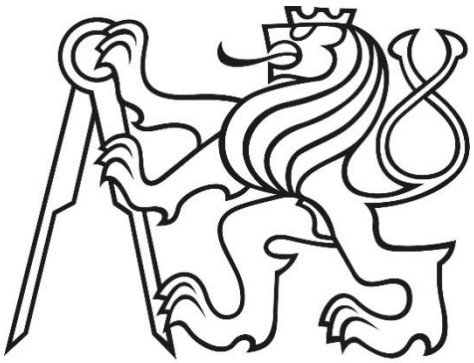


ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA DOPRAVNÍ

Václav Rada

SOFTWARE PRO ŘÍZENÍ STROJŮ A EXPERIMENTÁLNÍCH ZAŘÍZENÍ

Bakalářská práce

2017

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
d ě k a n
Konviktská 20, 110 00 Praha 1

K618 .....Ústav mechaniky a materiálů

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Václav Rada

Kód studijního programu a studijní obor studenta:

B 3710 – DOS – Dopravní systémy a technika

Název tématu (česky): **Software pro řízení strojů a experimentálních zařízení**

Název tématu (anglicky): Control Software for Machines and Experimental Devices

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte osnovou uvedenou v následujících bodech:

- V rámci ústavu K618 jsou vyvíjena a využívána experimentální zařízení vlastní konstrukce pracující zpravidla na stejných principech jako zařízení pro průmyslovou automatizaci a robotiku. V rámci této práce vznikne univerzální softwarový nástroj pro řízení těchto zařízení vytvořený v programovacím jazyce Python a kompatibilní s platformou LinuxCNC. Práce navazuje na přechozí studentské projekty a má vhodně doplnit vyvinuté modulární řídicí prvky o kvalitní softwarovou platformu.
- Cílem práce je vytvořit softwarový nástroj, který bude využit pro řízení experimentálních zařízení vlastní konstrukce. Řešení by se mělo skládat z rámcového ovládacího rozhraní obsahující prvky společné pro všechna zařízení (např. referenční poloha, koncové spínače, ruční posuvy) a vybraných modulů pro ovládání specifických funkcí, včetně zdrojových kódů využitelných při tvorbě dalších modulů a rozšíření.
- Funkčnost jednotlivých prvků bude ověřena použitím softwaru pro řízení zvoleného experimentálního zařízení nebo stroje.

Rozsah grafických prací: nebyl stanoven

Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury:

LinuxCNC Documentation

http://linuxcnc.org/docs/2.7/pdf/

Gürocak H., Industrial Motion Control, Wiley, 2015, DOI: 10.1002/9781118403211

Summerfield M., Rapid GUI Programming with Python and QT, Prentice Hall, 2007, ISBN: 0132354187

Vedoucí bakalářské práce: **Ing. Petr Zlámal, PhD.**
Ing. Tomáš Fíla
Ing. Nela Krčmářová

Datum zadání bakalářské práce: **3. října 2016**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **28. srpna 2017**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

L. S.

prof. Ing. Ondřej Jiroušek, Ph.D.
vedoucí
Ústavu mechaniky a materiálů

prof. Dr. Ing. Miroslav Svítek, dr. h. c.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

Václav Rada
jméno a podpis studenta

V Praze dne ..... 3. října 2016

Poděkování

Na tomto místě bych rád poděkoval všem, kteří mi poskytli podklady pro vypracování této práce. Zvláště pak děkuji Ing. Petru Zlámalovi, Ph.D., Ing. Tomáši Fílovi a Ing. Nele Krčmářové za odborné vedení a konzultování bakalářské práce a za rady, které mi poskytovali po celou dobu mého studia. Dále pak Ústavu teoretické a aplikované mechaniky AV ČR, v. v. i. a Ústavu mechaniky a materiálů Fakulty dopravní ČVUT, kde mi bylo poskytnuto zázemí pro vznik této práce a zároveň odborná pomoc od jejich pracovníků. V neposlední řadě je mou milou povinností poděkovat svým rodičům a blízkým za morální a materiální podporu, které se mi dostávalo po celou dobu studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací. Nemám žádný závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 24. srpna 2017

_____

Václav Rada

Název práce:	Software pro řízení strojů a experimentálních zařízení
Autor:	Václav Rada
Studijní program:	Technika a technologie v dopravě a spojích
Obor:	Dopravní systémy a technika
Druh práce:	Bakalářská práce
Vedoucí práce:	Ing. Petr Zlámal, Ph.D. Ing. Tomáš Fíla Ing. Nela Krčmářová
Rok vydání:	2017

Abstrakt: Na Ústavu mechaniky a materiálů Fakulty dopravní ČVUT jsou prováděny experimenty pokročilými experimentálními metodami, což vyžaduje vývoj experimentálních zařízení vlastní výroby. Předložená práce se zabývá vývojem modulárního řídicího software v jazyce Python na platformě systému LinuxCNC s využitím rozhraní Python Interface, kterým je možné tato zařízení ovládat. Řídicí software byl vytvořen jako množina samostatných pluginů (zásuvných modulů), které jsou do aplikace připojitelné, a tím lze modifikovat funkce řídicího software a efektivně jej přizpůsobit pro konkrétní zařízení. Řídicí software umožňuje ovládat zařízení s polohováním pracujícím na principu krokového motoru, popř. servomotoru, ovládat periferie a vyčítat veličiny měřené během experimentu (signál typu mV/V). Software obsahuje také řadu bezpečnostních mechanismů (např. zastavení stroje při přetížení siloměru apod.). Nakonec byl software implementován do řídicí jednotky, která disponuje veškerou hardwarovou funkcionalitou, a byl proveden pilotní experiment, aby byla ověřena funkčnost jednotlivých pluginů a správnost pořízených dat.

Klíčová slova: CNC, řízení, LinuxCNC, Python Interface, Python, krokový motor

Title: Control Software for Machines and Experimental Devices

Abstract: At the Department of Mechanics and Materials of the Faculty of Transportation Sciences, CTU are made experiments using advanced experimental methods and custom-based experimental devices are required. In this thesis, development of a modular control software, which allows for control the custom experimental devices, is described. The control software was developed in Python programming language using LinuxCNC control system and its Python Interface. The control software is designed as a set of separate plugins, which can be imported to the application to manage requirements and specifics of a particular experimental device. The control software is capable of controlling devices actuated by stepper motors or servo-motors, handling peripherals and obtain measured data during the experiment (mV/V signal type). The control software also provides many security mechanisms (e.g. stopping the machine when the load cell is overloaded etc.). The software was implemented into a control unit, which provides hardware functionality and a pilot experiment was performed to verify software functionality and measured data precision.

Keywords: CNC, controlling, LinuxCNC, Python Interface, Python, stepper motor

Obsah

Seznam obrázků

Seznam použitých zkratek

CNC	Computer Numerical Control	*číslicové řízení*
GUI	Graphical User Interface	*grafické uživatelské rozhraní*
RTAI	RealTime Application Interface	*real-time rozšíření systému Linux*
HAL	Hardware Abstraction Layer	*abstrakční hardwarová vrstva*
XML	eXtensible Markup Language	*rozšiřitelný značkovací jazyk*
CPU	Central Processing Unit	*centrální procesorová jednotka*
LPT	Line Printer Terminal	*paralelní port*
USB	Universal Serial Bus	*univerzální sériová sběrnice*
MDI	Manual Data Input	*manuální datový vstup*

1 Úvod

Číslicově řízené (Computer Numerical Control – CNC) stroje a zařízení prochází v současné době velkým vývojem a jsou využívány v nejrůznějších odvětvích (strojírenství, lékařství atd.). Ve strojírenství se jedná typicky o velká obráběcí zařízení s posuvem os v řádu centimetrů (někdy i metrů), ale své využití mají i zařízení s posuvem os v řádu mikrometrů. Takto přesná zařízení mohou nalézt uplatnění v laboratorních sestavách, například k experimentálnímu zjišťování materiálových vlastností.

V této oblasti naleznou tato zařízení využití v celém spektru aplikací a mohou být využívána např. pro vysoce přesné testování materiálů na mikrometrické úrovni (zatěžování jednotlivých prvků mikrostruktury), nedestruktivní testování (měření tvrdosti) nebo pro pokročilé experimentální metody v mechanice (měření v počítačovém tomografu). Další velkou výhodou těchto zařízení je, že mohou experimenty provádět automatizovaně s využitím řídicích programů. Jako příklad lze uvést automatizované měření tvrdosti v mnoha bodech vzorku. Přesnost experimentů závisí na přesnosti polohování os stroje, ale také na přesnosti vyčítání fyzikálních veličin (například posuv, síla apod.). Rozvoj výrobních technologií senzorů vedl ke zpřesnění vyčítání fyzikálních veličin a pokrok v oblasti výpočetní techniky umožnil dostupnost zařízení s dostatečným výpočetním výkonem.

Na Ústavu mechaniky a materiálů Fakulty dopravní ČVUT jsou k provádění experimentů s výhodou používána číslicově řízená zařízení. Jsou to zejména zatěžovací zařízení (například mikroindentor), nebo polohovací zařízení (například stolice pro polohování optické soustavy). Tato zařízení se skládají z funkčně podobných komponent (například motorizované osy poháněné krokovými elektromotory, siloměry, koncové spínače apod.). Každé zařízení je ale určené k provádění jiného druhu experimentu, a proto mají i určité komponenty odlišné (různý počet motorizovaných os, různé principy vyčítání fyzikálních veličin, různé typy siloměrů apod.). Řídicí software musí tyto specifické potřeby zohledňovat (například tlačítka, zobrazovací oblasti apod.) a je nutné již ve fázi jeho návrhu k tomuto přihlížet. Jedno z možných řešení je vybudovat řídicí software na stabilním jádře s modulárními ovládacími prvky. Takové řešení umožňuje jednoduše a rychle implementovat potřebné ovládací prvky, popřípadě upravit jejich vlastnosti, a tím je efektivně přizpůsobit pro dané zařízení a pro daný experiment.

V rámci této práce byl vytvořen řídicí software s modulárními ovládacími prvky pro řízení experimentálních zařízení na Ústavu mechaniky a materiálů. V textu práce jsou tento software i všechna dílčí programová řešení popsány. Funkčnost softwaru byla ověřena při pilotním experimentu se zatěžovacím zařízením.

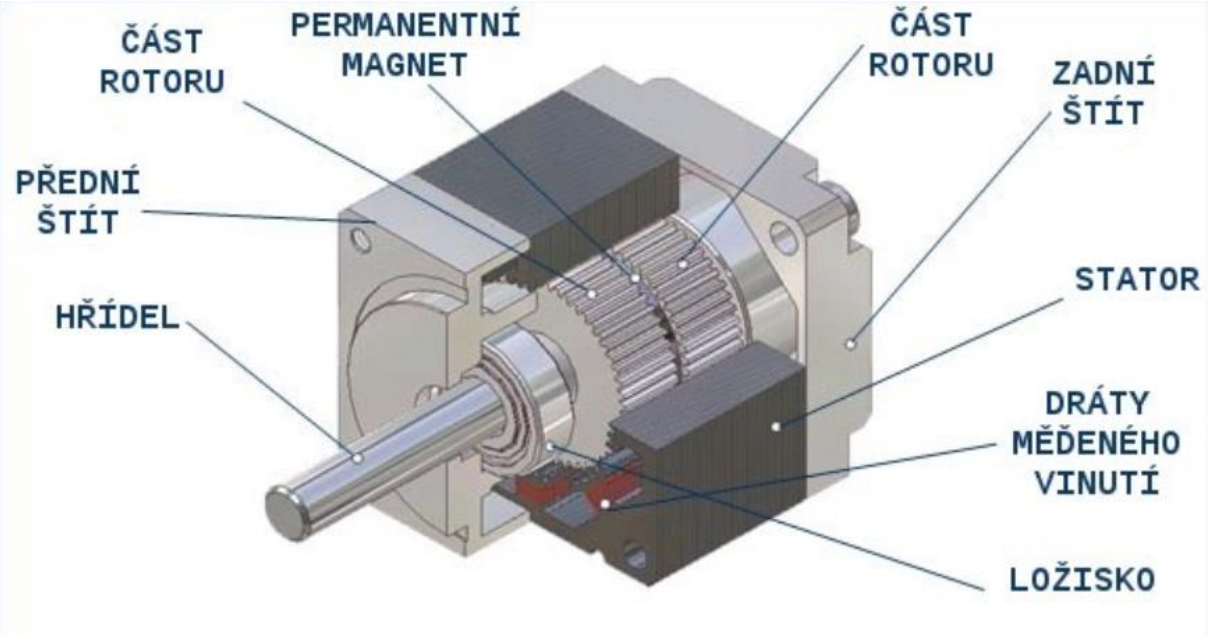
2 Teoretický úvod

Experimentální zařízení používaná na Ústavu mechaniky a materiálů Fakulty dopravní jsou zpravidla tvořena motorizovanými osami, siloměry a dalšími periferiemi. Jako motorizované osy jsou převážně používány jednotky vybavené krokovým motorem spojeným přes pružnou spojku s pohybovým šroubem s lineárním vedením. Pohyb jednotlivých os je monitorován enkodéry, které slouží ke kontrole pozice osy. K zajištění bezpečného provozu se používají koncové spínače a tlačítko nouzového zastavení E-STOP. K měření síly se používají siloměry převážně fungující na principu tenzometrických snímačů zapojených do Wheatstoneova můstku.

2.1 Krokové motory

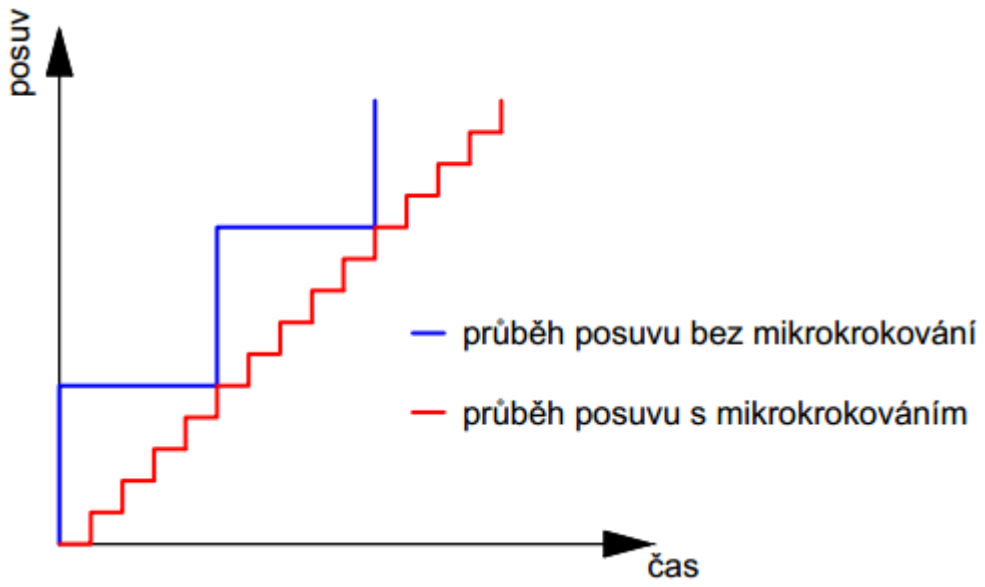
Základem krokového motoru je rotor tvořený permanentními magnety (póly) a stator tvořený cívkami. Průtokem elektrického proudu cívkami se kolem nich vytvoří magnetické pole, které otočí rotorem do nejbližší stabilní polohy. Při přepojení obvodu na vedlejší cívku dojde k opětovnému pootočení rotoru. Postupné přepojování cívek statoru tedy zajišťuje otáčivý pohyb.

Impulzy k přepojení cívek jsou posílány do motoru pomocí driveru motoru. Každý impulz má za následek pootočení motoru o určitý úhel. Tyto impulzy se nazývají kroky. K otočení motoru o celou jednu otáčku je potřeba několik stovek kroků (nejčastěji 200). Čím větší počet kroků je potřeba k otočení motoru o jednu otáčku, o to menší úhel se otočí na jeden krok. Tím se zvyšuje jeho přesnost. Na rozdíl od běžných elektromotorů proto může být krokový motor využit pro relativně jednoduché, přesné, stabilní a opakovatelné řízení polohy i bez zpětné vazby. Výhodou krokových motorů oproti klasickým servomotorům (elektromotorům vybavených zpětnovazebním enkodérem) je nižší cena, relativně nízké provozní stejnosměrné napětí (tj. nízké rušení měřených veličin) a jednoduchá řídicí elektronika. Hlavní nevýhodou ve srovnání se servomotorem je nižší krouticí moment, nižší dosažitelné otáčky a pevná poloha kroků.



Obrázek č. 1: Schéma krokového motoru, převzato z [2]

Protože pohyb motoru probíhá v krocích, jeho pohyb není spojitý, nýbrž skokový. Z toho důvodu umožňují moderní drivery tzv. mikrokrokování (microstepping). Účelem mikrokrokování je zvýšit plynulost otáčení motoru a zajistit hladší chod. Mikrokrokování lze hypoteticky použít i ke zvýšení přesnosti motoru za předpokladu, že odpory, které motor překonává, jsou v řádu jednotek procent jeho maximálního krouticího momentu.



Obrázek č. 2: Schématické znázornění mikrokrokování

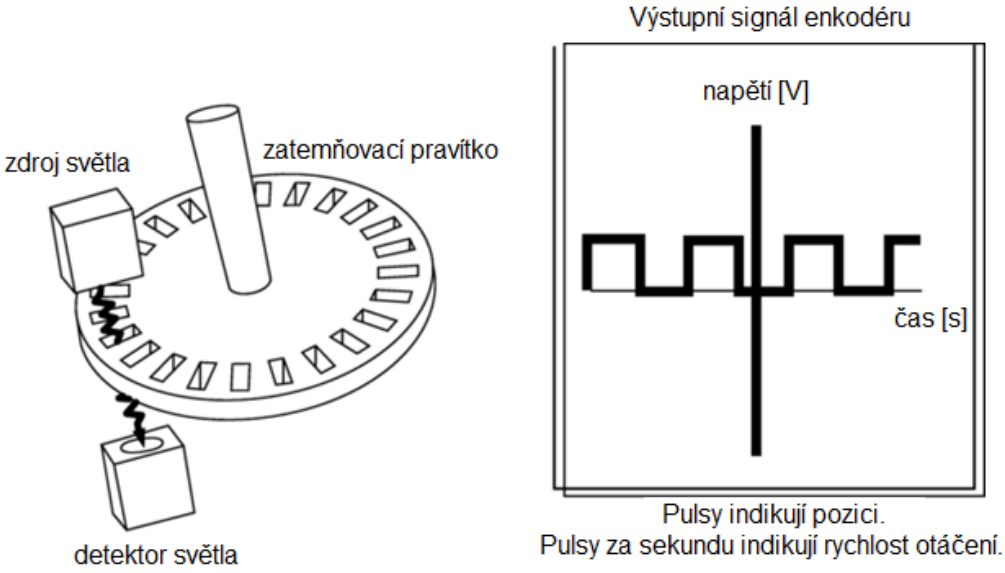
Při používání krokových elektromotorů může nastat situace, kdy krok nemusí být motorem vykonán, přestože driverem byl poslán impulz k přepojení cívek. Takový jev se

nazývá ztráta kroku. Nejčastější příčinou je nedostatečný krouticí moment motoru. Driver odvozuje polohu motoru podle počtu impulzů, které byly do motoru vyslány, a tím může určit úhel natočení. Ve chvíli, kdy impulz nezpůsobí pootočení motoru o krok, driver ztrátu kroku nezaznamená (jedná se o řízení bez zpětné vazby). Poloha vypočítaná driverem neodpovídá skutečné poloze motoru.

2.1.1 Enkodér

Enkodér je elektromechanická součástka, která je schopna detekovat pohyb motoru (rychlost, směr otáčení). Tuto informaci předává v podobě standardizovaného signálu do řídicího systému. Pro přesné řízení krokových motorů lze porovnávat polohu spočítanou driverem a polohu odečtenou enkodérem, a tím doplnit systém o zpětnou vazbu. Enkodéry se dělí podle fyzikálního principu (magnetické, odporové, optické) nebo podle typu výstupního signálu na analogové a digitální. Princip funkce je v rámci této práce demonstrován na jednoduchém optickém enkodéru.

Optické enkodéry fungují na principu detekce světla propouštěného přes zatemňovací pravítko. Při stabilní poloze motoru pravítko propouští světlo, které je detekováno detektorem (například fotorezistorem). Během provádění kroku dojde vlivem pohybu motoru k zamezení průchodu světla přes pravítko. Při dokončení kroku (uvedení motoru do další stabilní polohy) je opět umožněn průchod světla přes pravítko. Tím lze určit skutečný počet úspěšné provedených kroků.



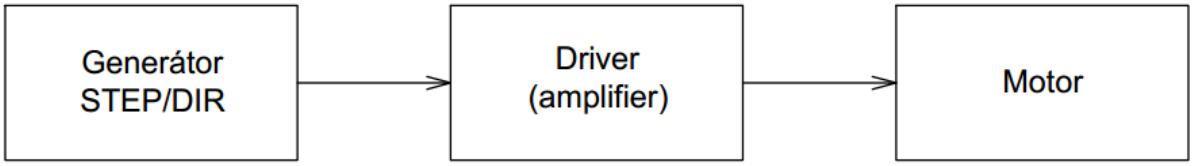
Obrázek č. 3: Schematicky znázorněný princip enkodéru, převzato a upraveno z [4]

Mimo enkodér, který zpětnovazebně určuje polohu osy, je pro zajištění bezpečnosti vhodné osy osadit koncovými spínači, aby osa nepřekročila své meze. Koncový spínač je součástka, kterou je zajištěno rozepnutí elektrického obvodu a zastavení pohybu v koncové poloze osy. Pokud by se pohyb osy nezastavil, může dojít k poškození zařízení. V případě, že nastane situace, kdy je potřeba neprodleně celý stroj zastavit a odvrátit hrozící nebezpečí, používá se zpravidla emergency-stop (tzv. E-STOP) tlačítko.

2.2 Řízení krokových motorů

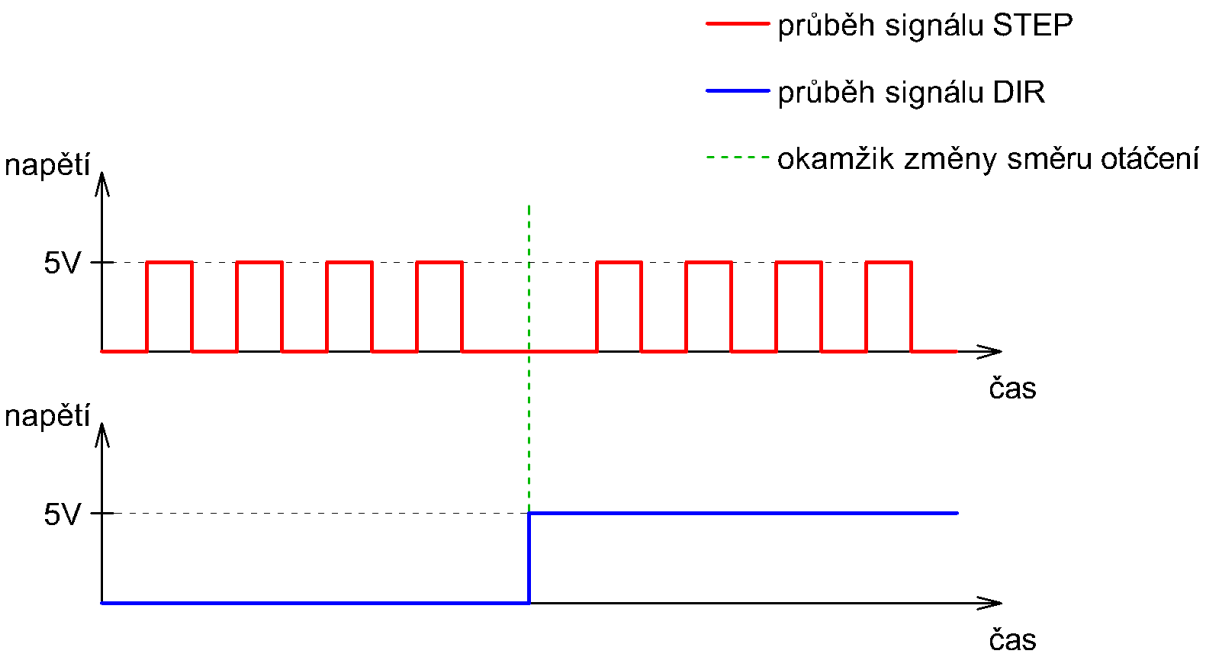
Řízení CNC strojů v podstatě spočívá v řízení pohybu os, potažmo v řízení zařízení, které pohyb osy vyvolává. Princip řízení je ukázán na příkladu, kdy se k pohonu os používají krokové elektromotory, protože jsou k tomu účelu na Ústavu mechaniky a materiálů nejčastěji používané (pohon mohou zajišťovat například i servomotory, lineární elektromotory apod.).

Pohyb krokových motorů je realizován postupným přepojováním cívek na statoru motoru. Přepojování cívek zajišťuje driver (amplifier) motoru. Driver motoru to činí na základě signálu STEP/DIR (step = krok, direction = směr), který do něj vstupuje. Princip řízení je znázorněn na následujícím schématu.



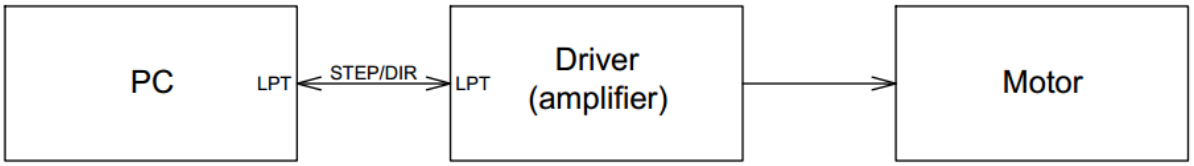
Obrázek č. 4: Schéma řízení

Signál STEP je obdélníkový signál, který udává, kolik kroků má motor vykonat. Motor může krok vykonávat při změně napětí STEP signálu z 0 V na 5 V (tzv. rising edge), nebo při změně napětí z 5 V na 0 V (tzv. sinking edge). Signál DIR udává, jakým směrem se má motor otáčet. Při napětí signálu DIR 0 V se motor otáčí jedním směrem (například po směru hodinových ručiček), při napětí 5 V se motor otáčí opačným směrem (analogicky například proti směru hodinových ručiček).



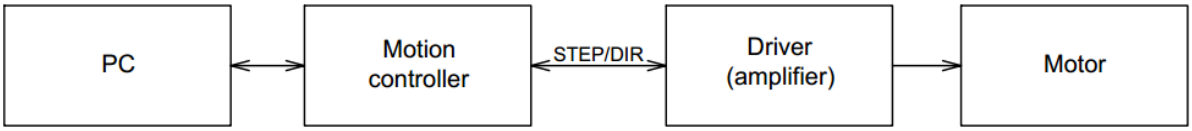
Obrázek č. 5: Schématické znázornění signálu STEP/DIR

Generovat signál STEP/DIR lze několika způsoby. Základní možností je generovat signál STEP/DIR procesorem počítače (CPU) a například pomocí paralelního portu (LPT, lze použít i USB, RJ45, RS232 apod.) jej posílat do driveru motoru. Taková varianta je jednoduše realizovatelná, ale má zásadní nedostatek. V případě, že je procesor počítače zahlcen vykonáváním dalších operací, STEP/DIR signál se nemusí vygenerovat včas a odezva zařízení na příkazy uživatele je pak pomalá a nepravidelná (systém má tzv. vysokou latenci). Motor tak nekoná pohyb v čase tak, jak uživatel požaduje.



Obrázek č. 6: Schéma řízení, generátor STEP/DIR je PC

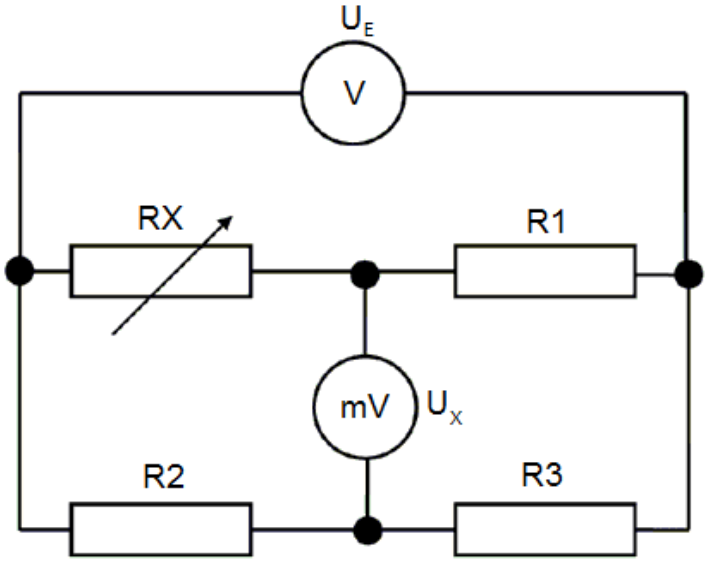
Sofistikovanější možností je použít hardware přímo určený ke generování STEP/DIR signálu (tzv. motion controller). Motion controllery slouží k několikanásobnému snížení latence oproti systému na obrázku č. 6, protože signál STEP/DIR není generován procesorem v PC, ale controllerem. Procesor posílá controlleru pouze příkazy, podle kterých controller signál STEP/DIR generuje.



Obrázek č. 7: Schéma řízení, generátor STEP/DIR je controller

2.3 Měření síly

Některé osy jsou vybaveny siloměrem, který je využíván pro měření vývoje síly ve vzorku v závislosti na posuvu osy (například zatěžovací osa indentoru apod.). Síla je měřena elektromechanickými siloměry, které jsou převážně založeny na principu tenzometrického (Wheatstoneova) můstku, který funguje jako dělič napětí. Napájecí napětí siloměru je U_E [V], výstupním signálem siloměru je napětí U_X [mV]. Proměnný rezistor RX symbolizuje tenzometr, jehož odpor se mění v závislosti na deformaci (úměrné aplikované síle). Rezistory R1, R2, R3 jsou rezistory se známým konstantním odporem.



Obrázek č. 8: Schématické znázornění Wheatstoneova můstku, tzv. čtvrt most

Z hlediska zapojení se může počet tenzometrů lišit. Na obrázku č. 8 je znázorněno zapojení s jedním tenzometrem a třemi konstantními odpory (tzv. čtvrt most). Při zapojení dvou tenzometrů a dvou konstantních odporů se jedná o tzv. půl most. Pokud jsou zapojeny čtyři tenzometry a žádný konstantní odpor, jedná se o tzv. celo most.

Každý siloměr má výrobcem udávané parametry (rozsah, faktor přetížení a citlivost). Rozsah siloměru F_{NOM} [N] udává základ pro jeho bezpečný provoz. Faktor přetížení k [–] udává, kolikrát lze překročit rozsah, aniž by došlo k poškození siloměru. Maximální možné zatížení silou F_{MAX} [N], po kterém již může dojít k poškození snímače, je dáno vztahem:

$$F_{MAX} = k \cdot F_{NOM} \tag{2.1}$$

Siloměr by se k této hranici neměl zatěžovat. Citlivost C [mV/V] udává závislost mezi výstupním napětím U_X [mV] a napájecím napětím U_E [V] na vstupních svorkách Wheatstoneova můstku (napájecí napětí siloměru). Výstupní napětí U_X je obtížně měřitelné, protože jeho velikost je v řádu milivoltů. Proto se používá zesilovač výstupního napětí s koeficientem zesílení G [–]. Převod výstupního signálu (napětí) U_X [mV] na sílu F [N] se uskutečňuje podle následujícího vztahu:

$$F = \frac{1000 \cdot F_{NOM}}{G \cdot U_E \cdot C} \cdot U_X \tag{2.2}$$

2.3.1 Zařízení pro vyčítání síly

Na Ústavu mechaniky a materiálů se v současnosti v rámci experimentálních zařízení používají dva typy elektroniky pro vyčítání síly. Prvním z nich je LabJack T7.

2.3.1.1 LabJack T7

LabJack T7 [4] je zařízení umožňující měřit veličiny pomocí různých senzorů (siloměry, tenzometry – měření prodloužení, termistory – měření teploty, fotorezistory – měření intenzity světla apod.). Pro měření je nutné ho použít v kombinaci se zesilovačem. Nejčastěji používaným zesilovačem na Ústavu mechaniky a materiálů je LJTick-InAmp [6]. Ten umožňuje nastavit různé koeficienty zesílení. Výstupní napětí ze siloměru je převáděno podle vztahu 2.2.

Obrázek č. 9: Ukázka zařízení LabJack T7 se zesilovačem LJTick-InAmp, převzato z [5]

2.3.1.2 Orbit Merret 502T

Další zařízení, které se na Ústavu mechaniky a materiálů používá k vyčítání síly, je Orbit Merret 502T. Orbit Merret 502T [7] je pětimístný zobrazovač pro tenzometrické můstky, který komunikuje přes sériovou linku (rozhraní RS232). V případě použití Orbit Merret 502T není potřeba přepočítávat výstupní signál podle vztahu 2.2. Výstupem je signál, jehož hodnota je přímo síla, protože nastavení parametrů snímače probíhá v aplikaci výrobce.

Obrázek č. 10: Ukázka Orbit Merret 502T, převzato z [7]

2.4 LinuxCNC

LinuxCNC [8] je nadstavba operačního systému umožňující řídit CNC stroje, jako jsou frézky, soustruhy apod. Systém LinuxCNC je vyvíjen jako open-source projekt licencovaný pod GNU General Public License. Běh prostředí zajišťuje jádro operačního systému Linux s real-time rozšířeními (RTAI nebo RT-PREEMT), protože všechny operace systému LinuxCNC musí být vykonávány v reálném čase.

Součástí systému LinuxCNC jsou kromě komponent pro samotné řízení strojů i nativně implementována různá grafická uživatelská rozhraní (GUI), která byla v rámci projektu vyvinuta (například Axis, Touchy apod.). Jsou optimalizována pro ovládání obráběcích strojů a jejich možnosti a rozšiřitelnost jsou značně omezené. Na Ústavu mechaniky a materiálů jsou používána zařízení, která se svou povahou liší od běžných úloh obráběcích strojů, a proto k jejich řízení nativní GUI nejsou vhodná. LinuxCNC ale umožňuje i tvorbu vlastních GUI. První možnosti přidat vlastní GUI formou přidruženého panelu se objevily v roce 2009 (ve verzi 2.3.0). K tomu byl používán nástroj PyVCP (Python Virtual Control Panel). Od roku 2012 (ve verzi 2.5.0) bylo možné k vytvoření vlastních GUI použít nástroj GladeVCP (Glade Virtual Control Panel). Tvorba vlastních GUI pomocí těchto nástrojů je omezená, protože lze použít pouze ovládací prvky, které byly vytvořeny komunitou vývojářů LinuxCNC. Vytvořit zcela nový ovládací prvek těmito nástroji je velmi obtížné a vyžaduje znalost programování dle specifických pravidel pro tvorbu nástrojů přidružených k LinuxCNC. Funkce ovládacích prvků, které slouží k řízení stroje, jsou u výše zmíněných nástrojů vytvořené pomocí Hardware Abstraction Layer (HAL).

HAL je programovací prostředí, které umožňuje pomocí softwaru přiřadit jednotlivým kontaktům (pinům) příslušné funkce. Například specifikovat, na který pin paralelního portu je zapojen signál pro tlačítko nouzového zastavení (E-STOP), zapnutí zařízení, pohyb motorů apod. Konfiguraci pro daný hardware obsahují *.hal soubory. Dalším důležitým souborem systému LinuxCNC je inicializační soubor ke konkrétnímu zařízení. Obsahuje informace nutné ke spuštění a parametry řízení (limity os, maximální rychlost posuvu, měřítko os, zvolené grafické uživatelské rozhraní atd.)

V roce 2014 (ve verzi 2.6.0) byla vydána kompletní dokumentace k řízení zařízení pomocí Python Interface. Python Interface je rozhraní pro LinuxCNC, které umožňuje

komunikovat se zařízením prostřednictvím programovacího jazyka Python. Příkazy k řízení jsou tak plně kompatibilní s jazykem Python, což je plnohodnotný programovací jazyk, a tím řízení i tvorba vlastních GUI prakticky není omezena.

2.5 Python

Python [9] je interpretovaný programovací jazyk, někdy označovaný i jako skriptovací jazyk. Výhody interpretovaných jazyků oproti kompilovaným jazykům tkví ve snadné přenositelnosti mezi platformami. Nevýhodou je, že bývají zpravidla pomalejší, protože se překládají až za běhu. Python je vyvíjen otevřenou komunitou jako open-source projekt a je dostupný pro všechny běžně používané platformy (Unix, Windows i Mac OS). Ve většině Linuxových distribucí je nativně nainstalován.

Python je víceparadigmatický jazyk, tzn. umožňuje při psaní programů využívat více paradigmat jako např. objektově orientované paradigma, procedurální i funkcionální paradigma. Programátor si tak může vybrat, jakým způsobem úlohu řešit, podle toho, co je pro danou úlohu nejvíce vhodné. Díky tomu je kód ve srovnání s ostatními jazyky velmi dobře čitelný a krátký. Klade velký důraz na efektivitu práce programátora, zápis kódu je velmi stručný. To jsou důvody, proč se v současné době těší velké oblibě. Dokonce bývá označován za nejvhodnější programovací jazyk z hlediska učení. Protože je Python vyvíjen otevřenou komunitou, má k dispozici nepřeberné množství volně dostupných knihoven. Přínos knihoven vytvořených komunitou je stěžejní z pohledu použití v projektech.

2.5.1 Knihovny jazyka Python

V informatice se knihovnou označuje množina funkcí (popř. procedur, objektů, datových typů apod.), kterou mohou počítačové programy sdílet. Usnadňují a zefektivňují práci programátora tím, že je lze snadno implementovat do programu.

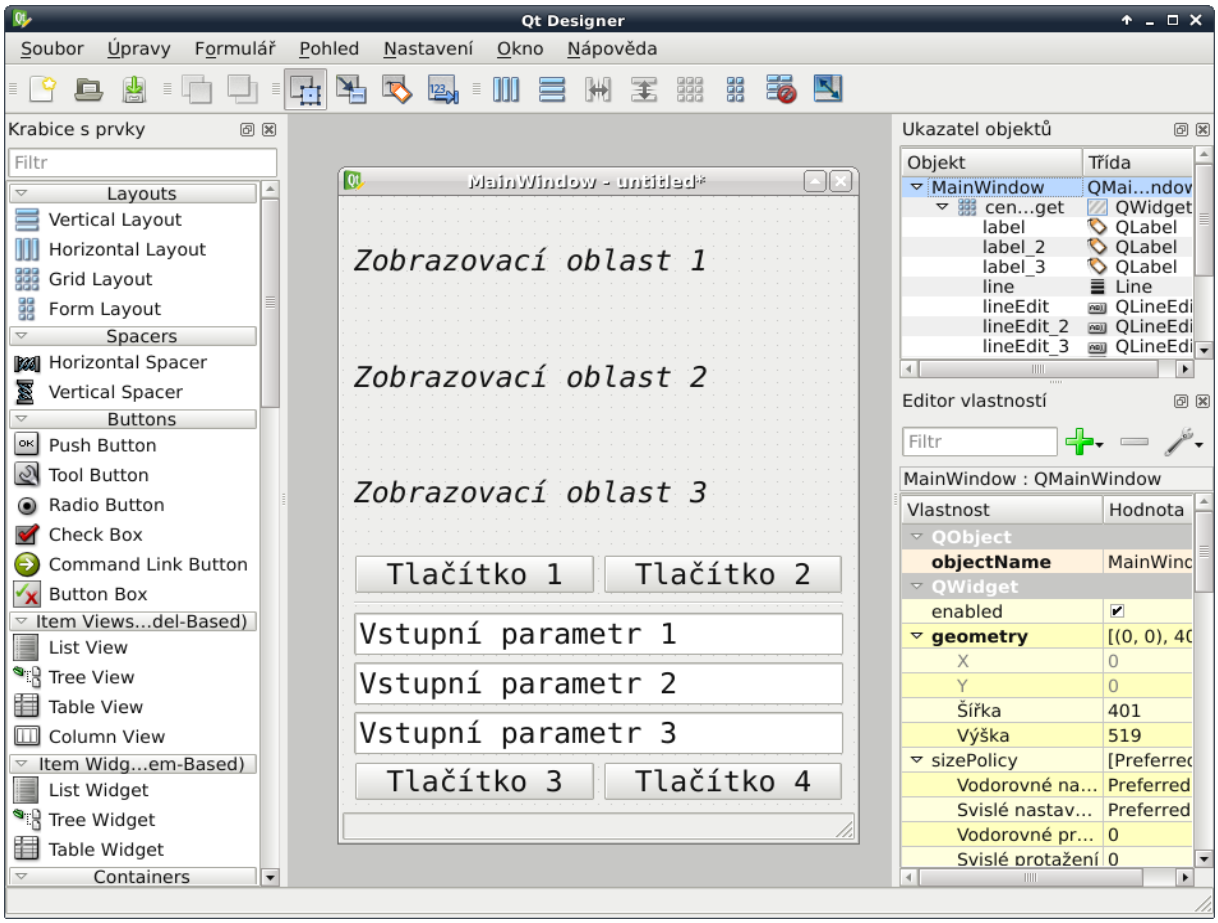
Knihovny umožňují tvořit programy k řešení nejrůznějších problémů. Pro jazyk Python existují knihovny určené k řešení matematických problémů (například knihovny math, NumPy, SciPy, SymPy apod.). Dále existují knihovny určené k editaci obrázků (například knihovna Pillow). Pro komunikaci s kamerami a obrazovou analýzu lze použít například openCV, VTK apod. Pro komunikaci přes sériovou linku a následné vyčítání siloměru například pomocí Orbit Merret 502T umožňuje knihovna pySerial. Různé knihovny pro jazyk Python vytváří sami výrobci hardwaru. Například k zařízení LabJack T7 poskytuje výrobce knihovnu labjack pro jeho snadné ovládání a komunikaci.

Každý vysokoúrovňový programovací jazyk by měl mít možnost vytvoření grafických uživatelských rozhraní (GUI), díky kterým jsou vytvořené aplikace uživatelsky přívětivé a komfortně ovladatelné. Pro Python existuje celá řada knihoven, které umožňují tvorbu GUI (například Tkinter, PySide, GTK+, PyGUI, PyQt apod.). Tyto knihovny zpravidla pracují na stejném principu, liší se svou komplexností a vhodností použití pro daný problém. Různí se také podporou a aktivitou komunity, která se podílí na jejím vývoji. Po zvážení potřeb pro tuto práci byla vybrána knihovna PyQt verze 4.

2.5.1.1 Knihovna PyQt

Knihovna PyQt [10] zajišťuje napojení (tzv. binding) mezi multiplatformním frameworkem Qt [11], který je vytvořen v programovacím jazyce C++, a programovacím jazykem Python. Hlavním důvodem, proč byla vybrána knihovna PyQt, je její zavedenost a aktivní komunita, která zaručuje stabilní podporu. Lze ji používat na různých platformách, a to i na mobilních zařízeních.

Velmi užitečným nástrojem frameworku Qt je Qt Designer. To je vývojové prostředí, které umožňuje velmi rychle a efektivně vytvářet návrhy ovládacích prvků, přičemž není potřeba psát zdrojový kód. Výstupem Qt Designeru je soubor, který neobsahuje zdrojový kód, nýbrž XML (eXtensible Markup Language) kód. Výhoda spočívá v tom, že nejsou vázány na programovací jazyk. XML kód v těchto souborech lze zkompilovat pro použití s programovacím jazykem Python, ale například i pro používání s jazykem C++.



Obrázek č. 11: Ukázka prostředí Qt Designeru, verze 4.8.2

Ke zkompilování XML kódu na zdrojový kód použitelný pro jazyk Python je určen nástroj pyuic. Zkompilovaný soubor pak obsahuje zdrojový kód s definicí GUI použitelný pro jazyk Python. Zkompilovaný soubor by sám uživatel neměl upravovat, aby byla zaručena funkčnost kódu při případně úpravě souboru v Qt Designeru a jeho opětovném zkompilování.

3 Výchozí stav

Práce navazuje na předchozí studentské projekty, ve kterých byla na Ústavu mechaniky a materiálů vyvíjena experimentální zařízení. Řídicí software byl zpočátku vyvíjen pomocí nástroje PyVCP (Python Virtual Control Panel).

3.1 PyVCP

PyVCP [12] je nástroj, který je součástí systému LinuxCNC a umožňuje tvořit vlastní grafické uživatelské rozhraní (GUI) jako přidružené panely k nativně implementovaným GUI LinuxCNC. Ovládací prvky využívané touto knihovnou jsou převzaty z knihovny Tkinter. Definice GUI se realizuje psaním XML kódu.

```
<pyvcp>
    <label text="This is a LED indicator"/>
    <led/>
</pyvcp>
```

Obrázek č. 12: Ukázka tvorby GUI pomocí PyVCP, převzato z [12]

Toto řešení je ale značně omezené. Například lze používat pouze ovládací prvky, které už byly někým z komunity vytvořeny, protože tvorba vlastních ovládacích prvků je velmi komplikovaná a náročná. Neexistují prvky, kterými by bylo možné definovat rozložení panelu (tzv. layout). Velmi složitá je i realizace ovládací logiky (například aby při probíhající operaci byla deaktivována tlačítka, jejichž použití by vedlo k nestabilitě aplikace apod.). Komplikovaná je i implementace duplicitních ovládacích prvků, protože všechny řídicí příkazy lze do zařízení posílat pouze prostřednictvím rozhraní HAL, které nepodporuje duplicitní přiřazování funkcí a v případě duplicit vyžaduje složité řetězení do řady logických operací. Pokročilejší možností je tvořit vlastní GUI pomocí nástroje GladeVCP.

3.2 GladeVCP

GladeVCP [13] je součást systému LinuxCNC umožňující vytvořit vlastní panel přidružený k nativně nainstalovaným GUI systému LinuxCNC (obdobně jako PyVCP). Využívá nástroj Glade [14], který se používá k návrhu GUI a používá ovládací prvky převzaté z knihovny GTK+. Jeho výstupem je soubor obsahující XML kód (obdobně jako Qt Designer). Tyto soubory se oproti Qt Designeru nekompilují, ale importují se do aplikace pomocí nástroje GtkBuilder. GladeVCP nabízí větší množství ovládacích prvků než PyVCP, ale řadu potřebných ovládacích prvků stále neobsahuje. Například chybí prvek, do kterého by bylo možné umístit graf, operace se soubory jsou komplikované, protože funkce ovládacích prvků jsou pro PyVCP i GladeVCP vytvořeny pomocí HAL.

V roce 2014 byla vydána kompletní dokumentace k Python Interface pro LinuxCNC. To umožňuje vytvořit ovládací prvky s řídicími funkcemi kompletně v jazyce Python. Tím je možné odstranit výše zmíněné nedostatky a velmi zjednodušit a zpřehlednit tvorbu vlastních ovládacích prvků. Jazyk Python například obsahuje celou řadu knihoven, které umožňují vykreslovat graf, což pomocí PyVCP i GladeVCP je velice obtížně realizovatelné.

4 Vlastní řešení

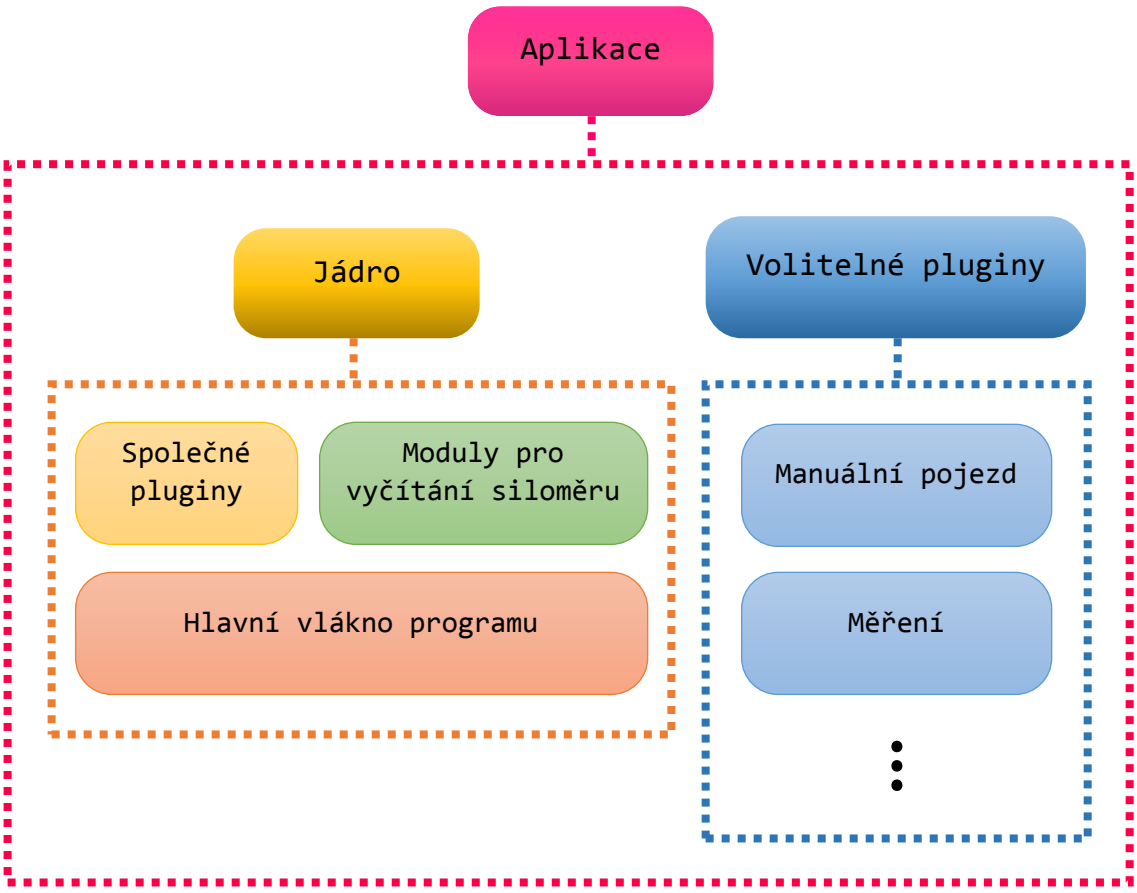
Práce se zabývá vývojem řídicího software experimentálních zařízení na platformě systému LinuxCNC s využitím rozhraní Python Interface. Cílem bylo vytvořit aplikaci (software), která je schopna díky své modularitě řídit současná i budoucí experimentální zařízení na Ústavu mechaniky a materiálů. Toho je docíleno tím, že se aplikace skládá z jádra a k němu byly vytvořeny pluginy (zásuvné moduly), které jsou importovány podle potřeb konkrétního zařízení. Princip modularity je demonstrován na konci kapitoly.

4.1 Úvod

Každé experimentální zařízení má svůj inicializační soubor, jehož účelem je nastavit parametry řízení. Udává například, jaké GUI bude použito k řízení, které osy jsou používány, které osy jsou měřící (na kterých osách je implementován siloměr), limity os, typ os (lineární nebo rotační) apod. Mimo to udává, kterou metodou (modulem) bude vyčítán siloměr, resp. jaký hardware je k tomu účelu v zařízení používán. V závislosti na tom, jsou automaticky generovány ovládací prvky aplikace.

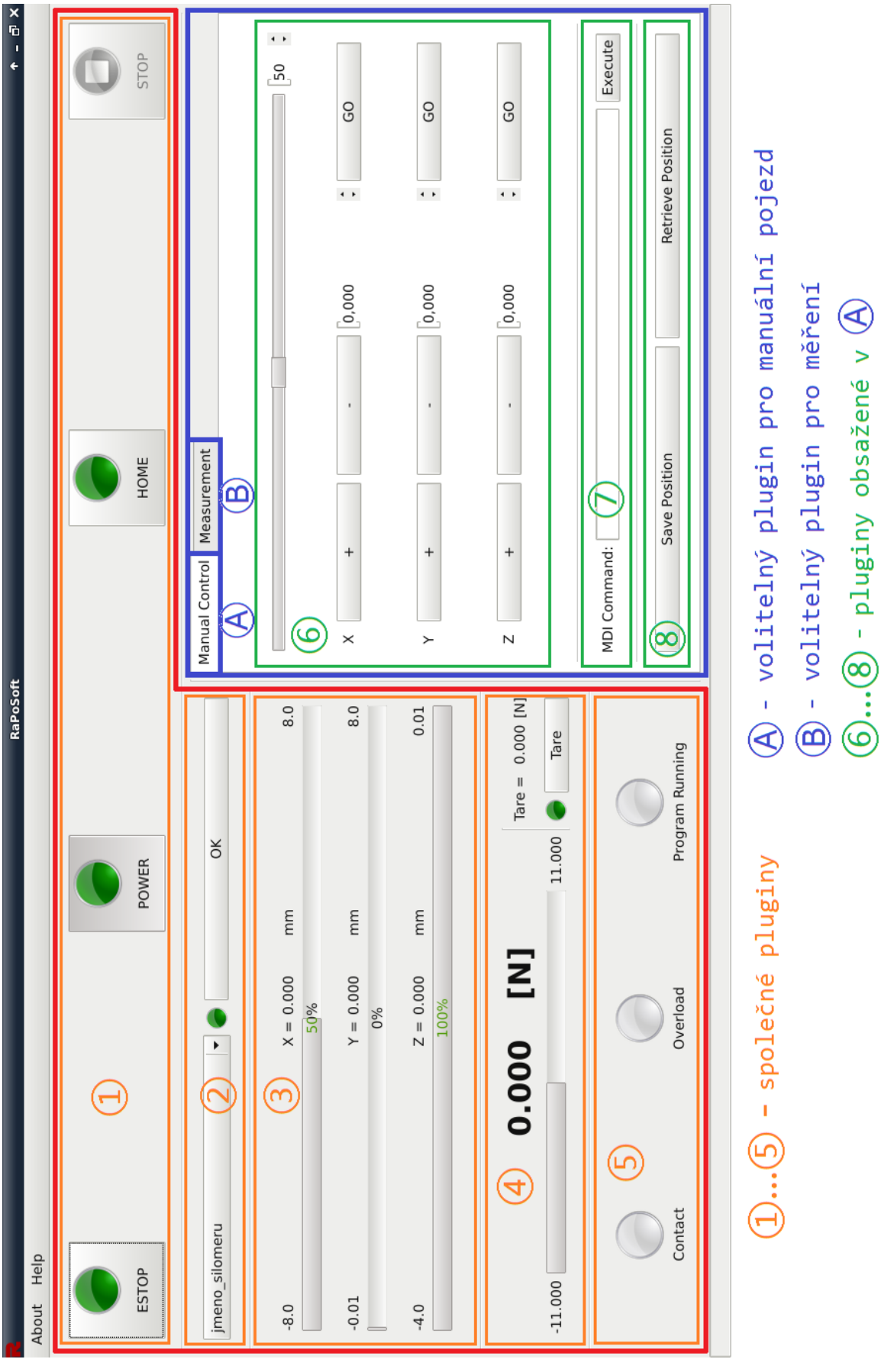
Aplikace obsahuje různé ovládací prvky. Obsahuje tlačítka, zobrazovací oblasti, řádky pro vstupní příkazy a parametry apod. Důležité pro správný běh programů je, aby bylo uživateli umožněno provádět pouze úkony, které nenaruší stabilitu aplikace. Toho je docíleno například zneaktivněním tlačítek ve chvílích, kdy by jejich použití narušilo chod aplikace. K ošetření nevhodných uživatelských zásahů je použita periodicky volaná funkce, která zkoumá, jaké události v programu probíhají a podle toho upravuje aktivní ovládací prvky. V případě textových (popřípadě číselných) vstupů je potřeba je podrobit kontrole (například jestli je daná souřadnice na ose v jejích mezích apod.).

Aplikace se funkčně skládá z jádra, které je pro všechna zařízení společné a z volitelných pluginů (zásuvných modulů). Díky modularitě jsou jednotlivé volitelné pluginy do aplikace snadno připojitelné, a proto lze aplikaci používat na různých experimentálních zařízeních. Předností aplikace je také, že volitelných pluginů může být připojen libovolný počet. Lze snadno používat další, v budoucnu vytvořené pluginy, což umožňuje aplikaci užívat v dlouhodobém horizontu.



Obrázek č. 13: Schéma funkčního členění aplikace

Na obrázku č. 13 je znázorněno funkční členění aplikace. Je na něm vyobrazeno, z jakých funkčních prvků se aplikace skládá, z jakých prvků se skládá jádro a oddělitelnost volitelných pluginů.



Obrázek č. 14: Rozložení pluginů v aplikaci

4.2 Jádro aplikace

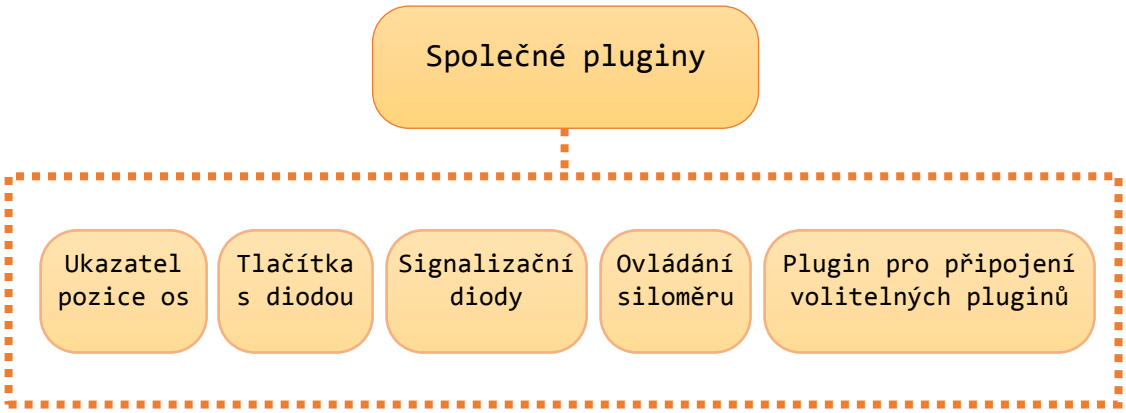
Jádro aplikace tvoří funkční základ programu. Obsahuje spustitelný kód (hlavní vlákno programu) a prvky, které jsou pro všechna zařízení společné (například tlačítko zapnout/vypnout, E-STOP tlačítko, ukazatel pozice os, signalizační diody apod.). Dále jádro obsahuje soubor funkcí (moduly) pro vyčítání siloměru, viz obrázek č. 13. Ke každému siloměru byly vytvořeny inicializační soubory, které obsahují název siloměru, parametry siloměru z kalibračního protokolu dodaného výrobcem (rozsah, faktor přetížení, citlivost) a práh, který odpovídá hodnotě síly indikující, že měřicí osa je v kontaktu se zkoušeným vzorkem.

4.2.1 Hlavní vlákno programu

Hlavní vlákno programu obsahuje kód, který obstarává spuštění a chod celé aplikace včetně inicializace systému LinuxCNC. Je to soubor funkcí a procedur, kterými se uskutečňují veškeré numerické operace v aplikaci.

4.2.2 Společné pluginy

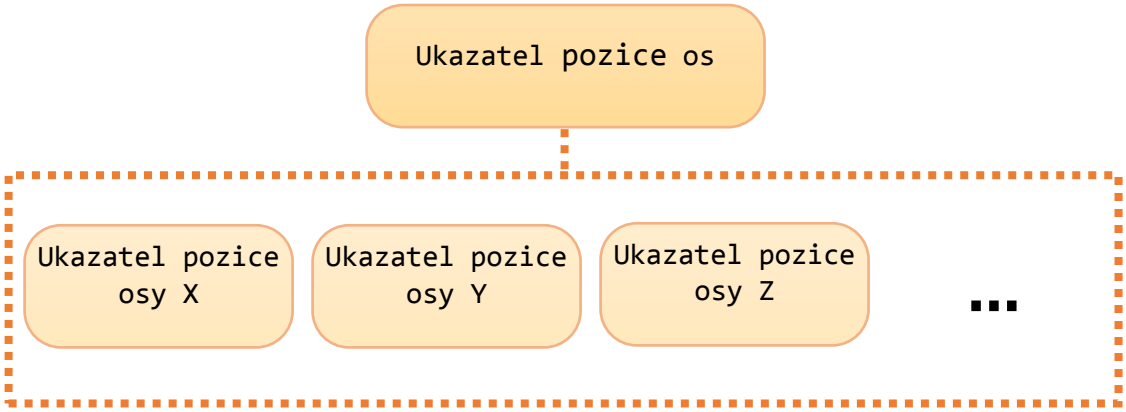
Společné pluginy aplikace jsou modulární ovládací prvky, které jsou automaticky generovány pro všechna zařízení bez rozdílu. Zajišťují základní funkcionalitu zařízení, proto jsou součástí jádra. Rozložení společných pluginů znázorňuje obrázek č. 14.



Obrázek č. 15: Schéma členění společných pluginů

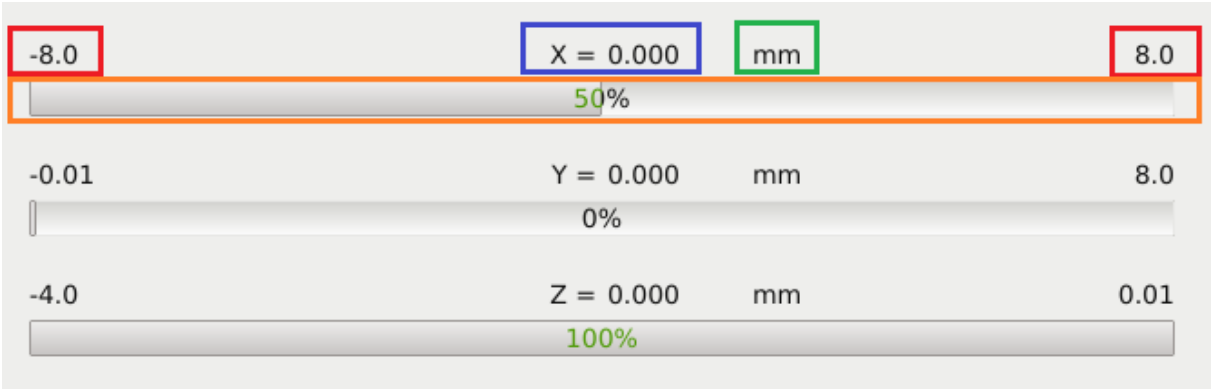
4.2.2.1 Ukazatel pozice os

Ukazatel pozice os je plugin, jehož úkolem je zobrazovat pozice, ve kterých se osy zařízení nacházejí. Pro každou osu je automaticky vygenerován vlastní ukazatel.



Obrázek č. 16: Schéma členění ukazatele pozice os

V inicializačním souboru zařízení uživatel specifikuje, pro které osy budou ukazatele vygenerovány. Parametr k tomu určený je nazván ACTIVE_AXES (aktivní osy) a v inicializačním souboru musí být umístěn v sekci [TRAJ].



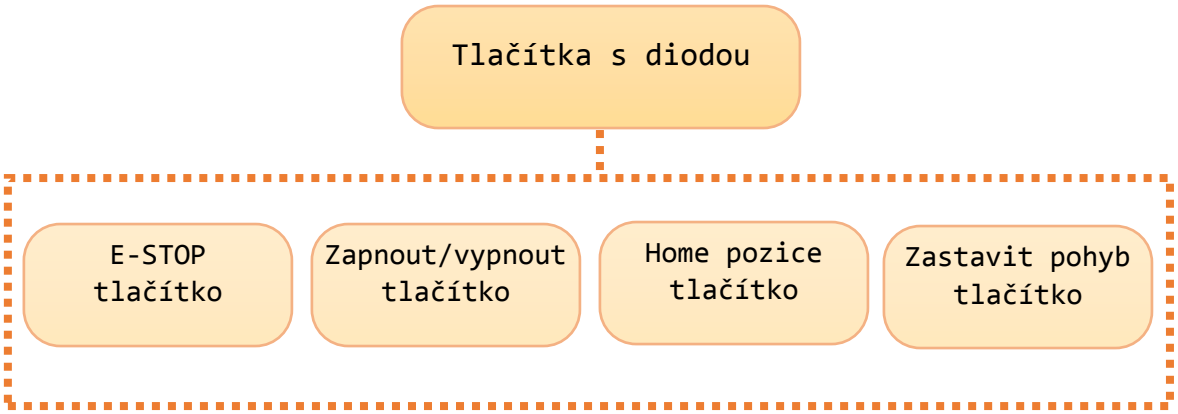
Obrázek č. 17: Ukázka ukazatele pozice os v aplikaci

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ③. Skladební prvky jsou demonstrovány na ukazateli pozice pro osu X, viz obrázek č. 17:

- Červená – minimální a maximální limit osy
- Zelená – délkové jednoty, ve kterých jsou hodnoty zobrazeny
- Modrá – souřadnice osy s přesností na tisícinu délkové jednotky
- Oranžová – pozice přepočtená a zaokrouhlená na celá procenta, 0 % - osa dosáhla minimálního limitu, 100 % - osa dosáhla maximálního limitu

4.2.2.2 Tlačítka s diodou

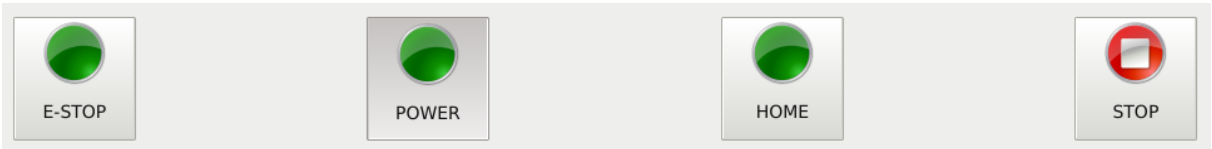
Tlačítka s diodou jsou pluginem, který zajišťuje nejzákladnější funkcionalitu. Dioda na tlačítku signalizuje stav zařízení (například zda je zařízení zapnuto apod.).



Obrázek č. 18: Schéma členění tlačítek s diodou

Nejdůležitějším ovládacím prvkem zařízení z hlediska bezpečnosti je E-STOP tlačítko. Při jeho stisknutí se pohyb zařízení okamžitě zastaví (hardwarově dosáhne stavu E-STOP), uvede se do stavu vypnuto a provádění veškerých úkonů se znemožní. Jediným úkonem, který lze v takové chvíli provést, je zrušit stav E-STOP. Když zařízení přestane být ve stavu E-STOP, uživateli se umožní ho uvést do stavu zapnuto (tlačítko POWER) a může ho začít znovu používat. V případě, že je potřeba zastavit pohyb zařízení a nehrozí žádné nebezpečí, stiskne uživatel tlačítko STOP, kterým zastaví jeho pohyb.

Tlačítko HOME slouží k nalezení referenční (home = domovské) pozice zařízení. Tato procedura se musí provést při každém zapnutí. Zařízení nelze používat, aniž by po jeho zapnutí nebyla dosažena home pozice. To je důležité především u mohutných zařízení, u kterých se po vypnutí mohou osy pohybovat vlivem vlastní tíhy. Pak by při opětovném spuštění neodpovídala pozice osy v aplikaci skutečné poloze.

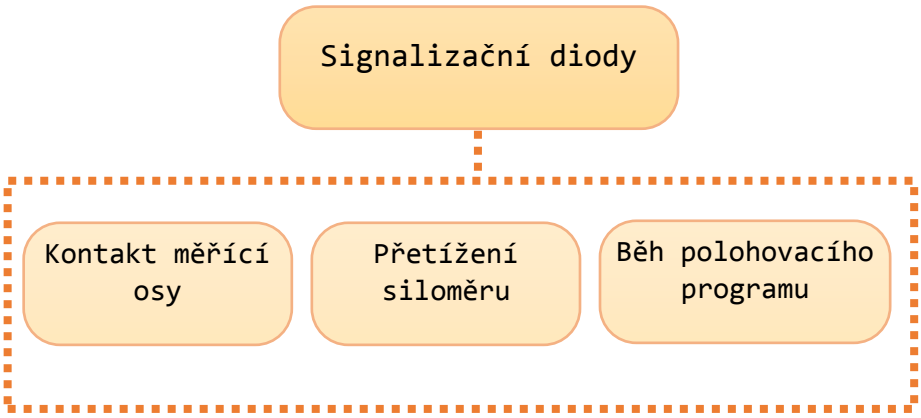


Obrázek č. 19: Ukázka tlačítek s diodou v aplikaci

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ①.

4.2.2.3 Signalizační diody

Plugin signalizačních diod podává základní informace o stavu siloměru (jestli je měřicí osa (osa, u které je implementován siloměr) v kontaktu se vzorkem, jestli je siloměr přetížen) a informace o tom, jestli zařízení provádí nějaký automatizovaný polohovací program.



Obrázek č. 20: Schéma členění signalizačních diod

Dioda signalizující kontakt měřící osy změní svou barvu na oranžovou v případě, že síla naměřená siloměrem na ose je větší než práh kontaktní síly z inicializačního souboru siloměru.

Dioda signalizující přetížení siloměru změní svou barvu na oranžovou v případě, že absolutní hodnota síly naměřené siloměrem překročí rozsah siloměru, ale nepřekročí maximální povolenou hranici síly pro bezpečný provoz siloměru podle vztahu (2.1). Jakmile absolutní hodnota síly překročí i hranici pro bezpečný provoz, dioda změní barvu na červenou. Výrobcem udávaný faktor přetížení používaný při výpočtu (2.1) je běžně 1,5. Pro bezpečný provoz experimentálních zařízení je v aplikaci používán faktor přetížení 1,1. Takový faktor přetížení je používán z důvodu, aby siloměr nebyl zatěžován až ke hranici svých možností, ale bylo zajištěno, že nedojde k jeho poškození. V aplikaci je ošetřeno, aby při přetížení siloměru nad bezpečnou hranici automaticky došlo k nouzovému zastavení přepnutím do režimu E-STOP, ale tato operace trvá řádově milisekundy. Proto je faktor přetížení záměrně snížen oproti hodnotě, kterou udává výrobce, aby byla kompenzována tato časová prodleva.

Dioda signalizující běh polohovacího programu, změní svou barvu na modrou v případě, že probíhá automatizovaný polohovací program, resp. že zařízení provádí polohovací

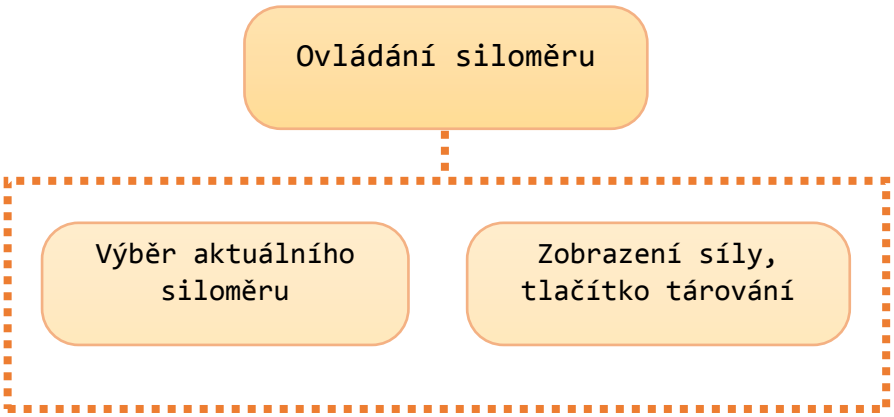
operace vykonáváním příkazů v tzv. G-kódu. Následující obrázek zobrazuje situaci, kdy je siloměr zatížen větší silou, než je kontaktní práh, ale zároveň menší, než je rozsah siloměru. Zařízení vykonává polohovací program.

Obrázek č. 21: Ukázka signalizačních diod v aplikaci

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ⑤.

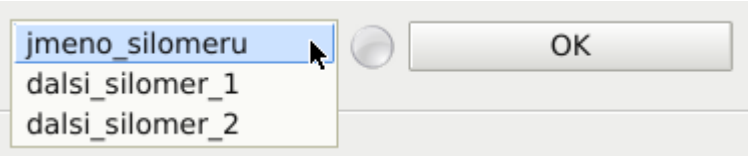
4.2.2.4 Ovládání siloměru

K ovládání siloměru slouží dva pluginy. Prvním pluginem je ze seznamu vybrán siloměr, který je implementován na konkrétním zařízení. Druhý plugin slouží k zobrazování aktuální síly a k tárování siloměru.



Obrázek č. 22: Schéma členění prvků pro ovládání siloměru

Při spuštění aplikace jsou z adresáře, který obsahuje inicializační soubory siloměrů, načteny všechny druhy siloměrů a jejich parametry (rozsah, faktor přetížení, citlivost, kontaktní práh). Po spuštění aplikace uživatel zvolí, jaký siloměr je implementován v zařízení a výběr potvrdí stiskem tlačítka OK. Jestliže výběr siloměru proběhl v pořádku, dioda nalevo od OK tlačítka změní barvu na zelenou. Tím jsou definovány všechny nutné parametry siloměru a aplikace z něho může začít vyčítat sílu.



Obrázek č. 23: Ukázka pluginu pro výběr siloměru

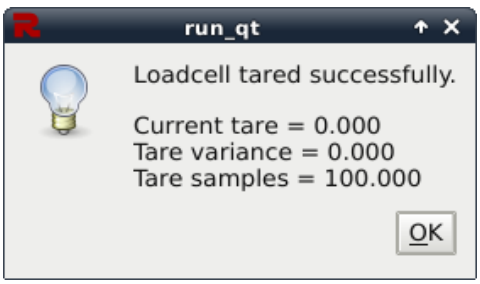
Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ②.

Další plugin slouží k zobrazování síly. Obsahuje řadu zobrazovacích oblastí a tlačítko, které slouží k tárování siloměru. Hodnota tara je vypočtena jako aritmetický průměr posledních 100 hodnot vyčtených ze siloměru. Jestliže nebylo ze siloměru vyčteno dostatečné množství hodnot, tara je vypočítána z dostupných hodnot. Pokud tárování proběhlo v pořádku, objeví se dialogové okno se zprávou o tárování, která obsahuje hodnotu tara, rozptyl hodnot a počet hodnot, ze kterých byla hodnota vypočtena.

Obrázek č. 24: Ukázka pluginu pro zobrazení síly a tárování siloměru

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ④. Skladební prvky jsou následující, viz obrázek č. 24:

- Modrá – síla vyčtená ze siloměru
- Červená – minimální a maximální limity pro provoz siloměru
- Oranžová – zatížení přepočtené a zaokrouhlené na procenta, 0 % - dosažen minimální limit, 100 % dosažen maximální limit pro provoz
- Zelená – hodnota tara
- Fialová – tlačítko tara a signalizační dioda



Obrázek č. 25: Ukázka zprávy o tárování

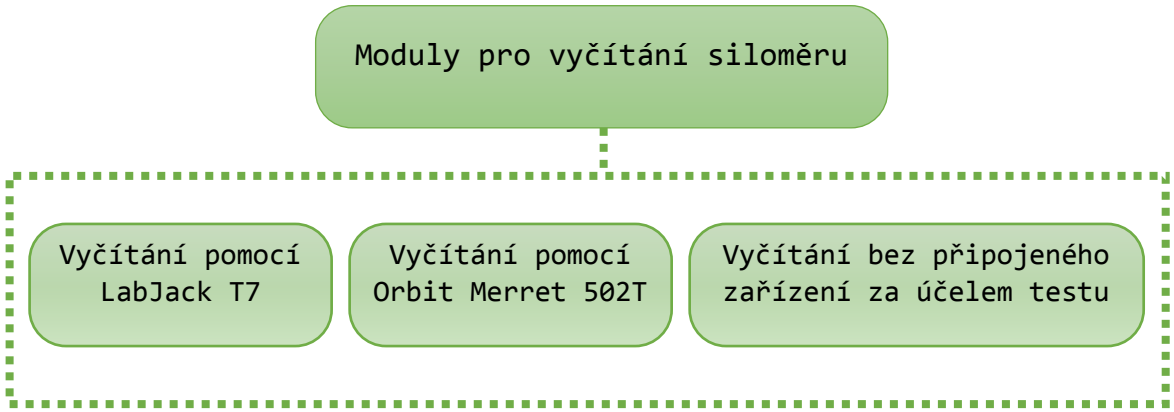
4.2.2.5 Plugin pro připojení volitelných pluginů

Volitelné pluginy se připojují do aplikace pomocí speciálního pluginu. Ten obsahuje všechny potřebné funkcionality pro správné fungování připojených pluginů (například synchronizační funkci, kterou se zajišťuje zpětná vazba). Připojení volitelných pluginů je poté velmi jednoduché a rychlé.

Na obrázku č. 14 nelze plugin přímo vidět, protože je to oblast, kam se volitelné pluginy přichycují. Volitelné pluginy se přichycují jako jednotlivé záložky, ze kterých si uživatel vybírá, jaký bude v danou chvíli používat.

4.2.3 Moduly pro vyčítání siloměru

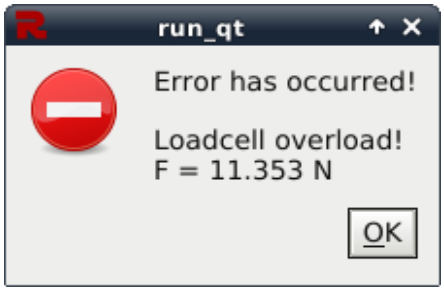
Součástí jádra aplikace jsou také moduly pro vyčítání siloměru. Moduly obsahují funkce a procedury, kterými je zajištěna komunikace se siloměrem v závislosti na použitém hardwaru. Jaký hardware je k vyčítání siloměru v zařízení použit, uživatel specifikuje v inicializačním souboru konkrétního zařízení. Parametr k tomu určený je nazván OM (zkratka Orbit Merret), který musí být umístěn v sekci [EMC].



Obrázek č. 26: Schéma členění modulů pro vyčítání siloměru

Aplikace dokáže vyčítat sílu ze siloměru pomocí zařízení LabJack T7 i pomocí Orbit Merret. Moduly k těmto zařízením mají jasně danou strukturu a vytváří přechodovou vrstvu mezi hardwarem použitým k vyčítání síly a hlavním vláknem programu, kde probíhají všechny numerické operace. Při případném používání jiného druhu hardwaru k vyčítání siloměru, než je LabJack T7 a Orbit Merret 502T, jej lze snadno implementovat vytvořením dalšího modulu podle struktury stávajících modulů, aniž by bylo nutné jakkoli zasahovat do struktury aplikace. To umožňuje aplikaci používat dlouhodobě, protože případný v budoucnu pořízený hardware je snadno implementovatelný.

Definice parametru OM=1 v inicializačním souboru zařízení znamená, že k vyčítání siloměru bude použit modul pro zařízení Orbit Merret 502T. Pokud parametr OM=0, k vyčítání siloměru se použije modul pro zařízení LabJack T7. Jakmile je síla ze siloměru vyčtena, je nutné zkontrolovat, zda není přetížený siloměr, aby bylo co nejdříve zabráněno případnému poškození siloměru. Pokud je siloměr přetížen, aplikace pošle zařízení příkaz k nouzovému zastavení (E-STOP) a objeví se chybová hláška, ve které jsou uživateli poskytnuty informace o tom, že došlo k přetížení siloměru a jaká byla velikost síly, která přetížení vyvolala.



Obrázek č. 27: Ukázka chybové hlášky o přetížení siloměru

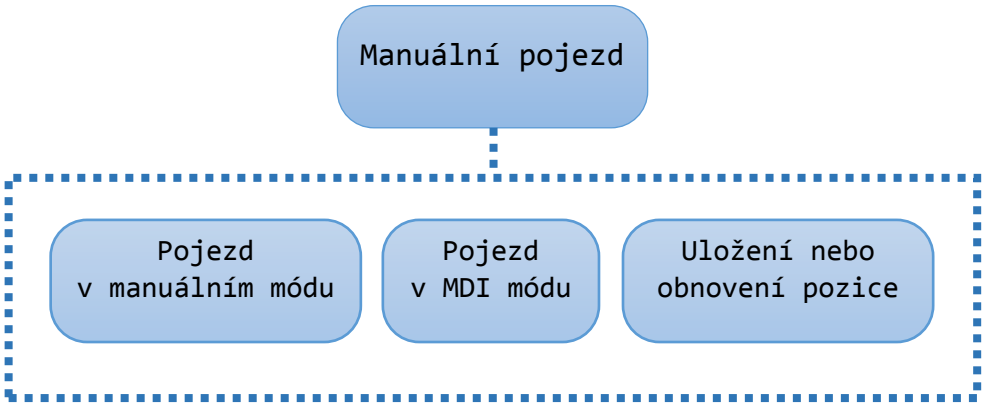
Kromě těchto dvou modulů byl vytvořen i třetí modul, kterým se simuluje komunikace se siloměrem. To umožňuje aplikaci používat, aniž by bylo připojené jakékoli zařízení k vyčítání siloměru. Tento modul sloužil pouze pro účely testování aplikace během jejího vývoje a lze ho použít při definici parametru OM=test.

4.3 Volitelné pluginy

Volitelné pluginy jsou zásuvné moduly, které lze velmi jednoduše implementovat do aplikace. Jejich implementace je zajištěna pomocí speciálního pluginu (viz 4.2.2.5). Volitelné pluginy jsou naprosto nezávislé na jádře aplikace, což umožňuje připojit libovolné pluginy, jak současné, tak i v budoucnu vytvořené. To dává aplikaci velký potenciál z hlediska dlouhodobého využití.

4.3.1 Manuální pojezd

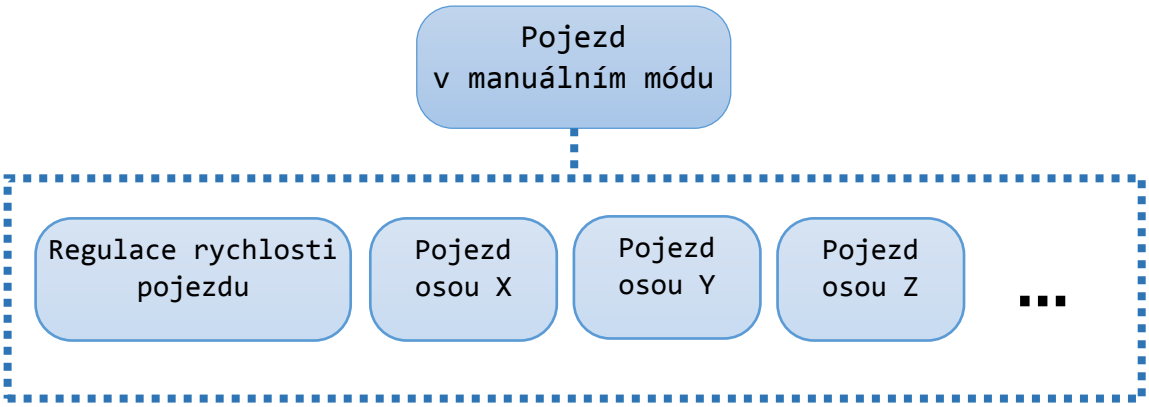
První volitelný plugin, který byl vytvořen slouží, k manuálnímu pojezdu zařízením. Obsahuje ovládací prvky pro pojezd v manuálním módu, MDI (Manual Data Input) módu a prvky pro uložení a obnovení pozice zařízení.



Obrázek č. 28: Schéma členění pluginu pro manuální pojezd

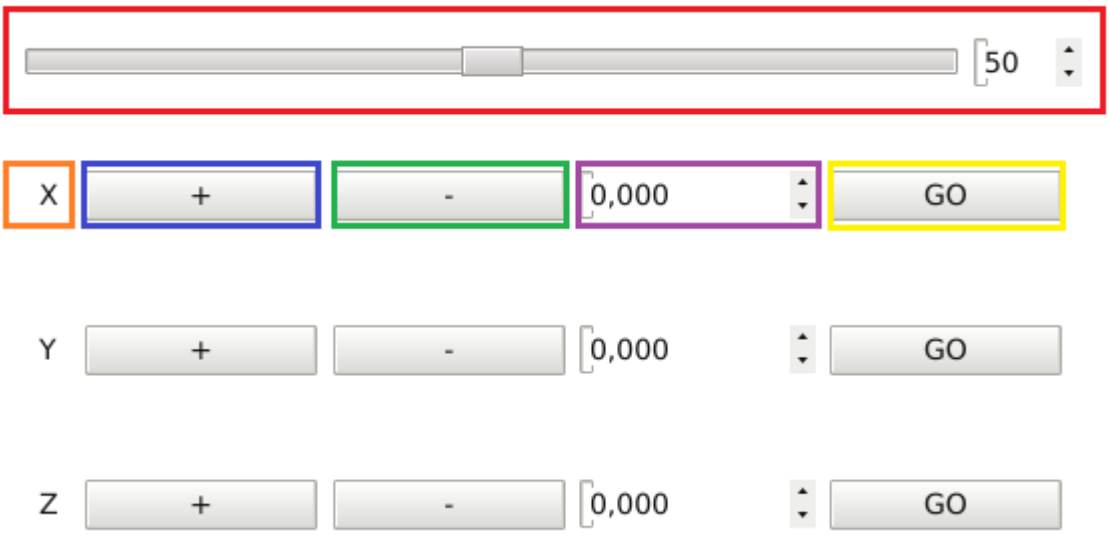
4.3.1.1 Pojezd v manuálním módu

Pojezd v manuálním módu umožňuje uživateli pojíždět osami zařízení v závislosti na tlačítku, které podrží (tzv. jogging). Plugin se skládá z posuvníku, který slouží k regulaci rychlosti pojezdu všech os a z ovládacích tlačítek každé osy, která jsou automaticky generována v závislosti na inicializačním souboru zařízení. V inicializačním souboru k tomu slouží parametr ACTIVE_AXES (obdobně jako u 4.2.2.1). Tím je zajištěna jednoduchá a rychlá variabilita. Na následujícím obrázku je zobrazeno schéma členění pluginu pro pohyb třemi osami X, Y, Z.



Obrázek č. 29: Schéma členění pluginu pro pojezd v manuálním módu

Pro pojezd osou byly vytvořeny ovládací prvky umožňující pojezd v kladném směru, v záporném směru pojezdu a pojezd do místa definovaného souřadnicí na ose. Posuvník reguluje rychlost pojezdu všech os. Jeho součástí je i oblast, která zobrazuje rychlost pojezdu jako procentuální podíl maximální rychlosti pojezdu. Tato oblast umožňuje i explicitně specifikovat, jakým procentuálním podílem maximální rychlosti pojezdu bude zařízení pojíždět, vepsáním celočíselné hodnoty v intervalu $\langle 0;100 \rangle$.



Obrázek č. 30: Ukázka pluginu pro pojezd v manuálním módu

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ⑥. Skladební prvky jsou demonstrovány na ovládacích prvcích pro pojezd osou X, viz obrázek č. 30:

- Červená – posuvník k regulaci rychlosti všech os včetně zobrazovací oblasti
- Oranžová – oblast s názvem osy, kterou ovládací prvky pohybují
- Modrá – tlačítko pro pojezd v kladném směru osy

- Zelená – tlačítko pro pojezd v záporném směru osy
- Fialová – oblast pro specifikaci souřadnice, do které má osa jet
- Žlutá – tlačítko k zahájení pojezdu do specifikované souřadnice

4.3.1.2 Pojezd v MDI módu

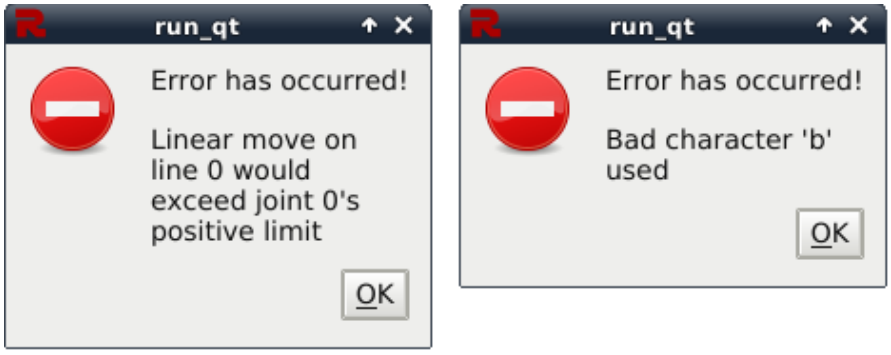
Plugin pro pojezd v MDI (Manual Data Input) módu umožňuje pojíždět zařízením na základě vstupního příkazu v G-kódu, který pak zařízení vykoná. Plugin se skládá ze vstupní oblasti pro polohovací příkaz v G-kódu a z tlačítka, kterým se zahájí vykonávání příkazu. Například pro pojezd do polohy (X, Y, Z) = (0, 0, 0) maximální možnou rychlostí je používán příkaz G0 X0 Y0 Z0. Pro větší komfort uživatele byl vytvořen i našeptávač, který obsahuje poslední provedené polohovací příkazy.

Obrázek č. 31: Ukázka pluginu pro pojezd v MDI módu

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ⑦. Skladební prvky jsou následující, viz obrázek č. 31:

- Oranžová – popisek
- Červená – oblast pro polohovací příkaz v G-kódu s našeptávačem
- Modrá – tlačítko k zahájení provádění příkazu

V případě, že uživatel zadá polohovací příkaz, ve kterém je některá pozice osy mimo své limity, nebo příkaz, který nemá správnou syntaxi, objeví se následující chybové hlášky.



Obrázek č. 32: Ukázka chybových hlášek pro nesprávný polohovací příkaz

4.3.1.3 Uložení nebo obnovení pozice

Během provozu zařízení může nastat situace, že zařízení je v poloze, která je z určitého důvodu důležitá. Při experimentu může být použita například kamera, která pořizuje obrazová data, a zařízení dosáhne pozice, kdy je výstupní obraz kamery správně zaostřený. V případě, že je nutné se zařízením provést další úkony, dojde ke ztrátě pozice, ve které byl obraz zaostřený. Proto byl vytvořen plugin, který umožňuje uložit pozici os a následně se do dané pozice vrátit (aby uživatel nemusel znovu hledat pozici, kdy byl obraz zaostřený).

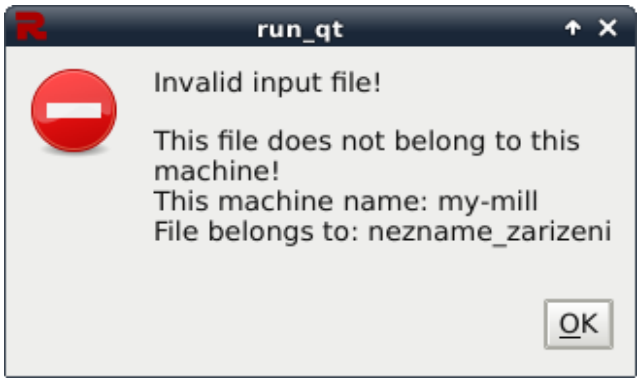
Obrázek č. 33: Ukázka pluginu pro uložení nebo obnovení pozice

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 14 pod číslem ⑧. Plugin se skládá z tlačítka, jehož stiskem se otevře dialogové okno, pomocí něhož uživatel pozici uloží do souboru. Uložený soubor nese informace o zařízení, ke kterému patří, datum, čas a souřadnice os.

```
MACHINE = my-mill
DATE = 2017-03-17
TIME = 23:35:31
X = 0.0
Y = 0.0
Z = 0.0
```

Obrázek č. 34: Ukázka obsahu souboru s uloženou pozicí

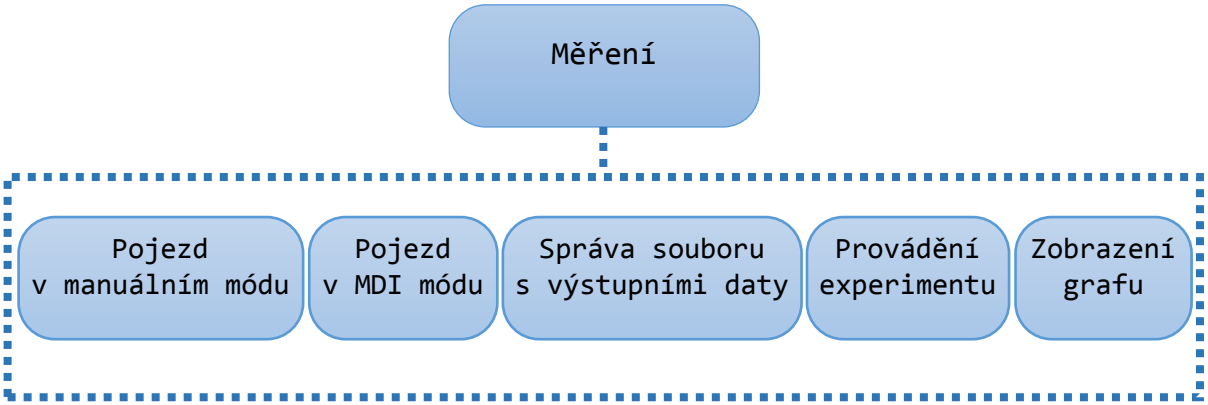
Stiskem tlačítka pro obnovení pozice se zkontrolují informace v něm obsažené (jestli soubor náleží danému zařízení a jestli jsou souřadnice v limitech os). Jestliže jsou vstupní data ze souboru v souladu s parametry zařízení, zahájí se pojezd do dané pozice maximální možnou rychlostí (pomocí příkazu G0). Pokud vstupní data nesouhlasí, objeví se chybová hláška. Následující obrázek zobrazuje chybovou hlášku, kdy nesouhlasí jméno zařízení se jménem zařízení ve vstupním souboru.



Obrázek č. 35: Ukázka chybové hlášky při obnovení pozice

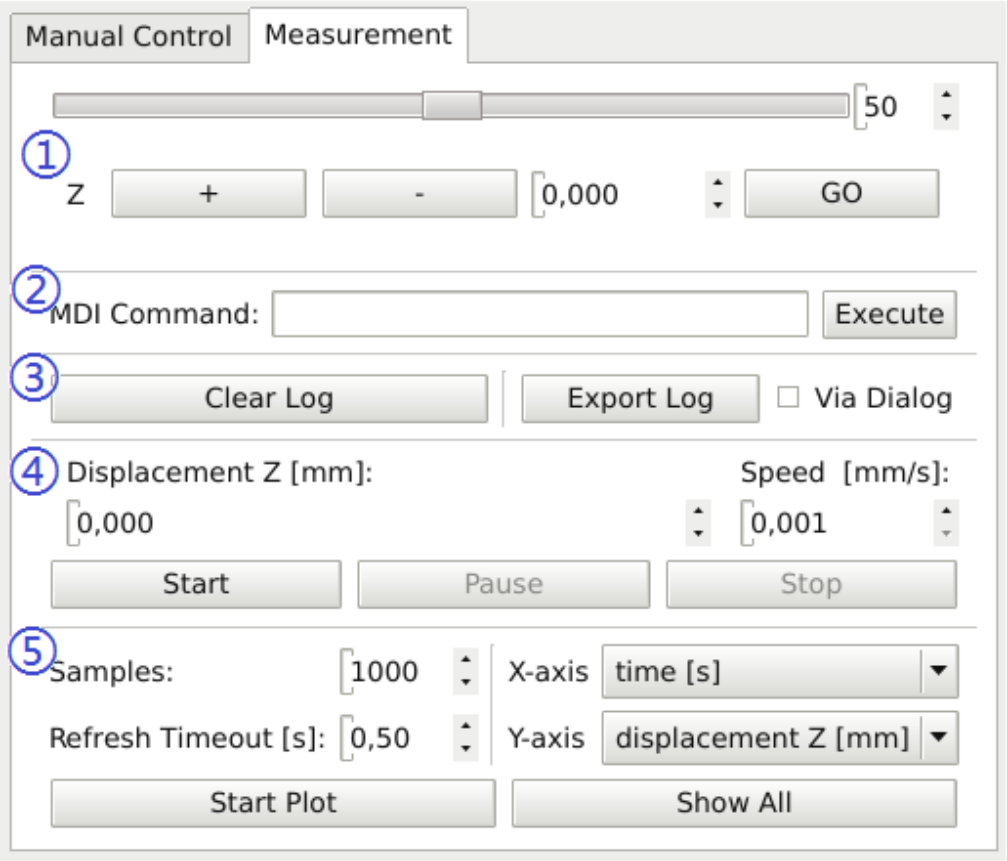
4.3.2 Měření

Další volitelný plugin, který byl v rámci práce vytvořen, je určený k provádění měření. Umožňuje pojíždět osami podobně jako volitelný plugin pro manuální pojezd (4.2) s tím rozdílem, že slouží k obsluze pouze os, na kterých je implementován siloměr (měřící osy). Dále umožňuje spravovat soubor s výstupními daty, kam jsou průběžně ukládány naměřené hodnoty (čas, síla, pozice os). Umožňuje provádět experiment a data pořízená v průběhu experimentu vykreslovat do grafu.



Obrázek č. 36: Schéma členění pluginu pro měření

Na obrázku č. 14 není volitelný plugin pro měření viditelný, protože jej překrývá záložka s volitelným pluginem pro manuální pojezd.



Obrázek č. 37: Rozložení volitelného pluginu měření

Obrázek č. 37 postihuje situaci, kdy uživatel aktivoval volitelný plugin (záložku) měření, tím pádem přešel do popředí a stal se viditelný. Volitelný plugin pro manuální pojezd přešel do pozadí.

4.3.2.1 Pojezd v manuálním módu

Plugin pro pojezd v manuálním módu ve volitelném pluginu pro měření slouží k pojezdu osami podobně jako 4.3.1.1. Tento plugin se ale liší tím, že umožňuje pojíždět pouze osou, která je určená pro provádění experimentu. Osu určenou pro měření (s implementovaným siloměrem) uživatel specifikuje v inicializačním souboru zařízení parametrem MEASUREMENT_AXES, který musí být umístěn v sekci [TRAJ]. Následující obrázek odpovídá definici parametru MEASUREMENT_AXES = Z.

Obrázek č. 38: Ukázka pluginu pro manuální pojezd pro měření

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 37 pod číslem ①. Skladební prvky jsou funkčně analogické s 4.3.1.1.

4.3.2.2 Pojezd v MDI módu

Tento plugin má naprosto stejnou funkci jako 4.3.1.2. Do záložky pro měření byl umístěn z důvodu, že je poměrně často používán a uživatel tak nemusí přepínat mezi záložkami. Umístění pluginu v aplikaci lze vyčíst z obrázku č. 37 pod číslem ②.

4.3.2.3 Správa souboru s výstupními daty

Po spuštění aplikace, když je vybrán siloměr implementovaný v zařízení, se ze siloměru začne vyčítat síla. Síla se ze siloměru vyčítá s určitou frekvencí (například 20 Hz). Ke každé hodnotě síly je zjištěn čas s přesností na milisekundy a pozice měřící osy (nebo i více měřících os v závislosti na parametru MEASUREMENT_AXES) s přesností na tři desetinná místa (nejčastěji mikrometry). Tato data jsou automaticky ukládána do matice v aplikaci, která má určenou strukturu, viz následující obrázek č. 39, který zobrazuje několik naměřených hodnot.

0.000	0.004	0.000
0.050	0.004	0.000
0.100	0.004	0.000
0.150	-0.001	0.000
0.200	-0.001	0.000
0.250	-0.001	0.000
0.300	-0.001	0.000

Obrázek č. 39: Ukázka výstupních dat

V červeném sloupci je zaznamenaný čas, kdy byla ze siloměru vyčtena síla, od počátku měření. V tomto případě byla síla vyčítána každých 0,05 s, tzn. s frekvencí 20 Hz. V modrém sloupci je síla vyčtená ze siloměru. Jak je vidět, siloměr není v tomto případě zatížen, protože se síla pohybuje v řádu mN. V zeleném sloupci je uložená pozice měřící osy. Každý řádek odpovídá jednotlivému stavu zařízení v závislosti na čase.

Data uložená v matici aplikace jsou průběžně ukládána do výstupního souboru, aby byly sníženy paměťové nároky a omezeno riziko ztráty dat. Aby bylo dosaženo co nejoptimálnějšího běhu aplikace, a tím pádem výstupní data byla co nejspolehlivější, bylo empiricky stanoveno, že jsou data ukládána pokaždé, když se počet řádků matice zvýší

o 200. Důvodem bylo, že každé otevření souboru s výstupními daty aplikaci stojí určitý čas, tudíž bylo nutné najít rovnováhu mezi časem nutným na otevření a zavření souboru a mezi časem nutným na vepsání dat do souboru.

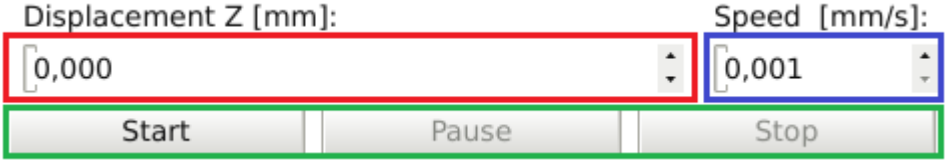
Obrázek č. 40: Ukázka pluginu pro správu souboru s výstupními daty

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 37 pod číslem ③. Plugin se skládá ze dvou tlačítek. První tlačítko slouží k vytvoření nového, čistého souboru pro výstupní data. Tento úkon je výhodné provést například bezprostředně před prováděním experimentu, aby ve výstupním souboru nebyly obsaženy záznamy, kdy bylo se zařízením pojížděno do pozice před experimentem.

Výstupní soubory jsou pojmenovány na základě času a data, kdy byly vytvořeny. To znamená, že z jejich názvu nelze přímo vyčíst, jaká data obsahují, z názvu lze vyčíst pouze, kdy byla pořízena. Proto plugin obsahuje druhé tlačítko, které slouží k exportování výstupního souboru. Plugin obsahuje i políčko, jejímž zaškrtnutím uživatel definuje, že výstupní soubor má být exportován pomocí dialogu, kde je možné dát souboru libovolný název a uložit do libovolného adresáře. Pokud políčko nebylo zaškrtnuto, při stisknutí tlačítka pro export se výstupní soubor uloží pod předdefinovaným názvem do předem stanoveného adresáře.

4.3.2.4 Provádění experimentu

Plugin pro provádění experimentu umožňuje provést zařízením experiment, jehož průběh řídí aplikace polohovacím programem. Před startem experimentu je potřeba specifikovat pozici měřící osy, ve které dojde k ukončení experimentu a rychlost pojezdu během experimentu. Plugin obsahuje 3 tlačítka, která slouží ke spuštění, pozastavení a předčasnému ukončení experimentu.



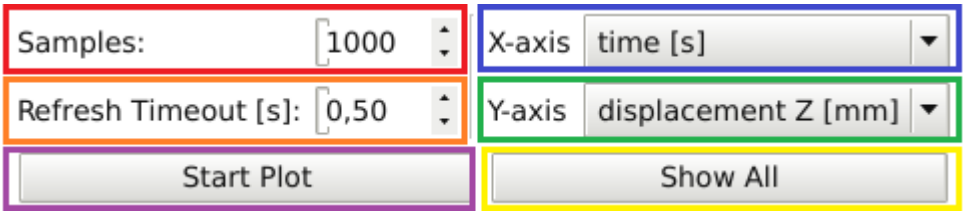
Obrázek č. 41: Ukázka pluginu k provádění experimentu

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 37 pod číslem ④. Skladební prvky jsou následující, viz obrázek č. 41:

- Červená – oblast pro specifikování pozice ukončení experimentu
- Modrá – oblast pro specifikování rychlosti pojezdu během experimentu
- Zelená – tlačítka k zahájení, pozastavení a předčasnému zastavení experimentu

4.3.2.5 Zobrazení grafu

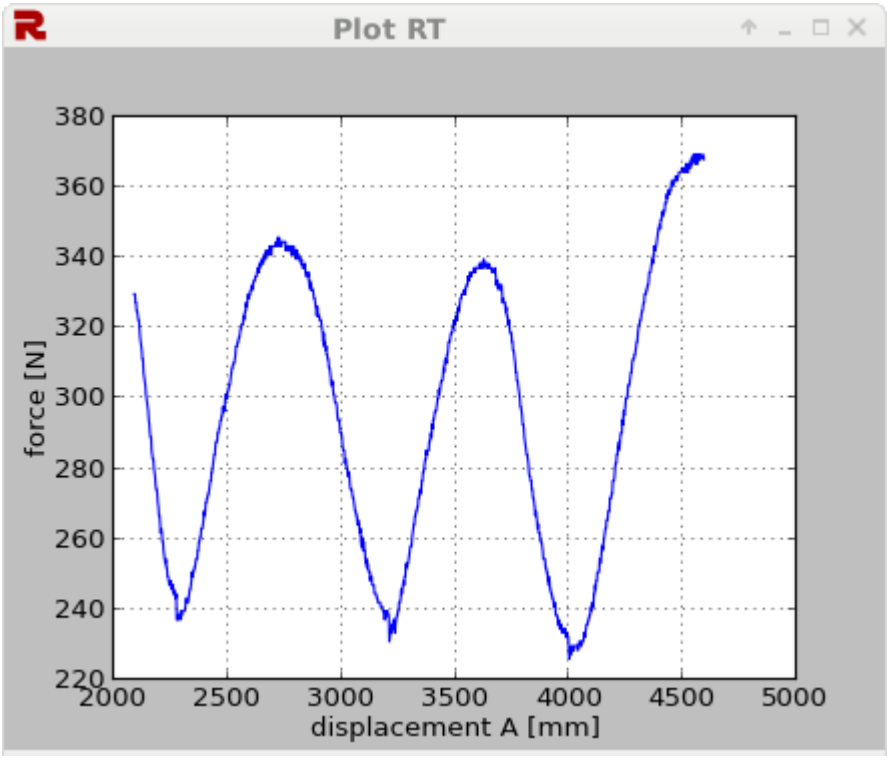
Aplikace umožňuje průběh experimentu vykreslovat v grafu, k čemuž slouží tento plugin. Umožňuje vykreslovat graf kontinuálně, a tak je uživateli umožněno sledovat, jak experiment probíhá. Uživatel si může zvolit veličiny grafu na obou osách (např. posunutí měřicí osy, síla, čas). Plugin obsahuje oblasti, jimiž jsou specifikovány parametry vykreslování grafu, tlačítko k zahájení kontinuálního vykreslování grafu a tlačítko, kterým se vykreslí graf ze všech naměřených hodnot.



Obrázek č. 42: Ukázka pluginu pro zobrazení grafu

Umístění pluginu v aplikaci lze vyčíst z obrázku č. 37 pod číslem ⑤. Skladební prvky jsou následující, viz obrázek č. 42:

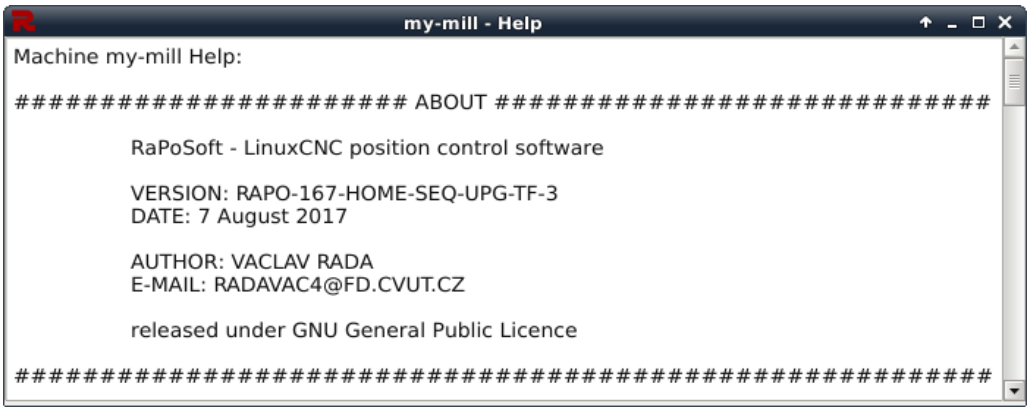
- Červená – oblast, kde uživatel specifikuje, jaký počet hodnot se má kontinuálně vykreslovat (čím větší počet, tím větší náročnost na vykreslení)
- Oranžová – oblast, kde uživatel specifikuje, s jakou periodou se má kontinuální graf vykreslovat (čím menší perioda, tím plynulejší vykreslování, ale větší náročnost)
- Modrá – hodnoty, které budou vykresleny jako definiční obor (čas, síla nebo posunutí)
- Zelená – hodnoty, které budou vykresleny jako obor hodnot (čas, síla nebo posunutí)
- Fialová – tlačítko k zahájení kontinuálního vykreslování
- Žlutá – tlačítko k vykreslení všech hodnot



Obrázek č. 43: Ukázka okna s vykresleným grafem

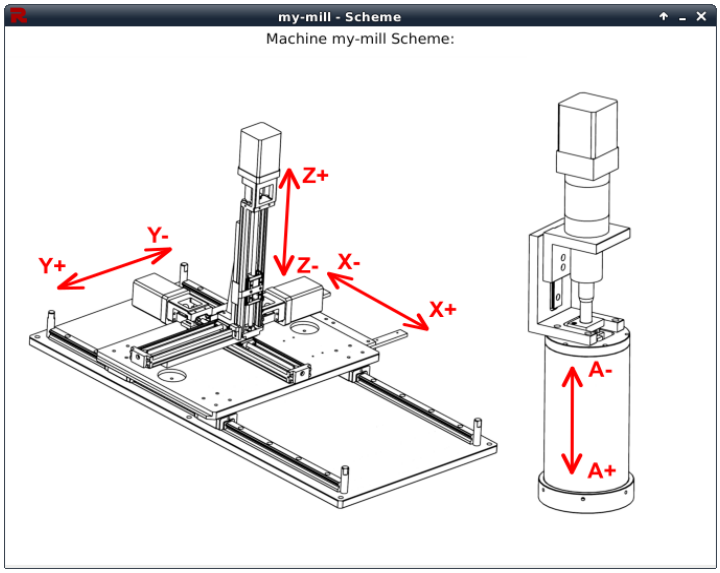
4.4 Ostatní funkce aplikace

Kromě výše zmíněných funkcí aplikace obsahuje i panel nabídek, kterým jsou uživateli poskytnuty informace o verzi software, změnách v poslední verzi (tzv. changelog) apod. Tyto informace jsou uloženy v textovém souboru, ze kterého si je aplikace načte a následně je zobrazí v separátním okně.



Obrázek č. 44: Ukázka okna s informacemi o aplikaci

Aplikace dále umožňuje zobrazit schéma stroje s vyznačenými osami a jejich orientací. To je důležitá funkce z toho důvodu, aby byl uživatel seznámen se zařízením. Jinak hrozí, že by bylo pojížděno jinou osou, než bylo zamýšleno, anebo by mohlo být pojížděno opačným směrem.



Obrázek č. 45: Ukázka okna se schématem zařízení

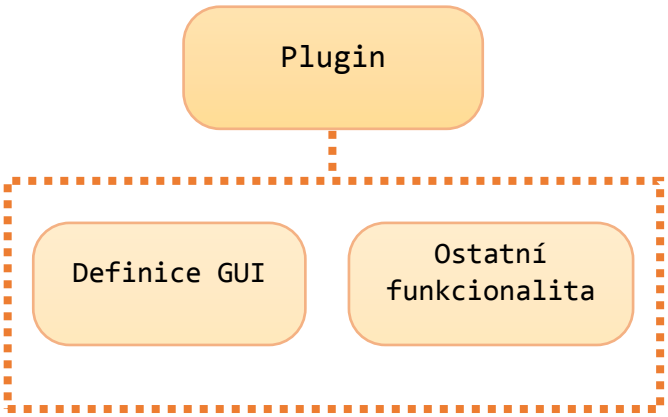
4.5 Princip modularity

4.5.1 Modularita pluginů

Všechny pluginy byly vytvořeny jako separátní ovládací prvky, přičemž každý plugin je uložen v samostatném adresáři. V adresáři jsou obsaženy zpravidla dva soubory se zdrojovým kódem.

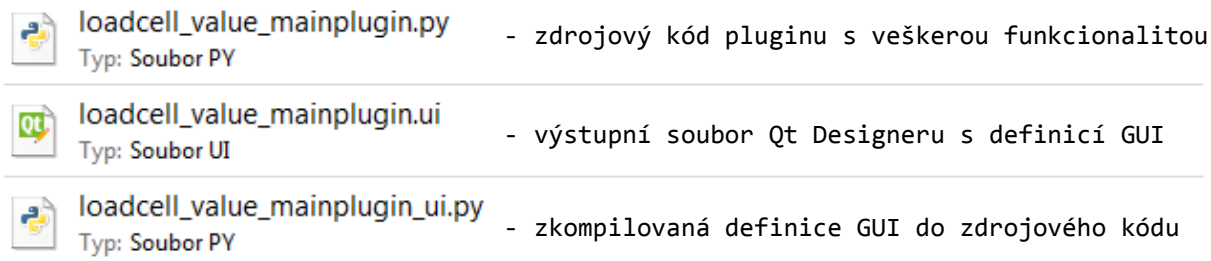
První soubor obsahuje zdrojový kód, který definuje GUI pluginu. Tento soubor byl obvykle vytvořen kompilací výstupního souboru nástroje Qt Designer, ale pro některé specifické pluginy, které jsou generovány parametricky, bylo potřeba vytvořit soubory s definicí GUI manuálně.

Druhý soubor obsahuje kompletní zdrojový kód pluginu. Všechny tyto soubory byly vytvořeny manuálně tak, že bylo importováno GUI z prvního souboru a bylo doplněno o veškerou funkcionalitu (např. pro tlačítka byly vytvořeny funkce, kterými jim byly přiřazeny akce např. při stisknutí, podržení apod.). Pro každý plugin byla vytvořena i synchronizační funkce, která je volána periodicky a obstarává, aby plugin zobrazoval aktuální stav stroje (např. aktuální pozice os, aktuální síla, jestli je stroj zapnutý apod.), ale také deaktivuje ovládací prvky ve chvíli, kdy by jejich použití mohlo vyvolat nestabilitu aplikace nebo jiné nebezpečí.



Obrázek č. 46: Schéma funkčního řešení pluginů

Obrázek výše schematicky znázorňuje oddělenost GUI pluginu od jeho funkcionality. To umožňuje GUI upravovat, aniž by bylo nutné zasahovat do zdrojového kódu pluginu. Následující obrázek znázorňuje členění jednotlivých souborů se zdrojovým kódem pluginu na příkladu pluginu k zobrazení aktuální hodnoty síly.



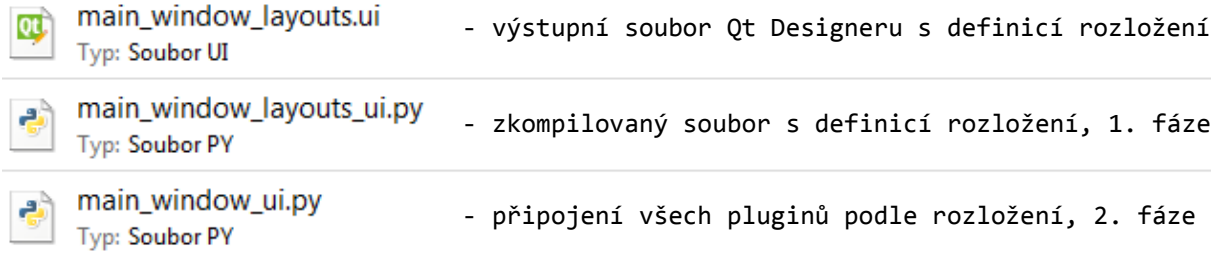
Obrázek č. 47: Popis souborů náležících pluginu

4.5.2 Modularita aplikace

Při návrhu aplikace byl kladen důraz na oddělitelnost a ucelenost jednotlivých pluginů, ale také na to, aby byl při vytváření GUI co nejvíce využit nástroj Qt Designer, a tím byl zefektivněn vývoj aplikace. Proto je definice GUI hlavního okna aplikace prováděna dvoufázově.

4.5.2.1 Fáze tvorby hlavního okna

V první fázi bylo pomocí Qt Designeru vytvořeno prázdné hlavní okno, ve kterém se nacházejí pouze prvky definující rozložení a umístění (zatím nepřipojených) pluginů v aplikaci (tzv. layouty). Ve druhé fázi jsou do prázdného hlavního okna připojeny pluginy na určené místo v okně, které bylo definováno v první fázi. Takové řešení znamená, že připojování společných pluginů do aplikace je realizováno ve třech krocích.



Obrázek č. 48: Popis souborů s definicí hlavního okna aplikace

4.5.2.2 Připojení společných pluginů

Připojení společného pluginu je realizováno ve třech krocích, protože společné pluginy jsou připojeny přímo do oblasti hlavního okna. K tomu je potřeba provést následující kroky:

1) Vytvořit novou oblast v definici hlavního okna, do které bude plugin zasazen (tzv. layout), tzn. modifikovat první fázi tvorby hlavního okna
2) Připojit plugin do připravené oblasti vytvořené v prvním kroku, tzn. modifikovat druhou fázi tvorby hlavního okna
3) Připojit synchronizační funkci nového pluginu do synchronizační funkce celé aplikace

4.5.2.3 Připojení volitelných pluginů

Připojení volitelných pluginů je zjednodušené, protože pluginy nejsou připojeny do oblasti hlavního okna, ale jsou připojeny do speciálního pluginu (viz 4.2.2.5), který disponuje nutnou funkcionalitou. K připojení volitelného pluginu je potřeba provést pouze jeden krok:

1) Připojit plugin do speciálního pluginu, tzn. modifikovat druhou fázi tvorby hlavního okna

Připojit synchronizační funkci není potřeba, protože to obstarává plugin pro připojení volitelných pluginů automaticky.

5 Implementace

Kapitola implementace se zabývá uvedením řídicího software do provozu. Řídicí software byl implementován do řídicí jednotky, která poskytuje veškerou hardwarovou funkcionalitu. Jednotka vznikla v roce 2016 jako předešlý studentský projekt, jehož účelem bylo vytvořit řídicí jednotku s unifikovaným hardwarovým rozhraním pro různá experimentální zařízení [15]. Řídicí software, jehož tvorbou se tato práce zabývá, měl tuto hardwarovou platformu vhodně doplnit a vytvořit tak univerzální řídicí systém, který je schopen řídit celou řadu experimentálních zařízení.

Řídicí jednotka funguje na principu řízení zobrazeném na obrázku č. 7. Ke generování signálu STEP/DIR byla použita řídicí karta vyrobená společností Mesa Electronics [16]. Konkrétně karta Mesa 5i25 [17] v kombinaci s kartou Mesa 7i76 (tzv. daughtercard) [18]. Systém založený na takových komponentách umožňuje řízení nezatížené latencí počítače, umožňuje řídit až 5 motorizovaných os, připojení enkodérů a dalších periferií (digitální i analogové vstupy/výstupy). Jako driver krokových motorů byly použity drivery od společnosti Leadshine Technology [19], konkrétně driver Leadshine M415B [20].

Obrázek č. 49: Ukázka karty Mesa 5i25, převzato z [16]

V rámci implementace vznikl hlavní HAL soubor, který nese informace nutné k řízení (například zapojení karet Mesa v řídicí jednotce apod.). Pak byl ke každému zařízení vytvořen doplňkový HAL soubor obsahující parametry řízení, které jsou v zařízeních odlišné (například informace o koncových spínačích apod.).

```
## AXIS 0 - X ##
setp    hm2_5i25.0.stepgen.00.dirsetup        [AXIS_0]DIRSETUP
setp    hm2_5i25.0.stepgen.00.dirhold         [AXIS_0]DIRHOLD
setp    hm2_5i25.0.stepgen.00.steplen         [AXIS_0]STEPLEN
setp    hm2_5i25.0.stepgen.00.stepspace       [AXIS_0]STEPSPACE
setp    hm2_5i25.0.stepgen.00.position-scale  [AXIS_0]
STEP_SCALE
setp    hm2_5i25.0.stepgen.00.step_type        0
setp    hm2_5i25.0.stepgen.00.control-type     0
setp    hm2_5i25.0.stepgen.00.maxaccel        [AXIS_0]
MAX_ACCELERATION
setp    hm2_5i25.0.stepgen.00.maxvel          [AXIS_0]
MAX_VELOCITY
```

Obrázek č. 50: Ukázka části hlavního HAL souboru

Ke každému zařízení byl vytvořen inicializační soubor, kterým se také nastavují parametry řízení. Obsahuje informace o umístění výše zmiňovaných HAL souborů, grafickém uživatelském rozhraní apod. Dále obsahuje parametry, které ovlivňují funkce řídicího software postupně popsané v kapitole 4.

```
[EMC]
MACHINE = 3AxisTable
DEBUG = 0

### RAPO - setup
OM=0
GAIN = 51
INVERT_LOADCELL_POLARITY = 1

[DISPLAY]
DISPLAY = run_qt
EDITOR = gedit
POSITION_OFFSET = RELATIVE
POSITION_FEEDBACK = ACTUAL
MAX_FEED_OVERRIDE = 1.2
INTRO_GRAPHIC = linuxcnc.gif
INTRO_TIME = 3
PROGRAM_PREFIX = ./ngc_files
INCREMENTS = 5mm 1mm .5mm .1mm .05mm .01mm .005mm
```

Obrázek č. 51: Ukázka části inicializačního souboru

Řídicí software je kompatibilní s parametry uvedenými v inicializačním souboru, a tak lze libovolně parametricky připojovat a odpojovat motorizované osy a periferie kompatibilní s řídicí jednotkou. Software se automaticky adaptuje definovaným parametrům a podle nich vygeneruje ovládací prvky, zobrazovací oblasti apod.

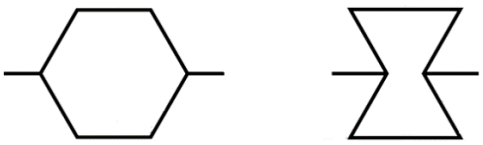
Po implementaci softwaru byla provedena kalibrace měřené síly porovnáním vyčítané síly ze softwaru s hodnotou vyčítanou z referenčního siloměru. Přesnost pojezdu os byla ověřena pomocí úchylkoměru. Byly také ověřeny všechny ochranné mechanismy řídicího software (např. při přetížení siloměru, funkce koncových spínačů apod.). Úvodní testování proběhlo úspěšně a pro ověření celkové funkčnosti systému byl proveden pilotní experiment.

6 Pilotní experiment

Řídicí systém s implementovaným řídicím software byl nasazen v pilotním experimentu, aby byla ověřena správná funkce ovládacích prvků a správnost pořízených dat. Byla zvolena zkouška jednoosým tlakem řízená posuvem se současným snímáním optickým systémem. Experiment byl proveden na vzorku auxetické struktury [21].

Materiály auxetické struktury mají oproti konvenčním materiálům záporné Poissonovo číslo. Dojde-li tedy k protažení materiálu v jednom směru, dojde k protažení i ve směru kolmém, analogicky pro stlačení. Taková vlastnost dává těmto materiálům schopnost pohltit velké množství deformační energie. Lze je využít k výrobě nárazníků, chráničů apod.

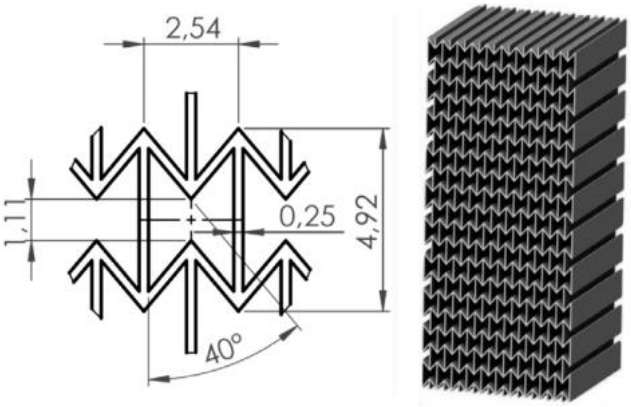
K experimentu byl vybrán vzorek s šestiúhelníkovými buňkami. Šestiúhelníkové buňky se nacházejí i u přírodních materiálů. Například včelí plástve mají tvar pravidelných šestiúhelníků, a proto se k označení šestiúhelníkových buněk ve struktuře auxetických materiálů vžil pojem honeycomb (anglicky plástev). Buňky, které mají tvar pravidelného šestiúhelníku, ale netvoří auxetickou strukturu. Musí být totiž modifikovány změnou vnitřních úhlů na tzv. inverted honeycomb, jak ukazuje následující obrázek.



Obrázek č. 52: Buňka honeycomb (vlevo) a inverted honeycomb (vpravo)

Výroba materiálu s takovou strukturou je poměrně náročná. Přestože auxetické materiály byly poprvé popsány koncem 20. let 20. století, důležitým mezníkem ve výrobě auxetických materiálů byl až rok 1987, kdy byla poprvé uměle vyrobena modifikací polymerní pěny struktura, která vykazovala záporné Poissonovo číslo. V současné době jsou auxetické struktury nejčastěji vyráběny pomocí 3D tiskáren.

Vzorek použitý k experimentu byl vyroben na 3D tiskárně a má strukturu 2D inverted honeycomb (existuje například i struktura 3D inverted honeycomb, viz [21]). Rozměry použitého vzorku před provedením experimentu byly $25{,}05 \times 25{,}40 \times 37{,}75$ mm (šířka, tloušťka, výška). Charakteristika geometrie buňky ve struktuře vzorku a ukázka vzorku jsou na následujícím obrázku.



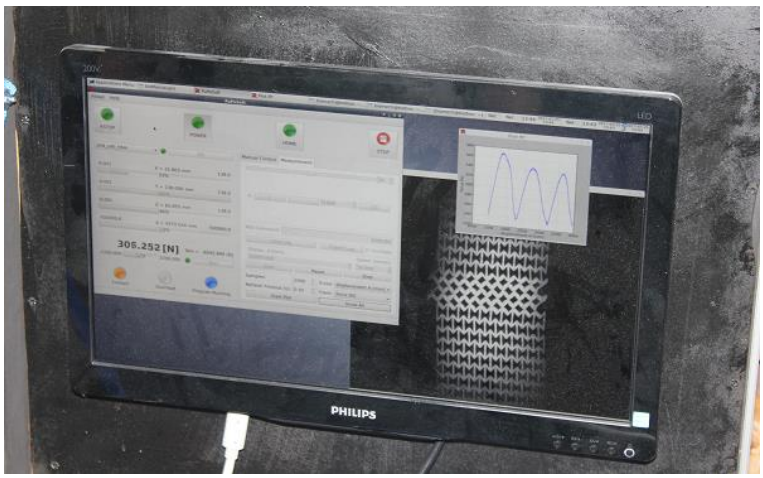
Obrázek č. 53: Ukázka auxetické struktury použité při experimentu, převzato z [21]

Při pilotním experimentu byl vzorek auxetického materiálu podroben zatěžování tlakem pomocí univerzálního jednoosého zatěžovacího zařízení vyvinuté na Ústavu mechaniky a materiálů. Jako senzor síly byl použit siloměr U9B vyrobený společností HBM [22] s rozsahem 2 kN. Pro vyčítání siloměru bylo použito zařízení LabJack T7 [5]. Byl také použit motorizovaný tříosý stolek pro polohování kamerou, která pořizuje obrazová data z experimentu.

Obrázek č. 54: Ukázka experimentální soustavy

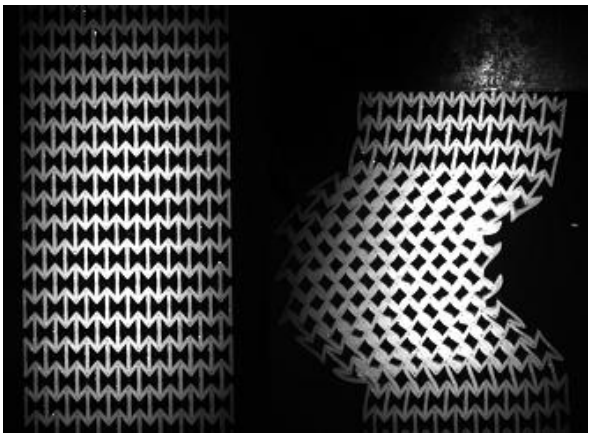
Řídicí software umožňuje ovládat zařízení jednotlivě, ale umožňuje modulárně připojit a ovládat více zařízení zároveň (tento případ), což velmi výrazně zvyšuje variabilitu řídicího systému. Tříosý stolek byl připojen na osy X, Y, Z a jednoosé zatěžovací zařízení bylo připojeno na osu A. V inicializačním souboru je taková kombinace definována

parametry ACTIVE_AXES = X Y Z A (4 aktivní osy) a MEASUREMENT_AXES = A (jedna měřící osa).

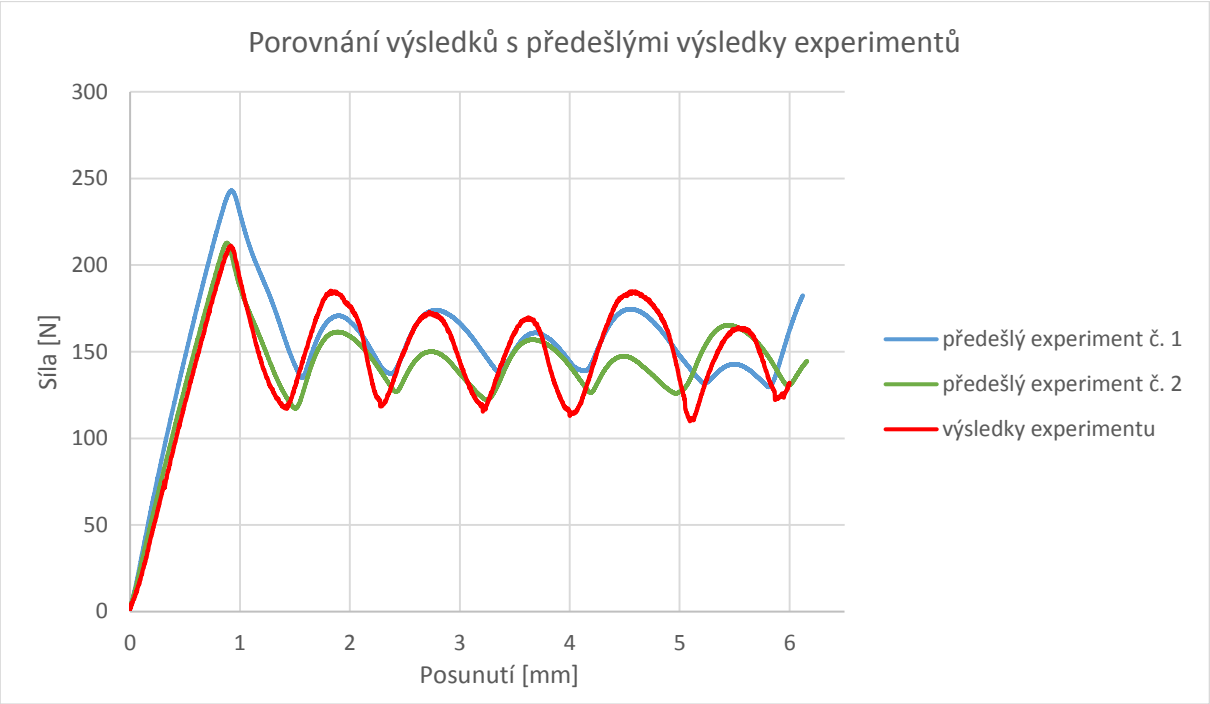


Obrázek č. 55: Ukázka běžící aplikace během experimentu

Po spuštění aplikace bylo polohováno kamerou do pozice, kdy byl obraz kamery zaostřený a zatěžovacím zařízením bylo polohováno do pozice, ve které bylo dosaženo kontaktu se vzorkem. Experiment byl spuštěn pomocí pluginu k tomu určeného (viz 4.3.2.4). Průběh zatěžovací zkoušky byl vykreslován do grafu (viz 4.3.2.5). Experiment byl řízen posunutím, rychlost posuvu byla 50 µm/s a předepsané posunutí bylo 15 000 µm, experiment trval 300 s.



Obrázek č. 56: Vzorek před (vlevo) a po experimentu (vpravo), pořízeno kamerou



Obrázek č. 57: Porovnání výsledků experimentu s výsledky předešlých experimentů

Cílem pilotního experimentu bylo ověřit funkčnost ovládacích prvků v reálném provozu, vyhodnotit pořízená data a ověřit jejich správnost.

Během přípravy experimentu i při samotném provádění experimentu nebyly zaznamenány žádné potíže. Byla ověřena funkčnost modulů pro vyčítání síly ze siloměru, pořízená data byla úspěšně ukládána do matice v aplikaci a průběžně ukládána do výstupního souboru. Po ukončení experimentu byl výstupní soubor úspěšně exportován pomocí dialogu. Pořízená data z výstupního souboru jsou znázorněna na obrázku č. 57. Obrázek č. 57 potvrzuje správnost pořízených dat porovnáním s daty z předchozích experimentů.

7 Závěr

V rámci této bakalářské práce byl vytvořen modulární řídicí software, kterým je možné řídit různá experimentální zařízení na Ústavu mechaniky a materiálů Fakulty dopravní. Práce navazuje na předchozí studentské projekty, v rámci kterých byla vyvinuta řídicí jednotka s veškerou hardwarovou funkcionalitou. Software vytvořený v rámci této práce doplňuje hardwarovou platformu o plnohodnotné softwarové řešení pro řízení stávajících i v budoucnu vyvinutých experimentálních zařízení.

K ovládání experimentálních zařízení byla vytvořena řada modulárních ovládacích prvků v jazyce Python. Aplikace obsahuje ovládací prvky zajišťující základní funkcionalitu zařízení (zapnutí, vypnutí, polohování, ukazatele pozice os apod.), ale také ovládací prvky přímo určené k provádění a obsluhování experimentu (spuštění, pozastavení a ukončení experimentu, správa souboru s výstupními daty, vykreslování grafu apod.). Také byly vytvořeny moduly pro vyčítání síly pomocí zařízení LabJack T7 a Orbit Merret 502T. Bezpečný provoz experimentálních zařízení zajišťuje aplikace pomocí různých ochranných mechanismů (například při přetížení siloměru, pojezd mimo limity os apod.). Funkčnost ovládacích prvků i ochranných mechanismů byla ověřena pomocí pilotního experimentu, který ukázal, že splnila očekávání a může být nasazena v reálném provozu.

Díky modularitě a parametrizaci softwarového řešení je implementace zařízení do řídicího systému zjednodušena, protože se aplikace skládá z jádra, které zajišťuje základní funkce zařízení a z volitelných pluginů, které jsou do aplikace snadno připojitelné. Jádro aplikace je pro všechna zařízení neměnné, všechny odlišné funkce konkrétních zařízení jsou obstarávány pomocí volitelných pluginů. Pokud by tedy bylo nutné implementovat další zařízení a stávající ovládací prvky by nedostačovaly, stačí vytvořit volitelný plugin s požadovanými funkcemi a ten do aplikace implementovat, aniž by bylo nutné jakkoli zasahovat do funkční struktury aplikace. Díky této vlastnosti má řídicí software velký potenciál a může být rozvíjen a využíván dlouhodobě.

Seznam použité literatury

[1] Douglas W. Jones. The University of Iowa – Department of Computer Science [online; navštíveno 25. 5. 2017]. Dostupné z: http://homepage.divms.uiowa.edu/~jones/step/

[2] Speciální krokové motory. [online; navštíveno 26. 5. 2017]. Dostupné z: http://www.servo-drive.cz/specialni_krokove_motory_krokove_motory_na_miru.php

[3] Vladimír Nekvasil. Ovládání krokových motorů – didaktická pomůcka, Bakalářská práce, ČVUT, Praha, 2008. Dostupné z: https://support.dce.felk.cvut.cz/mediawiki/images/7/78/Bp_2008_nekvasil_vladimir.pdf

[4] National Instruments. NI Community – National Instruments [online; navštíveno 2. 6. 2017]. Dostupné z: https://forums.ni.com/t5/Counter-Timer/help-to-check-my-project-convert-digital-signal-count-edge-to/td-p/3190805

[5] T7 Datasheet | LabJack. LabJack | Measurement & Automation [online; navštíveno 10. 6. 2017]. Dostupné z: https://labjack.com/support/datasheets/t7

[6] LJTick-InAmp | LabJack. LabJack | Measurement & Automation [online; navštíveno 10. 6. 2017]. Dostupné z: https://labjack.com/accessories/ljtick-inamp

[7] OM 502T | ORBIT MERRET, spol. s. r. o. Výstavy v roce 2017 [online; navštíveno 11. 6. 2017]. Dostupné z: http://www.merret.cz/produkty/pristroje-pro-mar/pristroje/zobrazovace-pro-tenzometry/om-502t

[8] LinuxCNC. LinuxCNC Documentation [online; navštíveno 22. 3. 2017]. Dostupné z: http://linuxcnc.org/docs/2.7/pdf/

[9] What is Python? Executive Summary | Python.org. Welcome to Python.org [online; navštíveno 15. 3. 2017]. Dostupné z: https://www.python.org/doc/essays/blurb/

[10] PyQt – Python Wiki. Python Software Foundation Wiki Server [online; navštíveno 18. 4. 2017]. Dostupné z: https://wiki.python.org/moin/PyQt

[11] About Qt – Qt Wiki [online; navštíveno 5. 6. 2017]. Dostupné z: https://wiki.qt.io/About_Qt

[12] Python Virtual Control Panel. LinuxCNC [online; navštíveno 13. 6. 2017]. Dostupné z: http://linuxcnc.org/docs/2.4/html/hal_pyvcp.html

[13] Glade Virtual Control Panel. LinuxCNC [online; navštíveno 17. 6. 2017]. Dostupné z: http://linuxcnc.org/docs/html/gui/gladevcp.html

[14] Glade – A User Interface Designer [online; navštíveno 17. 6. 2017]. Dostupné z: https://glade.gnome.org/

[15] Jan Šleichrt. Modulární jednotka řízení experimentálních zařízení pro měření mechanických vlastností materiálů, Diplomová práce, ČVUT, Praha, 2016, Dostupné z: http://mech.fd.cvut.cz/presentation/sleichrt_DP_2016.pdf

[16] Mesa Electronics [online; navštíveno 14. 7. 2017]. Dostupné z: http://mesanet.com/

[17] 5i25 SuperPort i/o card. Mesa Electronics [online; navštíveno 14. 7. 2017]. Dostupné z: http://www.mesanet.com/pdf/parallel/5i25man.pdf

[18] 7i76 step/dir plus i/o daughtercard. Mesa Electronics [online; navštíveno 14. 7. 2017]. Dostupné z: http://www.mesanet.com/pdf/parallel/7i76man.pdf

[19] Leadshine Technology – Home [online; navštíveno 19. 7. 2017]. Dostupné z: http://www.leadshine.com/

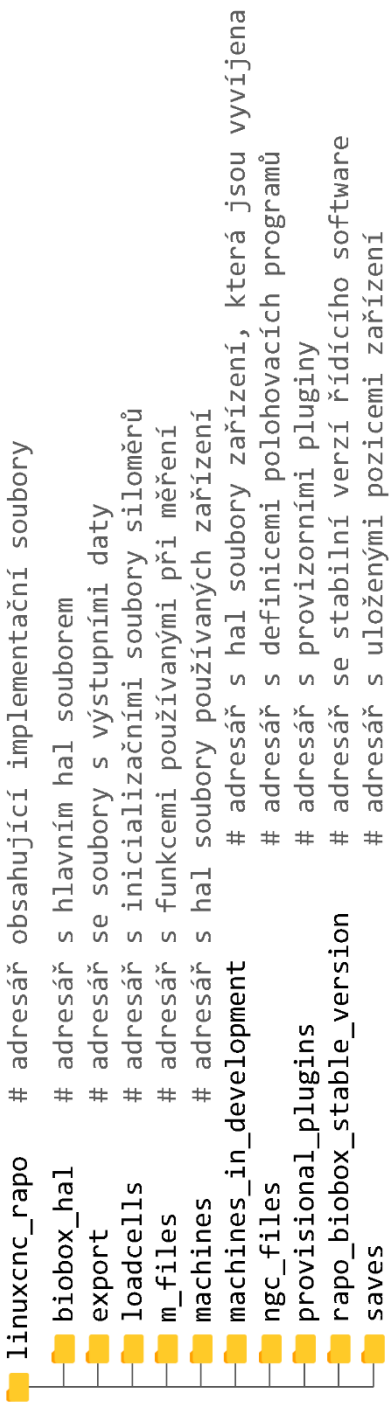
[20] M415B Micro Microstepping Driver. Leadshine Technology [online; navštíveno 22. 7. 2017]. Dostupné z: http://www.leadshine.com/UploadFile/Down/M415Bd.pdf

[21] Michaela Neuhäuserová. Mechanické vlastnosti auxetických struktur určené kvazi-statickými zkouškami, Bakalářská práce, ČVUT, Praha, 2015, Dostupné z: http://mech.fd.cvut.cz/presentation/bak_neuhauserova.pdf

[22] About Us | HBM. HBM Test and Measurement: Transducers, Load Cells [online; navštíveno 4. 6. 2017]. Dostupné z: https://www.hbm.com/en/0011/about-us/

Seznam příloh

Příloha A: Popis adresářové struktury implementačních souborů



Příloha B: Popis adresářové struktury aplikace

Seznam příloh na CD

Textová část bakalářské práce ve formátu PDF

Popis adresářové struktury implementačních souborů ve formátu PDF

Popis adresářové struktury aplikace ve formátu PDF

Adresář s aplikací, verze 167

Adresář s implementačními soubory, verze 0.81