TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2612 – Elektrotechnika a informatika

Studijní obor: Informatika a logistika

Aplikování vývojového prostředí LabVIEW při návrhu řízení malého robota LEGO MINDSTORMS NXT

Application of LabVIEW development system in design of control of small robot LEGO MINDSTORMS NXT

Bakalářská práce

Autor: **Petr Škoda**

Vedoucí práce: Ing. Jaroslav Vlach

Konzultant: Josef Havlíček

V Liberci 16.5.2011

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL 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 toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum

Podpis

Abstrakt

Tato bakalářská práce se zaměřuje na možnost programování a následného ovládání malého robota LEGO MINDSTORMS NXT za pomoci vývojového prostředí LabVIEW. V teoretické části se zabývá způsobem vytváření programů v prostředí LabVIEW. Dále se pak zabývá robotem LEGO MINDSTORMS NXT, jeho komponenty a způsobem komunikace s osobním počítačem. Na konci teoretické části jsou uvedeny jiné možnosti programování robota LEGO MINDSTORMS NXT.

Praktická část se zaměřuje na konkrétní aplikace, naprogramované v prostředí LabVIEW. Jsou zde odzkoušeny některé senzory, popsáno jejich fungování a případně zde jsou uvedeny jejich nedostatky. Jednotlivé programy jsou vytvářeny jako stavové automaty, což zpřehledňuje výsledný program a umožňuje snadnou změnu funkce programu.

Klíčová slova: LabVIEW, LEGO MINDSTORMS NXT, stavový automat

Abstrakt

The bachelor thesis is focused on the possibility of programming and controlling of small LEGO MINDSTORMS NXT robot through development system LabVIEW. The theoretical part is concerned with mode of creating programs in LabVIEW. It also deals with LEGO MINDSTORMS NXT robot, his components and method of communication with personal computer. In the end of theoretical part are noted other alternative programs for programming the LEGO MINDSTORMS NXT robot.

The practical part is focused on the concrete aplications, programmed in LabVIEW. There are examined some of the sensors, described their functions and eventually there are noted their embarrassments. The programs are created as state machine. It makes final program more synoptical and allows easy change of the program.

Keywords: LabVIEW, LEGO MINDSTORMS NXT, state machine

Obsah

Seznam pojmů a zkratek

ASCII (American Standard Code for Information Intechange) – základní znaková abeceda

Bluetooth – bezdrátová komunikace pro propojení dvou a více přístrojů. Pracuje v ISM pásmu, tedy ve volném frekvenčním pásmu.

Flash EEPROM – mžikové, elektricky mazatelné paměti (Flash Electric erasable Programmable Read Only Memory) – tuto paměť lze vícekrát přeprogramovat. Počet přeprogramování je až několik statisíc.

LabVIEW - Laboratory Virtual Instrument Engineering Workbench

PC – osobní počítač (Personal Computer) .

Robotika – obor zabývající se roboty, jejich návrhem, výrobou a účelem

ROM – paměť pouze pro jeden zápis a následné čtení (Read Only Memory)

NI – National Instruments – firma, jejímž produktem je mimo jiné i vývojové prostředí LabVIEW

USB – univerzální sériová sběrnice (Universal Serial Bus)

Úvod

V dřívějších dobách, kdy poptávka po zboží nebyla příliš vysoká, si lidé mohli dovolit vykonávat komplexnější práci, jeden produkt prošel od začátku až do konce výroby rukama jednoho či několika málo dělníků. Se vzrůstající poptávkou se však tento způsob výroby ukázal jako neefektivní a práce se začala rozdělovat na více lidí, kde každý měl svou danou specializaci. Postupem času se došlo do stádia, kdy velká část zaměstnanců vykonává velmi jednoduchou práci. V tu chvíli nastal čas přemýšlet nad tím, zda-li by nebylo výhodnější nahradit lidi stroji, které by stejně jednoduchou práci dokázali vykonat za kratší dobu a byly by finančně méně náročné.Mohl tak začít vývoj a rozvoj robotiky, tedy vědy zabývající se roboty, jejich funkcí a návrhem.

V dnešní době se setkáváme s roboty téměř na každém kroku, ať už jde o jednoduché kuchyňské roboty či daleko složitější roboty na výrobních linkách. Je tedy patrné, že robotika se stává velmi zajímavým odvětvím s širokou škálou využitelnosti.

Dánská firma LEGO, zabývající se výrobou hraček zareagovala na rozvoj elektroniky a robotiky a vytvořila pro děti a mladistvé řadu programovatelných stavebnic LEGO MINDSTORMS Úlohy pro funkci robota lze vytvářet jednak v programovém prostředí dodávaném s robotem, případně v prostředí LabVIEW od společnosti National Instruments. Tato bakalářská práce se zabývá programováním robota LEGO MINDSTORMS NXT v prostředí LabVIEW.

Teoretická část je rozdělena na dvě části. V první části se zabýváme popisem a vlastnostmi programového prostředí LabVIEW, ve kterém se vytváří praktické úlohy. Cílem je uvedení do problematiky grafického programování. Druhá část se pak zabývá stavebnicí LEGO MINDSTORMS NXT, popisem jejích technických prvků a programového vybavení. Praktická část se pak zabývá programy vytvořenými v programu LabVIEW za pomoci rozšíření LabVIEW Module pro LEGO MINDSTORMS NXT. Je zde uveden popis jejich funkcí a případných nedostatků. Ukázky programu jsou pak uvedeny v příloze.

Cíl práce

Cílem práce je ukázka možnosti komunikace robota LEGO MINDSTORMS NXT s osobním počítačem za pomoci vývojového prostředí LabVIEW. Z toho plyne seznámení se jak s vlastnostmi NXT robota, tak se základními vlastnostmi a způsobem grafického programování v prostředí LabVIEW.

Po seznámení se s NXT a LabVIEW je cílem sestavit robota a vytvořit program tak, aby bylo možné zjistit výhody, popřípadě nevýhody řízení robota za pomoci PC, vyzkoušet vzdálenou komunikaci, její funkčnost a zpoždění..

1. LabVIEW

1.1 Úvod

LabVIEW (**Lab**oratory **V**irtual **I**nstrument **E**ngineering **W**orkbench) je prostředí pro grafické programování od firmy NATIONAL INSTRUMENTS. Právě grafický způsob sestavování programů je to, čím se nejvíce liší od textových programovacích jazyků, jako jsou například C, C#, JAVA. Jednotlivé funkce nejsou reprezentovány textem, ale grafickou ikonou, která má různý počet vstupů a výstupů v závislosti na tom, co vykonává. Jednoznačnou výhodou tohoto stylu programování je jeho rychlost a přehlednost. Není potřeba se starat o syntaxi, stačí pouze propojovat jednotlivé ikony. Jednoduché úlohy s ním tedy zvládne vytvořit každý.
Programovací prostředí LabVIEW je primárně určeno pro řešení problémů v oblasti měření, analýzy a prezentace dat. Pro tyto činnosti je v LabVIEW naprogramováno mnoho funkcí podprogramů, které podstatně snižují čas potřebný k vyřešení a dokončení práce.

1.2 Historie

Již od roku 1983 se pracovníci firmy National Instruments (NI) snažili hledat způsob, jak urychlit a zefektivnit vytváření měřicích nástrojů. Jejich myšlenkou je grafické vytváření virtuálních nástrojů.
V roce 1986 byla uvedena první verze LabVIEW 1, která pracovala pod operačním systémem MacOS. Důvod byl jednoduchý: v té době nebylo mnoho systémů, které by mohly podpořit grafické programování a MS DOS to ještě neumožňoval.
V roce 1990 byla uvedena na trh verze LabVIEW 2. Rychlost překladu programu byla srovnatelná s rychlostí překladu v jazyce C. V roce 1992 bylo LabVIEW 2 zpřístupněno také pro operační systémy Windows a SunOS (UNIX).
V roce 1993 přichází nová verze LabVIEW 3, která podporuje všechny tři uvedené systémy a současně umožňuje přenos programů vytvořených v LabVIEW z jedné platformy na jinou.
V roce 1996 byla vypuštěna další verze LabVIEW 4, která zpříjemnila uživatelské prostředí a přinesla několik nových vylepšení. Hned o dva roky později přichází nová verze LabVIEW 5, která se zaměřila na využití internetu.
V roce 2000 vzniká verze LabVIEW 6, která mimo toho, že je zpřístupněna též pro platformu LINUX, podporuje nově objektově orientované programování a vícevláknové programování.

Od té doby přichází NI každý rok s novou, něčím vylepšenou verzí. Nejnovější verze je LabVIEW 10.0, vydaná v roce 2010.

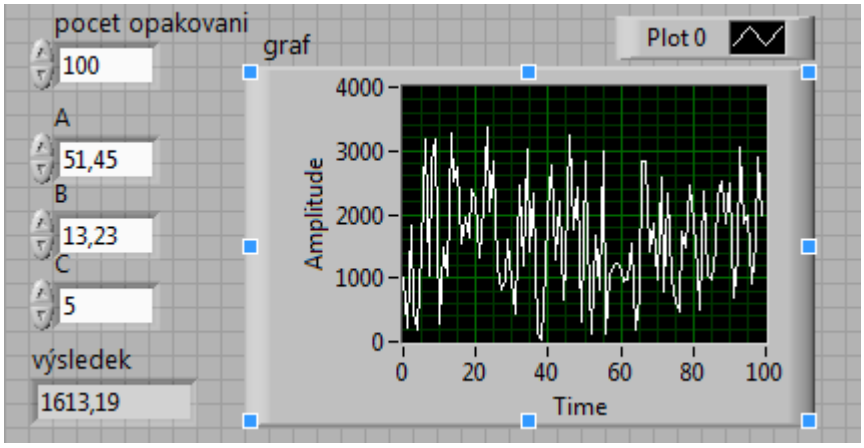
1.3 Virtuální instrumentace (VI)

Virtuální instrumentace je využití přizpůsobitelného programového vybavení a technického vybavení k vytvoření uživatelem požadovaného systému. Tradiční technické přístroje jsou vyrobeny za určitým, předem definovaným účelem, který nelze v případě potřeby změnit. Proto vznikla virtuální instrumentace (tedy „zdánlivá“), u níž je velká část funkce technického vybavení nahrazena programovým vybavením. To umožňuje rychlé vytváření funkcí systému podle potřeby dané aplikace. Prostředky programového prostředí LabVIEW umožňují vytvářet takovéto virtuální instrumenty.

1.4 Čelní panel a Blokový diagram

Při vytváření programů (tedy virtuálních instrumentů - VI) v LabVIEW pracujeme se dvěma základními programovacími prostředími: čelní panel a blokový diagram.

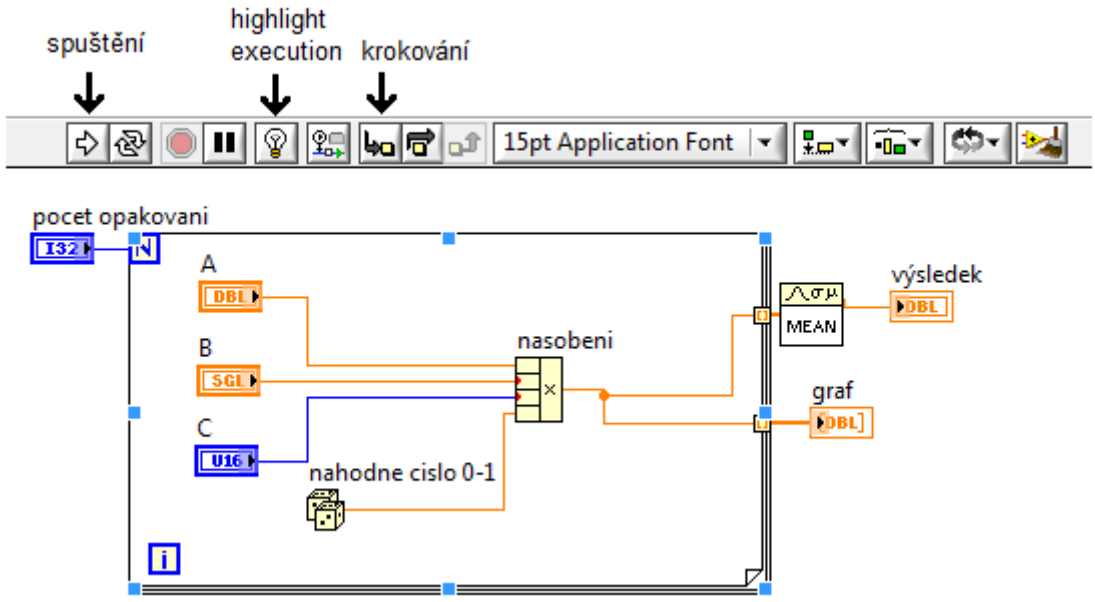
Čelní panel (Front panel) slouží pro vytváření grafického prostředí, které bude reprezentovat samotný nástroj. Jde tedy o rozhraní mezi uživatelem a aplikací. Jsou zde umístěny ovládací prvky, prvky s grafickým výstupem, kontrolky a mnoho dalších nástrojů. Po vytvoření aplikace (*.exe) je čelní panel zprostředkovatelem přenosu informací mezi uživatelem a aplikací.



Obr. 1.1: **Ukázka čelního panelu programu (VI) se čtyřmi numerickými vstupy, jedním numerickým výstupem a grafickým výstupem**

Na Obr 1.1 je ukázka čelního panelu s programem, který zvoleným počtem opakování násobí tři zadaná čísla náhodným číslem. Všechny vypočítané hodnoty se pak zapíší do grafu a vypočítá se z nich aritmetický průměr. Jednotlivé komponenty pro vložení na čelní panel (ovládací a zobrazovací prvky) získáme v paletě Controls (viz Obr 1.3 vlevo), kterou vyvoláme kliknutím pravého tlačítka myši na čelním panelu.

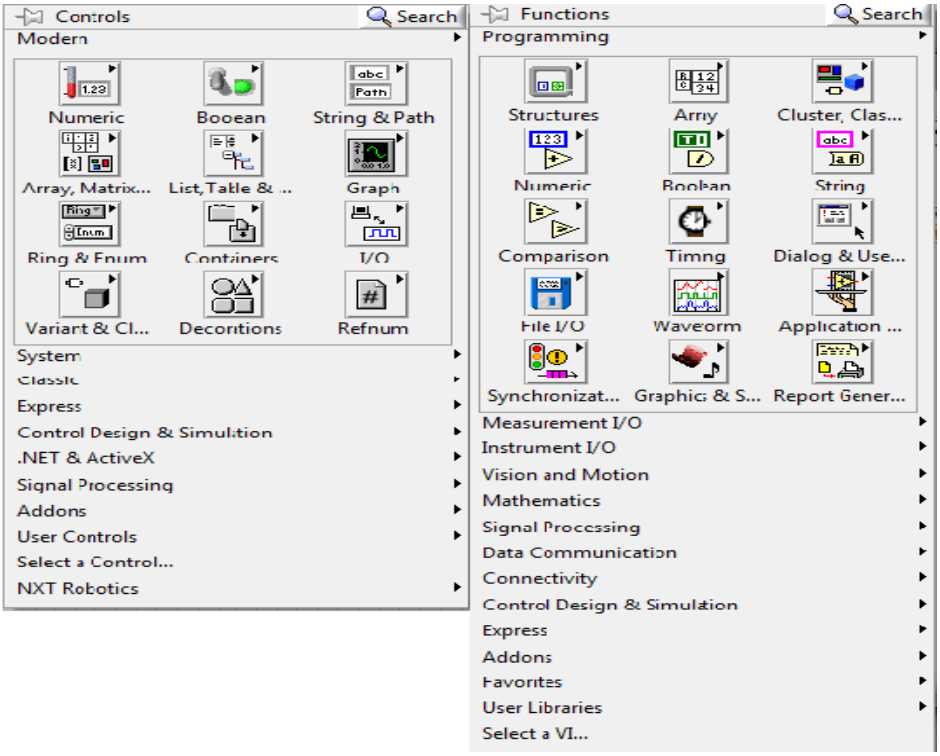
Blokový diagram (Block diagram) reprezentuje vlastní zápis programu v grafickém kódu. V okně blokového diagramu lze na horní liště plynule spustit program nebo jej spouštět krok po kroku pro lepší odhalování chyb a snadnější představu o funkci programu. Je zde také tlačítko se symbolem žárovky (tzv. highlight execution), po jehož stisknutí se program zpomalí natolik, aby bylo možné sledovat tok dat a vstupní a výstupní hodnoty na jednotlivých prvcích blokového diagramu. Zároveň při spuštění programu blokový diagram zešedne a jeho jednotlivé části jsou zvýrazňovány ve chvíli, kdy se datový tok dostane až k nim.



Obr. 1.2: Zdrojový kód příkladu z Obr. 1.1

Na Obr. 1.2 je patrný způsob propojování jednotlivých bloků. Bloky A, B, C, výsledek a graf vznikly samy po umístění jim odpovídajících kontrolek na čelním panelu. Ostatní ikonky a smyčka For byly dodány z palety Functions (Obr. 1.3 vpravo), která se stejně jako v případě palety Controls vyvolává stiskem pravého tlačítka myši na okně blokového diagramu. Stejně

jako jiné programovací jazyky, i LabVIEW pracuje s různými datovými typy, jako jsou boolean, integer, double, string a mnoho jiných. Například pro formát čísel pracuje LabVIEW s celkem 15 různými typy. LabVIEW však není silně typovým jazykem, jako například jazyk C. Je tedy možné slučovat dohromady různé datové typy s tím, že si je LabVIEW samo převede na požadované. Toto je dobře patrné na obr 1.2, kdy do bloku násobení je přiveden typ DBL (datový typ s pohyblivou čárkou s dvojitou přesností), typ SGL (datový typ s pohyblivou čárkou s jednoduchou přesností) a typ U16 (celočíselný 16-bitový datový typ). V místě propojení bloku násobení s U16 a SGL vznikne červená značka, která signalizuje datovou konverzi. Pokud není možná přímá konverze (např. číslo na řetězec znaků), nelze tyto bloky propojit.

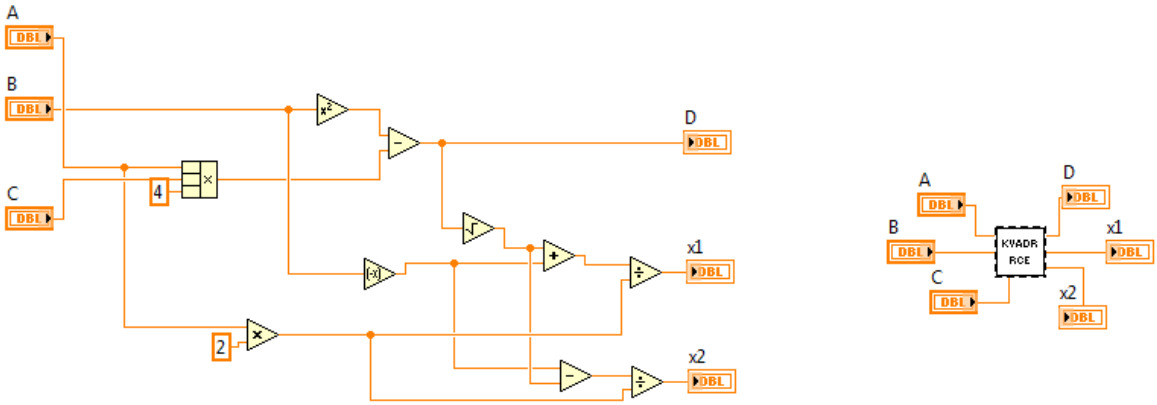


Obr. 1.3: Příklady palet: vlevo Controls, vpravo Functions

1.5 Vytváření SubVI

Při vytváření složitějších a rozsáhlejších projektů často dojde k tomu, že na obrazovce bude velké množství prvků, které budou propojené ještě větším množstvím spojů. Může tak snadno nastat situace, že program bude příliš velký, pak se nedá snadno kontrolovat a bude velmi nepřehledný. Proto je možné a velmi užitečné vytvářet si tzv. SubVI. SubVI je obdobou

vytváření podprogramů v klasických programovacích jazycích, respektive objektů v objektově orientovaných programovacích jazycích. Máme-li nějaký funkční celek, který je příliš rozsáhlý, můžeme jej zabalit do samostatného virtuálního nástroje a v hlavním programu pracovat pouze s jeho ikonkou.

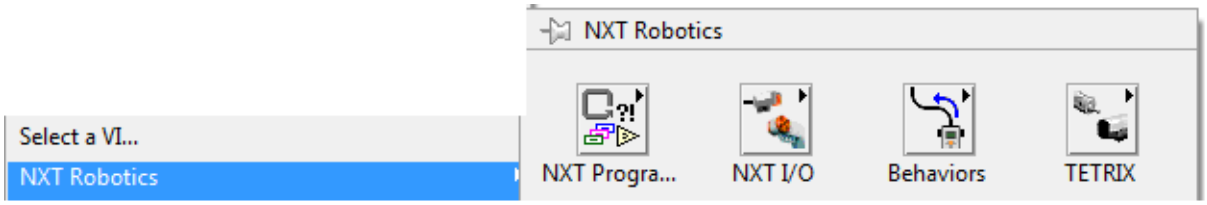


Obr. 1.4: Vytvoření SubVI

Na Obr. 1.4 je vidět, kolik místa se dá vytvořením SubVI ušetřit. Program v obou případech vypočítává kořeny kvadratické funkce, jednou je ale zabalen (vpravo) a jednou ne (vlevo).
SubVI se vytvoří tak, že označíme část programu, kterou chceme zabalit, v menu blokového diagramu zvolíme položku Edit a následně Create SubVI, poté si můžeme na čelním panelu vytvořit vlastní ikonku a rozmístit vstupy a výstupy nově vzniklého bloku.

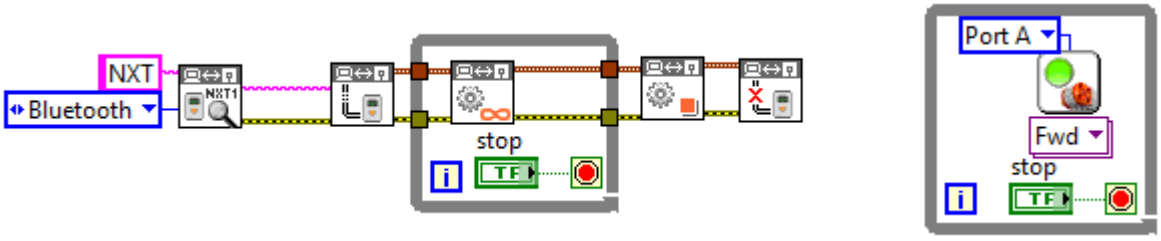
1.6 LabVIEW Module pro LEGO MINDSTORMS NXT

Pro komunikaci mezi robotem lego MINDSTORMS NXT a programem LabVIEW verze 9.0 a vyšší je určen přídavný NXT Module. Jeho předchůdcem byl NXT Toolkit (určený pro starší verze LabVIEW). NXT Module má většinu funkcí předešlého modulu, navíc má však spoustu nových funkcí, které jsou zaměřeny na lepší a pohodlnější komunikaci mezi robotem a počítačem. Pro jejich využití bylo třeba do řídicí jednotky robota (tzv. kostky) nahrát novou verzi vnitřního programového vybavení (firmware) místo verze 1.05 na verzi 1.26. Po nainstalování NXT module se v paletě Functions objeví nová záložka s řadou prvků pro ovládání robota, viz Obr. 1.5.



Obr. 1.5: Nová záložka po instalaci NXT Module

Nejvýraznější změna nastala u způsobu sestavování programu, kdy odpadá nutnost blokově vytvořit spojení mezi robotem a LabVIEW. Na Obr. 1.6 vlevo je vidět způsob vytváření programu ve starším modulu NXT Toolkit. Vpravo je potom ukázka, jakým způsobem se vytváří program v novém NXT Module. Oba tyto programy jsou určené pro vzdálené řízení (tzv. Remote control). Jde o řízení, kdy program je uložen v paměti počítače a ten komunikuje nepřetržitě s robotem. Druhým způsobem řízení je řízení pomocí inteligentní kostky (tzv. OnBoard control). V tomto případě je program nahrán do paměti kostky a tam se vykonává. V NXT Module je řízení OnBoard zrušeno, důvodem byla nejspíš malá paměť kostky.



Obr. 1.6: Rozdílné programování mezi NXT Toolkit(vlevo) a NXT Module (vpravo)

Na Obr. 1.6 jsou funkčně zcela stejné programy, které naváží komunikaci přes komunikační rozhraní bluetooth, roztočí servomotor na portu A na předdefinovanou rychlost, po stisknutí tlačítka „stop“ ukončí pohon servomotoru a pak i programu. Je zde velmi dobře vidět, že programování v NXT Module je podstatně rychlejší a při náročnějších aplikacích i daleko přehlednější. Malou nevýhodou, která by se dala najít je fakt, že při novém způsobu programování nelze ošetřovat možné vzniklé chyby.

1.7 Výhody a nevýhody

Výhody

Za hlavní výhody programu LabVIEW lze považovat jednoduché a intuitivní programování pomocí předprogramovaných prvků a rychlost, se kterou se dají naprogramovat rozsáhlé, především měřicí projekty. Jedná se o grafický programovací jazyk, proto není důležitá znalost syntaxe, takže je snadnější se s programem seznámit.

Nevýhody

Tím, že jsou všechny bloky předem naprogramovány, nemůže programátor dojít k cíli libovolnou cestou. Vždy je něčím omezen a často musí vymýšlet, jak tato omezení obejít.

2 LEGO MINDSTORMS NXT

2.1 Úvod

Historie robota MINDSTORMS sahá do roku 1980, kdy byla ve společnosti LEGO založena divize vzdělávacích produktů. Ve spolupráci s univerzitou MIT (Massachusetts Institute of Technology) vznikla v roce 1998 první inteligentní kostka, označená jako RCX. Programovala se v jazyce NQC (Not Quite C) a komunikovala s počítačem přes infračervené rozhraní. NQC je založeno na jazyce C a je vytvořeno pro programování LEGO robota.
V roce 2006 přišla na trh nová řada LEGO robotů označená jako NXT 1.0. Tato sada byla používána pro účely této bakalářské práce a je popsána níže.
Od roku 2009 je v prodeji stavebnice NXC 2.0. Oproti verzi 1.0 se liší menším počtem LEGO dílků a světelný senzor byl nahrazen senzorem barevným, který je schopný rozpoznat šest základních barev (bílou, černou, žlutou, červenou, zelenou a modrou).

2.2 Technické vybavení

Stavebnice Lego MINDSTORMS NXT ve verzi 9797-1, se kterou jsem se seznamoval, obsahuje kromě velkého množství stavebních dílů také jednu řídicí jednotku, celkem tři elektrické motory, dva dotykové senzory, jeden ultrazvukový senzor a jeden zvukový senzor.

2.2.1 Elektronické vybavení v sadě LEGO MINDSTORMS 9797

Řídicí jednotka – NXT kostka

Obr. 2.1: Řídicí jednotka (NXT kostka)

Technická specifikace:

- 32-bitový mikroprocesor ARM7 s 256KB flash pamětí a 64KB RAM
- 8-bitový mikroprocesor s 4KB flash pamětí a 512B RAM
- bezdrátová komunikace Bluetooth třída II v2.0
- USB 2.0 port
- 4 vstupní porty pro připojení senzorů
- 3 výstupní porty pro elektrické motory
- 8kHz reproduktor
- zobrazovač 60x100 pixelů
- zdroj energie 6x 1,5V monočlánek AA, respektive lithiové baterie

Funkce:

NXT kostka je hlavní součástí (jádrem) stavebnice. S její pomocí se dají ovládat jednotlivé senzory a motory, ať už přímo, programem nainstalovaným do kostky nebo na dálku pomocí PC, mobilního telefonu atd. Má tři výstupní konektory označené písmeny A, B a C pro připojení servomotorů, čtyři vstupní konektory označené 1 až 4 a čtyři tlačítka pro ovládání programu na kostce. Oranžové tlačítko uprostřed zapíná robota a dále pak slouží k potvrzení výběru. Dolní šedé tlačítko je určeno pro návrat o úroveň níž a vypnutí robota. Dvě postranní šedá tlačítka ve tvaru šipek slouží pro listování v menu kostky.

Servomotor

Obr. 2.2: Servomotor

Technická specifikace:

- Zabudovaný rotační senzor pro měření úhlu s přesností na 1° v rozsahu 0-360° nebo měření počtu otáček
- Systém ozubených kol pro přenos síly na výstup

Funkce:

Umožňuje pohyb robota. Rychlost se dá nastavit v hodnotách v relativním rozsahu 0-100, kde hodnota 0 odpovídá klidovému stavu a 100 je hodnota nejvyšší rychlosti. Motor se může otáčet jak směrem dopředu, tak dozadu.

Omezení:

Servomotor má v sobě zabudovaný senzor na měření úhlu s přesností na 1°. S takovouto přesností však nelze dosáhnout požadované hodnoty. Ovládat servomotor lze pouze přes jeho rychlost a při rychlosti menší než 5 ze 100 (dolní hranice se může měnit v závislosti na síle potřebné k otočení) se servomotor přestává pohybovat a nelze tak dosáhnout požadované hodnoty úhlu.

Dotykový senzor

Obr. 2.3: Dotykový senzor

Použití: umožňuje detekovat objekt, do kterého robot narazil. Indikovány jsou dvě hodnoty a to buď 0, kdy dotykový senzor nezaznamenal překážku, nebo 1, což signalizuje stisknutí

čelního tlačítka senzoru. Ze zkušeností je však nutno říci, že tento senzor je pro praktické aplikace těžko použitelný. Chceme-li totiž dosáhnout změny stavu, musíme namířit senzor poměrně přesně na překážku tak, aby bylo dosaženo stisknutí tlačítka na jeho čelní straně. Jeho dalším možným využitím je spouštění robota. To by bylo docela užitečné v případě, že by nešel spustit na dálku.

Zvukový senzor

Obr. 2.4: Zvukový senzor

Technická specifikace:

- Měření akustického tlaku až do 90 dB
- Detekce zvukového pásma dBA (frekvence v pásmu slyšitelném pro člověka) a dB (frekvence bez ohledu na sluch člověka) v rozsahu 0-100%

Použití:

Se zvukovým senzorem je možné například spustit robota na dálku vytvořením nějakého většího hluku. Bohužel, tím tento senzor vyčerpává své možnosti, jelikož při spuštění robota servomotory vydávají vibrace a zvuk dost velký na to, aby znemožnil zvukovému senzoru dále rozeznávat změny v hlasitosti. Možností by bylo nějak senzor zvukově odizolovat.

Světelný senzor

Obr. 2.5: Světelný senzor

Technická specifikace:

- Zaznamenává intenzitu světla v rozsahu 0-100%
- Zaznamenává odrazivost povrchu v rozsahu 0-100%
- Lze použít se zapnutým či vypnutým infračerveným světlem

Použití:

Senzor se dá použít jako sledovač trasy, pokud máme na zemi dva podklady, z nichž každý má jinou odrazivost světla. Potom lze sledovat a řídit pohyb pouze po jednom z nich.

Ultrazvukový senzor

Obr. 2.6: Ultrazvukový senzor

Technická specifikace:

- Zaznamenává objekty a pohyb před sebou. Jeho princip spočívá ve vyslání a následném vyhodnocení odražené zvukové vlny
- Zaznamenává vzdálenost od objektu v rozsahu 0-250 cm s přesností na 3 cm

Použití:
Ultrazvukový senzor je jednoznačně nejpoužitelnější ze všech standardně dodávaných senzorů. Jeho hlavní výhodou je detekování objektů na dálku a možnost se jim dále vyhnout. Bohužel senzor nezaznamená vše. Problémy mu dělají měkčí materiály a tvary od kterých se zvuk odrazí mimo směr senzoru. Senzor také nezaznamená velký objekt, je-li k němu nastaven pod větším úhlem.

2.2.2 Další doplňky

Mimo výše zmíněné technické vybavení existuje ještě celá řada dalších samostatně prodávaných doplňků. Jejich výrobou se zabývá jak samotná firma LEGO, tak další společnosti. Velké množství různých senzorů vyrábí firma HiTechnic (gyroskop, akcelerometr, úhlový senzor, kompas, senzor magnetického pole atd.). Dalším výrobcem senzorů pro NXT MINDSTORMS je firma LogIT, která se specializuje především na výrobu senzorů schopných měřit různé fyzikální veličiny. Výrobou náročnějších senzorů se zabývá firma Mindsensors, která vyrábí například kamerový systém pro NXT. Dalšími výrobci

senzorů jsou Verner a CODATEX. Konstrukčními prvky se zabývá firma Pitsco, která nahrazuje plastové konstrukční prvky hliníkovými částmi.

2.3 Programové vybavení

Ve stavebnici LEGO MINDSTORMS NXT se setkáme se dvěma druhy programového vybavení. Prvním a velmi důležitým je firmware, který je nahraný ve flash paměti inteligentní kostky. Bez přítomnosti firmwaru by robot nebyl funkční. Druhým dodávaným programovým vybavením je pak vývojové prostředí NXT-G.

2.3.1 Firmware

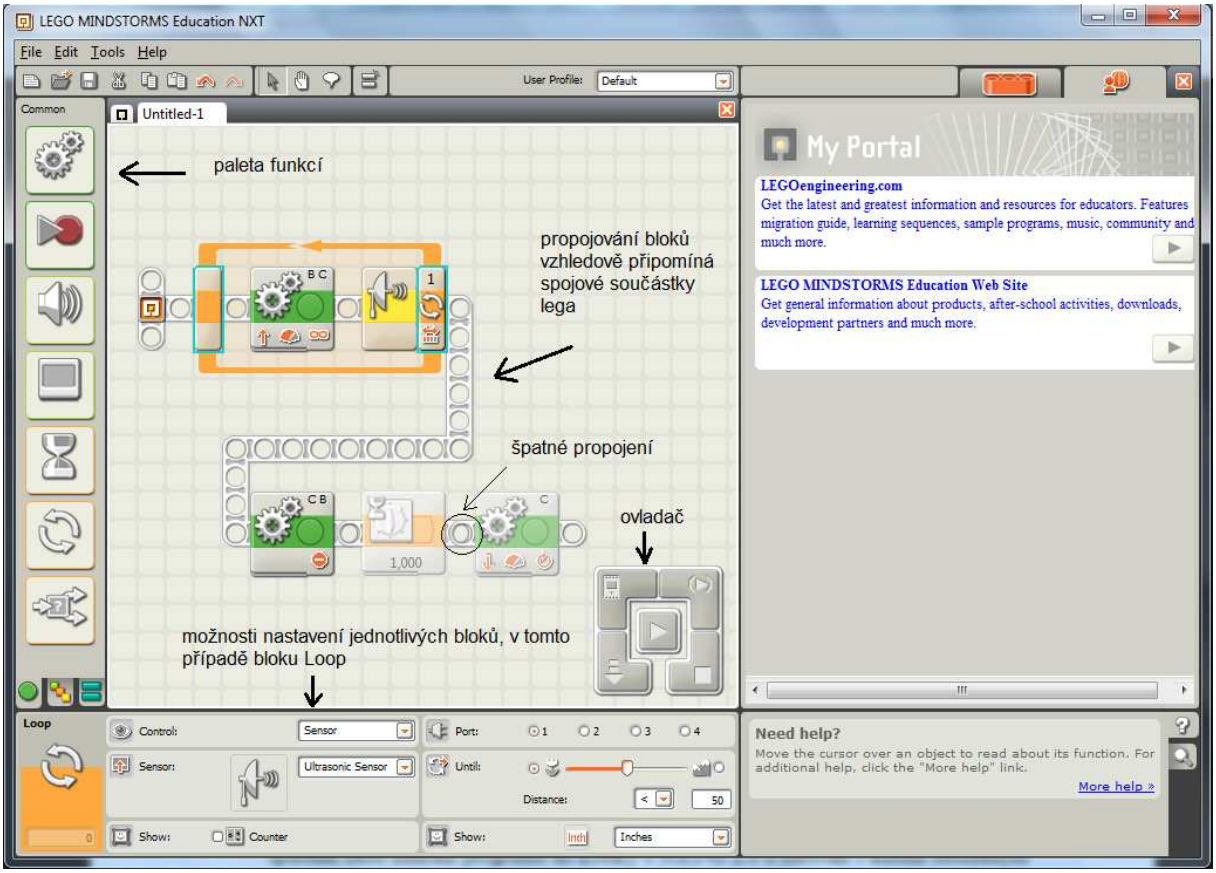
Firmware je jakýmsi operačním systémem elektronického zařízení, v tomto případě robota NXT. Firmware je vždy předem nahrán od výrobce do paměti, která je nezávislá na připojení elektrické energie, tedy nepotřebuje napájení. Tyto paměti se nazývají nevolatilní a jsou jimi například paměti ROM, flash EEPROM, atd. V našem případě je použita flash paměť, takže její obsah lze přepisovat a je možné nahrávat nové verze firmwaru, které firma Lego vydává, popřípadě nahrát firmware od jiné společnosti, která se vývojem aplikací pro Lego MINDSTORMS NXT zabývá. Firmware umožňuje uživateli ovládat samotnou inteligentní kostku a zařízení k ní připojená. Původní verze firmwaru tohoto robota byla 1.05. Od nových verzí 1.2x se odlišuje tím, že je pomalejší. Tato verze také nepodporuje nový NXT Module pro LabVIEW 2009 a vyšší. Pro tuto práci bylo tedy třeba původní firmware přehrát a nahradit jej verzí 1.26, která NXT Module již podporuje.

2.3.2 Vývojové prostředí NXT-G

Vývojové prostředí NXT-G je prostředí dodávané s NXT robotem. Toto grafické vývojové prostředí vzniklo (stejně jako programové prostředí LabVIEW) u firmy National Instruments a jedná se o prostředí s grafickým způsobem vytváření programů. NXT-G umožňuje jednoduché základní ovládání robota. Programování se provádí výběrem a spojováním jednotlivých programových bloků. Jsou zde bloky umožňující ovládání motorů a senzorů. Je k dispozici též jedna smyčka (Loop) pro opakování určité části programu. Také lze pracovat se zobrazovačem na NXT kostce.

Z grafického zpracování prostředí NXT-G jde vidět, že tento program je určen pro děti. Je zaměřen na vizuální stránku, aby jeho ovládání bylo intuitivní a aby zaujal malé

programátory. Vývojové prostředí je jednoduché, přehledné, ikonky jednotlivých funkcí jsou velké. Vše působí velmi příjemně, do chvíle než se začne programovat. Spojování jednotlivých funkčních bloků je provedeno v podobě jakýchsi spojovacích Lego modulů, avšak také trochu nešikovně. Spoje se vytváří automaticky ve chvíli, kdy se dva bloky přiblíží, někdy však ke spojení z nějakého důvodu nedojde. Přizpůsobování se Loop smyčky velikosti vložených bloků je také nešikovné a snad i trochu nedodělané. Ve výsledku sice působí prostředí pěkně, ale dokáže odradit od dalšího programování.



Obr. 2.7: Prostředí NXT-G

Na Obr. 2.7 je vytvořen program (v tomto případě záměrně nefunkční proto, abych ukázal špatné propojování bloků, po odstranění špatného spoje je program zcela funkční), který po spuštění roztočí motory na portu B a C stejnou rychlostí a zastaví je ve chvíli, kdy ultrazvukový senzor zaznamená před sebou překážku ve vzdálenosti menší než 50cm. Poté motory zastaví, počká 1 vteřinu, roztočí motor na portu C a otočí se doleva.

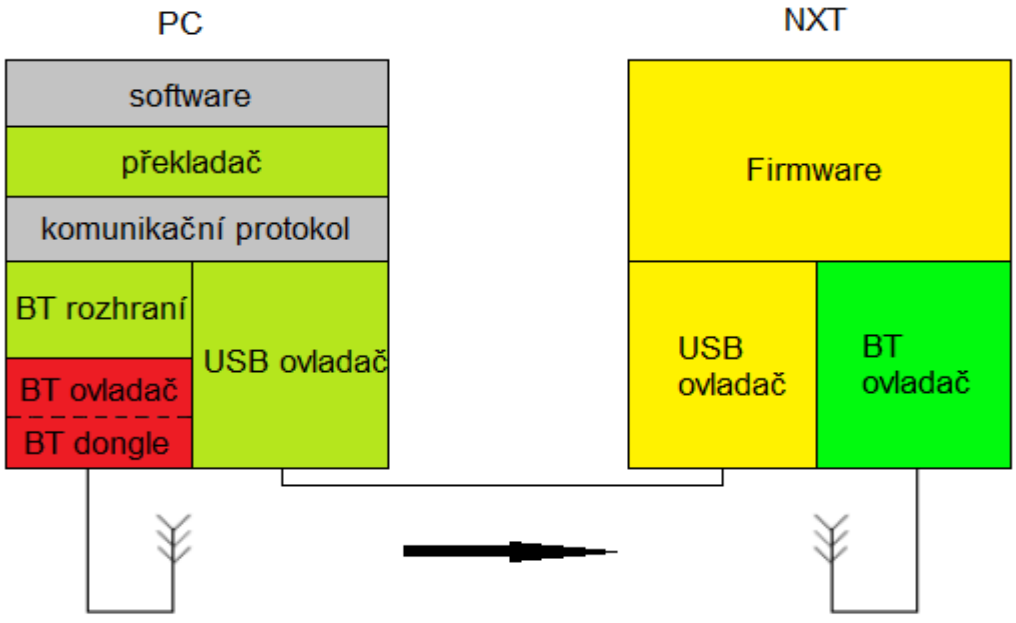
2.3.3 Další vývojová prostředí

Kromě zmíněného základního programového prostředí NXT-G, resp. v prostředí LabVIEW lze pracovat s LEGO robotem i v některých dalších vývojových prostředích. Příkladem je grafický způsob programování pomocí RoboLab, což je program založený na LabVIEW, vytvořený na Tuftské Univerzitě. Je možné programovat robota také v jazyce C, a to buď v běžném vývojovém prostředí pro jazyk C, nebo v prostředí přímo k tomu vytvořeném, které se jmenuje RobotC. Pro programátory vyznávající jazyk Java existuje prostředí leJOS NXT. Při používání leJOS NXT je nutné nahrát firmware k tomuto programu vytvořený, neboť s původním nepracuje. Ostatní uvedené programy pracují se základním firmwarem dodaným firmou Lego.

2.4 Komunikace

Komunikace mezi nadřazeným počítačem a robotem může probíhat v zásadě dvěma způsoby. Prvním způsobem je komunikace pomocí dodávaného USB kabelu a druhým pak komunikace přes bezdrátovou technologii Bluetooth.

Spojení robota NXT MINDSTORMS pomocí Bluetooth lze navázat s jakýmkoliv zařízením, které tuto technologii využívá, například s PC, mobilním telefonem a PDA zařízením. USB by tuto možnost teoreticky také umožňovalo, omezením je tu však koncovka USB typ A, kterou nemá každé zařízení. Možností by tak bylo pořízení vhodného převodníku, pro mobilní telefony třeba USB/microUSB.



Obr. 2.8: Komunikační vrstvy

Na Obr. 2.8 jsou znázorněny jednotlivé komunikační vrstvy, kterými prochází program, vytvořený na počítači a způsob, jakým se výsledný signál dostane do NXT kostky. Tedy za pomoci USB respektive Bluetooth rozhraní.

2.4.1 Komunikace pomocí USB kabelu

V základní sadě dodávané k robotu je k dispozici 2 metry dlouhý USB kabel, jehož jedna koncovka je USB typu A, která se zapojuje do počítače a druhá USB typu B, ta se připojuje do NXT kostky.

Výhody: Jednoduché a vždy funkční propojení. Navázání komunikace je okamžité.

Nevýhody: Vzhledem k pevnému spojení s počítačem a délce kabelu pouze 2 metry nelze toto spojení použít tam, kde se předpokládá pohyb po větší ploše.

2.4.2 Komunikace pomocí Bluetooth

NXT kostka má v sobě zabudované Bluetooth rozhraní, pomocí něhož může bezdrátově komunikovat s jiným zařízením podporujícím tuto technologii.

Výhody: Bezdrátová technologie nabízí možnost volného pohybu robota na větší vzdálenost, konkrétně na vzdálenost kolem 10 metrů od ovládacího zařízení. Přes Bluetooth je také možné propojit až čtyři NXT kostky, jedna NXT kostka je řídicí (master) a tři jsou podřízené (slave). Komunikace je ale navázána vždy pouze s jednou podřízenou kostkou.

Nevýhody: Navázání komunikace chvíli trvá. Při ztrátě spojení se musí komunikace znova pracně obnovit. Pro komunikaci mezi PC a robotem je ve většině případů zapotřebí použít USB klíčenku (tzv. Bluetooth dongle), který ale nezaručuje, že se propojení podaří. Některé počítače, zejména pak typu notebook, bývají vybaveny rozhraním Bluetooth již v základním provedení (např. HP, MSI).

3 Vytvořené programy

Cílem této bakalářské páce je ukázat, jak lze pomocí programového prostředí LabVIEW ovládat robota LEGO MINDSTORMS NXT. K tomuto účelu je nutné nejprve robota sestavit. K dispozici je sada plastových dílů, s jejichž pomocí robota lze složit. Je možné postavit robota buď podle návodu (ten lze získat buď oficiálně v knižní podobě nebo na internetu od uživatelů), nebo lze postavit robota podle vlastní fantazie a intuice. Druhou cestou jsem se vydal já.

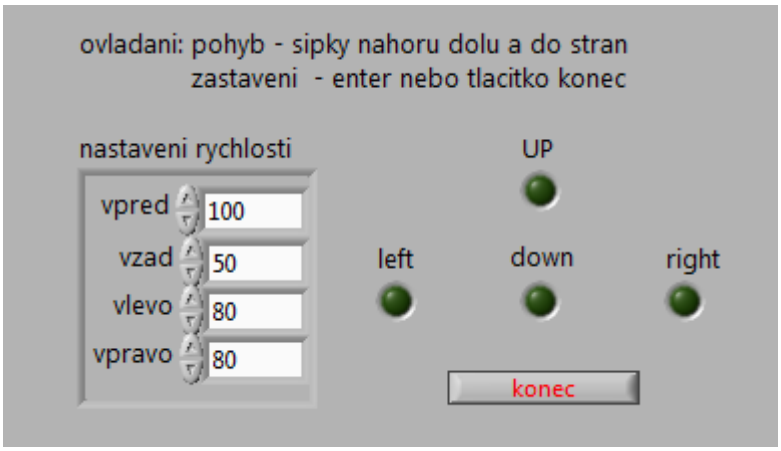
Sestavil jsem jednoduchého robota na kolečkách ovládaných dvěma motory a jedním kolečkem bez pohonu. Toto kolečko jsem umístil dozadu na střed robota tak, aby robot byl vyvážený, nepadal a zároveň měl snadné ovládání. Nejdříve jsem se sice snažil sestavit vše tak, aby výsledný robot jezdil na čtyřech kolech, tak jak je tomu u automobilů. Problém byl však v zatáčení, kdy dvě kola přidělaná k pevné ose bez možnosti natáčení podstatně zhoršovala jízdní vlastnosti. Tento problém by se dal vyřešit pohyblivým zavěšením os kol, což by vyřešilo potíže při zatáčení a ani jízdě vpřed by to nějak neuškodilo. Bylo by však nemožné jet s robotem směrem dozadu, což je jedna z funkcí robota na ovládání.
Dále jsem robotu přidělal ultrazvukový senzor, který je posazený na třetím elektromotoru, a zvukový senzor. K dispozici byly další 3 senzory: 2 x dotykový senzor a 1 x světelný senzor. Jejich využití je však velmi slabé. V jedné úloze jsem se snažil využít světelný senzor jako ochranu robota před spadnutím z výšky, pro případ, že by byl umístěn na vyvýšeném prostoru. Teoreticky taková ochrana má význam, prakticky však bohužel (jak bude popsáno dále) nikoliv. Je zde však příliš pomalá odezva na změnu intenzity světla, než aby robot včas zastavil.

3.1 Robot na ovládání

Tento program umožňuje pohybovat s vytvořeným robotem za pomoci směrových kurzorů na klávesnici. Při spuštění programu robot čeká na stisknutí libovolné směrové klávesy na klávesnici. Po jejím stisknutí se uvede do pohybu v daném směru a v tomto pohybu setrvá tak dlouho, dokud není určen jiný směr. Při pohybu dopředu a dozadu se roztočí dva servomotory ovládající kola stejnou rychlostí. Tu lze nastavit na čelním panelu (Obr. 3.1), zároveň se rozsvítí LED na čelním panelu, která indikuje aktuální směr jízdy. Při jízdě vlevo nebo vpravo

se upraví rychlost jednoho motoru na 0, tedy motor se zastaví, a druhý motor získá rychlost, kterou lze opět nastavit na čelním panelu, tímto se dosáhne zatočení robota.

Robot je dále vybaven ultrazvukovým senzorem, který zkoumá, zda-li není ve směru jízdy nějaká překážka. Jestliže robot vyhodnotí, že je před ním překážka, robot se zastaví a popojede mírně dozadu. To z toho důvodu, aby při následném otočení opět ultrazvukový senzor nezaznamenal ten samý objekt a znovu se mu nevyhýbal. Ukončení programu lze provést dvěma způsoby, a to buď stisknutím tlačítka „Konec“ na čelním panelu nebo stisknutím tlačítka Enter na klávesnici počítače.



Obr. 3.1: Čelní panel programu Robot na ovládání

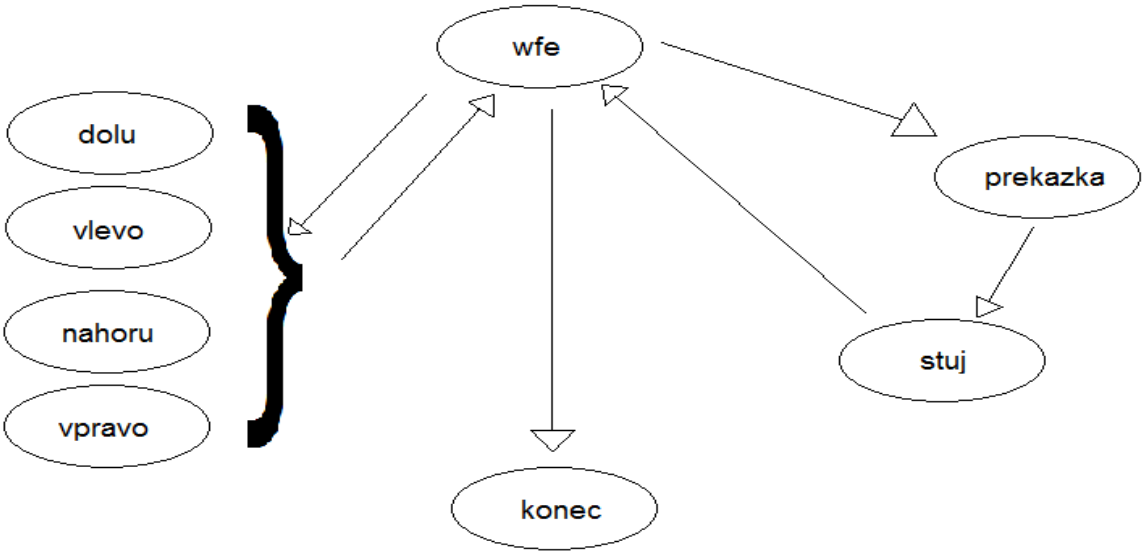
3.1.1 Účel programu Robot na ovládání

Snažil jsem se zde především ukázat dvě věci. Tou první je předvedení možnosti využít klávesnici jako ovládací zařízení robota. Tou druhou (a podstatnější) věcí je ukázka způsobu komunikace. Je zde patrné, že probíhá neustálá komunikace mezi robotem a počítačem, kdy robot reaguje na změnu provedenou uživatelem. Program tedy není po spuštění nahrán do paměti ovládací kostky robota (byl by zde i problém s malou vnitřní pamětí kostky), ale je uložen v paměti počítače, s robotem neustále komunikuje a odesílá instrukce.

3.1.2 Struktura programu Robot na ovládání

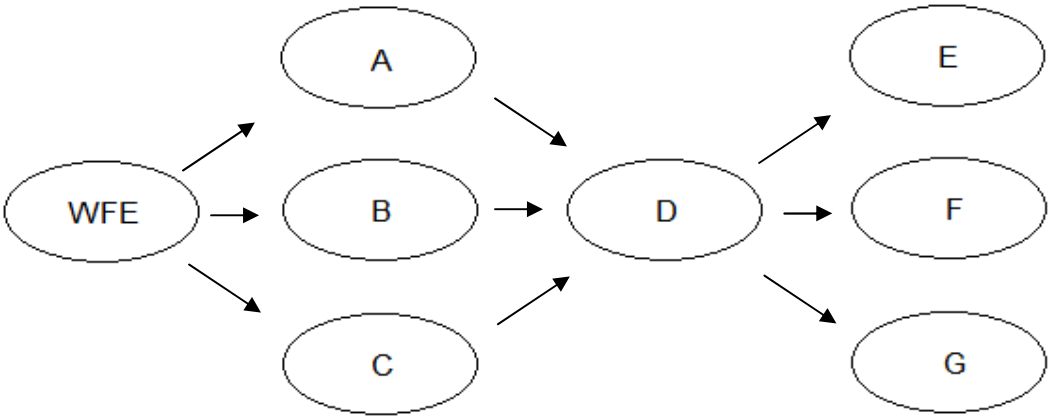
Jako programovou strukturu jsem zvolil základní typ programování konečného stavového automatu. Jeho výhodou je snadné a přehledné programování. Programové části jsou rozděleny do více stavů, kde každý stav vykonává jednoduchou úlohu. Cílem tohoto způsobu programování je rozdělit komplexní program na řadu dílčích jednoduchých programů. Každý takto vytvořený program má přesně daný úkol a je v něm jasně určeno, do jaké části programu

má po ukončení přejít. Díky tomuto způsobu programování se zlepšuje přehlednost programu a je daleko lehčí vytvořený program později upravovat podle potřeb. Nevýhodou tohoto stavového automatu je, že každý stav odkazuje pouze na jeden následující stav. Tento problém je vysvětlen na Obr. 3.3.



Obr3.2: Schéma Robota na ovládání

Na Obr. 3.2 je znázorněno použité schéma popisovaného programu. Ve stavu „wfe“ se čeká na nějakou událost, jako je třeba stisknutí směrového tlačítka. Po stisknutí tlačítka se přechází na zvolenou funkci, kde se vykoná příslušné nastavení rychlosti otáčení servomotoru. Po nastavení se signál opět vrací do stavu „wfe“, kde se opět čeká na další změnu směru. V případě, že k žádné změně nedochází, kontroluje si zde průběžně, jestli není před robotem překážka. Zjistí-li, že ano, přejde do stavu „prekazka“, kde nastaví zpětný chod motorů a jede směrem vzad do doby, než se vzdálí od překážky na určitou vzdálenost. Potom přechází do stavu „stuj“. Tam zastaví motory a opět se posouvá do stavu „wfe“. Po každé akci se tedy program vždy vrací do stavu „wfe“, odkud se pak dá ukončit.



Obr. 3.3: Ukázka problému, který stavový automat neřeší

Na Obr.3.3 je ukázka posloupnosti stavů, která se nedá jednoduše řešit pomocí stavového automatu. Ze stavu „WFE“ máme možnost přejít do tří dalších stavů, podle zvolené události. Z každého z těchto tří stavů se potřebujeme dostat do stavu „D“ a odtud plynule do jednoho z dalších tří stavů. Problémem je, že ve stavu „D“ je pouze jeden odkaz na další stav. Proto se nedá přejít do libovolného následujícího stavu. Toto řeší stavový automat s frontou, který je popsán níže.

3.1.3 Problémy

Menším problémem v tomto programu je fakt, že se klávesnice nedá nakonfigurovat podle vlastních požadavků. Velká část funkčních tlačítek klávesnice má svoji samostatnou událost, ale z nějakého důvodu například klávesa „mezerník“ a „tabulátor“ jsou brány jako ASCII klávesy spolu s mnoha dalšími tlačítky, jsou tedy pod jednou skupinou a nelze je tak samostatně programovat.

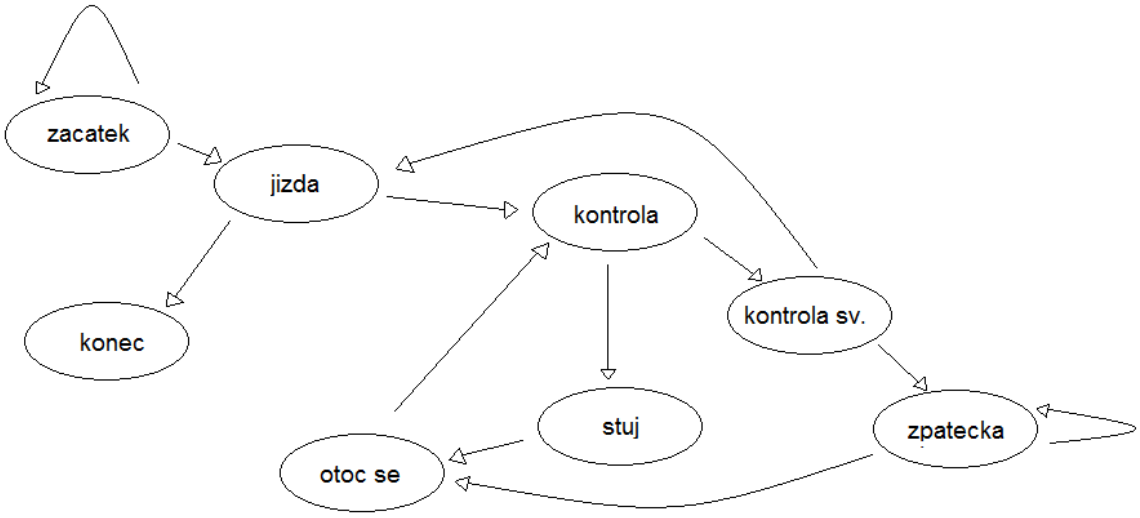
3.2 Automatický robot

Tento program umožňuje robotu volně se pohybovat po prostoru a v případě překážky změnit směr. V tomto případě směr vždy mění vlevo. Program se spouští přes zvukový senzor, který čeká na nějakou hlasitější událost (tlesknutí, úder do země atd.). Po této události se robot rozjede v přímém směru. Přiblíží-li se k překážce na určitou vzdálenost, zastaví a otočí se doleva. Dále pak pokračuje v jízdě. V případě, že po otočení zjistí před sebou další překážku ve vzdálenosti menší než je povolená, otočí se znovu. Rozjede se pouze tehdy, není-li před ním žádná překážka.

V programu je také využit světelný senzor. Ten může mít dvě úlohy: buď vymezuje prostor, po kterém se robot smí pohybovat (jestliže je prostor rozdělen na dvě barvy, kde jedna má výrazně jinou odrazivost světla než druhá), nebo (pokud je robot na vyvýšené rovné ploše) zamezuje spadnutí robota z této desky. To ale není moc použitelné, protože reakce robota na změnu světla je příliš pomalá. Tato funkce se dá použít pouze při zvolení velmi pomalého pohybu vpřed.

3.2.1 Struktura programu Automatický robot

Použitá struktura je stejná jako u předchozího modelu, tedy v podobě konečného stavového automatu.



Obr. 3.4: Schéma Automatického robota

Z Obr. 3.4 lze vyčíst způsob funkce programu. Ve stavu „zacatek“ čeká robot na zvukový signál. V případě, že nepřichází, kontroluje svůj stav do doby, než signál přijde. Poté se přesune na jízdu. Ta v každém cyklu přechází do stavu „kontrola“, kde nejdřív kontroluje vzdálenost robota od překážky, a pokud vyhodnotí, že překážka před robotem není, přechází na kontrolu světelné změny. V případě, že překážka před robotem je, robot se zastaví, otočí a opět proběhne kontrola vzdálenosti a intenzita světla. Pokud se změna světla nezmění, přechází robot na jízdu, pakliže barevný senzor vyhodnotí změnu, robot přejde do stavu „zpatecka“, kde popojede od okraje. Ze stavu „zpatecka“ pak přechází na stav „otoc se “ a vše probíhá znovu. Ukončit program lze pouze ze stavu „jizda“.

3.2.2 Účel programu Automatický robot

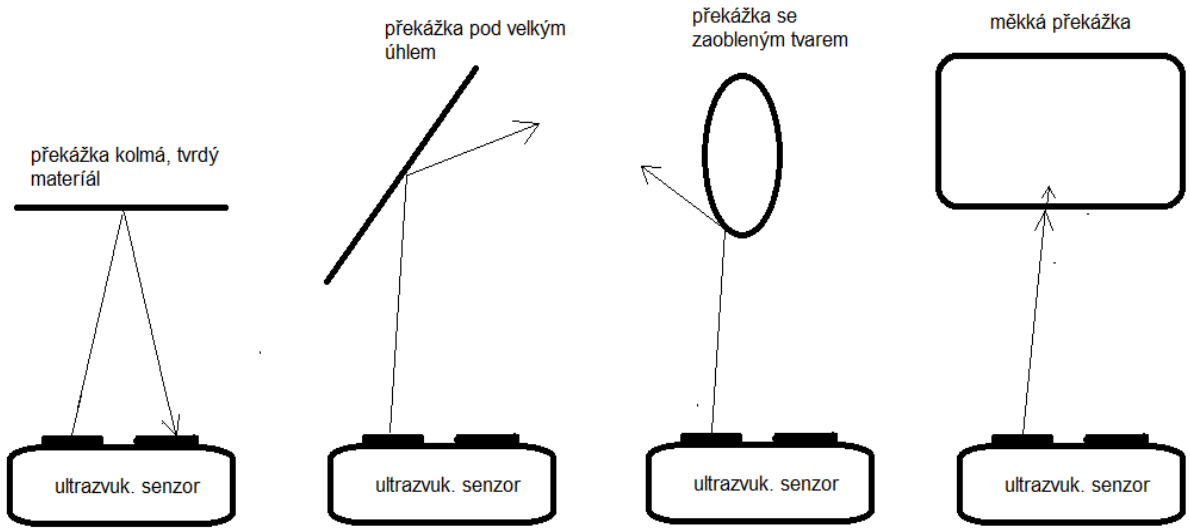
Tímto programem jsem se snažil předvést nevýhodu v programování pomocí stavového automatu. Takto popisovaný automat totiž zpracovává vždy jen jeden stav v daný okamžik. Problém nastává v případě, že se vytvoří řada stavů, které na sebe navazují bez možnosti se z této posloupnosti dostat. Zde se problém projeví tak, že se robot pohybuje ve směru vpřed a každou chvíli kontroluje, zda-li není před ním překážka a následně, zda-li není na hraně. Tyto úkoly trvají nějakou dobu, i když třeba zanedbatelnou. Po zjištění, že je před robotem překážka, popřípadě že je pod ním prázdný prostor, se robot zastaví až se zpožděním, které je způsobenou vedle setrvačnosti motorů právě i zpožděním při přechodu z jednoho stavu do jiného. Toto zpoždění je u kontroly prostoru pod robotem velmi závažné.

3.2.2 Problémy

Jelikož je robot ovládán automaticky bez možnosti korigovat směr jeho jízdy, projevuje se zde to, že servomotory, ač jsou nastaveny stejně, se stejně netočí. Příčin může být několik. Kola mohou být různě zatížená, tedy i síla potřebná k pohybu je u obou motorů rozdílná, nebo mohou být kola různě veliká. Dále je možné, že motory jsou jen lehce odlišné (větší vůle převodových koleček), což má za následek mírné zatáčení na jednu stranu.

Problémem ultrazvukového senzoru je jeho horší schopnost detekovat předměty pod vysokým úhlem, předměty zaoblené tak, že neodrážejí signál směrem k přijímači a měkké předměty, které ultrazvuk pohlcují, viz Obr. 3.5. Proto může robot snadno narazit do předmětu, který je pro něj neviditelný.

Posledním problémem je možnost využít zvukový senzor pouze na spuštění vzhledem k tomu, že po aktivaci servomotorů vzniká hluk, který zcela zahltí senzor.



Obr. 3.5: Způsob detekce objektu. Vlevo objekt, který senzor zaznamená. Na dalších obrázcích příklady objektů, které ultrazvukový senzor nezachytí. nezachytí (odražený zvuk se nevrátí do senzoru zpět).

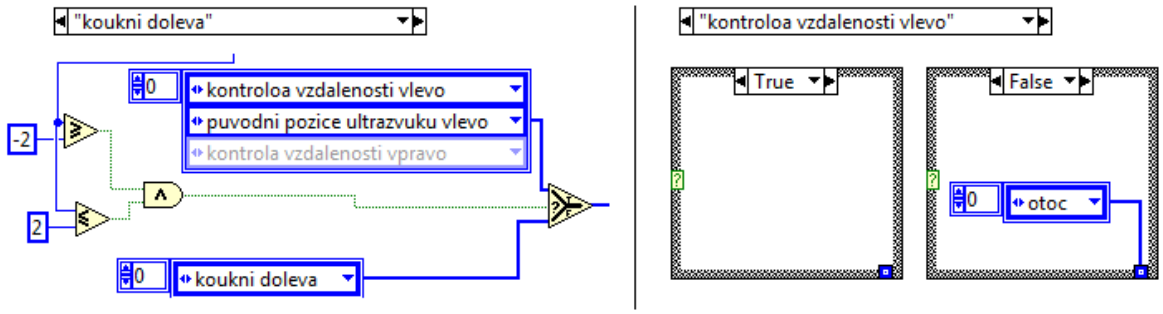
3.3 Automatický robot 2

Tento program je vylepšenou verzí programu Automatický robot popsaného výše. Jeho odlišnost je v tom, že ultrazvukový senzor je upevněn na třetí servomotor tak, aby se mohl otáčet ve vodorovném směru. Při spuštění se robot rozjede a pokračuje v jízdě, dokud před sebou nedetekuje nějakou překážku. Když zjistí, že ve směru jeho jízdy je překážka, zastaví se a natočí ultrazvukový senzor směrem doleva. Pokud vlevo nezjistí žádnou překážku, vrátí ultrazvukový senzor do původní polohy, otočí se doleva a pokračuje v jízdě. Zjistí-li, že mu vlevo něco překáží, otočí senzorem doprava a opět se „podívá“, jestli tam není překážka. Pokud žádnou nezjistí, vrátí senzor do původní polohy, otočí se doprava a pokračuje v cestě. V opačném případě vrátí senzor do původní polohy a ukončí svou činnost.

3.3.1 Struktura programu Automatický robot 2

Je zde použit konečný stavový automat s frontou. Rozdíl mezi automatem s frontou a bez fronty je ten, že u stavového automatu bez fronty má každý stav pouze jeden platný výstupní stav. U stavového automatu s frontou lze vytvořit posloupnost stavů, které mají být splněny. Pokud je tedy výstupu ze stavu přiřazena posloupnost stavů, na které se má přejít, uloží se tyto stavy do fronty. Poté se vezme první stav z fronty, přejde se na jeho vykonání a na konec fronty se zapíše nový stav.

Pro lepší pochopení uveďme příklad na vytvořeném programu Automatický robot 2.



Obr. 3.6: Způsob práce stavového automatu s frontou

Na Obr. 3.6 je v levé části zobrazen výstup ze stavu „koukni doleva“. V případě, že ultrazvukový senzor není natočen doleva o 90° ± 2°, do fronty se zapíše pouze jeden stav a ten se ihned vykoná. Ve chvíli, kdy senzor dosáhne požadovaného natočení, zapíší se do fronty dva stavy a to „kontrola vzdalenosti vlevo“ a „puvodni pozice ultrazvuku“. Poté se vykoná první stav. Výstupem z tohoto stavu je buď stav „otoc se doleva“, který se do fronty zařadí na konec v případě, že ultrazvukový senzor nenajde vlevo žádnou překážku, nebo výstupem bude prázdný stav, který do fronty nezařadí nic. Pokud překážka vlevo nebyla detekována, tak jsou v tu chvíli ve frontě stavy „puvodni pozice ultrazvuku“ a „otoc se doleva“. Vykoná se tedy stav „puvodni pozice ultrazvuku“, jejíž výstup v tomto případě je prázdný stav, vymaže se z fronty a přejde se na vykonání posledního stavu „otoc se doleva“. Pokud je vlevo detekován předmět, přejde se do stavu „puvodni pozice ultrazvuku“, kde se do fronty zařadí nový stav „koukni doprava“.

3.3.2 Účel programu Automatický robot 2

Účelem je tu vytvoření programu s používanější strukturou a složitější vyhodnocovací logikou. U předchozího programu docházelo pouze k vyhodnocování překážky před robotem a případnému otočení doleva. To by mohlo mít v určitých podmínkách (robot v malém uzavřeném prostoru) za následek, že se bude neustále otáčet dokola. Zde si robot kontroluje, zda-li má možnost zabočit do stran. V případě, že tuto možnost nemá, ukončí svou činnost.

3.3.3 Problémy

Hlavním problémem je špatná schopnost natáčet servomotor o požadovaný úhel. I když je v přídavném NXT Module funkce otáčení o určitý úhel předem naprogramována, je velmi nepřesná a je nutné tuto funkci výrazně poupravit. I tak však není servomotor schopen natočit se o daný úhel s odchylkou menší jak 3°. Při několika málo pohybech se to může zdát jako nepatrný problém, ale bohužel chyby se v tomto případě sčítají a může dojít k nepředvídané chybě. Ve většině případů se však naštěstí chyba po několika otáčkách sama napraví.
Další nedostatky jsou stejné jako u předchozího programu.

Závěr

V práci byla vyzkoušena možnost naprogramovat a následně ovládat malého robota LEGO MINDSTORMS NXT za pomoci vývojového prostředí LabVIEW rozšířeném o doplněk pro ovládání robota LabVIEW Module for LEGO MINDSTORMS NXT. Toto rozšíření umožňuje jednoduše ovládat jednotlivé senzory, servomotory a zobrazovač na inteligentní NXT kostce. Je zde odstraněna možnost nahrát vytvořený program do paměti NXT kostky, nyní lze ovládat robota pouze vzdáleně z osobního počítače. Ubyla nutnost vkládat do programu bloky pro navázání komunikace s robotem přes rozhraní Bluetooth. Nyní toto navázání probíhá automaticky. Firma LEGO spolu s National Instruments se tedy vydala cestou zpříjemnění programování pro koncového uživatele.

Konkrétní sestavení LEGO robota je čistě na fantazii a schopnostech uživatele. Mnou sestavený robot má výhodu dobré stability při pohybu a umístění ultrazvukového senzoru v popředí robota, což umožňuje včasné detekování objektů před robotem. Nevýhodou této konstrukce je velká vzdálenost poháněných kol od sebe, z čehož plyne horší ovladatelnost.

Pro komunikaci s počítačem jsem využil technologie Bluetooth. Ta má dosah přibližně 10 metrů. Se vzrůstající vzdáleností se však zhoršuje komunikace. Proto je dobré nastavit ochranu před nárazem, zajišťovanou ultrazvukovým senzorem, na větší vzdálenost od objektu.

Pro vlastní programování jsem použil metodu stavových automatů. Ta rozděluje program na množství menších, jednoduchých programů, které jsou mezi sebou provázány. Stavové automaty jsou přehlednější při náročnějších aplikacích a umožňují relativně snadné úpravy programu.

Robot LEGO MINDSTORMS NXT je určen pro výuku robotiky a jeho využití ve skutečném provozu je prakticky nemožné. Robot v základní sadě NX 1.0 by mohl přemisťovat malé a lehké objekty z pásu, ale pouze tam, kde není za potřebí rychlé a přesné práce.

Robot LEGO MINDSTORMS NXT je tedy výborný výukový robot, který dá uživateli představu o robotice. Zde však jeho úloha končí a reálnou práci přenechává profesionálním robotům.

Seznam použité literatury

[1] National Instruments [online] URL: http://www.ni.com

[2] Travis Jeffrey, Kring Jim : LabVIEW for Everyone: Graphical Programming Made Easy and Fun, Third Edition, Prentice Hall 2006, 1032 stran

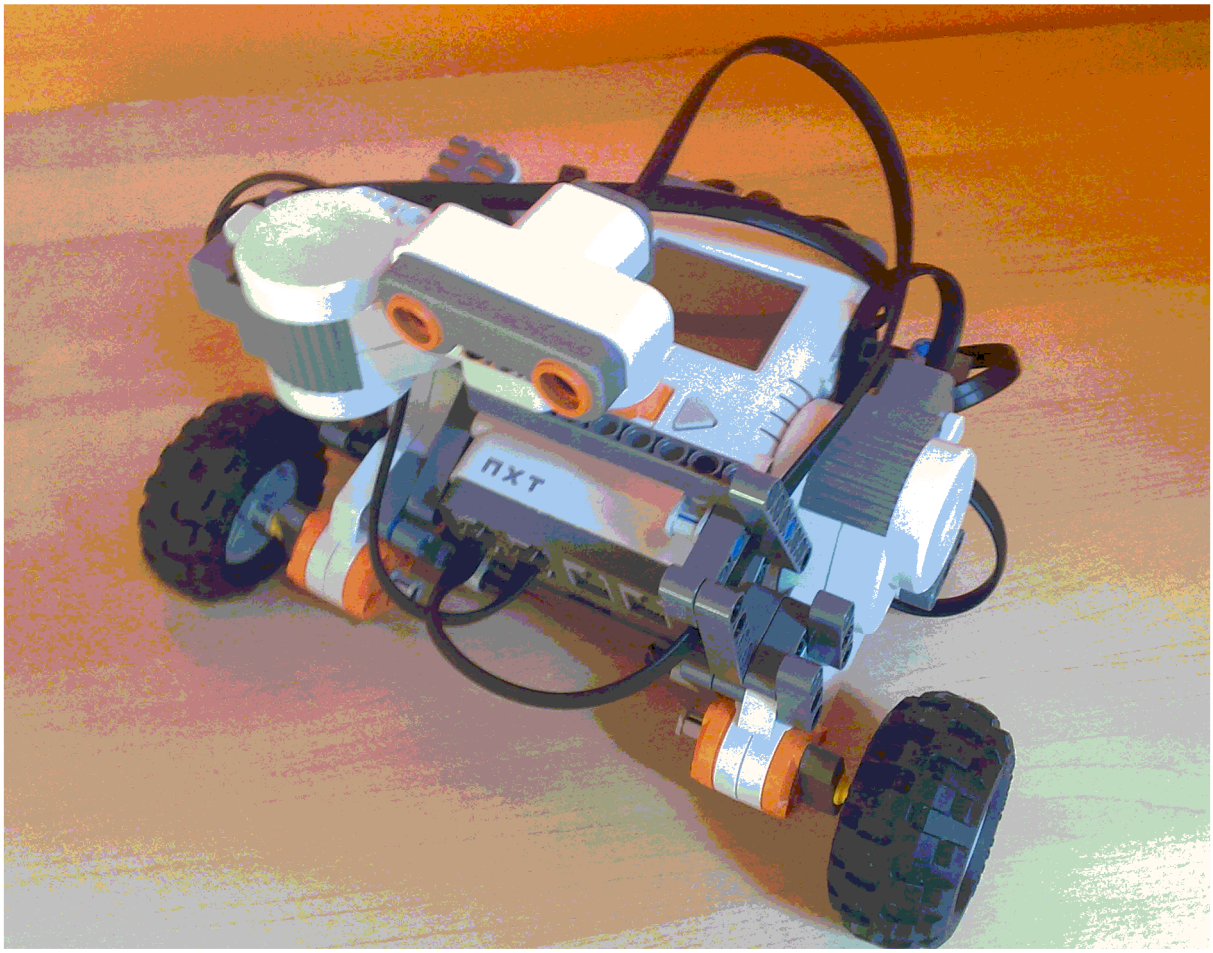
[3] Pokorný Jan: Řízení malého robota LEGO MINDSTORMS NXT ve vývojovém prostředí LabVIEW, bakalářská práce, FM TU Liberec 2010

[4] LEGO MINDSTORMS [online]. URL: http://mindstorms.lego.com

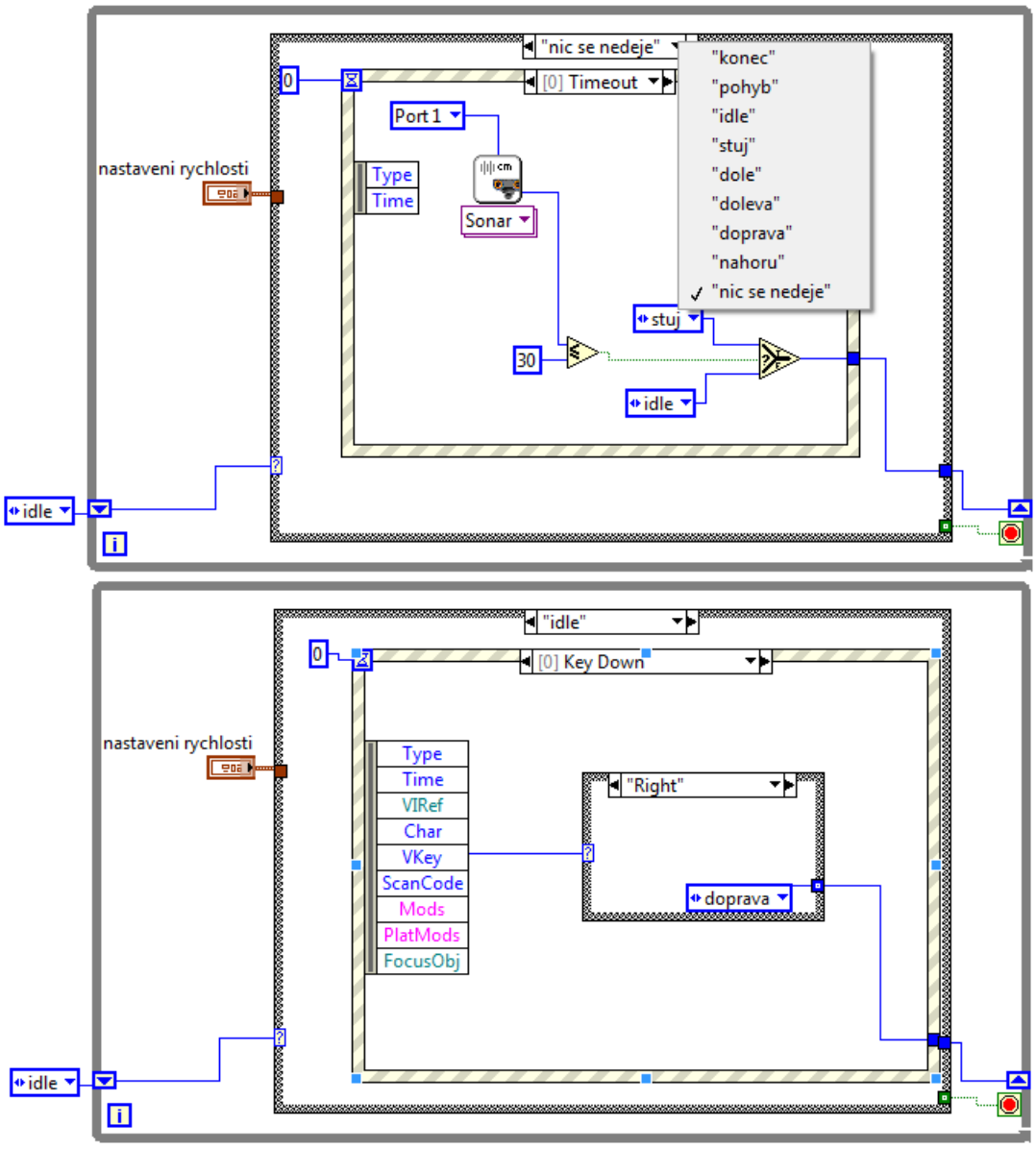
[5] Vlach, J., Havlíček, J., Vlach, M: Začínáme s LabVIEW, BEN Praha 2008, 248 stran

[6] internetový slovník Wikipedie [online] URL: http://www.wikipedia.org

Přílohy

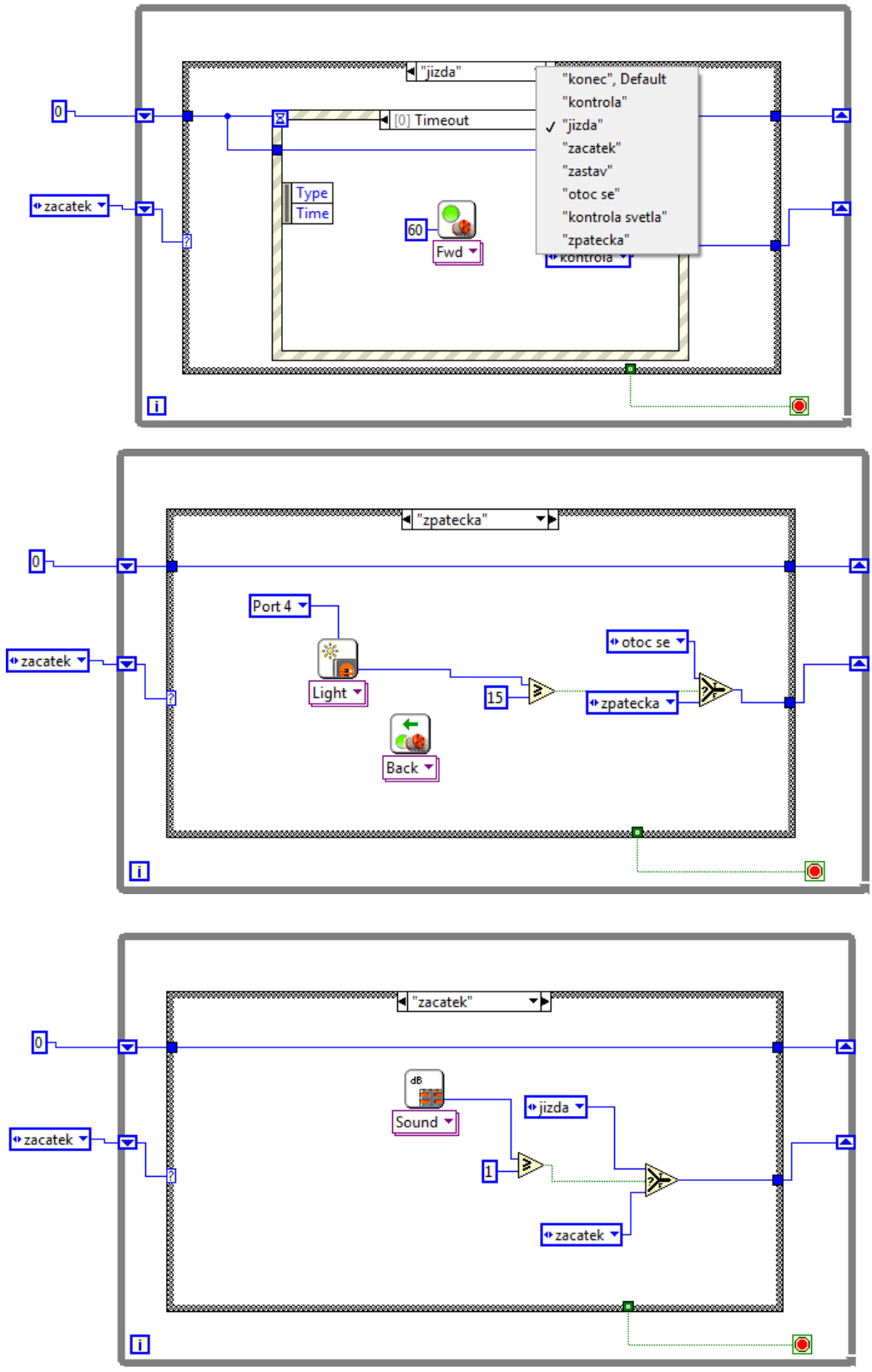


Vytvořený robot



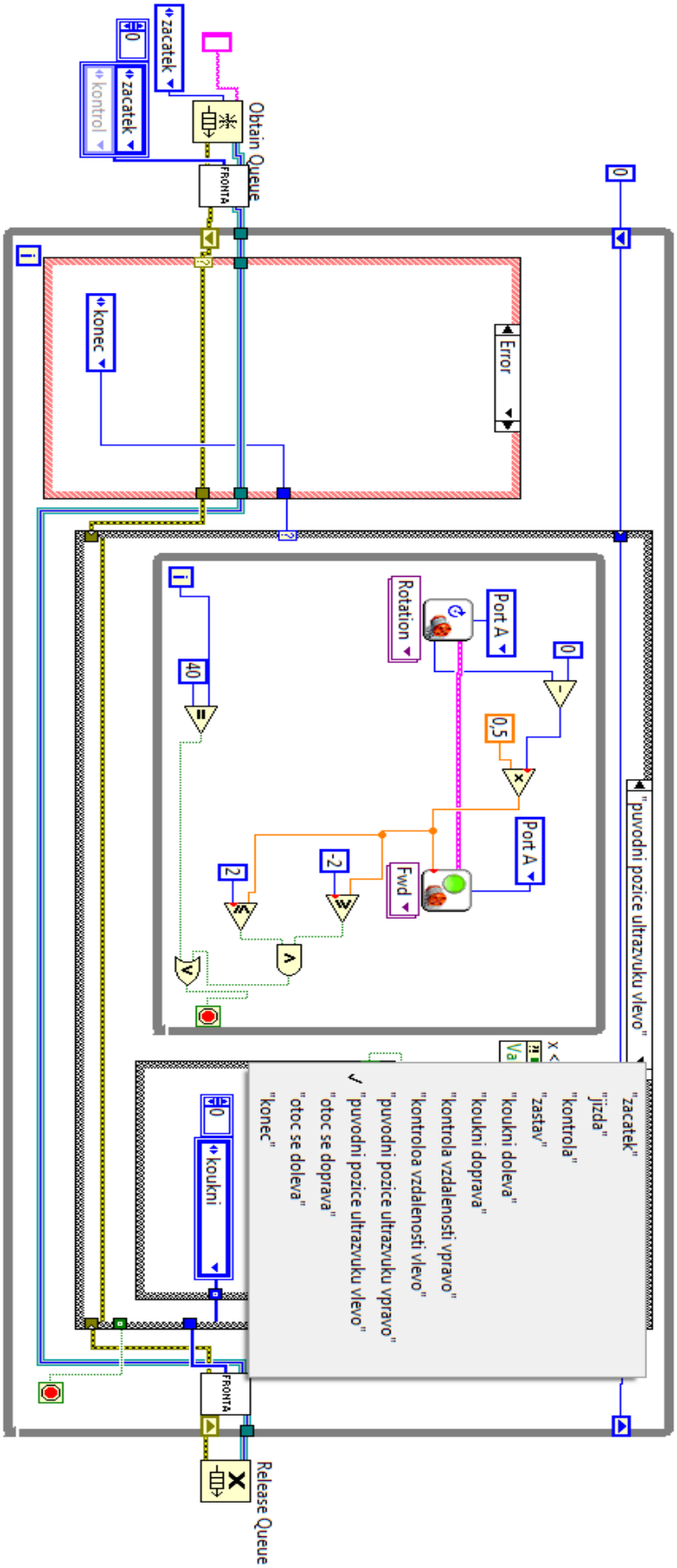
Ukázka části programu

Automatický robot



Ukázka částí programu automatický robot

Automatický robot 2



Ukázka části programu automatický robot 2