ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

FAKULTA
STROJNÍ

BAKALÁŘSKÁ PRÁCE

2019

PAVEL
ABRAHAM

České vysoké učení technické v Praze

Fakulta strojní

Ústav přístrojové a řídící techniky

Obor: Informační a automatizační technika

Systém pro řízené vybíjení elektrochemických článků

BAKALÁŘKSKÁ PRÁCE

Vypracoval: Pavel Abraham

Vedoucí práce: prof. Ing. Jaroslav Novák, CSc.

Rok: 2019

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Abraham** Jméno: **Pavel** Osobní číslo: **434233**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídicí techniky**

Studijní program: **Strojírenství**

Studijní obor: **Informační a automatizační technika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Systém pro řízené vybíjení elektrochemických článků

Název bakalářské práce anglicky:

System for controlled discharging of electrochemical cells

Pokyny pro vypracování:

1. Navrhněte blokovou strukturu zařízení a specifikujte hlavní funkčnosti
2. Navrhněte a realizujte obvodové řešení výkonové a řídicí části zařízení s budoucí možností rozšíření na testování Li článků při nabíjení a vybíjení
3. Navrhněte a realizujte SW vybavení
4. Ověřte funkčnost zařízení při vybíjení suchých elektrochemických článků

Seznam doporučené literatury:

Cenek, M.: Akumulátory od principu k praxi, FCC Public 2003, Praha, ISBN 80-86534-03-0
Chyský, J., Novák, J., Novák, L.: Elektronické aplikace ve strojírenství, skriptum ČVUT v Praze, FS, Praha 1998, ISBN 80-01-01744-3

Jméno a pracoviště vedoucí(ho) bakalářské práce:

prof. Ing. Jaroslav Novák, CSc., odbor elektrotechniky FS

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **26.04.2019** Termín odevzdání bakalářské práce: **12.06.2019**

Platnost zadání bakalářské práce: _____

prof. Ing. Jaroslav Novák, CSc. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Michael Valášek, DrSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

26-04-2019

Datum převzetí zadání

Podpis studenta

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně s tím, že její výsledky mohou být dále použity podle uvážení vedoucího bakalářské práce jako jejího spoluautora. Souhlasím také s případnou publikací výsledků diplomové práce nebo její podstatné části, pokud budu uveden jako její spoluautor.

V Praze dne: ……

……
Pavel Abraham

Poděkování

Děkuji prof. Ing. Jaroslavu Novákovi, CSc. za připomínky při vedení této bakalářské práce a současně i rodině za podporu během celého studia.

Název práce:	Systém pro řízené vybíjení elektrochemických článků
Autor:	Abraham Pavel
Obor:	Informační a automatizační technika
Druh práce:	Bakalářská práce
Vedoucí práce:	prof. Ing. Jaroslav Novák, CSc. Ústav přístrojové a řídící techniky Fakulty strojní České vysoké učení technické v Praze
Abstrakt:	Tato bakalářská práce se zabývá návrhem a realizací zařízení pro řízené vybíjení článků. Ověřuje koncept výkonové části při vybíjení článků velikosti AA. Naměřené hodnoty se ukládají na SD kartu. K ovládání slouží webové rozhraní, které zobrazuje naměřené hodnoty a vykresluje je do grafu.
Klíčová slova:	Arduino, zařízení, vybíjení, elektrochemický článek, vybíjecí křivka, kapacita, napětí, zátěž.

Title:	System for controlled discharging of electrochemical cells
Author:	Abraham Pavel
Abstract:	This bachelor thesis deals with designing and realization of device for controlled discharging of cells. Verifies the power section concept when discharging AA size cells. The measured values are stored on SD card. It is controlled by a web interface that displays the measured values and plots them in a graph.
Keywords:	Arduino, device, discharging, electrochemical cell, discharging curve, capacity, voltage, load.

Obsah

Seznam zkratek

I	Current (Proud)
U	Voltage (Napětí)
R	Resistance (Odpor)
C	Capacity (Kapacita)
F	Frequency (Frekvence)
A	Attenuation (Zeslabení)
GND	Ground (Uzemění)
PWM	Pulse Width Modulation (Pulsně šířková modulace)
SD	Secure Digital card (SD karta)
HTML	Hypertext Markup Lanuage
HTTP	Hypertext Transfer Protocol
CSS	Cascading Style Sheets
JS	JavaScript
SVG	Scalable Vector Graphics

Úvod

Zvláště u některých typů dobíjecích článků je nutné dbát na skladovací napětí. Většina nabíječek neumí vybíjet, případně jen malým vybíjecím proudem. To u velkých baterií může být zdlouhavé.

Cílem této práce je navrhnout a realizovat zařízení, které bude schopno řízeně vybíjet elektrochemické články. Koncept vybíjecí části spočívá v použití tranzistoru jako proměnlivého odporu a drátového rezistoru. Práce se zabývá ověřením tohoto konceptu na článcích velikosti AA s nominálním napětím 1,5 V a vhodnosti tohoto konceptu k vybíjení více článkových Li-Pol baterií.

Z důvodu možnosti vzniku požáru při vybíjení je potřeba mít zařízení na bezpečném místě. Z důvodu dlouhých vybíjecích časů velkých článků není možné vybíjet pod soustavným dozorem. Z tohoto důvodu je potřeba mít vzdálený přístup k zařízení. Práce se zabývá řešením vzdáleného přístupu pomocí připojení zařízení k ethernetové síti a ovládání pomocí webového rozhraní, které je poskytováno zařízením.

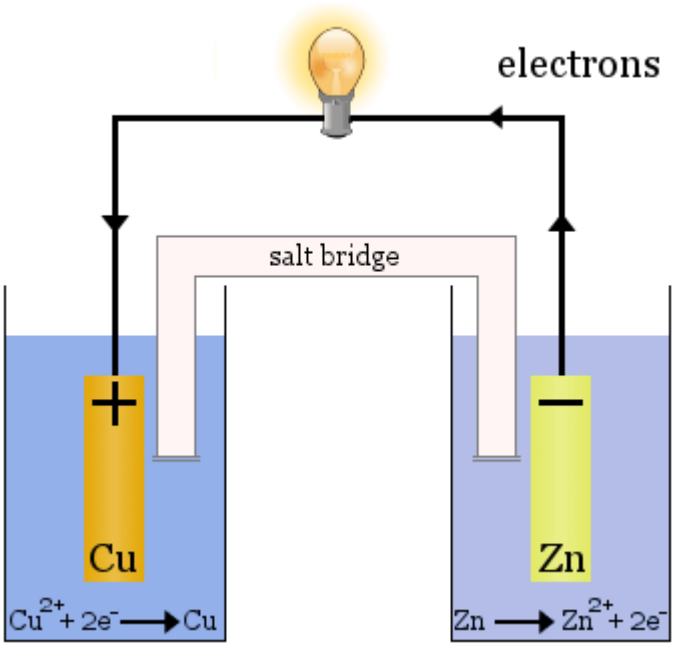
1. Teoretická část

1.1 Elektrochemické články

Elektrochemické články jsou zařízení, která přeměňují chemickou energii v elektrickou. [1]

Tato chemická reakce je rozdělena na dvě reakce na elektrodách. Jedna vypouští elektrony a druhá je pohlcuje. Tento tok elektronů tvoří proud, který může článek dodat. [2]

Základem je galvanický článek, který je tvořen dvěma poločlánky. Poločlánek tvoří elektroda ponořená v elektrolytu. Poločlánky jsou odděleny solným můstkem nebo porézní membránou. [3]



Obr 1 Schéma galvanického článku [4]

Skládá se ze dvou elektrod (kladné a záporné) a elektrolytu. Napětí vzniká z rozdílu potenciálu na elektrodách v důsledku chemických reakcí. Tyto reakce můžou vznikat samovolně nebo jsou vyvolané průchodem elektrického proudu. [3]

1.1.1 Primární články velikosti AA

Jedná se o články na jedno vybití, též nazývány galvanickými články. Obsahují omezené množství reaktantů účastnících se elektrochemických reakcí. Při spotřebování těchto reaktantů články ztrácejí svoji funkčnost. [3]

Zinko-uhlíkový článek

Bývá často označován jako „suchý článek". Katodu tvoří zinkový kalíšek, který současně tvoří vnější obal článku. Anodu tvoří oxid manganičitý v práškové formě, který je ke zlepšení vodivosti smíchán s práškovým uhlíkem. Elektrolyt není v článku v kapalné formě, ale je nasáknut do anody, odtud pochází název „suchý článek". [2][3]

Alkalický článek

Anodu tvoří rozpuštěný zinkový prášek v gelu hydroxidu sodného. Katodou je směs pasty oxidu manganičitého a práškového uhlíku. [1][2]

Lithiový článek

Anodou je lithium a katodu tvoří disulfid železa. Tento článek slouží k nahrazení alkalických článků v případě, že je potřeba vyššího vybíjecího proudu. [1][2]

1.1.2 Sekundární články velikosti AA

Stejně jako články primární obsahují omezené množství reaktantů. Po jejich vybití lze však elektrickým proudem z vnějšího okruhu reaktanty převést zpět na aktivní materiály. Elektrická energie se v článcích akumuluje, odtud tedy název akumulátor. [3]

Nikl-metal hydridový akumulátor

Jedná se nejrozšířenější akumulátor velikosti AA a AAA. Dříve trpěl vysokou hodnotou samovybíjení. Dnes se prodávají články označené jako LSD *(Low Self-Discharge),* často označovány jako *ready to use*. Životnost se pohybuje od 1000 do 3000 nabíjecích cyklů. [2]

Nikl-kadmiový akumulátor

Dnes již zastaralé akumulátory, které byly nahrazeny NiMH akumulátory. Mají nízkou kapacitu, trpí paměťovým efektem a mají vysokou hodnotu samovybíjení. [2]

Nikl-zinkový akumulátor

Nikl-zinkové články mají vyšší nominální napětí a to 1.6 V až 1.65 V. Dokáží dodávat proud dokonce vyšší než NiMH za cenu nižší životnosti. Životnost se pohybuje od 200 do 300 nabíjecích cyklů. [2]

1.1.1 Porovnání článků velikosti AA

Článek	Označení IEC	Označení ANSI/NEDA	Kapacita [mA]	Jmenovité napětí [V]	Maximální vybíjecí proud [A]
Zinko-uhlíkový	R6	15D	800-1100	1,5	0,5
Alkalický	LR6	15A	2000-3000	1,5	1,5
Lithiový	FR6	15LF	3000-3500	1,5 (1,7)	5
NiMH		1.2H2	2000-2500	1,2	5
NiCd	KR6	1.2K2	1000	1,2	3
NiZn	ZR6		1500-1800	1,6	7

Tabulka 1 Porovnání AA článků

1.1.2 Parametry článků

Jmenovité napětí

Jmenovité napětí udává napětí článku při 50 % kapacity. [1]

Kapacita

Představuje maximální hodnotu energie, která může být dodána článkem. Udává se obvykle v Ah, případně v mAh, ale může být také udávána ve Wh. [3]

Míra samovybíjení

Určuje možnou délku skladování článků. [3]

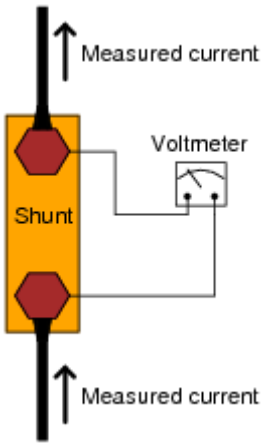
Vnitřní odpor

Vnitřní odpor článku má za následek pokles napětí při zatížení článku. Vypočítává se z rozdílu elektromotorického napětí a napětí při zátěži, poděleného dodávaným proudem. U primárních článků je závislý na teplotě. U akumulátorů na teplotě a počtu dobíjecích cyklů, který akumulátor prodělal. Pohybuje se v řádech desítek až stovek mΩ. [2]

1.2 Metody měření proudu

1.2.1 Bočník

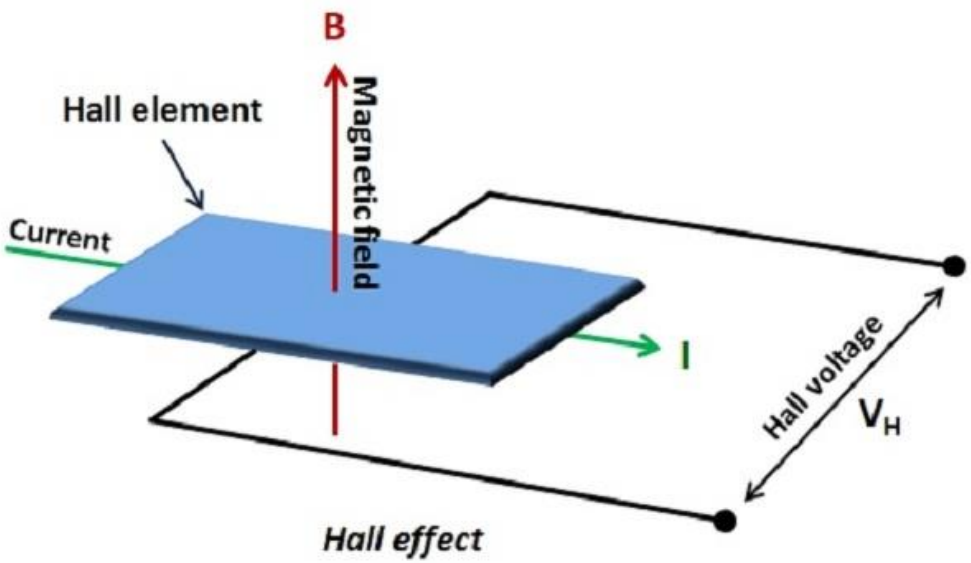
Tato metoda využívá k měření nízko ohmový odpor, který je sériově zapojený se zátěží. Prochází-li obvodem proud, vzniká na tomto odporu úbytek napětí, který je dle Ohmova zákona přímo úměrný velikosti procházejícího proudu. Měří se úbytek napětí na tomto odporu. [5]



Obr 2 Princip měření na bočníku [5]

1.1.1 Senzor proudu využívají Hallova jevu

Jedná se o zařízení, které měří velikost magnetického pole, které generuje procházející proud. [6]



Obr 3 Princip měření využívající Hallova jevu [6]

1.3 Webová stránka

Je typ dokumentu, který je možné pomocí webového prohlížeče zobrazit. [7]

1.3.1 HTML

HTML je zkratka pro Hypertext Markup Language. Jedná se o značkovací jazyk používaný k tvorbě webových stránek. Vytváří dokument, který je na začátku definován tagem <!DOCTYPE html>, obsah je ohraničen tagem <html>. Elementy k zobrazení jsou ohraničeny tagem <body>. Informace o dokumentu jsou ohraničeny tagem <head>. [7]

SVG

SVG je zkratka pro Scalable Vector Graphics. Jedná se o vektorový formát založený na XML. HTML umožnuje vykreslení SVG grafiky. Používá k tomu tag <svg>. [7]

Canvas

Jedná se o JavaScriptové API, pomocí kterého se dá programově kreslit. Používá k tomu tag <canvas>. [7]

1.3.2 CSS

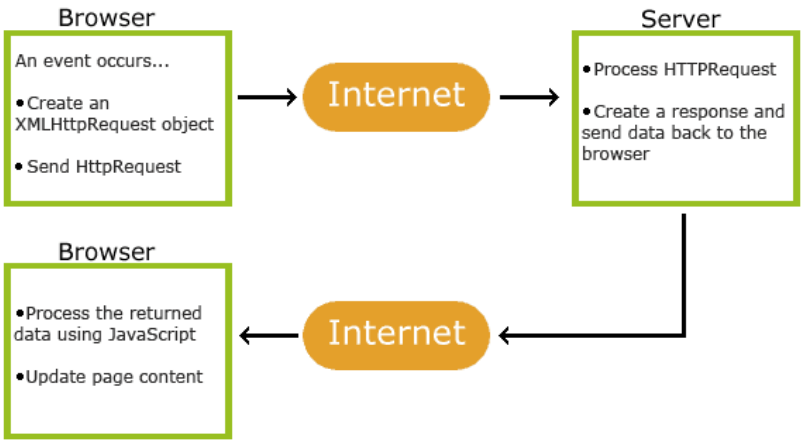
CSS je zkratka pro Cascading Style Sheets. Jedná se o jazyk používaný k popisu způsobu zobrazení elementů ve webových stránkách. [8]

1.3.3 JAVASCRIPT

JavaScript je objektově orientovaný skriptovací jazyk. Často je vkládán přímo do HTML kódu webových stránek. Interpretaci skriptů zajišťuje prohlížeč. [9]

AJAX

AJAX je zkráceně Asynchronous Javascript And XML. Jde o metodu komunikace mezi serverem a prohlížečem. Na základě skriptu je serveru odeslaná žádost typu HTTP Get nebo HTTP Post. Server poté odpoví formou dat ve formátu JSON nebo XML.



Obr 4 Diagram funkce AJAX [10]

1.3.4 Knihovna Bootstrap

Knihovna Bootstrap je CSS Framework, který velice usnadňuje práci s kaskádovými styly. Rozděluje stránku do bloků, díky kterým je možné stránku zobrazovat nezávisle na rozlišení. Stránku je poté možné zobrazovat na jakémkoliv zařízení. [11]

1.3.5 Knihovna jQuery

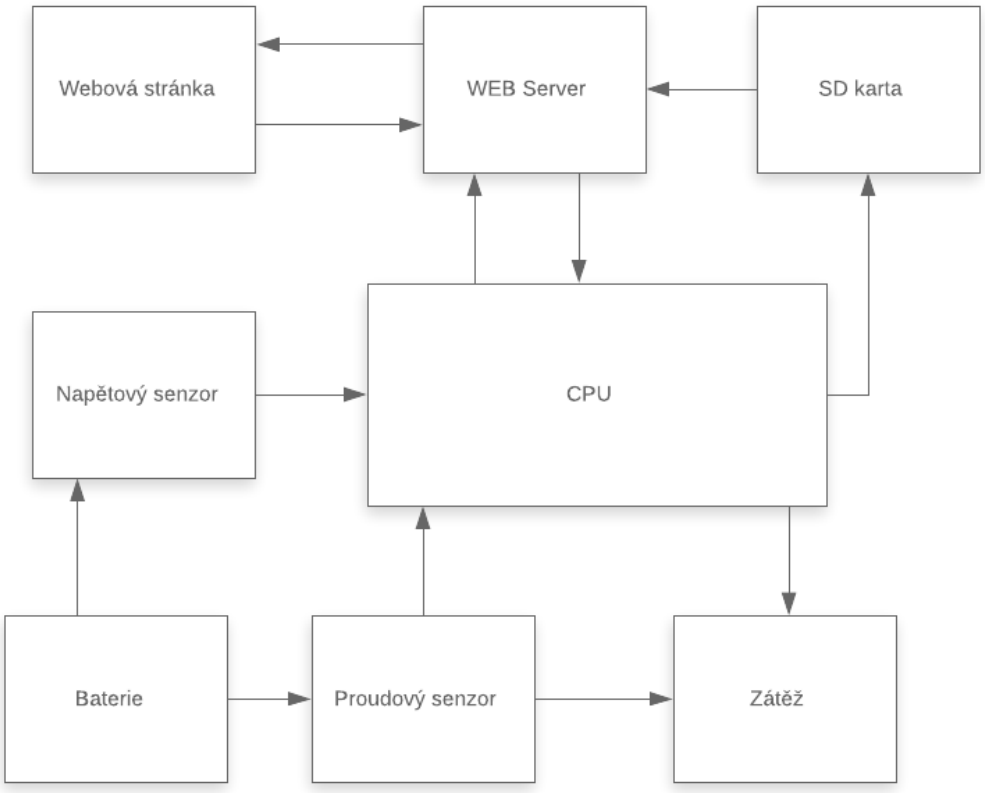
jQuery je JavaScriptová knihovna, se značně zjednodušenou syntaxí. [12]

1.3.6 Vykreslení grafů

K vykreslení grafů slouží JavaScriptové knihovny. Většina knihoven podporuje čistý JavaScript pro vykreslení grafu, některé jsou závislé například na knihovně jQuery. Graf se může vykreslit buď pomocí SVG nebo Canvas. [13]

2. Návrhová část

2.1 Bloková struktura zařízení



Obr 5 Blokové schéma zařízení

2.2 Stanovení funkcí zařízení

Ovládání pomocí webového rozhraní

Měření napětí baterie

Měření proudu

Měření vybité kapacity

Vybíjení baterií nastaveným proudem na nastavené napětí

Ukládání měřených dat na SD kartu

3. Hardwarové řešení

3.1 Řídící část

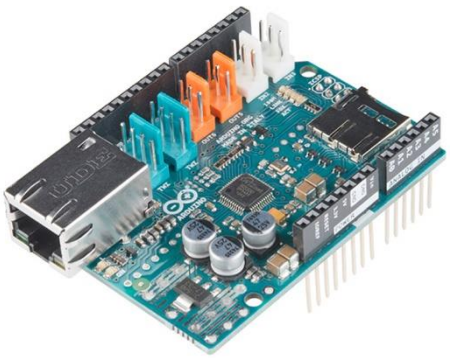
3.1.1 Arduino Due

Z důvodu možnosti rozšíření pro vybíjení Li-Pol článků jsem k řízení vybral vývojovou desku Arduino Due. Při vybíjení Li-Pol baterií je potřeba sledovat napětí jednotlivých článků. Tato deska obsahuje 12bit analogový převodník se 12 vstupy. Dále obsahuje, na rozdíl od většiny ostatních Arduino desek, 12 PWM výstupů s 12bit rozlišením. [14]

Obr 6 Arduino Due [14]

3.1.2 Arduino Ethernet Shield

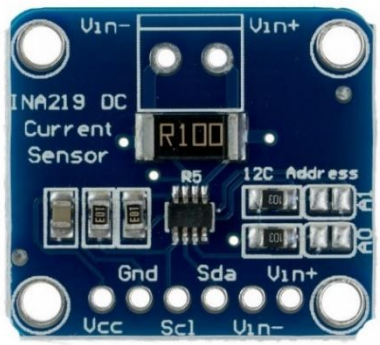
Ke komunikaci po Ethernetu jsem zvolil Arduino Ethernet Shield, který je kompatibilní s Arduinem Due. Rozšiřující modul zároveň obsahuje slot na MicroSD kartu, takže slouží i k ukládání dat. Umožňuje rozšíření o PoE (Power over Ethernet) modul a tím napájet celé zařízení. [15]



Obr 7 Arduino Ethernet Shield [15]

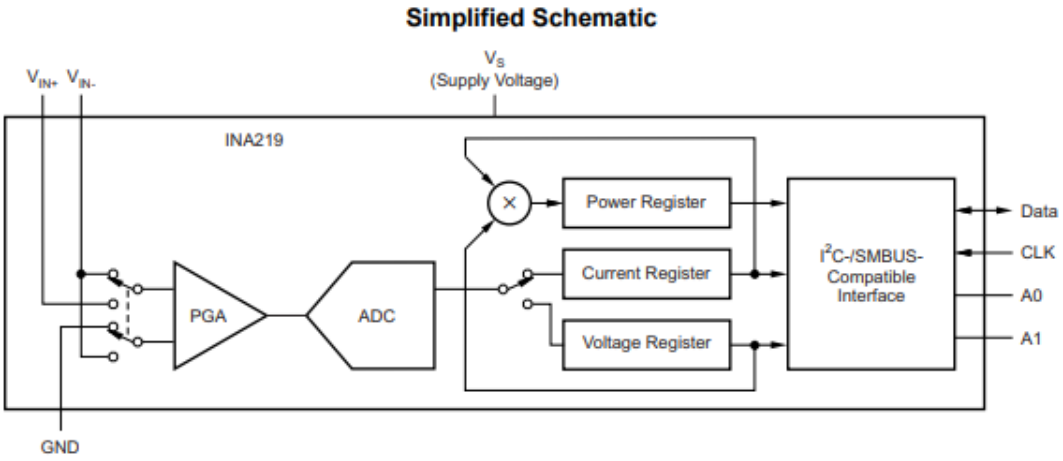
3.1.3 Senzor proudu

Jako senzor proudu jsem zvolil hotový modul, využívající senzor INA219.Senzor měří úbytek napětí na odporu R100. Modul potřebuje k chodu napájecí napětí 3 V až 5,5 V.



Obr 8 Senzor proudu INA219 [16]

Výhodou senzoru je, že měří i napětí.



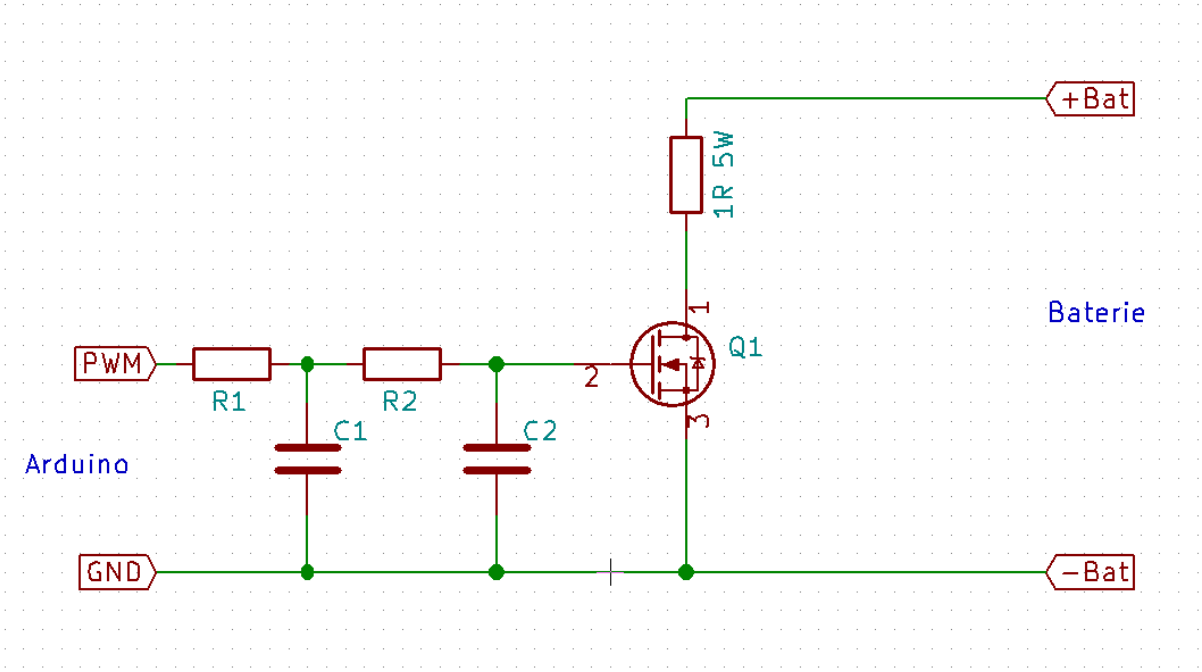
Obr 9 Zjednodušené schéma senzoru [17]

S Arduinem komunikuje pomocí sběrnice I^2C. Arduino IDE obsahuje knihovnu pro tento modul.

Výrobce udává přesnost měření proudu a napětí ±0.5%. [16]

3.2 Výkonová část

Cílem výkonové části je řízeně mařit energii z baterie. K tomu slouží kombinace drátového odporu a tranzistoru. Obvod je dimenzován na jmenovité napětí baterie 1,5V a maximální vybíjecí proud při plně nabité baterii 1 A.



Obr 10 Návrhové schéma výkonové části

Drátový odpor byl zvolen 1R 5W v keramice. Při použití odporu 0,47Ω došlo sice k navýšení maximálního vybíjecího proudu, ale za cenu snížení přesnosti regulace, a to konkrétně na ±25mA. Při použití odporu 1Ω došlo k zpřesnění regulace, a to na ±10mA, a současně k převedení větší části zátěže na odpor. Z tohoto důvodu tranzistor nepotřebuje chladič.

3.2.1 Tranzistor

Tranzistor jsem zvolil IRLZ44NPb. Jedná se o MOSFET s kanálem typu N.

	Parameter	Min.	Typ.	Max.	Units	Conditions
$V_{(BR)DSS}$	Drain-to-Source Breakdown Voltage	55	—	—	V	V_{GS} = 0V, I_D = 250µA
$\Delta V_{(BR)DSS}/\Delta T_J$	Breakdown Voltage Temp. Coefficient	—	0.070	—	V/°C	Reference to 25°C, I_D = 1mA
$R_{DS(on)}$	Static Drain-to-Source On-Resistance	—	—	0.022	Ω	V_{GS} = 10V, I_D = 25A ④
		—	—	0.025		V_{GS} = 5.0V, I_D = 25A ④
		—	—	0.035		V_{GS} = 4.0V, I_D = 21A ④
$V_{GS(th)}$	Gate Threshold Voltage	1.0	—	2.0	V	V_{DS} = V_{GS}, I_D = 250µA
g_{fs}	Forward Transconductance	21	—	—	S	V_{DS} = 25V, I_D = 25A
I_{DSS}	Drain-to-Source Leakage Current	—	—	25	µA	V_{DS} = 55V, V_{GS} = 0V
		—	—	250		V_{DS} = 44V, V_{GS} = 0V, T_J = 150°C
I_{GSS}	Gate-to-Source Forward Leakage	—	—	100	nA	V_{GS} = 16V
	Gate-to-Source Reverse Leakage	—	—	-100		V_{GS} = -16V
Q_g	Total Gate Charge	—	—	48	nC	I_D = 25A
Q_{gs}	Gate-to-Source Charge	—	—	8.6		V_{DS} = 44V
Q_{gd}	Gate-to-Drain ("Miller") Charge	—	—	25		V_{GS} = 5.0V, See Fig. 6 and 13 ④
$t_{d(on)}$	Turn-On Delay Time	—	11	—	ns	V_{DD} = 28V
t_r	Rise Time	—	84	—		I_D = 25A
$t_{d(off)}$	Turn-Off Delay Time	—	26	—		R_G = 3.4Ω, V_{GS} = 5.0V
t_f	Fall Time		15			R_D = 1.1Ω, See Fig. 10 ④
L_D	Internal Drain Inductance	—	4.5	—	nH	Between lead, 6mm (0.25in.) from package and center of die contact
L_S	Internal Source Inductance	—	7.5	—		
C_{iss}	Input Capacitance	—	1700	—	pF	V_{GS} = 0V
C_{oss}	Output Capacitance	—	400	—		V_{DS} = 25V
C_{rss}	Reverse Transfer Capacitance	—	150	—		f = 1.0MHz, See Fig. 5

Obr 11 Základní parametry tranzistoru IRLZ44NPb [18]

Tranzistor je v obvodu zapojen v zesilovacím režimu a slouží jako proměnný odpor.

3.2.2 Filtr

K funkci tranzistoru je potřeba na něj přivést analogové napětí. Arduino Due sice obsahuje digitálně analogový převodník, ale maximální výstup je 2,8V. K plnému rozsahu 0–3,3V je zapotřebí použít PWM výstup v kombinaci s filtrem typu dolní propust. Filtr funguje jako integrátor a převede PWM signál na analogové napětí.

K výpočtu filtru jsem použil následující rovnice [19] [20] [21]:

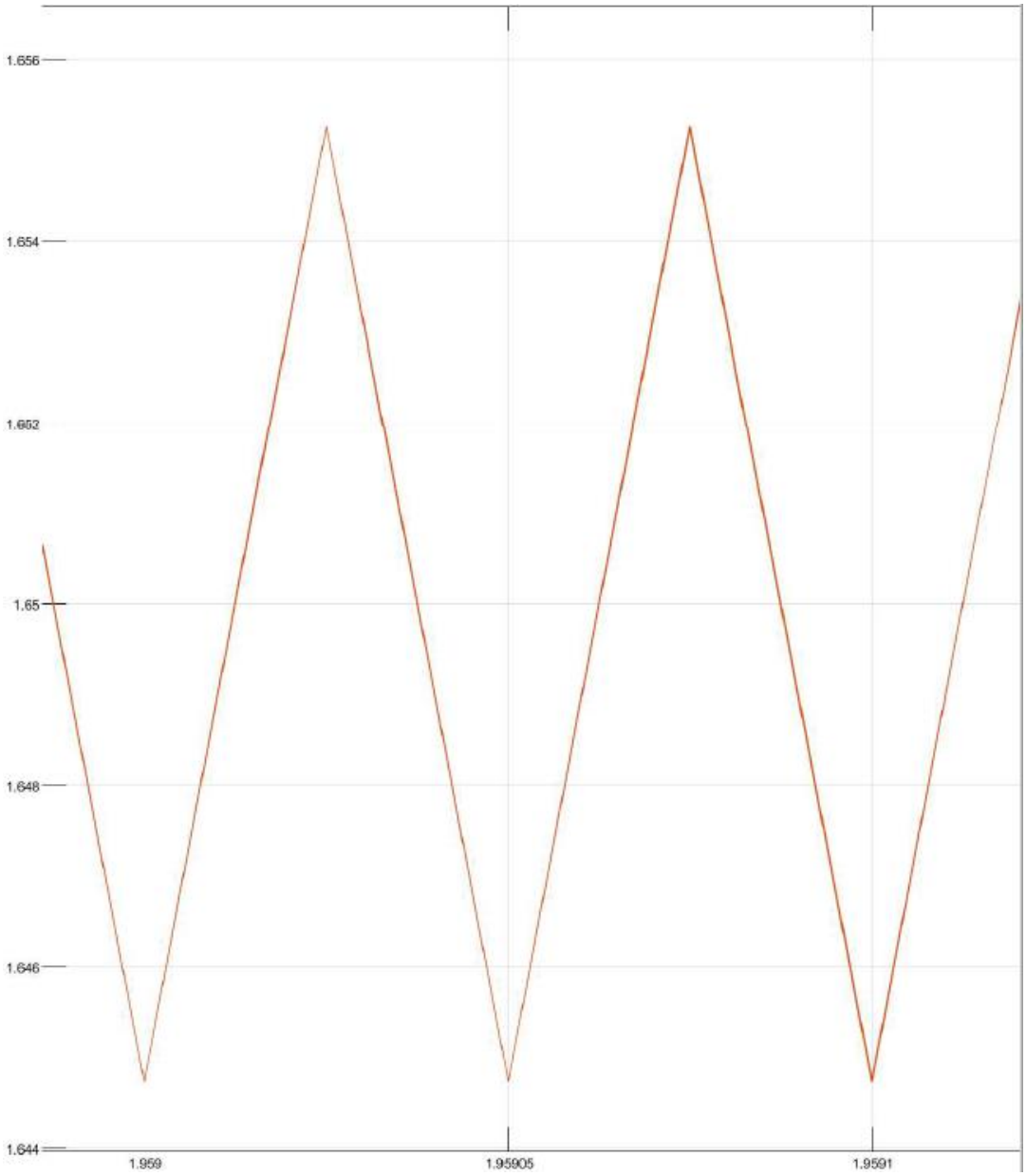
$$A_{dB} = 20 * \log\left(\frac{V_{Ripple}}{V_{PWM}}\right) \tag{3.2.1}$$

$$f_{3dB} = f_{PWM} * 10^{-\frac{A_{dB}}{Slope}} \tag{3.2.2}$$

$$R_F = \frac{1}{2*\pi*C_F*f_{3dB}} \tag{3.2.3}$$

Frekvence PWM je 1000 Hz a napětí PWM je 3.3 V, kapacitu kondenzátoru jsem zvolil 220 nF. Při zvlnění 0,005 V vyšel výsledný odpor na 18585 Ω.

Filtr jsem následně simuloval v programu MATLAB.

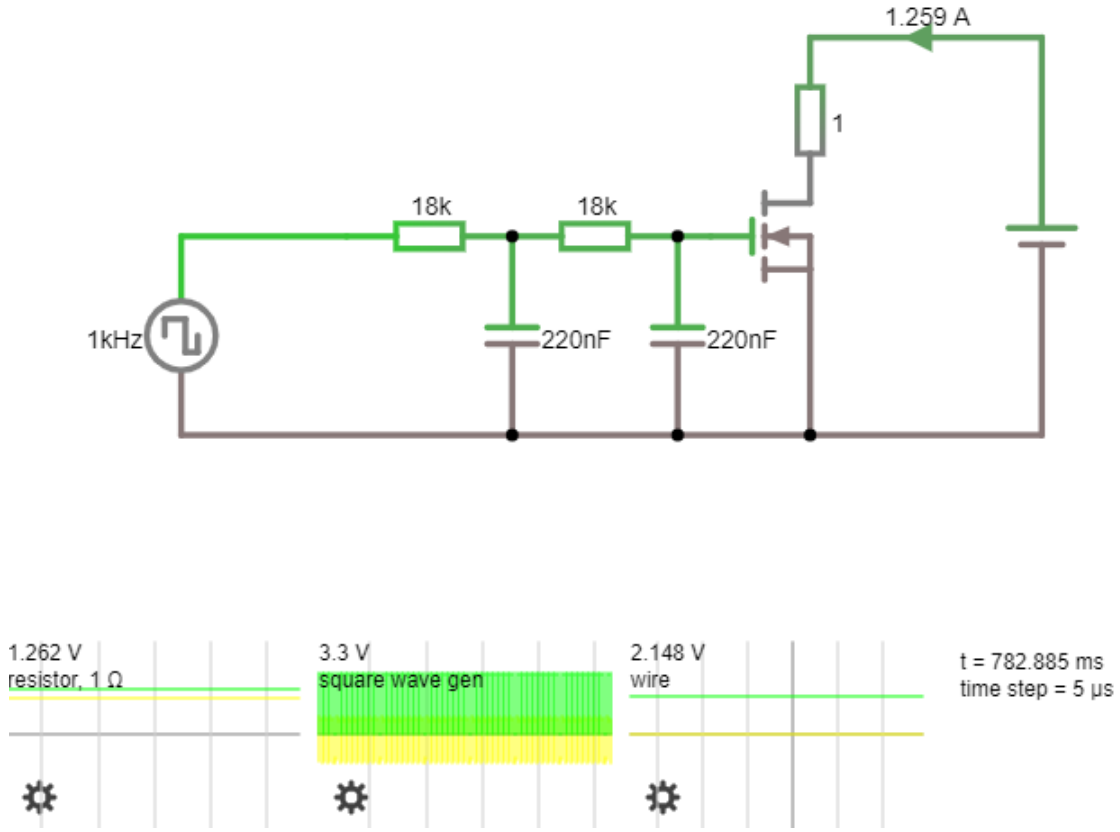


Obr 12 Simulace v programu MATLAB

Zvlnění v simulaci vycházelo stejně jako zvolené při výpočtu.

3.2.3 Simulace obvodu

Vybíjecí část jsem následně simuloval ve webové aplikaci Falstad. [22]



Obr 13 Simulace ve webové aplikaci Falstad [22]

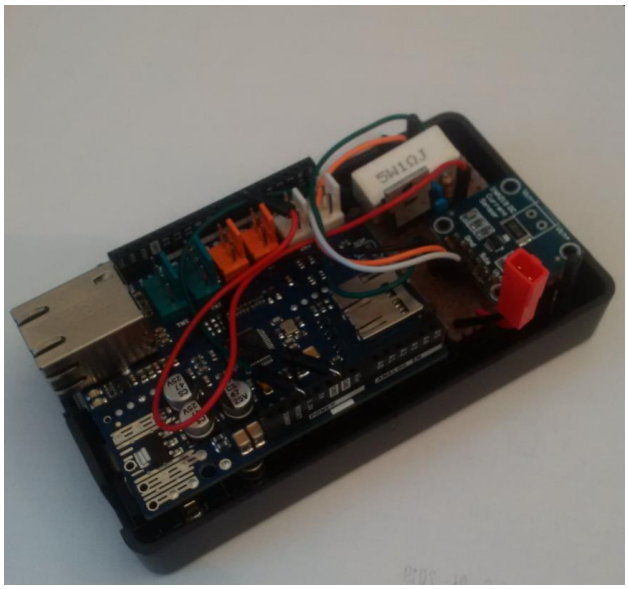
3.2.4 Použité součástky

Součástka	Název	Hodnota	Kusů
R1 a R2	Rezistor 18k	18000Ω	2
R3	Drátový rezistor 1R/5W	1Ω 5W	1
Q1	Tranzistor IRLZ44N	55V 49A	1
C1 a C2	Kondenzátor 220nF	220nF	2

Tabulka 2 Použité součástky

3.3 Realizace

Výkonový obvod jsem zhotovil za pomocí univerzálního plošného spoje tak, aby se vešel do originální Arduino krabičky. Připojení baterií je řešeno JST konektorem.

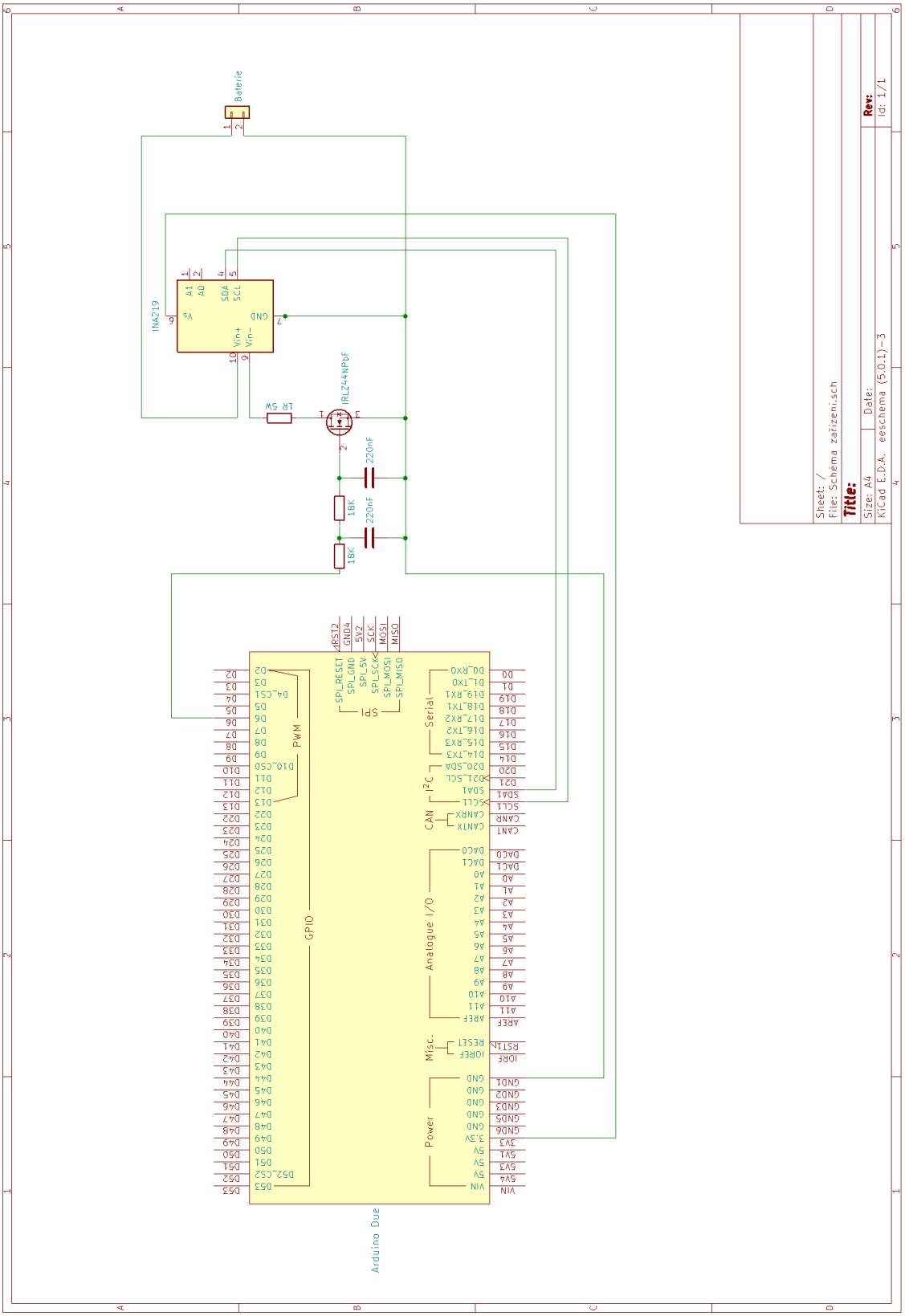


Obr 14 Zapojení řídící a výkonové části

Pro připojení AA baterií jsem zvolil 5 držáků článků, které jsou paralelně spojené a opatřené JST konektorem. Držák baterií je poté přidělán ke krabičce pomocí nalepovacího suchého zipu.

Obr 15 Hotové zařízení

3.4 Schéma zařízení



Obr 16 Schéma zařízení

4. Softwarové řešení

4.1 Arduino

Úkolem programu je regulovat vybíjecí proud, hlídat napětí baterie a ukládat naměřené hodnoty v průběhu vybíjení na SD kartu. Současně Arduino slouží jako webserver, který v případě připojení, pošle webovou stránku uloženou na SD kartě a následně na ní zobrazuje měřené hodnoty a umožňuje nastavení a zahájení vybíjecího cyklu.

4.1.1 Nastavení proměnných

```
1.  double Voltage = 0;
2.  double current = 0;
3.  int setpoint=1000;
4.  double reg = 0;
5.  boolean START = 0;
6.  float cutoffVoltage = 0;
7.  double capacity = 0;
```

Z důvodu nemožnosti použití delay(), kvůli webserveru, veškeré funkce, které se vykonávají v definovaném intervalu, jsou řešeny následovně:

```
1.  if (currentMillis - startMillis >= period)
2.   {
3.          ...
4.        startMillis = currentMillis;
5.   }
```

K tomu jsou přiřazené následující proměnné:

```
1.  unsigned long currentMillis;
2.  unsigned long currentcapMillis;
3.  unsigned long currentsaveMillis;
4.  unsigned long startMillis;
5.  unsigned long startcapMillis;
6.  unsigned long startsaveMillis;
7.  unsigned long capMillis;
8.  const unsigned long period = 10;
9.  const unsigned long periodcap = 100;
10. const unsigned long periodsave = 5000;
```

Poté je ve funkci setup() při zapnutí zařízení nastaven startovní čas.

```
1.  startMillis = millis();
2.  startcapMillis = millis();
3.  startsaveMillis = millis();
4.  capMillis = millis();
```

Jako poslední je potřeba ve funkci loop() nastavit čas od zahájení programu.

```
currentMillis = millis();
currentcapMillis = millis();
currentsaveMillis = millis();
```

4.1.2 Regulace vybíjecího proudu

Prvně je potřeba nastavit daný výstup.

```
int PWMPin = 6;
pinMode(PWMPin, OUTPUT);
```

PWM výstup je v základu nastaven na 8bitů. K nastavení na 12bitů slouží následující příkaz.

```
analogWriteResolution(12);
```

Samotná regulace spočívá ve dvou if() funkcích. První kontroluje, zdali je měřený proud menší než nastavený a pokud ano, zvýší PWM výstup o 1. Druhá snižuje výstup o 1, je-li měřený proud vyšší než nastavený.

```
 if (current()<setpoint &&  START == 1  && reg<4095)
{
    if (currentMillis - startMillis >= period)
     {
      reg++;
      startMillis = currentMillis;
     }

}
else if (current()>setpoint &&  START == 1  && reg>0)
{

 if (currentMillis - startMillis >= period)
     {
       reg=reg-1;
      startMillis = currentMillis;
     }

}
```

4.1.3 Měření proudu a napětí

Proudový senzor komunikuje pomoci I^2C, jsou tedy potřeba patřičné knihovny.

```
#include <Wire.h>
#include <Adafruit_INA219.h>
#define ADDR 0x40
```

Následně je potřeba zahájit komunikaci se senzorem.

```
Adafruit_INA219 ina219(ADDR);
ina219.begin();
ina219.setCalibration_32V_2A();
```

Hodnoty proudu a napětí se poté získají pomocí těchto příkazů:

```
current = ina219.getCurrent_mA();
Voltage = ina219.getBusVoltage_V();
```

4.1.4 Výpočet kapacity

Výpočet kapacity probíhá v intervalu, který je definovaný proměnnou periodcap. Jelikož je proud měřen v mAh, stačí přičíst k předchozí hodnotě proud vynásobený časem, který uběhl od posledního intervalu, převedeným na hodiny.

```
if (currentcapMillis - startcapMillis >= periodcap)
{
capacity = capacity + current*(millis()-capMillis)/3600000;
capMillis = millis();
startcapMillis = currentcapMillis;
}
```

4.1.5 Ukládání hodnot na SD kartu

Měřené hodnoty se ukládají na SD kartu ve formátu CSV. Funkce ukládá hodnoty pouze v případě spuštěného cyklu vybíjení, a to v intervalu, který je definovaný proměnnou periodsave.

```
String dataString = String(Voltage) + ", " + String(current) + ", " + String(r
eg) + ", " + String(capacity,0);

if (START==1 && currentsaveMillis - startsaveMillis >= periodsave){
File logFile = SD.open("log.csv", FILE_WRITE);
if (logFile)
{
logFile.println(dataString);
logFile.close();
 }
startsaveMillis = currentsaveMillis;
}
```

4.1.6 Zpracování žádosti posílané webovou stránkou

Pro Ethernet Shield je nejprve potřeba načíst potřebné knihovny.

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

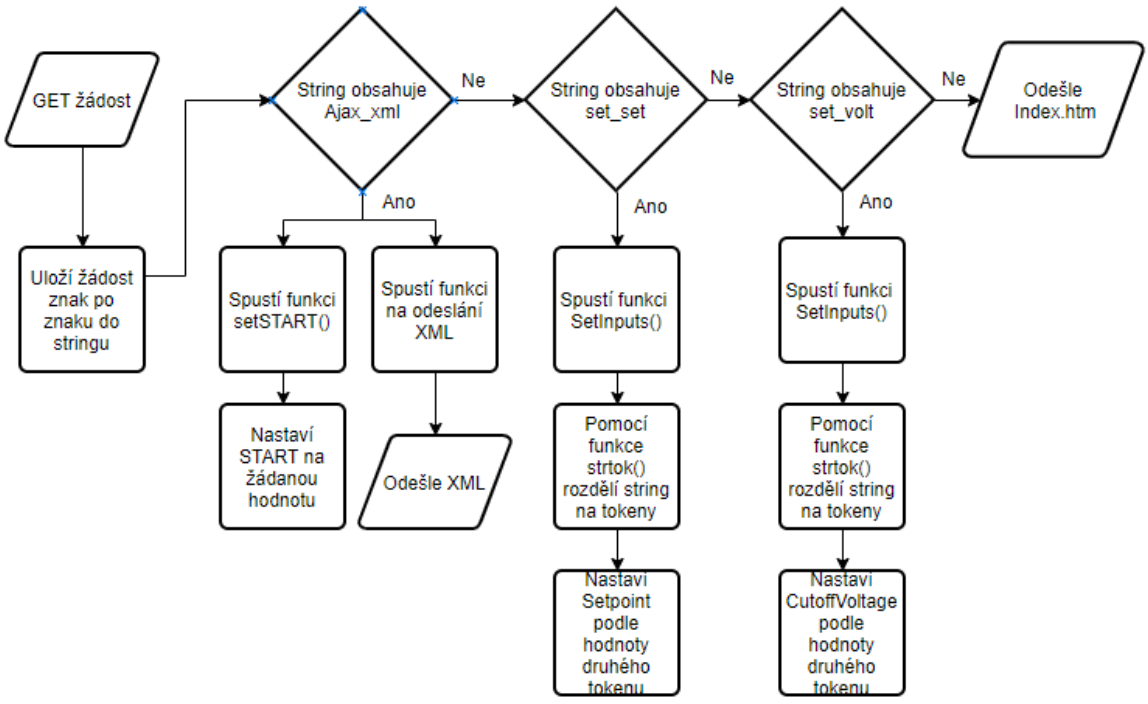
Následně je potřeba nastavit potřebné parametry.

```
byte mac[] = { ... };
IPAddress ip(10, 0, 0, 2);
EthernetServer server(80);
File webFile;
char HTTP_req[REQ_BUF_SZ] = {0};
char req_index = 0;
```

Web server je poté spuštěn následným příkazem.

```
Ethernet.begin(mac, ip);
server.begin();
```

Z důvodu délky funkce zpracovávající HttpRequest je zde zjednodušený vývojový diagram.



Obr 17 Zjednodušený vývojový diagram

Funkce zpracovává GET žádost a následně volá funkce dle žádosti. Kód celé funkce je na další straně.

```
EthernetClient client = server.available();
if (client) {
        boolean currentLineIsBlank = true;
        while (client.connected()) {
            if (client.available()) {
                char c = client.read();
                if (req_index < (REQ_BUF_SZ - 1)) {
                    HTTP_req[req_index] = c;
                    req_index++;
                    }
                if (c == '\n' && currentLineIsBlank) {
                    client.println("HTTP/1.1 200 OK");
                    if (StrContains(HTTP_req, "ajax_xml")) {
                        client.println("Content-Type: text/xml");
                        client.println("Connection: keep-alive");
                        client.println();
                        SetSTART();
                        XML_response(client);
                    }
                    else if (StrContains(HTTP_req, "set_set")) {
                        client.println("Content-Type: text/xml");
                        client.println("Connection: keep-alive");
                        client.println();
                      SetSetpointInput();
                    }
                     else if (StrContains(HTTP_req, "set_volt")) {
                        client.println("Content-Type: text/xml");
                        client.println("Connection: keep-alive");
                        client.println();
                      SetVoltInput();
                    }
                    else {
                        client.println("Content-Type: text/html");
                        client.println("Connection: keep-alive");
                        client.println();
                        webFile = SD.open("index.htm");
                        if (webFile) {
                            while(webFile.available()) {
                                client.write(webFile.read());
                                }
                            webFile.close();
                            }
                        }
                    req_index = 0;
                    StrClear(HTTP_req, REQ_BUF_SZ);
                    break;
                }

                if (c == '\n') {

                    currentLineIsBlank = true;
                    }
                else if (c != '\r') {

                    currentLineIsBlank = false;
                   }
               }
            }
        delay(1);
        client.stop();
    }
```

4.1.7 Nastavení parametrů z webové stránky

V případě, že žádost obsahuje parametry k vybíjení, je zavolána patřičná funkce.

Funkce pomocí strtok() *(string to token)* rozdělí string, obsahující žádost, na tokeny, které jsou odělovány dvojtečkou. Při druhém oddělení obsahuje token žádanou hodnotu. Tato hodnota je uložena do proměnné pomocí atoi() *(array to integer)*, případně atof() *(array to float)*.

```
1.  void SetSetpointInput(void)
2.  {
3.    char * strtokSet;
4.
5.    strtokSet = strtok(HTTP_req,":");
6.
7.    strtokSet = strtok(NULL, ":");
8.    setpoint = atoi(strtokSet);
9.  }
10.
11. void SetVoltInput(void)
12. {
13. char * strtokVolt;
14.
15.   strtokVolt = strtok(HTTP_req,":");
16.
17.   strtokVolt = strtok(NULL, ":");
18.   cutoffVoltage = atof(strtokVolt);
19. }
```

4.1.8 Tlačítko START

V případě, že žádost obsahuje informaci o tlačítku, je tato informace vyhodnocena a je podle toho nastavená proměnná START.

```
1.  void SetSTART(void)
2.  {
3.    if (StrContains(HTTP_req, "START=1")) {
4.        START = 1;
5.    }
6.    if (StrContains(HTTP_req, "START=0")) {
7.         START = 0;
8.    }
9.  }
```

O vypnutí cyklu vybíjení se stará podmínka, která kontroluje, že napětí baterie je vyšší než nastavené napětí pro vypnutí. V případě poklesu napětí pod nastavenou se nastaví proměnná START na 0. Dojde k vypnutí vybíjecího cyklu a informace o proměnné START je posláno pomoci XML webové stránce.

```
1.  if (Voltage < cutoffVoltage)
2.  {
3.    START = 0;
4.  }
```

V případě, že je START = 0, se nastaví PWM výstup na 2000, z důvodu, aby se při regulaci nemuselo čekat. Prahového napětí tranzistoru se dosáhne zhruba při výstupu 2120. Současně se nastaví proud na 0.

```
if ( START == 0)
  {
    reg=2000;
    current = 0;

  }
```

4.1.9 Posílaní hodnot v XML

V případě, že žádost obsahuje „ajax_xml", je zavolána funkce, která odešle XML odpověď, kde mezi jednotlivými tagy jsou měřené hodnoty. Současně posílá zpět informaci, je-li START tlačítko zapnuto nebo vypnuto.

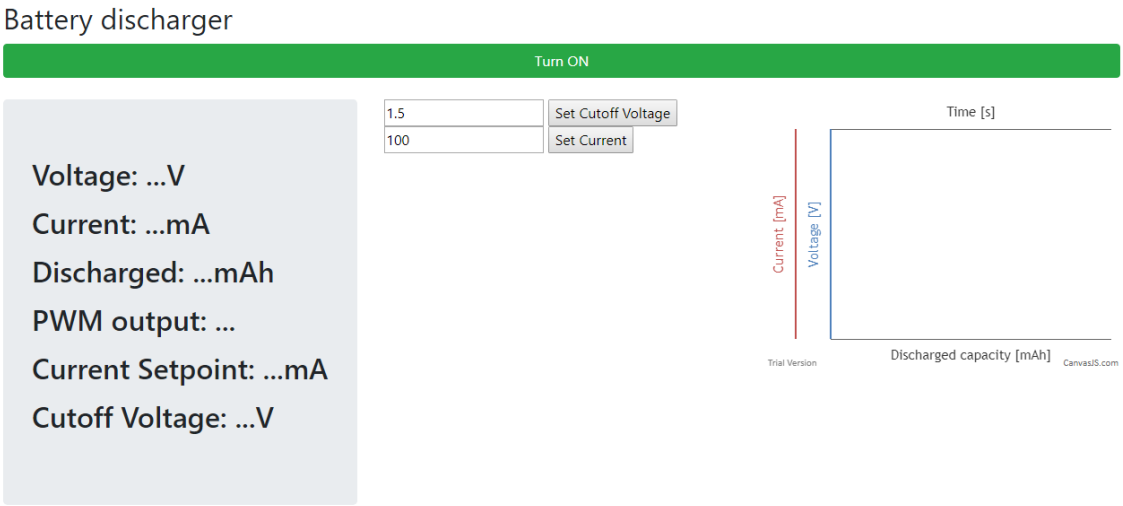
```
void XML_response(EthernetClient cl) {

cl.print("<?xml version = \"1.0\" ?>");
cl.print("<inputs>");

cl.print("<voltageinput>");
cl.print(Voltage);
cl.println("</voltageinput>");

cl.print("<currentinput>");
cl.print(current);
cl.println("</currentinput>");

cl.print("<capacityinput>");
cl.print(capacity);
cl.println("</capacityinput>");

cl.print("<pwminput>");
cl.print(reg);
cl.println("</pwminput>");

cl.print("<setpointinput>");
cl.print(setpoint);
cl.println("</setpointinput>");

cl.print("<cutoffvoltageinput>");
cl.print(cutoffVoltage);
cl.println("</cutoffvoltageinput>");


cl.print("<START>");
  if (START) {
      cl.print("on");
  }
  else {
      cl.print("off");
  }
cl.println("</START>");

cl.print("</inputs>");
}
```

4.2 Webová stránka

Webová stránka slouží k ovládání zařízení. Skládá se ze 4 částí, jejichž řešení je dále popsáno.

Stránka je optimalizována pro zobrazení na mobilním zařízení pomocí knihovny Bootstrap. Jednotlivé části jsou v blocích, které se na mobilním zobrazení zarovnají pod sebe.

Samotná stránka je psaná pomocí HTML a čistého Javascriptu. Knihovna jQuery, která je potřeba pro funkci knihovny Bootstrap, není jinak využita. Graf je řešen knihovnou CanvasJS.



Obr 18 Webové rozhraní

Na stránce je spouštěcí tlačítko START. Pod ním jsou v řadě bloky, kde první blok zobrazuje hodnoty, které jsou posílány z Arduina. V druhém bloku jsou textová pole pro nastavení parametrů a ve třetím bloku je graf, který vykresluje průběh proudu v čase a závislost napětí na vybité kapacitě.

4.2.1 Tlačítko START

START tlačítko na webové stránce slouží k zapnutí či vypnutí funkce vybíjení.

Po stisknutí tlačítka se zavolá funkce GetSTART()

```
<button type="button" class="btn btn-success btn-
block" id="START" onclick="GetSTART()">Turn ON</button>
```

Funkce GetSTART() zkontroluje, zda-li je hodnota tlačítka, kterou vrací Arduino v rámci XML, rovna 1. Pokud ano, nastaví stav tlačítka na 0 a zároveň nastaví proměnou strSTART, která je zpět posílána do Arduina.

```
function GetSTART()
        {
            if (START_state === 1) {
                START_state = 0;
                strSTART = "&START=0";
            }
            else {
                START_state = 1;
                strSTART = "&START=1";
            }
        }
```

K načtení stavu stavu z XML slouží příkaz níže, který je součástí funkce GetArduinoInputs() (dále probráno níže).

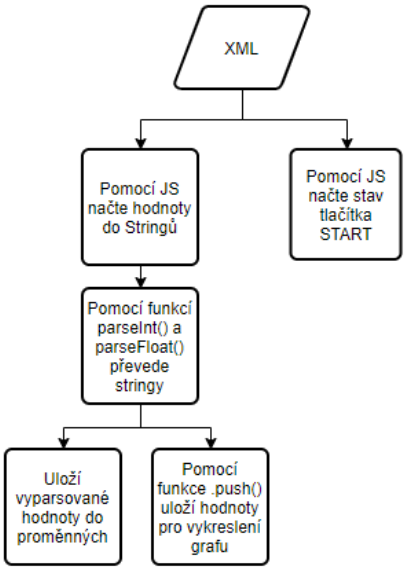
Příkaz kontroluje, zdali je vrácená hodnota z Arduina „on“. V případě, že ano, nastaví stav na 1 a přepíše hodnotu na tlačítku.

V případě, že ne, nastaví stav na 0, přepíše text na tlačítku a současně nastaví proměnou strSTART. Díky tomu je možné vybíjení vypnout, jak ze strany Arduina, tak ze strany uživatele.

```
//Načtení stavu tlačítka
if (this.responseXML.getElementsByTagName('START')[0].childNodes[0].nodeValue
=== "on")
{
document.getElementById("START").innerHTML = "Turn OFF";
    START_state = 1;
}
else
{
document.getElementById("START").innerHTML = "Turn ON";
    START_state = 0;
        strSTART = "&START=0";
}
```

4.2.2 Načtení a zobrazení měřených hodnot z Arduina

K načtení dat slouží funkce GetArduinoInputs(). Z důvodu délky funkce je zde kód rozdělen na dílčí části. Ke snazšímu pochopení je zde zjednodušený vývojový diagram.



Obr 19 Zjednodušený vývojový diagram

Funkce pracuje na principu AJAXu, kdy pošle na webserver žádost typu HttpRequest, která obsahuje text „ajax_xml“, webserver žádost zpracuje a odešle prohlížeči data ve formě XML.
Funkce probíhá jen v případě, že je navázáno spojení s webserverem. Funkce setTimeout() opakuje danou funkci každou vteřinu. Následně se proměnná strSTART vyprázdní a není tak dále posílána v rámci žádosti.

```
1.  function GetArduinoInputs()
2.  {
3.   request.onreadystatechange = function()
4.   {
5.    if (this.readyState == 4)
6.    {
7.     if (this.status == 200)
8.     {
9.      if (this.responseXML != null)
10.     {
11.     ... //Načtení hodnot z XML
12.     ... //Načtení stavu tlačítka
13.     ... //Načtení hodnot vyparsovaných z XML k zobrazení
14.     }
15.    }
16.   }
17.  }
18.
19. request.open("GET", "ajax_xml" + strSTART + nocache, true);
20. request.send(null);
21. setTimeout('GetArduinoInputs()', 1000);
22. strSTART= "";
23. }
```

K načtení dat z XML do stringu je realizováno pomocí Javascriptu.

Následně se z těchto stringů vyparsují hodnoty pomocí funkcí parseInt() a parseFloat().

```
1.  //Načtení hodnot z XML
2.  var rawVoltage = this.responseXML.getElementsByTagName('voltageinput')[0]
3.                              .childNodes[0].nodeValue;
4.  var rawCurrent = this.responseXML.getElementsByTagName('currentinput')[0]
5.                              .childNodes[0].nodeValue;
6.  var rawCapacity = this.responseXML.getElementsByTagName('capacityinput')[0]
7.                              .childNodes[0].nodeValue;
8.  var rawPWM = this.responseXML.getElementsByTagName('pwminput')[0]
9.                              .childNodes[0].nodeValue;
10. var rawSetpoint = this.responseXML.getElementsByTagName('setpointinput')[0]
11.                             .childNodes[0].nodeValue;
12. var rawCutoffVoltage = this.responseXML.getElementsByTagName('cutoffvoltageinp
    ut')[0]
13.                             .childNodes[0].nodeValue;
14.
15. Voltage = parseFloat(rawVoltage);
16. Current = parseInt(rawCurrent, 10);
17. Capacity = parseInt(rawCapacity, 10);
18. Setpoint = parseInt(rawSetpoint, 10);
19. PWM = parseInt(rawPWM, 10);
20. CutoffVoltage = parseFloat(rawCutoffVoltage);
```

Vyparsované hodnoty jsou následně přiřazeny k elementům podle Id.

```
1.  //Načtení hodnot vyparsovaných z XML k zobrazení
2.  document.getElementById('voltage').innerHTML = Voltage;
3.  document.getElementById('current').innerHTML = Current;
4.  document.getElementById('capacity').innerHTML = Capacity;
5.  document.getElementById('pwm').innerHTML = PWM;
6.  document.getElementById('setpoint').innerHTML = Setpoint;
7.  document.getElementById('cutoff').innerHTML = CutoffVoltage;
```

O zobrazení se starají HTML prvky, ke kterým byli přiřazeny hodnoty podle Id.

```
1.  <p>Voltage: <span id="voltage">...</span>V</p>
2.  <p>Current: <span id="current">...</span>mA</p>
3.  <p>Discharged: <span id="capacity">...</span>mAh</p>
4.  <p>PWM output: <span id="pwm">...</span></p>
5.  <p>Current Setpoint: <span id="setpoint">...</span>mA</p>
6.  <p>Cutoff Voltage: <span id="cutoff">...</span>V</p>
```

4.2.3 Nastavení vybíjecích parametrů

K nastavení parametrů slouží dvě textová pole se dvěma tlačítky.

```
<input type="text" id="Vol" value="1.5">
<button onclick="setVolt()">Set Cutoff Voltage</button><br>

<input type="text" id="Set" value="100">
<button onclick="setSet()">Set Current</button>
```

Po stisknutí tlačítka je zavolána patřičná funkce, která načte hodnotu z textového pole do proměnné a pošle HttpRequest, kde je patřičný identifikátor a žádaná hodnota v proměnné ohraničená dvojtečkami.

Následně se proměnná vyprázdní.

```
function setSet() {

  strSet = document.getElementById("Set").value ;

    request.open("GET", "set_set:" + strSet + ":" + nocache, true);
        request.send(null);

      strSet= "";
   }

function setVolt() {

  strVolt = document.getElementById("Vol").value ;

   request.open("GET", "set_volt:" + strVolt + ":" + nocache, true);
        request.send(null);

     strVolt= "";
}
```

4.2.4 Graf

Na hlavních osách zobrazuje závislost napětí na vybité kapacitě a na sekundárních osách proud v průběhu času. Vykresluje se v bloku, který je značený jako chartContainer.

```
function chart() {
var chart = new CanvasJS.Chart("chartContainer",
  {
      axisX: {
        title: "Discharged capacity [mAh]"
      },
      axisX2: {
        title: "Time [s]",
      },
      axisY: [
       {
         title: "Voltage [V]",
         lineColor: "#4F81BC",
         tickColor: "#4F81BC",
         labelFontColor: "#4F81BC",
         titleFontColor: "#4F81BC",
         lineThickness: 2
      },
      {
         title: "Current [mA]",
         lineColor: "#C0504E",
         tickColor: "#C0504E",
         labelFontColor: "#C0504E",
         titleFontColor: "#C0504E",
         lineThickness: 2
      }
      ],
         data: [{
             type: "spline",
             dataPoints: dataPoints1,
           },
        {
        type: "spline",
          axisXType: "secondary",
          axisYIndex: 1,
       dataPoints: dataPoints2,
        }
        ]
    });
chart.render();
}
```

Funkce setInterval následně vykresluje graf každé 2 vteřiny.

```
1.  setInterval(function(){chart()},2000);
```

Graf načítá hodnoty z pole. Do pole se hodnoty vkládají funkcí .push(), která nové hodnoty vkládá vždy na konec pole. Z důvodu nezávislosti na ostatních funkcích se hodnoty parsují ze stringů, do kterých byly načteny hodnoty z XML.

```
1.  dataPoints1.push({x: parseFloat(rawCapacity),y: parseFloat(rawVoltage)});
2.  dataPoints2.push({x: parseFloat(seconds),y: parseFloat(rawCurrent)});
```

K vytvoření časové osy slouží následující funkce, která každou vteřinu přičte 1.

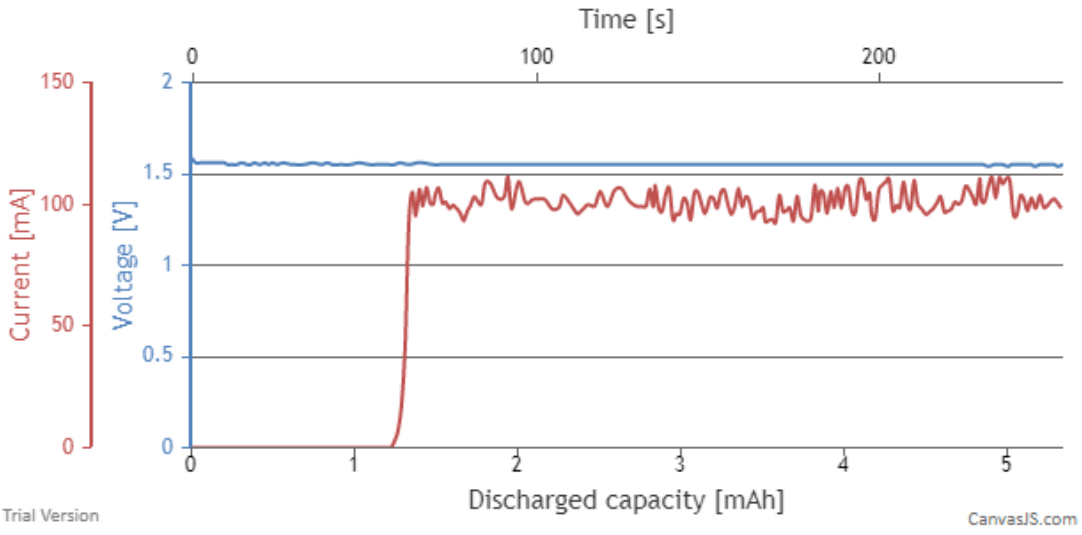
```
1.  var seconds = 0;
2.        setInterval(function () {
3.        seconds++;}, 1000);
```

5. Ověření funkčnosti

K otestování zařízení jsem provedl vybíjení na třech různých typech baterií. Vybíjecí obvod má při plně otevřeném tranzistoru odpor 1,25Ω, proto je maximální proud při téměř vybité baterii 0,72A.

Pro zinko-uhlíkovou a alkalickou baterii jsem provedl dvoje měření, jedno při 0,1A a druhé při 0,72A. Pro NiMH baterii jsem provedl měření pouze pro 0,72A.

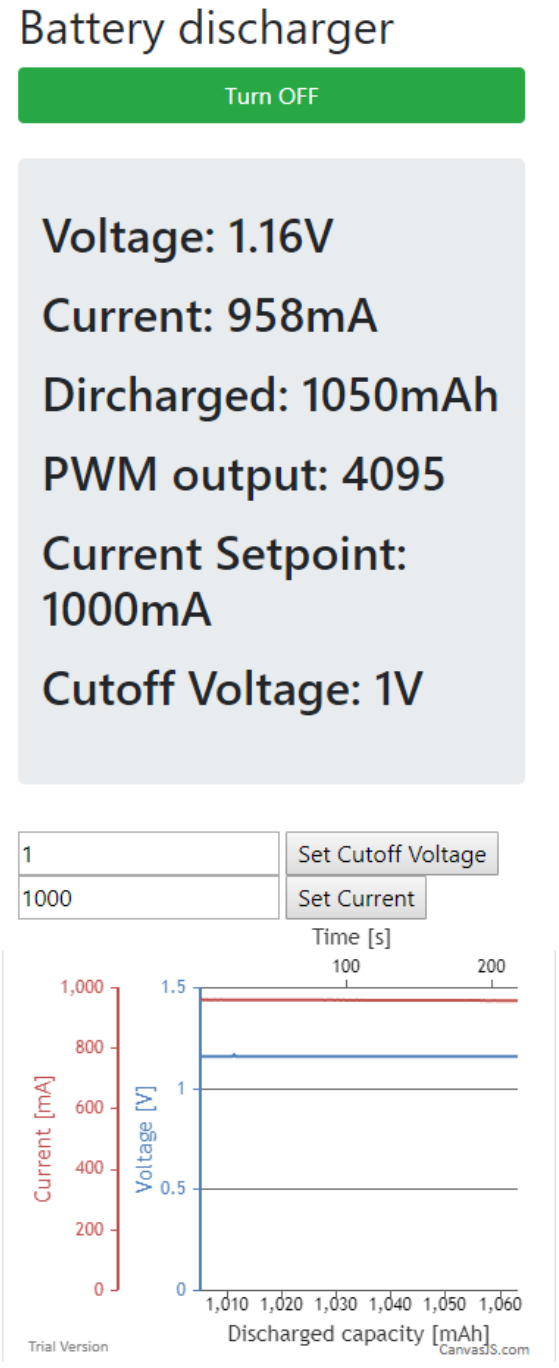
Při nastavení proudu 0,1A byl proud regulován s přesností ±10 mA. PWM výstup při tom měnil jen o ±1. Z toho vyplývá, že bych při použití PI regulátor nezpřesnil regulaci. Jediné, co by se zlepšilo, by byla doba přechodu, která je v případě žádaného proudu 1 A zhruba 5 vteřin. Výhodou takhle dlouhé doby přechodu je možnost vidět v grafu pokles napětí článku v závislosti na stoupajícím proudu.



Obr 20 Graf z webového rozhraní

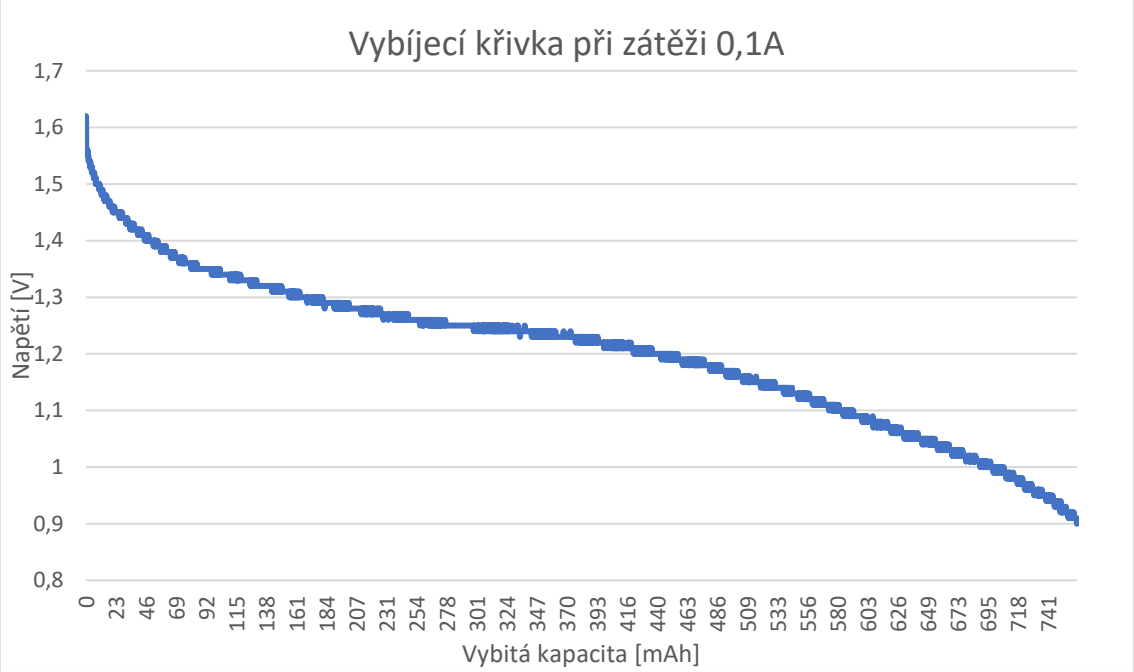
Webové rozhraní bylo plně použitelné na mobilním zařízení. Nevýhodou se stal graf, který vykresluje 4 body každé dvě vteřiny. To mělo za následek pád prohlížeče po zhruba půl hodině. Řešením by bylo nastavit graf jako průběžný, který by vykresloval omezený počet bodů. Jinou možností je přidat funkci na vypnutí grafu.

Dále by bylo vhodné přidat tlačítko na vynulování naměřené kapacity. To se provést jen restartem zařízení.

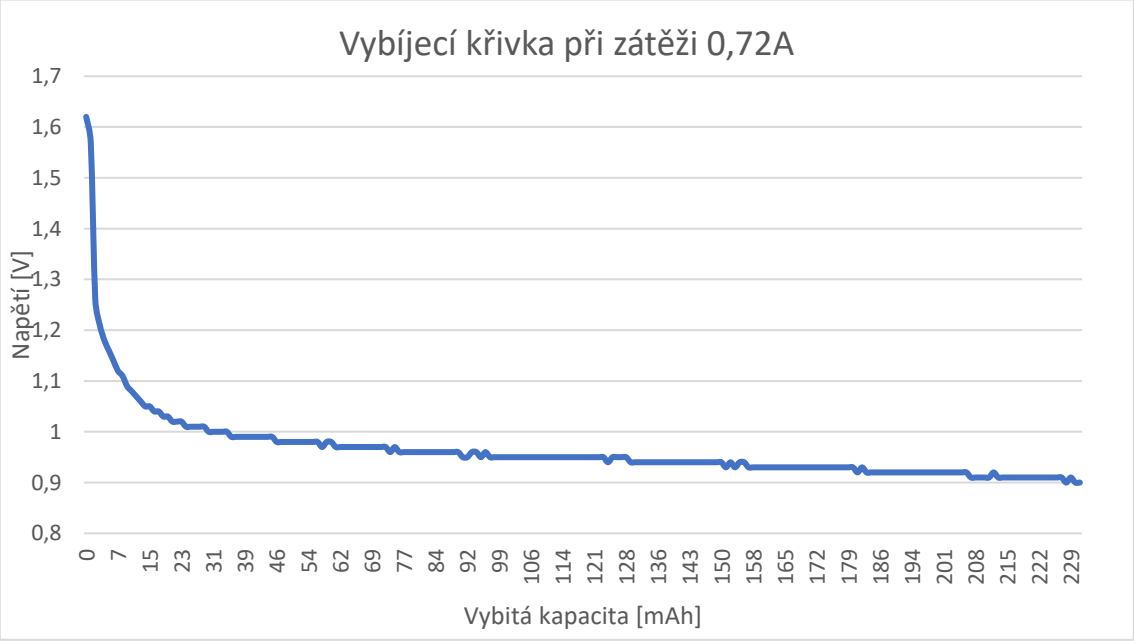


Obr 21 Webové rozhraní v mobilním zobrazení

5.1 Zinko-uhlíková baterie



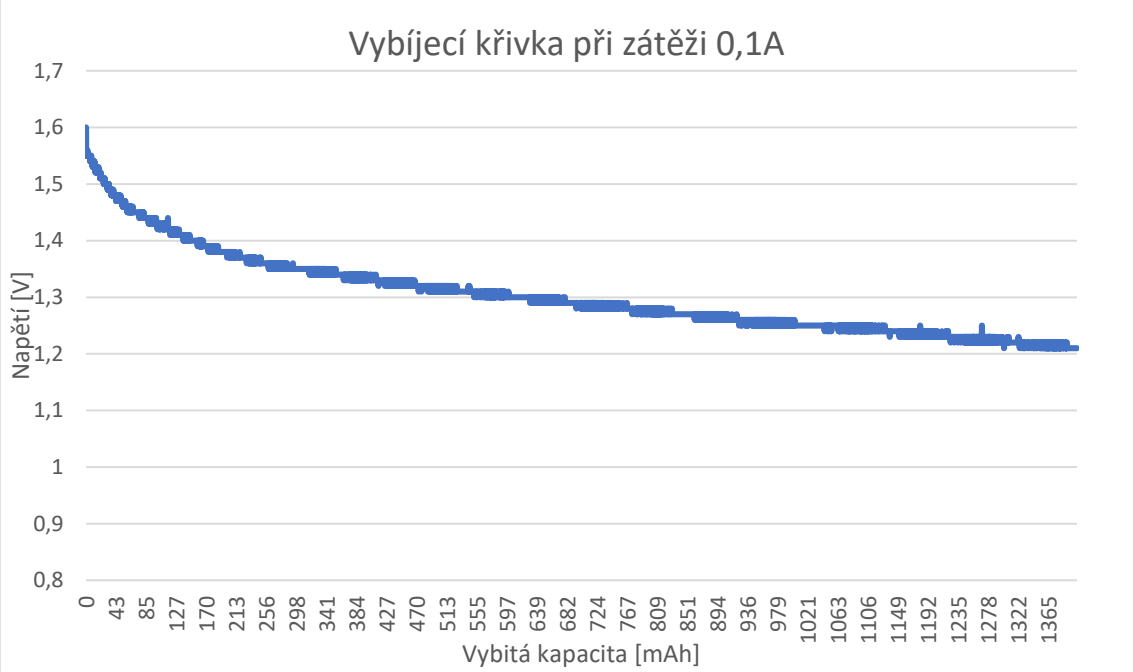
Graf 1 Vybíjecí křivka Zinko-uhlíkové baterie při zátěži 0,1A



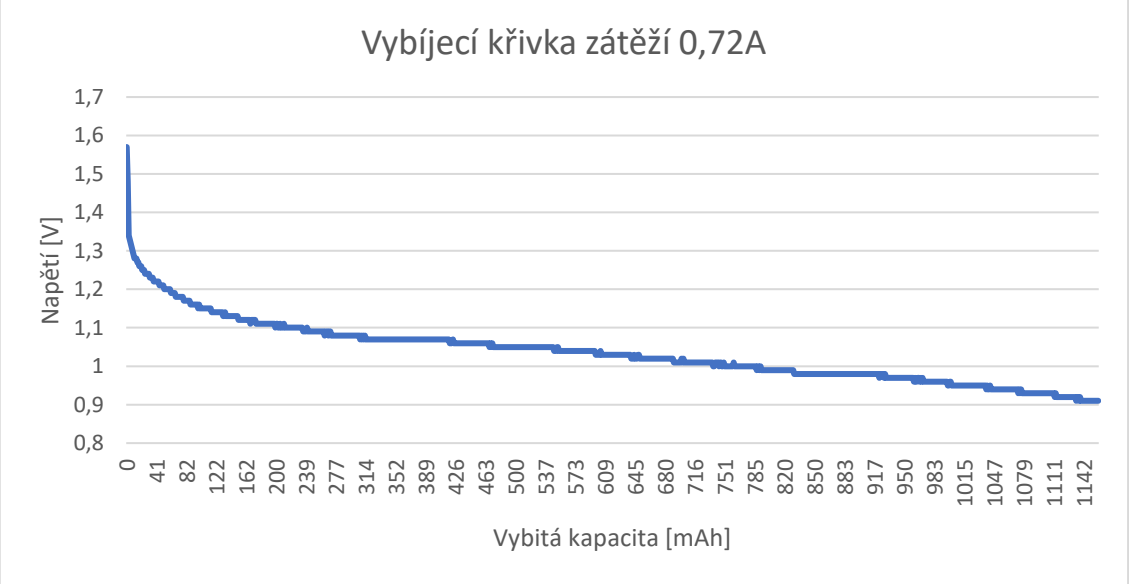
Graf 2 Vybíjecí křivka Zinko-uhlíkové baterie při zátěži 0,72A

Jak je vidět z druhého grafu, baterie není vhodná pro zatěžování většími proudy.

5.2 Alkalická baterie



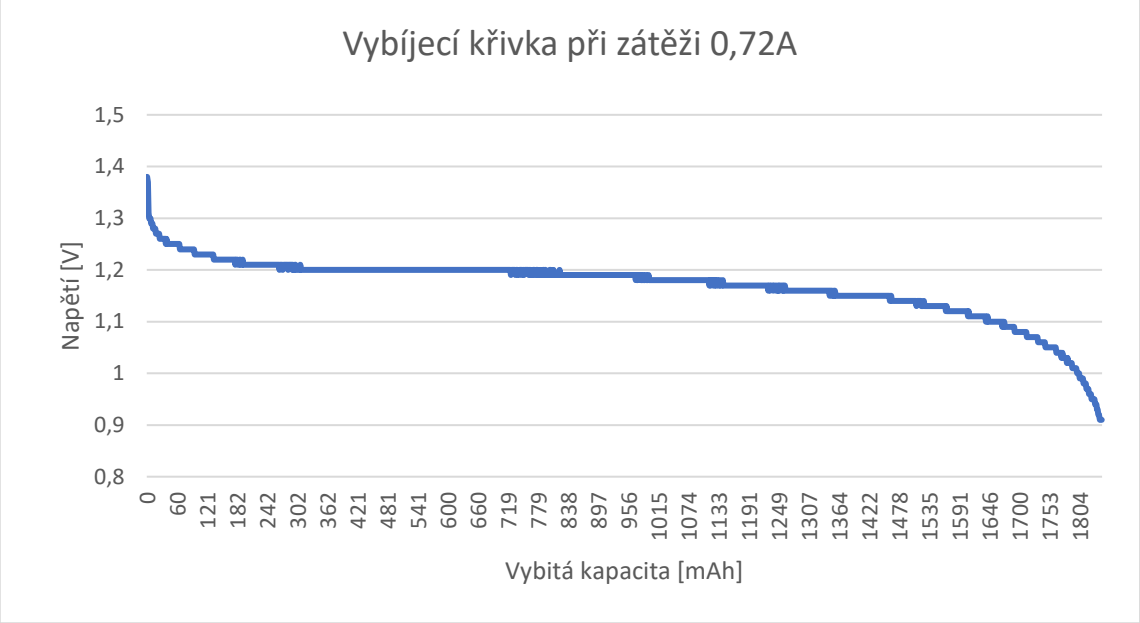
Graf 3 Vybíjecí křivka alkalické baterie při zátěži 0,1A



Graf 4 Vybíjecí křivka alkalické baterie při zátěži 0,72A

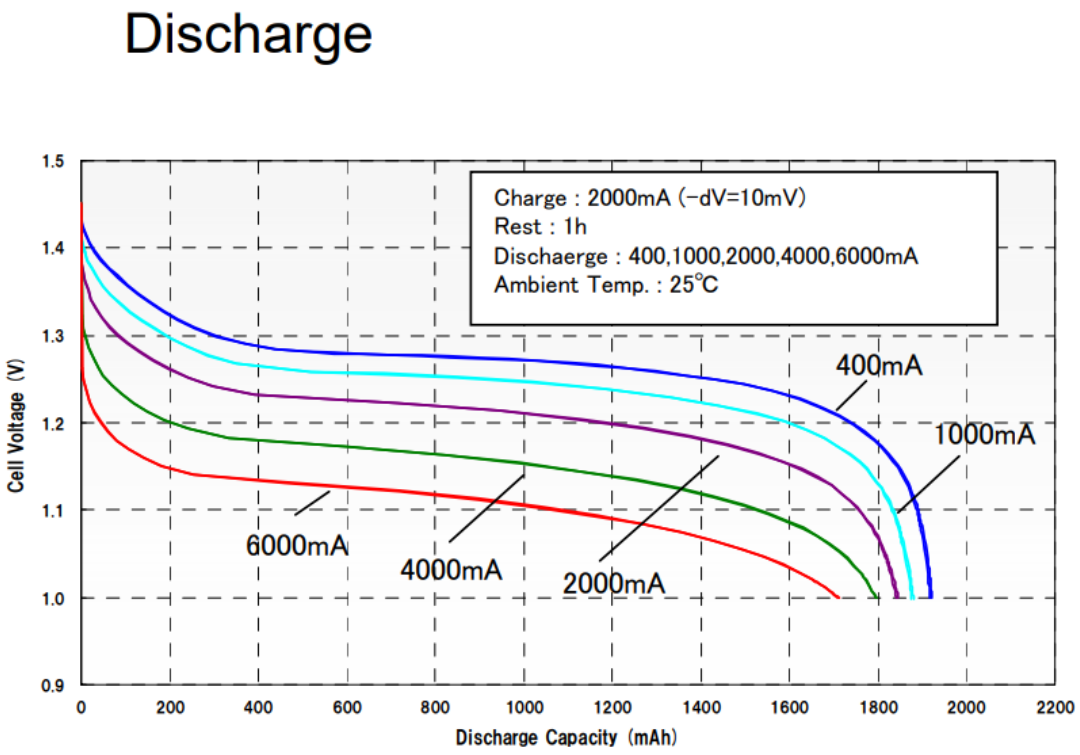
I přes vyšší pokles napětí je alkalická baterie vhodná pro větší zátěž.

5.3 NiMH baterie



Graf 5 Vybíjecí křivka NiMH baterie při zátěži 0,72A

NiMH baterie po většinu svého vybíjecího cyklu držela napětí 1,2V a vybíjecí křivka se shoduje s křivkou v katalogovém listě dané baterie.



Obr 22 Vybíjecí křivky z katalogového listu výrobce [23]

6. Závěr

Tato práce se zabývala návrhem a realizací zařízení k řízenému vybíjení elektrochemických článků o velikosti AA. O řídící část se stará Arduino Due. Jako senzor proudu a napětí byl zvolen senzor INA219. Výkonovou část tvoří kombinace tranzistoru v zesilovacím režimu a drátový odpor. Řízení probíhá pomocí PWM signálu převedeného na analogový signál pomocí filtru druhého řádu typu dolní propust. Držák článků je připojen pomocí JST konektoru, takže je možné připojit jakoukoliv baterii s nominálním napětím 1,5 V, na které je zařízení dimenzováno. Zařízení je schopno regulovat vybíjecí proud v rozmezí 0 až 0,72A při napětí baterie 0,9 V. Průběh vybíjení se ukládá na SD kartu v zařízení ve formátu .csv. Zařízení se ovládá pomocí webového rozhraní, které současně zobrazuje měřené hodnoty a vykresluje je do grafu.

Výkonová část se prokázala jako vhodná. Při rozšíření na Li-Pol baterie s více články je nutno brát ohled na daleko vyšší vybíjecí výkon. Pro výkonovou část je pak vhodná kombinace tranzistoru v pouzdře SOT-227 a drátového odporu, který je v hliníkovém pouzdře, osazené na chladiči.

Seznam obrázků

Seznam tabulek

Seznam grafů

Seznam použitého SW

Arduino IDE
KiCad
MATLAB 2018b
RJ TextEd

Seznam použité literatury a zdrojů

[1] SKUNDIN, A. M. - V. S. B. *Elektrochemické zdroje proudu*, Praha: SNTL, 1987

[2] KIEHNE, H. A. *Battery technology handbook*. 2nd ed. New York: Marcel Dekker, 2003. ISBN 08-247-4249-4.

[3] CENEK, Miroslav. *Akumulátory a baterie*, Praha: STRO.M, 1996.

[4] *Definition of Electrochemical Cell* [online]. [cit. 2019-05-27]. Dostupné z: https://www.chemicool.com/definition/electrochemical-cell.html

[5] *Ammeter Design* [online]. [cit. 2019-05-27]. Dostupné z: https://www.allaboutcircuits.com/worksheets/ammeter-design/

[6] *Using a ACS712 hall-effect current sensor* [online]. [cit. 2019-05-27]. Dostupné z: http://dangerousprototypes.com/blog/2012/01/27/interfacing-a-pic-microcontroller-with-the-acs712-hall-effect-current-sensor/

[7] *HTML5* [online]. [cit. 2019-05-27]. Dostupné z: https://www.w3schools.com/html/default.asp

[8] *CSS* [online]. [cit. 2019-05-27]. Dostupné z: https://www.w3schools.com/css/default.asp

[9] *JavaScript* [online]. [cit. 2019-05-27]. Dostupné z: https://www.w3schools.com/js/default.asp

[10] *AJAX Introduction* [online]. [cit. 2019-05-27]. Dostupné z: https://www.w3schools.com/js/js_ajax_intro.asp

[11] *Bootstrap 4* [online]. [cit. 2019-05-27]. Dostupné z: https://www.w3schools.com/bootstrap4/

[12] *JQuery* [online]. [cit. 2019-05-27]. Dostupné z: https://www.w3schools.com/jquery/default.asp

[13] 9 Best JavaScript Charting Libraries [online]. [cit. 2019-05-27]. Dostupné z: https://hackernoon.com/9-best-javascript-charting-libraries-46e7f4dc34e6

[14] *Arduino Due* [online]. [cit. 2019-05-27]. Dostupné z: https://store.arduino.cc/due

[15] *Sparkfun - Arduino Ethernet Shield 2* [online]. [cit. 2019-05-27]. Dostupné z: https://www.sparkfun.com/products/11166

[16] *NightShade Electronics* [online]. [cit. 2019-05-27]. Dostupné z: https://ns-electric.com/product/current-voltage-power-sensing-module-twii2c-ina219/

[17] Katalogový list INA219 [online]. [cit. 2019-05-27]. Dostupné z: http://www.ti.com/lit/ds/symlink/ina219.pdf

[18] Katalogový list IRLZ44NPBf [online]. [cit. 2019-05-27]. Dostupné z: http://www.irf.com/product-info/datasheets/data/irlz44n.pdf

[19] Using PWM Output as a Digital-to-Analog Converter ona TMS320F280x Digital Signal Controller [online]. [cit. 2019-05-27]. Dostupné z: http://www.ti.com/lit/an/spraa88a/spraa88a.pdf

[20] Method for Converting a PWM Output to an Analog Output When Using Hall Effect Sensor ICs [online]. [cit. 2019-05-27]. Dostupné z: https://www.allegromicro.com/en/Design-Center/Technical-Documents/Hall-Effect-Sensor-IC-Publications/Method-for-Converting-a-PWM-Output-to-an-Analog-Output-When-Using-Hall-Effect-Sensor-ICs.aspx

[21] Low-Pass Filter a PWM Signal into an Analog Voltage [online]. [cit. 2019-05-27]. Dostupné z: https://www.allaboutcircuits.com/technical-articles/low-pass-filter-a-pwm-signal-into-an-analog-voltage/

[22] Electronic Circuit Simulator [online]. [cit. 2019-05-27]. Dostupné z: http://www.falstad.com/circuit/index.html

[23] Katalogový list Panasonic BK-3MCC [online]. [cit. 2019-05-27]. Dostupné z: https://www.kronium.cz/uploads/BK-3MCCE.pdf