



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**

BRNO UNIVERSITY OF TECHNOLOGY



**FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH  
TECHNOLOGIÍ**

**ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY**

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION  
DEPARTMENT OF CONTROL AND INSTRUMENTATION

## **MATICOVÁ ZOBRAZOVACÍ JEDNOTKA**

MATRIX DISPLAY

**DIPLOMOVÁ PRÁCE**

MASTER'S THESIS

**AUTOR PRÁCE**

AUTHOR

**Bc. MATEJ GÁBIK**

**VEDOUCÍ PRÁCE**

SUPERVISOR

**doc. Ing. ZDENĚK BRADÁČ, Ph.D.**

BRNO 2015



VYSOKÉ UČENÍ  
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky  
a komunikačních technologií

Ústav automatizace a měřicí techniky

# Diplomová práce

magisterský navazující studijní obor  
**Kybernetika, automatizace a měření**

**Student:** Bc. Matej Gábik

**ID:** 134478

**Ročník:** 2

**Akademický rok:** 2014/2015

**NÁZEV TÉMATU:**

**Maticová zobrazovací jednotka**

## POKYNY PRO VYPRACOVÁNÍ:

V rámci diplomové práce navrhnete a realizujete autonomní informační portál, který umožní zobrazovat data a animace na maticovém displeji BUSE.

1. Provedte literární rešerši.
2. Navrhnete a realizujete koncepci systému jako kompaktního mikropočítačového systému s funkcí měření fyzikálních parametrů v interiéru.
3. Vytvořte vhodné komunikační rozhraní a připojení do nadřazeného systému.
4. Navrhnete a realizujete elektroniku systému, osadíte ji součástkami a oživíte.
5. Vytvořte komplexní programové vybavení, které umožní ovládání a administrování systému.
6. Otestujte funkčnost a demonstřujte ji.

## DOPORUČENÁ LITERATURA:

Pavel Herout: Učebnice jazyka C, KOPP, 2004, IV. přepracované vydání, ISBN 80-7232-220-6  
Dle pokynů vedoucího práce.

**Termín zadání:** 9.2.2015

**Termín odevzdání:** 18.5.2015

**Vedoucí práce:** doc. Ing. Zdeněk Bradáč, Ph.D.

**Konzultanti diplomové práce:**

**doc. Ing. Václav Jirsík, CSc.**

*Předseda oborové rady*

## UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

## **Abstrakt**

Cieľom diplomovej práce je realizovať autonómny informačný portál s použitím maticového displeja BUSE BS210. Prvá časť práce sa zaoberá teoretickým riešením problému a možnosťami riadenia displeja. V druhá časť je venovaná návrhu potrebného hardvéru a v poslednej časti práce je popísané programové vybavenie mikrokontroléra a mikropočítača Raspberry Pi.

## **Kľúčové slová**

Raspberry Pi, AtMega 128, BUSE BS210, I<sup>2</sup>C, UART, teplota, tlak, vlhkosť, MySQL.

## **Abstract**

The aim of this thesis is to implement autonomous information portal using matrix display BUSE BS210. The first part deals with the theoretical problem-solving and management options display. The second part is dedicated to the development of necessary hardware and the last part is the software described herein microcontroller and microcomputer Raspberry Pi.

## **Keywords**

Raspberry Pi, AtMega 128, BUSE BS210, I<sup>2</sup>C, UART, temperature, pressure, humidity, MySQL

### **Bibliografická citace:**

GÁBIK, M. *Maticová zobrazovací jednotka*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. 79s. Vedoucí diplomové práce byl doc. Ing. Zdeněk Bradáč, Ph. D..

## **Prohlášení**

„Prohlašuji, že svou diplomovou práci na téma Maticová zobrazovací jednotka jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **18. května 2015**

.....  
podpis autora

## **Poděkování**

Děkuji vedoucímu diplomové práce doc. Ing. Zdeněkovi Bradáči, Ph. D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **18. května 2015**

.....  
podpis autorara

# Obsah

1.	Úvod.....	11
2.	Teoretický rozbor.....	12
2.1	Riadenie panelu BUSE BS210.....	13
2.1.1.	Priamy výber stĺpca pomocou mikrokontroléra .....	15
2.1.2.	Výber stĺpca pomocou posuvného registra .....	16
2.1.3.	Výber pomocou dekóderu 1 z n .....	17
2.2	Požiadavky na hardvér .....	17
3.	Hardvér .....	19
2.4	Napájací zdroj .....	19
2.4.1	Znižujúce napäťové regulátory.....	21
3.2.	Obvod s mikrokontrolérom .....	22
3.2.1.	JTAG .....	24
3.2.2.	I <sup>2</sup> C zbernica .....	25
3.2.3.	UART .....	25
3.3.	RTC hodiny .....	25
3.4.	Výber riadku.....	26
3.4.1.	Zapojenie logickej časti pre výber.....	26
3.4.2.	Budiče.....	29
3.5.	Raspberry Pi.....	31
3.6.	Meteostanica .....	32
3.6.1.	Meranie teploty.....	32
3.6.2.	Meranie atmosférického tlaku .....	34
3.6.3.	Meranie vlhkosti.....	36
3.6.4.	Oživenie hardvéru .....	37
4.	softvér .....	38
4.2.	Programové vybavenie AtMega128.....	38
4.2.1.	global.h.....	38
4.2.2.	uart.h.....	39
4.2.3.	matrix.h.....	42
4.2.4.	main.c .....	46
4.3.	Programové vybavenie Raspberry Pi .....	46
4.3.1.	Operačný systém .....	46
4.3.2.	Web server.....	49
4.3.3.	Databáza .....	49
4.3.4.	Spracovanie dát zo snímačov .....	51
4.3.5.	Užívateľské rozhranie .....	56
4.3.6.	Komunikácia s mikrokontrolérom.....	64
4.3.7.	Predvedenie činnosti.....	64

5. Záver .....	66
Literatúra .....	68
Zoznam príloH .....	70

## ZOZNAM OBRÁZKOV

Obr. 1 Detail dvoch diskov z panelu BUSE.....	12
Obr. 2 Viditeľné poškodenie displeja .....	13
Obr. 3 Bloková schéma posuvného registra.....	16
Obr. 4 Navrhnutá koncepcia zariadenia .....	18
Obr. 5 Schéma zapojenia 24 a 12 V vetvy zdroja .....	20
Obr. 6 Schéma zapojenia 5 V časti zdroja .....	21
Obr. 7 Osadený napájací zdroj .....	21
Obr. 8 Rozloženie pinov mikrokontroléra AtMega128[9].....	23
Obr. 9 Osadená doska s mikrokontrolérom.....	24
Obr. 10 Rozloženie pinov na JTAG konektore.....	24
Obr. 11 Zapojenie hodín reálneho času .....	26
Obr. 12 Schéma zapojenia logickej časti .....	28
Obr. 13 DPS logickej a výkonovej časti .....	29
Obr. 14 Zapojenie jedného kanála obvodu ULN2803A [14].....	29
Obr. 15 Vnútorne zapojenie jedného kanála obvodu TD62783 [15].....	30
Obr. 16 Rozloženie pinov obvodu TD62783 [15] .....	30
Obr. 17 Mikropočítač Raspberry Pi .....	31
Obr. 18 základné zapojenie termočlánku [17] .....	33
Obr. 19 Blokové zapojenie snímača TMP102[18].....	34
Obr. 20 Zapojenie barometra BMP180[19] .....	35
Obr. 21 Vývojový diagram funkcie doAnswer() .....	41
Obr. 22 Vývojový diagram funkcie setGlobalDataCmd() .....	42
Obr. 23 Vývojový diagram funkcie writePicture() .....	44
Obr. 24 Program Win32DiskImager .....	47
Obr. 25 Okno konfigurácie Raspberry Pi.....	47
Obr. 26 Program PuTTY .....	48
Obr. 27 Výpis stĺpcov z tabuľky canvas .....	51
Obr. 28 Neformátované menu.....	56
Obr. 29 Menu formátované pomocou CSS.....	57
Obr. 30 Hlavička tabuľky z webovej stránky .....	57
Obr. 31 Grafické zobrazenie zmeraných hodnôt .....	59
Obr. 32 Dialógové okno pre výber obrázku.....	61
Obr. 33 Rozloženie ovládacích prvkov .....	61
Obr. 34 Načítanie dát z databázy a vloženie do obrazu .....	62
Obr. 35 Vytvorená znaková sada pre displej BUSE .....	63
Obr. 36 Vytvorený textový reťazec .....	64
Obr. 37 Vykreslený textový reťazec .....	64
Obr. 38 Vykreslené dáta z meteostanice .....	65

## **ZOZNAM TABULIEK**

Tab. 1 Význam jednotlivých pinov na paneli BUSE BS210 .....	14
Tab. 2 Pravdivostná tabuľka dekóderu 74HC238 .....	17
Tab. 3 Spotreba jednotlivých prvkov .....	19
Tab. 4 Formát odosielaných a prijímaných dát .....	39
Tab. 5 Načítanie kalibračných údajov .....	52
Tab. 6 Čítanie teploty .....	52
Tab. 7 Čítanie tlaku .....	53
Tab. 8 Výpočet skutočnej teploty .....	53
Tab. 9 Výpočet skutočného tlaku .....	53

# 1. ÚVOD

Cieľom práce bolo vytvoriť informačný portál, ktorý umožní zobrazovať dáta a animácie na maticovom displeji BUSE BS210, ktorý je primárne používaný v mestskej hromadnej doprave pre zobrazenie čísla linky a smeru jazdy. Prvá časť práce je venovaná samotnému displeju a možnosťami jeho riadenia. Sú spomenuté jeho výhody, nevýhody a návrh možného riešenia pomocou vhodného hardvéru.

V druhej časti sa venujeme popisu potrebného hardvéru od napájacieho zdroja po výber vhodného mikrokontroléra, ktorý slúži ako radič displeja. V podkapitolách vysvetľujeme základné princípy použitých rozhraní ako je JTAG, použitý pre naprogramovanie mikrokontroléra, dvojvodičová zbernica I2C, ktorú používame pre komunikáciu so snímačmi a sériová linka UART, použitá pre prenos dát z mikropočítača Raspberry Pi do mikrokontroléra. Záverečná časť kapitoly je venovaná druhej časti zadania, meteostanici. Zariadenie by malo byť v konečnej fáze schopné zobrazovať základné fyzikálne veličiny ako je teplota, tlak a relatívna vlhkosť vzduchu. Kapitola 3.6 sa zaoberá základnými možnosťami merania týchto veličín a výberom snímačov pre toto meranie.

Kapitola 4 obsahuje popis programovateľného vybavenia všetkých častí. V prvej polovici sa jedná o programové vybavenie mikrokontroléra AtMega 128. Popisujeme koncepciu jednotlivých vytvorených modulov pre riadenie displeja, a pre lepšie pochopenie sú uvedené vývojové diagramy zložitejších funkcií. Druhá polovica začína popisom inštalácie operačného systému na platformu Raspberry Pi a spustenie SSH serveru. Nasleduje spustenie webového serveru pomocou aplikácie Apache 2 a vytvorenie databázy pomocou jazyka SQL. Spracovanie dát zo snímačov prebieha pomocou skriptu napísaného v jazyku python, ktorý je pre tento mikropočítač veľmi obľúbený.

Posledná časť tejto kapitoly sa zaoberá vytvorením vhodného užívateľského rozhrania pre riadenie displeja a zobrazenie zosnímaných dát zo snímačov. Vytvorili sme niekoľko stránok pomocou kombinácie jazykov HTML, PHP a JavaScript. Tieto stránky sú dostupné po pripojení mikropočítača na sieť pomocou ethernetového rozhrania a zadaním pridelenej IP adresy do prehliadača na počítači. Pre pohodlné ovládanie displeja sme vytvorili niekoľko možností vytvoriť požadovaný obraz. Je možné zadávať text priamo z klávesnice, ktorý sa automaticky vkladá do obrázka. Ďalej je možné vložiť už vytvorený obraz alebo načítať posledné zmerané dáta zo snímačov a vložiť ich do zobrazovanej časti. Pre opravy alebo vytvorenie je možné taktiež meniť farbu jednotlivých pixelov ručne. Tým získava užívateľ plnú kontrolu nad zobrazovaným textom alebo obrazom.

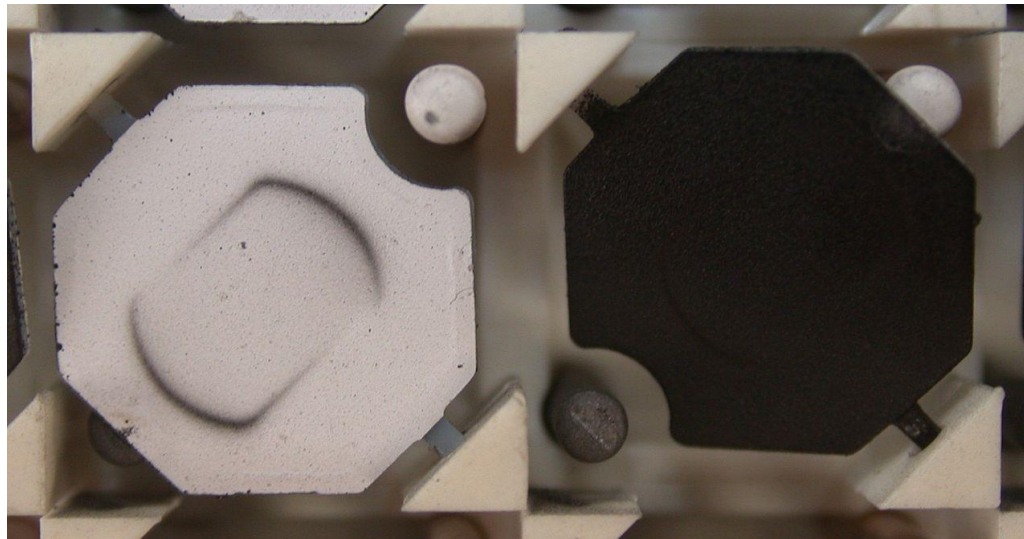
Pre účel animácií nie je tento displej najvhodnejší, preto sme implementovali len základný pohyb vytvoreného obrazu na displeji o zvolený počet krokov a zadaným časovým oneskorením.

## 2. TEORETICKÝ ROZBOR

Hlavnou súčasťou celého zariadenia je maticová zobrazovacia jednotka. Jedná sa o zariadenie, ktoré sa používa najmä v interiéroch mestskej hromadnej dopravy, kde informuje cestujúcich o číslach linky a nasledujúcej alebo konečnej zastávky. Displej má rozmer 140 stĺpcov a 19 riadkov. Celý panel je poskladaný z niekoľkých menších panelov o veľkosti 28 stĺpcov a 19 riadkov. Tieto panely sú paralelne prepojené, čím sa budeme zaoberať v práci v kapitole týkajúcej sa návrhu hardvéru a softvéru.

V anglickej literatúre sa tieto panely označujú ako flip-dot display, v preklade displej flip-disk. Toto označenie priamo súvisí s tým, ako panel zobrazuje dáta. Tieto zobrazovacie jednotky sú elektromechanické maticové displeje, ktoré sa používajú pre veľké vonkajšie ale aj vnútorné označenie, najmä tam kde sú vystavené priamemu slnečnému svetlu.

Displej sa skladá z mriežky malých plechových diskov, v našom prípade krúžky o priemere 10 mm, ktoré sú na jednej strane čierne a na druhej obvykle bielej, prípadne pre lepšiu viditeľnosť je použitá reflexná žltá farba. Veľkou výhodou tohto spôsobu zobrazovania informácií a dát je, že po pretočení panel zostane v aktuálnom stave aj po odpojení elektrickej energie. Na nasledujúcom obrázku Obr. 1 je možné vidieť detail dvoch čierno/bielých diskov, ktoré majú magnet umiestnený priamo vo vnútri disku.



*Obr. 1 Detail dvoch diskov z panelu BUSE*

Každý disk je pripojený k osi, na ktorej je tiež malý permanentný magnet. Možností umiestnenia magnetu je viac, napríklad priamo na disku, prípadne dva magnety na protiľahlých stranách. Pod každým diskom sa nachádza cievka, ktorá podľa smeru prúdového impulzu, ktorý na ňu privedieme vytvorí magnetické pole, a tým pritiahne k sebe stranu disku, kde je opačná polarita magnetického poľa, ako je to vyvolané cievkou. Hlavnou nevýhodou napríklad oproti LED displejom je to, že

informácie zobrazené na displeji nie sú viditeľné v noci. Tento nedostatok býva často riešený pridaním LED podsvietenia pod zobrazovaciu časť. Toto riešenie je podľa môjho názoru celkom neefektívne, keďže kombinuje dve technológie a tým sa strácajú výhody ako jedného, tak aj druhého spôsobu zobrazovania informácií.

Maticové zapojenie znamená, že cievky v jednom riadku majú jeden kontakt spoločný. To isté platí pre cievky v jednom stĺpci kde je tiež jeden spoločný kontakt. Spôsobov na riadenie takto zapojeného displeja je viac. Jednou z možností je prepísanie displeja zhora nadol, kde sa privedie napätie na prvý riadok displeja, a postupne sa zopínajú jednotlivé stĺpce zľava doprava. Druhým spôsobom je postupné spínanie stĺpcov, pričom začneme zľava postupujem až na posledný stĺpec.

Tieto typy zobrazovacích jednotiek sú stále veľmi rozšírené ale skôr v starších zariadeniach a aplikáciách. V dnešnej dobe ich spoľahlivo nahradili už spomínané LED displeje, ktoré síce vyžadujú trvalý prívod elektrickej energie, no na druhej strane je viditeľnosť za všetkých podmienok a nemajú žiadne mechanické časti, teda vyžadujú minimálnu údržbu. Častou chybou u mechanických vecí je ich opotrebovanie a u mechanických maticových displejoch je to chyba celkom zásadná. Výsledok potom vyzerá ako na obrázku Obr. 2.



*Obr. 2 Viditeľné poškodenie displeja*

U nášho displeja, ktorý je primárne určený na použitie v mestskej hromadnej doprave sa životnosť predlžuje len občasným používaním, keďže sa displej prepisuje často len na konečných staniach linky. Informácie o displejoch a technológiách sú prebraté z [1] a [2].

## **2.1 Riadenie panelu BUSE BS210**

Každý panel BUSE dodávaný českou firmou BUSE s. r.o. je vybavený komunikačnými rozhraniami IBIS, TIA/EIA – 485, CAN a IBIS, BUSE, prípadne ďalšími komunikačnými rozhraniami na požiadavku zákazníka. V cene každého panelu sú dodávané aj editačné programy gBUSE0 a gB123, ktoré umožňujú zákazníkovi

pracovať s ľubovoľnými znakovými sadami a vytvárať tiež všetky potrebné piktogramy. Panel, ktorý máme k dispozícii bol bez pôvodnej riadiacej elektroniky. K dispozícii sme mali len stručný popis panelu a rozloženie vstupov 50 pinového konektora ,ktorý je zobrazený v nasledujúcej tabuľke Tab. 1, ktorá bola prebratá z [3].

*Tab. 1 Význam jednotlivých pinov na paneli BUSE BS210*

<b>Pin</b>	<b>Popis</b>	<b>Pin</b>	<b>Popis</b>
<b>1</b>	Ovládanie riadku 1	<b>26</b>	-
<b>2</b>	Ovládanie riadku 2	<b>27</b>	-
<b>3</b>	Ovládanie riadku 3	<b>28</b>	-
<b>4</b>	Ovládanie riadku 4	<b>29</b>	GND
<b>5</b>	Ovládanie riadku 5	<b>30</b>	1. bit pre výber skupiny stĺpcov
<b>6</b>	Ovládanie riadku 6	<b>31</b>	2. bit pre výber skupiny stĺpcov
<b>7</b>	Ovládanie riadku 7	<b>32</b>	0. bit pre výber stĺpca
<b>8</b>	Ovládanie riadku 8	<b>33</b>	1. bit pre výber stĺpca
<b>9</b>	Ovládanie riadku 9	<b>34</b>	2. bit pre výber stĺpca
<b>10</b>	Ovládanie riadku 10	<b>35</b>	0. bit pre výber panelu
<b>11</b>	Ovládanie riadku 11	<b>36</b>	1. bit pre výber panelu
<b>12</b>	Ovládanie riadku 12	<b>37</b>	2. bit pre výber panelu
<b>13</b>	Ovládanie riadku 13	<b>38</b>	enable pre zápis
<b>14</b>	Ovládanie riadku 14	<b>39</b>	0. bit pre výber skupiny stĺpcov
<b>15</b>	Ovládanie riadku 15	<b>40</b>	-
<b>16</b>	Ovládanie riadku 16	<b>41</b>	-
<b>17</b>	Ovládanie riadku 17	<b>42</b>	-
<b>18</b>	Ovládanie riadku 18	<b>43</b>	24 V
<b>19</b>	Ovládanie riadku 19	<b>44</b>	24 V
<b>20</b>	-	<b>45</b>	GND
<b>21</b>	-	<b>46</b>	24 V
<b>22</b>	-	<b>47</b>	12 V
<b>23</b>	-	<b>48</b>	GND
<b>24</b>	-	<b>49</b>	GND
<b>25</b>	-	<b>50</b>	5 V

Z tabuľky vidieť, že pre zopnutie spoločného kontaktu cievok v jednom riadku máme k dispozícii jeden pin. Pin 1 pre prvý riadok až po pin 19 pre posledný riadok. Zapojenie spoločných kontaktov v stĺpcoch panela je o niečo zložitejšie. Ako bolo spomenuté panel sa skladá z piatich menších panelov, o veľkosti 28 x 19 bodov. Tento

panel je ďalej rozdelený na štyri skupiny po sedem stĺpcov teda vo výsledku nám zostalo dvadsať skupín po sedem stĺpcov. Ak by sme chceli riadiť každý stĺpec panelu jedným vývodom z mikrokontroléra bolo by to značne neefektívne ak nie nemožné. Na riadenie je použité iba deväť pinov, ktorými sa riadia logické obvody, pomocou ktorých vyberieme konkrétny stĺpec.

Postup výberu je nasledovný: pinom 35, 36 a 37, ktoré sú pripojené na logický obvod, konkrétne sa jedná o dekóder 1 z n, vyberieme jeden z piatich panelov. To znamená, že nastavíme na adresových vstupoch adresu pomocou troch bitov a tým je aktívny len jeden panel. Pomocou pinov 30 a 31 vyberieme jednu zo štyroch skupín stĺpcov už na konkrétnom vybratom paneli. Ďalšími tromi adresovými bitmi, ktoré sú na pinoch 32, 33 a 34 určíme konkrétny stĺpec, ktorý bude aktívny. Aby sme dokázali určitý bod vo vybratom stĺpci pretočiť, je nutné vybrať aj konkrétny riadok. Spôsobom výberu riadku sa budeme zaoberať ďalej. Po vybratí riadku a stĺpca, teda jednej cievky umiestnenej pod bodom, ktorý chcem otočiť, privedieme prúdový impulz o veľkosti 350 mA na cievku tak, že privedieme log. 1, teda 5 V na pin 38 po dobu 0,5 ms. Tento pin je pripojený na enable pin dekodérov, teda po jeho nastavení sa nastavia všetky adresy, cievka sa zopne a bod sa preklopí. Pre otočenie toho istého bodu na opačnú stranu je potrebné zmeniť polaritu napájania cievky, to prevedieme prepísaním hodnoty na pine 39. Ak je na pine 39 log. 0 na panel zapisujeme, tým je na spoločný kontakt cievok v stĺpci privedená zem, na riadky teda privádzame kladné napätie 24 V. Pri log. 1 na pine 39 pretáčame body na čiernu stranu, teda panel mažeme. Polarita napájania sa tým vymení, čím vznikne opačné magnetické pole tvorené cievkou. Podrobnejší popis ako vybrať nejaký bod a pretočiť ho bude vysvetlený pri návrhu hardvéru a softvéru v nasledujúcich kapitolách.

### **2.1.1. Priamy výber stĺpca pomocou mikrokontroléra**

Asi najjednoduchšou možnosťou ako vybrať jeden z devätnástich riadkov na displeji je použiť mikrokontrolér, ktorý ma dostatok voľných výstupov. Jeden výstup z mikrokontroléra bude pripojený na jeden riadok. Zápisom hodnoty log. 0 na určitý riadok by sme body v riadku mažeme, naopak hodnotou log. 1 na panel do riadku zapisujeme.

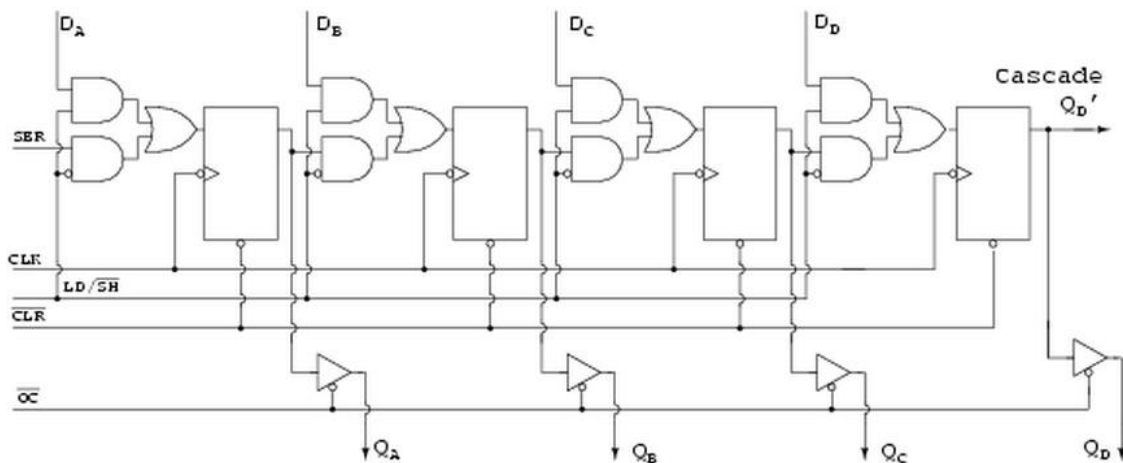
Tento spôsob vyzerá naozaj jednoducho ale prináša aj značné nevýhody. Prvým problémom je použitie veľkého množstva výstupov z mikrokontroléra. K výstupom musíme pripočítať ďalšie potrebné výstupy na riadenie výber stĺpca, a pripojenie ostatných periférií, ktoré budú v práci k mikrokontroléru pripojené.

Druhou veľkou nevýhodou je, že výstup z mikrokontroléra musíme výkonovo prispôbiť na pretekajúci prúd 350 mA. Pri výbere stĺpca je tento problém vyriešený tranzistorovými poliami, ktoré umožňujú maximálny pretekajúci prúd 500 mA. Ak by sme pri písaní riadiaceho softvéru pre výber riadku spravili chybu a vybrali by sme dva, prípadne viac riadkov, tranzistorové polia by sme pri niekoľkonásobne vyššom prúde

určite zničili. Možností ako sa tomuto vyhnúť je viacero, napríklad prúdové obmedzenie napájacieho zdroja, poistky a podobne.

### 2.1.2. Výber stĺpca pomocou posuvného registra

Posuvný register je logický obvod, ktorý sa skladá z rady klopných obvodov spojených tak, že každý klopný obvod prenáša informáciu zo svojho výstupu na vstup nasledujúceho klopného obvodu. Posuv informácie prichádza s nábežnou hranou hodinového signálu. Posuvné registre môžeme rozdeliť podľa typu vstupných a výstupných dát. Tie môžu byť sériové alebo paralelné. Na trhu sú bežne dostupné posuvné registre všetkých typov a kombinácií. Zapojenie posuvného registra pomocou klopných obvodov je zobrazené na obrázku Obr. 3 [4].



Obr. 3 Bloková schéma posuvného registra

Tento posuvný register umožňuje paralelný aj sériový vstup, rovnako tak aj výstupy. Pre náš účel by bol vhodný použiť posuvný register o veľkosti devätnásť bitov, so sériovým vstupom a paralelným výstupom. Devätnásť bitový register však nezoženieme, preto by sme museli vhodne spojiť jeden šesťnásť bitový a jeden trojbitový. Výstup by sme museli rovnako ako pri predchádzajúcom riešení výkonovo prispôbiť. Týmto riešením odstránime však len jednu nevýhodu oproti navrhovanému riešeniu s riadením každého riadku samostatne, a to, že na riadenie použijeme len jeden výstup z mikrokontroléra pre hodinový signál a jeden pre predávanie informácie na sériový vstup.

Problém so zopnutím viacerých riadkov však zostáva, pretože sériová informácia na vstupe môže byť akákoľvek, teda niekoľko hodnôt log. 1 a log. 0 za sebou. Nevýhodou tohto zapojenia je aj to že ak sme napríklad na druhom riadku a chceme znova nastaviť prvý, museli by sme nájsť register, ktorý dokáže posúvať oboma smermi, alebo v horšom prípade prejsť osemnásť krokov a vrátiť sa tak k prvému riadku.

### 2.1.3. Výber pomocou dekóderu 1 z n

Dekóder je kombinačný logický obvod, ktorý zo vstupných dát v určitom kóde vytvára na výstupe iné, často jednoduchšie dáta. Funkcia dekóderu je inverznou funkciou k funkcii kóderu. Typickým príkladom dekóderu je binárny dekóder či demultiplexor, obvod, ktorý binárne kódovaný vstupný signál o n bitoch prevádza na  $2^n$  výstupov, kódovaných ako 1 z  $2^n$ .

Tab. 2 Pravdivostná tabuľka dekóderu 74HC238

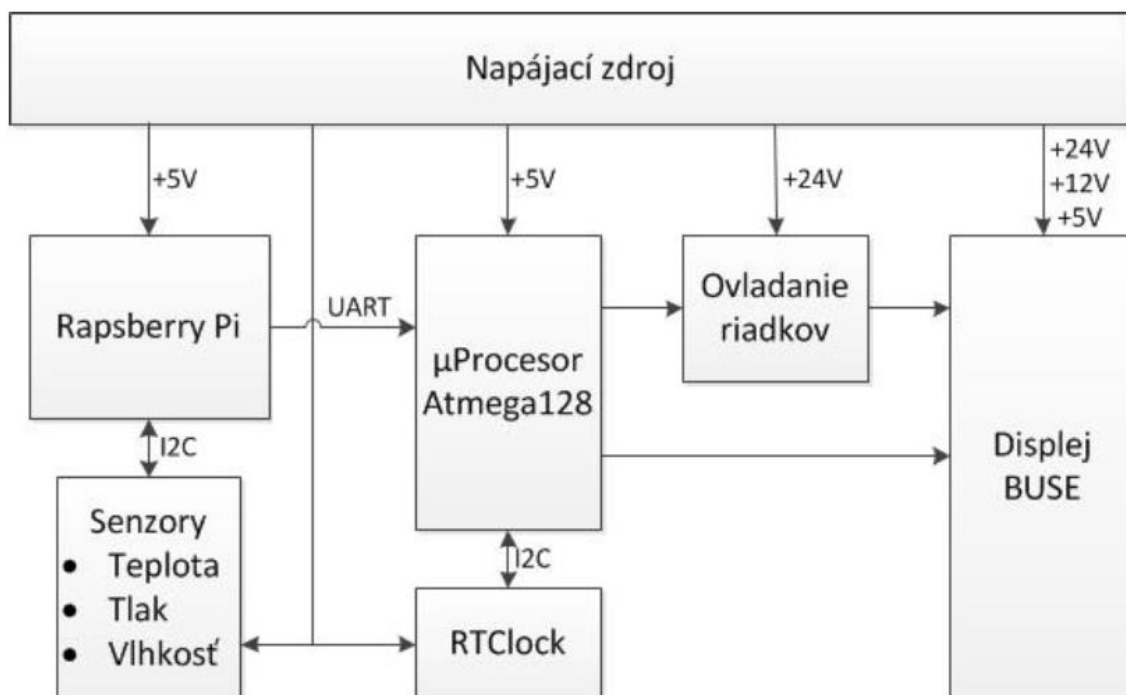
Vstupy						Výstupy							
E1	E2	E3	A0	A1	A2	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
1	X	X	X	X	X	0	0	0	0	0	0	0	0
X	1	X	X	X	X	0	0	0	0	0	0	0	0
X	X	0	X	X	X	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	1	0	0	0	1	0	0	0	0	0
0	0	1	1	1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	1	0	0	0	0	1	0	0	0
0	0	1	1	0	1	0	0	0	0	0	1	0	0
0	0	1	0	1	1	0	0	0	0	0	0	1	0
0	0	1	1	1	1	0	0	0	0	0	0	0	1

V tabuľke Tab. 2 je zobrazená pravdivostná tabuľka dekóderu 74HC238 dostupná napríklad v dokumentácii [5]. Z tabuľky vidíme, že má šesť vstupov, nám však postačia štyri. Vstupy označené ako E1, E2 a E3 slúžia ako enable, teda nastavenú adresu na vstupoch A0, A1 a A2 zapíšeme a aktivujeme tak jeden z výstupov. Piny E1 a E2 uzemníme a tak na enable potrebujeme jeden výstup z mikrokontroléra a ďalšie tri na nastavenie adresy, teda výber konkrétneho výstupu. Tým, že použijeme tento typ súčiastky sa zbavíme hlavného problému a to je možnosť zopnúť viac riadkov v jednom momente, keďže dekóder má aktívny vždy práve jeden výstup. Druhým dôvodom prečo sme sa rozhodli práve pre tento spôsob výberu riadku je skutočnosť, že rovnaký spôsob používa výrobca pri vyberaní stĺpcov. V kapitole 3.4 je uvedené výsledné zapojenie aj so slovným popisom.

## 2.2 Požiadavky na hardvér

Podľa zadania práce sme navrhli celý koncept zariadenia tak ako je uvedené na obrázku Obr. 4. Pozostáva z displeja BUSE BS210, obvodu pre výber riadku, mikrokontroléra AtMega128, ktorý spolu s obvodom pre ovládanie riadkov tvorí radič pre displej. Úlohu meteostanice zastávajú snímače pre meranie atmosférického tlaku,

teploty a vlhkosti vzduchu. Tieto snímače komunikujú s mikropočítačom Raspberry Pi po dvojvodičovej zbernici I<sup>2</sup>C. Mikropočítač zároveň komunikuje po sérovej zbernici UART s radičom displeja, najdôležitejšiu úlohu ale zastáva ako web server. Pre napájanie všetkých menovaných súčastí je potrebné navrhnuť vhodný zdroj napájania. Musí poskytovať napájanie pre displej a to 24 V a 12 V pre spínanie cievok. Ďalšou samostatnou vetvou bude 5 V pre napájanie mikropočítača Raspberry Pi a logických súčiastok ako v obvode s mikrokontrolérom tak aj v displeji BUSE BS210. Hodnoty prúdového zaťaženia sme podľa jednotlivých dokumentácií k navrhnutým súčiastkam a komponentom odhadli na 1 A pre každú vetvu zdroja.



Obr. 4 Navrhnutá koncepcia zariadenia

### 3. HARDVÉR

V tejto kapitole budú postupne prebraté jednotlivé časti podľa blokovej schémy na obrázku Obr. 4. Každá časť bude prebratá z hľadiska výberu vhodných súčiastok a celkovej skladby jednotlivých modulov.

#### 2.4 Napájací zdroj

Pre napájanie všetkých častí potrebujem celkom tri úrovne napájania a to 24 V pre napájanie displeja BUSE, ktoré slúži na otáčanie jednotlivých bodov (spínanie cievok), +12 V pre displej BUSE, o ktorom sa nám nepodarilo zistiť účel, ale v predchádzajúcej práci s panelom je táto úroveň napätia na zbernici uvedená. V poslednom rade potrebujeme +5 V pre napájanie mikro počítača Rapsberry Pi a použitých integrovaných obvodov. Pre návrh zdroja sme museli ako prvé určiť na aké prúdy má byť zdroj dimenzovaný, preto sme zostrojili tabuľku Tab. 3, v ktorej sú prehľadne zobrazené spotreby jednotlivých prvkov v obvode. V čase návrhu sme mali k dispozícii len dokumentáciu k jednotlivým prvkom, a preto nemusia byť hodnoty úplne presné. S touto skutočnosťou sme samozrejme počítali pri dimenzovaní zdroja.

Tab. 3 Spotreba jednotlivých prvkov

U [V]	Zariadenie	I [mA]	súčet [mA]	poznámka
24	BUSE	350,00	350	veľkosť prúdového impulzu potrebného pre otočenie pixelu
12	BUSE	500,00	500	???
5	Raspberry	1000,00	1272	výrobcom odporúčaná hodnota pre napájanie
	AtMega128	200,00		
	RTClock	0,05		hodiny reálneho času
	BMP085	0,65		barometer
	TMP100	0,10		teplomer
	HH10D	0,18		senzor vlhkosti
	LED	60,00		v prípade, že všetky tri LED svietia
	74HCT238	8,00		multiplexory
	ULN2803	2,00		Budič
	TD62783	1,20		Budič

Z tabuľky vidíme, aké napätia budeme na zdroji potrebovať, a taktiež akým prúdom budú zaťažované. Pri navrhovaní sme uvažovali niekoľko variant.

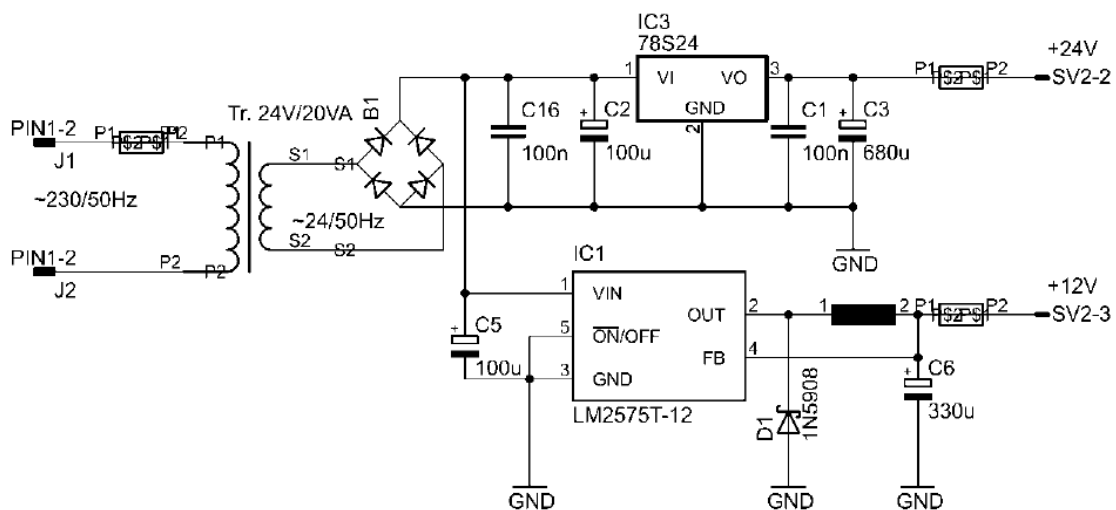
Prvým nápadom bolo použiť sieťové spínané adaptéry. Toto riešenie nevyjde najlacnejšie, pretože cena jedného adaptéra sa pohybuje v rozmedzí 100-200 Kč, v závislosti na maximálnom dodávanom prúde. Ďalšou veľkou nevýhodou je, že by sme

museli vymýšľať rôzne ochrany proti prepólovaniu, prehodeniu jednotlivých adaptérov a podobne. Jednoduchým riešením tohto problému by bolo upraviť konektory adaptérov tak, aby nám ich konštrukcia prepólovanie alebo prípadnú zámenu zamedzila, toto riešenie nám však neprišlo moc elegantné.

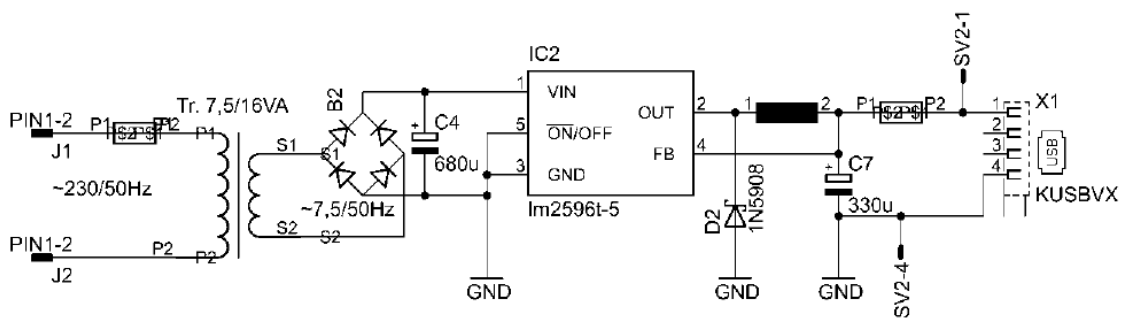
Ďalšou možnosťou ako vyrobiť jednoduchý, dostatočne výkonný a účinný zdroj s tromi rôznymi výstupnými napätiami bolo použitie sieťového transformátora a troch kaskádovo zapojených monolytických stabilizátorov. Už na prvý pohľad vidno, že toto zapojenie by mohlo fungovať, problém je však v tom, že ak by sme z 24 V chceli pomocou monolytického stabilizátora na 12 V upraviť napätie pri prúde asi 1A, na stabilizátore vznikne stratový výkon 12W. Táto hodnota je len približná pretože transformátor s výstupom ~24V má po dvojcestnom usmernení hodnotu asi 32 V, čo je omnoho vyššie. Problém by nebol v tom, že by stabilizátor nevydržal, ale takýto zdroj by bol veľmi neefektívny. Druhým problémom by bolo spomínaných 12 W odvieť vhodným chladičom.

Preto sme sa rozhodli pre riešenie s dvoma sieťovými transformátormi, jedným na 24V, druhým pre 7,5 V. Výstup prvého transformátora je usmernený dvojcestným usmerňovačom a privedený na monolytický stabilizátor s výstupom +24 V. Výkonová strata na tomto stabilizátore je rapídne menšia ako v prvom prípade, aj z dôvodu, že z 24 V výstupu bude odoberaný podstatne menší prúd. Usmernený výstup z transformátora je rovnako privedený na DC/DC menič LM2575T-12 s pevným výstupom 12V. O týchto meničoch sa zmienime bližšie v podkapitole 3.1.1.

Druhá vetva zdroja je vytvorená pomocou druhého sieťového transformátora s výstupom 7,5 V. Toto napätie je rovnako dvojcestne usmernené a privedené na DC/DC menič s výstupom 5 V. Výstup z tejto vetvy je privedený na USB konektor, ktorý bude slúžiť pre napájanie mikro počítača Rapsberry Pi a rovnako aj na štvorpinový konektor, na ktorom sú vyvedené všetky napätia vrátane signálu GND. Pre lepšiu predstavivosť o skutočnom zapojení uvádzame obrázky jednotlivých vetiev zdroja.



Obr. 5 Schéma zapojenia 24 a 12 V vetvy zdroja



Obr. 6 Schéma zapojenia 5 V časti zdroja

Návrh dosky plošného spoja je uvedený v prílohe tejto práce. Obrázok osadeného a oživeného zdroja je uvedený nižšie.



Obr. 7 Osadený napájací zdroj

### 2.4.1 Znižujúce napät'ové regulátory

Tieto regulátory môžeme nájsť aj pod menom DC-DC meniče alebo prevodníky. Jedná sa o elektronické obvody, ktoré prevádzajú jednosmerné napätie o určitej veľkosti na jednosmerné napätie inej veľkosti. Taktiež sa klasifikujú ako napájacie prevodníky.

Prevodníky sú veľmi používané v prenosných elektronických zariadeniach ako sú napríklad mobilné telefóny alebo prenosné počítače, ktoré sú primárne napájané batériami. U týchto zariadení by bolo nevýhodné používať batérie, ktoré by mali rovnaký napät'ový výstup aký vyžaduje aplikácia. Často krát je to nemožné z dôvodu rôznych napät'ových úrovní použitých v jednom zariadení. Na trhu sú dostupné prevodníky s pevným výstupom napríklad 3,3 V, 5 V, 12 V a podobne, ale aj prevodníky, ktoré sú schopné výstupné napätie regulovať v určitom rozmedzí.

Spôsobov ako dosiahnuť zníženia jednosmerného napätia je hneď niekoľko. Pre nás bude zaujímavá iba skupina elektronického prevodu. Do tejto skupiny patria lineárne regulátory, spínané regulátory, magnetické a kapacitné regulátory.

#### **2.4.1.1 Lineárne regulátory**

Lineárne regulátory sa používajú len pre malé zmeny vstupného napätia. Sú veľmi neefektívne pre veľký pokles napätia a veľkom prúde, pretože na nich vzniká veľký stratový výkon. Odvádzanie tepla od výkonových zdrojov je už samo o sebe problém a hlavnou úlohou je sa tohto problému zbaviť.

Ich použitie je vhodné vtedy, keď odoberáme z ich výstupu malý prúd, pri malom poklese napätia. Vtedy je rozptýlený výkon regulátorom malý a nie je potrebné žiadne prídavné chladenie. Využitie nájdú taktiež v aplikáciách, kde je vstupné napätie len o niečo vyššie ako požadujeme. Na regulátore vznikne úbytok napätia a na výstupe dostaneme stabilizovanú hodnotu napätia požadovanej veľkosti. Lineárne regulátory sú obľúbené medzi používateľmi hlavne kvôli priaznivej cene, jednoduchosti zapojenia a pri použití v správnej aplikácii aj spoľahlivosti. [7].

#### **2.4.1.2 Spínané regulátory**

Spínaný DC-DC regulátor prevádza jednu úroveň jednosmerného napätia na inú tým, že dočasne ukladá vstupnú energiu a potom ju uvoľňuje. Ukladanie energie môže byť pomocou cievok, transformátorov alebo kondenzátorov. Tento prevod je 78% až 98% efektívnejší ako pri lineárnych stabilizátoroch kde sa energia rozptyľuje vo forme nežiaduceho tepla. Účinnosť sa zväčšujú použitím výkonových FET tranzistorov v spojení s diódou, čím sa znižujú prepínacie straty a prevodník môže pracovať na vysokej pracovnej frekvencii, rádovo desiatky až stovky kHz.

Nevýhodou je citlivosť na elektrické a magnetické rušenie (EMI) a to, že sa pohybujú vo vyššej cenovej relácii. V dnešnej dobe sú takéto meniče dostupné ako integrované obvody, ktoré potrebujú pre svoju činnosť minimálne množstvo ďalšej elektroniky. Tiež sa dajú zakúpiť ako hotové moduly, ktoré sú určené pre použitie priamo v danej aplikácii. [8].

### **3.2. Obvod s mikrokontrolérom**

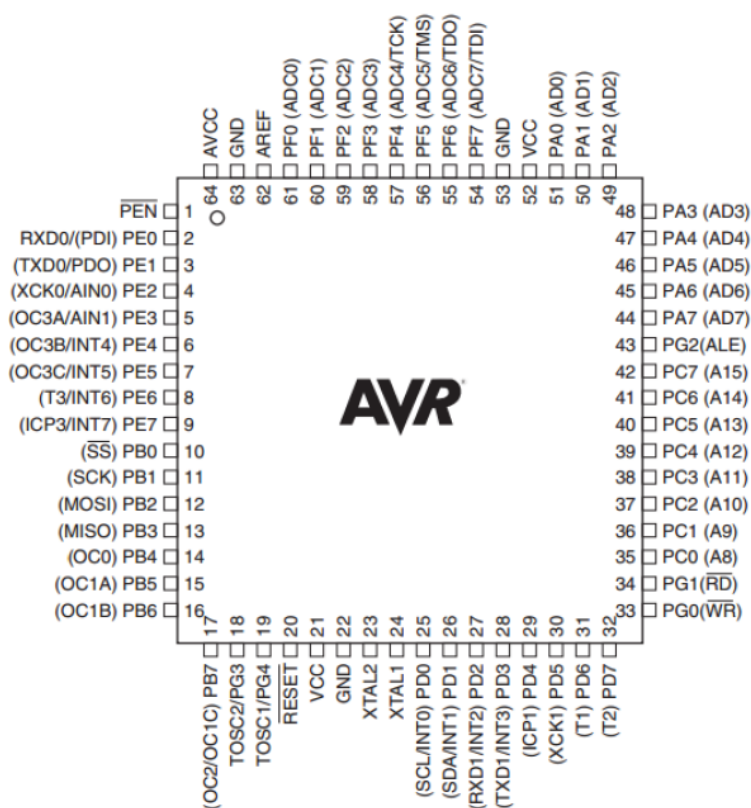
[9] Mikrokontrolér bude spolu s blokom pre riadenie riadkov slúžiť ako radič pre zobrazovaciu jednotku. To znamená že mu odošleme z nadradeného zariadenia (Raspberry Pi) príkaz, ktorý spracuje a následne nastaví svoje výstupné porty tak, aby sa na displeji BUSE zobrazila požadovaná informácia. Konkrétny mikrokontrolér sme vybrali podľa nasledujúcich požiadaviek:

- musí obsahovať rozhranie pre komunikáciu s nadradeným zariadením,
- musí mať dostatok pamäte pre uloženie znakovkej sady displeja (Flash),
- rozhranie pre preprogramovanie (ISP, JTAG),

- rozhranie pre pripojenie modulu RTClock (I2C),
- dostatok voľných potrov pre prípadné rozšírenia.

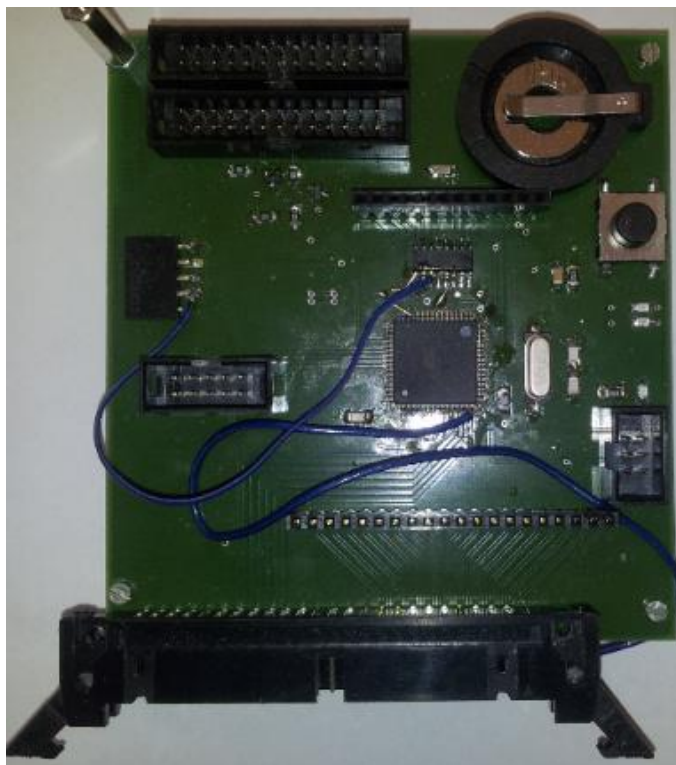
Po predchádzajúc skúsenostiach s programovaním 8-bitových mikrokontrolérov od firmy Atmel bol vybraný mikrokontrolér ATmega128, ktorý obsahuje všetky spomínané požiadavky. Konkrétne disponuje so 128 KB programovateľnou pamäťou Flash, ktorú využijeme pre uloženie znakovkej sady. Rozhrania I2C, UART, ISP a JTAG sú samozrejmosťou už pri nižších radách mikrokontrolérov ATmega.

Mikrokontrolér je dostupný aj vo verzii pre povrchovú montáž (SMD), preto sme zvolili túto konkrétnu variantu. Rozmiestnenie pinov na puzdre mikrokontroléra je zobrazené na obrázku Obr. 8.



Obr. 8 Rozloženie pinov mikrokontroléra AtMega128[9]

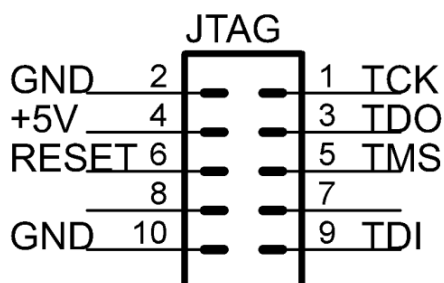
Pre zjednodušenie fáze vývoja a testovania sme na voľný port pripojili dve indikačné LED diódy a jedno tlačítko. Ďalšie pripojené periférie budú popísané v nasledujúcich podkapitolách. Z dôvodu veľkých rozmerov celej schémy zapojenia tohto obvodu sme sa rozhodli uviesť ju do príloh, ktoré sú súčasťou práce. Hotová a oživená doska plošných spojov je uvedená na obrázku Obr. 9.



Obr. 9 Osadená doska s mikrokontrolérom

### 3.2.1. JTAG

Pre pripojenie AVR zariadenia k programátoru pomocou rozhrania JTAG je potrebných minimálne 6 vodičov a to: TCK, TDO, TDI, TMS, UTG, GND. Ďalším rozširujúcim signálom je nSRST. Tento signál je použitý pre riadenie a monitorovanie resetovacieho vodiča na cieľovom zariadení. Pre programovanie AVR zariadení máme k dispozícii programátor ATVRDragon, rovnako od firmy Atmel. [10] Keďže sa predpokladá, že bude pri vývoji a testovaní skutočne použitý zapojili sme programovací konektor podľa programátora a tým sme sa vyhli nepríjemnému a neprehľadnému prepájaniu pomocou jednotlivých káblov. V našom prípade stačí pre pripojenie plochý 10-žilový vodič. Zapojenie konektor je zobrazené na obrázku Obr. 10.



Obr. 10 Rozloženie pinov na JTAG konektore

### 3.2.2. I<sup>2</sup>C zbernica

Zbernica I<sup>2</sup>C je dvojvodičová zbernica typu master-slave, vyvinutá firmou Philips Semiconductors. Primárne bola určená na komunikáciu v televíznych prijímačoch s perifériami umiestnenými na plošnom spoji. Dnes ju možno považovať za štandard, ktorý sa používa v mnohých zariadeniach. Podporujú ju poprední výrobcovia integrovaných obvodov a mikroprocesorov, používa sa na komunikáciu s displejmi, senzormi, prevodníkmi a mnohými ďalšími súčiastkami.

Samotná zbernica pozostáva z dvoch vodičov. Dátový vodič označovaný ako SDA (Serial\_Data) slúži na prenos dát, ktorých platnosť je daná hodinovým signálom, prenášaným vodičom SCL (Serial\_Clock). Všetky zariadenia pripojené ku zbernici musia mať vstupno-výstupný port s otvoreným kolektorom. Vysoká úroveň H je potom určená pull-up rezistorom a nízka úroveň L pripojením vodiča na zem jedným zo zariadení. Žiadne s pripojených zariadení nesmie nastaviť úroveň H na zbernici nastavením vysokej úrovne na výstupnom porte. Napäťové úrovne sú definované jako  $0,3 \cdot V_{cc}$  pre úroveň L a  $0,7 \cdot V_{cc}$  pre úroveň H. [11]

### 3.2.3. UART

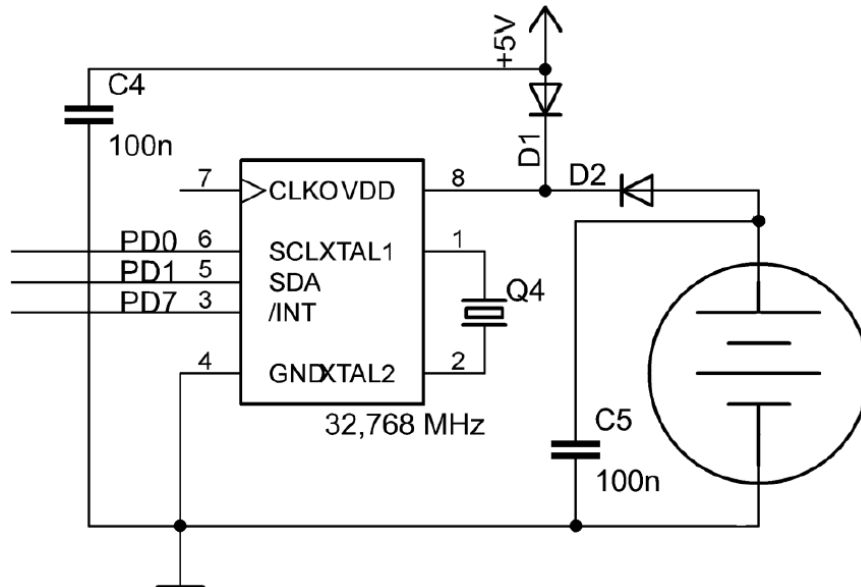
Skratka UART vznikla z anglického názvu Universal Asynchronous Receiver/Transmitter. V preklade sa jedná o univerzálny asynchrónny prijímač a vysielateľ. Je to vlastne časť hardwaru, ktorý prevádza dáta medzi paralelnou a sériovou podobou. UART sa bežne používa v spojení s komunikačnými štandardami ako je RS-232, RS-422 alebo RS-485. Univerzálne označenie znamená, že parametre prenosu sú konfigurovateľné. UART býva bežne zahrnutý v mikrokontroléroch.

Komunikácia prebieha tak, že jedno zariadenie (vysielateľ) vezme byte dát a po jednom bite ich prenáša po linke. Druhé zariadenie spätne zostavuje prijaté bity do kompletných bytov. Každý UART obsahuje posuvný register, ktorý je základným prvkom medzi paralelnou a sériovou formou dát. Sériový prenos informácií (bitov) pomocou jedného vodiča je menej nákladné ako paralelný prenos, na druhú stranu je pomalší. [12].

## 3.3. RTC hodiny

Aby sme sa vyhli situácii, že sa odpojí napájanie zariadenia a tým sa nám stratí aktuálny čas, doplnili sme do obvodu hodiny reálneho času. Jedná sa o obvod, ktorý má vstavaný vlastný alebo pripojený externý oscilátor a pamäť. Ako oscilátor kmitá, inkrementuje sa premenná a zapisuje sa do pamäte. Hodnota tejto premennej zodpovedá počtom kmitov oscilátora. Väčšina dostupných obvodov na trhu obsahuje v pamäti registre, z ktorých je možno priamo vyčítať hodnotu rokov, mesiacov, dní, hodín, minút a sekúnd. Týmto sme sa zbavili povinnosti prepočítavať hodnotu.

Pre našu aplikáciu postačí jeden z jednoduchších obvodov, z ktorého dokážeme zistiť aktuálny čas. Vybrali sme cenovo prijateľný obvod PCF8563 [13]. Tento obvod je optimalizovaný pre nízku spotrebu. Dostupný je výstup hodín obvodu, výstup generujúci prerušenie a detektor poklesu napätia. Komunikácia je vyriešená pomocou dvojvodičovej zbernice I2C pre ktorú je stanovená maximálna rýchlosť 400 kbit/s. Zapojenie je na obrázku Obr. 11.



Obr. 11 Zapojenie hodín reálneho času

V zapojení sme mysleli aj nad výpadkom napájania. Preto je v zapojení pridaná záložná batéria, ktorá zabezpečí plynulú činnosť aj počas výpadku primárneho napájania. Ako bolo spomínané komunikácia prebieha cez dvojvodičovú zbernicu I<sup>2</sup>C, piny 5 a 6 sú pripojené priamo na piny mikrokontroléra SDA a SCL.

### 3.4. Výber riadku

Pre našu aplikáciu sme vybrali dekódér 74HC238. Tento dekódér má tri binárne váhované adresové vstupy (A0, A1, A2) a keď je povolený enable, jeden z ôsmich výstupov je nastavený na vysokú úroveň, za predpokladu, že je nastavená adresa. Ak nie je na adresových vstupoch žiadna informácia, na výstupoch je stav vysokej impedancie. Prvok má tri enable vstupy. E1 a E2, ktoré sú aktívne pri log. 0 a E3, ktorý je aktívny pri log. 1. Aby bol enable aktívny, musia byť všetky vstupy pripojené na príslušnú logickú úroveň.

#### 3.4.1. Zapojenie logickej časti pre výber

Na displeji je 19 riadkov, ktoré musíme byť schopný spínať po jednom a to pre oba smery prúdu. Ak prechádza cievkou prúd v jednom smere, otočí sa pixel na čiernu stranu, ak v opačnom smere, otočí sa na žltú. Preto by sme potrebovali celkovo 38

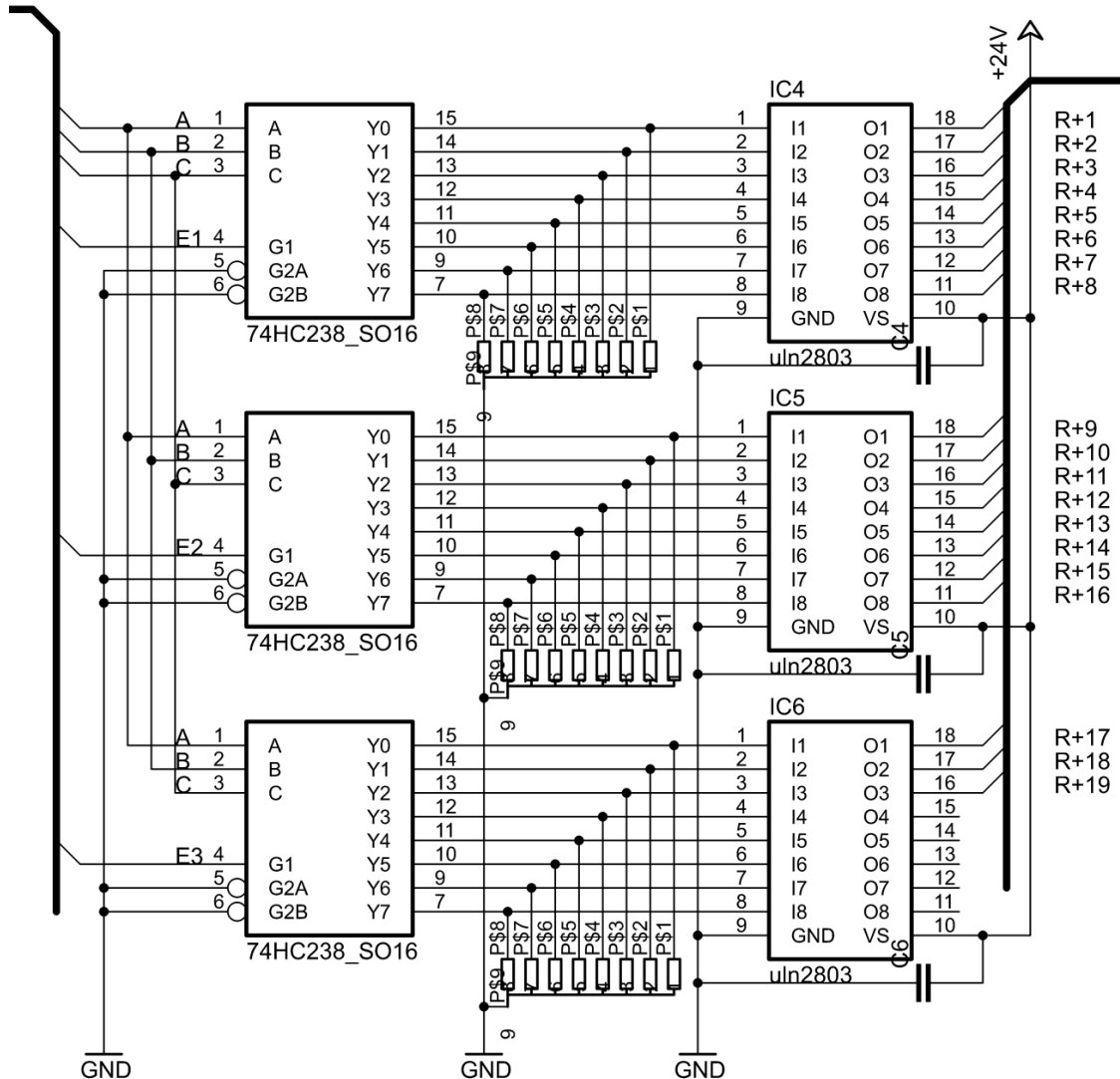
výstupov z mikrokontroléra. Toto riešenie je celkom neefektívne a pri programovaní by bolo celkom máťúce. Pre pretočenie jedného pixelu potrebujeme prúdový impulz o veľkosti 350 mA, čo je pre výstup z mikrokontroléra neprijateľná hodnota, preto musíme výstup výkonovo prispôbiť. Použijeme dva typy tranzistorových polí pre každý riadok. Aby sme zredukovali počet výstupov z mikrokontroléra, inšpirovali sme sa pôvodnou elektronikou panelu pre výber stĺpca. Použijeme jeden osem bitový demultiplexor u ktorého si pomocou najvyššieho adresového bitu budeme vyberať či chceme zapisovať alebo mazať riadok. To znamená, že vyberieme horné štyri výstupy ale spodné štyri. Na troch horných výstupoch budú pripojené ďalšie tri osem bitové demultiplexory a na ich výstupoch tranzistorové polia pre kladný smer prúdu. Na spodných troch výstupoch budú rovnaké demultiplexory s rozdielom, že na ich výstupoch budú tranzistorové polia umožňujúce opačný smer prúdu.

Proces výberu riadku bude spočívať teda v tom, že nastavíme najvyšší adresový bit prvého demultiplexora, ktorým vyberieme zápis alebo mazanie pixelu. Nastavením druhých dvoch adresových bitov vyberieme jeden z troch demultiplexorov vo výstupnej vrstve. Nastavením adresy vo výstupnej vrstve vyberieme jeden konkrétny riadok, ktorý bude aktívny, pre jeden alebo pre druhý smer prúdu. Adresové vstupy pre výstupné demultiplexory je rovnaká, avšak iba jeden je vždy aktívny.

Pri návrhu sme uvažovali nad tým, ako budú reagovať tranzistorové budiče, ktoré sú pripojené na práve neaktívne výstupy demultiplexorov. Ak demultiplexor nie je aktívny na jeho výstupy sú v stave vysokej impedancie. Z tohto dôvodu sa na vstupoch budičov môže indukovať neželaný šum, ktorý by teoreticky mohol spôsobiť nechcenú aktiváciu riadku. Preto sme do návrhu zapojili pull-down rezistory, ktoré by mali pri spomínanom stave pripojiť vstupy tranzistorových budičov na signál GND. V dokumentácii pri budičoch sa nám nepodarili nájsť, ako sa obvod zachová pri nepripojenom vstupe, preto zapojenie po zostavení odskúšame s pull-down rezistormi a bez nich. Výsledky porovnáme a podľa toho osadíme alebo neosadíme tieto rezistory.

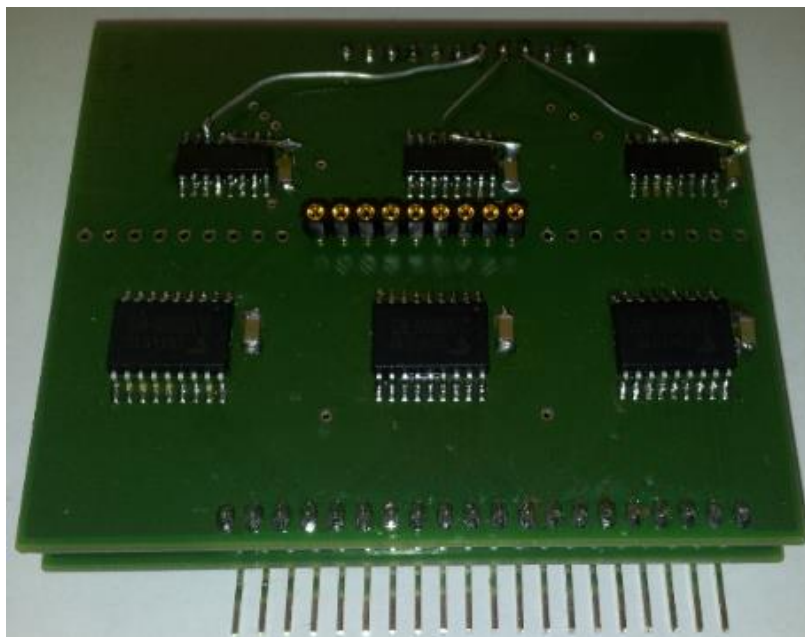
Pre lepšie pochopenie skutočného zapojenia sa môžeme pozrieť na obrázok Obr. 12, na ktorom je zapojenie logickej časti pre kladný smer prúdu. Adresové vstupy demultiplexorov sú pripojené priamo na mikrokontrolér. Každý z enable vstupov demultiplexorov je pripojený na jeden výstup z hlavného demultiplexora, ktorým sa vyberá, ktorý z troch bude aktívny. Na druhej polovici logickej časti sú adresové vstupy pripojené na rovnaké výstupy z mikrokontroléra. Celkovo teda použijeme dva výstupy na adresovanie osmice riadkov, tri výstupy na adresovanie konkrétneho riadku, jeden výstup na výber zápisu alebo mazania a jeden výstup na enable vstup prvého demultiplexora. Po aktivovaní enable vstupu prvého demultiplexora sa prevedie zapísanie alebo vymazanie konkrétneho pixelu. To však závisí aj na výbere konkrétneho stĺpca. Výber stĺpca sa prevádza pomocou elektroniky, ktorá už je súčasťou panela a funguje na rovnakom princípe. Troma výstupmi si nastavíme výber konkrétneho panela na ktorom sú štyri skupiny po sedem stĺpcov. Dvoma výstupmi si vyberieme jednu

skupinu a ďalšími troma výstupmi nastavíme výber konkrétneho stĺpca. Ešte potrebujem vybrať smer prúdu, teda či sa pixel otočí na čiernu alebo žltú stranu. To prevedieme nastavením jedného bitu, ktorý je na 50 pinovom konektore na pine 39. Tento pin je spojený s výstupom z mikrokontroléra, ktorý je pripojený na výber zapisovania/mazania na ovládání riadkov. Rovnako aj enable prvého demultiplexora pre riadenie riadkov a stĺpcov je spoločný.



Obr. 12 Schéma zapojenia logickej časti

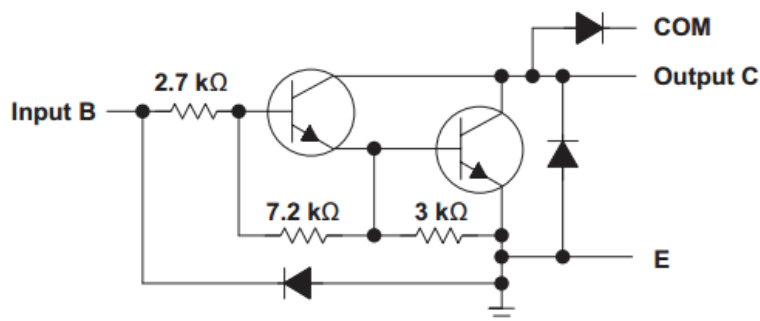
Na obrázku Obr. 13 sú zobrazené hotové dosky plošných spojov. Po odskúšaní s pull-down rezistormi a s nimi sa žiadne napätie na neaktívnych výstupoch neobjavilo, preto sme ich s výsledného zapojenia vylúčili.



Obr. 13 DPS logickej a výkonovej časti

### 3.4.2. Budiče

V aplikáciách, kde požadujeme veľké prúdové zosilnenie sa používa tzv. Darlingtonove zapojenie tranzistorov. Toto zapojenie si môžeme predstaviť ako jeden tranzistor, ktorého výhodou je veľký vstupný odpor spôsobený malým prúdom prvého tranzistora a veľký prúdový zosilňovací činiteľ, daný približne súčinom dielčích prúdových zosilňovacích činiteľov jednotlivých tranzistorov. Nevýhodou je približne dva-krát väčšie napätie medzi bázou a emitorom a predovšetkým podstatne väčšia veľkosť saturačného napätia. Za nevýhodu sa dajú považovať aj väčšie zbytkové prúdy. Bežne vyrábané typy Darlingtonových tranzistorov majú integrované rezistory v obvode báza-emitor a ochrannú diódu v obvode kolektor-emitor. Prúdový zosilňovací činiteľ býva v prípade výkonových typov niekoľko stoviek, u nízko-výkonových niekoľko desiatok tisíc.

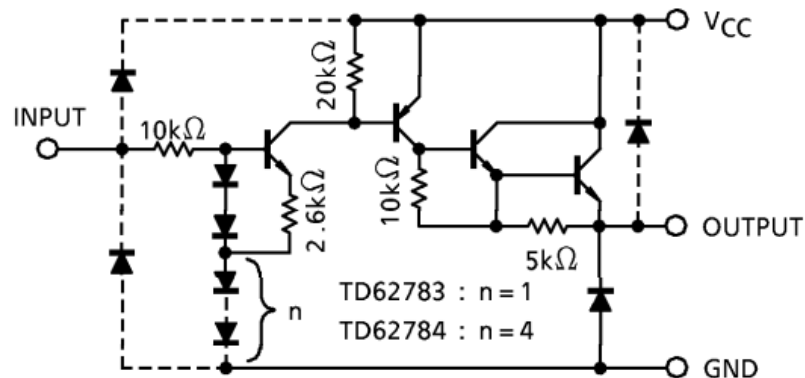


Obr. 14 Zapojenie jedného kanála obvodu ULN2803A [14]

Pre našu aplikáciu by sme takýchto Darlingtonových tranzistorov potrebovali celkovo 19, preto sme sa rozhodli použiť tzv. tranzistorové pole, kde sa v jednom púzdre integrovaného obvodu nachádza týchto tranzistorov 8. Konkrétne sme vybrali obvod od firmy Texas Instruments, ULN2803A. Tento obvod sa bežne používa ako driver pre relé, LED diódy alebo ako driver pre zobrazovacie jednotky [14].

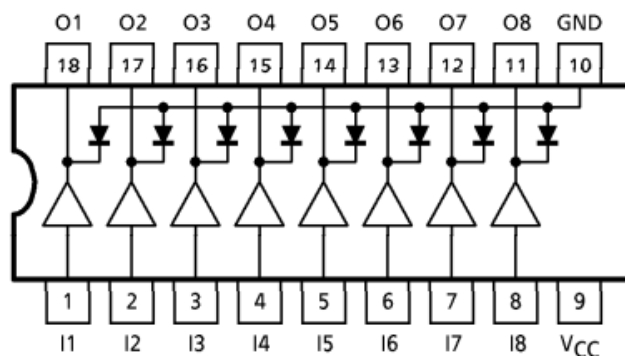
Tento obvod podporuje vstupné napätie až 30 V, my požadujeme pre displej BUSE 24 V, takže máme dostatočnú rezervu. Výstup každého tranzistora môže byť zaťažený prúdom až 500 mA, čo je pre naše použitie taktiež dostačujúce. Najviac, na každom výstupe sú zapojené diódy so spoločnou uzemnenou katódou, čo je vhodné pre spínanie indukívnej záťaže.

Pre spínanie prúdu v opačnom smere sme vybrali obdobný obvod TD62783 vyrábaný firmou TOSHIBA [15]. Znovu sa jedná o tranzistorové pole kde sa v jednom čipe nachádza celkovo osem tranzistorov. Zapojenie jedného kanála je uvedené na obrázku Obr. 15.



Obr. 15 Vnútorne zapojenie jedného kanála obvodu TD62783 [15]

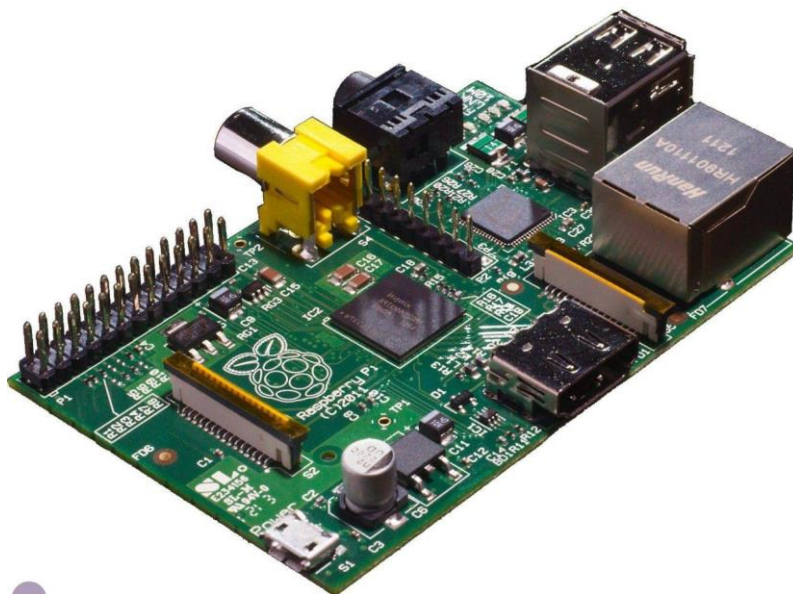
Obvod podporuje napájanie až 50 V, typická hodnota je 35 V. Výstupný prúd pre jeden výstup je -500 mA. Použitie tohto obvodu je rovnaké ako pri predchádzajúcom. Vnútorne zapojenie obvodu a rozloženie vývodov z púzdra je uvedené na obrázku Obr. 16.



Obr. 16 Rozloženie pinov obvodu TD62783 [15]

### 3.5. Raspberry Pi

[16] Jadrom systému Raspberry Pi je procesor typu SoC Broadcom BCM2835, ktorý obsahuje hlavný aj grafický procesor, zvukový a komunikačný hardware a je integrovaný do jednej súčiastky pod pamäťovým čipom. Pamäťový čip má kapacitu 256 MB a je umiestnený v strede základovej dosky, ktorá je zobrazená na obrázku Obr. 17.



Obr. 17 Mikropočítač Raspberry Pi

Procesor používa inú architektúru inštrukčnej sady, ktorá sa označuje ako ARM. Táto architektúra sa používa hlavne pri mobilných zariadeniach ako mobilné telefóny a tablety. Vyniká jednoduchou architektúrou s redukovanou inštrukčnou sadou, a nízkou spotrebou. Preto sa oproti procesorom stolných počítačov s vysokými nárokmi na napájanie používa hlavne u mobilných zariadení. Napájanie tohto počítača vyžaduje 5 V s maximálnym odberom 1 A. Pre napájanie sa používa integrovaný mikro USB port. Nie je dokonca potrebné ani žiadne prídavné chladenie ani pri zložitých výpočtových operáciách. My ale z dôvodu umiestnenia počítača do uzavretého krytu displeja BUSE použijeme chladič aspoň na procesor.

Operačný systém vhodný pre Raspberry Pi je GNU/Linux. Oproti iným bežne používaným operačným systémom je tento OpenSource, to znamená, že je možné stiahnuť celý zdrojový kód systému a čokoľvek v ňom podľa požiadaviek užívateľa zmeniť.

Počítač je vďaka nízkej spotrebe elektrickej energie a tichému chodu vhodný na použitie aj k obsluhu málo vyťažovaných webových stránok v miestnej sieti alebo v Internete. Veľká časť webových serverov je založená na kombinácii operačného systému Linux, webového serveru Apache a databáze MySQL a jazyka PHP. Tento spôsob využitia bude bližšie popísaný v nasledujúcich kapitolách, kde sa budeme týmto problémom priamo zaoberať.

## **3.6. Meteostanica**

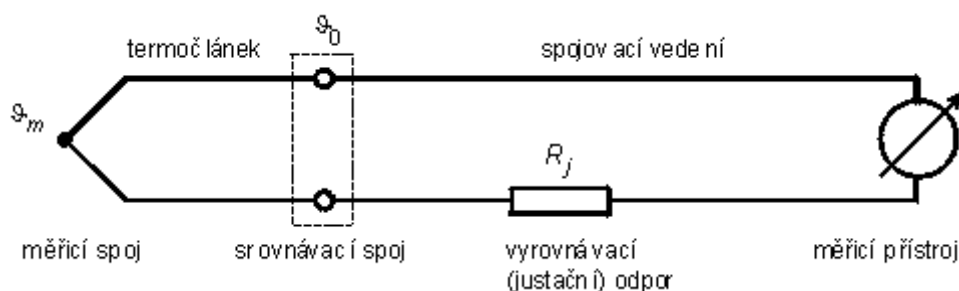
Jedným z bodov zadania tejto diplomovej práce je aby zariadenie splňovalo úlohu meteostanice. Rozhodli sme sa preto implementovať tri snímače na meranie fyzikálnych veličín a to atmosférického tlaku, teploty a vlhkosti vzduchu. Snímač tlaku a vlhkosti je schopný merať aj teplotu. Pripojené sú k mikropočítaču Raspberry Pi cez dvojvodičovú zbernicu I<sup>2</sup>C a v pravidelných intervaloch sú hodnoty zosnímané a uložené do databázy v mikropočítači. Ich podrobnejší popis a princípy sú popísané v nasledujúcich podkapitolách.

### **3.6.1. Meranie teploty**

Teplota je najčastejšie meraná veličina. Dá sa to očakávať pretože väčšina fyzických, elektrických, mechanických a biologických systémov sú teplotou ovplyvňované. Niektoré chemické reakcie, biologické procesy a elektronické obvody dosahujú optimálnych výsledkov len v určitom teplotnom rozsahu. Existuje mnoho spôsobov ako určiť aktuálnu teplotu priestoru alebo určitého objektu. Teplotu možno merať prostredníctvom priameho kontaktu so zdrojom tepla alebo na diaľku bez priameho kontaktu. Na trhu existuje široká škála teplotných čidiel založených na rôznych princípoch ako sú napríklad termočlánky, termistory, infračervené a polovodičové snímače [17].

#### **3.6.1.1. Termočlánok**

Termočlánok alebo termočlánkov sonda premieňa rozdiel teploty na napätia vďaka termoelektrickému javu. Napätie vzniká na spoji dvoch rôznych kovov. Veľkosť napätia je úmerná rozdielu teplôt a mení sa aj v závislosti od typu termočlánku. Teplomery, ktoré merajú teplotu týmto spôsobom merajú toto napätie a prevádzajú ho na teplotu. Výhodou termočlánku je možnosť vytvoriť veľmi malú sondu s rýchlou odozvou prípadne tvarovať snímaciu časť podľa povrchu vyžarujúceho tepla. Presnosť merania je menšia ako napríklad pri odporových snímačoch teploty. Problémom u termočlánkov je problematické predlžovanie vedenia. Z princípu vyplýva, že je nevyhnutné použiť rovnaký materiál na termočlánok aj na vedenie a svorky. Spoje nesmú byť spájkované ale skrutkované prípadne lisované alebo zvárané aby nevznikal ďalší termočlánok. Vedenia nad 10 metrov je zväčša potrebné aj tieniť aby sme odstránili rušenie.



Obr. 18 základné zapojenie termočlánku [17]

### 3.6.1.2. Termistory

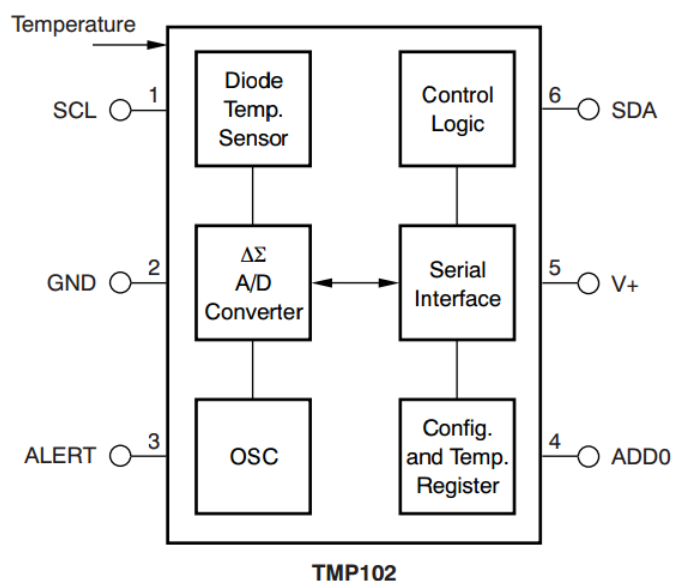
Termistor je elektronická súčiastka, ktorej elektrický odpor závisí na teplote. Existujú dva typy termistorov. NTC, je termistor s negatívnym teplotným koeficientom, čo znamená, že s jeho zahrievaním jeho odpor klesá. PTC termistor, tiež označovaný ako pozistor svoj odpor zvyšuje s rastúcou teplotou.

Pri termistoroch môžeme pracovať len s malými prúdmi, približne do 50 uA, preto sa používajú len vo veľmi citlivých meracích prístrojoch. Majú veľký vnútorný odpor, preto je odpor ich prírodných vodičov zanedbateľný. Ich veľkosť umožňuje takmer bodové meranie teploty a spolu s vysokou citlivosťou patria medzi najpresnejšie snímače teploty. Nevýhodou termistorov je ich časová nestabilita a značná nelineárna závislosť odporu na teplote. Rozmedzie ich použitia sa pohybuje v intervale od -200 do +300 °C.

### 3.6.1.3. Polovodičové senzory s PN prechodom

Polovodičové monokryštalické senzory sú založené na teplotnej závislosti zmien vodivosti PN prechodu. Polovodičové materiály ako kremík a germánium majú pri nízkej teplote veľmi malú vodivosť. Je dané tým, že všetky voľné elektróny sú viazané ako valenčné elektróny v kryštalickej mriežke. So zvyšovaním teploty sa zvyšuje ich energia a prekonávajú bariéru valenčnej oblasti a prechádzajú do vodivostného pásma. Ako senzory teploty sa používajú diódy, tranzistory a integrované obvody.

Pre našu aplikáciu sme vybrali senzor teploty využívajúce práve spomínané diódy. Jedná sa o senzor TMP102, ktorý komunikuje po dvojvodičovej I2C zbernici. Vyznačuje sa dobrou presnosťou v širokom rozsahu teplôt, výrobca udáva presnosť 0,5 °C v rozsahu od -25 °C po +85 °C. Rozlíšenie je 12 bitov a maximálna spotreba 10 uA. Jeho vnútorné blokové zapojenie možno vidieť na obrázku Obr. 19.



Obr. 19 Blokové zapojenie snímača TMP102[18]

Z obrázka vidieť, že obsahuje diódový senzor teploty, ktorého analógový výstup je prevádzaný pomocou sigma-delta prevodníka na digitálnu formu, sériové rozhranie a registre pre konfiguráciu a vyčítanie zmeranej teploty.

### 3.6.2. Meranie atmosférického tlaku

[11] Tlak patrí medzi stavové veličiny, to znamená, že je to veličina ktorá ukazuje zmenu stavu plynu. Medzi tlakom a objemom platí, že súčin tlaku a objemu a pri stálej teplote je konštantný. Táto zákonitosť je známa od roku 1662, objavili ju Robert-Boyle a Edma Mariot a zákon nesie ich mená. Grafickým vyjadrením Boyle-Mariottovho zákona je rovnoosá hyperbola. Ak narastá tlak, znižuje sa objem a naopak. Veľká pohyblivosť molekúl plynu a vzdialenosti medzi nimi sú príčinou tekutosti, rozpínavosti a ľahkej stlačiteľnosti plynov. Hlavná jednotka tlaku podľa normy SI je  $\text{N/m}^2$ . Táto jednotka sa nazýva Pascal, označuje sa Pa a je definovaná ako sila o veľkosti 1 N pôsobiaca na plochu  $1 \text{ m}^2$ . Atmosférický alebo tiež barometrický tlak je vytvorený tiažou vzduchu, ktorý nás obklopuje.

Pre meranie tlaku sa používa veľké množstvo snímačov. Jednotlivé snímače môžu byť rozdelené napríklad podľa výstupného signálu na mechanické, hydrostatické a elektrické. Podľa ich funkčného princípu by sme mohli tlakomery rozdeliť na piestové, deformačné a kvapalinové. Pre meranie atmosférického tlaku sa používajú barometre, ktoré možno rozdeliť na ortuťové a číslicové barometre a barometre s citlivým deformačným členom.

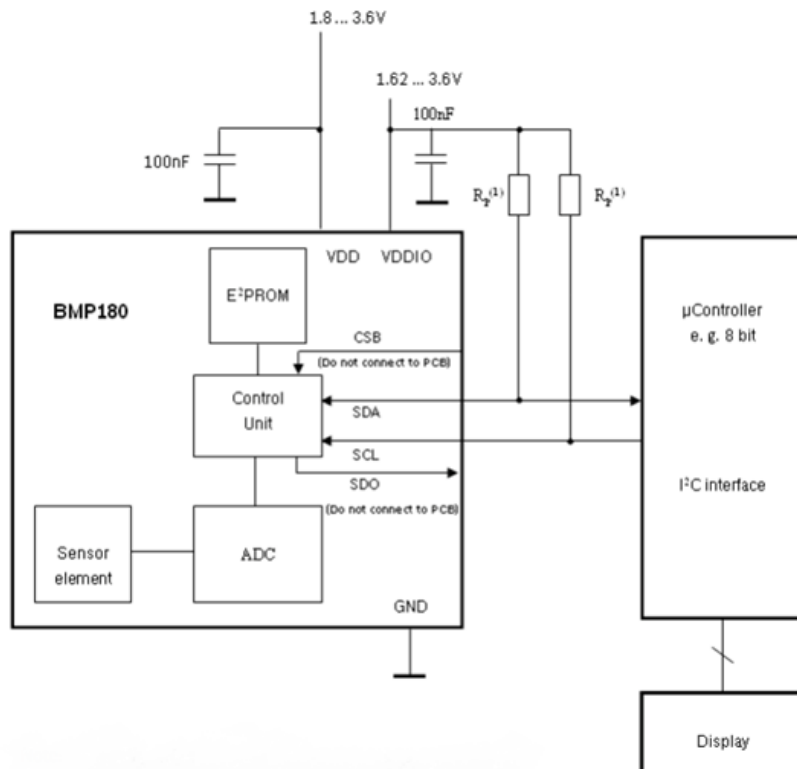
Zaujímavou skupinou číslicových barometrov sú piezo-elektrické barometre. Piezoelektrický snímač je založený na využití piezoelektrického javu. Piezoelektrický jav spočíva v tom, že vo vnútri dielektrika vzniká vplyvom deformácie polarizácia,

ktorá vedie ku vzniku elektrického nápoja. Na elektródach doštičky, ktorá má piezoelektrické vlastnosti dostaneme napätie, ktoré je priamoúmerné veľkosti deformácie.

Na obdobnom princípe fungujú aj piezorezistívne prevodníky tlaku, kde je meracím členom kremíková membrána s integrovanými piezorezistívnymi snímačmi. Jednotlivé rezistory sú vo väčšine prípadov zapojené do Wheatsonovho mostíka, ktorý je pôsobením atmosférického tlaku elektricky rozvažovaný. Výhodou tohto typu snímačov sú ich miniatúrne rozmery a schopnosť pracovať pri vysokých teplotách.

Pre naše účely sme si vybrali práve snímač pracujúci na piezo-elektrickom princípe. Jedná sa o tlakový senzor BMP180 vyrábaný firmou BOSH spadajúci do kategórie MEMS snímačov [19]. Toto označenie znamená, že sa jedná o systém na čipe alebo inteligentnom snímači. Je tu prítomný mechanický subsystém, ktorý sa používa ku transformácii fyzikálnej veličiny na elektrickú a elektronický subsystém, ktorý zaisťuje jej následné spracovanie. Bloková schéma je zobrazená na obrázku Obr. 20. Medzi základné vlastnosti tohto senzoru patria:

- Rozsah meraného tlaku: 300 ... 1100 hPa.
- Napájacie napätie: 1,8 V ... 3,6 V.
- Spotreba 5 uA na 1 vzorku/s.
- Schopnosť merať teplotu.



Obr. 20 Zapojenie barometra BMP180[19]

Elektrický náboj vzniknutý vplyvom mechanickej deformácie sa spracováva A/D prevodníkom a kontrolnou jednotkou s E2PROM pamäťou, v ktorej sú uložené kalibračné dáta na kompenzáciu zmeraných hodnôt tlaku a teploty. Snímač má vysokú citlivosť, rýchlu konverzáciu zmeraných dát a vysokú teplotnú stabilitu.

Snímač je navrhnutý pre mobilné zariadenia a komunikáciu po I<sup>2</sup>C zbernici. V E2PROM pamäti je uložených 176 bitov kalibračných dát, ktoré sú pre každý snímač iné. Získanie dát zo snímača a ich spracovanie bude rozobraté v kapitole 4, kde sa zaoberáme softvérom.

### 3.6.3. Meranie vlhkosti

Vlhkosť je definovaná pomocou koncentrácie molekúl vody v priestore vyplnenom vlhkým plynom. Na určenie vlhkosti sa používa niekoľko parametrov. Jedným z nich, najbežnejšie používaným je relatívna vlhkosť označovaná RH. Definuje sa ako pomer medzi skutočným tlakom pary a tlakom pary pri nasýtení pri istej teplote. Hodnota RH sa pohybuje v rozsahu od 0% po 100%. Existuje mnoho spôsobov na meranie vlhkosti. Všetky metódy využívajú určité fyzikálne alebo chemické vlastnosti molekuly vody, čím sa umožňuje rozlíšenie medzi vodou a ostatnými zložkami vlhkého plynu.

**Metódy využívajúce odstraňovanie vodnej pary:** zo vzorky vlhkého plynu sa odstráni molekuly vody a určí sa hmotnosť obidvoch oddelených zložiek.

Saturačné metódy: vzorka plynu sa privedie do stavu nasýtenia vodnou parou. Zistí sa bod nasýtenia, pričom odpovedajúci parameter procesu nasycovania odpovedá pôvodnej vlhkosti. Nasýtenie sa dá dosiahnuť pridaním vody, znížením teploty, stlačením alebo adiabatickým rozpínaním.

**Parametrické metódy:** fyzikálne vlastnosti vlhkého plynu závisia od obsahu vody v ňom. Medzi parametre, ktoré sa dajú využiť ako miera vlhkosti, patrí koeficient pohltienia elektromagnetickej energie, index lomu, dielektrická konštanta, tepelná vodivosť alebo rýchlosť zvuku.

**Absorpčné metódy:** hygroskopický materiál vystavený vlhkému plynu absorbuje molekuly vody, kým sa nedosiahne rovnováha medzi vlhkosťou plynu a obsahom vlhkosti v látke. Predchádzajúca parametrická metóda sa teraz používa na vlhký materiál. Medzi mnohé parametre, ktoré sú citlivé na obsah vlhkosti, patria napríklad hmotnosť, rozmer a elektrické charakteristiky ako napríklad odpor a kapacita.

Snímač vlhkosti, ktorý sme vybrali využíva absorpčnú metódu. Senzor HTU21D od firmy measurement SPECIALTIES používa laserovo vyrobený, plastový kapacitný citlivý člen.

### 3.6.4. Oživenie hardvéru

Všetky dosky plošných spojov boli vyrobené firmou, ktorá sa touto problematikou zaoberá profesionálne. Dosky sú obojstranné, každá s niekoľkými prekoveniami. Ako prvý som osádzal plošný spoj napájacieho zdroja, po premeraní studených spojov a možných skratov, zdroj po zapojení fungoval bezproblémovo.

Po osadení oboch dosiek s logickými a výkonovými obvody sme prišli na to, že pri návrhu boli prehodené enable piny na dekóderoch. Vstup, ktorý mal byť pripojený na mikrokontrolér bol prehodený s pinom, ktorý mal byť na GND. Táto chyba nebola až tak závažná, preto sme sa chybu rozhodli opraviť namiesto výroby nových DPS. Vodičom sme jeden pin pripojili na GND a druhý sme vyhli a priletovali na správne miesto. Rovnaká chyba sa vyskytla aj na dekóderi, ktorý sa nachádza na doske s mikrokontrolérom, kde sme problém vyriešili rovnako.

Na doske s mikrokontrolérom sme objavili pri oživovaní ešte jednu chybu a to, že sme nedopatrením zabudli pripojiť jeden výstup z mikrokontroléra na konektor pre displej BUSE. Problém sme vyriešili rovnako ako ostatné, pomocou káblovej prepajky od mikrokontroléra ku konektoru.

Pred samotným programovaním mikrokontroléra sme otestovali jeho vstupy a výstupy pomocou pripojených LED diód a tlačítka. Program sa podarilo nahráť bez problémov, teda JTAG funguje. Samotný program fungoval aj v debug móde, ktorý sme pri ďalšom programovaní veľmi ocenili pri hľadaní chýb v napísanom kóde.

Zapojenie konektora pre displej sme najskôr skontrolovali vizuálne podľa schémy zapojenia. Potom sme napísali program, ktorý nastavoval postupne jednotlivé piny odpovedajúce stĺpcom a pomocou logického analyzátora kontrolovali plnú funkčnosť zapojenia.

Po otestovaní so samotným displejom sme došli k rovnakému záveru a tým sme usúdili, že hardvér pracujúci s displejom je funkčný.

K mikropočítaču Raspberry Pi žiadny prídavný hardvér nebol, okrem snímačov pre meranie fyzikálnych parametrov v interiéri. Tie sme otestovali tak, že sme ich pripojili k mikropočítaču a následne sa ich pomocou príkazu `sudo i2cdetect -y`. Po zadaní tohto príkazu nás terminál informoval o všetkých pripojených zariadeniach na tejto zbernici medzi ktorými boli aj naše snímače.

Po otestovaní funkčnosti všetkých navrhnutých častí sme prešli k druhej časti práce, ktorým je programové vybavenie mikrokontroléra a mikropočítača. Vývoju softvéru je venovaná celá nasledujúca kapitola.

## 4. SOFTVÉR

Programové vybavenie je rozdelené do niekoľkých samostatných častí. Radič displeja, teda mikrokontrolér AtMega128 je programovaný v jazyku C. Mikropočítač Raspberry Pi obsahuje programy, ktoré sú písané v jazyku Python. Webové stránky sú vytvorené pomocou kombinácie niekoľkých nástrojov. Bol použitý značkový jazyk HTML v kombinácii s CSS, skriptovací jazyk PHP a skriptovací jazyk JavaScript. Výber jazykov bol založený na požiadavkách a vlastných skúsenostiach s danými nástrojmi.

### 4.2. Programové vybavenie AtMega128

Ako už bolo spomenuté, mikrokontrolér je programovaný v jazyku C. Ide o štandardný programovací jazyk, ktorý patrí medzi najpoužívanejšie. Je to jazyk na úrovni bližšie hardvéru a je podobný strojovo orientovaným jazykom viac ako väčšina jazykov vyššej úrovne.

Kvôli prehľadnosti sme celý program rozdělili na niekoľko samostatných modulov, kde každý obsahuje svoj hlavičkový `.h` súbor a k nemu zdrojový `.c` súbor. Modul `uart.h` obsahuje funkcie, ktoré obsluhujú sériovú komunikáciu medzi mikrokontrolérom a mikropočítačom Raspberry Pi a zároveň sa starajú o kontrolu prijatých dát po sériovej linke a odoslanie odpovede. Tento modul používa funkciu zo súboru `CRC.h`, ktorá slúži na vytvorenie kontrolného súčtu prijatých dát. V súbore `matrix.h` nájdeme všetky potrebné funkcie k spracovaniu dát, ktoré sa majú na displej vykresliť, potrebné konštanty a funkciu pre inicializáciu portov a globálnych premenných. Tieto premenné sú definované v súbore `global.h`.

#### 4.2.1. global.h

Zoznam globálnych premenných:

```
unsigned int gX;  
unsigned int gY;  
unsigned char gColour;  
struct command gCmd;  
unsigned char gOriginal[3][140];  
unsigned char gReciveBuffer[8];  
unsigned char gTransmitBuffer[8];
```

Celočíselné premenné `gX` a `gY` slúžia ako pomocné premenné pre indexovanie riadkov a stĺpcov na displeji. Premennú `gX` používame v rozsahu 0 – 140 a slúži na index stĺpcov, prenná `gY` 0-19 pre indexovanie riadkov.

Premenná gColour nadobúda hodnoty 1 a 0, kde hodnota 1 znamená, že vypísaný obraz alebo text na displeji bude žltou farbou na čiernom podklade, hodnota 0 farby invertuje.

V štruktúre gCmd sú uložené príkazy pre obsluhu zobrazeného obrazu. Jej výpis je nasledovný:

```
struct command
{
    unsigned char direct;
    unsigned char step;
    unsigned char units;
    uint32_t delay;
};
```

Význam jednotlivých položiek v štruktúre je vysvetlený v kapitole uart.h. Do poľa gOriginal sa ukladá obraz, ktorý má byť následne vykreslený. Posledné dve premenné gReceiveBuffer a g TransmitBuffer sú polia, ktoré používame pri odosielaní a prijímaní dát.

## 4.2.2. uart.h

Ako hovorí samotný názov, táto knižnica obsluhuje sériovú komunikáciu a spracovanie prijatých dát. Pre odosielanie a prijímanie dát sme vytvorili jednotný formát paketu, ktorý je zobrazený v tabuľke Tab. 1.

Tab. 4 Formát odosielaných a prijímaných dát

1B	1B	1B	3B	2B
NoP	Length	CMD	Data	CRC

Formát dát, ktoré prijímame a odosielame je rovnaký, rozdiel je len v obsahu položiek CMD a Data. Význam jednotlivých položiek je nasledovný:

**NoP (number of packet)** – označuje číslo paketu, ktorý odosielame z mikropočítača Raspberry Pi, pri odpovedi z mikrokontroléra je v odpovedi na tomto mieste rovnaké číslo, aby sa dalo jednoznačne identifikovať, že odoslaná odpoveď je na paket, ktorý sa prijal. Číslo paketu má ešte jeden dôležitý význam pri ukladaní prijatých dát. Pri prijatí paketu sa celý paket pomocou niekoľkých funkcií, ktorých význam bude vysvetlený neskôr, skontroluje, zostaví sa odpoveď, a pokiaľ je kladná tak prijaté dáta sa uložia na príslušný index do poľa gOriginal.

**Length** – označuje dĺžku paketu. Slúži len na kontrolu, aby sme si mohli byť istý, že nám prišli všetky dáta, ktoré sme chceli odoslať. Dĺžku paketu pred odoslaním spočítame a vložíme na požadované miesto, teda na druhý bajt.

**CMD** – v prípade komunikácie smerom od Raspberry Pi k mikrokontroléru, sú dve možnosti nastavenia tohto bajtu. Ak je  $CMD = 0x05$ , vieme že odosielame dáta pre vykreslenie na displej. Ak sa  $CMD = 0xA0$ , vieme že dáta ktoré nám prišli slúžia na nastavenie štruktúry `gCmd`, ktorá obsahuje príkazy na pracovanie s odrazom ako je napríklad posun obrázku alebo textu po displeji, smer posunu a veľkosť. V prípade komunikácie v opačnom smere, teda odpovedaní na prijaté dáta `CMD` hovorí o tom či prijaté dáta boli v poriadku alebo nie.  $CMD = ACK = 0x0F$  alebo záporná odpoveď  $CMD = NAK = 0xF0$ .

**Data** – tu sa nachádzajú dáta pre vykreslenie v prípade že `NoP` je v rozmedzí od 0 do 139, ak je `NoP` 140, tak sú tu umiestnené 3 bajty príkazov pre štruktúru `gCmd`.

**CRC** – kontrolný súčet, slúži na kontrolu prijatých dát. V oboch prípadoch komunikácie sa kontrolný súčet spočíta z celého paketu od `NoP` po `Data` a vloží sa na posledné dva bajty v reťazci.

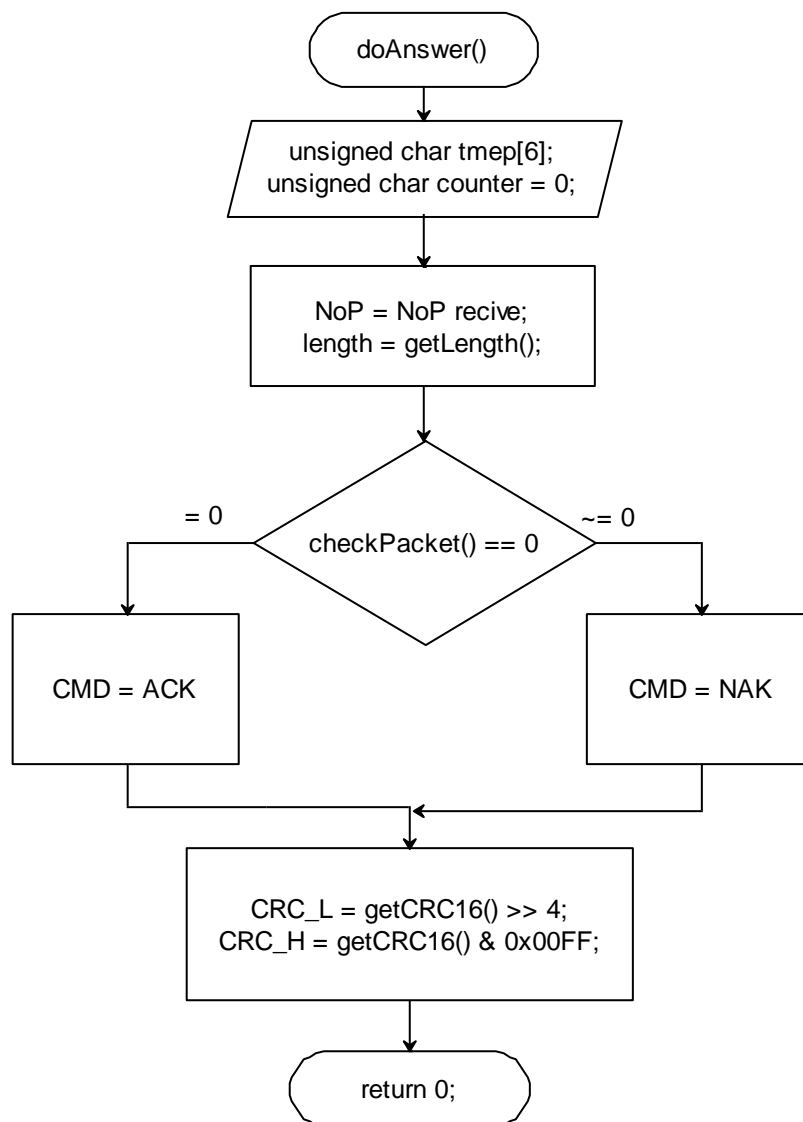
Funkcie, ktoré obsahuje hlavičkový súbor `uart.h` s ich vysvetlením.

```
unsigned char checkPNO(unsigned char packetNumber);
unsigned char checkCMD(unsigned char command);
unsigned char checkLENGTH(unsigned char length);
unsigned char checkCRC(unsigned char crcMSB, unsigned char crcLSB);
unsigned char checkPACKET(void);
```

Všetky vypísané funkcie slúžia na kontrolu prijatých dát. Vstupné parametre sú jednotlivé bajty z prijatého paketu. Každá funkcia vráti 0 pri správnom formáte prijatých dát, 1 pri nesprávnom. Posledná funkcia `checkPACKET(void)` spraví súčet všetkých návratových hodnôt a pokiaľ je rovný 0, vieme že prijaté dáta sú správne a môžeme s nimi ďalej pracovať. Ak funkcia vráti inú hodnotu ako 0, niektoré dáta sú nesprávne a pri vytváraní odpovede na miesto `CMD` vložíme `NAK`.

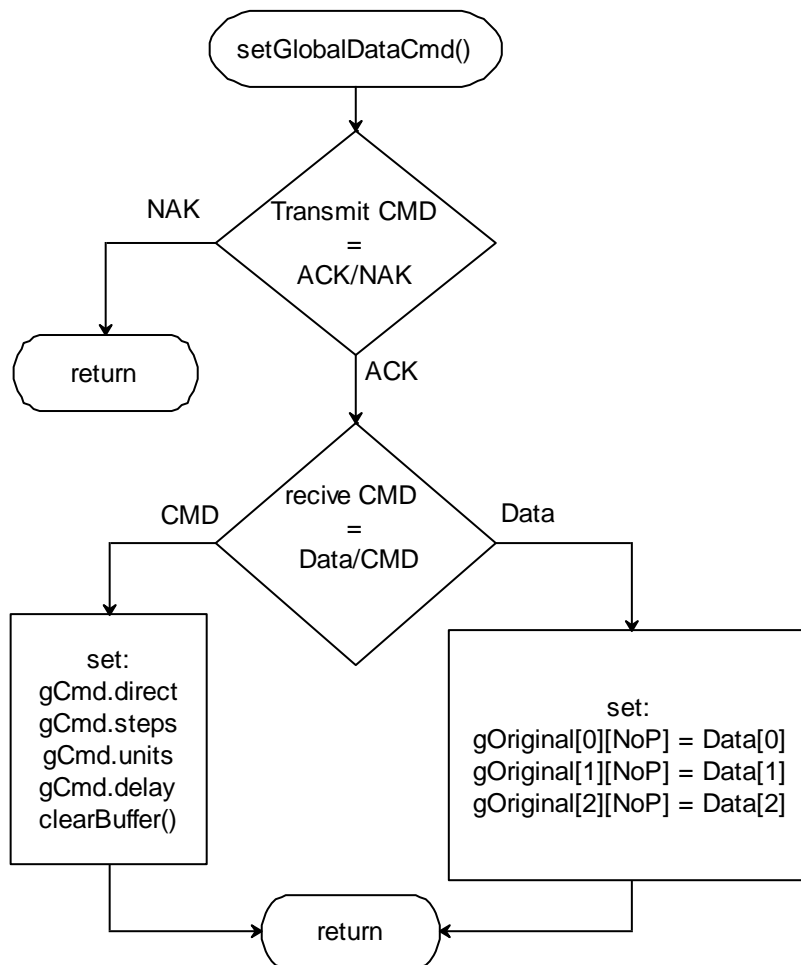
```
unsigned char doAnswer(void);
```

Funkcia `doAnswer(void)` je vysvetlená na vývojovom diagrame na obrázku Obr. 21. Používame dve pomocné premenné, pole o veľkosti 6 bajtov `temp[6]`, ktoré slúži na spočítanie kontrolného súčtu odpovede, a `counter`, ktorá sa inkrementuje pri kontrole paketu. Ak je celá odpoveď správne poskladaná, nemala by obsahovať žiadny bajt s hodnotou `0x00`. Vo funkcii je volaná funkcia pre kontrolu paketu a pomocou jej návratovej hodnoty nastavíme bajt `CMD` v odpovedi na `ACK` alebo `NAK`. Ďalšou volanou funkciou je `getCRC16()`, ktorou spočítame kontrolný súčet a výsledok vložíme na posledné dva bajty odpovede.



Obr. 21 Vývojový diagram funkcie doAnswer()

Dalšou funkciou v tomto hlavičkovom súbore je funkcia pre spracovanie prijatých dát `void setGlobalDataCmd(void)`, ktorá podľa bajtu CMD (Data alebo CMD) uloží prijaté dáta do globálnej premennej gOriginal alebo do premennej gCmd. Vývojový diagram tejto funkcie je na obrázku Obr. 22. Poslednou použitou funkciou je funkcia `void clearBuffer(unsigned char buff[], unsigned char size)` pre vyčistenie bufferu pre prijaté a odoslané dáta. Funkcia má dva vstupné parametre, kde jeden určuje pole, ktoré sa má vyčistiť a druhý určuje jeho veľkosť.



Obr. 22 Vývojový diagram funkcie setGlobalDataCmd()

### 4.2.3. matrix.h

Hlavičkový súbor s názvom matrix.h obsahuje funkcie pre vykresľovanie na displej, definície niektorých portov a použitých konštánt. Na porte C sú pripojené dve LED diódy, ktoré sme používali pri odladovaní softvéru, keďže navrhnutá elektronika je vo vnútri panelu využitie týchto LED diód v hotovom zariadení je zbytočné. Preto jednu z LED diód rozsvietime na začiatku programu a tým signalizuje aspoň stav napájania. Na porte C používame ďalšie dva piny ako výstupné a to pin 6 a 7. Pinom PC6 volíme zapisovanie, PC7 zápis povolíme, teda nastavíme enable na prvých dekodéroch pre riadky aj stĺpce na hodnotu log. 1. Výpis makier pre port C je pod textom.

```

#define LED1          (1<<PC1)
#define LED2          (1<<PC0)
#define WRITE         (1<<PC7)
#define ENABLE        (1<<PC6)
  
```

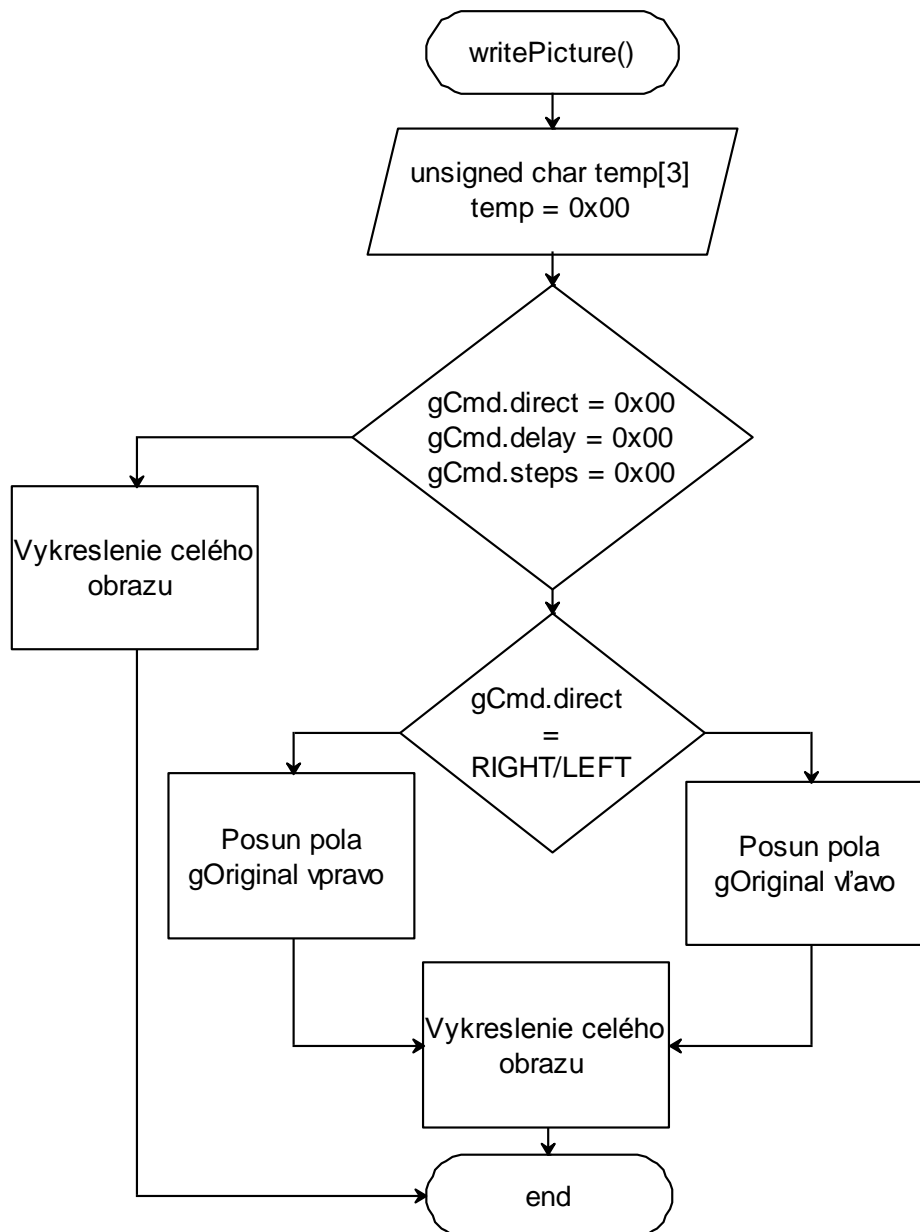
Pre vybratie určitého riadku a stĺpca je potrebné správne nastaviť adresové vstupy na dekodéroch. Pre tento účel sme vytvorili dve statické polia, ktoré sú zároveň konštanty a to `static const unsigned char rowAddr[19]` pre výber riadku a `static const unsigned char columnAddr[140]` pre výber stĺpca.

Prvou funkciou v tomto súbore je funkcia pre nastavenie smeru portov A, B a C s názvom `initPort(void)`. Funkcia nemá žiadnu vstupnú ani návratovú hodnotu. Rovnako ako funkcia `initGlobal(void)`, ktorá inicializuje globálne premenné na začiatku programu.

Pomocou funkcií `setRow(unsigned int nOfRow)` a `setColumn(unsigned int nOfColumn)` nastavíme na príslušné porty také hodnoty z tabuľky, aby sa vybral príslušný riadok a stĺpec. Pomocou `writeAll(void)` a `clearAll(void)` sme schopný celý displej vymazať prípadne celý displej vyfarbiť.

Samotný výpis obrázku, prípadne textu na displej zaobstaráva funkcia `writePicture(void)`, ktorej výpis je na obrázku Obr. 23. Na začiatku tejto funkcii si vytvoríme pole o veľkosti troch bajtov, ktoré požívame pri posune obrázku ako pomocné, na uloženie prvého, prípadne posledného stĺpca v závislosti na smere posunu. Pri pokusoch posúvať obraz smerom hore a dole sme sa rozhodli tieto smery posunu vynechať. Obraz bol vzhľadom na veľkosť displeja nečitateľný. Samotný displej nie je určený na vykresľovanie animácii, preto aj posun vpravo a vľavo pri vyšších rýchlostiach asi pod jednu sekundu nie je esteticky moc vhodný. Pokiaľ má jeden z parametrov štruktúry hodnotu 0x00 vieme, že obraz ktorý vykreslíme sa nebude pohybovať, preto obraz rovno vykreslíme. Ak sú parametre nastavené, skontrolujeme smer, na ktorý sa má obraz posunúť, obraz posunieme a vykreslíme po jednotlivých stĺpcoch. Na konci funkcie je doplnené oneskorenie, ktoré vyberie zo štruktúry `gCmd` hodnotu oneskorenia a jednotky. Potom sa daný počet krát vykoná funkcia `_delay_ms`, ktorá beh programu na nastavenú hodnotu zastaví. Zápis vo funkcii vyzerá nasledovne:

```
for (int l = 0; l < gCmd.delay; l++)
{
    if (gCmd.units == MS)
        _delay_ms(1);
    else if (gCmd.units == S)
        _delay_ms(1000);
    else
    {
        break;
    }
}
```



Obr. 23 Vývojový diagram funkcie writePicture()

Blok „Vykreslenie celého obrazu“ je nasledujúci úsek kódu:

```

for (int i = 0; i < 140; i++)
{
    writeColumn(i);
}
  
```

Výpis prebieha tak, že postupne vykresľujeme stĺpce poľa gOriginal. Jeden stĺpec obsahuje devätnásť bodov. V poli to predstavuje dva celé bajty a tri bity z tretieho bajtu. Na vykreslenie celého bajtu používame ešte jednu pomocnú funkciu, ktorá síce nebola nutná ale kód vyzerá prehľadnejšie. Vykreslenie bajtu prebieha tak, že funkcii

`writeByte(unsigned char symbol)` predáme celý bajt z poľa `gOriginal`, do pomocnej premennej uložíme hodnotu tohto bajtu posunutú štyri krát vpravo a tým dostaneme len vrchné štyri bity. Tieto bity vykreslíme a do pomocnej premennej uložíme tento krát hodnotu bajtu zamaskovanú hexadecimálnou hodnotou `0x0F`, čím dosiahneme to, že dostaneme spodné štyri bajty, ktoré následne rovnako vykreslíme. Vykreslenie posledných troch bitov je pomocou `switch-u`, kde kontrolujeme jeho hodnotu. Keďže kódovanie obrazu máme spravené tak, že v poslednom bajte sú nastavené len prvé tri bity a zvyšok sú nuly, posunieme celý bajt znova štyri krát vpravo a dostaneme osem možných kombinácií čísla, na základe ktorého príslušné bity vykreslíme na displej. Časť výpisu funkcie `writeFourBit(unsigned char symF)` je ukázaná nižšie. Na rovnakom princípe je postavená aj funkcia `writeThreeBit(unsigned char symT)`.

```
void writeFourBit(unsigned char symF)
{
    unsigned char temp = gColour;

    switch(symF)
    {
        case 0x00 : writeXY(~temp); gY++; writeXY(~temp); gY++;
                   writeXY(~temp); gY++; writeXY(~temp); gY++;
                   break;
        case 0x01 : writeXY(~temp); gY++; writeXY(~temp); gY++;
                   writeXY(~temp); gY++; writeXY(temp); gY++;
                   break;
        .
        .
        .
        .
        default   : break;
    }
}

void writeXY(unsigned char colour)
{
    setColumn(gX);
    setRow(gY);
    writeC(colour);
}
```

Funkcia `writeXY(unsigned char colour)`, nastaví príslušný riadok a stĺpec zapíše alebo zmaže bod podľa farby. To znamená, že ak je globálna premenná nastavená na `log. 1`, znamená to, že obrázok bude vykreslený žltou farbou na čiernom podklade, inak povedané nastavené bity v poli `gOriginal` sú žltou farbou, nenastavené čiernou. Má to značnú výhodu, pretože odoslaný obraz nemusíme ručne prekresľovať ak chceme invertovať jeho farbu ale stačí zmeniť hodnotu premennej `gColour` na `log. 0`

a farba obrazu je invertovaná. Do funkcie `writeXY(~temp)` predávame teda ako vstupný parameter farbu ako má byť text vykreslený, ak je hodnota `gColour` log. 1, v prvom case `0x00` budú všetky bity zmazané, teda čierne. V druhom case `0x01`, sú prvé tri bity čierne a posledný žltý. Obdobným princípom sú pokryté všetky kombinácie o hodnoty `0x00` po `0x0F`.

#### 4.2.4. main.c

Po pripojení napájania k mikrokontroléru sa začne vykonávať hlavná funkcia `main.c`. táto funkcia na začiatku inicializuje porty mikrokontroléra, následne sériovú linku a v nekonečnej slučke kontroluje, či je nastavený príznak od sériovej linky. Ak je, dáta sa prijmu, skontrolujú a uložia do odpovedajúcich premenných. Do jedného globálneho poľa sa ukladajú dáta pre vykreslenie a do globálnej štruktúry príkazy pre posun obrazu. Po prijatí všetkých dát sa obraz následne vykreslí na displej a v prípade že sa má posúvať po displeji a tvoriť tak animáciu, neustále sa prekresľuje.

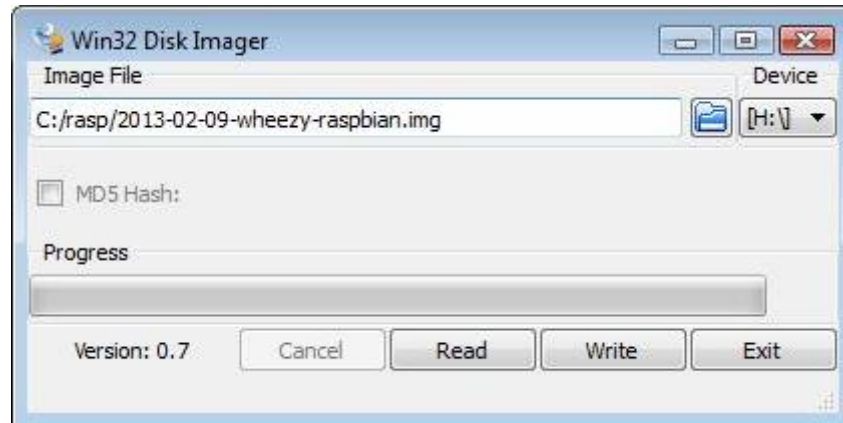
### 4.3. Programové vybavenie Raspberry Pi

Táto kapitola sa zaoberá popisom vytvoreného programového vybavenia pre mikropočítač Raspberry Pi. Je rozdelená do niekoľkých podkapitol, v ktorých popisujeme nahrať operačného systému, spustenie webového serveru, vytvorenie databázy a práca s ňou. Na konci tiež opíšeme obslužný softvér meteostanice a komunikáciu po sériovej linke s mikroprocesorom AtMega.

#### 4.3.1. Operačný systém

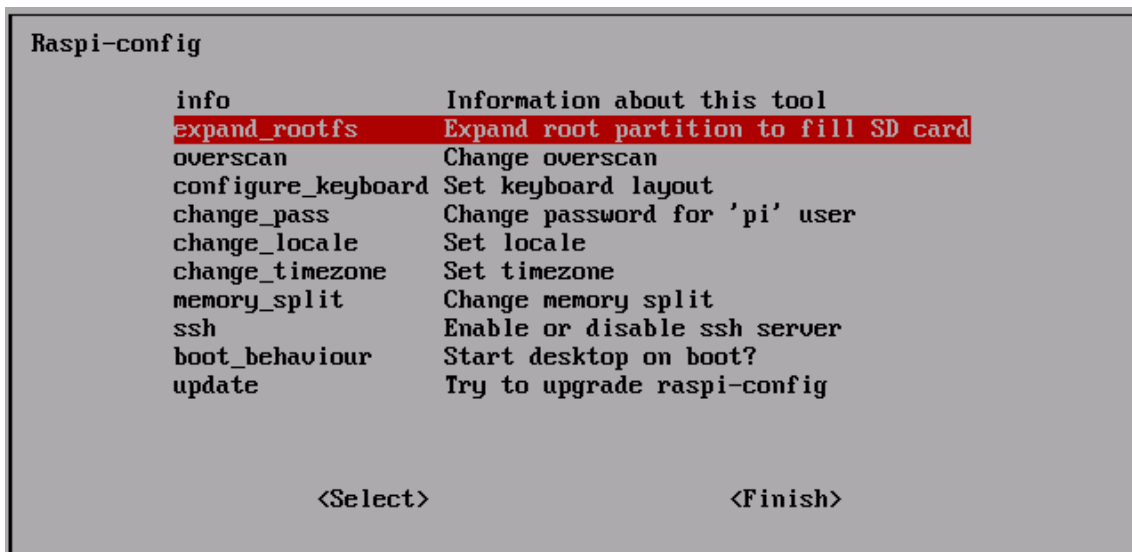
Táto kapitola popisuje základnú inštaláciu systému na SD kartu a jeho nasledovné spustenie. Základný systém sa vojde na kartu o veľkosti 2 GB, my sme použili 8 GB, označenú ako Class 10, čo udáva jej rýchlosť.

Z oficiálnej stránky Raspberry Pi, [22] sme stiahli obraz distribúcie Raspbian „wheezy“ a po jeho rozbalení sme získali súbor s príponou `.img`. Ďalej sme potrebovali nástroj pre Windows, pomocou ktorého dostaneme image súbor na kartu. Zvolili sme najjednoduchšiu variantu a tou je nástroj `Win32DiskImager`. Po jeho stiahnutí, rozbalení a inštalácii sme vybrali príslušnú SD kartu, image súbor a kliknutím na `Write` sa súbor na kartu zapísal. Prostredie programu je ukázané na obrázku Obr. 24. Na karte sa vytvorili dva oddiely, z ktorých vidíme len jeden, pretože druhý je naformátovaný ako `ext4`, čo je linuxový súborový systém.



Obr. 24 Program Win32DiskImager

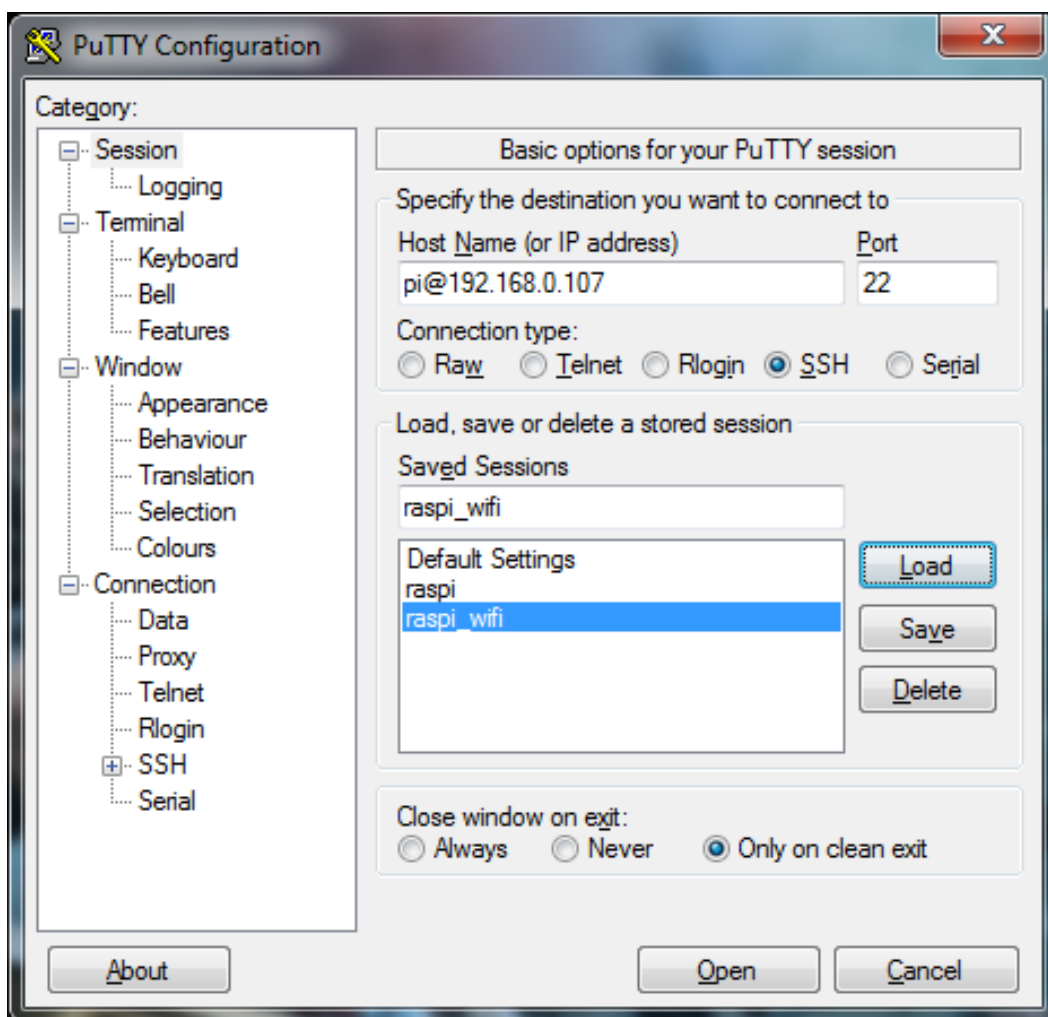
Po zápise sme kartu vložili do mikropočítača a po spustení systému sa zobrazila úvodná obrazovka, ktorá slúži ku konfigurácii. Zaujímavou položkou je „expand\_rootfs“, ktorú sme zvolili a tým sme rozšírili oddiel karty na maximálnu veľkosť a zabezpečili tak dostatok miesta pre systém.



Obr. 25 Okno konfigurácie Raspberry Pi

Ďalším krokom bolo pripojenie mikropočítač a k sieti. Na výber sme mali dve možnosti a to pripojenie pomocou štandardného ethernetového káblu RJ45 alebo bezdrôtovo. Súčasný modely Raspberry Pi nie sú štandardne vybavené žiadnym modulom pre bezdrôtové pripojenie máme však možnosť počítač rozšíriť o podporu Wi-Fi pomocou bezdrôtového adaptéru USB. Pomocou takého zariadenia sa je možné pripojiť k rôznym typom bezdrôtových sietí vrátane tých, ktoré používajú vysokorýchlostný štandard 802.11n. Konfigurácia siete prebieha v systéme Linux a na trhu je dostupné množstvo literatúry ako sieť nakonfigurovať, preto sa nebude touto problematikou podrobnejšie zaoberať.

[23] Aby sme pri práci s počítačom nemuseli využívať monitor, klávesnicu a myš, pripájame sa pomocou sieťového pripojenia. To je možné vďaka tomu, že distribúcia Linuxu Wheezy obsahuje službu SSH. Jedná sa o sieťový protokol zabezpečujúci bezpečný prenos dát. Po zapnutí počítača sa automaticky na pozadí spustí SSH démon. Na počítači s operačným systémom Windows sme použili klientský program s názvom putty.exe, ktorý je voľne dostupný z mnohých internetových zdrojov. Po spustení programu sa objaví úvodná obrazovka, kde je nutné vybrať spôsob pripojenia, zvolili sme SSH. Ďalej je nutné zadať IP adresu zariadenia, na ktoré sa chceme pripojiť, v našom prípade bola mikropočítaču pridelená adresa 192.168.0.107, pred ktorú sme zadali meno užívateľa „pi“. Po potvrdení sa za predpokladu, že je zariadenie na sieti pripojené objavilo terminálové okno a po zadaní správneho hesla „raspberry“ sme sa dostali do rovnakého terminálového okna ako by sme mali monitor pripojený priamo k mikropočítaču. Konfiguračné okno programu putty.exe s nastaveným pripojením je na obrázku Obr. 26.



Obr. 26 Program PuTTY

### 4.3.2. Web server

Veľká časť moderných webových serverov je založených na kombinácii systému Linux, webového serveru Apache, databáze MySQL a jazyka PHP, ktorých kombinácia sa označuje ako sada LAMP. Systém Linux slúži ako základný operačný systém, pomocou MySQL vytvárame a spravujeme databázu, Apache slúži ako webový server a pomocou jazyka PHP vytvárame skripty pre dynamické stránky.

Systém Linux náš počítač už obsahoval, preto bolo potrebné doinštalovať zostávajúce tri balíčky. To sme previedli zadaním nasledujúcich príkazov do terminálového okna:

```
sudo apt-get update
```

```
sudo apt-get install apache 2 php5 php5-mysql mysql-server
```

pomocou kľúčového slova *sudo* sme schopný používať príkazy supeusera, ktoré by sme použiť nemohli ak sme prihlásený ako užívateľ „pi“. Príkaz *update* skontroluje a stiahne všetky dostupné aktualizácie od posledného použitia. Tento príkaz je vhodné použiť vždy po dlhšej neaktivite, aby sme si boli istý, že systém a jeho doplnky sú posledné dostupné. Pomocou druhého príkazu sa spustí inštalácia sady LAMP. Asi v polovici priebehu inštalácie sa zobrazila výzva databázy MySQL pre zadanie hesla. Zvolili sme heslo „database112“ a toto heslo budeme používať pri pristupovaní do databázy. Po úspešnom dokončení inštalácie sa servery MySQL a Apache spustia na pozadí.

Po otestovaní funkčnosti všetkých súčastí sme prešli k vytváraniu vlastných stránok. Postup tvorby týchto stránok je uvedená v kapitole 4.3.5.

### 4.3.3. Databáza

Databáza bude v našej aplikácii súžiť na ukladanie dát získaných zo snímačov a pre ukladanie dát, ktoré sa budú vykresľovať na displej BUSE. Ako prvé je nutné spustiť MySQL klienta. To prevedieme zapísaním príkazu *mysql -u root -p* do okna terminálu. Po potvrdení by sa mal spustiť MySQL, pokiaľ sa nespustí pravdepodobne nie je aktívny na pozadí a je nutné ho spustiť zápisom príkazu *mysqlpd#*. Pre ukončenie klienta sa používa príkaz *quit* a všetky príkazy musia byť ukončené bodkočiarkou.

Po úspešnom prihlásení sme prešli k vytvoreniu samotnej databázy. Pomocou príkazu *create database data* sme vytvorili databázu, ktorej meno je „data“. Úspešné vytvorenie databázy je zrejmé z výpisu *Query OK, 1 row affected (0.01 sec)*. Rovnako si môžeme overiť úspešné vytvorenie príkazom *show database*, ktorý nám vypíše všetky databázy. Aby klient vedel, s akou databázou budeme pracovať je nutné ju vybrať pomocou príkazu *use data*. Aby sme do databázy nepristupovali ako užívateľ „root“, tak sme zadaním nasledujúcich príkazov vytvorili nového užívateľa:

```
mysql> CREATE USER 'matej112'@'localhost' IDENTIFIED BY 'database112';
mysql> GRANT ALL PRIVILEGES ON data.* TO 'matej112'@'localhost';
mysql> FLUSH PRIVILEGES;
mysql> quit;
```

Meno užívateľa je „matej112“ a jeho prístupové heslo je „database112“. Tento nový užívateľ nemá žiadne práva pre prácu s databázou. Na pridelenie práv slúži druhý riadok vo výpise. Tým sme užívateľovi pridelili všetky privilégia, no pri reálnej aplikácie je vhodnejšie pridelovanie práv zväziť a obmedziť ich iba na potrebné úkony, ktoré bude užívateľ s databázou vykonávať. Po úspešnom vytvorení užívateľa sa ďalej budeme prihlasovať pomocou *mysql -u matej112 -p* a zadaním hesla „database112“.

V každej databáze je možnosť vytvoriť niekoľkých tabuliek. My sme vytvorili prvú tabuľku pre ukladanie dát získaných zo snímačov teploty, tlaku a vlhkosti a druhú pre ukladanie dát, ktoré sa majú zobrazovať na displeji BUSE.

```
CREATE TABLE senzor_data (tdate DATE, ttime TIME, temp1 FLOAT, temp2, FLOAT,
temp3 FLOAT, press FLOAT, humidity FLOAT);
```

Zadaním tohto príkazu sme vytvorili tabuľku s názvom „senzor\_data“, ktorá obsahuje sedem stĺpcov. Prvé dva stĺpce sa volajú „tdate“ a „ttime“, ktorých dátové typy sú DATE a TIME a slúžia pre uloženie dátumu a času, kedy boli dáta zo snímačov zosnímané. Ostatné stĺpce sú pre ukladanie konkrétnych dát, ktoré sú ukladané ako desatinné čísla ktoré prezentuje dátový typ FLOAT. S ukladaním dát do databáze pomocou jazyka python je vysvetlené v nasledujúcej kapitole, ktorá je venovaná zberu dát zo snímačov a ich uloženie do príslušných stĺpcov tabuľky s názvom „senzor\_data“.

Rovnakým postupom sme vytvorili aj druhú tabuľku, ktorá slúži na uloženie vytvoreného obrázku, ktorý sa má vykresliť na displeji. Táto tabuľka obsahuje šesť stĺpcov. V prvom stĺpci je uvedené ID záznamu, ktorého hodnota sa pri vložení novej položky automaticky inkrementuje, teda jej hodnotu pri ukladaní neuvádzame. V druhom stĺpci tabuľky sa nachádza samostatný obrázok, ktorý je prezentovaný dátovým typom BLOB, o ktorom sa bližšie zmienime pri konverzii dát do tohto typu a pri jeho vyčítaní. Posledné štyri stĺpce sú venované príkazom, ktoré sa odosielajú po sériovej linke spolu s obrázkom a slúžia na špecifikáciu posunu obrázku po displeji. Dva z nich sú prezentované ako celočíselné hodnoty INT a dva nadobúdajú len dve hodnoty takže sme použili dátový typ BIT. Tabuľka je zobrazená na Obr. 27.

```
mysql> SHOW COLUMNS FROM canvas FROM data;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id         | int(11)   | NO   | PRI | NULL    | auto_increment |
| canvas     | blob      | YES  |     | NULL    |                |
| delay      | int(11)   | YES  |     | NULL    |                |
| step       | int(11)   | YES  |     | NULL    |                |
| unit       | bit(1)    | YES  |     | NULL    |                |
| direction  | bit(1)    | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)

mysql> █
```

Obr. 27 Výpis stĺpcov z tabuľky canvas

### 4.3.4. Spracovanie dát zo snímačov

Pre každý použitý snímač sme vytvorili samostatné moduly v jazyku python, ktoré obsahujú všetky potrebné funkcie pre správne vyčítanie dát zo snímačov. Knižnicu pre riadenie komunikácie po I2C zbernici je voľne dostupná na mnohých internetových stránkach, preto sme si jednu vybrali a použili. Jednotlivé postupy a funkcie použité pri čítaní a spracovaní dát zo snímačov budú bližšie rozobraté v nasledujúcich podkapitolách.

#### 4.3.4.1. Barometer a teplomer BMP085

[19] Tento snímač je umožňuje merať okrem atmosférického tlaku aj teplotu. Hodnoty teploty a tlaku, ktoré vyčítame z príslušných registrov sú nekalibrované. Pre získanie skutočných hodnôt vo fyzikálnych jednotkách je nutné pri inicializácii snímača vyčítať kalibračné dáta z jeho E2PROM pamäte. Tieto hodnoty sú pre každý jeden snímač rôzne a do pamäte sa zapisujú pri výrobe a kalibrácii snímača. Adresy registrov, kde sú uložené jednotlivé konštanty sú vypísané v tabuľke Tab. 5.

Počítanie skutočnej hodnoty teploty je s presnosťou 0,1 °C a tlaku 1 Pa. V nasledujúcich tabuľkách je uvedený podrobný algoritmus pre získanie kalibračných dát, nekompenzovanej hodnoty teploty a tlaku a výpočet skutočných hodnôt.

1. Pred začatím meranie je nutné vyčítať kalibračné dáta. Pre tento účel sme vytvorili prvú funkciu, ktorej skrútený výpis vyzerá nasledovne:

```
def readCalibrationData(self):
    "Reads the calibration data from the sensor"
    self._cal_AC1 = self.readS16(self.__BMP085_CAL_AC1)
    self._cal_AC2 = self.readS16(self.__BMP085_CAL_AC2)
    .
    self._cal_MC = self.readS16(self.__BMP085_CAL_MC)
    self._cal_MD = self.readS16(self.__BMP085_CAL_MD)
```

Tab. 5 Načítanie kalibračných údajov

čítanie registrov z EEPROM, 16 bit, MSB prvý	
AC1 (0xAA, 0xAB)	(16 bit)
AC2 (0xAC, 0xAD)	(16 bit)
AC3 (0xAE, 0xAF)	(16 bit)
AC4 (0xB0, 0xB1)	(16 bit)
AC5 (0xB2, 0xB3)	(16 bit)
AC6 (0xB4, 0xB5)	(16 bit)
B1 (0xB6, 0xB7)	(16 bit)
B2 (0xB8, 0xB9)	(16 bit)
MB (0xBA, 0xBB)	(16 bit)
MC (0xBC, 0xBD)	(16 bit)
MD (0xBE, 0xBF)	(16 bit)

2. V druhom kroku zo snímača vyčítame nekompensovanú hodnotu teploty:

```
def readRawTemp(self):
    "Reads the raw (uncompensated) temperature from the sensor"
    self.i2c.write8(self.__BMP085_CONTROL, self.__BMP085_READTEMPCMD)
    time.sleep(0.005) # Wait 5ms
    raw = self.readU16(self.__BMP085_TEMPDATA)
    return raw
```

Tab. 6 Čítanie teploty

zápis 0x2E do registra 0xF4, čakanie 4.5 ms
čítanie reg. 0xF6 (MSB), 0xF7 (LSB)
UT=MSB << 8 + LSB

3. Následne čítame nekompenzovanú hodnotu tlaku (funkcia je obsiahlejšia preto neuvádzame celý jej výpis):

```
def readRawPressure(self):
```

*Tab. 7 Čítanie tlaku*

zápis 0x34 + (oss << 6) do reg. 0xF4, čakanie
čítanie reg. 0xF6 (MSB), 0xF7 (LSB), 0xF8 (XLSB)
UP=(MSB<<16 + LSB<<8 + XLSB) >> (8 - oss)

4. Vo štvrtom kroku vypočítame s použitím získaných nekompenzovaných hodnôt tlaku a teploty a kalibračných dát skutočné hodnoty.

```
def readTemperature(self):
```

```
"Gets the compensated temperature in degrees celcius"
```

```
def readPressure(self):
```

```
"Gets the compensated pressure in pascal"
```

*Tab. 8 Výpočet skutočnej teploty*

$X1 = (UT - AC6) * AC5 / 2e15$ $X2 = MC * 2e11 / (X1 + MD)$ $B5 = X1 + X2$ $T = (B5 + 8) / 2e4$
---

*Tab. 9 Výpočet skutočného tlaku*

$B6 = B5 - 4000$ $X1 = (B2 * (B6 * B6 / 2e12)) / 2e11$ $X2 = AC2 * B6 / 2e11$ $X3 = X1 + X2$ $B3 = ((AC1 * 4 + X3) << oss + 2) / 4$ $X1 = AC3 * B6 / 2e13$ $X2 = (B1 * (B6 * B6 / 2e12)) / 2e16$ $X3 = ((X1 + X2) + 2) / 2e2$ $B4 = AC4 * (\text{unsigned long})(X3 + 32768) / 2e15$ $B7 = ((\text{unsigned long})UP - B3) * (50000 >> oss)$ $\text{if } (B7 < 0x80000000) \{ p = (B7 * 2) / B4 \}$ $\text{else } \{ p = (B7 / B4) * 2 \}$ $X1 = (p / 2e8) * (p / 2e8)$ $X1 = (X1 * 3038) / 2e16$ $X2 = (-7357 * p) / 2e16$ $p = p + (X1 + X2 + 3791) / 2e4$
--

Knižnica obsahuje ešte jednu funkciu, ktorá slúži na inicializáciu senzoru. V tejto funkcii je použitá aj funkcia pre zistenie kalibračných dát z prvého kroku.

Rovnako aj funkcie z kroku dva a tri sú použité vo funkciách pre výpočet skutočných hodnôt. Vo výslednom programe, ktorý zabezpečuje získanie a uloženie dát do databázy v úvode zavoláme funkciu `def __init__(self, address=0x77, mode=1)`, ktorou snímač inicializuje a následne funkcie z bodu štyri, ktorých návratová hodnota je skutočná hodnota tlaku a teploty.

#### 4.3.4.2. Snímač vlhkosti a teplomer HIH6130

[21] Tento snímač vlhkosti umožňuje rovnako ako použitý barometer aj meranie teploty. Výrobca udáva meranie teploty s presnosťou na 0,3 °C v rozsahu od -40 °C do 125 °C. relatívnu vlhkosť v rozsahu 0 – 100 % s presnosťou na 2 % RH. Výpočet skutočnej relatívnej vlhkosti a skutočnej teploty je o niečo jednoduchšie ako pri výpočte atmosférického tlaku.

Pre prácu so snímačom sme vytvorili knižnicu, ktorá obsahuje dve metódy. Jednu pre inicializáciu snímača a druhá pre vyčítanie dát. Oproti predchádzajúcemu riešeniu s barometrom sme vytvorili triedu s názvom HIH6130 a v nej metódu `__init__(address = 0x27)`, pomocou ktorej snímač inicializujeme a druhú metódu `read()`.

```
def read(self):
    try:
        self._buffer = self.i2c.read_i2c_block_data(self.address, 0, 4)
    except:
        raise IOError("Could not read from i2c device located at %s." %
            self.address )

    self.timestamp = datetime.now()
    self.status = self._buffer[0] >> 6 & 0x03
    self.rh = round(((self._buffer[0] & 0x3f) << 8 | self._buffer[1])*
        100.0 / (2**14 - 1), 2)
    self.t = round((float((self._buffer[2] << 6) + (self._buffer[3] >>
        2)) / (2**14 - 1)) * 165.0 $

    return
```

Táto metóda nám vráti čas, kedy bola hodnota zmeraná, relatívnu vlhkosť vzduchu a teplotu. Túto metódu zavoláme v skripte `read_data.py` a získané hodnoty spolu s hodnotou atmosférického tlaku a druhej teploty uložíme do databázy.

#### 4.3.4.3. Získanie a uloženie dát

Pre získavanie a ukladanie dát do databázy sme vytvorili ďalší skript s názvom `read_data.py`, ktorý obstaráva celú komunikáciu po zbernici I2C. Na začiatku importujeme súbory `BMP085.py`, `HIH6130.py`, `time.py` a knižnicu pre prácu

s databázou. Následne inicializujeme oba snímače a získame z nich už skutočné prepočítané hodnoty.

```
import BMP085
import HIH6130
import time
import MySQLdb

# inicializácia snímačov
bmp = BMP085(0x77)
rht = HIH6130()

# získanie dát pomocou vytvorených funkcií
temp1 = bmp.readTemperature()
pressure = bmp.readPressure()
temp2 = rht.t
humidity = rht.rh
```

Tieto hodnoty ukladáme do vytvorenej databázy, preto je nutné sa najskôr so databázy prihlásiť a následne hodnoty uložiť do tabuľky.

```
# prihlásenie
db = MySQLdb.connect("localhost","matej","database112","data")
curs = db.cursor()

# vloženie dát
try:
    curs.execute ("""INSERT INTO sensor_data values(CURRENT_DATE()-
INTERVAL 1 DAY, NOW(), %s, %s, %s, %s, %s)""",(temp1, temp2,
temp3,pressure/100.0, humidity))

    db.commit()
    print "Data commited"

except:
    print "Error: the database is being rolled back"
    db.rollback()
# nakoniec databázu zatvoríme
db.close()
```

### 4.3.5. Uživatelské rozhranie

Cieľom práce bolo navrhnúť užívateľské rozhranie pre zobrazenie zmeraných dát zo snímačov a ovládanie displeja BUSE. Pomocou skriptovacieho jazyka PHP a značkovacieho jazyka HTML sme vytvorili niekoľko internetových stránok, kde môže užívateľ zobrazovať dáta a ovládať displej.

Pre pohybovanie sa medzi jednotlivými stránkami sme vytvorili jednoduché dvojúrovňové menu, pomocou ktorého je možné prechádzať medzi jednotlivými stránkami. Menu je zobrazené na obrázku Obr. 28 a jeho HTML kód je nasledovný:

```
<hr><nav align="center">
<ul>
  <li><a href="index.php"><b>Introduction</b></a></li>
  <li><a href="#"><b>Weather Station</b></a>
  <ul>
    <li><a href="temperature_1.php">Temperature 1</a></li>
    <li><a href="temperature_2.php">Temperature 2</a></li>
    <li><a href="temperature_3.php">Temperature 3</a></li>
    <li><a href="humidity.php">Humidity</a></li>
    <li><a href="press.php">Pressure</a></li>
  </ul></li>
  <li><a href="#"><b>Display BUSE BS210</b></a>
  <ul>
    <li><a href="basic_info">Basic Info</a></li>
    <li><a href="basic_control">Basic Control Display</a></li>
    <li><a href="#">Timetable</a></li>
  </ul></li></ul>
</nav>
<hr>
```

- **Introduction**
- **Weather Station**
  - Temperature 1
  - Temperature 2
  - Temperature 3
  - Humidity
  - Pressure
- **Display BUSE BS210**
  - Basic Info
  - Basic Control Display
  - Timetable

*Obr. 28 Neformátované menu*

Menu zobrazené na obrázku nevyzerá moc prepracovane, preto sme pomocou jednoduchého mechanizmu, ktorý sa označuje ako CSS alebo kaskádové štýly vytvorili na pohľad príjemnejší vzhľad. Formátované menu je na nasledujúcom obrázku.



*Obr. 29 Menu formátované pomocou CSS*

Prejdením myšou na niektorú z položiek sa nám rozbalia vnorené položky. Po kliknutí na niektorú z nich sa pomocou hypertextového odkazu dostaneme na príslušnú stránku. Formátovanie dokumentu pomocou CSS je možné viacerými spôsobmi. Jeden zo spôsobov je vložiť kód CSS do hlavičky .html alebo .php dokumentu. Tento spôsob sa využíva najmä vtedy ak používame formátovanie len raz. My máme však naše menu na všetkých stránkach, aby sme sa mohli späť vrátiť na ľubovoľnú stránku, preto sme si vytvorili samostatný dokument test.css, kde je tento kód napísaný a do hlavičky dokumentu vložíme odkaz na tento súbor, ktorý vyzerá nasledovne:

`<link rel="stylesheet" href="test.css" type="text/css">`. Tým dosiahneme to, že všade kde bude tento odkaz vložený bude formátovanie menu rovnaké. Obsah súboru test.css je uvedený na CD, ktoré je súčasťou prílohy.

Všetky stránky pre zobrazenie dát, temperature1.php, temperature2.php, temperature3.php, pressure.php a humidity.php, majú rovnaký celý obsah. Rozdiel je len v hlavičke html dokumentu a dátach, ktoré sa zobrazujú. Z tohto dôvodu popíšeme len jednu z nich a to temperature1.php. vo vrchnej časti stránky je hlavný nadpis a pod ním sú popísané základné vlastnosti daného snímača ako presnosť merania danej fyzikálnej veličiny, v tomto prípade teploty, a princíp na akom snímač funguje. Pod základnými informáciami sa nachádza spomínané formátované menu. Nasleduje tabuľka zmeraných dát, ktorá obsahuje tri stĺpce, dátum, čas a zmeranú hodnotu teploty. V tabuľke sa zobrazuje posledných tridsať zmeraných hodnôt. Hlavička tabuľky je zobrazená na obrázku Obr. 30. V druhom riadku v tabuľke by mala byť posledná zmeraná hodnota ale pri písaní tohto textu sme nemali prístup do databáze.

Date	Time	Temperature [C]
\$.Srow["tdate"]."	\$.Srow["ttime"]."	\$.Srow["temp1"]."

*Obr. 30 Hlavička tabuľky z webovej stránky*

Dáta sú síce usporiadané v tabuľke, no na prvý pohľad to môže vyzerat' trochu chaoticky. Z tohto dôvodu sme sa rozhodli pridať aj grafické zobrazenie dát. Graf, ktorý

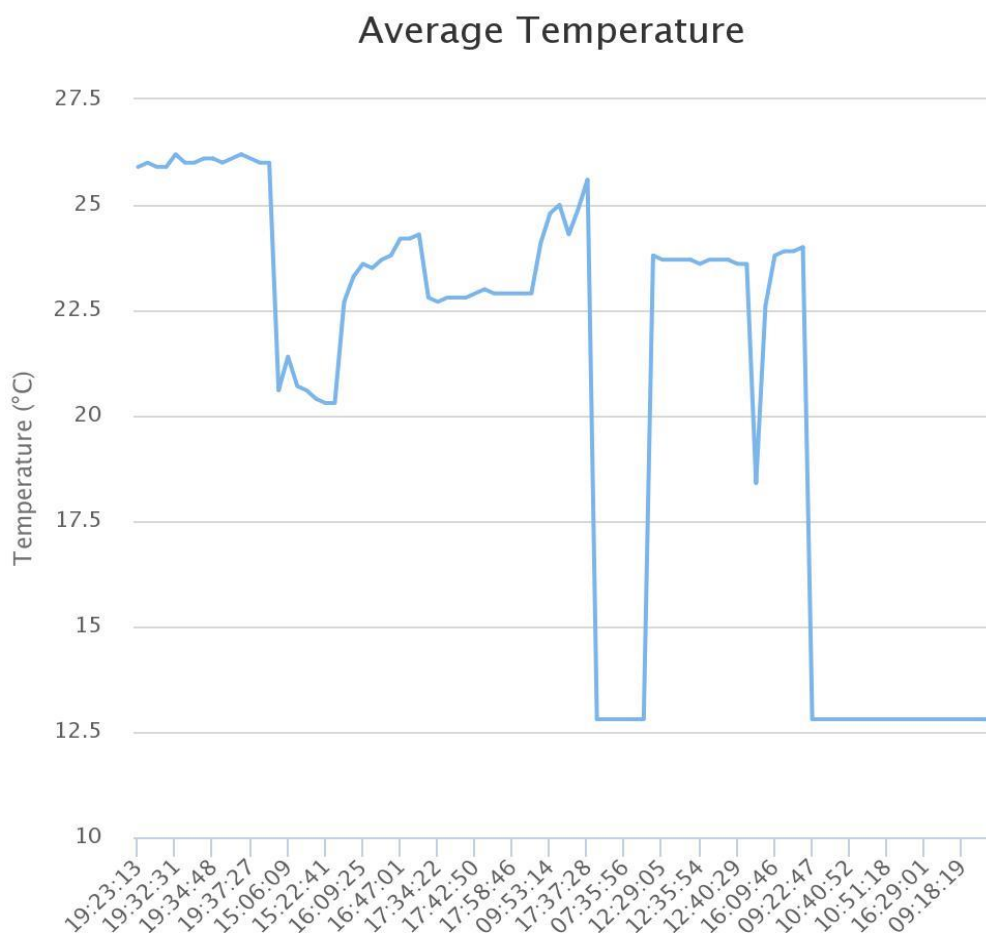
sme na stránku pridali je písaný v skriptovacom jazyku Java. S týmto jazykom som sa stretol po prvý krát, preto som sa rozhodol použiť knižnice pre tvorbu grafov, ktoré sú pre nekomerčné účely voľne dostupné na internete. Konkrétne sa jedná o knižnicu exporting.js a highcharts.js. Jazyk Java neumožňuje priamo pristupovať do SQL databáze preto sme museli dáta z databáze dostať pomocou jazyka PHP a jednoduchého skriptu. Pomocou PHP sa najskôr prihlásime do databázy:

```
<?php
    /*Otvoríme spojenie s MySQL databázou */
    $mysqli = new mysqli("localhost", "matej", "database112", "data");
    /* Kontrola pripojenia. */
    if (mysqli_connect_errno()) {
        printf("Connect failed: %s\n", mysqli_connect_error());
        exit();
    }
    /* Vyberieme dáta všetky dáta z tabuľky sensor_data */
    $data=mysqli_query($mysqli,"SELECT * FROM sensor_data");
?>
```

Jednoduchým skriptom potom vytvoríme dve polia, myData kde uložíme hodnoty teploty a myLabels kde uložíme čas kedy bola hodnota zmeraná.

```
<script>
    var myData=[<?php
    while($info=mysqli_fetch_array($data))
        echo $info['temp1'].' ','?>];
    <?php
        $data=mysqli_query($mysqli,"SELECT ttime FROM sensor_data");
    ?>
    var myLabels=[<?php
    while($info=mysqli_fetch_array($data))
        echo ''.$info['ttime'].' ','?>];
</script>
```

Takto získané polia použijeme vo funkcii a vykreslíme požadovaný graf. Príklad tohto grafu pre teplotu získanú z barometra je na obrázku Obr. 31.



*Obr. 31 Grafické zobrazenie zmeraných hodnôt*

Pre ovládanie displeja sme vytvorili ďalší súbor canvas.html. Táto webová stránka umožňuje užívateľovi vytvoriť obraz, ktorý sa následne uloží do databázy a pomocou skriptu napísaného v jazyku python, je možné uložený obraz alebo text odoslať po sériovej linke mikrokontroléru a ten ho následne vykreslí. Komunikácia a spôsob odoslania dát je podrobnejšie popísané v kapitole 4.3.6.

Vo vrchnej časti stránky sa nachádza pole o šírke 980 a výške 133 pixelov. Toto pole predstavuje displej, kde jeden bod na displeji je prezentovaný štvorčekom o veľkosti strany sedem pixelov. Rozmer sme volili tak, aby bolo pole zobrazené na stránke dostatočne veľké kvôli čitateľnosti a aby bolo možné bezproblémovo jednotlivé body vyberať. Pod týmto polom sa nachádzajú štyri vstupné polia, pomocou ktorých je užívateľ schopný nastaviť, čo sa má s uloženým obrazom po odoslaní diať. Jedná sa príkazy pre mikrokontrolér AtMega, pomocou ktorých zvolíme posunutie obrazu do určitého smeru o nastavený počet krokov a s nastaveným časovým oneskorením vo zvolených časových jednotkách. Ak je niektorá z položiek direct, steps, units alebo delay rovná nule uložený obraz po odoslaní len vykreslí.

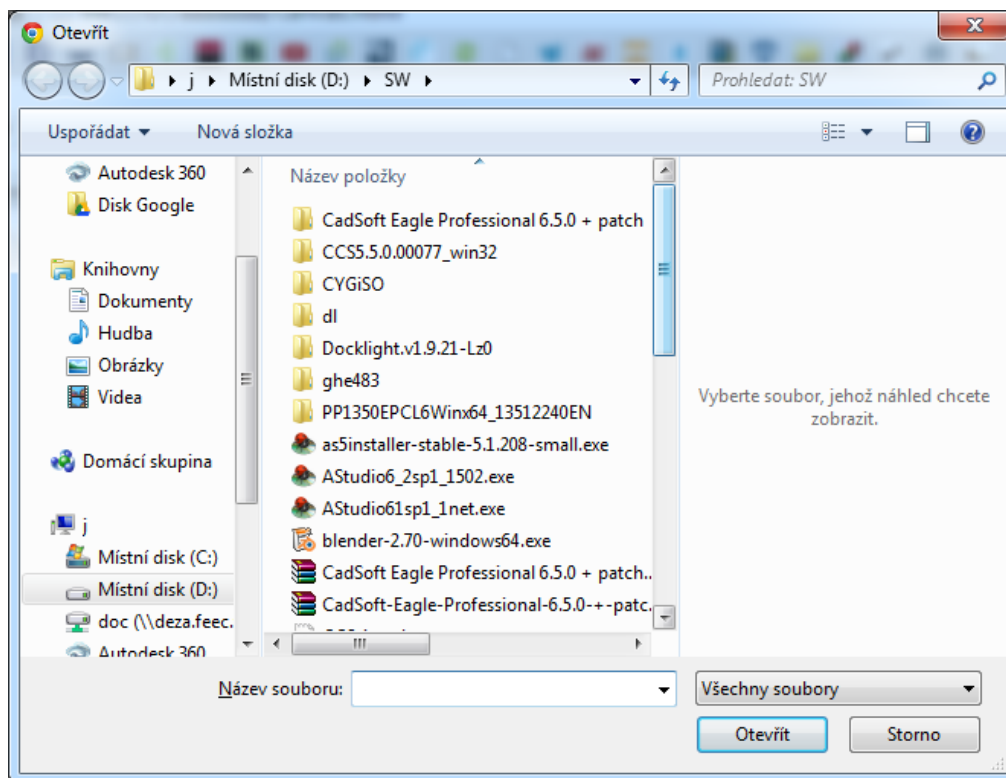
Ďalej má užívateľ k dispozícii štyri tlačítka. Tlačítkom Clear sa pole prepíše na nové, celé biele. Kliknutím na Save to database uložíme vytvorený obraz do databázy vrátane príkazov pre posun obrázka. Dáta z poľa pred uložením do databázy musíme upraviť. Keďže sme ako prvé písali programové vybavenie na strane mikrokontroléra, dáta upravíme tak aby sme mohli už napísaný a otestovaný kód bez zásahov použiť. Prvou vecou je, že dáta sú v poli prezentované reťazcom ‚true‘ alebo ‚false‘. Tieto hodnoty najskôr zmeníme na hodnoty 0 a 1 a rozdelíme ich na jednotlivé stĺpce. Takto získaný textový reťazec obsahuje teda iba znaky 0 a 1. Tento reťazec prevedieme pomocou funkcie parseInt(data, 2) na 32 bitové číslo. Týchto čísel máme celkom 140, pre každý stĺpec na displeji jedno. Následne tieto dáta prevedieme na dátový typ BLOB a uložíme spolu s príkazmi pre mikrokontrolér do odpovedajúcich stĺpcov tabuľky v databáze. Celá obsluha tlačítka Save:

```
$('#save').click(function () {
    var direction = $('#direction option:selected').val();
    var time = $('#time option:selected').val();
    var late = $('#late').val();
    var step = $('#step').val();
    var data = [];

    for (var j = 0; j < 140; j++) {
        var temp = "";
        for (var i = 0; i < 19; i++) {
            if (array[i][j])
                temp += 1;
            else
                temp += 0;
        };
        data.push(parseInt(temp, 2));
    };
    $.post( "/canvas.php", {'save' : true, 'direction' : direction,
'canvas' : data, 'late' : late, 'time' : time, 'step' : step}, null,
'json').done(function(data) {
        console.log(data);
    });
});
```

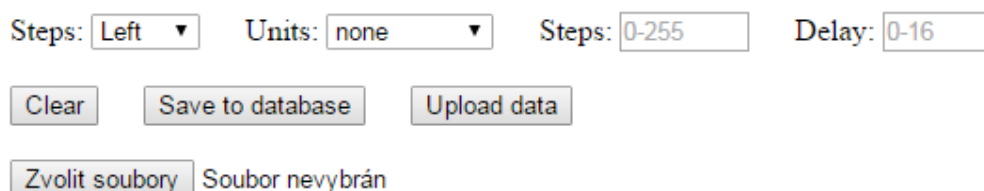
Aby sme mohli zobrazit' aj dáta zo snímačov implementovali sme tlačítko Upload data, pomocou ktorého načítame posledné zmerané hodnoty dát z databáze a v predpísanom formáte ich vložíme do poľa. V prípade, že už máme nejaký obraz vytvorený a uložený napríklad v počítači, nemusíme tento obrázok prekresľovať do nášho poľa ale jednoducho ho vyberieme a nahráme do poľa. Obrázok musí byť

veľkosti 140 krát 19 pixelov, inak vyskočí chybové hlásenie a obraz sa nenahrá. Po kliknutí na tlačítko Zvoliť súbory sa otvorí klasické dialógové okno pre výber obrázku, ktoré je zobrazené na obrázku Obr. 32, pod textom.



Obr. 32 Dialógové okno pre výber obrázku

Menším obmedzením je možnosť vyberať len súbory formátu wbmp. Jedná sa o monochromatický grafický formát súboru optimalizovaný pre mobilné výpočtové zariadenia. Tento obraz je čierne biely, takže jeho veľkosť je obmedzená na minimum. Čierny pixel je prezentovaný hodnotou nula, biely jedna. Pre prekonvertovanie súboru napríklad vo formáte bitmapy .bmp na súbor formátu .wbmp je možné pomocou rôznych konvertorov, my sme používali online konvertor dostupný na [24]. Grafické rozloženie týchto ovládacích prvkov na webovej stránke je zobrazené na obrázku Obr. 33.



Obr. 33 Rozloženie ovládacích prvkov

Užívateľ má celkovo štyri možnosti ako obraz vytvoriť. Prvou je spomínané nahranie obrazu z počítača a druhou je načítanie meteorologických dát z databázy. Pre zobrazenie dát sme vybrali teplotu zmeranú pomocou barometra BMP180, atmosférický tlak a relatívnu vlhkosť vzduchu. Tieto dáta sa zobrazujú v predpísanom formáte, tak ako je možné vidieť na obrázku Obr. 34.



Obr. 34 Načítanie dát z databázy a vloženie do obrazu

Dáta vyberieme z databázy pomocou PHP :

```
$data = mysqli_query($db, "SELECT temp1, press, humidity FROM  
    sensor_data ORDER BY tdate DESC, ttime DESC LIMIT 1");  
echo json_encode(mysqli_fetch_array($data));
```

a následne ich vypíšeme do poľa pomocou funkcie napísanej v jazyku JavaScript, ktorá sa vyvolá po kliknutí na tlačítko Load Sensor:

```
$('#load_sensors').click(function () {  
    $.post( "/canvas.php", {'save' : false}, null, 'json')  
    .done(function(data) {  
        writeData(data['temp1'], data['press'], data['humidity'])  
    });  
});
```

Tretou možnosťou je vkladať text, a to písaním priamo z pripojenej klávesnice. Implementovali sme len jeden font o veľkosti 8 x 5 pixelov na znak. Font je napísaný ručne pre každý znak. Jednotlivé znaky, tak ako vyzerajú vo webovom prehliadači, sú na obrázku Obr. 35. Pri zvažovaní možnosti o viacero znakových sád a štýlov písma nás odradili obmedzené možnosti displeja. Pre zložitejšie fonty je displej malých rozmerov a písmo by na ňom nevyniklo. Text vkladáme do poľa tak, že po stlačení nejakej klávesy, ktorá sa nachádza v našej znakovnej sade, sa zavolá funkcia `writing(data)`, ktorá volá funkciu `writeChar(data.keyCode)`. Druhá spomínaná funkcia obsahuje `switch`, pomocou ktorého vyberiem znak, ktorý bol stlačený a ten sa následne vykreslí do obrázku. Pre predstavu ako je prezentovaný jeden znak uvádzame výpis kódu pre písmeno A:

```

case 'A'.charCodeAt(0):
    pattern = [ [0,1,1,1,0],
                [1,0,0,0,1],
                [1,0,0,0,1],
                [1,0,0,0,1],
                [1,0,0,0,1],
                [1,1,1,1,1],
                [1,0,0,0,1],
                [1,0,0,0,1],
                [1,0,0,0,1],
                [1,0,0,0,1], ];
    break;

```

Na konci tejto funkcie posunieme kurzor aby sme jednotlivé znaky neprepisovali a pridáme jeden voľný stĺpec medzi jednotlivé znaky. Text sa vypisuje od prvého riadku zľava a keď dôjdeme na koniec riadku, kurzor sa presunie na druhý riadok úplne vľavo a v písaní môžeme pokračovať.



*Obr. 35 Vytvorená znaková sada pre displej BUSE*

Pre prípad, že by užívateľ chcel na displej dostať niečo iné ako hotový obrázok alebo text vo vytvorenom štýle pridali sme štvrtú možnosť ovládania displeja a to je označovanie jednotlivých pixelov (7 x 7) ručne, pomocou myši. Pre zmenu farby jednotlivých bodov sme vytvorili funkciu, ktorá po kliknutí na určitý bod v poli najskôr zistí jeho aktuálnu hodnotu (farbu) a potom ju zmení na opačnú. Tým máme možnosť pri vytváraní obrázku opraviť prípadné chyby alebo urobiť dotatočné úpravy bez toho, aby sme museli celú plochu zmazať pomocou volania funkcie `clearCanvas(array2D)`, ktorá vytvorí nové, prázdne pole a aktuálne týmto prepíše. Táto funkcia sa volá po stlačení tlačidla Clear. Funkcia pre zmenu farby jednej bunky je uvedená tu:

```

function updateCanvas(data) {
    $("#canvas > table > tbody > tr").each(function(i, array) {
        $(this).children().each(function(j, array) {
            if (data[i][j] == true) {
                $(this).css("background-color", "black");
            }
            else {
                $(this).css("background-color", "white");
            }
        });
    });
}

```

```
    array = data;  
}
```

Veľkou výhodou je, že užívateľ môže využívať všetky funkcie zároveň. Teda môžeme vložiť nejaký obrázok a dopísať do neho text, prípadne pomocou myši upraviť niektoré body a následne výsledok uložiť do databázy.

#### 4.3.6. Komunikácia s mikrokontrolérom

Programové vybavenie, ktoré má na starosti komunikáciu po sériovej linke s mikrokontrolérom je veľmi podobná ako v samotnom mikrokontroléri. Mikropočítač Raspberry Pi je master a komunikáciu riadi. Po príkaze na odoslanie dát sa spustí skript, ktorý na začiatku inicializuje sériovú linku a následne sa prihlási do SQL databázy. Z tabuľky canvas vyberie posledné uložené dáta a postupne z nich tvorí pakety, ktoré následne odosiela a kontroluje prichádzajúce odpovede. Pokiaľ sa v prijatej odpovedi nachádza v bajte pre príkaz NAK, čo znamená že mikrokontrolér prijal nesprávne dáta, sa paket odošle znova a inkrementuje sa počítadlo. To sa stane aj v prípade, že nie je správny kontrolný súčet alebo dĺžka paketu. Po desiatich neúspešných pokusoch sa počítadlo vynuluje a skript sa ukončí. Po prijatí kladnej odpovede pokračujeme v odosielaní až pokiaľ nemáme číslo paketu rovné 140. To znamená, že sme odoslali všetky dáta pre vykreslenie a v poslednom pakete odošleme príkazy pre mikrokontrolér. Po úspešnom odoslaní sa znova znuluje počítadlo a skript sa ukončí.

#### 4.3.7. Predvedenie činnosti

Pre predvedenie funkčnosti celého zariadenia sme vytvorili pomocou webovej stránky textový reťazec zadávaním znakov z klávesnice, ktorý sme uložili a následne vykreslili na displej. Vytvorený reťazec je na obrázku Obr. 36 a vykreslený na obrázku Obr. 37.

A screenshot of a text string in a monospace font, displayed in a white box. The text is arranged in two lines: "DIPLOMOVÁ PRACA 2015" on the top line and "MATICOVÝ DISPLEJBS210" on the bottom line. The characters are black on a white background.

Obr. 36 Vytvorený textový reťazec



Obr. 37 Vykreslený textový reťazec

Aby sme ukázali aj funkčnosť meteostanice pomocou rovnakej webovej stránky sme načítali dáta z databázy, ktoré sa následne vložili do poľa pre vytvorenie obrazu. Ten sme znova uložili, odoslali a vykreslili na displej. Výsledok je na nasledujúcom obrázku.



*Obr. 38 Vykreslené dáta z meteostanice*

## 5. ZÁVER

Úlohou práce bolo vytvoriť informačný portál pre zobrazovanie dát a animácii. Pre zobrazovanie máme k dispozícii maticový displej BUSE BS210, ktorý používa technológiu Flip-Flop disk. Jedná sa o otočné disky, ktoré majú jednu stranu inej farby ako druhú. Na každom disku sa nachádza magnet a pomocou prúdových impulzov do cievk pod diskami sa pretáčajú.

V prvej časti práce sme sa zaoberali popisom displeja a technológie, ktorú využíva. V druhej časti sme prebrali možnosti riadenia tohto typu displeja a rozhodli sme sa pre riadenie pomocou mikrokontroléra a dekóderov 1 z n. Výstupy týchto dekóderov sú výkonovo posilnené tranzistorovými poľami. Pre napájanie bol navrhnutý napájací zdroj, ktorý bude aj s ostatnou elektronikou umiestnený vo vnútri zariadenia. Obvod s mikrokontrolérom spolu s logickou časťou tvoria jeden celok a slúžia pre displej ako radič. Pre odosielanie príkazov pre tento radič sme zvolili nadradený systém, mikropočítač Raspberry Pi. Jedná sa o plnohodnotný systém s operačným systémom distribúcie Linux, ktorý zároveň slúži ako webový server.

Pre splnenie jedného z bodov zadania, konkrétne funkcie meteostanice sme implementovali snímače pre meranie teploty, atmosférického tlaku a relatívnej vlhkosti vzduchu. Hodnoty z týchto snímačov sú čítané pomocou skriptu, ktorý beží na pozadí po spustení operačného systému. Tento skript dáta prečíta a následne spolu s časovou značkou uloží do vytvorenej databázy do tabuľky `sensor_data`.

Užívateľ má možnosť si takto zosnímané hodnoty prezrieť na webových stránkach práce vo forme tabuľky alebo grafu vytvoreného pomocou voľne dostupných skriptov v skriptovacom jazyku JavaScript.

Užívateľ zariadenia má k dispozícii niekoľko možností ako vytvoriť obraz, ktorý sa následne má zobrazovať na displeji. Pre tento účel sme vytvorili ďalšiu internetovú stránku s názvom Basic Control BUSE. Po prejdení na túto stránku je možné zadávať text priamo s klávesnice pripojenej k počítaču. Text sa zobrazuje v nami vytvorenom štýle, ktorý obsahuje základné znaky abecedy a čísla. Nevýhodou displeja je jeho malý rozmer, čo nás obmedzilo pri experimentovaní s iným štýlom písma aký sme použili. Pre zobrazenie dát z meteostanice sme vytvorili funkciu, ktorá vytiahne posledné hodnoty teploty, tlaku a vlhkosti z databázy a vykreslí do poľa pre zobrazenie obrazu. Tieto dáta sa zobrazujú v predpísanom formáte, ktorý je možné jednoduchým zásahom do zdrojového kódu upraviť. Ďalšou možnosťou je označovanie jednotlivých pixelov v poli pomocou myši alebo načítanie hotového obrázku odpovedajúcich rozmerov a formátu `.wbmp`. Všetky možnosti sú prístupné zároveň a vzájomne sa prekresľujú. Výsledný obraz sa ukladá do databázy vo formáte BLOB spolu s ďalšími informáciami, ktoré slúžia pre mikrokontrolér. Jedná sa o údaje, ktoré určujú ako sa bude obraz alebo text na displeji pohybovať.

Zadanie práce sme splnili vo všetkých bodoch. Pokračovaním v práci by sa dalo napríklad možnosťou vytvoriť časový plán pre zobrazovanie dát na displeji.

Práca mala pre mňa veľký prínos hlavne v oblasti softvéru, kde som sa prvý krát stretol s tvorbou webových stránok a jazykmi ako sú PHP, HTML a JavaScript. Taktiež bola pre mňa neoceniteľná skúsenosť s jazykom python, pomocou ktorého sme vytvárali skripty a knižnice pre zber dát zo snímačov a obsluhu sériovej linky.

# LITERATÚRA

- [1] BUSE s.r.o. *BUSE* [online]. [cit. 2015-05-05]. Dostupné z: <http://www.buse.cz/>
- [2] Flip-disc display. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2015, 7 April 2015 [cit. 2015-05-05]. Dostupné z: [http://en.wikipedia.org/wiki/Flip-disc\\_display](http://en.wikipedia.org/wiki/Flip-disc_display)
- [3] PILNÝ, J. Informační portál s WWW rozhraním. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 74 s. Vedoucí diplomové práce doc. Ing. Zdeněk Bradáč, Ph.D..
- [4] All about circuits: parallel-in/parallel-out universal shift register. [online]. 2015 [cit. 2015-05-05]. Dostupné z: [:www.allaboutcircuits.com](http://www.allaboutcircuits.com)
- [5] *74HCT238: Datasheet* [online]. 2007[cit. 2015-01-06]. Dostupné z: <http://www.nxp.com/documents>
- [6] Raspberry Pi. *Raspberry Pi* [online]. 2015 [cit. 2015-05-05]. Dostupné z: <https://www.raspberrypi.org>
- [7] *Monolitické stabilizátory napětí* [online]. 2013[cit. 2015-01-06]. Dostupné z: [-moravec.net/elektronika/integrované-obvody](http://moravec.net/elektronika/integrované-obvody)
- [8] KREJČÍŘÍK, Alexandr. *DC/DC měniče*. 1. vyd. Praha: BEN – technická literatura, 2001, 111 s. ISBN 80-730-0045-8.
- [9] 8-bit Atmel Microcontroller: AtMega128. *ATMEL* [online]. 2011 [cit. 2015-05-05]. Dostupné z: <http://www.atmel.com/>
- [10] *AVR Dragon: Evaluation board* [online]. 2012[cit. 2015-01-06]. Dostupné z: <http://www.atmel.com/webdoc>
- [11] GÁBIK, M. *MEMS barometr s mikrokontrolérem*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2013. 62 s. Vedoucí bakalářské práce doc. Ing. Zdeněk Bradáč, Ph.D..
- [12] Universal asynchronous receiver/transmitter. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-01-06]. Dostupné z: [http://en.wikipedia.org/wiki/Universal\\_asynchronous\\_receiver/transmitter](http://en.wikipedia.org/wiki/Universal_asynchronous_receiver/transmitter)
- [13] *PCF8563T: Datasheet* [online]. 2008. vyd. [cit. 2015-01-06]. Dostupné z: <http://www.gme.cz/pcf8563t-smd-p959-259>

- [14] *ULN2803: Datasheet* [online]. 1997[cit. 2015-01-06]. Dostupné z: <http://www.ti.com/lit>
- [15] *TD62783: Datasheet* [online]. 1998[cit. 2015-01-06]. Dostupné z: <http://pdf1.alldatasheet.com/datasheet>
- [16] UPTON, Eben. *Raspberry Pi: uživatelská příručka*. 1. vyd. Brno: Computer Press, 2013, 232 s. ISBN 978-80-251-4116-8.
- [17] REGTIEN, Paul, Martin HALAJ a Eva KUREKOVÁ. *Meranie teploty* [online]. [cit. 2015-01-06].
- [18] Low Power Digital Temperature Sensor: TMP102. *Texas Instruments* [online]. 2008 [cit. 2015-05-05]. Dostupné z: [www.sparkfun.com](http://www.sparkfun.com)
- [19] *BMP185: Datasheet* [online]. 2009[cit. 2015-01-06]. Dostupné z: [www.sparkfun.com/datasheets](http://www.sparkfun.com/datasheets)
- [20] REGTIEN, Paul. Meranie vlhkosti: Modul M17. In: *Meranie vlhkosti* [online]. [cit. 2015-05-05]. Dostupné z: <http://www.kam.sjf.stuba.sk/>
- [21] HIH6130: Datasheet. 2015. *Honeywell HumidCon™ Digital* [online]. [cit. 2015-05-11]. Dostupné z: <http://sensing.honeywell.com/>
- [22] [Http://www.raspishop.cz/](http://www.raspishop.cz/): Instalace systému Raspbian na SD kartu. *Raspishop* [online]. 2013 [cit. 2015-05-05]. Dostupné z: [www.adafruit.com](http://www.adafruit.com)
- [23] NORRIS, Donald. *Raspberry Pi: projekty*. 1. vyd. Brno: Computer Press, 2015, 264 s. ISBN 978-80-251-4346-9.
- [24] Online image to wbmp converter: [online-convert.com](http://online-convert.com). *Convert image* [online]. [cit. 2015-05-05]. Dostupné z: <http://image.online-convert.com/>

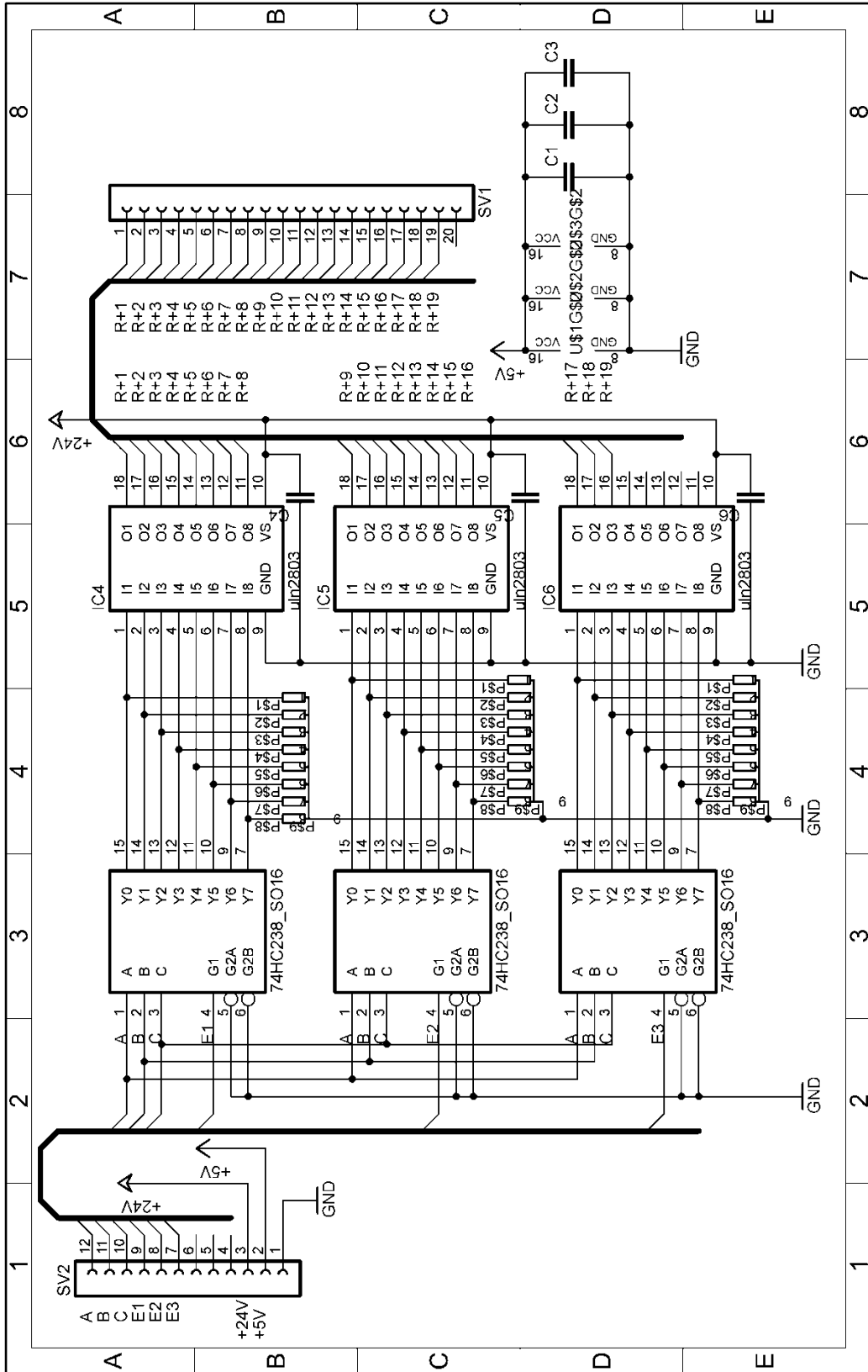
# ZOZNAM PRÍLOH

# ZOZNAM PRÍLOH

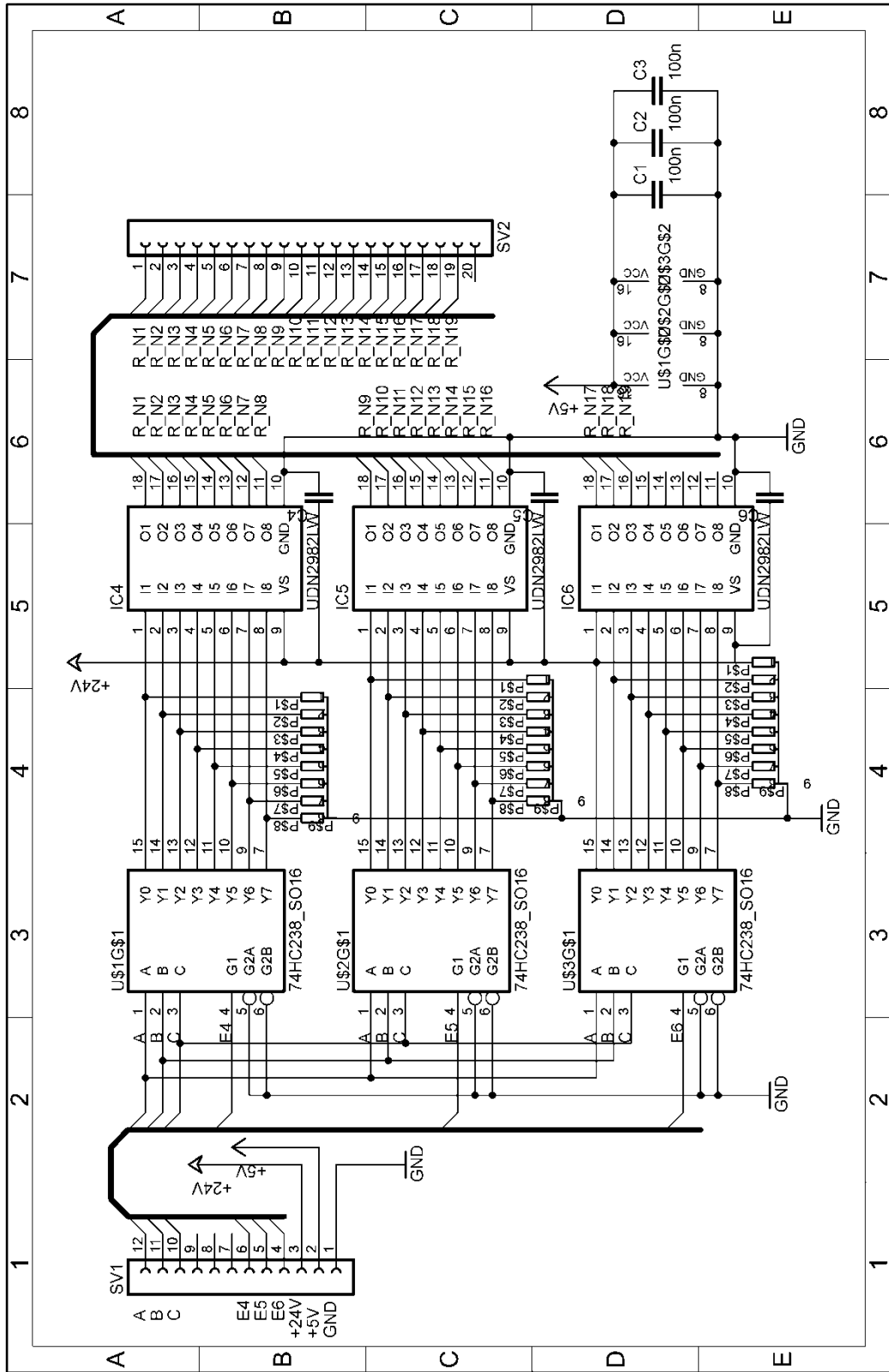
A	schémy zapojenia .....	I
A.1	Obvod s mikrokontrolérom .....	I
A.2	Výber riadku pre kladný smer prúdu .....	II
A.3	Výber riadku pre záporný smer prúdu .....	III
A.4	Napájací zdroj .....	IV
B	dosky plošných spojov .....	V
B.1	Obvod s mikrokontrolérom .....	V
B.2	Obvod s mikrokontrolérom .....	V
B.3	Výber riadku pre kladný smer prúdu .....	VI
B.4	Výber riadku pre záporný smer prúdu .....	VI
B.5	Napájací zdroj (top).....	VII
B.6	Napájací zdroj (bottom) .....	VIII
C	osadzovacie výkresy .....	IX
C.1	Obvod s mikrokontrolérom (top) .....	IX
C.2	Obvod s mikrokontrolérom (bottom).....	IX
C.3	Výber riadku pre kladný smer prúdu .....	X
C.4	Výber riadku pre záporný smer prúdu .....	X
C.5	Napájací zdroj .....	XI
D	zoznam súčiastok .....	XII
D.1	Obvod s mikrokontrolérom .....	XII
D.2	Výber riadku pre kladný smer prúdu .....	XIII
D.3	Výber riadku pre záporný smer prúdu .....	XIII
D.4	Napájací zdroj .....	XIII
E	CD.....	XIV
E.1	Text práce (.pdf).....	XIV
E.2	Schémy a dosky plošných spojov (Eagle).....	XIV
E.3	Zdrojové kódy webových stránok (PHP a HTML).....	XIV
E.4	Použité JavaScript (.js).....	XIV
E.5	Skripty v jazyku python (.py).....	XIV
E.6	Programové vybavenie AtMega128 (.atsln) .....	XIV
E.7	Kópia SD karty z RaspberryPi (image).....	XIV



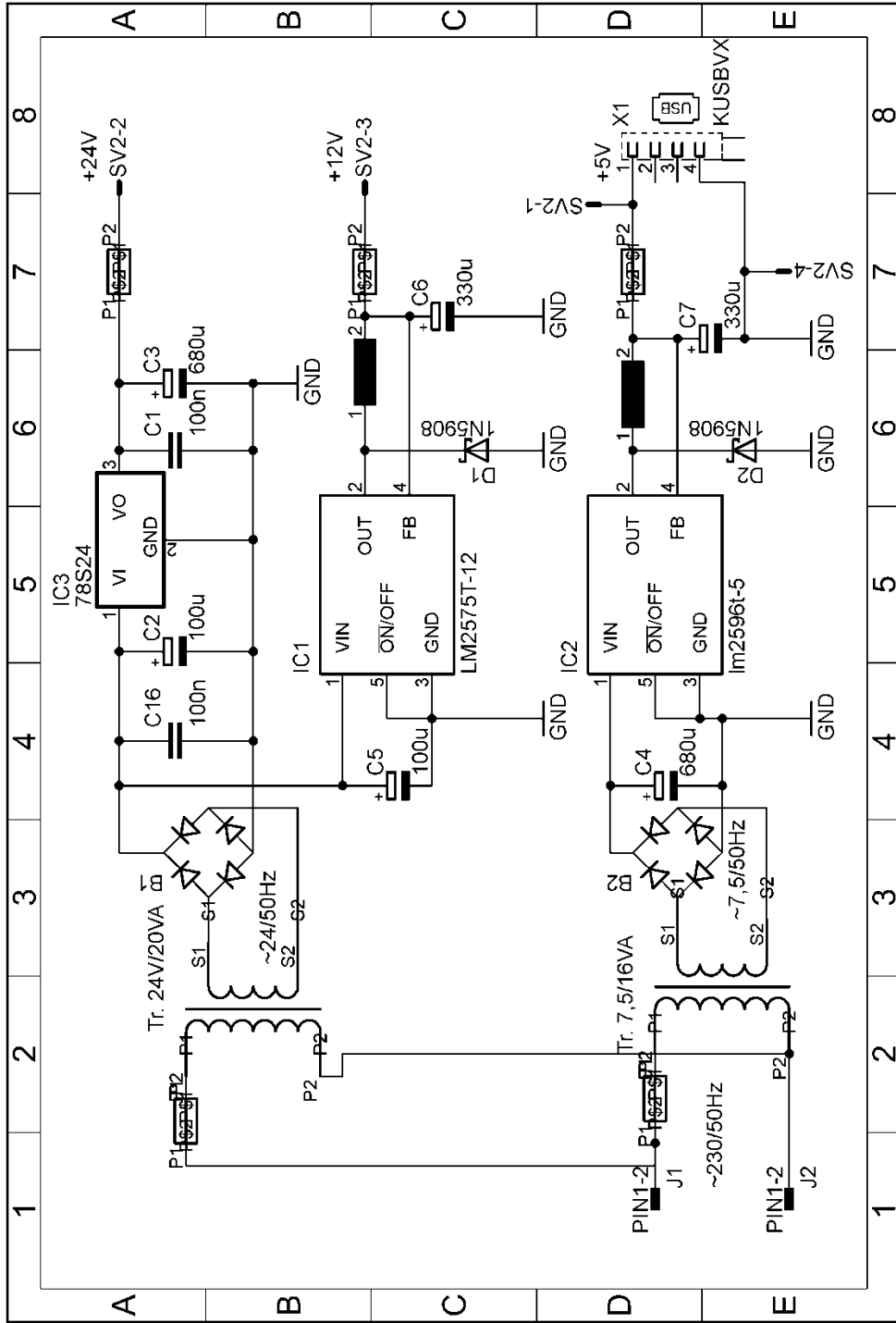
## A.2 Výber riadku pre kladný smer prúdu



### A.3 Výber riadku pre záporný smer prúdu

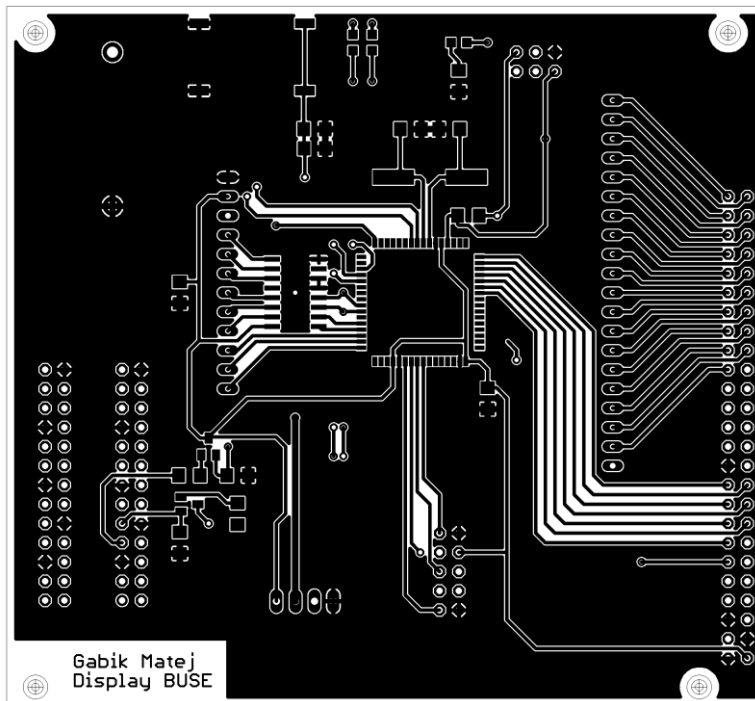


## A.4 Napájecí zdroj

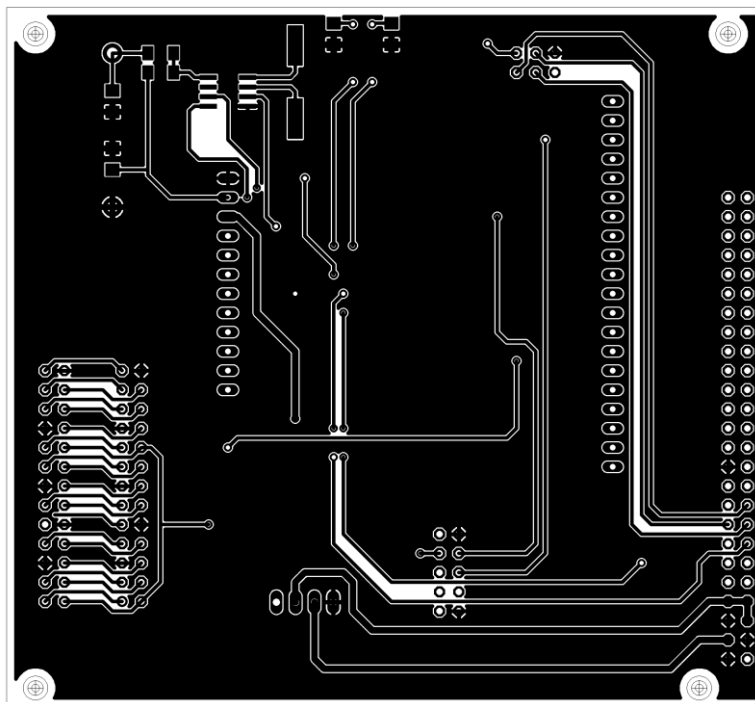


# B DOSKY PLOŠNÝCH SPOJOV

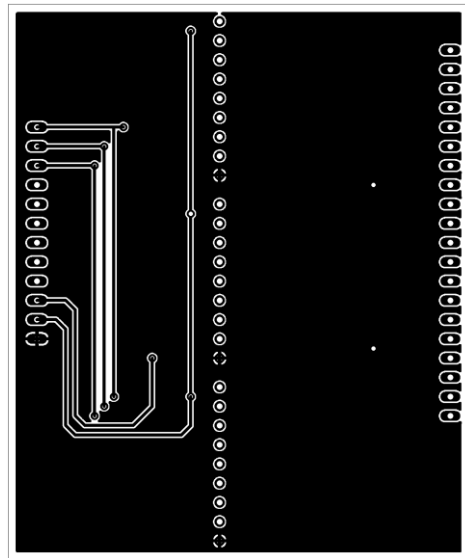
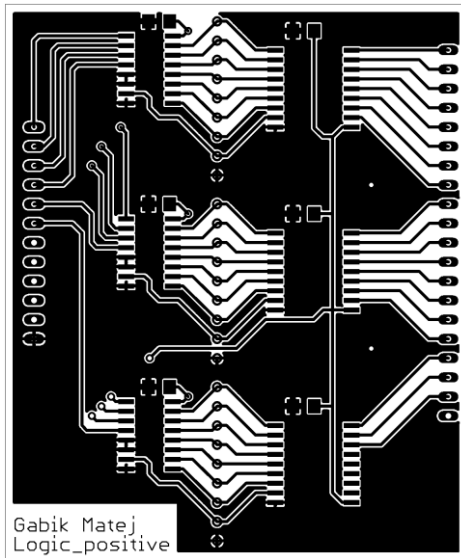
## B.1 Obvod s mikrokontrolérom



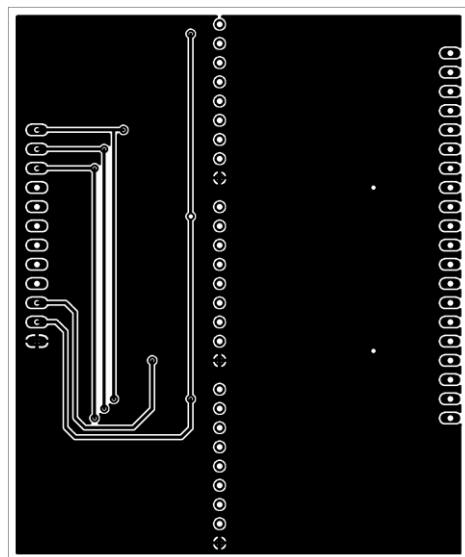
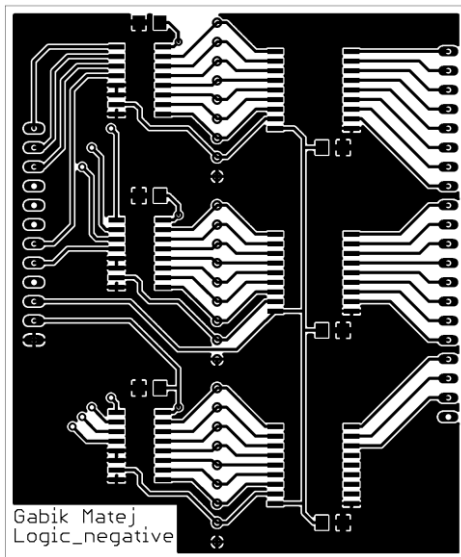
## B.2 Obvod s mikrokontrolérom



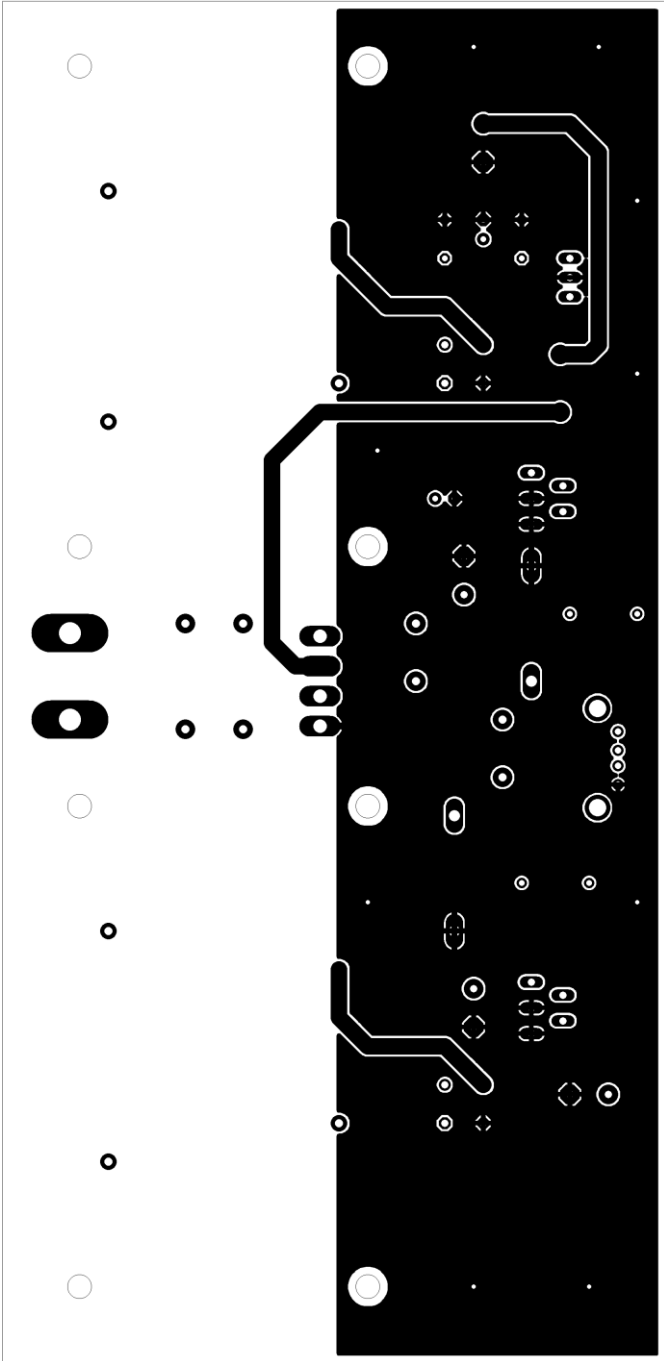
### B.3 Výber riadku pre kladný smer prúdu



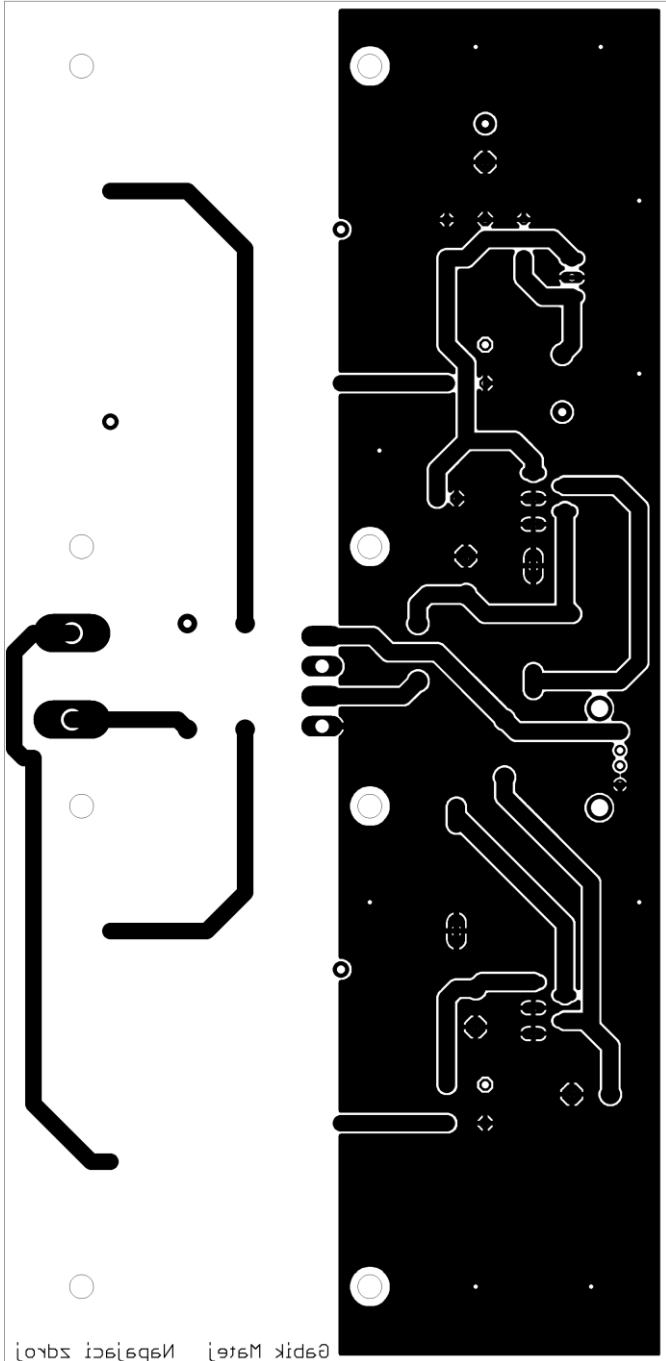
### B.4 Výber riadku pre záporný smer prúdu



**B.5 Napájací zdroj (top)**

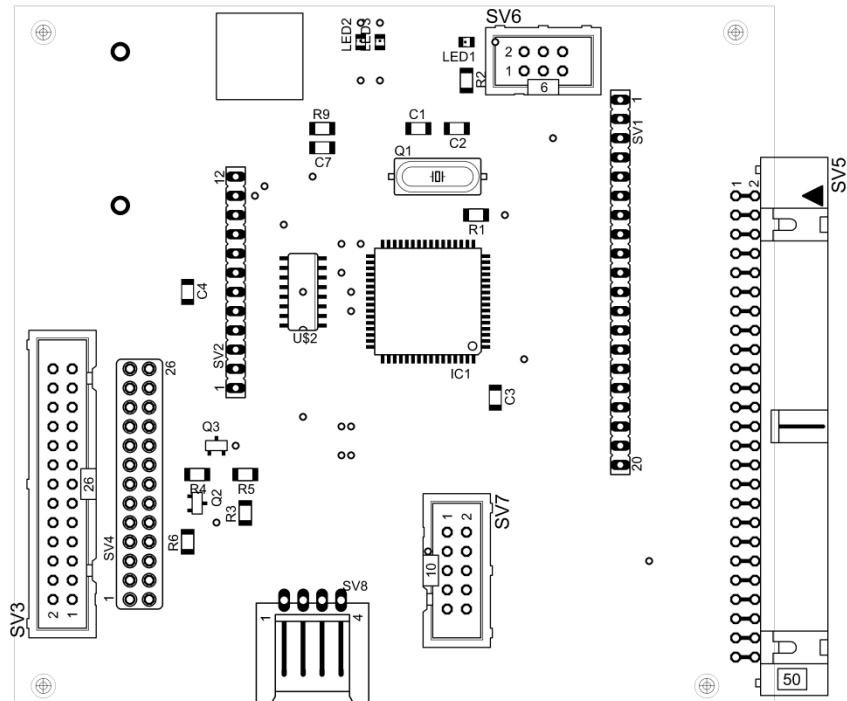


### B.6 Napájací zdroj (bottom)

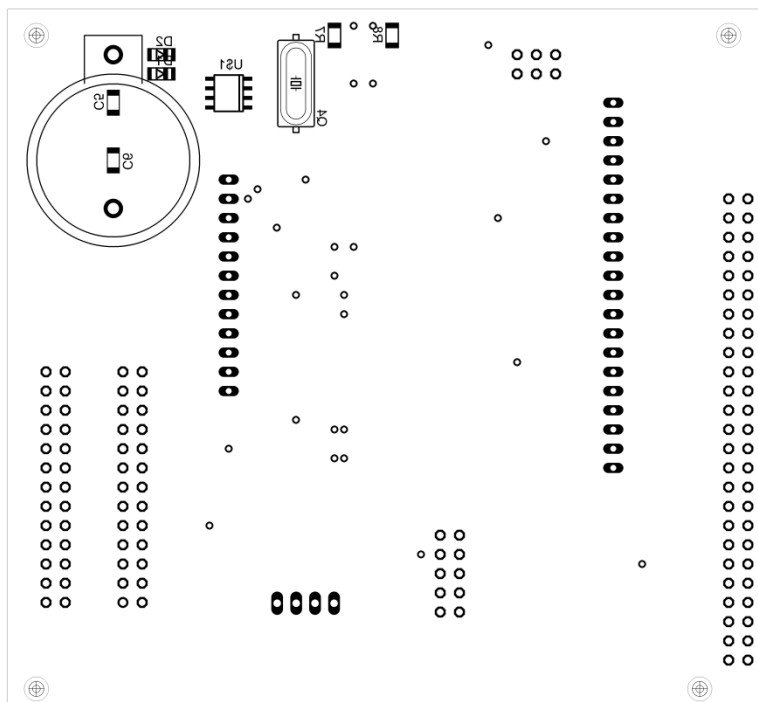


# C OSADZOVACIE VÝKRESY

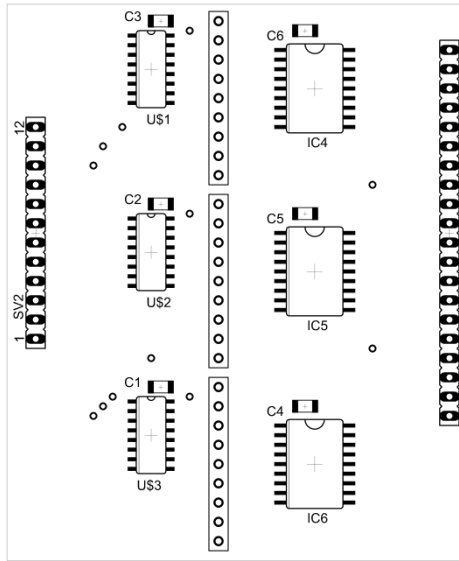
## C.1 Obvod s mikrokontrolérom (top)



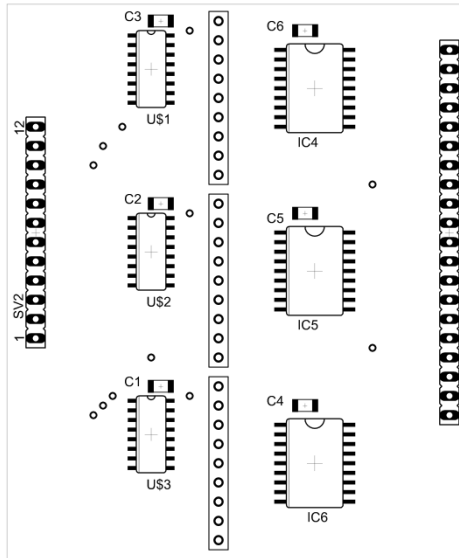
## C.2 Obvod s mikrokontrolérom (bottom)



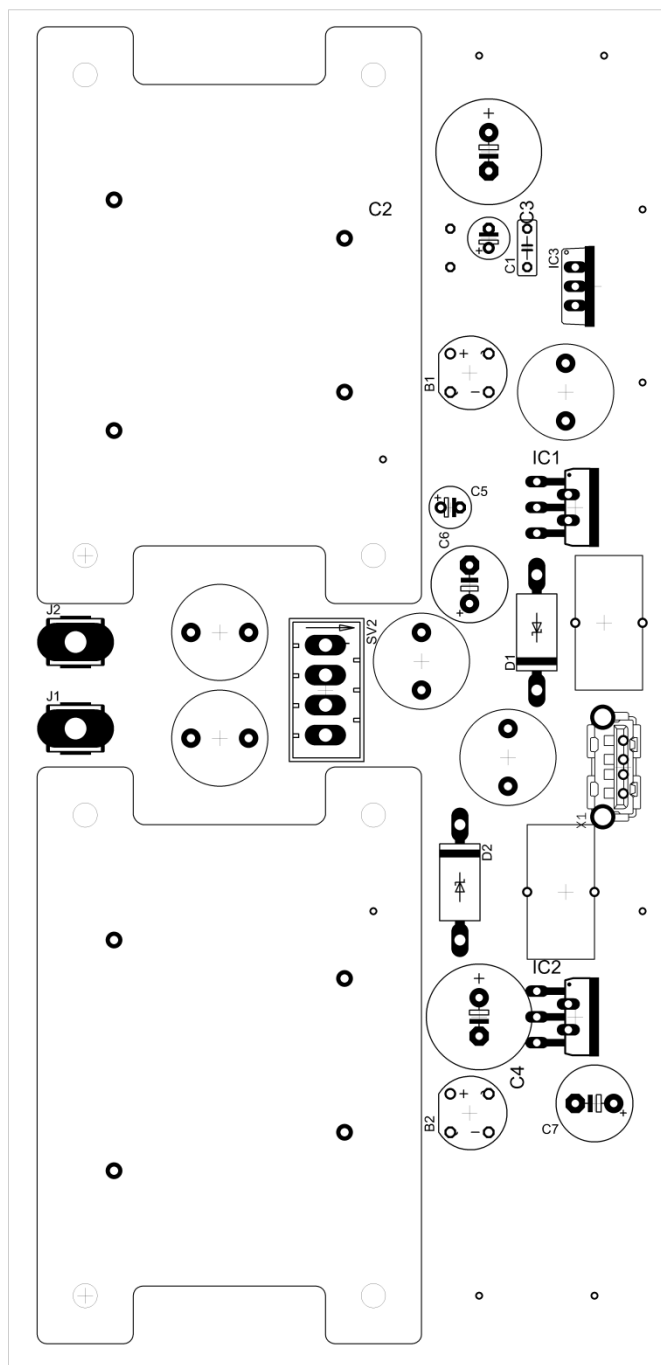
### C.3 Výber riadku pre kladný smer prúdu



### C.4 Výber riadku pre záporný smer prúdu



## C.5 Napájecí zdroj



# D ZOZNAM SÚČIASTOK

## D.1 Obvod s mikrokontrolérom

Qty	Value	Device	Package	Parts
1		MT6-4	MT6-4	SV2
2		RB1A	RB1A	B1, B2
2	100n	C-EU050-025X075	C050-025X075	C1, C16
2	100u	CPOL-EUE2.5-6	E2,5-6	C2, C5
2	1N5908	1N5908	C4111-15	D1, D2
2	330u	CPOL-EUE5-10.5	E5-10,5	C6, C7
2	33uH	INDUCTOR	INDUCTOR	U\$3, U\$4
2	680u	CPOL-EUE5-13	E5-13	C3, C4
1	78S24	7805TV	TO220V	IC3
1	KUSBVX	KUSBVX	KUSBVX	X1
1	LM2575T-12	LM2596T	T05D	IC1
2	PIN1-2	PIN1-2	62221	J1, J2
5	POJISTKA	POJISTKA	POJISTKA	U1, U5, U6, U7, U8
2	TRAFO_A	TRAFO_A	TRAFO_A	TRAFO_24V/20VA
1	lm2596t-5	LM2596T	T05D	IC2

## D.2 Výber riadku pre kladný smer prúdu

Qty	Value	Device	Package	Parts
6		C-EUC1206	C1206	C1, C2, C3, C4, C5, C6
1		FE12-1	FE12	SV2
1		FE20-1	FE20	SV1
3	74HC238	74HC238_SO16	SO16	U\$1, U\$2, U\$3
3	RR	RR	RR	U\$4, U\$5, U\$6
3	uIn2803	UDN2982LW	SO18W	IC4, IC5, IC6

## D.3 Výber riadku pre záporný smer prúdu

Qty	Value	Device	Package	Parts
3		C-EUC1206	C1206	C4, C5, C6
1		FE12-1	FE12	SV1
1		FE20-1	FE20	SV2
3	100n	C-EUC1206	C1206	C1, C2, C3
3	74HC238	74HC238_SO16	SO16	U\$1, U\$2, U\$3
3	RR	RR	RR	U\$4, U\$5, U\$6
3	UDN2982LW	UDN2982LW	SO18W	IC4, IC5, IC6

## D.4 Napájací zdroj

Qty	Value	Device	Package	Parts
1		MT6-4	MT6-4	SV2
2		RB1A	RB1A	B1, B2
2	100n	C-EU050-025X075	C050-025X075	C1, C16
2	100u	CPOL-EUE2.5-6	E2,5-6	C2, C5
2	1N5908	1N5908	C4111-15	D1, D2
2	330u	CPOL-EUE5-10.5	E5-10,5	C6, C7
2	33uH	INDUCTOR	INDUCTOR	U\$3, U\$4
2	680u	CPOL-EUE5-13	E5-13	C3, C4
1	78S24	7805TV	TO220V	IC3
1	KUSBVX	KUSBVX	KUSBVX	X1
1	LM2575T-12	LM2596T	T05D	IC1
2	PIN1-2	PIN1-2	62221	J1, J2
5	POJISTKA	POJISTKA	POJISTKA	U1, U5, U6, U7, U8
2	TRAFO_A	TRAFO_A	TRAFO_A	TRAFO_24V/20VA
1	lm2596t-5	LM2596T	T05D	IC2

## **E CD**

- E.1 Text práce (.pdf)**
- E.2 Schémy a dosky plošných spojov (Eagle)**
- E.3 Zdrojové kódy webových stránok (PHP a HTML)**
- E.4 Použité JavaScript (.js)**
- E.5 Skripty v jazyku python (.py)**
- E.6 Programové vybavenie AtMega128 (.atsln)**
- E.7 Kópia SD karty z RaspberryPi (image)**