processing. [20]

1.3.2 Arduino IDE

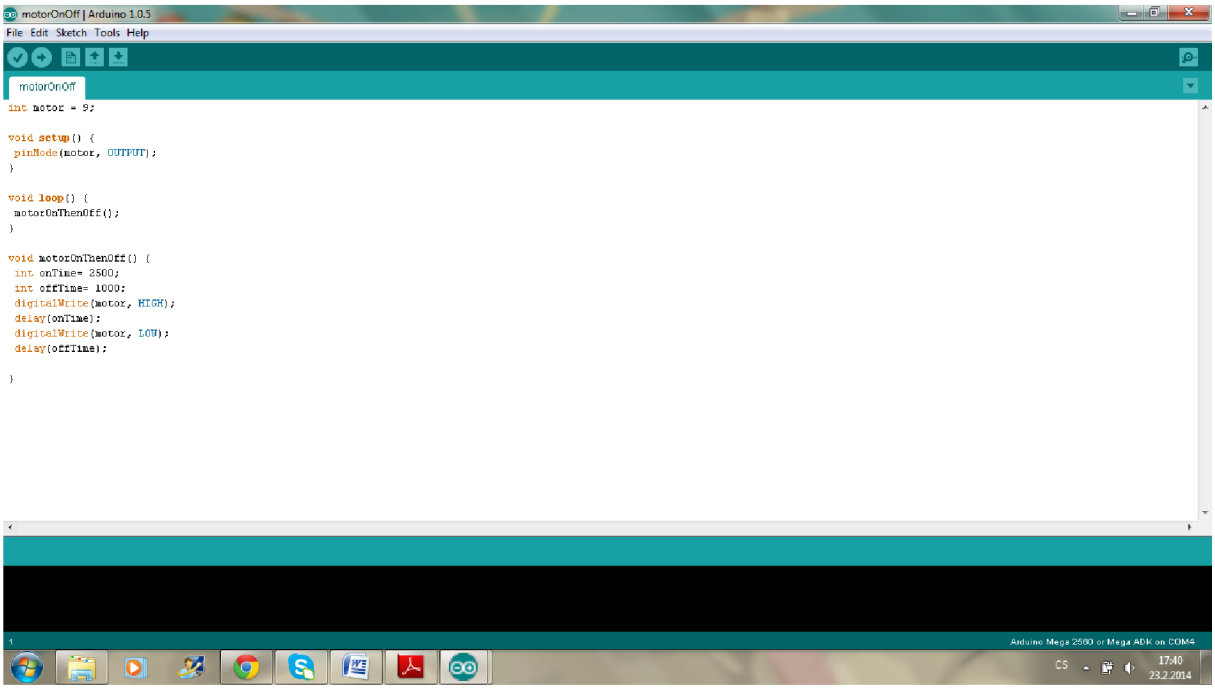
Zkratka IDE v informatice značí Integrated Development Environment tedy integrované vývojové prostředí. Vzhledem k otevřenosti projektu je ho možno bezplatně stáhnout na stránkách originálního výrobce desek Arduino. [6]

Programovací jazyk, ve kterém jsou psány programy (zde nazývané sketch), vychází z jiného a to z jazyka Wiring. Pro přehlednost je vložena část článku na odborném serveru *root.cz*:

„Programovací jazyk, v němž můžete pro Arduino psát, se jmenuje Wiring (lépe řečeno „vychází z jazyka Wiring“, ale rozdíly jsou minimální). Podle autorů jde o „jazyk podobný C++“ – ve skutečnosti je to jakýsi metajazyk či „vylepšená sada maker“ nad C++. Wiring původně vznikl pro vývojový kit podobný Arduinu a vychází z dalšího open source projektu, který se jmenuje Processing (můžete si všimnout určité podobnosti v logu Arduina a Processingu).“

MALÝ, Martin. Arduino: Jak pro něj začít programovat. In: *Root.cz: informace nejen ze světa Linuxu* [online]. 8.7.2010 [cit. 2014-05-23]. Dostupné z: http://www.root.cz/clanky/arduino-jak-pro-nej-zacit-programovat/

Prostředí (obr. 10) je velice přehledné a vše je uspořádáno, tak aby se i úplný začátečník neztratil. Nedílnou součástí je hlavní panel s klasickými nabídkami (File, Edit...), pod kterým je umístěno několik tlačítek s nejpoužívanějšími funkcemi. Pod nimi se nachází karta s názvem souboru a pracovní prostor připomínající prostý textový editor.



Obr. 10: Arduino IDE.

2 Moduly Xbee

Xbee je rádiový modul na bázi mikrokontroléru, který je hojně využíván v oblasti bezdrátové komunikaci a řízení. Přenos je založen na protokolu ZigBee, známý též jako IEEE 802.15.4, který poskytuje, hlavně z nákladového a napájecího hlediska, kvalitní bezdrátovou komunikaci, která se uplatňuje ponejvíce v průmyslu. [21, 22]

2.1 Charakteristiky Xbee

Své použití nalezl Zigbee standart zejména u bateriově napájených aplikací kvůli nízké spotřebě energie při relativně velkém dosahu. Pro porovnání s technologiemi, které se v daném odvětví dají použít, je zde umístěn obrázek 11. Tato tabulka nabízí zajímavé srovnání rozhodujících parametrů při volbě způsobu komunikace mezi danými objekty, avšak každá technologie má své výhody i nevýhody a jejich použití závisí na konkrétním zadání. [18,19]

Obchodní jméno Standard	GPRS/GSM 1xRTT/CDMA	Wi-Fi™ 802.11b	Bluetooth™ 802.15.1	ZigBee™ 802.15.4
Aplikační zaměření	Široké oblasti Hlas & Data	Web, Email, Video	Náhrada za kabel	Monitorování & Řízení
Systémové zdroje (paměť)	16MB a více	1MB a více	250KB a více	4KB - 32KB
Životnost baterií (dny)	1-7	0.5 - 5	1 - 7	100 – 1 000 i více
Max. velikost sítě (počet uzlů/síť)	1	32	7	65 000 (příp. až 2^{64})
Přenosová rychlost (Kb/s)	64 – 128	11 000	720	20 - 250
Komunikační dosah (m)	1 000 i více	1 - 100	1 - 10	1 – 100
Výhody	Dosažitelnost, Kvalita	Rychlost, Flexibilita	Cena, Jednoduchost	Spolehlivost, Výkon/Cena

Obr. 11: Tabulka srovnání vybraných bezdrátových technologií přenosu dat. [21]

2.2 Odlišná provedení

Provedení modulů se z důvodu, co nejvhodnějšího řešení, liší, původní generace (series) jsou nahrazovány novými. Bohužel tyto generace nejsou mezi sebou kompatibilní. [20]

2.2.1 Antény

Další odlišností, která je patrná nejvíce, jsou antény. Provedení antén je různé a každé lépe vyhovuje jiné situaci. Zde jsou základní typy antén [22, 23] :

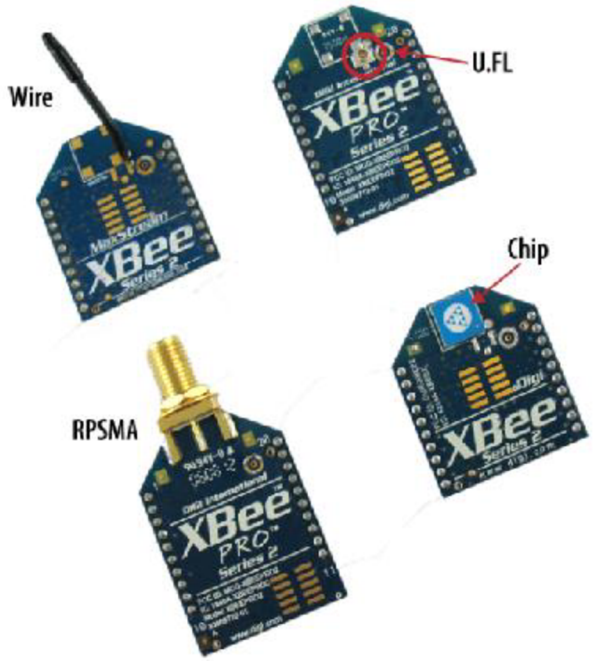
a) Drátové antény: Klasické, velmi často používané antény. Výhodou tohoto nejjednoduššího provedení je fakt, že tyto antény zajišťují kruhové pokrytí, tedy radiace signálu je ve všech směrech stejná. Provedení je na obrázku 10.

b) Čipová anténa: Jedná se o keramický čip umístěný na těle modulu Xbee, jenž díky svým malým rozměrům výrazně snižuje velikost zástavbového prostoru a zároveň dělá celou konstrukci robustnější. Tyto výhody jsou ale podmíněny vyšší cenou. Radiace signálu již není ve všech směrech stejná, nýbrž má tvar kardioidy (srdcovitý tvar). Ukázka provedení čipové antény je taktéž na obrázku 12.

c) PCB anténa: Zkratka PCB v překladu znamená deska plošných spojů. V tomto konkrétním případě značí, že anténa je součástí desky, tedy je na ni přímo tištěna. Tento typ má stejné výhody a nevýhody jako čipová anténa, avšak s tím rozdílem, že její výroba je levnější, což by se mělo projevit na prodejní ceně.

d) U.FL konektor (obr. 10): Tento malý konektor umožňuje připojení vlastní antény. V některých případech, např. je-li Xbee součástí zařízení, které je opatřeno krytem, je potřeba aby anténa dosahovala nad úroveň tohoto krytu, což by nebylo možné u antény drátové, která má omezenou délku. Jedním z dalších případů je, když potřebujeme anténu orientovat jiným směrem než samotný modul (např. použití u vysokoziskových antén).

e) RPSMA konektor: Tento konektor je obecně robustnější a vzhledově jiný než U.FL konektor. Rozdílem je, že na závit tohoto konektoru se anténa přišroubuje, zatímco v případě předchozím, je třeba spojení za pomoci drátů.

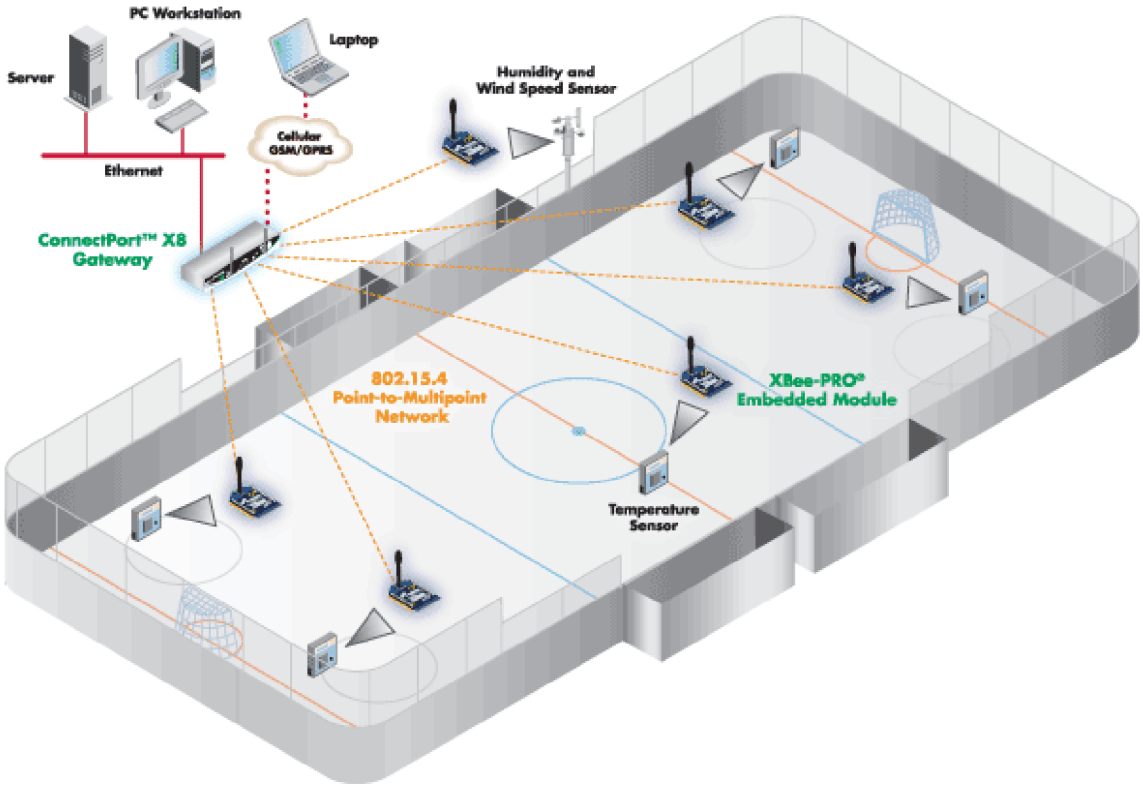


Obr. 12: Vybrané druhy antén modulů XBee. [21]

2.2.2 Series

Prakticky žádný výrobek nezůstane beze změny od svého prvního provedení, obzvláště ne v elektronice. Xbee disponuje prozatím 2 – 3 generacemi.

Ty se od sebe v určitých ohledech značně liší Jedno z možných použití modulů Xbee je na obrázku 13. [21, 22, 23]

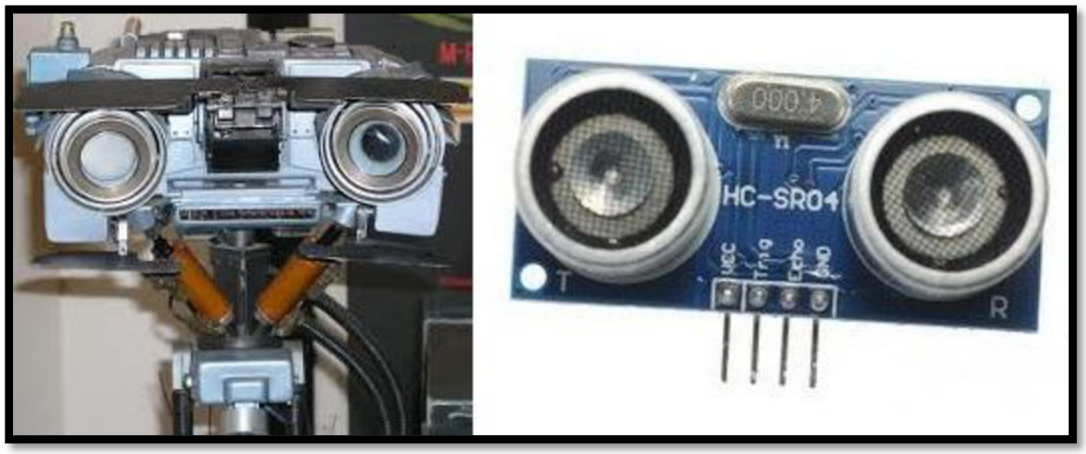


Obr. 13: Použití modulů Xbee a jejich zapojení point-to-multipoint. [23]

3 Ultrazvukový senzor HC-S04

Jedná se o piezoelektrický senzor, využívající piezoelektrického jevu, který objevili bratři Curieové v 19. stol (viz bod 1.2.6). Vnější stavba na pohled připomíná hlavu robota Johny 5 z filmu Číslo 5 žije (1986)- toto odlehčující srovnání je na obrázku 14.

Jedno z vysílacích „ok“ je označeno T (transmit – vysílat) a druhé je označeno R (recieve – přijímat). Vstupní napětí je přiváděno na pin s označením VCC. Pro zaručení správné funkce senzoru by napětí na vstupu mělo odpovídat 5 V. Některé jiné senzory pracují s napětím 3,3 V. Dalšími vývody jsou označeny jako TRIG a ECHO. TRIG je odpovědné za vysílání pulzů (anglicky trigger- spoušť), oproti tomu ECHO slouží k odečítání těchto pulzů. Poslední je GND (z angl. ground- země), který odpovídá 0 V. Podle datasheetu (příloha B) je s tímto senzorem dosáhnout za ideálních podmínek přesnosti až 3 mm.



Obr. 14: Johnny 5 a čelní pohled na senzor HC-S04. [25, 26- upraveno]

3.1 Ultrazvuk

Mechanické kmitání částic prostředí, které je nad hranicí slyšitelnosti lidského ucha, se nazývá ultrazvuk. Definovat se dá také za pomoci kmitočtu (frekvence), jako kmitání o kmitočtu vyšším než je 20 kHZ. Druhý způsob určení je vhodnější, protože horní hranice slyšitelnosti značně klesá s věkem, může být mezi 20-8 kHz. Běžně se v praxi využívá ultrazvuku o frekvenci 30 MHz v laboratorních podmínkách, což činí ultrazvuk, alespoň z pohledu kmitočtu, jako velmi široký pojem. Ultrazvuk se řídí stejnými zákony jako zvuk slyšitelný, avšak s tím rozdílem, že intenzita ultrazvukových vln je mnohonásobně vyšší. Díky této skutečnosti bylo nalezeno v praxi značného uplatnění téměř ve všech oborech lidské činnosti. [27, 28]

3.2 Základní rovnice a pojmy kmitání a vlnění

Periodický pohyb částic lze rozložit na harmonický pohyb se sinusovým průběhem. Jako analogie je možné použít těleso zavěšené na pružinu, které je uvedeno do pohybu. Jestliže uvažujeme netlumenou soustavu, budou výchylky vždy stejné a při splnění určitých předpokladů popsatelné vlnovou rovnicí [27]:

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2}$$

jejímž partikulárním řešením je rovnice pro okamžitou výchylku y:

$$y = A\, sin\omega(t - \frac{x}{c})$$

kde: A- amplituda výchylky
c – rychlost šíření vlny
x- poloha částice v čase
t- čas
ω- úhlová rychlost
ϕ= ω x/c – fázový úhel

Z tohoto vztahu lze postupnými časovými derivacemi získat postupně rychlost a následně zrychlení. Další nezbytnými rovnicemi pro výpočty spojené s kmitáním jsou [27, 28]:

$$\omega = 2\pi f$$

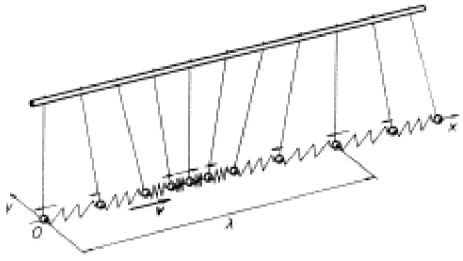
$$f = 1/T$$

kde T je perioda kmitání, což je doba, za kterou se okamžitá výchylka y opakuje. Následně je definována vlnová délka lambda jako podíl rychlosti a frekvence [27, 28]:

$$\lambda = c/f$$

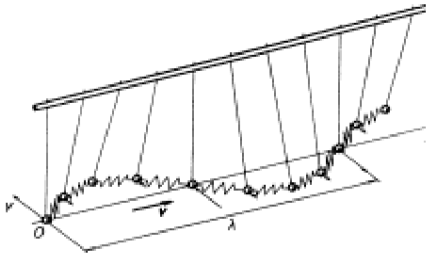
3.3 Dělení vln (podle šíření)

a) Podélné vlnění (obr. 15) - částice kmitají po přímkové dráze ve směru šíření a lokálně nahušťují a zřeďují prostředí, kterým vlna prochází. Je nejčastěji použitým druhem vlnění v praxi. [27]



Obr. 15: Podélné vlnění. [29]

b) Příčné vlnění (obr. 16) - částice kmitají po přímkové dráze avšak kolmo na směr šíření. [27]



Obr. 16: Příčné vlnění. [29]

c) Ostatní druhy vlnění jako povrchové vlnění, deskové vlnění a další. [27]

3.4 Aplikace ultrazvuku

Ultrazvuk se uplatňuje v mnoha oblastech a aplikacích např:

1. Zkoušení materiálu (myšleno zejména mech.vlastností)
2. Ultrazvuková defektoskopie
3. Měření hloubek vodních ploch
4. Kontrola stavu hladiny v nádržích
5. Ultrazvuková sonografie (lékařství)
6. Rozbíjení ledvinových kamenů
7. Senzory mapující okolní prostředí
8. A jiné…

4 MATLAB

„MATLAB je integrované prostředí pro vědeckotechnické výpočty, modelování, návrhy algoritmů, simulace, analýzu a prezentaci dat, paralelní výpočty, měření a zpracování signálů, návrhy řídicích a komunikačních systémů. MATLAB je nástroj jak pro pohodlnou interaktivní práci, tak pro vývoj širokého spektra aplikací.“

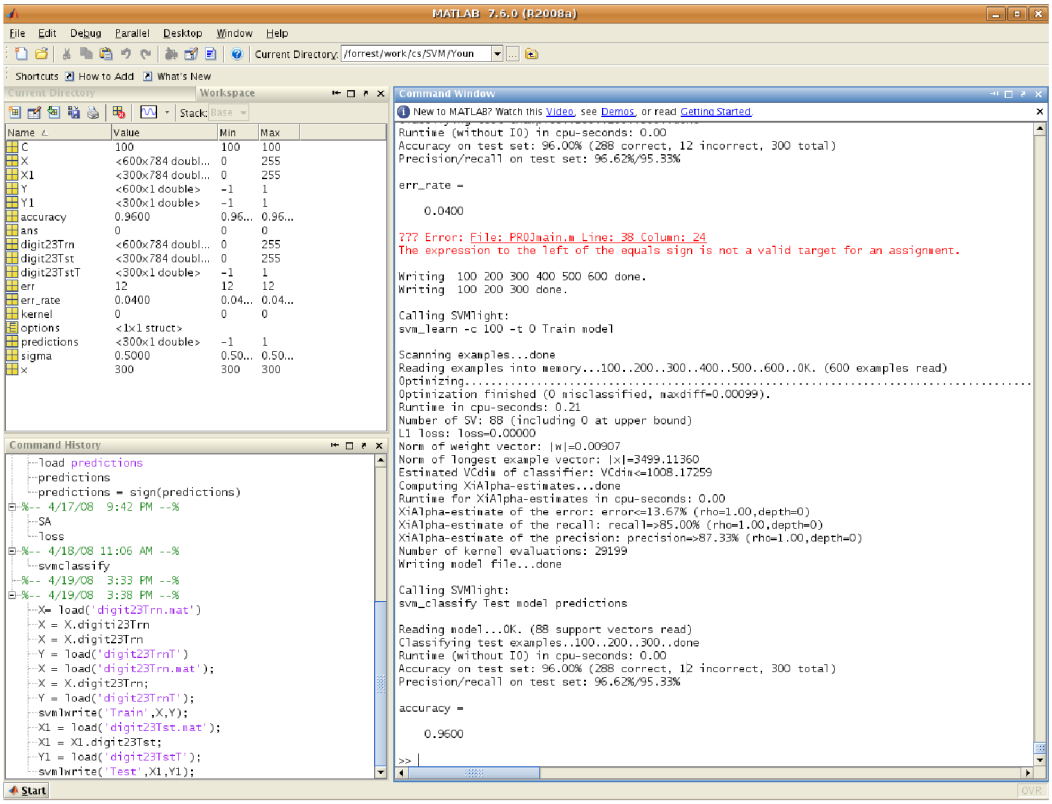
Matlab. HUMUSOFT. *Humusoft: Technické výpočty, řídící technika, simulace* [online]. © 1991 – 2014 [cit. 2014-05-23]. Dostupné z: http://www.humusoft.cz/produkty/matlab/matlab/

Nespornou výhodou MATLAB (název vznikl z Matrix Laboratory), patrnou již z názvu, je práce s maticemi a vektory. Tento program má zároveň obrovské množství rozšiřujících modulů (nazývaných toolboxy), které jsou vždy specificky orientovány na danou problematiku. Kupříkladu modul SIMULINK je orientovaný na vstup/výstup systémy

a pomocí něj se dají řešit nejen úlohy dynamiky ale i mnoho dalších. Právě pro svá rozšíření je MATLAB oblíbený ponejvíce mezi „inženýrskou“ komunitou, pro kterou by i původně koncipován. [30, 31]

MATLAB vychází z jazyka FORTRAN (z formula transaltion), což byl jedním z prvních vyšších programovacích jazyků. Byl vytvořen Johnem Backusem pro společnost IBM v roce 1954, avšak jeho komerční potenciál byl spatřen až později. Jazyk nepatří k nejmodernějším, avšak je dodnes používán zejména ve vědeckých a průmyslových aplikacích, přičemž je stále aktualizován. [31]

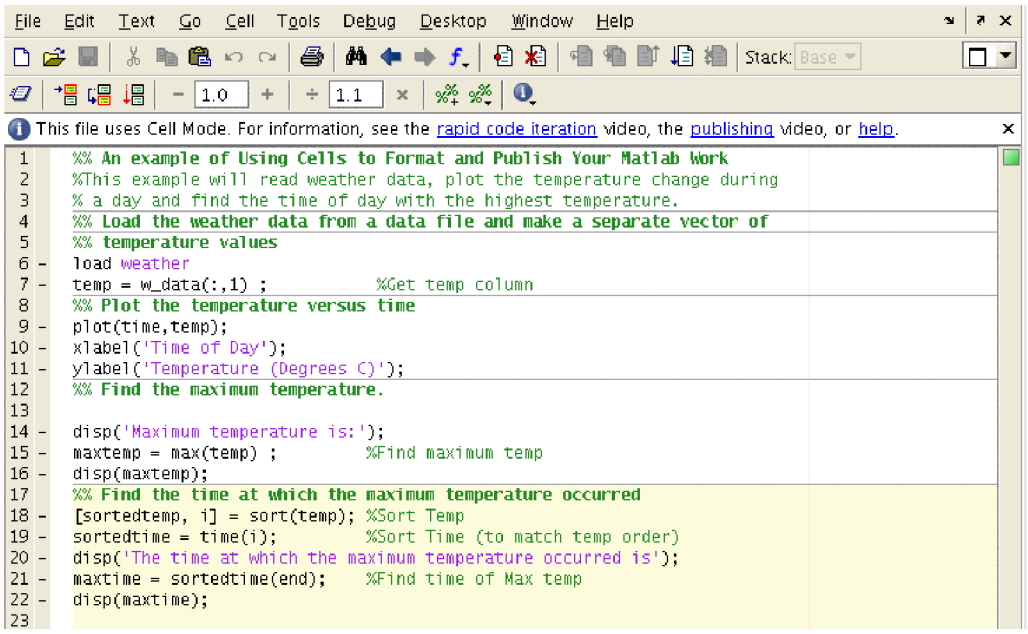
Při spuštění MATLAB se zobrazí hlavní okno (obr. 17), které zahrnuje části Command Window, kde probíhají veškeré výpisy a je možné zde pracovat jako s kalkulačkou, dále Command History, kde je historie použitých příkazů a také soupis inicializací programu- vše chronologicky seřazené a naposled podokno Workspace se seznamem a vlastnostmi proměnných, které lze přepínat pomocí panelu s Current Folder, kde je zobrazena složka se soubory typu jako m-file (.m), či figures (.fig)a další, které jsou MATLAB vlastní.



Obr. 17: Hlavní okno MATLAB.

[32]

Další důležitou částí je Editor (obr. 18) k tvorbě právě zmiňovaných souborů. V tomto editoru, připomínající klasický zápisník, jsou vytvářeny soubory, které jsou spustitelné buďto stisknutím tlačítka Run na horní liště nebo přes Command Window.



Obr. 18: Editor MATLAB. [33]

Praktická část

5 Rozbor úlohy

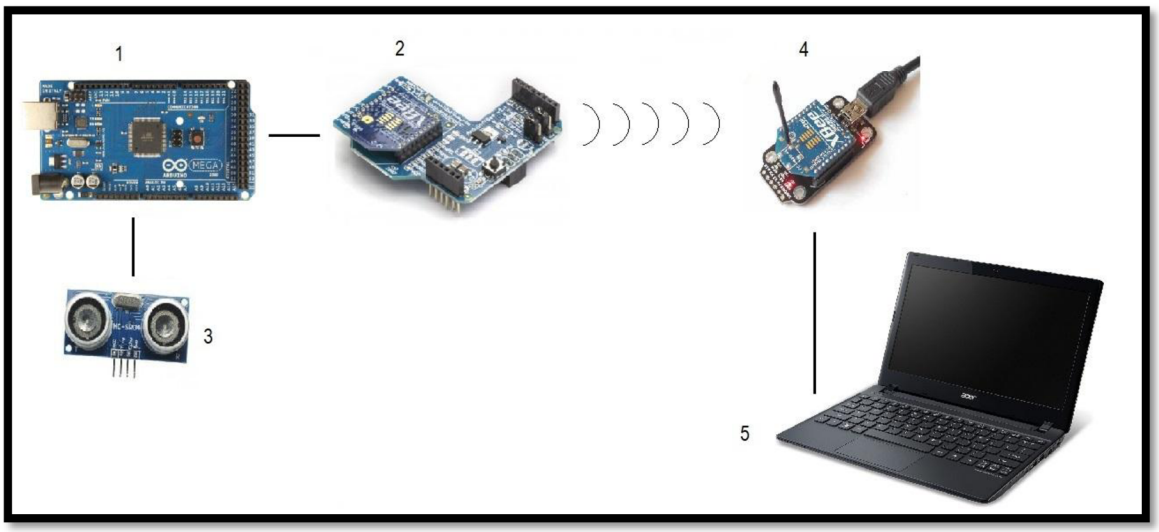
Úkolem je navrhnout obvod k měření vzdálenosti pomocí ultrazvukového senzoru HC – S04 jehož naměřená data budou přenášena bezdrátově do počítače a v reálném čase vykreslena. K tomuto bude použito programovatelné desky Arduino a rádiových modulů Xbee, dále pak programů Arduino a MATLAB. U MATLAB bude využita školní licence FSI VUT v Brně. Přehled použitých částí je v soupisu použitých součástek.

6 Návrh a zapojení obvodu

6.1 Popis měřícího obvodu

Schéma měřícího elektronického (obr 19) vychází ze zapojení desky Arduino (1) osazené Xbee shieldem s modulem XBee (2) a ultrazvukového senzoru (3). Toto zapojení má své vlastní napájení v podobě 9 V baterie upevněné ve svorkách (alternativou je adaptér) a z hlediska energetického tvoří nezávislý celek.

Druhá část obvodu je tvořena sesterským modulem XBee přijímající signál, které je upevněné v adaptéru (4). Tento adaptér je spojen s počítačem USB kabelem nejen z důvodu přenosu dat, ale také kvůli napájení osazeného adaptéru.



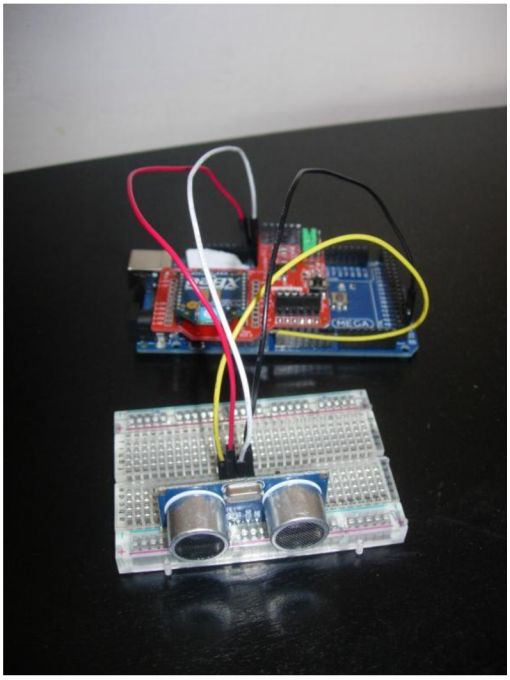
Obr. 19: Schéma zapojení měřícího obvodu. [6, 25, 34, 35- upraveno]

6.2 Zapojení senzoru

Senzor, jak již bylo napsáno, pracuje při napětí 5 V, proto zapojíme přes nepájivé pole vývod označený jako VCC pomocí drátku (žlutý) na 5 V na desce Arduino. GND na senzoru zapojíme na GND (černý) na desce, tímto máme vyřešeno napájení senzoru, který si ke své práci bere elektrickou energii skrze power piny na Arduinu.

Dále pak je třeba zapojit ECHO na libovolný digitální pin např. č. 8, stejný postup aplikujeme na TRIG a použijeme např. č. 8. Tuto volbu musíme zohlednit při psaní sketch. Lépe si lze představit zapojení při pohledu na fotografii na obr. 20. Senzor je tím pádem zapojený a připravený pracovat podle programu.

Pozn.: Barvy v textu odpovídají barvám drátků na obrázku.



Obr. 20: Zapojení měřící části obvodu.

7 Vytvoření sketche k měření vzdálenosti

Celý sketch je dostupný v papírové podobě v příloze C, dále pak na přiloženém DVD. Nyní bude sketch po částech rozebrán a okomentován.

7.1 Označení pinů a tvorba klíčové proměnné

```
int trigPin = 9;
int echoPin = 8;
long doba = 0;
```

V této části byly zavedeny dvě proměnné typu int. Někdy je možno vidět konkrétně tyto proměnné psané s *const* před celým výrazem, protože hodnoty obou proměnných se v celém sketch dají považovat za konstantu a označují čísla pinů, kam byly příslušné póly zapojeny. Je lepší hodnoty pinů, tak jako vše, uložit do proměnných, aby při jejich změně nebylo nutné přepisovat sketch na více místech.

Poslední proměnná *doba* je ve formátu long, který umožňuje využít 32 bitů- to vše z důvodu toho, že dopředu není možné zjistit, za jak dlouho se signál do přijímače vrátí. *Doba* bude dále znamenat čas, za jaký se vrátí vyslaný pulz. Její počáteční hodnota je rovna nule. [6]

7.2 Nastavení pinů a zahajení komunikace

```
void setup()
{
  //Nastaveni pinu a zahajeni komunikace
  pinMode(trigPin,OUTPUT);
  pinMode(echoPin,INPUT);
  Serial.begin(9600);
}
```

Setup znamená nastavení. Toto nastavení se v rámci sketch provede jednou a jeho hodnoty jsou uchovány. Tato část je nezbytnou částí každého sketch.

Znaky // označují jednořádkový komentář – řetězec za těmito znaky je jen poznámkou pro autora, či člověka, který se sketch přijde do styku. Komentáře slouží k lepší orientaci a je to vítaná pomoc při objasnění či osvěžení smyslu použitých funkcí a postupů.

Funkce pinMode() nastavuje, je- li pin použit jako vstup (INPUT) či jako výstup (OUTPUT). Na prvním místě argumentu je požadováno číslo pinu, který chceme nastavit, na druhém pak hodnota nastavení. Obojí lze nastavit číselně, avšak uživatelsky příjemnějším je shledáno definování proměnnou a slovně.

TRIG vysílá ultrazvuk, je tedy výstupem– bylo nastaveno OUTPUT. Naopak ECHO je příjemcem pulzu, a tedy bylo nastaveno jako vstup- INPUT.

Na poslední řádek byla vložena funkce Seriál.begin(), která zahajuje komunikaci a která jako argument požaduje modulační rychlost (ang. baud rate). Rychlost 9600 se jeví jako standardní a zcela dostačující pro danou práci. Tato rychlost musí být za všech okolností stejná v rámci celého obvodu, což musí být později zohledněno i v MATLAB.

7.3 Generování pulzů

```
void loop()
{
  // Vyslani a prijem ultrazvuku
 digitalWrite(trigPin,LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin,HIGH);
 delayMicroseconds(10);

 // Ziskani vzdalenosti
 doba= pulseIn(echoPin,HIGH);
 long vzdalenost_mm;
 vzdalenost_mm= vzdalenost(doba);


//Odeslani dat pres serial com
 Serial.println(vzdalenost_mm);
  delay(1000);
}

// Vypocet vzdalenosti
long vzdalenost(long time)
{
  int zvuk_rychlost;
 zvuk_rychlost = 343;
 long rovnice;
 rovnice = 0.5*(zvuk_rychlost*0.001)*doba;

 return rovnice;

}
```

V předchozí části byl nastaven tzv. SETUP. V části LOOP se s těmito vstupními parametry dále pracuje. Loop znamená v angličtině smyčka a soubor instrukcí v ní obsažené jsou vykonávány periodicky, což v tomto sketchi zapříčiňuje stále vysílání a přijímání ultrazvukových pulzů a následný přepočet.

První blok se věnuje, jak je již zřejmé z úvodního komentáře, vyslání a přijímání ultrazvuku. Funkce digitalWrite() očekává dva parametry a to označení pinu (číslo nebo proměnná) a stav, kterého chceme prostřednictvím této funkce dosáhnout. Tyto stavy mohou být dvojí- HIGH (zapnutý/spuštěný/aktivní) nebo LOW (vypnutý). Funkce delay() či delayMicroseconds() nastavují časovou prodlevu, liší se pouze jednotkami- delay() se zadává v milisekundách, zatímco v druhém případě je zadáváno v mikrosekundách. Celý tento blok je dán možnostmi senzoru a jeho vysvětlení by mohlo znít asi takto:

1. Nastavení stavu LOW na TRIG a ponechání v tomto stavu po dobu 2 μs.
2. Po této odmlce je nastaven stav HIGH na TRIG a tento stav je ponechán po dobu 10 μs.
3. Tento sled instrukcí se periodicky opakuje.

7.4 Vzdálenost

V dalším bodě se již částečně dostáváme k získání vzdálenosti měřené senzorem. Pro přehlednost je blok vložen znovu:

```
// Ziskani vzdalenosti
 doba= pulseIn(echoPin,HIGH);
 long vzdalenost_mm;
 vzdalenost_mm= vzdalenost(doba);
```

Zde je vyjádřena *doba* jako výstup funkce pulseln(). Tato funkce potřebuje pro svůj chod dva argumenty a označení pinu, na kterém je umístěn přijímač (ECHO). Princip fungování je následující:

Při nastavení druhého argumentu na HIGH, je vyčkáno, dokud pin nedojde do tohoto stavu (může být nastaveno i LOW).

Je započato měření času, dokud pin nedojde do stavu LOW, kdy je měření zastaveno. Jde tedy o časové vyjádření délky pulzu. Výstupem je doba trvání pulzu v mikrosekundách.

Na dalším řádku je nadefinována proměnná *vzdalenost_mm*, která je výstupem zatím neznámé funkce s jediným argumentem *doba.*

7.5 Přenos dat

Další blok řeší odeslání dat. Byla použita instrukce Serial.printl(), která odešle data a jejich výpis se bude zobrazovat na novém řádku. Data, které je potřeba odeslat v této práci jsou hodnoty *vzdálenost_mm*, se kterými se následně pracuje v MATLAB. Časová prodleva odesílání byla zvolena a její hodnota je 1s.

7.6 Výpočet

Naposled, je tedy potřeba provést zjednodušený výpočet. Je známo, že při rovnoměrném přímočarém pohybu je dráha součinem rychlosti a času. Rychlost zvuku je známá veličina a je z definice pohybu brána jako konstanta, její hodnota byla pro 20°C vyčtena z tabulky [37].

Jedinou proměnnou je tedy čas. Ten je však také známý a byl zjištěn měřením a jeho hodnota je uložena v proměnné *doba*. Je třeba si uvědomit, že dráha pulzu je dvojnásobná, protože pulz je vyslán ze senzoru a následně se odrazí od překážky zpět. Výsledná podoba rovnice vypadá tedy takto:

$$s = v.\frac{t}{2} \quad (1)$$

Vlastní použití této rovnice je v programu zde:

```
// Vypocet vzdalenosti
long vzdalenost(long time)
{
  int zvuk_rychlost;
 zvuk_rychlost = 343;
 long rovnice;
 rovnice = 0.5*(zvuk_rychlost*0.001)*doba;

 return rovnice;
}
```

Nejprve je zavedena proměnná *zvuk_rychlost* a poté *rovnice*. S ohledem na rovnici (1) je jediný problém pouze s jednotkami. Zatímco rychlost zvuku je dosazena v m/s, proměnná doba je v µs. Výstupními jednotkami byly zvoleny mm a tak je třeba přidat do součinu činitel 0,001, který nám vykompenzuje nesourodost jednotek. Hodnota proměnné rovnice je jako výstup vrácena.

7.7 Nahrání sketche

Prvním krokem je napsání sketche v Arduino IDE, což bylo vykonáno v předchozím bodě. Poté propojíme desku a počítač USB kabelem. Ve správci zařízení vybereme porty a zjistíme na kterém portu je zařízení připojeno, port bude označován jako COMX (místo X bude celé číslo), deska bude tedy např. COM4 (pozn. takto jsou porty označeny v os. Windows, v Linuxu bude označení jiné). Důležitým úkonem je předat informaci o označení portu a o typu desky programu. To je provedeno následovně:

1. V horní liště vybereme postupně: Tools -> Seriál port -> vybrat označení portu (např COM4)

2. Opět v horní liště: Tools -> Board -> vybrat typ desky (např. Arduino Micro)

Jestliže vybereme špatný port, poznáme to okamžitě. Ihned po kompilaci nám program zahlásí chybné nastavení portu. Daleko větším problémem je chybné označení typu desky, program samotný typ desky nerozpozná. Pokud tak provedete, sketch může pracovat špatně, avšak s větší pravděpodobností nebude pracovat vůbec, což bylo několikrát potvrzeno na deskách MEGA a DUEMILANOVE. Oba zadané parametry si můžeme zkontrolovat v pravém dolním rohu okna.

Jakmile máme splněné vše, co bylo napsáno dříve, stiskem druhého tlačítka zleva-UPLOAD v horní liště (detail obr. 21), budete nuceni sketch uložit, následně bude kód nejprve zkompilován a poté odeslán USB kabelem na desku. Pokud chceme nejprve zkontrolovat, zda jsme se nedopustili formální chyby při psaní sketche, stiskem tlačítka VERIFY dojde k ověření sketche a případnému chybovému upozornění. Při experimentování tato funkce nalézá uplatnění, zejména při práci bez samotné desky. Jestliže máme zapojené zařízení se všemi náležitostmi (správný port atd.) UPLOAD před samotným odesláním také provede tuto kontrolu.

Obr. 21: Detail horní lišty.

8 Tvorba a rozbor kódu v MATLAB

Kompletní kód, včetně použitých uživatelských funkcí, je v příloze D. Zde budou okomentovány jeho nejpodstatnější části.

Hlavními body kódu jsou:

1. Příprava proměnných pro příjem dat a otevření Serial portu.
2. Vytvoření měřící smyčky.
3. Uvnitř smyčky vytvořit graf s ukazatelem vzdálenosti pro měření v reálném čase.
4. Uzavření portu.
5. Jednoduchá filtrace extrémně vysokých nesourodých hodnot.
6. Vykreslení filtrovaných dat.

8.1 Příprava proměnných pro příjem dat a otevření Serial portu

```
clear all; close all; clc
y = zeros(1,120);                                    %urceni poctu
mereni(casu)
s1 = serial('COM8');
s1.BaudRate=9600;                                  %musi byt v souladu
s Arduino

%otevreni portu
fopen(s1);
```

Prvním řádkem, vymažeme všechny proměnné, zavřeme všechna okna a vymažeme obsah Command Window. Dále je nadefinován vektor y, který má 1 řádek a 120 sloupců a jehož všechny hodnoty jsou rovny 0. Tento vektor bude později po jednom přepsán na hodnoty získané ze senzoru.

Dalším řádkem vytvoříme seriál objekt na portu COM8, kde je přes USB připojeno přijímající Xbee. U tohoto objektu nadefinujeme Baud Rate na hodnotu 9600, tato hodnota se musí shodovat s Arduino.

Následně je třeba zahájit komunikaci příkazem fopen(), kde jsme na místo argumentu dosadili náš objekt.

Tímto je vše připraveno na tvorbu měřící smyčky.

8.2 Vytvoření měřící smyčky

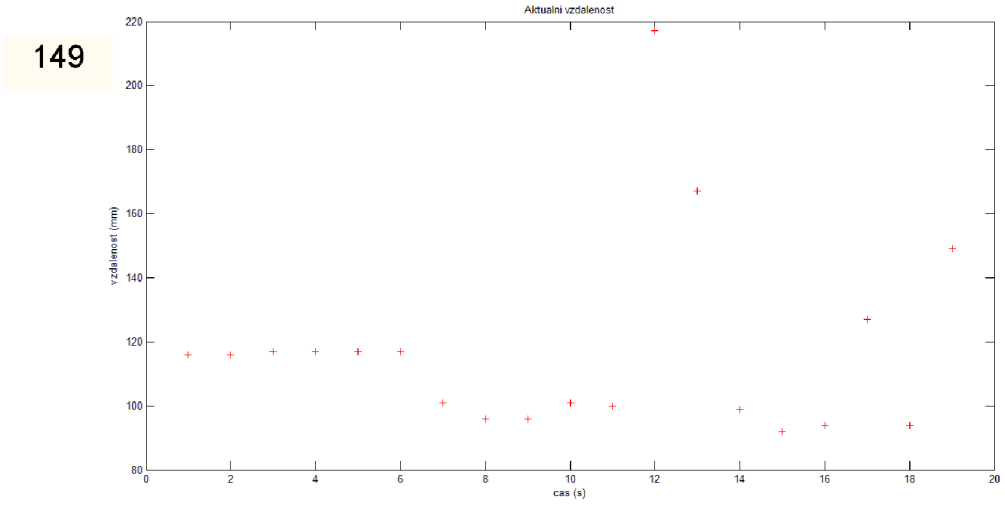
```
for i= 1:120                                          %120 mereni
korespondence s y
    data=fscanf(s1);                              %precteni dat ze senzoru
    y(i) = str2double(data);                        %prevod ze string na
ciselne hodnoty
    h0=figure(1)
```

```
h1=uicontrol('Units','Normalized','Position',[0,0.8,0.1,0.1],'Style'
,'Text',...
    'String',y(i),'Fontsize',30)
    h2=plot(i,y(i),'+r')
    title('Aktualni vzdalenost');
    xlabel('cas (s)');
    ylabel('vzdalenost (mm)');
    drawnow;
    hold on
end
```

Měřící smyčka byla vytvořena za pomocí for cyklu. Pro 120 hodnot, kdy počet hodnot odpovídá času měření v sekundách (vycházející ze sketch), jsou postupně provedeny tyto úkony:

1. Je přečtena jedna hodnota dat za pomocí příkazu fscanf().
2. Tato hodnota je převedena z řetězce na číselnou hodnotu a přiřazena na odpovídající místo vektoru y.
3. V okně figure(1) je vytvořen ukazatel vzdálenosti jako objekt uicontrol s danými parametry.
4. Graf závislosti vzdálenosti na čase je vykreslen, editován a jeho stávající hodnota „podržena“ příkazem hold on (v dalším kroku cyklu je navázáno).

Tento sled instrukcí se opakuje, dokud hodnota *i* nedosáhne vymezeného okraje intervalu. Průběžný graf závislosti vzdálenosti na čase je na obr. 22.



Obr. 22: Pruběžná (real- time) závislost vzdálenosti v čase.

8.3 Uzavření portu

Uzavřením portu je ukončeno získávání dat a je možno přikročit k práci s daty. Toto je realizováno příkazem fclose().

8.4 Jednoduchá filtrace extrémně vysokých nesourodých hodnot

```
%odfiltrovani extremne vysokych hodnot, u kterych je pravdepodobnost
chyby
a=filtr(y);

%urceni intervalu, na kterem chceme vykreslit data
x=1:length(y);
```

Proměnná a bude v sobě uchovávat výstup funkce filtr() se vstupním parametrem *y*. Tato uživatelská funkce byla vytvořena, protože vznikla potřeba odfiltrovat některé extrémní hodnoty, které při měření vznikaly. Při měření s přibližujícím se objektem nevznikaly viditelně chybné hodnoty, avšak když se objekt vzdaloval, byly pozorovány nepříliš časté, ale přesto jasně chybné extrémní hodnoty. Vznik těchto chyb je pravděpodobně spojený s orientaci měřené plochy.

```
function [oprava] = filtr(dat)
%jednoduchy filtr dat
delka=length(dat);
h=zeros(1,delka);

for i=1:(delka-2)
    if (dat(i+1)-dat(i))>(2.5*dat(i+2))
        X=[i,i+2,i+1];
        Y=[dat(i),dat(i+2)];
        h(i+1)=primka(X,Y);

    else
        h(i+1)=dat(i+1);

    end
end
h(delka)=dat(delka);
oprava=h;

end
```

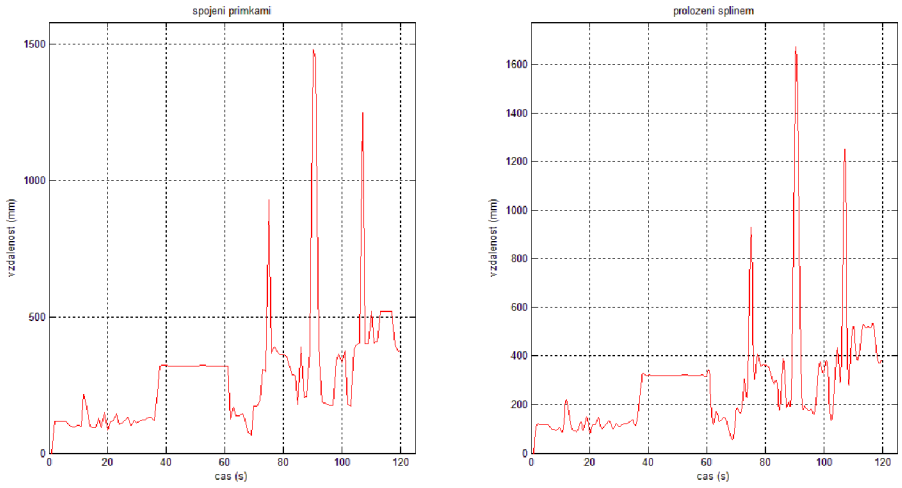
Jednoduchá uživatelská funkce nejprve zjistí počet hodnot vstupní proměnné a na základě této znalosti vytvoří řádkový vektor stejného rozměru. For cyklem jsou zkontrolovány všechny hodnoty kromě první a poslední. První je ponechána na nule, zatímco poslední je ponechána na hodnotě původních dat.

Samotná kontrola probíhá tak, že jsou kontrolovány vždy trojice po sobě jdoucích bodů. Jestliže hodnota rozdílu prvních dvou po sobě jdoucích bodů je větší než 2,5 násobek hodnoty třetího, tedy z dané trojice posledního, bodu, provede se interpolace (pozn. funkce primka() je další vnořená uživatelská funkce). První a poslední bod z trojice jsou pomyslně propojeny přímkou a hodnota chybného bodu posunuta právě na tuto přímku.

Tento filtr má mnoho předpokladů, které je potřeba splnit, aby jeho použití bylo opodstatněné. V opačném případě nás obírá o cenná data.

8.5 Vykreslení filtrovaných dat

Data byla vykreslena dvojím způsobem (obr. 23), prvně byla data pospojována klasicky přímkami na daném intervalu. V druhém sledu jsou stejná data proložena kubickým splinem. Data jsou proložena polynomy vyšších stupňů, při zachování návaznosti těchto polynomů- celá křivka působí mnohem plynuleji.

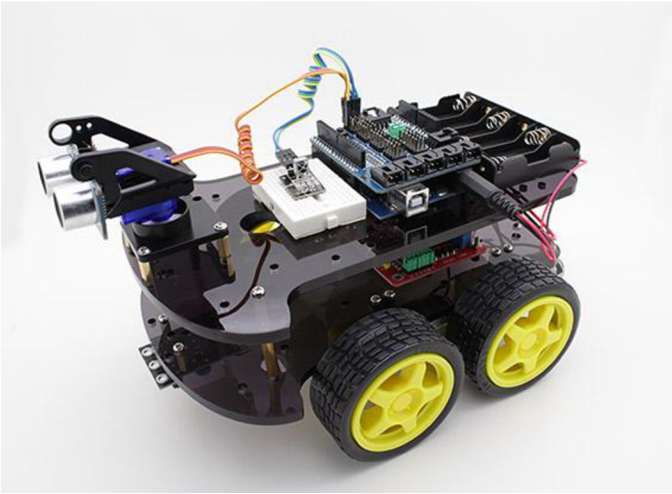


Obr. 23: Vykreslení filtrovaných dat.

9 Praktické využití dané práce

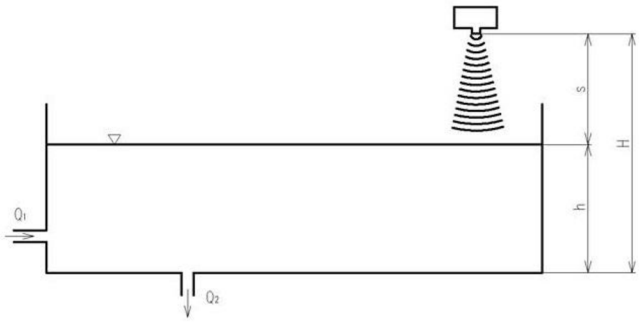
Obecně má měření vzdálenosti ultrazvukem široké uplatnění. Nejznámější aplikace v praxi je použití ultrazvuku v parkovacích senzorech automobilů. Tato práce by se k tomuto účelu dala použít, avšak bylo by třeba jistých úprav. Musela by být přidána větev se zvukovou signalizací a zároveň by bylo nutné zvýšit frekvenci měření tak, aby byly zachyceny rozdíly ve změně vzdálenosti co nejrychleji a řidič tak měl možnost pružně reagovat na relativně rychle se měnící situaci při parkování.

Dalším častým využitím ultrazvukových senzorů je mapování prostoru. Tento druh mapování používají např. autonomní mobilní roboti, kteří jsou schopni se na základě znalosti okolního terénu ze senzorů vyhýbat překážkám. Obvykle je užito buďto více senzorů snímajících v rozdílných směrech nebo jeden uchycený tak že je otáčen servomotorem. Takový robot je na obrázku 24.



Obr. 24: Mobilní robot
s ultrazvukovým senzorem. [38]

V dané konfiguraci by bylo možné použít tento konkrétní měřící obvod na měření výšky hladiny v nádrži, kde by síla přítoku neměla vliv tvar hladiny- tedy by hladina byla téměř rovinná. Je možné, že by obvod pracoval spolehlivě i při rozvíření, avšak to by bylo třeba experimentálně potvrdit. Na obrázku 25 je možného použití s nádrží (příloha E – fotogalerie a schémata).



Obr. 25: Schéma snímání výšky
hladiny v nádrži.

Závěr

Tématem bakalářské práce bylo měření vzdálenosti v reálném čase ultrazvukovým senzorem za použití programovatelné desky Arduino a bezdrátových modulů Xbee. Práce byla rozdělena na část teoretickou a část praktickou.

V teoretické části byly popsány součástky, ze kterých se skládá měřící obvod, a také práce s nimi. Byly popsány všechny nezbytné komponenty uvedených v zadání bakalářské práce.

Praktická část byla věnována konkrétnímu zapojení měřícího okruhu dle připraveného schématu a ultrazvukovým senzorem byla snímána vzdálenost. Naměřená data byla v reálném čase odesílána rádiovými moduly do počítače vybaveného programem MATLAB.

V tomto prostředí byla data, opět v reálném čase, vykreslována. Po ukončení měření byla data odfiltrována od zjevně chybných hodnot a proložena křivkami, tak aby výsledkem byly spojité grafy.

Díky této práci bylo možné vyzkoušet si základní práci se senzory, programování desky i programování v MATLAB, otestovat si funkčnost modulů Xbee s přidruženými součástmi, přičemž si práce zachovává vysokou praktickou využitelnost. Dále je možné porovnat dvě metody vytváření spojitých grafů - vhodnost jejich použití se liší.

Soupis použitých součástí

Počet	Typ	Odkaz
1x	Arduino MEGA 2560	http://www.czechduino.cz/?18,arduino-mega-2560
2x	XBee S2	Nyní už neprodejné
1x	Ultrazvukový senzor HC-SR04	http://www.easyduino.cz/Ultrazvukovy-senzor-HC-SR04-d43.htm
4x	Drát- samec	http://www.gme.cz/prislusenstvi-pro-nepajiva-kontaktni-pole
1x	Nepájivé pole	http://www.gme.cz/nepajiva-kontaktni-pole
1x	USB kabel A- B	http://www.seeedstudio.com/depot/TypeB-USB-cable-for-Arduino-Diecimila-and-Freeduino-p-130.html
1x	XBee USB adaptér	https://www.sparkfun.com/products/8687
1x	USB kabel A – mini	https://www.sparkfun.com/products/11301
1x	Napájecí adaptér	https://www.sparkfun.com/products/298

Seznam použité literatury

[1]	BANZI, Massimo. *Getting started with Arduino*. 2nd ed. Farnham: O'Reilly, 2011. ISBN 978-144-9309-879.
[2]	Introduction to Arduino Part 1. In: *Creativec0d3r* [online]. 2012 [cit. 2014-05-09]. Dostupné z: http://creativec0d3r.blogspot.cz/2012/09/introduction-to-arduino-part-1.html
[3]	MONK, Simon. *30 Arduino projects for the evil genius*. New York: McGraw-Hill, c2010, xiii, 191 p. ISBN 00-717-4133-X.
[4]	OXER, Jonathan. *Practical Arduino: cool projects for open source hardware*. Berkeley, CA: Apress, c2009, xx, 423 p. ISBN 978-1-4302-2477-8.
[5]	LEUNG, Kenneth. Arduino: A Brief History. In: *Ken Leung* [online]. [cit. 2014-05-09]. Dostupné z: http://www.kenleung.ca/_portfolioassets/PDF/HistoryOfArduino_KenLeung.pdf
[6]	ARDUINO. *Arduino* [online]. 2014 [cit. 2014-05-21]. Dostupné z: http://www.arduino.cc/
[7]	GADRE, Dhananjay V. *Programming and customizing the AVR microcontroller* [online]. New York: McGraw-Hill, 2001, xxi, 339 s. [cit. 2014-05-21]. ISBN 00-713-4666-X. Dostupné z: http://1001.kiev.ua/MPT/Programming_and_Customizing_the_AVR_Microcontroller.pdf
[8]	FIALA, Adam. Raspberry Pi místo Apple TV a jiné vychytávky. [online]. 2012 [cit. 2014-05-09]. Dostupné z: http://zpravy.aktualne.cz/ekonomika/technika/raspberry-pi-misto-apple-tv-a-jine-vychytavky/r~i:article:736526/
[9]	Microprocessors Flourish: TMS 1000 4-Bit microcontroller, TI, 1976. *Computer History Museum* [online]. C 2014 [cit. 2014-05-09]. Dostupné z: http://www.computerhistory.org/revolution/digital-logic/12/284/1564
[10]	Microcontroller In Circuit Seriál Programming (ICSP) with Microchip PIC and Atmel AVR. LIRTEX. *Lirtex: TECHNOLOGY ON THE EDGE OF TIME* [online]. 2014 [cit. 2014-05-21]. Dostupné z: http://www.lirtex.com/embedded/microcontroller-icsp-in-circuit-programming/
[11]	Vysvětlení jevu. *Encyklopedie fyziky* [online]. © 2006-2014 [cit. 2014-05-21]. Dostupné z: http://fyzika.jreichl.com/main.article/view/419-vysvetleni-jevu
[12]	MORAVEC, Stanislav. OSCILÁTORY, KRYSTALOVÉ OSCILÁTORY. *SPŠE-slaboproud* [online]. 2001 [cit. 2014-05-21]. Dostupné z: http://slaboproud.sweb.cz/elt2/stranky1/elt045.htm
[13]	CLARK, Scott H. *Osobní počítač*. Vyd. 1. Brno: Computer Press, 2004, xiii, 235 s. Jak je to snadné!. ISBN 80-251-0145-2.
[14]	*Techfan* [online]. 2013 [cit. 2014-05-21]. Dostupné z: http://techfan.gr/2013/12/%CE%B7-%CE%B5%CF%80%CF%8C%CE%BC%CE%B5%CE%BD%CE%B7-usb-type-c-%CF%85%CF%80%CE%BF%CE%B4%CE%BF%CF%87%CE%AE-%CE%B8%CE%B1-%CF%80%CF%81%CE%BF%CF%83%CF%86%CE%AD%CF%81%CE%B5%CE%B9-%CE%B4%CE%AF%CF%80%CE%BB/
[15]	*Gumbolabs* [online]. 2014 [cit. 2014-05-24]. Dostupné z: www.gumbolabs.org
[16]	EARL, Bill. Memories of an Arduino. *Adafruit* [online]. 2.8.2013, 22.5.2014 [cit. 2014-05-22]. Dostupné z: https://learn.adafruit.com/memories-of-an-arduino
[17]	512Kb SRAM expansion for the Arduino Mega (build). *Andy's workshop* [online]. 2011 [cit. 2014-05-24]. Dostupné z: http://andybrown.me.uk/wk/2011/08/28/512kb-sram-expansion-for-the-arduino-mega-build/
[18]	*Root.cz: informace nejen ze světa Linuxu* [online]. 2014 [cit. 2014-05-23]. ISSN 1212-8309. Dostupné z: http://www.root.cz/n/arduino/

[19]	*Processing* [online]. 2004 [cit. 2014-05-23]. Dostupné z: http://processing.org/
[20]	Processing book. In: *Abstractmachine* [online]. 2014 [cit. 2014-05-23]. Dostupné z: http://www.abstractmachine.net/blog/processing-book/
[21]	ZigBee: novinka na poli bezdrátové komunikace. In: *Hw.cz: vše o elektronice a programování* [online]. © 1997 – 2014 [cit. 2014-05-23]. Dostupné z: http://www.hw.cz/navrh-obvodu/rozhrani/zigbee-novinka-na-poli-bezdratove-komunikace.html
[22]	Xbee Buying Guide. *SparkFun* [online]. 2009 [cit. 2014-05-23]. Dostupné z: https://www.sparkfun.com/pages/xbee_guide
[23]	Wireless Sensor Networking. *Digi* [online]. © 1996-2014 [cit. 2014-05-23]. Dostupné z: http://www.digi.com/technology/drop-in-networking/wireless-sensor-networking
[24]	FALUDI, Robert. *Building wireless sensor 46ouš46rk*. 1st ed. Beijing: O´Reilly, 2010, xviii, 300 s. ISBN 978-0-596-80773-3.
[25]	Sensor De Distancia Ultrasonico Hc-sr04. *Mercado libre* [online]. © 1999-2014 [cit. 2014-05-23]. Dostupné z: http://articulo.mercadolibre.com.co/MCO-410101265-sensor-de-distancia-ultrasonico-hc-sr04-_JM
[26]	IT'S THE SAME XBOX 360, BUT IT'S ALL SHINY AND STUFF!. In: *Giant Bomb* [online]. 2011 [cit. 2014-05-24]. Dostupné z: http://www.giantbomb.com/xbox-360/3045-20/forums/its-the-same-xbox-360-but-its-all-shiny-and-stuff-430615/
[27]	OBRAZ, Jaroslav. *Zkoušení materiálu ultrazvukem*. Vyd. 1. Praha: SNTL, 1989, 460 s. Redakce strojírenské a metalurgické literatury. ISBN 80-030-0097-1.
[28]	*Ultrazvuk a jeho použití v praxi*. 2. 46ouš.vyd. Praha: Elektrotechnický svaz českomoravský, 1945, 167 s.
[29]	Kmitání a vlnění. In: JANDORA, Radek. *Radek Jandora sweb* [online]. 2008 [cit. 2014-04-24]. Dostupné z: radek.jandora.sweb.cz
[30]	Matlab. HUMUSOFT. *Humusoft: Technické výpočty, řídící technika, simulace* [online]. © 1991 – 2014 [cit. 2014-05-23]. Dostupné z: http://www.humusoft.cz/produkty/matlab/matlab/
[31]	KARBAN, Pavel. *Výpočty a simulace v programech Matlab a Simulink*. Vyd. 1. Brno: Computer Press, 2006, 220 s. ISBN 80-251-1301-9.
[32]	Matlab. In: *Wikipedia* [online]. 2008 [cit. 2014-05-23]. Dostupné z: http://de.wikipedia.org/wiki/Matlab
[33]	Formatting Your Matlab Program With Cells. In: *Dartmouth* [online]. © 2014 [cit. 2014-05-23]. Dostupné z: https://www.dartmouth.edu/~rc/classes/intro_matlab/Format_Program_Cells.html
[34]	Acer TravelMate B113-E-987B4G32akk (UK). In: *Hardware.info* [online]. © 1998-2014 [cit. 2014-05-24]. Dostupné z: http://uk.hardware.info/productinfo/178107/acer-travelmate-b113-e-987b4g32akk-uk
[35]	Xbee Shield. In: *Arduino* [online]. 2014 [cit. 2014-05-24]. Dostupné z: http://store.arduino.cc/index.php?main_page=product_info&cPath=37&products_id=4
[36]	DFRobot - XBee Explorer USB. *RoboSavvy* [online]. 2014 [cit. 2014-05-24]. Dostupné z: http://robosavvy.com/store/product_info.php/cPath/25/products_id/984
[37]	HALLIDAY, David. *Fyzika: Vysokoškolská učebnice obecné fyziky*. 1. vyd. Brno/Praha: VUTIUM/PROMETHEUS, 2000, 1198 s. ISBN 80-214-1869-9.
[38]	Arduino 4WD Ultrasonic Robot Kit. In: *Hobby King: The ultimate hobby experience* [online]. 2014 [cit. 2014-05-24]. Dostupné z:https://www.hobbyking.com/hobbyking/store/__37891__Arduino_4WD_Ultrasonic_Robot_Kit.html

Seznam příloh

Příloha	**Obsah**
Příloha A	Arduino pinout
Příloha B	Specifikace HC – SR04
Příloha C	Arduino sketch
Příloha D	MATLAB kódy
Příloha E	Fotografie, grafy
CD	Elektronická podoba práce, programy