Univerzita Palackého v Olomouci

Přírodovědecká fakulta

Společná laboratoř optiky

Přístrojová fyzika

Programování sbRIO robota v prostředí NI LabVIEW Robotics

Bakalářská práce

Vypracovala: Pavlína Popelková

Vedoucí: RNDr. Jiří Pechoušek, Ph.D.

2014

Bibliografická identifikace

Jméno a příjmení autora	Pavlína Popelková
Název práce	Programování sbRIO robota v prostředí NI LabVIEW Robotics
Typ práce	bakalářská
Pracoviště	Společná laboratoř optiky
Vedoucí práce	RNDr. Jiří Pechoušek, Ph.D.
Rok obhajoby práce	2014
Počet stran	51
Počet příloh	7
Jazyk	český
Abstrakt	Tato bakalářská práce se zabývá programováním robota Starter Kit 1.0, založeného na Single-Board RIO kontroléru, v prostředí NI LabVIEW Robotics. Obsahem práce je detailní popis veškerého hardwaru a softwaru, který je nutný pro správnou funkčnost a komunikaci s robotem. Dále jsou zde podrobně vysvětleny všechny programy, prostřednictvím nichž může i neznalý člověk robota jednoduše ovládat, a jejichž vytvoření bylo mimo jiné hlavním cílem této práce.
Klíčová slova	robot, LabVIEW

Bibliographical identification

Author's first name and surname	Pavlína Popelková
Title	Programming of Starter Kit 1.0 robot in NI LabVIEW Robotics environment
Type of thesis	bachelor
Department	Joint Laboratory of Optics
Supervisor	RNDr. Jiří Pechoušek, Ph.D.
Year of presentation	2014
Number of pages	51
Number of appendices	7
Language	czech
Abstract	This thesis deals with programming of Starter Kit 1.0 robot, based on Single-Board RIO controller, in NI LabVIEW Robotics environment. The thesis includes a detailed description of all the hardware and software essential for the proper functionality and comunication. There are also full explanations of all the programs by which anyone can control the robot and which are the main goal of this thesis.
Keywords	robot, LabVIEW

Čestné prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně. Veškeré použité podklady, z nichž jsem čerpala informace, jsou uvedeny v seznamu použitých zdrojů a citovány v textu.

V Olomouci dne ....

podpis

Poděkování

Chtěla bych poděkovat RNDr. Jiřímu Pechouškovi, Ph.D. za jeho vstřícnost, rady a odbornou pomoc při vypracování této bakalářské práce.

Obsah

Úvod

Robotika

Robotika je věda, která se zabývá roboty, jejich konstrukcí, designem a obecně jejich ovládáním. Počátky robotiky spadají do 20. století, kdy se vůbec poprvé objevilo slovo robot. Výraznější rozvoj ovšem pozorujeme až od druhé poloviny tohoto století, kdy byl vytvořen první plně automatizovaný stroj. V roce 1942 byly navíc vědecko-fantastickým spisovatelem Isaacem Asimovem formulovány zákony robotiky.

V dnešní době je využití robotů velmi rozšířené, a to z mnoha důvodů. Práce robotů je mnohem efektivnější, přesnější a levnější. Nahrazují také člověka při práci v příliš znečištěných nebo nebezpečných prostředích. Příkladem může být využití robotiky v automobilovém průmyslu, kde roboti na sériových linkách sestavují nové automobily. Pro armádní účely se naopak roboti využívají například k odstraňování bomb. Zajímavé je také využití exoskeletonů, což jsou bionické konstrukce, ovládané nervovými signály z mozku. Slouží zejména pro rehabilitační účely v lékařství, kde mimo jiné nalezneme i roboty provádějící různé chirurgické zákroky. Zřejmě nejsledovanějším robotem je ale v poslední době robot Curiosity, jehož posláním je dlouhodobý výzkum planety Mars [1]. Úkolem Curiosity je zjistit, je-li prostředí na této planetě schopné podpořit vývoj malých forem života – mikrobů. Robot zde byl vysazen roku 2012 a od té doby nafotil mnoho snímků. Nejaktuálnější z nich zachycuje přípravu vrtného testu do kamene nazvaného Windjana, jehož cílem je odběr práškového vzorku materiálu. Fotografie je na obr. 1.

Obr. 1: Příprava vrtného testu na Windjaně.

(http://www.nasa.gov/jpl/msl/pia18089/#.U2NIwVdq5hE)

Již od počátků robotiky je ale snahou vytvořit robota, který bude plně autonomní, co nejvíce podobný člověku jak chováním, tak i vzhledem. V současné době je studium robotiky velmi rozšířené a je umožněno již žákům na středních školách. Vývoj robotiky je navíc podpořen mnoha projekty, v nichž žáci soutěží se svými vlastními roboty o hodnotné ceny. Proto můžeme do budoucna zcela jistě očekávat, že se obor robotiky bude dále rychle rozvíjet a nepochybně velmi zdokonalovat.

1. Hardware

1.1 Starter Kit 1.0

Starter Kit 1.0 je robot navržený firmou National Instruments. Je založen na Single-Board RIO kontroléru (sbRIO – viz 1.2 sbRIO 9631), který je upevněn na horní části konstrukce robota. Pomocí digitálních vstupů/výstupů (dále jen I/O) je k sbRIO desce připojen ultrazvukový senzor Paralax PING))), díky kterému se robot orientuje v prostředí (viz 1.3 Parallax PING))) ultrazvukový senzor vzdálenosti), servo motor, na kterém je tento senzor připevněn a dva motory na stejnosměrný proud, které pohání kola robota (4 Pitsco Education TETRIX kola, průměr 10,16 cm). [2]

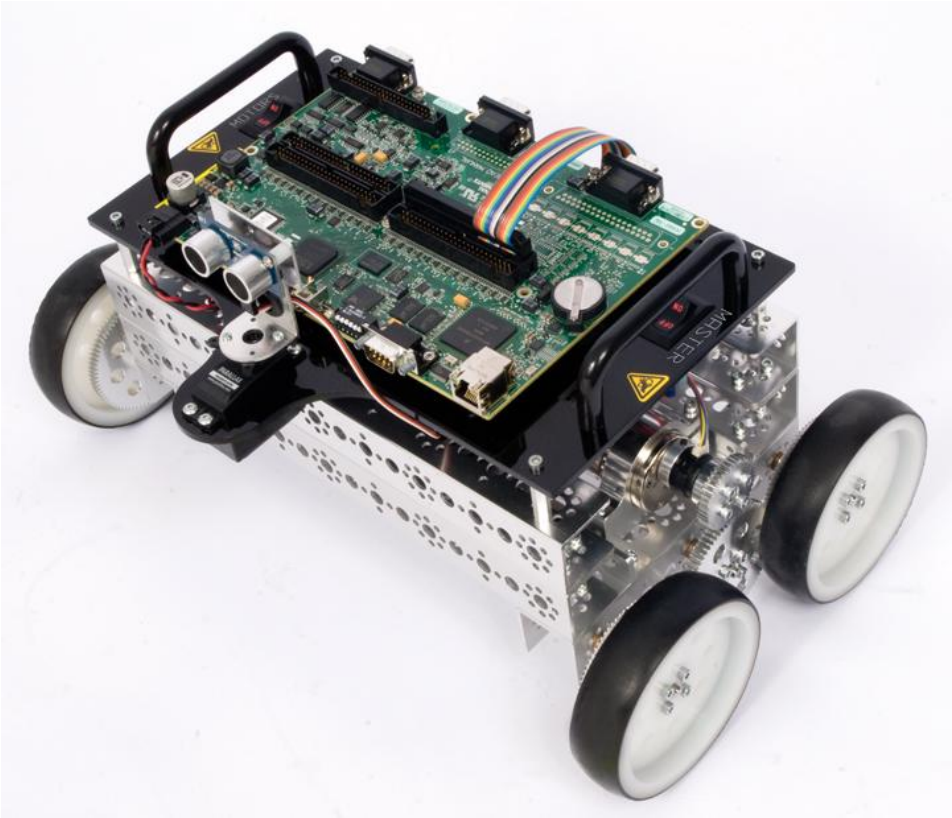
Parametry motorů na stejnosměrný proud:

- Napájecí napětí 12 V,
- točivý moment 300 oz-in.,
- otáčky za minutu 152 ot/min.

Aby bylo možné s robotem komunikovat, je potřeba následující software (viz 2. SOFTWARE):

- LabVIEW,
- LabVIEW Real-Time Module,
- LabVIEW FPGA Module,
- LabVIEW Robotics Module,
- RIO ovladače.

Robot je zobrazen na obr. 2.



Obr. 2: Robot Starter Kit 1.0.

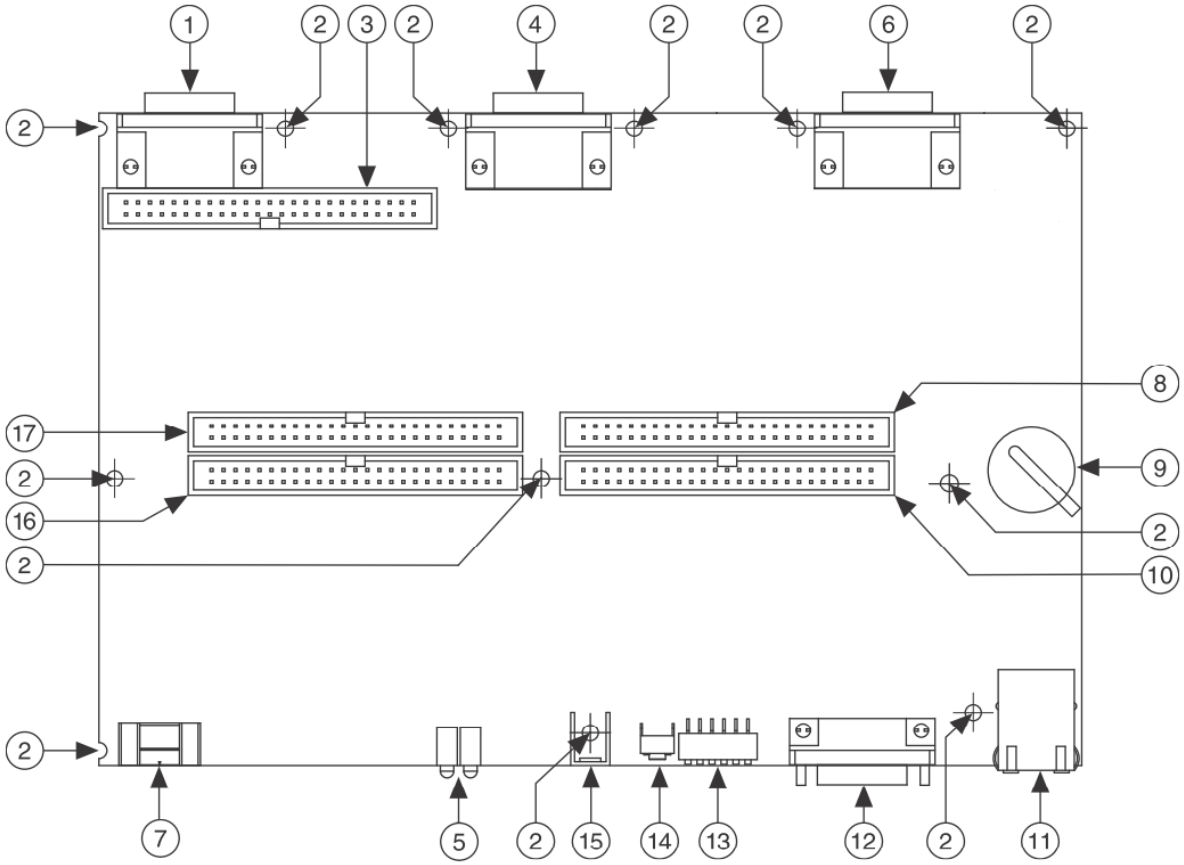
1.2 sbRIO 9631

Modul NI sbRIO 9631 je vestavné řídicí a zpracovávající zařízení, na kterém se nachází real-time procesor, FPGA (Field Programmable Gate Array – programovatelné hradlové pole = integrovaný obvod, určený pro nakonfigurování dle vlastní potřeby vývojáře) a analogové a digitální I/O. [3,4]

Specifikace:

- 266 MHz procesor,
- 1M hradel Xilinx FPGA,
- 110 3,3V (5V/TTL) digitálních I/O, 32 se společnou zemí a 16 diferenciálních 16 bitových analogových vstupů se vzorkováním 250 kS/s, 4 16 bitové analogové výstupy se vzorkováním 100 kS/s,
- 3 konektory pro rozšíření I/O použitím přídavných C modulů,
- provozní teplota od - 20 do 55 °C, vstupní napětí 19 V až 30 V,
- paměť 64 MB DRAM, 128 MB pro uložení programů a záznam dat,
- 10/100 Mbits/s Ethernet port, RS-232 sériový port.

Na obr. 3 je znázorněno rozložení jednotlivých konektorů na sbRIO desce.



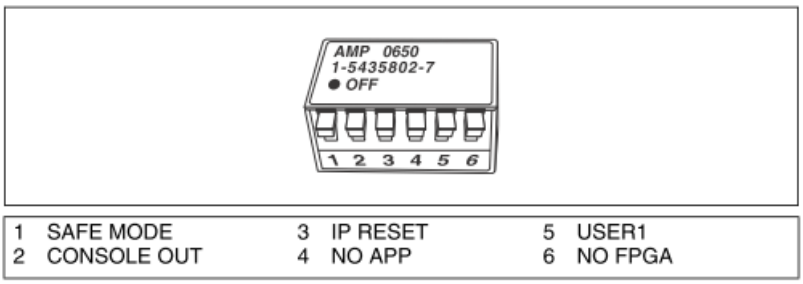
Obr. 3 : NI sbRIO 9631- rozložení konektorů.

Jednotlivé konektory (viz obr. 1):

1 J10, konektor pro přídavný C modul 3
2 Montážní otvory
3 J7, analogový I/O konektor
4 J9, konektor pro přídavný C modul 2
5 LED indikátory
6 J8, konektor pro přídavný C modul 1
7 J3, konektor pro nabíjení
8 P4, 3,3 V digitální I/O
9 Záložní baterie
10 P2, 3,3 V digitální I/O
11 J2, RJ-45 Ethernetový port
12 J1, RS-232 sériový port
13 DIP spínače
14 Reset tlačítko
15 P1, zemnicí svorka
16 P3, 3,3 V digitální I/O
17 P5, 3,3 V digitální I/O

1.2.1 DIP Spínače

Na obr. 4 jsou znázorněny jednotlivé spínače.



Obr. 4: DIP Spínače na modulu.

SAFE MOD Spínač

Na spínači SAFE MODE závisí, připojí-li se LabVIEW Real-Time při spuštění. Pro běžné operace by měl být v pozici OFF. Je-li spínač při spuštění v pozici ON, NI sbRIO se připojí pouze ke službám nezbytným pro update a instalaci softwaru, ale LabVIEW Real-Time se nespustí.

V případě, že je software na NI sbRIO poškozený, nebo je nutné přeformátovat disk, spínač musí být v pozici ON. Není-li na zařízení nainstalován žádný software, zařízení se automaticky zapne v safe módu, a to i v případě, že spínač je v pozici OFF.

CONSOLE OUT Spínač

Pomocí terminálové služby lze využít sériový port pro čtení IP adresy a verze firmwaru sbRIO zařízení. Propojení sériového portu zařízení s počítačem se realizuje pomocí kabelu nulového modemu. Terminálový program musí při komunikaci mít nastaveny tyto hodnoty:

- 9600 bitů za sekundu,
- 8 datových bitů,
- žádná parita,
- 1 stop bit,
- žádné řízení toku.

Při běžných operacích musí být CONSOLE OUT spínač v pozici OFF, jinak software LabVIEW Real-Time není schopen komunikovat se sériovým portem.

IP RESET Spínač

Dojde-li k restartování NI sbRIO zařízení zatímco bude spínač IP RESET v pozici ON, IP adresa se resetuje na hodnotu 0.0.0.0. Je-li zařízení připojené k lokální síti a současně je spínač v pozici ON, objeví se s takto nastavenou adresou i v programu pro správu zařízení (MAX), v němž je potom možné nakonfigurovat novou IP adresu.

NO APP Spínač

NO APP spínač v pozici ON zabraňuje LabVIEW Real-Time aplikacím jejich spuštění při startu. Permanentní zákaz spouštění je třeba nastavit v LabVIEW. Pro spuštění aplikace při startu je nutné přepnout spínač do pozice OFF, vytvořit aplikaci pomocí LabVIEW Application Builder a nakonfigurovat ji pro automatické připojování při startu.

USER1 Spínač

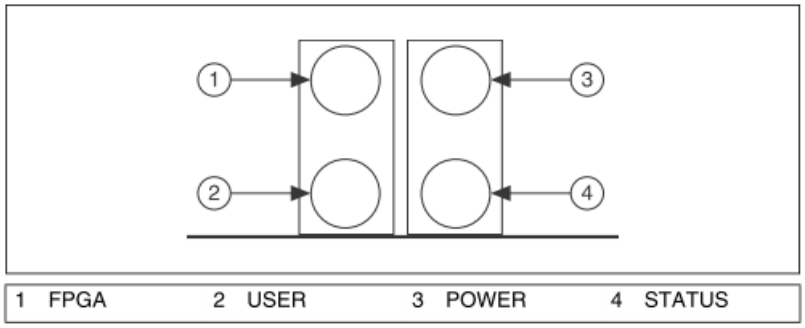
Spínač USER1 je možné nadefinovat pro vlastní aplikaci.

NO FPGA Spínač

Je-li NO FPGA spínač v pozici ON, zabraňuje načtení LabVIEW FPGA aplikací při spuštění.

1.2.2 LED Indikátory

Rozložení jednotlivých LED indikátorů je znázorněno na obr. 5.



Obr. 5: LED Indikátory na modulu.

FPGA LED

FPGA LED slouží k doladění aplikací nebo snadnému získání informace o jejich statusu. Může být nadefinována tak, aby její činnost odpovídala požadavkům jakékoliv uživatelské aplikace. K jejímu nakonfigurování je potřeba užít LabVIEW FPGA Module a NI-RIO software.

USER LED

User LED může být nadefinována tak, aby její činnost odpovídala požadavkům jakékoliv uživatelské aplikace. K jejímu nakonfigurování slouží RT LEDs VI v LabVIEW.

POWER LED

Power LED svítí vždy, když je zařízení zapnuto. Indikuje tak, že napětí 5 V a 3,3 V jsou stabilní.

STATUS LED

Během běžných operací je status LED vypnutá. Specifickým počtem bliknutí indikuje chybové stavy (viz tab. 1).

1 (jedno bliknutí každých pár sekund)	Zařízení není nakonfigurováno.
2	Zařízení zjistilo chybu v softwaru. Tato chyba vzniká převážně v situacích, kdy dojde k přerušení upgradu softwaru. Je nutné přeinstalovat software zařízení.
3	Spínač SAFE MODE je v pozici ON a zařízení je tedy v safe módu.
4	Software zařízení dvakrát zkolaboval, přičemž mezitím nedošlo k jeho restartování. Tento problém se většinou objeví, dojde-li k vyčerpání velikosti paměti. Je nutné znovu prohlédnout RT VI a zkontrolovat využití paměti zařízení.
Neustálé blikání nebo svícení.	Zařízení zjistilo neopravitelnou chybu. Je nutné zformátovat hard disk zařízení. Pokud problém přetrvává, je potřeba kontaktovat National Instruments.

Tab. 1: Indikace chybových stavů.

1.3 Ultrazvukový senzor vzdálenosti Parallax PING)))

Ultrazvukový senzor vzdálenosti Parallax PING)))™ poskytuje přesné a bezkontaktní měření vzdálenosti od zhruba 2 cm do 3 m. Dá se velmi jednoduše připojit k mikrokontrolérům, jako jsou např. BASIC Stamp ®, Propeller chip, nebo Arduino, pomocí pouze jednoho I/O pinu. [5]

1.3.1 Vlastnosti

- Dosah: 2 cm až 3 m,
- LED dioda slouží k indikaci aktivity senzoru,
- obousměrné TTL rozhraní na jediném I/O pinu může komunikovat s 5V TTL nebo 3,3V CMOS mikrokontroléry,
- vstupní trigger: kladný TTL impulz, 2 µs min, 5 µs typ.,
- echo pulz: kladný TTL impulz, 115µs min až 18,5 ms max,
- v souladu se směrnicí RoHS.

1.3.2 Hlavní parametry

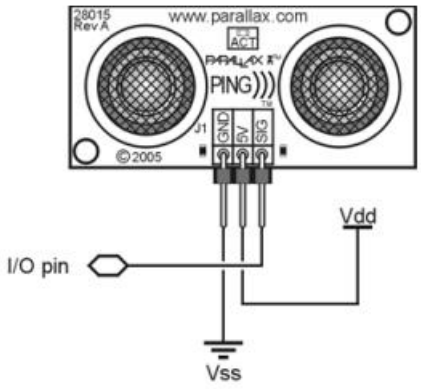
- Napájecí napětí: +5 V,
- napájecí proud: 30 mA; 35 mA max.,
- komunikace: kladný TTL impulz,

- připojení: 3pinový konektor,
- provozní teplota: 0 – 70° C,
- rozměry: 22 mm V x 46 mm Š x 16 mm H.
- hmotnost: 9 g.

1.3.3 Specifikace pinů

- GND – Země (Vss),
- 5 V (Vdd),
- SIG – Signál (I/O pin).

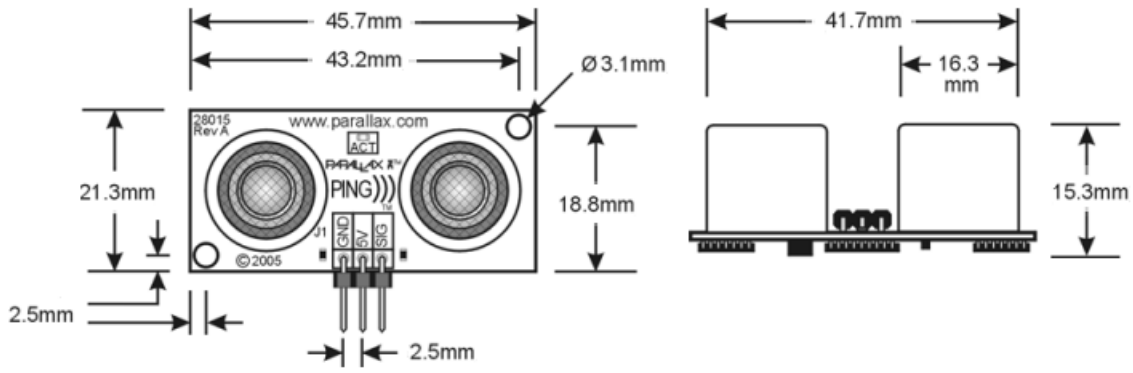
PING))) senzor má 3pinový konektor, který slouží pro připojení napájení, země a signálu uzemnění (viz obr. 6).



Obr. 6: Konektor PING))) senzoru.

1.3.4 Rozměry

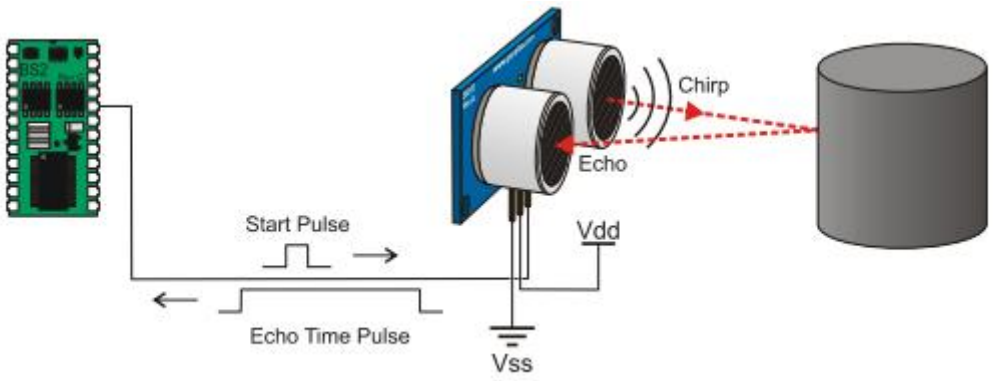
Rozměry senzoru jsou zobrazeny na obr. 7.



Obr. 7: Rozměry PING))) senzoru.

1.3.5 Komunikační protokol

PING))) senzor detekuje objekty tak, že vyšle krátký 40 kHz ultrazvukový signál (pro člověka vysoko nad pásmem slyšitelnosti) a současně začne generovat impulz. Vyslaný signál se šíří vzduchem a jakmile narazí do objektu, odrazí se zpět k senzoru. Celková doba od vyslání signálu až po obdržení jeho odrazu potom odpovídá šířce výstupního impulzu ze senzoru. Změřením této šířky pak lze vzdálenost objektu od senzoru jednoduše dopočítat. Princip komunikace je znázorněn na obr. 8.

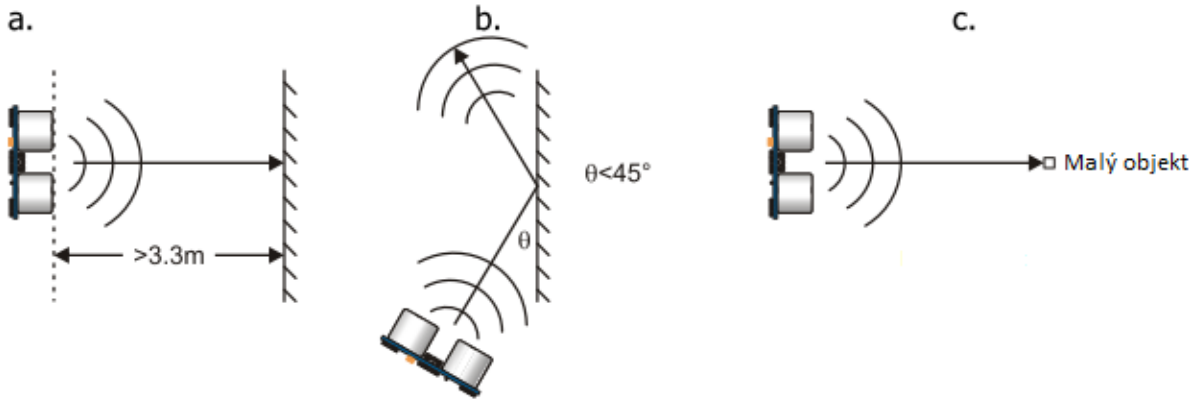


Obr. 8: Princip PING))) senzoru.

1.3.6 Praktická doporučení pro užívání

Určování polohy objektů

PING))) senzor nedokáže přesně měřit vzdálenost objektů jestliže: a) jsou dál než 3 m, b) mají reflexivní povrch a vyslaný paprsek se tak nemůže odrazit zpět k senzoru (platí zejména pro signály vyslané v příliš ostrém úhlu), c) jsou příliš malé na to, aby odrazily dostatečnou úroveň ultrazvukového signálu zpět k senzoru. Graficky jsou všechny situace znázorněny na obr. 9. Jestliže je navíc senzor umístěn na zařízení příliš nízko, může senzor detekovat signál odrážející se od podlahy.



Obr. 9: PING))) senzor v praxi.

Materiál objektů

Objekty, které absorbují ultrazvuk nebo mají měkký či nepravidelný povrch, jako například plyšová zvířata, nemusí odrážet dostatek ultrazvuku pro přesnou detekci. PING))) senzor je schopen detekovat povrch vody, nicméně není určen pro venkovní použití nebo kontinuální použití ve vlhkém prostředí. Srážení vody na jeho převodnících může ovlivnit jeho výkon a životnost.

Teplota vzduchu

Teplota vzduchu ovlivňuje rychlost ultrazvuku. Jestliže je teplota známa, rychlost může být jednoduše dopočítána ze vztahu:

$$c_{\text{air}} = 331{,}5 + (0{,}6 \times T_{\text{C}})\ m/s,$$

kde c_{air} je rychlost signálu a T_C teplota vzduchu.

Je-li tedy např. teplota vzduchu T_{C} = 24,6 °C, ultrazvukový signál se bude prostředím šířit rychlostí c_{air} = 346,26 *m/s*. Při vyšší teplotě, např. T_{C} = 40 °C, bude rychlost signálu odpovídat hodnotě c_{air} = 355,5 *m/s*.

2. Software

2.1 LabVIEW

LabVIEW (tedy zkratka pro Laboratory Virtual Instrument Engineering Workbench) je vývojové prostředí, které prostřednictvím grafického programování umožňuje vytváření různých aplikací. Bylo vyvinuto firmou National Instruments.

Toto grafické programování, také nazývané jako „G“ je založeno na takzvaném programování pomocí datového toku. Kód v LabVIEW je uspořádán do blokového schématu, jehož funkční části jsou propojeny pomocí vodičů. Tyto vodiče dále rozvádí proměnné se zde vyskytující do jednotlivých uzlů programu, přičemž každý z těchto uzlů se může uskutečnit až v okamžiku, kdy jsou na jeho vstupu dostupná všechna data. Jednotlivé bloky takto vytvořených programů tedy probíhají postupně, ve specifickém pořadí, což je princip programovacích jazyků datového toku.

Programy vytvářené v LabVIEW se nazývají virtuální instrumenty (v originále Virtual Instruments, dále jen VI). Skládají se ze tří částí: blokový diagram, čelní panel a konektorový panel. Prostřednictvím konektorového panelu, jsou VI reprezentovány v blokovém diagramu jiných VI. Na čelním panelu se nachází ovládací prvky a indikátory. Ovládací prvky slouží jako vstupy – umožňují uživateli zadat hodnoty, které se následně pošlou do VI. Indikátory naopak slouží jako výstupy – zobrazují výsledky z VI. Blokový diagram obsahuje zdrojový kód programu. Nachází se zde struktury a funkce, které uskutečňují operace v závislosti na hodnotách přivedených z ovládacích prvků a naopak dodávají výsledné hodnoty do indikátorů. [6,7]

LabVIEW FPGA Modul je doplnkový software pro FPGA čipy na NI nekonfigurovatelných I/O (RIO) hardwarech. [8]

LabVIEW Real-Time Modul je doplňkový software pro LabVIEW, který slouží k vytváření spolehlivých deterministických aplikací. [9]

LabVIEW Robotics Modul je softwarový balíček pro LabVIEW, který umožňuje vyvíjet aplikace robotiky. Obsahuje:

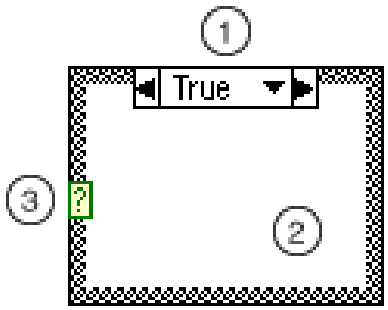
- VI a přístrojové ovladače – Robotics Modul obsahuje algoritmy pro navigaci, řízení a mnohé další. Dále poskytuje ovladače pro senzory a různé akční členy.
- Setup Wizards
    - Hardware Setup Wizard – provádí konfiguraci základních nastavení NI RIO hardwaru a instaluje na ně potřebný software.
    - Robotics Project Wizard – umožňuje vytvoření nového LabVIEW projektu.

- Simulátor – vestavěný robotický simulátor, který umožňuje vykreslení 3D simulací a modelů reálných robotů v reálném prostředí.
- Ukázky VI. [10]

2.1.1 Struktury a smyčky

2.1.1.1 Case Structure

Slouží k větvení algoritmu (viz obr. 10).



Obr. 10: Case struktura.

Skládá se z jednoho nebo více rámců s kódem, přičemž může probíhat vždy pouze jeden. Hodnota přivedená na výběrový terminál udává, který z rámců se provede.

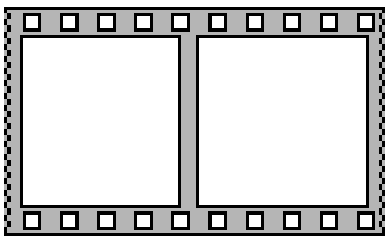
1. Štítek názvu rámce (case selector label) – zobrazuje hodnotu, pro kterou se daný rámec spustí. Je možné zadat jednu hodnotu, nebo interval hodnot. Také slouží pro definování přednastaveného rámce.

2. Rámec (case) - obsahuje kód, který se provede, jestliže se hodnota připojená na výběrový terminál shoduje s hodnotou ve štítku názvu rámce.

3. Výběrový terminál (selector terminal) – na základě hodnoty vstupních dat určuje, který rámec se vykoná. Vstupní data mohou být logické hodnoty, řetězce, celá čísla, výčtového typu nebo chybové klastry.

2.1.1.2 Flat Sequence

Slouží k jednoznačnému zavedení postupného provádění kódu (viz obr. 11).



Obr. 11: Flat sekvence.

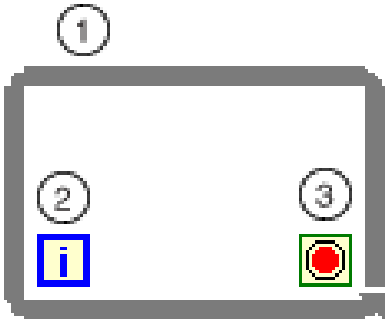
Skládá se z jednoho nebo více rámců, které se provádějí postupně. Používá se tam, kde je potřeba vykonat rámce požadovaném pořadí.

Datový tok flat sekvence se liší od datových toků ostatních struktur. Rámce ve flat sekvenci probíhají postupně, tedy z leva doprava a pouze v případě, že jsou všechna data připojená k rámci dostupná. Ukončí-li jeden rámec svoji činnost, všechna data přecházejí do druhého. To znamená, že vstup jednoho může záviset na výstupu dalšího.

Přidají-li se nebo se odeberou rámce do flat sekvence, struktura se automaticky sama přizpůsobí.

2.1.1.3 While Loop

Slouží k zavedení opakovaného provádění kódu, který je uvnitř smyčky (viz obr. 12).



Obr. 12: While smyčka.

Opakuje kód do doby, než je splněna ukončovací podmínka. Vždy proběhne alespoň jednou.

1. Rámec (subdiagram) – obsahuje kód, který probíhá při každém opakování.

2. Iterační terminál (iteration terminal) – udává současnou hodnotu opakování. Po prvním průběhu vždy začíná od nuly. Jestliže počet opakování přesáhne počet 2 147 483 647 (2^{31}-1), iterační terminál zůstane na hodnotě 2 147 483 647 pro všechna další opakování. Vyššího počtu opakování lze dosáhnout použitím tzv. shift registrů s větším číselným rozsahem.

3. Terminál ukončovací podmínky (conditional terminal) – vyhodnocuje hodnotu logického vstupu a následně rozhoduje, zda pokračovat ve vykonávání smyčky nebo ji ukončit. Má-li se smyčka ukončit při hodnotě TRUE nebo FALSE lze jednoduše nakonfigurovat.

2.1.2 Robotics VIs

2.1.2.1 Initialize Starter Kit 1.0 (sbRIO-9631) VI

Slouží k inicializaci zařízení (viz obr. 13).

Obr. 13: Initialize Starter Kit 1.0 (sbRIO-9631) VI.

Zahajuje komunikační relaci mezi FPGA na robotovi a vrací referenci potřebnou pro čtení nebo zápis na FPGA dalším následným funkcím. Toto VI musí být voláno před přístupem na I/O na FPGA.

IP address – určuje IP adresu sbRIO kontroléru připojeného k počítači. Jako výchozí nastavení toto VI používá IP adresu pro sbRIO definovanou v Projektovém Exploreru.

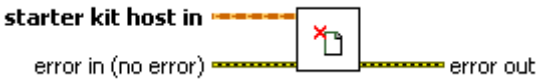
Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Tento vstup poskytuje standardní funkci error in.

FPGA starter kit host 1.0 out – je odkazem na komunikační relaci mezi VI a FPGA na robotovi. Tento výstup se zapojuje do jiných Starter Kit VI.

Error out – obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

2.1.2.2 Close Starter Kit VI

Slouží k ukončení komunikace se zařízením (viz obr. 14).



Obr. 14: Close Starter Kit VI.

Ukončí komunikační relaci s FPGA na skutečném nebo simulovaném Starter Kit robotovi. Jakmile dojde k ukončení komunikace, hnací motory, senzor vzdálenosti a servo motory zastaví svůj provoz.

Starter kit host in – je odkazem na komunikační relaci mezi VI a FPGA na robotovi.

Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Až na následující výjimku tento vstup poskytuje standardní funkci error in.

Uzel poběží normálně i v případě, že se chybový stav objeví před jeho započetím.

Error out - obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

2.1.2.3 Read PING))) Senzor Distance VI

Funkce je zobrazena na obr. 15.

Obr. 15: Read PING))) Senzor Distance VI.

Funkce přečte a vrátí hodnotu vzdálenosti mezi PING))) ultrazvukovým senzorem a nejbližší překážkou, kterou senzor detekuje.

Pokud senzor nedetekuje žádnou překážku, výstup timed out? vrací TRUE.

Starter kit host in - je odkazem na komunikační relaci mezi VI a FPGA na robotovi.

Deafault distance – udává hodnotu vzdálenosti, kterou toto VI vrací, pokud senzor nedetekuje žádnou překážku. To umožňuje nastavit kód tak, aby běžel, nejsou-li v dosahu robota žádné překážky. Výchozí hodnota je NaN (Not a Number).

Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Tento vstup poskytuje standardní funkci error in.

Starter kit host out - je odkazem na komunikační relaci mezi VI a FPGA na robotovi. Tento výstup zapojujeme do jiných Starter Kit VI.

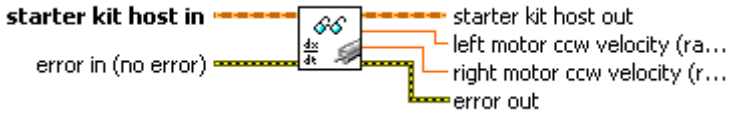
Distance – vrací hodnotu vzdálenost v metrech mezi senzorem a nejbližší překážkou, pokud senzor nějakou detekuje. Jinak vrací hodnotu připojenou na vstup default distance.

Timed out? – TRUE značí, že senzor nedetekoval žádnou překážku. Jestliže je timed out? TRUE, výstup distance vrací hodnotu připojenou na vstup default distance.

Error out - obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

2.1.2.4 Read DC Motor Velocities VI

Funkce je zobrazena na obr. 16.



Obr. 16: Read DC Motor Velocities VI.

Přečte a vrátí rychlost levého a pravého hnacího motoru na Starter Kit robotovi. Rychlost vrací proti směru hodinových ručiček. Pozici levého a pravého motoru je definována při pohledu na robota zezadu.

Starter kit host in - je odkazem na komunikační relaci mezi VI a FPGA na robotovi.

Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Tento vstup poskytuje standardní funkci error in.

Starter kit host out - je odkazem na komunikační relaci mezi VI a FPGA na robotovi. Tento výstup zapojujeme do jiných Starter Kit VI.

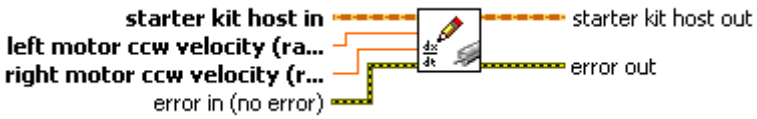
Left motor ccw velocity – vrací rychlost, v radiánech za sekundu, jakou se pohybuje levý motor. Hodnota, kterou toto VI vrací je ta, která otáčí levými koly proti směru hodinových ručiček (při přímém pohledu na levá kola). Takže je-li hodnota kladná, robot jede dopředu, je-li záporná dozadu.

Right motor ccw velocity – vrací rychlost, v radiánech za sekundu, jakou se pohybuje pravý motor. Hodnota, kterou toto VI vrací je ta, která otáčí pravými koly proti směru hodinových ručiček (při přímém pohledu na pravá kola). Takže je-li hodnota záporná, robot jede dopředu, je-li kladná dozadu.

Error out - obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

2.1.2.5 Write DC Motor Velocity Setpoints VI

Funkce je zobrazena na obr. 17.



Obr. 17: Write DC Motor Velocity Setpoints VI.

Aplikuje hodnoty rychlosti na hnací motory na Starter Kit robotovi. Pozice levého a pravého motoru je definována při pohledu na robota zezadu.

Starter kit host in - je odkazem na komunikační relaci mezi VI a FPGA na robotovi.

Left motor ccw velocity – udává hodnotu rychlosti, v radiánech za sekundu, kterou požadujeme na levém motoru. Nastaví-li se větší rychlost než je hodnota maximální rychlosti, motor se bude pohybovat touto maximální rychlostí. Rychlost, kterou nastavíme, bude otáčet levými koly proti směru hodinových ručiček (při přímém pohledu na levá kola). Takže je-li hodnota kladná, robot jede dopředu, je-li záporná dozadu.

Right motor ccw velocity (rad/s) – udává hodnotu rychlosti v radiánech za sekundu, kterou požadujeme na pravém motoru. Nastaví-li se větší rychlost než je hodnota maximální rychlosti, motor se bude pohybovat touto maximální rychlostí. Rychlost, kterou nastavíme, bude otáčet pravými koly v proti směru hodinových (při přímém pohledu na pravá kola). Takže je-li hodnota záporná, robot jede dopředu, je-li kladná dozadu.

Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Tento vstup poskytuje standardní funkci error in.

Starter kit host out - je odkazem na komunikační relaci mezi VI a FPGA na robotovi. Tento výstup zapojujeme do jiných Starter Kit VI.

Error out - obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

2.1.2.6 Read Sensor Servo Offset VI

Funkce je zobrazena na obr. 18.

Obr. 18: Read Senzor Servo Offset VI.

Přečte a vrátí odchylku, o kterou se nulová neboli přímá pozice ultrazvukového senzoru vzdálenosti liší od nulové pozice servo motorku.

Starter kit host in - je odkazem na komunikační relaci mezi VI a FPGA na robotovi.

Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Tento vstup poskytuje standardní funkci error in.

Starter kit host out - je odkazem na komunikační relaci mezi VI a FPGA na robotovi. Tento výstup zapojujeme do jiných Starter Kit VI.

Ccw offset angle – vrací úhel, v radiánech, o který se nulová neboli přímá pozice ultrazvukového senzoru vzdálenosti liší od nulové pozice serva. Úhel je měřen proti směru hodinových ručiček od středu robota, díváme-li se na něj shora. Takže kladné hodnoty jsou vlevo, záporné vpravo od středu.

Error out - obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

2.1.2.7 Write Sensor Servo Angle VI

Funkce je zobrazena na obr. 19.

Obr. 19: Write Senzor Servo Angel VI.

Zapíše hodnotu úhlu do serva, na kterém je umístěn senzor vzdálenosti, pro otočení serva na tento úhel. Úhel serva se liší od úhlu senzoru v případě, že servo a snímač nejsou srovnány.

Maximální rozsah otočení serva je ±π/2 nebo ±90°.

Starter kit host in - je odkazem na komunikační relaci mezi VI a FPGA na robotovi.
Ccw servo angle – určuje úhel, v radiánech, do něhož se má otočit servo. Úhel je měřen proti směru hodinových ručiček od středu robota, díváme-li se na něj shora. Takže kladné hodnoty otáčí servem doleva, záporné hodnoty doprava.
Error in – popisuje chybové stavy, které se mohou objevit před spuštěním uzlu. Tento vstup poskytuje standardní funkci error in.
Starter kit host out - je odkazem na komunikační relaci mezi VI a FPGA na robotovi. Tento výstup zapojujeme do jiných Starter Kit VI.
Error out - obsahuje informaci o chybě, jestliže se nějaká objeví. Tento výstup poskytuje standardní funkci error out.

3. Realizace práce

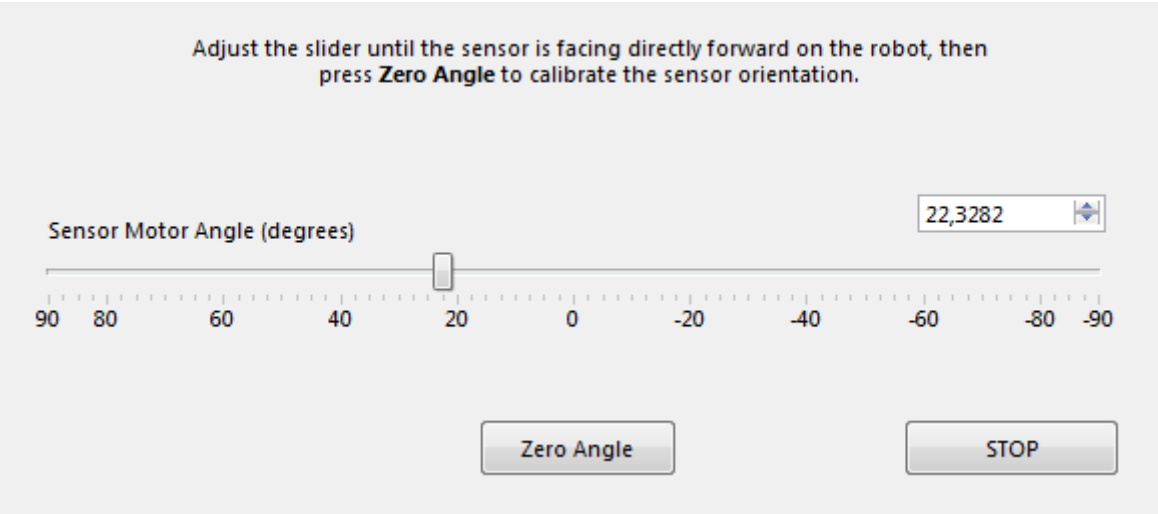
V následující kapitole jsou popsány veškeré aplikace, pomocí nichž je možné robota ovládat. První tři z nich, tedy 3.1 Calibrate Sensor Angle, 3.2 Test Ultrasonic Sensor a 3.3 Test Motors, byly vyvinuty již firmou National Instruments a slouží pouze ke správnému nastavení ultrazvukového senzoru a motorů.

3.1 Calibrate Sensor Angle

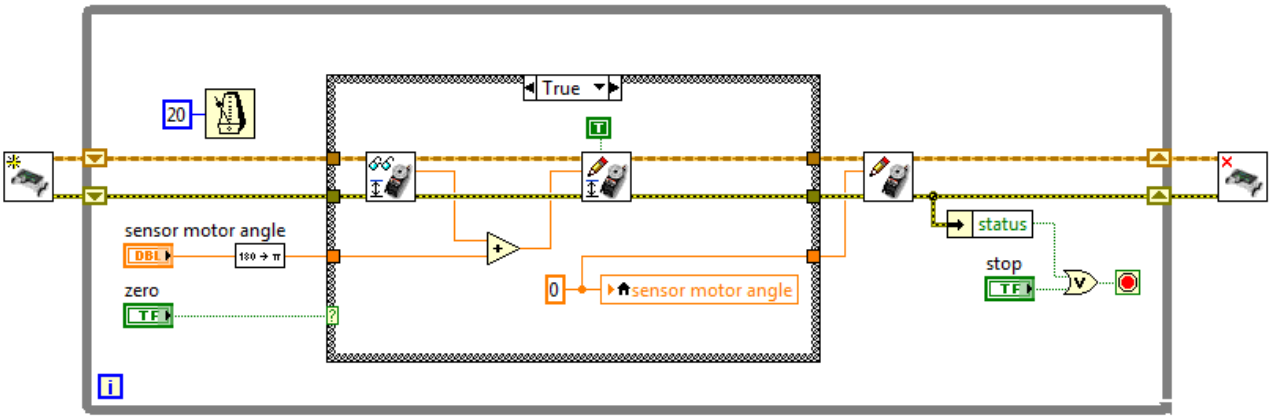
Hlavním účelem tohoto programu je kalibrace úhlu natočení servomotoru se senzorem.

Při spuštění se zahájí komunikace s FPGA na modulu pomocí funkce Initialize Starter Kit 1.0 (sbRIO-9631). Na čelním panelu se nachází posuvný ovladač, pomocí něhož může uživatel úhel senzoru měnit. Je-li podle něj správně nastaven, tedy míří-li senzor rovně před robota, může stisknout tlačítko Zero Angle. Stisknutím tlačítka se ve zdrojovém kódu provede obsah case struktury pro možnost true. Na vstup ccw offset angle funkce Write Sensor Servo Offset se zapíše součet hodnoty úhlu nastavené uživatelem a původní hodnoty ccw offset angle funkce Read Sensor Servo Ofsset. Poté dojde k vynulování hodnoty posuvníku a vstupu ccw servo angle funkce Write Sensor Servo Angle. Zpoždění mezi jednotlivými cykly programu je realizováno funkcí Wait Until Next ms Multiple. Ukončení programu se provádí buď stisknutím tlačítka STOP, nebo automaticky objeví-li se nějaká specifická chyba. Komunikaci s robotem ukončuje funkce Close Starter Kit.

Na obrázcích 20 a 21 se nachází čelní panel a blokový diagram programu.



Obr. 20 : Čelní panel programu Calibrate Sensor Angle.



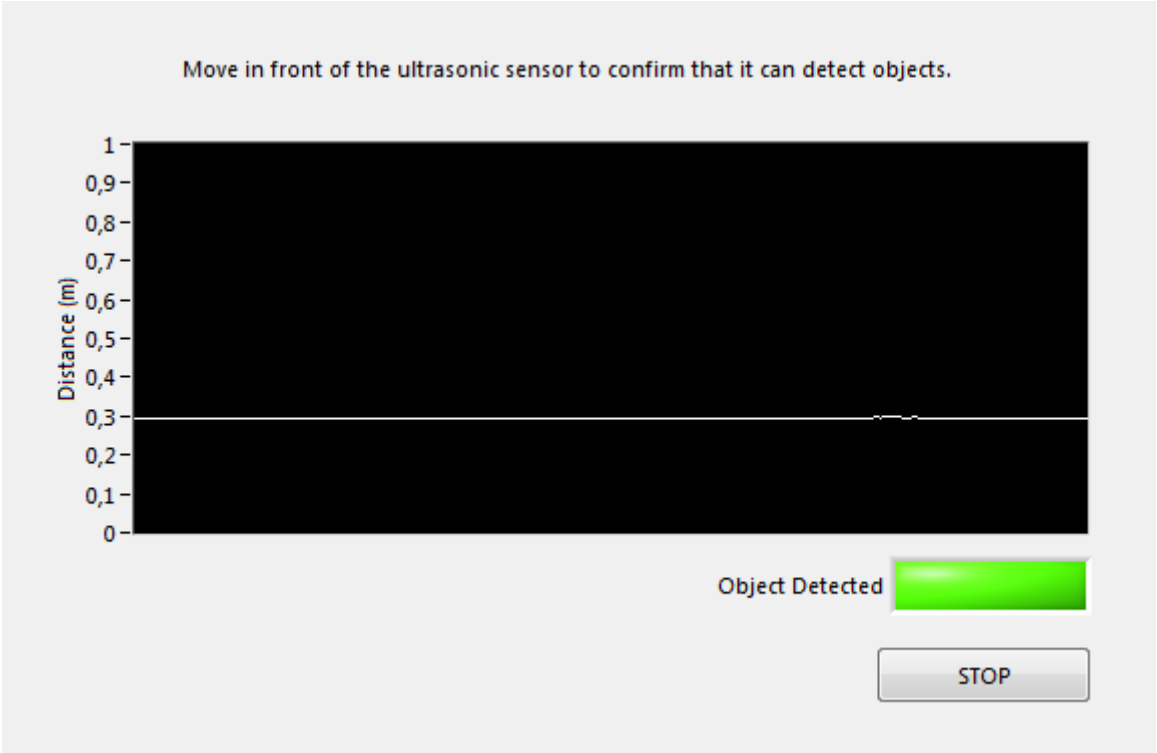
Obr. 21: Blokový diagram programu Calibrate Sensor Angle.

3.2 Test Ultrasonic Sensor

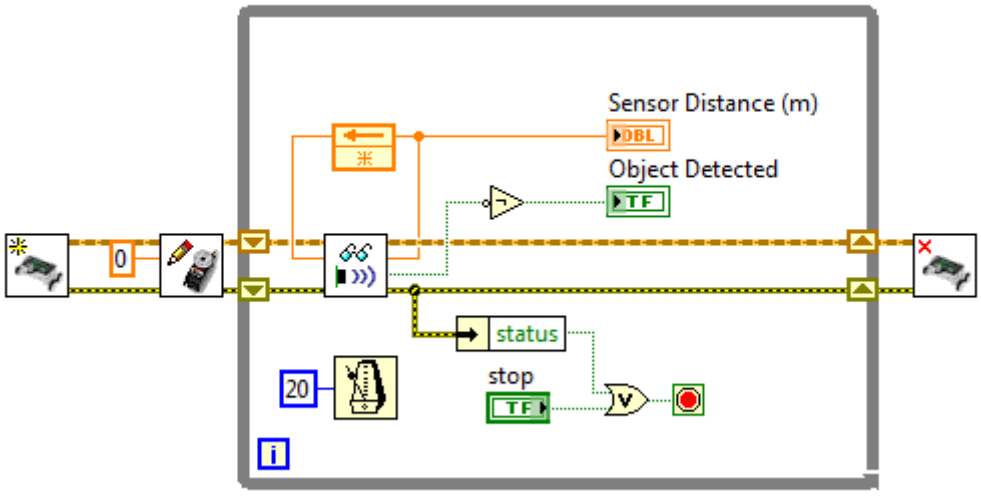
Tento jednoduchý program slouží k otestování správné funkčnosti ultrazvukového senzoru vzdálenosti.

Spustí-li se program, funkce Initialize Starter Kit 1.0 (sbRIO-9631) ihned zahájí komunikaci s FPGA. Přivedení nuly na vstup ccw servo angle funkce Write Sensor Servo Angle zajišťuje vynulování úhlu senzoru a tedy jeho přímý pohled rovně před robota. Ve while smyčce dochází se zavedeným zpožděním, které obstarává funkce Wait Until Next Ms Multiple, k opakovanému čtení hodnoty vzdálenosti ze senzoru funkcí Read PING))) Senzor Distance. Hodnota je vyvedena z funkce výstupem distance do grafu, jehož průběh lze sledovat na čelním panelu, a přes zpětnovazebný uzel také zpět do vstupu default distance. Zpětnovazebný uzel slouží k uchování hodnot z předchozích cyklů. Detekuje-li navíc robot nějakou překážku, rozsvítí se LED dioda na čelním panelu. Program se ukončí buď stisknutím tlačítka STOP, nebo při výskytu nějaké konkrétní chyby. Funkce Close Starter Kit poté přeruší komunikaci s FPGA.

Na obrázcích 22 a 23 se nachází čelní panel a blokový diagram programu.



Obr. 22: Čelní panel programu Test Ultrasonic Sensor.



Obr. 23 : Blokový diagram programu Test Ultrasonic Sensor.

3.3 Test Motors

Cílem tohoto programu je otestovat, pracují-li motory správně.

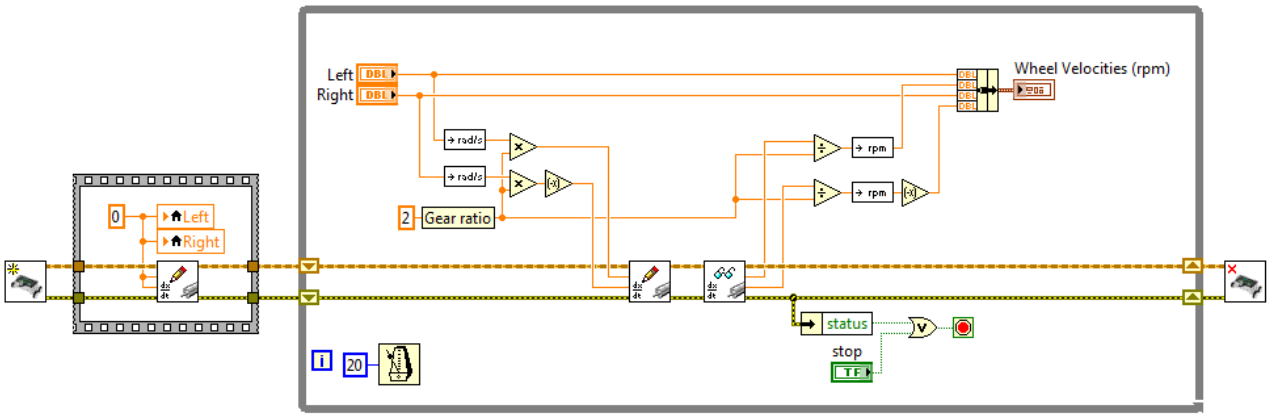
Po spuštění programu funkce Initialize Starter Kit 1.0 započne komunikační relaci s robotem. Ve flat sekvenci dochází k vynulování hodnot rychlostí motorů na vstupech funkce Write DC Motor Velocity Setpoints a hodnot posuvníků, jimiž uživatel nastavuje rychlost motorů

robota. To slouží jako opatření proti nechtěnému pohybu robota. Jestliže se totiž program ukončí, zůstanou hodnoty rychlostí stále zapsané na posuvných ovladačích a na vstupech funkce. Při opětovném zapnutí programu by se tak robot ihned rozjel. Proto je nutné zajistit, aby při každém spuštění došlo k přepsání stávajících hodnot rychlostí motorů na nulu.

Nastaví-li uživatel prostřednictvím posuvníků na čelním panelu hodnoty rychlostí, zapíšou se v blokovém diagramu na vstupy left motor ccw velocity a right motor ccw velocity téže funkce, nacházející se však ve while smyčce. Následující funkce Read DC Motor Velocities tyto hodnoty přečte a z výstupů left motor ccw velocity a right motor ccw velocity je vede do funkce Bundle, do níž zároveň vstupují hodnoty z posuvníků. Tato funkce vytvoří z jednotlivých prvků do ní vstupujících klastr hodnot. Ten je poté vyveden do grafu, který zobrazí data v grafické podobě. Program se ukončuje stisknutím tlačítka Stop nebo automaticky při jakémkoliv výskytu chyby. Funkce Close Starter Kit poté ukončí komunikaci s robotem.

Na obrázcích 24 a 25 se nachází čelní panel a blokový diagram programu.

Obr. 24: čelní panel programu Test Motors.



Obr. 25: Blokový diagram programu Test Motors.

3.4 Start

Program slouží k jednoduchému ovládání robota, jeho rychlosti přímého pohybu a ultrazvukového senzoru. Naměřená data ze senzoru se zobrazují v grafu.

Na počátku funkce Initialize Starter Kit 1.0 (sbRIO-9631) zahájí komunikaci s FPGA. V následující a jediné flat sekvenci, obsahující funkci Write DC Motor Velocity Setpoints VI, dochází při každém spuštění programu k vynulování hodnot rychlostí obou motorů.

Ve vnitřní while smyčce se nachází funkce Write Sensor Servo Angle VI, která uživateli prostřednictvím posuvného ovladače na čelním panelu umožňuje nastavit úhel natočení ultrazvukového senzoru vzdálenosti. Bude-li uživatel chtít tento úhel opět vynulovat, nemusí tak činit pracným hledáním nulové polohy posuvníku, stačí mu pouze kliknout na tlačítko Zero. To v blokovém diagramu ovládá case strukturu, a je-li její hodnota true, tedy je-li stisknuto tlačítko, dojde k vynulování vstupní hodnoty funkce a tedy k vynulování úhlu.

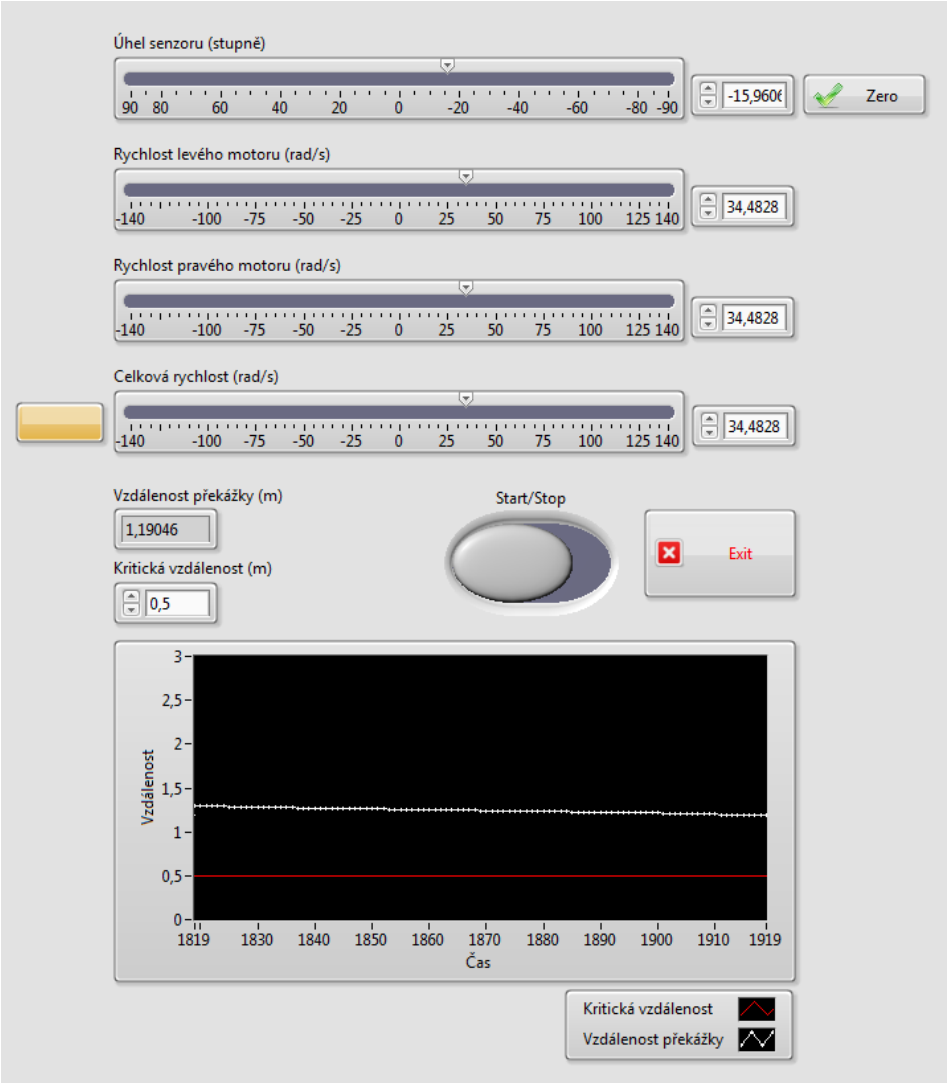
Vnitřní while smyčka také obsahuje samotný kód pro ovládání pohybu robota. Dále se zde objevuje filtr nízkých naměřených hodnot vzdáleností ze senzoru. Filtr bylo nutné přidat z důvodu občasných výskytů nuly na výstupu senzoru, které díky ukončovací podmínce způsobovaly nechtěné ukončení programu (viz dále). Filtr je sestaven z case struktury a shift registru. Jeho princip je následující. Na výběrovém terminálu se porovnává příchozí hodnota ze senzoru s pevnou hodnotou 0,04 m. Je-li příchozí hodnota větší než tato pevná, probíhá rámec false, a tedy hodnota ze senzoru prochází dále. Je-li ovšem menší, nahrazuje se předchozí hodnotou z shift registru a opět pokračuje dál. Takto zpracované hodnoty vzdálenosti může uživatel sledovat číselně na indikátoru a graficky vykreslené v grafu na čelním panelu.

Zpracovaná hodnota vzdálenosti slouží dále i jako srovnávací hodnota v již dříve zmíněné ukončovací podmínce. Ta je opět tvořena case strukturou, na jejímž výběrovém terminálu se porovnává rychlost robota s nulou. Je-li rychlost menší než nula, a tedy robot couvá, neděje se nic (případnému střetu robota s překážkou musí zabránit sám uživatel, jelikož servomotor se senzorem vzdálenosti má rozsah otočení pouze 180°). Jede-li však robot dopředu, tedy je-li rychlost kladná, probíhá smyčka false a srovnává se hodnota vzdálenosti s nastavitelnou minimální hodnotou. Jestliže je potom změřená vzdálenost menší nebo rovna této, hrozí náraz robota do překážky. Hodnoty rychlostí motorů se přepíšou na nulu, rozsvítí se FPGA LED na modulu a program se zastaví. Přerušit průběh programu ale může uživatel i ručně, a to přepnutím tlačítka Start/Stop na čelním panelu do pozice Stop. Pro opětovné spuštění programu je v obou případech nutné přepnout tlačítko zpět do pozice Start.

Výsledek rozhodnutí nachází-li se robot v minimální vzdálenosti před překážkou je dále přiveden na vstup další case struktury, v níž se nachází již samotné ovládání rychlostí motorů. Pomocí posuvníku na čelním panelu uživatel nastavuje tyto hodnoty. Stisknutím žlutého tlačítka se navíc může rozhodnout, chce-li ovládat oba motory zároveň, nebo každý zvlášť.

V programu se nachází i funkce Wait Until Next Ms Multiple, která se stará o zpoždění mezi jednotlivými průběhy kódu. Program se ukončuje stisknutím tlačítka Exit na čelním panelu, případně automaticky, vyskytne-li se chyba. Ukončení komunikace s FPGA na modulu zajišťuje funkce Close Starter Kit VI.

Čelní panel programu je vyobrazen na obr. 26, blokový diagram je obsažen v příloze 1.



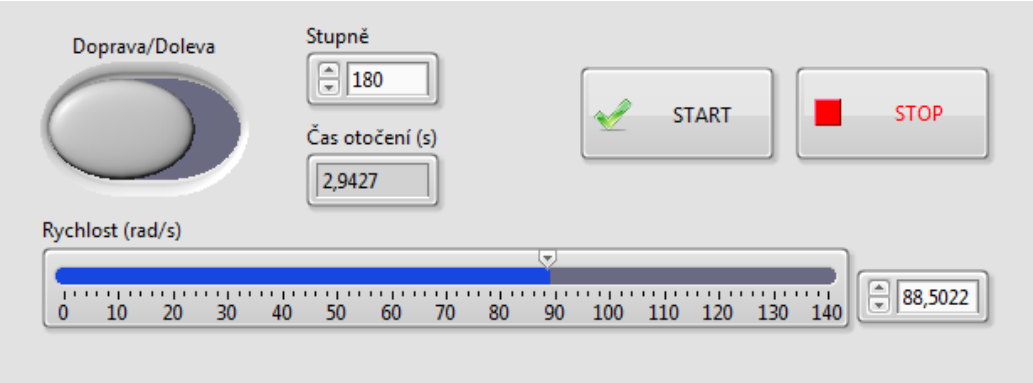
Obr. 26: Čelní panel programu Start.

3.5 Otočení

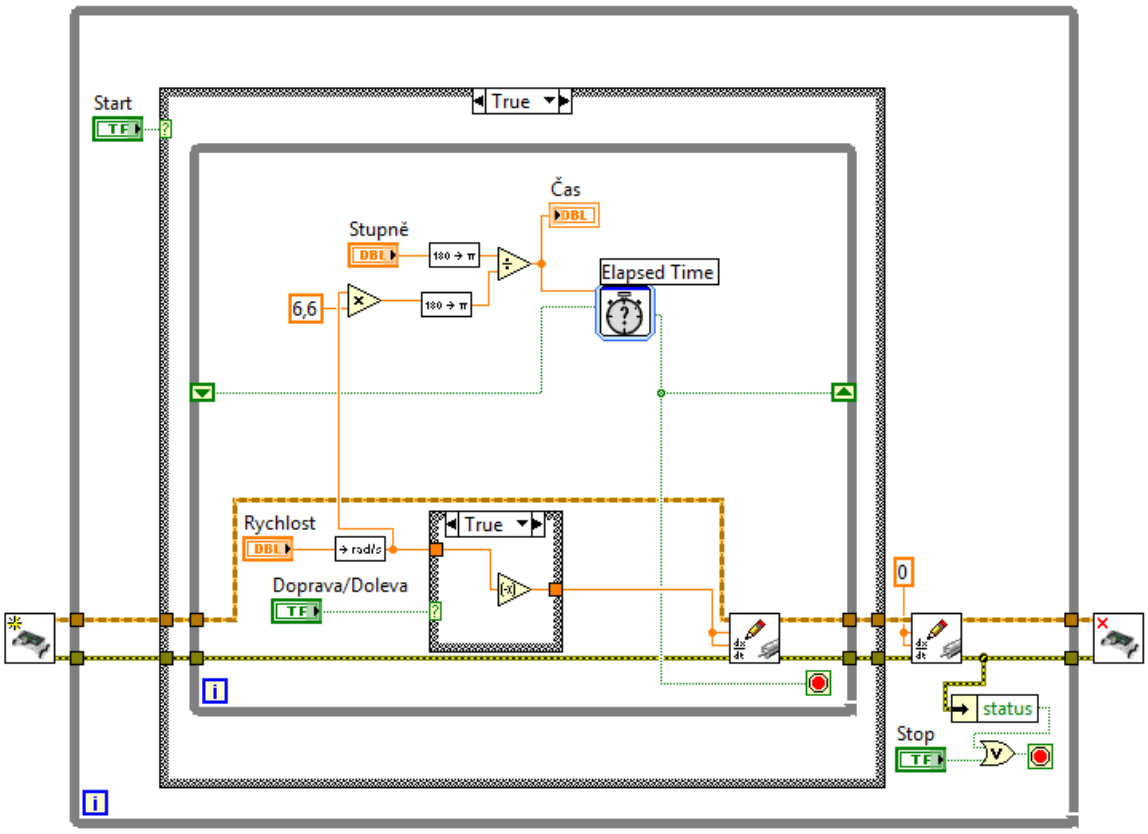
Program slouží k otočení robota. Uživatel má možnost nastavit jakou rychlostí a o jaký úhel se robot má otočit.

Komunikaci při spuštění programu opět navazuje funkce Initialize Starter Kit 1.0 (sbRIO-9631). Samotné otočení se spouští tlačítkem Start, jehož hodnota je v blokovém diagramu přivedena na výběrový terminál vnější case struktury. Po jeho stisknutí se provede obsah vnitřní while smyčky, která obsahuje hlavní kód programu. Nachází se zde hodnoty úhlu a rychlosti otočení, které se, jak již bylo řečeno na počátku, nastavují na čelním panelu pomocí jednoduchých ovladačů. Prostým výpočtem, tedy podílem úhlu, o který se má robot otočit a rychlosti jakou to má provést, lze zjistit dobu trvání tohoto otočení. Aby však byl tento podíl uskutečnitelný, musí se převést jednotky veličin, neboť úhel zadává uživatel ve

stupních, rychlost v otáčkách za minutu. Stupně lze převést na radiány funkcí Degrees to Radians. Jednotky rychlosti, tedy otáčky za minutu potom funkcí RPM to Radians per Second. Tato hodnota je však ta, jíž se otáčí ozubené kolečko poháněné motorem, nikoliv kolo robota. Při otočení o 3,14 radiánů, neboli 1 π se robot otočí o úhel zhruba 21°. Pro 1 radián pak úhel popojetí robota odpovídá hodnotě cca 6,6° (tuto hodnotu nelze určit zcela přesně, vždy bude, zejména pro různé rychlosti, mírně odlišná, neboť kola robota mohou prokluzovat, robot na příkazy nezareaguje vždy ihned, skutečná rychlost motorů není úplně přesná atd). Vynásobí-li se tedy rychlost motorů v radiánech za sekundu tímto číslem a převede opět na radiány, je možné provést konečné dělení, jehož výsledkem je doba otočení robota. Takto vypočtený čas je poté přiveden na vstup time target funkce Elapsed Time (tato funkce má mnoho vstupů, v tomto programu jsou ale využity pouze 3 z nich). Vstup time target udává, jaká doba musí uběhnout, aby hodnota výstupu time has elapsed byla true. V tomto programu onen výstup slouží jako ukončovací podmínka vnitřní while smyčky a současně vstupuje do shift registru, jehož hodnota je vedena na vstup reset a způsobuje vynulování výstupní hodnoty time has elapsed po každém proběhnutí smyčky. Volba směru otočení je řešena posuvným tlačítkem na čelním panelu, jehož hodnota v blokovém diagramu vstupuje na výběrový terminál case struktury. Tou prochází hodnota rychlostí motorů a při výběru otočení doprava je znegována. V obou případech jsou nastavené rychlosti vedeny do vstupů left motor ccw velocity a right motor ccw velocity funkce Write DC Motor Velocity Setpoints. Nestiskne-li uživatel tlačítko Start, nebo neprobíhá-li vnitřní while smyčka, jsou hodnoty rychlostí motorů nastaveny na 0. Program se ukončí stlačením tlačítka Stop, nebo výskytem chyby. Ukončení komunikace s robotem obstarává opět funkce Close Starter Kit.
Na následujících obrázcích 27 a 28 je zobrazen čelní panel i blokový diagram programu.



Obr. 27: Čelní panel programu Otočení.



Obr. 28: Blokový diagram programu Otočení.

3.6 Graf

Program slouží ke skenování okolí robota.

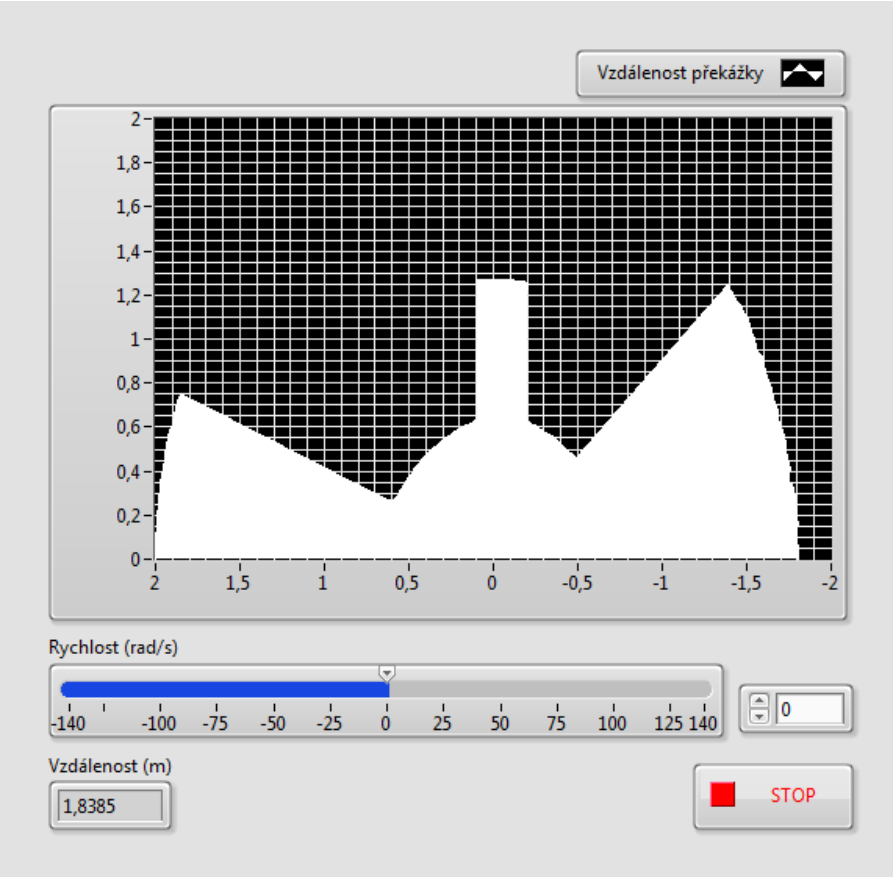
Zdrojový kód programu opět začíná funkcí Initialize Starter Kit 1.0 (sbRIO-9631), která zahájí komunikaci s FPGA. Flat sekvence s funkcí Write DC Motor Velocity Setpoints obstarává, jak již bylo zmíněno dříve, vynulování hodnot rychlostí motorů při spuštění programu. Hlavní kód se nachází ve while smyčce. Funkce Read PING))) Senzor Distance na výstupu distance vrací hodnotu vzdálenosti k nejbližší překážce, kterou robot pomocí ultrazvukového senzoru detekuje. Tato hodnota následně prochází dvěma filtry. První z nich nahrazuje všechny hodnoty menší než 0,04 m hodnotami z předchozího cyklu (popsán v 4.1 Start VI). Druhý filtr naopak porovnává vzdálenost s hodnotou 2 m, a je-li větší, přepíše ji touto hodnotou.

Funkce Initialize Scan Angle Data má 4 vstupy. Max angle, tedy úhel, po který senzor skenuje po levé straně, min angle – úhel skenování na pravé straně, scan angle, neboli krok, po kterém ke skenování dochází a initial angle – počáteční úhel skenování. Z těchto informací je vytvořen klastr hodnot, který je veden skrz funkci Get Next Scan Angle. Výstup new angle

této funkce je nejaktuálnější hodnota úhlu natočení serva se senzorem. Ta vstupuje přes zpětnovazebný uzel do vstupu scan angle funkce Update Scan Distance. Jejím druhým vstupem je new scan distance a je do něj přivedena hodnota vzdálenosti nejbližší překážky. Klastr hodnot, procházející i touto funkcí, se dále ukládá do shift registru a současně vstupuje do funkce Unbundle By Name, která umožňuje oddělit jednotlivé prvky klastru. V tomto případě tedy na výstupu obdržíme pole hodnot vzdáleností a úhlů. Tato pole dál prochází funkcí Polar to Re/Im Function, která převádí polární složky komplexního čísla na složky pravoúhlé. Takto upravená pole poté přes funkci Bundle, která z nich vytvoří opět klastr hodnot, vstupují do XY grafu, v němž uživatel může na čelním panelu pozorovat do jaké vzdálenosti robot „vidí“ při jednotlivých úhlech natočení senzoru. Pomocí posuvného ovladače lze také ovládat rychlost motorů robota. Robot je tedy schopen při svém pohybu skenovat okolí. Uživatel ovšem musí dávat pozor na případný střet robota s překážkou. I v tomto programu je přidané zpoždění mezi průběhy kódu. Ukončení programu se provádí stisknutím tlačítka Stop. Vyskytne-li se při průběhu programu chyba, dojde k automatickému ukončení. Přerušení komunikace s FPGA opět zajišťuje funkce Close Starter Kit VI.

Na obr. 29 je fotografie robota, který skenuje okolí se dvěma překážkami. Čelní panel s odpovídajícím grafem je na obr. 30. Blokový diagram aplikace je zobrazen v příloze 2.

Obr. 29: Robot skenující okolí.



Obr. 30: Čelní panel programu Graf.

3.7 Převod rychlosti

Tento program slouží k převodu jednotek rychlosti z radiánů za sekundu na metry za sekundu. Robot bude jezdit mezi dvěma vzdálenostmi, které uživatel zadá ovladači umístěnými na čelním panelu. Z celkové ujeté vzdálenosti a času, za který ji ujel, se pak jednoduše vypočítá rychlost v metrech za sekundu. Úsek ujede vždy minimálně jednou, ale počet opakování lze změnit pomocí ovladače.

Na začátku funkce Initialize Starter Kit 1.0 (sbRIO-9631) naváže komunikaci s robotem. Následně dojde ve flat sekvenci k vynulování hodnoty rychlosti motorů, a také hodnoty času, který je potřebný pro výpočet. Ve while smyčce se nachází filtr nízkých hodnot vzdáleností ze senzoru, který nahradí všechny hodnoty menší než 0,04 m hodnotami z předchozího cyklu. Takto přefiltrované a navíc funkcí Wait Until Next ms Multiple zpožděné hodnoty se porovnávají s hodnotami z předchozích průběhů smyčky, a jsou-li menší, znamená to, že robot se pohybuje směrem dopředu (průchod hodnot musí být zpožděn, neboť senzor neměří dokonale přesně). Na čelním panelu tedy bude svítit LED indikátor červeně. Jestliže bude

robot couvat, LED bude mít zelenou barvu a jeho hodnota bude true. Výsledná logická hodnota z tohoto porovnání dále vstupuje do dvou logických součtů.

Druhým vstupem prvního logického součtu (1) je výsledek porovnávání vzdálenosti k překážce a kritické hodnoty vzdálenosti vepředu (a). Je-li první hodnota menší než druhá, vychází true. Hodnota z porovnání (a) se v každém případě zapíše do shift registru. Výsledek logického součtu (1) je poté přiveden na výběrový terminál case struktury. Bude-li true, nastaví se v této case struktuře rychlost motorů na zápornou hodnotu. Jinými slovy jestliže se robot nachází v kritické přední vzdálenosti a zároveň jede dopředu, musí začít couvat.

Logická hodnota porovnání (a) dále vstupuje do dalšího logického součtu (3), jehož druhým vstupem je hodnota téhož porovnání, ovšem z shift registru. Výsledek je veden přes negaci do logického součtu (5) do něhož současně vede i výstup součtu (1). Hodnota koncového součtu (5) je přivedena na výběrový terminál case struktury, v níž je-li její hodnota true, se do shift registru, jehož počáteční hodnota je 0, přičte 1. Jinak řečeno jestliže se robot dostal do kritické vzdálenosti vepředu, cyklus zpátky v ní nebyl a zároveň jede dopředu, pak indikátor, který počítá, kolikrát byl robot v této kritické pozici, se zvýší o 1.

Případ, kdy se robot nachází v kritické vzdálenosti vzadu a couvá, je řešen obdobně. Do logického součtu (2) však nyní vstupuje znegovaná logická hodnota LED indikátoru, a výsledná hodnota porovnání velikostí kritické hodnoty vzadu a filtrované hodnoty vzdálenosti překážky (b). Ta opět vstupuje do shift registru. Bude-li nyní hodnota přivedená ze součtu (2) na výběrový terminál true, rychlost bude nastavena jako kladná a robot se tedy rozjede dopředu.

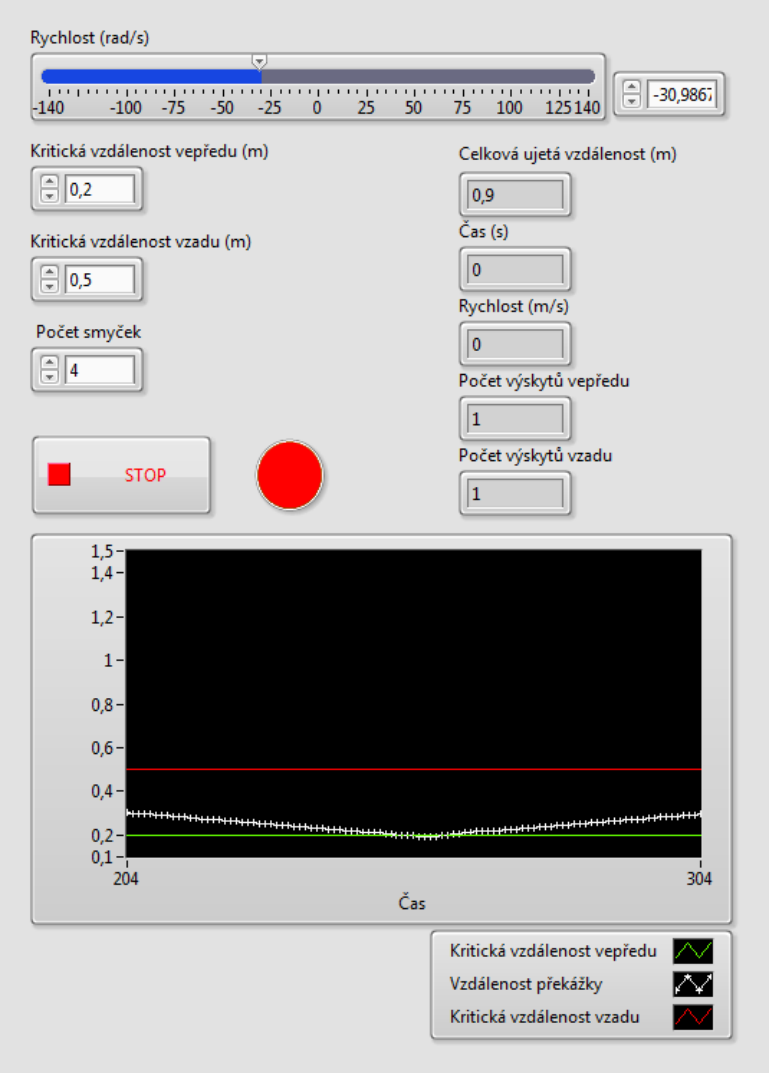
Navyšování hodnoty indikátoru výskytu robota v kritické vzdálenost vzadu, pracuje na témže principu jako indikátor výskytů vepředu.

Jakmile se robot poprvé v průběhu programu dostane do kritické vzdálenosti (ať už vepředu nebo vzadu), uloží se funkcí Tick Count hodnota systémového času. Hodnoty z obou indikátorů výskytů robota v kritických vzdálenostech se sečtou a porovnají s počtem smyček, které má robot ujet. Bude-li počet splněn, while smyčka se ukončí.

V poslední fázi se funkcí Tick Count opět sejme hodnota systémového času, od níž se odečte první uložená. Výsledkem bude čas, za který robot ujel celkovou vzdálenost, jednoduše vypočtenou ze součinu počtu smyček a rozdílu kritických vzdáleností. Konečná rychlost v metrech za sekundu je potom snadno spočtena pomocí vztahu $v = \frac{s}{t}$, kde v (m/s) je rychlost, s (m) dráha a t (s) je čas.

Proběhnou-li všechny operace, nebo dojde-li při běhu programu chybě, ukončí se. Funkce Close Starter Kit potom ukončí komunikaci s robotem.

Na obr. 31 je zobrazen čelní panel programu. V přílohách se potom nachází blokový diagram programu (příloha 3) a tabulka převodu rychlostí (příloha 4) s následným grafickým vyobrazením převodu rychlosti robota v radiánech za sekundu na rychlost v metrech za sekundu (příloha 5).



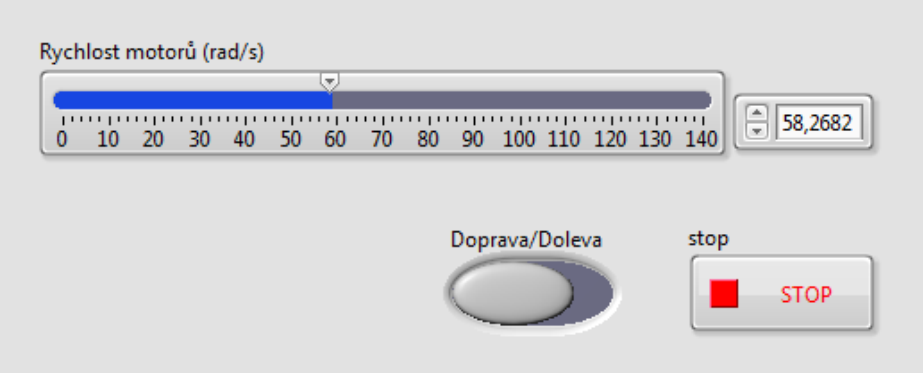
Obr. 31: Čelní panel programu Převod rychlosti.

3.8 Jízda

Po spuštění tohoto programu bude robot samovolně jezdit zadanou rychlostí a v případě, že před sebou detekuje nějakou překážku, otočí se o 90°.

Na počátku funkce Initialize Starter Kit 1.0 (sbRIO-9631) zahájí komunikaci s robotem. Funkce Read PING))) Senzor Distance z výstupu distance vede přes filtr nízkých hodnot vzdálenost překážky změřenou senzorem do porovnání, které ovládá výběrový terminál case struktury. Porovnává se zde naměřená filtrovaná hodnota vzdálenosti s pevně stanovenou minimální vzdáleností překážky. Dostane-li se robot do této minimální vzdálenosti, provede se obsah case struktury pro možnost true, což je v tomto případě subVI Otočení (viz 3.5 Otočení). Po jeho dokončení se opět sejme hodnota vzdálenosti k nejbližší překážce a robot se rozhodne, má-li se znovu otočit nebo pokračovat v přímé jízdě. Program může být ukončen buď výskytem chyby, nebo stisknutím tlačítka Stop. Komunikaci ukončuje funkce Close Starter Kit.

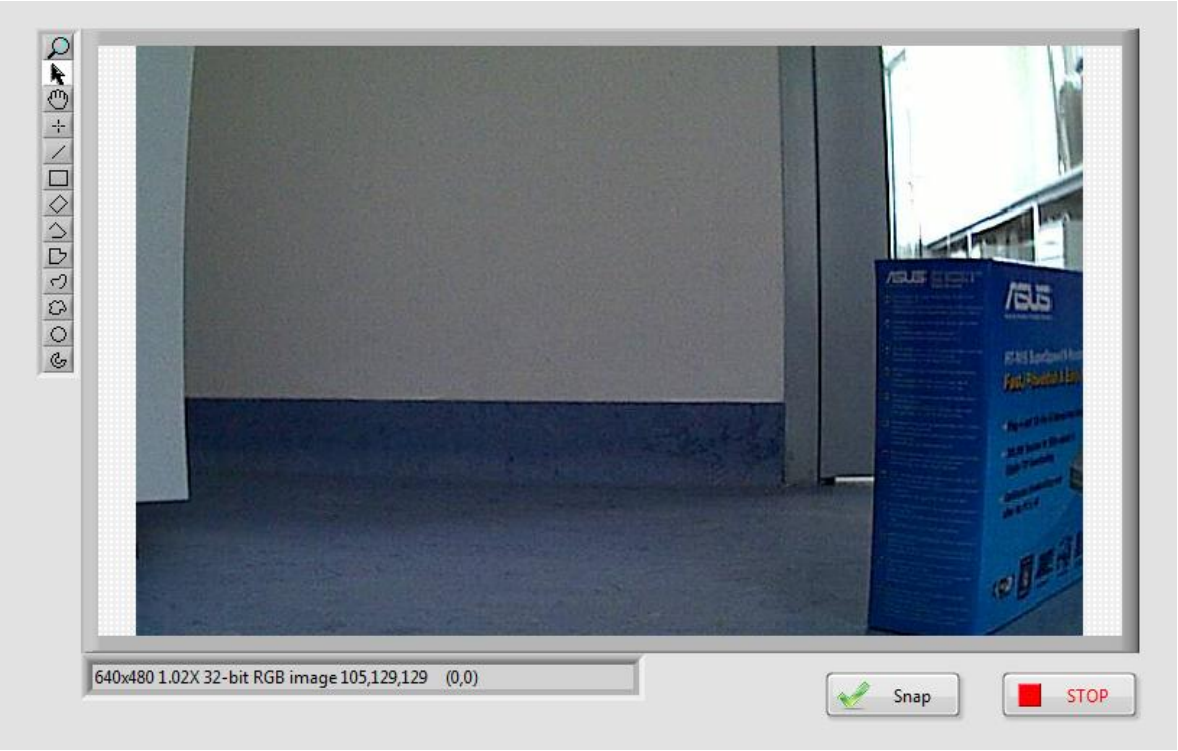
Čelní panel programu je na obr. 32, blokový diagram potom v příloze 6.



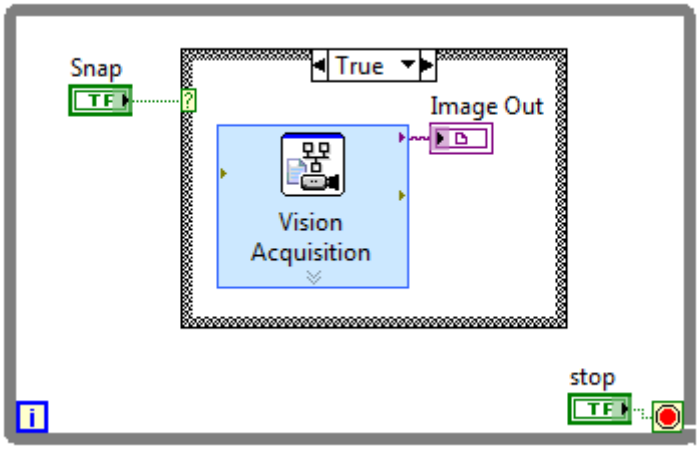
Obr. 32: Čelní panel programu Jízda.

3.9 Kamera

Tento program slouží k ovládání kamery Axis M1011. Kameru je možné mechanicky připevnit k robotovi, a poté je-li program spuštěn, pořídit snímek z kamery stisknutím tlačítka Snap. Všechny pořízené snímky se ukládají do vybrané složky. Na obr. 33 je zobrazen čelní panel programu s fotografií. Ta byla pořízena při průběhu skenování robota, jehož výsledný graf je na obr. 30. Ukončení se provádní tlačítkem stop.Blokový panel tohoto jednoduchého programu se nachází na obr. 34.



Obr. 33: Snímek z Axis kamery.



Obr. 34: Blokový panel programu Kamera.

Závěr

První kapitola této práce seznamuje čtenáře s hardwarem robota. Je zde popsán sbRIO 9631 modul, který umožňuje komunikaci s robotem a jeho programování. Dále je zde vysvětlen princip ultrazvukového senzoru Paralax PING))), který měří vzdálenost překážek před robotem, díky čemuž mu tak umožňuje orientovat se po okolí.

Druhá kapitola pojednává o softwaru, který je nutný pro samotné programování. Nachází se zde jednoduchý popis vývojového prostředí LabVIEW a jeho přídavných modulů. Nechybí ani podrobnější výčet všech struktur a funkcí, které jsou pro vytváření robotických aplikací zcela klíčové.

Třetí kapitola se zabývá již vlastní tvorbou. Ačkoliv se v průběhu práce vyskytlo mnoho různých technických problémů (spálená pojistka, robot byl občas „náladový“ – odmítal komunikovat, nebo používal pouze jeden z motorů), ve výsledku se nakonec podařilo vytvořit celkem 5 aplikací pro ovládání robota. Z těchto 5 aplikací vyžaduje neustálou obsluhu pouze program Start, případně program Otočení, u něhož je k provedení akce nutné stisknutí tlačítka start. Zbylé 3 mohou po počátečním nastavení vstupních hodnot (rychlostí motorů, úhlu natočení senzoru atd.) pracovat zcela samostatně. K robotovi byla dále připevněna kamera Axis M1011, což umožnilo snímání fotografií při jeho volném pohybu.

Do budoucna by pro další práci s robotem bylo nepochybně vhodné zajistit komunikaci přes wifi síť, neboť ethernetový kabel, přes který je komunikace momentálně řešena, velmi omezuje pohyb robota a je celkově v dnešní „bezdrátové době“ velmi nepraktický.

Seznam použitých zdrojů

[1] http://www.nasa.gov/mission_pages/msl/index.html#.U2NNTldg5hE

[2] http://zone.ni.com/reference/en-XX/help/372983B-01/lvrobogsm/robo_skit_overview/

[3] NI sbRIO-961x/963x/964x and NI sbRIO-9612XT/9632XT/9642XT User Guide, 2010. Dostupné na http://digital.ni.com/manuals.nsf/websearch/00E7F65295734CA486257A8700681784

[4] http://sine.ni.com/nips/cds/view/p/lang/cs/nid/205894

[5] PING))) Ultrasonic Distance Sensor Product Guide, 2013. Dostupné na http://www.parallax.com/product/28015

[6] http://labviewwiki.org/Home

[7] http://zone.ni.com/reference/en-XX/help/371361J-01/

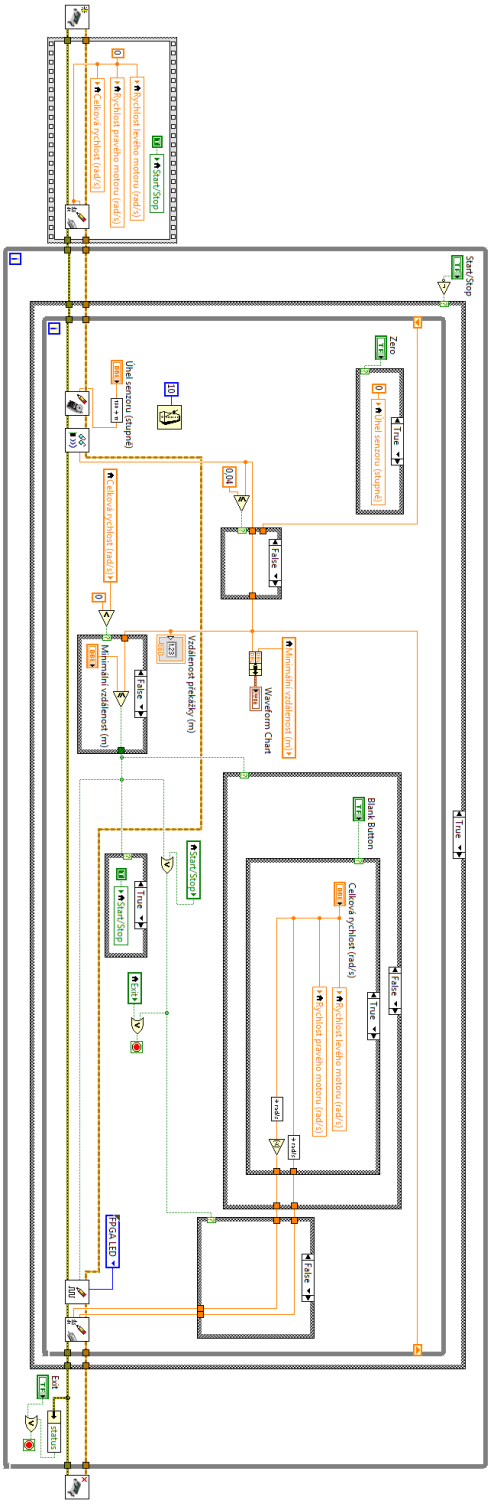
[8] http://sine.ni.com/nips/cds/view/p/lang/cs/nid/11834

[9] http://sine.ni.com/nips/cds/view/p/lang/cs/nid/209855

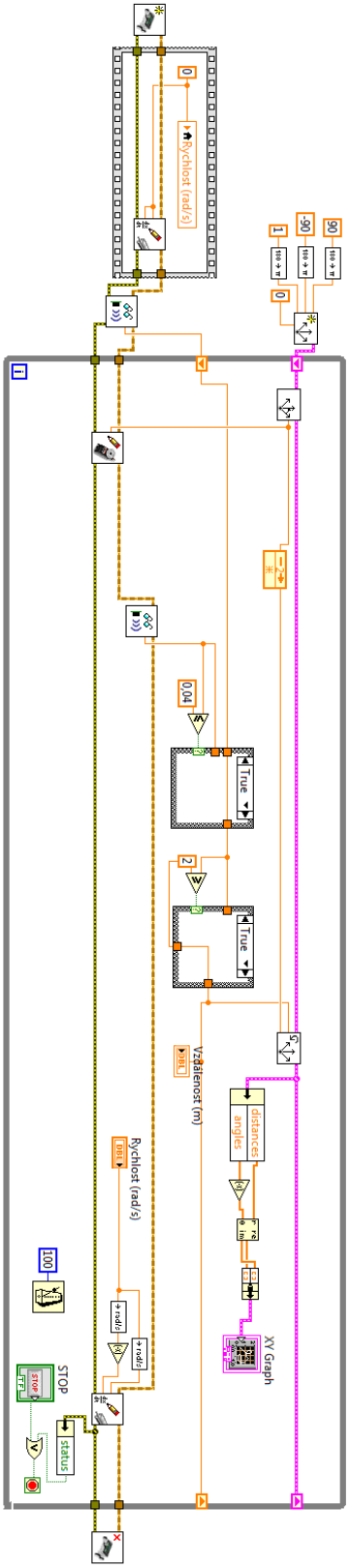
[10] http://sine.ni.com/nips/cds/view/p/lang/cs/nid/209856

Seznam příloh

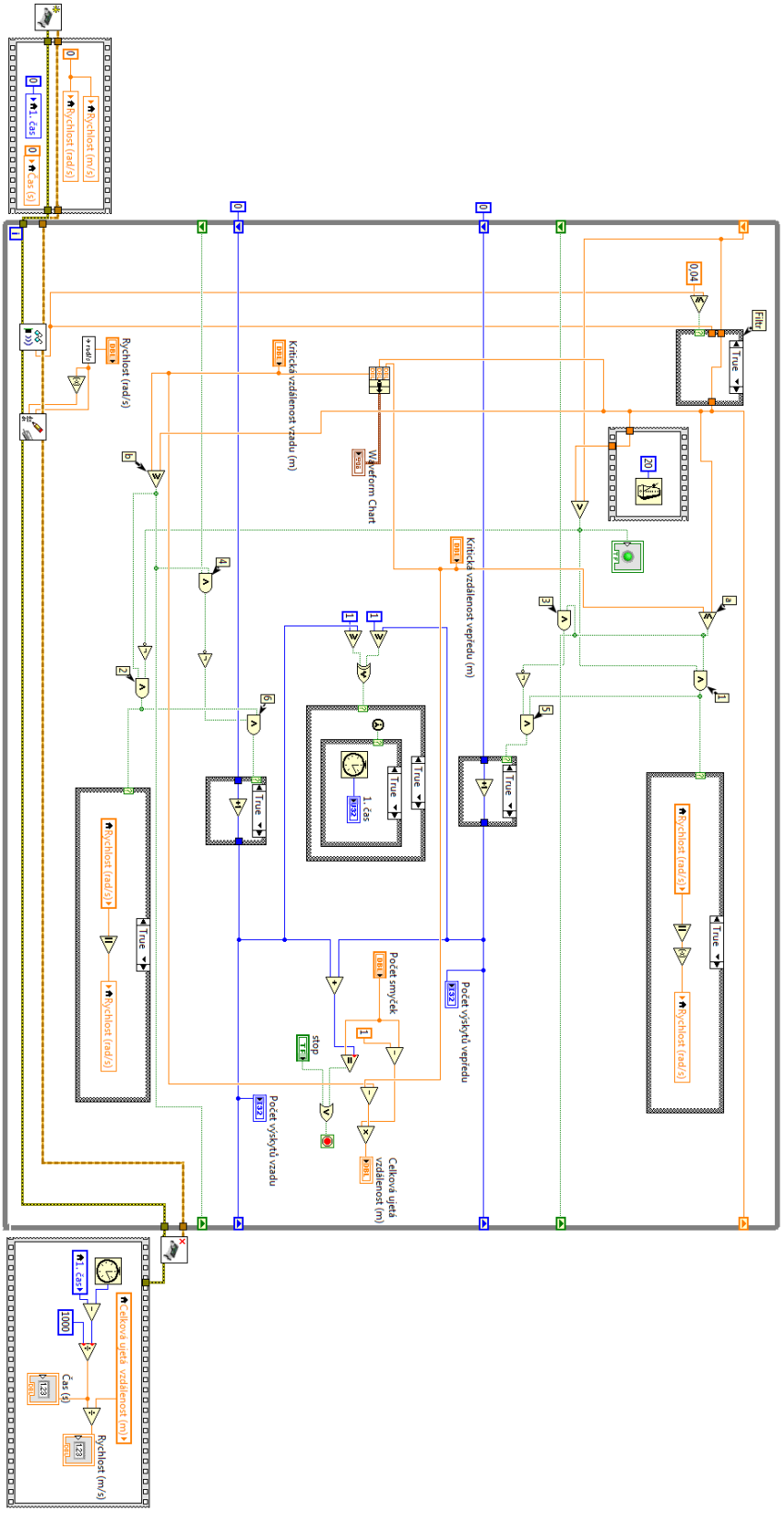
Příloha 1: Blokový diagram programu Start.



Příloha 2: Blokový diagram programu Graf.



Příloha 3: Blokový diagram programu Převod rychlosti.

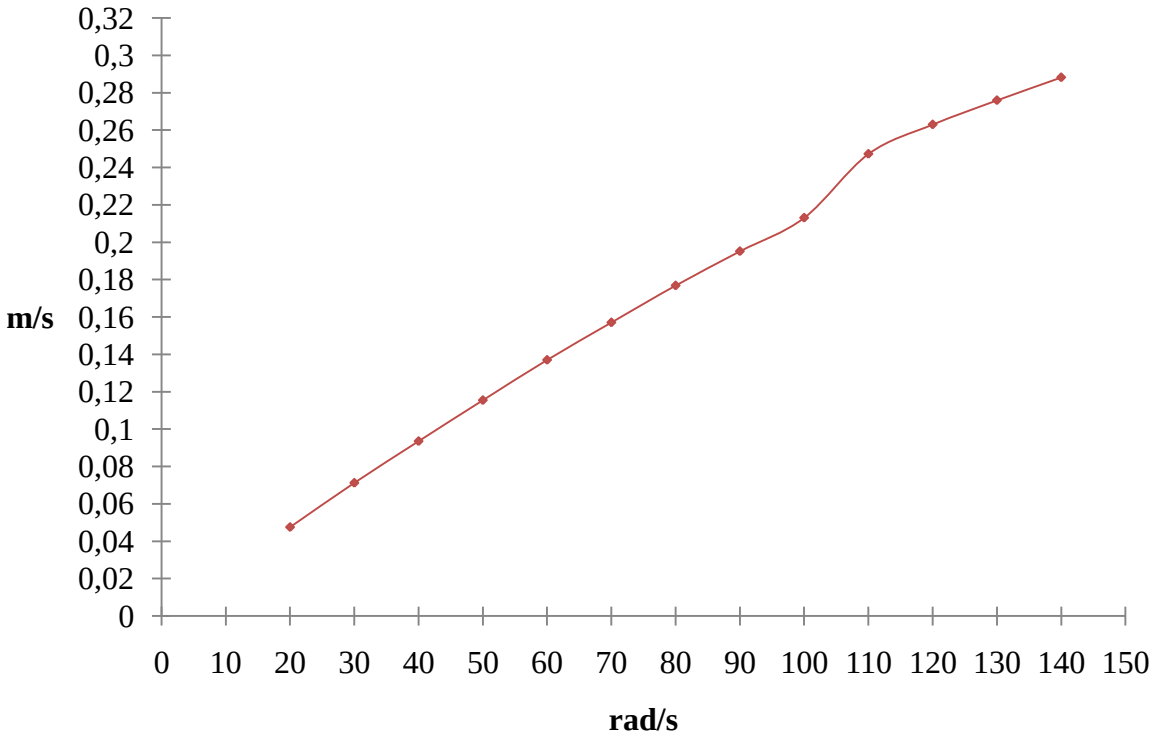


Příloha 4: Tabulka rychlostí v radiánech za sekundu a příslušných hodnot v metrech za sekundu.

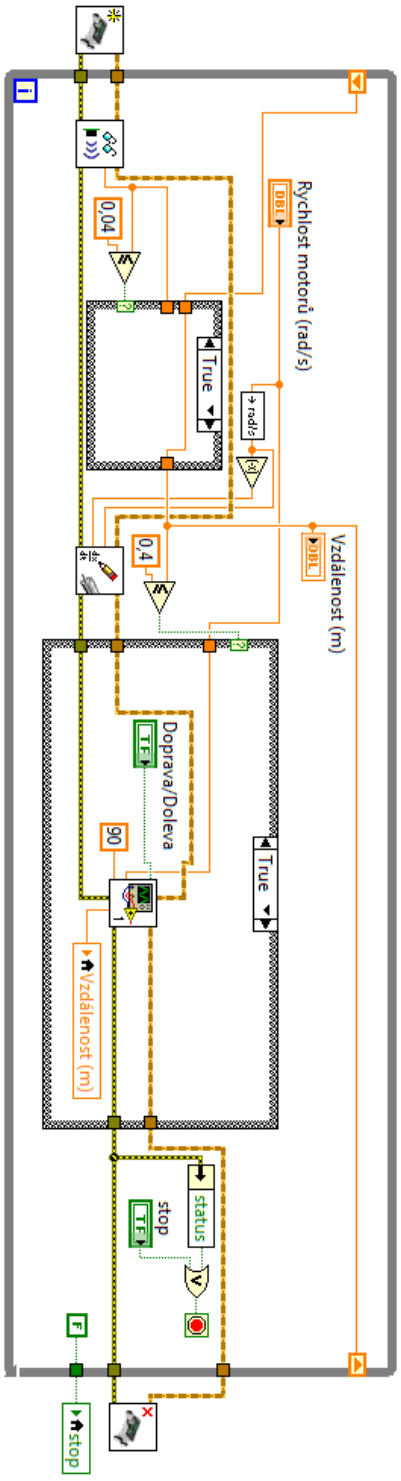
v [rad/s]	v [m/s]	t [s]		v [m/s]	t [s]	v_p [m/s]
20	0,04778	20,927		0,04810	20,789	0,04756
	0,04787	20,889		0,04787	20,891	
	0,04673	21,399		0,04778	20,930	
	0,04797	20,848		0,04777	20,933	
	0,04782	20,910		0,04556	21,950	
30	0,07127	14,032		0,07108	14,068	0,07123
	0,07147	13,991		0,07086	14,113	
	0,07127	14,032		0,07117	14,051	
	0,07089	14,106		0,07159	13,968	
	0,07108	14,069		0,07159	13,968	
40	0,09335	10,712		0,09369	10,673	0,09353
	0,09338	10,709		0,09408	10,629	
	0,09369	10,673		0,09338	10,715	
	0,09355	10,689		0,09314	10,736	
	0,09390	10,650		0,09321	10,728	
50	0,11534	8,670		0,11472	8,717	0,11546
	0,11488	8,708		0,11585	8,632	
	0,11559	8,651		0,11482	8,709	
	0,11665	8,573		0,11563	8,648	
	0,11585	8,632		0,11532	8,672	
60	0,13750	7,273		0,13712	7,239	0,13702
	0,13646	7,328		0,13755	7,270	
	0,13644	7,329		0,13676	7,312	
	0,13644	7,328		0,13755	7,270	
	0,13682	7,309		0,13751	7,272	
70	0,15620	6,402		0,15691	6,373	0,15708
	0,15840	6,313		0,15652	6,389	
	0,15741	6,353		0,15652	6,389	
	0,15743	6,352		0,15800	6,329	
	0,15699	6,370		0,15645	6,392	
80	0,17640	5,669		0,17765	5,629	0,17686
	0,17759	5,631		0,17627	5,673	
	0,17637	5,670		0,17640	5,669	
	0,17693	5,652		0,17693	5,652	
	0,17634	5,671		0,17768	5,628	

90	0,19493	5,130		0,19425	5,148	0,19518
	0,19486	5,132		0,19501	5,128	
	0,19410	5,152		0,19639	5,092	
	0,19346	5,169		0,19643	5,091	
	0,19486	5,132		0,19751	5,063	
100	0,21227	4,711		0,21222	4,712	0,21311
	0,21236	4,709		0,21250	4,706	
	0,21313	4,692		0,21340	4,673	
	0,21218	4,713		0,21409	4,671	
	0,21409	4,671		0,21432	4,666	
110	0,24576	4,069		0,24795	4,033	0,24728
	0,24956	4,007		0,24564	4,071	
	0,24606	4,464		0,24552	4,073	
	0,24606	4,064		0,25056	3,991	
	0,24716	4,046		0,24851	4,024	
120	0,26076	3,835		0,26511	3,772	0,26302
	0,26253	3,809		0,26385	3,790	
	0,26253	3,809		0,26261	3,808	
	0,26247	3,810		0,26378	3,791	
	0,26392	3,789		0,26267	3,807	
130	0,27556	3,629		0,27996	3,572	0,27598
	0,27049	3,697		0,28019	3,569	
	0,27556	3,629		0,27910	3,583	
	0,27382	3,650		0,27548	3,630	
	0,27405	3,649		0,27556	3,629	
140	0,28703	3,484		0,28835	3,468	0,28824
	0,28969	3,452		0,28694	3,485	
	0,28637	3,492		0,28969	3,452	
	0,28596	3,497		0,28835	3,468	
	0,29002	3,448		0,29002	3,448	

Příloha 5: Graf převodu rychlosti robota z radiánů za sekundu na metry za sekundu.



Příloha 6: Blokový diagram programu Jízda.



Příloha 7: CD, které obsahuje kopii práce v pdf formátu a LabVIEW projekt se všemi popsanými aplikacemi.