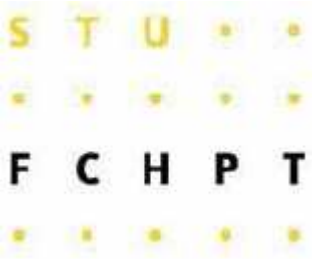
SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

Ústav informatizácie, automatizácie a matematiky



RIADENIE LEGO ROBOTOV

Bakalárska práca

Vypracoval: Jarmila Grancová

Školiteľ: Ing. Michal Kvasnica, PhD.

2009

Čestné vyhlásenie

Týmto podpisom potvrdzujem, že som túto prácu vypracovala so spoluziačkou Katarínou Chroustovou a to tak, že každá z nás vypracovala svoju časť samostatne s použitím prameňov, ktoré uvádzam v zozname literatúry a že som ich použila spôsobom vo vede obvyklým.

.....

Jarmila Grancová

Poďakovanie

Ďakujem vedúcemu mojej bakalárskej práce, Ing. Michalovi Kvasnicovi, PhD., za odborné usmernenie, pomoc a trpezlivosť, ktoré mi počas prípravy tejto práce poskytol.

zadanie

Abstrakt

Grancová, Jarmila: Riadenie LEGO robotov. Bakalárska práca, Slovenská Technická Univerzita. Fakulta chemickej a potravinárskej technológie. Vedúci bakalárskej práce: Ing. Michal Kvasnica, PhD. Bratislava 2009. 52 s.

Hlavným cieľom práce je riadenie robotov stavebnice LEGO Mindstorms pomocou softvéru Stateflow, ktorý je prepojený so Simulinkom. Prvé dve kapitoly sú venované teoretickému oboznámeniu so softvérom a stavebnicou Lego Mindstorms NXT. Dôraz sa kladie na prácu s balíkom stateflow a na popis jednotlivých súčastí stavebnice Lego Mindstorms NXT. Poznatky uvedené v prvých kapitolách sa využívajú pri praktických úlohách, ktorých riešenie je obsahom ďalších kapitol. Práca popisuje riadenie jednotlivých úloh v Stateflow, skladajúcich sa z čiastkových riešení, ktorých vzájomné prepojenie vytvára konečnú celkovú úlohu. Výsledkom práce je popísanie jednotlivých postupov pri riešení úloh pomocou prostredia Stateflow, ktoré môžu slúžiť ako pomôcka na laboratórnych cvičeniach.

Kľúčové slová: stateflow, LEGO Mindstorms, NXT kocka, senzory, servomotory.

Abstract

Grancová, Jarmila: Control of LEGO Robots. Bachelor's thesis. Slovak Technical University, Faculty of Chemical and Food Technology. Supervisor of the bachelor's thesis: Ing. Michal Kvasnica, PhD. Bratislava 2009. 52 pp.

The purpose of this work is to describe control of the 'LEGO Mindstorms building system' robots using the Stateflow software which is interconnected with Simulink. The first two chapters are devoted to provide the theoretical background information on the software and the LEGO Mindstorms NXT building system. The focus is put on the work with the Stateflow package and on the description of the constituent elements of the LEGO Mindstorms building system. The theoretical information stated in the first chapters of the thesis is then utilized in the practical exercises solving in the following chapters. The thesis aims to depict the operation of individual tasks in Stateflow that consist of partial solutions. The final task is formed by the interconnection of these partial solutions. The overall result of the thesis is to describe the individual procedures when solving the problems using the Stateflow software which can serve as an aid in laboratory exercises.

Key words: Stateflow, LEGO Mindstorms, NXT cube, sensors, servo motors

Obsah

Zoznam obrázkov

Úvod

Riadenie procesov je čoraz aktuálnejšou témou, neustále sa stretávame s riadiacimi systémami, či už o tom vieme alebo nie. S každým novým dňom prichádza potreba veci vylepšovať, zdokonaľovať, uľahčovať prácu ľudí a nahrádzať ju prácou inteligentných robotov. Správne riadenie procesov môže zredukovať neefektívnosť a chyby vyplývajúce z nadbytočnosti informácií a akcií. Dobre navrhnutý systém nám uľahčuje prácu a čas.

Zdokonaľovanie výpočtovej techniky umožňuje pomerne rýchlo vytvárať modely aj zložitých systémov. Tvorbou modelov a ich simuláciami sa zaoberá aj programový balík Matlab. K produktom využívaným na modelovanie udalostných systémov patrí Stateflow od spoločnosti MathWorks, ktorého popis je obsahom prvej kapitoly práce.

Pre potreby vzdelávania v oblasti robotiky bola vyvinutá špeciálna pomôcka - stavebnica Lego Mindstorms. Lego Mindstorms NXT kombinuje inteligentnú kocku s mikropočítačovým mozgom, mnohými senzormi a s jednoduchým softvérom. Umožňuje vytvoriť rôzne konštrukcie robotov na plnenie rozmanitých úloh a činností. Vytvorenie riadeného systému a jeho overenie v reálnom priestore a čase pomocou robota prispieva k väčšiemu záujmu študentov, k pochopeniu princípov a k predvídaniu udalostí, ktoré môžu pri návrhu systému nastať.

Účelom práce je spojenie získaných teoretických poznatkov s praktickým riešením úloh rozdielnych typov. Začiatok každej úlohy tvorí jej stručný popis, v ktorom je uvedená činnosť, ktorú má robot vykonať a dostupné senzory pre daný typ úlohy. Ďalej je uvedené prepojenie NXT kocky so vstupmi a výstupmi. Celková úloha je pre lepšie pochopenie rozčlenená do viacerých čiastkových úloh, pre ktoré je následne vysvetlené vytvorenie stavových diagramov v prostredí Stateflow. Ku každému riešeniu je pre správne pochopenie pripojený obrázok s vytvorenou schémou v Stateflowe a simulinku.

Práca je určená pre tých, ktorí sú len začiatočníkmi v oblasti riadiacich systémov alebo prichádzajú s týmto pojmom do styku prvýkrát, ale ako pomôcka môže poslúžiť aj pokročilejším. S Lego Mindstorms NXT môže pracovať tak jednotlivec ako aj tímy, čo vedie k väčšej motivácii a k získaniu skúseností potrebných pre prax. Riadiaci systém je v prevažnej miere výsledkom skupinovej práce, kde jednotlivec prispeje svojím dielom k vytvoreniu jednotného fungujúceho celku.

1 Softvér

Stateflow je graficko-návrhový a vývojový nástroj na simulovanie komplexných reakčných systémov. Jedinečnosť programu je v jeho jednoduchom prepojení s modelom reálneho zariadenia v simulinku. Poskytuje efektívne prostredie pre návrh vstavaných systémov a jazykové prvky potrebné na opis zložitých logických systémov v ľahko čitateľnej a zrozumiteľnej forme. Tento produkt sa dá spustiť pod rôznymi operačnými systémami, ako sú napríklad MS Windows, Solaris, Mac OS X alebo Linux. [1]

1.1 Stateflow

Softvér je primárne určený na modelovanie udalostných systémov, teda systémov, ktoré reagujú na určitú udalosť, prípadne sekvenciu udalostí. Tieto systémy prechádzajú z jedného operačného módu do ďalšieho na základe reakcie na udalosti a podmienky. Udalostne riadené systémy sú obvykle modelované podľa teórie konečných automatov, kotré predstavujú operačné módy ako stavy. Stateflow poskytuje grafické objekty, ktoré stačí uchytiť myšou a jednoducho premiestniť na požadované miesto a sériu prechodov na označenie smeru prechodov z jedného stavu do ďalšieho. [2]

Zakladá sa na teórii konečného stavového automatu. [1] V takomto systéme dochádza k prechodu z jedného stavu do druhého predpísaného stavu. Prechod sa vykoná, len ak je splnená podmienka. Na začiatku sa automat nachádza v počiatočnom stave. Do nového stavu sa dostane po splnení definovanej podmienky. V tomto stave ostane, až kým nebude splnená výstupná podmienka. Na vytvorenie grafickej prezentácie takéhoto stavového automatu slúži Stateflow.

Stateflow stavy umožňujú grafické znázornenie hierarchie a paralely stavov a prechodov medzi nimi. Zväčšuje tradičné možnosti stavových diagramov tokmi stavov, vstavanými funkciami v matlabe, grafickými funkciami, pravdivostnými tabuľkami, dočasnými operátormi a podporou pre integráciu C kódu. Podporuje zbernice signálov, vektorov, matíc a pevne stanovených údajov. [1]

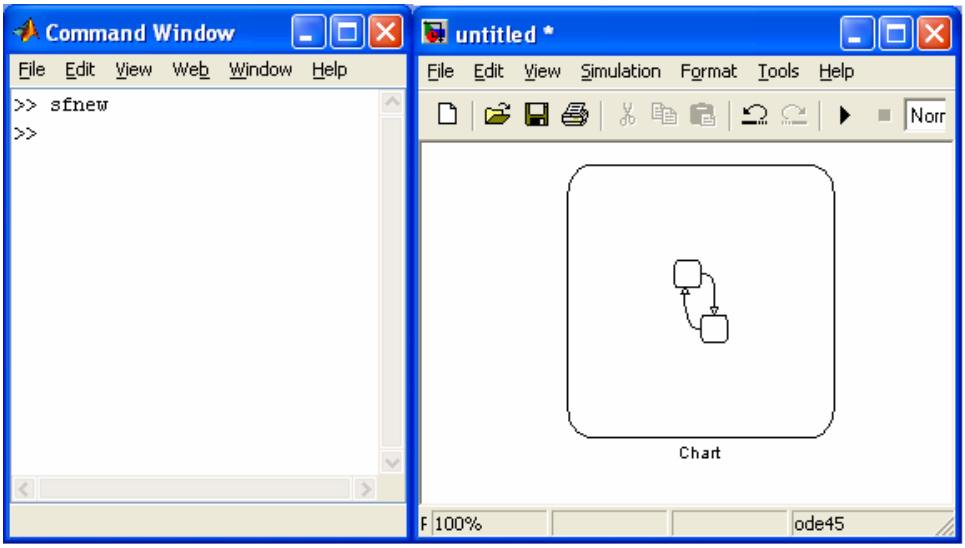
V Stateflowe môžeme:

- Vizualizovať model a simulovať komplexné reakčné systémy založené na logickom automate a konečnom počte stavov.
- Navrhnúť a vyvíjať deterministické, dozorné, diagnostické a riadiace systémy.
- Ľahko meniť návrh, vyhodnotiť výsledky a overiť správanie sa systému v rôznych etapách nášho návrhu.
- Využiť výhodu integrácie matlabu a simulinku na model, simuláciu a analýzu systému.

Stateflow sa používa spolu so simulinkom, správanie stateflow modelov dopĺňa algoritmické správanie modelu simulink. Diagramy Stateflowu sú začlenené do simulinku, aby ho rozšírili o schopnosť využívať riadenie pomocou udalostí.

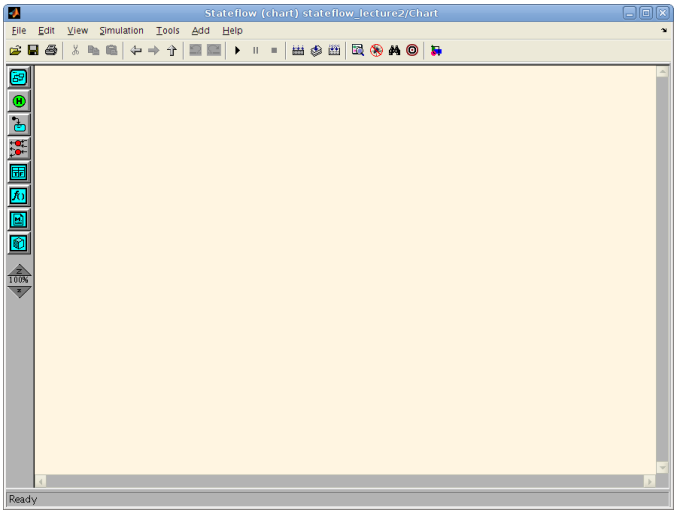
1.2 Vytváranie stavových diagramov

Diagramy v Stateflowe predstavujú konečný stavový automat, kde stavy a prechody medzi nimi tvoria základné bloky celého systému. Stavy môžu byť aktívne alebo neaktívne. Ak je stav aktívny, systém sa v tomto stave nachádza a naopak, ak je stav neaktívny, systém v tomto móde nie je. Najvyšším stavom je samotný graf (Chart), nazývame ho aj rodičovský stav. Systém sa môže pridať do schémy zo Simulink Library Browser alebo pomocou príkazu v „Matlab Command Window: „sfnew“..[2]



Obr. 1 Stateflow v prostredí simulink

Dvojklikom na blok chart sa otvorí Stateflow editor, kde môžeme vytvoriť udalostne riadený systém.



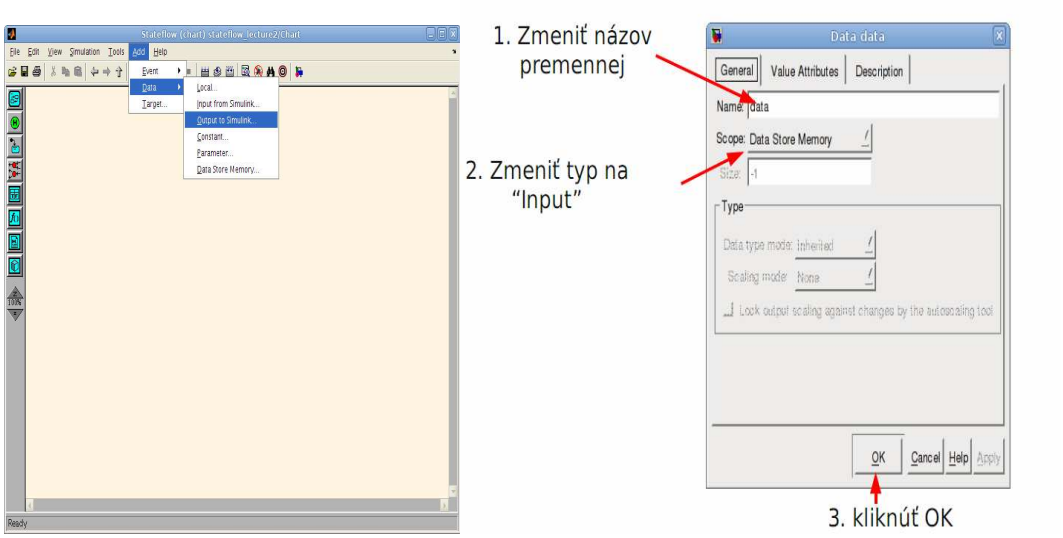
Obr. 2 Stateflow editor

Vytvorenie diagramu je bližšie vysvetlené na jednoduchom príklade otvárania a zatvárania brány.

1.2.1 Pridanie udalostí

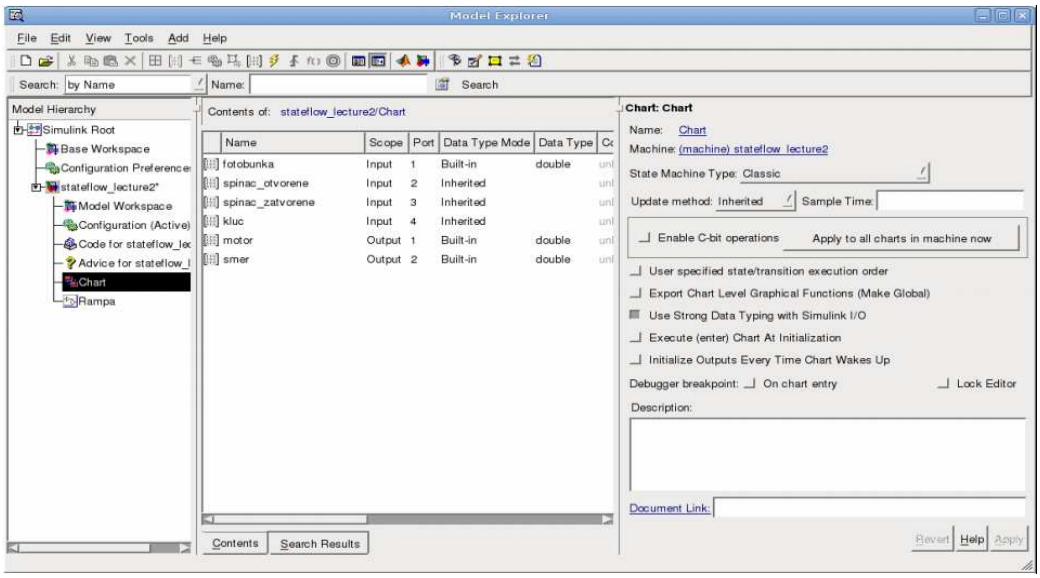
Stateflow rozpoznáva dve udalosti a takzvané volanie funkcie. Z dvoch udalostí môže reagovať na nábehovú (rising trigger), dobehovú (falling trigger) a obe hrany (either trigger). Udalosti vieme definovať ako vstupy do simulinku, lokálne a ako výstupy zo simulinku. [2] V našom prípade sú vstupmi do systému fotobunka, spínač pre otvorenie, spínač pre zatvorenie a kľúč.

Vstupy sa definujú v Stateflow editore. Na hornej lište sa nachádza možnosť „Add“ (pridať). Vyberie sa kolonka „Data“ a v nej „Input from Simulink“. Otvorí sa okno „Data data“, v ktorom sa do kolonky „Name“ napíše názov premennej a možnosť „Scope“ treba zmeniť na „input“. Nakoniec sa pridanie potvrdí kliknutím na „Ok“. Takto sa pridajú všetky vstupy, ktoré sa budú v riadení využívať.



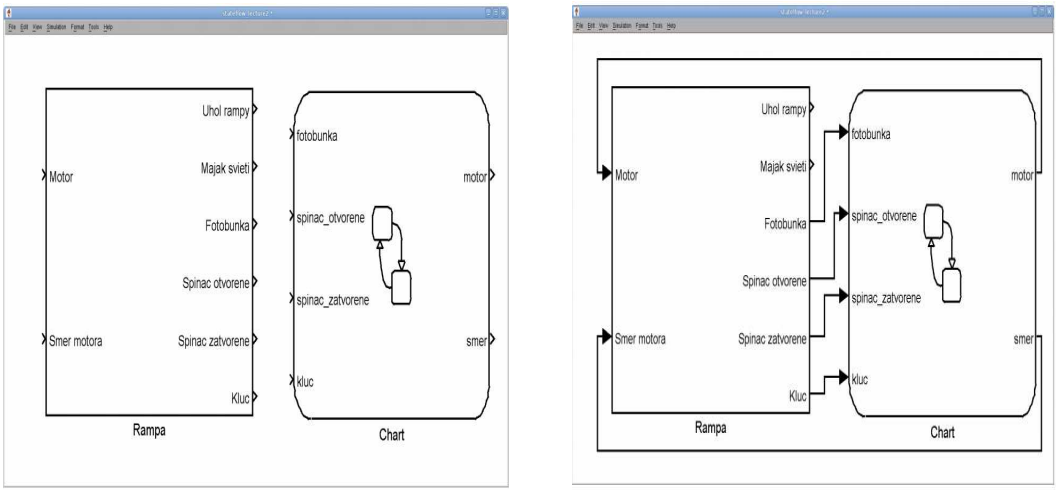
Obr. 3 Pridanie vstupov

Podobným spôsobom pridáme výstupy zo simulinku. Výstupmi z riadiaceho systému sú motor a smer motora. V možnosti „Add“ vyberieme položku „Data“ a potom „Output to Simulink“. V novootvorenom okne v kolónke „Name“ uvedieme názov premennej a zmeníme typ na „output“. Obdobne môžeme pridávať aj lokálne premenné. Pomôckou na pridávanie udalostí je „Model Explorer“, v ktorom vidíme celú štruktúru „Chartu“ a môžeme meniť vlastnosti. Využívame ho aj, keď sa pri pridávaní vstupu alebo výstupu pomýlime. Nachádza sa v možnosti „Tools“ na hornej lište v Stateflow editore.



Obr. 4 Model Explorer

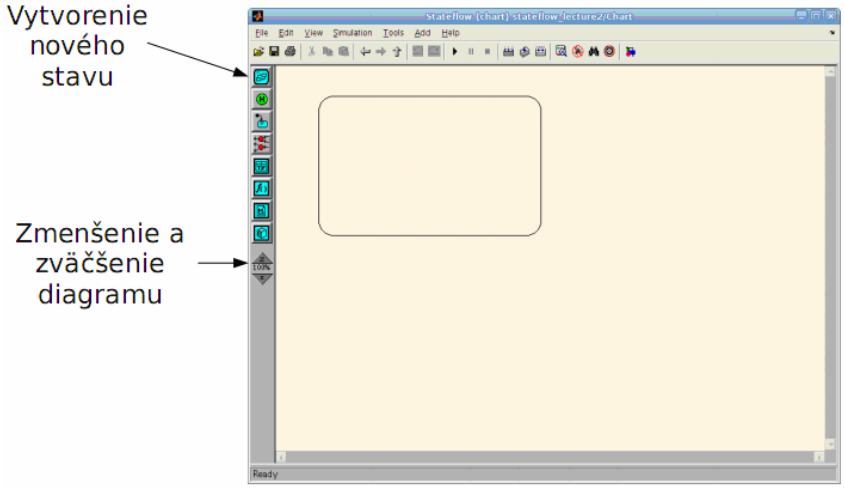
Po pridaní všetkých vstupov a výstupov prepojíme riadenie a model.



Obr. 5 Schéma prepojenia riadenia a modelu

1.2.2 Vytvorenie stavu

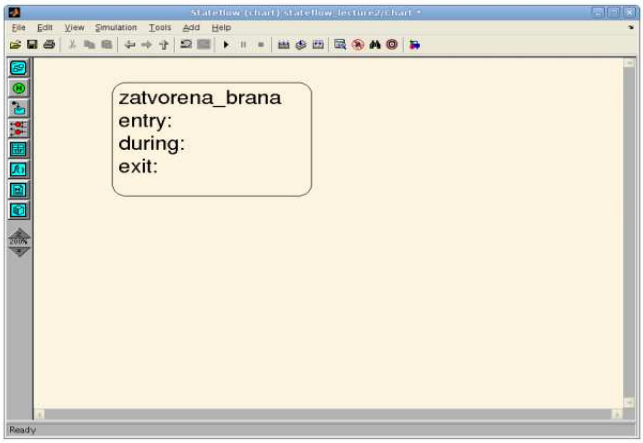
Základom Stateflowu je objekt reprezentujúci „stav“ automatu.



Obr. 6 Pridanie stavu

Všetky stavy musia byť pomenované. V našom príklade je názov pre prvý stav „zatvorena_brana“. Názov stavu nesmie obsahovať diakritiku, medzery, čiarky, bodky, nesmie začínať číslom a musí byť napísaný v prvom riadku bloku. Každý stav musí mať iné pomenovanie. Stav má tri režimy prevádzky - „entry“, „during“ a „exit“. Vlastnosť „entry“ hovorí, čo sa má vykonať pri vstupe do stavu. Akcie definované v režime

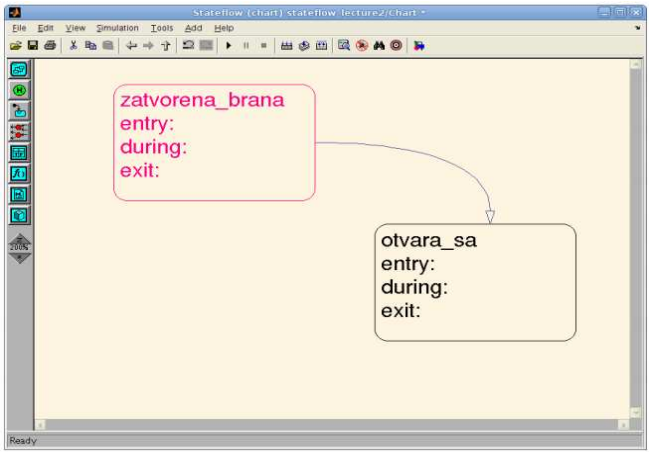
„during“ sa vykonávajú v každom kroku simulácie. Pri opúšťaní stavu sa vykonajú príkazy definované v režime „exit“.



Obr. 7 Vloženie názvu stavu a obsahu

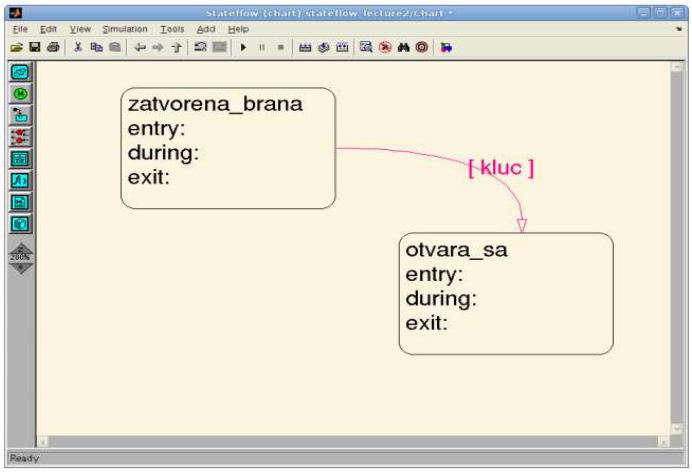
1.2.3 Vytvorenie prepojení medzi stavmi a pridanie podmienok

Ďalším dôležitým krokom je vytvoriť prepojenia medzi jednotlivými stavmi. Prechod je v Stateflowe reprezentovaný šípkou, pričom dôležitý je smer šípky. Pre náš prípad to znamená, že brána môže prejsť zo stavu „zatvorena_brana“ len do stavu „otvara_sa“, nie naopak. Ak sa systém môže dostať zo stavu do viacerých stavov, prednosť má prechod označený číslom jeden. Stateflow čísluje podmienky v smere hodinových ručičiek od prvého horného rohu vzostupne. Prechod vložíme nasledujúcim postupom. Po kliknutí ľavým tlačidlom myši na hranu stavu (nesmieme kliknúť do rohu) kurzor zmení tvar na krížik. Tlačidlo myši držíme stále stlačené a kurzorom ukážeme na hranu stavu, do ktorého sa chceme dostať. Prepojenie je vytvorené.



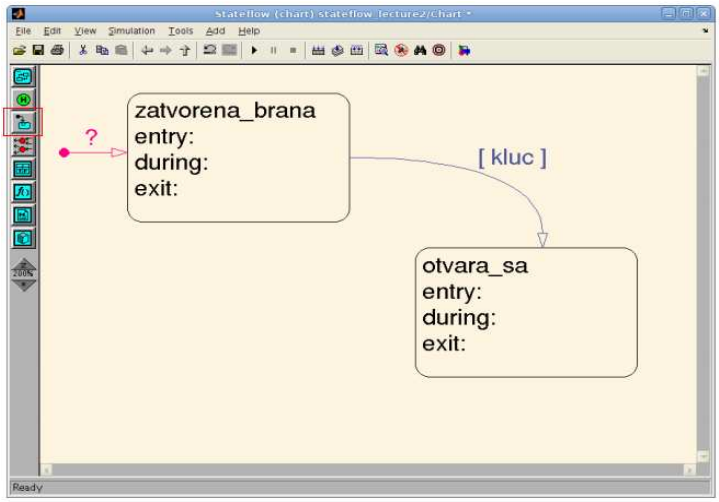
Obr. 8 Vytvorenie prepojenia medzi stavmi

Zmena stavu môže nastať len vtedy, ak je splnená podmienka prechodu, ktorú musíme zadefinovať. Ak nad šípkou nie je definovaná žiadna podmienka, prechod sa vykoná automaticky. Našou podmienkou je, že brána sa môže otvoriť až po vložení kľúča. Na šípku klikneme ľavým tlačidlom, pričom sa zmení jej farba na ružovú a vytvorí sa symbol otáznika. Ľavým tlačidlom klikneme do symbolu a napíšeme podmienku. Po jej napísaní stačí kliknúť na voľnú plochu, nemusíme stláčať enter. Podmienky prechodu musia byť napísané v hranatých zátvorkách. Kým nebude splnená výstupná podmienka z počiatočného stavu, automat v tomto stave ostane. Stateflow rozoznáva logické podmienky, akou je už aj v našom prípade spomínaný kľúč a ďalšie, ktoré môžu byť typu: podmienka1 alebo podmienka2, alebo druhý typ: podmienka1 a zároveň podmienka2. Ďalším typom sú podmienky vo forme nerovností, kedy je hodnota premennej väčšia alebo menšia ako zadaná hodnota, alebo väčšia a zároveň rovná (obdobne menšia a zároveň rovná) danej hodnote.



Obr. 9 Vloženie podmienky prechodu

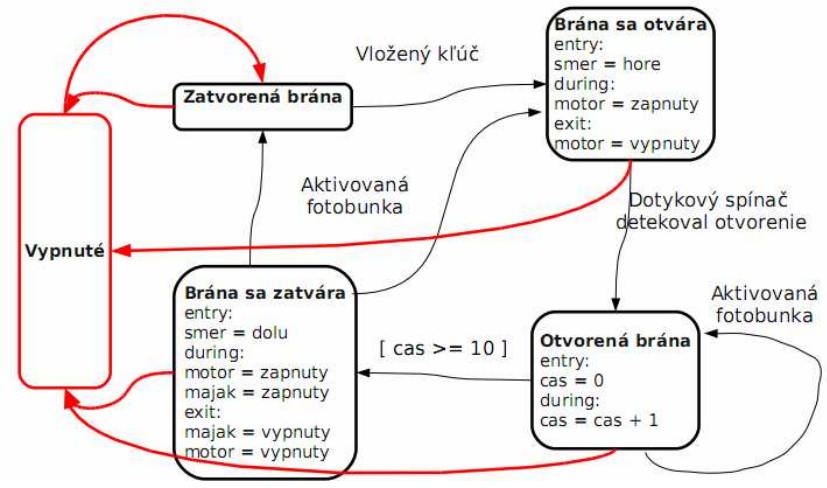
Pri vytváraní stavového diagramu nesmieme zabudnúť určiť prvý aktívny stav, teda do ktorého stavu automat vstúpi po spustení simulácie. Kliknutím na ikonu pre vloženie prvého aktívneho stavu sa zobrazí šípka s bodkou, ukážeme ňou na hranu stavu, ku ktorému ju chceme pripojiť. Táto šípka neobsahuje žiadnu podmienku.



Obr. 10 Určenie prvého aktívneho stavu

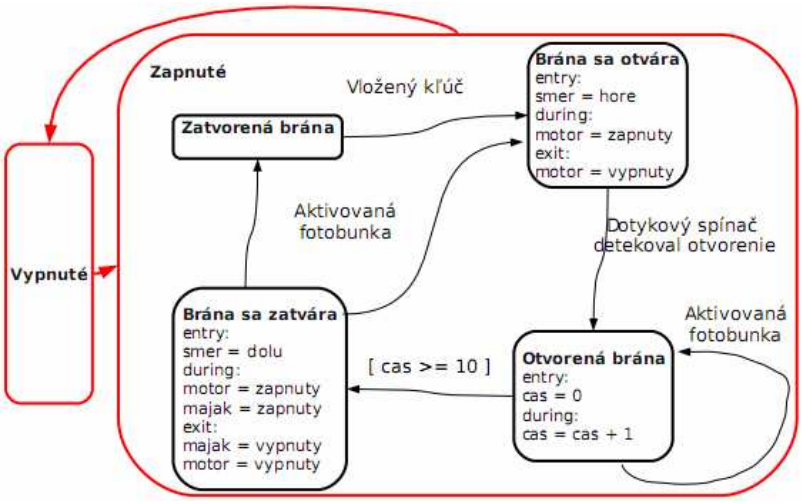
1.3 Stavy v stavoch

Pri navrhovaní a riadení systémov sa častokrát stretávame s veľmi komplikovanými a neprehľadnými diagramami. Takéto úlohy môžeme zjednodušiť vytvorením stavov v stavoch. Takýmto usporiadaním získame zjednodušenie celkového diagramu. Tento diagram je prehľadnejší, zjednodušujú sa prechodové podmienky, vieme definovať históriu. Pridajme do nášho ukážkového príkladu s bránou stav, v ktorom je brána vypnutá. Do tohto stavu sa môže dostať z akéhokoľvek aktívneho stavu. Stavový diagram by vyzeral následovne:



Obr. 11 Riadenie bez superstavu

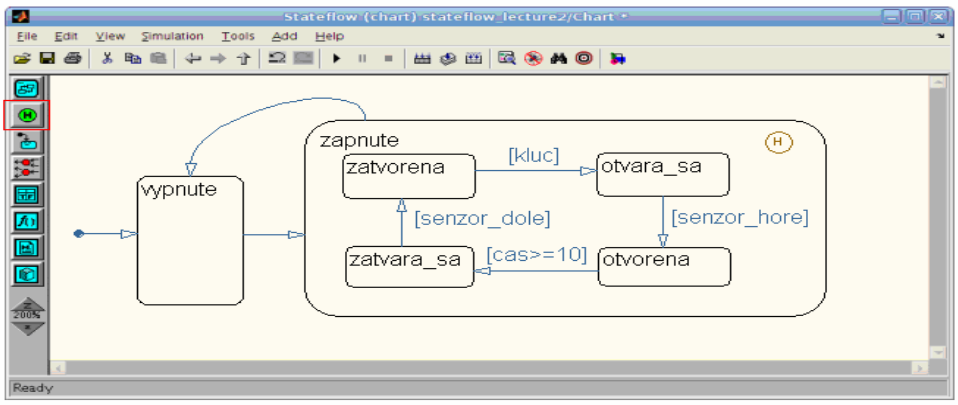
Vidíme, že schéma je neprehľadná a komplikovaná. Použime na riadenie superstav, kde prvý hlavný stav predstavuje vypnutú bránu a druhý zapnutú.



Obr. 12 Riadenie pomocou stavov v stavoch

Pri takomto type riadenia platia určité podmienky, ktoré musíme pre funkčné riadenie dodržať. Každý stav musí byť pomenovaný, názov musí spĺňať podmienky uvedené pre pomenovanie stavu. Nesmieme zabudnúť označiť defaultné stavy, aj počiatočný stav vo vnútri hlavného stavu. Hranice stavu nesmú presahovať prechodové šípky, podmienky, ani jednotlivé podstavy.

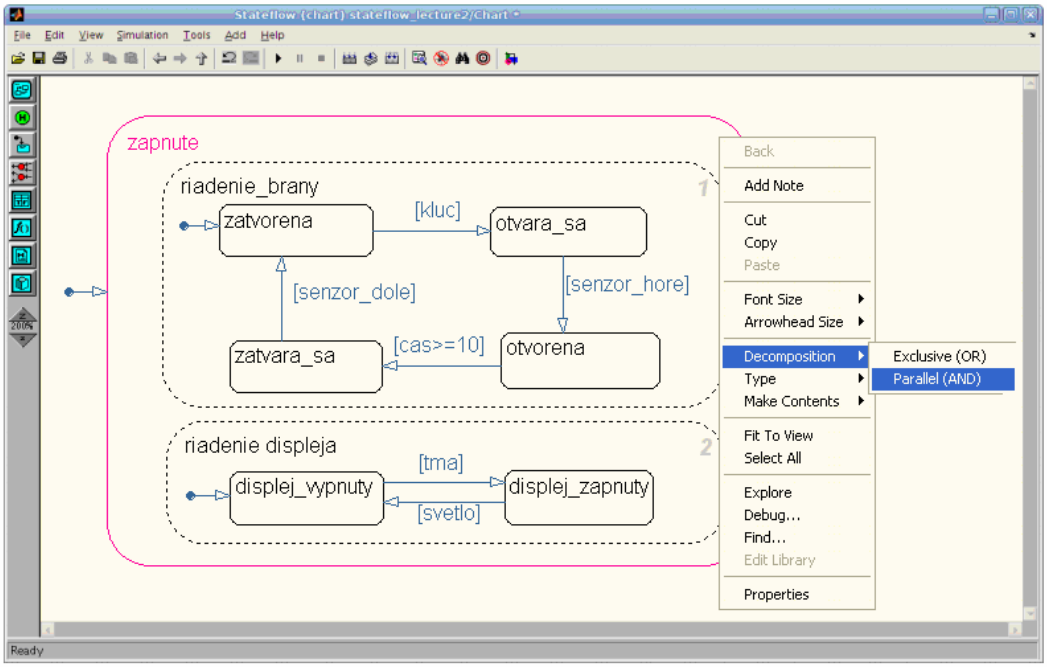
Výhodou takéhoto usporiadania je aj možnosť použitia „History junction“. Tento blok zabezpečí, že systém sa v hlavnom stave vráti do toho stavu, z ktorého predtým odišiel. Ak je história prázdna, systém vojde do defaultného stavu. Ak však história prázdna nie je, systém vojde do posledného navštíveného stavu. Teda ak brána opustila hlavný stav „zapnute“ v priebehu zatvárania, pri ďalšom vstupe do tohto stavu sa opäť dostane do procesu zatvárania.



Obr. 13 Pridanie histórie

1.4 Paralelná dekompozícia

Každý stav má svoju dekompozíciu, ktorá hovorí, aký typ podstavov stav obsahuje. Prvý typ dekompozície je exkluzívny stav, v tomto type môže byť aktívny iba jeden podstav. Druhým typom je paralelný stav, v ktorom môžu byť aktívne viaceré podstavy naraz. Využívame ho, ak sa v riadiacom systéme vykonávajú viaceré činnosti naraz. Tento typ dekompozície urýchľuje riadenia a umožňuje vytvoriť menej komplikovanú schému. Pridajme k riadeniu brány senzor svetla. Ak je v okolí tma, displej zariadenia má svietiť. Táto úloha sa vykonáva nezávisle od toho, v akom režime sa brána nachádza. Displej má svietiť, ak senzor zaznamená tmu, respektíve nezaznamená svetlo. V hlavnom stave vytvoríme dva stavy, ktoré sa budú riadiť nezávisle. Prvý stav bude riadenie brány a druhý riadenie displeja. Po vytvorení celej schémy klikneme pravým tlačidlom do materského stavu, v menu vyberieme položku „decomposition“ a následne označíme „paralel“. V okne Stateflowu sa zmenia plné čiary hraníc stavov pracujúcich súčasne na čiarkované.



Obr. 14 Vytvorenie paralelnej dekompozície

2 Lego Mindstorms NXT

Najinteligentnejší a najrozmanitejší robot. "Mozgom" tohto robota je výkonný 32-bitový mikroprocesor s dostatočným množstvom pamäti. . Pre komunikáciu s okolím a napríklad pre možnosť jeho programovania disponuje USB portom, ale tiež Bluetooth 2.0 pripojením. [3]

Robot obsahuje zvukové, svetelné a hmatové senzory, patriace k novej generácii robotiky. S pribaleným návodom, nenáročným na používanie alebo aj s vlastnou predstavivosťou, si môžu začiatočníci a aj pokročilí tvoriť humanoidov, vozidlá, zvieratá a vlastné podoby robota, schopné vykonať každý príkaz. So zabudovanou podporou bluetooth sa môže dokonca ovládať aj cez mobilný telefón. [3]

Zabudované svetelné senzory dokážu detekovať rôzne farby a intenzity svetla, zatiaľ čo zvukový senzor mu umožňuje reagovať na rozličné tóny. Jeho software je programovateľný, takže sa dá prispôsobiť vlastným predstavám.

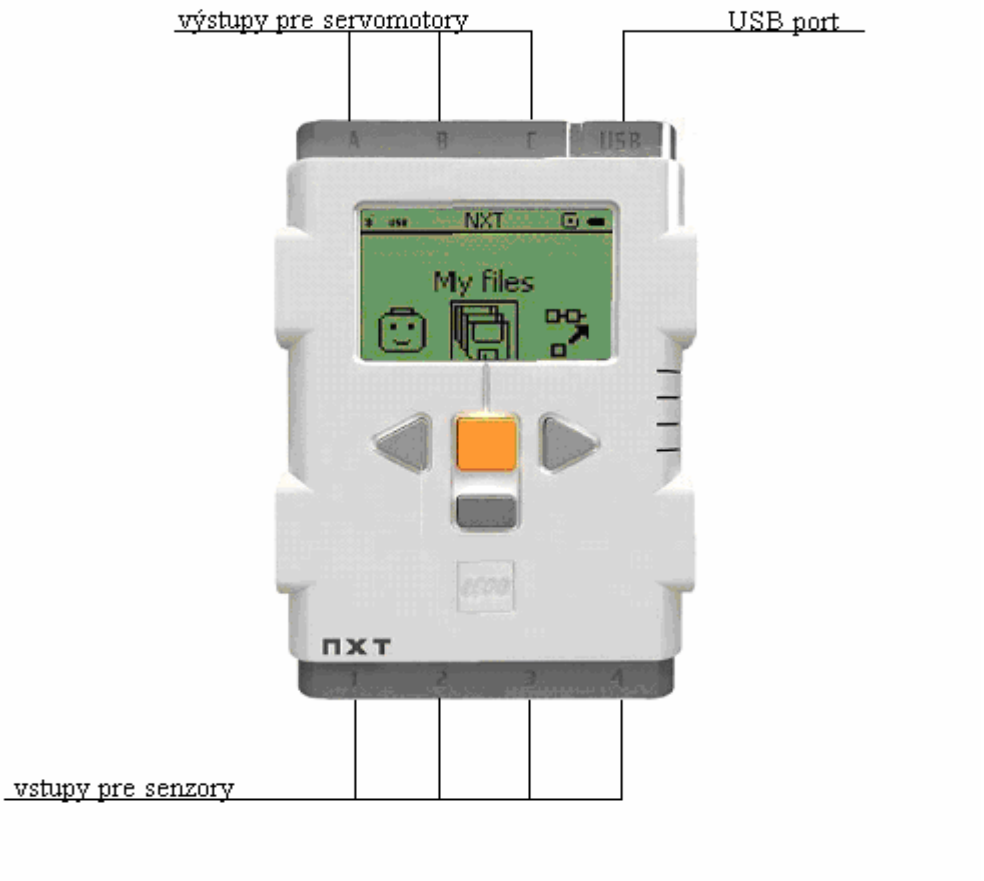
Obr. 15 Riadiaca kocka NXT so zapojenými akčnými členmi a senzormi

2.1 NXT kocka

NXT je mozgom Mindstorms robotov. Je to inteligentná, počítačom riadená lego kocka, ktorá realizuje rôzne operácie, ktoré dávajú robota do pohybu. [4]

Má tri výstupy, na ktoré sa pripájajú servomotory – výstup A, B a C a štyri vstupy pre pripojenie senzorov. Vstupy sú označené číslami 1, 2, 3 a 4. USB port sa nachádza na vrchu kocky, pri výstupoch. Do tohto portu sa pripája USB kábel, ktorým sa

nahrávajú programy z počítača do NXT kocky, alebo sa sťahujú data z robota do počítača. Taktiež sa dá na sťahovanie a nahrávanie použiť bezdrôtové Bluetooth pripojenie. Kocka NXT obsahuje aj zvukový výstup, ktorý je možné použiť pre signalizáciu, prípadne prezentáciu. [4]



Obr. 16 Riadiaca kocka NXT

Lego kocka NXT obsahuje štyri tlačidlá, ktoré môžu byť použité ako vstup do programu. Oranžové tlačidlo: On/Enter /Run sa používa na spustenie NXT kocky a bledosivé šípky sa používajú na výber a pohyb vľavo a vpravo v NXT menu. Tmavosivé tlačidlo slúži na vynulovanie a vrátenie sa späť. [4]

Kocka NXT je prioritne napájaná 6 AA batériami. Preto pri písaní programov, ktoré pracujú dlhšiu dobu, je potrebné vedieť zistiť stav, v akom sa batérie nachádzajú. Tento stav nám signalizuje ikona baterky na displeji. [5]

2.1.1 Technické parametre

- 32-bit ARM7 mikrokontrolór
- 256 Kbytes FLASH, 64 Kbytes RAM
- 8-bit AVR mikrokontrolór
- 4 Kbytes FLASH, 512 Byte RAM
- Bezdrôtová Bluetooth komunikácia (Bluetooth trieda II V2.0 vyhovujúca)
- Plnerýchlostný USB port (12 Mbit/s)
- 4 vstupné porty, 6-článkový kábel digitálnej platformy (jeden port obsahuje IEC 61158 Type 4/EN 50 170 kompatibilne rozšírený port pre budúce použitie)
- 3 výstupné porty, 6- článkový kábel digitálnej platformy
- 100 x 64 pixel LCD grafický displej
- Reproduktor - 8 kHz zvuková kvalita. Zvukový kanál s 8-bitovým rozlíšením a 2-16 KHz vzorkovací kmitočet.
- Napájanie: 6 AA batérie [4]

2.2 Senzory

2.2.1 Dotykový senzor

Dotykový senzor dáva robotovi možnosť dotyku. Zisťuje, kedy je senzor stlačený a kedy uvoľnený. [6] Dotykový senzor nadobúda hodnoty 1 pri stlačení alebo 0 pri uvoľnení senzora. Na základe toho môže vykonávať činnosti, ktoré podliehajú danému stlačeniu alebo uvoľneniu senzora.

Dotykový senzor sa môže použiť pri zdvíhaní vecí, napríklad robotické rameno je vybavené dotykovým senzorom, ktorý umožňuje robotovi poznať, či drží alebo nedrží niečo v ruke. Alebo sa môže použiť na vykonávanie príkazov pre robota. Stlačený senzor môže uviesť robota do pohybu alebo ho prinútiť hovoriť, zatvoriť dvere, zapnúť televízor a podobne. [6]

Obr. 17 Dotykový senzor

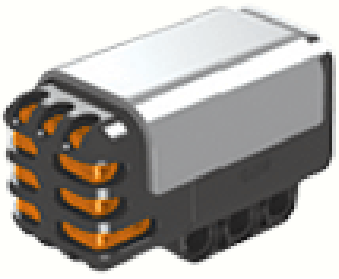
2.2.2 Zvukový senzor

Zvukový senzor môže zaznamenávať nenastavené, ale aj presne nastavené decibely. [8] Decibel (*dB*) je meracia jednotka, udávajúca pomer medzi dvomi veličinami. Často sa používa ako vyjadrenie miery intenzity zvuku. [7] Meria výkon, útlm, hladinu intenzity zvuku alebo akustický tlak.

Pri zaznamenávaní nastavených decibelov (dBA) je citlivosť senzora prispôsobená citlivosti ľudského ucha. Inak povedané, sú to zvuky, ktoré je ucho schopné počuť. V zaznamenávaní štandardných (nenastavených) decibelov (dB) sú všetky zvuky merané s rovnakou citlivosťou. Tieto zvuky môžu zahŕňať pre ľudské ucho veľmi vysoké, ale aj veľmi nízke tóny. [8]

Zvukový senzor môže merať akustický tlak približne do 90 dB, čo zodpovedá hluku bežiacej sekačky na trávu. Hladiny akustického tlaku sú veľmi komplikované, preto zvukový senzor zaznamenáva hodnoty na displeji v percentách. Napr.

- 4-5% je ako tichá obývačka
- 5-10% môže byť ako keď niekto hovorí z určitej vzdialenosti
- 10-30% je normálna konverzácia blízko senzora
- 30-100% sú ľudia, ktorí kričia alebo hudba, ktorá hrá pri vysokej hlasitosti [8]

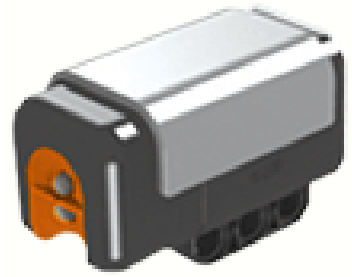


Obr. 18 Zvukový senzor

2.2.3 Svetelný senzor

Svetelný senzor je jedným z dvoch senzorov, ktoré umožňujú robotovi „vidieť“, čiže rozoznávať svetlo a tmu (druhým takým senzorom je ultrazvukový senzor). Svetelný senzor sníma intenzitu svetla v miestnosti a meria svetelnú intenzitu farebných povrchov. [9] K zaznamenávaniu farieb sa môže k svetelnému senzoru použiť aj prisvetlenie. V Stateflowe sa hodnoty jednotlivých farieb pohybujú v rôznych rozpätiach. Čím je farba tmavšia, tým sú hodnoty svetelného senzora menšie. Hodnoty čiernej farby sa pohybujú okolo 20 – 50 a hodnoty bielej farby sú približne 500 – 600.

Svetelný senzor sa môže použiť napr. pri výrobe robota ako poplachového alarmu proti vlámaniu. Keď zlodej zapne svetlo v izbe, robot zareaguje na ochranu vášho majetku. Taktiež robot môže pomocou tohto senzora triediť veci podľa farieb. [9]



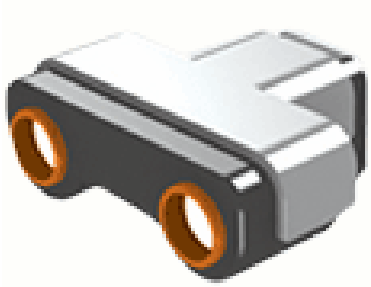
Obr. 19 Svetelný senzor

2.2.4 Ultrazvukový senzor

Ultrazvukový senzor je jeden z dvoch senzorov, ktoré umožňujú robotovi „vidieť“. Vďaka tomuto senzoru robot vidí, zachytáva objekty alebo môže slúžiť pri vyhýbaní sa prekážkam, meraní vzdialenosti a zachytávaní pohybov. [10]

Ultrazvukový senzor meria vzdialenosť v centimetroch a v palcoch. Dokáže zachytiť vzdialenosť od 0 po 255 cm s presnosťou +/- 3 cm. Funguje na rovnakom vedeckom princípe ako netopiere. To znamená, že meria vzdialenosť na základe výpočtu času, ktorý meria dobu od vyslania zvukovej vlny, ktorá narazí na predmet

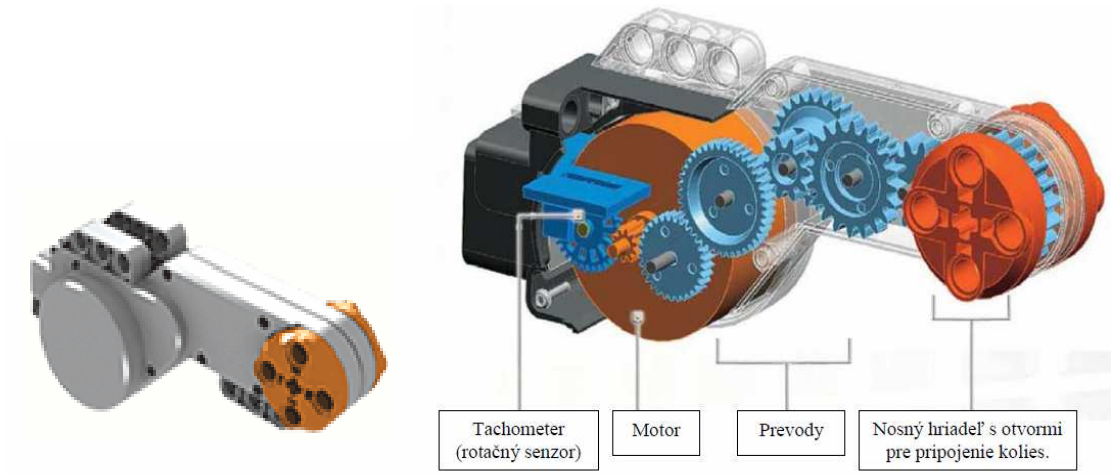
a vráti sa - rovnako ako echo. Obrovské predmety s tvrdým povrchom sú najlepšie na čítanie pre ultrazvukový senzor. Predmety vyrobené z mäkkej tkaniny alebo ktoré sú oblé (napr. ako lopta) alebo veľmi tenké a malé môžu byť horšie pre zachytávanie senzora. Dva alebo viac ultrazvukových senzorov pracujúcich v tej istej miestnosti sa môžu navzájom ovplyvňovať a prerušiť každé druhé čítanie. [10]



Obr. 20 Ultrazvukový senzor

2.3 Servomotory

Robot je vybavený troma servomotormi, ktoré mu umožňujú pohyb. Každý servomotor má v sebe zabudovaný rotačný senzor, ktorý umožňuje presne kontrolovať pohyby robota. Rotačné snímače merajú otáčky motora v stupňoch alebo v plných otáčkach s presnosťou +/- jeden stupeň. Jedna otáčka sa rovná 360 stupňom, takže ak sa nastaví motor na otáčanie o 180 stupňov, jeho výstupný hriadeľ urobí pol otáčky. Zabudovaný rotačný senzor v každom motore sa môže nastaviť na rôznu rýchlosť. [11]



Obr. 21 Interaktívny servomotor

3 Úloha 1

3.1 Riadenie brány

Cieľom je riadenie brány, ktorá reaguje na rôzne užívateľské kľúče. Na začiatku je brána zatvorená, čo zodpovedá uhlu menšiemu ako 0 stupňov. Ak sa brána otvára a uhol nadobudne hodnotu väčšiu ako 200 stupňov, automaticky sa zastaví a čaká na príkaz na zatvorenie - stlačenie tlačidla. V prípade registrovania pohybu fotobunkou počas zatvárania, sa brána znovu začne otvárať.

K dispozícii máme štyri rôzne kľúče: základný, víkendový, uzamykací a servisný. Po vložení jedného z nich a stlačení tlačidla sa brána dostáva do zodpovedajúceho režimu.

Pri riešení úlohy sme použili dotykový a ultrazvukový senzor a dva servomotory. Dotykový senzor predstavuje tlačidlo, ktorým otvárame a zatvárame bránu. Ultrazvukový zodpovedá fotobunke. Jednotlivé kľúče sú realizované na základe hodnôt uhla, ktorý určuje servomotor. Pohyb brány zabezpečuje druhý servomotor.

Celková úloha sa dá riešiť pomocou čiastkových úloh, ktoré sú detailnejšie popísané v ďalších podkapitolách.

3.1.1 Definovanie kľúčov

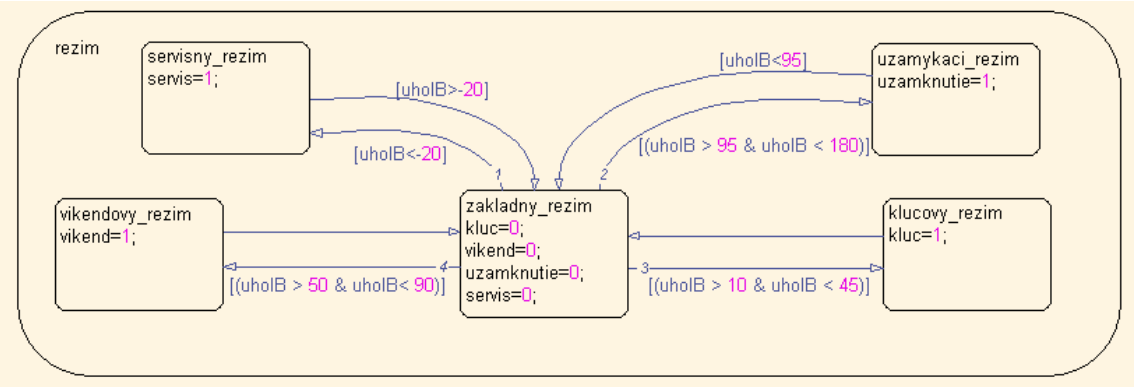
Prvý krok, ktorý je potrebný na začatie úlohy, je zadefinovanie kľúčov pomocou servomotora. Vytvoríme stupnicu s hodnotami od -20 do 180 stupňov a rozdelíme ju na štyri oblasti. Každá oblasť zodpovedá jednému kľúču.

- **Konštrukcia**

Na servomotor sme pripojili otočnú páku, ktorá nám uľahčí otáčanie nosného hriadeľa a tým aj nastavenie uhla. Servomotor sa pripojí káblom k NXT kocke do portu B. Je dôležité zapojiť ho do správneho portu, pretože pri programovaní v matlabe je potrebné definovať konkrétny port, na ktorý je servomotor napojený. Ak by sme ho zapojili do iného portu, tak by robot nefungoval. Tento akčný člen využívame v celom riešení úlohy.

- **Riešenie v Stateflowe**

V simulinku musíme pripojiť vstup – „uholB“. Je potrebné vložiť lokálne premenné- „kluc“, „vikend“, „uzamknutie“ a „servis“.



Obr. 22 Stavový diagram pre definovanie kľúčov

1. *Základný režim*

Vytvorili sme stav, ktorý predstavuje základný režim, keď nie je vložený žiadny kľúč a brána je zatvorená. Hodnoty všetkých premenných v tomto stave sú nulové a po každom vrátení sa do tohto stavu sa nastavia na nulu.

2. *Kľúčový režim*

Podmienkou pre prechod do kľúčového režimu je rozpätie „uhluB“ medzi 10 – 45 stupňov. Ak je táto podmienka splnená, brána prejde do kľúčového režimu a premennej „kluc“ sa priradí hodnota 1. Tento stav automaticky prejde do základného režimu. A cyklí sa dovtedy, až kým sa zmení podmienka.

3. *Víkendový režim*

Ak „uholB“ nadobudne hodnoty od 50 do 90 stupňov, podmienka pre prechod do víkendového režimu bude splnená a tento stav sa stane aktívnym. Premenná „vikend“ sa nastaví na hodnotu 1. Tento stav opäť automaticky prejde do základného režimu. Cyklí sa dovtedy, až kým sa zmení podmienka.

4. Uzamykací režim

Pri hodnote „uhluB“ od 95 do 180 stupňov sa nastaví uzamykací režim. Premenná „uzamknutie“ nadobudne hodnotu 1. Túto hodnotu má dovtedy, kým „uholB“ nebude menší ako 95 stupňov, čím sa splní podmienka pre výstup z tohto stavu. Pri splnení tejto podmienky brána prejde do základného stavu, kde premenná „uzamknutie“ sa nastaví na hodnotu 0.

5. Servisný režim

Pre prechod do servisného režimu musí byť splnená podmienka, že „uholB“ je menší ako -20 stupňov. Premenná „servis“ má hodnotu 1. Ak bude „uholB“ väčší ako -20 stupňov, prejde do základného režimu, kde premenná „servis“ bude mať hodnotu 0.

Bránu budeme riadiť pomocou dvoch hlavných stavov, ktoré budú bežať paralelne, preto týchto 5 stavov je vložených do jedného stavu s názvom „režim“.

3.1.2 Otvorenie a zatvorenie brány

Brána sa má po vložení kľúča začať otvárať. Otvára sa dovtedy, kým nedosiahne uhol zodpovedajúci otvoreniu brány, čiže kým brána nebude vo vertikálnej polohe. Po dosiahnutí definovaného uhla pre otvorenie, ostane brána otvorená, až kým sa nestlačí tlačidlo. Po stlačení tlačidla sa brána začne zatvárať. Zatvorí sa po dosiahnutí uhla, ktorý predstavuje vodorovnú polohu brány. Brána sa zatvorí a ostane v stave „zatvorena“ až do ďalšieho vloženia kľúča.

- **Konštrukcia**

Na riešenie úlohy zostavíme lego robota. Pripojíme všetky senzory, ktoré pripájame do NXT kocky ako vstupy. K dispozícii je len dotykový senzor. Výstup z tohto senzora vložíme do postu 1 na NXT kocke. Akčným členom je servomotor, ovládajúci rameno brány. Pripojíme ho do portu A.

Aby sa rameno brány neotváralo veľmi rýchlo, nepripevníme ho rovno na nosný hriadeľ interaktívneho servomotora, ale vytvoríme prevod. Na servomotor vložíme menšie ozubené koliesko, do ktorého zapadá väčšie ozubené koliesko. Dosiahneme tým, že rameno brány sa otvorí o uhol menší ako uhol, o ktorý sa otočí servomotor. To vysvetľuje aj hodnoty uhlov pre zatvorenie a otvorenie brány. Kým servomotor sa otočí

o uhol 200 stupňov, rameno brány sa otočí približne o 90 stupňov. Uhol definovaný pre otvorenie brány je potom 200 stupňov.

Do NXT kocky pripojíme ešte servomotor určený na definovanie kľúčov. Ten vložíme do portu B.

Obr. 23 Obrázok lego robota zostaveného pre otváranie a zatváranie brány

- **Riešenie v Stateflowe**

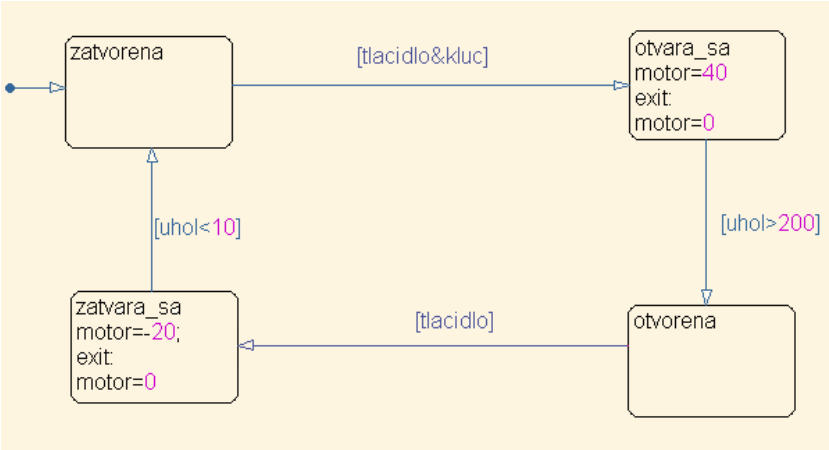
Ak má brána vykonávať funkciu otvárania a zatvárania, potrebujeme vytvoriť štyri stavy. Pri spustení schémy v Stateflowe vidíme, v ktorom stave sa brána nachádza. Každý stav je potrebné pomenovať a napísať príkazy, ktoré sa majú v danom stave vykonať.

Prvý aktívny stav predstavuje zatvorenú bránu, čo v diagrame symbolizuje šípka s guličkou na konci. Brána sa nachádza v základnej polohe, ktorú predstavuje hodnota uhlu 0 stupňov. Na to, aby sa brána začala otvárať, je potrebné vložiť kľúč. Samotný kľúč predstavuje hodnota „uhluB“, ktorá musí byť v rozpätí od 10 po 45 stupňov. Ak je hodnota „uhluB“ v požadovanom intervale, priradí sa premennej „kluc“ hodnota 1. Vloženie kľúča predstavuje stlačenie tlačidla.

Ďalší stav, do ktorého sa brána dostane po splnení prechodových podmienok, je stav „otvara_sa“. V tomto stave je potrebné zadefinovať rýchlosť motora, akou sa brána bude otvárať. Zvolíme hodnotu 40. Brána sa dá do pohybu a zdvíha sa, až kým nie je úplne otvorená. Úplnú otvorenosť charakterizuje hodnota „uhluB“ väčšia ako 200

stupňov. Pri vyšších rýchlostiach otvárania sa môže stať, že senzor nestihne rýchlo zareagovať na požadovanú hodnotu uhla pre otvorenie a brána sa otvorí viac ako je potrebné. Aby sa predišlo možným haváriám či poškodeniu systému, na začiatku volíme radšej nižšiu rýchlosť motora. Pri odchode z tohto stavu motor získa hodnotu 0, to znamená, že sa zastaví a brána prejde do stavu „otvorena“.

V stave „otvorena“ systém čaká, až kým nebude znovu stlačené tlačidlo, ktoré dá pokyn na zatvorenie brány. Ak je hodnota dotykového senzoru 1, aktívnym sa stane posledný stav nazvaný „zatvara_sa“. Tu treba opäť nastaviť rýchlosť motora, ale so záporným znamienkom, pretože brána sa má pohybovať opačným smerom ako pri otváraní. Rýchlosť motora je nastavená na -20. Brána sa začne pohybovať opačným smerom a zastaví sa, keď „uhol“ nadobudne hodnotu menšiu ako 10 stupňov. Splnením poslednej podmienky sa systém dostane do počiatočného stavu, v ktorom ostane, kým nebudú splnené podmienky pre prechod do ďalšieho stavu.



Obr. 24 Otvorenie a zatvorenie brány

3.1.3 Fotobunka

Rozšírenie predchádzajúcej úlohy je zavedenie fotobunky do systému. Ak pri zatváraní brány fotobunka zaregistruje pohyb v jej okolí, brána sa začne znovu otvárať bez toho, aby bolo znovu stlačené tlačidlo a vložený kľúč a bez toho, aby brána bola vo východiskovej polohe.

- **Konštrukcia**

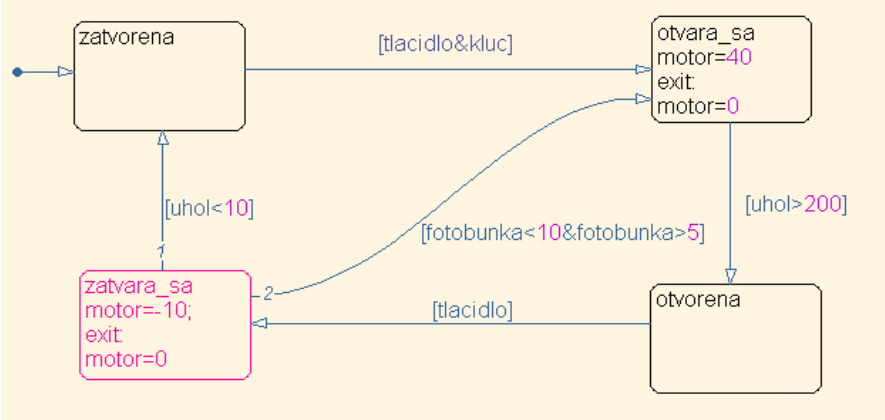
Fotobunku predstavuje ultrazvukový senzor, ktorý sa pripojí ako vstup do NXT kocky do portu 3.

Obr. 25 Obrázok lego robota zostaveného na riadenie brány s pomocou fotobunky

- **Riešenie v Stateflowe**

Úloha sa dá riešiť tak, že do stavu „zatvara_sa“ pripojíme podmienku, ktorá hovorí, že ak fotobunka nadobudne hodnoty v rozpätí od 5 po 10, tak systém prejde automaticky do stavu „otvara_sa“ a brána sa začne znovu otvárať. V Stateflowe pridáme nový vstup do systému s názvom „fotobunka“.

V tomto prípade sú k stavu „zatvara_sa“ pridané dve podmienky. Aby tieto dve podmienky neboli vykonávané naraz, treba im určiť poradie. Prioritnou podmienkou je, že brána sa zatvára, až kým uhol nebude menší ako 10 stupňov. Podmienka „fotobunka“ je druhá v poradí. Ak by bola táto podmienka splnená, tak systém zareaguje a prejde hneď do stavu „otvara_sa“. Brána sa začne otvárať.



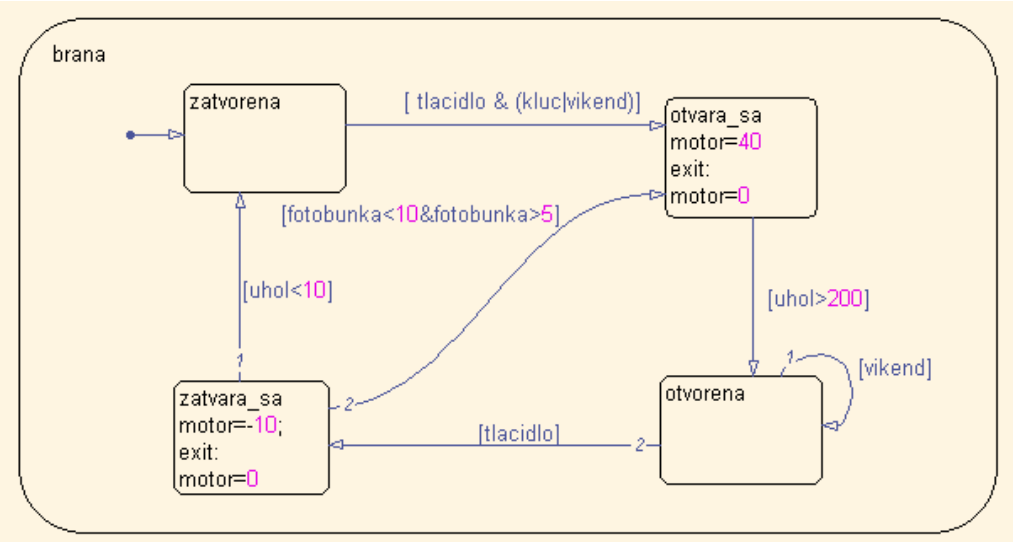
Obr. 26 Pridanie fotobunky

3.1.4 Víkendový režim

Víkendový režim znamená, že ak bol do brány vložený víkendový kľúč, brána sa začne otvárať a ostane otvorená, až kým nebude tento kľúč vytiahnutý a nebude stlačené tlačidlo.

- **Riešenie v Stateflowe**

Stavy a podmienky pre otvorenie brány sú rovnaké ako v predošlých úlohách, len do prvej podmienky, pri prechode zo stavu „zatvorena“ do stavu „otvara_sa“, je pridaná podmienka „vikend“. Do stavu „otvorena“ pridáme ďalšiu podmienku. Ak má premenná „vikend“ hodnotu 1, brána prejde do stavu „otvorena“. Systém sa bude cykliť, kým premenná „vikend“ nebude mať hodnotu 0, po stlačení tlačidla sa brána začne zatvárať. Podmienka pre prechod do víkendového režimu má vyššiu prioritu ako podmienka pre zatváranie brány. Ak by hodnoty premennej „vikend“ a „tlačidla“ nadobudli naraz hodnotu 1, prednosť by mal víkendový režim.



Obr. 27 Pridanie víkendového režimu

3.1.5 Uzamykací režim

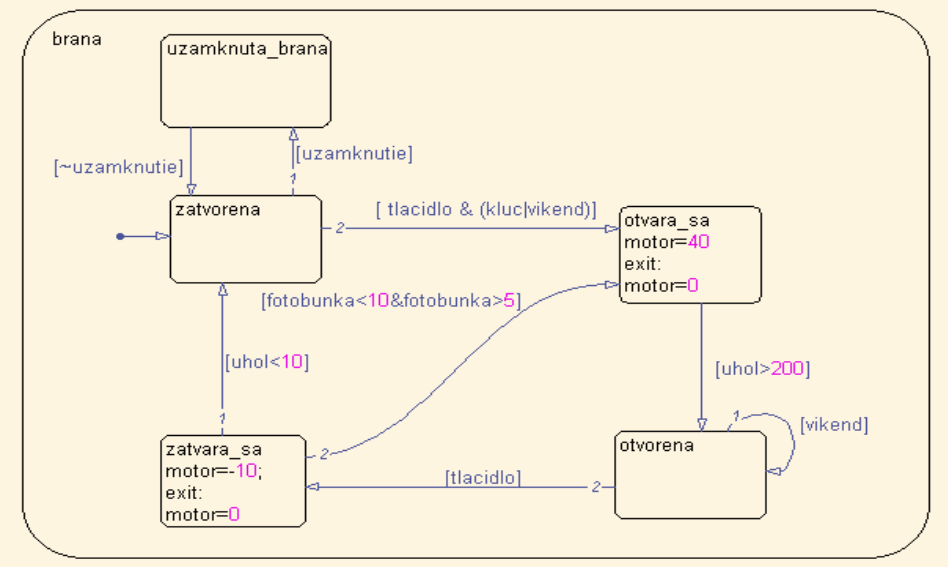
Ak do systému vložíme uzamykací kľúč, brána ostane zatvorená a nereaguje na žiadne iné kľúče, až kým tento uzamykací kľúč nie je znovu vytiahnutý.

- **Riešenie v Stateflowe**

Na riešenie úlohy treba vytvoriť nový stav „uzamknuta_brana“. Tento stav je aktívny, keď hodnota premennej „uzamknutie“ bude 1. Táto premenná nadobúda túto

hodnotu, keď sa do brány vloží uzamykací kľuč. Tento kľúč definuje podmienka, že hodnota „uhluB“ sa musí nachádzať v rozpätí od 95 po 180 stupňov.

Keď je táto podmienka splnená, systém prejde do nového stavu „uzamknuta_brana“ a zotrvá tam. Systém v tomto stave nereaguje na žiadne iné kľúče ani na tlačidlo, až kým hodnota „uhluB“ nebude menšia ako 95 stupňov. Toto zodpovedá vytiahnutiu kľúča z brány, hodnota „uzamknutia“ je 0. V tom momente systém vstupuje opäť do stavu „zatvorena“ a začína reagovať na ostatné užívateľské kľúče.



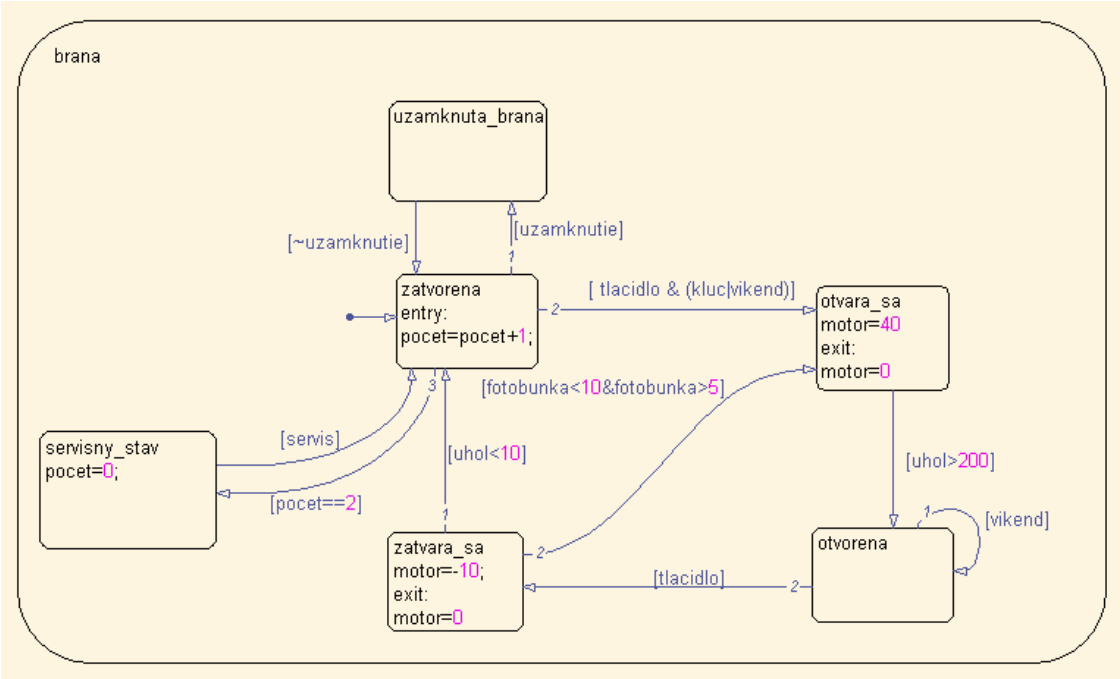
Obr. 28 Pridanie uzamykacieho režimu

3.1.6 Servisný režim

Posledná úloha je, aby sa brána dostala po určitom počte otvorení a zatvorení do servisného režimu, v ktorom sa vykonávajú údržbárske práce na bráne.

- **Riešenie v Stateflowe**

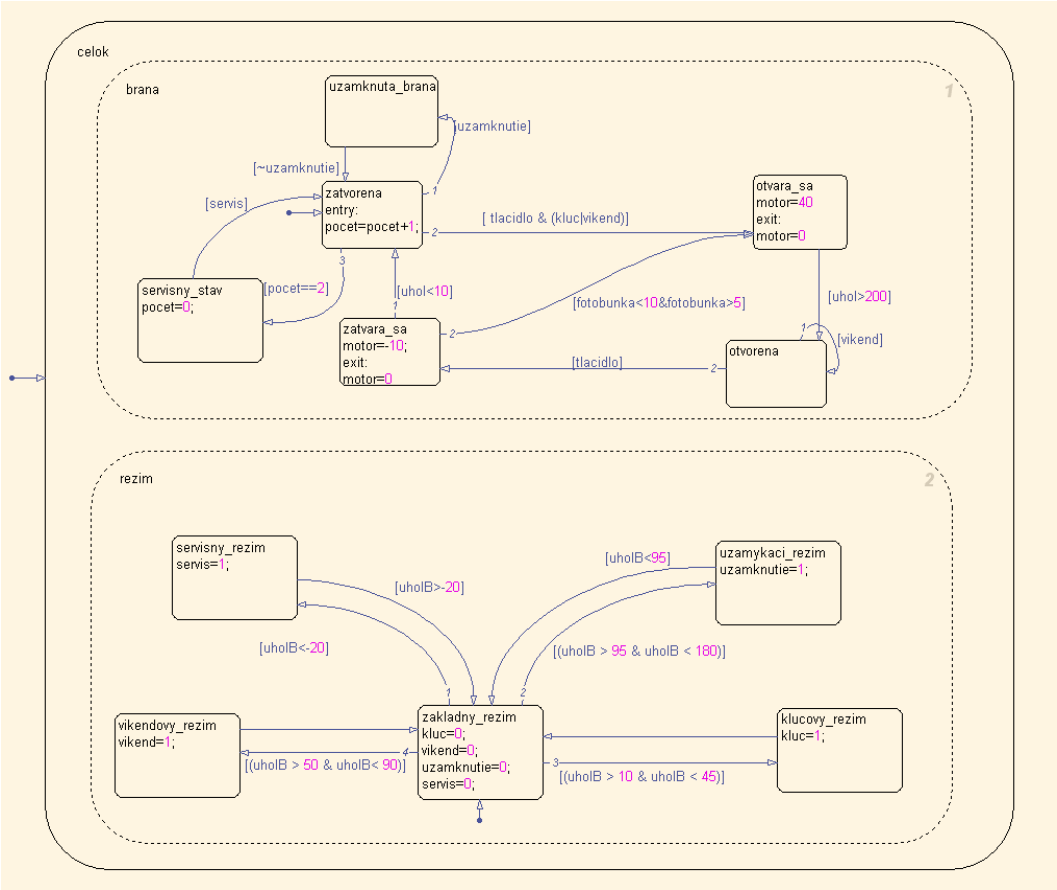
Do systému musíme zaviesť počítadlo, ktoré bude počítať zatvorenia brány. Do východiskového stavu „zatvorena“ treba vložiť príkaz na počítanie. Keď sa systém dostane do tohoto stavu, vykoná sa príkaz a „pocet“ sa zvýši o jeden. Keď dosiahne požadovanú hodnotu, v tomto prípade, keď sa „pocet“ rovná 2, prejde do stavu „servisny_stav“. Tu sa počítadlo automaticky vynuluje. Brána ostane v „servisnom“ režime, kým hodnota premennej „servis“ nebude 1, kým nebude vložený servisný kľúč. Po splnení tejto podmienky prechádza do stavu „zatvorena“.



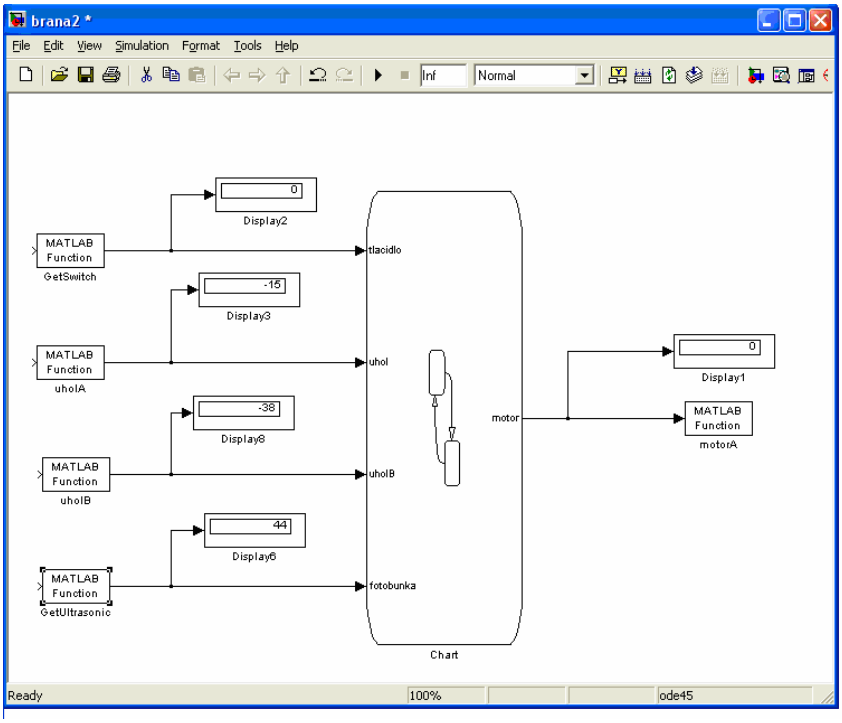
Obr. 29 Pridanie servisného režimu

3.1.7 Celková úloha

Definovanie kľúčov a prechody do rôznych stavov sa vykonávajú paralelne. Preto vložíme diagram pre kľúče do väčšieho stavu s názvom „režim“ a diagram pre riadenie brány do stavu „brana“. Tieto stavy vložíme do jedného hlavného stavu s názvom „celok“, nastavíme paralelné vykonávanie stavov a pridáme symbol pre prvý aktívny stav k stavu „celok“.



Obr. 30 Stavový diagram riadenia brány



Obr. 31 Simulinková schéma riadenia brány

4 Úloha 2

4.1 Meranie reakcií

Úloha spočíva v meraní ľudských reakcií na určitý podnet. V našom prípade ide o odozvu na pohyb ramena robota, ktorá sa musí vykonať v stanovenom čase. Po stlačení tlačidla začne bežať čas. Po uplynutí vygenerovaného času sa rameno začne hýbať. Po zaregistrovaní pohybu ramena treba čo najrýchlejšie tliesknuť. Ak sa tliesknutie vykoná v čase kratšom ako je stanovený, zaráta sa úspešný pokus. Ak sa vykoná pred pohybom ramena alebo po ukončení pohybu, zaráta sa neúspešný pokus.

4.1.1 Konštrukcia robota

K splneniu úlohy musíme zostaviť robota s pohyblivým ramenom. Ramenom hýbe servomotor zapojený do NXT kocky do výstupu A. Servomotor určuje aj uhol potrebný na zastavenie ramena. Aby sa rameno nehýbalo príliš rýchlo vytvoríme prevod z menšieho ozubeného kolieska a z väčšieho ozubeného kolieska, ktoré do seba zapadajú. Zabezpečíme tým, že otočenie servomotora otočí rameno o menší uhol ako bez prevodu.

Na zachytávanie zvuku potrebujeme zvukový senzor, ktorý pripojíme ako vstup do portu 2. Tlačidlo predstavuje dotykový senzor zapojený do vstupu 1. Označenia portov, do ktorých sme akčné členy a senzory pripojili, sa udávajú aj v schéme v simulinku. Zadané porty v simulinku musia byť rovnaké ako ich reálne zapojenie. Inak robot nebude vykonávať požadovanú funkciu.

Obr. 32 Obrázok lego robota zostaveného pre úlohu merania reakcií

4.1.2 Pohyb ramena

Rameno je v základnej horizontálnej polohe. Do pohybu sa dá po stlačení tlačidla a pohybuje sa, až kým nedosiahne zvolenú hodnotu uhla. Po dosiahnutí tejto hodnoty sa začne pohybovať opačným smerom a vráti sa do pôvodnej polohy.

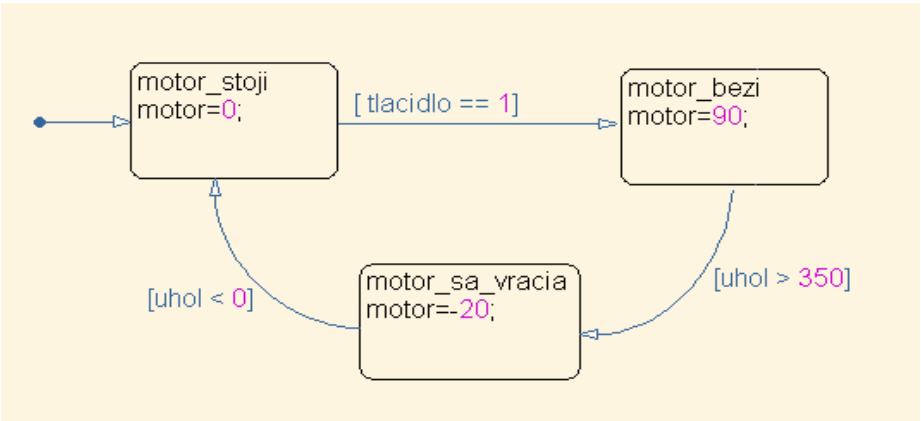
- **Riešenie v Stateflowe**

V Stateflow editore vytvoríme tri stavy a zadefinujeme všetky vstupy, výstupy aj lokálne premenné. Vstupmi sú uhol, počítaný servomotorom a hodnota dotykového senzora, reprezentovaná premennou „tlacidlo“. Výstupom je rýchlosť motora, premennú nazveme „motor“.

V prvom stave je motor vypnutý a rameno je v pôvodnej polohe. Názov stavu je „motor_stoji“ a rýchlosť motora je 0, čo zadefinuje ako „motor=0“. Podmienkou pre prechod do ďalšieho stavu je stlačenie tlačidla.

Ak je hodnota premennej „tlacidlo“ 1, aktívnym stavom sa stane stav „motor_bezi“. Rameno sa začne pohybovať smerom nahor. Rýchlosť pohybu má byť čo najvyššia, ale taká, že pri rýchlej reakcii na pohyb stihneme tliesknuť a zvukový senzor zaznamená zvuk. Volíme preto rýchlosť 90. Rameno sa hýbe po hodnotu „uhlu“ 350. Hodnota uhlu je prispôsobená už spomínanému prevodu servomotora. V skutočnosti sa rameno otočí o menší uhol. Pri vyšších rýchlostiach volíme menší uhol ako podmienku na zastavenie ramena, pretože servomotor pomalšie zareaguje na dosiahnutie tohto uhla.

Po prekročení veľkosti zadefinovaného uhla sa motor začne vracať do pôvodnej polohy. Aktívnym sa stáva stav „motor_sa_vracia“. Motor sa hýbe opačným smerom, preto rýchlosť motora je -20. Táto rýchlosť môže byť nižšia, servomotor tak rýchlejšie zaregistruje uhol definovaný na zastavenie. Ak bude hodnota uhlu menšia ako nula, motor sa zastaví a systém sa dostane do prvého stavu.



Obr. 33 Pohyb ramena

4.1.3 Pridanie funkcie „hodiny“

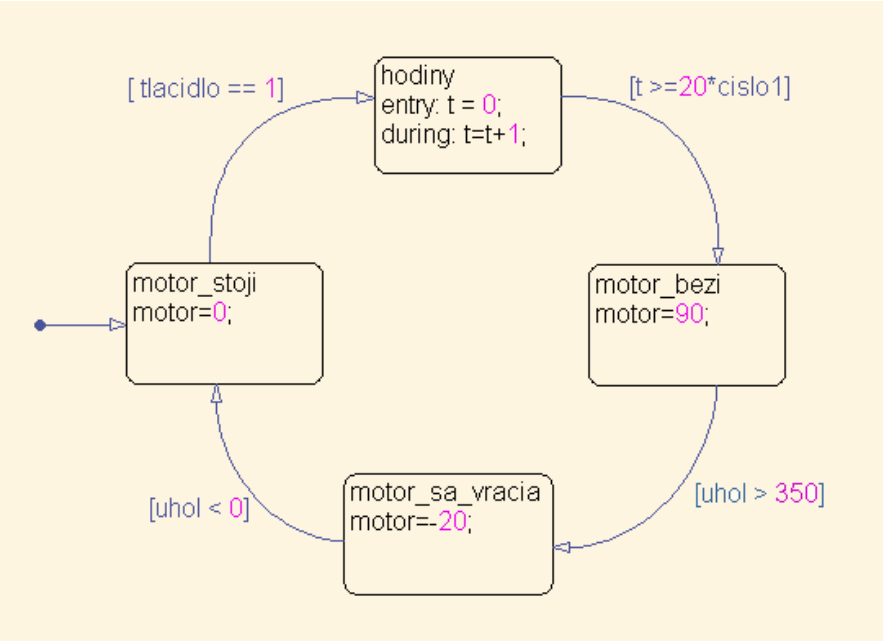
Treba zabezpečiť, aby sa rameno začalo hýbať po náhodnom čase od stlačenia tlačidla. Testovaná osoba tak nebude vedieť, kedy sa rameno začne hýbať.

- **Riešenie v Stateflowe**

Do simulinkovej schémy pripojíme nový vstup, v ktorom sa budú generovať náhodné čísla. Zadefinujeme ich ako premennú „cislo1“. V Stateflow editore vytvoríme nový stav „hodiny“ a pridáme lokálnu premennú „t“.

Pri vstupe do stavu „hodiny“ sa priradí premennej „t“ hodnota 0. Počas celého trvania tohto stavu sa hodnota premennej „t“ zvyšuje o jednotku, v každej sekunde bude hodnota tejto premennej vyššia o jednotku. Systém z tohto stavu vystúpi, ak hodnota premennej „t“ bude vyššia, nanajvýš rovná dvadsaťnásobku náhodne vygenerovaného čísla, teda ak „t >=20*cislo1“. Touto podmienkou zabezpečíme pohnutie ramena po náhodnom čase od stlačenia tlačidla, keďže „cislo1“ je náhodne generované číslo a je vždy rôzne.

Stav „hodiny“ prepojíme s prvým stavom „motor_stoji“. Ak sa podmienka prechodu z tohto stavu splní, automat prejde do stavu hodiny, kde čaká na uplynutie náhodného času. Po splnení podmienky pre východ zo stavu hodiny sa rameno dá do pohybu.



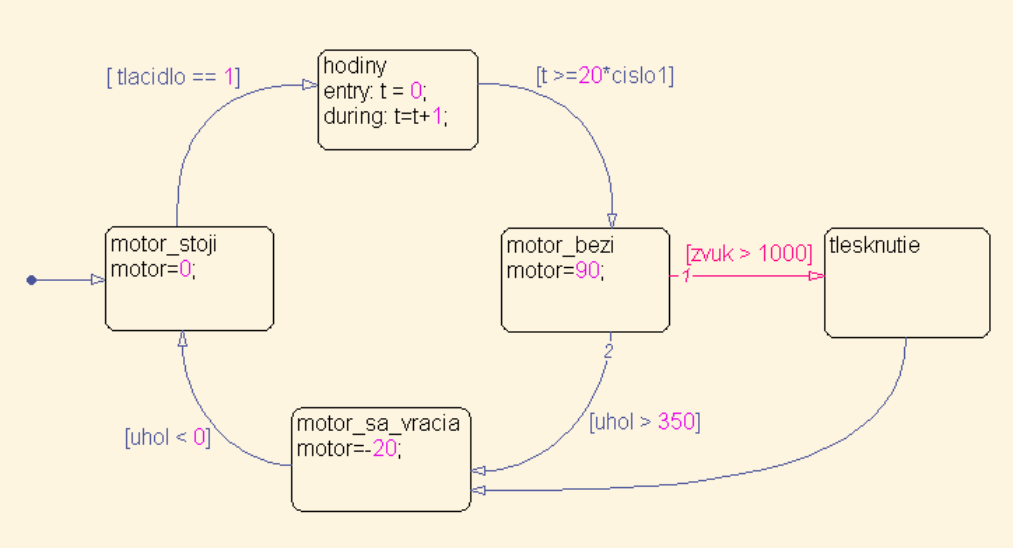
Obr. 34 Pridanie funkcie hodiny

4.1.4 Zaznamenania úspešného a neúspešného pokusu

Pri zaregistrovaní pohybu ramena robota, treba tliesknuť. Úspešný pokus sa zaráta vtedy, ak sa tliesknutie vykonalo počas pohybu ramena, teda kým uhol nebol väčší ako 350. Ak zvukový senzor zaznamenal tliesknutie pred pohybom ramena alebo po ukončení pohybu, zaráta sa neúspešný pokus.

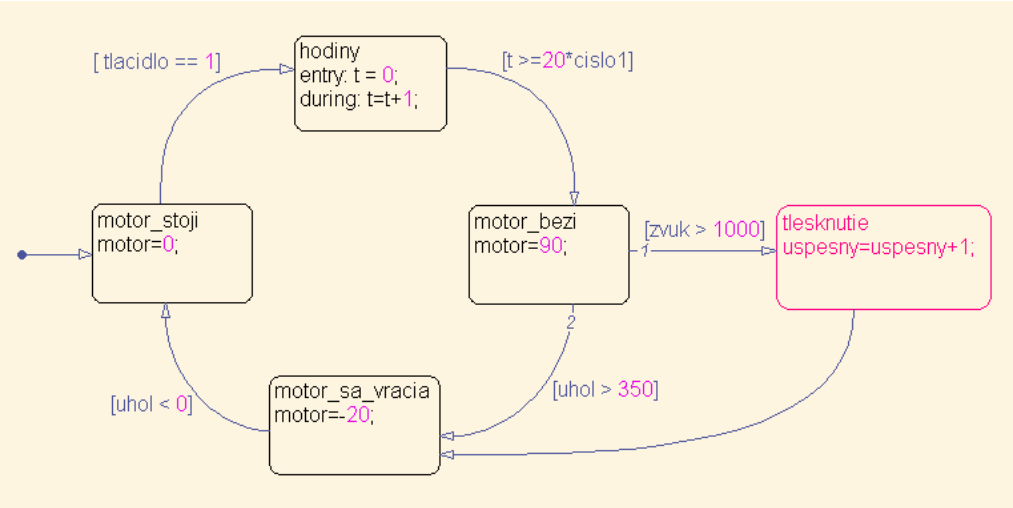
- **Riešenie v Stateflowe**

Do simulinkovej schémy musíme pripojiť vstup zo zvukového senzora, ktorý sa bude ukladať do premennej „zvuk“. Pridáme aj lokálne premenné „uspesny“ a „neuspesny“, ktoré bude rátať počet úspešných a neúspešných pokusov. Úspešný pokus je ten, keď sa tlieskne pri pohybe ramena. Teda ak je aktívny stav „motor_bezi“ a zvukový senzor zaznamená zvuk. Vytvoríme nový stav „tlesknutie“. Do tohto stavu sa systém môže dostať zo stavu „motor_bezi“. Podmienkou pre prechod je, že hodnota premennej zvuk je vyššia ako 1000. Zo stavu „tlesknutie“ sa systém automaticky dostane do stavu „motor_sa_vracia“ a rameno sa začne vracať do pôvodnej polohy.



Obr. 35 Zaznamenanie tlesknutia

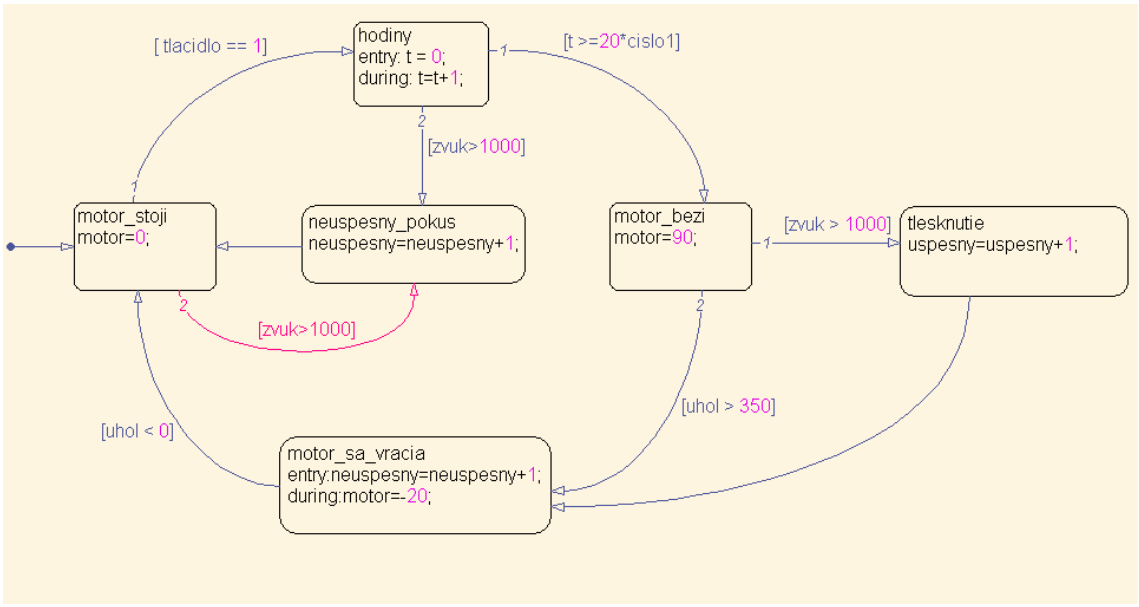
Pri vstupe do tohto stavu sa zaráta úspešný pokus, hodnota premennej „uspesny“ sa zvýši o jednotku. Do stavu pridáme príkaz „uspesny = uspesny +1“. Keďže systém do tohto stavu vstúpi a hneď vystúpi, príkaz nemusíme písať do „entry“ príkazu.



Obr. 36 Zarátanie úspešného a neúspešného pokusu

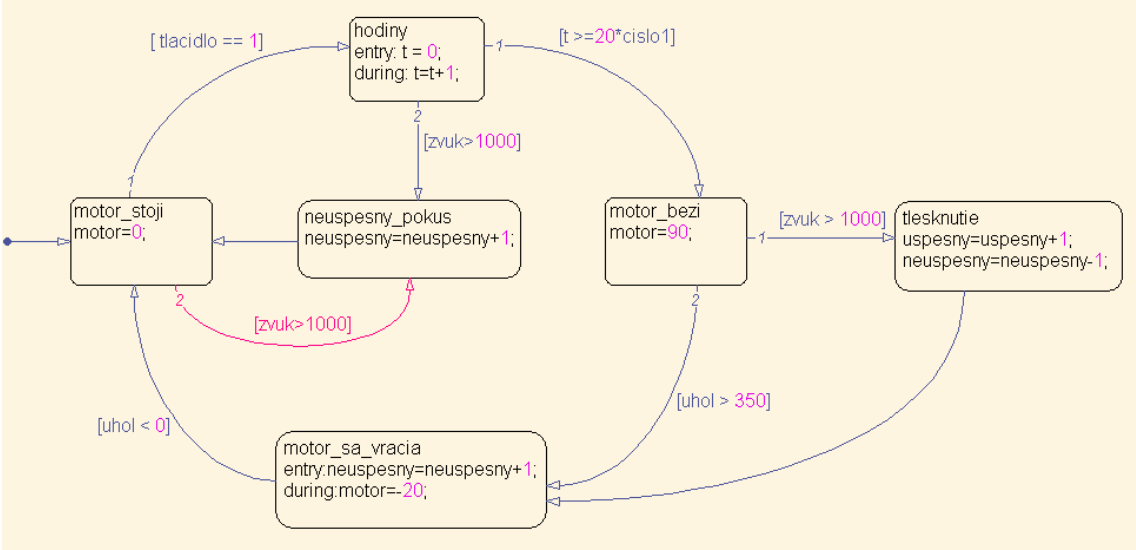
Ak sa zvuk zaznamená v stave „motor_stoji“ alebo v stave „hodiny“, zaráta sa neúspešný pokus. Vytvoríme nový stav „neuspesny_pokus“. V tomto stave sa hodnota premennej „neuspesny“ zväčší o jednotku. Do stavu „neuspesny_pokus“ sa systém môže dostať zo stavu „motor_stoji“ aj zo stavu „hodiny“. Podmienka pre prechod je rovnaká. Hodnota premennej zvuk musí byť väčšia ako 1000. Z tohto stavu sa systém automaticky presunie do prvého stavu.

Neúspešný pokus sa zaráta aj v tom prípade, že sa tlieskne až po ukončení pohybu ramena alebo sa netlieskne vôbec. Do stavu „motor_sa_vracia“ pridáme podmienku, že pri vstupe do tohto stavu sa hodnota premennej „neuspesny“ zväčší o jednotku.

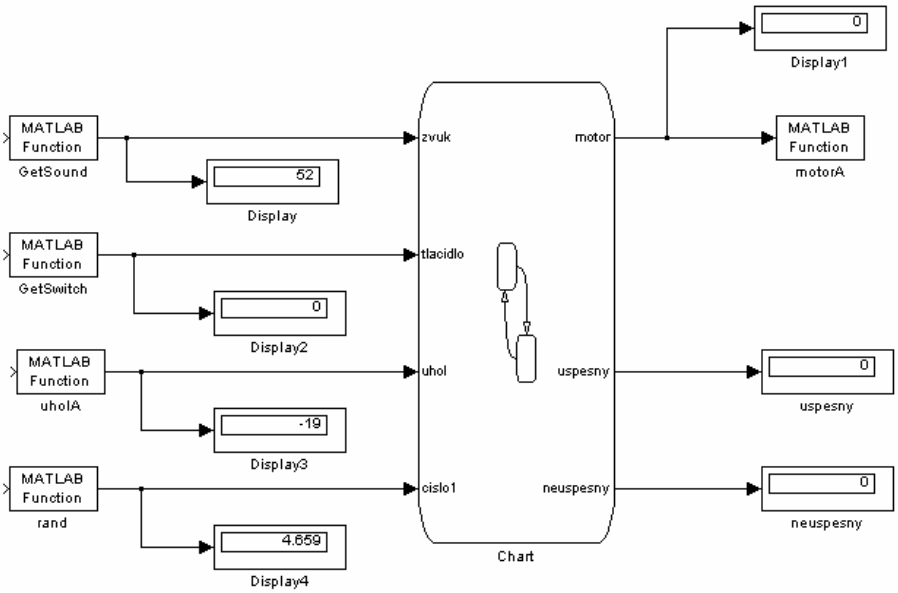


Obr. 37 Ďalší prípad zarátanie neúspešného pokusu

Problémom je, že ak sa pri vstupe do tohto stavu zaráta neúspešný pokus, tak aj po prechode zo stavu „tlesknutie“ sa zaráta neúspešný pokus. Tak by sa v stave „tlesknutie“ zarátal úspešný a hneď na to aj neúspešný pokus. Riešením tohto problému je, že do stavu „tlesknutie“ pridáme príkaz, aby sa hodnota premennej „neuspesny“ zmenšila o jednotku. Teda pri vstupe do tohto stavu sa hodnota premennej „neuspesny“ zmenší o jednotku a po prechode z tohto stavu do stavu „motor_sa_vracia“ sa zväčší o jednotku. Jej hodnota tak ostane nezmenená. Problém je vyriešený a systém pracuje tak, ako bolo požadované.



Obr. 38 Celkové riešenie úlohy merania reakcií



Obr. 39 Simulinková schéma pre úlohu merania reakcií

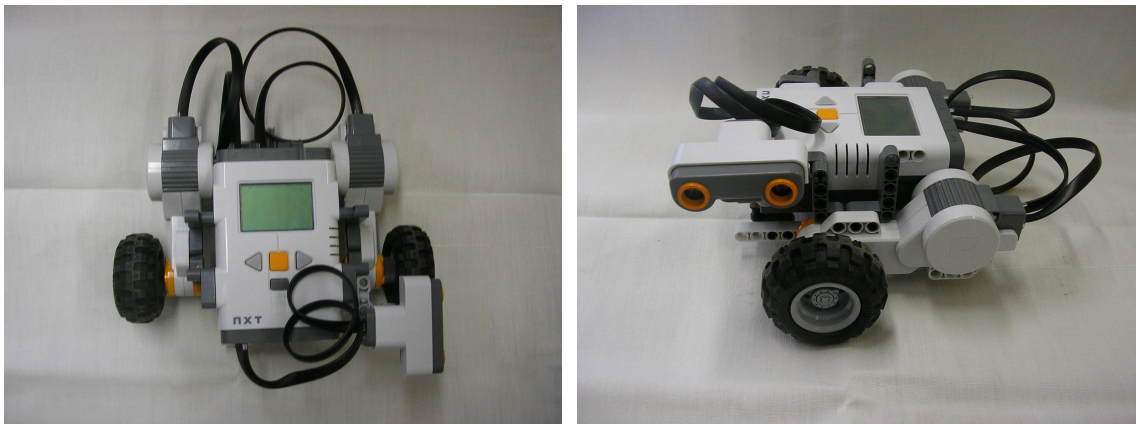
5 Úloha 3

5.1 Obchádzanie objektu

Úlohou robota je obchádzať objekt dookola, teda má sa pohybovať okolo vonkajších stien objektu a udržať konštantnú vzdialenosť medzi stenou a jeho vlastnou polohou. K dispozícii sú dva akčné členy - nezávislé servomotory, na ktorých sú pripevnené kolesá umožňujúce pohyb robota. „Očami“ robota je ultrazvukový senzor merajúci vzdialenosť robota od steny. Robot sa otáča okolo rohov (hrán) zmenou rýchlosti jedného zo servomotorov. Rýchlosť jedného kolesa je konštantná a rýchlosť druhého kolesa je menšia alebo väčšia v závislosti od toho, ktoré koleso ovládame.

5.1.1 Konštrukcia robota

Po obidvoch stranách NXT kocky pripevníme servomotory. Kolesá spojíme s nosným hriadeľom tak, aby sa robot mohol hýbať. Servomotor pre pravé koleso vložíme do portu B na NXT kocke a servomotor pre ľavé koleso do A portu. Na zaznamenávanie vzdialenosti od steny potrebujeme ultrazvukový senzor. Pripevníme ho na jednu stranu robota takým spôsobom, aby bol schopný merať vzdialenosť. Ultrazvukový senzor vložíme do portu 1.



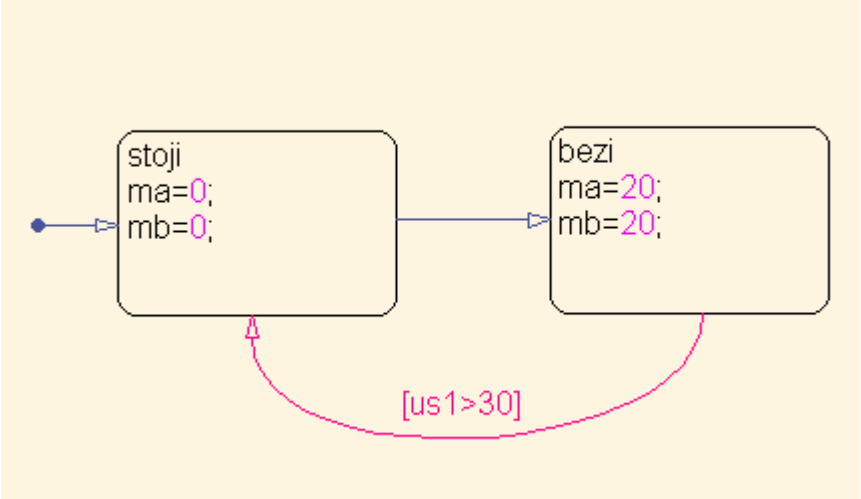
Obr. 40 Obrázok lego robota zostaveného pre úlohu obchádzania objektu

5.1.2 Pohyb okolo rovnej steny

Po spustení programu sa má robot pohybovať smerom dopredu popri stene objektu v určitej zvolenej vzdialenosti. Po prekročení tejto vzdialenosti sa zastaví.

- **Riešenie v Stateflowe**

Na riešenie úlohy vytvoríme dva stavy. Prvý stav s názvom „stoji“ a druhý s názvom „bezi“. Je potrebné si zadefinovať v Stateflow editore premenné „ma“ a „mb“ ako výstupy. Hodnoty týchto premenných predstavujú rýchlosť, ktorou sa servomotory pohybujú. V stave „stoji“ sú motor „ma“ aj motor „mb“ nastavené na hodnotu 0, to znamená, že servomotory sú vypnuté a robot stojí. Údaje z ultrazvukového senzora, ktorý meria vzdialenosť medzi robotom a stenou, vložíme do premennej „us1“, ktorú zadefinuje ako vstup. Po spustení simulácie systém automaticky prejde z prvého stavu do druhého stavu „bezi“. Robot sa dá do pohybu. Obidva servomotory sú nastavené na konštantnú rýchlosť 20, čo zabezpečuje rovný pohyb. Ak sa robot dostane na koniec steny alebo sa pri poruche od steny príliš vzdiali, teda hodnota premennej „us1“ bude väčšia ako 30, robot sa zastaví. Opäť prejde do prvého stavu.



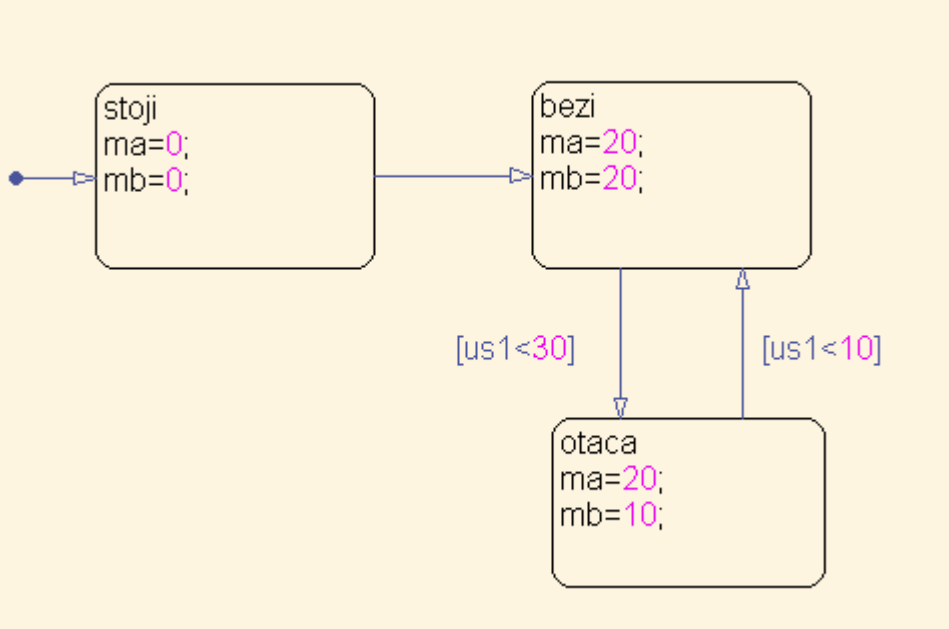
Obr. 41 Pohyb robota okolo rovnej steny

5.1.3 Otáčanie okolo objektu

V tejto úlohe je rovný pohyb rozšírený o možnosť otáčania okolo rohov, keď robot príde na koniec steny.

- **Riešenie v Stateflowe**

Princíp riešenia tejto úlohy spočíva v zaznamenávaní vzdialenosti od steny. Vytvorili sme nový stav „otaca“. Keď sa systém nachádza v stave „bezi“, pohybuje sa konštantnou rýchlosťou - motory „ma“ a „mb“ majú hodnotu 20. Ultrazvukový senzor zaznamenáva vzdialenosť od steny. Keď premenná „us1“ nadobudne hodnotu väčšiu ako 30, systém prejde do stavu „otaca“. V tomto stave rýchlosť motora „ma“ ostane konštantná a rýchlosť motora „mb“ sa zmenší na 10. Zmenšením rýchlosti jedného z motorov sa začne robot otáčať. V tomto prípade sa bude otáčať doľava. Akonáhle ultrazvukový senzor zaregistruje vzdialenosť menšiu ako 10, prejde znovu do stavu „bezi“, motory nadobudnú rovnaké rýchlosti a robot sa znovu pohybuje rovno pozdĺž steny. Tento proces sa vykonáva dovtedy, kým simuláciu nezastavíme.



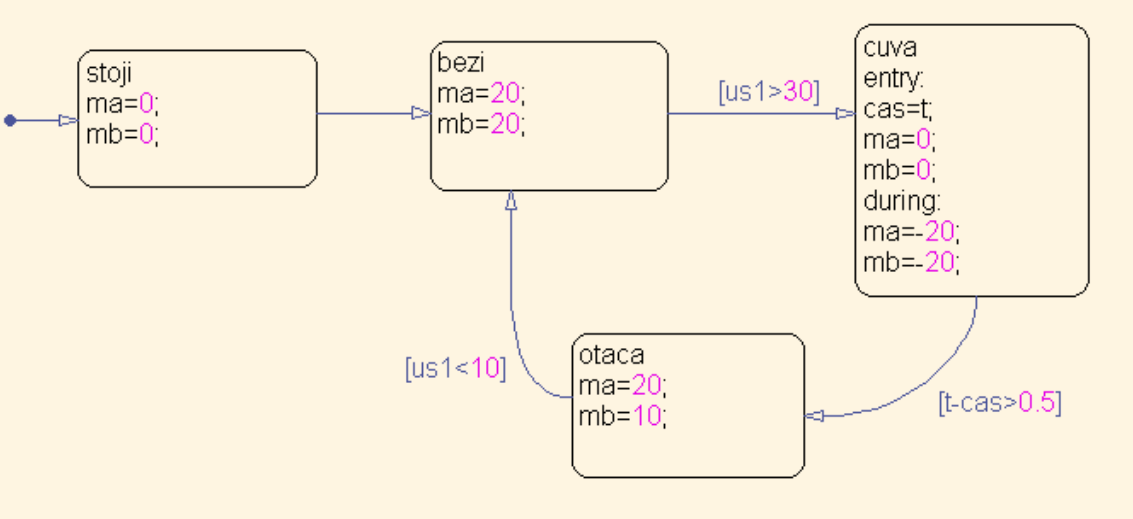
Obr. 42 Otáčanie robota okolo objektu

5.1.4 Cúvanie

Počas simulácie sa robot môže začať otáčať o pár sekúnd neskôr ako je pôvodne predpokladané alebo sa môže otáčať veľmi pomaly. To môže mať za následok veľký uhol otočenia, kedy robot už nemusí opätovne zachytiť stenu a bude sa otáčať stále dookola. Aby sme sa vyhli takejto situácii, pridáme do riadeného systému nový stav.

- **Riešenie v Stateflowe**

Stavy „bezi“ a „otaca“ rozšírime o stav „cuva“. Pridáme nové lokálne premenné „cas“ a „t“. Po vstupe do tohto stavu sa do premennej „cas“ uloží hodnota premennej „t“, ktorá predstavuje reálne bežiaci čas. Motory sa vypnú, majú hodnoty 0 a následne sa zmenia na záporné hodnoty -20, ktoré znamenajú, že sa kolesá začnú otáčať opačným smerom a robot začne cúvať. Po uplynutí pol sekundy od vstupu do stavu, systém prejde do stavu „otaca“ a začne sa otáčať ako je uvedené v časti 5.1.3. Keďže premenná „t“ predstavuje čas, ktorý beží a v premennej „cas“ je uložená hodnota „t“ pri vstupe do stavu, ich rozdiel bude tvoriť čas, ktorý uplynul od vstupu do stavu. Podmienku pre výstup zo stavu „cuva“ zapíšeme „t-cas>0,5“.



Obr. 43 Cúvanie robota

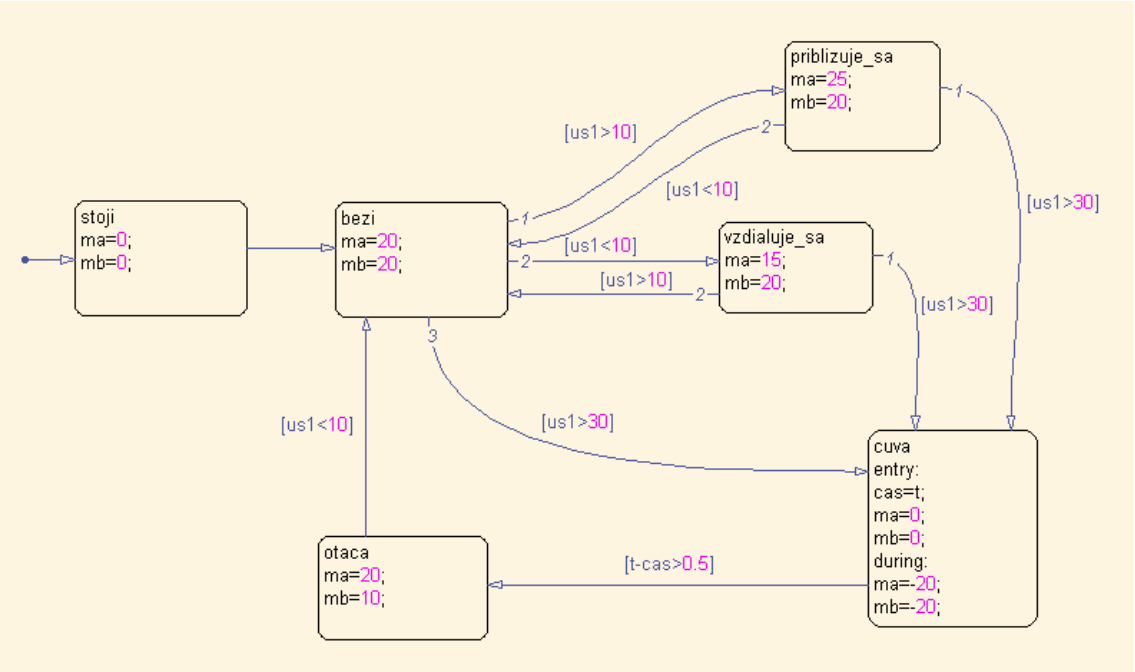
5.1.5 Udržiavanie konštantnej vzdialenosti od objektu

Úlohu môžme rozšíriť aj o možnosť udržania konštantnej vzdialenosti od objektu, pretože objekt nemusí byť postavený rovnomerne s dráhou robota a robot môže stratiť konštantnú vzdialenosť. Môže to mať za následok vybočenie robota z jeho pôvodnej dráhy a následné otáčanie robota okolo vlastnej osi.

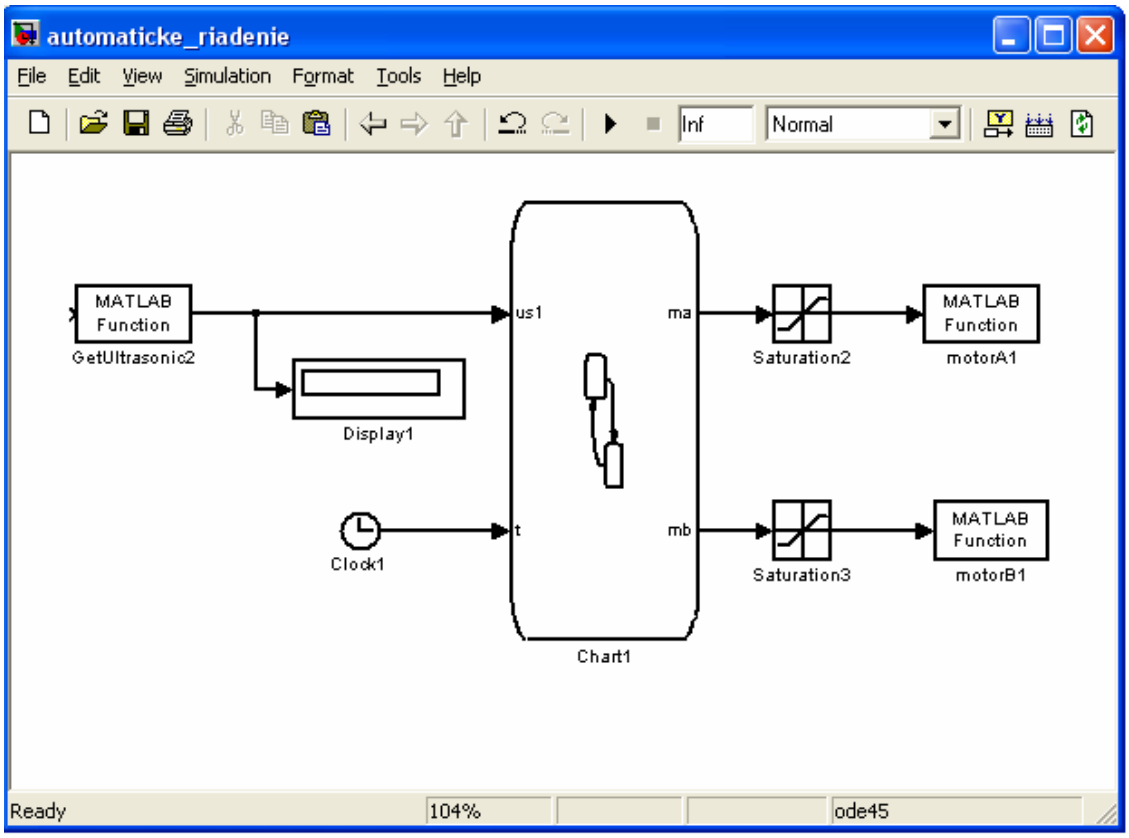
- **Riešenie v Stateflowe**

Vzdialenosť robota od steny zaznamenáva ultrazvukový senzor. Ak pri rovnom pohybe robota dôjde k prekročeniu povolených hodnôt tohto senzora, robot sa začne pomaly otáčať k stene (pri vzdialení sa od danej hodnoty) alebo od steny (pri priblížení sa k stene). Vytvorili sme stavy „priblizuje_sa“ a „vzdialuje_sa“. V stave „bezi“ sa robot pohybuje rovno a pritom sleduje ultrazvukovým senzorom vzdialenosť od objektu. Na základe rôznych hodnôt sa robot buď približuje, vzďaľuje alebo otáča. Ak premenná „us1“ je väčšia ako 10, tak sa aktivuje stav „priblizuje_sa“. To znamená, že motor „ma“ bude mať hodnotu 25, čiže o trochu väčšiu ako motor „mb“, ktorému ostane hodnota 20. Robot sa bude mierne otáčať doľava, až kým znovu nebude vzialenosť menšia ako 10, potom pôjde znovu rovno. Ak by hodnota us1 bola mešia ako 10, tak sa aktivuje stav „vzdialuje_sa“. Hodnota motora „ma“=15 bude v tomto prípade menšia ako hodnota motora „mb“=20. Robot sa začne pohybovať mierne doprava, až kým nebude hodnota premennej „us1“ väčšia ako 10.

Pri riešení tejto úlohy musíme myslieť aj na prípad, keď systém bude vyrovnávať svoju polohu a súčasne sa bude nachádzať na konci steny, kedy má dôjsť k otáčaniu. Stavy „priblizuje_sa“ a „vzdialuje_sa“ prepojíme so stavom „cuva“. Robot na nachádza na rohu steny ak je hodnota premennej „us1“ väčšia ako 30, podmienkou prechodu do tohto stavu je teda „us1>30“. Splnením tejto podmienky sa systém dostane do stavu „cúva“.



Obr. 44 Udržiavanie konštantnej vzdialenosti od objektu



Obr. 45 Simulinková schéma pre úlohu obchádzania objektu

6 Záver

Hlavným cieľom bakalárskej práce bolo ukázať riadenie robotov LEGO Mindstorms pomocou programu Stateflow na jednotlivých zadaných úlohách. Popísali sme činnosti technických parametrov senzorov a akčných členov, ktoré sa pri riadení robota využívali.

Celá práca pozostáva z piatich kapitol. Výsledkom prvej kapitoly je oboznámenie so softvérom Stateflow, všeobecná charakteristika a vysvetlenie vytvárania jednotlivých častí riadenia v Stateflow prostredí. Je v nej popísané vytváranie stavov a stavových diagramov, pridanie udalostí a rôzne možnosti prepojenia jednotlivých stavov.

V ďalšej časti práca popisuje technické parametre NXT kocky, ktorá je hlavnou riadiacou kockou, technické parametre senzorov a akčných členov a ich možnosti využitia.

V posledných častiach sú rozobraté jednotlivé zadania úloh. Vypracované sú tri úlohy, z ktorých každá tvorí samostatnú kapitolu. Úlohy obsahujú zadanie, popísanie konštrukcie robota (t. j. ktoré akčné členy a senzory sa pri danej úlohe použili a do ktorých portov boli zapojené a obrázok konštrukcie robota). V každej úlohe sa popisuje celková úloha pomocou čiastkových riešení, pričom ku každej úlohe sú riešenia doplnené o obrázky, ktoré znázorňujú riešenie v Stateflow.

Na základe týchto úloh sa dá pochopiť, ako ľahko sa programom Stateflow dajú riešiť aj mnohé iné zložité úlohy. Dajú sa ním naprogramovať a riadiť aj problémy bežného života, ako je napr. fungovanie klimatizácie alebo riadenie križovatky a mnoho iných.

LEGO robotov môžeme riadiť aj s využitím bluetooth pripojenia, čo rozširuje možnosti pohybu robota v priestore a umožňuje riadiť mnohé zaujímavé úlohy. Táto funkcia sa však v práci nevyužíva.

Táto práca môže byť aj pomôckou pre budúce riešenie úloh na laboratórnych cvičeniach z predmetu projektovanie riadiacich a informačných systémov. Umožňuje s ľahkosťou pochopiť riadenie lego robotov aj študentom, ktorí sa s robotikou zatiaľ ešte nestretli.

Zoznam použitej literatúry

[1] http://www.mathworks.com/products/stateflow/description1.html

[2] http://www.atpjournal.sk/casopisy/atp_08/pdf/atp-2008-07-68.pdf

[3] http://dinosaur.sk/next-mindstorms-p-114.html?zenid=oi9lh59fu26hbtaec41mgpemo5

[4] http://mindstorms.lego.com/eng/Overview/The_NXT.aspx

[5] http://www.ai-cit.sk/source/prirucka-kasanicky/prirucka.pdf

[6] http://mindstorms.lego.com/eng/Overview/Touch_Sensor.aspx

[7] http://sk.wikipedia.org/ wiki/Decibel

[8] http://mindstorms.lego.com/eng/Overview/Sound_Sensor.aspx

[9] http://mindstorms.lego.com/eng/Overview/Light_Sensor.aspx

[10] http://mindstorms.lego.com/eng/Overview/Ultrasonic_Sensor.aspx

[11] http://mindstorms.lego.com/eng/Overview/Interactive_Servo_Motors.aspx

Prílohy

Príloha A: CD médium – práca v elektronickej podobe.