STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10: Elektrotechnika, elektronika a telekomunikace

Tester baterií

Pavel Horský
Jihomoravský

Brno 2022

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10: Elektrotechnika, elektronika a telekomunikace

Tester baterií

Battery tester

Autor: Pavel Horský
Škola: Gymnázium Brno-Řečkovice, Terezy Novákové 2, 621 00 Brno
Kraj: Jihomoravský
Konzultant: prof. Dr. Ing. Zdeněk Kolka

Brno 2022

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Brně dne 7. 2. 2022 ……

Poděkování

Chtěl bych poděkovat mému vedoucímu práce, panu prof. Dr. Ing. Zdeňku Kolkovi za cenné rady, odborný dohled a konzultace, díky kterým jsem tuto práci dovedl do zdárného konce. Dále bych chtěl poděkovat internímu konzultantovi práce Mgr. Zdeňku Votavovi za formální vedení a připomínky k textu.

Anotace

Ve své práci SOČ jsem se zabýval principy činnosti a praktickými vlastnostmi elektrochemických akumulátorů. Na základě tohoto rozboru jsem navrhl, realizoval a odzkoušel univerzální tester baterií, který umožňuje měřit všechny základní parametry akumulátorů, včetně výkonových, i celkovou energii, uschovávanou v akumulátoru. Tester obsahuje funkční bloky: řídící jednotka Arduino nano, proudový snímač s integrovaným obvodem INA219, elektronická zátěž včetně snímání zátěžového proudu, obvody ochrany proti přepólování, obvody pro měření napětí baterie na jednotlivých článcích a snímač teploty a řízení ventilátoru. Tester byl vyroben (včetně návrhu a realizace krabičky a desky plošného spoje) a proměřením bylo prokázáno, že splňuje očekávané parametry.

Klíčová slova

elektronická zátěž; proudový snímač; řídící jednotka Arduino nano; univerzální tester baterií.

Annotation

In my SPA work I studied principles of electrochemical batteries and their properties. Based on this analysis, I designed, implemented and tested a universal battery tester, which allows me to measure all basic parameters of batteries, including load characteristic and total energy stored in the battery. The tester contains the following functional blocks: Arduino nano control unit, current sensor with integrated circuit INA219, electronic load including load current sensing, reverse polarity protection circuits, circuits for measuring battery voltage and individual cells voltage, a temperature sensor and a fan control. The tester was manufactured (including the design and implementation of the box and dashboard) and the measurement proved that it meets all expected parameters.

Keywords

Electronic load, current sensor, Arduino nano control unit, universal battery teste.

Obsah

1 Úvod 8
2 Teoretická část 9
2.1 Elektrochemický akumulátor 9
2.1.1 Princip 9
2.1.2 Rozdělení elektrických akumulátorů 9
2.1.3 Vybíjení 14
2.2 Požadavky na tester baterií 14
2.2.1 Základní charakteristiky 15
2.3 Specifikace testeru podle návrhu Testeru 16
3 Praktická část 17
3.1 Hardware 17
3.1.1 Řídící jednotka Arduino nano 18
3.1.2 Proudový snímač s integrovaným obvodem INA219 18
3.1.3 Elektronická zátěž včetně snímání zátěžového proudu 19
3.1.4 Obvody ochrany proti přepólování 20
3.1.5 Obvody pro měření napětí baterie a jednotlivých článků 21
3.1.6 Snímač teploty a řízení ventilátoru 22
3.1.7 Celkové schéma zapojení 22
3.1.8 Deska plošných spojů 23
3.2 Software 25
3.2.1 Struktura menu 25
3.2.2 Měření charakteristik 27
3.2.3 Sériový výstup 32
3.2.4 Software v počítači 33
3.3 Mechanická konstrukce 34
3.4 Realizované zařízení 36
3.5 Měření 41
3.5.1 Měření vlastností 41
3.5.2 Měření zátěžové charakteristiky baterií 46
3.5.3 Měření zátěžové charakteristiky LiIon baterie 47
3.5.4 Měření zátěžové charakteristiky zdrojů 48
3.5.5 Měření kapacity baterie LiIon 49

3.5.6 Měření kapacity NiMh baterie 50
4 Závěr 52
5 Literatura 53
6 Seznam obrázků 54
7 Seznam tabulek 56
8 Přílohy 57
8.1 Seznam součástek 57
8.2 Program v testeru 59
8.3 Makro v Excelu 85

1 ÚVOD

Akumulátor je elektrochemický zdroj umožňující skladování elektrické energie. Poskytuje stejnosměrné napětí o různé velikosti v závislosti na počtu článků, použitém materiálu elektrod a elektrolytu.

V dnešní době je značná část přístrojů, používaná v každodenním životě, poháněna elektrickou energií z akumulátorů, což umožňuje vysokou mobilitu zařízení. V některých případech se bez akumulátoru nelze obejít, například při nedostupnosti elektrické sítě. V současné době se použití akumulátoru, jako hlavního zdroje pohonu, začíná uplatňovat i v osobní dopravě. Jako zdroje energie slouží pro různá jízdní elektrokola, elektrokoloběžky i pro automobily.

Moderní akumulátory zpravidla umožňují práci v jakékoliv poloze a s nízkou závislostí na okolní teplotě, existují i akumulátory optimalizované pro krátkodobý vysoký odběr proudu i jiné, určené pro dlouhodobé zálohování zdrojů. Snahou je vyrobit akumulátory s co nejvyšší koncentrací energie vztaženou na vlastní hmotnost akumulátoru, případně s co nejnižšími nároky na prostor.

Tato práce je zaměřena na návrh a konstrukci testeru pro měření výkonové kapacity a zátěžové charakteristiky různých typů akumulátorů. Tester dále umožňuje měření zátěžové charakteristiky napájecích zdrojů a může být použit jako elektronická zátěž s nastavitelným proudem. Měření je realizováno vybíjením akumulátoru nastavitelným proudem. Během vybíjení je měřen vybíjecí proud, napětí akumulátoru a je počítána kapacita. Vybíjení je ukončeno při dosažení nastaveného minimálního (koncového) napětí na celé baterii případně na jednotlivých článcích. Navrhovaný tester umožňuje akumulátor / zdroj zatížit až do proudu 8 A a maximálního výkonu na zátěži 100 W.

Tester je řízen jednodeskovým počítačem Arduino nano. Pro přesné měření proudu a napětí je využit integrovaný obvod INA219. Elektronická zátěž je realizována obvodem, který řídí vybíjecí proud procházející výkonovým tranzistorem na masivním chladiči s aktivním chlazením. Dále je tester vybaven obvodem ochrany proti přepólování, obvody pro měření napětí jednotlivých článků baterie a snímačem teploty a řízením ventilátoru.

Při realizaci jsem navrhl zapojení a desku plošných spojů, kterou jsem osadil a oživil. Napsal jsem a odladil program pro řízení testeru. Nakreslil jsem a na 3D tiskárně vytisknul krabičku. Tester jsem změřil, ověřil jeho činnost a vyzkoušel při měření různých baterií a napájecích zdrojů.

Práce je rozdělena do dvou částí. V první teoretické části je proveden teoretický rozbor vlastností různých typů akumulátorů a slouží jako základ pro návrh pracovních parametrů testeru.

V druhé, praktické části práce je popsán návrh, konstrukce a výroba testeru a desky plošných spojů. Dále následuje popis struktury ovládacího SW. Následuje kalibrace testeru a měření charakteristik vybraných akumulátorů.

2 TEORETICKÁ ČÁST

2.1 Elektrochemický akumulátor

Nejběžnější typy akumulátorů jsou založeny na elektrochemickém principu. Sekundární články jsou elektrochemické akumulátory, které využívají přeměnu elektrické energie na energii chemickou, kterou je možno v případě potřeby transformovat zpět na elektrickou energii.

2.1.1 Princip

Procházející proud v elektrochemickém akumulátoru vyvolá vratné chemické změny, které se projeví rozdílným elektrochemickým potenciálem na elektrodách. Z elektrod se pak dá čerpat na úkor těchto změn elektrická energie zpět. Protože jsou napětí na článcích elektrochemických akumulátorů relativně malá (okolo 1,2–3,7 V), jsou tyto články také sdružovány do akumulátorových baterií pro dosažení vyššího napětí.

2.1.2 Rozdělení elektrických akumulátorů

2.1.2.1 Podle principu

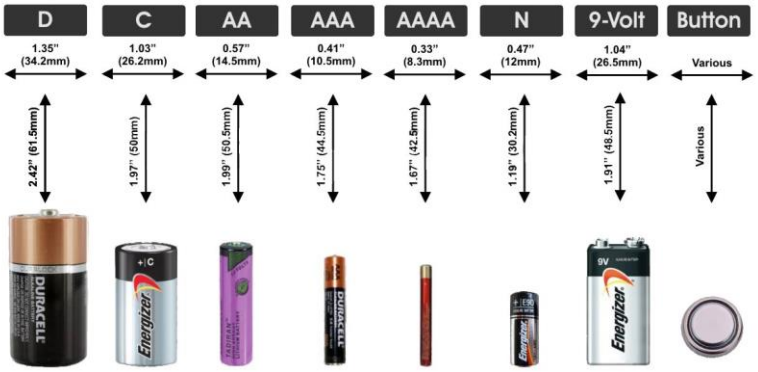
- olověný (Pb),
- nikl-kadmiový (NiCd),
- nikl-metal hydridový (NiMH),
- nikl-železný (Ni-Fe),
- nikl-zinkový (Ni-Zn),
- stříbro-zinkový,
- lithium-iontový (LiIon),
- lithium-polymerový (LiPo),
- lithium-železo-fosfátový akumulátor $LiFePO_4$ (Li-$FePO_4$),
- sodíkovo-sírový (Na-S),
- sodíkovo-chlorid nikelnatý (Na-$NiCl_2$),
- alkalický (RAM),
- ostatní

2.1.2.2 Podle použití

- průmyslové akumulátory,
- spotřební elektronika,
- vojenské aplikace,
- pro vysoké odběrové proudy,
- rychlonabíjecí,
- pro trvalé dobíjení,
- pro vysoké teploty,
- s MBU (Memory Back-up),
- ...

2.1.2.3 Podle tvaru a velikosti pouzdra

- válcové (tužkové)
  - AAAA (½AAAA),
  - AAA (⅓AAA, ½AAA, ⅔AAA, AAA, 5/4AAA, 5/3AAA),
  - AA (⅓AA, ½AA, ⅔AA, AA, 5/4AA),
  - A (⅔A, 4/5A, A, 4/3A),
  - ostatní válcové (A f, Cs, C, D, F, SF, N, …)
- prizmatické
  - malé prizmatické
- diskové (knoflíkové)
  - podle průměru (např. ⌀ 6,8 mm, ⌀ 11,5 mm, ⌀ 15,5 mm, ⌀ 25 mm, ...),
  - oválné
- hranolovité (+ jejich sestavy)[2]
  - různé typy konektorů

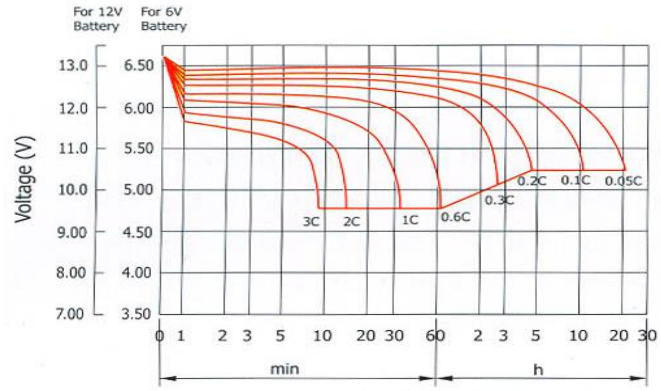


Obr. 1: Tvary baterií [1]

2.1.2.4 Podle principu

2.1.2.4.1 Pb

Olověný akumulátor je sekundární galvanický článek s elektrodami na bázi olova, jehož elektrolytem je kyselina sírová. Hlavní výhodou olověných akumulátorů je schopnost dodávat vysoké rázové proudy. Tato vlastnost, spolu s jejich nízkou cenou, je výhodná např. pro startování automobilu. Baterie těchto akumulátorů se vyrábějí v kapacitách řádově od 1 do 10 000 Ah [3]. Umožnují vytvořit tvrdý napájecí zdroj.



Obr. 2: Vybíjecí charakteristika olověného akumulátoru [4]

2.1.2.4.2 NiCd

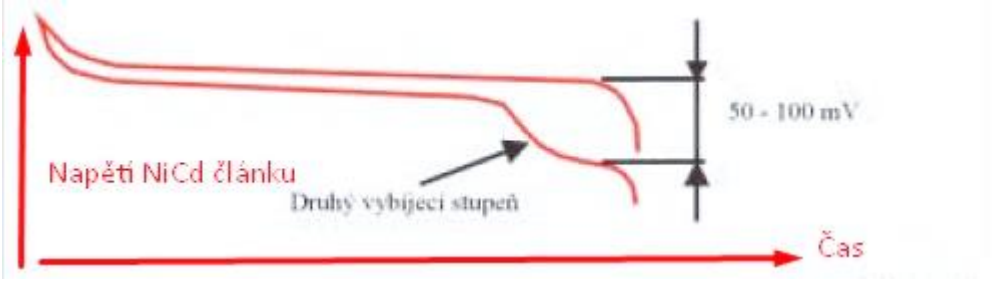
Nikl-kadmiový akumulátor, zkráceně **NiCd**, je druh galvanického článku. Vyrábí se jednak se zaplavenými elektrodami a kapalným elektrolytem (velké staniční baterie) a jednak jako hermetizovaný (akumulátory do přístrojů jako jsou akumulátorové vrtačky).

Mezi jeho výhody patří, že mu příliš nevadí skladování ve vybitém stavu a s tím související odolnost vůči hlubokému vybití. Určitou nevýhodou ve srovnání s NiMH a LiIon akumulátory je jeho relativně nižší měrná kapacita. Problematickým rysem tohoto akumulátoru je jedovatost kadmia, z něhož se skládá záporná elektroda a tedy nezbytnost sběru opotřebovaných NiCd akumulátorů (stejně jako v případě Pb akumulátorů). Svými vlastnostmi se jinak podobá novějšímu NiMH akumulátoru. Jmenovité napětí jednoho článku je 1,2 V. V plně nabitém stavu dosahuje napětí k 1,35 V a vybitý článek má 0,8–1,0 V [5] .

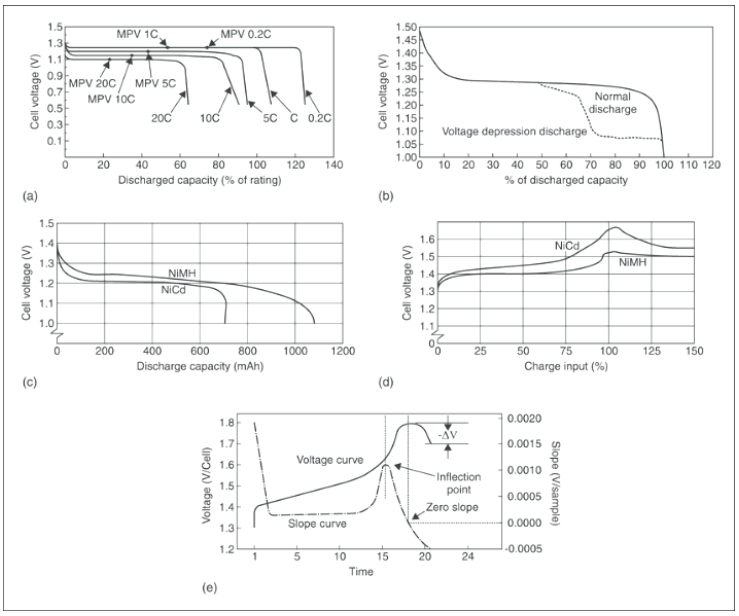
Pokud je niklokadmiový akumulátor vybíjen pouze z části a pak opětovně nabíjen na plnou kapacitu, může se stát, že po čase si „zapamatuje“ jako výchozí stav hodnotu, na které začalo nabíjení a pak není ochoten vydat celou svou kapacitu, ale jen její část.

Jedná se o tzv. druhý vybíjecí cyklus u NiCd akumulátorů, kdy při opakovaném vybíjení NiCd akumulátorů (minimálně 50–100× za sebou) na malou, ale vždy stejnou hloubku vybití, kdy dochází ke změně velikosti částic aktivní hmoty kadmiových elektrod a u akumulátoru pak při vybíjení dojde ke skokovému poklesu napětí o 0,05–0,1 V.

Tento jev je reverzibilní, takže stačí takhle „nemocný akumulátor“ několikrát hluboce vybít a nabít rychlonabíječkou, dojde k tzv. rekrystalizaci záporné elektrody akumulátoru a efekt zmizí [6]. Jsou to obyčejné dobíjecí baterie, ale jsou jedovaté.



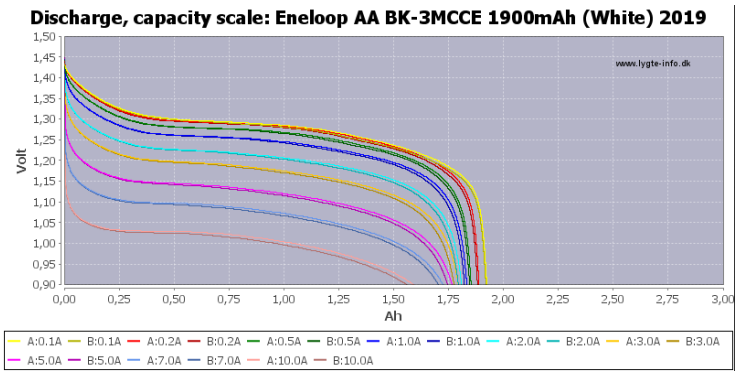
Obr. 3: Vybíjecí charakteristika NiCd článku [6]



Obr. 4: Vybíjecí charakteristika NiMh a NiCd článku [7]

2.1.2.4.3 NiMh

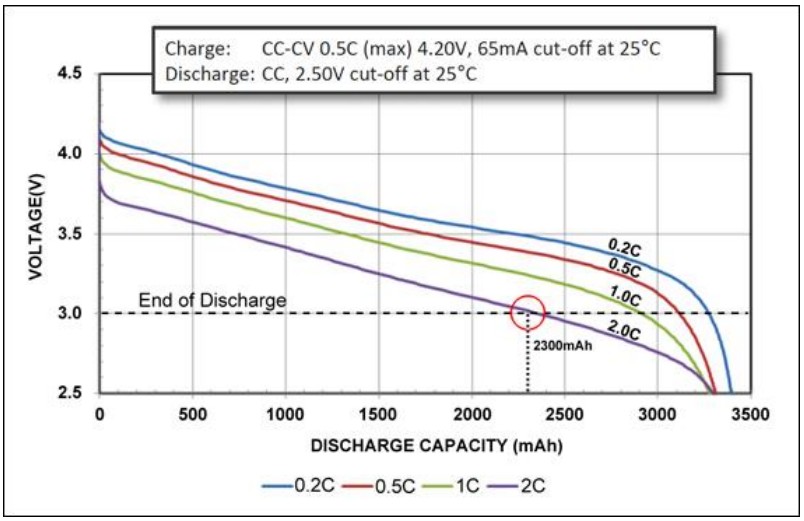
Nikl-metal hydridový akumulátor, zkráceně **NiMH**, je druh galvanického článku. V první dekádě 21. století to byl jeden z nejčastěji používaných druhů akumulátorů. Ve srovnání s jemu podobným nikl-kadmiovým akumulátorem má přibližně dvojnásobnou kapacitu. Hlavními důvody jeho velkého rozšíření je jeho značně velká kapacita a schopnost dodávat poměrně velký proud spolu s přijatelnou cenou [8]. Jsou to kvalitní akumulátory, nejedovaté.



Obr. 5: Vybíjecí charakteristika NiMh článku [9]

2.1.2.4.4 LiIon

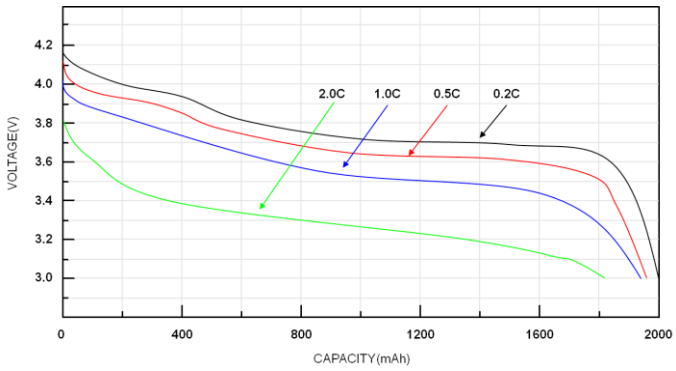
Lithium-iontová baterie (zkráceně **LiIon** baterie) je druh nabíjitelné baterie běžně používané ve spotřební elektronice. Kvůli vysoké hustotě energie vzhledem k objemu se výborně hodí pro přenosná zařízení. V současnosti je to v této oblasti asi nejpoužívanější typ. Chemický princip je velmi podobný jako v lithium-polymerových bateriích [10]. Nejběžnější, moderní, velmi vysoká kapacita.



Obr. 6: Vybíjecí charakteristika LiIon článku [11]

2.1.2.4.5 LiPo

Lithium-polymerový akumulátor (Li-pol, LiPo) je relativně nový typ elektrického akumulátoru. Lze jej použít téměř ve všech osobních elektronických zařízeních (např. mobilní telefony, fotoaparáty, notebooky, RC modely, …). Jsou vyvinuty z Lithium-iontových akumulátorů (LiIon) a zlepšují jejich vlastnosti (nízká hmotnost, relativně vysoká kapacita, minimální samovybíjení a velká výkonnost). Výroba akumulátorů je technologicky i energeticky náročná [12]. Velmi vysoká kapacita.



Obr. 7: Vybíjecí charakteristika LiPo článku [13]

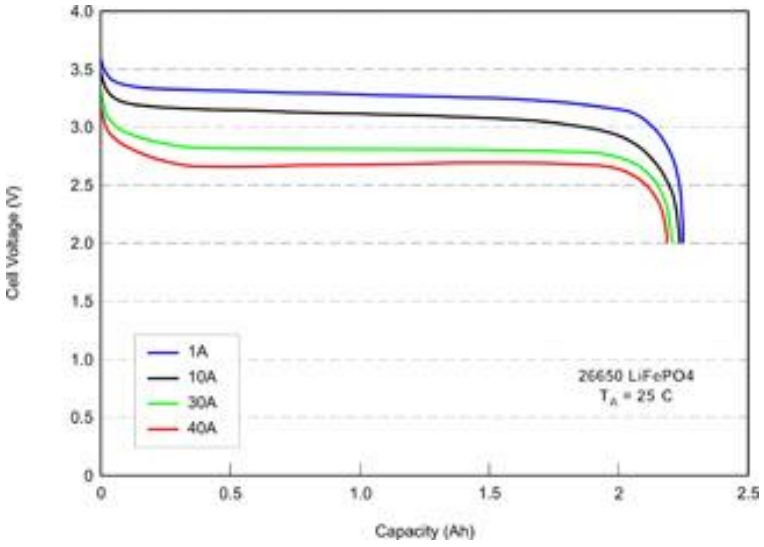
2.1.2.4.6 LiFe

LiFe jsou v současné době brány jako jedny z nejbezpečnějších baterií celkově, z hlediska požárního i obecně bezpečnostního. Nevytékají, nehoří a nedochází k explozím. Jsou velmi odolné proti vzplanutí při nechtěném mechanickém poškození na rozdíl od LiPo baterií. Výrobce udává teplotu kompletního vzplanutí 400 stupňů Celsia. Tedy asi bude spíše hořet všechno kolem díky žáru, než vlastní baterie. Naproti tomu LiPol baterie se mohou samovznítit už při teplotě varu vody.

LiFe baterie mají životnost 2000–3000 cyklů při nabíjení na 100 % a vybíjení na 20 % kapacity, nebo se uvádí životnost až 20 let ve staničních UPS a aplikacích, kde nejsou cyklovány, ale ani

potom nejsou na vyhození. Pouze jim klesne kapacita na cca 75-80 % kapacity a takto fungují dalších cca 1000 cyklů.

LiFe (vojenské provedení) mají rozsah pracovních teplot −20 až +80 stupňů. LiFe v provedení commercial mají uváděný rozsah 0 až +60 stupňů při zachování kapacity 90 % [14].



Obr. 8: Vybíjecí charakteristika LiFe článku [15]

2.1.3 Vybíjení

Doporučený vybíjecí proud je zpravidla napsán na baterii ve tvaru xC, kde x je předepsaný násobek a C je kapacita daného akumulátoru. Toto značení je použito i v obrázcích.

2.2 Požadavky na tester baterií

Každým rokem je víc a víc zařízení napájených místo ze sítě z baterie. Baterie ale mají oproti zdrojům tu nevýhodu, že jim klesá kapacita a výkon, a není to na nich bez měření zřejmé, proto je důležitý tester, který skutečné vlastnosti konkretní baterie prověří. Proto bych se chtěl zaměřit na baterie, které jsou běžně dostupné. Také se zajímám o RC modely, ve kterých jsou baterie, u nichž je kapacita a výkon velice důležitá.

Protože baterie mají různé tvary, velikosti a jiné způsoby připojení, tak je důležité, aby je bylo možno k navrženému testeru jednoduše připojit. To jsem vyřešil tak, že na tester budou upevněny univerzální konektory (4 mm banánky, které snesou více než 10 A) a k nim se poté připojí baterie přes redukční konektor.

U lithiových baterií pro RC modely bývá navíc balanční konektor. Je to konektor, který umožňuje měřit jednotlivé články ze série. Normálně u vybíjení není vyžadován, ale protože u lithiových baterií hrozí při podbití zničení, tak jej budu používat, protože některý článek ze série může mít menší kapacitu než ostatní a mohlo by se stát, že celkově se baterie ještě zdá v pořádku, ale některý z článků už může být podbitý. Toho se taky dá využít, pokud zjistíme,

že jeden článek je pokažený a zbylé jsou v pořádku, tak stačí vyměnit vadný článek a nemusíme vyhodit celou baterii.

Většina běžně dostupných baterií, se kterými jsem se doposud setkal, má napětí do 50 V a řádově kapacitu jednotek až desítek ampérhodin. Existují i výkonné baterie, které jsou mimo rozsah této práce. Kdybychom ale chtěli měřit třeba baterii, která má 12 V a okolo 50 Ah (např akumulátor do auta), tak kdybychom ji chtěli vybít třeba za hodinu, museli bychom přeměňovat na teplo okolo 500 W, což je příliš velký výkon na to, aby tester měl rozumnou velikost a váhu. Proto u větších baterií se budeme muset smířit s pomalejším vybíjením a nemožností udělat zátěžovou charakteristiku (u výše zmíněné autobaterie při startu tečou stovky ampér, z toho plyne, že bychom museli „pálit“ tisíce wattů, což by vyžadovalo několikanásobně větší tester).

Rozhodl jsem se, že bude stačit vybíjet výkonem okolo 100 wattů, k tomu použiji chladič od procesoru počítače.

2.2.1 Základní charakteristiky

2.2.1.1 Vybíjecí charakteristika a celková kapacita

Vybíjecí charakteristika je graf, ve kterém je zobrazována závislost napětí baterie na čase vybíjení. Z tohoto grafu lze vypočítat celkovou výkonovou kapacitu baterie. Ta se rovná součinu doby vybíjení a vybíjecího proudu (při vybíjení konstantním proudem).

Protože každý typ baterie, ale i stejné typy různých výrobců mají různé koncové napětí, při kterém se musí ukončit vybíjení, tak v programu jsou nastavena pouze orientační napětí, která můžete vidět v tabulce níže, a uživatel si vždy napětí musí dostavit sám podle požadavků výrobce měřeného článku.

Tab. 1: Přednastavené napětí pro 1 článek v programu

Typ článku	**Minimální napětí na jednom článku**
LiIon	3 V
LiPo	3 V
LiFe	3 V
NiMH	0,9 V
NiCd	0,9 V
Olověný Pb	1,8 V

2.2.1.2 Zátěžová charakteristika

Zátěžová charakteristika slouží k zjištění jak velkých proudů je zdroj/baterie schopna dodat, tak aby napětí příliš nekleslo.

Ze zátěžové charakteristiky je možné jednoduše spočítat dynamický odpor zdroje. Je dán podílem změny napětí ke změně proudu.

2.3 Specifikace testeru podle návrhu Testeru

Na základě výše uvedených rozborů jsem navrhl k realizaci uvedenou specifikaci, která je pro běžné užití optimální.

pro měření článků typu	LiIon, LiPo, LiFe, NiCD, NIMH, Pb	
pro počet článků	1; 2	
měření napětí	0,5 V až 25 V	±0,2%
měření proudu	50 mA až 8 A	±0,3%
měření času vybíjení	1 s až 24 hod	0,01%
maximální ztrátový výkon	100 W, testováno do 60 W	
teplota okolí	(23 ± 5)° C	
napájecí zdroj	USB port / 5 V až 12 V max. 1A	

3 PRAKTICKÁ ČÁST

Tato část obsahuje popis zvoleného řešení testeru.

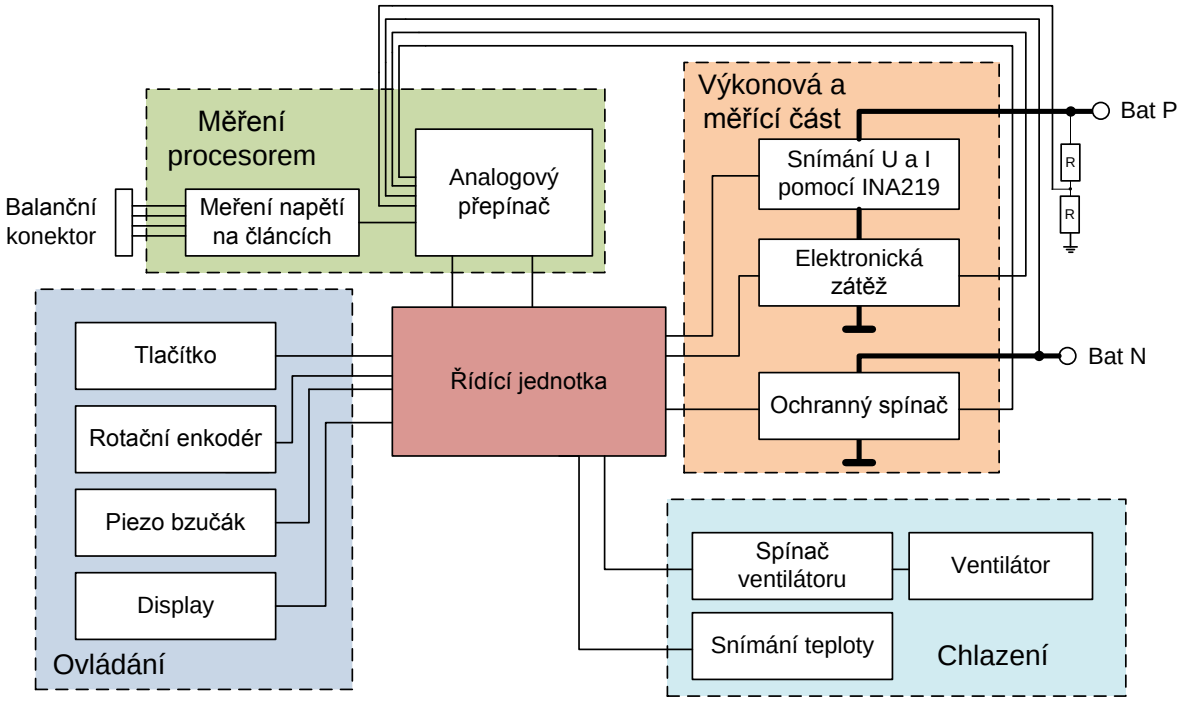
3.1 Hardware

Elektronické obvody testeru baterií sestávají z následujících částí:

- Řídící jednotka Arduino nano,
- Proudový snímač s integrovaným obvodem INA219,
- Elektronická zátěž včetně snímání zátěžového proudu,
- Obvody ochrany proti přepólování,
- Obvody pro měření napětí baterie a jednotlivých článků,
- Snímač teploty a řízení ventilátoru,
- Další pomocné obvody,
- Display se sériovým řízením,
- Rotační ovladač a tlačítko

Elektronické obvody jsou umístěné na desce plošných spojů, kterou jsem si pro tento účel navrhnul v programu Eagle, nechal vyrobit a potom osadil a oživil.

Blokové schéma zapojení testeru je znázorněno na obr. 9.



Obr. 9: Blokové schéma zapojení

Následuje popis jednotlivých obvodů na desce plošných spojů.

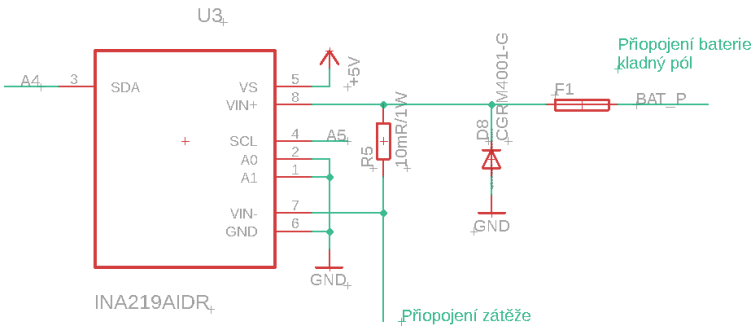
3.1.1 Řídící jednotka Arduino nano

Arduino je název malého jednodeskového počítače založeného na mikrokontrolerech ATmega od firmy Atmel a bylo poprvé představeno v roce 2005. [17]

Arduin jsou různé velikosti s různou výkonností. Já jsem si pro tester baterií vybral Arduino Nano.

Arduino Nano je minimalizovaná vývojová deska s mikrokontrolérem AVR ATmega328. Funkčně je shodná s verzí Arduino UNO, je ale přizpůsobená pro zasunutí do nepájivého kontaktního pole. Kvůli minimalizaci chybí konektor pro externí napájení a USB konektor je zmenšen na verzi mini USB. Samotná deska obsahuje 14 digitálních vstupů/výstupů a 8 analogových vstupů. Konektor pro externí napájení je nahrazen dvojicí pinů pro připojení 5–20 V. [18]

3.1.2 Proudový snímač s integrovaným obvodem INA219



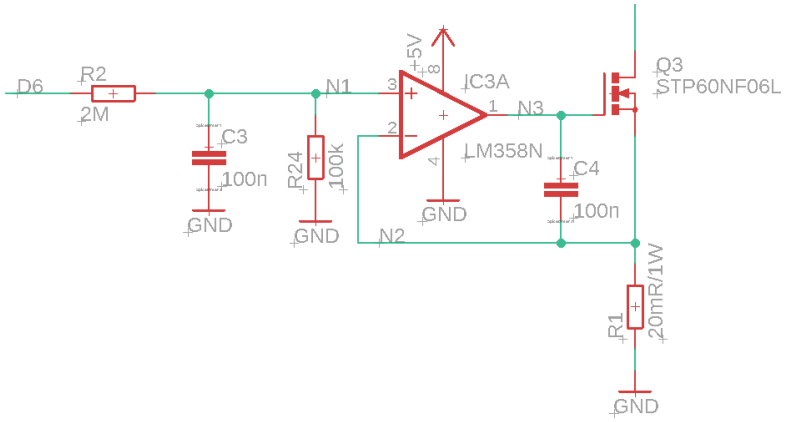
Obr. 10: Schéma zapojení měření napětí a proudu s integrovaným obvodem INA219

Proudový snímač s integrovaným obvodem INA219 umožňuje měření proudu pomocí měření napěťového úbytku na bočníku a také měření napětí. Obvod disponuje čtyřmi napěťovými rozsahy a dvěma proudovými rozsahy. Změřená hodnota je převedena dvanácti bitovým A/D převodníkem a výsledek je komunikován pomocí I2C sběrnice do Arduina.

Maximální vstupní napětí obvodu je 26 V, to je tedy maximální napětí, které může být přivedeno na vstup testeru. Při použití rozsahu 16 V je rozlišení 4 mV a pro rozsah 32 V je rozlišení 8 mV.

V zapojení používám externí bočník o velikosti 10 mΩ, což je optimální hodnota vzhledem k citlivosti i potřebnému výkonu a tedy i chlazení. Používám kvalitní součástku v SMD provedení s malou teplotní závislostí. Tomu odpovídají proudové rozsahy do 4 A, 8 A, 16 A a 32 A. V programu používám rozsah 4 A. Při použití Arduino knihovny pro ovládání obvodu INA219 používám rozsah 400 mA, protože mám 10× menší hodnotu odporu bočníku a výslednou hodnotu násobím 10×. [19][20]

3.1.3 Elektronická zátěž včetně snímání zátěžového proudu



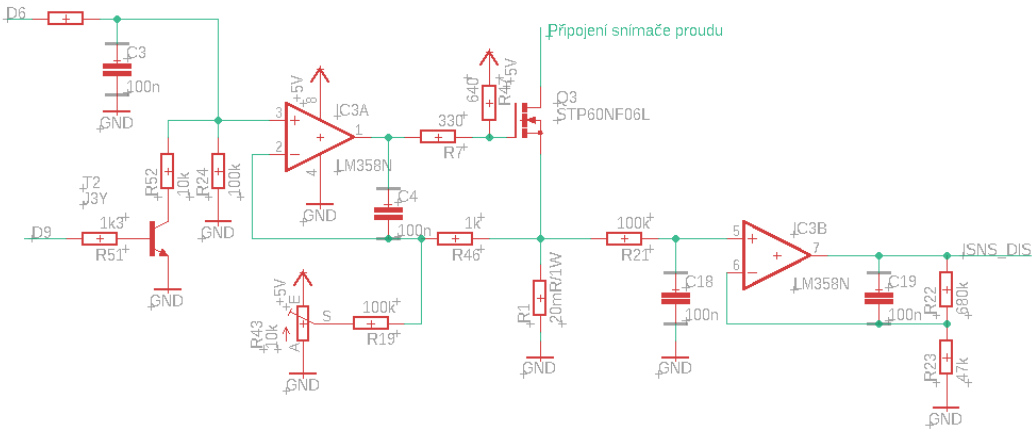
Obr. 11: Zjednodušené zapojení obvodu elektronické zátěže

Elektronická zátěž je realizována výkonovým tranzistorem typu P60NF06 [23], který je umístěný na masivním chladiči s aktivním chlazením. Chladič původně sloužil k chlazení procesoru v PC.

PWM výstup Arduina na pinu D6 je filtrován a napětí je poděleno přibližně 21×. Tím je možné nastavovat napětí na uzlu N1 v rozsahu od 0 V do 238 mV (5 V / 21) s krokem přibližně 0,9 mV (PWM má 8 bitů, tomu odpovídá 255 úrovní).

Operační zesilovač IC3A slouží k regulaci napětí na hradle tranzistoru Q3 tak, aby na snímacím rezistoru R1 bylo stejné napětí, jako je podělené napětí na uzlu N1. Proud protékající rezistorem R1 a tedy možné nastavovat v rozsahu od 0 do 11,9 A (238 mV / 20 mΩ) s krokem 45 mA (0,9 mV / 20 mΩ).

Pro měření malých baterií je možné změnit dělící poměr děliče signálu z pinu D6 tím, že k rezistoru R24 připojíme paralelně rezistor R52 pomoci tranzistoru T2 a napětí v tomto případě bude poděleno přibližně 221×. Takto zmenšíme maximální vybíjecí proud i krok nastavení proudu přibližně 10×.



Obr. 12: Zapojení obvodu elektronické zátěže

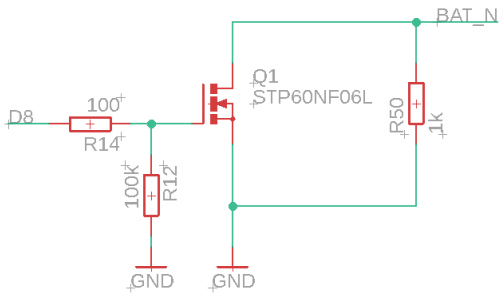
Pro dostavení nulového proudu z důvodu napěťové nesymetrie vstupů operačního zesilovače (to je takzvaná kompenzace ofsetu) je použit trimr R43. Rezistory R7, R47 slouží k zvětšení proudového rozsahu, protože použitý operační zesilovač má omezené maximální výstupní napětí.

Obvod na obr. 12 dále umožňuje snímání vybíjecího proudu pomocí měření zesíleného napětí na rezistoru R1 – součástky R21, C18, IC3B, C19, R22a R23 na obr. 12. Tato část není využívána, protože k přesnému měření proudu je použit obvod INA219.

3.1.4 Obvody ochrany proti přepólování

Protože měřená baterie nebo napájecí zdroj jsou k testeru připojeny přes banánky a je proto možné i přepólování, je tester vybaven ochranným obvodem. Tento obvod připíná záporný pól baterie na zem přístroje pouze tehdy, když je baterie správně připojena.

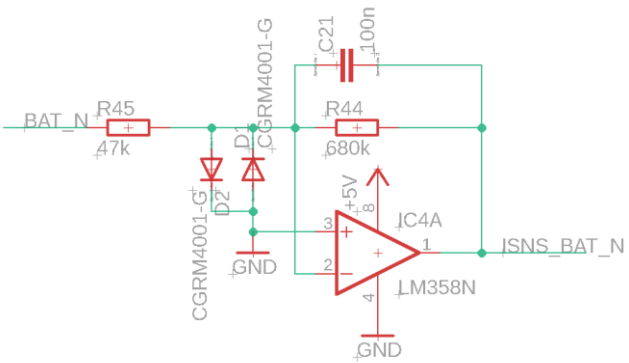
Spínač pro připojení záporného pólu baterie je realizován výkonovým MOS tranzistorem typu P60NF06, který je řízený z Arduina pinem D8.



Obr. 13: Schéma zapojení ochrany proti přepólování

Rezistor R14 je ochranný, rezistor R12 zajišťuje vypnutí spínače pokud Arduino neřídí pin D8 a rezistor R50 nastavuje potenciál na baterii tak, aby před sepnutím spínače polarity bylo možné měřit napětí na kladném a záporném pólu baterie a vyhodnocovat polaritu.

Dále je použit obvod pro měření úbytku napětí na spínači pro připojení záporného pólu baterie.



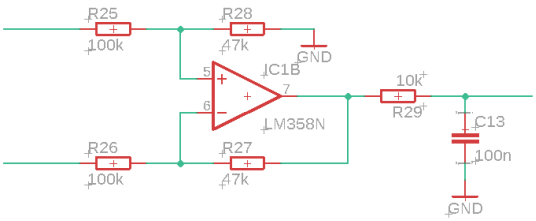
Obr. 14: Schéma zapojení obvodu pro měření úbytku napětí na ochranném spínači

Tento obvod zesílí úbytek napětí 14,5×. Zesílené napětí je přivedeno na A/D převodník v Arduinu a může sloužit ke kompenzaci chyby způsobené úbytkem na ochranném tranzistoru. Úbytek napětí na tomto tranzistoru je tak malý, že prakticky kompenzace není zapotřebí a není proto implementována v programu.

3.1.5 Obvody pro měření napětí baterie a jednotlivých článků

Pro měření napětí baterie je použit odporový dělič R10 a R11, filtr a ochranná Zenerova dioda D4 na obr. 15.

Měření jednotlivých článků baterií připojených na balanční konektor je umožněno pomocí přístrojového zesilovače, který rozdílové napětí na vstupu převede na napětí proti zemi na výstupu. Jeho zapojení je:



Obr. 15: Schéma zapojení obvodu pro měření napětí na jednom článku

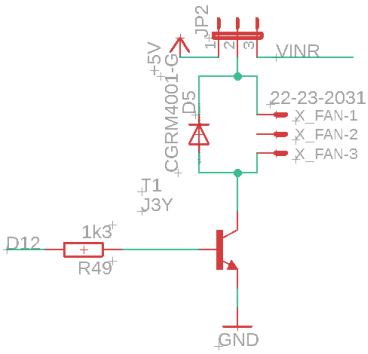
Aby bylo možné měřit různá napětí, je použit analogový elektronicky řízený přepínač 4051N [21]. Tento přepínač umožňuje přepnutí 8 různých vstupních napětí na jeden výstup.

Tab. 2: Tabulka připojení přepínače 4051N

Vstup č.	A (D4)	B (D7)	C (A3)	Vstupní napětí	Zesílení
0	0	0	0	Kladná vstupní svorka	¼
1	1	0	0	Vybíjecí proud	0,02 až15,5 mA
2	0	1	0	Zesílený úbytek napětí na spínači v záporné vstupní svorce	14,5
3	1	1	0	Napětí na 3. článku baterie	0,47
4	0	0	1	Napětí na 2. článku baterie	0,47
5	1	0	1	Napětí na 1. článku baterie	0,47
6	0	1	1	Napětí na nulovém vstupu balančního konektoru	1
7	1	1	1	Napětí na záporné vstupní svorce	1

3.1.6 Snímač teploty a řízení ventilátoru

Pro snímání teploty na chladiči je v blízkosti výkonového tranzistoru umístěn snímač DS18B20. Tento snímač umožňuje měřit teplotu v rozsahu od −55 °C do +125 °C s přesností ±0,5 °C v rozsahu od −10°C do +85°C [23].

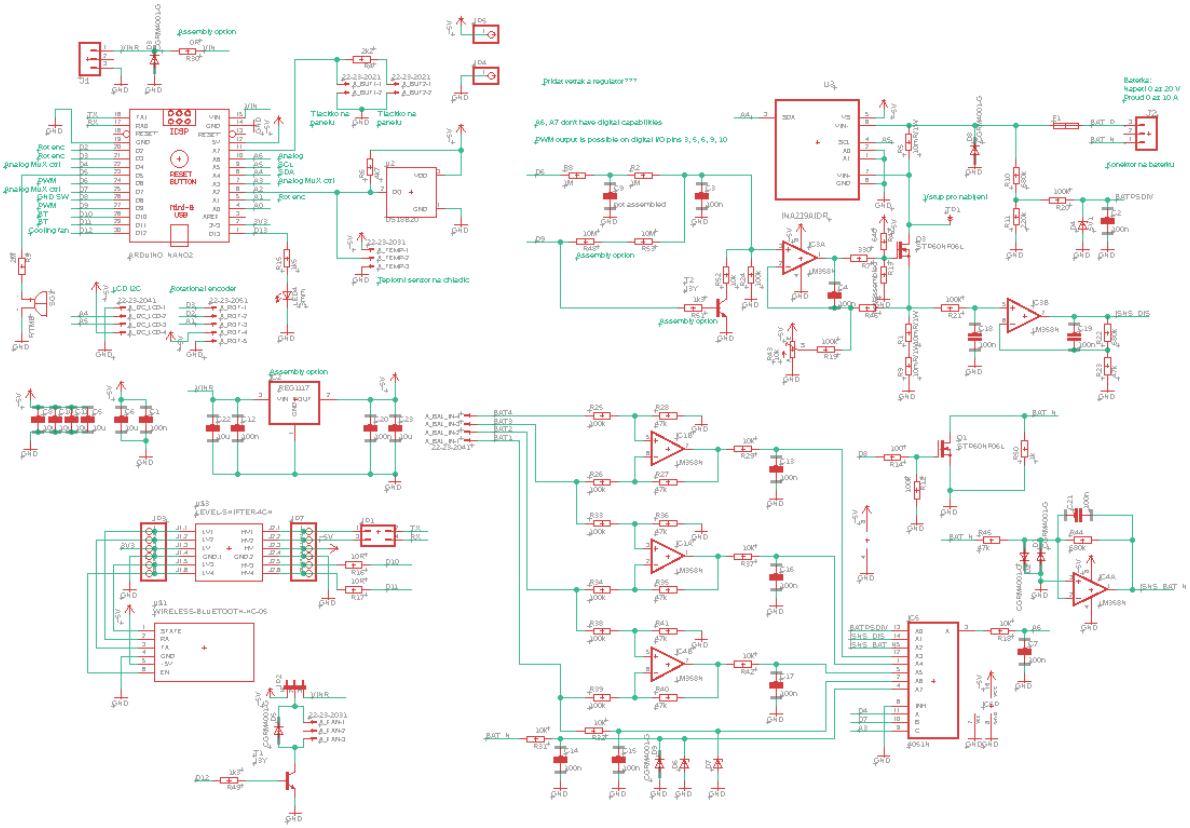


Obr. 16: Schéma zapojení snímání teploty na chladiči

Ventilátor chladiče je ovládán výstupem D12 a je spínán pomocí tranzistoru T1.

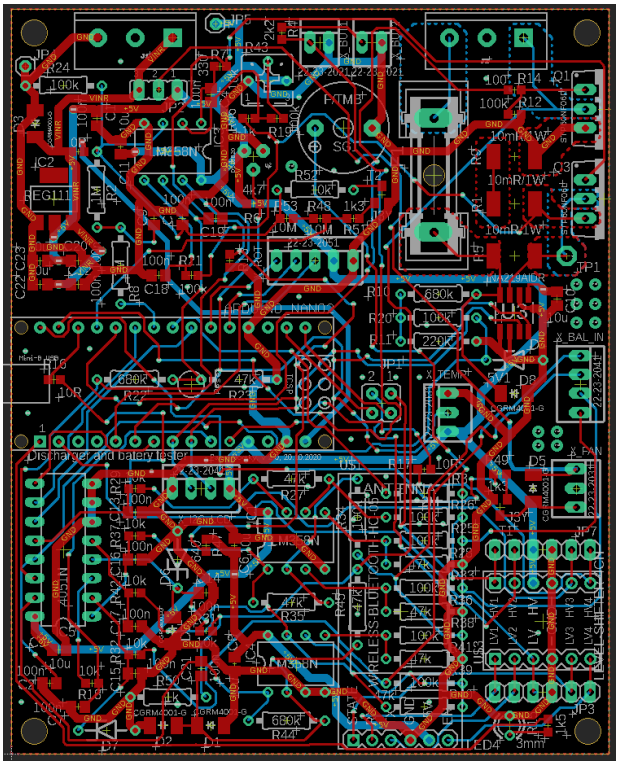
Pomocí jumperu JP2 je možné volit napájení ventilátoru, buď 5 V pro Arduino nebo vstupní napětí na napájecím konektoru, např. 12 V.

3.1.7 Celkové schéma zapojení

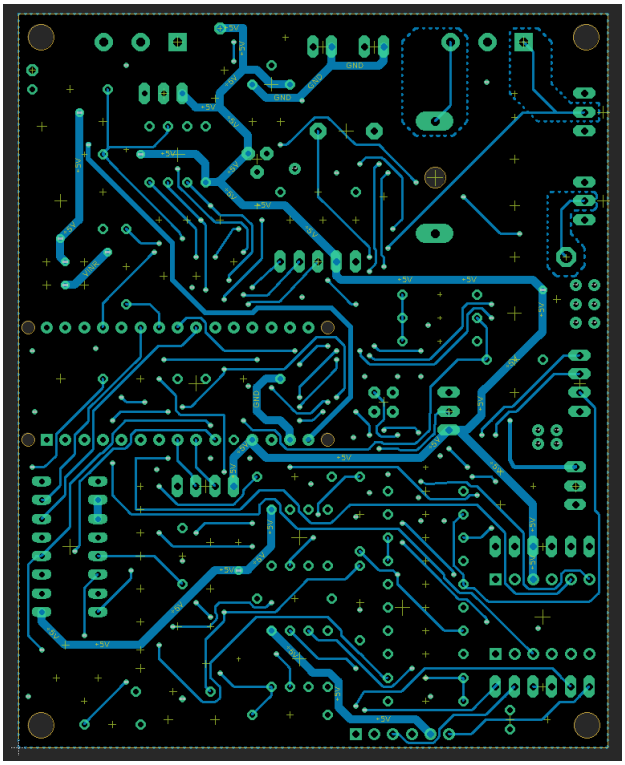


Obr. 17: Celkové schéma zapojení

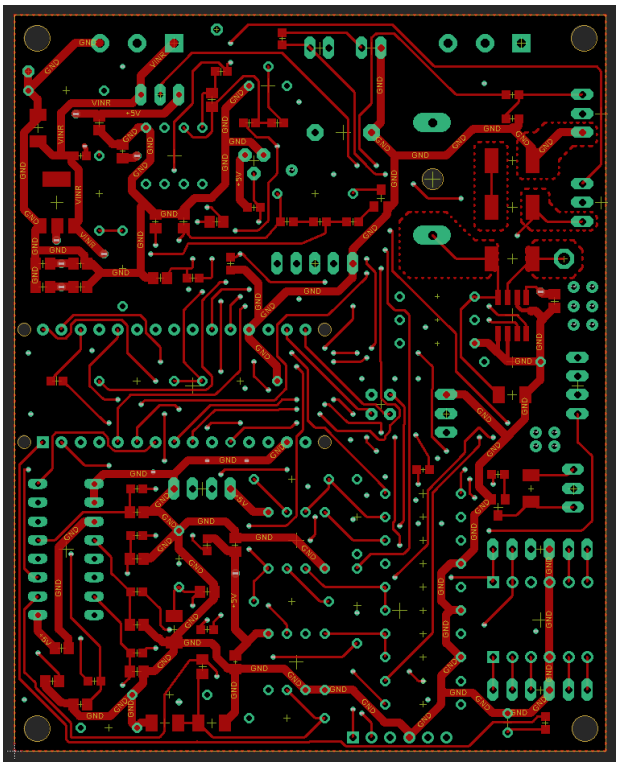
3.1.8 Deska plošných spojů



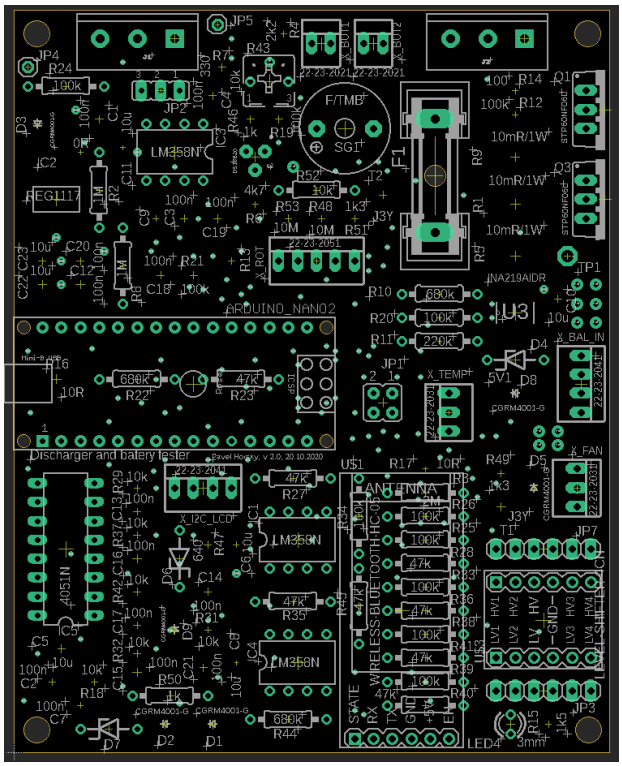
Obr. 18: Celá deska plošných spojů v reálné velikosti



Obr. 19: Spodní vrstva PCB v reálné velikosti



Obr. 20: Vrchní vrstva PCB v reálné velikosti



Obr. 21: Osazovací plán PCB v reálné velikosti

3.2 Software

Software je rozdělen na dvě části, na hlavní část software v testeru a na pomocný software v počítači. Program v počítači umožňuje vyčítat hodnoty ze zařízení a vytvořit z nich graf, případně nastavit hodnoty pro měření a spustit měření. Tester je schopen pracovat samostatně a zobrazovat výsledky na displeji, program v počítači pouze zlepšuje uživatelský komfort.

Program v testeru je psán ve vývojovém prostředí Arduino IDE v jazyce Wiring, který je postaven okolo C a C++. Program jsem vymýšlel sám, ale pro ovládání jednotlivých modulů jsem použil knihovny dostupné na internetu. Program se skládá ze dvou hlavních částí, nastavení vstupních údajů v menu a vlastního měření.

3.2.1 Struktura menu

Menu je navrženo tak, aby při standardních nastaveních bylo potřeba pouze otáčet a „od entrovávat“ encodérem. V případě špatného nastavení je zde tlačítko zpět. Pro nastavení jsem zvolil popis v angličtině.

- Prvním výběrem je výběr typu baterie nebo zdroje z možností LiIon, LiPo, LiFe, NiMh, NiCd, Pb, Source. Tento výběr probíhá otáčením rotačního encodéru doprava nebo doleva, než uživatel výběr potvrdí.

Obr. 22: Foto menu při výběru typu baterie

- Pokud uživatel vybere baterii LiIon nebo LiPo, tak dalším bodem je nastavení počtu sériově zapojených článků, které jsou potřeba znát pro program. Možnosti jsou tři a to 1 nebo 2 články v sérii. Tento výběr probíhá stejně jak ten první.

Obr. 23: Foto menu při výběru počtu článků v sérii

- Dalším, již společným nastavením je minimální napětí. Uživateli se zde vždy objeví předpokládané minimální napětí podle typu baterie a počtu článků. Krok, po kterém je možné nastavovat je zvolen na 0,1 V.

Obr. 24: Foto menu při nastavení minimálního napětí

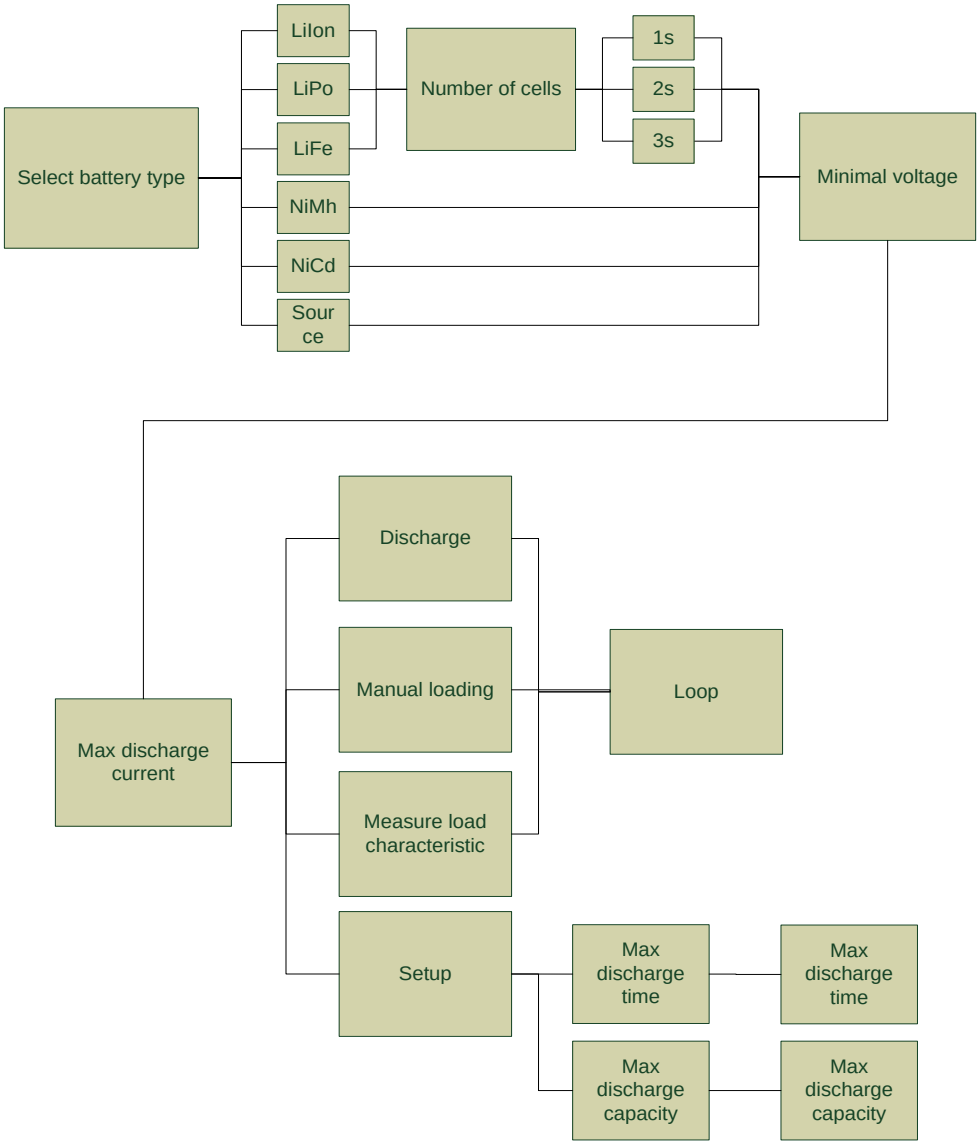
- Další je nastavení vybíjecího proudu, pro rychlejší nastavení se krok mění, u malých proudů je jemnější a u velkých je hrubší, protože se nepředpokládá, že by někdo chtěl třeba nastavovat jednotlivé miliampéry, když by chtěl vybíjet baterii 10 A.

Obr. 25: Foto menu při nastavení maximálního proudu

- Dále se vybírá mezi Discharge, Manual loading, Measure load characteristic a Setup.

Obr. 26: Foto menu při výběru mezi typy vybíjení

- V Setupu je možné upravit již předem nastavené hodnoty a to maximální dobu vybíjení a maximální kapacitu, kterou je nám schopna baterie dodat aniž by se poničila.



Obr. 27: Struktura menu

3.2.2 Měření charakteristik

Jsou zde volitelné 3 druhy vybíjení:

- **Discharge**
- **Manual loading**
- **Measure load characteristic.**

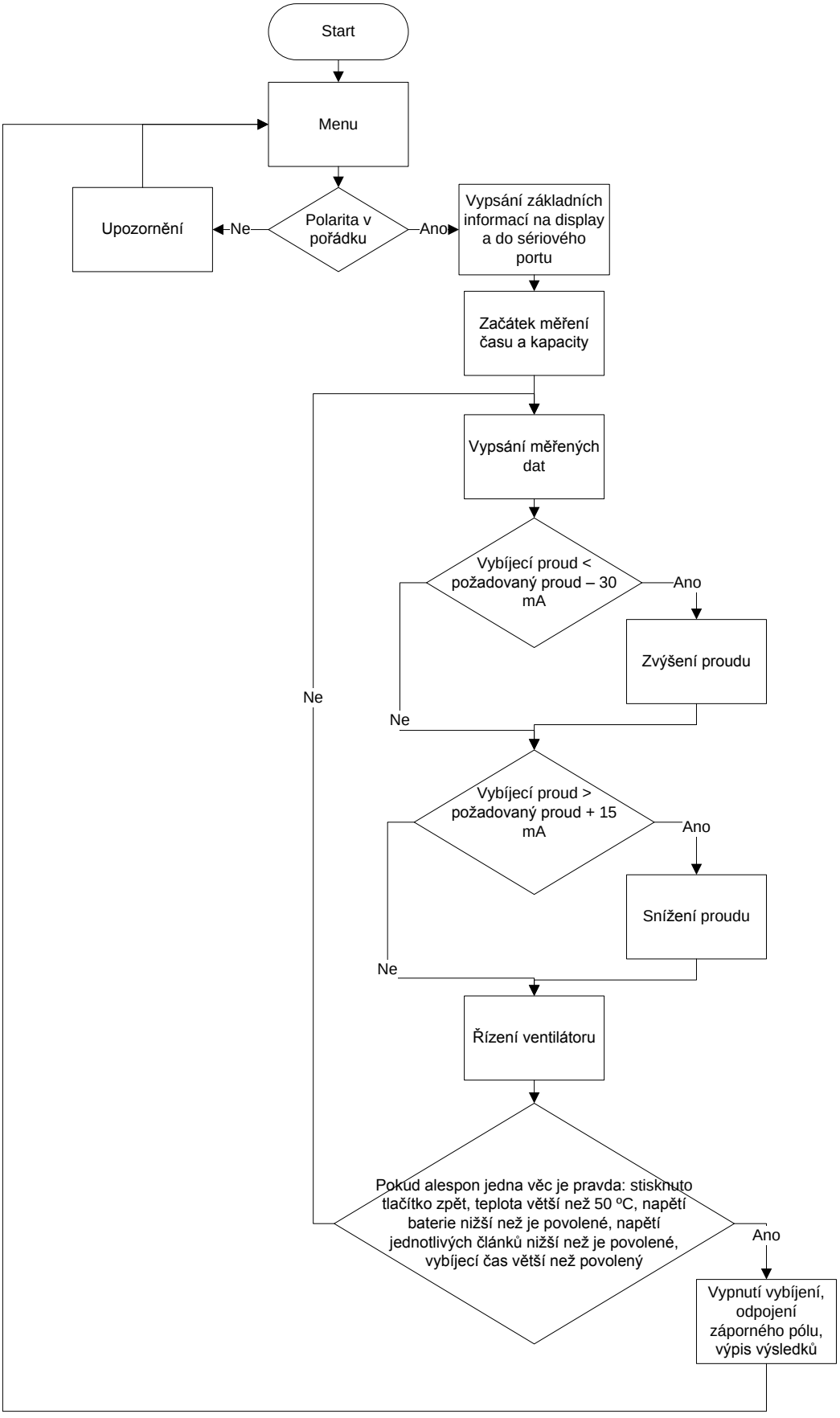
U **Discharge** se na displeji zobrazí DIS a postupně naběhne vybíjecí proud a tester bude baterii vybíjet. Po ukončení vybíjení vypíše, jaká kapacita byla vybita, při jakém napětí se vybíjení zastavilo a pokud během vybíjení běžel i program na počítači, tak se v počítači zobrazí graf. Níže můžete vidět vývojový diagram vybíjení.

Obr. 28: Foto při vybíjení se zobrazením času

Při více článcích v sérii se na místě času zobrazí střídavě napětí jednotlivých článků a hodiny

Obr. 29: Foto při vybíjení se zobrazením napětí na prvním článku

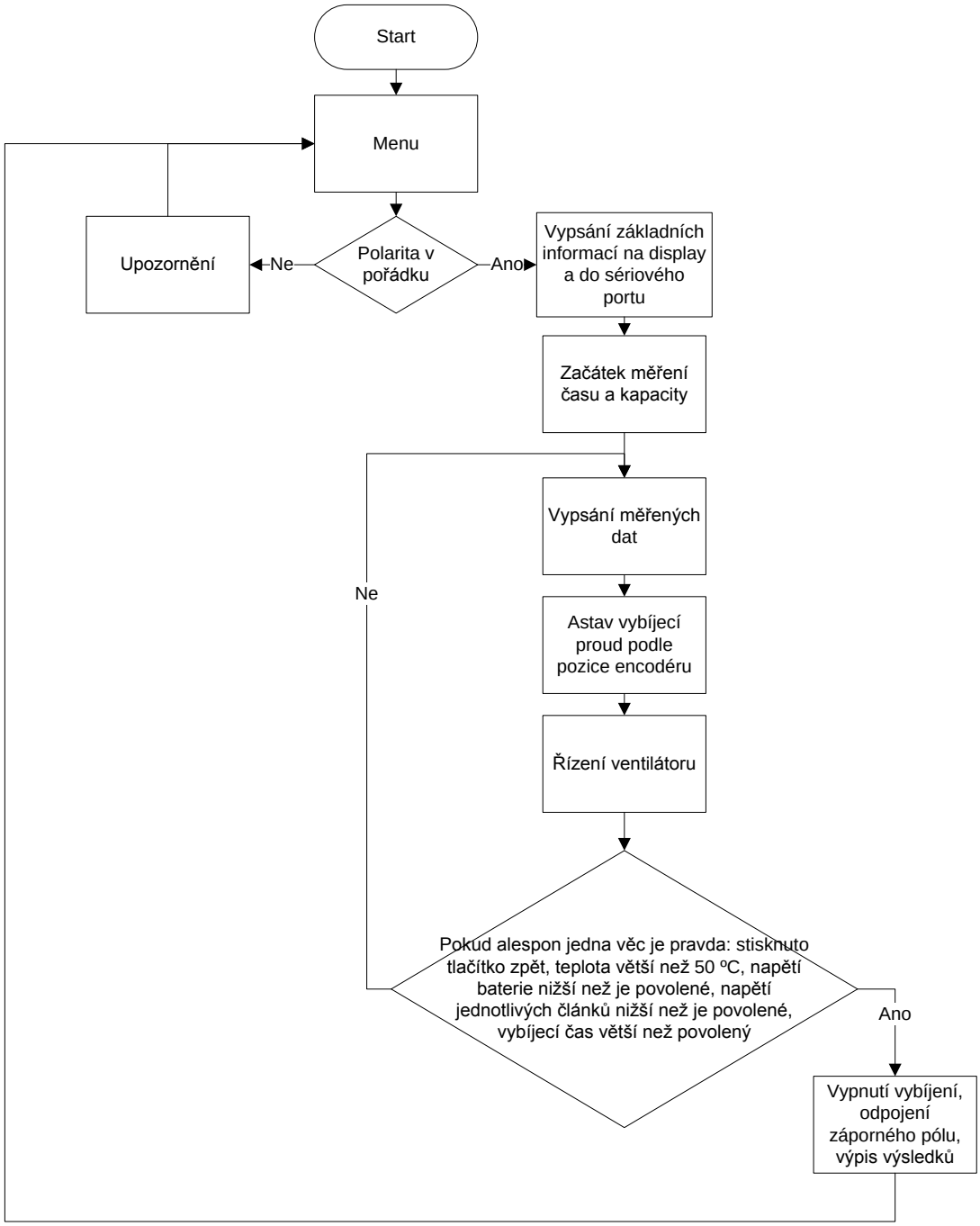
Obr. 30: Foto při vybíjení se zobrazením napětí na druhém článku



Obr. 31: Vývojový diagram programu při Discharge

U **Manual loading** se zobrazí na displeji MLD a uživateli se zobrazí napětí baterie a vybíjecí proud, který bude pomocí rotačního encodéru nastavovat.

Obr. 32: Foto výpisu na display při manual loading

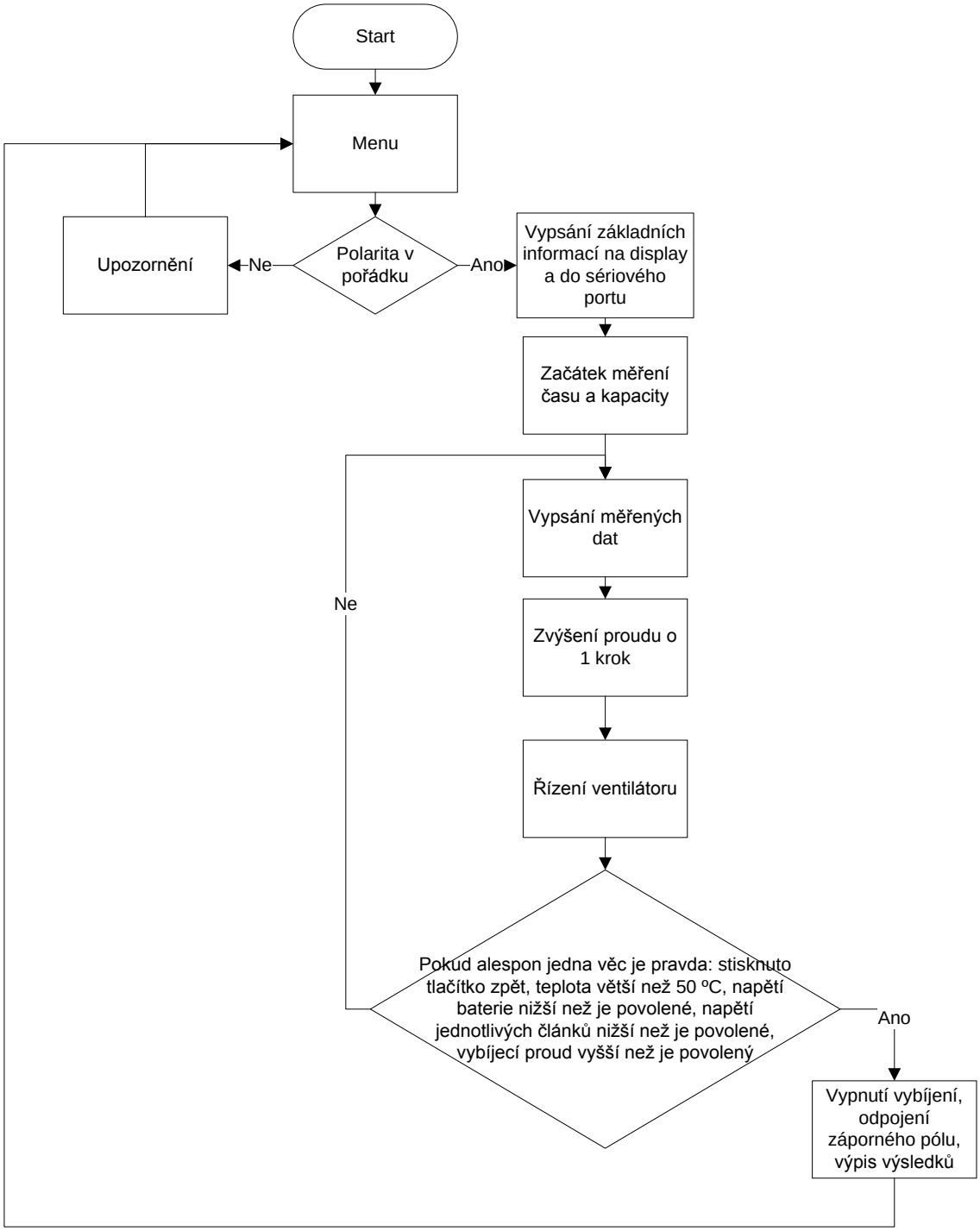


Obr. 33: Vývojový diagram programu při Manual loading

Pokud uživatel zvolí **Measure load characteristic**, tak se na displeji zobrazí LOA a tester automaticky postupně zvedá proud a odesílá informace do počítače. Při dosažení maximálního povoleného proudu nebo při poklesu napětí pod povolenou hodnotu se vybíjení ukončí, a na displeji se zobrazí vnitřní odpor baterie nebo zdroje v ohmech a zároveň se zobrazí graf v počítači.

Obr. 34: Zobrazení při zátěžové charakteristice

Obr. 35: Zobrazení při ukončení zátěžové charakteristiky



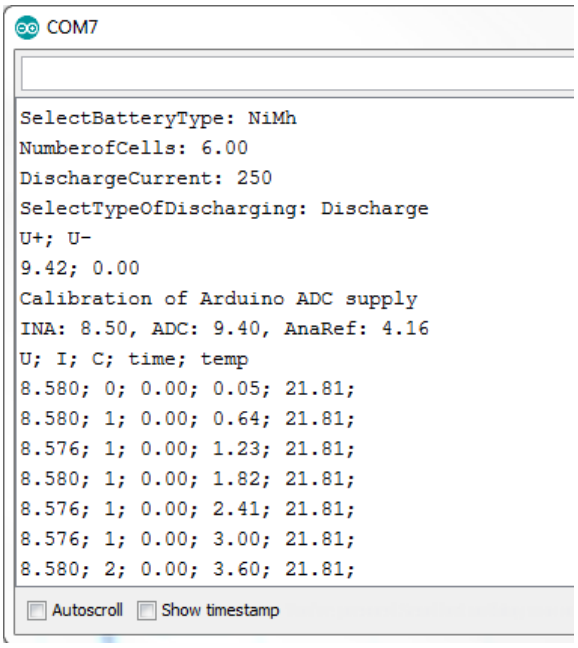
Obr. 36: Vývojový diagram při Measure Load Characteristic

3.2.3 Sériový výstup

Kromě zobrazení na displeji je použit sériový port přes USB, kterým jsou odesílány údaje o nastavení a změřená data v průběhu měření. Tato sériová data mohou být přijímána na počítači a dále zpracována. Z těchto dat je možné vynést vybíjecí charakteristiky nebo zátěžové charakteristiky.

Přijímat data ze sériového portu na PC je možné různými programy. Arduino IDE obsahuje monitor sériových dat, dále je možné použít třeba program PuTTY nebo napsat makro v Excelu, které bude data přijímat a rovnou vykreslovat grafy jak je uvedeno v následující kapitole.

Ukázka přijímaných dat je na obr. 37

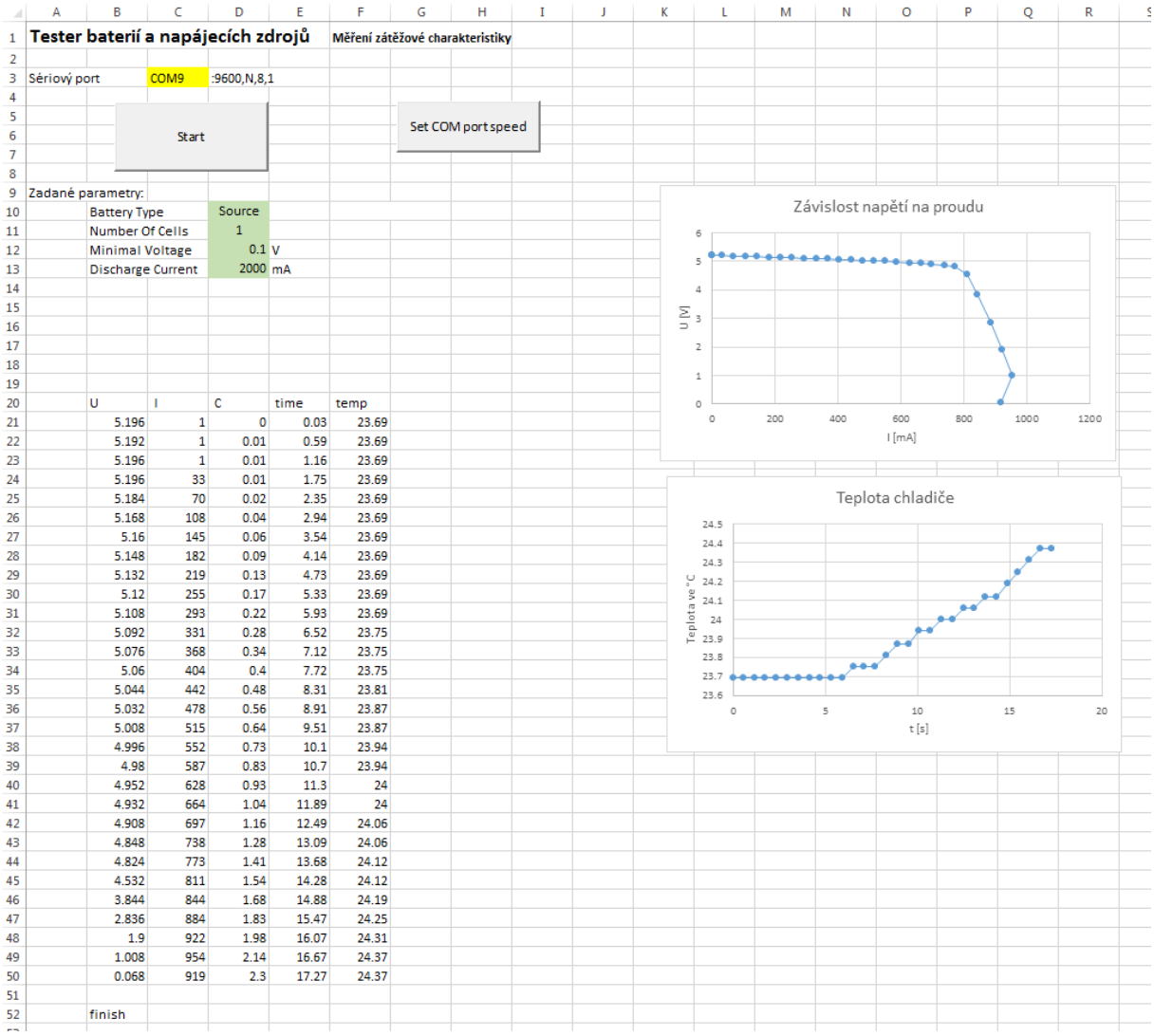


Obr. 37: Ukázka přijímaných dat vypsaných do počítače

3.2.4 Software v počítači

Pro zobrazení hodnot naměřených zařízením jsem napsal program v Excelu ve Visual Basic for Applications (VBA), který umožnuje zobrazit grafy z načtených hodnot nebo i nastavit parametry měření a spustit měření.

1. **Vyčítání hodnot**
   Během tohoto programu si Excel vyčítá ze sériového portu to, co se na testeru nastavilo v menu a následně vypisuje naměřené hodnoty a rovnou tvoří grafy. Tato část je na dvou listech v Excelu, a to jeden list pro zátěžovou charakteristiku a jeden list pro vybíjení, liší se pouze jedním grafem, a to tak že při zátěžové charakteristice je na ose x proud v mA a při vybíjení je na ose x čas v sekundách.
2. **Nastavení a vyčítání hodnot**
   Tato část je také na dvou listech, jeden pro zátěžovou charakteristiku a druhý pro vybíjení. Během tohoto programu Excel vyšle uživatelem nastavené dodnoty a typ měření z Excelu do testeru, ten je zpracuje a začne měřit. Dále to už probíhá stejně, Excel vyčítá hodnoty ze sériového portu, vypisuje je a kreslí grafy.

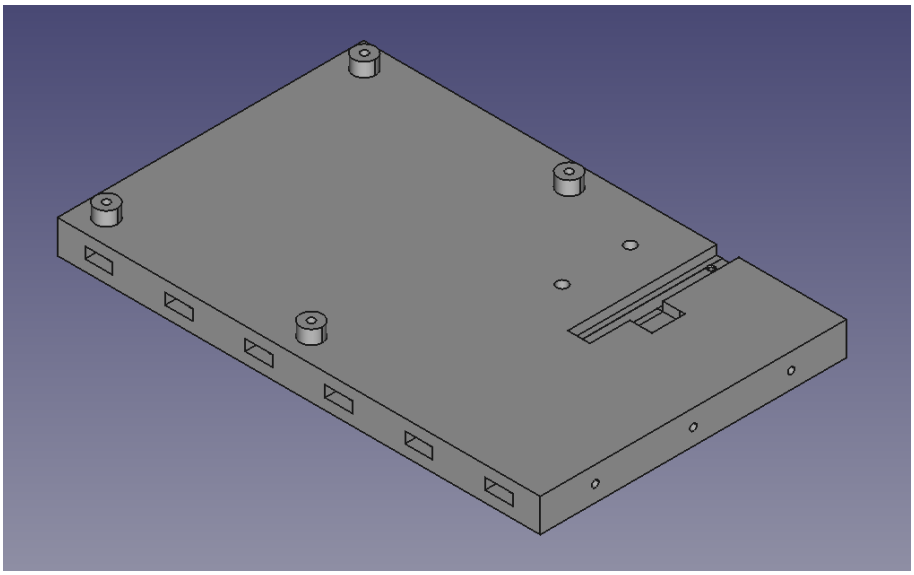


Obr. 38: Nastavení a vyčítání hodnot v Excelu

3.3 Mechanická konstrukce

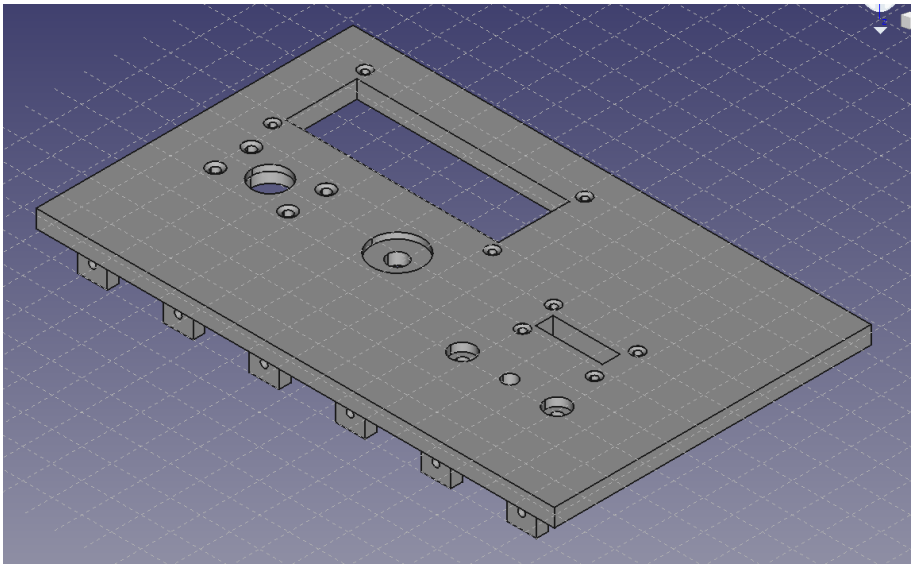
Protože je tester atypických rozměrů, navrhl jsem si pro něj krabičku ve FreeCadu na 3D tiskárnu. Na čelní stranu jsem přilepil zalaminovaný štítek.

Krabička se skládá ze čtyř částí, z podstavy, z přední a zadní strany a z krytu. Na podstavě je připevněna deska plošných spojů a chladič. K podstavě se zepředu a zezadu připevňuje přední a zadní deska vždy pěti šroubky zespodu, aby se pro každé odšroubování nemusel strhávat štítek a lepit nový.



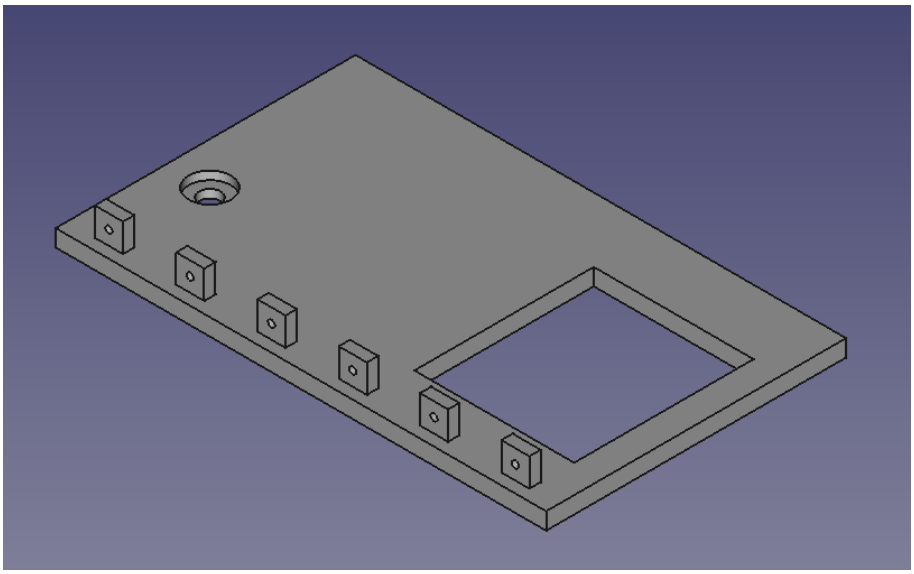
Obr. 39: Podstava krabičky

Na přední straně je navrhnuta úprava pro připevnění displeje, zpětného tlačítka, rotačního encodéru, balančních konektorů, banánků a LED diody. Všechny části jsou přišroubovány šroubky se zápustnými hlavičkami. Aby nebylo vidět nechtěné části, tak je to celé přelepeno samolepícím zalaminovaným štítkem, na kterém jsou popisky.



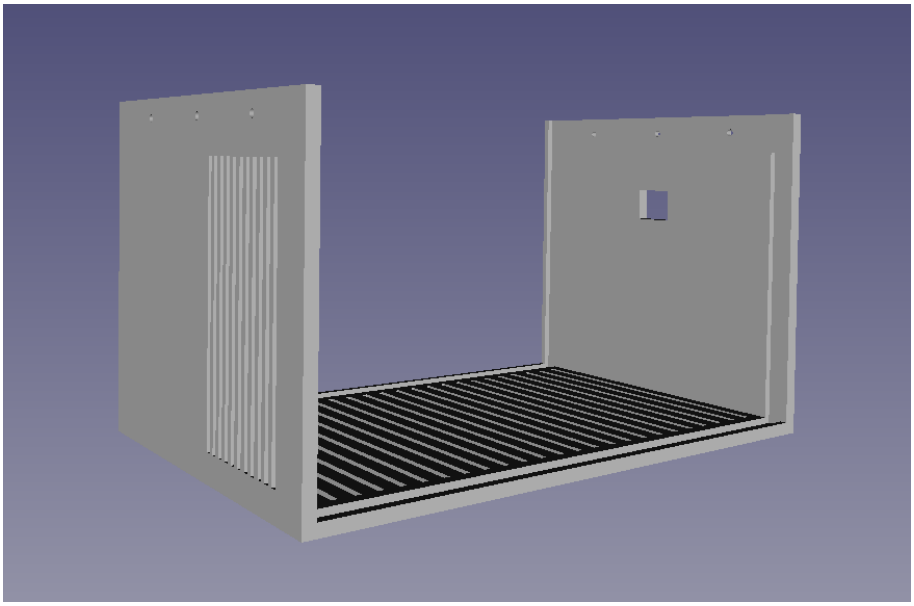
Obr. 40: Přední strana krabičky

Na zadní straně je velký otvor pro výduch chladiče a otvor pro napájecí konektor.



Obr. 41: Zadní strana krabičky

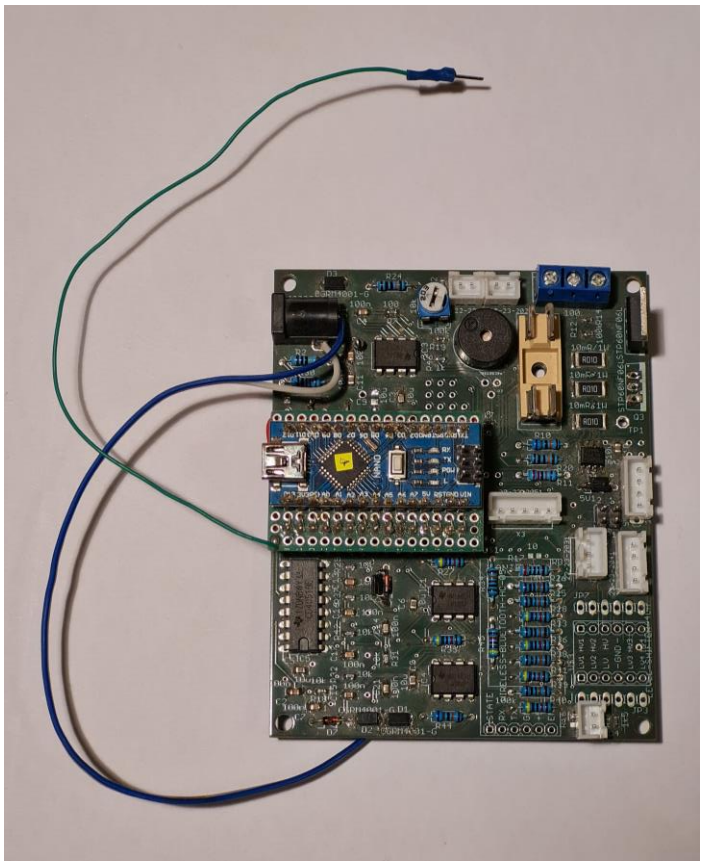
Vrchní kryt je v pravé straně dírkovaný, aby tudy chladič mohl nasávat vzduch. Tvarem dírek jsem se inspiroval u jednoho vysavače. Vlevo je otvor pro připojení sériového portu pro výpis do počítače. Okolo přední a zadní strany je drážka jednak pro zpevnění konstrukce, tak i pro ochranu přilepeného štítku, aby se jej nedalo snadno poškodit. Pro zpevnění konstrukce je vrchní strana krabičky zevnitř vlnitá.



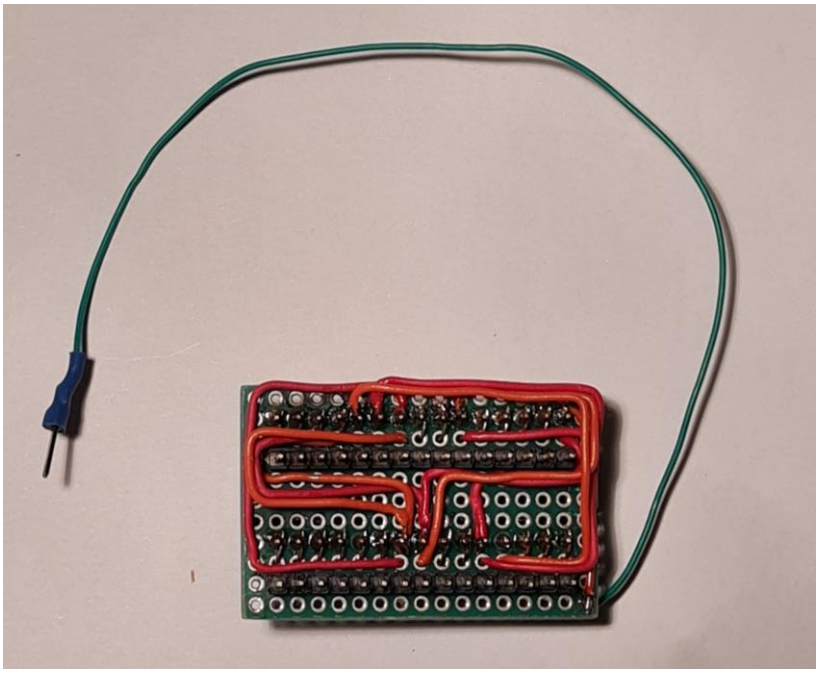
Obr. 42: Vrchní strana krabičky

3.4 Realizované zařízení

První varianta plošného spoje neměla správné připojení Arduina a musela obsahovat plošný spoj s drátovými propoji, je uvedena na obr. 43 a propojovací deska na obr. 44.



Obr. 43: PCB verze 1



Obr. 44: Upravené Arduino zespodu pro PCB verze 1

První pokusnou krabičkou byla krabička ze dřeva, ve které jsem si zkoušel rozmístění konektorů a tlačítek.

Obr. 45: Krabička verze 1 ze dřeva

Obr. 46: Krabička verze 1 ze dřeva, pohled zepředu

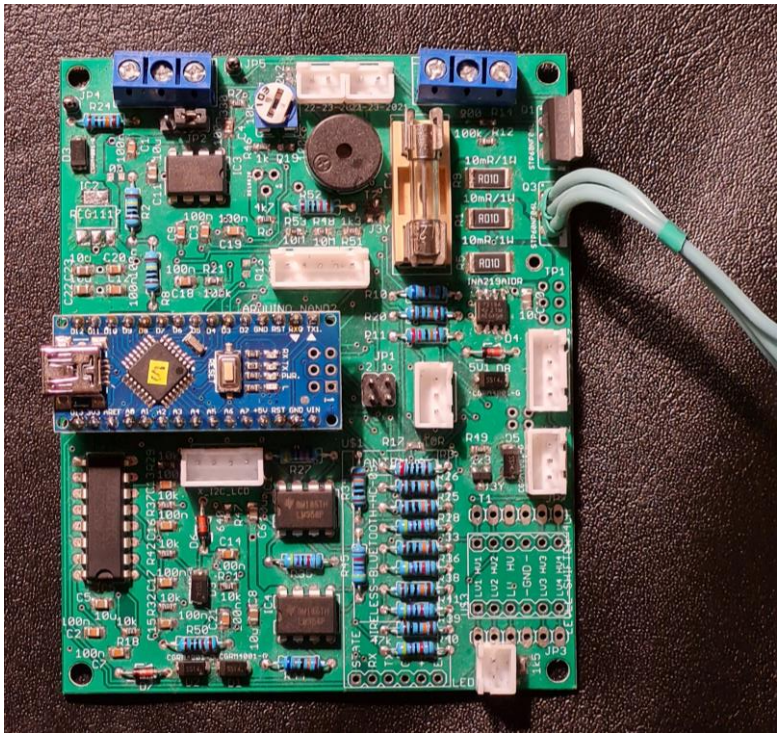
Na obrázku níže můžete vidět novou krabičku tištěnou na 3D tiskárně v pohledu zboku.

Obr. 47: Finální krabička testeru

Na následujícím obrázku můžete vidět přímý pohled.

Obr. 48: Pohled přímo na tester zepředu (konektor 3 cells není používán)

Obr. 49: Pohled dovnitř testeru



Obr. 50: PCB verze 2

3.5 Měření

3.5.1 Měření vlastností

3.5.1.1 Kontrola měření napětí

Metoda

Kalibraci funkce měření napětí jsem prováděl tak, že jsem k testeru připojil paralelně regulovatelný externí zdroj a multimetr Fluke. Teplota okolí byla (23 ± 5) °C.

Etalon

Použil jsem multimetr Fluke 289, který byl kalibrován v akreditované laboratoři 18. 10. 2020 s platností kalibrace do 17. 10. 2022. Specifikace multimetru Fluke, potvrzená kalibrací, je 0,025 % pro stejnosměrné napětí na rozsazích 5 a 50 V. Specifikace vyhovuje potřebám testeru. Kalibraci jsem provedl v devatenácti různých hodnotách napětí od 0,2 V do 30 V, což pokrývá rozsah měření testeru. Protože zařízení přepíná rozsah při 15 V, tak jsem okolo tohoto bodu měřil z obou stran co nejblíže.

Pro každý bod jsem provedl 10 měření, jak na referenčním multimetru, tak na kontrolovaném testeru.

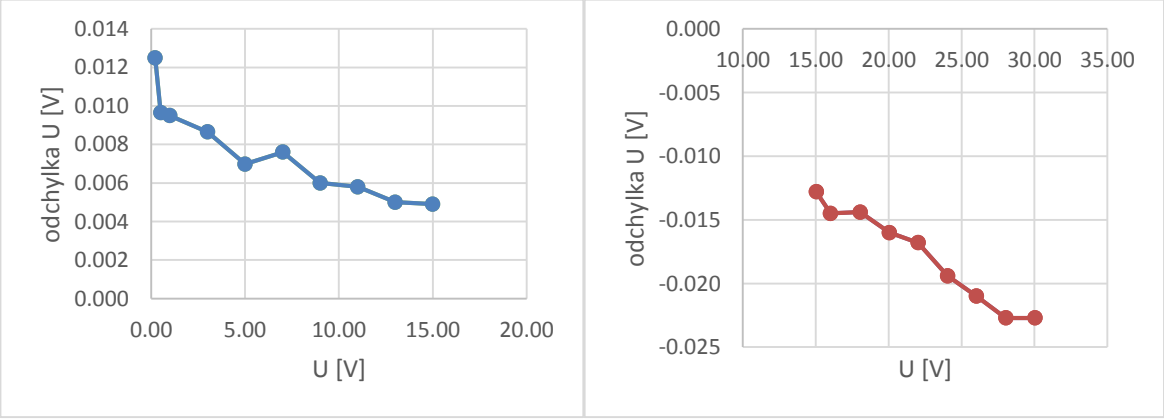
Tab. 3: Kontrola přesnosti měření INA 219 v rozsahu 16 V pomocí multimetru (DMM) Fluke 289, v. č. 44870089

zařízení	měření č.:	měření ve V									
DMM	1	0.2115	0.5096	1.0015	3.0122	5.0065	7.017	9.006	11.009	13.001	14.991
	2	0.2115	0.5096	1.0015	3.0123	5.0065	7.017	9.006	11.009	13.001	14.991
	3	0.2115	0.5096	1.0015	3.0124	5.0065	7.017	9.006	11.009	13.001	14.991
	4	0.2115	0.5096	1.0015	3.0125	5.0065	7.017	9.006	11.009	13.001	14.991
	5	0.2115	0.5097	1.0015	3.0126	5.0065	7.017	9.006	11.009	13.001	14.991
	6	0.2115	0.5097	1.0015	3.0127	5.0065	7.017	9.006	11.009	13.001	14.991
	7	0.2160	0.5097	1.0015	3.0128	5.0067	7.017	9.006	11.009	13.001	14.991
	8	0.2160	0.5097	1.0015	3.0129	5.0067	7.017	9.006	11.009	13.001	14.992
	9	0.2160	0.5097	1.0015	3.0130	5.0067	7.018	9.006	11.009	13.001	14.992
	10	0.2160	0.5097	1.0015	3.0131	5.0067	7.018	9.006	11.009	13.001	14.992
Tester	1	0.2000	0.5000	0.9920	3.0040	5.0000	7.008	9.000	11.004	12.996	14.984
	2	0.2000	0.5000	0.9920	3.0040	5.0000	7.008	9.000	11.004	12.996	14.988
	3	0.2000	0.5000	0.9920	3.0040	4.9960	7.012	9.000	11.000	12.996	14.984
	4	0.2000	0.5000	0.9920	3.0040	5.0000	7.012	9.000	11.004	12.996	14.988
	5	0.2040	0.5000	0.9920	3.0040	5.0000	7.012	9.000	11.004	12.996	14.984
	6	0.2000	0.5000	0.9920	3.0040	5.0000	7.012	9.000	11.004	12.996	14.988
	7	0.2000	0.5000	0.9920	3.0040	5.0000	7.008	9.000	11.004	12.996	14.984
	8	0.2000	0.5000	0.9920	3.0040	5.0000	7.008	9.000	11.004	12.996	14.988
	9	0.2000	0.5000	0.9920	3.0040	5.0000	7.008	9.000	11.000	12.996	14.988
	10	0.2040	0.5000	0.9920	3.0040	5.0000	7.008	9.000	11.004	12.996	14.988
DMM	průměr	0.2133	0.5097	1.0015	3.0127	5.0066	7.017	9.006	11.009	13.001	14.991
Tester	průměr	0.2008	0.5000	0.9920	3.0040	4.9996	7.010	9.000	11.003	12.996	14.986
Odchylka		-0.013	-0.010	-0.009	-0.009	-0.007	-0.008	-0.006	-0.006	-0.005	-0.005

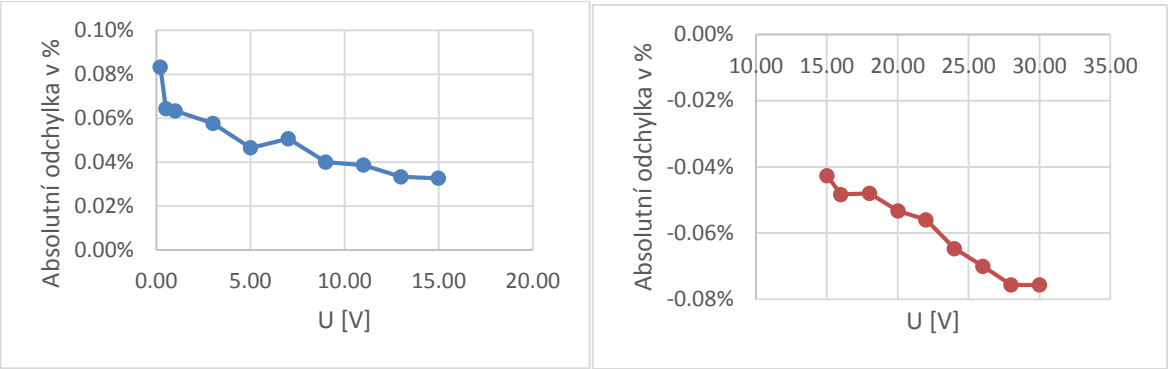
Tab. 4: Kontrola přesnosti měření INA 219 v rozsahu 32 V pomocí multimetru (DMM) Fluke 289, v. č. 44870089

zařízení	měření č.:	měření ve V								
DMM	1	15.030	15.999	18.025	20.032	22.000	24.015	26.007	28.004	30.015
	2	15.030	15.999	18.025	20.032	22.000	24.015	26.007	28.004	30.015
	3	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.016
	4	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.017
	5	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.017
	6	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.017
	7	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.017
	8	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.017
	9	15.030	15.999	18.027	20.032	22.000	24.015	26.007	28.005	30.017
	10	15.030	16.000	18.027	20.032	22.000	24.015	26.007	28.004	30.017
Tester	1	15.044	16.012	18.040	20.048	22.016	24.032	26.028	28.024	30.040
	2	15.040	16.012	18.044	20.048	22.016	24.036	26.028	28.024	30.040
	3	15.044	16.016	18.040	20.048	22.016	24.032	26.028	28.024	30.040
	4	15.044	16.016	18.036	20.048	22.020	24.036	26.028	28.028	30.036
	5	15.044	16.012	18.044	20.048	22.020	24.036	26.028	28.028	30.036
	6	15.040	16.012	18.040	20.048	22.016	24.036	26.028	28.028	30.036
	7	15.044	16.012	18.040	20.048	22.016	24.032	26.028	28.028	30.036
	8	15.044	16.012	18.040	20.048	22.016	24.036	26.028	28.028	30.040
	9	15.044	16.016	18.040	20.048	22.016	24.032	26.028	28.028	30.044
	10	15.040	16.016	18.040	20.048	22.016	24.036	26.028	28.028	30.044
DMM	průměr	15.030	15.999	18.026	20.032	22.000	24.015	26.007	28.004	30.017
Tester	průměr	15.043	16.014	18.040	20.048	22.017	24.034	26.028	28.027	30.039
Odchylka		0.013	0.014	0.014	0.016	0.017	0.019	0.021	0.023	0.023

Z vypočítaných odchylek jsem zjistil, že tester má velkou reservu přesnosti a proto není potřeba zavádět korekce v programu.
V grafu s odchylkou je vidět, že při změně rozsahu se naměřená odchylka měření změní, ale je to pouze v zanedbatelné hodnotě v setinách voltu.



Obr. 51: Grafy odchylek napětí ve voltech měřený INA 219, modře pro rozsah do 16V a hnědě pro rozsah do 32V

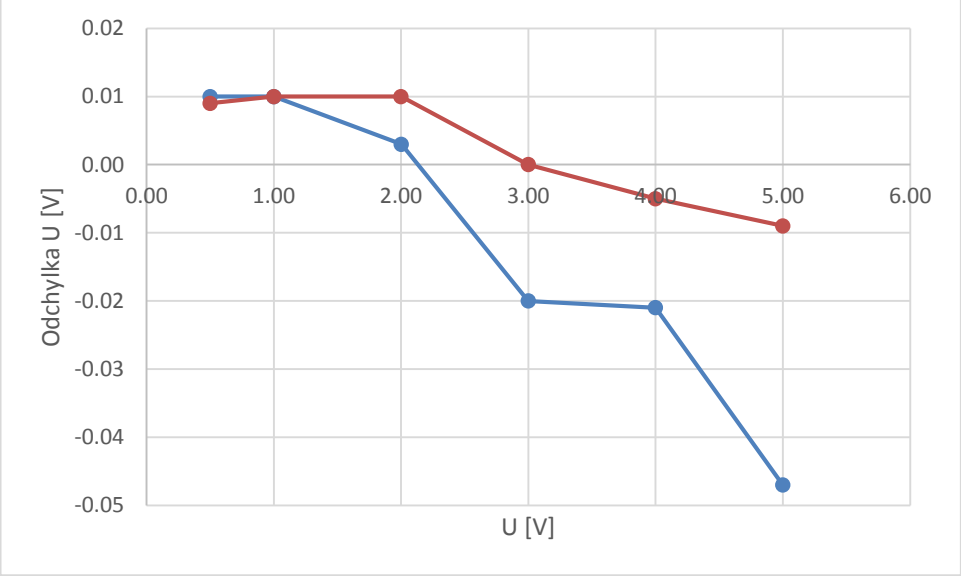


Obr. 52: Grafy absolutních odchylek napětí v procentech měřený INA 219, modře pro rozsah do 16V a hnědě pro rozsah do 32V

Napětí na jednotlivých článcích je měřeno pomocí AD převodníku v procesoru a jako reference je použito napájecí napětí procesoru. Při napájení Arduina z USB je napájecí napětí oddělené Schottkyho diodou a není moc přesné. Abych docílil větší přesnosti měření, tak vždy nejprve v programu změřím napětí celé baterie AD převodníkem procesoru a porovnám jej s přesným napětím z INA219 a vypočítám si odchylku, kterou použiji při měření článků. Protože pro každý článek je jiný obvod a zesílení je ovlivněno tolerancí použitých rezistorů, tak napětí ještě vynásobím druhým koeficientem, který je rozdílný pro každý článek.

Protože mám k disposici pouze baterie složené ze dvou článků v sérii, kalibroval jsem pouze pro toto zapojení. K balančnímu konektoru jsem si připojil dva zdroje a dva multimetry a měřil napětí.

Z grafu obr 52 je vidět, že odchylka měření napětí na jednotlivém článku v nejpoužívanějším rozsahu (1 až 4 V) je pro dané užití zanedbatelná.



Obr. 53: Graf odchylek napětí článku 1 modře a článku 2 hnědě

Tab. 5: Tabulka napětí na prvním článku porovnaná s multimetrem

zařízení	měření č.:	měření ve V					
DMM1	1	0.50	1.00	2.00	3.00	4.00	5.00
	2	0.50	1.00	2.00	3.00	4.00	5.00
	3	0.50	1.00	2.00	3.00	4.00	5.00
	4	0.50	1.00	2.01	3.00	4.00	5.00
	5	0.50	1.00	2.00	3.00	4.00	5.00
	6	0.50	1.00	2.00	3.00	4.00	5.00
	7	0.50	1.00	2.00	3.00	4.00	5.00
	8	0.50	1.00	2.00	3.00	4.00	5.00
	9	0.50	1.00	2.01	3.00	4.00	5.00
	10	0.50	1.00	2.01	3.00	4.00	5.00
Tester	1	0.49	0.99	2.00	3.02	4.02	5.04
	2	0.49	0.99	2.00	3.02	4.02	5.05
	3	0.49	0.99	2.00	3.02	4.02	5.05
	4	0.49	0.99	2.00	3.02	4.02	5.05
	5	0.49	0.99	2.00	3.02	4.02	5.05
	6	0.49	0.99	2.00	3.02	4.03	5.04
	7	0.49	0.99	2.00	3.02	4.02	5.04
	8	0.49	0.99	2.00	3.02	4.02	5.05
	9	0.49	0.99	2.00	3.02	4.02	5.05
	10	0.49	0.99	2.00	3.02	4.02	5.05
DMM1	průměr	0.50	1.00	2.00	3.00	4.00	5.00
Tester	průměr	0.49	0.99	2.00	3.02	4.02	5.05
Odchylka		-0.01	-0.01	0.00	0.02	0.02	0.05

Tab. 6: Tabulka napětí na druhém článku porovnaná s multimetrem

zařízení	měření č.:	měření ve V					
DMM2	1.00	0.50	1.00	2.00	3.00	4.00	5.00
	2.00	0.50	1.00	2.00	3.00	4.00	5.00
	3.00	0.50	1.00	2.00	3.00	4.00	5.00
	4.00	0.50	1.00	2.00	3.00	4.00	5.00
	5.00	0.50	1.00	2.00	3.00	4.00	5.00
	6.00	0.50	1.00	2.00	3.00	4.00	5.00
	7.00	0.50	1.00	2.00	3.00	4.00	5.00
	8.00	0.50	1.00	2.00	3.00	4.00	5.00
	9.00	0.50	1.00	2.00	3.00	4.00	5.00
	10.00	0.50	1.00	2.00	3.00	4.00	5.00
Tester	1.00	0.49	0.99	1.99	3.00	4.00	5.01
	2.00	0.50	0.99	1.99	3.00	4.00	5.01
	3.00	0.49	0.99	1.99	3.00	4.01	5.01
	4.00	0.49	0.99	1.99	3.00	4.01	5.01
	5.00	0.49	0.99	1.99	3.00	4.00	5.01

	6.00	0.49	0.99	1.99	3.00	4.00	5.01
	7.00	0.49	0.99	1.99	3.00	4.01	5.00
	8.00	0.49	0.99	1.99	3.00	4.00	5.01
	9.00	0.49	0.99	1.99	3.00	4.01	5.01
	10.00	0.49	0.99	1.99	3.00	4.01	5.01
DMM2	průměr	0.50	1.00	2.00	3.00	4.00	5.00
tester	průměr	0.49	0.99	1.99	3.00	4.01	5.01
Odchylka		-0.01	-0.01	-0.01	0.00	0.00	0.01

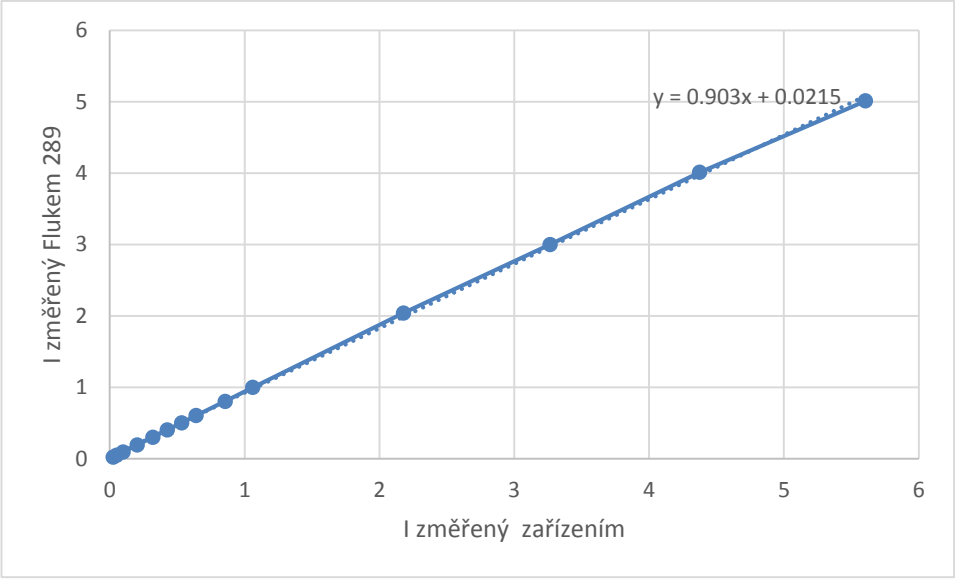
3.5.1.2 Proud

Kalibraci měření proudu jsem prováděl tak, že jsem testeru připojil do série baterii a multimetr Fluke 289. Použil jsem baterii, protože nemám zdroj, který by byl schopen dodat větší proudy. Nejprve jsem si proměřil proudy od 0 do 5 A. Z toho jsem si vynesl v Excelu graf.

Teplota okolí byla (23 ± 2) °C.

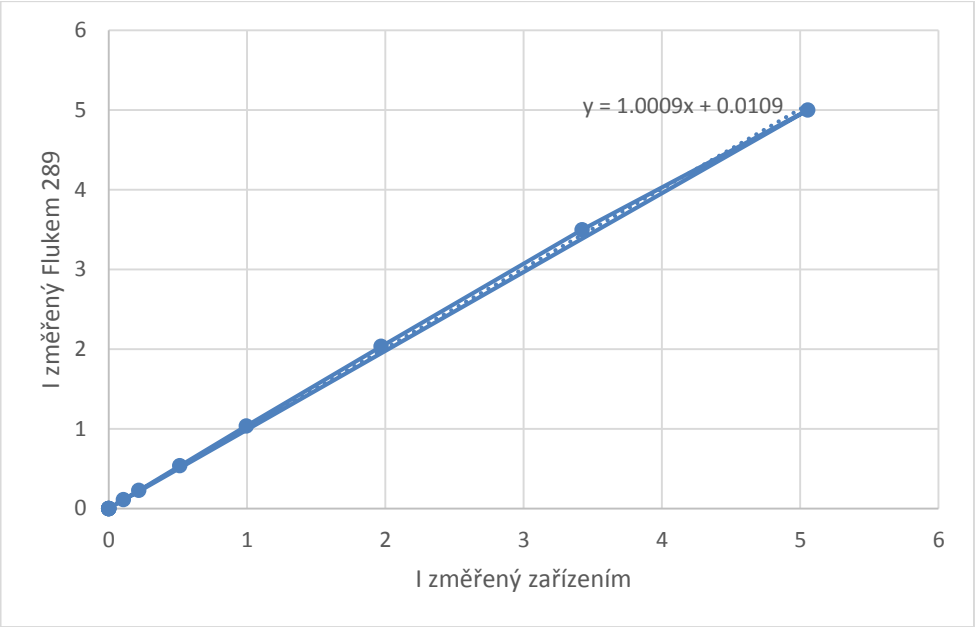
Etalon

Použil jsem multimetr Fluke 289, který byl kalibrován v akreditované laboratoři 18. 10. 2020 s platností kalibrace do 17. 10. 2022. Specifikace multimetru Fluke je 0,15 % pro DC I. Specifikace potřebám vyhoví.



Obr. 54: Graf s odchylkou proudu změřeného testerem s DDM před dostavením v programu

Protože odchylka byla téměř lineární, použil jsem korekci v programu. Odchylka je způsobená převážně tolerancí použitého bočníku. Následně jsem měření opakoval a již tester měřil téměř přesně.



Obr. 55: Graf s odchylkou proudu změřeného testerem s MDD

3.5.1.3 Kapacita baterie

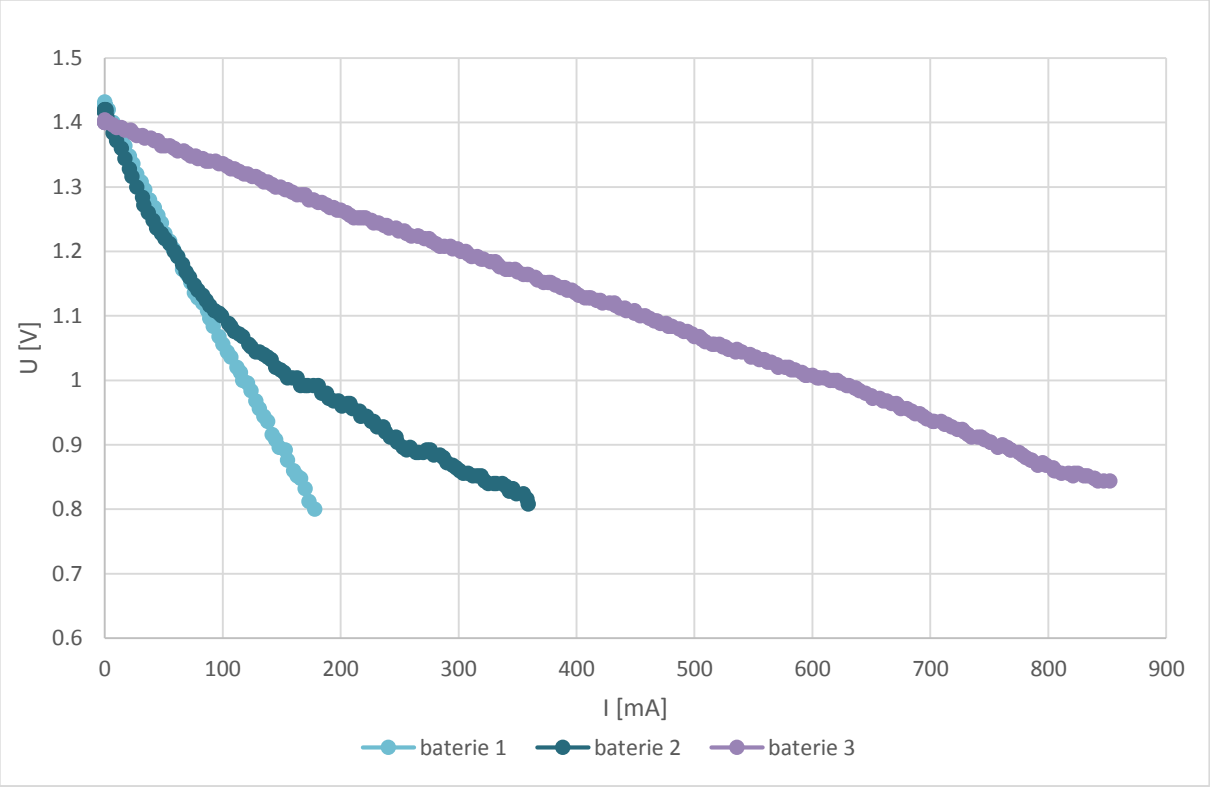
Pro kontrolu správného měření kapacity baterie jsem si k tester připojil baterii a při manual loading jsem si nastavil konstantní vybíjecí proud a změřil čas, za jaký se určitá kapacita vybila. Z výpočtu jsem si zjistil, že tester měří s reservou k předpokládané specifikaci a vzhledem k užití přesně.

3.5.2 Měření zátěžové charakteristiky baterií

V rámci validace bylo provedeno kontrolní měření. Pro měření AA baterií jsem si k testeru připojil držák na AA baterie a provedl několik měření.

Obr. 56: Připojená AA baterie k testeru při měření zátěžové charakteristiky

Pro měření byl nastaven maximální proud 850 mA a minimální napětí 0,8 V. Tester postupně automaticky zvyšoval proud až do maximální hodnoty, pokud nepokleslo napětí na baterii pod 0,8 V. Během měření tester odesílal změřené napětí na baterii a změřený zatěžovací proud. Výsledky měření můžete vidět v grafu na obr. 57. Záměrně byly vybrány staré baterie (nejméně 10 až 20 let), aby se každá chovala jinak.

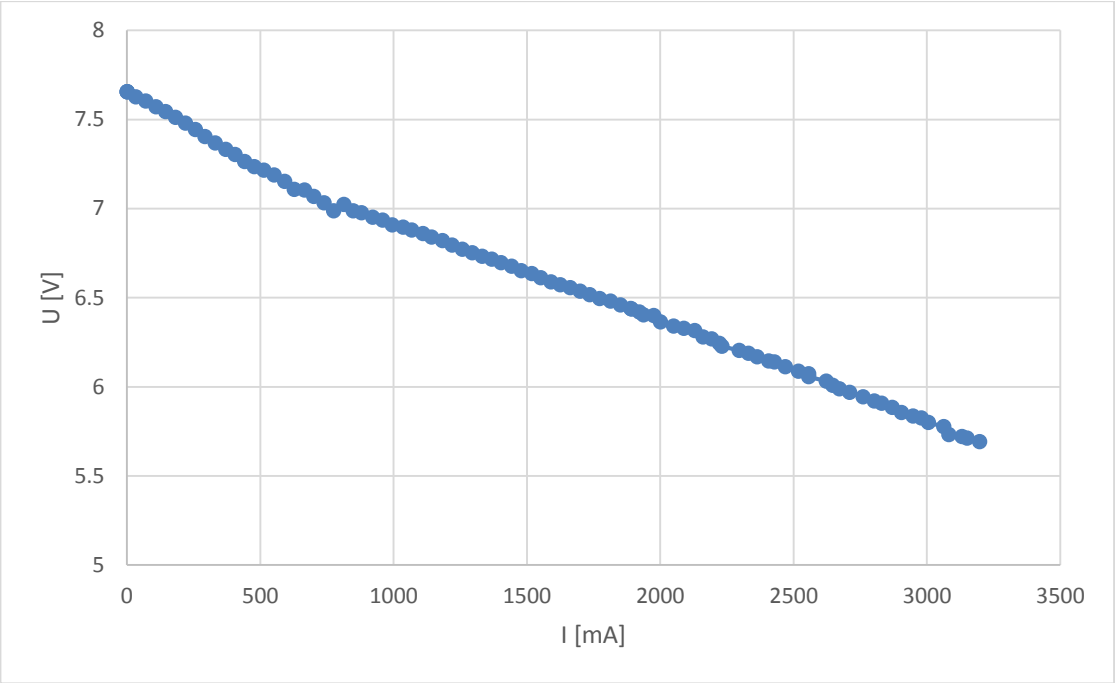


Obr. 57: Zatěžovací charakteristiky 3 různých AA baterií

Obr. 58: Měřené AA baterie, očíslovány 1 až 3 zleva doprava

3.5.3 Měření zátěžové charakteristiky LiIon baterie

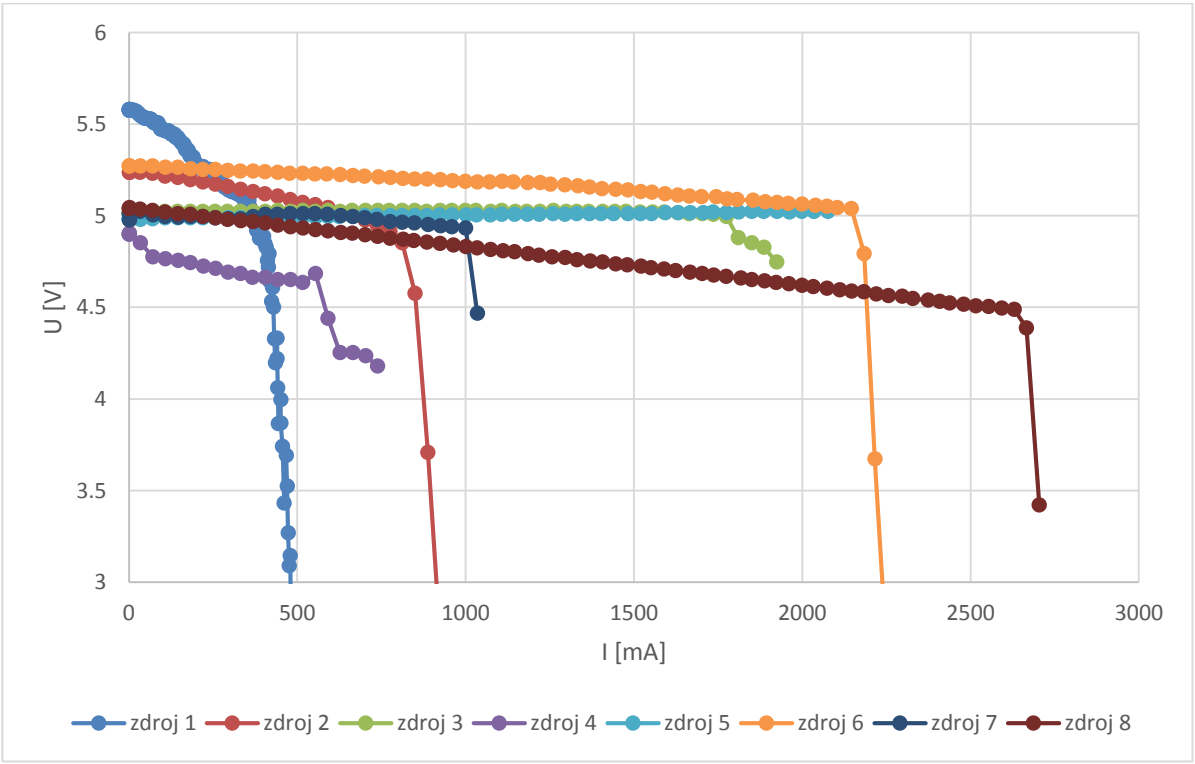
Změřil jsem charakteristiku LiIon baterie ze 4 článků Samsung ICR18650, 2 dvoj-články v sérii, která je na obr. 63. Graf volt-ampérové charakteristiky můžete vidět níže na obr. 59.



Obr. 59: Zatěžovací charakteristika LiIon baterie

3.5.4 Měření zátěžové charakteristiky zdrojů

Provedl jsem měření zátěžových charakteristik osmi 5V zdrojů (USB nabíječek pro mobilní telefony). Z grafu níže můžete vidět změřené zatěžovací charakteristiky. Měření ukázalo, že ne všechny zdroje jsou schopny dodat výrobcem udaný proud, ale naopak některé byly výkonnější.



Obr. 60: Zatěžovací charakteristiky napájecích zdrojů pro mobily

Zdroj 1 až 8 v pořadí zleva doprava:

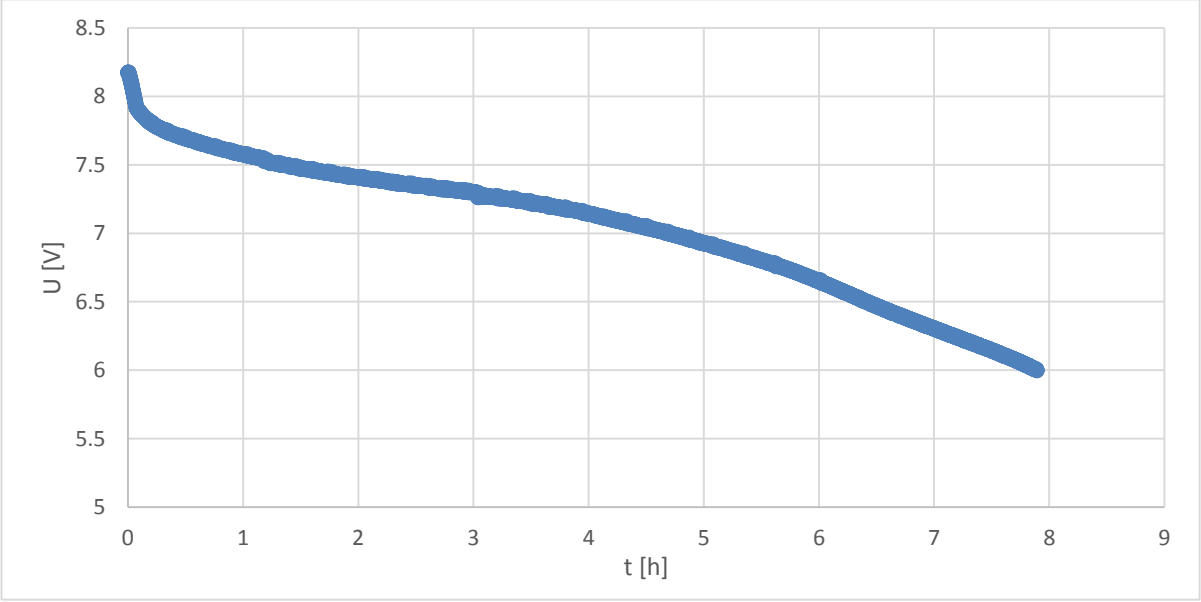
Obr. 61: Změřené zdroje od mobilů

Tab. 7: Tabulka specifikací zdrojů porovnaná s výsledky

zdroj	Výrobce	U (V)	I (A)	změřený proud (A)
1	Courier	5	0.5	0,37
2	neudán	5	2	0,77
3	Samsung	5	2	1,78
4	Samsung	5	1	0,55
5	Samsung	5	2	2,10
6	Chicony	5,35	2	2,15
7	Pengisi Daya	5	1	1,00
8	Baseus	5	2,4	2,60

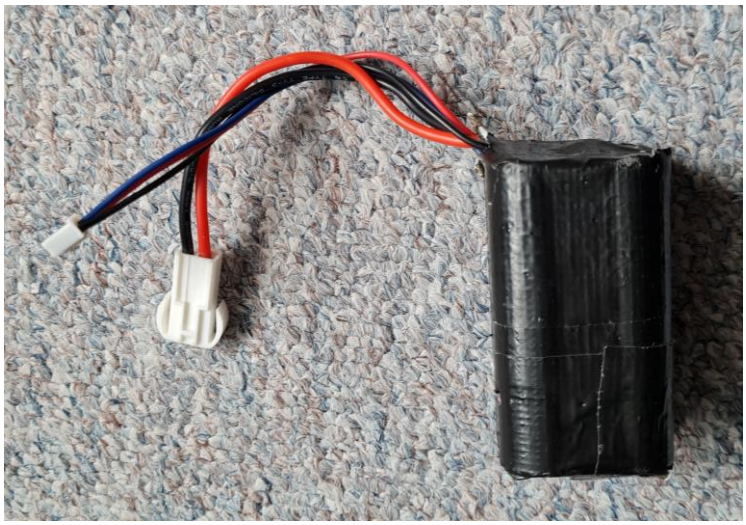
3.5.5 Měření kapacity baterie LiIon

Na grafu níže můžete vidět změřenou závislost napětí baterie (obr. 63) na čase.



Obr. 62: Graf závislosti napětí na čase LiIon baterie

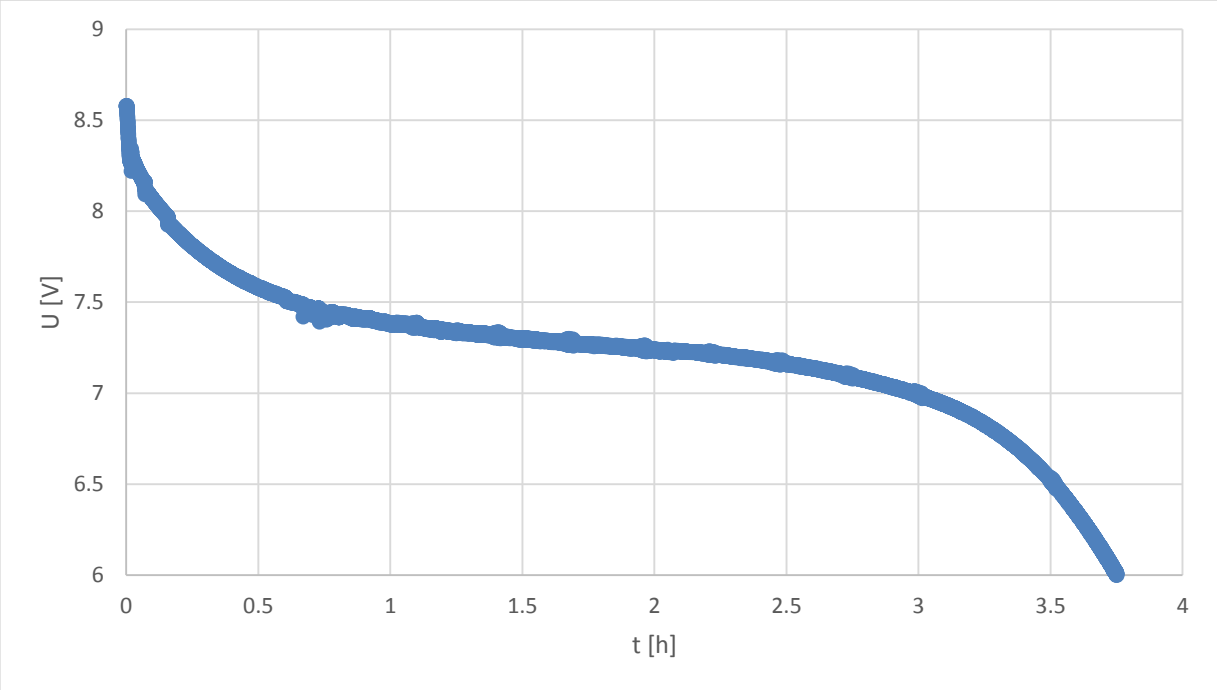
Dvoj-článková Lion baterie na obr 62 má podle měření skutečnou naměřenou kapacitu 3200 mAh místo původních 5200 mAh podle specifikace.



Obr. 63: Měřená LiIon baterie pro RC modely složená ze čtyř článků

3.5.6 Měření kapacity NiMh baterie

Níže můžete vidět změřenou napěťovou závislost na čase při vybíjení NiMh baterie. Šesti-článková NiMh baterie na obr 64 má podle měření skutečnou naměřenou kapacitu 940 mAh místo specifikovaných 900 mAh.



Obr. 64: Graf závislosti napětí na čase NiMh baterie

Obr. 65: NiMh baterie pro RC modely

4 ZÁVĚR

Ve své práci jsem se seznámil s různými typy akumulátorů a jejich vlastnostmi. Na základě těchto znalostí jsem navrhl a zkonstruoval tester akumulátorů a zdrojů řízený mikroprocesorem Arduino a napsal řídící SW pro ovládání testeru. Realizovaný tester jsem funkčně odzkoušel, zkalibroval a změřil zatěžovací charakteristiky několika typů běžně dostupných článků a zdrojů.

V testeru použitý měřicí integrovaný obvod INA219 se osvědčil a pracuje s dostačující přesností pro měření napětí a proudu v užité aplikaci. Velikost chladiče byla správně odhadnuta pro použití vybíjení běžných domácích akumulátorů. Změřené zátěžové charakteristiky odpovídají teoretickým předpokladům.

Na mou práci lze navázat rozšířením o ovládání z počítače, ale také vytvoření Android aplikace pro ovládání a vynášení grafu do mobilního telefonu přes bluetooth.

5 LITERATURA

[1] https://technoluxpro.com/cs/batarejki/vidy.html
[2] https://cs.wikipedia.org/wiki/Akumul%C3%A1tor
[3] https://cs.wikipedia.org/wiki/Olov%C4%9Bn%C3%BD_akumul%C3%A1tor
[4] https://www.elnika.cz/cz/podpora/informace-akumulatorech/kompletni-specifikace-nabijeni-vybijeni-skladovani-konstrukce/
[5] https://cs.wikipedia.org/wiki/Nikl-kadmiov%C3%BD_akumul%C3%A1tor
[6] https://www.velofiala.cz/n/co-je-pametovy-efekt-povida-se-ze
[7] https://www.researchgate.net/figure/Characteristics-of-Ni-based-batteries-a-Ni-Cd-cells-voltage-versus-discharge-and_fig2_224225487
[8] https://cs.wikipedia.org/wiki/Nikl-metal_hydridov%C3%BD_akumul%C3%A1tor
[9] https://lygte-info.dk/review/batteries2012/Eneloop%20AA%20BK-3MCCE%201900mAh%20(White)%202019%20UK.html
[10] https://cs.wikipedia.org/wiki/Lithium-iontov%C3%BD_akumul%C3%A1tor
[11] https://batteryuniversity.com/article/bu-501a-discharge-characteristics-of-li-ion
[12] https://cs.wikipedia.org/wiki/Lithium-polymerov%C3%BD_akumul%C3%A1tor
[13] https://www.dnkpower.com/lithium-polymer-battery-guide/
[14] https://www.abctech.cz/default.asp?show=wm&wmpart=article&wmaid=87
[15] http://sustainableskies.org/dr-eric-darcy-building-better-batteries/typical-lifepo4-discharge-curve/
[16] https://www.theengineeringprojects.com/2019/10/what-is-the-voltage-source.html
[17] https://cs.wikipedia.org/wiki/Arduino
[18] https://www.alza.cz/arduino-nano-v3-0-d569054.htm
[19] https://navody.dratek.cz/navody-k-produktum/proudovy-snimac-ina219.html
[20] https://www.ti.com/lit/ds/symlink/ina219.pdf?ts=1642047271902
[21] https://www.ti.com/lit/ds/symlink/cd74hc4051.pdf?HQS=dis-mous-null-mousermode-dsf-pf-null-wwe&ts=1642110379982
[22] https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf
[23] https://datasheetspdf.com/pdf/546199/STMicroelectronics/P60NF06/1

6 SEZNAM OBRÁZKŮ

Obr. 1: Tvary baterií [1] 10
Obr. 2: Vybíjecí charakteristika olověného akumulátoru [4] 11
Obr. 3: Vybíjecí charakteristika NiCd článku [6] 11
Obr. 4: Vybíjecí charakteristika NiMh a NiCd článku [7] 12
Obr. 5: Vybíjecí charakteristika NiMh článku [9] 12
Obr. 6: Vybíjecí charakteristika LiIon článku [11] 13
Obr. 7: Vybíjecí charakteristika LiPo článku [13] 13
Obr. 8: Vybíjecí charakteristika LiFe článku [15] 14
Obr. 9: Blokové schéma zapojení 17
Obr. 10: Schéma zapojení měření napětí a proudu s integrovaným obvodem INA219 18
Obr. 11: Zjednodušené zapojení obvodu elektronické zátěže 19
Obr. 12: Zapojení obvodu elektronické zátěže 19
Obr. 13: Schéma zapojení ochrany proti přepólování 20
Obr. 14: Schéma zapojení obvodu pro měření úbytku napětí na ochranném spínači 20
Obr. 15: Schéma zapojení obvodu pro měření napětí na jednom článku 21
Obr. 16: Schéma zapojení snímání teploty na chladiči 22
Obr. 17: Celkové schéma zapojení 22
Obr. 18: Celá deska plošných spojů v reálné velikosti 23
Obr. 19: Spodní vrstva PCB v reálné velikosti 23
Obr. 20: Vrchní vrstva PCB v reálné velikosti 24
Obr. 21: Osazovací plán PCB v reálné velikosti 24
Obr. 22: Foto menu při výběru typu baterie 25
Obr. 23: Foto menu při výběru počtu článků v sérii 25
Obr. 24: Foto menu při nastavení minimálního napětí 25
Obr. 25: Foto menu při nastavení maximálního proudu 26
Obr. 26: Foto menu při výběru mezi typy vybíjení 26
Obr. 27: Struktura menu 27
Obr. 28: Foto při vybíjení se zobrazením času 28
Obr. 29: Foto při vybíjení se zobrazením napětí na prvním článku 28
Obr. 30: Foto při vybíjení se zobrazením napětí na druhém článku 28
Obr. 31: Vývojový diagram programu při Discharge 29
Obr. 32: Foto výpisu na display při manual loading 30
Obr. 33: Vývojový diagram programu při Manual loading 30
Obr. 34: Zobrazení při zátěžové charakteristice 31
Obr. 35: Zobrazení při ukončení zátěžové charakteristiky 31
Obr. 36: Vývojový diagram při Measure Load Characteristic 32
Obr. 37: Ukázka přijímaných dat vypsaných do počítače 33
Obr. 38: Nastavení a vyčítání hodnot v Excelu 34
Obr. 39: Podstava krabičky 35
Obr. 40: Přední strana krabičky 35
Obr. 41: Zadní strana krabičky 36

Obr. 42: Vrchní strana krabičky 36
Obr. 43: PCB verze 1 37
Obr. 44: Upravené Arduino zespodu pro PCB verze 1 37
Obr. 45: Krabička verze 1 ze dřeva 38
Obr. 46: Krabička verze 1 ze dřeva, pohled zepředu 38
Obr. 47: Finální krabička testeru 39
Obr. 48: Pohled přímo na tester zepředu (konektor 3 cells není používán) 39
Obr. 49: Pohled dovnitř testeru 40
Obr. 50: PCB verze 2 40
Obr. 51: Grafy odchylek napětí ve voltech měřený INA 219, modře pro rozsah do 16V a hnědě pro rozsah do 32V 42
Obr. 52: Grafy absolutních odchylek napětí v procentech měřený INA 219, modře pro rozsah do 16V a hnědě pro rozsah do 32V 43
Obr. 53: Graf odchylek napětí článku 1 modře a článku 2 hnědě 43
Obr. 54: Graf s odchylkou proudu změřeného testerem s DDM před dostavením v programu 45
Obr. 55: Graf s odchylkou proudu změřeného testerem s MDD 46
Obr. 56: Připojená AA baterie k testeru při měření zátěžové charakteristiky 46
Obr. 57: Zatěžovací charakteristiky 3 různých AA baterií 47
Obr. 58: Měřené AA baterie, očíslovány 1 až 3 zleva doprava 47
Obr. 59: Zatěžovací charakteristika LiIon baterie 48
Obr. 60: Zatěžovací charakteristiky napájecích zdrojů pro mobily 48
Obr. 61: Změřené zdroje od mobilů 49
Obr. 62: Graf závislosti napětí na čase LiIon baterie 49
Obr. 63: Měřená LiIon baterie pro RC modely složená ze čtyř článků 50
Obr. 64: Graf závislosti napětí na čase NiMh baterie 50
Obr. 65: NiMh baterie pro RC modely 51

7 SEZNAM TABULEK

Tab. 1: Přednastavené napětí pro 1 článek v programu 15
Tab. 2: Tabulka připojení přepínače 4051N 21
Tab. 3: Kontrola přesnosti měření INA 219 v rozsahu 16 V pomocí multimetru (DMM) Fluke 289, v. č. 44870089 41
Tab. 4: Kontrola přesnosti měření INA 219 v rozsahu 32 V pomocí multimetru (DMM) Fluke 289, v. č. 44870089 42
Tab. 5: Tabulka napětí na prvním článku porovnaná s multimetrem 44
Tab. 6: Tabulka napětí na druhém článku porovnaná s multimetrem 44
Tab. 7: Tabulka specifikací zdrojů porovnaná s výsledky 49

8 PŘÍLOHY

8.1 Seznam součástek

Součástka	Hodnota	Pouzdro
ARDUINO_NANO2		ARDUINO_NANO
C1	100n	C0805
C2	100n	C0805
C3	100n	C0805
C4	100n	C0805
C5	10u	C0805
C6	10u	C0805
C7	100n	C0805
C8	10u	C0805
C9	not assembled	C0805
C10	10u	C0805
C11	10u	C0805
C12	100n	C0805
C13	100n	C0805
C14	100n	C0805
C15	100n	C0805
C16	100n	C0805
C17	100n	C0805
C18	100n	C0805
C19	100n	C0805
C20	100n	C0805
C21	100n	C0805
C22	10u	C0805
C23	10u	C0805
D1	CGRM4001-G	SOD-123_MINI-SMA
D2	CGRM4001-G	SOD-123_MINI-SMA
D3	CGRM4001-G	SOD-123_MINI-SMA
D4	5V1	ZDIO-7.5
D5	CGRM4001-G	SOD-123_MINI-SMA
D6		ZDIO-7.5
D7		ZDIO-7.5
D8	CGRM4001-G	SOD-123_MINI-SMA
D9	CGRM4001-G	SOD-123_MINI-SMA
F1		GSH15
IC1	LM358N	DIL08
IC2	REG1117	SOT223
IC3	LM358N	DIL08
IC4	LM358N	DIL08
IC5	4051N	DIL16

J1		SCREWTERMINAL-5MM-3
J2		SCREWTERMINAL-5MM-3
JP1		JP2Q
JP2		JP2
JP3		1X06
JP4		1X01
JP5		1X01
JP7		1X06
l.04	3mm	LED3MM
Q1	STP60NF06L	TO220BV
Q3	STP60NF06L	TO220BV
R1	10mR/1W	R2512
R2	1M	0207/10
R3	2M	0207/10
R4	2k2	R0603
R5	10mR/1W	R2512
R6	4k7	R0603
R7	330	R0603
R8	1M	0207/10
R9	10mR/1W	R2512
R10	680k	0207/10
R11	220k	0207/10
R12	100k	R0603
R13	not assembled	R0603
R14	100	R0603
R15	1k5	R0603
R16	10R	R0603
R17	10R	R0603
R18	10k	R0603
R19	100k	R0603
R20	100k	0207/10
R21	100k	R0603
R22	680k	0207/10
R23	47k	0207/10
R24	100k	0207/10
R25	100k	0207/10
R26	100k	0207/10
R27	47k	0207/10
R28	47k	0207/10
R29	10k	R0603
R30	0R	R0603
R31	10k	R0603
R32	10k	R0603
R33	100k	0207/10
R34	100k	0207/10

R35	47k	0207/10
R36	47k	0207/10
R37	10k	R0603
R38	100k	0207/10
R39	100k	0207/10
R40	47k	0207/10
R41	47k	0207/10
R42	10k	R0603
R43	10k	B25P
R44	680k	0207/10
R45	47k	0207/10
R46	1k	R0603
R47	640	R0603
R48	10M	R0603
R49	1k3	R0603
R50	1k	0207/10
R51	1k3	R0603
R52	10k	0207/10
R53	10M	R0603
SG1	F/TMB	F/TMB
T1	J3Y	SOT23-BEC
T2	J3Y	SOT23-BEC
TP1	TPPAD1-17	P1-17
U$1	WIRELESS-BLUETOOTH-HC-05	WIRELESS-BLUETOOTH-HC-05
U$3	LEVEL-SHIFTER-4CH	LEVEL-SHIFTER-4CH
U2	DS18B20	TO-92-TO92127P495H2044-3
U3	INA219AIDR	SOIC127P600X175-8N
X_BAL_IN	22-23-2041	22-23-2041
X_BUT1	22-23-2021	22-23-2021
X_BUT2	22-23-2021	22-23-2021
X_FAN	22-23-2031	22-23-2031
X_I2C_LCD	22-23-2041	22-23-2041
X_ROT	22-23-2051	22-23-2051
X_TEMP	22-23-2031	22-23-2031

8.2 Program v testeru

```
#include <LiquidCrystal_I2C.h>              // inicializace LCD
LiquidCrystal_I2C lcd(0x3F, 16, 2);

#include <OneWire.h>
#include <DallasTemperature.h>          // inicializace teploměru
#include <Wire.h>
#define ONE_WIRE_BUS A2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
```

```
#include <Adafruit_INA219.h>            //inicializace INA
#define ADDR_INA 0x40
// inicializace senzoru s nastavenou adresou z knihovny
Adafruit_INA219 ina219(ADDR_INA);


//Rotary encoder
#define encoderPinA 3
#define encoderPinB 2
#define encoderButton A1

// definitions for rotary encoder
const uint8_t LATCHSTATE = 3;
int8_t oldState = 3;
const int8_t KNOBDIR[] = {
  0, -1,  1,  0,
  1,  0,  0, -1,
  -1,  0,  0,  1,
  0,  1, -1,  0
};

// globalni promenne
int counter = 0;
int BatType;
float x = 0;
int cells; //pocet je + 1
int Idis = 500;
int disch = 0;
//int Idism = 1000;
int set = 0;
int mtime = 300;
float Vstop;
int Mcap = 2000;
int positionInternal = 0;
int positionExternal = 0;
int oldRotPos = 0;
int RotPos = 0;
int oldPositionExternal;
int zpet = 7;
int ok = A1;
int tlacitko, tlacitkoencoder, stisk0 = 0, stisk1 = 0, enter = 0;
byte pinfan = 12;
int fan;
float temp;
float VBatC, ILoad;
int measure;
float k;//koeficient
double starttime;
int capacity = 0;
float AnaRef = 4.6;
double t = 0;
int current = 0;
double dischargedCapacity = 0;
long int lasttime;
int ct = 0;
int z = 0;
float Cell1;
float Cell2;
float Voltage;
float U1 = 0, U2 = 0, I1 = 0, I2 = 0, U3 = 0, I3 = 0, U4 = 0, I4 = 0, Ustart = 0;
#define DebugMd  false   //degug mode active =1  Display messages on LCD

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
void setup() {

  lcd.init();             //inicializace LCD
```

```
  lcd.backlight();
  lcd.setCursor(0, 0);

  pinMode(zpet, INPUT_PULLUP);      //Inicializace pinů
  pinMode(4, OUTPUT);//levelshifter
  pinMode(7, OUTPUT);
  pinMode(A3, OUTPUT);
  pinMode(13, OUTPUT);//led
  pinMode(8, OUTPUT);
  digitalWrite(8, LOW);
  analogWrite(9, 0);
  analogWrite(6, 0);

  Serial.begin(9600);             // Inicializace serioveho portu
  sensors.begin();

  // initialize rotational encoder
  pinMode (encoderPinA, INPUT_PULLUP);
  pinMode (encoderPinB, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(encoderPinA), tick, CHANGE);
  attachInterrupt(digitalPinToInterrupt(encoderPinB), tick, CHANGE);
  RotPos = positionExternal;

  lcd.clear();                        //vypsání zapínací obrazovky
  lcd.setCursor(0, 0);
  lcd.print(F("Discharger and "));
  cells = counter;
  lcd.setCursor(0, 1);
  lcd.print(F("battery tester"));
  delay(1500);

  ina219.begin();
  // nastavení kalibrace, k dispozici jsou 3 režimy
  // režim 32V a 2A má největší rozsahy, ale nejmenší přesnost
  // ina219.setCalibration_32V_2A();
  // režim 32V a 1A má lepší rozlišení průchozího proudu
  // ina219.setCalibration_32V_1A();
  // režim 16V a 400mA má nejlepší rozlišení proudu i napětí
  ina219.setCalibration_16V_400mA();
  // Serial.println (F("INA init finished"));
}

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
void loop() {

  SelectBattery();
  back();
  encoder();

  do {
    encoder();
  } while (enter == 0 && Serial.available() == 0 && positionExternal == 0);

  if (Serial.available() != 0) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Remoted from PC");
    do {

      String SerialData = ReadSerialPort();

      if (SerialData == "REMOTE") {
        //    lcd.setCursor(0, 1);
        //    lcd.print("R");
        //    delay (2000);
      }
```

```
      if (SerialData.substring(0, 13) == "Battery type:") {
        //    lcd.setCursor(0, 1);
        //    lcd.print("BT ");
        //    lcd.print(SerialData.substring(14));
        //    delay(2000);
        x = x + 3;
      }
      if (SerialData.substring(0, 16) == "Number of cells:") {
        //    lcd.setCursor(0, 1);
        //    lcd.print("Cells ");
        cells = SerialData.substring(17).toInt() - 1;
        //    lcd.print(cells);
        //    delay (2000);
        x = x + 3;
      }
      if (SerialData.substring(0, 12) == "Min Voltage:") {
        //    lcd.setCursor(0, 1);
        //    lcd.print("V ");
        Vstop = SerialData.substring(13).toFloat();
        //    lcd.print(Vstop);
        //    delay (2000);
        x = x + 3;
      }
      if (SerialData.substring(0, 12) == "Max Current:") {
        //    lcd.setCursor(0, 1);
        //    lcd.print("mA ");
        Idis = SerialData.substring(13).toFloat();
        //    lcd.print(Idis);
        //    delay (2000);
        x = x + 3;
      }
      if (SerialData.substring(0, 22) == "Type of discharging: 0") {
        disch = 0;
        //    lcd.setCursor(0, 1);
        //    lcd.print(disch);
        //    delay (2000);
        x = x + 3;
      }

      if (SerialData.substring(0, 22) == "Type of discharging: 2") {
        disch = 2;
        //    lcd.setCursor(0, 1);
        //    lcd.print(disch);
        //    delay (2000);
        x = x + 3;
      }

      delay(1000);
    } while (x < 15);
  }

  do {                          //nastavení v menu, které probíhá, dokud není vše nastaveno pro
vybíjení
    if (x == 1) {
      SelectBatteryType();      //výběr mezi typy baterií
    }
    if (x == 2) {
      if (BatType < 3) {         //v případě Litiové baterie, vypsání počtu článků
        x++; lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print(F("Number of cells"));
        cells = counter;
        lcd.setCursor(14, 1);
        lcd.print(F("1s"));
      }
      else x = x + 2;
```

```
    }
    if (x == 3) {         //v případě Litiové baterie, nastavení počtu článků
      NumberOfCells();
    }
    if (x == 3.5) {       //nastavení minimálního napětí, do kterého se může při vybíjení dostat
      MinimalVoltage();
    }
    if (x == 4) {
      BaseDischargeCurrent();   //Vypsání vybíjecího proudu
    }
    if (x == 5) {
      DischargeCurrent();   //nastavení maximálního vybíjecího proudu
    }
    if (x == 6) {
      x++;                                //vypsání pro vybrání mezi typy vybíjení a dodatečné
nastavení
      Serial.print(F("SelectTypeOfDischarging: "));
    }
    if (x == 7) {
      SelectTypeOfDischarging();        //vybírání mezi typy vybíjení a dodatečné nastavení
    }
    if (x == 8) {                    //v případě dodatečného nastavení se dá nastavit maximální
vybíjecí čas
      lcd.clear();
      if (disch == 3) {
        lcd.setCursor(1, 0);
        lcd.print(F("Max disch time"));
        lcd.setCursor(1, 1);
        lcd.print(F("              "));
        x++;
      }
      else x = 15;
    }
    if (x == 9) {
      if (disch == 3) {
        Setup();
      }
      else x = 15;
    }
    if (x == 10) {
      if (set == 0) {             //v případě dodatečného nastavení se dá nastavit maximální
vybíjecí čas výpis
        lcd.setCursor(0, 0);
        lcd.print(F("Max disch time  "));
        lcd.setCursor(0, 1);
        lcd.print(F("                "));
        lcd.setCursor(14, 1);
        lcd.print(F("5h"));
        x++;
        enter = 0;
        delay(100);
      }
      if (set == 1) {            //nebo maximální vybitá kapacita výpis
        lcd.setCursor(0, 0);
        lcd.print(F("Max disch capac."));
        lcd.setCursor(0, 1);
        lcd.print(F("                "));
        lcd.setCursor(9, 1);
        lcd.print(F("2000mAh"));
        x++;
        enter = 0;
        delay(100);
      }
    }
    if (x == 11) {
      if (set == 0) {
```

```
        MaxDischTime();     //maximální vybíjecí čas nastavení
      }
      if (set == 1) {
        MaxDischCapac();    //maximální vybitá kapacita nastavení
      }
    }
  } while (x < 15);
  x = 0;
  lcd.clear();              //příprava pro změření polarity
  digitalWrite(8, LOW);
  analogWrite(9, 0);
  analogWrite(6, 0);

  float VbatNdir ;
  float Voltage;

  Serial.println("U+; U-");   //vypsání napětí nepřipojené baterie
  Voltage = VBatP();
  Serial.print(Voltage);
  Serial.print("; ");

  VbatNdir = MeasVbatNdir ();
  Serial.println(VbatNdir);

  if ((Voltage == VbatNdir) || (Voltage < VbatNdir) || (VbatNdir != 0)) {   //v případě špatné
polarity vypsání změřených hodnot a zapískání
    Serial.println(F("POLARITY NOT OK"));

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(F("POLARITY NOT OK"));
    lcd.setCursor(0, 1);
    lcd.print(F("U+"));
    lcd.setCursor(3, 1);
    lcd.print(Voltage);
    lcd.setCursor(8, 1);
    lcd.print(F("U-"));
    lcd.setCursor(11, 1);
    lcd.print(VbatNdir);

    for (int i = 0; i <= 8; i++) {
      analogWrite(5, 10);
      delay(50);
      analogWrite(5, 0);
      delay(50);
    }

    enter = 0;
    do {            //vyčkání na potvrzení uživatelem
      encoder();
      delay(10);
    } while (enter == 0);   //vrácení se pro výběr typu vybíjení
    x = 6;
    z = 1;
    enter = 0;
    lcd.clear();
  }
  float Voltage5 = 0;
  VBatC = ina219.getBusVoltage_V();
  Voltage = VBatP();

  AnaRef = AnaRef * VBatC / Voltage;    //změření nepřesnosti měření napětí pro celly a následná
kalibrace
  if (AnaRef > 5.5 || AnaRef < 4.0) AnaRef = 4.6;

  Serial.println ("Calibration of Arduino ADC supply");   //vypsání základních změřených hodnot
do počítače
```

```
  Serial.print (F("INA: "));
  Serial.print (VBatC);
  Serial.print (F(", ADC: "));
  Serial.print (Voltage);
  Serial.print (F(", AnaRef: "));
  Serial.println (AnaRef);

  if (z == 0) {
    analogWrite(5, 10);         //vypsání základních věcí do počítače i na display
    delay(100);
    analogWrite(5, 0);
    if (disch == 0) {
      lcd.print(F("DIS      A     V"));
    }
    if (disch == 1) {
      lcd.print(F("MLD      A     V"));
    }
    if (disch == 2) {
      lcd.print(F("LOAD     A     V"));
    }
    if (cells == 0) {
      Serial.println(F("U; I; C; time; temp"));
    }
    else {
      Serial.println(F("U; I; C; time; temp; U; Uc1; Uc2"));
    }
    starttime = millis();             //začátek měření času
    x = 0; stisk0 = 0; enter = 0;
    digitalWrite(8, HIGH);           //podle nastaveného maximálního proudu se nastaví rychlost
kroku
    if (Idis < 950) {
      digitalWrite(9, HIGH);
    }
  }
  while (z == 0) {    //pokud vše předchozí proběhlo bez problémů, tak z == 0 a může se vybíjet
    back();
    encoder();
    if     (disch     ==     0)     {                                    //vybíjení     (Discharge)
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000
      Ina219();             //změření a vypsání napětí a proudu
      if (VBatC > 15 or ILoad > 3500) {
        // režim 32V a 1A má lepší rozlišení průchozího proudu
        ina219.setCalibration_32V_1A();
      }
      else {
        // režim 16V a 400mA má nejlepší rozlišení proudu i napětí
        ina219.setCalibration_16V_400mA();
      }
      if (ILoad > 0) {          //sepnutí kontrolní led diody, pokud prochází nějaký proud
        digitalWrite(13, HIGH);
      } else {
        digitalWrite(13, LOW);
      }
      dischCapacity();      //vypsaní a změření kapacity
      if (Idis >= 950) {
        if (ILoad < (Idis - 30)) {      //nastavení vybíjecího proudu
          current++;
        }
        else {
          if (ILoad > (Idis + 15)) {
            current--;
          }
        }
      } else {
        if (ILoad < (Idis - 2)) {      //nastavení vybíjecího proudu
          current++;
```

```
        }
        else {
          if (ILoad > (Idis + 2)) {
            current--;
          }
        }
      }
      analogWrite(6, current);
      Serial.print(dischargedCapacity / 3600000);
      Serial.print("; ");
      dischtime();
      Serial.print(t / 1000);
      Serial.print("; ");

      vetrak();             //sepnutí/vypnutí větráku, v závislosti na teplotě
      if (cells == 1) {    //pokud baterie je typu Li a má více cellů, vypíší se i jednotlivé
celly
        Voltage = VBatP();
        Serial.print(Voltage);
        Serial.print(F("; "));
        Cell1 = MeasCell1 ();
        Serial.print(Cell1);
        Serial.print(F("; "));
        Cell2 = MeasCell2();
        Serial.println(Cell2);
      } else {
        Serial.println(F(""));
      }
      if (cells == 0) {      //postupné vypisování času a napětí na cellech na jednom místě
        dischtime();
        writetime();
      }
      napeticlanku();
      if (stisk0 == 1) {      //při zmáčknutí tlačítka back
        x++;
      }
      if (cells == 1) {
        if (x != 0 || temp > 100 || VBatC < Vstop || Cell1 < (Vstop / 2) || Cell2 < (Vstop / 2)
|| (t / 60000) >= mtime) {     //ukončení programu, pokud nějaká podmínka neplatí
          endDischarge();
        }
      }
      else {
        if (x != 0 || temp > 100 || VBatC < Vstop || (t / 60000) >= mtime) {
          endDischarge();
        }
      }
    }
    if    (disch    ==    1)    {                          //ruční    zátěž    (Manual    loading)
1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
1111111111111111111111111111111111111111
      Ina219();
      if (VBatC > 15 or ILoad > 3500) {
        // režim 32V a 1A má lepší rozlišení průchozího proudu
        ina219.setCalibration_32V_1A();
      }
      else {
        // režim 16V a 400mA má nejlepší rozlišení proudu i napětí
        ina219.setCalibration_16V_400mA();
      }
      if (ILoad > 0) {
        digitalWrite(13, HIGH);
      } else {
        digitalWrite(13, LOW);
      }
      dischCapacity();
      if (cells == 0) {
```

```
        dischtime();
        writetime();
      } else {
        napeticlanku();
      }

      if (positionExternal != oldRotPos) {                    //nastavení proudu podle otočení
encoderem
        current = current + positionExternal - oldRotPos;
        if (current <= 0) {
          current = 0;
        }
        if (current > 255) {
          current = 255;
        }
        oldRotPos = positionExternal;
      }
      analogWrite(6, current);
      Serial.print(dischargedCapacity / 3600000);
      Serial.print(F("; "));
      dischtime();
      Serial.print(t / 1000);
      Serial.print(F("; "));

      vetrak();
      if (cells == 1) {
        Voltage = VBatP();
        Serial.print(Voltage);
        Serial.print(F("; "));
        Cell2 = MeasCell2();
        Serial.print(Cell2);
        Serial.print(F("; "));
        Cell1 = MeasCell1 ();
        Serial.println(Cell1);
      } else {
        Serial.println(F(""));
      }
      if (stisk0 == 1) {
        x++;
      }
      if (cells == 1) {
        if (x != 0 || temp > 100 || VBatC < Vstop || Cell1 < (Vstop / 2) || Cell2 < (Vstop / 2)
|| (t / 60000) >= mtime) {
          endDischarge();
        }
      }
      else {
        if (x != 0 || temp > 100 || VBatC < Vstop || (t / 60000) >= mtime) {
          endDischarge();
        }
      }
    }
    if (disch == 2) {            //zátěžová charakteristike (Measure load characteristic)
22222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
22222222222222222
      Ina219();
      if (VBatC > 15 or ILoad > 3500) {
        // režim 32V a 1A má lepší rozlišení průchozího proudu
        ina219.setCalibration_32V_1A();
      }
      else {
        // režim 16V a 400mA má nejlepší rozlišení proudu i napětí
        ina219.setCalibration_16V_400mA();
      }
      if (ILoad > 0) {
        digitalWrite(13, HIGH);
      } else {
```

```
        digitalWrite(13, LOW);
      }
      //změření dynamického odporu baterie nebo zdroje
      if (Ustart == 0) {
        Ustart = VBatC;
      }
      U4 = U3;
      I4 = I3;
      U3 = U2;
      I3 = I2;
      U2 = U1;
      I2 = I1;
      U1 = VBatC;
      I1 = ILoad;

      if (-100 <= (((U4 + U3) - (U2 + U1)) / (((I4 + I3) - (I2 + I1)) / 1000)) && (((U4 + U3) -
(U2 + U1)) / (((I4 + I3) - (I2 + I1)) / 1000)) <= 100) {
        lcd.setCursor(8, 1);
        lcd.print(F("Rd      "));
        lcd.setCursor(11, 1);
        lcd.print(((U4 + U3) - (U2 + U1)) / (((I4 + I3) - (I2 + I1)) / -1000));
      }
      else {
        lcd.setCursor(8, 1);
        lcd.print(F("Rd      "));
        lcd.setCursor(12, 1);
        lcd.print(F("nan"));
      }
      dischargedCapacity = dischargedCapacity + ILoad * (millis() - lasttime);
      lasttime = millis();

      if (ILoad < Idis) {
        current++;
      }
      analogWrite(6, current);
      Serial.print(dischargedCapacity / 3600000);
      Serial.print(F("; "));
      dischtime();
      Serial.print(t / 1000);
      Serial.print(F("; "));

      vetrak();
      if (cells == 1) {
        Voltage = VBatP();
        Serial.print(Voltage);
        Serial.print("; ");
        Cell2 = MeasCell2();
        Serial.print(Cell2);
        Serial.print("; ");
        Cell1 = MeasCell1 ();
        Serial.println(Cell1);
      } else {
        Serial.println(F(""));
      }
      if (cells == 0) {
        dischtime();
        writetime();
      } else {
        napeticlanku();
      }
      if (stisk0 == 1) {
        x++;
      }
      if (cells == 1) {
        if (x != 0 || temp > 100 || VBatC < Vstop || Cell1 < (Vstop / 2) || Cell2 < (Vstop / 2)
|| (t / 60000) >= mtime || ILoad > Idis) {
          endLOA();
```

```
        }
      }
      else {
        if (x != 0 || temp > 100 || VBatC < Vstop || (t / 60000) >= mtime || ILoad > Idis) {
          endLOA();
        }
      }
    }
  }
  current = 0;
  dischargedCapacity = 0;
  enter = 0;
  z = 0;
}

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

void SelectBattery()            //vypsání pro výběr typu baterie
{
  lcd.clear();                  //vyčištění a vypsání na display
  lcd.setCursor(0, 0);
  lcd.print("Select battery");
  lcd.setCursor(0, 1);
  lcd.print("type");

  batterytype();
  x++;
  Serial.print("SelectBatteryType: ");      //výpis do počítače
}

void SelectBatteryType() {      //výběr pomocí rotačního encodéru typ baterie
  //načtení tlačítek
  encoder();

  BatType = BatType + positionExternal - oldRotPos;
  if (BatType > 6) {
    BatType = 0;
  }
  if (BatType < 0) {
    BatType = 6;
  }
  oldRotPos = positionExternal;

  batterytype();
  if (enter == 1) {
    if (BatType == 0) {
      Serial.println("LiIon");
    }
    if (BatType == 1) {
      Serial.println("LiPo");
    }
    if (BatType == 2) {
      Serial.println("LiFe");
    }
    if (BatType == 3) {
      Serial.println("NiMh");
    }
    if (BatType == 4) {
      Serial.println("NiCd");
    }
    if (BatType == 5) {
      Serial.println("Pb");
    }
    if (BatType == 6) {
      Serial.println("Source");
    }
```

```
    if (BatType < 3) {
      x++; enter = 0; counter = 0;
      Serial.print("NumberOfCells: ");
    }
    else {
      x = 3.5; enter = 0; counter = 0;
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Minimal voltage");
      if (BatType == 6) {
        Vstop = 2;
        lcd.setCursor(12, 1 );
        lcd.print("2.0");
        lcd.setCursor(15, 1 );
        lcd.print("V");
      }
      else {
        if (BatType == 5) {
          Vstop = 10.8;
          lcd.setCursor(11, 1 );
          lcd.print("10.8V");
        }
        else {
          if (BatType == 4) {
            Vstop = 0.5;
            lcd.setCursor(12, 1 );
            lcd.print("0.5V");
          }
          else {
            if (BatType == 3) {
              Vstop = 5.4;
              lcd.setCursor(12, 1 );
              lcd.print("5.4V");
            }
          }
        }
      }
      Serial.print("NumberofCells: ");
    }
  }
}

void batterytype() {
  if (BatType == 0) {         //vypsání typu baterie, pokd je to při prvním vypsání po startu,
tak nemá smyl, ale při vrácení zpět se vypíše již minule zvolený typ
    lcd.setCursor(10, 1);
    lcd.print(" LiIon");
  }
  if (BatType == 1) {
    lcd.setCursor(10, 1);
    lcd.print("  LiPo");
  }
  if (BatType == 2) {
    lcd.setCursor(10, 1);
    lcd.print("  LiFe");
  }
  if (BatType == 3) {
    lcd.setCursor(10, 1);
    lcd.print("  NiMh");
  }
  if (BatType == 4) {
    lcd.setCursor(10, 1);
    lcd.print("  NiCd");
  }
  if (BatType == 5) {
    lcd.setCursor(10, 1);
    lcd.print("    Pb");
```

```
  }
  if (BatType == 6) {
    lcd.setCursor(10, 1);
    lcd.print("Source");
  }
}

void NumberOfCells() {
  encoder();
  back();

  if (positionExternal != oldRotPos) {
    cells = cells + positionExternal - oldRotPos;
    if (cells > 2) {
      cells = 0;
    }
    if (cells < 0) {
      cells = 2;
    }
    oldRotPos = positionExternal;
    if (cells == 0) {
      lcd.setCursor(14, 1);
      lcd.print("1s");
    }
    if (cells == 1) {
      lcd.setCursor(14, 1);
      lcd.print("2s");
    }
    if (cells == 2) {
      lcd.setCursor(14, 1);
      lcd.print("3s");
    }
  }
  if (enter == 1) {
    x = x + 0.5; enter = 0; counter = 0; Serial.println(cells + 1);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Minimal voltage");
    Serial.print("MinimalVoltage: ");

    Vstop = 3 * (cells + 1);
    lcd.setCursor(14, 1 );
    lcd.print(Vstop);
    lcd.setCursor(15, 1 );
    lcd.print("V");
  }
  if (stisk0 == 1) {
    x = x - 2; enter = 0; counter = 0; stisk0 = 0;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Select battery");
    lcd.setCursor(0, 1);
    lcd.print("type");
    Serial.println("X");
    Serial.print("SelectBatteryType: ");
  }
}

void MinimalVoltage()
{
  back();
  encoder();

  if (positionExternal != oldRotPos) {
    counter = counter + positionExternal - oldRotPos;
    if (counter > 0) {
      if (Vstop < 50.1) Vstop = Vstop + 0.1;
```

```
    }
    if (counter < 0) {
      if (Vstop > 0.1)  Vstop = Vstop - 0.1;
    }
    oldRotPos = positionExternal;
    counter = 0;

    if (Vstop == 0) {
      lcd.setCursor(11, 1);
      lcd.print("    0");
      lcd.setCursor(15, 1 );
      lcd.print("V");
    }
    else {
      if (Vstop < 10) {
        lcd.setCursor(9, 1);
        lcd.print("     ");
        lcd.setCursor(12, 1);
        lcd.print(Vstop);
        lcd.setCursor(15, 1 );
        lcd.print("V");
      }
      else {
        if (Vstop < 100) {
          lcd.setCursor(9, 1);
          lcd.print("     ");
          lcd.setCursor(11, 1);
          lcd.print(Vstop);
          lcd.setCursor(15, 1 );
          lcd.print("V");
        }
      }
    }
  }
  delay(10);
  if (enter == 1) {
    x = 4; enter = 0; counter = 0;
    Serial.println(Vstop);
  }
  if (stisk0 == 1) {
    if (BatType < 3) {
      x = 2;
      stisk0 = 0;
      Serial.println("X");
      Serial.print("NumberofCells: ");
    }
    else {
      x = 1; enter = 0; counter = 0; stisk0 = 0;
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Select battery");
      lcd.setCursor(0, 1);
      lcd.print("type");
      Serial.println("X");
      Serial.print("SelectBatteryType: ");
    }
  }
}

void BaseDischargeCurrent() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Max. discharge");
  lcd.setCursor(0, 1);
  lcd.print("current");
  lcd.setCursor(11, 1 );
  lcd.print("500mA");
```

```
  x++;
  Serial.print("DischargeCurrent: ");
  delay(200);
}

void DischargeCurrent() {
  back();
  encoder();
  counter = counter + positionExternal - oldRotPos;
  if (counter > 0) {
    if (Idis < 10000)if (Idis >= 1000) {
        if (Idis >= 5000) {
          Idis = Idis + 500;
        }
        else Idis = Idis + 100;
      }
      else {
        Idis = Idis + 50;
      }
  }
  if (counter < 0) {
    if (Idis > 1000) {
      if (Idis > 5000) {
        Idis = Idis - 500;
      }
      else Idis = Idis - 100;
    }
    else {
      if (Idis >= 50)Idis = Idis - 50;
    }
  }
  oldRotPos = positionExternal;
  counter = 0;
  if (Idis == 0) {
    lcd.setCursor(9, 1);
    lcd.print("    0");
  }
  else {
    if (Idis < 100) {
      lcd.setCursor(9, 1);
      lcd.print("     ");
      lcd.setCursor(12, 1);
      lcd.print(Idis);
    }
    else {
      if (Idis < 1000) {
        lcd.setCursor(9, 1);
        lcd.print("     ");
        lcd.setCursor(11, 1);
        lcd.print(Idis);
      }
      else {
        if (Idis < 10000) {
          lcd.setCursor(9, 1);
          lcd.print("     ");
          lcd.setCursor(10, 1);
          lcd.print(Idis);
        }
        else {
          if (Idis < 100000) {
            lcd.setCursor(9, 1);
            lcd.print("     ");
            lcd.setCursor(9, 1);
            lcd.print(Idis);
          }
        }
      }
```

```
    }
  }
  delay(50);

  if (enter == 1) {
    x++; enter = 0; counter = 0; Serial.println(Idis);
  }
  if (stisk0 == 1) {
    if (BatType != 4 && BatType != 6) {
      x = 2;
      stisk0 = 0;
      Serial.println("X");
      Serial.print("NumberofCells: ");
    }
    else {
      x = 3.5; enter = 0; counter = 0; stisk0 = 0;
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Minimal voltage");
      if (BatType == 6) {
        Vstop = 2;
        lcd.setCursor(12, 1 );
        lcd.print("2.0");
        lcd.setCursor(15, 1 );
        lcd.print("V");
      }
      else {
        if (BatType == 5) {
          Vstop = 10.8;
          lcd.setCursor(11, 1 );
          lcd.print("10.8V");
        }
        else {
          if (BatType == 4) {
            Vstop = 0.5;
            lcd.setCursor(12, 1 );
            lcd.print("0.5V");
          }
          else {
            if (BatType == 3) {
              Vstop = 5.4;
              lcd.setCursor(12, 1 );
              lcd.print("5.4V");
            }
          }
        }
      }
      Serial.println("X");
      Serial.print("MinimalVoltage: ");
    }
  }
}

void back() {
  tlacitko = analogRead(zpet);

  if (tlacitko == 0) {
    stisk0++;
    do {
      tlacitko = analogRead(zpet);
    } while (tlacitko == 0);
  }
}

void encoder() {
  tlacitkoencoder = analogRead(ok);
```

```
  if (tlacitkoencoder == 0) {
    enter++;
    do {
      tlacitkoencoder = analogRead(ok);
    } while (tlacitkoencoder == 0);
  }
}

void SelectTypeOfDischarging() {
  lcd.setCursor(0, 0);
  lcd.print((char)127);
  lcd.setCursor(15, 0);
  lcd.print((char)126);
  lcd.setCursor(0, 1);
  lcd.print((char)127);
  lcd.setCursor(15, 1);
  lcd.print((char)126);
  back();
  encoder();
  disch = disch + positionExternal - oldRotPos;
  if (disch < 0) {
    disch = 3;
  }
  if (disch > 3) {
    disch = 0;
  }
  oldRotPos = positionExternal;
  if (disch == 0) {
    lcd.setCursor(1, 0);
    lcd.print("  Discharge   ");
    lcd.setCursor(1, 1);
    lcd.print("              ");
  }
  if (disch == 1) {
    lcd.setCursor(1, 0);
    lcd.print("Manual loading");
    lcd.setCursor(1, 1);
    lcd.print("              ");
  }
  if (disch == 2) {
    lcd.setCursor(1, 0);
    lcd.print(" Measure load ");
    lcd.setCursor(1, 1);
    lcd.print("characteristic");
  }
  if (disch == 3) {
    lcd.setCursor(1, 0);
    lcd.print("    Setup     ");
    lcd.setCursor(1, 1);
    lcd.print("              ");
  }
  if (enter == 1) {
    x++; enter = 0; counter = 0;
    if (disch == 0) {
      Serial.println("Discharge");
    }
    if (disch == 1) {
      Serial.println("Manual loading");
    }
    if (disch == 2) {
      Serial.println("Measure load characteristic");
    }
    delay(100);
  }
  if (stisk0 == 1) {
    x = x - 3; enter = 0; counter = 0; stisk0 = 0;
    Serial.println("X");
```

```
  }
}

void Setup() {
  back();
  encoder();

  if (positionExternal != oldRotPos) {
    set = set + positionExternal - oldRotPos;
    if (set < 0) {
      set = 3;
    }
    if (set > 3) {
      set = 0;
    }
    oldRotPos = positionExternal;
    if (set == 0) {
      lcd.setCursor(1, 0);
      lcd.print("Max disch time");
      lcd.setCursor(1, 1);
      lcd.print("              ");
    }
    if (set == 1) {
      lcd.setCursor(1, 0);
      lcd.print("Max discharge ");
      lcd.setCursor(0, 1);
      lcd.print("    capacity   ");
    }
    if (set == 2) {
      lcd.setCursor(1, 0);
      lcd.print("  Max load I  ");
      lcd.setCursor(1, 1);
      lcd.print("              ");
    }
    if (set == 3) {
      lcd.setCursor(1, 0);
      lcd.print("  Fan Power   ");
      lcd.setCursor(1, 1);
      lcd.print("              ");
    }
  }
  if (enter == 1) {
    x++; enter = 0; counter = 0;
    Serial.println("setup selected");
    Serial.println(x);
    Serial.println(enter);
  }
  if (stisk0 == 1) {
    x = x - 3; enter = 0; counter = 0; stisk0 = 0;
  }
}

void MaxDischTime() {
  back();
  encoder();

  if (positionExternal != oldRotPos) {
    counter = counter + positionExternal - oldRotPos;
    if (counter > 0) {
      if (mtime < 1381) {
        if (mtime >= 60) {
          if (mtime >= 120) {
            mtime = mtime + 60;
          }
          else mtime = mtime + 30;
        }
        else {
```

```
          mtime = mtime + 10;
        }
      }
    }
    if (counter < 0) {
      if ((mtime < 1441) and (mtime > 0)) {
        if (mtime <= 120) {
          if (mtime <= 60) {
            mtime = mtime - 10;
          }
          else mtime = mtime - 30;
        }
        else {
          mtime = mtime - 60;
        }
      }
    }
    oldRotPos = positionExternal;
    counter = 0;

    if (mtime == 0) {
      lcd.setCursor(10, 1);
      lcd.print("  0min");
    }
    else {
      if (mtime < 120) {
        lcd.setCursor(10, 1);
        lcd.print("   min");
        lcd.setCursor(11, 1);
        lcd.print(mtime);
      }
      else {
        if (mtime < 600) {
          lcd.setCursor(10, 1);
          lcd.print("     h");
          lcd.setCursor(14, 1);
          lcd.print(mtime / 60);
        }
        else {
          if (mtime < 100000) {
            lcd.setCursor(10, 1);
            lcd.print("     h");
            lcd.setCursor(13, 1);
            lcd.print(mtime / 60);
          }
        }
      }
    }
    delay(10);
  }
  if (enter == 1) {
    x = 8; enter = 0; counter = 0;
    delay(100);
  }
}

void MaxDischCapac() {
  back();
  encoder();

  if (positionExternal != oldRotPos) {
    counter = counter + positionExternal - oldRotPos;
    if (counter > 0) {
      if (Mcap < 100000)
        if (Mcap >= 1000) {
          if (Mcap >= 5000) {
            if (Mcap >= 10000) {
```

```
          Mcap = Mcap + 1000;
        }
        else {
          Mcap = Mcap + 500;
        }
      }
      else Mcap = Mcap + 100;
    }
    else {
      Mcap = Mcap + 50;
    }
}
if (counter < 0) {
  if (Mcap > 1000) {
    if (Mcap > 5000) {
      if (Mcap > 10000) {
        Mcap = Mcap - 1000;
      }
      else {
        Mcap = Mcap - 500;
      }
    }
    else Mcap = Mcap - 100;
  }
  else {
    if (Mcap >= 50)Mcap = Mcap - 50;
  }
}
oldRotPos = positionExternal;
counter = 0;
if (Mcap == 0) {
  lcd.setCursor(9, 1);
  lcd.print("    0");
}
else {
  if (Mcap < 100) {
    lcd.setCursor(9, 1);
    lcd.print("     ");
    lcd.setCursor(12, 1);
    lcd.print(Mcap);
  }
  else {
    if (Mcap < 1000) {
      lcd.setCursor(9, 1);
      lcd.print("     ");
      lcd.setCursor(11, 1);
      lcd.print(Mcap);
    }
    else {
      if (Mcap < 10000) {
        lcd.setCursor(9, 1);
        lcd.print("     ");
        lcd.setCursor(10, 1);
        lcd.print(Mcap);
      }
      else {
        if (Mcap < 100000) {
          lcd.setCursor(9, 1);
          lcd.print("     ");
          lcd.setCursor(9, 1);
          lcd.print(Mcap);
        }
      }
    }
  }
}
delay(10);
```

```
  }
  if (enter == 1) {
    x = 8; enter = 0; counter = 0;
    delay(100);
  }
}

void temperature()
{
  // Send the command to get temperatures
  sensors.requestTemperatures();
  //print the temperature in Celsius
  temp = sensors.getTempCByIndex(0);
  Serial.print(temp);
  Serial.print("; ");
}

void vetrak()
{
  temperature();
  if (temp >= 30)analogWrite(pinfan, 255);
  if (temp < 28)analogWrite(pinfan, 0);
}

void Ina219() {
  VBatC = ina219.getBusVoltage_V();
  ILoad = ina219.getCurrent_mA() * 10.0;
  ILoad = 0.903 * ILoad + 0.0215;

  Serial.print(VBatC, 3);
  Serial.print("; ");
  Serial.print(ILoad, 0);
  Serial.print("; ");

  lcd.setCursor(11, 0);
  lcd.print(VBatC);
  lcd.setCursor(15, 0);
  lcd.print("V");

  if (ILoad >= 0) {
    if (ILoad >= 10000) {
      lcd.setCursor(4, 0);
      lcd.print(ILoad / 1000);
      lcd.setCursor(9, 0);
      lcd.print("A ");
    }
    else {
      if (ILoad >= 1000) {
        lcd.setCursor(3, 0);
        lcd.print("    ");
        lcd.setCursor(5, 0);
        lcd.print(ILoad / 1000);
        lcd.setCursor(9, 0);
        lcd.print("A ");
      }
      else {
        if (ILoad < 1) {
          lcd.setCursor(3, 0);
          lcd.print("    0mA ");
        }
        else {
          if (ILoad < 100) {
            lcd.setCursor(3, 0);
            lcd.print("    ");
            lcd.setCursor(6, 0);
            lcd.print(ILoad);
            lcd.setCursor(8, 0);
```

```
            lcd.print("mA ");
          }
          else {
            lcd.setCursor(3, 0);
            lcd.print("    ");
            lcd.setCursor(5, 0);
            lcd.print(ILoad);
            lcd.setCursor(8, 0);
            lcd.print("mA ");
          }
        }
      }
    }
  }
}

float VBatP() {
  digitalWrite(4, LOW);
  digitalWrite(7, LOW);
  digitalWrite(A3, LOW);
  delay(500);
  int AnalogValue = analogRead(A6);
  float Voltage = AnalogValue / 1024.0 * AnaRef * 4.55;
  return Voltage;
}

void ISNS_DIS() {
  digitalWrite(4, HIGH );
  digitalWrite(7, LOW);
  digitalWrite(A3, LOW);
  delay(500);
  measure = analogRead(A6);
  Serial.print(measure);
  Serial.print("; ");
}

float MeasVbatNamp() {
  digitalWrite(4, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(A3, LOW);
  delay(500);
  int AnalogValue = analogRead(A6);
  float Voltage = AnalogValue / 1024.0 * AnaRef / -15.0;
  return Voltage;
}

void Bal1() {
  digitalWrite(4, HIGH );
  digitalWrite(7, HIGH);
  digitalWrite(A3, LOW );
  delay(500);
  measure = analogRead(A6);
  Serial.print(measure);
  Serial.print("; ");
}

float MeasCell1 () {
  digitalWrite(4, LOW);    //a LSB
  digitalWrite(7, LOW);    //b
  digitalWrite(A3, HIGH ); //c MSB
  delay(500);
  int AnalogValue = analogRead(A6);
  float Voltage = AnalogValue / 1024.0 * AnaRef * 2.39 ; //meri presne
  return Voltage;
}

float MeasCell2() {
```

```
  digitalWrite(4, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(A3, HIGH);
  delay(500);
  int AnalogValue = analogRead(A6);
  float Voltage = AnalogValue / 1024.0 * AnaRef * 2.35 ; //momentalne o 2 setiny podmeruje
  return Voltage;
}

float MeasVbatNbal() {
  digitalWrite(4, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(A3, HIGH);
  delay(500);
  int AnalogValue = analogRead(A6);
  float Voltage = AnalogValue / 1024.0 * AnaRef;
  return Voltage;
}


float MeasVbatNdir () {
  digitalWrite(4, HIGH);
  digitalWrite(7, HIGH);
  digitalWrite(A3, HIGH);
  delay(500);
  int AnalogValue = analogRead(A6);
  float Voltage = AnalogValue / 1024.0 * AnaRef;
  return Voltage;
}


void dischtime() {
  t = millis() - starttime;

}

void dischCapacity() {
  dischargedCapacity = dischargedCapacity + ILoad * (millis() - lasttime);
  lasttime = millis();
  if ((dischargedCapacity / 3600000) >= 10000) {
    lcd.setCursor(9, 1);
    lcd.print(dischargedCapacity / 3600000000);
    lcd.setCursor(14, 1);
    lcd.print("Ah");
  }
  else {
    if ((dischargedCapacity / 3600000) >= 1000) {
      lcd.setCursor(9, 1);
      lcd.print(dischargedCapacity / 3600000);
      lcd.setCursor(13, 1);
      lcd.print("mAh");
    }
    else {
      if ((dischargedCapacity / 3600000) >= 100) {
        lcd.setCursor(9, 1);
        lcd.print("       ");
        lcd.setCursor(10, 1);
        lcd.print(dischargedCapacity / 3600000);
        lcd.setCursor(13, 1);
        lcd.print("mAh");
      }
      else {
        if ((dischargedCapacity / 3600000) >= 10) {
          lcd.setCursor(9, 1);
          lcd.print("       ");
          lcd.setCursor(11, 1);
          lcd.print(dischargedCapacity / 3600000);
```

```
          lcd.setCursor(13, 1);
          lcd.print("mAh");
        }
        else {
          lcd.setCursor(9, 1);
          lcd.print("       ");
          lcd.setCursor(12, 1);
          lcd.print(dischargedCapacity / 3600000);
          lcd.setCursor(13, 1);
          lcd.print("mAh");
        }
      }
    }
  }
}

void endDischarge() {
  digitalWrite(8, LOW);
  analogWrite(9, 0);
  analogWrite(6, 0);
  digitalWrite(13, LOW);
  lcd.setCursor(0, 1);
  lcd.print(F("      "));
  dischtime();
  analogWrite(5, 10);
  delay(1000);
  analogWrite(5, 0);
  lcd.setCursor(0, 0);
  if ((t / 60000) >= mtime) {
    if (disch == 0) {
      lcd.print(F("DIS time  "));
    }
    if (disch == 1) {
      lcd.print(F("MLD time  "));
    }
    Serial.println(F("finish time"));
  } else {
    if (disch == 0) {
      lcd.print(F("DIS done  "));
    }
    if (disch == 1) {
      lcd.print(F("MLD done  "));
    }

    Serial.println(F(""));
    Serial.println(F("finish"));
  }
  do {
    encoder();
    delay(10);
  } while (enter == 0);
  z = 2;
  x = 0;
}

void endLOA() {
  if (current > 2) {
    if (current > 20) {
      current = current * 0.6;
    }
    else {
      current = current / 2;
    }
    analogWrite(6, current);
    delay(100);
    Ina219();
```

```
    lcd.setCursor(8, 1);
    lcd.print("Rd      ");
    lcd.setCursor(11, 1);
    lcd.print(((Ustart - VBatC) / ILoad) * 1000);


  } else {
    lcd.setCursor(9, 1);
    lcd.print("Rd     ");
    lcd.setCursor(12, 1);
    lcd.print("nan");
  }
  digitalWrite(8, LOW);
  analogWrite(9, 0);
  analogWrite(6, 0);
  digitalWrite(13, LOW);
  lcd.setCursor(0, 1);
  lcd.print("      ");
  dischtime();

  analogWrite(5, 10);
  delay(1000);
  analogWrite(5, 0);

  lcd.setCursor(0, 0);
  if ((t / 60000) >= mtime) {
    lcd.print("LOA time  ");
    Serial.println("finish time");
  } else {
    lcd.print("LOA done  ");
    Serial.println("");
    Serial.println("finish");
  }


  do {
    encoder();
    delay(10);
  } while (enter == 0);
  z = 2;
  x = 0;
}

void writetime() {
  lcd.setCursor(0, 1);
  lcd.print(t / 3600000);
  lcd.setCursor(4, 1);
  lcd.print("h");
}

void napeticlanku() {
  if (cells == 1) {
    if (ct == 0 or ct == 1) {
      lcd.setCursor(0, 1);
      lcd.print(F("      "));
      dischtime();
      writetime();
      ct++;
    } else {
      if (ct == 2 or ct == 3) {
        lcd.setCursor(1, 1);
        lcd.print(Cell1);
        lcd.setCursor(0, 1);
        lcd.print(F("A"));
        lcd.setCursor(5, 1);
        lcd.print(F("V"));
        ct++;
```

```
      } else {
        if (ct == 4 or ct == 5) {
          lcd.setCursor(1, 1);
          lcd.print(Cell2);
          lcd.setCursor(0, 1);
          lcd.print(F("B"));
          lcd.setCursor(5, 1);
          lcd.print(F("V"));
          if (ct == 5) {
            ct = 0;
          } else {
            ct++;
          }
        }
      }
    }
  }
}

String ReadSerialPort()
{
  uint8_t x = 0,  answer = 0, y = 0;
  char response[30];
  char resp;
  unsigned long previous;


  memset(response, '\0', 28);    // Initialize the string

  x = 0;
  y = 0;
  previous = millis();
  if (DebugMd == true) lcd.setCursor(0, 0);
  // this loop waits for the answer
  do
  {
    if (Serial.available() != 0) {
      resp = Serial.read();
      if (resp > 31) {
        response[x] = resp;
        if (DebugMd == true && x < 16) {
          lcd.print((char)resp);
        }
        x++;
      }

    }
  }
  while ((resp != 10) && ((millis() - previous) < 1000) && (x < 110));

  return response;

}

void tick() {
  int sigA = digitalRead(encoderPinA);
  int sigB = digitalRead(encoderPinB);
  int8_t thisState = sigA | (sigB << 1);
  if (oldState != thisState) {
    positionInternal += -KNOBDIR[thisState | (oldState << 2)];
    if (thisState == LATCHSTATE)
      positionExternal = positionInternal >> 2;
    oldState = thisState;
  }
}
```

8.3 Makro v Excelu

```
Private Sub CommandButton1_Click()

Dim COM_Byte As Byte
Dim Input_Buffer As String
Dim Output_Buffer As String
Dim arrSplitStrings1() As String


ComSetting = Me.Cells(3, 3) + ":9600,N,8,1"
DischargeStarted = False
'Me.Range("D10:D14") = ""
Me.Range("B19:I1000") = ""


    'Open "COM7:9600,N,8,1" For Random As #1 Len = 1
    Open ComSetting For Random As #1 Len = 1
    Input_Buffer = ""
    ''CharsRemaining = 0

    Application.Wait (Now + TimeValue("0:00:5"))

    WriteStringToFile ("REMOTE" + vbCrLf)
   Application.Wait (Now + TimeValue("0:00:1"))

   WriteStringToFile ("Battery type: " + Me.Cells(10, 4) + vbLf)
    Application.Wait (Now + TimeValue("0:00:1"))
    WriteStringToFile ("Number of cells: " + CStr(Me.Cells(11, 4)) + vbLf)
   Application.Wait (Now + TimeValue("0:00:1"))
    WriteStringToFile ("Min Voltage: " + CStr(Me.Cells(12, 4))) + vbLf
    Application.Wait (Now + TimeValue("0:00:1"))
    WriteStringToFile ("Max Current: " + CStr(Me.Cells(13, 4))) + vbLf
    Application.Wait (Now + TimeValue("0:00:1"))
    WriteStringToFile ("Type of discharging: 2") + vbLf

'    WriteStringToFile ("Command: LoadChar" + vbCrLf)



    'Get #1, , COM_Byte
LineNumber = 20
 Do
    Get #1, , COM_Byte
    'Debug.Print Input_Buffer   ' print it
    If COM_Byte Then
        'Debug.Print Input_Buffer
        If COM_Byte = 13 Then           ' look for CR line termination
            Debug.Print Input_Buffer   ' print it



              If Left(Input_Buffer, 5) = "U; I;" Then
                DischargeStarted = True

            End If

            If DischargeStarted = True Then

              If InStr(1, Input_Buffer, ";", vbTextCompare) > 0 Then
                arrSplitStrings1 = Split(Input_Buffer, ";", 8, vbTextCompare)
                For i = LBound(arrSplitStrings1, 1) To UBound(arrSplitStrings1, 1)
                  Me.Cells(LineNumber, 2 + i) = arrSplitStrings1(i)
                Next i
                Else
                  Me.Cells(LineNumber, 2) = Input_Buffer
```

```
              End If
              If Left(Input_Buffer, 6) = "finish" Then
                Close
                Exit Sub
              End If

            LineNumber = LineNumber + 1

            End If
            Input_Buffer = ""
        Else
            If COM_Byte > 31 Then Input_Buffer = Input_Buffer & Chr(COM_Byte) ' assemble output
buffer
        End If
    End If
    DoEvents
    Loop




    Application.Wait (Now + TimeValue("0:00:10"))

    Close #1



End Sub


Private Sub WriteStringToFile(Data)
Dim ByteToSend As Byte
  For i = 1 To Len(Data)
    ByteToSend = Asc(Mid(Data, i, 1))
    Put #1, , ByteToSend
  Next i
End Sub

Private Sub CommandButton2_Click()
Dim strBatchName As String
  strBatchName = "c:\temp\SetComPort.bat"
  ComSetting = "mode.com " + Me.Cells(3, 3) + ":9600,N,8,1"

  Open strBatchName For Output As #1
  Print #1, ComSetting
  Close #1

  Shell strBatchName
End Sub
```