

Ovládací panel s vývojovým kitem Arduino

Bc. Milan Kadlec

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2013/2014

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Milan KADLEC**
Osobní číslo: **A11346**
Studijní program: **N3902 Inženýrská informatika**
Studijní obor: **Bezpečnostní technologie, systémy a management**
Forma studia: **kombinovaná**

Téma práce: **Ovládací panel s vývojovým kitem Arduino**

Téma anglicky: **An Arduino-based Control Panel**

Zásady pro vypracování:

1. Seznamte se s historií a vývojem projektu Arduino.
2. Popište konkrétní kit Arduino s Ethernet shieldem a vývojové prostředí.
3. Seznamte se s programovacím jazykem vývojového kitu Arduino.
4. Navrhněte a demonstруйте pomocí vývojového kitu Arduino a Ethernet shieldu webový server, přes který lze ovládat připojené periferie a získávat data z programovatelných pinů vývojového kitu Arduino.
5. Navrhované řešení otestujte a proveďte závěrečné zhodnocení daného projektu.

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. BANZI, Massimo. Getting Started with Arduino. Second Edition. U.S.A.: O'Reilly Media, 2011. ISBN 978-1-449-30987-9.
2. MATOUŠEK, David. Práce s mikrokontroléry ATMEL AVR. 2. vyd. Praha: BEN, 2006, 375 s. ISBN 80-7300-209-4.
3. PINKER, Jiří. Mikroprocesory a mikropočítače. Praha: BEN – technická literatura, 2004, 159 s. ISBN 80-730-0110-1.
4. ARDUINO [online]. Dostupné z: <http://www.arduino.cc>
5. CADY, Fredrick M. Microcontrollers and microcomputers: principles of software and hardware engineering. 2nd ed. New York: Oxford University Press, 2010, xii, 477 s. ISBN 978-0-19-537161-1.
6. Technical Devices for Supervising of a Household via Internet Based on Arduino Microcontroller. Home Page. 2012. Dostupné z: <http://www.wseas.us/e-library/conferences/2012/Istanbul/AICBE/AICBE-40.pdf>
7. BUMBA, Jiří. Programování mikroprocesorů: praktický návod nejen pro mikroprocesory PIC. Vyd. 1. Brno: Computer Press, 2011, 135 s. ISBN 978-80-251-2838-1.
8. VÁŇA, Vladimír. ARM pro začátečníky. 1. vyd. Praha: BEN-Technická literatura, 2009. ISBN 978-80-7300-2.

Vedoucí diplomové práce:

Ing. Martin Pospíšilík

Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce:

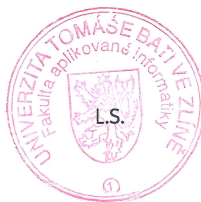
7. února 2014

Termín odevzdání diplomové práce:

27. května 2014

Ve Zlíně dne 7. února 2014

prof. Ing. Vladimír Vašek, CSc.
děkan



doc. RNDr. Vojtěch Křesálek, CSc.

ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

ABSTRAKT

Cílem diplomové práce je možnost využití vývojového kitu Arduino jako webového serveru. V teoretické části jsou shrnuty poznatky získané studií odborné literatury zabývající se základními pojmy, historií a vývojem mikropočítačů. A dále seznámení se s projektem Arduino od vývojových desek, přes doplňkové desky až k vývojovému prostředí a programovacího jazyka.

V praktické části je navrženo konkrétní řešení webového serveru pomocí vývojového kitu Arduino Mega 2560 a Ethernet shieldu. Popsány jsou použité komponenty včetně jejich elektronického zapojení. Přes webový prohlížeč lze ovládat připojené periferie či získávat data z programovatelných pinů. Navrhované řešení je fyzicky zapojeno a odzkoušeno. V závěru práce je uvedeno zhodnocení celého projektu.

Klíčová slova: Arduino, mikrokontrolér, Ethernet

ABSTRACT

The aim of this thesis is a possibility of using the Arduino development kit as a Web server. The theoretical part includes a literature search focused on basic terms, history and development of microcomputers. A further presents with the project from the Arduino development boards through additional boards to the development environment and programming language.

In the practical part is designed the specific Web server solution using a development kit Arduino Mega 2560 and the Ethernet Shield. There are described components including their electrical wiring. Through a web browser is possible to control connected peripherals or obtain data from programmable pins. The proposed solution is physically connected and tested. The project is appraised in the end of the thesis.

Keywords: Arduino, microcontroller, Ethernet

Tímto bych chtěl poděkovat panu Ing. Martinu Pospíšilíkovi, Ph.D., za jeho věnovaný čas, cenné rady a věcné připomínky.

Dále bych také velmi rád poděkoval mojí rodině, která mě po celou dobu studia podporovala a byla mi velkou oporou.

OBSAH

ABSTRAKT	5
ABSTRACT	5
ÚVOD	9
I TEORETICKÁ ČÁST	10
1 ZÁKLADNÍ POJMY	11
1.1 MIKROPROCESOR (MPU)	11
1.2 JEDNOČIPOVÝ MIKROPOČÍTAČ	11
1.3 MIKROKONTROLÉR (MCU)	12
1.4 DIGITÁLNÍ SIGNÁLOVÝ PROCESOR (DSP)	13
1.5 DIGITÁLNÍ SIGNÁLOVÝ KONTROLÉR (DSC)	13
2 MIKROPOČÍTAČE	14
2.1 HISTORIE MIKROPOČÍTAČŮ	14
2.2 ROZDĚLENÍ MIKROPOČÍTAČŮ	15
2.3 INSTRUKČNÍ SOUBORY	15
2.3.1 CISC.....	15
2.3.2 RISC.....	16
2.4 KONCEPCE MIKROPROCESORŮ	17
2.4.1 Von Neumannova koncepce.....	17
2.4.2 Harvardská koncepce.....	17
3 ARDUINO	18
3.1 HISTORIE ARDUINA	18
3.2 HARDWARE ARDUINA	19
3.2.1 Vývojová deska Arduina.....	19
3.2.2 Doplnkové desky - shieldy.....	20
3.3 VÝVOJOVÉ PROSTŘEDÍ	21
3.3.1 Instalace a nastavení Arduino IDE.....	21
3.3.2 Arduina IDE.....	22
3.3.3 Programovací jazyk Wiring.....	23
4 PROGRAMOVÁNÍ ARDUINA	26
4.1 HARDWAROVÉ VYBAVENÍ	26
4.2 SOFTWAREOVÉ VYBAVENÍ	28
II PRAKTICKÁ ČÁST	29
5 POUŽITÉ KOMPONENTY	30
5.1 ARDUINO MEGA 2560	30
5.1.1 Specifikace Arduina Mega 2560.....	32

5.2	ETHERNET SHIELD	33
5.2.1	Specifikace Ethernet shieldu.....	34
5.3	LED BARGRAF	34
5.4	TRIMRY	35
5.5	MIKROSPÍNAČE	36
5.6	ZKUŠEBNÍ NEPÁJIVÉ KONTAKTNÍ POLE	37
6	POPIS ELEKTRONICKÉHO ZAPOJENÍ.....	38
6.1	ZAPOJENÍ JEDNOTLIVÝCH SOUČÁSTEK	38
6.2	VÝPOČET VSTUPNÍCH A VÝSTUPNÍCH OKRUHŮ ARDUINA	39
6.3	SCHÉMA ZAPOJENÍ	40
7	NÁVRH PROGRAMU.....	41
7.1	ANALÝZA ZADÁNÍ A NÁVRH ŘEŠENÍ	41
7.2	NÁVRH WEBU	41
7.3	VÝVOJOVÝ DIAGRAM	42
7.4	VYTVOŘENÍ ZDROJOVÉHO KÓDU	44
8	TESTOVÁNÍ WEBOVÉHO SERVERU.....	47
	ZÁVĚR.....	48
	ZÁVĚR V ANGLIČTINĚ.....	50
	SEZNAM POUŽITÉ LITERATURY.....	51
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	54
	SEZNAM OBRÁZKŮ.....	55
	SEZNAM TABULEK.....	56
	SEZNAM PŘÍLOH.....	57

ÚVOD

Vývoj mikropočítačů či mikrokontrolérů jde neustále mílovými kroky vpřed. Při zachování svých malých rozměrů, jsou stále výkonnější a opatřeny čím dál lepšími vlastnostmi. To vede k tomu, že se již dnes můžeme setkat s mikrokontrolérem téměř na každém kroku. Mikrokontroléry jsou použity nejen v domácích spotřebičích, automobilech, ale také v nej-různějších průmyslových zařízeních. Zkrátka tam, kde je zapotřebí získávat a vyhodnocovat data z různých periférií a následně na tyto změny patřičně reagovat.

Pro pochopení této problematiky je dobré si ujasnit základní pojmy jako je mikropočítač, mikrokontrolér, mikroprocesor, digitální signálový procesor či digitální signálový kontrolér. Dále pak získat základní znalosti mikropočítače včetně jejich rozdělení podle koncepce a instrukčních souborů.

Diplomová práce se zaměřuje na vývojový kit Arduino, který bude plnit funkci webového serveru, přes který bude možné ovládat a získávat data z připojených periférií. Pro připojení vývojové desky Arduino k Ethernetu je zapotřebí rozšířit Arduino o doplňkovou desku zvanou Ethernet shield. Tato deska zprostředkovává komunikaci s Ethernetovým prostředím. Po vytvoření webového serveru se bude k Arduino přistupovat přes webové stránky, které budou zobrazovat aktuální hodnoty naprogramovaných analogových a digitálních pinů nebo bude prostřednictvím webu možné ovládat dané digitální piny. K vývojové desce Arduino jsou připojeny mikrosvítače, trimry a LED diody zabudované v bargrafu. Mikrosvítače jsou připojeny k digitálním pinům, které jsou naprogramovány jako vstupní piny. Při stisku svítače se změní stav (rozsvítí/zhasne) patřičné LED diody, která je ovládána přes digitální pin Arduina. Použité trimry, jež jsou připojeny k vývojové desce přes analogové piny, mohou v našem případě simulovat měření fyzikálních veličin jako je teplota, vlhkost, světelný tok atd.

V případě použití tohoto jednoduchého webového serveru je nutné mít na paměti, že komunikace mezi klientem a serverem není nijak zabezpečena, tudíž může docházet k neoprávněným přístupům na webový server či posílání falešných požadavků, které mohou negativně ovlivnit chod celého systému. Proto je vhodné tento webový server používat výhradně v Ethernetové síti.

I. TEORETICKÁ ČÁST

1 Základní pojmy

Často se pojmy jako je mikroprocesor, mikropočítač či mikrokontrolér nerozlišují a nazývají se jako jeden a tentýž integrovaný obvod, který zpracovává číslicové signály. Hlavním rozdíl uvedených pojmů tkví v tom, že mikroprocesor je jako centrální řídicí jednotka (CPU) součástí mikropočítače či mikrokontroléru, ve kterém provádí jednotlivé výpočty. Připojíme-li k mikroprocesoru pomocné obvody jako je vstupně-výstupní rozhraní, paměti dat a další periferie, nazýváme zařízení jako mikropočítač. Pokud všechny části, které obsahuje mikropočítač jsou zahrnuty v jediném integrovaném obvodu (IO), nazývá se takový IO jako jednočipový mikropočítač. Obsahuje-li jednočipový mikropočítač navíc i pomocné obvody jedná se již o tzv. mikrokontrolér. (Frýza, 2013)

1.1 Mikroprocesor (MPU)

Mikroprocesor (MPU - Microprocessor Unit) slouží jako centrální řídicí jednotka (CPU - Central Processing Unit), která je hlavní součástí každého mikropočítače. Jedná se o složitý obvod složený z milionů tranzistorů, které vykonávají matematické operace. Hlavní funkce mikroprocesoru spočívá ve:

- vykonávání instrukcí obsažených ve zdrojovém kódu pomocí aritmetických a logických operací
- řízení toku dat mezi vstupně-výstupních obvodů a mikropočítačem
- zpracování dat v paměti

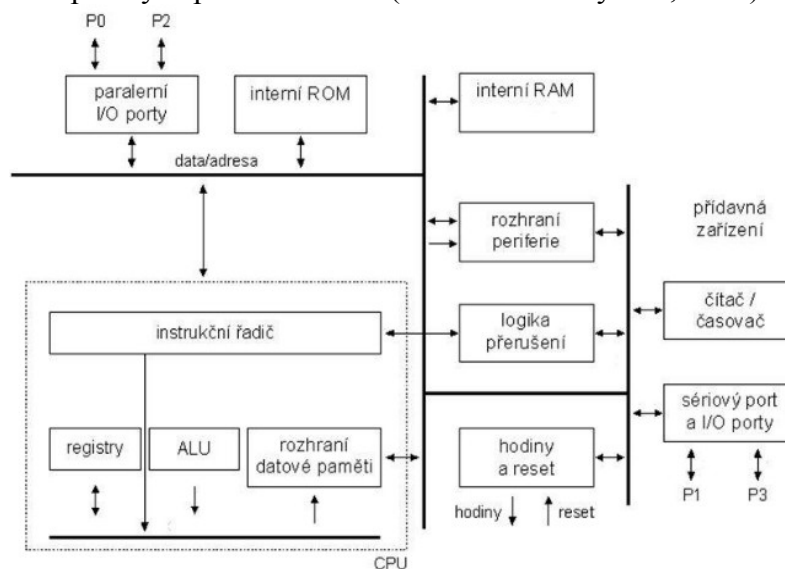
Přičemž každý mikroprocesor je složen ze základních součástí:

- ALU (Arithmetic-Logic Unit) - aritmetická a logická jednotka, která zajišťuje provádění aritmetické a logické operace s daty
- Řadič – řídí komunikaci v procesoru při vykonávání strojových instrukcí
- Registry – paměť dat (Mikrokontroléry PIC, 2013)

1.2 Jednočipový mikropočítač

Jednočipový mikropočítač je monolitický integrovaný obvod viz.obr.1, který se díky svým vlastnostem podobá menšímu univerzálnímu počítači. Pro správnou funkci a soběstačnost se takto sestavený mikropočítač skládá z:

- mikroprocesoru
- operační paměti pro paměť dat (RAM)
- paměti pro uložení programu (EEPROM, FLASH či ROM)
- obvodů vstupně výstupního rozhraní (Mikrokontroléry PIC, 2013)



Obr. 1 Obecné blokové schéma 8bitového jednočipového mikročítače

Zdroj: LabTF, © 2013

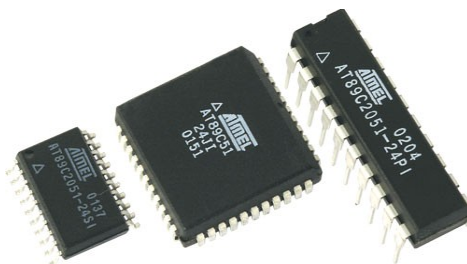
1.3 Mikrokontrolér (MCU)

Jedná se o jednočipový mikročítač, který má v sobě integrovány obdobné obvody stejně jako jednočipový mikročítač s tím rozdílem, že je dále doplněn o další pomocné obvody:

- analogově-digitální a digitálně-analogový převodník
- komparátory a modulátory (PWM – pulzně-šířková modulace)
- dále pak různé řadiče a časovače

Ve srovnání s jednočipovými mikročítači, obsahuje mikrokontrolér méně vstupů a výstupů, což se projevuje nižšími náklady potřebnými k řešení realizace externích obvodů. Z tohoto důvodu je také mikrokontrolér rozměrově menší a zároveň také levnější. Díky těmto pozitivním vlastnostem a vysoké míře spolehlivosti se mikrokontroléry používají zejména pro jednoúčelové operace. Mezi jednoúčelové operace se řadí monitorování, vyhodnocování, zobrazování, řízení, ovládání, regulace různých mechanismů a procesů, které vykonávají předem definovanou činnost vestavěného systému (tzv. embedded systému). (Mikrokontroléry PIC, 2013)

Obr. 2 Mikrokontrolér



Zdroj: Transfer Multisort Elektronik, © 2014

1.4 Digitální signálový procesor (DSP)

Architektura digitálního signálového procesoru je uzpůsobena pro rychlé vykonávání algoritmů na zpracování číslicových signálů. Tato struktura vyniká rychlejším a dokonalejším zpracováním dat. Pro rychlé zpracování dat jsou v obvodu této struktury procesoru použity:

- hardwarový násobič
- duální přístupy do paměti
- vícenásobný datový a adresový sběrnice
- víceúrovňový pipelining

Digitální signálový procesor neobsahuje paměť ani analogově-digitální a digitálně-analogový převodník. V praxi tento typ architektury nachází uplatnění ve zpracování zvuků či obrazů atd. (Nevoral, 2008)

1.5 Digitální signálový kontrolér (DSC)

Jde stejně jako u mikrokontroléru o jednočipový mikropočítač, jež je doplněn o přidané podpůrné obvody. Značnou předností digitálního signálového kontroléru vzhledem k mikrokontroléru je výkonný digitální signální procesor, který slouží pro zpracování číslicových signálů. Uvedený typ kontroléru je hojně využit k řízení nejrůznějších elektrických pohonů. (Elektrorevue, 06/2010, s. 76)

2 Mikropočítače

V současné době se pojmem mikropočítač rozumí počítač, který je rozměrově podstatně menší než klasický počítač, přičemž koncepce zůstává zachována. Jedná se o nejrozšířenější typ počítače na světě, jehož využití je velmi široké. Oblíbenost uvedeného typu spočívá zejména díky jeho malým rozměrům, dostupnosti na trhu, spolehlivosti a také ceně. Každý mikropočítač se skládá z:

- mikroprocesoru
- paměti pro data a program
- vstupně-výstupní obvody

2.1 Historie mikropočítačů

V průběhu let došlo v oblasti mikropočítačů k výrazným změnám. Největší změnou je tzv. příchod třetí generace počítačů, která nastala v první polovině 60. let minulého století a trvala až do 80. let minulého století. Díky výrazné změně se pojem mikropočítač a minipočítač začínal postupně vryvat do podvědomí lidí. V této době zabíraly klasické počítače celé místnosti a pojem mikropočítač byl označován pro zmenšený model, jež mohl obsluhovat jeden člověk.

První mikroprocesor uvedla firma Intel v roce 1971 pod označením Intel 4004. Jednalo se o předchůdce dnešních mikroprocesorů. Ve stejném roce vyvinula společnost TEXAS INSTRUMENTS první mikrokontrolér s označením TMS 1000, který byl uveden do prodeje o 3 roky později. Firma Intel zanedlouho na to vyvinula svůj vlastní mikrokontrolér 8048. V 1974 byl uveden první komerční mikropočítač společností Intel. Jednalo se spíše o stavebnici, která se prodávala pod obchodní značkou Altair 8800 s mikroprocesorem Intel 8080. V rámci konkurenčního boje se na trhu objevila společnost Apple se svým mikropočítačem Apple II, dále následovala firma Commodore International s Commodore 64 (C64). (Augarten, 1983)

Vlastnosti dnešních mikropočítačů zůstávají zachovány, jsou spojeny v jediném integrovaném obvodu, který je označen jako jednočipový mikropočítač. (Switch2Mac, 2010)

2.2 Rozdělení mikropočítačů

Mikropočítače můžeme rozdělit podle nejrůznějších kritérií. Nejčastěji jsou mikropočítače děleny podle použitého instrukčního souboru (CISC a RISC struktury) a podle použitého procesoru v mikropočítači (Harvardská koncepce a Von Neumannova koncepce procesoru).

2.3 Instrukční soubory

Na začátku padesátých a šedesátých let minulého století, kdy se datují začátky vývoje počítačů, používal každý výrobce vlastní technologii. V průběhu let došlo k výrazným odlišnostem a inovacím jednotlivých technologií. Tím postupně stále více vznikal problém, protože program byl přesně napsán pro určitý typ počítače a nemohl být přenesen na jiný. Data musela být složitě konvertována do jiných formátů, což vedlo nejen k časové ale také finanční náročnosti. Proto začala koncem šedesátých let výpočetní technika dostávat určité pravidla nejen pro software ale i hardware, aby bylo uvedeným komplikacím včas předcházeno. Díky vzniku pravidel bylo dosaženo zvýšení výkonnosti a nižší ceny počítačů. Konkrétně tak u procesorů a dále pak u mikroprocesorů vznikla architektura nazývaná po označení CISC, později přibyla architektura RISC. Z těchto architektur vznikly hybridy jako EPIC a VLIW. (Root.cz, © 1998 – 2014)

2.3.1 CISC

Z anglického názvu Complex Instruction Set Computer je patrné, že se jedná o rozsáhlou sadu instrukcí, která má proměnou délku. Některé instrukce jsou prováděny s více funkcemi najednou, čím se snižuje efektivní využití výkonu procesoru.

Hlavní výhody a nevýhody CISC architektury :

- vysoká spotřeba energie
- vhodnější pro vysokoúrovňové jazyky
- komplexní návod omezuje využití paměti
- komplexní návod může trvat jako jediný krok k dokončení
- instrukce mohou být dokončena v mnoha hodinových cyklech
- nákladná architektura

- instrukce se liší ve velikosti
- méně prostor pro registry
- je požadováno více tranzistorů (Ozden, © 2013)

2.3.2 RISC

RISC obsahují pouze jednoduchou (redukovanou) sadu instrukcí s pevně danou délkou (nejčastěji 32bitové instrukce). Provádění složitějších operací je podstatně rychlejší než u procesoru s instrukční sadou CISC. Negativní vlastností této struktury je nutnost větší paměti k uložení programu.

Výhody a nevýhody RICS architektury :

- efektivní, rychlá, levná architektura
- krátká doba pro provedení
- nízká spotřeba energie
- pokyny jsou stejné délky , takže to může být provedena v jednom hodinovém cyklu
- vzhledem k tomu, že instrukce jsou vyplněny v jednom hodinovém cyklu , což umožňuje procesoru provádět více úkolů najednou
- vyšší rychlosti hodin
- velký počet registrů
- potřebuje pro přístup k paměti pouze pro zatížení a uložení pokynů
- pevné pokyny délka
- komplexní instrukce mohou dokončit v mnoha krocích , a to může zvýšit velikost kódování
- méně tranzistory potřeby (Ozden, © 2013)

2.4 Koncepce mikroprocesorů

Konceptí mikroprocesorů je myšleno hardwarové propojení řadiče a aritmeticko-logické jednotky s pamětí obsahující data a program. V odborné literatuře se setkáváme se dvěma základními koncepcemi a to Von Neumannovu a Harvardskou koncepcí mikroprocesoru. Dodnes se můžeme setkat s oběma typy, i když je již uvedené rozdělení u moderních procesorů poněkud zavádějící. Dnešní procesory jsou částečně sestaveny z obou koncepcí. Větší podíl u mikrokontrolérů je převládající v Harvardské architektuře.

2.4.1 Von Neumannova koncepce

Von Neumannova koncepce popisuje vnitřní propojení počítače s jediným paměťovým prostorem, který obsahuje jak data, tak i samotný program. Komunikace mezi pamětí a procesorem (řadičem a aritmeticko-logickou jednotkou) probíhá po jedné sběrnici. Nevýhodou této architektury je fakt, že nelze současně pracovat s programem a daty, což se projevuje ve zpomalené komunikaci s pamětí. Velkou hrozbu této koncepce představuje přepsání programu daty, protože jsou uloženy pouze v jedné paměti. Výhoda ovšem spočívá v možnosti libovolné modifikace programu nebo dat. Von Neumannova koncepce bývá využívána u osobních počítačů, mikropočítačů či mobilních telefonech.

2.4.2 Harvardská koncepce

První použití této koncepce bylo u počítače Harvard Mark I, jež nese nynější název koncepce. Tato architektura nevyužívá pouze jednu paměť (data + program), ale dvě paměti. Jedna je aplikována čistě na program a druhá je přizpůsobena na data. Obě paměti komunikují s procesorem paralelně po své sběrnici, čímž se u Harvardské architektury dosahuje rychlejší komunikace. Další výhodou koncepce tkví v použití různých druhů a parametrů paměti (RAM, ROM, EPROM, FLASH), jež stále užívá digitální signálové procesory (DSP) a mikrokontroléry.

3 Arduino

Arduino je projekt, který jako celek obsahuje hardware, software a komunitu. Oficiální webová stránka je veřejně dostupná na internetové stránce <http://www.arduino.cc>. V uvedeném odkazu lze nalézt veškeré informace, které se uvedeného projektu týkají. Například si zde lze stáhnout informace o vývojovém prostředí, knihovnu, detailní popis jednotlivých produktů, ukázky s různými příklady programování, popis příkazů, podpora-forum a další důležité informace.). Arduino je založen na open-source platformě. Otevřený zdrojový kód i volně dostupné schémata všech oficiálních modelů využívají zajímavě řešený hardware, který pracuje na mikrokontrolerech firmy Atmel s kombinací jednoduchého grafického vývojového prostředí „Arduino IDE“ (Integrated Development Environment).

Americká nezisková organizace Creative Commons Attribution Share Alike (CC-BY-SA) na základě licence umožňuje uživatelům upravovat, přestavovat, rozšiřovat či vylepšovat software i hardware. Je zde ovšem stanovena podmínka, kdy všechny takto provedené změny musí být také licencovány jako Share Alike (SA) a musí se uvádět jako autor Attribution (BY). Na základě splnění výše uvedených podmínek lze takto upravené kopie dále prodávat. Díky této změně se platforma velmi rychle rozšířila a nabrala ohromné oblibě hlavně mezi domácí kutily. Mezi nejznámější kompatibilní upravené kity Arduina patří FreeDui-no, Seeeduino, Boarduino či Bare Bones Boards (BBB). (Lupa.cz, © 1998 – 2014)

3.1 Historie Arduina

Projektu Arduino vznikl v roce 2005 na bývalé vysoké škole „Interaction Design Institute Ivrea“ v Itálii. Za zakladatele jsou považováni Massimo Banzi a David Cuartielles, jejichž myšlenka vedla k vytvoření jednoduché vývojové prostředí a hardware. Dle různých literárních zdrojů je pojmenování projektu odvozeno dle významné historické postavy Arduin of Ivrea či název vznikl dle názvu pro baru, ve kterém se po škole scházeli studenti a zaměstnanci uvedené univerzity. (Core77.com, © 2014)

Po velkém úspěchu začali postupem času vznikat další a další verze jako Uno, Leonardo, Due, Mega2560, Yún, Tre. Již pouhý rok od vzniku získal projekt významné ocenění Prix Ars Electronica 2006, který je jedním z nejvýznamnějších ocenění v oblasti elektroniky, interaktivního umění, počítačové animace, digitální kultury a hudby. Výrobu všech originálních výrobků Arduino zajišťuje Italská firma SmartProjects, která rovněž vlastní ochrannou známku Arduino. (Root.cz, © 1998 – 2014)

3.2 Hardware Arduina

Základem celého vývojového kitu je vývojová deska Arduina tzv. board, který pracuje na mikrokontroléru z řady Atmega, jež byl vyvinut norskou firmou Atmel. Na vývojovou desku Arduino lze připojit různé doplňkové desky nazývané jako shieldy. Tyto shieldy rozšiřují Arduino o další funkce, které např. dokáží zprostředkovávat komunikaci mezi Arduinem a jiným systémem (ethernet, WiFi, GSM, XBee). V případě využití průchozích pinů se může na sebe „naskládat“ na sebe několik desek a tím využít další možnosti Arduina. Dále pak lze k desce připojit různé příslušenství jako jsou LCD displeje, ovládací prvky, USB převodníky atd.

3.2.1 Vývojová deska Arduina

Srdcem celé vývojové desky je mikrokontrolér typu RISC, který pracuje na harvardské koncepci. Skládá se z CPU pracující nejčastěji na frekvenci 8MHz nebo 16MHz, vnitřní paměti FLASH, RAM a EEPROM, časovače, A/D převodníky, paralelní a sériové rozhraní (UART), analogové a digitální piny viz Tab.1. Dále na desce najdeme USB port s převodníkem (u starších verzí seriový port) pro komunikaci s počítačem, napájecí 12V konektor se stabilizátory napětí (3,3V a 5V), signalizační diody, resetovací tlačítko, vstupně-výstupní piny a ICSP konektor (pro přímé programování mikrokontroléru).

Tab. 1. Druhy Arduino desek

Název	Mikrokontrolér	Frekvence CPU	Analog In/Out	Digital IO/PWM	EEPROM	SRAM [KB]	Flash [KB]	UART
Uno	ATmega328	16 Mhz	6/0	14/6	1	2	32	1
Due	AT91SAM3X8E	84 Mhz	12/2	54/12	-	96	512	4
Leonardo	ATmega32u4	16 Mhz	12/0	20/7	1	2.5	32	1
Mega 2560	ATmega2560	16 Mhz	16/0	54/15	4	8	256	4
Mega ADK	ATmega2560	16 Mhz	16/0	54/15	4	8	256	4
Micro	ATmega32u4	16 Mhz	12/0	20/7	1	2.5	32	1
Mini	ATmega328	16 Mhz	8/0	14/6	1	2	32	-
Nano	ATmega168	16 Mhz	8/0	14/6	0.512	1	16	1
Nano	ATmega328	16 Mhz	8/0	14/6	1	2	32	1
Ethernet	ATmega328	16 Mhz	6/0	14/4	1	2	32	-
Esplora	ATmega32u4	16 Mhz	-	-	1	2.5	32	-
ArduinoBT	ATmega328	16 Mhz	6/0	14/6	1	2	32	1
Fio	ATmega328P	8 Mhz	8/0	14/6	1	2	32	1
Pro (168)	ATmega168	8 Mhz	6/0	14/6	0.512	1	16	1
Pro (328)	ATmega328	16 Mhz	6/0	14/6	1	2	32	1
Pro Mini	ATmega168	8(16) Mhz	6/0	14/6	0.512	1	16	1
LilyPad	ATmega168V	8 Mhz	6/0	14/6	0.512	1	16	-
LilyPad	ATmega328V	8 Mhz	6/0	14/6	0.512	1	16	-
LilyPad USB	ATmega32u4	8 Mhz	4/0	9/4	1	2.5	32	-
LilyPad Simple	ATmega328	8 Mhz	4/0	9/4	1	2	32	-
LilyPad SimpleSnap	ATmega328	8 Mhz	4/0	9/4	1	2	32	-

Zdroj: Arduino, © 2014

3.2.2 Doplnkové desky - shieldy

Shieldy, doplnkové desky neboli nástavby, se využívají k rozšíření vývojové desky. Díky těmto vlastnostem tak umožňují Arduino další možnosti využití. Komunikace mezi deskami probíhá po sběrnici. Současně lze najednou použít více desek. Oficiálně Arduino nabízí uvedené shieldy:

- **Ethernet Shield** – osazen ethernetovým čipem Wiznet W5100 s 16K vnitřní pamětí. To umožňuje Arduino se pomocí SPI sběrnice připojit k ethernetové či internetové síti pomocí konektoru RJ-45. Rychlost připojení 10/100Mb

- **WiFi Shield** – komunikaci mezi Arduinem a WiFi čipem HDG104 zajišťuje mikrokontrolér ATmega32UC3 pomocí sběrnice SPI. K WiFi lze připojit přes WiFi standard 802.11b/g, bez zabezpečení nebo se zabezpečením typu WEP a WPA2 Personal
- **GSM Shield** – umožňuje pomocí čipu Quectel Cellular Engine M10 připojit Arduino do mobilní sítě přes GSM/GPRS ve frekvenčním pásmu GSM850MHz, GSM900MHz, DCS1800MHz a PCS1900MHz.
- **Wireless Proto Shield** – tato přídatná deska je stavěná na připojení bezdrátového modulu Xbee od společnosti Digi. Lze však připojit i jakýkoliv jiný modul se stejnou stopou. Nejčastěji se však připojuje modul Xbee 802.15.4(1.serie) nebo Xbee Znet 2.5 či ZB (2.serie).
- **Motor Shield** – touto doplňkovou deskou, která je osazena dvojitým regulátorem motorů L298 lze plnohodnotně a na sobě nezávisle ovládat dva DC motory (2x2A) nebo jeden krokový motor (1x4A). (Arduino, © 2014)

3.3 Vývojové prostředí

Je software, který usnadňuje programátorovi práci se psáním zdrojového kódu. V našem případě jde o Arduino IDE. Jedná se o multiplatformní program, který je napsán v jazyce Java. Díky Javě je nejen dosažena možnost práce pod jakýmkoli operačním systémem ale také podporuje Java Virtual Machine (MS Windows, Linux, Mac OS atd), ale také stejně vypadající pracovní prostředí. Používaný programovací jazyk vychází z jazyka Wiring. (Root.cz, © 1998 – 2014)

3.3.1 Instalace a nastavení Arduino IDE

Na oficiálních webových stránkách Arduina (<http://www.arduino.cc>) v záložce Download si lze zdarma stáhnout nejaktuálnější vývojové prostředí (Arduino 1.0.5), které je kompatibilní s operačním systémem Window, Linux i Mac OS. Po stáhnutí a rozbalení za dekomprimovaného programu v počítači s OS Window 7, připojíme vývojovou desku k počítači. Po automatické snaze operačního systému o instalaci ovladače dochází k varovnému hlášení a ukončení instalace z důvodu nenalezení ovladače. Ovladač je nutné tedy nainstalovat ručně a to následujícím způsobem:

- v záložce „správce zařízení“ vybereme nabídku „Porty (COM a LPT)“, kde je položka „Arduino xxx (COMy)“ (xxx je typ desky, y číslo obsazeného portu)
- následně aktualizujeme software ovladače přes pravé tlačítko myši a vybereme možnost vyhledat ovladač v počítači
- po nalezení složky Arduino a jejím následném rozkliknutí, otevřeme podsložku Drivers
- po potvrzení by měl Windows úspěšně instalaci ovladače dokončit
- po takto úspěšné instalaci je vývojová deska Arduino připravena k následnému programování.
- Pro spuštění programu Arduino IDE můžeme dle potřeb desku nastavit a nakonfigurovat. Konkrétní typ vývojové desky, který chceme naprogramovat je potřeba přiřadit ve vývojovém prostředí Arduino IDE v záložce Tools/board /typ_desky. Dále je nutné zjistit na jakém portu je Arduino připojeno (v PC-správce zařízení) a v IDE přiřadit port v záložce Tools/Serial Port/COMx.
- Vše je nyní připraveno k nahrání zdrojového kódu do Arduina.

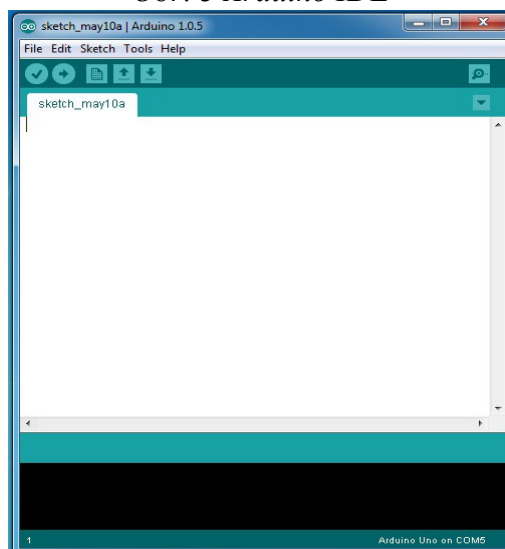
3.3.2 Arduina IDE

Pomocí vývojového prostředí si můžeme zdrojové kódy vytvářet, ukládat a přímo nahrávat do Arduina a to prostřednictvím jednoduchého programu Arduino IDE obr. 3. Jedná se o intuitivní prostředí, které můžeme rozdělit na 6 částí:

1. Titulní pruh – zde je zobrazen název uloženého zdrojového kódu následuje oddělovací svíslá čára za kterou je verze Arduina IDE
2. Panel nabídek – obsahuje položky pro práci se soubory.
 - File – práce se souborem (otevření, uložení, načtení, nastavení)
 - Edit – úprava a práce se zdrojovým kódem
 - Sketch – kompilace kódu, vložení knihoven, přístup k textovým souborům
 - Tools – nastavení portu, vývojové desky, programátoru
 - Help – nápověda při řešení problémů

3. Panel nástrojů pro rychlý přístup k funkcím.
 - Verify (Compile) – Překlad a ověření zdrojového kódu
 - Upload – Nahrání kódu do Arduina
 - New – Otevření nového textového editoru
 - Open – Otevření uloženého kódu
 - Save – Uložení zdrojového kódu
 - Serial Monitor – Zapnutí monitorování sériové linky
4. Textový editor – psaní nebo editování zdrojového kódu který zvýrazňuje syntaxi u klíčových slov
5. Hlášení překladače – výpis informací o provedené kompilaci (stav kompilace a nahrávání, velikost zdrojového kódu či chybového hlášení)
6. Stavový řádek – v levé části zobrazeno číslo řádku v textovém editoru na kterém se právě kursor nachází. Vpravo je informace o nastaveném typu vývojové desky a na jaký port bude nahráván zdrojový kód.

Obr. 3 Arduino IDE



Zdroj: Snímek z programu Arduino IDE

3.3.3 Programovací jazyk Wiring

Zdrojový kód pro programování vývojových desek Arduino se píše v Arduino Programming Language, který vychází z jazyka Wiring. Rozdíly jsou velmi zanedbatelné, můžeme říct, že pro programování Arduina se používá programovací jazyk Wiring. Tento jazyk je

v podstatě programovací jazyk C/C++ s vylepšenými sady maker. Wiring vznikl jako podobný open-source projekt Processing, ale pro hardware. Je zaměřen zejména pro začátečníky a pro uživatele, kteří se snaží proniknout do světa programování. Porovnáme-li logy Arduina a Processingu, zjistíme, že jsou si značně podobné. (Root.cz, © 1998 – 2014)

Základní struktura programovacího jazyka Wiring pro Arduino se musí vždy skládat minimálně ze dvou bloku (částí) programu. Tyto bloky jsou ohraničeny složenými závorkami.

První blok je obsažen ve funkci `void setup() {}`. Jedná se o funkci, kterou Arduino volá pouze jednou a to při spuštění programu (po připojení napájení nebo resetu). Do tohoto nastavovacího bloku se vepisují zejména příkazy, které v sobě nesou inicializační parametry jako např. nastavení vstupních a výstupních pinů či komunikačních pravidel. Po vykonání tohoto bloku přechází program do druhé části. (Hwkitchen.com, © 2014)

Druhá část programu `void loop() {}` nebo-li nekonečná smyčka. Obsahuje programový kód, který začne běžet po funkci `void setup() {}` a běží v nekonečné smyčce do odpojení napájení. Tato výkonová část provádí většinu činnosti jako čtení vstupů, ovládání výstupů, výpočty, posílání, ukládání či zapisování dat atd. (Hobbyrobot.cz, © 2014)

Níže je napsána jednoduchá struktura programu Arduino IDE, kde si můžeme všimnout, že před prvním blokem programu (`void setup() {}`) lze pro celý program deklarovat proměnné (`int led = 13;`) či použít knihovny (`#include <jméno_knihovny.h>`).

Struktura programu:

```
1 // #include <jméno_knihovny.h>
2 int led = 13;
3
4 void setup() {
5 // Jednořádkový komentář – vše za dvojitým lomítkem (//) v daném řádku se
6 // jedná o komentář, který se nekompile se zdrojovým kódem
7 pinMode(led, OUTPUT);
8 }
9
```

```
10 void loop() {  
11 /* vícenásobný komentář – text psaný mezi „lomítkem, hvězdičkou“ a „hvězdičkou,  
12 lomítkem“ je také považován za komentář, který na rozdíl od jednořádkového komentá  
13 ře může obsahovat libovolný počet řádků. */  
14  
15 digitalWrite(led, HIGH);  
16 delay(1000);  
17 digitalWrite(led, LOW);  
18 delay(1000);  
19 }
```

Tento program ovládá 13tý digitální výstup na vývojové desce Arduino tak, že 1000ms (1 vteřinu) je na pinu napětí 5V poté dalších 1000ms se napětí na pinu rovná 0. Tento kód lze použít pro vteřinové blikání LED diody.

4 Programování Arduina

K programování mikrokontrolérů je potřeba mít patřičný hardware (převodník mezi mikropočítačem a počítačem) a k dispozici patřičný software (vývojové prostředí). U Arduina je nutné mít vývojovou desku Arduino, která již obsahuje převodník a jako software je zde Arduino IDE.

4.1 Hardwarové vybavení

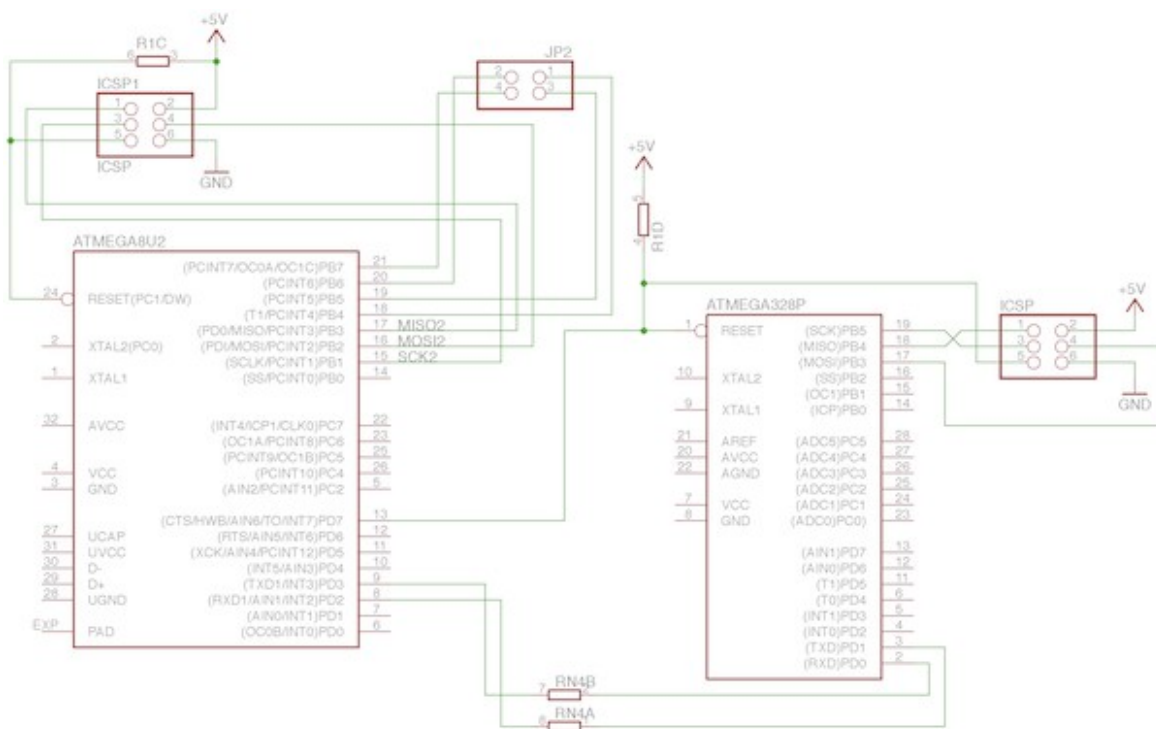
Pro naprogramování mikrokontroléru nutné použít převodník, jak již bylo detailně popsáno výše. Pro mikrokontroléry firmy Atmel, které se používají ve vývojových deskách Arduino lze naprogramovat pomocí sériového nebo paralelního převodníku.

Programování přes paralelní port se již dnes prakticky nepoužívá, protože při změně strojového kódu je potřeba vyjmutí integrovaného obvodu a naprogramování v externím programátoru. Dále pak moderní počítače ani již portem nedisponují.

Sériový převodník je mnohem efektivnější ve srovnání s paralelním převodníkem nejen z pohledu ale i stability s menším počtem vodičů. U programování se jedná zejména o převodník s mikroprocesorem, který konvertuje posílané data z PC na srozumitelnou posloupnost impulsů, které následně mikrokontrolér zpracovává. Některé programátory používají kromě mikroprocesoru i podpůrný obvod FTDI, který slouží jako rozhraní mezi USB a asynchronní sériovou linkou RS232 či TTL logiku.

Vývojová deska Arduino již obsahuje USB-seriál převodník. U starších vývojových desek s USB převodníkem se používaly čipy FTDI, jež se hlásila jako klasická sériová linka. U novějších desek se již místo FTDI používá mikrokontrolér. Ten pak slouží jako USB-UART převodník a z pohledu počítače se tváří jako USB zařízení. Oba mikrokontroléry jsou spojeny a komunikují přes UART rozhraní viz [obr.4](#). (μArt.cz, © 2014)

Obr. 4. Blokové schéma propojení Arduina a USB převodníku



Zdroj: *μArt.cz*, © 2014

Další možnost naprogramovat vývojovou desku Arduino je přes ICSP konektor (In Circuit Serial Programming), který je zobrazen na obr.5. Popis pinů podle Obr.5 je následující

1. MISO (Master In/Slave Out) – vstupní pin ze strany programátoru a výstupní ze strany mikrokontroléru
2. +Vcc – kladné napájecí napětí (+5V)
3. SCK (Serial Clock) – zdroj hodinového signálu určující rychlost komunikace (nastavování posuvných registrů)
4. MOSI (Master Out/Slave In) – výstupní datový pin z programátoru a vstupní ze strany mikrokontroléru
5. Reset – nulovací vstup, resetuje mikrokontrolér (propojením s Gnd)
6. Gnd (Ground) – uzemnění (záporná pól)

Přes tento konektor lze nejen programovat ale také využít v běžícím programu. Pokud bychom však chtěli ICSP použít k ladění programu, je nutné v mikrokontroléru povolit OCDS (On-Chip Debug System).

Obr. 5. ICSP konektor



Zdroj: Arduino, © 2014

4.2 Softwarové vybavení

Z hlediska požadavku na softwarové vybavení pro programování mikrokontrolérů je vývojové prostředí a softwarové rozhraní. Každé vývojové prostředí je určeno pro určité typy mikrokontrolérů (většinou podle výrobce PIC, Atmel atd.) s odpovídajícím vývojovým kódem. Dříve se mikrokontroléry programovaly výslovně v nižším programovacím jazyku symbolických adres (Assembleru). Jelikož programování složitějších úloh bylo časově dosti náročné, začalo se přecházet na vyšší programovací jazyk (např. C/C++). Tento programovací jazyk umožnil jednodušším způsobem vytvořit složitější program i za cenu menší efektivity a rozsáhlejšího strojového kódu.

Softwarové rozhraní má za cíl přeložit zdrojový kód na strojový a ten poslat přes převodník a programátor do mikrokontroléru. Po napsání kódu a jeho kompilaci vznikne soubor s příponou hex (*.hex), který se přes komunikační rozhraní programátoru do mikrokontroléru nahraje.

U Arduina se zdrojový kód píše ve vývojovém prostředí Arduino IDE. Celý projekt se ukládá do souboru s příponou *.ino, který je uložen ve složce se stejným názvem. Uložený kód se v Arduino nazývá sketch. Tento sketch může být rozdělen do více souborů. Před nahrání vývojového kódu do Arduina se kód nejprve zkompiluje (zjistí zda kód neobsahuje hrubé chyby), poté převede do strojového kódu a následně nahraje do mikrokontroléru.

II. PRAKTICKÁ ČÁST

5 POUŽITÉ KOMPONENTY

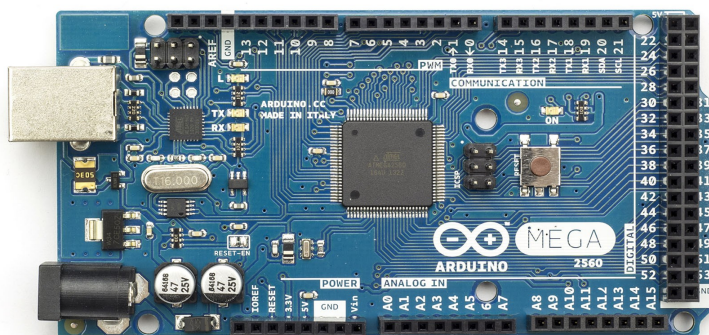
Cílem praktické části je ukázat práci s jednoduchým webovým serverem Arduino, přes který by bylo možné ovládat výstupní piny a získávat ze vstupních pinů informace, které by se zobrazovali ve webovém prohlížeči.

Hlavním srdcem celého systému je vývojová deska. Z široké škály byl vybrán typ Arduino Mega 2560, který má narozdíl od klasického Arduina Uno rychlejší procesor ATmega 2560. K propojení Arduina s Ethernetem je nutno použít doplňkovou desku, která by zprostředkovala komunikaci. Jedná se o desku zvanou Ethernet shield, která je osazena zásuvkou RJ-45 a řadičem WIZnet W5100. Aby mohlo být fyzicky viděno ovládání výstupních pinů, jsou na tyto piny připojeny LED diody. Pro simulaci analogových vstupů použijeme potenciometry a pro digitální vstupy mikropínače.

5.1 ARDUINO MEGA 2560

Arduino Mega 2560 je nástupcem známé desky Arduino Mega 1280. Kromě všech funkcí předchozího Arduina, má nové Arduino Mega 2560 dvojnásobnou flash paměť a používá jiný USB-serial převodník, který je postavený na mikrokontroléru Atmel ATmega8U2 místo staršího FTDI čipu. To umožňuje rychlejší přenosové rychlosti a není třeba instalovat jakékoli ovladače pro Mac a Linux (vše, co potřebujete na OS Windows je jednoduchý konfigurační soubor *.inf). Pomocí nového USB-serial převodníku je v počítači Arduino zobrazeno jako klasické USB zařízení (klávesnice, myš, joystick, MIDI, atd.)

Obr. 6. Arduino Mega 2560

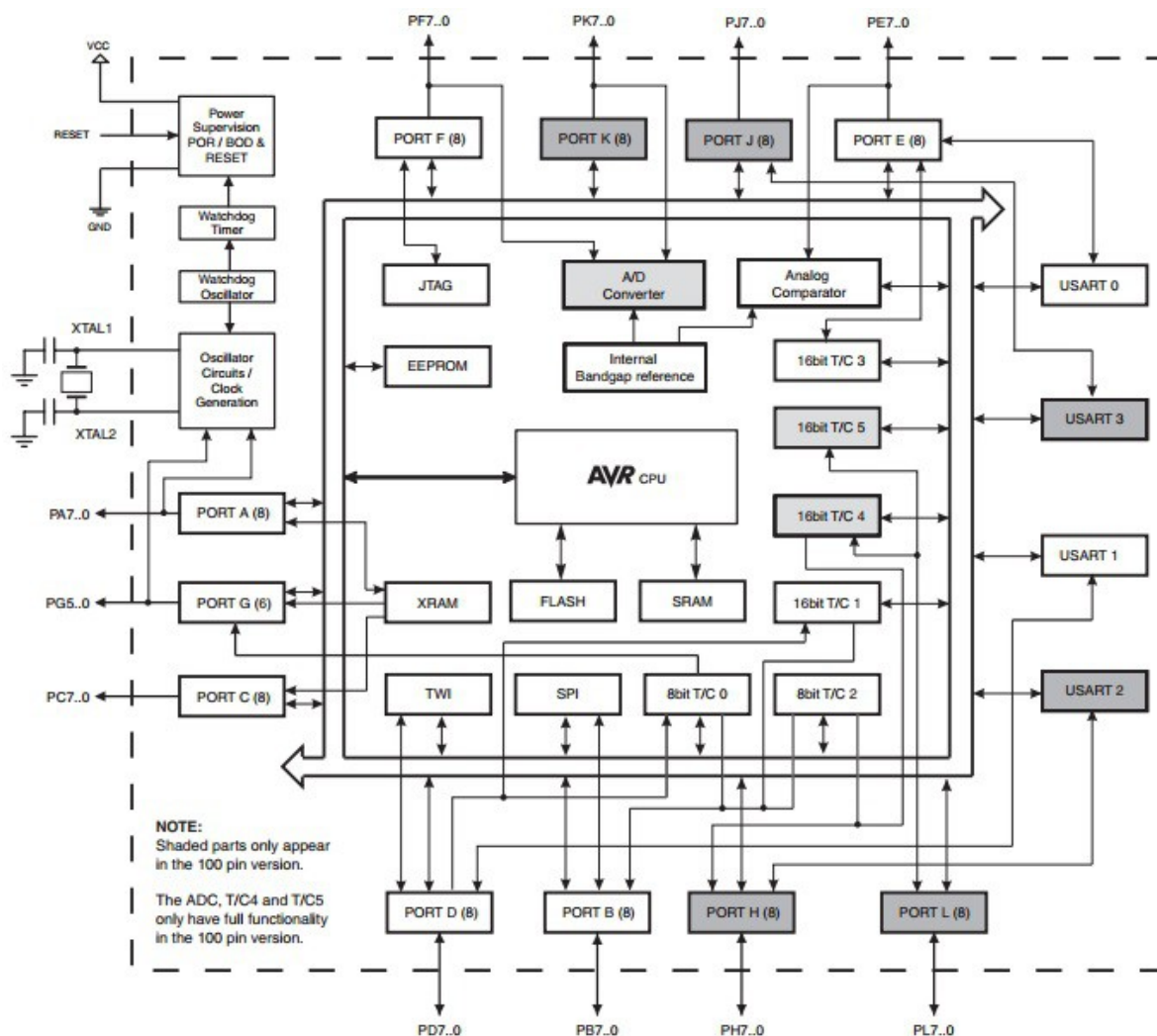


Zdroj: Arduino, © 2014

Deska Arduino Mega 2560 je založena na výkonné ATmega2560. Má 54 digitálních vstupních / výstupních pinů (z nichž 14 lze použít jako výstupy PWM), 16 analogových vstupů, 4 UART (hardware sériové porty), 16 MHz krystalový oscilátor, připojení USB, napájecí konektor, záhlaví ICSP a tlačítko reset. Obsahuje vše potřebné k podpoře mikrokontroléru. Napájet lze pomocí USB kabelu nebo externího adaptéru (230V AC – 12V DC) či baterie (7-12V). Mega je kompatibilní s většinou shieldů určené pro Arduino Due-milanove nebo Diecimila.

Hlavními příslušenství pro vývojovou desku Arduino Mega 2560 je USB kabel pro připojení k PC a DC adaptér. Deska má certifikaci CE i FCC.

Obr. 7. Blokové schéma ATmega 2560



Zdroj: HW Kitchen, © 2014

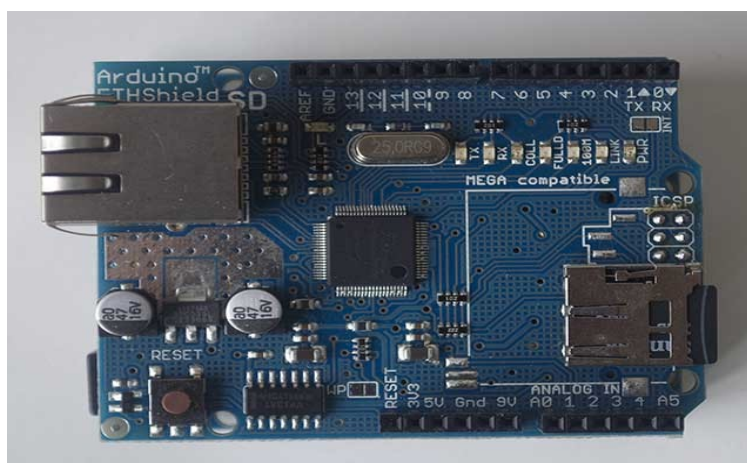
5.1.1 Specifikace Arduina Mega 2560

- mikrokontrolér – ATmega2560
- převodník – USB-serial (Atmel ATmega8U2)
- I/O analogových pinů – 16 pinů (10bitů – 0-1023)
- I/O digitálních pinů – 54 pinů
 - PWM výstupů – 14 8bitových výstupů (2-13, 44-46 pin)
 - sériové linky (UARTS) – 4 linky
 - seriál 0 – 0(RX) - 1(TX)
 - seriál 1 – 14(RX) - 15(TX)
 - seriál 2 – 16(RX) - 17(TX)
 - seriál 3 – 18(RX) - 19(TX)
 - externí přerušení – 6
 - přerušení 0 – 2 pin
 - přerušení 1 – 3 pin
 - přerušení 2 – 21 pin
 - přerušení 3 – 20 pin
 - přerušení 4 – 19 pin
 - přerušení 5 – 18 pin
 - SPI rozhraní – 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS) pin
 - TWI (I2C) rozhraní – 20 (SDA) a 21 (SCL) pin
- vstupní napětí – 7-12V
- max. proud na I/O pin – 40mA DC
- max. odběr proudu z 3,3 V pinu – 50mA DC
- paměť flash – 256kB (pro strojový kód)
- frekvence – 16MHz
- rozměry – 108x53x15mm (Arduino, © 2014)

5.2 ETHERNET SHIELD

Arduino Ethernet shield umožňuje vývojové desce Arduino se pomocí UTP kabele připojit na Ethernet. Deska obsahuje i resetovací mikrospínač, kterým po stisknutí vyresetujeme celé Arduino. Dále pak mohou některé desky obsahovat i slot na Micro SD kartu, která slouží pro čtení nebo ukládání potřebných dat. Pro práci s kartou je nejdříve potřeba danou kartu naformátovat na systém souborů typu FAT16 nebo FAT32 (pro lepší komunikaci se však doporučuje FAT16). Kompatibilitu desky s Arduinem Mega 2560 poznáme tak, že pod signalizačními LED diody najdeme nápis bílými písmeny MEGA compatible.

Obr. 8. Arduino Ethernet shield

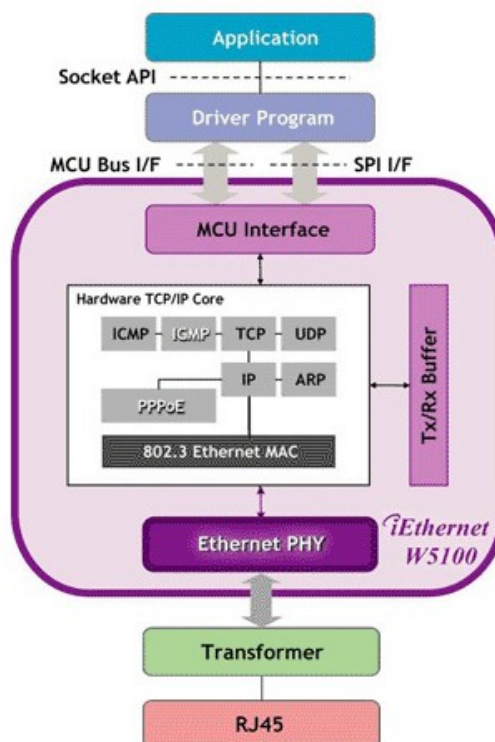


Zdroj: Arduino, © 2014

Spojení desek se provede nasunutím Ethernetové desky na Mega2560 desku tak, aby byli zachováni číselné označení IO pinů a tím i seděl ICSP konektor. Spojení mezi deskami probíhá přes PWM piny (10 až 13). U shieldu s integrovanou Micro SD kartou je komunikace s Arduinem řešena přes 4tý PWM pin. S ostatními piny můžeme bez problémů pracovat.

Komunikaci mezi Ethernetem a Arduinem zajišťuje řadič. Dříve se Ethernet shieldů používaly čipy ENC28J60, které jsou dnes výhradně nahrazovány čipy WIZnet W5100 (viz. Obr.x). Pro zjednodušenou komunikaci s ethernetem je vhodné použít podle typu řadiče odpovídající ethernet knihovnu.

Obr. 9. Blokové schéma řadiče W5100



Zdroj: Wiznet, © 2009 - 2014

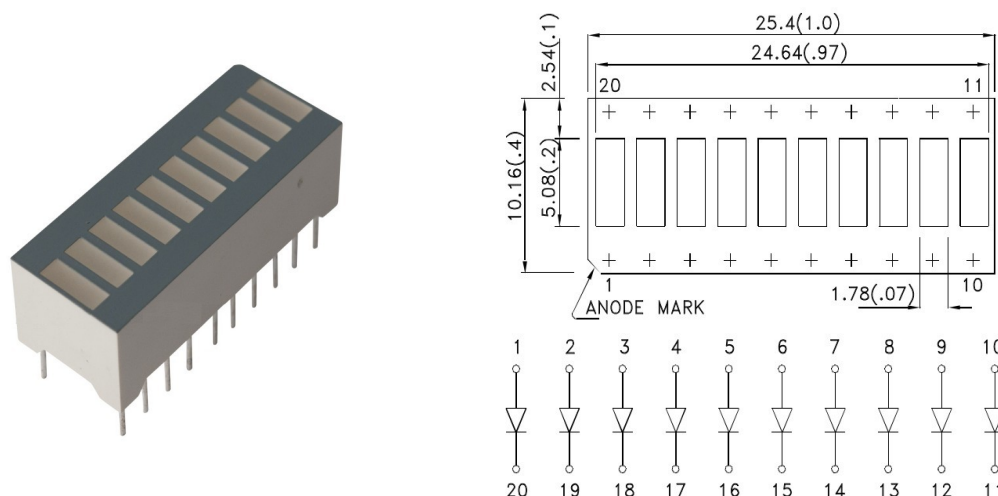
5.2.1 Specifikace Ethernet shieldu

- podpora TCP/IP protokoly TCP a UDP
- rychlost 10/100Mb
- vnitřní paměť 16Kb
- podporuje Auto MDI/MDIX
- síťové napájení PoE (Power over Ethernet)

5.3 LED Bargraf

10ti sloupcový LED displej s technologií Galium od firmy Kingbright, vyroben z 10 LED diod zalitých do jednoho těla viz. levá část obr. 10. Zapojení LED diod vyobrazeno vedle na pravé části obr. 10. Pro demonstrování aktuálních stavů výstupních pinů vývojové desky Arduina nám tento bargraf bohatě postačí. Připojení k vývojové desce bude realizováno přes vypočítané předřadné rezistory, které slouží k omezení napětí a protékajícího proudu LED diodou. Hodnota rezistoru se musí stanovit tak, aby velikost proudu nepřekročila max. povolený proud LED diody a zároveň max. odebíraný proud z Arduina.

Obr. 10. LED bargraf a jeho vnitřní zapojení



Zdroj: Alldatasheet.com, © 2009 – 2014, GM electronic, © 1990 – 2014

Vlastnosti bargrafu:

- barva: zelená
- dominantní vlnová délka: 568nm
- špičková vlnová délka: 565nm ,polo-šířka spektrální čáry: 30nm
- napětí v propustném směru: 2,2V (max. 2,5V)
- max. proud LED diodou: 20mA ,závěrný saturační proud: 10 μ A ,kapacita: 15pF
(Alldatasheet.com, © 2009 – 2014)

5.4 Trimry

Odporový trimr je pasivní elektronická součástka, která má podobné vlastnosti jako rezistor. Oproti rezistoru lze u trimru měnit lineární velikost odporu a to od 0 Ω do max. hodnoty trimru. Trimr má 3 vývody, kde mezi dvěma vývody je rovnoměrně nanášena odporová vrstvička s přesně danou odporovou hodnotou. Třetí vývod je připojen na kovového jezdece, který se posouvá po odporové vrstvičce. Pomocí tohoto jezdece lze nastavit přesnou odporovou hodnotu. Sečteme-li odporovou hodnotu na vývodech (1-3 a 2-3) musí se odpor rovnat hodnotě na vývodech 1-2. Použitý trimr je v ležatém provedení s vertikální osou (osičkou).

Obr. 11. Trimr



Zdroj: GM electronic, © 1990 – 2014

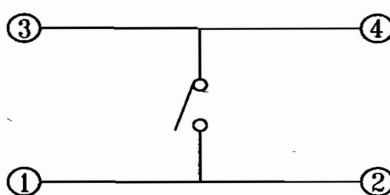
Vlastností trimrů:

- Max.odpor - 10KOhm (lineárně)
- Tolerance - 20 %
- Ztrátový výkon - 0,25 W
- Úhel rotace elekt. - 245°
- Teplotní koeficient - +200/-300 ppm/°C (GM electronic, © 1990 – 2014)

5.5 Mikrospínače

Jedná se o mikrospínače výrobce WEALTH METAL FACTORY v provedení DIP (Dual in-line package) určené do DPS (desky plošných spojů). Mikrospínače disponují 4mi vývody viz. obr. 12, kde podle vnitřního zapojení je vidět propojení dvakrát dvou vývodů. Vzdálenosti mezních pracovní poloh páčky (hmatníku) je 1,5 mm. (GM electronic, © 1990 – 2014)

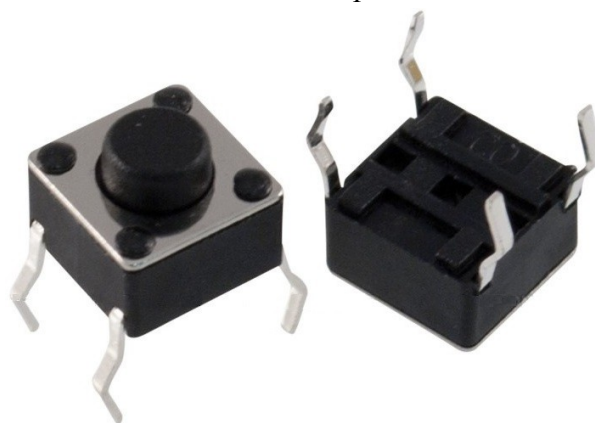
Obr. 12. Blokové schéma mikrospínače



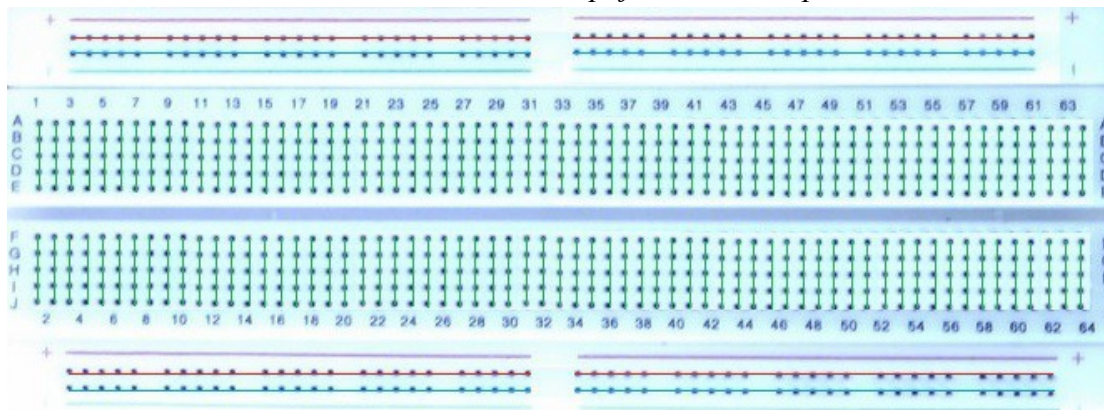
Zdroj: GM electronic, © 1990 – 2014

Vlastnosti mikrospínače:

- 1-pólový, spínací ON-OFF
- Jmenovité napětí = 12V
- Jmenovitý proud = 0,05A
- Přechodový odpor = 100 MOhm (*GM electronic, © 1990 – 2014*)

Obr. 13. Mikrospínač*Zdroj: GM electronic, © 1990 – 2014***5.6 Zkušební nepájivé kontaktní pole**

Slouží pro vyvíjení, testování a odzkoušení jednodušších el. zapojení, které nejsou nijak ověřeny (pouze navrženy a spočítány). Realizace propojení je podstatně rychlejší než osazování na DPS. Na obr. 14 jsou dokreslené červené, modré a zelené čáry, které zobrazují fyzické propojení zdírek. Zdířky které jsou pod červenými čáry se používají na kladné napájecí napětí, modré na záporné napětí či GND (zem) a zelené na propojení mezi jednotlivými el. součástkami.

Obr. 14. Zkušební nepájivé kontaktní pole*Zdroj: Vlastní zpracování*

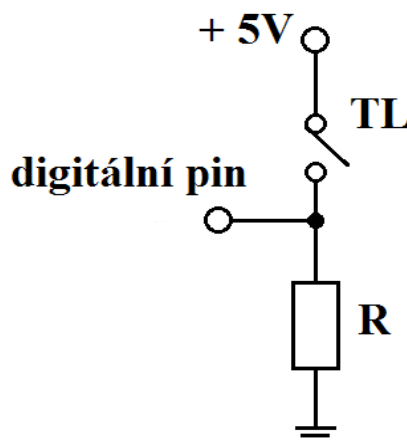
6 POPIS ELEKTRONICKÉHO ZAPOJENÍ

Pro vytvoření webového serveru Arduino je potřeba nejen zvolení správné vývojové desky a doplňkové desky Ethernet shield, ale také zvolení počtu analogových a digitálních vstupů a digitálních výstupů. Ovládání digitálních výstupů přes piny a webové stránky bude pomocí graficky rozmístěných ovládacích tlačítek ve stylu numerické klávesnice (1-9). Tím nám postačí 9 digitálních vstupů a 9 výstupů. Pro zobrazení aktuálních hodnot analogových vstupů využijeme 6 pinů z Ethernet shieldu (A0-A5).

6.1 Zapojení jednotlivých součástek

K získání analogových hodnot pro analogové vstupy použijeme trimry, které pracují jako děliče napětí tzn. že pomocí změny odporu se mění napětí na vstupu desky. Digitální výstupy budou podle daných pravidel rozsvěcovat nebo zhasínat LED diody. Podle toho poznáme zda je na daných pinech aktuálně 0V nebo 5V (převědeme-li to na dvojkovou soustavu tak log.0 či log.1). U diod je však potřeba vypočítat předřadné odpory, kterými se omezí jak max. proud tekoucí přes diodu, tak max. výstupní proud z Arduina. Tím se zabrání zničení LED diod, pinů Arduina či celého mikrokontroléru. A nakonec u digitálních vstupů budou použity mikrospínače pomocí kterých se bude dostávat napětí na piny. V klidovém stavu mikrospínačů bude na pinech nulové napětí (log. 0). Při stisknutí hmatníku napětí dosáhne 5V (log. 1). Do obvodu s mikrospínačem je nutné zapojit rezistor a to tak jak je nakresleno na obr. 15. Pokud by se rezistor do obvodu nezapojil, hrozilo by při stisku mikrospínače zkrat (propojení kladného napětí se zemí).

Obr. 15. Schéma zapojení tlačítka



Zdroj: Vlastní zpracování

6.2 Výpočet vstupních a výstupních okruhů Arduina

Velikost odporové hodnoty trimrů, které budou připojeny na vstupní analogové piny Arduina, zvolíme $10\text{K}\Omega$ dle doporučeného zapojení výrobce. Napěťový rozsah analogových vstupů může nabývat hodnot od 0V do 5V . A/D převodník má 10 bitové rozlišení. Tzn. že rozsah vstupního napětí je rozdělen na 2^{10} (1024) dílků. Jeden dílek tedy odpovídá cca:

$$U = \frac{U_{max}}{\text{bitů}}$$

$$U = \frac{5}{1024} = 4,88 \text{ mV}$$

Rozlišení velikosti hodnoty odporu na jeden dílek $10\text{K}\Omega$ trimru bude zhruba:

$$R = \frac{R_{max}}{\text{bitů}} \quad R = \frac{10\,000}{1024} = 9,76 \Omega$$

Rozlišení úhlu pootočení osičky trimru na jeden dílek odpovídá přibližně:

$$\alpha = \frac{\alpha_{max}}{\text{bitů}}$$

$$\alpha = \frac{245}{1024} = 0,24^\circ$$

Na digitální vstupy vývojové desky Arduino budou zapojeny mikrospínače s odpory o hodnotě také $10\text{K}\Omega$, jak je uvedeno na oficiálních stránkách Arduina. (Arduino, © 2014)

Výstupní digitální piny, které poslouží k ovládání LED diod mohou být max. zatíženy proudem 40mA na pin. Max. proud LED diody může být 20mA . Předřadný odpor při proudu 20mA bude:

$$R = \frac{U}{I}$$

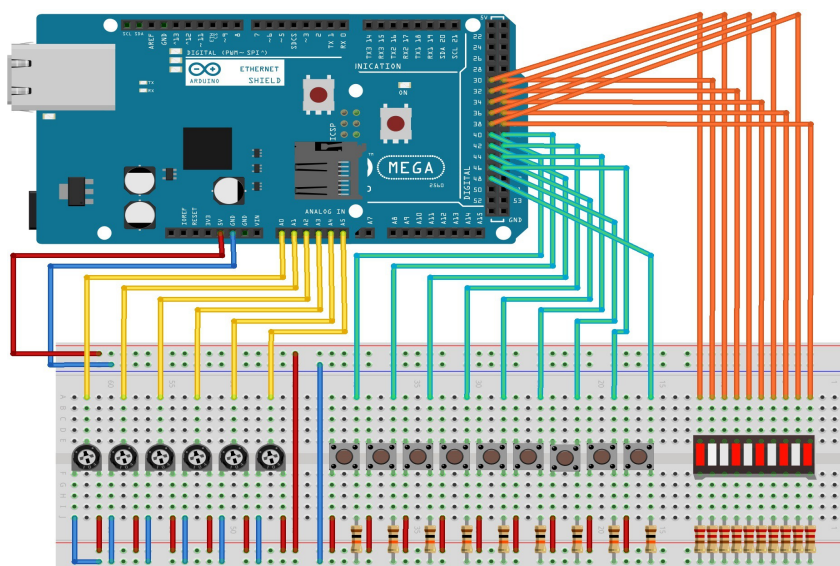
$$R = \frac{5}{20} = 250\Omega$$

Jelikož 250Ω rezistor ve standardní odporové řadě E12 ani E24 není. Použijeme nejbližší vyšší, tedy 270Ω .

6.3 Schéma zapojení

Schéma zapojení je zobrazeno na obr.16. Obsahuje vývojovou desku Arduino Mega2560, která je umístěna v horní části. Na ni je připevněna Ethernetová doplňková deska tzv. Ethernet shield zajišťující komunikaci mezi ethernetem a Arduinem. Pod těmito deskami je nepájivé kontaktní pole, které je osazeno šesti trimry, devíti mikropsínači s rezistory a LED bargrafem s předřadnými rezistory. Napájet vývojovou desku lze přes USB nebo přes externí napájení s jmenovitým napětím 7-12V. Nepájivé pole je napájeno z Arduina přes piny 5V a GND pomocí červeného (+5V) a modrého vodiče (GND – zem).

Obr. 16. Schéma zapojení Arduina



Zdroj: v programu Fritzing

Jezdci trimru jsou zapojeny na analogové piny desky (A0-A5) pomocí žlutých vodičů, mikropsínače na digitální piny (40-48) zelenými vodiči a LED bargraf také na digitální piny (30-39) vývojové desky Arduino prostřednictvím oranžových vodičů. Předřadné rezistory bargrafu jsou zapojeny sériově mezi záporným napětím a LED. Podobné propojení je i mezi rezistory mikropsínačů.

7 NÁVRH PROGRAMU

Při návrhu softwarového řešení je potřeba z analyzovat?? zadání, co je potřeba udělat. Dále pak navrhnou řešení, jak budeme postupovat. Tento postup je dobré přenést do vývojového diagramu, který nám pomůže lépe se zorientovat v programu a tak předejít chybám, kterých by jsme se mohli dopustit. Nakonec pomocí vývojového diagramu lze jednodušeji vytvořit zdrojový kód, který pak přeložíme a nahrajeme do mikrokontroléru.

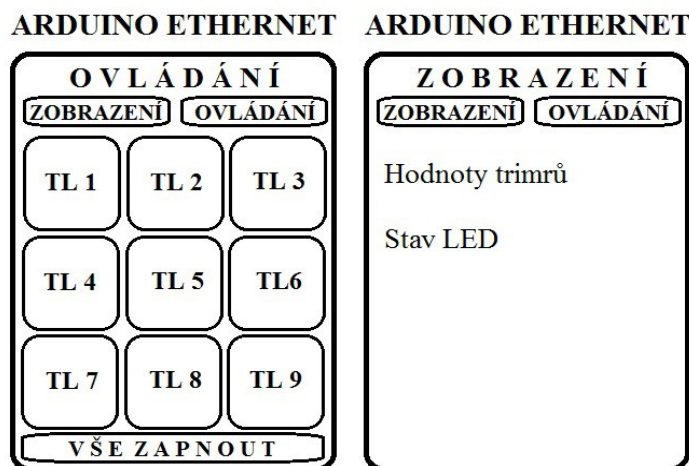
7.1 Analýza zadání a návrh řešení

Pomocí webového prohlížeče ovládat a získávat data z programovatelných pinů vývojového kitu Arduino, Po nakreslení blokového zapojení viz. obr. 16 už víme co, kde, čím a jak budeme komunikovat s jednotlivými prvky. Nyní je potřeba softwarově naprogramovat posílání aktuálních hodnot z analogových pinů na web a tam je následně zobrazovat. Pomocí mikropínačů ovládat LED diody a zároveň propisovat aktuální stavy LED diod na webové stránky.

7.2 Návrh webu

Než se pustíme do samotného programování, tak si nejprve navrhne a nakreslíme webové prostředí, pomocí kterého budeme ovládat výstupní zařízení či získávat data.

Obr. 17. Návrh webu



Zdroj: Vlastní zpracování

Návrh webové stránky je vyobrazen na obr. 17. kde levá část obrázku bude automaticky zobrazována při každém přístupu na web. Nadpis „ARDUINO ETHERNET“ bude trvale zobrazen. Přepínání mezi záložkami bude možné pomocí tlačítek „ZOBRAZENÍ“ a „OVLÁDÁNÍ.“ Při přepnutí tlačítkem „ZOBRAZENÍ“ se zobrazí pravá část obr. 17 a

ztuční se písmo na tlačítku. Zde se vypíše aktuální hodnoty trimrů a stavy LED diod na bargrafu. Pro vrácení se na záložku ovládání je nutné kliknout na tlačítko s nápisem „OVLÁDÁNÍ.“ Zobrazí se webová stránka ve stylu levého obr. 17 a opět se nápis ztuční. Tady je možné simulovat stisk mikropínačů a tím ovládat LED diody. Např. při stisku mikropínače č.2 se rozsvítí LED dioda č.2 a na webu se ztuční nápis „TL 2“. Při opětovném stisku LED dioda zhasne a nápis „TL 2“ na webu, již nebude tučně napsáno. Ovládání přes webové tlačítka je totožné. Při stisku daného webového tlačítka se změní stav příslušné LED diody a podle změny LED se změní i nápis. U tlačítka „VŠE ZAPNOUT“ se rozsvítí všech devět LED diod a nápis se změní na „VŠE VYPNOUT“. Po opětovném stisku se opět všechny diody zhasnou a nápis se opět změní na „VŠE ZAPNOUT“

7.3 Vývojový diagram

Znázorňuje posloupnost jednotlivých kroků či stavbu programu. Obsahuje začátek programu, vstupy/výstupy, příkazy a konec programu. V našem případě konec programu není nakreslen z důvodu nekonečné smyčky. Všechny kroky musí být řádně pospojovány spojovacími čáry. Níže je na obr. 18 vyobrazen stručný vývojový diagram návrhu programu pro vývojovou desku Arduino.

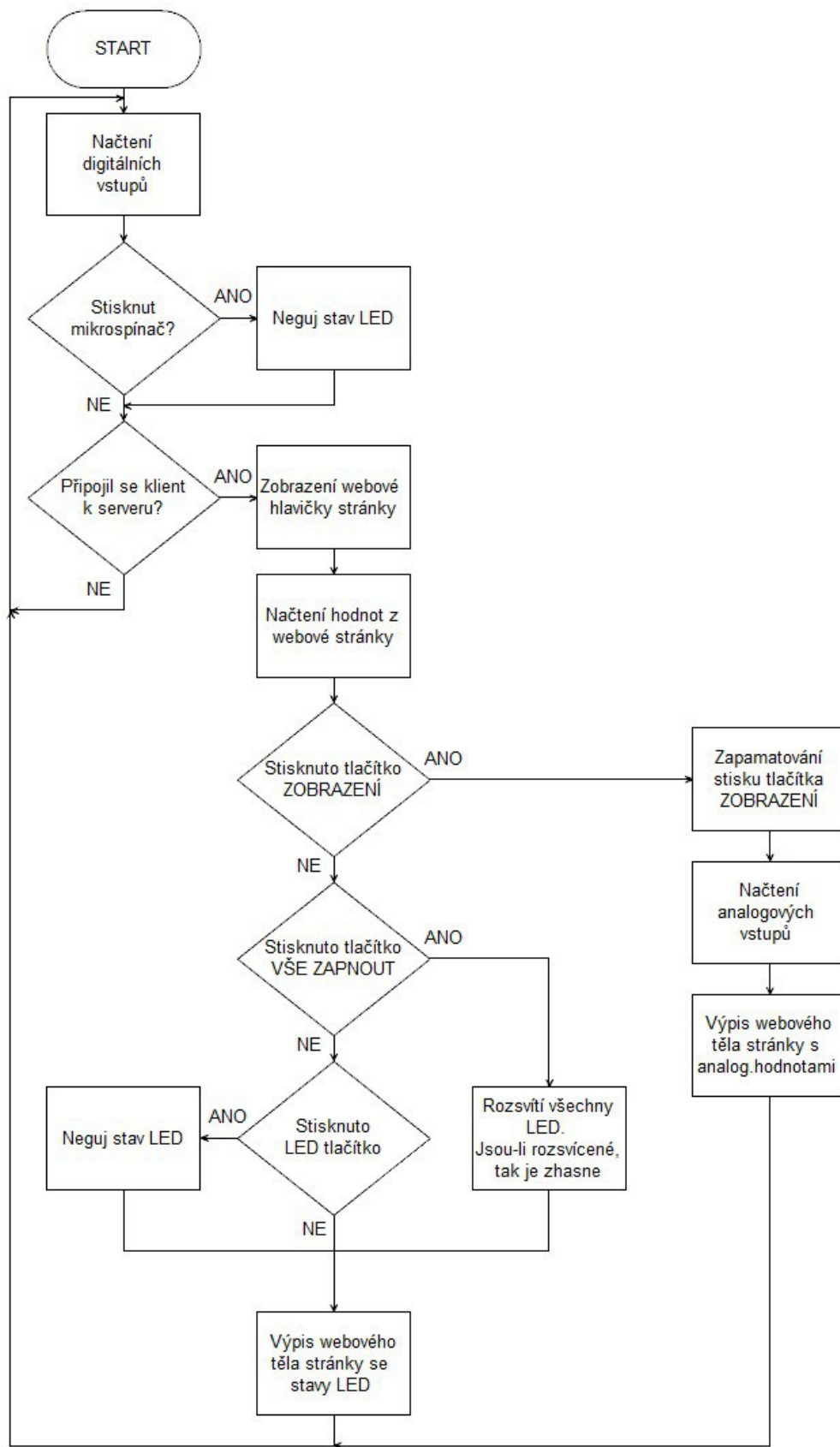
Popis vývojového diagramu:

Při startu Arduina se načtou knihovny, inicializační parametry a deklarují proměnné.

Po tomto kroku běží ve smyčce načtení digitálních vstupů (stavy mikropínačů) a jeho vyhodnocení (zda nebyl stisknut mikropínač), v případě stisku změna dané LED diody (rozsvícení/zhasnutí), a kontrola zda se někdo nechce připojit k Arduinu.

V případě připojení na webové stránky se zobrazí hlavička stránky, načtení dat ze stránky (zda nebylo stisknuto webové tlačítko) následné vyhodnocení (o které tlačítko šlo), příslušná reakce na stisk webového tlačítka (změna stavu LED či přepnutí záložky na webové stránce) a propsání aktuálních hodnot nebo změna stavu LED diod do webového těla stránky.

Obr. 18. Vývojový diagram



Zdroj: Exportovaný obrázek z programu Diagram Designer

7.4 Vytvoření zdrojového kódu

Pomocí vývojového diagramu víme jak budeme postupovat při programování zdrojového kódu v Arduino IDE. Před puštěním se do programování samotného Arduina si nejprve vytvoříme webové stránky, prostřednictvím kterých budeme s Arduinem komunikovat.

Navržené stránky již máme na obr. 17, Nyní je jen potřeba pomocí webových jazycích HTML a CSS napsat webové stránky, které by graficky odpovídaly navrženým stránkám. Zdrojový kód obou stránek je vypsán v přílohách „P1: Webová stránka Ovládání“ a „P2: Webová stránka Zobrazení.“ Výsledné grafické znázornění z webu je na obr. 19.

Obr. 19. Grafické znázornění webové stránky



Zdroj: Exportovaný a ořezaný obrázek z webového prohlížeče

Po vytvoření zdrojového kódu stránek můžeme začít programovat v Arduinu IDE. Abychom se nemuseli složitě zabývat tvorbou zdrojového kódu pro vytvoření síťového serveru a zajištění komunikace mezi vývojovou deskou Arduino a Ethernte shieldem, usnadníme si práci (pomůžeme si) s knihovnamy SPI.h a Ethernet.h. Poté nám stačí jednoduše zadat IP adresu, masku podsítě, výchozí bránu, na jakém TCP portu bude server naslouchat a MAC adresa Arduina. Např.

```
byte ip[] = { 10, 0, 0, 200 };
byte gateway[] = { 10, 0, 0, 110 };
byte subnet[] = { 255, 255, 255, 0 };
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
EthernetServer server = EthernetServer(80);
```

```
void setup(){
  Ethernet.begin(mac, ip, gateway, subnet);
  server.begin();
}
```

Tímto jsou zajištěny základní nastavovací parametry pro přístup k Arduinu.

Připojí-li se nějaký klient k Arduinu, zjistíme tak, že jsou od klienta do Arduina odeslána data. To zjistíme pomocí metody `available`. V našem případě

```
EthernetClient client = server.available();
```

Pro načítání digitálního vstupu, vyhodnocení stisknutí mikrospínače a následné změny LED diody bude zdrojový kód vypadat takto

```
int tlac1 = digitalRead(42);

if (tlac1 == 1)
{
  digitalWrite(22, !cteniPin[0]);
  cteniPin[0] = !cteniPin[0];

  while (tlac1 == 1)
  {
    tlac1 = digitalRead(42);
    delay(10);
  }
}
```

První řádek nám načte aktuální stav mikrospínače (1-sepnutý / 0-rozepnutý). Pokud bude spínač v sepnutém stavu, změní se stav LED diody která je se spínačem spjatá. První spínač připojený na digitálním pinu č.42 ovládá první LED diodu, která je na digitálním pinu 22. Změna stavu paměti diody se promítne na první pozici řetězce „cteni[0] (string)“. Po zapsání přejde do cyklu `while`, kde bude načítat digitální vstup a testovat dokud nedojde k puštění spínače. Tím je při stisku tlačítka ošetřeno neustálé zapisování změny LED diody.

Trimry jsou načítány v cyklu `for`, do proměnné „hodnotaVstupu“ a vypisovány na webovou stránku. Viz níže.

```
for (int i = 0; i < 6; i++) {
  int hodnotaVstupu = analogRead(i);
  client.print("<tr><td>Trimr ");
  client.print(i+1);
  client.print(" : ");
  client.print(hodnotaVstupu);
  client.println("</td></tr>");
}
```

U vypisování stavu LED diod na webové stránky v záložce „ZOBRAZENÍ“ je obdobné jak u vypisování hodnot trimrů. Pokud jde však o změnu webových tlačítek v záložce „OVLÁDÁNÍ“, jsou nejprve všechny stavy LED diod načteny do řetězce „cteniPin[i]“. Ze kterého jsou příslušná tlačítka správně zobrazena. Viz. níže

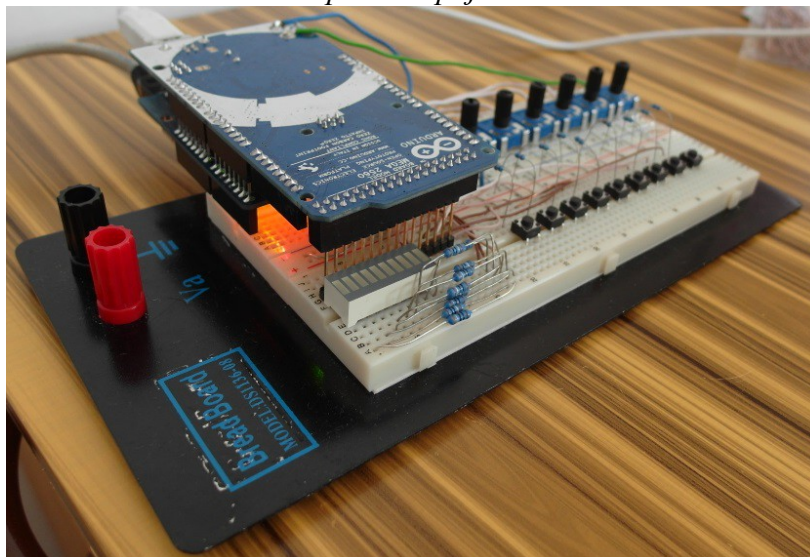
```
for (int i = 0; i < 10; i++) {  
    cteniPin[i] = digitalRead(i + ZacVystLed);  
}
```

Tak je popsána komunikace s mikrospínači, trimry a LED diody v bargrafu. Celý zdrojový kód je vypsán v příloze „P3: zdrojový kód ARDUINA.“

8 TESTOVÁNÍ WEBOVÉHO SERVERU

Po kompletním zapojení všech součástí (viz. obr. 20) dle schématu zapojení (viz. obr. 16) a nahrání zdrojového kódu do vývojové desky Arduina, lze přejít na závěrečné testování.

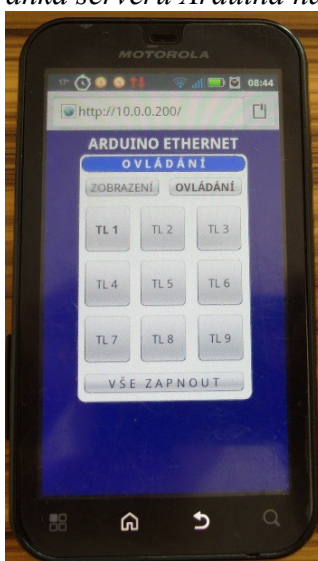
Obr. 20. Kompletní zapojení Arduina



Zdroj: Vlastní foto

Webový server je zapojen do lokální sítě LAN pomocí kříženého UTP kabelu, který je přiveden do WIFI routeru. Napájení Arduina je řešeno přes datový USB kabel (5V). Po otevření webového prohlížeče a připojení se na IP adresu webového serveru Arduino, se načte daná webová stránka (viz. obr. 21). Otestování všech funkčních tlačítek v prohlížeči, trimrů i mikrospínačů na nepájivém proběhlo bez větších problémů. I po cca. 2hodinovém provozu se zdál webový server stabilní.

Obr. 21. HTML stránka serveru Arduina na mobilním telefonu



Zdroj: Vlastní foto

ZÁVĚR

Cílem diplomové práce bylo seznámení se s projektem Arduino a jeho využití jako webový server. Pro vytvoření serveru byla použita vývojová deska Arduino Mega 2560 a doplňková deska Ethernet shield, která zprostředkovává komunikaci mezi Arduinem a Ethernetem. Na použité programovací piny byli připojeny LED diody, trimry a mikrospínače.

V úvodu teoretické části jsou vysvětleny základní pojmy jako je mikroprocesor, mikropočítač, mikrokontrolér, digitální signálový procesor či digitální signálový kontrolér. Poté je část práce věnována problematice mikropočítačů a její historii. Mikropočítače se dají rozdělit podle koncepce či instrukčních souborů. Následuje seznámení se s projektem Arduino, kde je popsána jeho historie, dále jsou rozebrány a porovnány vývojové i doplňkové desky, seznámení se s vývojovým prostředím a s jeho programováním. V praktické části jsou již popsány konkrétní použité komponenty. Tento rozbor zahrnuje charakteristiku jednotlivých součástek včetně jejich elektronického zapojení, schématu a výpočtů. V závěru praktické části je uveden postup vytvoření daného programu od počáteční analýzy zadání, návrhu řešení, přes vývojový diagram až k vytvoření samotného zdrojového kódu.

Hardwarové zapojení k naprogramovaným analogovým pinům, ke kterým jsou zapojeny odporové trimry, pomocí kterých můžeme lineárně měnit vstupní napětí. Na naprogramovaných digitálních vstupních pinech jsou zapojeny mikrospínače, pomocí nichž můžeme ovládat LED diody na bargrafu, které jsou k Arduinu připojeny pomocí digitálních výstupních pinů. Tyto výstupní piny můžeme ovládat i vzdáleně přes webový server.

Arduino je naprogramováno tak, aby uživatel, který se vzdáleně připojí na webový server i pomocí mobilního telefonu měl automaticky přizpůsobený obsah webové stránky na displeji a tím se mohl pohodlně přepínat mezi záložkami ovládání a zobrazení. V záložce ovládání jsou rozmístěny tlačítka podobně jako u numerické klávesnice. Ta zajišťuje pohodlné ovládání digitálních výstupů. V záložce zobrazení najdeme aktuální hodnoty analogových vstupů (trimrů) a stavů LED diod, které se každých 10s aktualizují, kdy dochází k obnovení stránky. V případě použití Arduina jako webového serveru bych doporučil používání pouze v ethernetové síti tj. v domácí nebo firemní síti. Jelikož Arduino zatím nedisponují dostatečným početním výkonem pro zabezpečení datové komunikace, nedoporučoval bych vzdálený přístup z internetu.

Shrneme-li závěrem praktické využití Arduina, lze říci, že díky své přijatelné ceně, otevřenému zdrojovému kódu a široké komunitě lidí, začíná být Arduino čím dál více populární.

ZÁVĚR V ANGLIČTINĚ

The aim of the thesis was to get to know with the Arduino project and its use as a Web server. To create a server was used development board Arduino Mega 2560 and an additional board Ethernet shield which arrange communication between Arduino and Ethernet. On the he used programming pins were connected LEDs, potentiometers and switches.

In the introduction of the theoretical part were explained the basic concepts such as a microprocessor, microcontroller, microcontroller, digital signal processor or digital signal controller. Then are described main terms of microcomputers, its history and then set distribution of instruction and concept. Following part is devote an introduction of Arduino project which describes its history, analysis and comparison of the development and option boards, explanation with the development environment as its programming. The practical part consists of the particular components including properties. Once described the involvement of individual electronic components, including a diagram calculations. Finally it explains how to create the program by entering the initial analysis, design solution, through development diagram to create the actual source code.

Hardware involvement of the programmed analog pins, which are connected resistive trimmers through which we can linearly change the input voltage . The programmed digital input pins are connected micro-switch with which you can control the LEDs on the bargraph and are connected to the Arduino via the digital output pins . These output pins can be controlled remotely via a web server.

Arduino is programmed to allow the user remotely connect to a Web server. Using a mobile phone should automatically adapted content of the web page on the screen and that could easily switch between tabs control and display. In the tab control buttons are arranged similarly as in the numeric keypad. It can control the digital outputs. In the tab view, see the current values of analog inputs (trimmers) and status LEDs, which are updated every 10 seconds so that there is a page refresh . When using the Arduino as a web server I would recommend using only an Ethernet network that is for home or business network. Because the Arduino not to dispose of enough numerical performance for data communication security, I would not recommend remote access from the Internet.

To summarize the conclusion of the practical use of Arduino we can say that thanks to its affordable price, open source code and a community of people Arduino is becoming increasingly popular.

SEZNAM POUŽITÉ LITERATURY

ALLDATASHEET.COM. Atmega 2560 Datasheet – ATMEL Corporation. © 2003 - 2014. [online]. [cit. 2014-05-03]. Dostupný z: <http://pdf1.alldatasheet.com/datasheet-pdf/view/107092/ATMEL/ATMEGA2560.html>

ARDUINO. © 2014. [online]. 21.10.2008 [cit. 2014-04-17]. Dostupný z: <http://arduino.cc/en>

AUGARTEN, Stan. State of the art: a photographic history of the integrated circuit. New Haven: Ticknor, 1983. ISBN 0-89919-195-9.

BANZI, Massimo. Getting Started with Arduino. Second Edition. U.S.A.: O'Reilly Media, 2011. ISBN 978-1-449-30987-9.

BUMBA, Jiří. Programování mikroprocesorů: praktický návod nejen pro mikroprocesory PIC. Vyd. 1. Brno: Computer Press, 2011, 135 s. ISBN 978-80-251-2838-1.

CADY, Frederick M. Microcontrollers and microcomputers: principles of software and hardware engineering. 2Nd ed. New Yourk: Oxford University Press, 2010, xii, 477 s. ISBN 978-0-19-537161-1.

CORE 77. com. King Arduino from Ivrea. © 2014 . [online]. 21.10.2008 [cit. 2014-04-17]. Dostupný z: http://www.core77.com/blog/technology/king_arduino_from_ivrea_11506.asp

ELEKTROREVUE: internetový časopis. Využití digitálních signálových kontrolérů při řízení elektrických pohonů. VŠB-Technická Univerzita Ostrava, Fakulta elektrotechniky a informatiky. 2010, č. 06. ISSN 1213 – 1539. Dostupný z: <http://www.elektrorevue.cz>

FRÝZA, Tomáš. Bloková struktura mikrokontroleru: Mikroprocesorova technika a embedded systemy. Přednáška 1. Vysoké učení v Brně. © 2013. [cit. 2014-04-10]. Dostupný z: http://www.urel.feec.vutbr.cz/~fryza/downloads/mpt_pred_01.pdf

GM electronic: elektronika, kterou neznáte. [online]. © 1990 – 2014. [cit. 2014-05-08]. Dostupný z: <http://www.gme.cz/l-bargraf-y-p512-106>

HOBBYROBOT.CZ. Malá robotika a mechanické modely. Arduino – příručka programátora. © 2014 . [online]. [cit. 2014-04-17]. Dostupný z: <http://www.hobbyrobot.cz/wp-content/uploads/ArduinoPriruckaProgramatora.pdf>

HWKITCHEN.COM. © 2014. [online]. [cit. 2014-04-17]. Dostupný z: <http://www.hwkit-chen.com/news/>

LABTF. Počítačová laboratoř v Tróji LabTF. Jednočipové mikropočítače. © 2013. [online]. [cit. 2014-04-17]. Dostupný z: <http://lucy.troja.mff.cuni.cz/~tichy/elektronika/kap9/jednocpoc.html>

LUPA.CZ, server o českém Internetu. © 1998 – 2014 Internet Info, s.r.o. Všechna práva vyhrazena. Powered by Linux. ISSN 1213-0702. [online]. [cit. 2014-04-17]. Dostupný z: <http://www.lupa.cz/clanky/creative-commons-8211-budoucnost-copyrightu/>

MOTOUŠEK, David. Práce s mikrokontroléry ATMEL AVR. 2.vyd. Praha: BEN, 2006, 375 s. ISBN 80_7300-209-4.

NEVORAL, Jiří. Řízení měniče. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 42 s. Vedoucí bakalářské práce doc. Ing. Pavel Fiala, Ph.D

OZDEN, Ozgur. Journey Towards Completing MSC Degree. CIS architecture and RISC architecture. [online]. 08.02.2013 [cit. 2014-04-17]. Dostupný z: <http://oozden.wordpress.com/2013/02/08/cisc-architecture-and-risc-architecture/>

PINKER, Jiří. Mikroprocesory a mikropočítače. Praha: BEN – technická literatura, 2004, 159 s. ISBN 80-730-0110-1.

ROOT.CZ. © 1998 – 2014. [online]. [cit. 2014-04-17]. Dostupný z: <http://www.root.cz>

SWITCH2MAC. Stručná historie mikropočítačů – Úsvit. In: Blogger [online]. [cit. 2014-04-17]. Dostupný z: <http://switch2mac.blog.zive.cz/2010/02/strucna-historie-mikropocitacu-usvit>

TECHNICAL DEVICES FOR SUPERVISING OF A HOUSEHOLD VIA INTERNET BASED ON ARDUINO MICROCONTROLLER. Home Page. 2012. Dostupné z: <http://www.wseas.us/e-library/conferences/2012/Instanbul/AICE/AICBE-40.pdf>

TRANSFER MULTISORT ELEKTRONIK, © 2014. [online]. [cit. 2014-05-06]. Dostupný z: http://www.tme.eu/html/PL/mikrokontrolery-8-bit-firmy-atmel-serii-at89/ramka_1571_PL_pelny.html

VÁŇA, Vladimír. ARM pro začátečníky. 1 vyd. Praha: BEN – technická literatura, 2009. ISBN 978-80-7300-2.

WIZNET. [online]. [cit. 2014-05-26]. Dostupný z:
http://www.wiznet.co.kr/UpLoad_Files/ReferenceFiles/W5100_Datasheet_v1.2.4.pdf

μART.CZ. Arduino a USB. © 2014. [online]. [cit. 2014-04-17]. Dostupný z:
<http://uart.cz/394/arduino-a-usb/>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ALU	Aritmetická a logická jednotka
BBB	Bare Bones Boards
CISC	Complex Instruction Set Computer
CPU	Centrální řídicí jednotka
DCS	Digitální signálový kontrolér
DPS	Deska plošných spojů
DSP	Digitální signálový procesor
FTDI	Future Technology Devices International
GND	Ground
LCD	Liquid Crystal Display
ICSP	In Circuit Serial Programming
IDE	Intergrated Development Enviroment
IO	Integrovaný obvod
LED	Dioda emitující světlo
MCU	Mikrokontrolér
MISO	Master In/Slave Out
MOSI	Master Out/Slave In
MPU	Mikroprocesor
OCDS	On-Chip Debug System
RISC	Reduced Instruction Set Computing
SA	Share Alike
UART	Universal Synchronous/ Asynchronous Receiver and Transmitter
USB	Universal Serial Bus
UTP	Unshielded Twisted Pair
PWM	Pulse Width Modulation

SEZNAM OBRÁZKŮ

Obr. 1 Obecné blokové schéma 8bitového jednočipového mikropočítače.....	12
Obr. 2 Mikrokontrolér.....	13
Obr. 3 Arduino IDE.....	23
Obr. 4. Blokové schéma propojení Arduina a USB převodníku.....	27
Obr. 5. ICSP konektor.....	28
Obr. 6. Arduino Mega 2560.....	30
Obr. 7. Blokové schéma ATmega 2560.....	31
Obr. 8. Arduino Ethernet shield.....	33
Obr. 9. Blokové schéma řadiče W5100.....	34
Obr. 10. LED bargraf a jeho vnitřní zapojení.....	35
Obr. 11. Trimr.....	36
Obr. 12. Blokové schéma mikropsínače.....	36
Obr. 13. Mikropsínač.....	37
Obr. 14. Zkušební nepájivé kontaktní pole.....	37
Obr. 15. Schéma zapojení tlačítka.....	38
Obr. 16. Schéma zapojení Arduina.....	40
Obr. 17. Návrh webu.....	41
Obr. 18. Vývojový diagram.....	43
Obr. 19. Grafické znázornění webové stránky.....	44
Obr. 20. Kompletní zapojení Arduina.....	47
Obr. 21. HTML stránka serveru Arduina na mobilním telefonu.....	47

SEZNAM TABULEK

Tab. 1. Druhy Arduino desek.....20

SEZNAM PŘÍLOH

Příloha P 1: Webová stránka Ovládání

Příloha P 2: Webová stránka Zobrazení

Příloha P 3: zdrojový kód ARDUINA

PŘÍLOHA P 1: WEBOVÁ STRÁNKA OVLÁDÁNÍ

```
<!DOCTYPE HTML>
<head>
<title>Ethernet Switching</title>

<meta name="description" content="Arduino ethernet"/>
<meta http-equiv="refresh" content="10; url=/">
<meta name="apple-mobile-web-app-capable" content="yes">
<meta name="apple-mobile-web-app-status-bar-style" content="default">
<meta name="viewport" content="width=device-width, user-scalable=no"/>
<meta charset="windows-1250">
<meta charset="UTF-8">
<style type="text/css">

html {height:100%;}
body {background-color: #000066;}

.nazev {
font-size: 21px;
text-align: center;
font-weight: bold;
color: #ffffff;}

.rozlozeni {
background-color: #ffffff;
margin: 2px;
border: solid 1px #000000;
border-radius: 10px;
margin-left:auto;
margin-right:auto;}

.list {
font-size: 16px;
text-align: center;
font-weight: bold;
color: #ffffff;
text-shadow: #000000 3px 0;
background-color: #0033cc;
border: solid 1px #000000;
border-radius: 10px;}

.prepinani0 {width: 109px;}
.prepinani1 {width: 109px; font-weight: bold;}
.tlacitka0 {width: 70px; height: 70px;}
.tlacitka1 {width: 70px; height: 70px; font-weight: bold;}
.vse0 {width: 220px;}
.vse1 {width: 220px; font-weight: bold;}
</style>
</head>
```

```

<body>
<div class="navez">ARDUINO ETHERNET</div>

<FORM>
<table class="rozlozeni" border="0" align="center">
<tr><td colspan="3" class="list">O V L Á D Á N Í</td></tr>

<tr><td colspan="3"><input type="button" class="prepinani0" value="ZOBRAZENÍ"
onClick="\parent.location=/?A\">
<input type="button" class="prepinani1" value="OVLÁDÁNÍ"
onClick="\parent.location=/?B\"></td></tr>

<tr><td><input type="button" class="tlacitka0" value="TL 1"
onClick="\parent.location=/?0\"></td>
<td><input type="button" class="tlacitka0" value="TL 2" onClick="\parent.location=/?
1\"></td>
<td><input type="button" class="tlacitka0" value="TL 3" onClick="\parent.location=/?
2\"></td></tr>

<tr><td><input type="button" class="tlacitka0" value="TL 4"
onClick="\parent.location=/?3\"></td>
<td><input type="button" class="tlacitka0" value="TL 5" onClick="\parent.location=/?
4\"></td>
<td><input type="button" class="tlacitka0" value="TL 6" onClick="\parent.location=/?
5\"></td></tr>

<tr><td><input type="button" class="tlacitka0" value="TL 7"
onClick="\parent.location=/?6\"></td>
<td><input type="button" class="tlacitka0" value="TL 8" onClick="\parent.location=/?
7\"></td>
<td><input type="button" class="tlacitka0" value="TL 9" onClick="\parent.location=/?
8\"></td></tr>
<tr><td colspan="3"><input type="button" class="vse0" value="V Š E Z A P N O U T"
onClick="\parent.location=/?9\"></td></tr>

</table></FORM>
</body></html>

```

PŘÍLOHA P 2: WEBOVÁ STRÁNKA ZOBRAZENÍ

```
<!DOCTYPE HTML>
<head>
<title>Ethernet Switching</title>

<meta name="description" content="Arduino ethernet"/>
<meta http-equiv="refresh" content="10; url=/">
<meta name="apple-mobile-web-app-capable" content="yes">
<meta name="apple-mobile-web-app-status-bar-style" content="default">
<meta name="viewport" content="width=device-width, user-scalable=no"/>
<meta charset="windows-1250">
<meta charset="UTF-8">
<style type="text/css">

html {height:100%;}
body {background-color: #000066;}

.nazev {
font-size: 21px;
text-align: center;
font-weight: bold;
color: #ffffff;}

.rozlozeni {
background-color: #ffffff;
margin: 2px;
border: solid 1px #000000;
border-radius: 10px;
margin-left:auto;
margin-right:auto;}

.list {
font-size: 16px;
text-align: center;
font-weight: bold;
color: #ffffff;
text-shadow: #000000 3px 0;
background-color: #0033cc;
border: solid 1px #000000;
border-radius: 10px;}

.prepinani0 {width: 109px;}
.prepinani1 {width: 109px; font-weight: bold;}
.tlacitka0 {width: 70px; height: 70px;}
.tlacitka1 {width: 70px; height: 70px; font-weight: bold;}
.vse0 {width: 220px;}
.vse1 {width: 220px; font-weight: bold;}
</style>
</head>
```

```
<body>
<div class="navez">ARDUINO ETHERNET</div>

<FORM>
<table class='rozlozeni' border='0' align='center'>
<tr><td colspan='3' class='list'>Z O B R A Z E N Í</td></tr>
<tr><td colspan='3'><input type='button' class='prepinani1' value='ZOBRAZENÍ'
onClick=\"parent.location=?A\">
<input type='button' class='prepinani0' value='OVLÁDÁNÍ' onClick=\"parent.location=?
B\"></td></tr>

<tr><td>Trimr 1 : 0-1023</td></tr>
<tr><td></td></tr>

<tr><td>LED 1 : 0/1</td></tr>
<tr><td></td></tr>

</table></FORM>
</body></html>
```

PŘÍLOHA P 3: ZDROJOVÝ KÓD ARDUINA

```
#include <Ethernet.h>
#include <SPI.h>

byte ip[] = { 10, 0, 0, 200 };
byte gateway[] = { 10, 0, 0, 110 };
byte subnet[] = { 255, 255, 255, 0 };
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
EthernetServer server = EthernetServer(80);

int ZacVystLed = 30;
int PosVystLed = 39;

char c;
char zasobnik = 'B';
char otaznik;
boolean cteni = false;
boolean hlavicka;
boolean cteniPin[10];

void setup(){
  Serial.begin(9600);

  for (int i = ZacVystLed; i <= PosVystLed + 1; i++) {
    pinMode(i, OUTPUT);
  }

  Ethernet.begin(mac, ip, gateway, subnet);
  server.begin();
  Serial.println(Ethernet.localIP());
}

void loop(){

  int tlac1 = digitalRead(40);
  int tlac2 = digitalRead(41);
  int tlac3 = digitalRead(42);
  int tlac4 = digitalRead(43);
  int tlac5 = digitalRead(44);
  int tlac6 = digitalRead(45);
  int tlac7 = digitalRead(46);
  int tlac8 = digitalRead(47);
  int tlac9 = digitalRead(48);

  if (tlac1 == 1){
    digitalWrite(30, !cteniPin[0]);
    cteniPin[0] = !cteniPin[0];
  }
}
```

```

while (tlac1 == 1)
{
  tlac1 = digitalRead(42);
  delay(10);
}
}

if (tlac2 == 1){
digitalWrite(31, !cteniPin[1]);
cteniPin[1] = !cteniPin[1];
  while (tlac2 == 1)
  {
    tlac2 = digitalRead(44);
    delay(10);
  }
}

if (tlac3 == 1){
digitalWrite(32, !cteniPin[2]);
cteniPin[2] = !cteniPin[2];
  while (tlac3 == 1)
  {
    tlac3 = digitalRead(46);
    delay(10);
  }
}

if (tlac4 == 1){
digitalWrite(33, !cteniPin[3]);
cteniPin[3] = !cteniPin[3];
  while (tlac4 == 1)
  {
    tlac4 = digitalRead(48);
    delay(10);
  }
}

if (tlac5 == 1){
digitalWrite(34, !cteniPin[4]);
cteniPin[4] = !cteniPin[4];
  while (tlac5 == 1)
  {
    tlac5 = digitalRead(50);
    delay(10);
  }
}

if (tlac6 == 1){
digitalWrite(35, !cteniPin[4]);
cteniPin[4] = !cteniPin[4];
}

```

```

while (tlac5 == 1)
{
  tlac5 = digitalRead(50);
  delay(10);
}
}

if (tlac7 == 1){
digitalWrite(36, !cteniPin[4]);
cteniPin[4] = !cteniPin[4];
  while (tlac5 == 1)
  {
    tlac5 = digitalRead(50);
    delay(10);
  }
}

if (tlac8 == 1){
digitalWrite(37, !cteniPin[4]);
cteniPin[4] = !cteniPin[4];
  while (tlac5 == 1)
  {
    tlac5 = digitalRead(50);
    delay(10);
  }
}

if (tlac8 == 1){
digitalWrite(38, !cteniPin[4]);
cteniPin[4] = !cteniPin[4];
  while (tlac5 == 1)
  {
    tlac5 = digitalRead(50);
    delay(10);
  }
}

checkForClient();
}

void checkForClient(){

EthernetClient client = server.available();

if (client) {
  hlavicka = false;

  while (client.connected()) {
    if (client.available()) {

```

```
if(!hlavicka){
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("Connnection: close");
  client.println();

  client.println("<!DOCTYPE HTML>");
  client.println("<head>");
  client.println("<title>Ethernet Switching</title>");

  client.println("<meta name='description' content='Arduino ethernet'/>");
  client.println("<meta http-equiv='refresh' content='10; url=/'>");
  client.println("<meta name='apple-mobile-web-app-capable' content='yes'>");
  client.println("<meta name='apple-mobile-web-app-status-bar-style'
content='default'>");
  client.println("<meta name='viewport' content='width=device-width, user-
scalable=no'/>");
  client.println("<meta charset='UTF-8'>");
  client.println("<style type='text/css'>");
  client.println("");

  client.println("html {height:100%;}");

  client.println("body {background-color: #000066;}");

  client.println(".nazev {");
  client.println("font-size: 21px;");
  client.println("text-align: center;");
  client.println("font-weight: bold;");
  client.println("color: #ffffff;}");

  client.println(".rozlozeni {");
  client.println("background-color: #ffffff;");
  client.println("margin: 2px;");
  client.println("border: solid 1px #000000;");
  client.println("border-radius: 10px;");
  client.println("margin-left:auto;");
  client.println("margin-right:auto;}");

  client.println(".list {");
  client.println("font-size: 16px;");
  client.println("text-align: center;");
  client.println("font-weight: bold;");
  client.println("color: #ffffff;");
  client.println("text-shadow: #000000 3px 0;");
  client.println("background-color: #0033cc;");
  client.println("border: solid 1px #000000;");
  client.println("border-radius: 10px;}");

  client.println(".prepinani0 {width: 109px;}");
  client.println(".prepinani1 {width: 109px; font-weight: bold;}");
```

```

client.println(".tlacitka0 {width: 70px; height: 70px;}");
client.println(".tlacitka1 {width: 70px; height: 70px; font-weight: bold;}");

client.println(".vse0 {width: 220px;}");
client.println(".vse1 {width: 220px; font-weight: bold;}");

client.println("</style>");
client.println("</head>");

client.println("<body>");
client.println("<div class='nazev'>ARDUINO ETHERNET</div>");
client.println();

hlavicka = true;
}

char c = client.read(); // GET /?3 HTTP/1.1

if(cteni){
  if(c == 'A'){
    zasobnik = 'A';
  }
  if(c == 'B'){
    zasobnik = 'B';
  }
  if(zasobnik == 'A'){
    vypisHodnot(client);
    cteni = false;
    break;
  }
  if(zasobnik == 'B'){

    if(c == '9'){
      if(cteniPin[9] == 0){
        for (int i = 0; i < 10; i++){
          digitalWrite(i + ZacVystLed, 1);
        }
      }else{
        for (int i = 0; i < 10; i++){
          digitalWrite(i + ZacVystLed, 0);
        }
      }
    } else{
      digitalWrite(atoi(&c) + ZacVystLed, !cteniPin[atoi(&c)]);
      digitalWrite(ZacVystLed + 9, 0);
    }
  }
  vypisTlacitek(client);
  cteni = false;
  break;
}

```

```

    if (c != '?'){
        cteni = false;
    }
}

if(c == '?') {
    cteni = true;
}

if (c == '\n'){
    if(zasobnik == 'A'){
        vypisHodnot(client);
        break;
    } if(zasobnik == 'B'){
        vypisTlacitek(client);
        break;
    }
}
}
}

delay(1);
client.stop();
}
}

void vypisTlacitek(EthernetClient client){

    for (int i = 0; i < 10; i++) {
        cteniPin[i] = digitalRead(i + ZacVystLed);
    }

    client.println("<FORM>");
    client.println("<table class='rozlozeni' border='0' align='center'>");
    client.println("<tr><td colspan='3' class='list'>O V L Á D Á N Í</td></tr>");

    client.println("<tr><td colspan='3'><input type='button' class='prepinani0'
value='ZOBRAZENÍ' onClick='\"parent.location=?A\">");
    client.println("<input type='button' class='prepinani1' value='OVLÁDÁNÍ'
onClick='\"parent.location=?B\"></td></tr>");

    if (cteniPin[0] == 0){
        client.println("<tr><td><input type='button' class='tlacitka0' value='TL 1'
onClick='\"parent.location=?0\"></td>");
    } else {
        client.println("<tr><td><input type='button' class='tlacitka1' value='TL 1'
onClick='\"parent.location=?0\"></td>");
    } if (cteniPin[1] == 0){
        client.println("<td><input type='button' class='tlacitka0' value='TL 2'
onClick='\"parent.location=?1\"></td>");
    } else {

```

```

        client.println("<td><input type='button' class='tlacitka1' value='TL 2'
onClick=\"parent.location=?1\"></td>");
        } if (cteniPin[2] == 0){
            client.println("<td><input type='button' class='tlacitka0' value='TL 3'
onClick=\"parent.location=?2\"></td></tr>");
        } else {
            client.println("<td><input type='button' class='tlacitka1' value='TL 3'
onClick=\"parent.location=?2\"></td></tr>");

        } if (cteniPin[3] == 0){
            client.println("<tr><td><input type='button' class='tlacitka0' value='TL 4'
onClick=\"parent.location=?3\"></td>");
        } else {
            client.println("<tr><td><input type='button' class='tlacitka1' value='TL 4'
onClick=\"parent.location=?3\"></td>");
        } if (cteniPin[4] == 0){
            client.println("<td><input type='button' class='tlacitka0' value='TL 5'
onClick=\"parent.location=?4\"></td>");
        } else {
            client.println("<td><input type='button' class='tlacitka1' value='TL 5'
onClick=\"parent.location=?4\"></td>");
        } if (cteniPin[5] == 0){
            client.println("<td><input type='button' class='tlacitka0' value='TL 6'
onClick=\"parent.location=?5\"></td></tr>");
        } else {
            client.println("<td><input type='button' class='tlacitka1' value='TL 6'
onClick=\"parent.location=?5\"></td></tr>");

        } if (cteniPin[6] == 0){
            client.println("<tr><td><input type='button' class='tlacitka0' value='TL 7'
onClick=\"parent.location=?6\"></td>");
        } else {
            client.println("<tr><td><input type='button' class='tlacitka1' value='TL 7'
onClick=\"parent.location=?6\"></td>");
        } if (cteniPin[7] == 0){
            client.println("<td><input type='button' class='tlacitka0' value='TL 8'
onClick=\"parent.location=?7\"></td>");
        } else {
            client.println("<td><input type='button' class='tlacitka1' value='TL 8'
onClick=\"parent.location=?7\"></td>");
        } if (cteniPin[8] == 0){
            client.println("<td><input type='button' class='tlacitka0' value='TL 9'
onClick=\"parent.location=?8\"></td></tr>");
        } else {
            client.println("<td><input type='button' class='tlacitka1' value='TL 9'
onClick=\"parent.location=?8\"></td></tr>");

        } if (cteniPin[9] == 0){
            client.println("<tr><td colspan='3'><input type='button' class='vse0' value='V Š E Z A
P N O U T' onClick=\"parent.location=?9\"></td></tr>");
        } else {

```

```

    client.println("<tr><td colspan='3'><input type='button' class='vsel' value='V Š E V Y  
P N O U T' onClick='\"parent.location=/'?9'\"></td></tr>");
    }

    client.println("</table>");
    client.println("</FORM>");
    client.println("</body></html>");
    }

void vypisHodnot(EthernetClient client){
    client.println("<FORM>");
    client.println("<table class='rozlozeni' border='0' align='center'>");

    client.println("<tr><td colspan='3' class='list'>Z O B R A Z E N Í</td></tr>");

    client.println("<tr><td colspan='3'><input type='button' class='prepinani1'  
value='ZOBRAZENÍ' onClick='\"parent.location=/'?A'\">");
    client.println("<input type='button' class='prepinani0' value='OVLÁDÁNÍ'  
onClick='\"parent.location=/'?B'\"></td></tr>");

    for (int i = 0; i < 6; i++) {
        int hodnotaVstupu = analogRead(i);
        client.print("<tr><td>Trimr ");
        client.print(i+1);
        client.print(" : ");
        client.print(hodnotaVstupu);
        client.println("</td></tr>");
    }

    client.print("<tr><td></td></tr>");

    for (int i = 0; i < 9; i++) {
        int hodnotaVstupu = digitalRead(i + ZacVystLed);
        client.print("<tr><td>LED ");
        client.print(i+1);
        client.print(" : ");
        client.print(hodnotaVstupu);
        client.println("</td></tr>");
    }

    client.println("</table>");
    client.println("</FORM>");
    client.println("</body></html>");
}

```