Vysoká škola polytechnická Jihlava
Obor počítačové systémy

Vizualizace a simulace vlakového uzlu s IPC

Bakalářská práce

Autor: Miloš Šimek

Vedoucí práce: Ing. Michal Bílek

Jihlava 2012

Vysoká škola polytechnická Jihlava

Tolstého 16, 586 01 Jihlava

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Autor práce:	**Miloš Šimek**
Studijní program:	Elektrotechnika a informatika
Obor:	Počítačové systémy
Název práce:	**Vizualizace a simulace vlakového uzlu s IPC**
Cíl práce:	Popište model a komunikační protokol pro řízení reálného modelu. Vytvořte vizualizace a simulace vlakového uzlu s IPC pro existující model vlakového uzlu v různých variantách dle počtu využitých vstupů a výstupů. Vizualizaci a simulaci navrhněte ve vhodně zvoleném prostředí tak, aby bylo možné odzkoušet řídící programy pro PLC. Vytvořte řídicí program v prostředí Twincat pro demonstraci vlastností příslušného modelu. Vytvořte vzorové řízení modelu dle vámi vybraného PLC. Navrhněte sadu zadání laboratorních úloh s odstupňovanou obtížností včetně vzorových řešení vše odzkoušejte přes vzdálenou laboratoř LABLINK.

Ing. Michal Bílek
vedoucí bakalářské práce

Ing. Bc. Michal Vopálenský, Ph.D.
vedoucí katedry
Katedra elektrotechniky a informatiky

Anotace

Cílem této bakalářské práce bylo vytvořit program, simulující chování fyzického železničního modelu, sloužícího pro studium programování programovatelných logických automatů. Tento simulační program přinese užitek studentům, kteří si budou moci otestovat funkčnost svých studijních programů přes vzdálenou laboratoř LabLink přímo z domova.

V této práci je popsán fyzický model, jeho stavba a fungování. Dále je popsána komunikace pomocí digitálních/analogových vstupů a výstupů společně s druhou možností komunikace, a to pomocí sériového portu RS232.

Další část popisuje strukturu a fungování navrženého simulačního programu.

Klíčové pojmy

Vizualizace, simulace, IPC, Beckhoff, TwinCAT, model, železniční uzel, řízení, program, programování, C++, strukturovaný text, vzdálená laboratoř, LabLink, studium.

The Summary

The purpose of this thesis for a bachelor´s degree is to create a program, which simulates the showing of physical railway model created for study of programmable logical controller programming. This simulation program will benefit students who will be able to test their study programs for functionality through a remote laboratory LabLink right from home.

The physical model, its construction and functioning are described in this thesis and also a communication with digital/analog inputs and outputs together with the second option - communication through a serial port RS232.

Another part describes the structure and functioning of designed simulation program.

Key words

Visualization, simulation, IPC, Beckhoff, TwinCAT, model, railway node, control, program, programming, C++, structured text, remote laboratory, LabLink, study.

Poděkování

Rád bych poděkoval Ing. Michalovi Bílkovi za zadání práce, která mi umožnila hlouběji porozumět tématice programování aplikací. Dále bych chtěl poděkovat za jeho odborné vedení, rady a připomínky, které mi poskytl při zpracování této bakalářské práce.

Prohlašuji, že předložená bakalářská práce je původní a zpracoval/a jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil/a autorská práva (ve smyslu zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, v platném znění, dále též „**AZ**“).

Souhlasím s umístěním bakalářské práce v knihovně VŠPJ a s jejím užitím k výuce nebo k vlastní vnitřní potřebě VŠPJ.

Byl/a jsem seznámen/a s tím, že na mou bakalářskou práci se plně vztahuje **AZ**, zejména § 60 (školní dílo).

Beru na vědomí, že VŠPJ má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom/a toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem VŠPJ, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených vysokou školou na vytvoření díla (až do jejich skutečné výše), z výdělku dosaženého v souvislosti s užitím díla či poskytnutím licence.

V Jihlavě dne

.....

Podpis

Obsah

1 Úvod

Problematika vzdáleného ovládání laboratoří je čím dál více používaná. Není se čemu divit. Dříve bylo studium látky zaměřené na praxi složitější, neboť si student mohl praktickou část ozkoušet jen v laboratoři a ne doma při studiu, když je to nejvíce potřeba. Byl tu i další problém, a to je to, že velice často je laboratorních modelů méně než studentů, což znemožňuje, aby studenti získali potřebnou praxi přímo na cvičení.

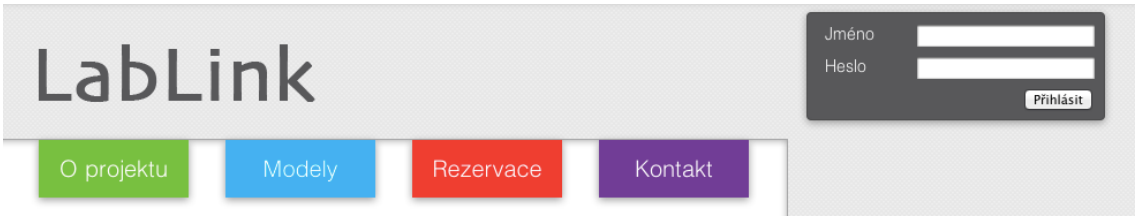
Proto vznikla díky rozmachu dnešních moderních technologií a hlavně internetu další možnost, a tou je právě to, že se student připojí do laboratoře vzdáleně, příkladně z domova a otestuje si své dovednosti. Tento způsob skýtá obrovské výhody. Student se může připojit do laboratoře kdykoli a odkudkoli, což usnadňuje práci nejenom studentům, ale i učitelům, kteří nemusí být v laboratoři přítomni.

Přispět k tomuto zjednodušení praktického studia je cílem této bakalářské práce. A to ve formě programu, který simuluje fungování laboratorního vlakového modelu, určeného pro řízení programovatelným automatem. Tento program pak lze spustit na dálku, kde se student bude pokoušet tento simulovaný model řídit. Při simulaci není potřeba přítomnosti žádného studenta ani učitele v laboratoři.

1.1 Rešerše

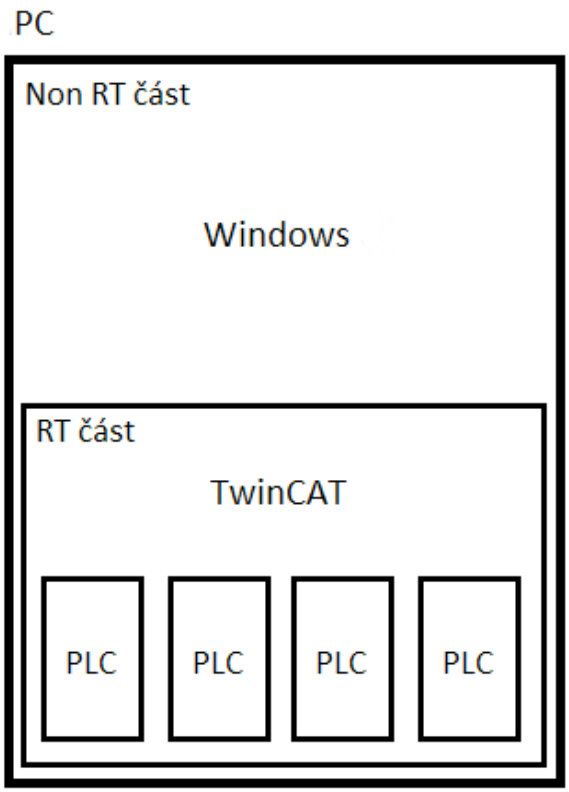
Ze začátku pár pojmů: „Vzdálená internetová laboratoř je skutečný reálný pokus (fyzikální, chemický …), připojený pomocí řídicího počítače (serveru) k celosvětové síti Internet. Vzdálení uživatelé prostřednictvím ovládacího webového rozhraní experimentu přes internetové spojení ovládají experiment a měří relevantní data.” [1]

„LabLink je vzdálenou laboratoří usnadňující výuku řídicí techniky. Hlavní výhody jsou časově i prostorově neomezený přístup do laboratoře, zapojení virtuálních a reálných modelů, jejich efektivní využití a použití standardního software, používaného v průmyslu pro řízení jednotlivých modelů. Systém obsahuje webovou aplikaci, server virtuálních ploch, reálné a virtuální modely. LabLink byl vytvořen na katedře řídicí techniky elektrotechnické fakulty ČVUT v Praze.” [2]



Obr. 1 – Přihlašovací obrazovka systému LabLink [2]

TwinCAT je systém od firmy Beckhoff, fungující pod operačním systémem Windows. Je schopen vytvořit až čtyři virtuální PLC. Dále umožňuje přes ethernetový port počítače připojit vstupně–výstupní terminály určené pro řízení industriálního procesu. TwinCAT umožnil v této práci vytvořit rozhraní terminálů stejné, jako má fyzický vlakový model.



Obr. 2 – Grafické znázornění systému TwinCAT [0]
RT část – část fungující v režimu reálného času
Non RT část – část nefungující v režimu reálného času

Laboratorní model železničního uzlu, který je v rámci této práce simulován, byl vytvořen na zakázku. Vzhledem k této skutečnosti, není k jeho simulaci pro vzdálený přístup žádná přímá konkurence (v době psaní této práce).

Vysoká škola polytechnická Jihlava (dále jen VŠPJ) ale vlastní program, simulující jiný vlakový model. Tento model je vytvořen pouze programově a nemá svoji fyzickou obdobu, což zamezuje ozkoušení studentských řídicích programů na reálném zařízení. Dále tento systém vyžaduje mít v laboratoři VŠPJ aktivní obrazovku, zobrazující daný model a webovou kameru, která obrazovku snímá. Simulační program vytvořený v rámci této bakalářské práce obrazovku ani webkameru v laboratoři nepotřebuje.

1.2 Cíl a řešení

Jak jsem již zmínil v úvodu, cílem této bakalářské práce je zjistit fungování železničního modelu (který bude popsán v další kapitole) a vytvořit program, který toto fungování napodobí. S tímto programem se pak bude pracovat přes vzdálenou laboratoř LabLink. Teoreticky by bylo možné i s fyzickým modelem pracovat přes vzdálenou laboratoř, ale toto řešení by mělo jedno velké úskalí a tím je chybějící bezobslužnost.

Fyzický vlakový model se ovládá přes vstupy a výstupy, na které se může napojit libovolný programovatelný automat (dále jen PLC). Toto PLC bude model řídit. Takto je možné využít PLC od různých výrobců, naprogramovaných v různých programovacích jazycích. Tento požadavek musí splnit i simulace modelu.

Finální řešení zadaného úkolu vypadá následujícím způsobem. Pro řešení byla zvolena sestava systému reálného času TwinCAT společně s programem simulujícím chod fyzického modelu pojmenovaného „Railway Simulator". Tyto dvě části spolu komunikují a každá řeší část problému.

Součástí tohoto systému programů je i vizualizace, vytvořená v prostředí TwinCAT, zobrazující aktuální stav simulovaného modelu. Tu si student bude moct zobrazit přes vzdálenou laboratoř LabLink. Kompletní popis fungování je popsán v kapitole „Simulátor vlakového modelu".

2 Popis fyzického modelu

Fyzický model byl sestaven jako nástroj pro studium programování programovatelných automatů. Obsahuje samotnou sestavu skládající se z kolejnic, 2 vlaků, výhybek, semaforů a řídící jednotky. Tato konstrukce umožňuje jak přímé ovládání pomocí vstupů a výstupů modelu, tak přes sériovou linku RS232. Díky tomu může student zkoušet ruční ovládání přímo z počítače a později navrhnout program, který bude model železničního uzlu (dále jen MŽU) řídit automaticky.

Řídící elektronika modelu je zařízena pomocí mikrokontroléru LPC1754 (Cortex–M3), který umožňuje výše popsané ovládání. [3]

Obr. 3 – Vzhled fyzického modelu [3]

2.1 Součásti modelu

Model je složen z více typů zařízení. Jsou zde dva vlaky, které se napájí pomocí koleček přes železnici. Železnice má tvar ležaté osmičky a je rozdělena do sedmi částí, které lze napájet zvlášť. Toto provedení umožňuje ovládat oba vlaky nezávisle na sobě z hlediska rychlosti. Je možné, aby byly oba vlaky na stejném napájecím segmentu. S tím se musí v kontrolním programu (programovatelného automatu připojeného k modelu) počítat, aby nedošlo ke srážce. Segmenty kolejiště jsou propojeny výhybkami, kterých je celkem šest.

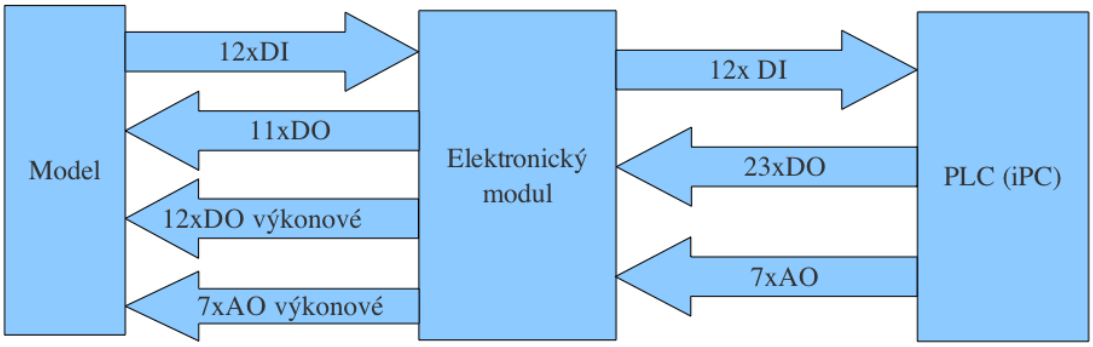
Pro úspěšné ovládání je potřeba, aby měl kontrolní program (nebo student) zpětnou vazbu ohledně pozice vlaků. Proto je po MŽU rozmístěno dvanáct indikátorů pozice, na které může program studenta reagovat.

Další součástí jsou semafory. Model jich obsahuje celkem jedenáct. Zajímavé je, že ovládání semaforů je naprosto nezávislé na ovládání vlaků, takže příkladně může jet vlak i na červenou. Jde o další záležitost, na kterou se bude muset v kontrolním programu myslet. Díky oddělenosti semaforů od vlaků, také může student semafory ignorovat a věnovat se jenom programu pro ovládání vlaků. Což umožnuje se rychleji seznámit s programováním modelu.

Na jedné straně modelu je i nádraží. Kromě domečku má i praktickou funkci. Z pohledu ovládání jsou zde 2 výhybky a 2 segmenty napájení vlaku. Toto rozložení umožňuje vzájemné prostřídání vlaků, kdy je jeden vlak odstaven na vedlejší kolej.

Celý model včetně elektroniky je napájen zabudovaným napájecím zdrojem poskytujícím 12 V a 8 A. [3]

2.2 Popis ovládání



Obr. 4 – Blokové schéma ovládání modelu [3]

Přímé ovládání systému je umožněno 30. vstupy a 12. výstupy. 12 digitálních výstupů je vyhrazeno signalizaci projetí vlaku. Tato signalizace je provedena tak, že na obou vlacích je na spodní straně umístěn magnet. V místech, kde se snímá pozice je mezi kolejemi jazýčkový spínač, jehož kontakty jsou vlivem magnetu při projíždění vlaku spojeny. V této chvíli se na výstupu objeví logická hodnota jedna. Vzhledem k tomu, že daný impulz je při vyšších rychlostech vlaku velice krátký, byla doba signálu projetí vlaku prodloužena na 2 sekundy. [3]

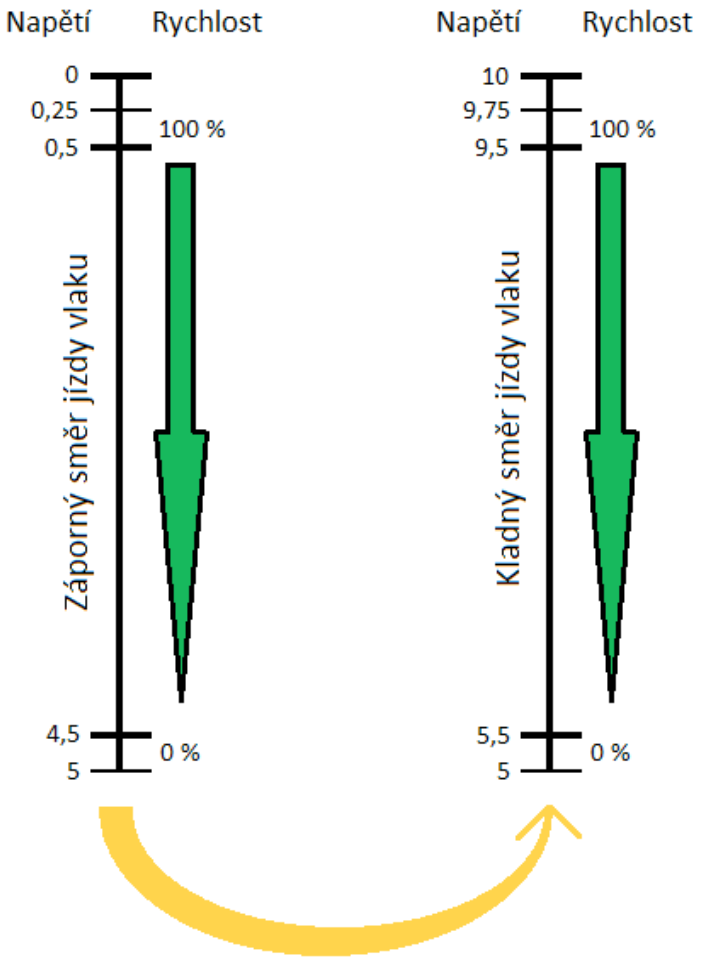
Pro řízení semaforů slouží 11 digitálních vstupů. Semafory není možné vypnout. Vstupy jsou pro ovládání svítící barvy semaforu. V případě, že je na vstupu semaforu

logická hodnota 0, semafor svítí zeleně. Na druhou stranu logická hodnota 1 přepne semafor, aby svítil červeně. Vždy ale bude svítit buď zelená nebo červená dioda. [3]

Dalšími 12 digitálními vstupy se ovládá přepínání výhybek. Na rozdíl od semaforů připadají na každou výhybku 2 digitální vstupy. Přepnutí do druhé polohy výhybky je aktivováno logickou hodnotou 1 na příslušném digitálním vstupu. Výhybka nepotřebuje být pro udržení pozice napájena neustále. Proud je vyžadován jen pro přepnutí výhybky. Mohlo by se stát, že pokud bude studentův řídící program chybný, bude do výhybky přiveden proud po dlouhou dobu. V takovémto případě hrozí přehřátí mechanického ovládání výhybky. Této poruše zabraňuje elektronika modelu, která omezuje dobu napájení. Omezení je nastaveno tak, že čas, kdy teče do výhybky proud, je osminový oproti době předchozího vypnutí výhybky. To znamená, že pokud se s výhybkou nebude pracovat 8 sekund, vnitřní elektronika omezí trvání přepínacího impulzu výhybky na maximálně jednu sekundu. [3]

Problém by mohl nastat, pokud najede vlak na špatně přehozenou výhybku. V tomto případě dojde ke zkratu a zastavení vlaku. Pokud ale přehodíme výhybku, tak bude model schopen pokračovat v provozu i bez ručního zásahu. [3]

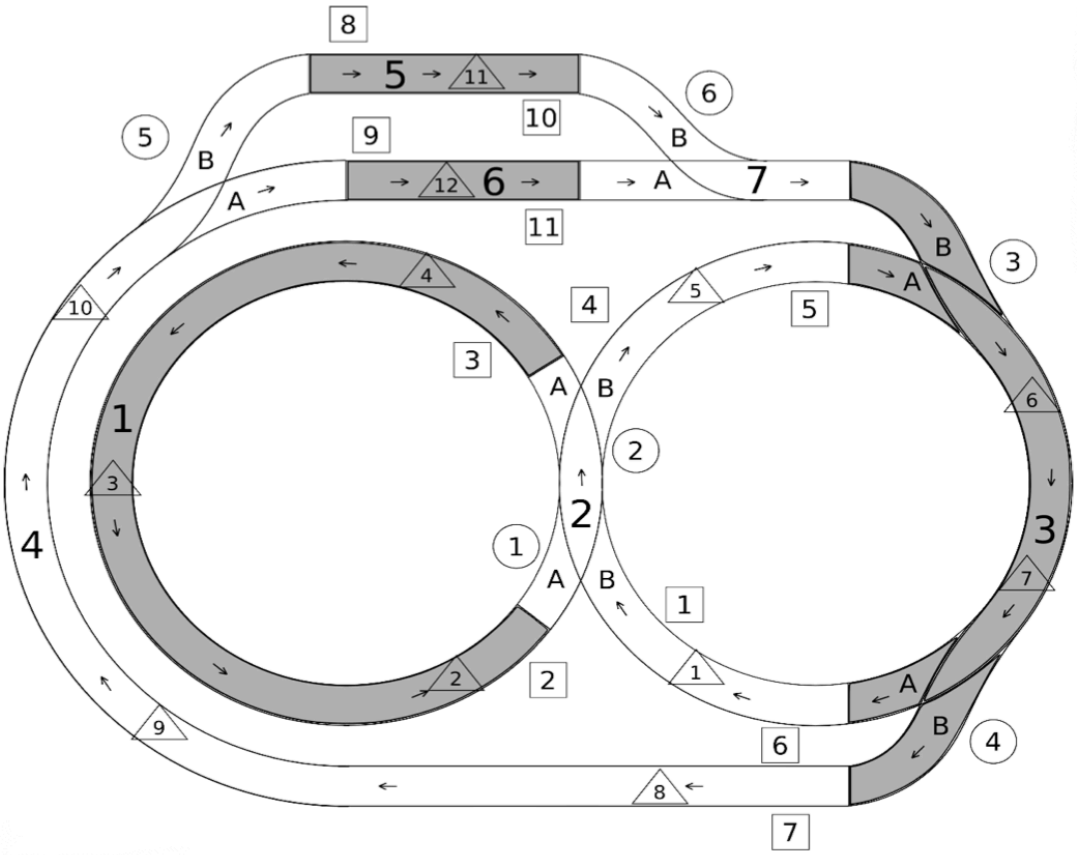
Sedm analogových vstupů ovládá napájení segmentů MŽU. Segmenty jsou ovládány napětím 0 – 10 V. Vlaky mohou jezdit oběma směry. Aby toho bylo možné dosáhnout s tím, že na analogových vstupech bude jenom kladné napětí, je napájecí rozsah rozdělen na dvě části. Při napájení segmentu napětím 0 – 5 V pojede vlak v jednom směru jízdy, a pokud bude napájen segment 5 – 10 V, pojede ve směru druhém. Z ochranných důvodů nemohou jezdit vlaky v obou směrech zároveň. [3]



Obr. 5 – Napětí v závislosti na rychlosti vlaku
(hodnoty napětí jsou ve voltech) [0]

2.3 Varianty ovládání

Kompletní ovládání MŽU je složitější záležitost, zvlášť pro studenta, který se bude tento systém učit ovládat. Proto jsem rozdělil ovládání do tří variant podle počtu vstupů a výstupů, které bude potřeba ovládat pro úspěšné řízení. Všechny tři varianty ovládání jsou implementované i v simulačním modelu, který bude popsán později.



Obr. 6 – Technický nákres MŽU s vyznačeným kladným směrem jízdy vlaku [3]

Legenda:

- číslo v kroužku – výhybka
- A, B – polohy výhybky
- číslo ve čtverci – semafor
- číslo v trojúhelníku – snímač projetí vlaku
- samotné číslo – napájecí segment

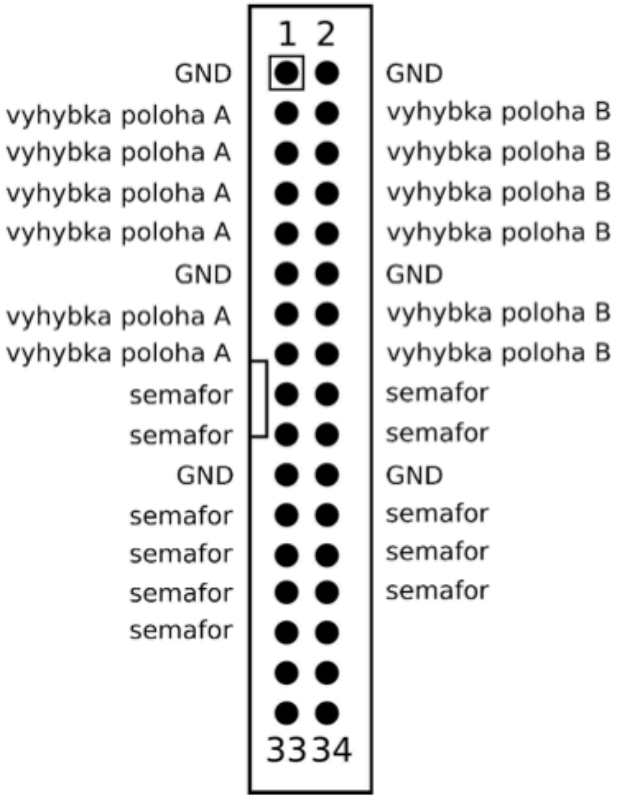
2.3.1 Minimální varianta ovládání

V této variantě se bude využívat nejmenší počet vstupů a výstupů, které je potřeba pro úspěšné ovládání modelu. Cílem je řízení průjezdu vlaku po vnějším obvodu. K tomu bude potřeba přístup ke čtyřem výhybkám s číselným označením: 3, 4, 5, 6 na obrázku číslo 6. Výhybky číslo 3 a 4 je potřeba ovládat, neboť jejich základní nastavení (pozice A) neumožňuje průjezd vlaku po vnějším obvodu, a proto je bude potřeba přehodit do pozice B. Přehazování výhybek 5 a 6 může být součástí zadání pro studenta. Každá výhybka vyžaduje 2 digitální vstupy, což je celkově 8 vstupů.

Jak již bylo výše řečeno, rychlost vlaků je řízena pomocí sedmi analogových vstupů. Pro minimální variantu ovládání bude stačit využití jednoho vstupu s tím,

že jeho napětí se přesměruje na zbylých 6. Toto řešení umožňuje řídit rychlost vlaku jedním vstupem pro celý model.

Každá úloha řízení modelu vyžaduje mít možnost zjišťovat pozici vlaku pomocí snímačů projetí. Proto i v minimální variantě budou zpřístupněny všechny snímače, což je 12 digitálních výstupů. V případě nutnosti by bylo možné toto číslo snížit o 5 opomenutím vnitřních snímačů.



Obr. 7 – Rozmístění digitálních vstupů modelu MŽU [3]

2.3.2 Střední varianta ovládání

Ve střední variantě je umožněno řízení vlaku na celém modelu. K tomu je potřeba řídit všechny výhybky, což je 12 digitálních vstupů. Řízení rychlosti vlaku se provede stejným způsobem, jako v minimální variantě, jedním analogovým vstupem. Pro zjišťování pozice se využije všech 12 snímačů projetí vlaku.

2.3.3 Maximální varianta ovládání

Při použití maximální varianty ovládání, bude možné ovládat celý model bez omezení. Odlišnost od střední varianty je ta, že rychlost řízení vlaku již není řízena

pomocí jednoho vstupu, ale pomocí všech sedmi vstupů. Toto řízení umožňuje různé rychlosti na různých částech MŽU. Možná chyba, kterou bude potřeba při řízení vzít v úvahu, je možnost napájení segmentů ve vzájemně protichůdných směrech. Pokud k této situaci dojde, tak se MŽU deaktivuje. Jde o řídící chybu, která by se při řízení neměla nikdy stát.

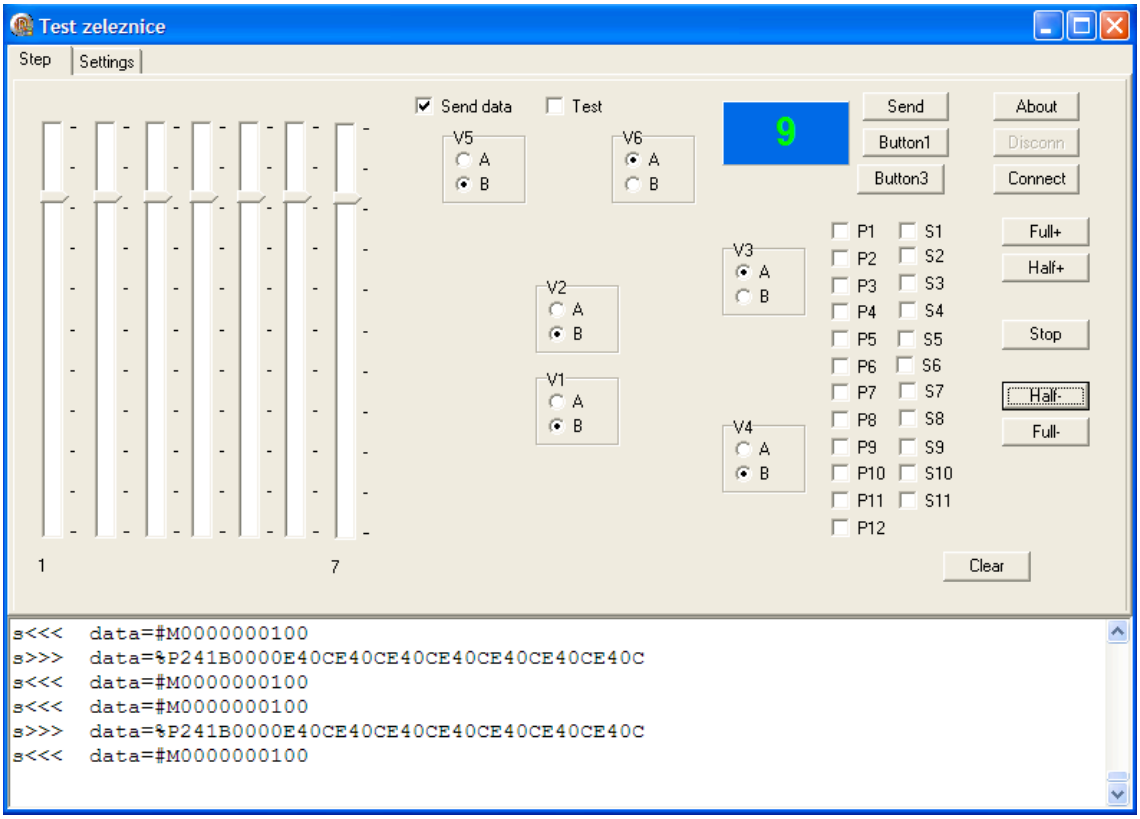
U maximální varianty je také možné využít řízení semaforů.

2.4 Popis ovládání pomocí RS232

Výše bylo popsáno přímé ovládání přes vstupy a výstupy. Další z možností ovládání, které umožňuje zabudovaný mikrokontrolér v MŽU je ovládání pomocí sériové linky RS232.

RS232 umožňuje propojení a vzájemnou sériovou komunikaci dvou zařízení. Na rozdíl od ostatních technologií přenosu, jako je Ethernet nebo USB se z pohledu modelu ISO/OSI jedná čistě o fyzickou vrstvu, kde jsou data přenášena v sérii. Datový přenos je typu full duplex, který data vysílá a přijímá nezávisle na sobě. Aby tento požadavek mohl být splněn, využívá se pár vodičů pro přenos v každém směru. RS232 má oproti technologii USB z hlediska řízení jednu obrovskou výhodu. A to je přenos způsobem vhodným pro „realtimové" využití. USB tento typ přenosu nenabízí, neboť v případě, že více zařízení spolu komunikuje přes USB, je každému zařízení přidělen čas pro komunikaci. A to, jestli stihne za toto časové kvantum odeslat všechna data, se neřeší. [4]

Pro ovládání a testování MŽU byla vytvořena aplikace, umožňující kontrolu všech prvků modelu nezávisle na sobě.



Obr. 8 – Aplikace ovládající MŽU přes RS232 [0]

V případě našeho vláčkového modelu má sériová komunikace tyto parametry:

- Rychlost přenosu: 38400 bd
- Počet datových bitů: 8
- Počet stop bitů: 1
- Parita žádná
- Kontrola průtoku dat: žádná

Z pohledu programového kódu se využívají pro komunikaci 2 datové struktury. Poté, co se do struktur vloží požadovaná data, se převedou na bitové pole a pošlou po sériové lince. Zde je jejich deklarace v jazyku Pascal:

```
typedef struct __attribute__((packed))
{
  unsigned char VA;
  unsigned char VB;
  unsigned short Sem;
  short Seg[7];
} tDataZel;


typedef struct __attribute__((packed))
{
  unsigned char VA;
  unsigned char VB;
  unsigned short VL;
  unsigned char in;
} tMerZel;
```

3 Sada zadání laboratorních úloh

V předchozí kapitole jsme se mohli dočíst o variantách ovládání MŽU. Tyto varianty slouží pro sadu zadání laboratorních úloh. Jde o tři úlohy, které se student může snažit splnit v rámci studia programovatelných automatů. Všechny úlohy jsou implementovány v simulátoru vlakového modelu.

První, nejsnazší úloha vyžaduje alespoň jednu aktivaci všech senzorů projetí vlaku po obvodu MŽU a obou senzorů u nádraží. Toto zadání vyžaduje alespoň jednou, při průjezdu vlaku prohodit obě nádražní výhybky (na obrázku 6 s číslem 5 a 6). Při průjezdu se nesmí stát žádná z řídicích chyb (které jsou taktéž implementovány v simulátoru vlakového modelu). Jmenovitě jde o: najetí vlaku na špatně přehozenou výhybku, přehození výhybky ve chvíli, kdy se na ní nachází vlak, napájení obou vstupů výhybky zároveň, projetí vlaku kolem semaforu s rozsvíceným červeným světlem a nastavení napájecích segmentů s protichůdným směrem jízdy vlaku. Ne všechny se ale v nejsnazší úloze mohou stát, neboť při plnění této úlohy je v implementaci aktivní minimální varianta ovládání.

Druhá úloha se střední obtížností implementuje střední variantu ovládání. Úkolem je alespoň jednou aktivovat všechny senzory pozice vlaku na celém MŽU. Tento úkol bude splněn pouze tehdy, pokud vlak projede všemi segmenty. Tato úloha (stejně jako ta předchozí) nechává prostor pro různé možnosti řešení, neboť není stanoveno, v jakém pořadí se musí senzory aktivovat. Jediné, co je pro úspěšné splnění nutné je to, aby u všech senzorů proběhla aktivace alespoň jednou.

Třetí úloha je velice podobná té druhé s tím rozdílem, že implementuje maximální variantu ovládání. Student zde musí projet vlakem celý MŽU a zároveň ho celý ovládat, včetně všech napájecích segmentů.

Simulátor dále implementuje ještě možnost bez úlohy, kdy nedochází ke kontrole průjezdu vlaků. Řídicí chyby jsou ale kontrolovány stále.

4 Simulátor vlakového modelu

V této kapitole je rozebrána architektura celého systému simulující MŽU. Po vysvětlení základního rozdělení systému se zaměřím na jednotlivé části a detailněji popíši jejich fungování.

4.1 Architektura simulačního systému

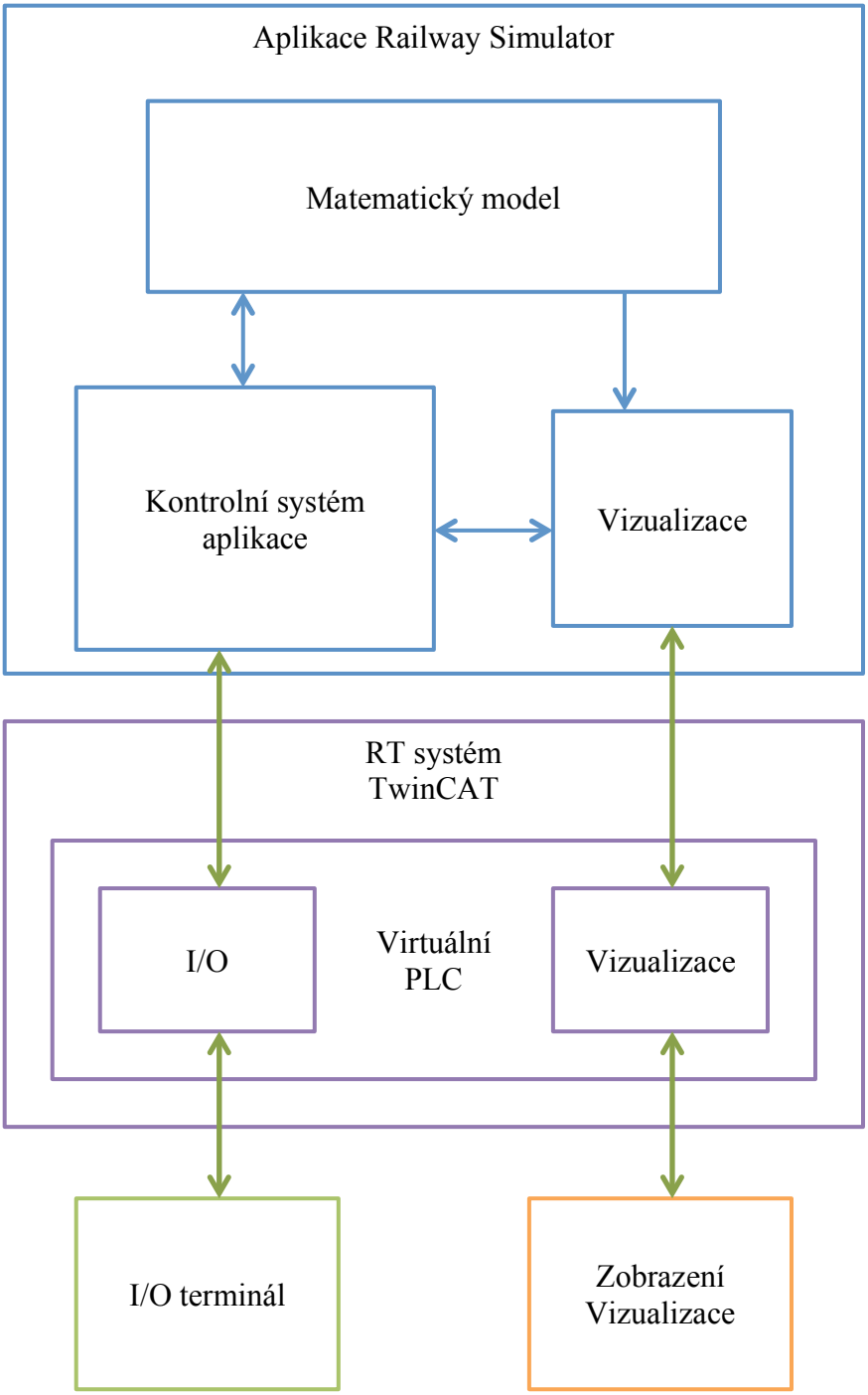
Simulační systém je složen ze dvou vzájemně propojených celků. Tím je nakonfigurovaný systém TwinCAT a aplikace „Railway Simulator" (vše je zobrazeno na obrázku číslo 9). Tyto celky fungují na počítači pod operačním systémem Windows. Pro fungování musí mít zmíněný počítač připojený vstupně–výstupní terminál (na obrázku jako I/O terminál), který je nakonfigurovaný, aby se k němu mohlo připojit PLC stejným způsobem, jako k MŽU.

Terminál komunikuje přímo se systémem TwinCAT, který musí mít nastaveno alespoň jedno virtuální PLC. V tomto PLC jsou dvě části naprogramované ve strukturovaném jazyku. Proměnné, sloužící pro komunikaci s terminálem (na obrázku popsané jako I/O) a grafická vizualizace, která zobrazuje aktuální stav simulovaného modelu a umožňuje ho ovládat.

S vstupně–výstupními proměnnými i s vizualizací ve virtuálním PLC komunikuje aplikace „Railway Simulator". Jde o aplikaci napsanou v programovacím jazyce C++ běžící pod Windows nativně. Tato aplikace se dá rozdělit na tři základní části: kontrolní systém aplikace, vizualizaci a matematický model.

Matematický model je jádro fungování celé aplikace. Jeho úkol se simulovat MŽU se všemi jeho částmi. Jde o nejrozsáhlejší programovou část.

Kontrolní systém aplikaci kontroluje (a tím zároveň celý simulační systém). Také ji umožňuje ovládat. Získává ze systému TwinCAT informace o vstupních proměnných (na obrázku I/O), které může i upravit a předat matematickému modelu. Od matematického modelu získává informace o výstupech, které předává zpátky do sytému TwinCAT, aby se mohly objevit na vstupně–výstupním terminálu. Dále také komunikuje s vizualizací.



Obr. 9 – Blokové schéma architektury simulátoru [0]

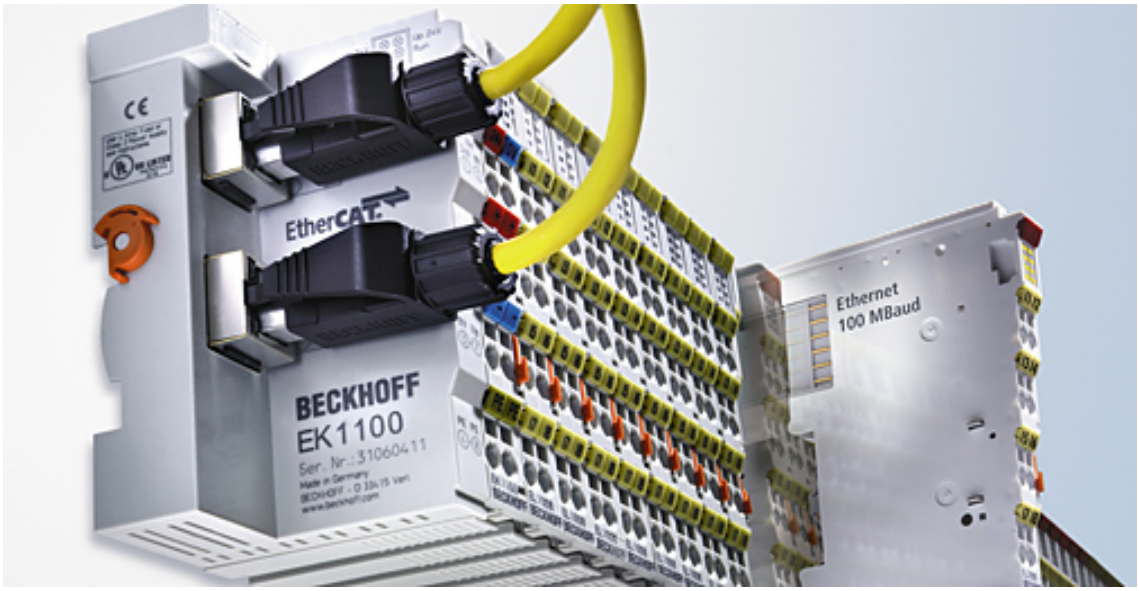
(Zeleně vyznačena meziprogramová komunikace.)

Vizualizační blok v aplikaci „Railway Simulator" je rozhraní mezi vizualizací v sytému TwinCAT a zbytkem kódu simulační aplikace napsané v C++. Toto rozhraní

automaticky získává aktuální stav matematického modelu, který předává do virtuálního PLC pro zobrazení. Jak je na obrázku vidět, vizualizace komunikuje s matematickým modelem jednosměrně. Jednosměrný tok informací totiž umožňuje, aby byl matematický model na vizualizaci nezávislý. Nezávislost matematického modelu na vizualizaci by se dala využít při příkladném budoucím rozšiřování aplikace.

4.2 Vstupy a výstupy

4.2.1 Terminály



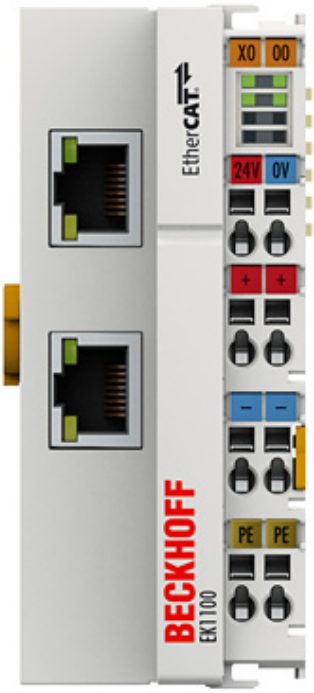
Obr. 10 – Zobrazení terminálů řady EtherCAT od Beckhoffu [5]

Vstupy a výstupy jsou řešeny pomocí terminálů od firmy Beckhoff řady EtherCAT a jsou k počítači připojeny pomocí ethernetového kabelu. Tyto terminály jsou vyrobeny takovým způsobem, aby je bylo možné libovolně kombinovat připojováním za sebe, jak je vidět na obrázku 10. Pro náš simulátor je konfigurace následující (v pořadí napojení):

- EK1100
- EL1008
- EL1008
- EL1008
- EL1008
- EL2008
- EL2008
- EL3068

Podle konfigurace je vidět, že je použito osm EtherCAT terminálů. Jejich detailní specifikaci lze nalézt na stránce www.beckhoff.com. Zde popíši jejich základní funkci a k čemu v simulátoru slouží.

Terminál EK1100 slouží pro propojení komunikace mezi ethernetovým rozhraním a ostatními terminály. Krom předávání informací všechny terminály také napájí.



Obr. 11 – EK1100 z přední strany [6]

Napájení obstarává napájecí zdroj od firmy AXIMA AXSP3P06, který je schopen dodávat 6 ampér při 24 voltech. Maximální příkon je udáván na 196 W. Zdroj využívá standardní napájení ze sítě, tzn. 230 V a 50 Hz. [7]

Obr. 12 – Napájecí zdroj AXIMA AXSP3P06 [7]

EL1008 je terminál digitálních vstupů. Obsahuje jich celkově osm a jsou řízeny napětím. Jako „logická nula" se bere hodnota napětí -3 až +5 V a jako „logická jednička" hodnota 15 až 30 V. Pro každý vstup je na terminálu vyhrazena dioda pro signalizaci logické hodnoty. V naší konfiguraci jsou tyto terminály celkově čtyři. [8]

První dva terminály EL1008 slouží pro vstupy výhybek. Těch je celkově 6 a každá potřebuje dva vstupy, což je celkově 12 vstupů. Pro jednu výhybku slouží „logická jednička" jednoho vstupu pro přehození výhybky do pozice A a „logická jednička" na druhém vstupu pro přehození do pozice B (pozice jsou znázorněny na obrázku 6).

Další dva terminály jsou vyhrazeny pro vstupy semaforů. Těch je celkově 11, kde „logická nula" znamená, že je na semaforu zelená barva a „logická jednička", že semafor svítí červeně. Nastavení je stejné jako u MŽU.

Obr. 13 – Přední pohled na EL1008 [8]

Terminály EL2008 jsou celkově dva. Jde o terminály zajišťující digitální výstupy. Na každém je jich osm. V rámci simulátoru jsou využity pro předávání hodnot senzorů projetí vlaku zpět k řídícímu PLC. Senzorů je celkem 12, což znamená, že 4 výstupy nebyly využity.

V konfiguraci je ještě jeden terminál EL3068. Obsahuje osm analogových vstupů. Vstupy jsou schopny přebírat hodnoty napětí od 0 do 10 V. Terminál je využit pro získávání hodnot napětí pro simulované napájecí segmenty (těch je celkově 7).

4.2.2 Vstupní a výstupní proměnné ve virtuálním PLC

Celkově se používá 42 vstupů a výstupů. Aby hodnoty těchto vstupů a výstupů bylo možné v simulátoru získávat, bylo potřeba vytvořit stejný počet proměnných ve virtuálním PLC (I/O na obrázku 9). Tyto proměnné bylo nutné v konfiguraci systému TwinCAT spárovat s hodnotami příslušných terminálů. Jak bude vysvětleno později, aplikace „Railway Simulator" získává hodnoty těchto proměnných pro matematický model. Pro naprogramování byl využit strukturovaný jazyk.

Zde je seznam vstupně výstupních proměnných:

Výhybky:

```
DI_V1_A  AT %I* : BOOL;        DI_V1_B  AT %I* : BOOL;
DI_V2_A  AT %I* : BOOL;        DI_V2_B  AT %I* : BOOL;
DI_V3_A  AT %I* : BOOL;        DI_V3_B  AT %I* : BOOL;
DI_V4_A  AT %I* : BOOL;        DI_V4_B  AT %I* : BOOL;
DI_V5_A  AT %I* : BOOL;        DI_V5_B  AT %I* : BOOL;
DI_V6_A  AT %I* : BOOL;        DI_V6_B  AT %I* : BOOL;
```

Semafory:

```
DI_SEM1   AT %I* : BOOL;       DI_SEM2   AT %I* : BOOL;
DI_SEM3   AT %I* : BOOL;       DI_SEM4   AT %I* : BOOL;
DI_SEM5   AT %I* : BOOL;       DI_SEM6   AT %I* : BOOL;
DI_SEM7   AT %I* : BOOL;       DI_SEM8   AT %I* : BOOL;
DI_SEM9   AT %I* : BOOL;       DI_SEM10 AT %I* : BOOL;
DI_SEM11 AT %I* : BOOL;
```

Senzory:

```
DO_SENSOR1   AT %Q* : BOOL := FALSE;
DO_SENSOR2   AT %Q* : BOOL := FALSE;
DO_SENSOR3   AT %Q* : BOOL := FALSE;
DO_SENSOR4   AT %Q* : BOOL := FALSE;
DO_SENSOR5   AT %Q* : BOOL := FALSE;
DO_SENSOR6   AT %Q* : BOOL := FALSE;
DO_SENSOR7   AT %Q* : BOOL := FALSE;
```

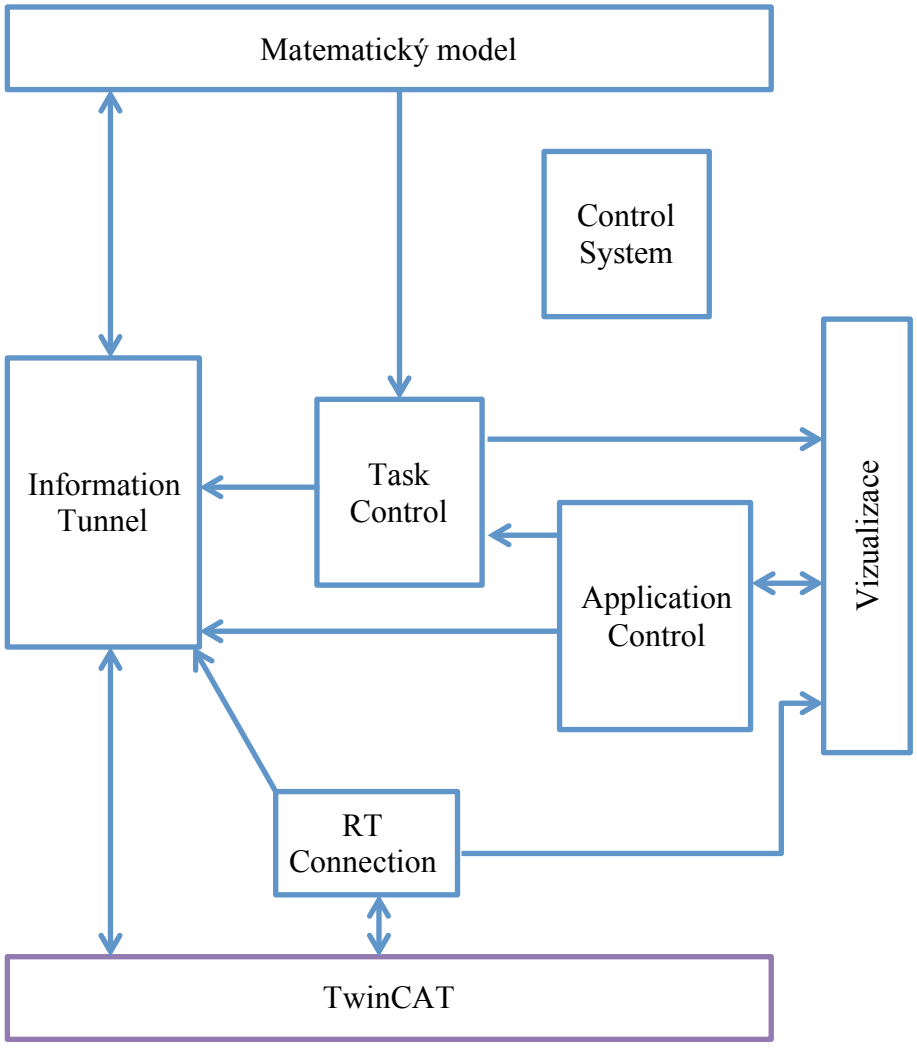
```
DO_SENSOR8   AT %Q* : BOOL := FALSE;
DO_SENSOR9   AT %Q* : BOOL := FALSE;
DO_SENSOR10 AT %Q* : BOOL := FALSE;
DO_SENSOR11 AT %Q* : BOOL := FALSE;
DO_SENSOR12 AT %Q* : BOOL := FALSE;
```

Napájecí segmenty:

```
AI_P_SEGMENT1 AT %I* : INT;      AI_P_SEGMENT2 AT %I* : INT;
AI_P_SEGMENT3 AT %I* : INT;      AI_P_SEGMENT4 AT %I* : INT;
AI_P_SEGMENT5 AT %I* : INT;      AI_P_SEGMENT6 AT %I* : INT;
AI_P_SEGMENT7 AT %I* : INT;
```

4.3 Kontrolní systém aplikace

Kontrolní systém aplikace je část programu „Railway Simulator", která je zodpovědná za řízení celého systému. Vnitřní uspořádání je zobrazeno na obrázku 14.



Obr. 14 – Zobrazení tříd kontrolního systému s meziobjektovou komunikací [0]

Nyní vysvětlím fungování všech objektů, tvořících kontrolní systém. Vše začíná u objektu třídy „ControlSystem”. Tento objekt celý kontrolní systém zapouzdřuje. Provádí alokaci všech objektů a při ukončování programu také dealokaci. Dále navazuje komunikaci s matematickým modelem a vizualizací.

„RTConnection” je třída, jejíž objekt navazuje komunikaci se systémem TwinCAT. To probíhá následujícím způsobem. TwinCAT má program „ADS router”. Tento program funguje podobně jako router na síti – umožňuje přístup k různým částem systému, jako příkladně k virtuálnímu PLC. K tomu je využit adresní systém, podobný IP adresám. „RTConnection” se k tomuto routeru připojí a získá přístup. Ten pak poskytuje ostatním objektům, které chtějí s virtuálním PLC komunikovat. Jmenovitě jde o objekt třídy „InformationTunnel” a o vizualizaci.

„InformationTunnel” zajišťuje předávání informací mezi systémem TwinCAT a matematickým modelem. Objekt této třídy má vytvářet mezi těmito částmi (systémem TwinCAT a matematickým modelem) takzvaný tunel, přes který budou protékat informace. Tato část programu je vytvořena tímto způsobem, aby informace procházející tunelem mohly být upravovány.

Při alokaci využije „informační tunel” již vytvořený objekt třídy „RTConnection” pro spojení s virtuálním PLC a prováže se s vstupně–výstupními proměnnými. Od této chvíle bude reagovat na každou změnu vstupů. Reakce na změnu bylo dosaženo využitím „událostmi řízeného programování”. Dále se „informační tunel” propojí při alokaci s matematickým modelem.

Výstupy z matematického modelu jsou virtuálnímu PLC předávány beze změny (konkrétně jde o hodnoty senzorů projetí vlaku). Vstupní hodnoty se ale ještě předtím, než jsou předány matematickému modelu, mohou upravit nebo zablokovat. Je to díky tomu, že „informační tunel” implementuje varianty ovládání (popsané v kapitole 2.3). Vše, co podle aktivní varianty nemá být ovládáno, je zablokováno. U minimální a střední varianty ovládání jsou ignorovány všechny analogové vstupy, až na první. Jeho hodnota je pak předána všem napájecím segmentům.

„TaskControl” má za cíl kontrolovat právě nastavenou úlohu, kterou má student na simulovaném modelu splnit (úlohy byly popsány v kapitole 3 – Sada zadání laboratorních úloh). Pro tento účel přebírá z matematického modelu informace

o chybách řízení modelu a o průjezdech vlaku senzory. Dále říká „informačnímu tunelu", jakou variantu ovládání má použít. Pokud bude nastavená úloha splněna nebo se objeví chyba řízení modelu, „TaskControl" o tom předá informaci vizualizaci, která úspěšnost úlohy zobrazí.

Poslední částí je třída „ApplicationControl". Její funkcí jsou věci, jako spuštění a zastavení simulace nebo změna úlohy uživatele. Pro tento účel komunikuje s vizualizací, z které povel pro danou akci získává.

4.4 Vizualizace

Vizualizace má jednu svoji část v programu „Railway Simulator" a druhou ve virtuálním PLC. Níže popíši obě části, včetně ovládání simulačního systému přes vizualizaci uživatelem.

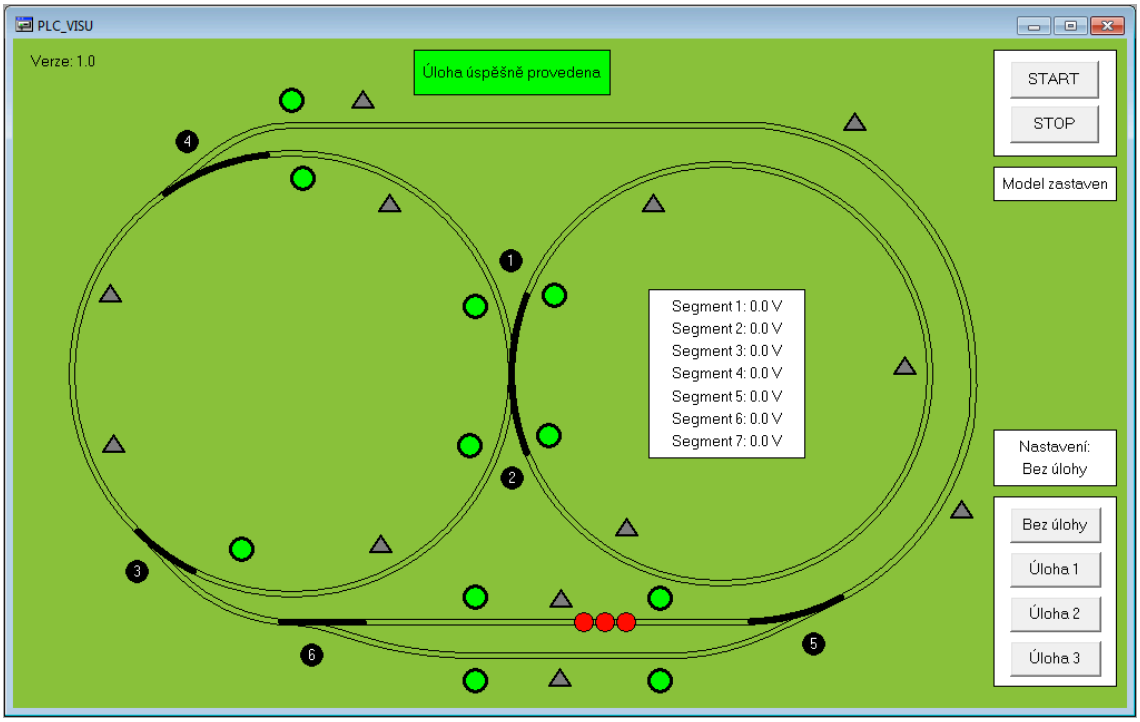
4.4.1 Vizualizace v simulačním programu

První část vizualizace sídlící v simulačním programu je třída (napsaná v C++). Během celé doby běhu programu získává aktuální stav matematického modelu a předává ho druhé části vizualizace ve virtuálním PLC. Dále z virtuálního PLC získává požadavky uživatele, zadané přes uživatelské rozhraní. Tyto informace vizualizace předá příslušným objektům kontrolního systému aplikace.

Komunikace s virtuálním PLC probíhá stejným způsobem, jakou používá „informační tunel" kontrolního systému. Vizualizace se připojí přes router systému TwinCAT a sváže svoje fungování s proměnnými ve virtuálním PLC, určenými pro vizualizaci.

4.4.2 Vizualizace ve virtuálním PLC

Druhá část vizualizace sídlí ve virtuálním PLC. Tato část je naprogramovaná ve strukturovaném textu. Hlavním úkolem tohoto kódu je správně zobrazovat virtuální model a jeho fungování. Uživatelská podoba vizualizace je zobrazena na obrázku 15.



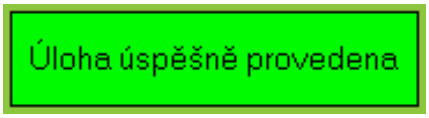
Obr. 15 – Vzhled vizualizace simulátoru [0]

Jak je vidět na obrázku, vizualizace graficky zobrazuje simulovaný model tak, aby se jeho vzhled podobal MŽU. Největší část zabírá zobrazení kolejnic. Po těchto kolejnicích se projíždí vlak, zobrazený třemi červenými kruhy.

Výhybky jsou označeny černými kruhy s bílým číslem. Černou barvou je zobrazeno to, jak je výhybka aktuálně nastavená. Pokud vlak najede na špatně přehozenou výhybku, tak se ve vizualizaci zastaví (stejně jako v matematickém modelu) a zobrazí se zpráva pro chybu řízení modelu. Tato zpráva se zobrazí v horní části vizualizace kdykoli, když nastane chyba řízení. Všechny tyto chyby byly popsány v kapitole 3.

Obr. 16 – Zpráva oznamující řídící chybu [0]

Pokud dojde k úspěšnému zvládnutí úlohy, zobrazí se taktéž v horní části vizualizace zpráva (obrázek 17).

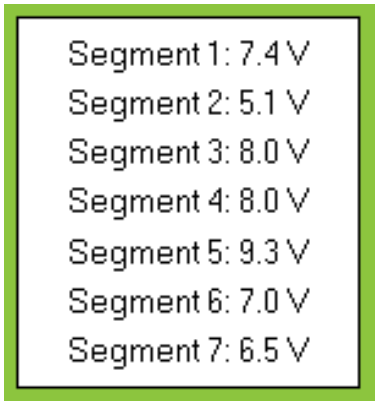


Obr. 17 – Zpráva oznamující úspěšné provedení úlohy [0]

Senzory průjezdu vlaku jsou ve vizualizaci taktéž zobrazeny. Zobrazuje je trojúhelník vyplněný šedou barvou. V případě, že přes senzor vlak přejede, šedá barva trojúhelníku se na dvě sekundy změní na oranžovou. Dvě sekundy byly zvoleny z toho důvodu, že MŽU drží senzor aktivní po stejnou dobu.

Semafory jsou ve vizualizaci zobrazeny jako kruh, vyplněný zelenou nebo červenou barvou podle toho, jaká barva má na semaforu aktuálně být.

Pro zjednodušení práce uživatele se systémem je ve vizualizaci také zobrazeno napětí na jednotlivých napájecích segmentech. To může sloužit uživateli k ověření, zda-li jeho řídící program, který má simulátor ovládat, funguje správně. Detail části vizualizace zobrazující napětí na segmentech je na obrázku 18.



Obr. 18 – Zobrazení napětí na jednotlivých napájecích segmentech [0]

Celý program je možno ovládat několika tlačítky ve vizualizaci. Spuštění nebo zastavení simulace je možné pomocí tlačítek START a STOP. Vizualizace zobrazuje i to, jestli je simulace právě spuštěna (detail na obrázku 19).

Obr. 19 – Okénko aktuálního stavu simulačního modelu [0]

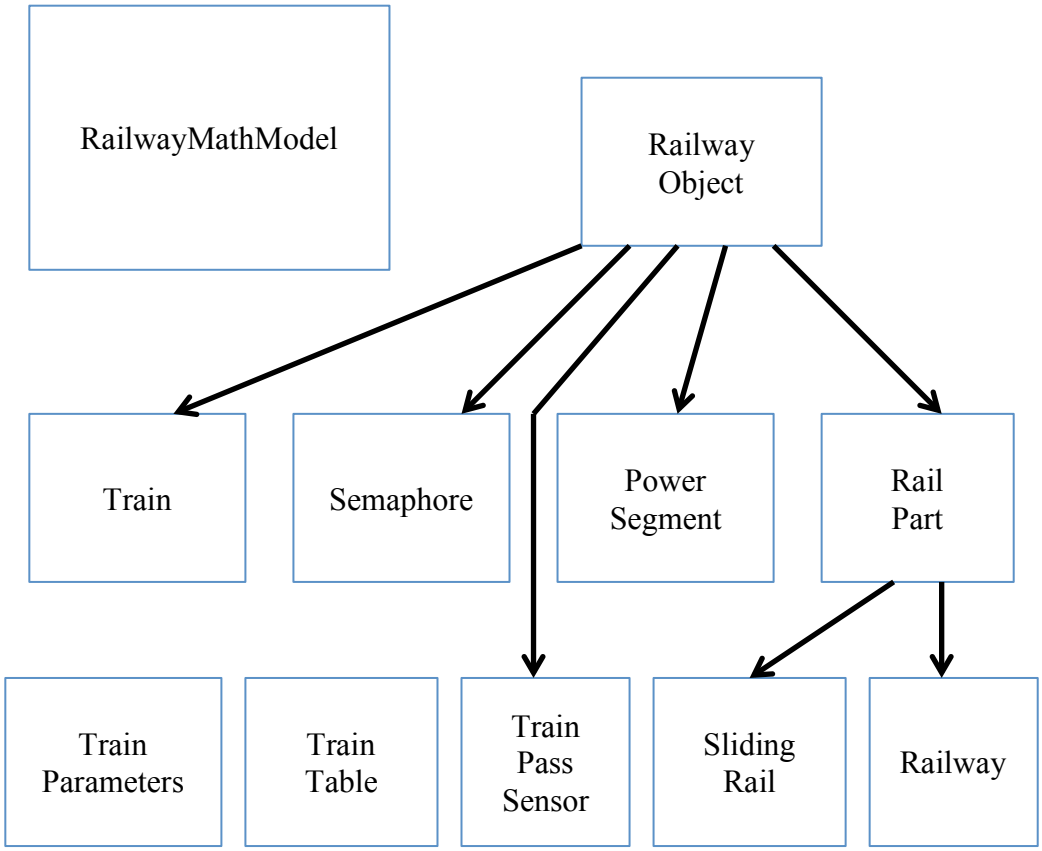
Poslední částí vizualizace jsou tlačítka pro nastavování úlohy (popsané v kapitole 3). Při spuštění simulátoru je simulace nastavena bez úlohy. Tu si může poté uživatel nastavit. Z funkčních důvodů lze měnit úlohu jenom v případě, že je simulování zastaveno. Detail pro ovládání úloh je na obrázku 20.

Obr. 20 – Ovládání úloh [0]

4.5 Matematický model

Matematický model je hlavní část programu „Railway Simulator". Jejím úkolem je simulovat MŽU. Dále musí získávat hodnoty vstupů z kontrolního systému aplikace a předávat mu informace o výstupech (což jsou senzory průjezdu vlaku). V neposlední řadě musí o svém aktuálním stavu poskytovat informace vizualizaci pro zobrazení.

Model tvoří 11 tříd. Tyto třídy simulují fungování různých částí MŽU (schéma je na obrázku 21). Za běhu je dohromady vytvořeno 52 objektů mat. modelu, které vzájemně spolupracují.



Obr. 21 – Schéma tříd matematického modelu s vyznačenou dědičností [0]

Celý mat. model zapouzdřuje třída „RailwayMathModel". Obsahuje ukazatele na všechny části mat. modelu a je zodpovědná za jejich vytvoření i odstranění. Tato hlavní třída je zároveň i rozhraním komunikace mat. modelu s ostatními částmi programu. Umožňuje také model spustit a zastavit metodami „start" a „stop".

Stejně jako MŽU musí být mat. model dynamický. Toho bylo dosáhnuto následovně. Metoda „start" vytvoří nové vlákno programu, které cyklicky volá metodu „update" hlavní třídy. Tato metoda volá metody „update" všech objektů, které potřebují aktualizovat stav. Každý objekt si tímto způsobem provede všechny potřebné změny a akce. Pro dosažení velké přesnosti simulace dochází k aktualizaci každou milisekundu. Fungování mat. modelu ve vlastním vlákně mu dává velkou výhodu v nezávislosti běhu na ostatních částech programu. Za zmínku stojí fakt, že pro práci s vlákny byla využita knihovna „thread" z nového standardu programovacího jazyka C++ z roku 2011.

Většina tříd jsou potomky třídy „RailwayObject". Po spuštění programu je v mat. modelu vytvořeno mnoho objektů některých tříd. Vzhledem k tomu, že tyto objekty spolupracují, vznikla potřeba, aby se mohli vzájemně identifikovat. Proto byla

tato funkčnost zabudována do „RailwayObject". Pro přehlednost programu jsou všechny třídy, které simulují reálnou část MŽU, potomky této třídy.

„PowerSegment" je třída realizující virtuální napájecí segment. Hlavní funkcí je uchovávání hodnoty napětí na daném segmentu. Hodnota je uchovávána a distribuována ostatním objektům.

„RailPart" je virtuální třída. Jde o společný základ pro jakoukoli železniční část. Obsahuje ukazatel na napájecí segment („PowerSegment"), do kterého patří. Dalším atributem je délka železniční části. „RailPart" obsahuje metodu „getConnection", která vrací napojení na ostatní části železnice, na které je v rámci simulování napojena. Metoda je definována v potomcích, neboť zmíněné fungování se může lišit. Této funkčnosti využívá vlak (který je popsaný níže). Potomky třídy „RailPart" jsou třídy „Railway" a „SlidingRail".

Deklarace třídy RailPart:

```
class RailPart : public RailwayObject {
    PowerSegment *powerSegment;
    float lengthInM;

    RailPart();
    void operator=(int);
public:
    RailPart(float lengthInM,
             PowerSegment *powerSegment,
             RailwayMathModel *model,
             char ID = -1);

    float               getLength() { return lengthInM; }
    virtual void        setConnection(RailPart *connection,
                                      char index) = 0;
    virtual RailPart*   getConnection(char index) = 0;
    PowerSegment*       getSegment() { return powerSegment; }
};
```

„Railway" realizuje železnici a „SlidingRail" železniční výhybku. „Railway" nemá žádnou funkčnost nad úroveň „RailPart". Třída železniční výhybky má navíc pár zajímavých funkčností. Její metoda „getConnection" vrací napojení podle aktuálního přehození. Ve chvíli, kdy je přehazována, si výhybka zkontroluje, jestli na ní není vlak. Pokud ano, tak vyvolá chybu řízení.

Třída „Semaphore" má funkci semaforu. Uchovává si informaci o rozsvícené barvě. Ve chvíli, kdy kolem projíždí vlak a na semaforu je rozsvícené červené světlo, vyvolá „Semaphore" chybu řízení. To platí ale jenom ve chvíli, kdy jede vlak ze směru, z kterého jsou světla semaforu vidět. Nastavení semaforů bylo převzato z MŽU, kde jsou světla viditelná jen z jednoho směru.

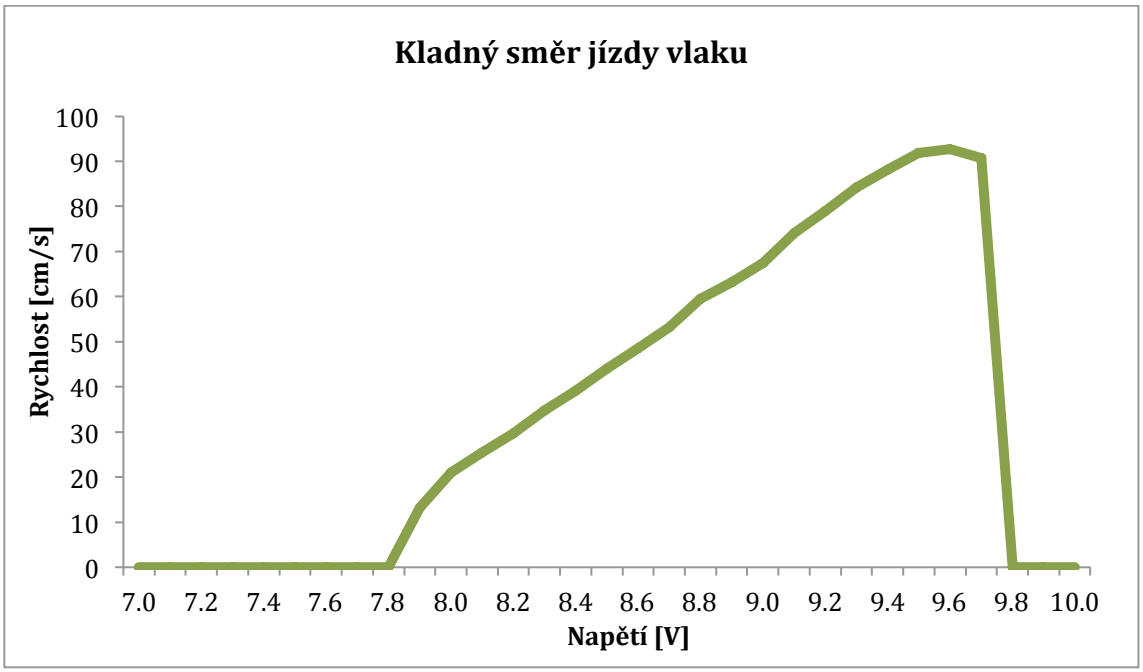
„TrainPassSensor" je, jak anglický název napovídá, třída senzoru průjezdu vlaku. Objekt této třídy je stejně jako semafor umístěn na určité pozici železniční části (třída „RailPart"). Když vlak tuto pozici přejede, objekt odešle informaci o aktivaci (která se projeví na výstupech) a čeká dvě sekundy. Po uplynutí této doby odešle informaci o deaktivaci.

„Train" je třída realizující vlak. Tento programový vlak si je schopen zjistit, na které železniční části je a jaké napětí má napájecí segment této části. Podle napětí, zrychlení a aktuální rychlosti si poté přepočítá novou rychlost. Hodnotu rychlosti a pozice nakonec využije pro výpočet nové aktuální pozice. Vzhledem k tomu, že tento přepočet probíhá v rámci metody „update" každou milisekundu, tak jde o numerické řešení diferenciálního výpočtu. Programový vlak si při nájezdu na výhybku kontroluje, zda-li je správně přehozena. V opačném případě je vyvolána chyba řízení.

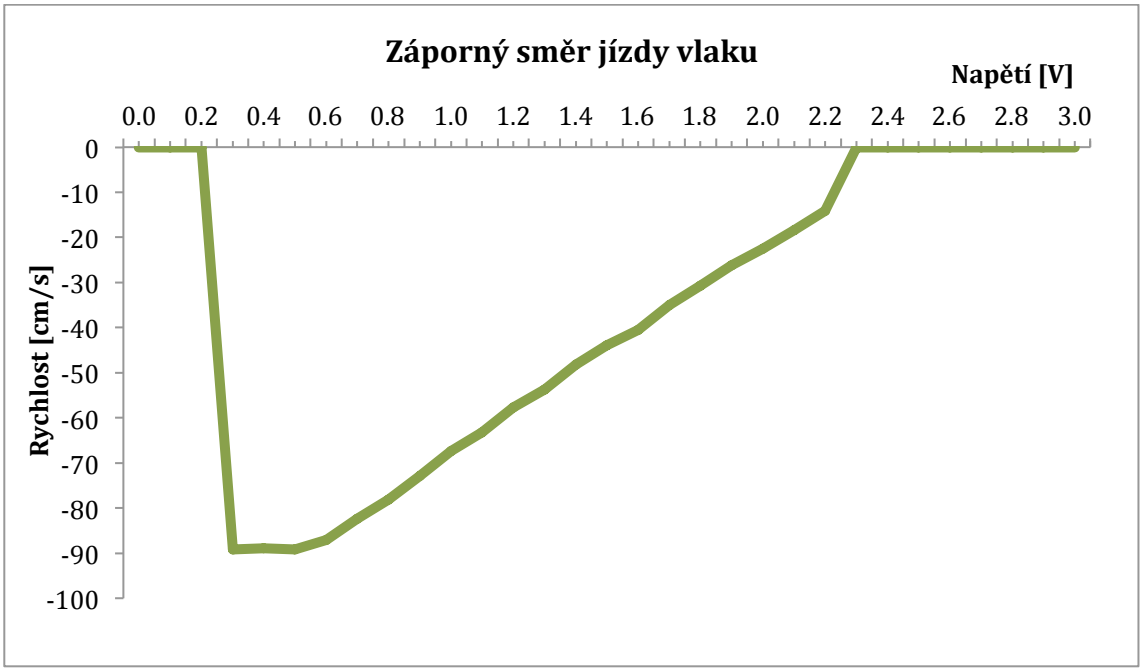
Třída „TrainTable" nemá žádnou obdobu v MŽU. Její hlavní funkce je uchovávat odkazy na všechny vlaky. Třída vznikla jako řešení programového paradoxu. Mnoho objektů vyžaduje pro svoji práci přístup k vlakům, ale vlaky je potřeba vytvořit až ve chvíli, kdy už je zbytek mat. modelu vytvořen. Příkladně při alokaci objektu třídy realizující senzor. Tento objekt potřebuje od začátku znát všechny vlaky. Ty ale ještě neexistují. A proto nebude senzor pracovat přímo s vlaky, ale s „tabulkou vlaků" (jak lze „TrainTable" přeložit), do které se vlaky doplní později.

„Vlaková tabulka” si umí všechny své vlaky kontrolovat na jejich vzájemnou srážku. Pokud by k tomu došlo, tak se vyvolá chyba řízení. Neboť ve výsledné verzi simulátoru je jen jeden vlak, toto fungování není využito.

Poslední třídou matematického modelu je třída „TrainParameters” („parametry vlaku”). Tento programový útvar slouží výhradně objektům třídy „Train”. Hlavní funkcí je přepočítávat dva parametry pro vlak a těmi jsou rychlost a zrychlení. Tyto parametry se mění v závislosti na napětí, které je k vlaku přivedeno. Vztah rychlost–napětí bylo potřeba na MŽU změřit pro různá napětí (znázorněno v grafech).



Graf 1 – Závislost rychlosti vlaku MŽU na napětí v kladném směru [0]



Graf 2 – Závislost rychlosti vlaku MŽU na napětí v záporném směru [0]

4.6 Příklad práce uživatele se systémem

Fungující simulátor v praxi vypadá příkladně takto. Na virtuálním počítači je nainstalován systém TwinCAT a program „Railway Simulator". K tomuto počítači jsou připojeny vstupně–výstupní terminály. Na terminály je napojeno PLC pro řízení simulátoru. Virtuální počítač je propojen se vzdálenou laboratoří LabLink.

Práce uživatele se simulátorem je následující. Uživatel (příkladně student) si spustí libovolný počítač s přístupem na internet. Připojí se přes LabLink k virtuálnímu počítači. Zde spustí program „Railway Simulator", který funguje jako konzolová aplikace a otevře vizualizaci v programu „TwinCAT PLC Control" nebo v programu „TwinCAT PLC HMI". Do PLC napojeného na terminály vzdáleně nahraje řídící program, který chce otestovat a spustí simulaci. Po ukončení testování se uživatel odpojí od virtuálního počítače.

5 Závěr

Hlavní cíl této bakalářské práce bylo vytvořit simulační program napodobující fungování MŽU. Tento cíl byl splněn. Kombinace všech částí simulačního systému (TwinCAT, „Railway Simulator", popřípadě LabLink) umožňuje získávat hodnoty vstupů na terminálech. Ty pak předá simulaci, která se zobrazuje ve vizualizaci. Přes vizualizaci může uživatel systém ovládat.

Tato práce by se dala rozšířit několika způsoby. Bylo by možné simulovat fungování více vlaků zároveň. Systém by mohl mít více vizualizací, mezi kterými by uživatel mohl přepínat (příkladně vizualizace podobná systémům zobrazení vlakového provozu v řídících centrech). Vizualizace by mohla být vytvořena jako samostatná aplikace pro operační systém Windows.

Bakalářská práce měla pro mě značný přínos. Umožnila mi si kompletně projít vývojem softwaru od stanovení požadavků, přes návrh architektury, rozdělení úkolů jednotlivým třídám, naprogramování, až po testování a odhalování chyb.

Velký přínos bude mít tato práce i pro budoucí studenty VŠPJ, kteří si budou moci otestovat své praktické znalosti, aniž by museli zůstávat ve škole, což jim umožní mít více času na studium problematiky ohledně PLC.

Seznam použitých zdrojů

[0] Zdroj vlastní

[1] Vzdálená internetová laboratoř. *Wikipedie* [online]. 2012 [cit. 2012-05-25]. Dostupné z: http://cs.wikipedia.org/wiki/Vzdálená_internetová_laboratoř

[2] About. *LabLink* [online]. [cit. 2012-05-25]. Dostupné z: http://www.lablink.cz/about

[3] *Model železničního uzlu*: *Uživatelská dokumentace*. Jihlava, 2011.

[4] RS-232. *Wikipedie* [online]. 2012 [cit. 2012-05-24]. Dostupné z: http://cs.wikipedia.org/wiki/RS-232

[5] EtherCAT Coupler. *Beckhoff* [online]. 2012 [cit. 2012-05-26]. Dostupné z: http://www.beckhoff.com/english.asp?ethercat/buskopl.htm?id=1633319839

[6] EK1100. *Beckhoff* [online]. 2012 [cit. 2012-05-26]. Dostupné z: http://www.beckhoff.com/english.asp?ethercat/ek1100.htm?id=1983920606

[7] AXSP3P06. *Axima* [online]. 2011 [cit. 2012-05-26]. Dostupné z: http://www.axima.cz/katalog/produkt/axsp3p06--230vac-24vdc-6a/AXSP3P06/911035dc.html

[8] EL1008. *Beckhoff* [online]. 2012 [cit. 2012-05-26]. Dostupné z: http://www.beckhoff.com/english.asp?ethercat/el1008.htm?id=1989211831288

Seznam obrázků

Seznam grafů

Seznam zkratek a pojmů

- VŠPJ – Vysoká škola polytechnická Jihlava
- PLC – Programovatelný logický automat
- MŽU – Model železničního uzlu
- Railway Simulator – název programu zajišťující simulaci MŽU
- LabLink – systém vyrobený na ČVUT v Praze pro vzdálený přístup
- TwinCAT – systém reálného času od firmy Beckhoff

Přílohy

- Program „Railway Simulator”
- Knihovny pro program „Railway Simulator” (od firmy Microsoft: www.microsoft.com)
- Popis konfigurace terminálů
- Model železničního uzlu – Uživatelská dokumentace
- Ukázkový program pro řízení simulátoru
- Postup instalace simulátoru
- Instalace systému TwinCAT
- Konfigurace systému TwinCAT
- Program pro virtuální PLC