ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA ELEKTROTECHNICKÁ

KATEDRA ŘÍDICÍ TECHNIKY

DIPLOMOVÁ PRÁCE

VIZUALIZACE DAT Z FREKVENČNÍHO MĚNIČE

Autor: Bc. Petr Adámek

Vedoucí: Ing. Peter Matisko

Praha, 2013

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Petr Adámek**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Vizualizace dat z frekvenčního měniče**

Pokyny pro vypracování:

1. Naprogramujte vizualizační rozhraní pro frekvenční měnič značky Powtran.
2. Program napište v jazyku C# nebo v Javě. Program bude obsahovat komunikační příkazy, ukládání/načítání dat, ukládání/načítání nastavení měniče.
3. Dále bude program zahrnovat zobrazovací modul typu osciloskop pro zobrazení průběhů signálů, které měnič poskytuje.
4. Otestujte program společně s měničem.

Seznam odborné literatury:

Voženílek P., Janoušek J.: Základy silnoproudé elektrotechniky
Glynn J. et al.: C# Programujeme profesionálně
Nash T.: C# 2010 - Rychlý průvodce novinkami a nejlepšími postupy
Schildt H.: Java 7 - Výukový kurz
Manuály k frekvenčnímu měniči značky Powtran

Vedoucí: Ing. Peter Matisko

Platnost zadání: do konce letního semestru 2013/2014

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 10. 10. 2012

Poděkování

Na tomto místě bych rád poděkoval Ing. Peteru Matiskovi za vedení mé práce, ochotu a čas při konzultacích, poskytování cenných rad a materiálů. Dále bych rád poděkoval své rodině za podporu během studia a v neposlední řadě svým přátelům.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, dne 9. 5. 2013 ..... Adámek .....

podpis

Abstrakt

Cílem této práce je vytvoření aplikace pro vizualizaci dat získaných z frekvenčního měniče pomocí sériové linky. Komunikace probíhá mezi frekvenčním měničem značky POWTRAN (typ PI8100) a nadřazenou jednotou (počítačem). Ke komunikaci je použita sběrnice RS-485 a FTDI převodník. K výměně zpráv mezi počítačem a měničem je použita datová sada k danému měniči a komunikační protokol MODBUS. Aplikace je grafické rozhraní vytvořené v Javě, které umožňuje uživateli komunikaci s frekvenčním měničem a vizualizaci měřených dat. Program nabízí uživateli i správu dat, uložení a načtení naměřených dat a uložení či načtení nastavených parametrů. Pro správu dat jsou vytvořené struktury, které umožňují aplikaci snadnější a rychlejší přístup k datům.

Klíčová slova

Frekvenční měnič, asynchronní motor, vizualizace, MODBUS, Java, XML, CSV, XLS.

Abstract

This thesis deals with creation of an application for visualization of the data from a frequency inverter. Communication is carried out between the frequency inverter (type Powtran PI8100 - slave) and a computer (master). The communication is used for the bus RS-485 and FTDI chip. The messages between devices and a computer are using a data set of inverters and the communication protocol MODBUS. The application is a graphical interface developed in Java that allows the user to communicate with the device and visualize the data. The application allows user to manage save and load the data, measured data and set the parameters of device.

Keywords

Frequency inverter, asynchronous motor, visualization, MODBUS, Java, XML, CSV, XLS.

Obsah

Obsah ..... **V**
Seznam obrázků ..... **VII**
Seznam tabulek ..... **VIII**
Seznam zkratek ..... **IX**
1 Úvod **1**
1.1 Motivace 1
1.2 Vizualizace dat 2
1.3 Struktura diplomové práce 2
2 Frekvenční měniče **4**
2.1 Základní popis frekvenčních měničů 4
2.2 Princip frekvenčního měniče 4
2.3 Použití frekvenčních měničů 6
2.4 Řízení frekvenčních měničů 9
2.5 Frekvenční měnič Powtran PI8100 11
3 Komunikační protokol MODBUS **13**
3.1 Obecný popis 13
3.2 Popis protokolu 13
3.3 Specifikace protokolu frekvenčním měničem Powtran PI8100 15
4 Správa dat **21**
4.1 Obecný popis 21
4.2 Jazyková modifikace 22
4.3 Načtení dat měniče 23
4.4 Správa projektu 25
4.5 Správa naměřených dat 25
5 Aplikace **27**
5.1 Výběr programovacího jazyka 27
5.2 Základní popis aplikace 27
5.3 Panel s menu 30
5.4 Pracovní plocha 36
5.5 Komunikace 40
5.6 Aktivní hlídání grafických prvků 40
5.7 Měřící vlákno 42
6 Závěr **44**
6.1 Funkčnost aplikace 44

6.2 Testování aplikace 45
6.3 Možnosti rozšíření aplikace 46
Literatura 47
Obsah přiloženého CD 49

Seznam obrázků

Obr. 2-1 Základní schéma třífázového frekvenčního měniče 5
Obr. 2-2 Frekvenční měnič Powtran PI8100 **[8]** 11
Obr. 2-3 Blokové schéma frekvenčního měniče PI8100 12
Obr. 3-1 Základní tvar rámce 13
Obr. 3-2 Komunikace master-slave s doručenou odpovědí 14
Obr. 3-3 Komunikace master-slave se ztrátou dat 15
Obr. 3-4 Ukázka délky rámce pro čtení jednoho registru (0x03) 17
Obr. 3-5 Ukázka délky rámce pro zápis jedné funkce (0x06) 18
Obr. 3-6 Ukázka délky rámce pro zápis do více registrů (0x10) 18
Obr. 3-7 Ukázka délky rámce pro zápis stavu přepínače (0x05) 18
Obr. 3-8 Ukázka délky rámce pro zápornou odpověď 19
Obr. 4-1 Výběr jazykové modifikace aplikace 22
Obr. 4-2 Stromová struktura XML souboru pro PI8100 23
Obr. 4-3 Stromová struktura XML souboru projektu 25
Obr. 5-1 Hlavní okno aplikace 28
Obr. 5-2 Zjednodušené schéma běhu aplikace 29
Obr. 5-3 Průvodce nastavením nového projektu 30
Obr. 5-4 Okno pro výběr průběhů 33
Obr. 5-5 Okno pro úpravu časové osy 34
Obr. 5-6 Průvodce změnou nastavení komunikace 35
Obr. 5-7 Ukázka okna Pomoc 35
Obr. 5-8 Rozdělení hlavního okna 36
Obr. 5-9 Ukázka grafu v offline režimu 38
Obr. 5-10 Ukázka informací o zařízení v offline režimu 38
Obr. 5-11 Ukázka zachování poměrů panelů v okně 41
Obr. 5-12 Zjednodušený vývojový diagram pro vlákno hlídající grafické prvky 41
Obr. 5-13 Zjednodušený vývojový diagram pro měřící vlákno 43
Obr. 6-1 Ukázka grafu vytvořeného v Excelu 44
Obr. 6-2 Ukázka grafu uloženého ve formátu PNG 45

Seznam tabulek

Tab. 3-1Specifikace komunikace pro RS-485 16
Tab. 3-2 Specifikace kódu funkce 16
Tab. 3-3 Specifikace kódu funkce pro zápornou odpověď 17
Tab. 3-4Základní skupiny funkcí 20

Seznam zkratek

EMC	Elektromagnetická kompatibilita (ElectroMagnetic Compatibility)
PLC	Programovatelný logický automat (Programmable Logic Controller)
PDU	Zpráva na úrovni protokolu (Protocol Data Unit)
ADU	Zpráva na aplikační úrovni (Application Data Unit)
CRC	Cyklický redundantní součet (Cyclic Redundancy Check)
CSV	Souborový formát (Comma-Separated Values)
XML	Značkovací jazyk (Extensuble Markup Language)
XLS	Souborový formát – Excel
USB	Sériová sběrnice (Universal Seriál Bus)
FTDI	Převodník mezi sběrnicemi (Future Technology Devices International)

1 Úvod

1.1 Motivace

V dnešní době je nejrozšířenějším typem elektrického pohonu asynchronní indukční motor. Hlavní výhodou asynchronních strojů je jejich konstrukční jednoduchost, vysoká spolehlivost a vyžadují pouze minimální údržbu. Asynchronní indukční motory lze rozdělit podle způsobu provedení rotoru na motory s kotvou nakrátko a motory s kroužkovou kotvou. Asynchronní motory nacházejí využití k pohonům mnoha zařízení s různým charakterem provozu například čerpadla, ventilátory a kompresory, pasové dopravníky, výtahy, jeřáby či obráběcí stroje atd. Každý charakter zátěže má různé nároky na technické parametry motoru, výkon, moment a jejich řízení.

U motorů požadujeme určitou rychlost otáček hřídele, ale kromě tohoto parametru můžeme požadovat i moment. Pro změnu rychlosti, při použití střídavého motoru, je nejefektivnější způsob změna frekvence napájecího napětí motoru a zároveň i regulace velikosti napájecího napětí. Tato regulace pomáhá snížit velikost odběru při rozbíhání pohonu a zajištění konstantního momentu. Případně může docházet i k jiné metodě regulace odběru při rozběhu motoru a zachování velikosti momentu. Změnu kmitočtu a velikosti napájecího napětí pohonu umožní frekvenční měnič. Zároveň může nabídnout ochrany (přepěťová a proudová ochrana) pro samotné elektrické stroje, které jsou k němu připojeny. Ochrany jsou buď automaticky nastaveny po detekování parametrů motoru pomocí měniče, nebo se parametry vloží do samotného měniče.

Vlastní měnič může mít ovládací panel, který slouží k zobrazení dat a k nastavení parametrů. Tyto parametry se mohou nastavit i pomocí komunikačních rozhraní, která měnič nabízí (připojení ke sběrnici: Ethernet, sériová linka, Profibus atd.). Použití tohoto rozhraní k nastavení parametrů měniče je snadnější a rychlejší, než když se nastavují pomocí ovládacího panelu. Pokud budeme chtít víc měničů se stejným nastavením, je použití rozhraní nutností. Připojení ke sběrnici nabízí i další možnosti než jen nastavení. Umožňuje monitorování dat, která lze vizualizovat, zpracovat a archivovat.

Software, který vznikl při vypracování této diplomové práce, by měl ulehčit nastavování frekvenčního měniče a sběr dat. Nastavení frekvenčního měniče umožní aplikace dvěma základními postupy, nabízí rychlé ovládání pro směr otáčení, spuštění, zastavení a nastavení frekvence. Cyklicky monitoruje základní měřená data frekvenčním měničem a poskytuje je uživateli v grafickém a textovém zpracování. Získaná data lze uložit v několika formách pro pohodlnější externí zpracování. Mimo jiné je možné uložit projekt do vlastního souboru, aby bylo možné se k němu vrátit nebo přenést na jiné zařízení. V tomto souboru jsou uloženy všechny parametry frekvenčního měniče.

1.2 Vizualizace dat

Vizualizace dat a procesů je v dnešní době nedílnou součástí většiny aplikací v různých odvětvích průmyslu. Je jednou ze základních a nejsnadnějších možností monitorování procesů a poskytování zpětné vazby uživateli, který může reagovat na vzniklé situace. Vizualizace dat může být realizována různě. Například textové výpisy hodnot a varování, nebo grafické prvky (grafy, tlačítka, grafické znázornění procesu a mnoho dalších). Ve složitějších aplikacích vznikají operátorská pracoviště, zodpovědná za chod jednotlivých procesů. Využití vizualizace dat je důležité i pro experimenty, které vedou k vývoji a testování.

Vizualizace a ukládání dat je důležité nejen z krátkodobého hlediska, ale i z pohledu delší doby. Starší data nám slouží při zpětné analýze, jako je například zjištění příčiny vzniku problému, porovnání kvality regulace při různých nastaveních nebo statistického vyhodnocení aj.

1.3 Struktura diplomové práce

Diplomová práce je rozdělena do několika kapitol, které postupně vysvětlují hardwarové a softwarové části diplomové práce. Hardwarovou část v diplomové práci představuje frekvenční měnič Powtran PI8100 a asynchronní motor. Frekvenčním měničem a motorem se zabývá převážně kapitola 2. Softwarová část je aplikace určená pro komunikaci s frekvenčním měničem. Kapitola 3 popisuje komunikační protokol,

který je použit pro komunikaci mezi frekvenčním měničem a počítačem. Následující kapitola 4 se snaží přiblížit datové soubory, se kterými aplikace pracuje a používá je pro ukládání a načítání dat nebo jako externí databázi. Kapitola 5 popisuje aplikaci, její grafické rozhraní a práci s ním, a základní myšlenky implementace. Závěrečná kapitola 6 shrnuje výsledek práce, možné rozšíření a další postup pro zavedení aplikace.

2 Frekvenční měniče

2.1 Základní popis frekvenčních měničů

Frekvenční měnič je elektronické zařízení, které mění jeden ze základních parametrů elektrického napětí či proudu (dále již jen napěťové frekvenční měniče), frekvenci. Ke změně kmitočtu dochází buď přímo (přímé měniče - cyklokonvertory) nebo nepřímo (nepřímé měniče – viz kap. 2.2). Kromě změny frekvence může měnit i amplitudu. Způsoby řízení frekvenčního měniče můžeme rozdělit na skalární a vektorové řízení.

Frekvenční měniče se s výhodou používají pro řízení asynchronních strojů s kotvou na krátko, protože umožňují efektivní změnu otáček ve velkém rozsahu při různém zatížení, plynulý rozběh a brždění stroje. Mimo jiné je lze použít i jako náhradu mechanických převodovek.

Velké rozšíření frekvenčních měničů umožnil vývoj polovodičové elektroniky a její cenová dostupnost. Nejdůležitějším parametrem, který ovlivňuje cenu je požadovaný výkon frekvenčního měniče. Poměr cena/výkon se liší podle výrobce, mezi které patří Siemens, ABB, Omron a další. U frekvenčních měničů do 3 kW se cena pohybuje většinou v základním provedení do 5.000 Kč, u měničů do 7,5 kW může cena dosáhnout i 30.000 Kč. Velké měniče používané pro jeřáby, nákladní výtahy apod., mají výkon až 700 kW a cena dosahuje až milionu korun.

2.2 Princip frekvenčního měniče

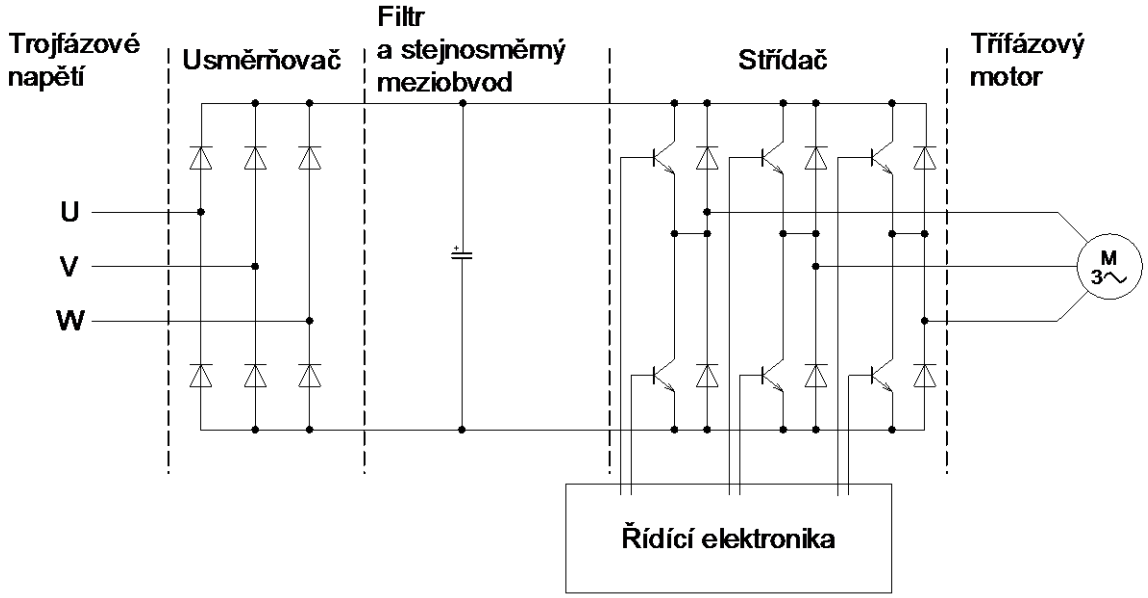
Základní princip frekvenčních měničů je stále stejný, skládají se ze základních částí, které jsou přizpůsobeny k potřebám jednotlivých měničů. Třífázový měnič (Obr. 2-1) můžeme rozdělit na části obvodu, které umožní změnu frekvence (**[1]**, **[2]**):

1. **Usměrňovač** – usměrňuje vstupní napětí. Může být řízený nebo neřízený.

2. **Filtr a stejnosměrný mezi obvod** – vyhlazuje usměrněné napětí a pracuje se stejnosměrným napětím, které poskytuje střídači.

3. **Střídač** – moduluje stejnosměrné napětí a vytváří tím střídavé výstupní napětí. Nejčastěji je tvořen IGBT tranzistory a je ovládán např. pulsně-šířkovou modulací.

4. **Řídící elektronika** – řídí střídač a určuje frekvenci výstupního napětí. Konstrukce řídícího obvodu je závislá na typu měniče (pulsně-šířkově modulovaný měnič, pulsně-amplitudově modulovaný měnič, proudově řízený měnič).



Obr. 2-1 Základní schéma třífázového frekvenčního měniče

Frekvenční měnič umožňuje mimo změny frekvence i změnu počtu fází, např. z jednofázového vedení na vstupu vytvoří třífázové vedení na výstupu. Tuto změnu můžeme snadno ukázat na obrázku (Obr. 2-1), v tomto zapojení stačí změnit vstupní obvod (usměrňovač). Místo třífázového usměrňovače použijeme jednofázový usměrňovač a zbytek obvodu ponecháme beze změny. Obvod střídače má k dispozici stále stejnosměrné napětí a za pomoci řídící elektroniky je schopen ze stejnosměrného napětí modulovat výstupní třífázové střídavé napětí. Jedinou nevýhodou této transformace je nižší výkon, než kdybychom použili na vstupu třífázové napětí. Jednofázové frekvenční měniče jsou dostupné jen pro malé výkony do 3 kW.

Změna amplitudy napětí může být realizována několika způsoby. Buď pomocí transformátorů, kdy se mění amplituda střídavého napětí na vstupu či výstupu

frekvenčního měniče. Tato metoda by byla poměrně drahá a zvyšovala by velikost a hmotnost měniče. Druhým řešením je úprava amplitudy ve stejnosměrném meziobvodu, např. přidání stejnosměrného měniče (typ buck-boost, který umožňuje zvýšení a snížení amplitudy). Přidáním stejnosměrného měniče se nijak výrazně nezvýší cena ani velikost frekvenčního měniče. Amplitudu lze modulovat pomocí střídače a řídící elektroniky, v tomto případě je možné dosáhnout maximální amplitudy, která je rovna stejnosměrnému napětí poskytovanému ze stejnosměrného meziobvodu.

Při návrhu frekvenčního měniče se musí brát v úvahu rušení (EMC – viz **[3]**), neboť na vstupu frekvenčního měniče je diodový usměrňovač, který ze sítě odebírá nesinusový proud (proud je odebírán pouze v případě, že napětí sítě je vyšší než napětí na filtračním kondenzátoru). Tím vzniká při odběru proudu zkreslení vyššími harmonickými. Použitím síťové tlumivky dojde k filtraci proudu, čímž poklesne efektivní hodnota síťového proudu až o 35%. Dalším zdrojem rušení je výstup frekvenčního měniče, který je spojen přímo s motorem. Zdrojem rušení je samotná modulace (pulsně šířková modulace) výstupního napětí, neboť dochází velkému napěťovému rozptylu při frekvencích v řádech kHz. Odběr proudu je při těchto frekvencích vlivem indukčnosti motoru prakticky sinusový, ale pokud je kabel mezi frekvenčním měničem a motorem veden blízko datových linek může docházet k jejich rušení. Frekvenční měnič dále musí splňovat normy pro vysokofrekvenční rušení, které se týkají vyzařování v pásmu 150 kHz až 30 MHz.

2.3 Použití frekvenčních měničů

Frekvenční měniče se s výhodou používají pro řízení rychlosti otáček asynchronních motorů. Otáčky těchto motorů jsou přímo úměrné frekvenci napájecího napětí, synchronní otáčky motoru jsou dány vztahem (1). U asynchronních motorů se otáčky hřídele a točivého magnetického pole liší, tato odchylka se dá vyjádřit pomocí skluzu (2).

$$n_1 = \frac{60 f_1}{p}, \tag{1}$$

kde n_1 jsou synchronní otáčky magnetického pole statoru, f_1 je frekvence napájecího napětí motoru a p je počet pólů motoru.

$$s = \frac{f_2}{f_1} = \frac{n_1 - n}{n_1}, \tag{2}$$

kde s je skluz, f_2 je frekvence otáček rotoru, f_1 je frekvence napájecího napětí, n_1 jsou synchronní otáčky magnetického pole statoru a n jsou otáčky rotoru.

Rychlost otáček se dá ovlivnit několika základními způsoby, které si zjednodušeně popíšeme:

1. **Změna statorového napětí**

   Tento způsob řízení se jeví jako nejsnadnější, protože měnit střídavé napětí je snadné například pomocí regulačního transformátoru. Moment asynchronního motoru je přímo úměrný druhé mocnině fázového napětí. Bohužel, pokud bychom vykreslili zatěžovací charakteristiku n(M), kde n jsou otáčky a M je moment, zjistili bychom, že křivky se nacházejí blízko sebe a řízení otáček je tedy možné pouze v malém rozsahu. Více k této problematice se dočteme v literatuře **[1]**.

2. **Změna počtu pólů statorového vinutí**

   Tato metoda je založena na přepínání počtu pólů pomocí kontaktního přepínače. Nevýhodou této metody je, že řízení otáček není spojité, ale skokové, a zároveň vyžaduje konstrukčně složitější řešení motoru. Což můžeme vidět, když počet pólů dosadíme do rovnice (1). Tento způsob řízení není vhodný do většiny aplikací.

3. **Změna frekvence napájecího napětí**

   Vyjdeme-li opět ze vztahu (1) je řízení otáček asynchronního stroje mnohem snadnější, protože ke změně otáček stačí změna frekvence. Tato regulace je na rozdíl od předchozího případu spojitá a má velký rozsah řízení otáček oproti případu, kdy měníme velikost napájecího napětí. Pro změnu frekvence je nejsnadnější použít frekvenční měnič.

Kromě řízení otáček můžeme chtít řídit i směr otáčení. Směr otáčení rotoru je závislý na točivém magnetickém poli, který vytváří stator. U třífázového stroje je vytvoření točivého pole a tím i směr otáčení podstatně snadnější než u jednofázového stroje. U třífázového stroje dochází k vytvoření točivého magnetického pole automaticky,

k otočení směru otáčení magnetického pole je zapotřebí prohodit libovolné dvě fáze mezi sebou. U jednofázového stroje musíme pro rozběh (i směr) použít přídavné rozběhové vinutí.

Rozběh motoru si popíšeme pro třífázový asynchronní motor s kotvou na krátko. Pro spuštění motoru máme následující možnosti:

1. **Přímé připnutí na síť**

   Tímto způsobem je povoleno rozbíhat pouze stroje malých výkonů (do 3 kW), protože záběrný proud může dosáhnout čtyř až osminásobku jmenovitého proudu. Pro spouštění stroje s vyšším výkonem musí mít stroj zvláštní přípojku a jištění.

2. **Přepínání hvězda – trojúhelník**

   Na začátku připneme statorové vinutí do hvězdy (záběrný proud a moment motoru se sníží třikrát) a po rozběhu motoru přepneme statorové vinutí do trojúhelníku. Toto řešení vyžaduje jen přepínač a využívá se často u strojů do 6 kW.

3. **Statorový spouštěč**

   Do série k přívodům motoru jsou připojeny rezistory, které zajistí snížení záběrného proudu na libovolnou žádanou hodnotu.

4. **Polovodičový řízený měnič napětí (softstartér)**

   Softstartér umožňuje pozvolný rozběh motoru, při kterém dochází ke snížení záběrného proudu a momentu. Řízení rozběhu probíhá buď pozvolným nárůstem napětí, nebo omezením proudu motoru během rozběhu.

5. **Dvojitá klec na rotoru (Boucherotova)**

6. **Vírová kotva**

Případy 5 a 6 jsou již velice speciální a jsou popsány v **[1]**.

Brzdění asynchronního motoru je vyžadováno v případě, kdy chceme zkrátit dobu doběhu stroje nebo ve speciálních případech aplikací. Brzdění asynchronního stroje lze provést:

1. **Brzdění protiproudem**

Základní myšlenkou tohoto brzdění je otočení směru otáčení magnetického pole statoru, čímž dochází k brzdění rotoru. Po dosažení nulových otáček je potřeba odpojit motor od sítě, aby nedošlo k roztočení rotoru na opačnou stranu. Tuto metodu je možné uplatnit jen krátkodobě, protože dochází k přetěžování motoru.

2. **Brzdění stejnosměrným proudem (dynamické brzdění)**

Asynchronní stroj je odpojen od sítě a na svorky stroje je přivedeno stejnosměrné napětí, které vytvoří stojící magnetické pole, jehož následkem dochází k vytvoření brzdného momentu rotoru.

3. **Brzdění v generátorickém chodu**

Tento způsob brzdění využívá nadsynchronních otáček rotoru. V tomto režimu pracuje motor jako generátor (dodává energii do sítě tzv. rekuperace) a je brzděn generátorovým momentem. Při použití frekvenčního měniče lze udržet motor v generátorovém režimu změnou frekvence v širokém rozsahu rychlostí.

2.4 Řízení frekvenčních měničů

Řízení frekvenčních měničů lze rozdělit na dva základní typy, skalární a vektorové řízení (**[4]**, **[5]**, **[6]** a **[7]**). Oba způsoby se snaží zachovat moment a rychlost při použití zpětné vazby a regulátoru, který ovlivňuje modulaci. Použití skalárního či vektorového řízení je závislé na typu aplikace, ve které je motor použit. Skalární řízení je podstatně jednodušší než vektorové, ale je vhodné pouze do omezeného množství aplikací, protože nedosahuje tak vysoké přesnosti řízení otáček.

1. **Skalární řízení**

Skalární řízení frekvenčních měničů vychází z rovnic pro ustálený stav asynchronního stroje. Model motoru nezahrnuje elektromagnetické jevy uvnitř stroje a z tohoto důvodu neumožňuje řízení okamžité hodnoty momentu, což má za následek horší dynamiku regulace rychlosti. Skalární řízení bylo pro svou

jednoduchost využíváno v pohonech starší generace a dále je využíváno v levných pohonech s nízkými nároky na dynamiku pohonu, například v pohonech čerpadel nebo ventilátorů.

Rychlost rotoru asynchronního stroje je přímo úměrná frekvenci napájecího napětí. Změnou frekvence napájecího napětí docílíme změnu otáček, pokud však chceme zachovat konstantní moment zvratu, musíme zachovat konstantní poměr (3). Motor lze řídit tímto způsobem ve velkém rozmezí frekvencí (do jmenovité hodnoty), u vyšších frekvencí však dochází k problému udržení konstantního poměru z důvodu velikosti napětí, čímž dochází k poklesu momentu.

$$\frac{\mathrm{U}}{f} = konst. \tag{3}$$

Zvýšení přesnosti řízení otáček pomocí skalárního řízení lze dosáhnout přidáním PID regulátoru. Regulátor je možné implementovat do řídící části frekvenčního měniče. V některých aplikacích lze dosáhnout srovnatelných výsledků s vektorovým řízením, při jednodušším způsobu realizace.

2. Vektorové řízení

Vektorové řízení asynchronních motorů vychází z modelu popisujícího elektromagnetické a elektromechanické jevy ve stroji. Pomocí tohoto modelu lze efektivně řídit okamžité hodnoty toků a proudů ve stroji a rovněž okamžitou hodnotu momentu stroje. Na asynchronní pohon s vektorovým řízením pak lze pohlížet jako na stejnosměrný pohon s cizím buzením, kde lze řídit nezávisle tok motorem a jeho moment.

Vektorové řízení nenastavuje pouze amplitudu statorového napětí (velikost magnetického toku a proudu), ale nastavuje i úhel mezi magnetickým tokem a proudem, neboť velikost momentu závisí nejen na velikosti těchto veličin, ale i na fázovém posunu mezi nimi. Největšího momentu lze dosáhnout, jestliže je proud posunut o 90° vůči magnetickému toku.

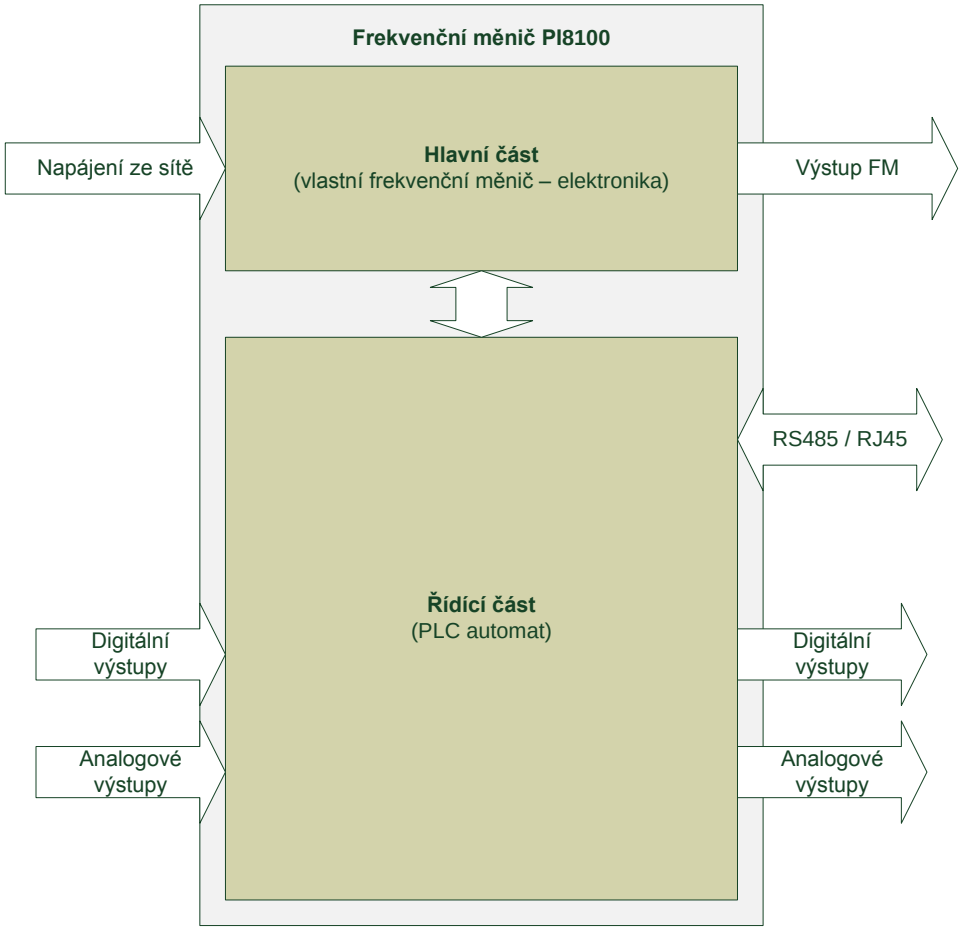
2.5 Frekvenční měnič Powtran PI8100

Frekvenční měnič Powtran PI8100 (Obr. 2-2) je třífázový frekvenční měnič, napěťového typu, který může pracovat i s jednofázovým vstupem. Skládá se ze dvou částí (Obr. 2-3), hlavní části (vlastní frekvenční měnič) a řídící části (PLC modul).

Obr. 2-2 Frekvenční měnič Powtran PI8100 **[8]**

Hlavní část představuje vlastní frekvenční měnič (viz kap. 2.2), u kterého je na vstupu měniče zapojena vstupní reaktance pro potlačení vyšších harmonických složek napájecího proudu. Dále je ke vstupu měniče připojen vstupní odrušovací filtr, který potlačí elektromagnetické rušení, jenž vyzařuje frekvenční měnič. Vstupní část je vybavena stykačem, který je určený pro bezpečnostní funkci, při které je potřebné odpojit napájení. K hlavní části se do meziobvodu dá připojit brzdný odpor, pro zvýšení účinku brždění asynchronního motoru (při brždění asynchronního motoru se chová motor jako generátor = vrací energii do měniče). K výstupu měniče je možné připojit odrušovací filtr, který filtruje proud odebíraný motorem a výrazně tím zabraňuje elektromagnetickému rušení. Zapojením výstupní reaktance měniče a vyladěním spínací frekvence měniče

odstraníme možné proudové přetížení měniče, ke kterému může docházet nevhodnou volbou spínací frekvence a volbou kabelu mezi měničem a motorem.



Obr. 2-3 Blokové schéma frekvenčního měniče PI8100

Řídící částí je PLC automat, který se stará o řízení vlastního měniče. PLC automat generuje řídící signály na základě nastavených parametrů a zpětných vazeb (vektorové řízení). Komunikuje s okolím pomocí analogových/digitálních vstupů a výstupů, a umožňuje připojení frekvenčního měniče do sítě pomocí komunikačního rozhraní (RS-485 a RJ-45).

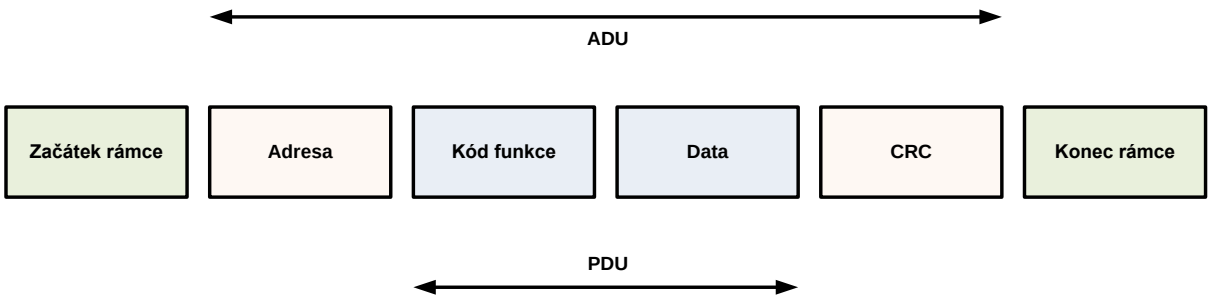
3 Komunikační protokol MODBUS

3.1 Obecný popis

Komunikační protokol MODBUS je standard pro sériovou komunikaci mezi různými typy zařízení na různých typech sítě. MODBUS původně publikovala firma Modicon (nyní Schneider Electric) v roce 1979, pro použití ke komunikaci mezi svými PLC automaty (viz **[9]**). Jedná se o komunikaci typu master-slave (klient-server), kde každá komunikace je vyvolána řídící jednotkou (zařízením typu master). Zařízení se mezi sebou dorozumívají pomocí datových zpráv. Komunikace může být provozována na různých typech komunikačních médií například sériové linky (RS-232, RS-422 a RS-485), optických nebo bezdrátových sítích, ale i na sítích Ethernet pomocí TCP/IP protokolu.

3.2 Popis protokolu

Protokol MODBUS definuje strukturu zprávy na dvou úrovních. Konkrétně na úrovni protokolu PDU (Protocol Data Unit), kde je definována struktura zprávy nezávisle na typu komunikační vrstvy, a na aplikační úrovni ADU (Application Data Unit), kde je zpráva rozšířena o další části (adresa, kontrolní součet). Základní tvar zprávy je znázorněn na následujícím obrázku (Obr. 3-1).



Obr. 3-1 Základní tvar rámce

Začátek a konec rámce jsou definovány jako klidový stav sběrnice, kdy nedochází k žádnému přenosu dat po předem definovaný čas. Adresa je identifikační kód zařízení, které definuje příjemce, případně odesílatele zprávy. Adresa zařízení musí být unikátní

(unikátní identifikační kód), čímž je jednoznačně určeno zařízení. V našem případě jsou možné adresy zařízení 1 až 127. Adresa 0 definuje jako příjemce všechna zařízení připojená ke sběrnici. Hodnoty 128 až 255 jsou rezervovány. Kód funkce definuje operaci, která se má vykonat, může nabývat hodnot 1 až 255, při čemž hodnoty 128 až 255 jsou vyhrazeny pro záporné odpovědi (chyby). Data blíže specifikují registry a hodnoty, se kterými má funkce pracovat. CRC je kontrolní součet, který slouží k zabezpečení komunikace.

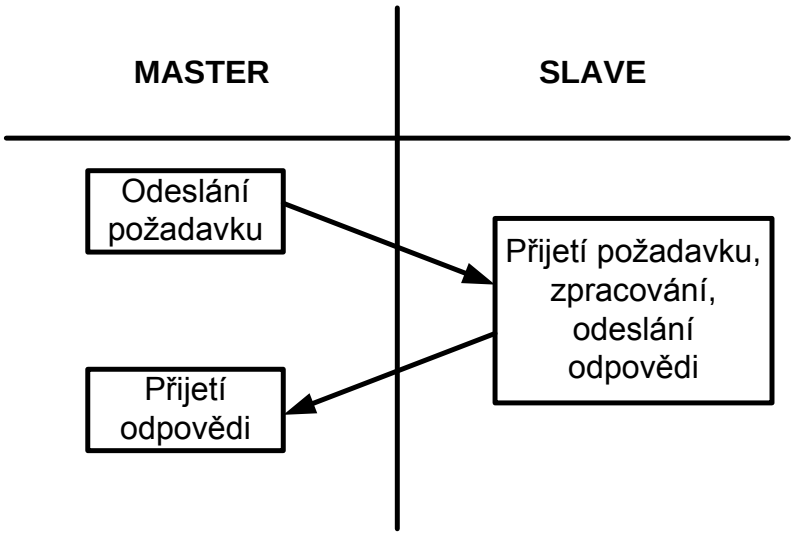
Zabezpečení přenášených dat nemusí být závislé pouze na kontrolním součtu, ale může se zabudovat i do komunikačního procesu jednotky master. Z pohledu mastera mohou nastat tři základní situace:

1. **Přenos proběhl v pořádku** (Obr. 3-2)

   Master vyslal požadavek, který slave zpracoval a odeslal kladnou odpověď.

2. **Jednotka slave odeslala zápornou odpověď** (Obr. 3-2)

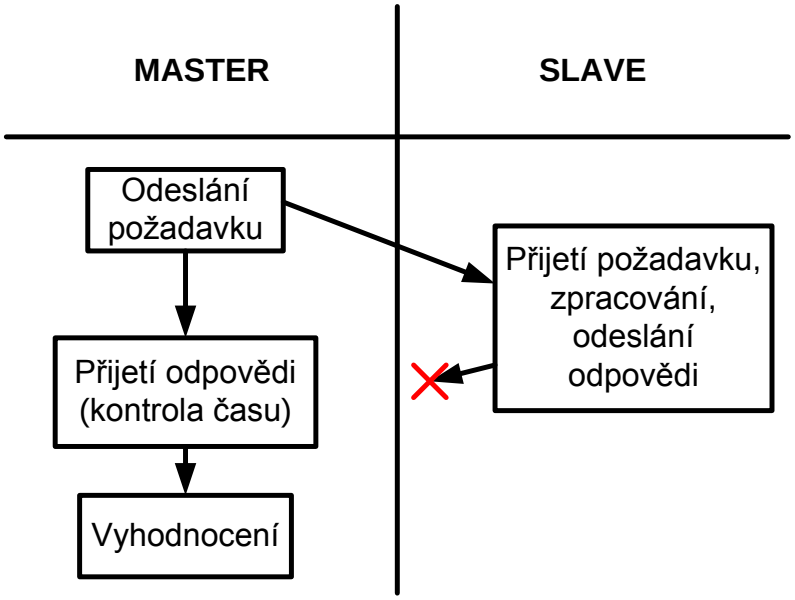
   Jednotka slave nemohla vykonat požadovanou operaci a odeslala jednotce master zprávu o chybě. Další reakce na chybnou odpověď záleží na jednotce master, která může zobrazit varování, že došlo k chybě a prozatím pozastavit proces než se chyba odstraní nebo může přenos opakovat, dokud nedojde ke kladné odpovědi.



Obr. 3-2 Komunikace master-slave s doručenou odpovědí

3. **Ztráta dat** (Obr. 3-3)

Mezi zařízeními došlo ke ztrátě dat, např. odpojení zařízení, napájení, porucha sběrnice. Opět je na jednotce master, aby si s nastalou situací poradila a případně rozpoznala příčinu poruchy a informovala uživatele.



Obr. 3-3 Komunikace master-slave se ztrátou dat

3.3 Specifikace protokolu frekvenčním měničem Powtran PI8100

V předcházejících částích této kapitoly jsme se seznámili s obecným popisem komunikačního protokolu MODBUS. V této podkapitole rozšíříme obecný popis o konkrétní použití protokolu na frekvenčním měniči Powtran PI8100, při použití sériové sběrnice RS-485. Všechny informace z této podkapitoly a mnohé jiné se dají najít v uživatelské příručce k danému měniči **[10]**.

Základní specifikace komunikace se dají shrnout do několika bodů (Tab. 3-1). Mezi nejvýznamnější parametry komunikace patří rychlost komunikace, která je v měniči defaultně nastavena na 19200 bps, ale je možné ji změnit, a typ komunikačního protokolu. Další parametry se týkají sestavení rámce na úrovni bitů a bytů.

Tab. 3-1Specifikace komunikace pro RS-485

Typ	Specifikace
Rychlost komunikace	38400/**19200**/9600/4800/2400/1200 bps, možno měnit
Komunikační protokol	MODBUS, formát RTU[1]
Vlastnosti rozhraní	Asynchronní komunikace, polo-duplexní Vyšší byte předchází nižší
Formát dat (přenos 1 bytu)	1 strat bit, 8 datových bitů, 1 stop bit, bez parity
Adresa slave zařízení	Možní adresy slave zařízení: 1 – 127 Adresa 0 ~ všechna zařízení Ostatní adresy jsou rezervovány

Rámec zachovává v každém přenosu základní tvar (Obr. 3-1), velikost jednotlivých částí se však může lišit s ohledem na použitém kódu funkce. Kód funkce je specifikován v uživatelské příručce, v diplomové práci si však vystačíme se čtyřmi základními funkcemi (Tab. 3-2). Jednotka slave (frekvenční měnič) má k dispozici navíc kódy funkcí pro záporné odpovědi (Tab. 3-3).

Tab. 3-2 Specifikace kódu funkce

Kód funkce	Význam
0x03	Funkce pro čtení jednoho registru
0x06	Funkce pro zápis jedné funkce
0x10	Funkce pro zápis do více registrů
0x05	Funkce pro zápis stavu přepínače

[1] Režim RTU – 8 bitový byte je posílán jako jeden znak

Tab. 3-3 Specifikace kódu funkce pro zápornou odpověď

Kód funkce	**Význam**
0xA0	Neplatná operace, nastavení stavu je neplatné
0xA1	Kód funkce je neplatný
0xA2	Porucha záznamu je prázdný
0xA3	Adresa registru je neplatná
0xA4	Jednotka slave je zaneprázdněna, EEPROM zpoždění
0xA5	Správce je omezen
0xA6	Nastavená hodnota je mimo limit
0xA7	Chyba CRC součtu
0xA8	Chyba rámce

Po kódu funkce následuje v rámci blok dat, který se liší v závislosti na kódu funkce a zda se jedná o příkaz od jednotky master nebo o odpověď od dotazované jednotky slave. Hexadecimální kód 0x03 kódu funkce odpovídá čtení registru, který je dán adresou registru a číslem registru. Adresa registru odpovídá konkrétní funkci a číslo registru může nabývat hodnot od 1 do 16. U většiny funkcí stačí číst pouze první registr. Kód funkce 0x06 odpovídá zápisu dat do určité funkce, která je dána adresou registru. Kód funkce 0x10 slouží pro zápis dat do více registrů. Adresa registru definuje konkrétní funkci a číslo registru (1-16) určí, do jakého registru se bude zapisovat. Funkce 0x05 pro zápis stavu přepínače slouží k připojení a odpojení výstupu frekvenčního měniče. Přepínač se může nacházet pouze ve dvou polohách, pro hodnotu výstupu 0xFF00 je spínač zapnut (ON) a při hodnotě výstupu 0x0000 je spínač rozepnut (OFF), jiné hodnoty jsou nepřípustné. Pro zápornou odpověď se rámec odpovědi zkrátí o data (Obr. 3-8).

Požadavek jednotky master

Začátek rámce	Adresa	Kód funkce	Adresa registru	Číslo registru	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	2 byty	2 byty	Interval ≥ 3,5 bytu

Odpověď jednotky slave

Začátek rámce	Adresa	Kód funkce	Čtený byte	Obsah čtení	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	1 byte	2 byty * číslo registru	2 byty	Interval ≥ 3,5 bytu

Obr. 3-4 Ukázka délky rámce pro čtení jednoho registru (0x03)

Požadavek jednotky master

Začátek rámce	Adresa	Kód funkce	Adresa registru	Data registru	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	2 byty	2 byty	Interval ≥ 3,5 bytu

Odpověď jednotky slave

Začátek rámce	Adresa	Kód funkce	Adresa registru	Data registru	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	2 byty	2 byty	Interval ≥ 3,5 bytu

Obr. 3-5 Ukázka délky rámce pro zápis jedné funkce (0x06)

Požadavek jednotky master

Začátek rámce	Adresa	Kód funkce	Adresa registru	Číslo registru	Obsah registru byte	Obsah registru	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byte	2 byte	1 byte	2 byty * číslo registru	2 byty	Interval ≥ 3,5 bytu

Odpověď jednotky slave

Začátek rámce	Adresa	Kód funkce	Adresa registru	Číslo registru	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	2 byty	2 byty	Interval ≥ 3,5 bytu

Obr. 3-6 Ukázka délky rámce pro zápis do více registrů (0x10)

Požadavek jednotky master

Začátek rámce	Adresa	Kód funkce	Adresa výstupu	Hodnota výstupu	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	2 byty	2 byty	Interval ≥ 3,5 bytu

Odpověď jednotky slave

Začátek rámce	Adresa	Kód funkce	Adresa výstupu	Hodnota výstupu	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	2 byty	2 byty	Interval ≥ 3,5 bytu

Obr. 3-7 Ukázka délky rámce pro zápis stavu přepínače (0x05)

Začátek rámce	Adresa	Kód funkce	CRC	Konec rámce
Interval ≥ 3,5 bytu	1 byte	1 byte	2 byty	Interval ≥ 3,5 bytu

Obr. 3-8 Ukázka délky rámce pro zápornou odpověď

V předchozím odstavci jsme zmínili konkrétní funkci, ta určuje adresu registru, do kterého se bude zapisovat nebo se z něj číst. Funkcí, které frekvenční měnič nabízí je nepřeberné množství a dají se rozdělit do skupin (Tab. 3-4), a ty se pak následně dělí na konkrétní funkce (viz **[10]** str. 24-99).

Kontrolní součet je posledním blokem v rámci, jeho výpočet specifikuje komunikační protokol MODBUS. V našem případě se jedná o MODBUS CRC-16 a pro jeho výpočet je použita adresa, kód funkce a data. MODBUS CRC-16 je definován dělícím polynomem (4). Výpočet CRC součtu odpovídá logické funkci XOR a implementace je volně dostupná na internetu. Kontrolní součet slouží k zabezpečení dat a umožňuje detekci chyb v přenosu, CRC je nezávisle vypočítáno po přijetí dat a porovnáno s CRC kódem ve zprávě.

$$G(x) = x^{16} + x^{15} + x^{2} + 1 \tag{4}$$

Tab. 3-4Základní skupiny funkcí

Kód funkce	**Název funkce**	**Popis funkce**	**Hex. kód funkce**
S	Monitorovací funkce	Monitorování frekvence, proudu aj.	0x0B
F	Základní funkce	Nastavení frekvence, mód řízení, doba náběhu a doběhu	0x00
A	Uživatelské funkce	Nastavení komunikace, monitorování aj.	0x01
o	IO funkce	Analogové a digitální vstupy, výstupní funkce	0x02
H	Multi-rychlostní PLC funkce	Běh PLC, multi-rychlostní nastavení	0x03
U	Funkce pro V/F parametry	Uživatelem definovaná V/F křivka	0x04
P	Funkce pro nastavení PID	Nastavení PID parametrů	0x05
E	Rozšiřující funkce	Nastavení konstantního tlaku vody aj.	0x06
C	Funkce pro rychlostní zpětnou vazbu	Nastavení rychlosti, měřící smyčky aj.	0x07
b	Funkce pro nastavení parametrů motoru	Nastavení parametrů motoru	0x08
y	Systémové funkce	Parametry resetu, informace o produktu aj.	0x09

4 Správa dat

4.1 Obecný popis

Správa dat je základem celé aplikace. Externí zdroje dat, které jsou načítány ze souborů, umožňují aplikaci větší univerzálnost a snadnější rozšiřitelnost o další zdrojové soubory. Aplikace vytvořená v rámci této diplomové práce pracuje se čtyřmi typy externích dat:

1. **Jazyková modifikace** – výběr jazyka aplikace, obsahuje překlady většiny textů, které jsou zobrazeny v grafických prvcích.

2. **Informace o konkrétním měniči** – informace o měniči získaná z manuálu k měniči, rozšířená o další data, která usnadní práci s komunikačním protokolem.

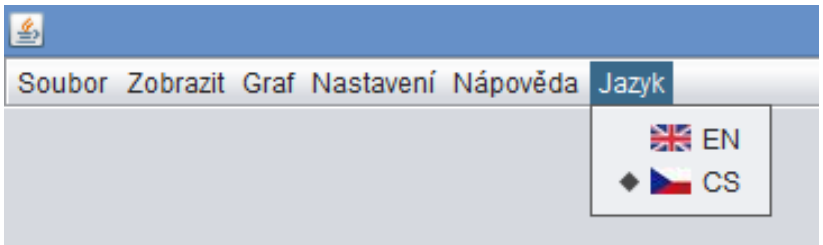
3. **Správa projektu** – uložení nebo načtení dat o projektu.

4. **Správa naměřených dat** – uložení nebo načtení naměřených hodnot zobrazených v grafech.

Data lze uložit do zdrojového kódu v samotné aplikaci, ale rozšíření o nová data by vyžadovalo větší programátorský zásah. Optimálním řešením je externí zdroj dat, který aplikace načte a uloží do paměti (proměnných - struktury), s nimiž se nadále pracuje. Rozšíření aplikace o nový zdroj dat vyžaduje do zdrojového kódu připsat několik řádků. U externího zdroje dat se pak musíme již jen rozhodnout, zda použít online nebo offline databázi. U online databáze je snadnější distribuce jejího rozšíření, ale za cenu připojení k internetové síti a přístupu k dané databázi. Naopak u offline databáze není potřeba připojení k síti, ale rozšíření databáze (aktualizace) je třeba distribuovat aktualizacemi aplikace (přidání datových souborů). Ukládání a načítání dat vytvořených samotnou aplikací je jen práce se soubory, které jsou přizpůsobeny konkrétním požadavkům.

4.2 Jazyková modifikace

Jazyková modifikace zajišťuje jazykovou flexibilitu celé aplikace. Jedná se o databázi s překlady jednotlivých názvů a textů grafických prvků, které jsou v aplikaci použity. Externí zdroj jazykové modifikace umožní pozdější přidání další jazykových překladů (aktuálně pouze dva čeština a angličtina). Ve vlastní aplikaci následně stačí přidat položku s výběrem dalšího jazyka (Obr. 4-1) a do zdrojového kódu připsat podmínku na výběr nového jazyka.



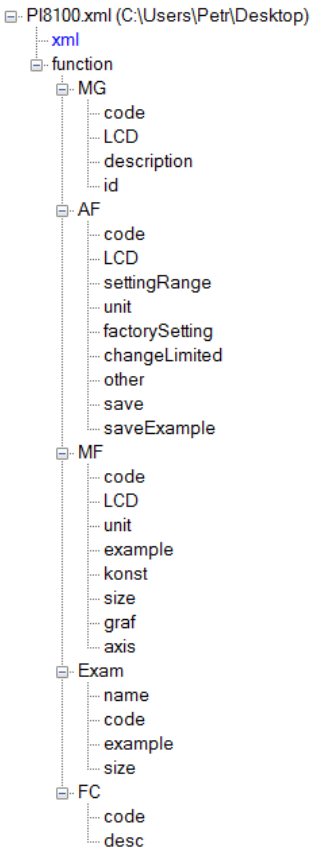
Obr. 4-1 Výběr jazykové modifikace aplikace

Vlastní změna textu v aplikaci je pro všechny grafické prvky vytvořena vždy, když dojde ke změně jazyka. Každý prvek je pojmenován identifikačním číslem, které specifikuje v souboru s jazyky příslušný text, vybraný jazyk určí konkrétní jazykový překlad. Soubor s jazykovou modifikací je formátu CSV, který má určitá pravidla. CSV soubor je tabulka přepsaná do textového souboru, jednotlivé sloupce tabulky jsou odděleny specifickým symbolem (např. čárkou) a řádek v textovém dokumentu představuje řádek v tabulce. Načtení CSV souboru v Javě lze zjednodušit přidáním externí knihovny, pomocí které načteme CSV soubor do dvou rozměrné tabulky typu *String*. Identifikační číslo prvku následně určí řádek v tabulce a vybraný jazyk určí sloupec v tabulce, tímto způsobem určíme konkrétní buňku tabulky, jejíž text se zobrazí v grafickém prvku. Při vytváření CSV souboru a přiřazení identifikačního čísla musíme postupovat synchronizovaně, abychom zaručili přiřazení textu správnému grafickému prvku.

4.3 Načtení dat měniče

Pod daty měniče rozumíme data uvedená v manuálu ke konkrétnímu frekvenčnímu měniči, v našem případě PI8100 **[10]**. V první řadě se jedná o přepis veškerých tabulek, které jsou rozšířené o další důležitá data, která popisují komunikaci jednotlivých funkcí. Dále se jedná o přidání dalších informací, která jsou pro správný chod aplikace nezbytná. Hlavním problémem je, jak velké množství různých dat uchovávat a zda použít online nebo offline přístup. Při použití online přístupu by bylo vhodné použít například SQL databázi, ale ne vždy je možné získat přístup k síťovému připojení. Z tohoto důvodu je vhodnější použít offline databázi, která by se při vývoji aplikace dala snadno měnit a co nejvíce usnadnit aplikaci přístup k datům. Vzhledem k tomu, že databáze obsahuje velké množství různých dat, je vhodné uložit data do XML souboru.

XML **[11]** je standard pro psaní strukturovaného textu, který vytvoří XML dokument (soubor), jenž obsahuje tagy (značky), elementy a entity. Načtení XML dokumentu je pro Javu poměrně snadné, protože již sama obsahuje parsery (SAX a DOM parser).



Obr. 4-2 Stromová struktura XML souboru pro PI8100

Data v XML dokumentu můžeme popsat stromovou strukturou (Obr. 4-2), kterou po načtení dokumentu přiřadíme do jednotlivých proměnných a dále s nimi v aplikaci pracujeme.

Hlavní větev je označená function (funkce), která se dále rozděluje na dalších pět uzlů, v nichž jsou obsažené konkrétní informace pro daný uzel.

Uzel MG (Menu Group – Skupina funkcí) obsahuje informace o funkcích, jsou to označení funkce (code), název funkce (LCD), popis funkce (description) a hexadecimální označení (id).

Uzel AF (All Function – Všechny funkce) obsahuje informace o všech funkcích (adresa registru), definuje funkce (code), název funkce (LCD), rozsah (settingRange), jednotku (unit), tovární nastavení (factorySetting), měnitelný rozsah (changeLimited), jiné – popis (other), uložení (save), příkaz pro uložení (saveExample).

Uzel MF (Monitoring Function – Sledované funkce) obsahují informace nezbytné k snadnému monitorování parametrů a informace důležité pro zobrazení hodnot v grafu, jsou to parametry označení (code), název (LCD), jednotka (unit), příkaz pro čtení hodnoty (example), konstanta pro dělení – úprava přečtené hodnoty do základních jednotek (konst), očekávaná délka odpovědi (size), číslo grafu, ve kterém se zobrazí hodnota (graf), osa (axis), na kterou se zobrazí jednotky.

Uzel Exam (Example – příklad) označuje funkce důležité pro nastavení běhu motoru (směr otáčení, spuštění / zastavení a zápis frekvence), obsahuje tyto parametry název (name), označení (code), hexadecimální kód (example) a délku očekávané odpovědi (size).

Posledním uzlem je FC (Function Code - Kód funkce), který definuje operace, které může aplikace použít. Tyto operace jsou definovány hexadecimálním kódem (code) a popisem (desc).

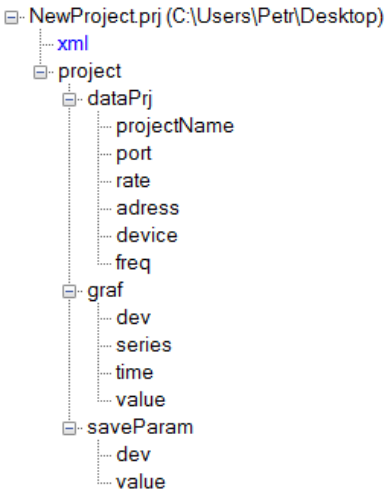
4.4 Správa projektu

Správa projektu zahrnuje uložení a načtení dat o projektu, která popisují základní informace o projektu důležitá pro aplikaci, informace získaná přečtením registrů frekvenčního měniče a naměřená data. Opět se jedná o velké množství různých dat, která uložíme do XML dokumentu s příponou *.pjr*.

Základní informace (uzel dataPrj) v sobě obsahují název projektu, informace o sériovém portu (číslo sériového portu a rychlost komunikace), adresy zařízení a jejich typ a hodnota frekvence, která byla nastavena naposledy.

Naměřená data (uzel graf) jsou uložena opět s adresou zařízení, aby mohla být při načtení projektu zpětně vykreslena do grafu k danému zařízení.

Informace získané přečtením registrů frekvenčního měniče (saveParam) jsou ukládány jednotlivě s adresou zařízení a hexadecimálním popisem dat.



Obr. 4-3 Stromová struktura XML souboru projektu

4.5 Správa naměřených dat

Správa naměřených dat v sobě obsahuje uložení a načtení naměřených dat v aktuálně aktivním zařízení. Lze proto ukládat naměřená data pouze z jednoho zařízení, v případě, že by se uživatel chtěl uložit data z více zařízení, musí je ukládat postupně. Uložení dat

z více zařízení by znamenalo složitější struktury při ukládání i načítání souborů. Aby bylo možné uložená naměřená data využít i v externích programech jako je například tabulkový editor Excel, je nutné data uložit přímo v *.xls* nebo CSV souboru *.csv*. Toto uložení je v Javě možné přidáním knihoven pro práci s těmito soubory, tyto knihovny umožní opětovné načtení dat. Dále pro rychlé uložení obrázku zobrazeného grafu je možné aktuálně aktivní grafy uložit jako obrázky ve formátu *.png*. Při načítání je možné načíst pouze XLS a CSV soubory, u kterých je před načtením nutné zvolit typ zařízení.

Při využívání dat v externích programech musíme mít na paměti, že data jsou uložená v anglickém formátu. Oddělovač desetinné části není čárka, ale tečka. Zároveň dochází k uložení dat, které byla označena jako „špatná“ (chybová hláška či špatný součet CRC) a tyto data jsou uložena s vyhodnocením *null*. Při vykreslení v aplikaci dojde k přerušení grafu v daném časovém bodě.

5 Aplikace

5.1 Výběr programovacího jazyka

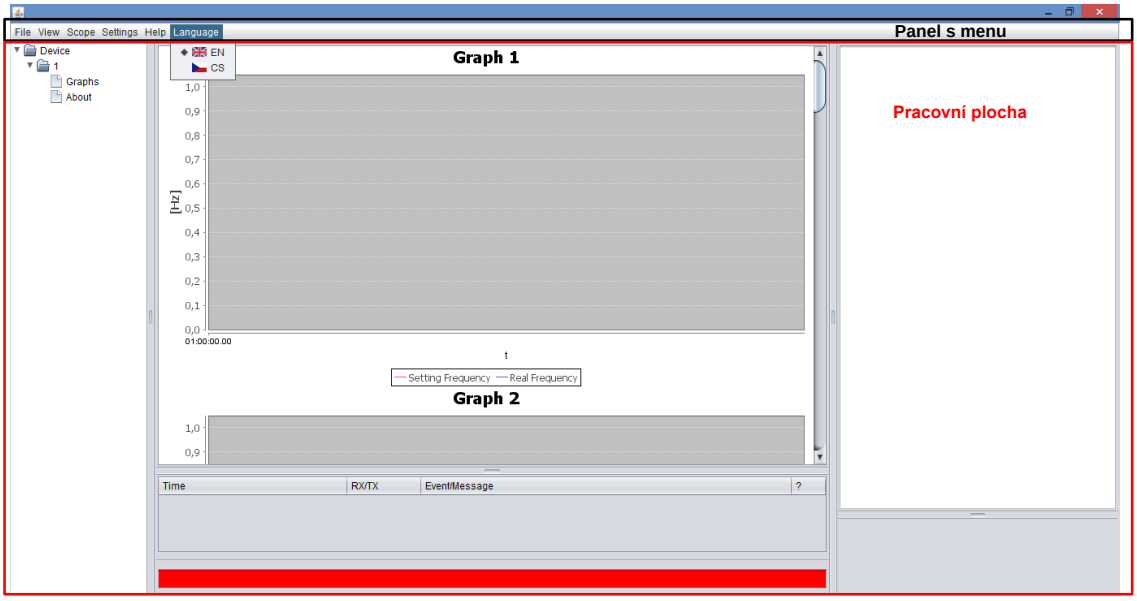
Před začátkem realizace (programování) projektu se musíme rozhodnout pro programovací jazyk. Výběr je závislý na použití aplikace, platformy, na které aplikace poběží, a programovacích nástrojů (grafické zpracování, podpora vláken). Cílem při realizaci aplikace je vytvoření grafického rozhraní, které usnadní komunikaci s frekvenčním měničem a zároveň zobrazí data získaná z měniče. Pro komunikaci mezi počítačem a frekvenčním měničem je použita sběrnice RS-485 s převodníkem na USB, musíme tedy zvolit program, který tento převodník podporuje, případně ho podporuje po vložení externí knihovny **[12]**. Od hotové aplikace chceme, aby byla nezávislá na operačním systému a byla více vláknová. Vše tyto požadavky splňuje Java, která nabízí možnost snadného vytvoření grafického rozhraní.

5.2 Základní popis aplikace

Aplikace se dá popsat ze dvou pohledů, jedná se grafické zpracování a implementaci. Grafické zpracování jsou okna, která umožňují komunikaci s programem a s frekvenčním měničem. Grafika zároveň poskytuje uživateli zpětnou vazbu pomocí textových výpisů a grafů. Implementace je vlastní zdrojový kód, který se stará o komunikaci, zpracování událostí vyvolaných uživatelem, správu dat atd.

1. **Grafické zpracování**

   Grafické zpracování aplikace v sobě zahrnuje všechna okna, která aplikace používá. Okna se dají rozdělit na hlavní okno aplikace (Obr. 5-1) a pomocná okna, která jsou vyvolána z hlavního. Hlavní okno se dá rozdělit na panel s menu (kap. 5.3) a pracovní plochu (kap. 5.4). Pomocná okna umožňují uživateli změnit některá nastavení aplikace, vytvoření projektu, ukládání a načítání dat aj. Všechna okna a jejich užití budou probrána v následujících kapitolách.



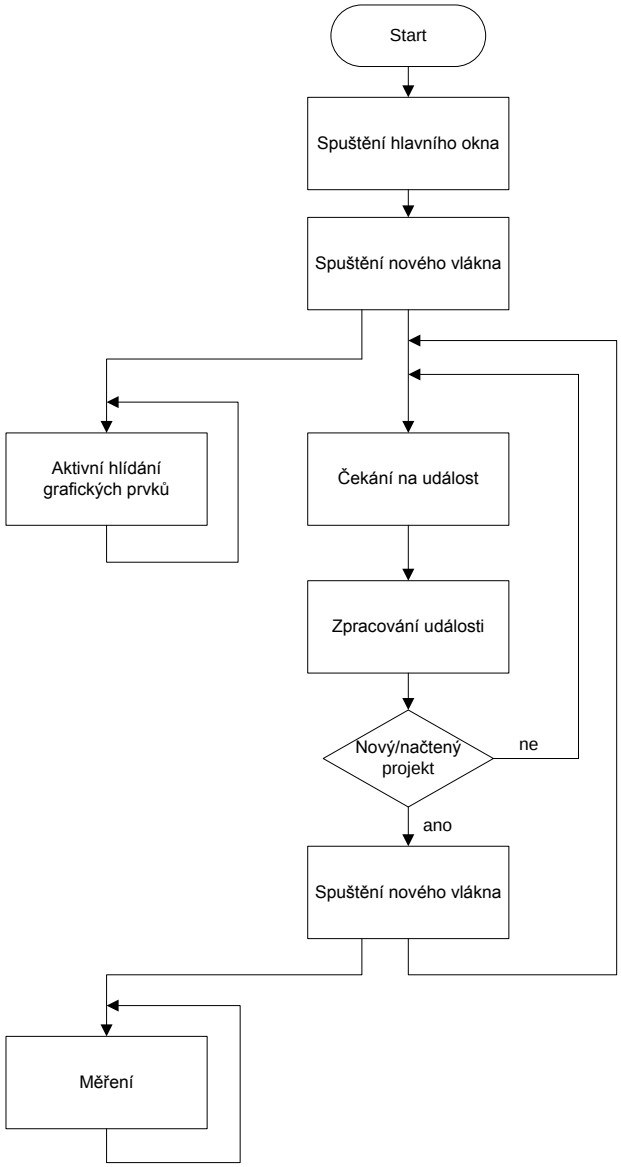
Obr. 5-1 Hlavní okno aplikace

2. Implementace

Implementace v sobě nezahrnuje jen zdrojový kód, ale i schéma průběhu aplikace, případně vývojový diagram. V tomto případě by byl celý vývojový diagram moc složitý a rozsáhlý, proto použijeme jen zjednodušené schéma, které popíše základní myšlenku implementace aplikace (Obr. 5-2). Po spuštění programu dojde k vytvoření hlavního okna, které spustí další dvě vlákna. Celkově tedy v aplikaci běží paralelně tři vlákna.

Hlavní vlákno (hlavní okno aplikace – kap. 5.3 a 5.4) je zodpovědné za vytvoření všech struktur, objektů a tříd v závislosti na událostech vyvolaných uživatelem (stisk tlačítka, položky v menu aj.). V konstruktoru hlavního vlákna je vyvoláno druhé vlákno (kap. 5.6), které nepřetržitě kontroluje a upravuje grafické zobrazení prvků (jejich velikost a pozici) jako reakci na změnu velikosti okna, tak aby byl zachován určitý poměr. Toto vlákno kontroluje příkazový řádek a zadanou posloupnost znaků, při chybě je příkazový řádek obarven na červeno, jako zpětná vazba uživateli, že není možno příkaz odeslat. Třetí vlákno (kap. 5.7) je vyvoláno spuštěným projektem a jedná se o měřící vlákno, toto vlákno nepřetržitě odesílá požadavky na čtení monitorovaných registrů frekvenčního měniče. Po přijetí odpovědi zkontroluje rámec a CRC součet a dle výsledku zapíše získaná data do paměti a zároveň je zapíše do grafu.

Paralelní programování sebou přináší problém souběhu, musíme tedy zamezit chybám v přístupu ke společným proměnným z různých vláken. Tento problém se netýká jen zápisu a čtení z proměnných, ale musíme při komunikaci přes sériovou linku zabezpečit nepřístupnost tohoto portu, pokud je již využíván jinou částí programu. K vyloučení přístupu k souběhu sériové linky je využito binárního semaforu a klíčového slova *synchronized*. Chyba přístupu k sériové lince může vzniknout z hlavního vlákna a z měřícího vlákna, zároveň se používá při potřebě pozastavit měřící vlákno, ve kterém je pak aplikováno aktivní čekání. Žádné jiné části programu již není třeba synchronizovat a nemělo by u nich tudíž docházet k synchronizačním chybám a deadlockům.



Obr. 5-2 Zjednodušené schéma běhu aplikace

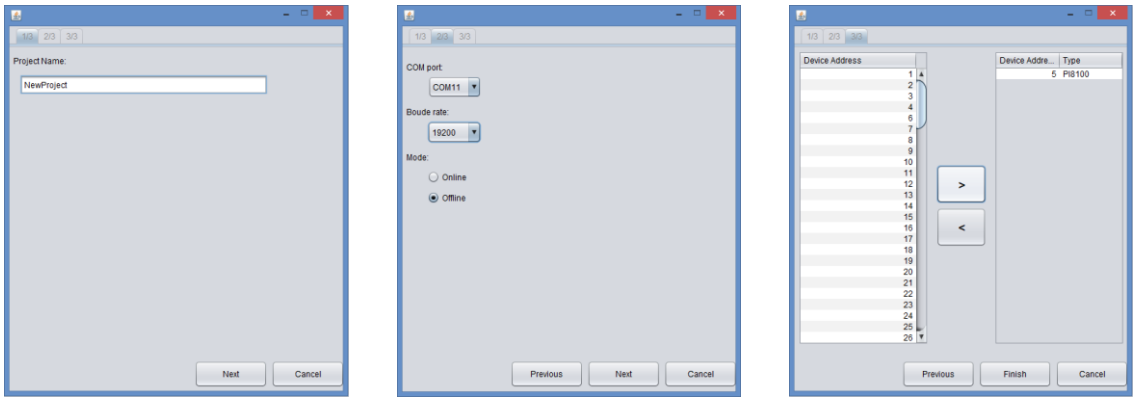
5.3 Panel s menu

Panel s menu (viz Obr. 5-1) nabízí uživateli základní funkce pro nastavení a práci s projektem, je tvořen šesti záložkami. Každá záložka má několik položek, které mají vliv na aplikaci, ve které vyvolají událost a ta je následně zpracována příslušnou obslužnou rutinou. Záložky v panelu jsou `Soubor` (`File`), `Zobrazit` (`View`), Graf (`Scope`), `Nastavení` (`Settings`), `Pomoc` (`Help`), `Jazyk` (`Language`). Každou ze záložek si popíšeme zvlášť, jejich význam a použití položek, které jednotlivé záložky nabízejí.

1. Soubor

Záložka `Soubor` slouží k práci s projektem a datovými soubory, konkrétně se jedná o založení nového projektu, uložení a načtení projektu, uložení a načtení grafů (naměřené hodnoty) a ukončení aplikace. Každá z těchto funkcí je implementována jednou položkou.

Položka `Nový Projekt` vyvolá v aplikaci ihned nové okno (Obr. 2-1) pro nastavení nového projektu a dojde k zablokování hlavního okna. V případě, že v aplikaci již je puštěný projekt, tak se aplikace uživatele zeptá, jestli si přeje ukončit stávající projekt a při kladné odpovědi dojde k ukončení puštěného projektu a až následně je vyvoláno okno pro nastavení nového projektu. Toto okno je rozděleno do tří kroků, při čemž v každém kroku je možné průvodce novým projektem ukončit nebo se vrátit na předchozí krok.



Obr. 5-3 Průvodce nastavením nového projektu

V prvním kroku je nastavení jména projektu.

Druhý krok umožní uživateli vybrat si z připojených sériových portů, pokud není žádný nalezen, přepne program automaticky do offline režimu a nabídne uživateli všechny sériové porty. Dále se v tomto kroku nastavuje rychlost komunikace a výběr offline či online režimu.

Třetí krok nabízí uživateli adresy připojených zařízení ke sběrnici, v online režimu prohledá aplikace sběrnici a nabídne pouze adresy, které jsou na sběrnici použity. V případě offline režimu jsou nabídnuty všechny přípustné adresy. Uživatel v tomto kroku vybere zařízení, která chce použít a to tak, že postupně přesouvá adresy z jedné tabulky do druhé a zároveň vybírá typ zařízení. Když je vybráno minimálně jedno zařízení, se kterým chce uživatel pracovat, aplikace umožní dokončení průvodce.

Po dokončení dojde k ukončení okna průvodce a je odblokováno hlavní okno, dále dojde k vytvoření struktury a načtení dat a ke spuštění měřícího vlákna. Vytvořená struktura (viz níže) v sobě uchovává všechna data, která byla zadána v průvodci nastavení (jméno projektu, vybraný sériová port, rychlost komunikace, vybraná zařízení a jejich typ a struktura pro ukládání naměřených dat). Je-li aplikace v online režimu, dojde k vytvoření objektu obsluhujícího sériový port.

```
// Základní struktura projektu
public class Project {
    String name; // jméno
    String port; // COM port
    String rate; // rychlost
    List<String> address; // adresy zařízení
    List<String> deviceName; // typ zařízení
    List<DatabaseSaxParser> device; // zdrojová data pro zařízení
    List<Graph> graf; // struktura pro uložení hodnot
    List<ControlPanel> cntp; // kontrolní panel

    public Project() {
        address = new ArrayList<String>();
        deviceName = new ArrayList<String>();
        device = new ArrayList<DatabaseSaxParser>();
        graf = new ArrayList<Graph>();
        cntp = new ArrayList<ControlPanel>();
    }
}
```

Položka `Uložení Projektu` nabídne uživateli klasické okno pro ukládání souboru. Soubor se ukládá s příponou *.pjr* a je vytvořen jako textový soubor v XML formátu (kap. 4.4). V případě online režimu jsou z jednotlivých zařízení na sběrnici vyčtena data a uložena do tohoto souboru.

Položka `Načtení Projektu` nabízí uživateli klasické okno pro načítání souborů (s příponou *.pjr*). Načtení souboru proběhne do struktury, která je následně využita pro nastavení projektu. Nastavení projektu probíhá obdobně jako u založení nového projektu, jen s tím rozdílem, že data uložená v souboru obsahují navíc naměřená data, která je nutno uložit k příslušným zařízením, a parametry načtená z jednotlivých zařízení, která je nutno opět uložit do příslušných zařízení.

Položka `Uložení Grafů` je tvořena oknem pro ukládání souborů a lze vybrat ze tří možností typu uložení (do XLS souboru, CSV souboru nebo PNG souboru). Po vybrání typu a stisknutí uložit dojde k uložení aktuálně vybraného zařízení. V případě XLS a CSV souboru dojde k uložení hodnot s názvy sloupců, v případě PNG souboru dojde k uložení každého grafu aktuálně vybraného zařízení do vlastního souboru.

Položka `Načtení Grafů` požaduje v okně pro načítání vybrání souboru typu XLS nebo CSV, po vybrání a stisknutí tlačítka načíst (otevřít), je uživatel dotázán na typ zařízení. Po vybrání zařízení jsou data načtena a uložena do struktury odpovídající typu zařízení a dojde k zobrazení dat v grafech v novém okně.

Pokožka `Ukončit` uvolní aktuálně užívaný sériový port a ukončí aplikaci.

2. Zobrazit

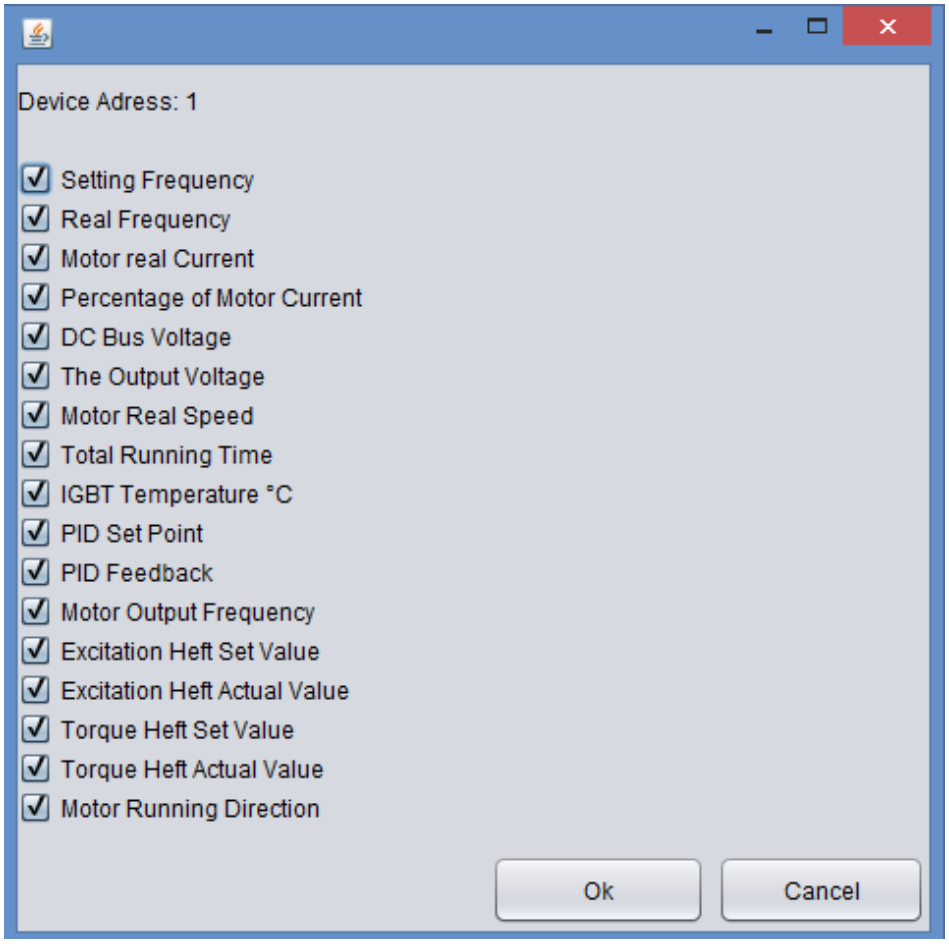
Záložka `Zobrazit` umožňuje uživateli jednoduchou úpravu zobrazených panelů na hlavní ploše. Záložka obsahuje tři položky, které mají binární charakter, protože se jedná o položky, které nastaví svou hodnotu na *true* či *false* podle toho, zda jsou vybrány. Pokud položka není vybrána, tak odebere panel, ke kterému se vztahuje z hlavního okna (panel se nezobrazí). Konkrétně se jedná o panel umožňující nastavení frekvence směru otáčení a připojení/odpojení výstupu frekvenčního měniče (kontrolní panel), panel zobrazující poslední přečtenou

hodnotu naměřených dat (monitorovací panel) a panel sloužící k ručnímu zadávání příkazů (příkazová řádka).

3. Graf

V záložce `Graf` najdeme dvě položky, které mají za úkol uživateli poskytnout možnost změnit monitorované parametry a jejich zobrazení v grafu.

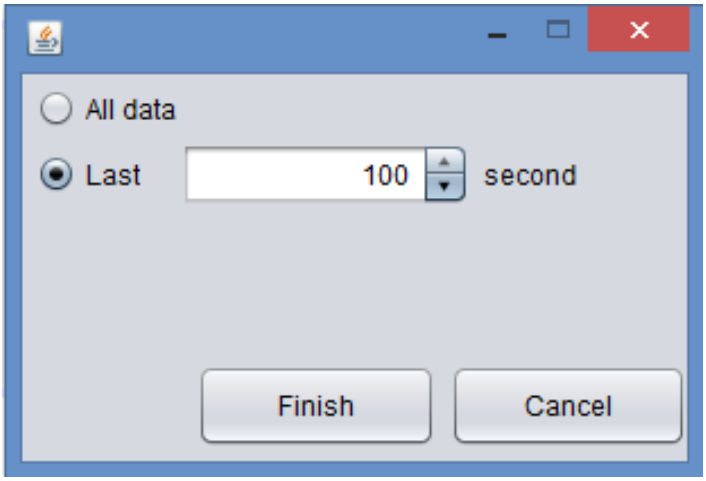
První položka `Výběr Průběhů` vyvolá okno, ve kterém jsou zobrazeny všechny funkce, jenž je možné monitorovat. Uživatel si může vybrat, které z funkcí chce sledovat a zaškrtne je. Po potvrzení volby dojde k uložení těchto monitorovaných funkcí a měřící vlákno začne přeskakovat dotazování na funkce, které nejsou vybrány (nejsou monitorovány).



Obr. 5-4 Okno pro výběr průběhů

Druhá položka `Časová Osa` vyvolá okno, které uživateli nabízí možnosti pro nastavení časové osy grafů. Jedná se o možnost sledovat celý průběh, kde dochází

k postupnému zvyšování rozsahu osy, a o možnost sledovat poslední sekundy, které uživatel zadá (plynulý posun časové osy).



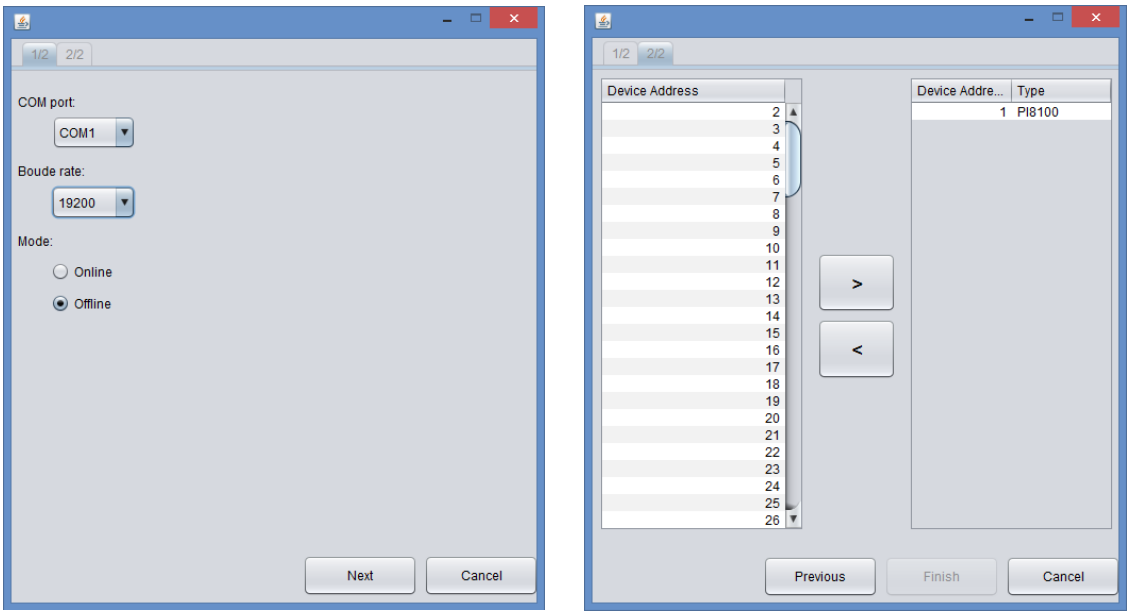
Obr. 5-5 Okno pro úpravu časové osy

4. Nastavení

Záložka `Nastavení` pomáhá uživateli s nastavením režimu komunikace, se změnou nastavení parametrů komunikace a s přidáním či odebráním zařízení.

Režim komunikace lze nastavit v položce `Mód`, která obsahuje dvě binární položky `Online` a `Offline`. Tyto položky se vzájemně vylučují, do online módu lze přepnout pouze, pokud je připojen sériový port, který je v projektu nastaven. Aplikace v offline módu nabízí všechny možnosti, jako v online módu, nedochází v něm pouze ke komunikaci se zařízeními ani k monitorování dat.

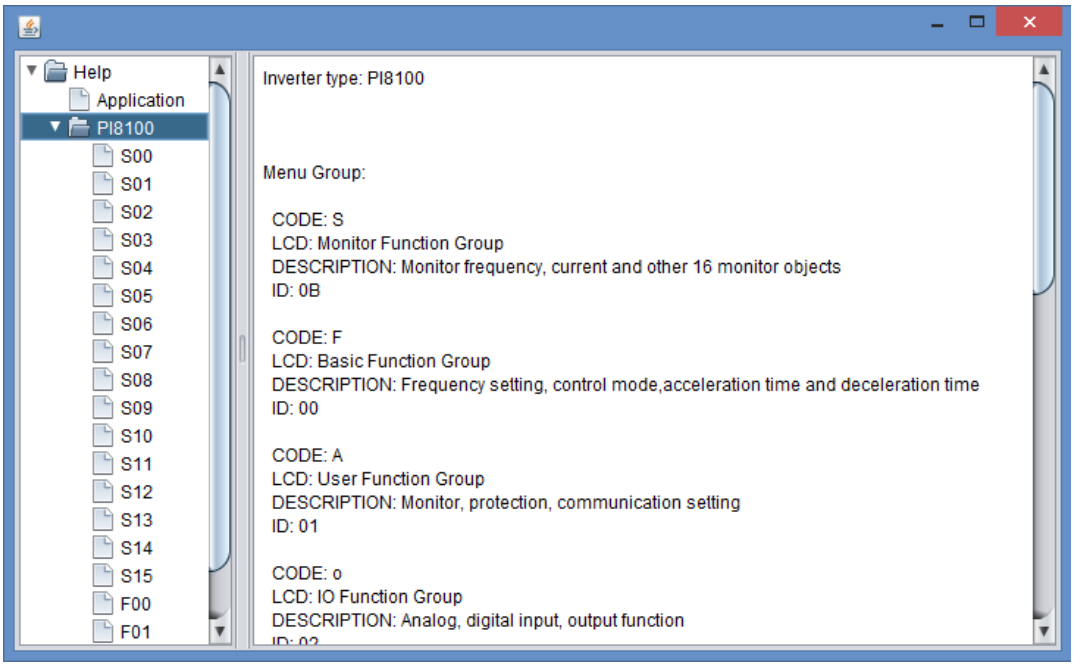
Změna parametrů komunikace a přidání nebo odebrání zařízení umožňuje položka `Komunikace`. Tato položka vyvolá okno podobné průvodci nastavením nového projektu. Uživatel nejprve vybere sériový port, rychlost a mód komunikace, ve druhém kroku provede uživatel správu zařízení, přidání, odebrání nebo ponechání stávajících zařízení. Po dokončení tohoto průvodce se uloží změny do struktury projektu.



Obr. 5-6 Průvodce změnou nastavení komunikace

5. Pomoc

Záložka `Pomoc` slouží pro základní popis aplikace a popis funkcí jednotlivých typů frekvenčních měničů. Data se zobrazují ve vlastním okně a jsou načítána pomocí dat uložených v externích souborech (zdrojové soubory k jednotlivým typům frekvenčních měničů).



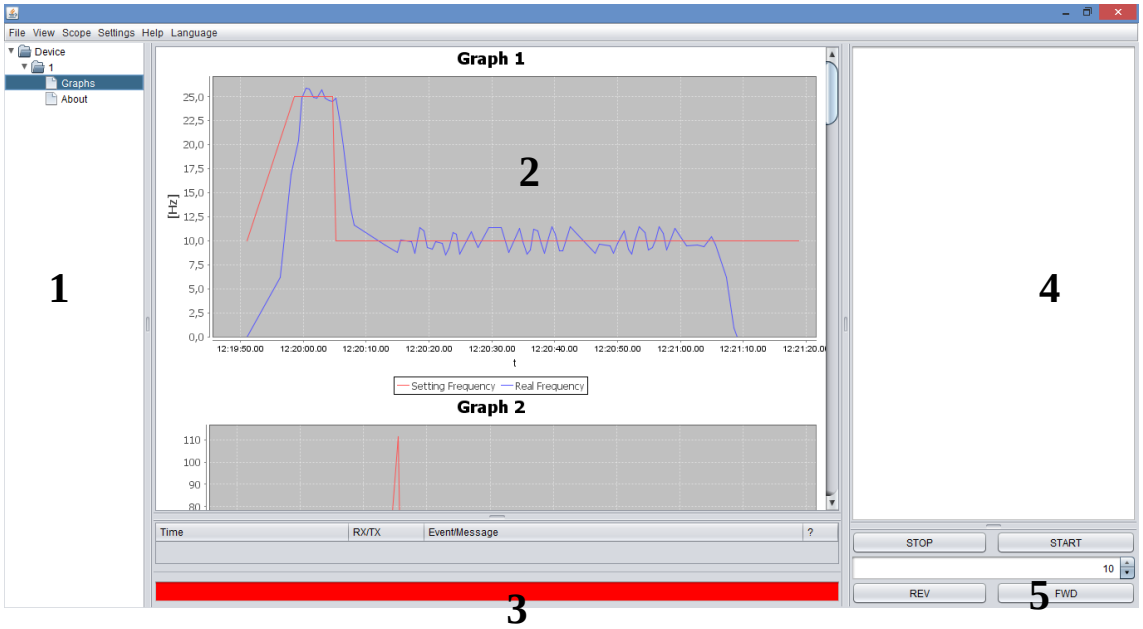
Obr. 5-7 Ukázka okna Pomoc

6. Jazyk

Záložka `Jazyk` slouží pro výběr jazykové modifikace aplikace (kap. 4.2), po výběru jazyka dojde k nastavení textů všech prvků, u kterých dochází k jazykové změně.

5.4 Pracovní plocha

Pracovní plocha je vizualizační prostor pro zobrazení všech důležitých prvků, které mají uživateli pomoci při práci s aplikací a poskytují mu vizuální zpětnou vazbu. Pracovní plocha je rozdělena do pěti panelů (Obr. 5-8). Každý z těchto panelů je určen pro jiný grafický objekt. V levé části se nachází panel se stromovou strukturou zařízení (Obr. 5-8 - 1). V prostřední části nalezneme dva panely, příkazovou řádku (Obr. 5-8 - 3) a panel pro zobrazení grafů a informací o zařízení (Obr. 5-8 - 2). V pravé části se nacházejí panel pro zobrazení textového výpisu (Obr. 5-8 - 4) posledních naměřených hodnot na zařízení a kontrolní panel (Obr. 5-8 - 5).



Obr. 5-8 Rozdělení hlavního okna

1. **Panel se stromovou strukturou**

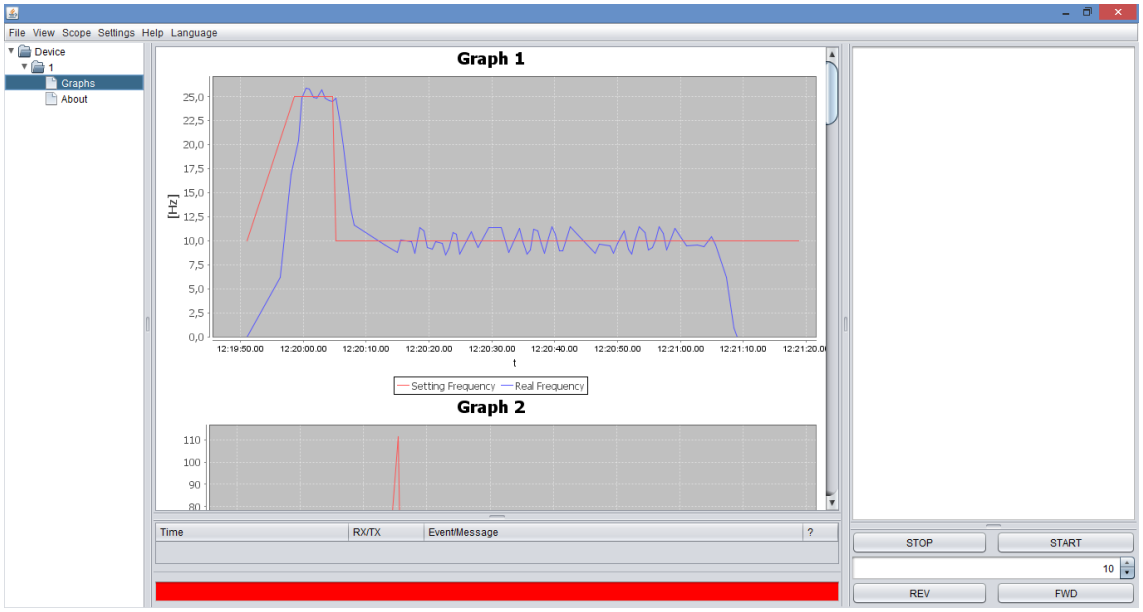
V tomto panelu se nachází stromová struktura všech zařízení, která uživatel do projektu vložil. Každé zařízení má v sobě dvě položky, graf a informace o zařízení. Stromová struktura slouží k přepínání mezi zařízeními, se kterými se komunikuje. Zároveň ovlivní i zobrazení v hlavním okně v závislosti na vybrané záložce. Každá změna se projeví v jednotlivých funkcích zejména v automatické komunikaci. Všechny ostatní panely pracují s aktuálně vybraným zařízením.

2. **Panel pro zobrazení grafů a informací**

V tomto panelu se zobrazí buď grafy s měřenými hodnotami zařízení, nebo informace o zařízení, které je aktuálně vybráno ve stromové struktuře a v závislosti na položce ve stromové struktuře, která byla vybrána.

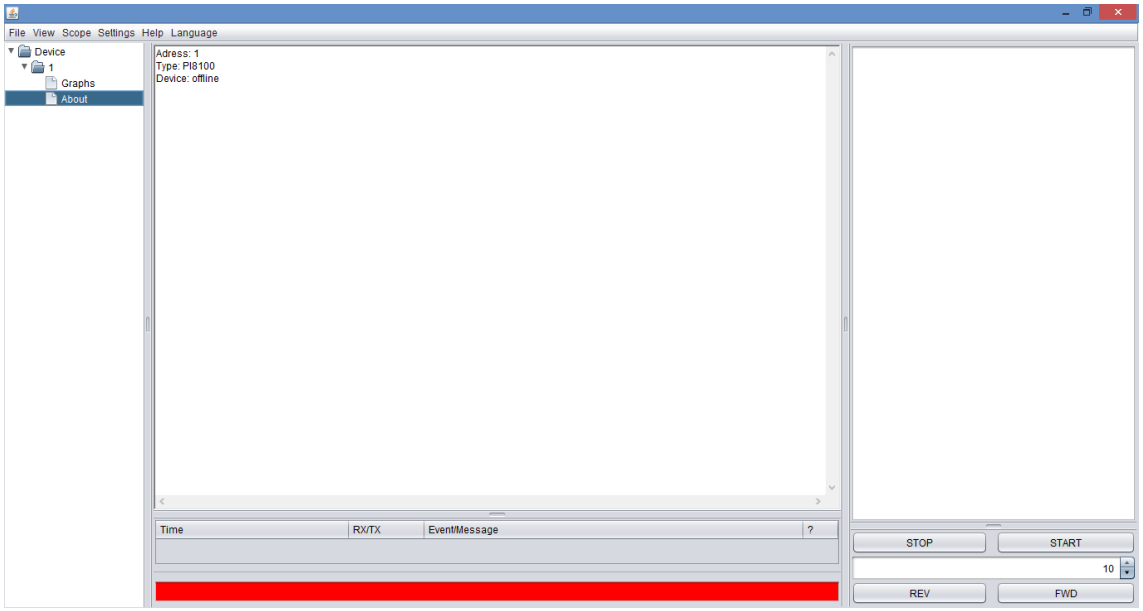
Pro zobrazení grafů je vhodné použít externí knihovnu JFreeChart **[13]**, která umožňuje snadnější zobrazení naměřených dat. Tato knihovna nabízí nespočet grafů, koláčové, histogramy, XY grafy aj. Pro zobrazení naměřených hodnot je nejvhodnější použít XY graf s časovou osou, který po vytvoření požaduje vstupní data, jimiž jsou uspořádané dvojice času a hodnota. Hodnota je vypočtena z naměřené hodnoty (přijatých dat, která jsou převedena na číslo typu *double*) a čas[2] je k nim přiřazen při zápisu uspořádané dvojice. Výhoda XY grafu s časovou osou je možnost formátování časové osy a zobrazení více průběhů do jednoho grafu. Časová osa je formátována do formátu *hodina:minuta:sekuna.milisekunda*. Do grafu jsou tedy zapisována data i se dnem, měsícem a rokem, ale nejsou zobrazeny v grafu, tuto skutečnost musíme zohlednit v ukládání grafů a případném pracování s daty v externím programu.

[2] Systémový čas Javy



Obr. 5-9 Ukázka grafu v offline režimu

Informace o zařízení zobrazí adresu, typ a informaci, zda je zařízení připojeno ke sběrnici.



Obr. 5-10 Ukázka informací o zařízení v offline režimu

3. Panel s příkazovou řádkou

Panel s příkazovou řádkou slouží k ručnímu zadávání příkazů a k zobrazení odeslaných zpráv uživatelem a odpovědí od zařízení. Panel se dá rozdělit na dvě části.

První část tvoří tabulka, ve které je zobrazena historie příkazů a je možné z ní vyvolat informace k danému příkazu. Tyto informace se zobrazují v samostatném okně při kliknutí na poslední sloupec v určitém řádku tabulky. Celý odeslaný rámec je pak rozebrán na jednotlivé části a pomocí dat načtených ze zdrojového souboru k danému zařízení jsou popsány.

Druhá část je samotný příkazový řádek, do kterého je možné zapsat příkaz v hexadecimálním tvaru, tento tvar je neustále kontrolován a odeslání příkazu je možné pouze v případě, že se aplikace nenachází v offline módu a není zadán špatný znak. Pokud se bude nacházet aplikace v offline režimu nebo bude zadán špatný znak nelze tedy odeslat zprávu a navíc se příkazový řádek podbarví červeně na znamení chyby (či offline režimu). Po odstranění chyby podbarvení zmizí.

4. Monitorovací panel

Monitorovací panel slouží pro textové výpisy naměřených hodnot, jedná se o hodnoty, které jsou zapisovány do grafů. Umožňuje to uživateli sledovat aktuální velikost všech sledovaných parametrů, uživatel tedy není závislý jen na grafickém zobrazení hodnot.

5. Kontrolní panel

Kontrolní panel je složen z prvků, které umožní nastavení nejzákladnějších parametrů měniče. Jedná se o čtyři tlačítka a jeden číselný panel. Tlačítka podporují nastavení směru otáčení motoru (vpřed, vzad) a připojení či odpojení výstupu. Číselný panel umožňuje nastavení frekvence výstupního napětí. Po stisku tlačítka je odeslána do příslušného aktivního zařízení zpráva definovaná ve zdrojovém souboru k danému zařízení.

5.5 Komunikace

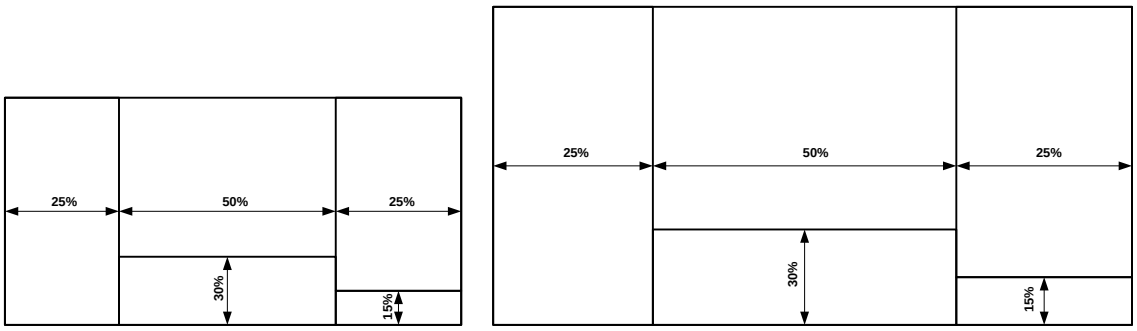
Komunikace mezi frekvenčním měničem a počítačem probíhá pomocí sériové linky, na straně frekvenčního měniče se jedná o RS-485 a na straně počítače je převodník RS-485/USB (FTDI čip). Počítač s tímto převodníkem pracuje přímo jako se sériovou linkou po nainstalování ovladačů. V Javě se dá s tímto převodníkem pracovat po vložení externí knihovny **[12]** (na stránkách jsou i příklady použití knihovny, nebo můžeme použít již existující software **[14]**, který přizpůsobíme svým potřebám).

Ke komunikaci je v programu vytvořena vlastní třída, která obsahuje inicializaci sériového portu a funkce potřebné ke komunikaci. Pokud je inicializován sériový port, musí třída rovněž obsahovat funkci, která sériový port opět uvolní. Funkce pro odesílání a příjem jsou obsluhovány pouze přes funkci *SendMessage*, kterou využívají ostatní části programu, pokud chtějí komunikovat se zařízeními. Tato funkce má čtyři parametry, zprávu, která má být odeslána (zpráva = adresa + kód funkce + data), očekávanou délku zprávy při kladné odpovědi. Dále pak obsahuje binární hodnotu (*true/false*), zda se má zobrazit odesílaná zpráva v příkazové řádce, a binární hodnotu (*true/false*), zda se má zobrazit přijatá zpráva v příkazové řádce. Před odesláním zprávy je vypočítán CRC součet, který je ke zprávě přidán, aby byl vytvořen kompletní rámec.

Při komunikaci musí být vyloučena chyba souběhu, k sériovému portu nesmí přistupovat dvě vlákna současně, aby nedocházelo k chybám. Tuto možnost vyloučíme vzájemným vyloučením (synchronizací) kritických sekcí (kap. 5.2).

5.6 Aktivní hlídání grafických prvků

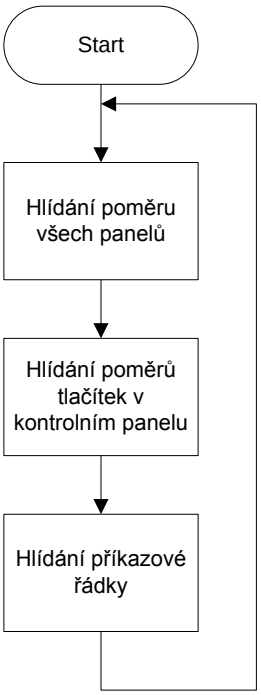
Aktivní hlídání grafických prvků je samostatně běžící vlákno spuštěné po spuštění grafického rozhraní aplikace. Má za úkol zachovávat velikost panelů v hlavním okně, aby například nedocházelo k přílišnému roztažení stromové struktury přes celé okno. Zároveň se stará o zachování poměrů velikosti panelů vzhledem k celkové velikosti okna. To znamená, že pokud dojde ke zvětšení či zmenšení hlavního okna, zvětší či zmenší se i panely, která jsou v něm zobrazeny a to při zachování poměru k velikosti okna (viz Obr. 5-11).



Obr. 5-11 Ukázka zachování poměrů panelů v okně

Kromě hlídání poměrů všech panelů, upravuje poměrové rozdělení v kontrolních panelech tak, aby tlačítka měla vždy 50% šířku přiděleného panelu.

V příkazovém řádku kontroluje zadaný řetězec uživatelem, pokud je chyba v zadaném řetězci podbarví se příkazová řádka červeně na znamení chyby a zablokuje se odeslání zprávy. V případě, že je řetězec správně zadaný, tak není příkazová řádka podbarvena. K podbarvení příkazové řádky může dojít ještě v jednom speciálním případě. Nachází-li se aplikace v offline módu je příkazová řádka podbarvena trvale červeně a je zablokována možnost odeslání zprávy. Zároveň dochází k poměrovému rozdělení buněk tabulky v závislosti na velikosti okna, v této tabulce se zobrazují zprávy odeslané a přijaté uživatelem.



Obr. 5-12 Zjednodušený vývojový diagram pro vlákno hlídající grafické prvky

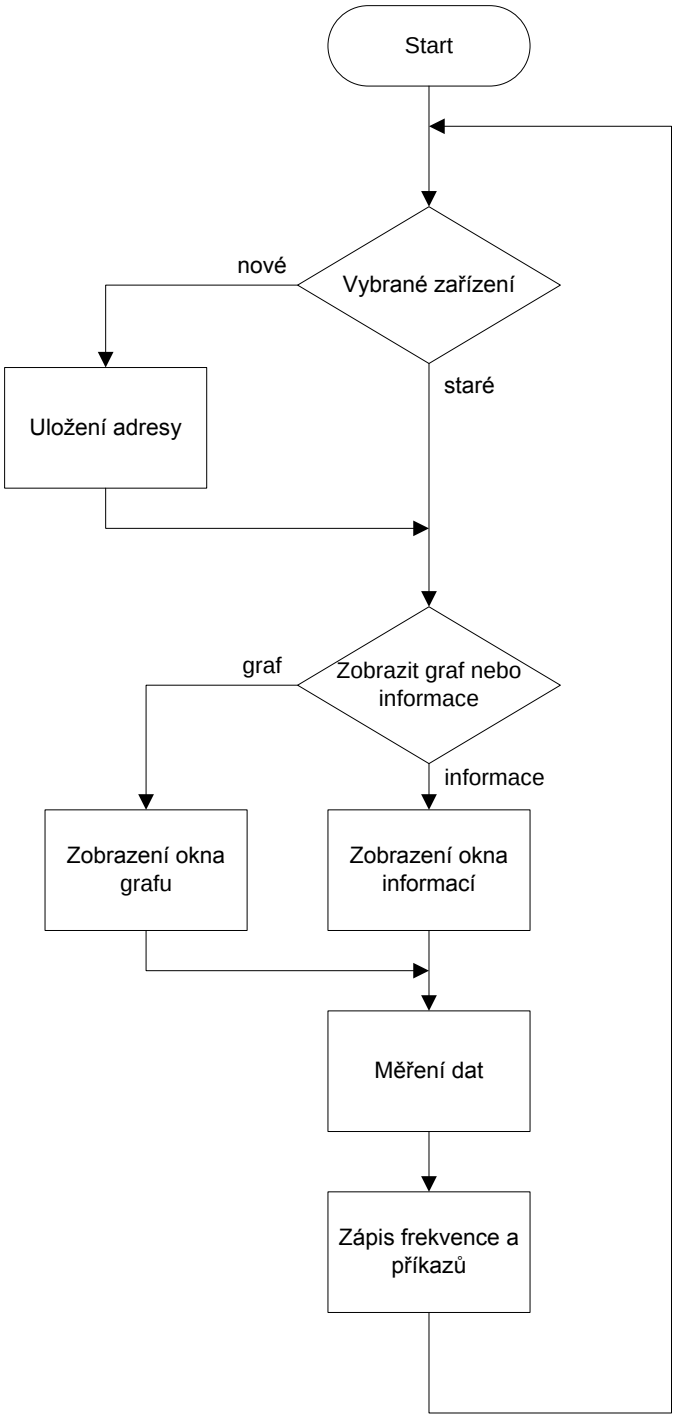
5.7 Měřící vlákno

Měřící vlákno slouží k automatickému získávání dat ze zařízení. Aktuálně vybrané zařízení je cyklicky dotazováno na monitorované funkce a z odpovědi, pokud je v pořádku, jsou vypočítány hodnoty. Příkaz je získán z externího souboru k danému zařízení a odpovídá monitorované funkci. Tyto funkce jsou pevně dány typem zařízení a jsou definovány ve zdrojovém souboru zařízení. Uživatel má možnost vybrat v záložce `Graf`, které funkce mají být monitorovány. Funkce, které nejsou vybrány, jsou v cyklickém monitorování přeskakovány a nejsou tudíž dotazovány. K dotazování dochází pouze v online módu.

Vypočtené hodnoty jsou zapisovány do paměti (do struktury odpovídající danému zařízení na příslušné adrese), zároveň se zapsáním do struktury jsou data automaticky zobrazena i v grafu a následně v monitorovacím panelu.

V tomto vlákně se kontroluje stav vybraného zařízení ve stromové struktuře a vybraná záložka, která se má zobrazit (graf, informace). To má za následek výběr zařízení, které je monitorováno, a zobrazení panelu v hlavním okně aplikace.

Měřící vlákno používá k měření třídu určenou ke komunikaci (kap. 5.5), u které nedochází k zobrazování odchozích a příchozích zpráv. Zprávy jsou před výpočtem kontrolovány a v případě zjištění chybné či poškozené zprávy jsou data zahozena.



Obr. 5-13 Zjednodušený vývojový diagram pro měřící vlákno

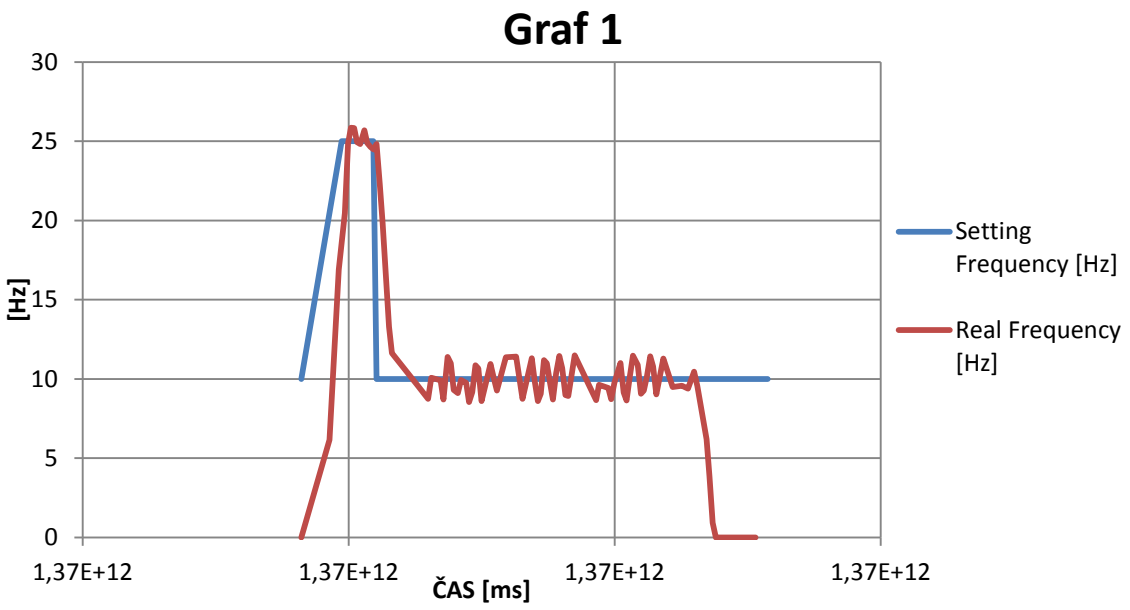
6 Závěr

6.1 Funkčnost aplikace

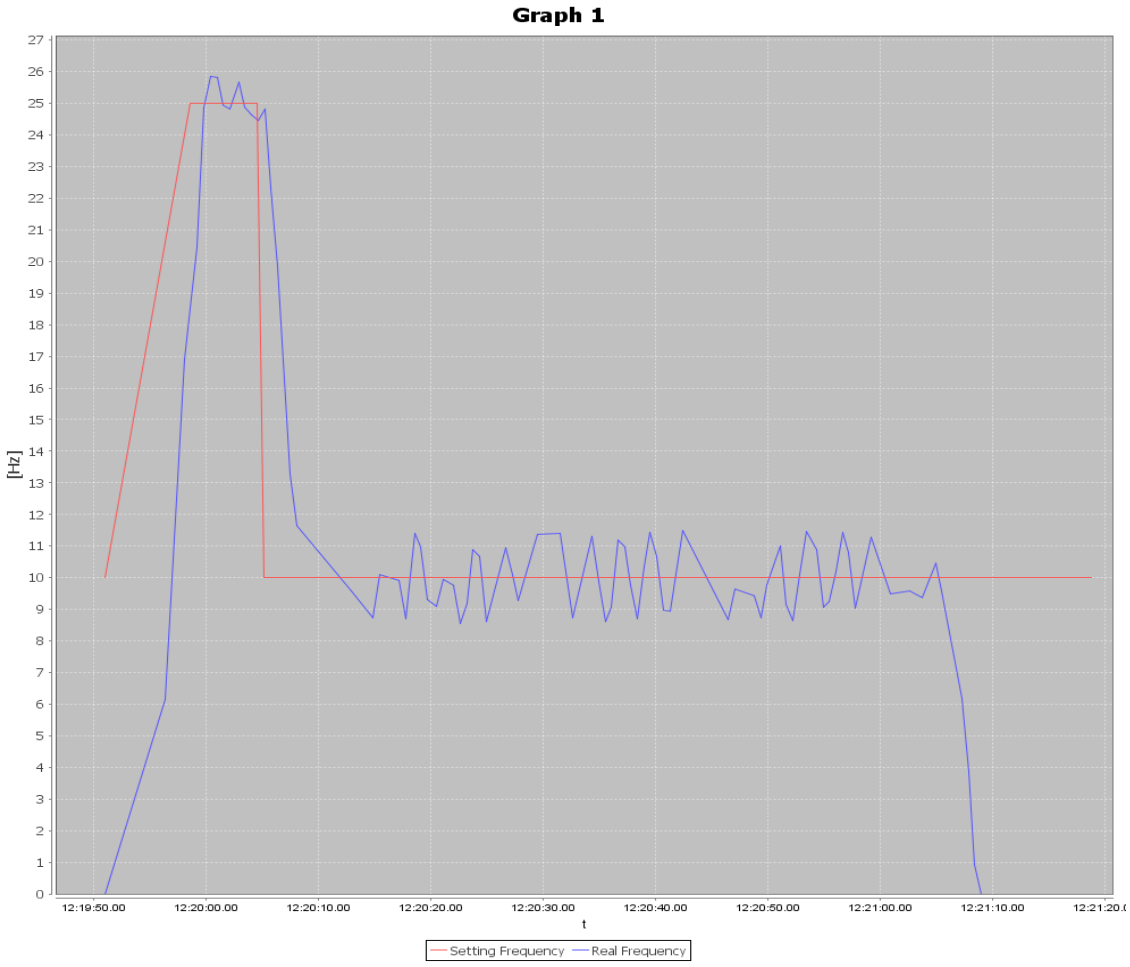
Vytvořená aplikace je funkční. Je schopná komunikovat se zařízením Powtran PI8100 pomocí sériového portu při použití komunikačního protokolu MODBUS. Naměřená data jsou zpracována a zobrazena uživateli.

Externí datové soubory jsou načítány a používány ke komunikaci a k dalším využitím v aplikaci. Prozatím však externí datové soubory, konkrétně datový soubor pro frekvenční měnič, není kompletní, z důvodu rozsáhlosti dat, která je nutná přepsat. Aktuální rozsah tohoto souboru je pouze pro testování funkčnosti aplikace.

Správa dat celého projektu a hodnot získaných měřením je bez problému v ukládání i načítání do souborů (viz Obr. 6-1 a Obr. 6-2). Ostatní možnosti aplikace se jeví jako funkční.



Obr. 6-1 Ukázka grafu vytvořeného v Excelu



Obr. 6-2 Ukázka grafu uloženého ve formátu PNG

6.2 Testování aplikace

Testování aplikace probíhalo prozatím jen v laboratorních podmínkách. Po napsání části zdrojového kódu ovlivňujícího chod aplikace byl proveden test nejprve na simulátoru a následně na reálném zařízení. Simulátor tvořil FTDI převodník při propojení (zkratování) pinů TX a RX a sledování odchozích a příchozích zpráv. Test na reálném zařízení tvořilo otestování funkcí na připojeném frekvenčním měniči Powtran PI8100 s asynchronním motorem. Veškeré chyby objevené při těchto testech byly odstraněny. Další testování by mělo probíhat ve spolupráci se třetí stranou (uživateli) a na základě zpětné vazby by mělo docházet k odladění programu.

Testování by v sobě mělo zahrnovat i adresaci všech zařízení 0x00, při připojení více zařízení na sběrnici. Testování aplikace probíhalo pouze na jednom zařízení, které bylo připojeno na sběrnici.

6.3 Možnosti rozšíření aplikace

Možnost rozšíření aplikace představuje hlavně přidání dalších externích souborů pro jiná zařízení a přidání těchto souborů ve zdrojovém kódu, tak aby s nimi aplikace mohla pracovat. Důležité je u těchto souborů udržet navržený formát (kap. 4).

Další možné rozšíření je v oblasti grafického rozhraní a přidání dalších automatických funkcí, na základě zpětné vazby od uživatelů.

Literatura

[1] JANOUŠEK, Josef; VOŽENÍLEK, Petr. *Základy silnoproudé elektrotechniky*, ČVUT, Praha, 2006

[2] JANOUŠEK, Josef; Petrásek, František; VOŽENÍLEK, Petr. *Základy silnoproudé elektrotechniky – Laboratorní cvičení*, ČVUT, Praha, 2005

[3] JELÍNEK, Richard. *Frekvenční měniče – EMC a použití příslušenství* [online], Elektroinstalatér č. 5, 2005, Dostupné z WWW: <http://www.tzb-info.cz/2953-frekvencni-menice-emc-a-pouziti-prislusenstvi>

[4] PAVELKA, Jiří. *Elektrické pohony*, ČVUT, Praha, 2007

[5] FRODL, Martin. *Řízení pohonu asynchronního motoru pomocí PLC*, ČVUT, Praha, 2011

[6] KOCMAN, Stanislav, *Asynchronní stroje* [online], FEI VŠB-TU, Ostrava, 2002, Dostupné z WWW: < http://p.kobrle.sweb.cz/stroje/as-skriptum.pdf>

[7] GAJDŮŠEK, Pavel. *Přehled metod řízení asynchronních motorů* [online], FEKT VUT, Brno, 2005, Dostupné z WWW: <http://www.elektrorevue.cz/clanky/05020/index.html#2_BSR>

[8] *Obrázek frekvenčního měniče Powtran PI8100* [online], Dostupné z WWW: <http://www.electricmotorshop.co.uk/index.asp?function=PRODUCTIMAGEWINDOW&closedelay=20000&SRC=http://www.electricmotorshop.co.uk/ekmps/shops/kbrewinds/images/powtran-pi8100-4.0kw-inverter-1-phase-input-3-phase-output-859-p[ekm]288x288[ekm].jpg>

[9] RONEŠOVÁ, Andrea. *Přehled protokolu MODBUS* [online], Plzeň, květen 2005, Dostupné z WWW: < http://home.zcu.cz/~ronesova/bastl/files/modbus.pdf>

[10] *Manuál k frekvenčnímu měniči POWTRAN PI8000/PI8100* [online], Dostupné z WWW:
< http://www.cyberma.cz/sites/default/files/Images/Manual%20PI8000-ENG.pdf>

[11] BRADLEY, Neil. *XML Kompletní průvodce*, Grada Publishing, Praha, 2000

[12] *Knihovna pro práci s FTDI převodníkem* [online], Dostupné z WWW:
<http://rxtx.qbang.org/wiki/index.php/Main_Page>

[13] *Knihovna pro práci s JFreeChart* [online], Dostupné z WWW:
<http://www.jfree.org/jfreechart/>

[14] ADÁMEK, Petr. *Klasické i moderní řízení spínaných měničů*, Praha, 2011, Dostupné z WWW:
<https://support.dce.felk.cvut.cz/mediawiki/images/f/f0/Bp_2011_adamek_petr.pdf>

[15] *Práce s tabulkami v Javě* [online], Dostupné z WWW:
<http://docs.oracle.com/javase/tutorial/uiswing/components/table.html>

[16] *Práce s externí knihovnou pro Excel v Javě* [online], Dostupné z WWW:
<http://www.java-tips.org/other-api-tips/jexcel/reading-data-from-excel-documents-at-runtime-4.html>

[17] *Ukládání obrázku v Javě* [online], Dostupné z WWW:
<http://www.javamex.com/tutorials/graphics/bufferedimage_save_png_jpeg.shtml>

[18] *Výpočet CRC16 MODBUS* [online], Dostupné z WWW:
<http://stackoverflow.com/questions/3215328/crc-16-ansi-library-for-for-java-groovy>

[19] *Použití Sax parseru* [online], Dostupné z WWW:
<http://www.javacodegeeks.com/2012/01/xml-parsing-using-saxparser-with.html>

Obsah přiloženého CD

- Diplomová práce v elektronické podobě
- Aplikace
- Zdrojové kódy
- Externí datové soubory