TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Elektrochemická impedanční spektroskopie Li-Ion článků na platformě STM32

Diplomová práce

Studijní program: N2612 Elektrotechnika a informatika
Studijní obor: Informační technologie

Autor práce: **Bc. Jan Kálecký**
Vedoucí práce: Ing. Miroslav Novák, Ph.D.
Ústav mechatroniky a technické informatiky

Liberec 2022

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání diplomové práce

Elektrochemická impedanční spektroskopie Li-Ion článků na platformě STM32

Jméno a příjmení: **Bc. Jan Kálecký**
Osobní číslo: M19000153
Studijní program: N2612 Elektrotechnika a informatika
Studijní obor: Informační technologie
Zadávající katedra: Ústav mechatroniky a technické informatiky
Akademický rok: **2021/2022**

Zásady pro vypracování:

1. Seznamte se s principem měření elektrochemické impedanční spektroskopie.
2. Navrhněte firmware pro mikrokontroler řady STM32, který bude provádět měření spektroskopie po jednotlivých frekvenčních krocích. Základem bude generování sinusového průběhu proudu PWM modulací zatěžovacího rezistoru a měření napěťové odezvy článku. Součástí FW bude komunikační rozhraní pro ovládání měření a odesílání výsledků do PC.
3. Program otestujte na prototypové desce. Výsledky porovnejte s daty zístkanými na aparatuře s měřicí kartou NI.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	40–50 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	Čeština

Seznam odborné literatury:

[1] CHEN, C H, J LIU a K AMINE, 2001. Symmetric cell approach and impedance spectroscopy of high power lithium-ion batteries. *Journal of Power Sources*. 8.

[2] KOSEOGLOU, M., E. TSIOUMAS, D. PAPAGIANNIS, N. JABBOUR a C. MADEMLIS, 2021. A Novel On-Board Electrochemical Impedance Spectroscopy System for Real-Time Battery Impedance Estimation. *IEEE Transactions on Power Electronics* [online]. 1–1. ISSN 1941-0107. Dostupné z: doi:10.1109/TPEL.2021.3063506

[3] NUSEV, G., Đ JURIČIĆ, M. GABERŠČEK, J. MOŠKON a P. BOŠKOSKI, 2021. Fast Impedance Measurement of Li-ion Battery Using Discrete Random Binary Excitation and Wavelet Transform. *IEEE Access* [online]. 1–1. ISSN 2169-3536. Dostupné z: doi:10.1109/ACCESS.2021.3058368

Vedoucí práce:	Ing. Miroslav Novák, Ph.D. Ústav mechatroniky a technické informatiky
Datum zadání práce:	12. října 2021
Předpokládaný termín odevzdání:	16. května 2022

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Ing. Josef Černohorský, Ph.D.
vedoucí ústavu

V Liberci dne 12. října 2021

Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

16. května 2022 Bc. Jan Kálecký

Poděkování

Chtěl bych velice poděkovat vedoucímu mé diplomové práce, jímž byl Ing. Novák Miroslav, Ph.D., a to za odbornou pomoc při konzultacích a za svěřený čas.

Dále děkuji mé přítelkyni a rodině za trpělivost a podporu při studiu.

Elektrochemická impedanční spektroskopie Li-Ion článků na platformě STM32

Abstrakt

Diplomová práce se zabývá elektrochemickou impedanční spektroskopií Li-Ion článků, kterou uvádí v praxi na platformě STM32. Cílem bylo seznámit se s metodou měření amplitudy a fáze vzorkovaného signálu a znalosti zúročit při navržení, implementaci a ladění systému pro měření a vizualizaci impedance článku a fáze signálu. Výsledek tvoří firmware pro měření impedanční spektroskopie na mikrokontroleru STM32H723ZG a klientská aplikace pro obsluhu zvoleného hardwaru a zobrazení výsledků. Práce popisuje konfiguraci všech potřebných periferií a dosažené výsledky porovnává s referenčním měřením.

Klíčová slova

elektrochemická impedanční spektroskopie, synchronní detekce, Li-Ion baterie, STM32H7

Electrochemical impedance spectroscopy of Li-Ion cells on the STM32 platform

Abstract

This thesis deals with electrochemical impedance spectroscopy of Li-Ion batteries, which is then used for practical implementation on the STM32 platform. The goal was to get to know the measurement method of amplitude and phase sampling and then to use the knowledge for designing and developing a system for measuring and visualizing battery impedance and signal phase. I present the impedance spectroscopy firmware for the STM32H723ZG microcontroller and a client application for its administrating and interpreting its results. The work describes proper configuration of all chosen peripherals and the achieved results are being compared with reference measurements.

Key words

electrochemical impedance spectroscopy, synchronous detection, Li-Ion batteries, STM32H7

Obsah

Seznam obrázků

Seznam zkratek

EIS – elektrochemická impedanční spektroskopie (Electrochemical Impedance Spectroscopy)

SoC – stav nabití baterie (State of Charge)

SoH – zdravotní stav, stárnutí baterie (State of Health)

SoF – stav funkce baterie (State of Function)

MOSFET – polem řízený tranzistor (Metal Oxide Semiconductor Field Effect Transistor)

MCU – mikrokontroler (Microcontroller)

DMA – přímý přístup do paměti (Direct Memory Access)

DPS – deska plošných spojů (PCB – Printed Circuit Board)

USB – univerzální sériová sběrnice (Universal Seriál Bus)

CDC – Communication Device Controller

ADC – A/D převodník (Analog to Digital Converter)

SPI – Serial Peripheral Interface

SCPI – Standard Commands for Programmable Instruments

PWM – pulzně šířková modulace (Pulse Width Modulation)

TIMx – časovač x (timer x)

APB – Advanced Peripheral Bus

BMS – obvod dohledu baterie (Battery Management System)

SNR – odstup signálu od šumu (Signal-to-Noise Ratio)

GPIO – General-purpose input/output

MISO – Master in Slave out

RCC – Reset and Clock Control

PLL – fázový závěs (Phase-locked Loop)

HSI – High Speed Internal Oscillator

HSE - High Speed External Oscillator

Úvod

V dnešním světě plném moderních technologií je třeba dbát také na udržitelnost celého systému, otázka znečištění životního prostředí nebyla nikdy tak důležitou. O elektromobilitě toho slyšíme mnoho ze všech stran, ale ať už stojíme na kterékoliv straně, přináší opět něco nového a s tím i rozvoj nových technologií. Ačkoliv otázka bezztrátového uchovávání energie doposud nebyla vyřešena, poměrně malá přenosná úložiště máme v podobě baterií. Ať už to mají být články do elektromobilů, nebo jiné baterie, je nutné, abychom porozuměli jejich fyzikálním vlastnostem. Recyklace lithiových článků činí globálně stále pouhá procenta, ale kdybychom věděli jak dosáhnout nejvyšší možné kvality baterií při co nejmenší výrobní ceně a při rozumné udržitelnosti, příroda by to jistě ocenila.

Metody pro zkoumání vlastností článků již existují – elektrochemická impedanční spektroskopie hraje významnou roli při sběru informací o používaných Li-Ion článcích. Samotná spektroskopie je komplikovaná a náročná na přesnost, články na přístroje pro měření musíme nejprve připojit a následně čekat poměrně dlouhou dobu, než se o nich něco dozvíme. Cílem této práce bylo zjednodušit celý proces měření impedanční spektroskopie baterií, a to s využitím moderního procesoru. Převést ho na jednoduché, levné zařízení, které by bylo možné použít pro automatické testování článků. Stačilo by tak připojit článek a za pár minut bychom mohli sledovat jeho charakteristiky.

Z toho vychází cíle práce. Nejprve se seznámit s principy měření elektrochemické impedanční spektroskopie a s těmito poznatky navrhnout firmware pro moderní procesor. Ten by měl provádět měření spektroskopie po jednotlivých frekvenčních krocích. Základem bude generování sinusového průběhu proudu PWM modulací zatěžovacího rezistoru a měření napěťové odezvy článku. Díky tomu zjistíme impedanční spektroskopii článku. FW bude obsahovat komunikační rozhraní, a to pro ovládání měření, a pro odesílání výsledků a jejich interpretaci. Celý systém je nutné řádně otestovat na prototypové desce. Tyto výsledky můžeme následně porovnat s referenčním měřením na aparatuře s měřící kartou NI.

Diplomová práce je rozdělena do 6 hlavních kapitol. Nejprve seznamuje s metodami elektrochemické impedanční spektroskopie a způsoby měření amplitudy a fáze vzorkovaného signálu. V průběhu měření totiž dochází k excitaci systému střídavým proudem, který HW vzorkuje, stejně jako elektrické napětí. Z těchto veličin získáme impedanci, která udává informace o zkoumaném článku. Hlavním cílem bylo vyvinout firmware pro mikrokontroler řady STM32, který metodu spektroskopie bude implementovat.

Další část práce popisuje dostupný HW a požadavky na něj – udává potřebné periferie a jejich konfiguraci. Přistupuje k výběru konkrétního procesoru, na němž vývoj probíhal, a na kterém byl celý systém laděn. Dále zpravuje o vlastním návrhu firmwaru. Nejprve zmiňuje generování sinusového proudu, které slouží k buzení článku, a to za pomoci časovačů. Zde nalezneme důležitou část programu pro nastavení požadované frekvence a obsluhu přerušení časovače pro průběžné výpočty. Samotné měření napětí a proudu článku obsluhují převodníky, které spojité signály vzorkují. Zmiňuji výpočet synchronní detekce, s jejíž pomocí získáme absolutní hodnotu impedance a fázový posuv. Popis navrženého firmwaru zakončí periferie pro měření teploty článku při měření, nastavení hodinových kmitočtů celého systému a princip komunikace s nadřazeným systémem.

Pro snadnou vizualizaci a interpretaci výsledků v rámci práce vzniká i klientská aplikace. Nejprve popíši způsob práce s touto aplikací a proces vizualizace naměřených dat. Následovat bude ukládání dat pro pozdější využití. Závěrečná kapitola osvětlí některá měření, metodiku kalibrace a výpočet kalibračních konstant. Zpravuje o celkovém testování navrženého a implementovaného systému, jehož výsledky dále porovnává s referenčním měřením.

1 Impedanční spektroskopie

Elektrochemická impedanční spektroskopie (EIS) se často využívá pro popis charakteristik baterií a článků nebo jiných elektrochemických systémů. Metoda je vhodná pro monitorování stavu těchto systémů v průběhu času a tedy pro kontrolu životnosti baterií apod. Konvenční způsob počítá s postupnou excitací systému za pomoci signálů sinusového průběhu na předem definovaných frekvencích. Měření se postupně opakuje na jednotlivých frekvencích pro pokrytí celého zvoleného spektra. Pro zpřesnění naměřených hodnot lze také na každé frekvenci měření několikrát opakovat a následně je průměrovat. Rozsah frekvencí, na němž má smysl metodu provádět, je poměrně široký – od jednotek μHz až po stovky MHz. V závislosti na frekvenci můžeme určit komplexní hodnotu impedance systému. Protože měření provedeme na vícero frekvencích, získáme tak celé souhrnné impedanční spektrum. Pokud chceme zjistit aktuální životnost baterie, nejvíce nás bude zajímat pásmo nižších frekvencí, na nichž právě excitace může zabrat poměrně hodně času. Zajistit stabilní podmínky při dlouhodobém experimentu je při měření klíčové, protože např. i teplotní podmínky mohou ovlivnit hodnoty impedance. [3]

Impedanční spektrum lze nejlépe interpretovat za pomoci grafů. Často se setkáme se zobrazením závislosti absolutní hodnoty impedance a fáze na frekvenci, případně se zobrazením imaginárních a reálných komponent impedance. Právě skrze tuto interpretaci jednoduše porovnáme stav baterie opakovaným měřením, kdy se degradace baterie projeví v impedančním spektru. Při podrobnější analýze lze navíc zhodnotit, kde k elektrochemickým jevům uvnitř článku dochází, protože frekvence, níž jsou děje charakterizovány, se různě liší – může tomu tak být na katodě článku, anodě, případně ve vrstvě mezi nimi. [1][2]

Například v hybridních automobilech, nebo elektromobilech se často využívá Li-Ion článků pro ukládání a zdroj elektrické energie. Bylo zjištěno, že impedance u článků 18650 se postupně zvyšuje s časem. Články jsou vyvíjeny tak, aby se dalo rozlišit, kde k nárůstu impedance dochází – tedy zdali je tomu tak na katodě, nebo na anodě. Při experimentech s uchováním článků ve stejném prostředí dochází k postupnému nárůstu impedance po zhruba první tři týdny a následně se stabilizuje. [1]

Měření EIS významně pomáhá při dalším vývoji Li-Ion baterií, které představují výhodný prostředek pro uchování elektrické energie s co možná největší hustotou. Tyto články disponují možností velmi rychlého nabití a vybití, a zároveň nedochází k tak velkému samovolnému úbytku energie, jako tomu může být u jiných baterií. Bohužel mají i řadu

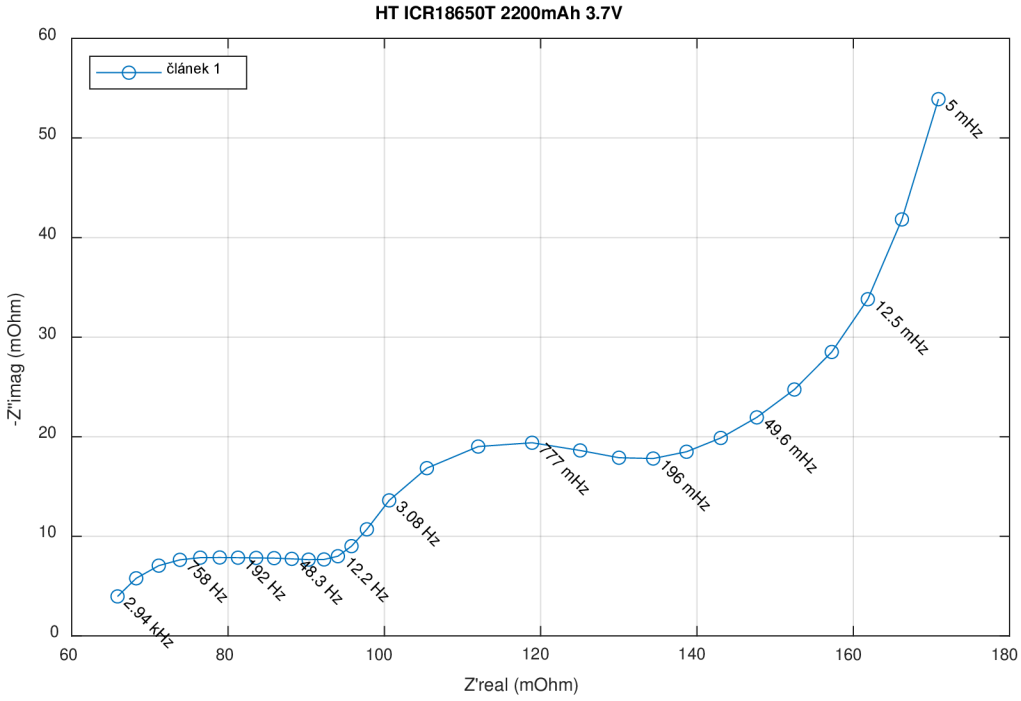
negativních vlastností – vyšší cena, musíme zvážit životnost takového článku v běžném provozu, omezený počet nabití, omezení teplotními podmínkami a bezpečnostní rizika. Pro ochranu správného fungování článku je nutné zajistit chod v přívětivých podmínkách. Pro práci s baterií a všemi jejími články přijde vhod znát parametry určující její stav. Často uváděnými parametry jsou state of charge (SoC), state of health (SoH) a state of function (SoF). Výstupem EIS ale není informace o nabití článku, případně jeho kapacitě, protože to není možné přímo zjistit. Namísto toho se právě tyto uvedené parametry snažíme odhadnout za pomoci modelu impedanční spektroskopie. Pro posuzování kvality článku ale často tato informace nebývá nejdůležitější, nýbrž právě vnitřní odpor článku, tedy impedance, slouží lépe k vlastnímu popisu. Pokud známe hodnotu proudu, který protéká článkem a jeho napětí, je poměrně jednoduché touto metodou impedanci naměřit. Výhodou celé metody je i fakt, že samotný článek, který chceme testovat, není nutné odpojit z příslušného obvodu, jako je tomu u metod jiných. EIS pak provádíme přímo v aplikaci tohoto článku, ale nesmíme opomenout vliv dalších parametrů obvodu a provozních podmínek. V průběhu měření metodou EIS by také nemělo dojít ke změně stavu článku. [2][4]

Pro měření impedančního spektra článku lze zvolit různé metody. Z hlediska přesnosti výsledků se jeví jako vhodný způsob excitace systému sinusovým signálem. V daný moment aplikujeme pouze jednu frekvenci z celého spektra, ve kterém se postupně posouváme. Po dokončení tohoto časově náročného měření získáme celé impedanční spektrum s co nejlepší přesností měření. Dále známe techniky Fourierových transformací, kdy se při měření aplikuje vícero signálů na různých frekvencích současně. Dílčí výsledky pak získáme právě za pomoci algoritmů Fourierových transformací. Tím se celé měření výrazně urychlí a s dnešní pokročilou technikou se lze spolehnout na velmi přesné měření. Může ale docházet k zašumění signálu pokud článek zatížíme až příliš velkou amplitudou. Navíc musíme dbát na to, aby se vzájemně měřené frekvence neovlivňovaly harmonickými frekvencemi. Tyto metody jsou tedy výrazně složitější na použitou techniku. [2][4]

Dále u EIS rozlišujeme potenciostatickou a galvanostatickou metodu. V průběhu měření systém vždy zatěžujeme periodickým sinusovým signálem. Chceme získat informaci o impedanci článku, zůstává nám tedy vybrat si, zda systém budeme excitovat elektrickým napětím, nebo proudem. Vždy si totiž zvolíme, kterou z veličin úmyslně ovlivníme, označíme ji tedy jako primární. Sekundární veličina a její vlastnosti se odvíjejí právě od měřeného systému. Jako potenciostatickou metodu označujeme měření tehdy, když volíme za primární veličinu napětí, galvanostatická metoda využívá zatěžování střídavým proudem. Jak je známo,

experimentálně bylo ověřeno, že pro diagnostiku článků a baterií pomocí EIS, je vhodnější systém ovlivňovat primárně změnou proudu. Následně naměříme jeho odezvu na změně napětí. [4]

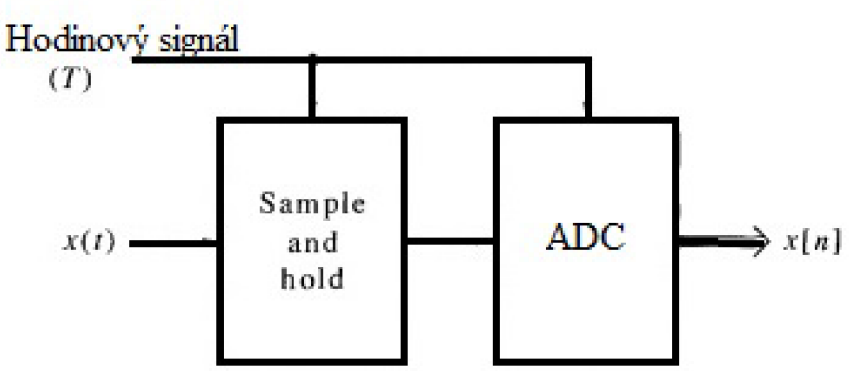
Na Obrázku 1 vidíme průběh reálné a imaginární složky impedance pro článek HT ICR18650T 2200 mAh, na kterém v rámci této práce probíhalo testování nově vzniklé aplikace a referenční měření. Právě takto může vypadat výstup z metody elektrochemické impedanční spektroskopie. Pro jednotlivé frekvence jsme získali impedanční charakteristiku článku.



Obrázek 1 – Referenční měření EIS na článku HT ICR18650T 2200 mAh

2 Měření amplitudy a fáze vzorkovaného signálu

Pro měření amplitudy a fáze vzorkovaného signálu je nejprve třeba porozumět procesu vzorkování signálu. Přesnost tohoto měření ovlivňují podmínky jako např. vzorkovací rychlost, nebo kvantovací chyba. Důležité jsou také transformace signálů – např. jejich násobení, kdy dochází ke změně amplitudy. Víme, že při měření vzniká šum v signálu, od kterého chceme často užitečnou informaci oddělit. Odstup signálu od šumu uvádíme jako SNR. Tato práce se zaměřuje na měření amplitudy a fáze signálu uvnitř elektrického obvodu. Protože HW je limitován, vznikají nedokonalosti a omezení, a to tří hlavních typů. Jedním z nich je saturace, protože napětí na obvodu je vždy v nějakém rozsahu hodnot dle konkrétního obvodu. Dále kvantizace – její chyba, případně šum, způsobený omezeným počtem bitů pro ukládání hodnot. Další omezení je způsobeno nelinearitou různých komponent zapojeného obvodu, např. tolerance tranzistorů, zesilovačů apod. Vzorkování signálu implementují převodníky A/D, které na vstupu přijímají v čase spojitý signál ve formě elektrického napětí. Výstupem tohoto zařízení je sekvence binárních hodnot. Převodníky na vstupu často obsahují sample-and-hold obvody, jejichž schematický diagram vidíme na Obrázku 2. Protože ADC pracuje při hodinovém taktu, je vhodné nejprve analogovou hodnotu podržet po dostatečně dlouhý interval, aby ji následně převodník stihl navzorkovat. Samotná konverze může trvat poměrně dlouho a mezitím může dojít k případnému úbytku napětí. Díky zavedení pomocného obvodu se hodnota napětí nemění a nedojde ke zkreslení. Binární hodnota napětí se ale může lišit až o polovinu kvantizační hladiny, což je nutné zvážit při výběru HW. Uvnitř převodníku může také docházet k průměrování hodnot, což odpovídá filtrování za pomoci dolní propusti. [5]



Obrázek 2 – Sample and hold obvod před vstupem vzorkovaného signálu do ADC [5]

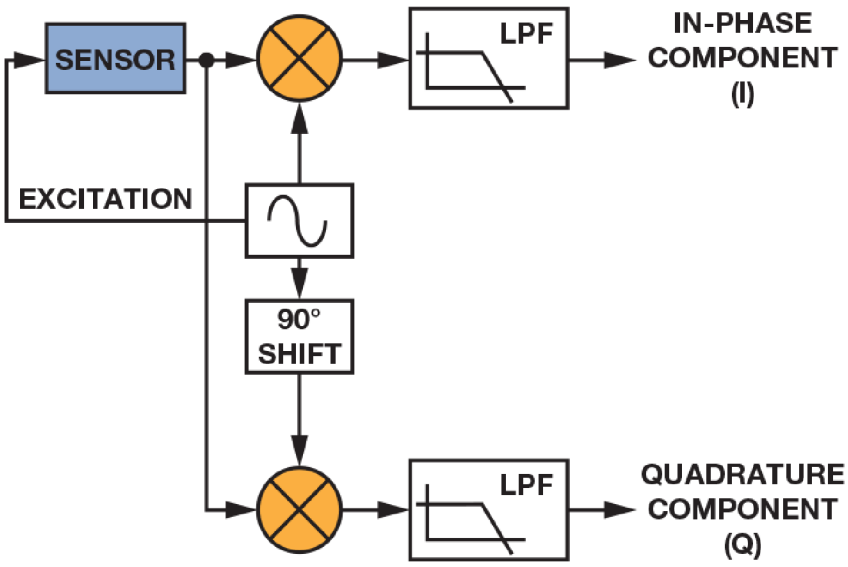
Tato práce využívá systému mikrokontroleru s ADC, který v čase vzorkuje analogový signál, s nímž chceme dále pracovat. ADC blíže určují jejich vlastní parametry. Bitová hloubka určuje kvantovací chybu, která se snižuje se zvětšujícím se dynamickým rozsahem. Převodník vzorkuje analogové hodnoty na co nejvyšší možné rychlosti, což je dále určeno vybraným HW. Výsledný signál vzniklý postupným vzorkováním využijeme pro popis systému a jeho vlastností.

Za pomoci výpočtu synchronní detekce můžeme extrahovat menší signály z šumu signálu, a to pro měření vlastností, jako je například nízký odpor. U mnoha systémů se amplituda šumu zvyšuje úměrně s frekvencí, která klesá k nule. Modulací signálu, tedy posunutím měření do vyšších frekvencí, lze zvýšit SNR a detekovat tak i slabší signály. Existuje vícero metod, jak modulovat excitační signál, např. jej opakovaně vypínat a zapínat (vhodné pro LED diody). Další možností je navrhnout filtr typu pásmové propusti, abychom vyfiltrovali pouze chtěnou frekvenci. Návrh takového filtru může být složitý, proto přistupujeme i k jiným metodám. Můžeme generovat signál sinusového průběhu pro modulaci naměřeného signálu, v některých případech si vystačíme s vlnou obdélníkového průběhu. Takový signál vygenerujeme velmi jednoduše např. analogovým spínačem, nebo tranzistorem MOSFET. Další zajímavostí je fakt, že pokud je mezi referenčním signálem a naměřeným signálem fázový posuv, tedy nemají stejnou fázi, výsledný signál vzniklý přenásobením signálů bude nabývat nižších výstupních hodnot, nežli kdyby měly fázi stejnou. Dostáváme tak možnost vytvořit systém, kde naměřený signál přenásobíme referenčním signálem, který bude zpožděn o 90 °. Ve výsledku získáme dvě komponenty – první bude shodné fáze, druhá posunutá. Z těchto dvou hodnot pak snadno zjistíme magnitudu a fázi signálu dle následujícího vzorce (1) a (2). [6]

$$Magnituda = \sqrt{(I^2 + Q^2)} \quad (1)$$

$$Fáze = \tan^{-1}\left(\frac{Q}{I}\right) \quad (2)$$

V programu tohoto systému výpočet synchronní detekce využívá, na Obrázku 3 vidíme schéma probíhající synchronní detekce. Excitační signál znázorňuje spínání rezistoru PWM, tedy odběr sinusového proudu z baterie. Senzorem je pak měření napětí na baterii. Pro potlačení spínací podstaty generování sinusového proudu probíhá průměrování naměřených hodnot ADC. Jako dolní propust program využívá součet hodnot přes 9 period signálu. První perioda se přeskakuje z důvodu možné vyšší chybovosti měření. O konkrétní implementaci výpočtu synchronní detekce dále hovoří kapitola 4.3.



Obrázek 3 – Schéma synchronní detekce s posunutím jedné z komponent o 90 ° [6]

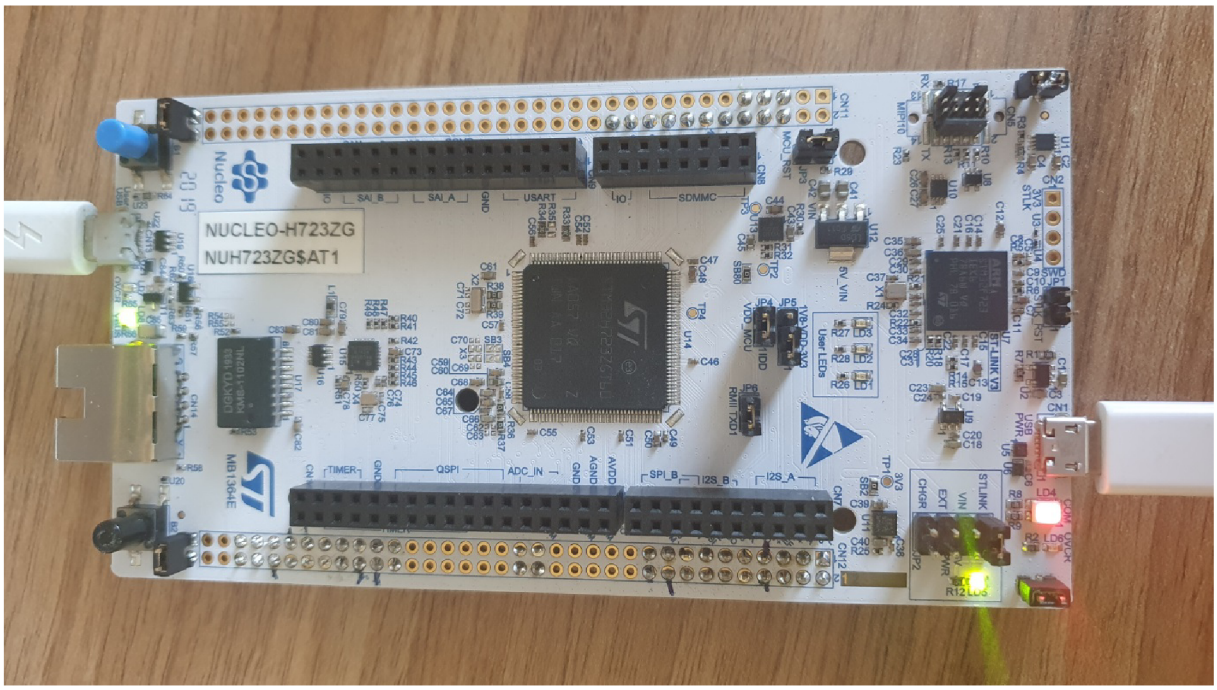
3 Popis HW

Pro potřeby této práce bylo nutné zvolit takový HW, abychom s jeho pomocí dosáhli co nejpřesnějších výsledků měření. Základní funkci představuje generování sinusového průběhu proudu PWM modulací zatěžovacího rezistoru a měření napěťové odezvy článku. Pro konfiguraci měření a interpretaci výsledků dále potřebujeme přívětivé komunikační rozhraní. Doplňkové periferie HW zaručí měření teploty připojeného článku pro kontrolu stavu, a to skrze termistor a digitální snímač. Další nárok stanovuje nutnost ukládat velké množství dat z ADC, HW tedy musí obsahovat dostatečně velké množství paměti. Systém bude nutné v průběhu řádně testovat a ladit a zvolený HW jednoduše konfigurovat.

Generování sinusového proudu za pomoci PWM modulace zatěžovacího rezistoru zajistí přesně nastavené HW časovače. Pro konfiguraci frekvence v co největším rozsahu potřebujeme časovač alespoň 32bitový. Následné měření napěťové odezvy článku zajistí 2 ADC. Převodníky musí stihnout navzorkovat dostatečný počet co nejpřesnějších hodnot i na vyšších frekvencích. Pro co nejrychlejší přenos dat mezi HW a nižší zátěž systému využijeme DMA. Měření teploty termistorem obslouží další ADC, na nějž nevznášíme vyšší požadavky. Zvolený digitální snímač připojíme za pomocí sběrnice SPI. Stabilní komunikaci pro přenos dat oběma směry zajistí vysokorychlostní USB. V SW programu využijeme přerušení vyvolaných periferiemi, procesor proto musí běžet na dostatečně vysoké frekvenci pro zpracování dat a pro výpočty nad čísly s plovoucí řádovou čárkou je zapotřebí jednotky pro zpracování těchto datových typů.

3.1 Volba procesoru

Všechny tyto požadavky splňuje mikrokontroler Nucleo-H723ZG, který vidíme na Obrázku 4. Pro vývoj firmwaru na této vývojové desce byl použit konkrétně procesor STM32H723ZGT6U v pouzdru LQFP144. Jádro tvoří ARM Cortex-M7, které operuje na rychlosti až 550 MHz, a které obsahuje matematický koprocesor.



Obrázek 4 – Zvolený MCU napájený pomocí USB

Zvolený procesor disponuje dvěma ADC s hloubkou 16 bitů a jedním ADC s rozlišením 12 bitů. Teoretická maximální rychlost dvou souběžně běžících ADC na nejvyšší bitové hloubce dosáhne až 1 MSPS. K dispozici má kanály DMA, které využijeme při odbavení ADC. Pro konfiguraci kmitočtů nám slouží vícero oscilátorů a registry děliček a násobiček. MCU lze napájet skrze USB, které také využijeme jako komunikační rozhraní. Program využije dostupnou paměť – 1 MB vestavěné FLASH paměti, celkem 564 KB SRAM. Dále FW spustí jeden časovač s PWM a jeden 32bitový časovač pro konfiguraci frekvence. [7] Další výhodou zvoleného mikrokontroleru tvoří vývojové prostředí STM32CubeIDE přímo od výrobce STMicroelectronics. Díky němu lze MCU jednoduše nakonfigurovat včetně všech jeho periferií, vygenerovat základní strukturu kódu, ladit systém atd.

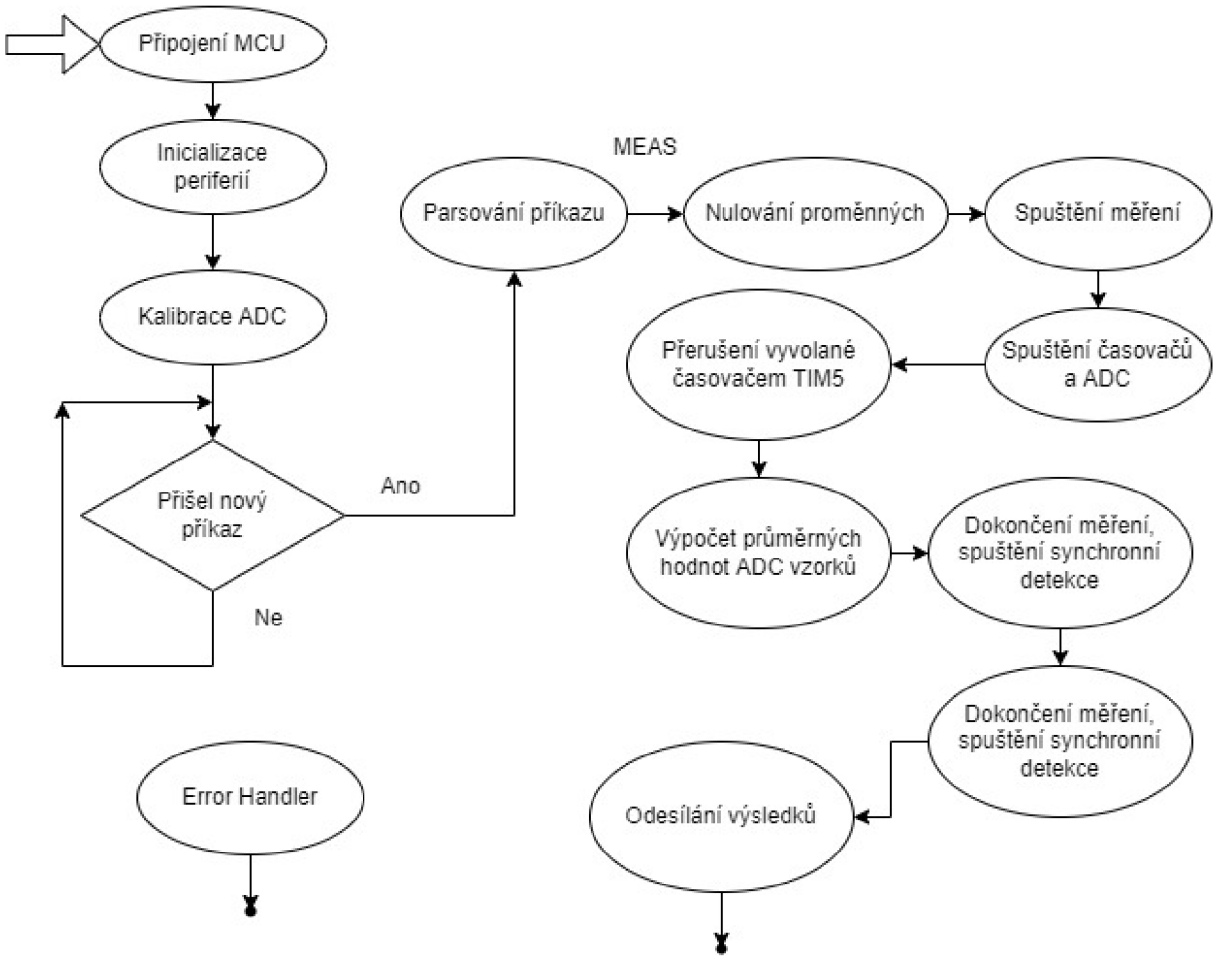
Dalšími vhodnými procesory od výrobce STMicroelectronics mohou být i další MCU ze série STM32H7, které často obsahují ADC s rozlišením 16 bit, dostatek paměti RAM a frekvenci hodin v řádu stovek MHz. Některé mikrokontrolery ze série STM32F3 by sice splnily požadavek na ADC, ale programovací paměť je ve srovnání velmi malá, stejně jako maximální frekvence hodin. Pokud se podíváme na výrobce NXP Semiconductors, který nabízí často ADC i s bitovým rozlišením 20 bit, některé MCU by splnily požadavky i na paměť, ale co se týče maximální hodinové frekvence, opět nedosahuje takových hodnot, jako série STM32H7. Výrobce Texas Instruments například nabízí MCU TMS320F28384S, který

obsahuje 16bitový a 12bitový převodník s maximální frekvence hodin procesoru 200 MHz. Na základě těchto informací byl vývoj proveden právě na MCU ze série STM32H7.

4 Návrh firmware

Celá koncepce firmwaru je vcelku jednoduchá – jedná se o jednovláknový lineární program, který se opírá o možnosti zvoleného HW. Právě díky němu lze využít přerušení od HW periferií a rychlého přenosu dat pomocí DMA. Komunikaci mikrokontroleru s dalšími systémy poskytuje vysokorychlostní USB. Program tedy obsluhuje pouze dvě základní věci – komunikaci a konfiguraci nebo spouštění HW periferií.

Životní cyklus programu vidíme na Obrázku 5. Při prvním spuštění se inicializují prvky HW a provádí se kalibrace všech ADC. Od této chvíle procesor čeká ve smyčce na příkazy, které mohou přijít přes terminál virtuálního sériového portu, nebo přes obslužnou Matlab aplikaci. Pokud program přijme požadavek na spuštění měření, provede jej a následně odesílá naměřená data. Výsledkem se stávají vypočítané fyzikální veličiny – impedance a fázový posuv, případně průběh naměřených vln. V průběhu měření dále firmware umí odesílat stavové zprávy např. o momentálním stavu nebo nastavení.



Obrázek 5 – Vývojový diagram chodu programu

Samotný HW, pokud je spuštěn, se stará o generování sinusového proudu (viz 4.1) za pomoci dvou časovačů. Časovač TIM2 zajišťuje generování PWM signálu, TIM5 pak řídí spínání TIM2 dle požadované frekvence. TIM5 též zapíná a vypíná ADC s DMA pro měření napětí a proudu článků (viz 4.2). Pokud vše proběhne úspěšně, na řadu se dostává výpočet synchronní detekce (viz 4.3), kde z detekovaných průběhů proudu a napětí program vypočítá impedanci a fázový posuv pro nastavenou frekvenci střídavého proudu.

O kalibraci ADC se stará sám HW na pokyn SW, což můžeme vidět na následující Ukázce kódu 1. Jedná se o funkci, která se vykoná při startu systému, ale je možné ji volat i po delší době, kdy HW pracuje, a to pro opětovnou kalibraci. Uvnitř postupně dochází ke kalibraci všech tří ADC, a to pro režim, ve kterém pracují. Pokud by došlo k chybě při kalibraci, funkce vyvolá chybový stav. Pro jistotu dále procesor čeká 10 ms, aby nedošlo k chybnému vzorkování před dokončením kalibrace.

```
1  void Calibrate_ADCs()
2  {
3       if (HAL_ADCEx_Calibration_Start(&hadc1, ADC_CALIB_OFFSET,
   ADC_SINGLE_ENDED) != HAL_OK) { Error_Handler(); }
4       if (HAL_ADCEx_Calibration_Start(&hadc2, ADC_CALIB_OFFSET,
   ADC_DIFFERENTIAL_ENDED) != HAL_OK) { Error_Handler(); }
5       if (HAL_ADCEx_Calibration_Start(&hadc3, ADC_CALIB_OFFSET,
   ADC_SINGLE_ENDED) != HAL_OK) { Error_Handler(); }
6
7       HAL_Delay(10);
8  }
```

Ukázka kódu 1 – Kalibrace všech ADC

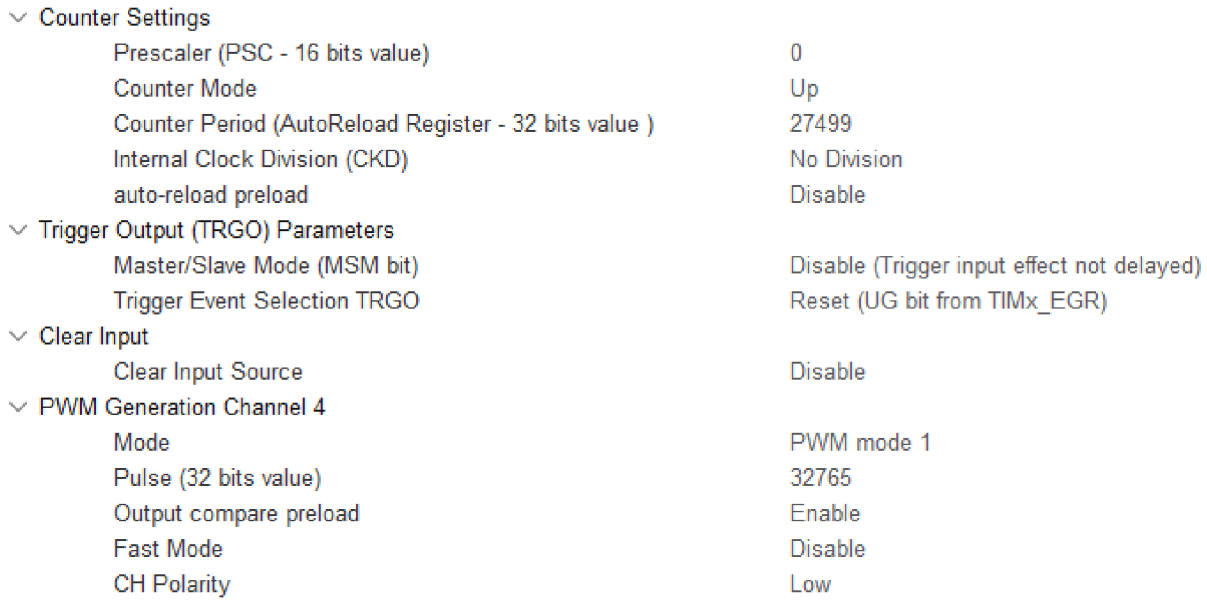
4.1 Generování sinusového proudu

Základní funkci celého systému tvoří generování sinusového průběhu proudu PWM modulací zatěžovacího rezistoru. Toho bylo docíleno za pomocí časovačů – jednoho pro samotné generování sinusového proudu, druhý slouží k obsluze systému na zvolené frekvenci.

4.1.1 Nastavení časovačů

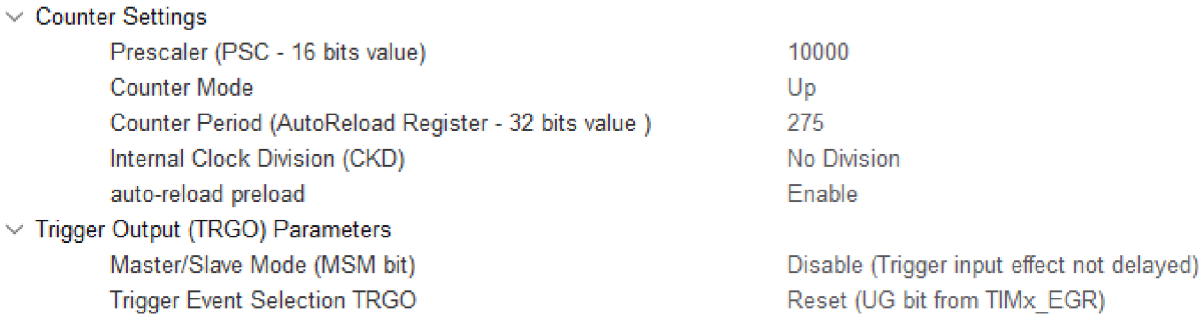
Nejdůležitější součást celé koncepce programu tvoří dva časovače, zejména pak časovač TIM5, který zajišťuje běh HW na předem nastavené frekvenci. Hodinový signál u obou časovačů zajišťují vnitřní hodiny mikrokontroleru, pracují tak na frekvenci 275 MHz, která se odvíjí od *APB1*. O nastavení hodinových kmitočtů dále rozpravuje kapitola 4.4. U časovače

TIM2 byl zvolen kanál 4 pro generaci signálu PWM, jeho vývod je na PB11. Obrázek 6 demonstruje jeho nastavení. Za chodu programu se konfigurace nemění, pouze se aktualizuje jeho porovnávací registr. Ten určuje střídu signálu (činitel plnění) PWM, která je daná jeho poměrem k registru vrcholu čítače (AutoReload). Podle hodnoty vrcholu čítače je nutné nastavit amplitudu hodnot tabulky sinu, která je předgenerovaná v programové paměti procesoru. To zaručuje, že střída generovaného PWM signálu se bude měnit o 0 do 100 %.



Obrázek 6 – Nastavení časovače TIM2 s PWM

Na následujícím Obrázku 7 pak vidíme nastavení hlavního časovače TIM5. Ten byl zvolen kvůli 32bitovému registru čítače a určuje přesné časování měřicího jádra programu. V průběhu kód přenastavuje jeho registry vrcholu čítače a nastavení předděličky podle požadované frekvence měření impedance článku. Čítač běží volně a v příslušných okamžicích generuje přerušení, ve kterém se vkládá nová komparační hodnota do časovače TIM2, čímž se generuje kosinusový odběr proudu. Dále se zde provádí zpracování dat naměřených ADC převodníky. Při konfiguraci časovače proto nelze opomenout parametr globálního přerušení, které pro správný chod musíme povolit.



Obrázek 7 – Výchozí nastavení časovače TIM5

4.1.2 Nastavení požadované frekvence

Velmi důležitou část kódu tvoří funkce pro nastavení frekvence časovače 5, protože se od kmitočtu odvíjí doba, jakou celý proces bude trvat, a tedy i celkový počet vzorků z AD převodníků, ze kterých se počítají průměrné hodnoty. S vyšší frekvencí a tedy nižším počtem vzorků vzrůstá míra nepřesnosti. Proto bylo zapotřebí určit maximální hodnotu frekvence, jakou lze nastavit pro výpočet. U vyšších frekvencí generování proudu (>100 Hz) je už počet naměřených vzorků velmi malý. Výpočet střední hodnoty napětí a proudu PWM signálu už je pak velmi hrubý. Proto program provádí pro vyšší frekvence zkrácení počtu vzorků pro generování sinu podle referenční tabulky. Principem je indexace s nastavitelným krokem. Pro nízké frekvence je krok 1 a délka tabulky cosinu je 128 hodnot. Při použití kroku 2 se používá každá 2. hodnota z tabulky. Jedna vlna cosinu se pak skládá z 64 hodnot. Pro správnou funkčnost musí být velikost kroku mocninou 2. Díky snížení frekvence aktualizace PWM modulátoru získává ADC více času na vzorkování a měření se tak opět zpřesňuje. Pokud požadavek zní: nastav frekvenci na hodnotu *x*, reálná frekvence časovače TIM5 je přenásobena délkou srovnávací tabulky, tedy hodnotou 128. Ve výsledku to znamená, že průběh celé jedné vlny se vzorkoval na právě zvolené frekvenci *x*. Nastavení frekvence v programu vidíme na následující Ukázce kódu 2, kde dle řádu frekvence nastavuji hodnoty dvou registrů časovače, a to předděličku a periodu. Nastavuje se také krok, dle kterého se prochází tabulka referenčních hodnot kosinu. V ukázce kódu záměrně vynechávám některé méně podstatné řádky – některá pásma frekvencí, výchozí akce při překročení pásma (odešle se informace uživateli o stávající frekvenci a zpráva, že jeho zvolená hodnota frekvence nelze nastavit), nastavení globálních proměnných, které vycházejí z nastavovaných registrů, atd.

```
1  void Set_Frequency(double freq)
2  {
3     int prescaler = 1;
4     int period = 2750000;
5     table_step = 1;
6     long lfreq = freq*1000;
7     freq *= TABLE_LEN;
8
9     switch (lfreq)
10    {
11          case 1 ... 9:              //1 mHz... 9 mHz
12                period = 2750000/freq;
13                prescaler = 100;
14                break;
15          case 300000 ... 599999:
16                table_step = 4;
17                period = 275000000/(freq/table_step);
18                break;
19          case 600000 ... 1200000:   //600 kHz...1.2 kHz
20                table_step = 8;
21                period = 275000000/(freq/table_step);
22                break;
23    }
24    __HAL_TIM_SET_AUTORELOAD(&htim5, period - 1);
25    __HAL_TIM_SET_PRESCALER(&htim5, prescaler - 1);
26 }
```

Ukázka kódu 2 – Funkce pro nastavení frekvence příslušnými registry časovače

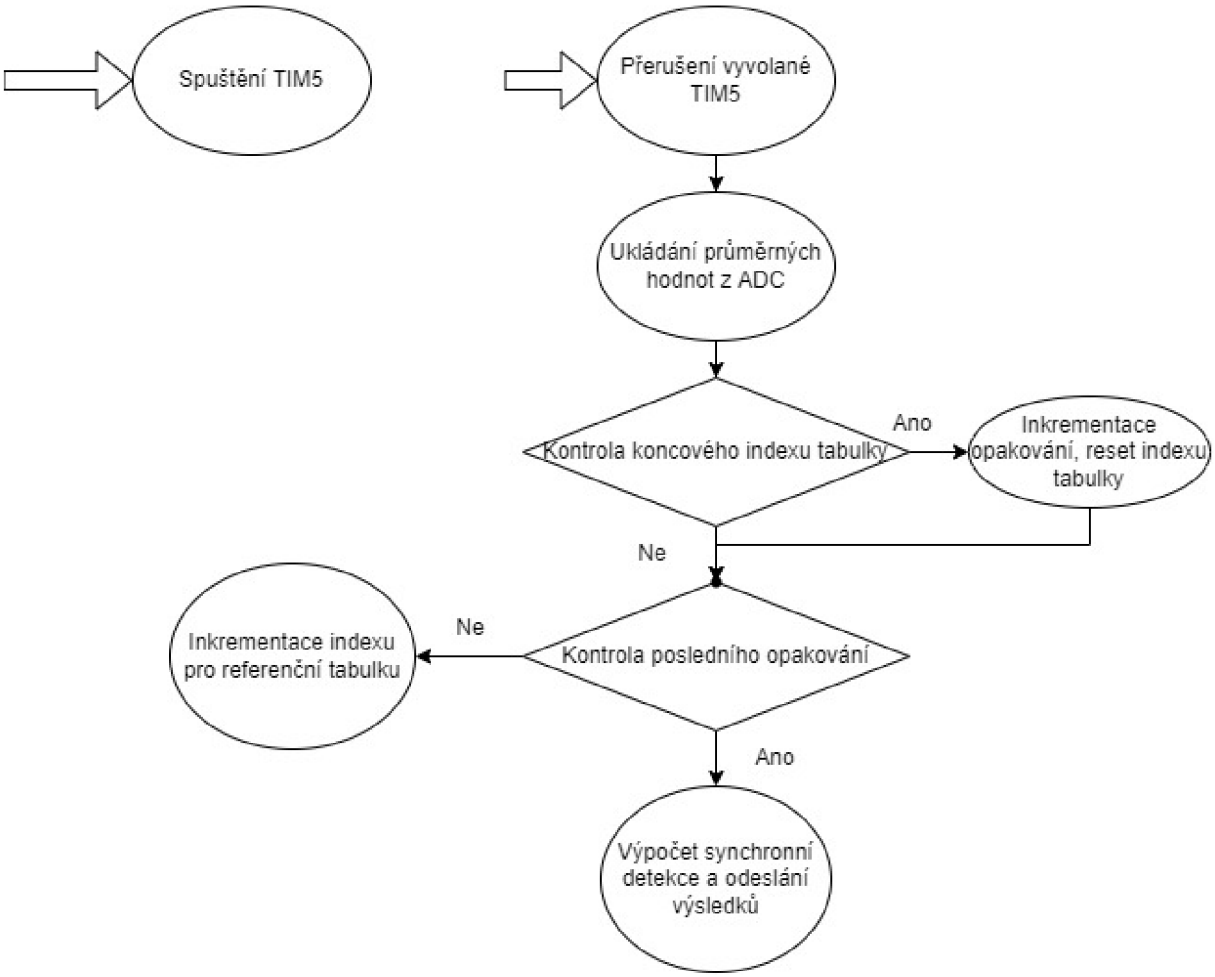
Vstupním parametrem funkce změny frekvence je hodnota požadované frekvence. Časovač je ale omezen parametry 16bitové předděličky a 32bitového čítače, jejichž hodnoty se mění dle řádu požadované frekvence. Výsledný kmitočet tak nekoresponduje pokaždé zcela přesně s tím požadovaným jedna ku jedné, a proto hodnotu program odesílá zpět pokaždé, když ho program správně přenastaví. V případě, že vstupní hodnota se nachází mimo povolený interval, frekvence se nemění, o čemž uživatele informuje krátká zpráva.

4.1.3 Přerušení časovače TIM5

Pokud časovač běží TIM5, při každém uplynutí nastavené periody se vyvolá přerušení, ve kterém se opakovaně spouští ADC1 a ADC2 společně s DMA přenosem výsledků. Zároveň se nastaví porovnávací registr u časovače TIM2 pro chod jeho PWM dle inkrementovaného indexu, pomocí něhož program sahá do předem vygenerované tabulky hodnot průběhu vlny *cos (t)*. V tomto přerušení dále dochází k inkrementaci indexů potřebných k obsluze PWM a

k průběžnému přepočtu naměřených vzorků z obou ADC. Pro DMA přenos byl zvolen duální synchronní režim. 16 bitová data z ADC1 a ADC2 jsou v tomto režimu uložena v společném 32bitovém slově, který DMA řadič přesunuje do bufferu v paměti. Tím je zajištěna synchronnost měření proudu a sníží se zátěž sběrnic procesoru. V programu je ale nutné data oddělit. To se provádí jednoduchým vymaskováním a bitovým posunem. Ve chvíli, kdy celý proces doběhne do svého konce tím, že index přeskočí nad svoji maximální hodnotu, se zastaví i oba časovače a přistupuje se dále k výpočtu synchronní detekce (viz 4.3).

Jak je vidět na Obrázku 8, po spuštění časovače TIM5 se celkem 10krát (v kódu konstanta *TOTAL_REPETITIONS*) opakuje měření (hodnota je nastavena na 10 period generovaného cosinu), kdy se do porovnávacího registru PWM časovače TIM2 nahraje hodnota z tabulky průběhu vlny *cosin(t)* na aktuálním indexu. Program obsahuje tabulku s délkou 128 prvků, takže časovač TIM5 vyvolá přerušení celkem 1280krát. S každým přerušením obě ADC vzorkují nové hodnoty, které se za pomoci DMA ukládají do paměti a procesor je následně v přerušení časovače zpracuje. Ve smyčce totiž postupně prochází všechny připravené vzorky, pro každé ADC zvlášť procesor počítá průměrnou hodnotu signálu přes interval měření. Následně ukládá průměrnou hodnotu do dvourozměrného pole pro pozdější zpracování.



Obrázek 8 – Životní cyklus časovače TIM5, obsluha přerušení

Časovač TIM5 také v průběhu nastavuje hodnoty dvou příznaků, a to jak pro úspěšné naměření jednotlivých vln, ale také pro úspěšné dokončení celého výpočtu. Tyto příznaky využívá stavový automat komunikace po USB, aby mikrokontroler získaná data odeslal po virtuálním sériovém portu tam, odkud přišel požadavek na výpočet.

4.2 Měření napětí a proudu článku

K měření napětí a proudu článku slouží dva AD převodníky ADC1 a ADC2. Běží v synchronním módu, aby nedocházelo k fázovému posuvu naměřených hodnot. V následujících podkapitolách rozebírám zapojení HW, nastavení ADC a DMA, prvotní způsob využití ADC, od kterého bylo nutné ustoupit a dále robustní řešení vzorkování.

Deska měření obsahuje hlavní části pro měření impedance, kterými jsou: zatěžovací rezistor spínaný PWM modulací, snímání napětí článku a snímání proudu rezistorem. Dále je deska obsahuje doplňkové funkce, a to měření teploty, které je prováděno dvěma metodami.

Rezonátor pro obvod reálného času ani záložní článek pro jeho napájení nebyly použity a na desce nejsou osazeny.

PWM signál je generován čítačem TIM2 na výstupu PB11. K vývodu je připojen push-pull budič MCP1401 [8] pro MOSFET tranzistor BUK7880-55A [9], Tento tranzistor byl vybrán s ohledem na malé napětí článku a poměrně malý pracovní proud. Požadována byla vysoká rychlost sepnutí a vypnutí, která je v podmínkách dle katalogového listu 60/26 ns. Druhým hlediskem je malá odpor v sepnutém stavu.

Doba sepnutí a vypnutí tranzistoru je prodloužena o průchozí zpoždění budiče a trvání náběžné respektive týlové hrany signálu. Tyto doby činí 54/50 ns. Naměřené hodnoty vzestupné a sestupné hrany jsou cca 4krát pomalejší. To způsobuje malé ovládací napětí U_{GS}, které činí pouze 5 V oproti referenčním podmínkám katalogového listu 10 V. Nižší napětí je použito, protože je přímo dostupné na vývojové desce mikrokontroleru. Také rezistor R8 100 Ω je větší oproti referenční hodnotě 10 Ω.

Chyba činitele plnění PWM modulace způsobená dobou sepnutí t_{ON} a vypnutí t_{OFF} je dána vztahem:

$$e = (t_{ON} + t_{OFF}) \div T_{PWM} = \left(400^{e^{-9}} + 400^{e^{-9}}\right) \div e^{-4} = 0{,}8\,\% \quad (3)$$

Vzhledem k tomu, že doby t_{ON} a t_{OFF} jsou téměř stejné, se chyba projeví pouze při požadavku na generování hodnot téměř nulových a téměř maximálních.

4.2.1 Schéma zapojení HW

Napětí Li-Ion článku se snímá odporovým děličem R2/R3. Dělicí poměr přizpůsobuje rozsah napětí článku 0 až 4,2 V rozsahu ADC převodníku 0 až 3,3 V. Voleny byly záměrně vysoké hodnoty odporu, aby se omezilo vybíjení článku. Mezní proud děličem je 33 μA. Za děličem následuje impedanční přizpůsobení, které provádí snímač U2B s operačním zesilovačem MCP6022. Vstup zesilovače je ochráněn proti přepětí Schottkyho diodou D1. Signál je pak veden přes RC článek na vstup ADC1. Celé schéma MCU a jeho piny ukazuje Obrázek 9. Pro měření vznikla deska plošných spojů se zvolenými periferiemi, jejíž vývody odpovídají zapojení na schématu. DPS vidíme na Obrázku 10.

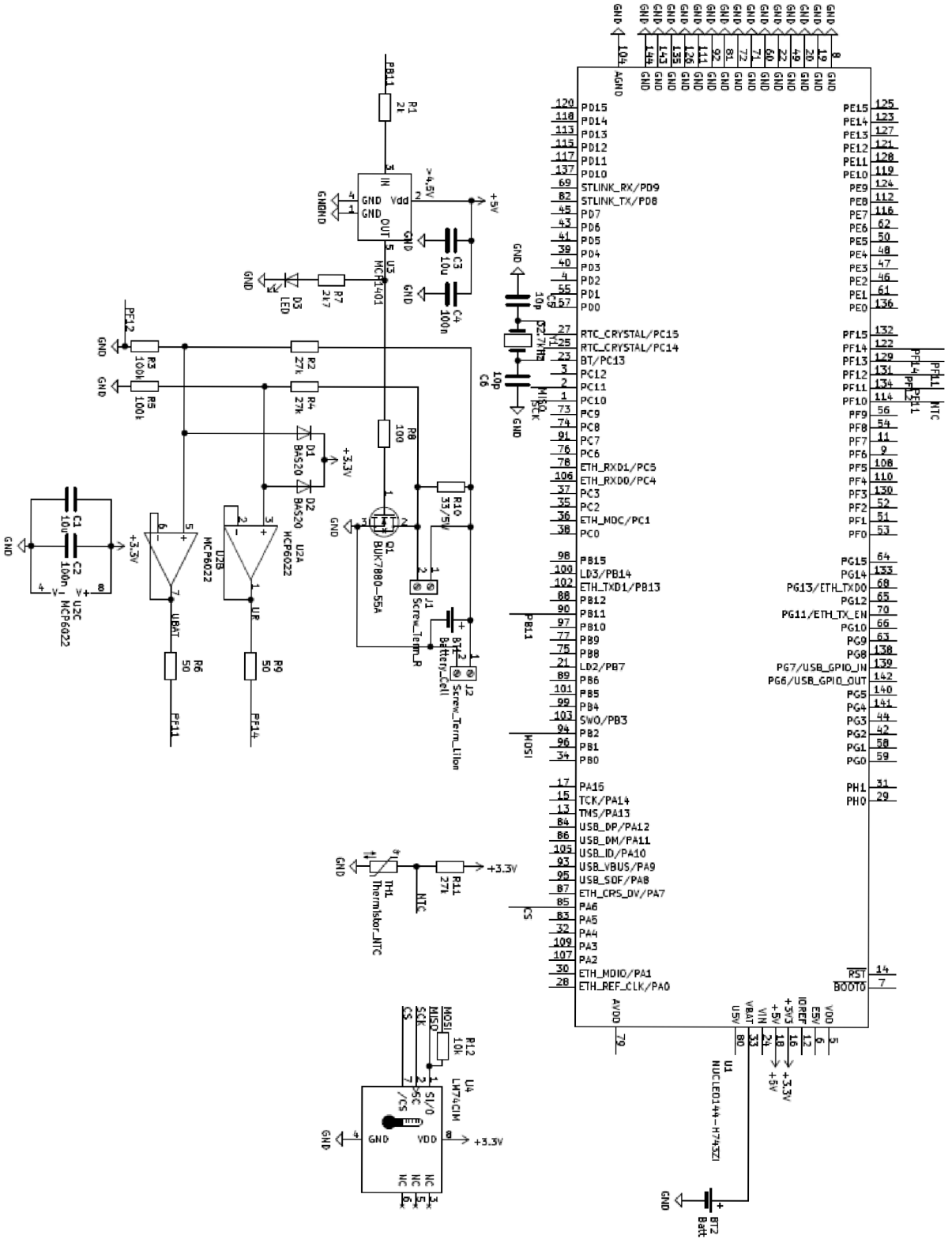
Rozlišení kanálu je dáno 16bitovým převodem podle vztahu

$$d_U = 4{,}2 \div 2^{16} = 64{,}09\ \mu\text{V} \quad (4)$$

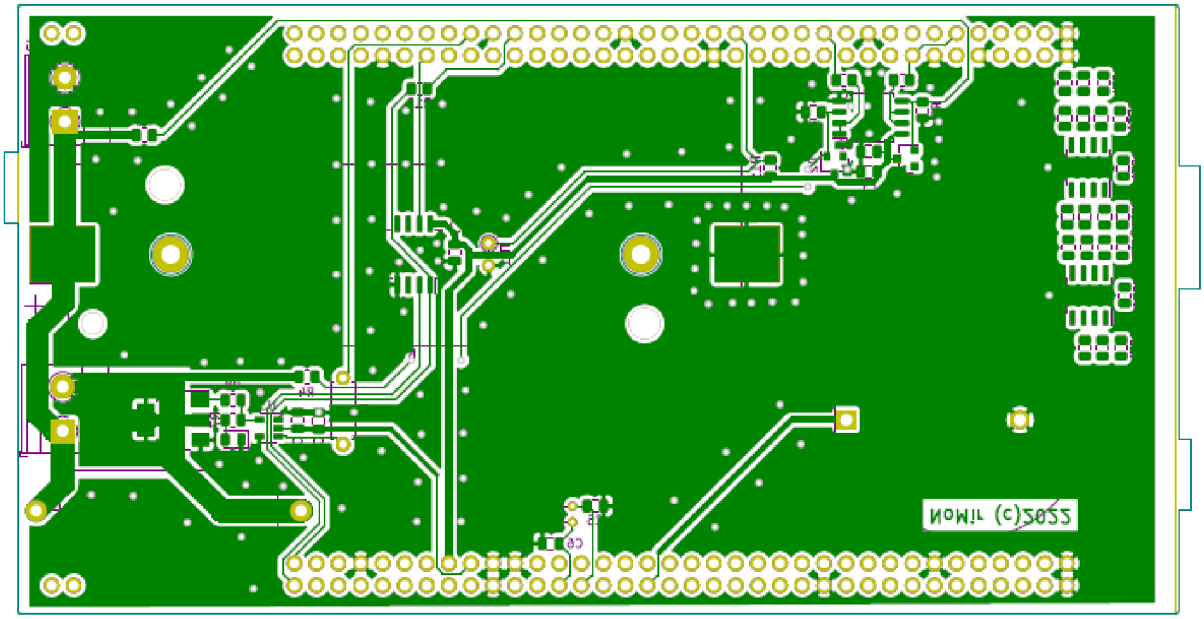
ADC1 je nastaven do single-ended režimu a využívá vývod PF11 pouzdra mikrokontroleru.

Modulační proud z článku Li-Ion potřebný k měření jeho vnitřní impedance je měřen nepřímo jako úbytek napětí na výkonovém rezistoru R10. Tento rezistor představuje balanční rezistor BMS. Jeho hodnota byla zvolena na 33 Ω, což odpovídá meznímu proudu $I = U_{max} / R = 4{,}2 \div 33 = 127{,}3$ mA. V průběhu řešení byla hodnota rezistoru zmenšena na polovinu paralelním připojením druhého kusu rezistoru. Získáme tak silnější signál pro vyhodnocování. Dosažitelný mezní proud je tedy 245,5 mA.

Zapojení vstupu ADC2 je identické jako u měření napětí. Ve vybíjecím obvodu článku se měří napětí uzlu mezi rezistorem R10 a spínacím tranzistorem Q1. Toto napětí je opět zmenšeno děličem R4/R5, omezeno ochrannou diodou D2 a impedančně přizpůsobeno sledovačem U2A. ADC2 je nastaven do diferenčního režimu a diferenciální vstupy jsou připojeny k vývodům PF13 a PF14. Převodník tedy převádí rozdílové napětí na R10. Hodnota je platná pouze při sepnutém tranzistoru. To musíme zohlednit při vyhodnocování dat a nízké hodnoty odprahovat.



Obrázek 9 – Schéma zapojení HW, piny MCU



Obrázek 10 – DPS pro potřeby měření obsahuje mimo jiné termistor a digitální snímač pro měření teploty na článku

4.2.2 Rychlost konverze

Rychlost samotného vzorkování určuje frekvence ADC a počet cyklů potřebných ke konverzi analogové hodnoty a počet cyklů, po které je signál zachytáván sample-hold obvodem. Dle [7] pro dodržení přesnosti při 16bitovém převodu při daných podmínkách by vstupní frekvence převodníku neměla přesáhnout 10 MHz. Hodinový signál je vůči jádru procesoru nastaven jako asynchronní. Tomu se uzpůsobily registry děliček při vstupu hodinového signálu do multiplexoru ADC, viz kapitola 4.4. Pro nejvyšší rychlost vzorkování zvolíme parametr sampling time 1,5 cyklu. Z důvodu použití 16bitového rozlišení ADC k celkové periodě vzorkování přičteme dalších 8,5 cyklů dle manuálu [10], dohromady tedy 10 cyklů. Další 2,5 cyklu se vkládá mezi převody při současném vzorkování ADC1 a ADC2. Díky tomu dosáhneme vzorkovací frekvence 800 kSPS. Nejvyšší možná rychlost pro mikrokontroler v pouzdru LQFP144 činí 1 milion vzorků za sekundu. Rozdíly v rychlostech konverze spatříme v Tabulce 1, která počítá s maximální frekvencí u jednotlivých konfigurací – ty zobrazuje Tabulka 2. [7]

Tabulka 1 – Maximální vzorkovací rychlost dvou souběžně běžících ADC u balíků LQFP [7]

Resolution	LQFP100		LQFP144		LQFP176		LQFP208	
	Direct	Fast	Direct	Fast	Direct	Fast	Direct	Fast
8	8.33	7.14	8.33	7.14	8	7	8.33	7.14
10	5.86	5.13	4.57	4.57	4.29	4.29	3.57	3.57
12	4.38	3.89	2.13	2.13	2	2	1.88	1.88
14	2.33	2.33	1.67	1.67	1.33	1.33	1.44	1.44
16	1.8	1.8	1	1	0.7	0.7	0.9	0.9

Tabulka 2 – maximální frekvence ADC pro různé balíčky (VREF=VDDA=3 V, T=25°C) [7]

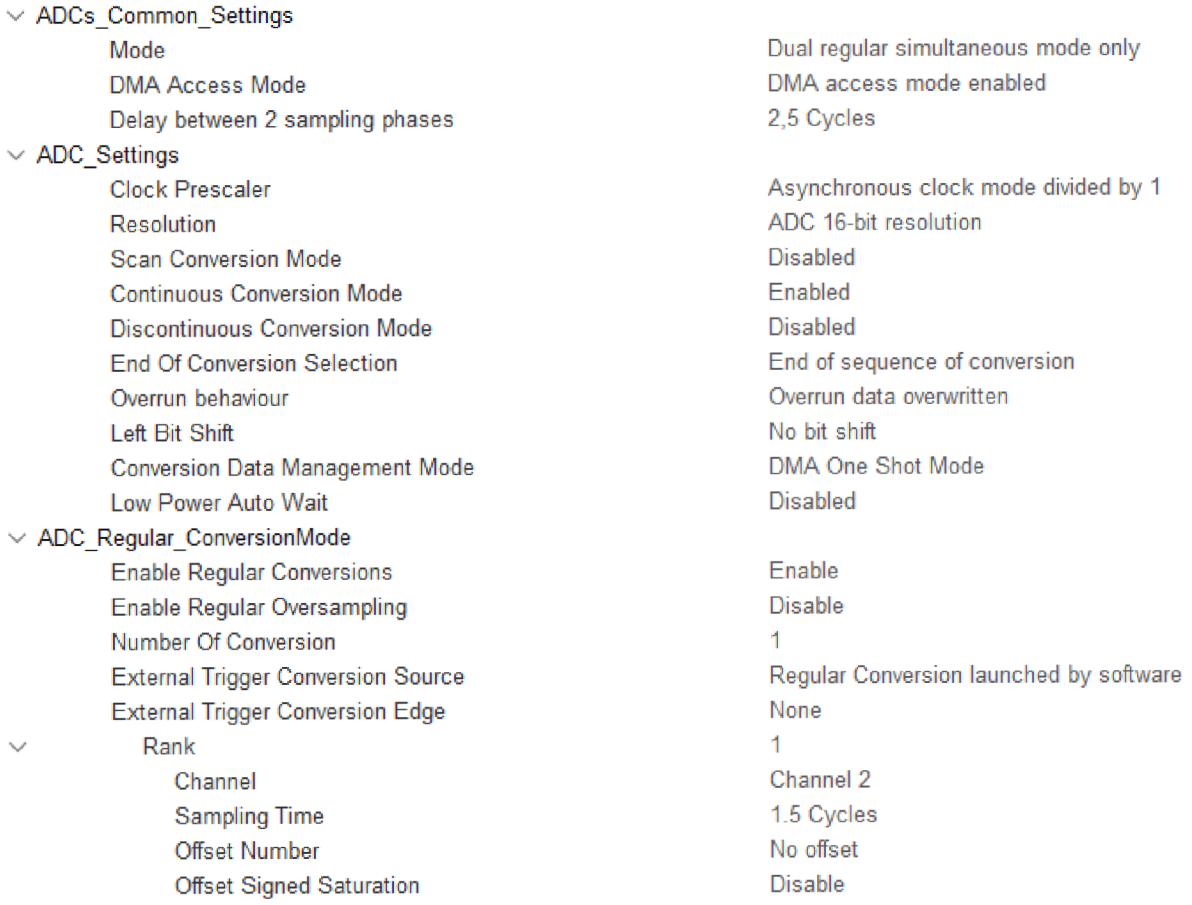
Number of ADCs	Resolution	BGA100	BGA169	BGA176	BGA240	LQFP100	LQFP144	LQFP176	LQFP208
1	8	50	50	50	50	50	50	50	50
	10	50	50	50	47	47	36	32	39
	12	50	50	50	38	38	29	20	19
	14	49	49	49	31	24	16	15	15
	16	33	38	39	25	19	12	10	10
2	8	50	50	50	50	50	50	48	50
	10	50	50	50	43	38	31	28	25
	12	50	50	50	34	35	17	16	15
	14	40	49	49	29	21	15	12	13
	16	26	37	38	22	19	10	7	10
3	8	50	50	50	50	50	38	33	50
	10	50	50	50	42	38	30	28	20
	12	50	50	50	34	25	16	15	15
	14	32	49	49	25	20	12	10	12
	16	25	35	37	19	18	7	7	7

4.2.3 Výsledné řešení a konfigurace

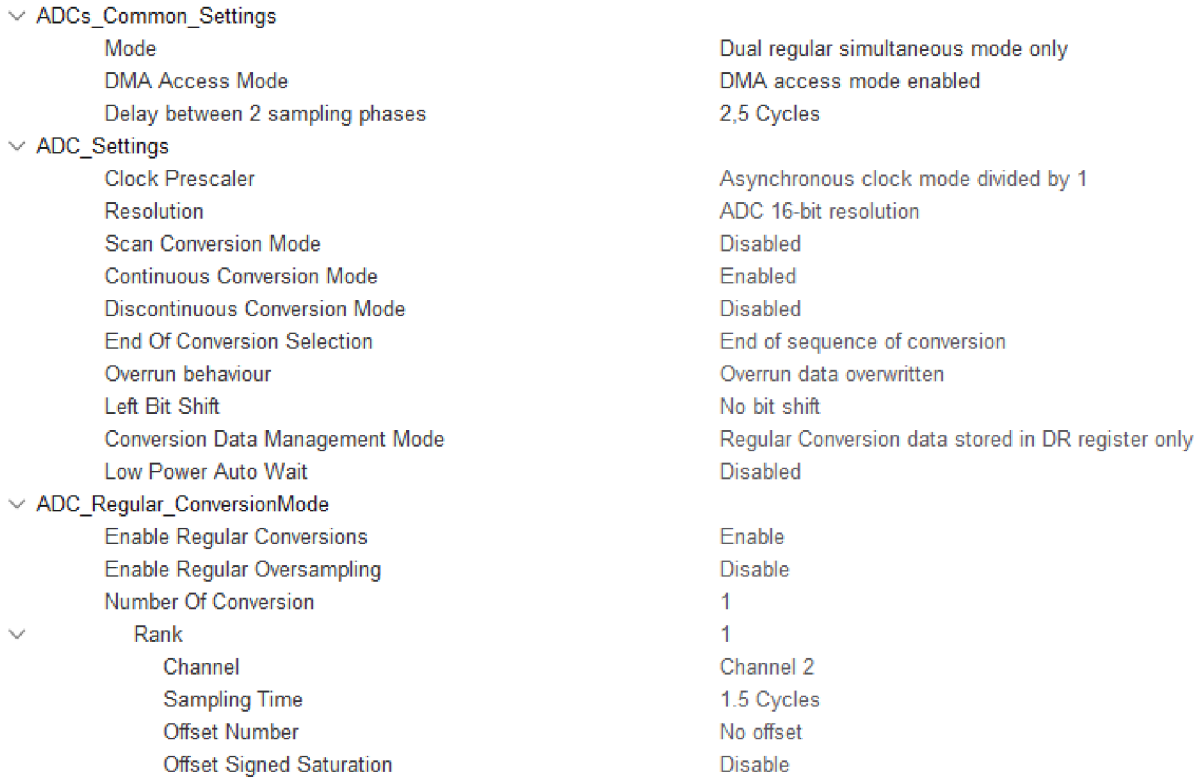
Přerušení procesoru od ADC lze vyvolat při ukončení konverze ADC převodníku nebo při naplnění paměťové oblasti DMA řadičem. Finální verze programu tato přerušení nevyužívá. Při prvních pokusech zprovoznění vzorkování na zvolené frekvenci však přerušení pomohlo zjistit jak spolu periferie – časovače a převodníky – souběžně pracují, a jak správně zajistit to, aby vždy ADC stihly navzorkovat dostatečný počet hodnot napětí a proudu. V nezávislém módu převodníky ADC1 a ADC2 nestíhaly vzorkovat teoreticky vypočtený počet hodnot, který vycházel z frekvence časovače a převodníku. Podrobnou analýzou bylo zjištěno, že doby konverze a DMA transferu jsou po startu měření proměnlivé a některé výrazně delší. Během několika převodů se časy stabilizují a pak převody probíhají víceméně korektně. Za tímto účelem byly v přerušení od časovače TIM5 ukládány v různých bodech vykonávání

přerušovací rutiny stavy čítače. Nestejnoměrnost časování pravděpodobně způsobuje používání cache pamětí a sdílením sběrnic paměti procesorem a periferiemi. Přerušení od DMA se tak nestihlo vyvolat dříve, než časovač TIM5 napočítal do maximální hodnoty svého čítacího registru. Průměrné hodnoty se měly kalkulovat a ukládat v přerušení DMA, a tak byl výsledný buffer často prázdný. Zavedení rezervy do očekávaného počtu převedených hodnot problém neřešilo dobře. Rezerva musela být veliká a snižovala tak přesnost měření. Při měření impedance na vyšších frekvencích byl díky nutné rezervě počet vzorku nepřijatelně malý. Docházíme tedy ke zjištění, že během prvních cyklů časovače nelze spoléhat na teoreticky správný výpočet požadovaných hodnot a přistoupíme tedy k řešení robustnějšímu.

Spolehlivě funkční návrh počítá s duálním módem ADC s využitím DMA bez přerušení – průměrné hodnoty procesor vypočítá v přerušení časovače z již hotových vzorků, namísto z předem určeného počtu. Kompletní konfigurace obou převodníků zobrazují následující obrázky 11 a 12. Výrazný rozdíl mezi ADC tvoří nastavení DMA, kdy sice oba mají přístup, ale pouze ADC1 jako master vzorky odesílá. To se děje díky společné sběrnici mezi oběma ADC, a tak jsou 16bitové hodnoty z obou převodníků sloučeny a ukládány jako 32bitové slovo. V přerušení, tedy při zpracování procesorem, se toto slovo zpětně rozkládá na dvě hodnoty.



Obrázek 11 – Konfigurace ADC1 s využitím DMA



Obrázek 12 – Nastavení ADC2, které pracuje jako slave ADC1

4.3 Výpočet synchronní detekce

Hlavní výstup programu tvoří naměřené absolutní hodnoty impedance a fázového posuvu na připojeném článku pro aktuálně nastavenou frekvenci. K těmto výsledkům se program dopočítá skrze synchronní detekci, kterou blíže popisuji v kapitole 2. Jak již práce zmiňuje, v průběhu měření se ukládají zprůměrované vzorky z obou ADC, a to pro napětí na článku a elektrický proud. K tomu dochází uvnitř přerušení časovače TIM5 (viz kapitola 4.1.3). Právě tato data slouží k výpočtu synchronní detekce. Ve chvíli, kdy program pro aktuální nastavení frekvence dokončí celé měření, se přistupuje k zastavení obou časovačů a ke spuštění kódu ve funkci synchronní detekce. V následujících odstavcích popisuji průběh této významné části programu.

Funkci pro výpočet synchronní detekci lze rozdělit na dvě části – nejprve probíhá akumulace naměřených hodnot, z kterých následně vypočítáme konkrétní hodnoty fyzikálních veličin. První část funkce znázorňuje Ukázka kódu 3. Ve smyčce postupně akumulujeme průměrné naměřené hodnoty z ADC, které se vždy přenásobí příslušnou hodnotou z referenční tabulky.

```
1  void Synchronous_Detection()
2  {
3     int i_;
4     int i_Qshift;
5     for (int i = TABLE_LEN; i < TABLE_LEN*TOTAL_REPETITIONS; i++)
6     {
7           i_ = (i*table_step) % TABLE_LEN;
8           int shift = 0;
9           if (i - table_quarter_len < 0) { shift = TABLE_LEN - 1; }
10          i_Qshift = (i - table_quarter_len/table_step +
   shift)*table_step % TABLE_LEN;
11
12          Uim += avg[0][i]*table[i_];
13          Ure += avg[0][i]*table[i_Qshift];
14          Iim += avg[1][i]*table[i_];
15          Ire += avg[1][i]*table[i_Qshift];
16    }
```

Ukázka kódu 3 – První část funkce pro výpočet synchronní detekce, akumulace

V kódu vystupuje několik konstant definovaných na začátku programu. *TABLE_LEN* deklaruje délku referenční tabulky *table* a je stanovena celkem na 128 hodnot. *TOTAL_REPETITIONS* udává počet opakování, tedy 10 vln. Od konstant se odvíjí počátek a konec smyčky – začínáme od 2. vlny, protože vzorky z 1. vlny mohou být defektní. Cyklus následně projde celé pole průměrných hodnot *avg*. Proměnné *i_* a *i_Qshift* slouží k indexaci v referenční tabulce. Pro výpočet imaginární složky napětí (proudu) průměrnou hodnotu napětí (proudu) přenásobíme hodnotou tabulky na indexu *i_* – např. průměr vzorků z 2. vlny na indexu 32 (i = 160) přenásobíme hodnotou referenční tabulky na indexu 32 (*i_* = 32). Pro výpočet reálné složky obou veličin je nutné spočítat sumu všech průměrných hodnot, které přenásobíme hodnotami z referenční tabulky. Zde ale vystupuje index *i_Qshift*, tedy posunutí o čtvrtinu délky tabulky zpět. Na řádcích 8 a 9 tohoto kódu si dále všimneme pomocné proměnné *shift* sloužící k překlopení indexu zpět z hodnot menší než nula zpět do správného intervalu. Výstup smyčky tvoří 4 akumulační proměnné, a to pro jednotlivé složky proudu a napětí. S nimi program dále pracuje v druhé části funkce pro výpočet synchronní detekce.

Zbytek funkce představuje Ukázka kódu 4. Nejprve vypočítáme normalizované hodnoty napětí a proudu z dílčích složek. Druhým krokem je kalibrace, a to za pomocí kalibračních konstant *Ugain* a *Igain*, které byly laboratorně vypočteny. Tento proces popisuje kapitola 6.2. Funkce dále počítá absolutní hodnotu impedance jako podíl napětí a proudu na článku. K výpočtu fáze potřebujeme zjistit hodnotu arkus tangens pro napětí a proud. Konkrétní hodnotu

fázového posuvu pak získáme jako rozdíl těchto dvou úhlů a výsledek převedeme z radiánů na stupně. U fáze dochází v případě záporného výsledku k překlopení hodnoty přičtením 360 °.

```
17    U = sqrt((Ure*Ure) + (Uim*Uim));
18    I = sqrt((Ire*Ire) + (Iim*Iim));
19
20    U = U * Ugain;
21    I = I * Igain;
22
23    fiU = atan2(Uim, Ure);
24    fiI = atan2(Iim, Ire);
25
26    R = U/I;
27    Rphase = (fiU - fiI)*180/M_PI;
28
29    if (Rphase < 0) {Rphase += 360.0;}
30 }
```

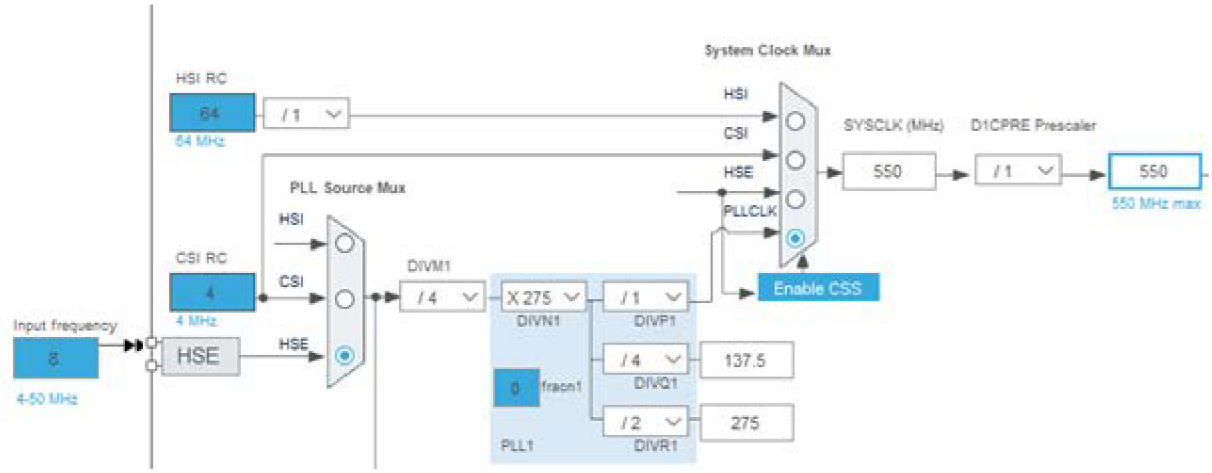
Ukázka kódu 4 – Druhá část funkce synchronní detekce, výpočet impedance a fáze

Výsledná absolutní hodnota impedance a fázový posuv se ukládají do paměti programu, stejně jako dílčí výsledky napětí a proudu. Po vykonání funkce program přenastaví příslušný příznak, který slouží ke kontrole odesílání výsledků, o čemž zpravuje kapitola 4.7. Program je nastaven tak, aby ihned výsledky odeslal skrze komunikační periferie, ale lze se na ně dotázat i později, pokud tedy mezitím nedošlo ke spuštění nového měření.

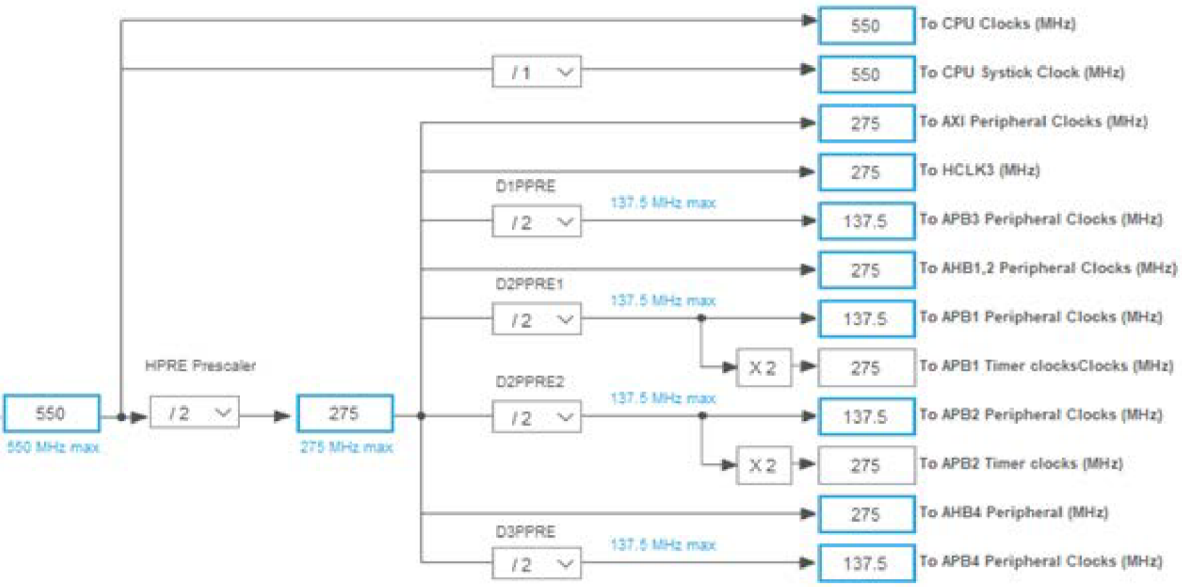
4.4 Nastavení hodinových kmitočtů jádra a periferií

Periferie, které systém používá, pracují na operační frekvenci udávané blokem RCC a jeho registry. Pro konfiguraci HW bylo nutné řadu registrů pozměnit pro správnou funkci programu. RCC spravuje generování hodinového cyklu a jeho resetu pro celý mikrokontroler. Nastavení proběhlo v programovacím prostředí STM32CubeIDE. Na Obrázku 13 právě z tohoto prostředí vidíme většinu potřebných hodnot frekvence a příslušných registrů. FW využívá časovačů a AD převodníků, jejichž vstupní frekvenci určují fázové závěsy PLL1 a PLL2, o kterých se dozvíme více v následujících odstavcích. Dále bylo zapotřebí správně určit vstupní signál pro SPI a USB. Při konfiguraci máme možnost v multiplexoru zvolit příslušný zdroj hodinového signálu. Mezi nimi máme na výběr mezi 4 interními oscilátory a 2 externími. Externí se dělí na vysokorychlostní (HSE) či nízkorychlostní (LSE) variantu.

Protože je žádoucí, aby procesor a zvolené časovače i převodníky běžely na co nejvyšší povolené frekvenci, zvolen byl právě HSE pro většinu z HW periferií. Například pro komunikaci běží USB na zdroji interního oscilátoru HSI48 – tedy na 48 MHz. [10]



Obrázek 13 – Nastavení kmitočtu hlavní větve hodinového signálu pro jádro

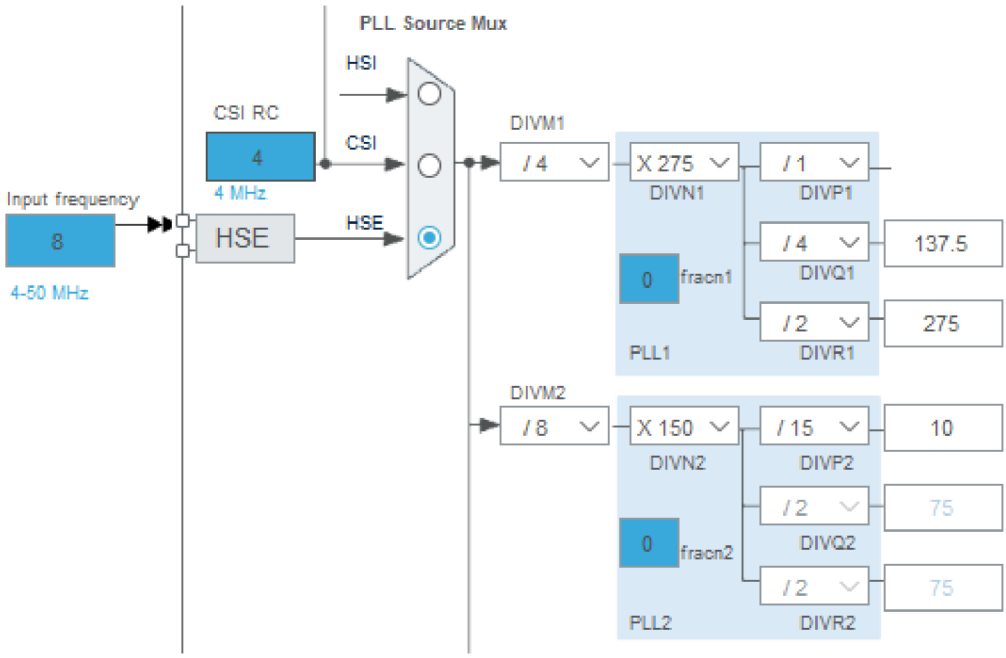


Obrázek 14 – Pokračování hlavní větve hodinového signálu pro jádro a využívané časovače na APB1

Jak již bylo popsáno v předchozích kapitolách, důležitou součást systému tvoří časovače TIM2 a TIM5. K těm je přiveden hodinový signál, který vychází z HSE, jehož vstupní frekvence je ve výchozím nastavení 8 MHz. Odtud se dostává k hlavnímu fázovému závěsu PLL1, kde se kmitočet hodinového cyklu před vstupem do jednotlivých PLL dále dělí. Uvnitř PLL1 se signál opět násobí, abychom dosáhli co nejvyšší hodnoty frekvence pro klíčové periferie. Výstupní signál již běží na 550 MHz a nemění se před vstupem do hodin procesoru.

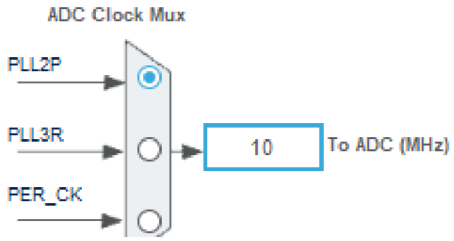
Pro vstup do periferií prochází předděličkou registru HPRE, aby signál nepřesáhl maximální možnou hodnotu frekvence podporovanou sběrnicí APB pro obsluhu periferií. Hodinový signál se dále větví podle sběrnic jednotlivých periferií. Časovače TIM2 i TIM5 nalezneme na sběrnici APB1. Výslednému vstupnímu kmitočtu hodinového signálu obou časovačů odpovídá hodnota 275 MHz, přesně podle Obrázku 14. Signál se před tím sice dělí dalším registrem pro potřeby ostatních APB1 periferií, ale pro časovače je opět přenásoben zpět na maximální hodnotu. Vnitřní signál časovačů pak nastavuje sám program za chodu, a to konfigurací příslušných registrů, jak již práce popisovala.

Pro potřeby ADC však není vhodné, aby vstup hodinového signálu přesáhl teoreticky maximální frekvenci 10 MHz pro souběžný chod dvou ADC za standardních podmínek. [7] Zdroj signálu pro ADC tedy tvoří fázový závěs PLL2. Jeho konfiguraci vidíme na Obrázku 15 hned pod nastavením PLL1.



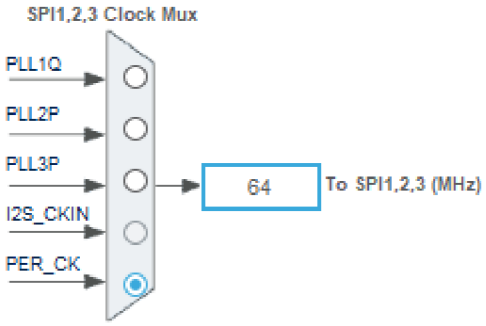
Obrázek 15 – Nastavení hodinových kmitočtů fázových závěsů PLL1 a PLL2

Vstupní frekvence z HSE je opět 8 MHz. Jednotlivé registry děliček a násobiček uvnitř PLL2 mají omezení na rychlost signálu, a proto musíme každý z registrů nastavit na požadovanou hodnotu. Protože výstupní signál PLL2 (vstupní signál pro ADC) chceme nastavit na hodnotu 10 MHz tak jak je na Obrázku 16, registr DIVM2 je nastaven na hodnotu 8, DIVN2 signál vynásobí 150krát a DIVP2 jej dále podělí 10.

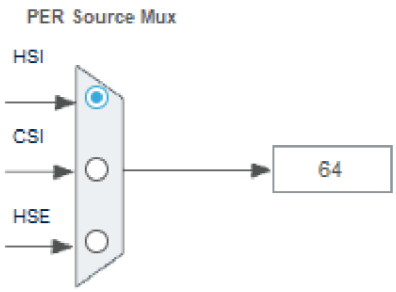


Obrázek 16 – Zdroj hodinového signálu na frekvenci 10 MHz určuje výstupní frekvence PLL2

Pro potřeby digitálního snímače, který měří teplotu, systém využívá SPI. I tato sběrnice vyžaduje zdroj hodinového signálu a ten spatříme na Obrázku 17. Nastaven je na 64 MHz, ale uvnitř SPI3 se signál dále dělí předděličkou s nastavenou hodnotou 8, abychom zajistili modulační rychlost 8 Mbit/s pro správnou funkci snímače. Hodnota vstupní frekvence závisí na zvoleném zdroji hodin PER_CK uvnitř multiplexoru. Jedná se opět pouze o výběr zdroje hodin dle Obrázku 18, tedy zda se použije HSE, HSI apod. Volbu HSI nabízí výchozí nastavení a není nutné ji měnit, protože si SPI3 signál sám zpomalí interní předděličkou. Druhou možností by samozřejmě bylo zvolit za zdroj HSE se vstupní frekvencí 8 MHz a dále nevyužívat interních registrů SPI3.



Obrázek 17 – Multiplexor pro určení vstupního hodinového signálu do SPI



Obrázek 18 – Možnost volby zdrojového oscilátoru pro vybrané periferie

4.5 Měření teploty NTC termistorem

Deska obsahuje 10kΩ NTC termistor, tedy termistor s negativním teplotním koeficientem. Umístěný je přímo pod článkem, kterého se dotýká. Měřená impedance článku je silně závislá na jeho teplotě. Z tohoto důvodu je vhodné před spuštěním měření zkontrolovat teplotu článku, případně i v jeho průběhu. Termistor při změně teploty mění svůj odpor, a protože byl použit NTC termistor, jeho odpor při zahřátí součástky klesá.

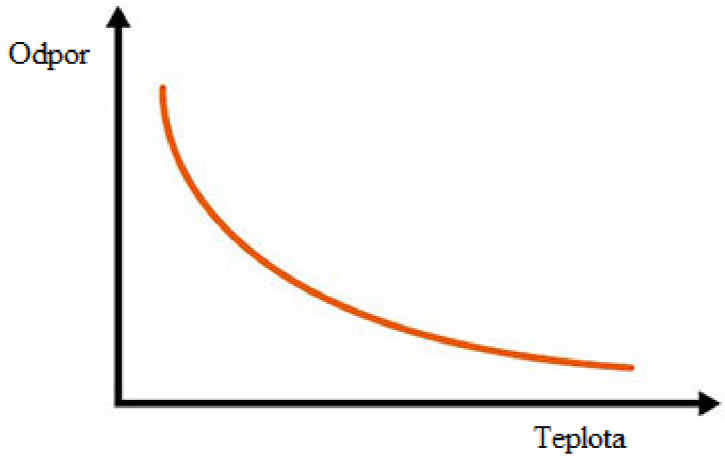
K interpretaci naměřených hodnot teploty na desce aktivujeme další z ADC, tedy ADC3. Jeho nastavení spatříme na Obrázku 19. Využívá pouze jeden kanál, a to vstupní kanál 6, jehož vývod nalezneme na PF10, ke kterému je přiveden vývod od termistoru. Jak popisuji v kapitole 4.4, všechny převodníky se řídí nastavenou vstupní frekvencí hodin 10 MHz. Nastavení předděličky ADC3 není potřeba měnit, je tak nastavena na hodnotu 1 a hodiny jsou asynchronní. Volíme nejvyšší dostupné rozlišení převodníku ADC3, tedy 12 bitů.

ADC_Settings	
Clock Prescaler	Asynchronous clock mode divided by 1
Resolution	ADC 12-bit resolution
Scan Conversion Mode	Disabled
Data Alignment	Right alignment
Continuous Conversion Mode	Disabled
Discontinuous Conversion Mode	Disabled
DMA Continuous Requests	Disabled
End Of Conversion Selection	End of single conversion
Overrun behaviour	Overrun data preserved
Left Bit Shift	No bit shift
Conversion Data Management Mode	Regular Conversion data stored in DR register only
Low Power Auto Wait	Disabled
ADC_Regular_ConversionMode	
Enable Regular Conversions	Enable
Enable Regular Oversampling	Enable
Oversampling Right Shift	No bit shift for oversampling
Oversampling Ratio	16
Regular Oversampling Mode	Oversampling Continued Mode
Triggered Regular Oversampling	Single trigger for all oversampled conversions
Number Of Conversion	1
External Trigger Conversion Source	Regular Conversion launched by software
External Trigger Conversion Edge	None
Sampling Mode	Normal
Rank	1
Channel	Channel 6
Sampling Time	2.5 Cycles
Offset Number	No offset
Offset Sign	Offset Sign Negative

Obrázek 19 – Kompletní nastavení ADC3 pro potřeby měření teploty termistorem

Převodník má konfigurovaný oversampling na šestnáctinásobek pro zpřesnění naměřené hodnoty – získáme pouze jednu hodnotu, která ale vznikla zprůměrováním 16 vzorků díky oversampleru. Tímto procesem dosáhneme vyššího SNR, [11] výsledkem je i nižší spotřeba elektrické energie oproti SW implementaci kvůli menší zátěži procesoru. Zbylé nastavení je prakticky standardní pokud chceme periferii spustit jednou za kvantum času a naměřit jednu konkrétní hodnotu. Globální přerušení chodu programu převodníkem program nevyužívá.

Měření teploty v programu obsluhuje vcelku jednoduchá funkce, která tento převodník spustí. Procesor počká na provedení konverze hodnoty, na což má k dispozici 100 ms. Pokud vše proběhne v pořádku, hodnota se ukládá a ADC3 se zastaví. Z důvodu využití oversamplingu a nastaveného rozlišení ADC se naměřená hodnota dělí hodnotou 4095 (což je rovno 2^16-1, tedy maximální 16bitové celočíselné hodnotě), dále pak dle [11] hodnotou 32. Přenásobením referenční hodnotou napětí, tedy 3,3 V, získáme přepočtené napětí. Nyní potřebujeme přepočítat naměřené napětí na odpor termistoru a z něho následně vypočítat jeho teplotu. Průběh změny odporu NTC termistoru v závislosti na teplotě vidíme na Obrázku 20.



Obrázek 20 – Průběh změny odporu termistoru NTC [15]

Pro správný přepočet odporu na teplotu vhodně zvolíme rovnici, která se může odvíjet od rovnice proložení křivkou Steinhart-Hart. Pro naše potřeby použijeme následující rovnici β:

$$\frac{1}{T} = \frac{1}{T_O} + \frac{1}{\beta} \times \ln\left(\frac{R}{R_O}\right) \quad (5)$$

Parametr R_O určuje elektrický odpor při referenční teplotě T_O. Hodnotu parametru β nalezneme ve specifikaci termistoru a dle [12] hodnota konstanty činí 3 435 K ± 1 %. Právě díky uvedené rovnici lze snadno vypočítat naměřenou teplotu. Nejprve podle [15] převedeme navzorkované napětí na elektrický odpor vzorcem:

$$R_{thermistor} = R_{balance} \cdot \left(\frac{U_S}{U_{out}} - 1\right) \quad (6)$$

V tuto chvíli již máme hodnotu napětí v podobě digitálního čísla díky ADC, parametr U_S určuje referenční hodnotu napětí – 3,3 V. U_{out} představuje již přepočtené napětí. $R_{balance}$ je hodnota odporu děličky napětí směrem k uzemnění. Tímto výpočtem se můžeme přesunout k předešlé rovnici a již přepočítat hodnotu odporu na teplotu za pomoci konstanty β. Nyní získáme teplotu ve stupních Kelvina, proto k převodu do Celsiovy stupnice od vypočtené hodnoty odečteme hodnotu 273,15.

Funkce výsledek ukládá do paměti mikrokontroleru. Pokud program dostane příkaz pro změření teploty, celá funkce se provede znovu pro zjištění aktuální teploty a naměřenou hodnotu teploty ve stupních Celsia odesílá po USB. Protože je deska osazena i digitálním snímačem pro měření teploty, odesílány jsou dvě hodnoty za sebou – jedna naměřená termistorem, druhá digitálním snímačem. Díky tomu snadno ověříme správnost výsledku dvěma různými způsoby měření.

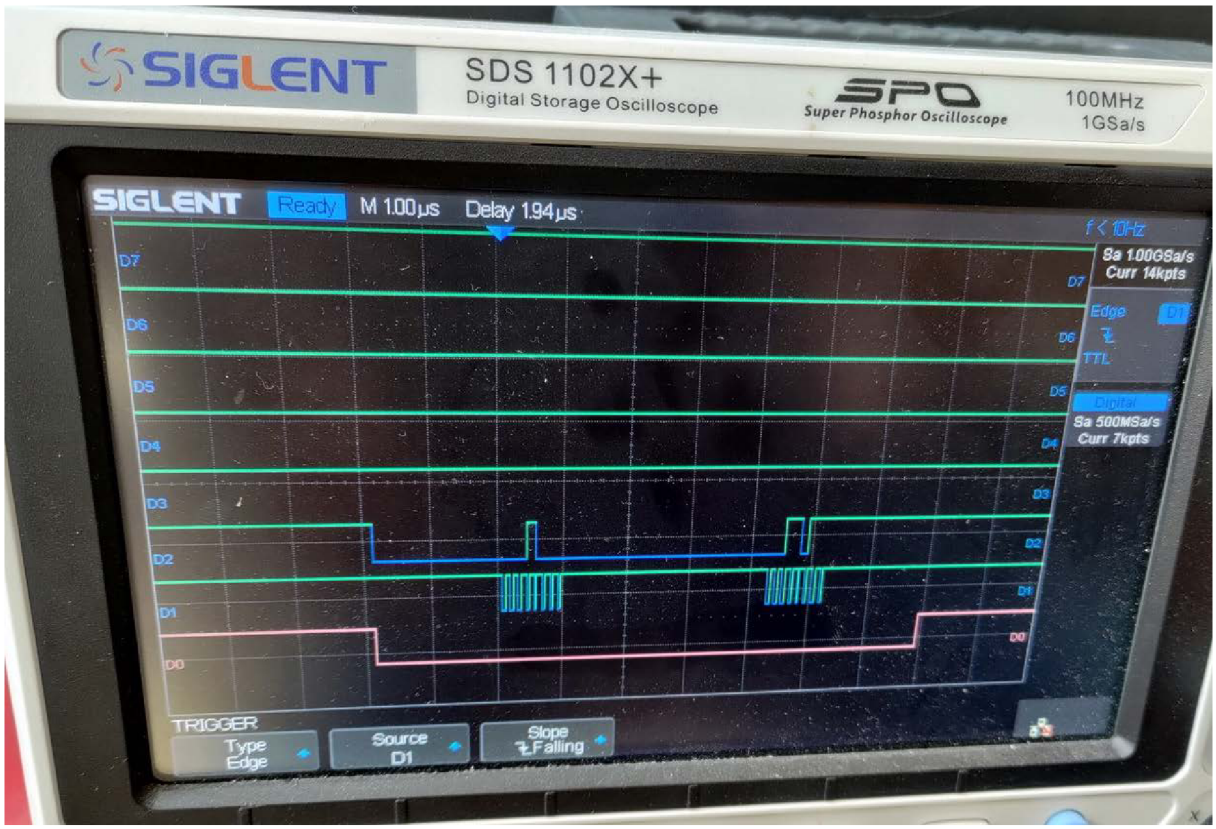
4.6 Měření teploty digitálním snímačem LM74

Druhým způsobem měření teploty článku je využití digitálního snímače. Z důvodu nízké spotřeby a jednoduché implementace periferie byl zvolen LM74, senzor teploty s AD převodníkem a sériovým periferním rozhraním SPI, které mikrokontroler podporuje lze jej dále konfigurovat. Procesor může v libovolný moment číst aktuální teplotu, ale snímač zároveň zaručuje velmi nízkou spotřebu ve chvíli, kdy není používán – spotřebuje méně nežli 10 μA. Dle katalogového listu produktu rozlišení teploty činí 0,0625 °C. [13] Samotný senzor na desce nalezneme přímo pod článkem a co nejblíže k němu, aby monitoroval jeho teplotu.

Pro implementaci snímání teploty bylo nutné v projektu správně nastavit SPI, konkrétně SPI3, což vidíme na Obrázku 21. Důležitý prvek tvoří nastavení frekvence vstupního hodinového signálu pro sběrnici, který blíže popisuje kapitola 4.4, a jejíž hodnota činí 64 MHz. Při nastavení předděličky signálu na hodnotu 8 získáme modulační rychlost 8 Mbit/s, data totiž budou mít délku 8 bitů. Nastavení GPIO zůstává ve výchozí konfiguraci, tedy vývody PC10 pro externí komunikační hodiny a pin PC11 pro komunikaci v režimu MISO pro příjem dat v režimu master. V celkové konfiguraci GPIO ale přibyl pin PA6 pro signál chip select přenášený přes SPI3. Jeho označení potřebujeme znát v programu, abychom jej využili, proto jej označíme jako *SPI3_CS*. [10]

Obrázek 21 – Konfigurace SPI3 pro potřebu měření teploty digitálním snímačem LM 74

Průběh měření teploty skrze tento snímač vidíme na Obrázku 22 měření osciloskopem Siglent SDS 1102X+. Na kanálu D1 logického analyzátoru osciloskopu spatříme hodinové pulsy externích komunikačních hodin generovaných rozhraním SPI mikrokontroleru, na D2 získáme datové slovo odpovídající naměřené teplotě – jednotlivé bity měření. Kanál D0 zobrazuje signál Chip select (CS) pro obvod LM74, kterým se aktivuje jeho komunikační rozhraní.



Obrázek 22 – Průběh signálů digitálního snímače při snímání teploty na osciloskopu

Funkce v programu, která obsluhuje činnost senzoru, neobsahuje složitější výpočty na rozdíl od způsobu měření teploty za pomoci termistoru. Probíhá pouhé nastavení GPIO pinu pro chip select na hodnotu *RESET*, čímž se spouští měření. Následně skrze SPI3 získáme 16 bitů dat – nejprve vyšších 8, následně spodních 8. Tímto je měření u konce a můžeme příslušný pin chip select nastavit zpět na hodnotu *SET*. Bity spojíme dohromady do jedné 16bitové proměnné. Rozlišení při konverzi dat do digitálních hodnot činí pouze 13 bitů. [13] Z toho důvodu provedeme bitový posun vpravo, a to o 3 bity. Výslednou hodnotu stačí již pouze přenásobit rozlišením teploty pro snímač LM74, tedy hodnotou 0,0625. Získáme tak teplotu snímače ve stupních Celsia. Stejně jako v případě měření termistorem, tento výsledek se ukládá do paměti mikrokontroleru, aby se společně s hodnotou prvního měření mohl odeslat uživateli skrze virtuální sériový port po USB. Zde vidíme stručnou Ukázku kódu 5 z funkce:

```
1  double Meas_CellTemperatureSPI()
2  {
3     uint8_t RxData1;
4     uint8_t RxData2;
5     uint16_t regbits;
6
7     HAL_GPIO_WritePin(SPI3_CS_GPIO_Port, SPI3_CS_Pin,
   GPIO_PIN_RESET);
8     HAL_SPI_Receive(&hspi3, (uint8_t *)&RxData1, 1, 100);
9     HAL_SPI_Receive(&hspi3, (uint8_t *)&RxData2, 1, 100);
10    HAL_GPIO_WritePin(SPI3_CS_GPIO_Port, SPI3_CS_Pin,
   GPIO_PIN_SET);
11
12    regbits = (RxData1 << 8) + RxData2;
13    regbits = (regbits >> 3);
14    return(regbits * 0.0625);
15 }
```

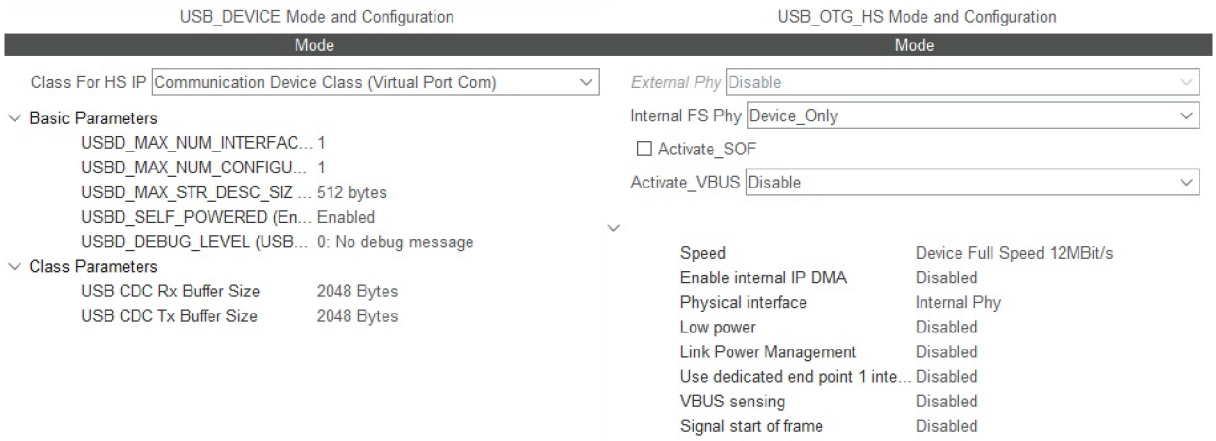
Ukázka kódu 5 – Měření teploty za pomoci digitálního snímače

4.7 Komunikace s nadřazeným systémem

Pro vizualizaci a ukládání naměřených dat z mikrokontroleru bylo nutné zprovoznit komunikaci oběma směry – přijímat příkazy a odesílat zprávy. K připojení k počítači bylo zvoleno vestavěné rozhraní USB. Při správné konfiguraci za pomoci příslušné knihovny spolehlivě zajišťuje přenos dat. Zvolený HW STM32H723ZG navíc podporuje vysokorychlostní komunikaci. V nastavení volíme Communication Device Class (CDC) pro komunikaci skrze virtuální sériový port. Díky tomu lze s mikrokontrolerem být ve spojení skrze libovolný terminál pro sériovou komunikaci. V rámci diplomové práce vznikla též klientská aplikace pro PC v prostředí Matlab, o které dále hovoří kapitola 5. Podrobněji o nastavení a o dostupných příkazech hovoří následující podkapitoly.

4.7.1 Nastavení USB

V prostředí STM32CubeIDE v souboru *.ioc* pro konfiguraci desky STM32H723ZG nalezneme *USB_OTG_HS*, tedy vysokorychlostní specifikaci USB, díky které mikrokontroler umí plnohodnotně komunikovat na obě strany. Jeho nastavení vyčteme z Obrázku 23. Tato definice ale samotná nestačí, nutností je dále zvolit middleware *USB_DEVICE* a jeho komunikační třídu. Volíme CDC, tedy komunikaci skrze virtuální sériový port. Třída obsahuje parametry pro délku jednotlivých bufferů pro příjem a odesílání dat, nastaveny jsou na maximální hodnotu, tedy 2 KB.



Obrázek 23 – Nastavení middleware USB a samotné sběrnice

Projekt generuje všechny potřebné soubory k práci s USB, neb o vše se stará middleware. V programu deklarujeme buffery pro obě strany komunikace, pro přijímání příkazů postačí pole délky 64 bytů, pro odesílání pak např. 255. V souboru middlewaru *usbd_cdc_if.c* musíme vytvořit globální pole pro příjem a ukládání dat tak jako v následujícím kódu:

```
1  extern uint8_t com_receive[64];
```

Ukázka kódu 6 – Deklarace bufferu pro komunikaci přes USB

4.7.2 Průběh komunikace a parsování dat

Průběh komunikace je vcelku jednoduchý proces. Nejprve middleware získá data a uloží je do přístupného pole, které program periodicky prochází a kontroluje, zda se v něm objeví nové hodnoty. Program totiž v hlavní smyčce neprovádí nic jiného, než kontrolu komunikace a kontrolu příznaků pro dokončené měření. V případě, že byla přijata nějaká data, přichází na řadu parser. Ten projde celé pole, sestaví z něho řetězec znaků a porovnává jej s předem definovanými příkazy, které deklaruje následující kapitola kapitola 4.7.3. Princip funkce parseru vyčteme z následující ukázky kódu uvnitř parseru.

```
1     if (strcasestr(msg, "RESU") != '\0')
2     {
3          if (measurement_completed) fsend_results();
4          else fsend_state();
5     }
6     if (strcasestr(msg, "WAVE") != '\0') { fsend_lastWave(); }
7     if (strcasestr(msg, "BUFF") != '\0') { fsend_adc_buffer(); }
8     if (strcasestr(msg, "STAT") != '\0') { fsend_state(); }
9     if (strcasestr(msg, "TEMP") != '\0') { fsend_temp(); }
```

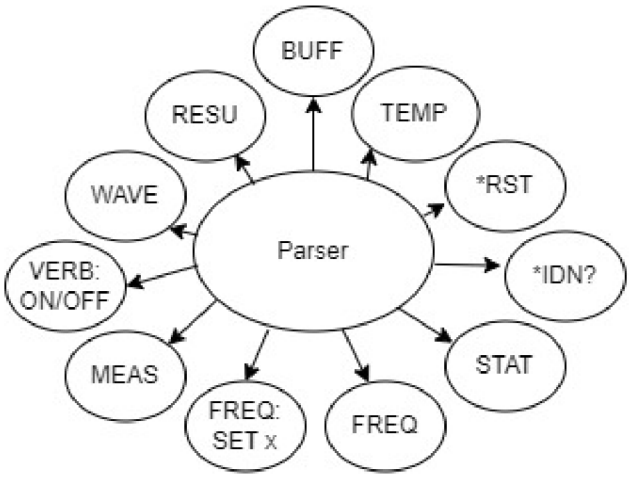
Ukázka kódu 7 – Část parseru pro kontrolu příchozích instrukcí

Vždy se porovná vstupní pole *msg* s definovaným setem instrukcí, nehledě na formu příchozí zprávy – důležité jsou první čtyři znaky, které mohou klidně být psané malými písmeny. Pokud dojde ke shodě, provádí se funkce v závislosti na nastaveném příznaku jednoduché, nebo rozvité komunikace.

Dle přečtených instrukcí program dále pracuje – spouští periferie, oznamuje svůj stav apod. FW obsahuje řadu funkcí, které komunikační rozhraní využívají. Mohou reagovat na příkazy, nebo na vnitřní podněty – např. dotaz na aktuální stav měření, odesílání aktuální teploty na NTC termistoru a digitálním snímači, viz kapitoly 4.5 a 4.6. Důležité je zajistit, aby komunikaci nevyvolávalo žádné z HW přerušení, což způsobuje chybové stavy. Pokud chceme odesílat data z mikrokontroleru, ale rozhraní USB je zaneprázdněno, middleware buffer neodešle a místo toho celou zprávu zahodí. To je ošetřeno v programu podmínkou, kde se stav USB kontroluje, a to při každém pokusu o odeslání zprávy. Při odesílání dat program využívá zpravidla stejného bufferu, který formátuje a ukládá do něj data. Podle příznaku komunikace funguje ve dvou režimech – odesílá speciální identifikační znak a hodnotu, či více oddělených hodnot, nebo zprávu ve formátu věty pro lepší čitelnost. První možnost pak využívá přidružená klientská aplikace pro minimalizaci toku přebytečných dat.

4.7.3 Příkazy SCPI

Sériovou komunikací lze mikrokontroleru odesílat řadu požadavků, které se často mohou shodovat s příkazy pro jiné programovatelné stroje. Z toho důvodu program využívá standardu SCPI, dle něho definuje seznam instrukcí. Instrukce by měly dodržet formát – první čtyři znaky velkými písmeny, další znaky malými písmeny. Parseru ale, jak již bylo zmíněno, na formě moc nezáleží, stačí mu rozeznat první čtyři znaky, které mohou být psány i malými písmeny. Celkový set instrukcí vidíme na Obrázku 24, zároveň následuje seznam s popisem každé z instrukcí:



Obrázek 24 – Celkový instrukční set pro komunikaci s MCU

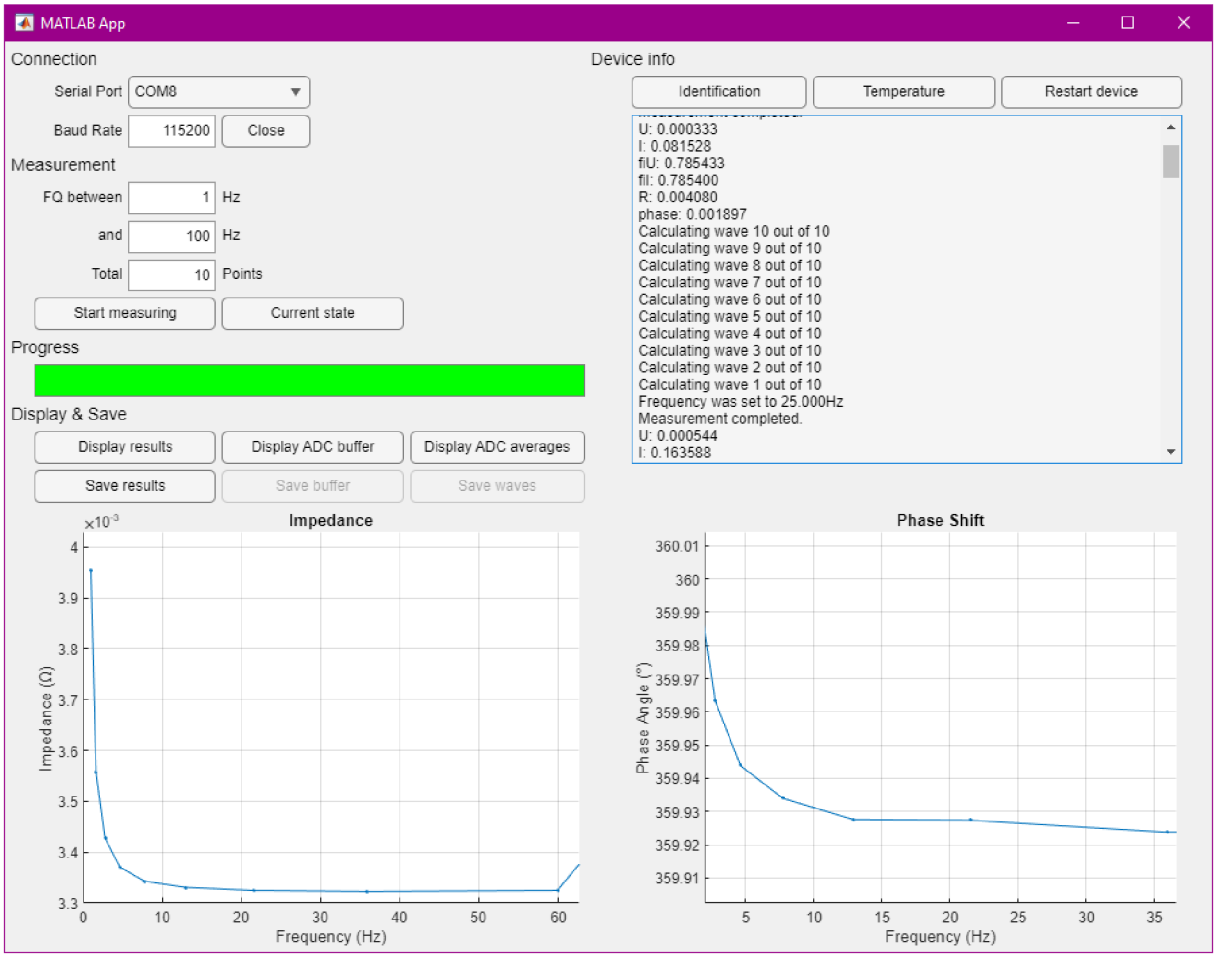
- *RST – slouží k restartu MCU.
- VERBose: [ON/OFF] – nastaví MCU do výřečného, nebo strohého módu. Dle toho se liší způsob výpisu.
- *IDN? – odesílá identifikační zprávu FW.
- FREQuency: SET x – nastaví časovač TIM5 na požadovanou frekvenci x. Může dojít na případ, že frekvence nelze nastavit z důvodu nedodržení rozsahu povolených frekvencí. V tu chvíli se stávající frekvence nemění a uživatel je o tom pouze informován. V případě, že frekvenci nelze konfigurovat na hodnotu x, ale ta spadá do přípustného rozsahu, nastaví se frekvence na nejbližší možnou hodnotu. I po úspěšném nastavení frekvence je uživatel zpraven o její aktuální hodnotě.
- FREQuency – slouží jako pouhý dotaz na aktuálně nastavenou frekvenci.
- MEASure – instrukce spouští měření na zvolené frekvenci. Nejprve nuluje všechny používané proměnné, následně spustí časovače a ADC. V průběhu měření uživateli oznamuje aktuální stav stejně jako příkaz STAT. Po skončení měření odesílá výsledky, jako kdyby dostal příkaz RESUlts.
- RESUlts – pokud bylo měření dokončeno, odešle jeho výsledky. Pokud ne, splní příkaz STAT.
- WAVE – dotaz na hodnoty z bufferů obou ADC průměrných hodnot. Odešlou se všechny hodnoty pro aktuální frekvenci za všech 10 opakování. Oba signály mají průběhy vln, odtud název instrukce.

- BUFF – odesílá pomocný buffer při 50% střídě.
- STAT – dotaz na aktuální stav MCU. Odpovědí může být, že měření dosud nebylo spuštěno, nebo bylo naopak dokončeno. V průběhu měření odesílá aktuální stav, tedy číslo aktuálního opakování a maximální počet opakování.
- TEMP – odesílá dvě hodnoty teploty, a to nejprve naměřené NTC termistorem, následně odešle teplotu naměřenou digitálním snímačem LM74.

Příkazy lze seskupit a odeslat jich více naráz, oddělují je dvojtečky. Např. v klientské aplikaci kód odesílá příkaz na nastavení frekvence časovače a spustí měření následovně: FREQ:SET %f:MEAS, parametr „%f“ je nahrazen hodnotou frekvence, která je reprezentována číslem s plovoucí řádovou čárkou. Druhým příkladem může být připojení aplikace k mikrokontroleru, kdy nejprve nastavuje formát komunikace na pouhý výpis hodnot, zeptá se na aktuálně nastavenou frekvenci a stav měření: VERB:OFF:FREQ:STAT. Samotné příkazy se tedy vyhodnocují v parseru programu, kde procesor postupně projde celý seznam instrukcí podle jejich priority.

5 Aplikace klienta na PC

Klientská aplikace je velmi jednoduchá, vznikla na platformě Matlab pro potřeby vizualizace výsledků souvislého měření. Aplikaci demonstruje Obrázek 25 – slouží k odesílání instrukcí do mikrokontroleru, zejména pro spuštění měření napětí a proudu článku, a to postupně na všech bodech logaritmické škály frekvencí, kterou určíme počáteční a koncovou hodnotou frekvence a celkovým počtem bodů. Povolený interval frekvencí, ve kterém lze provést měření, je shodný s nastavením firmwaru, a to 1 mHz až 1,2 kHz.



Obrázek 25 – Klientská aplikace na PC pro správu HW a vizualizaci dat

Aplikace uživateli průběžně oznamuje předpokládaný čas celého výpočtu a vykresluje naměřené hodnoty. S každou naměřenou vlnou dílčích frekvencí se také aktualizuje ukazatel průběhu. Po dokončení měření na všech bodech lze naměřená data ukládat, viz kapitola 5.2. Vedle zobrazení výsledků – naměřené impedance a fázového posuvu – můžeme zobrazit i průběh vln jednotlivých frekvencí, nebo konkrétní naměřené hodnoty obou ADC. I tyto hodnoty lze skrze odpovídající tlačítka ukládat. Pokud měření probíhá na velmi nízkých

frekvencích a výpočet tak trvá delší dobu, je možné opět zjistit aktuální stav měření příslušným tlačítkem. K výpisu hodnot slouží konzole, k vizualizaci dat pak dva grafy – jeden pro každé ADC.

5.1 Popis práce s aplikací

Běžná práce s aplikací funguje následovně. Pomocí USB kabelu a virtuálního sériového portu připojíme klientskou aplikaci k mikrokontroleru při zvolené modulační rychlosti. Pokud připojení proběhne úspěšně, odemknou se další ovládací prvky aplikace a odesílá se instrukční set VERB:OFF:FREQ:STAT pro vypnutí verbálního módu komunikace, zjištění aktuálně nastavené frekvence a stavu mikrokontroleru. Pomocí tlačítek můžeme vyslat požadavky na identifikaci mikrokontroleru, HW reset, vypsání aktuální naměřené teploty termistorem a digitálním snímačem (viz 4.5 a 4.6), nebo instrukci pro zjištění aktuálního stavu desky – ta může obsahovat předešlé měření a jeho výsledky, které lze vizualizovat.

Pro spuštění nového měření nastavíme příslušná pole počáteční frekvence a frekvence koncové. Aplikace při kliknutí na tlačítko pro spuštění měření vygeneruje pole frekvencí s požadovanou délkou a odesílá první požadavek pro nastavení frekvence na první hodnotu a spuštění měření. S každým příchozím výsledkem se znovu nastaví frekvence na další hodnotu z pole frekvencí a měření se opět spustí. Mezitím jsou výsledky vykreslovány do grafů a konzole se plní stavovými informacemi. Jak vidíme na Obrázku 25, po dokončení všech měření již celý ukazatel průběhu zezelená a data můžeme ukládat, případně zobrazit hodnoty vln poslední naměřené frekvence (viz Obrázek 29 a Obrázek 28), nebo bufferu vzorků z ADC (viz Obrázek 27 a Obrázek 26).

Protože je mikrokontroler nastaven tak, aby odesílal stavové informace a naměřené hodnoty ihned (viz kapitola 4.7), také klientská aplikace obsahuje callback funkci pro manipulaci s příchozími daty. Funkce se spouští pokaždé, když po virtuálním sériovém portu přijme novou zprávu. Následně dochází k parsování této zprávy. Mikrokontroler je přes automaticky odesílaný příkaz VERB:OFF v módu čisté komunikace, takže vždy odesílá nejprve znak pro označení typu zprávy a následně hodnoty, které jsou mezi sebou řádně oddělené. Aplikace obsahuje stavový automat pro čtení přijatých zpráv a pro každý identifikační znak má definovanou vlastní funkci, kde se dále data zpracují. Zprávy se interpretují v konzoli aplikace, a pokud obsahují data k vizualizaci, ihned se dokreslují do obou grafů.

Komunikaci směrem z klientské aplikace vždy řídí příslušná tlačítka, pokud na ně uživatel klikne. Jak vidíme na Ukázce kódu 8 v jazyce Matlab, při vyvolání události kliknutí na

tlačítko kontroly stavu se program pokusí odeslat příkaz skrze virtuální sériový port. Pokud by došlo k jakékoliv chybě v komunikaci – přístroj byl odpojen apod., je uživateli předána chybová zpráva a sériová komunikace ukončena. Obsluha aplikace se pak pro další práci musí opět pokusit o navázání komunikace.

```
1    function StateButtonPushed(app, event)
2         try
3              write(serialportHandle,"STAT","uint8");
4         catch
5              app.OutputTextArea.Value = ["Serial port connection
  error!"; app.OutputTextArea.Value];
6              resetConnection(app);
7         end
8    end
```

Ukázka kódu 8 – Funkce pro obsluhu stlačení jednoho z tlačítek v klientské aplikaci

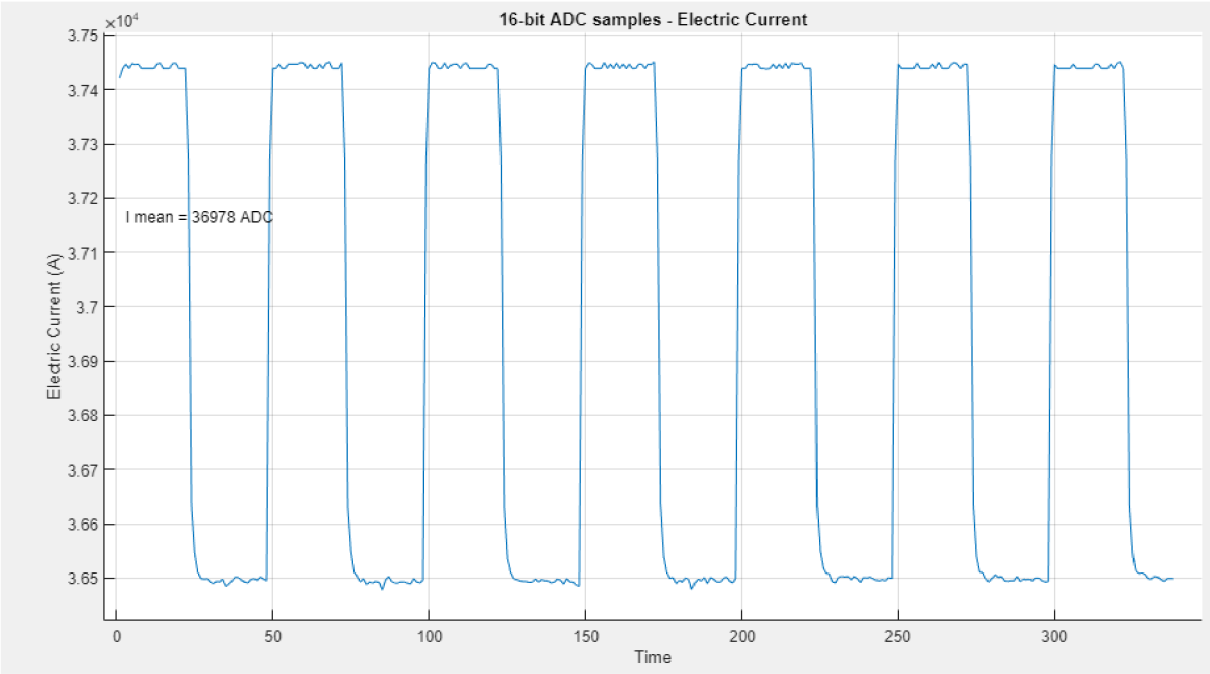
5.2 Vizualizace a ukládání dat

V sekci aplikace pod nastavením frekvence a spouštěním měření nalezneme oddíl pro vizualizaci a ukládání naměřených dat mikrokontrolerem. Zde se nachází celkem 6 tlačítek uskupených po sloupcích – vždy se jedná o tlačítko pro vykreslení naměřených dat do grafů a tlačítko pro uložení právě těchto hodnot. Výsledky jsou ukládány do souboru s příponou *.mat* pro pozdější využití v prostředí Matlab.

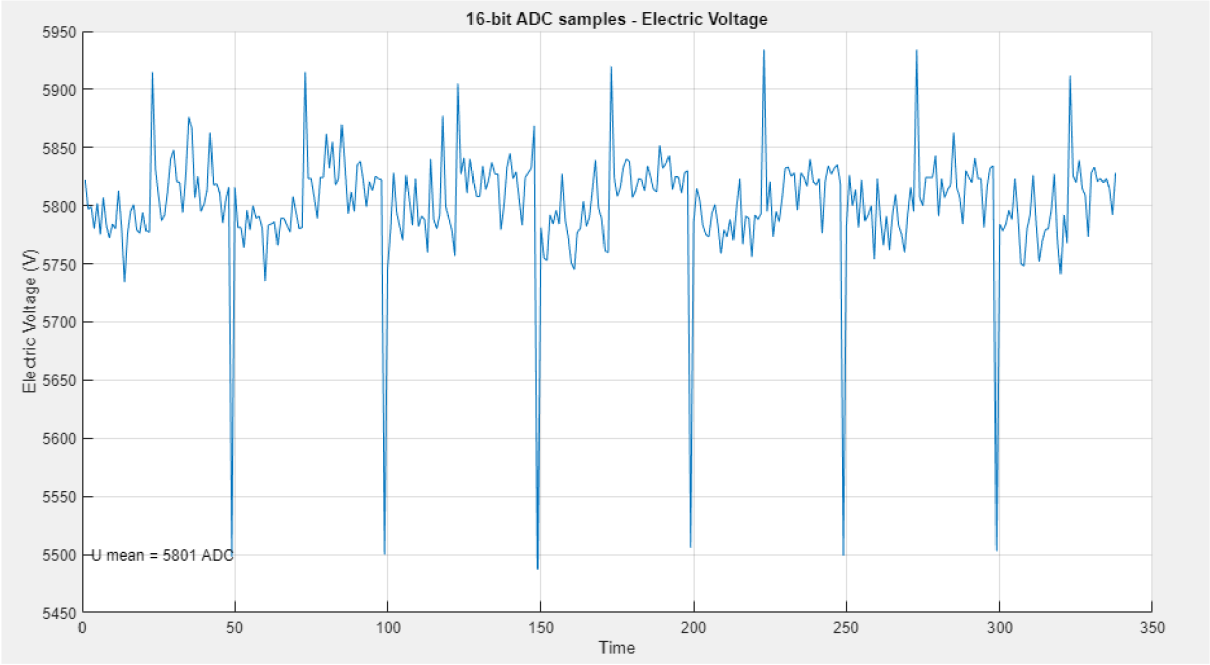
První sloupec obsahuje příkaz vizualizace naměřené impedance a fázového posuvu. Právě tato data se v grafu překreslují již za běhu měření. Pokud se ale klientská aplikace právě připojila k mikrokontroleru, který drží již naměřené hodnoty, lze je také získat a vykreslit. Tlačítka pro vizualizaci fungují jako malý stavový automat, protože skrze něj se mění typ zobrazovaných dat v grafech a jejich popisky. Dále lze uložit naměřené hodnoty impedance a fázového posuvu do dvou polí společně s polem frekvencí, na kterých tyto data mikrokontroler získal.

Druhý sloupec – vizualizace a ukládání hodnot z bufferu ADC – kliknutím na vrchní tlačítko aplikace získá pole hodnot pro naměřené elektrické napětí a elektrický proud. Jedná se o všechny naměřené vzorky obou ADC z poslední vlny (tedy 10. opakování měření) na poslední aktuálně vypočtené frekvenci, a to při 50% střídě signálu. Při vizualizaci na Obrázku 26 a Obrázku 27 vidíme čistý průběh proudu v závislosti na čase při spínání tranzistoru, kdežto v grafu napětí spatříme překmity při spínání, viz kapitola 4.2. Graf navíc

obsahuje střední hodnotu obou naměřených veličin, ale z důvodu snadného výpočtu střední hodnotu aplikace neukládá. Do souboru se vloží pouze obě pole bufferu napětí a proudu.



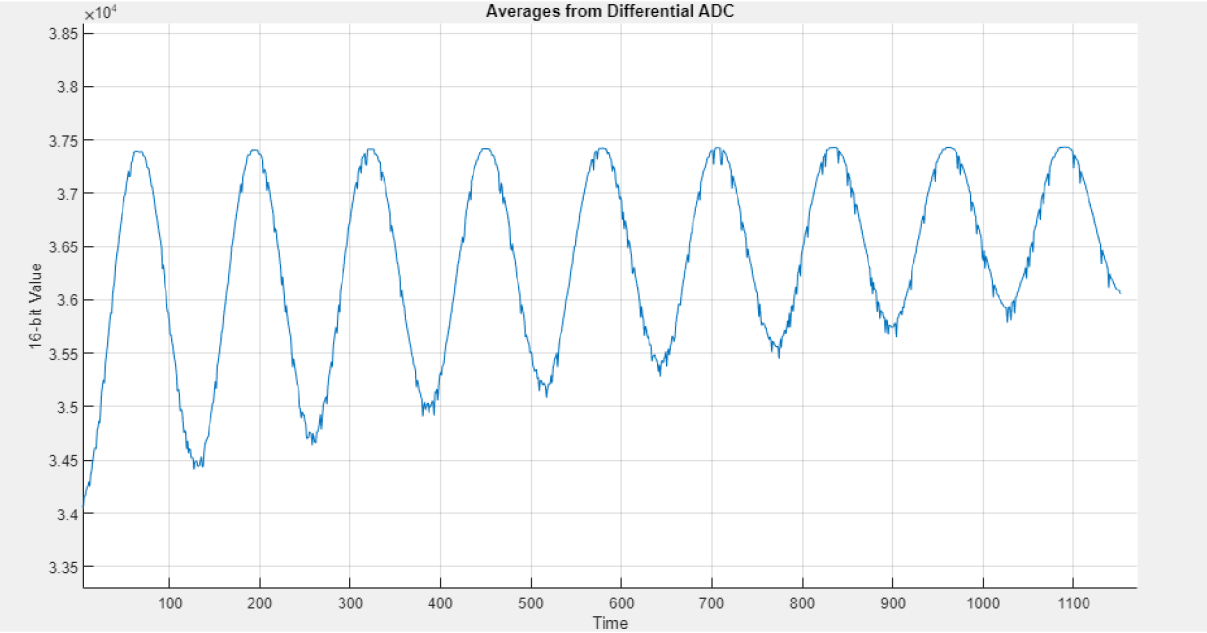
Obrázek 26 – Hodnoty elektrického proudu z bufferu ADC2 při 50% střídě na frekvenci 1 Hz



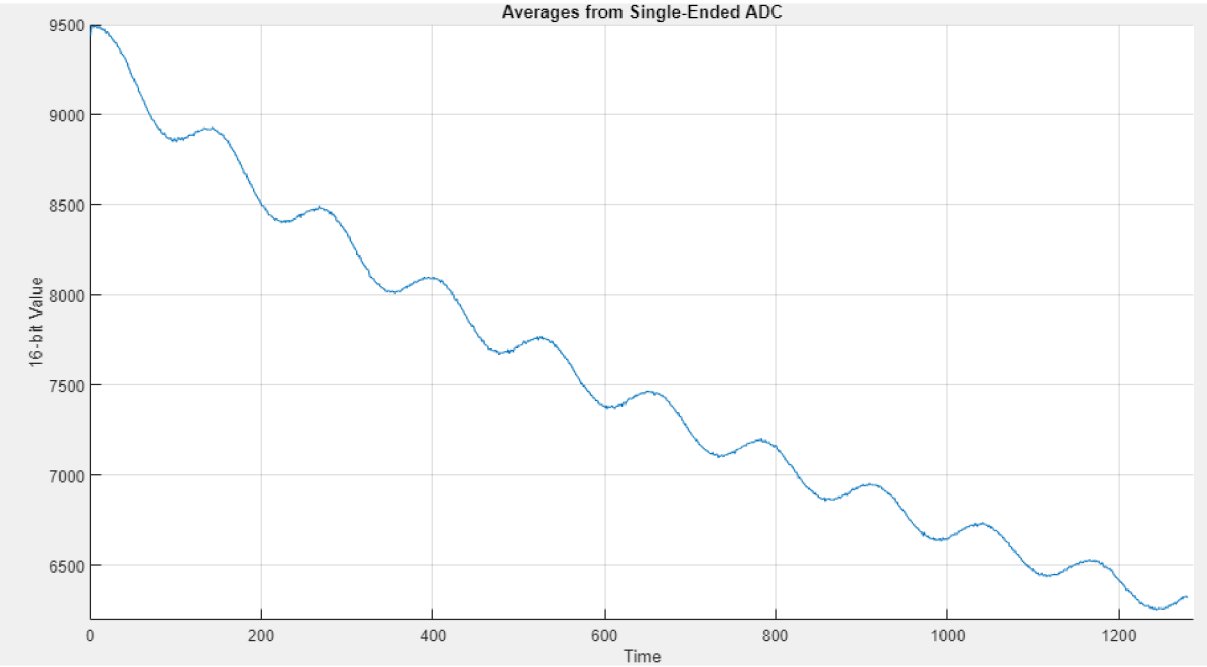
Obrázek 27 – Hodnoty elektrického napětí z bufferu ADC1 při 50% střídě na frekvenci 1 Hz

Tlačítko pro zobrazení průměrovaných vzorků z ADC ve třetím sloupci získá z mikrokontroleru všechny postupně vypočítané průměrné hodnoty měření napětí a proudu při stávající frekvenci – v grafech na Obrázku 28 a Obrázku 29 je lze interpretovat jako vlny,

které odpovídají kmitání PWM. V průběhu proudu vidíme stálý vlnitý průběh, který se v čase víceméně periodicky opakuje a nedochází k úbytku hodnot amplitudy. U napětí v průběhu času dochází k jeho úbytku a amplituda vln postupně klesá s přibývajícím časem. Tyto hodnoty jsou ukládány do dvou polí – elektrický proud i napětí (hodnoty ze single-ended a diferenciálního ADC), a k tomu je do souboru zaznamenán údaj o konkrétní frekvenci, na které měření proběhlo.



Obrázek 28 – Průměrné hodnoty elektrického proudu z ADC2 od 2. po 10. opakování měření na frekvenci 1 Hz

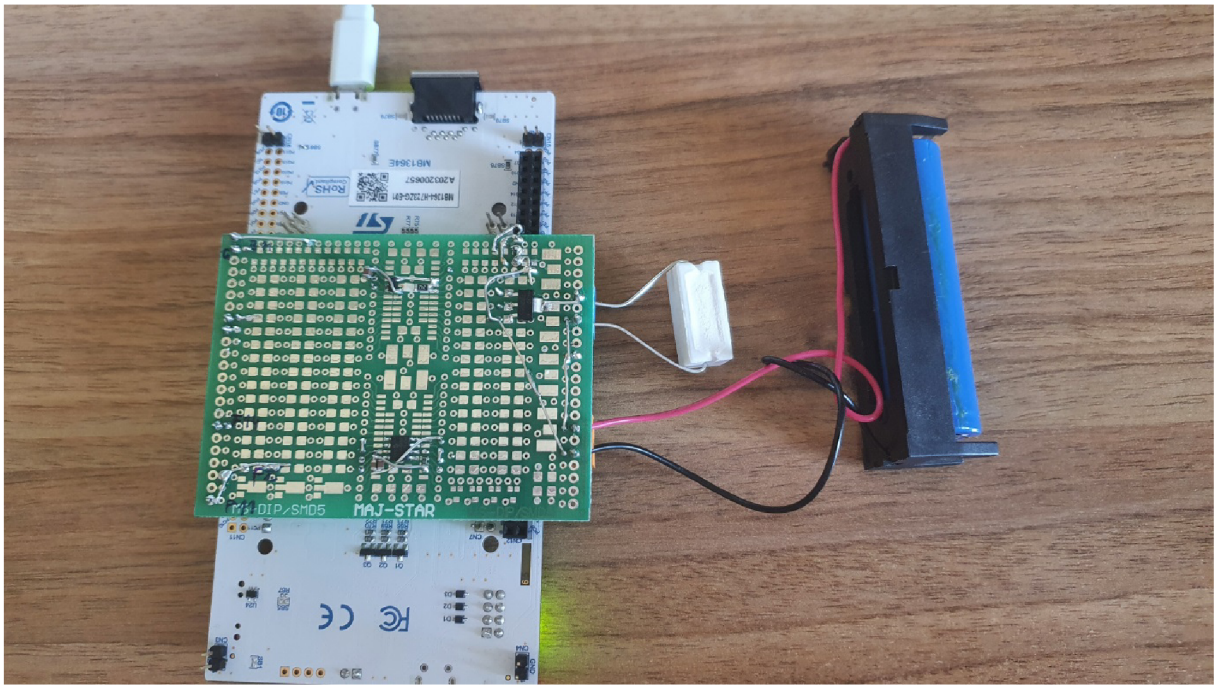


Obrázek 29 – Průměrné hodnoty elektrického napětí z ADC1 od 2. po 10. opakování měření na frekvenci 1 Hz

6 Testování systému

Celý systém – průběh elektrochemické impedanční spektroskopie skrze FW pro HW na platformě STM32 a přidružená klientská aplikace v prostředí Matlab pro vizualizaci měření a jeho spuštění – byl důkladně testován, a to již v průběhu vývoje pro ladění systému, ale zejména pak po zprovoznění celého navrženého řešení. Zapojený systém vidíme na Obrázku 30. Veškeré měření probíhalo na článku HT Li-Ion ICR18650T. V začátcích projektu mikrokontroler nebyl osazen všemi periferiemi – deska neobsahovala např. digitální snímač teploty, ani NTC termistor – a vývoj tak probíhal na testovacím kusu, který spatříme na Obrázku 31.

Obrázek 30 – Testovaný MCU s DSP pro potřeby měření a s připojeným článkem HT Li-Ion ICR18650T



Obrázek 31 – Prvotní prototyp DPS pro měření napětí a proudu na článku

Výsledné testování sloužilo např. pro kontrolu průběhu PWM za pomoci osciloskopu, pro určení kalibračních konstant pro výpočet synchronní detekce, pro ověření správné funkčnosti komunikace mezi mikrokontrolerem a klientské aplikace, nebo také pro porovnání s referenčním měřením.

Měření Li-Ion článku navrženým a sestaveným systémem je dále zdokumentováno v následujících podkapitolách. Ty nejprve popisují průběh PWM přeměřený osciloskopem v laboratoři, dále naměřené napětí a proudu a postup kalibrace těchto veličin, ze kterých následně FW získá hodnoty impedance a fázového posuvu. Výsledně měření vizualizují grafy z klientské aplikace, jejichž průběh dále popisuji. Závěr testování popisuje porovnání naměřených hodnot s referenčním měřením.

6.1 Průběh PWM z osciloskopu

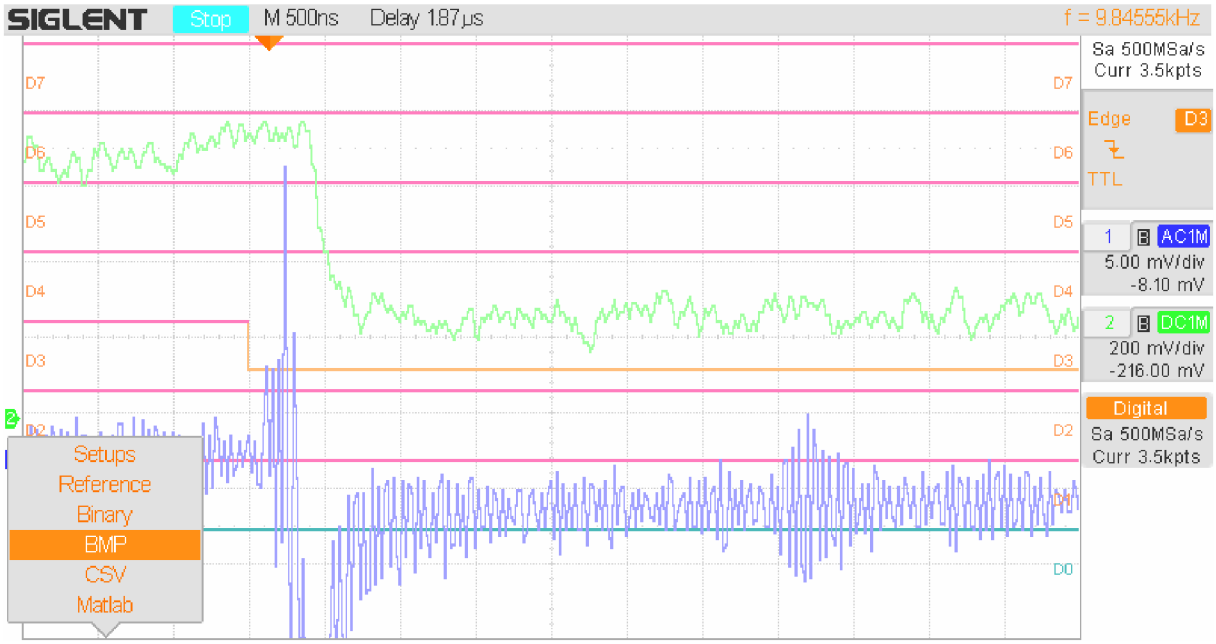
Průběh signálu při sepnutí a vypnutí tranzistoru ukazuje Obrázek 32, respektive Obrázek 33. Napětí v grafu označuje signál s modrou barvou, zelenou má elektrický proud. Můžeme sledovat prodlevu mezi sepnutím (nebo vypnutím) tranzistoru a změnou v napětí (proudu), která lze také vyčíst z grafu a činí zhruba 400 ns. Tranzistory mají konečnou rychlost spínání, proto spočítáme chybu konečnou rychlostí spínání dle vzorce 7:

$$E_s = \frac{t_{on} + t_{off}}{T_{modulace}} (7)$$

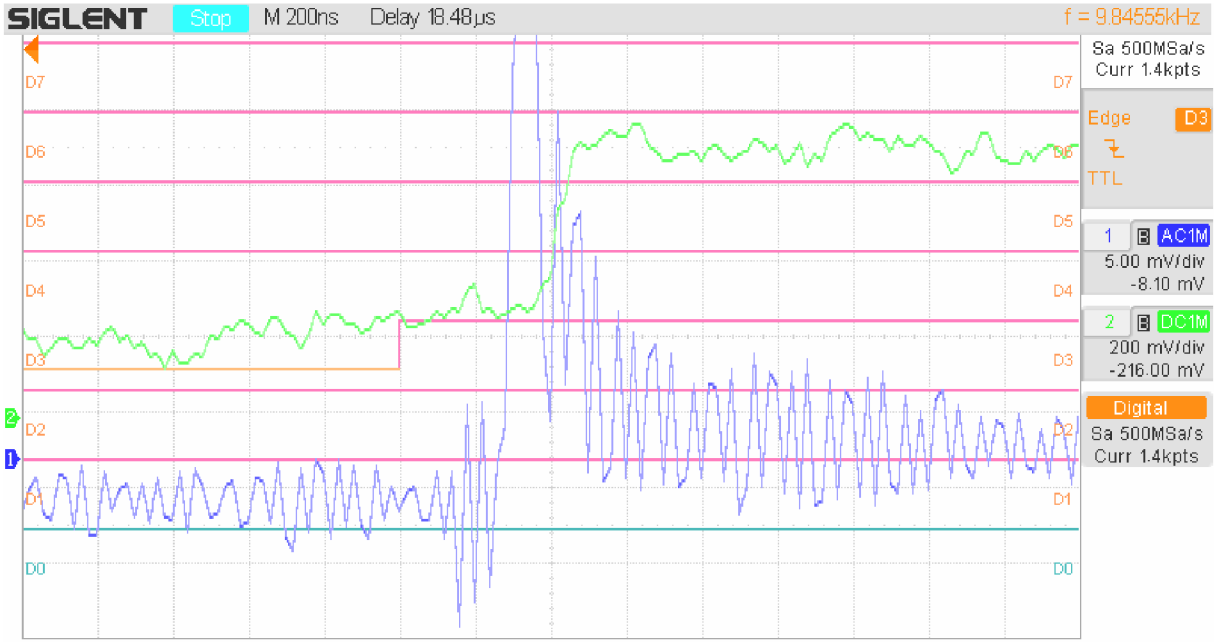
Do něho dosadíme dobu spínání t_{on} rovno 400 ns, stejně tak t_{off}. PWM běží na frekvenci 10 kHz, odtud $T_{modulace}$ = 0,001. Získáváme hodnotu chyby, která činí 0,8 %:

$$0{,}008 = \frac{400\,ns + 400\,ns}{\frac{1}{10}kHz} (8)$$

Proto musíme počítat s tím, že pokud chceme pro PWM nastavit hodnotu z referenční tabulky např. 275, reálná hodnota se může lišit o ± 0,8 % a hodnota tedy může nabývat hodnot cca 273 až 277.



Obrázek 32 – Osciloskopem změřený průběh signálu při sepnutí tranzistoru



Obrázek 33 – Osciloskopem změřený průběh signálu při vypnutí tranzistoru

V grafech je dále viditelný překmit u napětí, který sledujeme i u firmwarem naměřených hodnot při 50% střídě, které vizualizuje klientská aplikace při zobrazení hodnot z bufferu ADC. Překmit se objevuje při úbytku napětí, ale také při jeho vzrůstu. U hodnot proudu k tomuto jevu takřka nedochází, což opět potvrzuje i vlastní měření při správné kalibraci pro výpočet synchronní detekce.

6.2 Kalibrace výpočtu synchronní detekce

Jak blíže popisuje kapitola 4.3, ve výpočtu synchronní detekce figurují dvě konstanty – nárůst napětí a proudu. Naměřené hodnoty z ADC se totiž mírně liší od naměřených hodnot na osciloskopu. To způsobuje pokles napětí při různých tepelných podmínkách. Dle [14] se vnitřní referenční napětí – parametr V_{REFINT} – může pohybovat v rozmezí 1,180 V až 1,255 V. Tento parametr vychází z vnitřního referenčního napětí V_{REF}, které je připojeno ke vstupním kanálům ADC. Pro potlačení tohoto rozdílu přenásobíme vypočtené hodnoty proudu a napětí příslušnými konstantami nárůstu. Následující podkapitoly podrobněji komentují zvolenou metodiku při výpočtu hodnot obou konstant a uvádí konkrétní postup měření a kalibrace.

6.2.1 Metodika kalibrace

Nejprve stanovíme konstantu pro kalibraci podle napětí na článku naprázdno při stejnosměrném proudu. Díky tomu určíme poměr výsledků referenčního měření osciloskopem a oběma ADC na mikrokontroleru. Následně provádíme kalibraci při střídavém proudu na

zvoleném pásmu frekvencí. Kalibrační konstanty bychom mohli určovat pro každou frekvenci zvlášť, ale jak bylo zjištěno při konkrétním měření, rozdíly mezi konstantami jsou víceméně zanedbatelné a pro jednodušší interpretaci byla zvolena střední hodnota vypočtených kalibračních konstant.

Při kalibraci podle napětí na článku naprázdno nejprve stanovíme hodnotu referenčního napětí, které naměří osciloskop. Známe vstupní referenční napětí pro ADC: ADC_{REF} = 3,3 V, převodový poměr vstupního odporového děliče $R_{divider}$ = 1,27 a kalibrační parametr U_{gain} = 1,007 pro výpočet napětí při stejnosměrném proudu. Dále pak nastavené rozlišení pro příslušné ADC, tedy ADC_{RES} = 16 bitů, z něhož vypočteme maximální hodnotu, které vzorky mohou nabývat výpočtem $2\text{^}ADC_{RES}$. Snadno pak vypočteme kalibrační konstantu napětí při stejnosměrném proudu U_{gainDC} dle rovnice 9, kde:

$$U_{gainDC} = ADC_{RES} \times R_{divider} \times U_{gain} \quad (9)$$

Dále spustíme proces impedanční spektroskopie na mikrokontroleru. Jakmile získáme výsledky, které zobrazí klientská aplikace při požadavku na vizualizaci hodnot z ADC bufferu, zjistíme jejich střední hodnotu ADC_{mean}. Vynásobením těchto dvou hodnot získáme naměřené napětí na ADC – parametr ADC_{meas}. Podělením hodnoty referenčního napětí z osciloskopu a naměřeného napětí na ADC získáme poměrnou chybu mezi měřeními tak jako u rovnice 9. Pokud na konci kalibrace opětovně dosadíme hodnoty do tohoto vzorce 10, výsledný poměr rovný U_{err} = 1 značí správný výsledek kalibrace.

$$U_{err} = \frac{U_{refosc}}{ADC_{meas}} \quad (10)$$

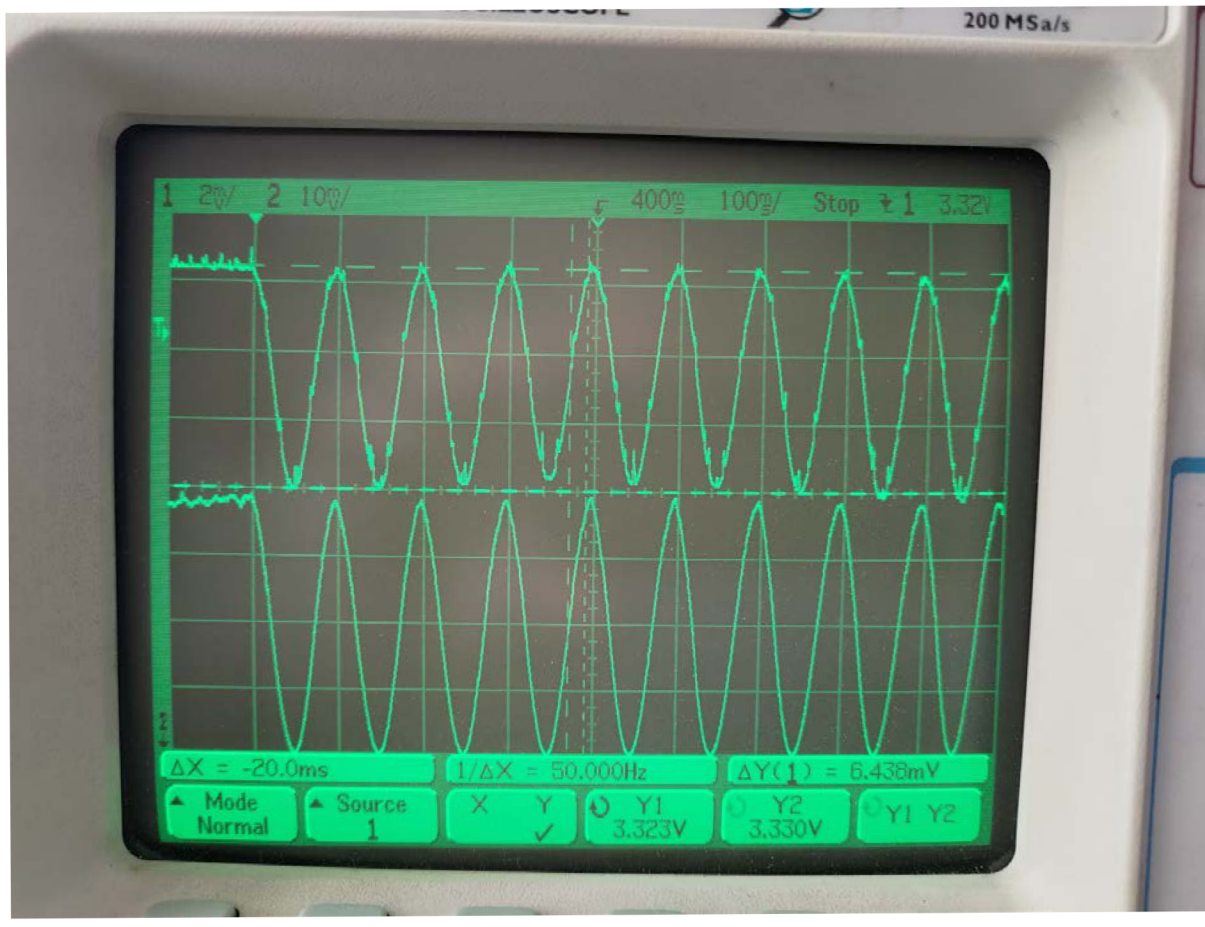
Pro kalibraci při střídavém proudu nejprve volíme pásmo zajímavých frekvencí, na němž se provede měření článku. Postupně zaznamenáme výsledky měření osciloskopem i z ADC. Jak již bylo zmíněno, při vyšších frekvencích se využívá vyššího krokování pro indexaci referenční tabulkou, což musíme uvážit při výpočtu kalibrační konstanty. Pokud je totiž krok nastaven např. na hodnotu 2, výsledná hodnota napětí vypočtená v algoritmu synchronní detekce bude obsahovat počet vzorků děleno 2.

Kalibrační konstantu pro napětí vypočteme jako podíl naměřených hodnot na osciloskopu a výsledných hodnot napětí z mikrokontroleru. Protože použijeme celé pásmo frekvencí, nakonec z něho spočítáme střední hodnotu. U výpočtu konstanty pro kalibraci naměřeného proudu definujeme odpor rezistoru bočníku viz 4.2.1. Obvod obsahuje dva tyto rezistory, proto je odpor R_{load} = 33/2 Ω. Při měření napětí zaznamenáváme i hodnoty proudu. Podělením

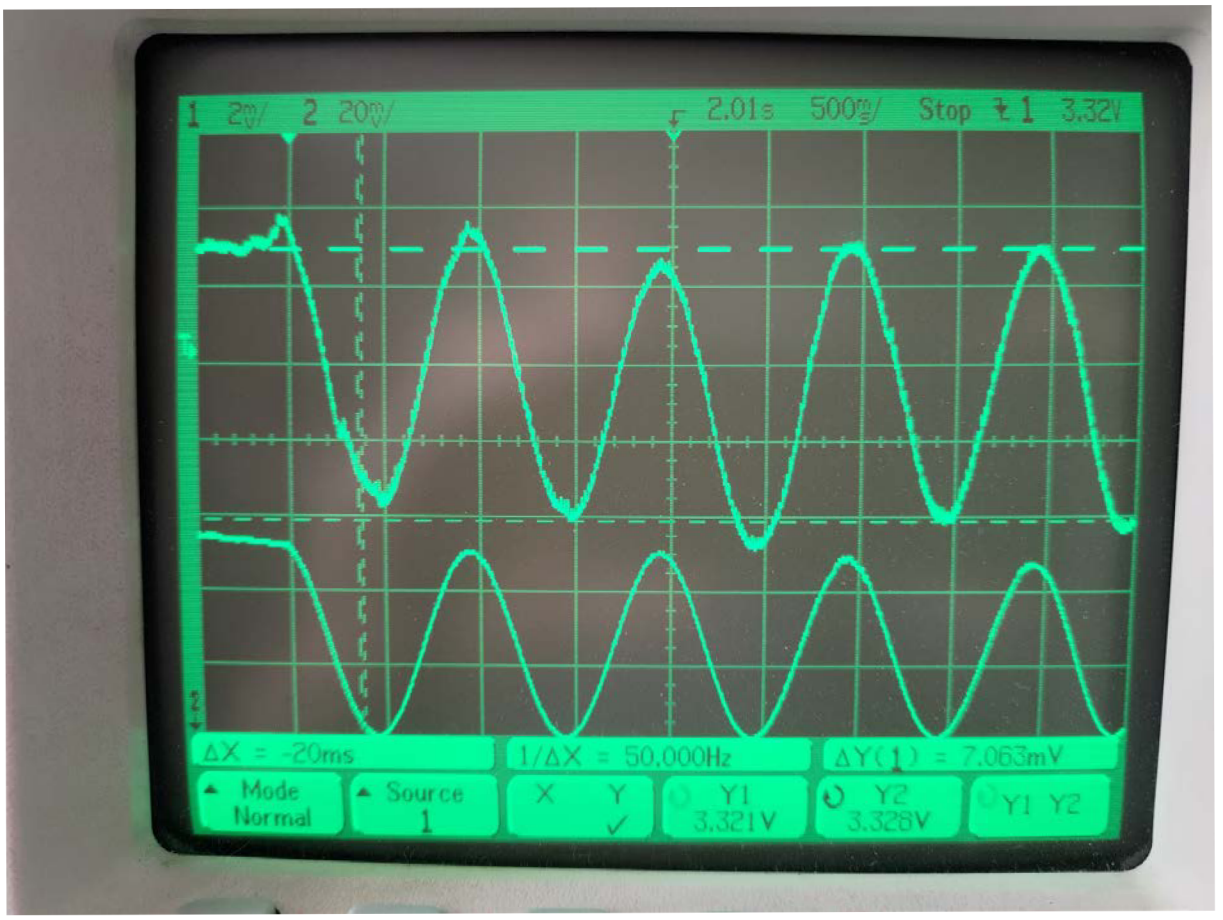
referenčního napětí a odporu R_{load} získáme referenční proud I_{REF}, který použijeme k výpočtu kalibrační konstanty pro každou frekvenci ze zvoleného pásma, a to pouhým poděleníím I_{REF} a naměřeného proudu I_{stm} z mikrokontroleru. Závěrem opět snadno nalezneme střední hodnotu kalibrace. Program obě konstanty implementuje a v příslušném kódu synchronní detekce ji využívá pro přepočet napětí a proudu. Díky tomu docílíme přesného výpočtu impedance jako podílu vypočteného napětí a proudu.

6.2.2 Výpočet kalibračních konstant

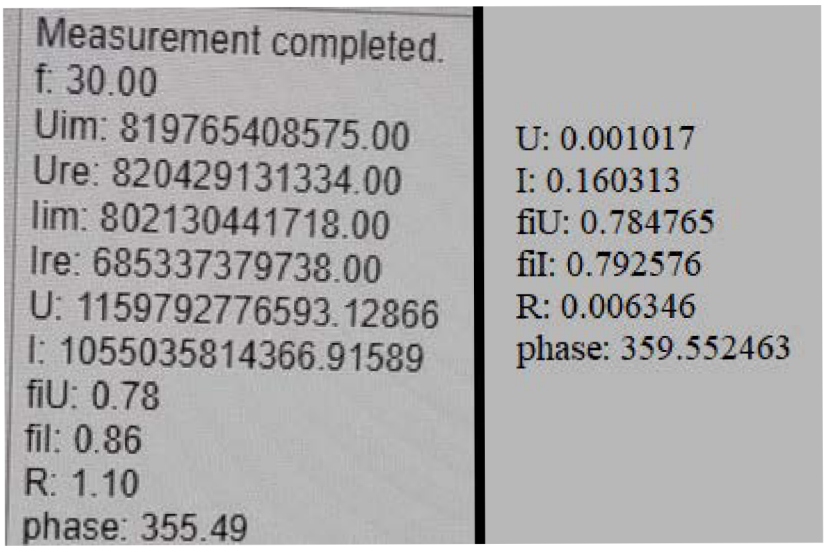
Pro kalibraci při střídavém proudu bylo zvoleno pásmo frekvencí, na kterém se provedlo měření, a to následovně – 300 mHz, 1 Hz, 10 Hz, 30 Hz, 70 Hz a 100 Hz. Referenční měření proběhlo na osciloskopu Agilent 54622D dne 3. 5. 2022. Na následujících obrázcích 34 a 35 vidíme průběhy napětí (vrchní signál) a proudu (spodní signál) článku na vybraném přístroji při konkrétní zvolené frekvenci z pásma. Hodnoty napětí a proudu určuje pozorovaný maximální rozkmit amplitud. Ty se postupně ukládají do prostředí Matlab, kde byl proveden výpočet. Dále na Obrázku 36 sledujeme vypočtené hodnoty mikrokontrolerem před a po dokončení kalibrace. Tyto hodnoty MCU odesílá do klientské aplikace, kde se vypisují do konzole. Parametry U a I z konzole aplikace opět ukládáme do paměti pro následný výpočet kalibračních konstant. Skript provedl jejich dílčí výpočty a vrátil střední hodnoty. Kalibrační konstanty v programu mikrokontroleru se nastavily na zjištěné hodnoty a proběhlo nové měření k ověření správnosti kalibrace. Výsledky pro U a I proto již nejsou nesmyslně vysokého řádu jako doposud, ale odpovídají reálnému napětí a proudu.



Obrázek 34 – Měření proudu a napětí článku osciloskopem pro výpočet kalibračních konstant na frekvenci 10 Hz



Obrázek 35 – Měření proudu a napětí článku osciloskopem pro výpočet kalibračních konstant na frekvenci 1 Hz

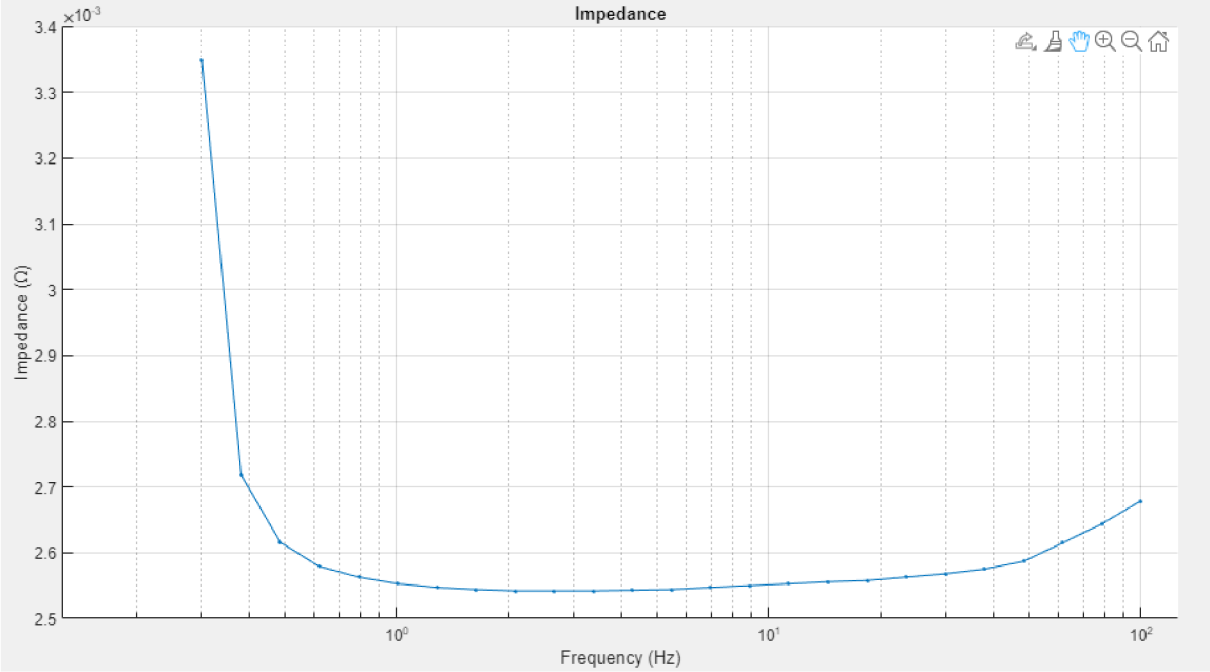


Obrázek 36 – Výstup z klientské aplikace při měření na frekvenci 30 Hz vlevo před zkalibrováním proudu a napětí, vpravo po kalibraci

6.3 Porovnání výsledků s referenčním měřením

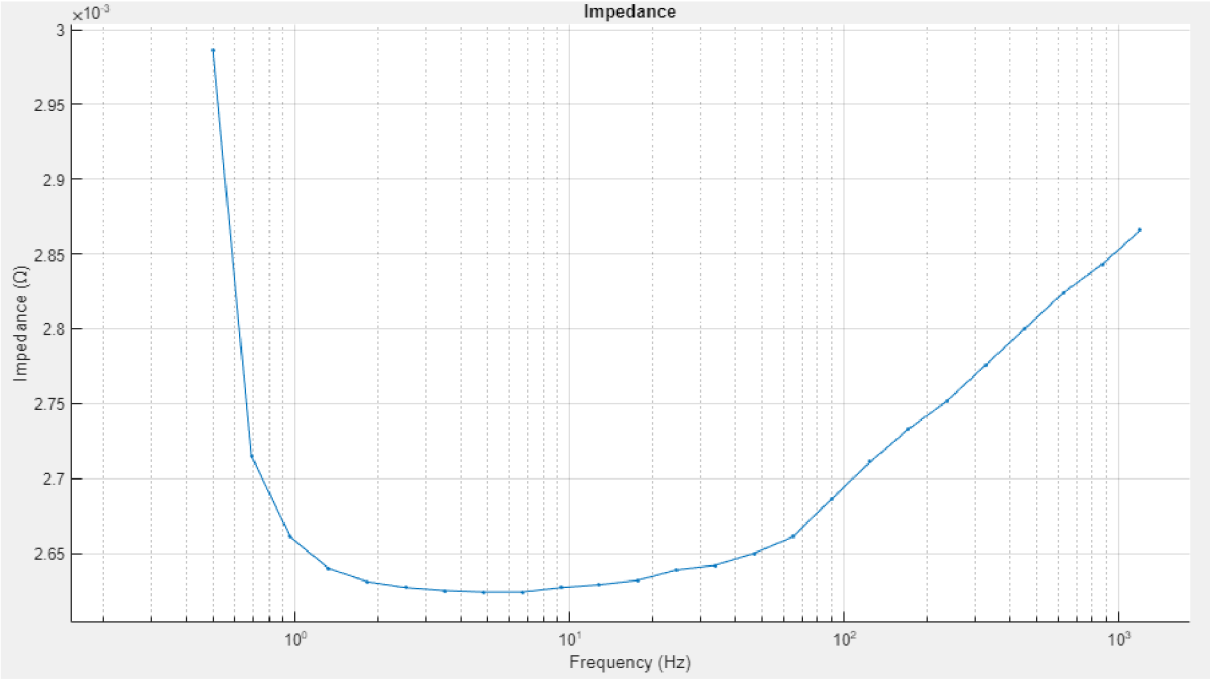
Celý navržený systém se dá velmi snadno ovládat díky klientské aplikaci a volit frekvenční pásmo, na kterém se provádí měření. Jak uvádí kapitola 5 i s obrázky z testování systému, výsledný naměřený proud i napětí vypadají reálně. Po kalibraci výpočtu synchronní detekce dostáváme reálné hodnoty impedance článku, což je vidět na Obrázku 37. Zde vidíme 25 bodů měření na frekvenčním pásmu od 0,3 Hz po 100 Hz. Toto měření trvá déle než 150 sekund kvůli opakování a nízkým frekvencím. Spuštění vlastního měření probíhalo skrze klientskou aplikaci. Program si tak sám vytvořil logaritmickou stupnici dílčích frekvencí, které postupně odesílal s příslušnou instrukcí do MCU.

Nutno podotknout, že vlastní měření probíhalo v domácím prostředí, nikoli v laboratoři a nešlo tak zajistit stálé podmínky, čímž může být měření ovlivněno. I z tohoto důvodu neuvádím měření na nejnižších frekvencích, protože doba trvání je velmi dlouhá a v tomto prostředí nešlo zaručit správné dokončení měření.



Obrázek 37 – Průběh měření absolutní hodnoty impedance na shodném frekvenčním spektru, na kterém byla provedena kalibrace

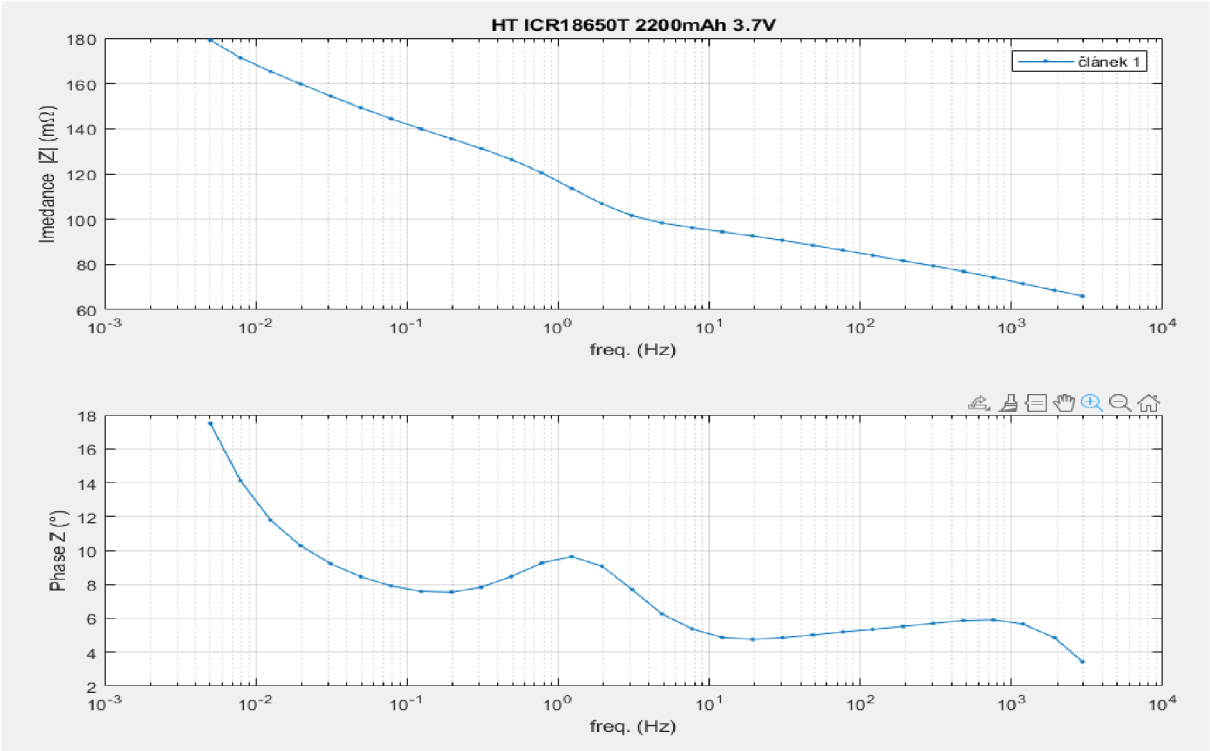
Další měření, které vidíme na Obrázku 38, bylo provedeno na širším pásmu frekvencí – od 0,5 Hz až po 1,2 kHz, kdy počet bodů je opět nastaven na 25. Toto měření zabralo celkem 72 sekund. Výsledná absolutní hodnota impedance opět nabývá velmi nízkých hodnot mezi 2,6 až 3 mΩ.



Obrázek 38 – Vlastní měření na širším pásmu frekvencí

V úvodní kapitole 13 o měření EIS jsme viděli Obrázek 1, který zobrazuje imaginární a reálnou složku impedance pro celé frekvenční pásmo pro článek HT ICR18650T 2200 mAh, 3.7 V. Jedná se o naměřenou impedanci a její imaginární a reálnou složku referenčním měřením na kartě NI CompactDAQ a zesilovačem VCCS. Výstup z vlastní práce ale tvoří grafy pro magnitudu impedance a fázový posuv. Z tohoto důvodu referenční měření Obrázek 39 uvádí i v této formě. Smyslem referenčního měření je nutnost ověření správnosti výsledků z vlastní aplikace. Obě měření probíhaly na stejném typu Li-Ion článku.

Jak vyplývá z grafů, vlastní měření a referenční měření dospívají k rozdílným výsledkům. Ačkoliv obě metody měřily stejný typ článku, nejednalo se o stejný kus, navíc měření neproběhlo ve stejných podmínkách. Z vlastních výsledků ale vidíme, že zvolená metoda funguje a moderní procesor s připojením k PC pohodlně poskytuje možnost testování článků skrze EIS. Ačkoliv výsledky vlastní impedanční spektroskopie se mohou zdát nedostačující nebo nepřesné, práce představuje funkční základ pro měření EIS Li-Ion článků.



Obrázek 39 – Magnituda impedance a její fázový posuv při referenčním měření na frekvenčním pásmu

Závěr

V rámci práce vznikl firmware pro měření elektrochemické impedanční spektroskopie Li-Ion článků, a to za pomoci vybraného hardwaru – mikrokontroleru s přídavnými periferiemi – a díky klientské aplikaci, která poskytuje možnost měření opakovaně spouštět na celém zvoleném frekvenčním pásmu při libovolném počtu bodů. Výsledky měření lze v aplikaci vizualizovat, případně je ukládat. Navržený systém je lehký, stabilní a funkční. Pro kontrolu stavu měření a stavu článku slouží funkce pro komunikaci a dále pro měření teploty na článku. Vybraný HW obsahuje moderní procesor a velmi výkonné periferie – ADC s dostatečnou přesností měření a rychlostí vzorkování, 32bitové časovače pro správnou konfiguraci volené frekvence atd.

Práce splňuje předem deklarované cíle – nejprve popisuje proces elektrochemické impedanční spektroskopie a její metody. Dále zpravuje o způsobech měření amplitudy a fáze vzorkovaného signálu. Dotýká se také modulace signálu za pomoci výpočtu synchronní detekce.

Druhým cílem práce bylo navrhnout jednoduchý firmware za použití předchozích poznatků pro měření EIS. Na vysoce výkonném moderním procesoru došlo k realizaci návrhu firmwaru, což popisuje celá kapitola 4. Práce nejprve seznamuje s principem generování sinusového proudu, jenž excituje signál uvnitř článku. K tomu bylo zapotřebí správně konfigurovat časovače a hlavní časovač TIM5 odbavit procesorem v ním vyvolaném přerušení. Následující bod a příslušná kapitola popisuje princip měření napětí a proudu článku, a to s využitím dvou ADC, které systém konfiguruje a kalibruje při svém spuštění. Pro výpočet impedanční spektroskopie při měření na MCU program využívá výpočtu synchronní detekce. Kapitola pokračuje s popisem nastavení hodinových kmitočtů jádra a zvolených periferií. Následuje princip měření teploty článku dvěma metodami – NTC termistorem a digitálním snímačem LM74. Systém dle požadavků obsahuje i komunikační rozhraní s instrukčním setem pro správu programu. Dále také obsahuje klientskou aplikaci, ve které lze měření ovládat, vizualizovat a ukládat výsledky.

Celý program byl testován na prototypové desce. Práce zmiňuje i kalibraci konstant pro zpřesnění výpočtu synchronní detekce. Dle zadání jsou výsledky porovnávány s referenčním měřením. Zde nalezneme nedostatky práce, protože ačkoliv výstup vlastního měření tvoří reálné a uvěřitelné výsledky, nelze je zcela srovnat s přesným referenčním měřením. Jak popisuje příslušná kapitola, vlastní experimenty byly prováděny pouze v domácím prostředí

s variabilními podmínkami v průběhu ladění systému. I tak ale práce pokládá funkční základ pro výpočet EIS za pomoci dostupného HW a vlastního počítače v libovolném prostředí. Výsledky impedanční spektroskopie systém udává jako absolutní hodnotu impedance a fázový posuv, vhodné by proto bylo doplnit systém o další standardní způsoby reprezentace výstupu měření. Kalibrace synchronní detekce byla provedena pouze v úzkém pásmu frekvencí, což je možné dále rozšířit a zpřesnit tak celý proces. Další možnost rozšíření práce představuje kalibrace konstant za pochodu, pokud bychom se zrovna nacházeli v jiných než běžných podmínkách testování.

Literatura

[1] CHEN, C H, J LIU a K AMINE, 2001. Symmetric cell approach and impedance spectroscopy of high power lithium-ion batteries. *Journal of Power Sources*. 8.

[2] KOSEOGLOU, M., E. TSIOUMAS, D. PAPAGIANNIS, N. JABBOUR a C. MADEMLIS, 2021. A Novel On-Board Electrochemical Impedance Spectroscopy System for Real-Time Battery Impedance Estimation. *IEEE Transactions on Power Electronics* [online]. 1–1. ISSN 1941-0107. Dostupné z: doi:10.1109/TPEL.2021.3063506

[3] NUSEV, G., Đ JURIČIĆ, M. GABERŠČEK, J. MOŠKON a P. BOŠKOSKI, 2021. Fast Impedance Measurement of Li-ion Battery Using Discrete Random Binary Excitation and Wavelet Transform. *IEEE Access* [online]. 1–1. ISSN 2169-3536. Dostupné z: doi:10.1109/ACCESS.2021.3058368

[4] PODHRÁZSKÝ, Adam. ELEKTROCHEMICKÁ IMPEDANČNÍ SPEKTROSKOPIE ELEKTROCHEMICKÝCH ZDROJŮ. Praha, 2019. Diplomová práce. České vysoké učení technické v Praze. Vedoucí práce Ing. Pavel Hrzina, Ph.D.

[5] BOAZ PORAT, A Course in Digital Signal Processing. Wiley, 1997. ISBN 10: 0471149616.

[6] *Synchronous Detectors Facilitate Precision, Low-Level Measurements* [online]. 2014 [cit. 2022-02-12]. Dostupné z: https://www.analog.com/en/analog-dialogue/articles/synchronous-detectors-facilitate-precision.html

[7] *AN5354: Getting started with the STM32H7 Series MCU 16-bit ADC* [online]. STMicroelectronics, 2020 [cit. 2022-04-19]. Dostupné z: https://www.st.com/resource/en/application_note/an5354-getting-started-with-the-stm32h7-series-mcu-16bit-adc-stmicroelectronics.pdf

[8] *MCP1401/02: Tiny 500 mA, High-Speed Power MOSFET Driver* [online]. Microchip, 2014 [cit. 2022-05-01]. Dostupné z: https://ww1.microchip.com/downloads/en/DeviceDoc/20002052D.pdf

[9] *BUK7880-55A: N-channel TrenchMOS standard level FET* [online]. nexperia, 2015 [cit. 2022-05-01]. Dostupné z: https://assets.nexperia.com/documents/data-sheet/BUK7880-55A.pdf

[10] *RM0468: STM32H723/733, STM32H725/735 and STM32H730 Value line advanced Arm®-based 32-bit MCUs* [online]. STMicroelectronics, 2021 [cit. 2022-02-04]. Dostupné z: https://www.st.com/resource/en/reference_manual/rm0468-stm32h723733-stm32h725735-and-stm32h730-value-line-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

[11] *AN4629: ADC hardware oversampling for microcontrollers of the STM32 L0 and L4 series* [online]. STMicroelectronics, 2015 [cit. 2022-03-24]. Dostupné z: https://www.st.com/resource/en/reference_manual/rm0468-stm32h723733-

stm32h725735-and-stm32h730-value-line-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

[12] *Specification of Thin Film NTC Thermistor* [online]. TEWA TEMPERATURE SENSORS, 2019 [cit. 2022-05-06]. Dostupné z: https://www.tme.eu/Document/85c74588be813b44821ab28dae36c629/TT6-10KC8-9-25.pdf

[13] *LM74 SPI/Microwire12-Bit Plus Sign Temperature Sensor* [online]. Texas Instruments, 2013 [cit. 2022-05-06]. Dostupné z: https://www.ti.com/lit/ds/symlink/lm74.pdf?ts=1651176021695&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM74

[14] *STM32H723VE STM32H723VG STM32H723ZE STM32H723ZG: Arm® Cortex®-M7 32-bit 550 MHz MCU, up to 1 MB Flash, 564 KB RAM, Ethernet, USB, 3x FD-CAN, Graphics, 2x 16-bit ADCs* [online]. STMicroelectronics, 2021 [cit. 2022-03-15]. Dostupné z: https://www.st.com/resource/en/datasheet/DM00701028.pdf

[15] *How to Measure Temperature with an NTC Thermistor* [online]. [cit. 2022-04-26]. Dostupné z: https://www.digikey.com/en/maker/projects/how-to-measure-temperature-with-an-ntc-thermistor/4a4b326095f144029df7f2eca589ca54