FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Bakalářská práce

Diagnostika vybraných sériových rozhraní pomocí STM32F722

Studijní program:	B2612 Elektrotechnika a informatika
Studijní obor:	Elektronické informační a řídicí systémy
Autor práce:	**Václav Matura**
Vedoucí práce:	Ing. Miroslav Holada, Ph.D. Ústav informačních technologií a elektroniky

Liberec 2023

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Zadání bakalářské práce

Diagnostika vybraných sériových rozhraní pomocí STM32F722

Jméno a příjmení:	**Václav Matura**
Osobní číslo:	M18000038
Studijní program:	B2612 Elektrotechnika a informatika
Studijní obor:	Elektronické informační a řídicí systémy
Zadávající katedra:	Ústav informačních technologií a elektroniky
Akademický rok:	2022/2023

Zásady pro vypracování:

1. Seznamte se s přípravkem s mikrokontrolérem STM32F722RI na pracovišti školitele. Proveďte rešerši možností monitorování základních komunikačních rozhraní – sériových sběrnic I2C, SPI, UART a RS485.
2. Navrhněte systém s STM32F7xx, který bude diagnostikovat tok dat (formát, časová posloupnost) a kvalitu signálu pro následující sériová rozhraní: UART, I2C, SPI a RS485, případně další (CAN, USB, OneWire). Zobrazení a vyhodnocení získaných dat řešte v nadřazeném PC (např. v prostředí Matlab).
3. Navržený systém realizujte na úrovni vývojového prototypu včetně vlastního firmwaru mikrokontroléru a zobrazovacího softwaru v nadřazeném PC.
4. Realizovaný systém ověřte v ukázkových úlohách.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	30-40 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	Čeština

Seznam odborné literatury:

[1] NOVÁK, Petr. Mobilní roboty: pohony, senzory, řízení. 1. vyd. Praha: BEN – technická literatura, 2005. ISBN 80-7300-141-1.

[2] HESS, Filip. Využití obvodu ESP32 pro virtuální průmyslovou sběrnici: Using the ESP32 circuit for the virtual fieldbus. Liberec: Technická univerzita v Liberci, 2020. Bakalářské práce. Technická univerzita v Liberci.

[3] www.st.com

Vedoucí práce:	Ing. Miroslav Holada, Ph.D. Ústav informačních technologií a elektroniky
Datum zadání práce:	24. října 2022
Předpokládaný termín odevzdání:	22. května 2023

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

prof. Ing. Ondřej Novák, CSc.
vedoucí ústavu

V Liberci dne 24. října 2022

Prohlášení

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

Jsem si vědom toho, že na mou bakalářskou 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 bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou 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 bakalářská 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.

22. května 2023 Václav Matura

Diagnostika vybraných sériových rozhraní pomocí STM32F722

Abstrakt

Tato bakalářská práce se věnuje diagnostice sériových rozhraní I2C, RS485, UART a SPI s využitím mikroprocesoru STM32F722, kterým je osazen přípravek na pracovišti vedoucího práce, a následné zobrazování v aplikaci v nadřazeném počítači vytvořené v Matlabu. V práci byl navržen firmware, který využívá možnosti vývojového kitu pro sbíraní dat k následnému zpracování. Souběžně byla vytvořena aplikace k zobrazení nasbíraných dat v nástroji App Designer, který je dostupný v prostředí Matlab.

Klíčová slova: STM32F722RE, monitorování sériových sběrnic, App Designer, Aplikace pro zobrazování dat.

Diagnostics of selected serial interfaces using STM32F722

Abstract

This bachelor thesis is devoted to the diagnosis of serial interfaces I2C, RS485, UART and SPI using the STM32F722 microcontroller, which is equipped with a fixture at the workplace of the supervisor of the thesis, and the subsequent display in the application in the parent computer created in Matlab. In this thesis, firmware has been designed that uses the capabilities of the development kit to collect data for post-processing. In parallel, an application was created to display the collected data in the App Designer tool, which is available in the Matlab environment.

Keywords: STM32F722RE, serial bus monitoring, App Designer, Data display application.

Poděkování

Chtěl bych poděkovat Ing. Miroslavu Holadovi,Ph.D. za odborné vedení, za pomoc a rady při zpracování této práce.

Obsah

Seznam zkratek

I2C	Inter-Integrated Circuit
UART	Universal Asynchronous Receiver-Transmitter
SPI	Serial Peripheral Interface
RS485	Recommended Standard 485
SMD	Surface Mount Device
LED	Light-Emitting Diode
GND	Ground
SWIM	Single Wire Interface Module
JTAG	Joint Test Action Group
HAL	Hardware Abstraction Layer
ACK	Acknowledgment
ADC	Analog-to-Digital Converter
DMA	Direct Memory Access
FAT	File Allocation Table

Seznam obrázků

1 Úvod

Motivací pro tuto práci byla možnost za pomocí mikroprocesoru STM32F722 sbírat data ze sériových sběrnic a usnadňovat tak práci při debuggování, využívaní a zkoumání dění na sériovém přenosu dat. Sériová komunikace se vyskytuje v mnoha aplikacích a projektech. Pro jejich monitoring je potřeba docela drahých zařízení jako jsou osciloskopy, nebo logické analyzátory, které jsou často spousty vývojářům nedostupné. Jsou zároveň náročné na prostor na pracovní ploše nebo na přemísťování. Nevýhodou je také nutnost připojení do elektrické sítě.

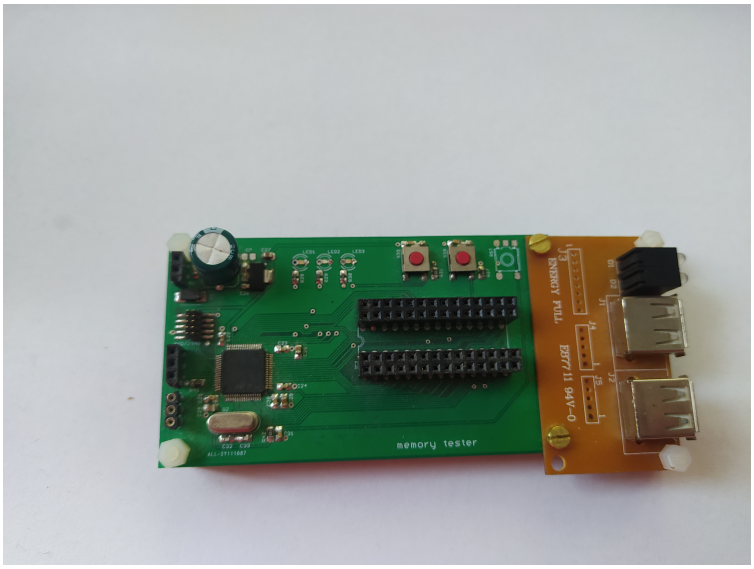
Úkolem bylo vytvoření zařízení na bázi prototypu s mikrokontrolerem STM32F722, které umožní sbírat data na sběrnicích jako jsou I2C, UART, SPI nebo RS485 a následné posíláni do nadřazeného počítače. Zařízení by mělo disponovat nejméně dvěma kanály pro sběr dat a komunikačním prostředkem s počítačem.

Dalším úkolem byla možnost zobrazení naměřených dat v nadřazeném zařízení. Pro tuto úlohu by mělo být vybráno vhodné prostředí pro práci s velkým počtem dat a následným přehledným zobrazení. Vše by mělo být odzkoušeno na ukázkových úlohách a tím ověřit funkci přípravku.

2 Rešerše a seznámení se s přípravkem s mikrokontrolerem STM32F722

2.1 Vývojový kit

Vývojový kit, který je k dispozici na pracovišti vedoucího práce se skládá z mikroprocesoru STM32F722, ze kterého jsou vyvedeny piny na dvouřadý hřeben. Mezi vyvedenými piny jsou k dispozici napájecí +5V a GND a volitelné vstupní a výstupní piny. Přípravek disponuje třemi SMD svítivými diodami červené, zelené a žluté barvy s označením LED1, LED2 a LED3, které jsou vyvedeny na porty PB13, PB14 a PB15. Vedle svítivých diod se nachází dvě tlačítka označená jako S29 a S33, které slouží jako vstupní digitální piny. Je zde i možnost přidání dalšího tlačítka pro funkci resetu. Mezi další vyvedené piny patří rozhraní pro programování mikrokontroleru JTAG a dva hřebeny, dvou pinový pro napájení +5V a GND a tří pinový pro sériové rozhraní UART tedy RX a TX piny. Kit také disponuje 16MHz krystalem, což je jeho velká výhoda oproti komerčně dostupným vývojovým kitům.



Obrázek 2.1: Připravek s mikrokontrolerem

Během práce bylo potřeba rozšíření o USB konektory typu A. USB konektor je pomocí drátků ze spodní strany připájen k vývodním pinům mikrokontreleru používaných pro připojení a komunikaci s USB zařízením. Desky jsou k sobě připojeny

pomocí šroubů.

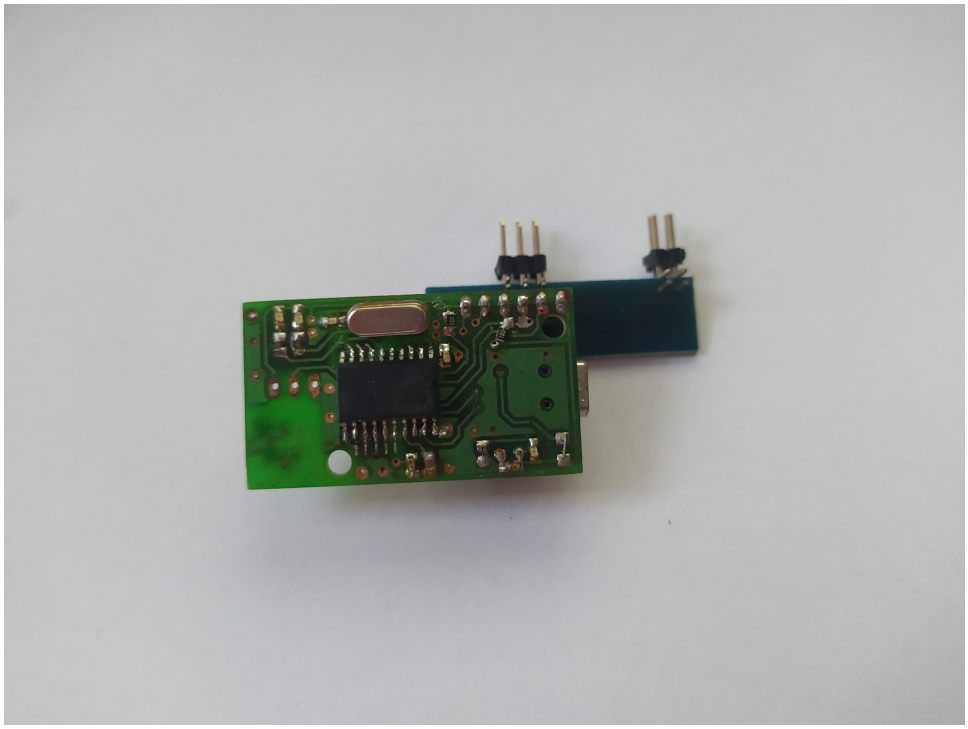
2.2 Programátor ST-LINK/V2

Programátor ST-LINK/V2 slouží pro nahrání a debuggování programu pro mikrokontrolery STM32. Pro komunikaci slouží SWIM rozhraní a rozhraní JTAG pro sériové debuggování. Lze připojit jak k STM8, tak STM32. Při zapojení s mikroprocesorem STM32 se využívá USB full-speed rozhraní ke komunikaci se STM32CubeIDE, nebo s jinými vývojovými prostředími třetích stran. Je napájen pěti volty z konektoru USB. K indikaci připojení ke COM portu slouží svítivá dioda. [1]

Obrázek 2.2: ST-LINK/V2

2.3 Obvod pro sériovou komunikaci

Na třípinový a dvoupinový hřeben je připojena destička, na které se nachází obvod MCP2200. Jedná se o sériový převodník z USB na UART. Destička slouží ke komunikaci s nadřazeným počítačem. Pomocí této destičky lze z USB portu počítače mikrokontroler napájet.



Obrázek 2.3: Obvod pro sériovou komunikaci

2.4 Vývojové prostředí STM32CubeIDE

Je vývojové prostředí pro mikroprocesory STM32, které lze naistalovat na Windows, Linux i macOS. Poskytuje podporu pro jazyk C a C++. Nabízí pokročilé možnosti při debuggingu a konfiguraci mikroprocesoru. Umožnuje automatické generování inicializačních kódů. Inicializaci lze kdykoliv v průběhu vývoje měnit bez zásahu do vlastně napsaného kódu, pokud je vložen do vyhrazených míst pro vlastní kód. Obsahuje analyzátory, které vývojářům poskytují užitečné informace o stavu projektu a paměti. Poskytuje též standartní a pokročilé funkce ladění včetně zobrazení základních registrů, nebo živé sledování proměnných.[2]

2.5 Knihovna funkcí HAL

HAL je knihovna poskytované společností STMicroelectronics pro jejich mikrokontrolery STM32. Poskytuje abstrakční vrstvu nad hardwarovými periferiemi mikrokontroleru. Díky této vrstvě je možné různé periferie jako GPIO, UART, I2C, časovače, AD převodníky a další nastavit bez přímého vstupu k nízko úrovňovým registrům.[3]

Kód vytvořený pomocí této knihovny není závislí na konkrétní variantě mikrokontroleru. Knihovna poskytuje konzistentní API napříč různými rodinami STM32. Kód lze tedy použít i pří změně modelu mikrokontroleru [3]

I když knihovna HAL zjednodušuje díky vyšší abstrakci programování, může představovat nežádoucí režii v určitých případech nemusí být nejefektivnější. [3]

2.6 Matlab App Designer

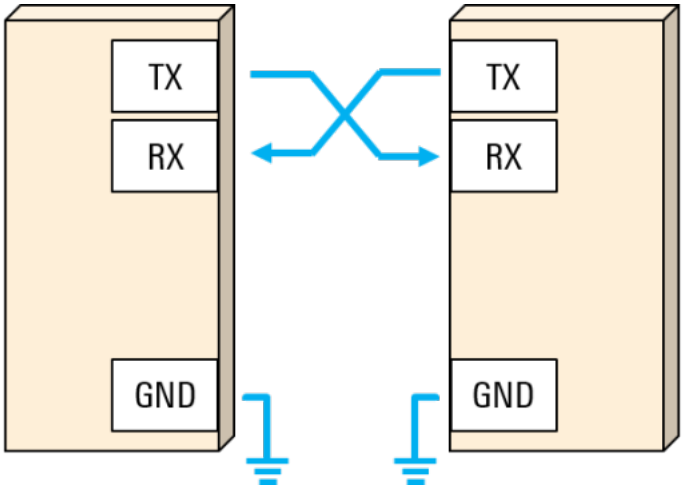
Pro naprogramování aplikace v nadřazeném PC je zvolen jazyk Matlab a jeho grafické vývojové prostředí App Designer Jedná se o grafické vývojové prostředí od společnosti MAthWorks. Umožnuje vývojářům vytvářet interaktivní grafické uživatelské rozhraní bez větší znalosti programování. Lze využívat přetahování funkčních komponent jako jsou tlačítka, posuvníky, grafy a následné nastavení vlastností a chování v editoru, nebo přímo napsáním kódu. Pomocí zpětných volání jako je klinutí na tlačítko, nebo změna nějaké vlastnosti prvku je možné spouštět vlastní kód, který vytváří logiku aplikace.[4]

MATLAB App Designer zjednodušuje proces vytváření aplikací MATLAB a umožňuje uživatelům rychle vytvářet prototypy a nasazovat interaktivní aplikace pro analýzu dat, vizualizaci, řídicí systémy, simulace a další. Nabízí intuitivní rozhraní, které kombinuje sílu MATLABu se snadným vizuálním vývojem, takže je přístupné uživatelům s různou úrovní programátorských zkušeností.[4]

Přichází jako součást SMT32CubeIDE spolu s ovladači zařízení middlewarem a ukázkovými úlohami.[4]

2.7 Sériové rozhraní UART

Zkratka UART znamená Universal Asynchronous Receiver and Transmitter. Definuje protokol pro sériový přenos dat mezi dvěma zařízeními. Je velmi jednoduchý. Používá dva vodiče mezi vysílačem a přijímačem v obou směrech. Oba konce musí být uzemněny. Komunikace může probíhat jedním směrem tzv. simplexní, nebo poloduplexní, při které vysílá vždy pouze jedna strana, nebo plně duplexní, kdy obě strany posílají data najednou.[5]

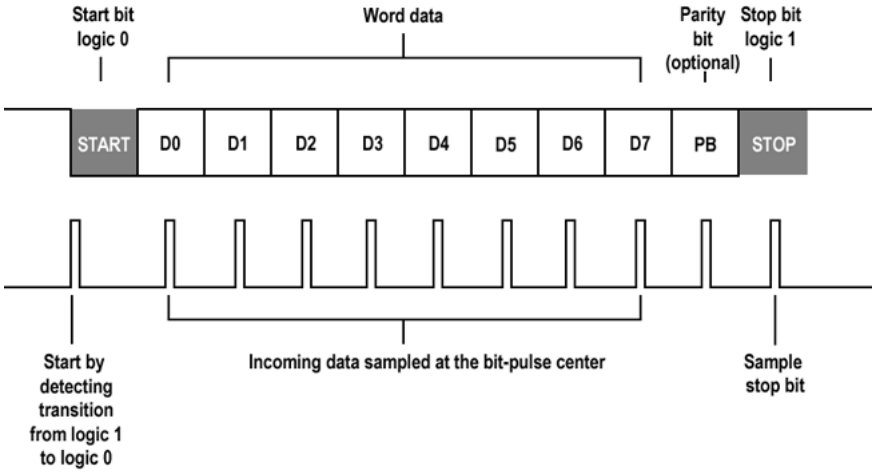


Obrázek 2.4: Zapojení zařízení na sběrnici UART [5]

Jedna z velkých výhod je jeho asynchronost tedy není nutný hodinový signál. Vysílač i přijímač musí vysílat stejnou rychlostí, aby měli stejné bitové časování, stejnou strukturu rámce a parametry jako je stop bit, či parita. [5]

Používá nejběžnější kódování logické 1 a 0 a to takové že vysoká úroveň je 1 a nízká úroveň je 0. V klidovém stavu je sběrnice udržována ve vysoké úrovni, kvůli snadnější detekci poškození vedení nebo zařízení.[5]

Kvůli detekci přicházejících datových bitů se používá start bit. Start bit je přechod z nečinného stavu vysoké úrovně do nízké úrovně. Po tomto stavu pak následují datové bity. Na konci datových bitů je stop bit, který buď setrvá ve vysokém stavu nebo přejde z nízkého na vysoký stav. Lze nakonfigurovat i druhý stop bit. Díky druhému stop bitu má přijímač více času připravit se na další rámec dat. [5]



Obrázek 2.5: UART rámec [6]

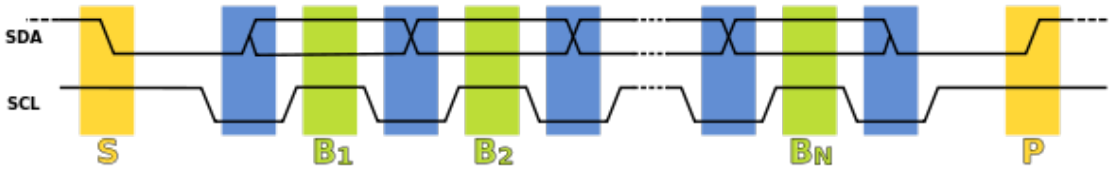
2.8 Sériová sběrnice I2C

I2c je multi-masterová sériová sběrnice vytvořená firmou Philips. Používá se k připojovaní nízko rychlostních zařízení. Je identická se sběrnicí TWI, kterou využívají výrobci místo chráněné značky I2C. [8]

Datové bity, tedy užitečných uživatelských dat je nejčastěji 7 až 8, ale je možno se pohybovat v rozmezí 5 až 9 bitů.[8]

Další bit může být volitelný paritní bit, který slouží pro detekci chyb. Vyskytuje se na konci datových bitů. Může být použita sudá nebo lichá parita. Zařízení na sběrnici se rozdělují na ty, které řídí komunikaci tzv. master a na ty které jsou řízené a mají svoji adresu. Těmto se říká slave. [8]

Sběrnice umožňuje propojení až 128 zařízení. Ty jsou propojeny pouze dvěma vodiči. Jeden se využívá pro hodinový signál a druhý pro přenos dat. Datový vodič se nazývá SDA a hodinový SCL. Každý vodič je připojen pull-up rezistorem, kvůli udržení vysoké úrovně v klidovém stavu. Logická úroveň na SDA se smí měnit pouze, když je SCL v nízké úrovni. Toto pravidlo je porušeno pouze při vysílaní start a stop bitů při zahájení a ukončení komunikace na sběrnici. I2C neumožnuje duplexní přenos, vysílat data může pouze jedno zařízení.[8]



Obrázek 2.6: časový diagram I2C[8]

Před každým přenosem se vyšlou dva start bity. Poté následuje sedmibitová

adresa zařízení, která má přijímout data a jeden read/write bit, který určí, jestli se budou zapisovat nebo číst data. Komunikace pokračuje bitem ACK, který potvrdí připravenost zařízení přijímat. Každý následující byte ukončuje ACK bit. Na konci komunikace je stop bit. [8]

U řízení komunikace po sběrnici se používá detekce kolizí. Když je sběrnice v klidovém stavu, každé zařízení může zahájit vysílaní dat. Pokud nastane rozdíl mezi skutečnými daty a stavem sběrnice je detekována kolize. Tento stav může začít, když jedna sběrnice vysílá vysokou úroveň a zároveň druhá nízkou úroveň. V některých případech může nastat stav takový že, zařízení posílají po sběrnici stejná data. Kolize pak není detekována. [8]

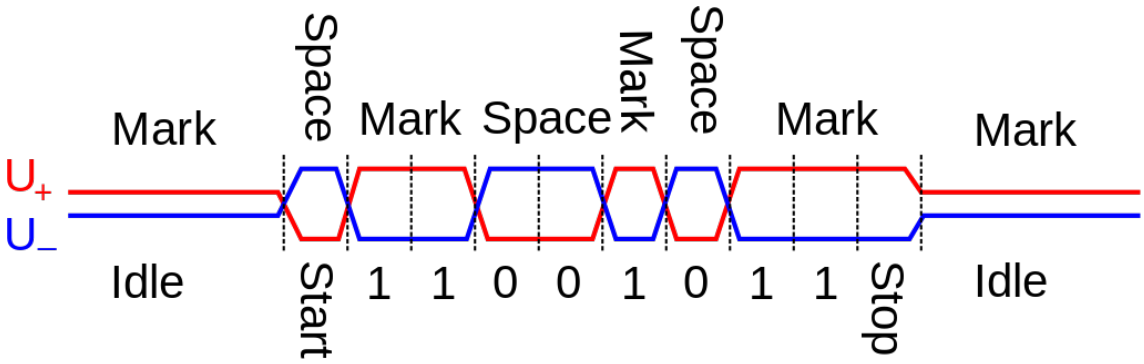
Každý uživatel sběrnice má svou vlastní sedmibitovou adresu. Po vyslání start bitu všechny zařízení porovnávají svou adresu s adresou, která je posílána z masteru. Při shodě adresy se svou adresou je poslán ACK bit. Existují adresy, které jsou vyhrazeny pro speciální účely. Například 000000 je adresa určena pro broadcast. [8]

2.9 Sériová komunikace po RS485

Jedná se o sériové rozhraní používané od roku 1983, Kdy bylo definováno sdružením EIA. Je využíváno hlavně v průmyslových prostředích. Od standartu RS-232 se liší pouze definicí napěťových úrovní. Na sběrnici může komunikovat až 32 zařízení, po použití opakovačů i více, na vzdálenost až 1200m. Při komunikaci na takto velké vzdálenosti musí být na konec vedení zapojeny odpory kvůli odrazům. [7]

RS-485 používá dva vodiče. Označují se A a B nebo + a -. Klidový stav se vyznačuje menším napětím na vodiči A. Logické úrovně jsou pak reprezentovány rozdílovým napětím mezi vodiči. Tento způsob je výhodný pro potlačení indukovaného rušivého signálu, který se přičítá ke každému vodiči stejně. Logická 1 se určí rozdílem mezi vodičem A a vodičem B menším než 200mV. Logický stav 0 naopak při rozdílu A a B větším než 200mV. [7]

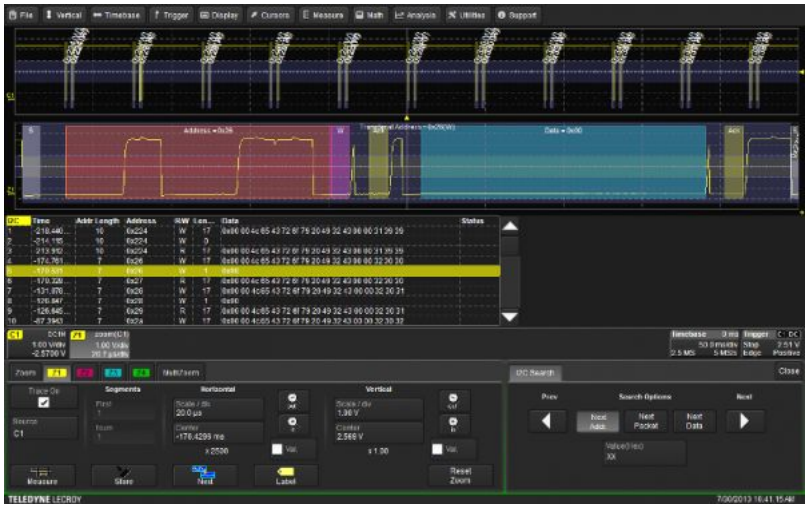
Přenos probíhá pomocí sedmibitového rámce s jedním nebo více stop bity. Lze nastavit i paritní bit. Startbit je kódován jako logická 0, stop bit s neaktivním stavem logickou 1.[7]



Obrázek 2.7: Ukázka rámce RS-485 [7]

2.10 Možnost monitorování pomocí dekodéru sériových sběrnic pro osciloskopy Teledyne LeCroy

Dekodéry sériových sběrnic pro osciloskopy Teledyne LeCroy nabízí pokročilé softwarové algoritmy. Rozloží zachycený průběh na protokolové údaje a následně zobrazí dekódované rámcové informace pomocí obdélníkových tvarů v různých barvách, které jsou vykresleny na základě změřeného signálu. Dekódované informace se upravují tak, aby byly čitelně zobrazeny, a to buď zhuštěním nebo roztahováním v závislosti na aktuálním nastavení časového rozmezí nebo přiblížení časové osy. Díky intuitivnímu způsobu zobrazení dekódovaných informací pomocí barevných obdélníků překládaných přes změřený signál jsou všechny informace a vztahy dobře srozumitelné i pro uživatele, kteří se teprve seznamují se sběrnicemi I2C, SPI, UART nebo RS-232. Dekódování je rychlé i pro velmi dlouhé záznamy signálu. Uživatel má možnost vybrat si zobrazení dekódovaných informací ve formátu Hex, Binary nebo ASCII. Dekodér je schopen fungovat i v případě, kdy není signál hodin zapnutý. [9]



Obrázek 2.8: Teledyne LeCroy][9]

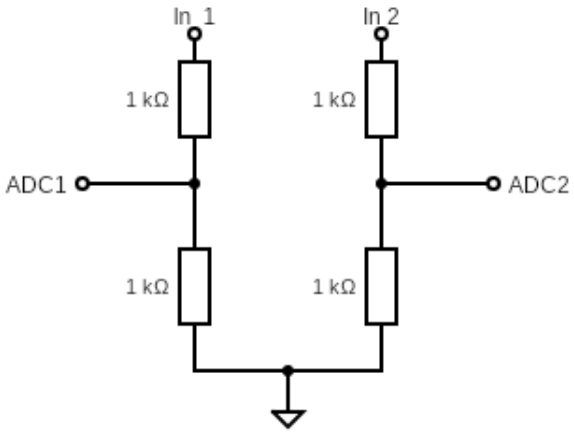
2.11 Možnost monitorování pomocí logického analyzátoru s mikrořadičem

Tento logický analyzátor byl navržen s důrazem na jednoduché obvodové řešení, s cílem sloužit výukovým účelům. Pro dosažení tohoto cíle byl použit mikrokontroler STM32F303RE na desce Nucleo a spojení s počítačem je zajištěno pomocí USB kabelu. Pro vizualizaci dat z logického analyzátoru byla využita stávající počítačová aplikace PulseView. Tato aplikace umožňuje nastavení parametrů logického analyzátoru a poskytuje také funkce pro analýzu digitálních komunikací.[10]

3 Návrh systému pro diagnostiku dat

3.1 Připojení měřeného systému k AD převodníkům

AD převodník mikrokontroleru STM32 má referenční napětí 3,3V. Kvůli tomu bylo nutné upravit signál ke zpracovaní převodníkem. Většina zkoumaných sériových rozhraní používá logiku o hodnotách napětí maximálně +5V. Byla tedy vytvořena prototypová destička se čtyřmi rezistory o hodnotě 1kΩ zapojenými do dvou napěťových děličů, kteří omezí příchozí napětí na polovinu. Odpory jsou umístěny v pájivém poli, které se za pomocí dvou nožičkového jumperu připojuje do hřebenu na piny mikrokontroleru.



Obrázek 3.1: Schéma odporového děliče

3.2 Nastavení analogově-digitálních převodníků

Pro první verzi bylo zvoleno dvoukanálové nastavení ADC1 převodníku s DMA kruhovým bufferem. Ukázalo se však, že dva kanály nemohou běžet na stejné frekvenci. Jeden kanál měl jinou vzorkovací frekvenci než druhý z důvodu jiného počtu cyklů mikrokontroleru potřebnou pro převod. Toto nastavení se tedy ukázalo jako slepá cesta.

Pro sbírání vzorků měřeních dat jsou tedy využity dva AD převodníky, a to ADC1 a ADC2. Oba převodníky jsou nastavené stejně a to do DMA kruhového režimu. Tento režim umožnuje vyvolání dvou přerušení z knihovny HAL, které se spustí, když je plná dolní polovina bufferu. Druhé přerušení se vyvolá, když je plná horní polovina bufferu. DMA zároveň umožnuje rychlý přesun dat z paměti na další periferie.

Oba převodníky jsou nastaveny do nezávislého módu. Rozlišení pro kvantování sledovaných signálů je nastaveno na 12 bitů. Díky tomuto rozlišení je každá konverze provedena během patnácti hodinových cyklů mikrokontroleru. Data jsou zarovnána zprava, jak bylo nastaveno automaticky

K vyvolání převodu je zvolena externí konverze pomocí časovače Timeru1. Převod se spustí pokaždé když je u časovače detekována náběžná hrana přetečení časovače.

V nastavení přerušení je povolen globální přerušení přímého přístupu do paměti.

3.3 Nastavení timeru

Pro určení vzorkovací frekvence byl vybrán časovač Timer 1. Jeho šestnáctibitový registr prescaler je nastaven na hodnotu 216. Tento registr umožňuje vydělení frekvence mikrokontroleru, která je nastavena na 216MHz. Registr counter period, který naopak násobí periodu přetečení je nastaven na hodnotu 1. Frekvence přetečení časovače byla určena na 1MHz a nastavena pomocí vzorce. 3.1.

$$f = \frac{f_{MCU}}{Prescaler + 1} \times CounterPeriod = \frac{216000000}{215 + 1} \times 1 = 1Mhz \tag{3.1}$$

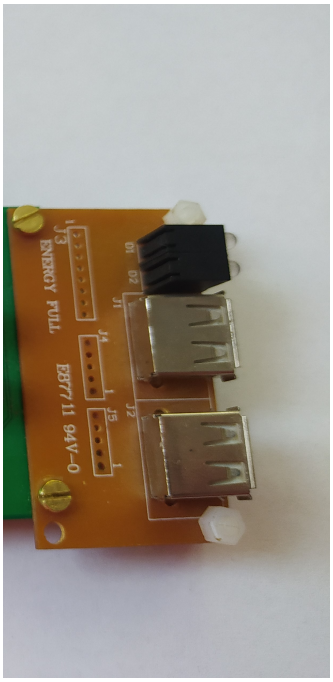
3.4 Komunikace s nadřazeným PC

Pro první verze prototypu bylo plánovaná komunikace s počítačem pomocí ukládaní dat na externí paměťové zařízení například flash disk. K mikrokontroleru byla přidána destička s konektorem USB pro připojení externí paměti. V nastavení mikrokontroleru byla provedena konfigurace USB hosta a souborového systému FAT.

Bylo zjištěno že tato varianta není příliš vhodná pro aplikace vyžadující rychlé zápisy a zpracování dat. Mikrokontroler se zpožďoval otevíráním a aktualizací dat v souboru a nestíhal další detekci hodnot na sběrnici. Byla tedy zvolena metoda přes sériovou linku, která se ukázala jako výhodnější pro tuto aplikaci.

3.5 Nastavení seriové komunikace s PC

Ke komunikaci s nadřazeným počítačem bylo zvoleno sériové rozhraní UART. Rychlost přenosu je nastavena 115200 Bit/s. Rychlost je zvolena adekvátně k délce vodiče připojenému k USB portu počítače. Délka rámce je osmibitová včetně paritního bitu, který není nakonfigurován. Na konci rámce se nachází jeden stop bit.



Obrázek 3.2: Destička s USB porty pro připojení USB

Stejně jako u přerušení analogově-digitálních převodníků je zvolena verze DMA přerušení.

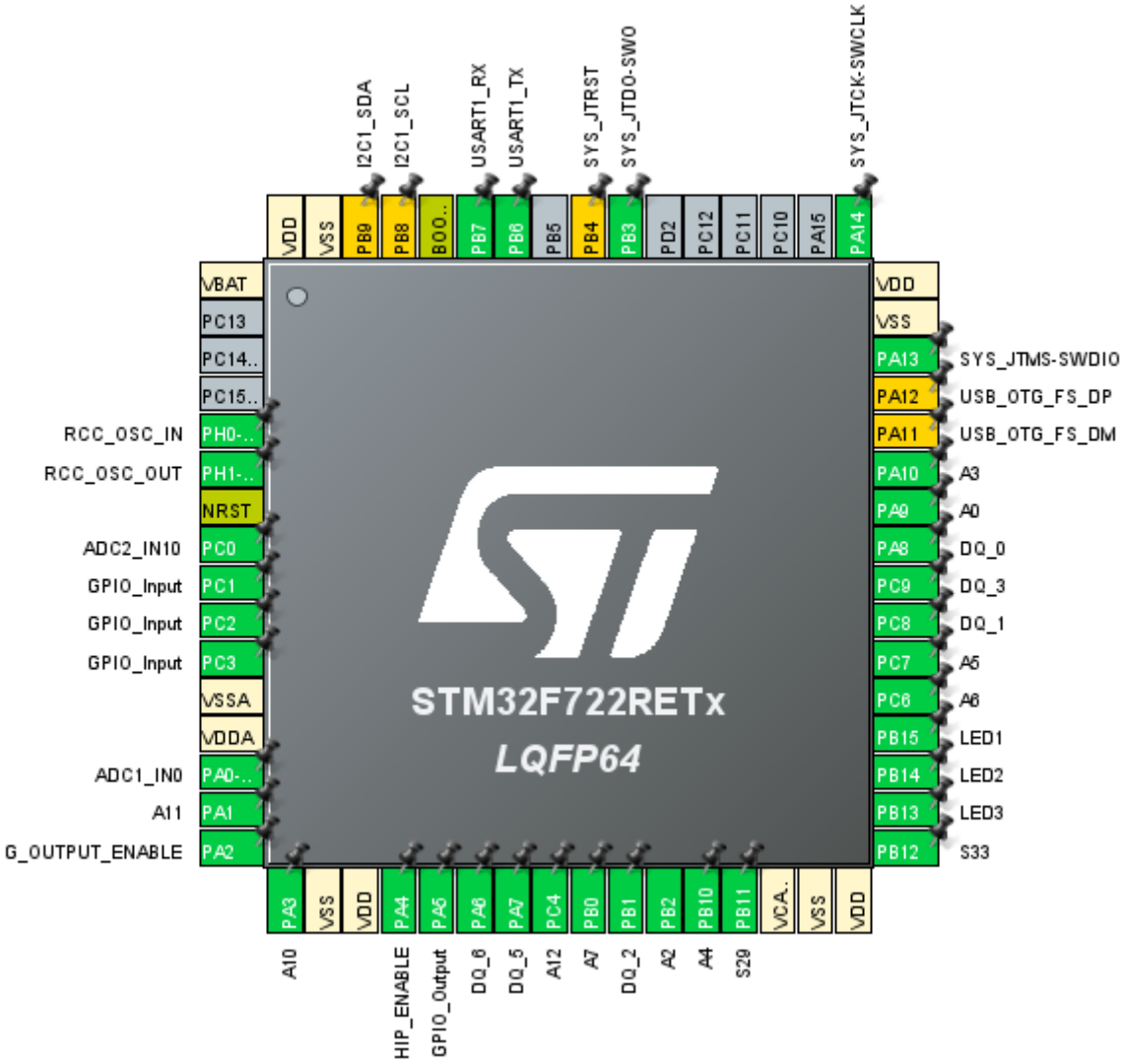
3.6 Indikační LED

Kvůli sledovaní vnitřního stavu firmwaru byly nakonfigurovány piny PB15, PB14 jako výstupní. Na těchto pinech se nachází svítivé diody. Jedna indikuje běh mikroprocesoru druhá odeslání dat.

3.7 Konfigurace hodinových signálů a pinů

Jelikož prototypový přípravek disponuje krystalickým oscilátorem o frekvenci 16Mhz, je vstupní frekvence mikrokontroleru nakonfigurována na stejnou hodnotu. Hodinový signál z oscilátoru prochází HSI registrem. Hlavní frekvence mikrokontroleru může díky oscilátoru mít maximální hodnotu a to 216MHz. Ostatní hodnoty děliček a násobiček frekvence jsou díky vývojovému prostředí nastaveny automaticky.

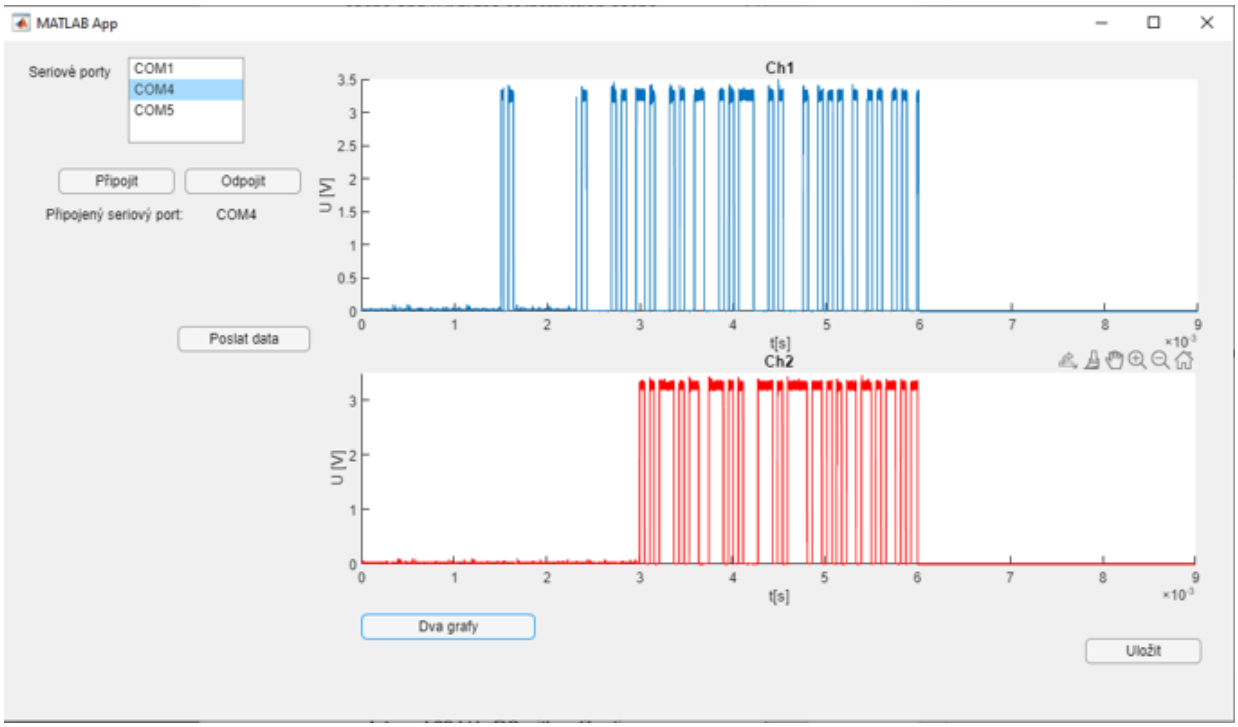
Před programováním firmwaru byly nastaveny vstupní, výstupní a ostatní funkce pinů procesoru. Piny PC0 a PA0 byly nakonfigurovány jako vstupy pro AD převodníky. PH0 a PH1 byly na nasteveny pro vstup a výstup oscilátoru. Pro odesílaní dat přes UART byl nakonfigurován port PB6 a k přijímání dat port PB7.



Obrázek 3.3: Pinout

3.8 Návrh uživatelského rozhraní aplikace

Uživatelské rozhraní se skládá z listboxu, ve kterém si uživatel může zvolit používaný COM port počítače. Je potřeba zvolit správný, například pomocí správce zařízení. K sériovému portu se lze připojit pomocí tlačítka Připojit a přerušit komunikaci lze tlačítkem Odpojit. Pro příjem dat je potřeba zmáčknout tlačítko Poslat data. Po provedení těchto kroků se získaná data zobrazí na osách grafu. Signály lze poté rozdělit na dva průběhy pro lepší přehlednost dat. Výsledné grafy s daty lze uložit tlačítkem s názvem Uložit.

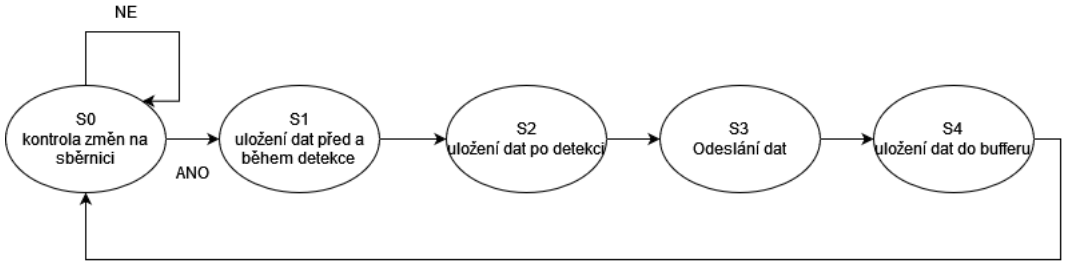


Obrázek 3.4: Uživatelské rozhraní aplikace

4 Realizace firmwaru a aplikace v Matlabu

Po konfiguraci periferií a vlastností mikrokontroleru a vytvoření grafického uživatelského rozhraní aplikace v App Designeru. Byl vytvořen firmware a model aplikace.

Firmware funguje jako stavový automat o pěti stavech. každý AD převodník má svůj stavový automat. Stavové automaty se prolínají ve stavech 0 při detekci změny na sériové sběrnici a ve stavu 3, kde probíhá posílání dat ven z mikrokontroleru.



Obrázek 4.1: Stavový automat

4.1 Inicializace

Po provedení inicializace byly vytvořeny dva adresové prostory pro oba analogově-digitální převodníky o délce 12000 bytů. Dále jsou vytvořeny čtyři buffery pro ukládání dat z převodníku, které se pak spojí do jednoho velkého pole dat k odeslání do nadřazeného PC. Potom byly potřeba dvě proměnné, které slouží k určení stavu stavového automatu.

4.2 Hlavní programová smyčka

V hlavní programové smyčce probíhá odesílaní balíku dat do nadřazeného počítače. Tento proces probíhá ve stavovém automatu ve stavu 3. Je potřeba, aby se do stavu 3 dostali obě stavové proměnné .

Dále zde probíhá indikace pomocí svítivých diod. LED1 indikuje funkčnost mikrokontroléru a funkci hlavní programové smyčky.K indikaci poslání dat slouží LED2, která se rozsvítí, když se je zahajeno odesílaní dat a zhasne po dokončení odeslání dat.

Listing 4.1: kód hlavní programové smyčky

```
while (1)
{
  HAL_GPIO_TogglePin(LED1_GPIO_Port, LED1_Pin);
        if(st2==3||st==3)
                {
                st2=202;
                st=102;
                    HAL_UART_Transmit_DMA(&huart1, (uint8_t *)
                       tr_buff,size);//1000 * 2 (uint16 - 2byte)
                        HAL_GPIO_TogglePin(LED2_GPIO_Port,
                           LED2_Pin);
                }
       HAL_GPIO_TogglePin(LED1_GPIO_Port, LED1_Pin);
}
```

4.3 Přerušení ConvHalfCpltCallback

Toto přerušení se vyvolá pokaždé, když je buffer analogově-digitálních převodníků z půlky naplněn.

Nejdříve je nutné zkontrolovat který AD převodník je zdrojem vyvolání přerušení. Následně dochází ke kontorole stavové proměnné k určení aktuálního stavu.

Při nulové hodnotě stavové proměnné probíhá kontrola změny z klidového stavu na zahájení komunikace. Pokud je podmínka splněna, hodnota stavových proměnných se změní na 1. Na konci se nahraje obsah půlky bufferu převodníku do pomocného bufferu.

Listing 4.2: Kód stavu 0

```
if(st==0)
        {
        for (int i=1;i<ADC1_BUFF_LENGHT/2;i++)
        {
      hr=adc_buff[i]-adc_buff[i-1];
        if(hr>EVal)
        {
                st=1;
                 st2=1;
      }
       }
 for (int i=0;i<ADC1_BUFF_LENGHT/2;i++)
                      half_buff[i]= adc_buff[i];
       }
```

Po stavu 0 následuje stav 1. V tomto stavu probíhá plnění pole dat pro odeslání. Na každý sudý index pole se nahrávají data z převodníku ADC1 a na každý lichý index se nahrávají hodnoty z ADC2 převodníku. Na začátek pole pro odesílání se nejdříve uloží data z pomocného bufferu. Jedná se o data, která byla nasbíraná

před detekcí změny. Do druhé třetiny se ukládají aktuální data v bufferu analogově-digitálního převodníku. Po konci nahrávaní se stavová proměnná nastaví na hodnotu 2.

Listing 4.3: Kód stavu 1

```
if(st==1)
{
            int j=0;
          for (int i=0;i<ADC1_BUFF_LENGHT/2;i++)
             {
                 tr_buff[j]= half_buff[i];
                 j=j+2;
             }
               j=0;
        for (int i=0;i<ADC1_BUFF_LENGHT/2;i++)
        {
        tr_buff[j+ADC1_BUFF_LENGHT/2]= adc_buff[i];
        }
                st=2;
}
```

Během stavu 2 probíhá nahrávání dat do poslední třetiny bufferu pro odesílání z aktuálně načtených dat. Následně se do stavové proměnné zapíše hodnota 3.

Listing 4.4: kód stavu 2

```
if(st==2)
{
        int j =0;
        for (int i=0;i<ADC1_BUFF_LENGHT/2;i++)
        {
                tr_buff[j+ADC1_BUFF_LENGHT] = adc_buff[i];
                j=j+2;
    }
        st=3;
}
```

4.4 Přerušení ConvCpltCallback

Toto přerušení se vyvolá vždy když je plná druhá půlka bufferu pro analogově-digitální převodníky. Díky nastavení kruhového režimu může převodník plnit první polovinu, když probíhá obsluha tohoto přerušení. Je zde podobný kód jako v ConvHalfCpltCallback přerušení. Jediným rozdílem je začátek indexovaní bufferu dat z analogově-digitálního převodníku, které musí začínat na hodnotě poloviny velikosti bufferu.

4.5 Přerušení UARTTxCpltCallbac

Tato funkce se spustí vždy, když se dokončí přenos dat. Po odeslání celého pole dat se do stavových proměnných zapíše hodnota 4. V tomto stavu se nahrají hodnoty z AD převodníku do pomocného bufferu a pokračuje se opět stavem 0.

4.6 Aplikace v Matlabu

Aplikace vytvořené v App designeru fungují na principu volání callback funkcí. Při spuštění aplikace se spustí funkce tartupFcn. Ve které se do Labelu nahraje text o nepřipojení sériového portu. Dále se do Listboxu nahrají všechny dostupný COM porty. Vytvoří se osa y u grafických os a zároveň do nich nahrají data složená ze samých jedniček, aby grafy nebyly prázdné při spuštění aplikace.

Listing 4.5: kód starup funkce

```
 function startupFcn(app)
app.SeriovportLabel.Text="neni pripojeno";
app.SeriovportyListBox.Items=serialportlist('available')
    ;
   period=1/1000000; %1Mhz;

 for t = 1:1:9000
     app.Yaxis(t)= period*t;

 end

app.PlotLine = plot(app.UIAxes,0:2000,ones(1,2001));
app.PlotLine2 = plot(app.UIAxes2,0:2000,ones(1,2001));

end
```

Dále byla naprogramována funkčnost tlačítek Připojit a Odpojit. Při připojení se spouští funkce stisknutí tlačítka, při které se vytvoří spojení se sériovým portem. Při vytváření je třeba zadat správnou hodnotu přenosové rychlosti. Odpojení poté zruší spojení mezi COM portem a aplikací.

Listing 4.6: kód funkce tlačítek

```
function ConectCom(app, event)
     app.Port = app.SeriovportyListBox.Value;
     delete(app.S);
     app.SeriovportLabel.Text=app.Port;
     app.S =serialport(app.Port,115200);
end
function DisconectCom(app, event)
    delete(app.S);
     app.SeriovportLabel.Text="neni pripojeno";
end
```

Funkce stisknutí tlačítka Poslat data vytvoří callback funkci, která se spustí po naplnění bufferu sériového portu určítým počtem bytů.

Listing 4.7: kód funkce tlačítka Poslat data

```
function SendData(app, event)
      configureCallback(app.S,"byte",36000,@app.ReadData);
  end
```

Funkce obsluhy přerušení po naplnění upraví data do správné formy. Jelikož z mikrokontroleru jsou vysílaná data přímo z AD převodníku, který má na svém vstupu odporový dělič, který hodnotu vstupního napětí rozdělí na půl, je potřeba data upravit na hodnotu napětí podle vzorce. **??**.

$$U = x \cdot \frac{3.3}{4096} \cdot 2 \tag{4.1}$$

Po úpravě dat je nutné data rozdělit na hodnoty z ADC1 a ADC2. Jelikož data z ADC1 jsou na sudých indexech a data z druhého AD převodníku na lichých stačí pouze v cyklu data zapsat do svých matic podle indexu.

Následně jsou data vykresleny v horních osách. Po stisknutí tlačítka Dva Grafy se data z prvního analogově-digitálního převodníku vykreslí do horních os a z druhého do spodních os. Osu x tvoří hodnoty napětí ve voltech a osou y je čas v sekundách, který je znám díky zvolené vzorkovací frekvenci 1 MHZ.

Listing 4.8: kód obsluhy přerušení serivého portu

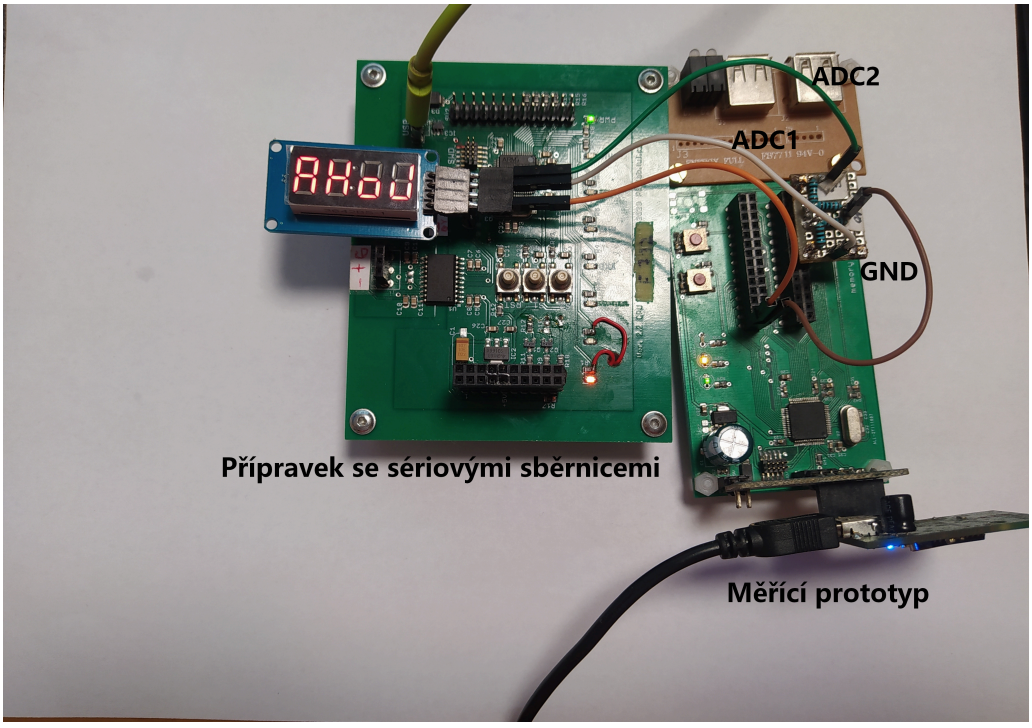
```
function ReadData (app,src, ~)
       r = read(src,18000,"uint16");
       app.Data=r*(3.3/4096)*2;

        j=1;
      for i=1:2:length(app.Data)
        if (i==1)
            app.Datachaneltwo(j)=app.Data(1);
         else
              app.Datachanelone(j)=app.Data(i-1);
             app.Datachaneltwo(j)=app.Data(i);
        end
        j=j+1;
      end

      app.Plotline=plot(app.UIAxes,app.Yaxis,app.Datachanelone,
         app.Yaxis,app.Datachaneltwo);
      app.UIAxes.Legend;

    end
end
```

Když jsou data zobrazené je možné je uložit jako graf a soubor dat s obsahem hodnot napětí na obou kanálech a časové osy. Soubor může sloužit pro další zpracování naměřených dat. Soubor se uloží do pracovní složky tedy do té, v které se aplikace nachází. Soubory mají název podle aktuálního data a času.

5 Ukazkové úlohy

V ukázkových úlohách byly stejné úlohy měřeny za pomocí osciloskopu a pomocí přípravku. Pro simulaci sběrnic mi byl poskytnut přípravek, který disponuje sběrnicemi UART, RS485 a I2C. Simulátor sběrnic mi byl poskytnut už s naprogramovaným firmwarem komunikací přes sériová rozhraní. Přes všechny sběrnice je posílán datagram obsahující slovo Ahoj a následují čísla od jedničky do desítky.

Analyzátor se na přípravek připojuje pomocí hřebenů a propojovacích kabelů, který mají na obou koncích samcový Dupont konektor. Po stisknutí tlačítka S2 přípravek komunikuje s displejem přes I2C a po stisknutí tlačítka S1 vysílá data skrze RS485 a UART. Spojení zemnících vodičů probíhá také přes kabely s Dupont konektorem.



Obrázek 5.1: Zapojení při měření sběrnice I2C

Hodnoty rychlostí přenosu data byly zvoleny u sběrnice UART 115200 bit/s. u RS485 9600 bit/s a u I2C 100Kb/s. Rozmezí bylo zvoleno pro vyzkoušení omezených vlastností prototypu.

5.1 Měření Sběrnic

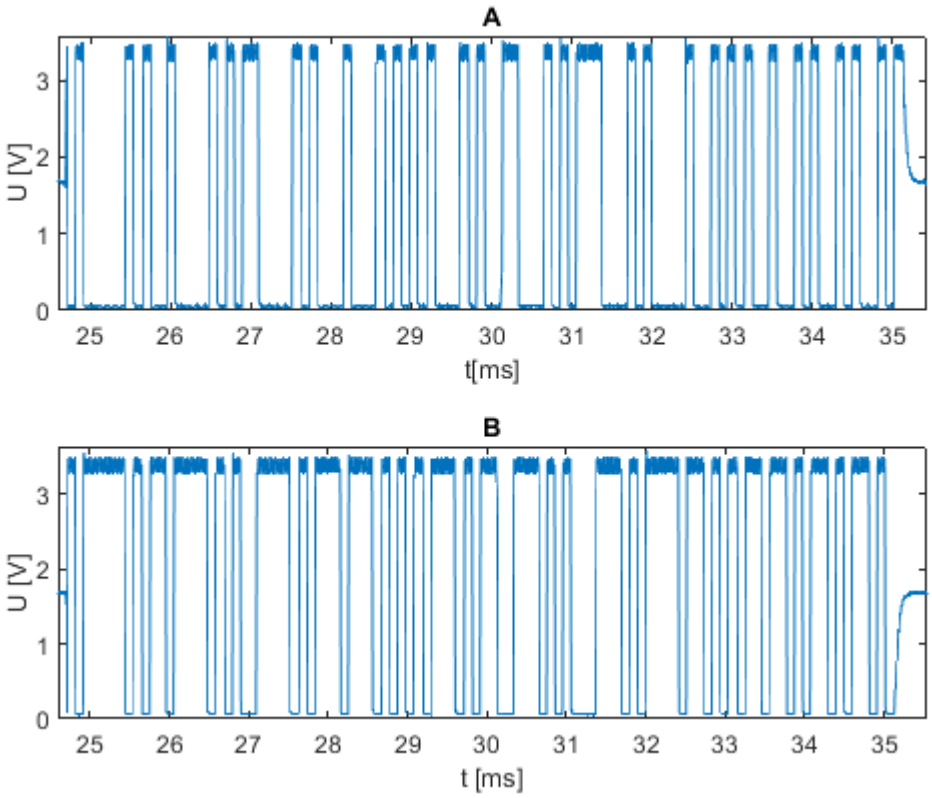
Měření proběhlo osciloskopem na pracovišti vedoucího. Přípravek pro simulování sběrnic byl napájen z USB portu osciloskopu. Změřená data byla uložena do souboru CSV a následně načtena a zobrazena ve skriptu v Matlabu.

Měření pomocí přípravku proběhlo doma, kde byl přípravek napájen s USB portu počítače. Výsledky se tedy mohou mírně lišit. Následně byly data zpracována pro lepší zobrazení v Matlabu.

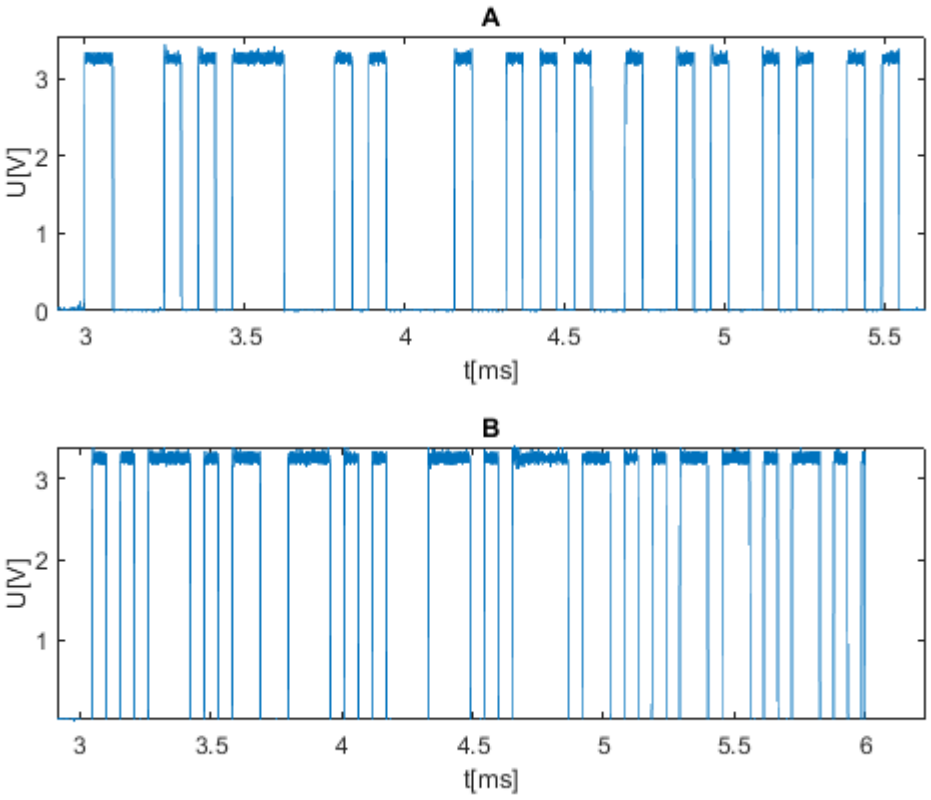
U měření RS485 můžeme vidět, že přípravek změřil jiný klidový stav než osciloskop. Kdy na osciloskopu se drží nad 1,5V a u přípravku se tato hodnota pohybuje kolem 0V. Můžeme též vidět vyšší vzorkovací frekvenci přípravku.

Při měření dat na sběrnici UART byla na osciloskopu desetkrát větší vzorkovací frekvence. Můžeme tedy vidět, že osciloskop poskytuje více podrobná data.

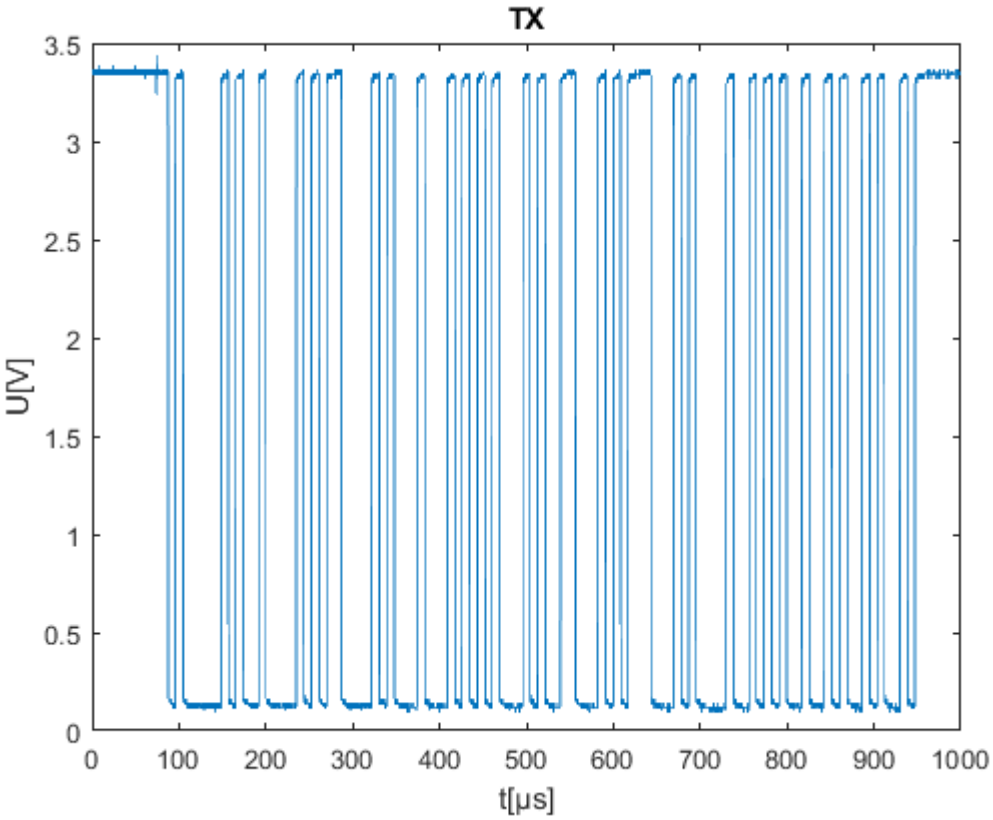
U sbíraní dat ze sběrnice I2C je možné vyčíst, že u přenosu po vodiči SDA jsou data dosti podobná. U přenosu po SCL tedy hodinového signálu sběrnice můžeme vidět strop přípravku, kdy už není dostatečná vzorkovací frekvence a dochází k nedostatku dat pro vyhodnocení signálu.



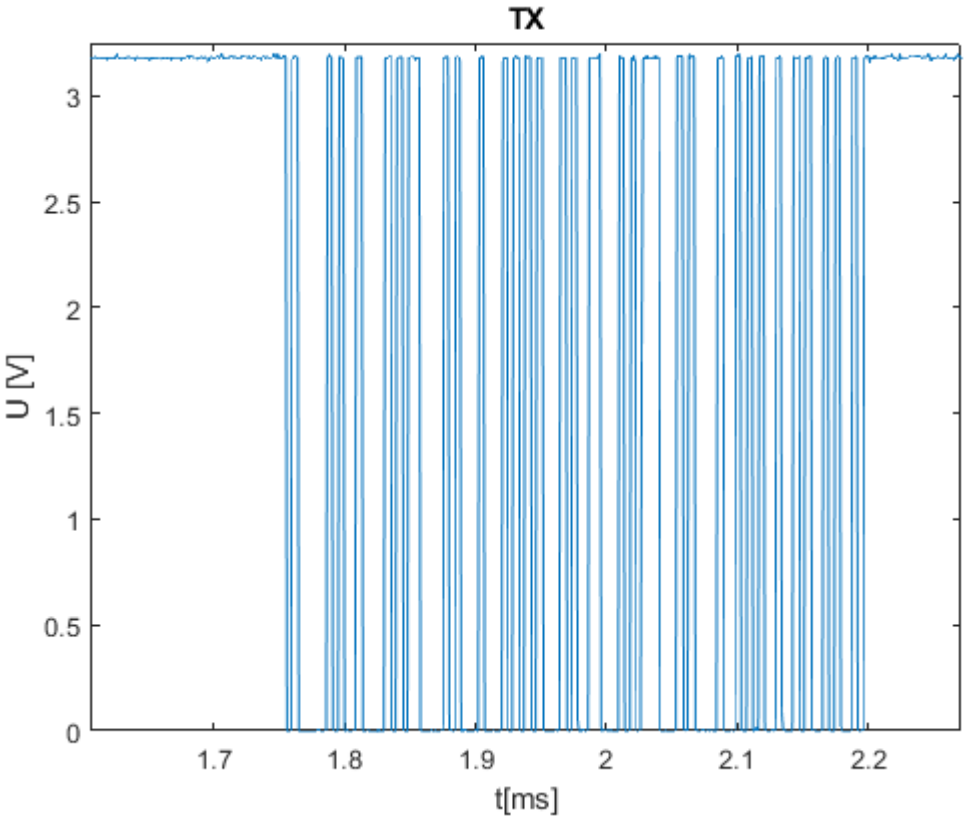
Obrázek 5.2: Časový průběh RS485 na oscyloskopu



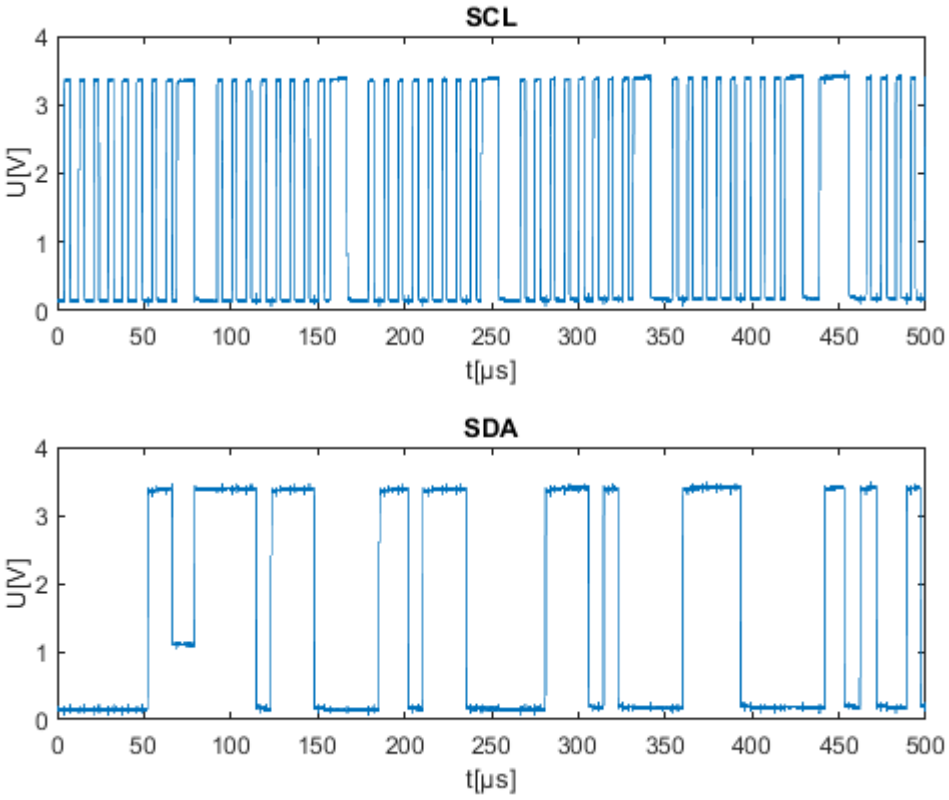
Obrázek 5.3: Časový průběh RS485 na přípravku



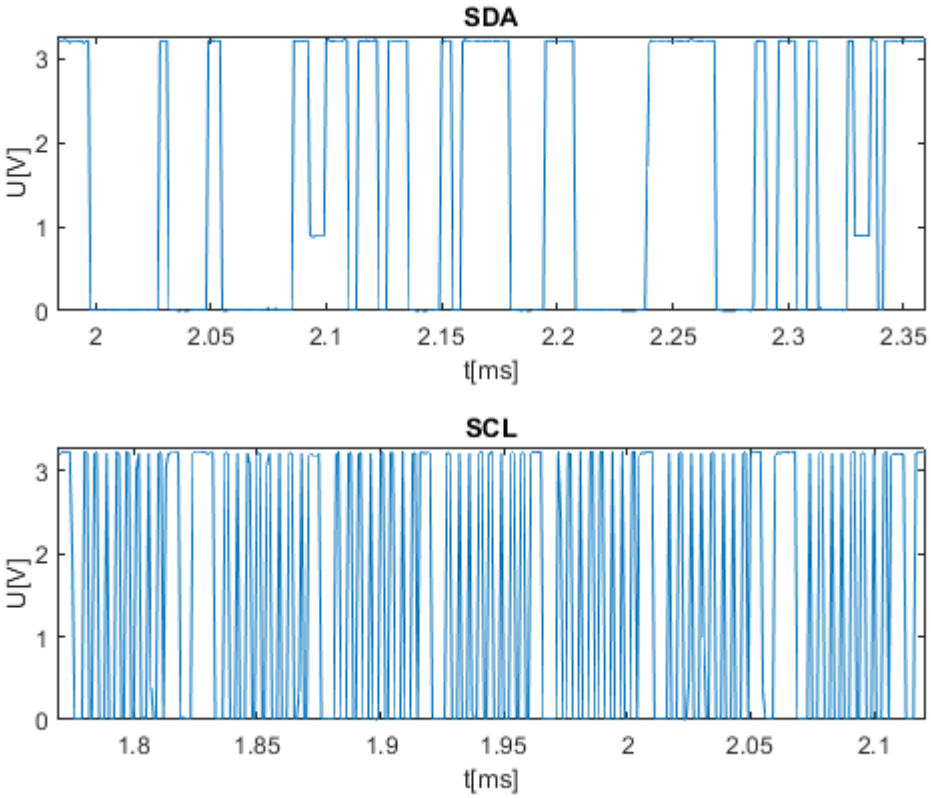
Obrázek 5.4: Časový průběh UART na oscyloskopu



Obrázek 5.5: Časový průběh UART na přípravku



Obrázek 5.6: Časový průběh I2C na oscyloskopu



Obrázek 5.7: Časový průběh I2C na přípravku

6 Závěr

V této bakalářské práci byl vytvořen prototyp analyzátoru sériových sběrnic za pomocí vývojového kitu. Na pracovišti byl vybrán ze dvou vývojových přípravků, přípravek memtest. Druhý přípravek byl následně poskytnut vedoucím pro ukázkové úlohy pro ověření funkčnosti. Přípravek byl vybrán na základě jeho výhodného rozmístění pinů pro AD převodníky.

V dalším kroku byla vypracována rešerše možností monitorování vybraných sériových sběrnic. Následně bylo potřeba zjistit vlastnosti sériových rozhraní jejich logické úrovně, počet vodičů atd. Z výsledků této rešerše byl vypracován návrh řešení pro diagnostiku toku dat.

Pro čtení dat z logickými úrovněmi vyššími jak 3,3V nebylo možné připojit sériové sběrnice přímo na hřeben mikrokontroleru. Kvůli pětivoltové logice byla vytvořena destička z pájivého pole, na které byly napájeny dva děliče napětí. K tomuto děliči byly připájeny dva vodiče na připojení k sériové sběrnici a jeden další vodič pro propojení země přípravku se zemnícím vodičem sledovaného sériového rozhraní.

Při naprogramování firmwaru byla nejdříve zvolena cesta dvou kanálů na jednom analogově-digitálním převodníku. Během ověřovaní funkčnosti na jednom kanále se zdálo, že vše funguje. Bohužel se na konec ukázalo že při vyšších frekvencích druhý kanál funguje s nižší vzorkovací frekvencí. Za pomocí knihovny HAL nebylo možné AD převodník nastavit tak, aby oba kanály vykonaly konverzi dat za stejný počet cyklů mikrokontroleru.

Bylo tedy nutné zvolit způsob se dvěma na sobě nezávislými AD převodníky. Kvůli tomuto rozhodnutí, bylo zapotřebí pozměnit kód firmwaru mikrokontroleru.

Po naprogramování firmwaru bylo potřebné přejít k vývoji zobrazovací aplikace pro nadřazený počítač. V prostředí App Designeru v Matlabu bylo vytvořeno uživatelské rozhraní pro připojení k sériovému portu a zobrazení získaných dat na časovou osu. Dále byla přidána možnost uložení dat do souboru s příponou .mat a uložení časové závislosti jako obrázku grafu.

Vše bylo následně úspěšně odzkoušeno na ukázkových úlohách se sériovými sběrnicemi I2C, UART a RS485. Nejdříve byly data změřena na osciloskopu u vedoucího práce a následně změřena i pomocí vytvořeného přípravku. Data se lišší hlavně kvůli omezené vzorkovací frekvenci přípravku 1 MHz. Osciloskop byl nastaven až na 10 MHz vzorkovací frekvenci.

Dalším postupem by mohlo být přidání dalšího kanálu, a tedy použití jiného vývojového kitu a zároveň přidání dalších funkcí do uživatelské aplikace v Matlabu.

7 Literatura

[1] www.st.com. ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32 [online]. [cit. 2021-5-30]. Dostupné z: https://www.st.com/en/development-tools/st-link-v2.html

[2] STM32Cube [online]. 2023 [cit. 2023-05-22]. Dostupné z: https://www.st.com/en/development-tools/st-link-v2.html

[3] STM32 HAL Library [online]. 2023 [cit. 2023-05-22]. Dostupné z: https://deepbluembedded.com/stm32-hal-library-tutorial-examples

[4] App Designer [online]. 2023 [cit. 2023-05-22]. Dostupné z: https://www.mathworks.com/products/matlab/app-designer.html

[5] Understanding UART [online]. [cit. 2023-05-22]. Dostupné z: https://www.rohde-schwarz.com/us/products/test-and-measurement/essentials-test-equipment/digital-oscilloscopes/understanding-uart254524.html

[6] UART Explained [online]. 2023 [cit. 2023-05-22]. Dostupné z: https://developer.electricimp.com/resources/uart

[7] Přispěvatelé Wikipedie, RS-485 [online], Wikipedie: Otevřená encyklopedie, c2023, Datum poslední revize 9. 03. 2023, 05:36 UTC, [citováno 22. 05. 2023] <https://cs.wikipedia.org/w/index.php?title=RS-485oldid=22528430>

[8] Přispěvatelé Wikipedie, RS-485 [online], Wikipedie: Otevřená encyklopedie, c2023, Datum poslední revize 9. 03. 2023, 05:36 UTC, [citováno 22. 05. 2023] <https://cs.wikipedia.org/w/index.php?title=RS-485oldid=22528430>

[9] Dekodéry sériových sběrnic I2C, SPI a UART-RS232 pro osciloskopy Teledyne LeCroy [online]. 2023 [cit. 2023-05-22]. Dostupné z: https://www.blue-panther.cz/dekodery-seriovych-sbernic-i2c-spi-a-uart-rs232-pro-osciloskopy-teledyne-lecroy

[10] Logický analyzátor s mikroradicem. Praha, 2020. Bakalářsjá práce. ČVUT. Vedoucí práce Doc. Ing. Jan Fischer, CSc.,.