Izvedba mehkega regulatorja hitrosti mobilnega robota s poljem programirljivih logičnih vrat

Jernej Otič, dr. Andreja Rojko
UM-FERI
Smetanova ulica 17
jernej.otic@uni-mb.si, andreja.rojko@uni-mb.si

Realization of the Fuzzy Logic Controller for a Mobile Robot by Using FPGA

In this work the realization of fuzzy control of DC motor is considered for the case, when Fields Programmable Gate Array is used as a platform. For more attractive and more real demonstration of the controller's efficiently we have built a mobile robot with DC motors and then tested the controller's efficiency on that robot.

This work presents details concerning the realization of the fuzzy logic controller in the Fields Programmable Gate Array (FPGA) and realization of graphical user interface for controlling and steering the robot's motors.

For programming of the FPGA circuits the program packet Xilinx ISE Web PACK was used. The simulation was realized in the program packet MATLAB/Simulink.

Kratek pregled prispevka

Delo obravnava problem izvedbe mehke regulacije hitrosti enosmernega motorja s poljem programirljivih logičnih vrat. Za atraktivnejšo in realnejšo predstavitev smo regulacijo vgradili in preizkusili na mobilnem robotu z enosmernimi motorji na pogonskem sklopu.

Opisana je izvedba mehkega regulatorja na polju programirljivih logičnih vrat (FPGA), in izdelavo grafičnega vmesnika za nadzor in krmiljenje motorjev.

Za programiranje vezij FPGA smo uporabili programski paket Xilinx ISE WebPACK. Simulacijo smo izvedli s pomočjo programskega paketa MATLAB/Simulink.

1 Uvod

Regulacijo sistema z mehko logiko uporabimo takrat, kadar želimo voditi sistem, katerega matematični model je slabo poznan ali netočen in kadar ima sistem izrazito nelinearno dinamiko. Danes zelo razširjen in preprost način za realizacijo mehke regulacije na realnem sistemu je z uporabo osebnega računalnika in ustrezne programske opreme. Vendar tak način ni vedno najboljši, saj so osebni računalniki dragi, veliki, kompleksni, energetsko potratni in največkrat še klub vsemu prepočasni. Tem pomanjkljivostim se izzognemo z implmentacijo mehkih regulatorjev na mikroprocesorjih, namenskih čipih, ali na poljih programirljivih logičnih vrat – FPGA.

Prednost FPGA je v veliki gostoti logičnih vrat, možnosti paralelnega delovanja, veliki prilagodljivosti in vsestranskosti. Zaradi vsestranskosti in prilagodljivosti je čas od razvoja do prodaje izdelka krajši, s čimer se znižajo tudi stroški razvoja novega izdelka.

Namen dela je opisati implementacijo mehkega regulatorja hitrosti mobilnega robota, temelječega na mehki logiki in implementiranega na polju programirljivih logičnih vrat – FPGA .

Za implementacijo mehke logike na FPGA smo uporabili programski paket Xilinx ISE in razvojno ploščo za FPGA, podjetja Digilent, serije Nexys 2.

Da smo lahko spreminjali in opazovali hitrost motorjev smo v programskem paketu Delphi 7 izdelali uporabniški grafični vmesnik za osebni računalnik.

Za našega mobilnega robota smo naredili tudi simulacijo v MATLAB/Simulinku, vendar v tem delu ne bo natančneje predstavljena.

2 Komunikacija z osebnim računalnikom

2.1 Grafični vmesnik

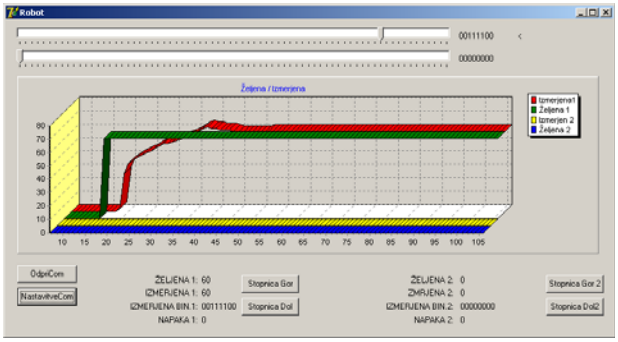
Grafični vmesnik za spreminjanje in spremljanje hitrosti je napisan v programskem paketu Delphi 7. Za komunikacijo z razvojno ploščo smo uporabili programsko knjižnico TComPort 3, ki nam omogoča enostavno vzpostavitev komunikacije z zunanjimi napravami, priključenimi na COM vrata, ki delujejo po protokolu RS232. Knjižnica TcomPort omogoča veliko nastavitev, ki so potrebne za pravilno delovanje komunikacije. Knjižnico sestavljajo naslednje komponente: ComPort, ComDataPacket, ComComboBox, ComRadioGroup, ComLed in ComTerminal. Za naše delo smo uporabili samo dve komponenti, in sicer komponento ComPort in Komponeto ComDataPacket.

Komponenta ComPort služi za komunikacijo in v njej lahko nastavimo lastnosti COM vrat, kot so: Baudna hitrost, število podatkovnih bitov, stop bit, vrata, na katera se bomo priključili, v našem primeru COM7, pariteto, maksimalni dovoljeni čas.

Komponento ComDataPacket pa smo uporabili, zato, ker je bil naš podatkovni paket dolg 2 bajta. Za nastavljanje želene hitrosti vsakega od motorjev smo uporabili po en bajt.

Poleg komunikacije z razvojno ploščo je grafični vmesniki omogočal opazovanje referenčne in dejanske hitrosti motorjev in jih izrisoval na grafu. Grafični vmesnik prikazuje tudi absolutno razliko med dejansko in želeno hitrostjo.

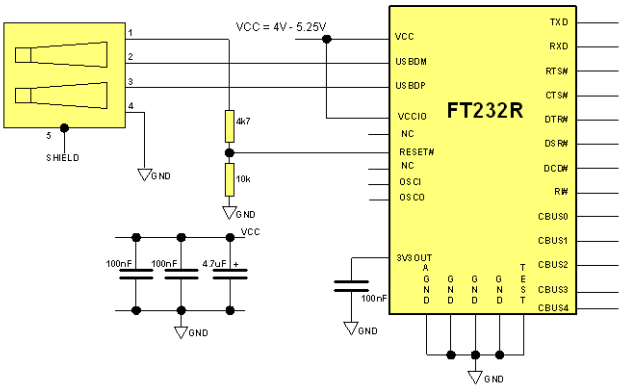
Grafični vmesnik nadalje omogoča tudi zvezdno ali pa stopnično nastavljanje referenčnih hitrosti. Slika 1 prikazuje grafični vmesnik.



Slika 1: Grafični vmesnik za nadzor motorjev robota.

2.2 Pretvornik iz USB na RS232

Komunikacijo med osebnim računalnikom in razvojno ploščo smo izvedli preko RS232, ker je ta protokol dobro poznan in enostaven za uporabo. Ker pa na večini sodobnih osebnih računalnikih ni več RS232 vrat, ampak samo še USB vrata smo morali narediti pretvorbo iz USB na RS232. V ta namen smo uporabili čip FT232R. Izdelali smo preprosto tiskanino po načrtu kot ga prikazuje slika 2, vendar smo za naš primer uporabili samo RX in TX liniji.



Slika 2: Načrt za komunikacijski vmesnik iz USB na RS232 [1].

2.3 Komunikacijski modul za FPGA

Komunikacijski modul je sestavljen je iz dveh delov prvi del je sprejemni del, drugi pa oddajni del.

Sprejemni del je narejen tako da opazujemo vhod signala in ko je signal pade na 0 je to indikator, da smo prejeli začetek prvega bajta.

Ker smo se odločili, da bo hitrost prenosa podatkov 19200 baudov zadostna za našo komunikacijo smo vhodni urni signal, ki znaša 50MHz delili z 19200. Da pa smo vhodni signal vzorčili ravno na sredini vsakega bita smo rezultat deljenja delili še z 2. Tako smo prvi bit detektirali pri vrednosti 1302, za detekcijo vsakega naslednjega bita pa smo število 1302 množili z 2*n-1 (torej z neparnimi števili). Tako smo prvi bit detektirali pri vrednosti izhoda 1302 drugega pri 3*1302 tretjega pri 5*1302 in tako naprej.

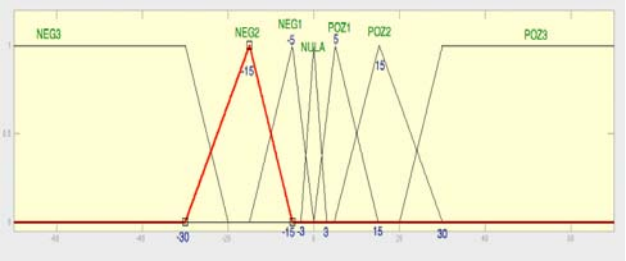
Drugi izhodni del je izveden tako da izhod držimo na želeni vrednosti natančno 2604 urinih ciklov.

3 Regulator z mehko logiko

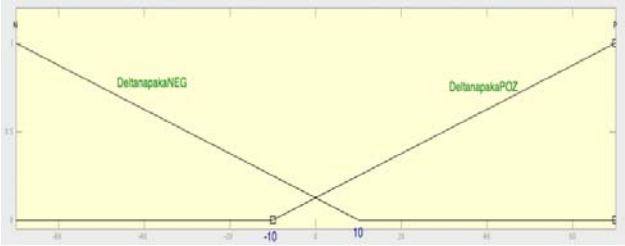
Za mehki regulator smo uporabili Sugeno tip mehkega regulatorja. Za vhodne spremenljivke smo uporabili pripadnostne funkcije trikotnih in trapeznih oblik. Za izhod pa smo vzeli konstante 3600, 500, 50, 1, -50, -500, -3600.

Iz podatka o dejanskem številu vrtljajev motorja in reference, ki smo jo dobili iz komunikacijskega modula smo izračunali pogrešek. Poleg pogreška smo za PI regulator potrebovali še razliko pogreškov med dvema tipanjima in pogrešek v prejšnjem tipanju.

Glede na razpon obratov smo določili razpon mehkih pripadnostnih množic za mehčanje. Pripadnostne množice za pogreške smo določili tako kot prikazuje slika 3., za odvod pogreška tako pa tako kot prikazuje slika 4. Pripadnostne množice za pogreške smo poimenovali NEG3, NEG2, NEG1, NULA, POZ1, POZ2 in POZ3. Za odvod pogreškov pa DeltanapakaNEG in DeltanapakaPOZ.



Slika 3: Oblika pripadnostnih množic za pogreške.



Slika 4: Oblika pripadnostnih množic za odvod pogreškov.

Ko smo določili pripadnostne funkcije smo interval pogreškov od -70 do +70 razdelili na odseke. Odseki so bili določeni glede nato kje stranica trikotnika seka abscisno os oziroma ima

Odsek	Računanje stopnje pripadnosti
Napaka >30	POZ3=1
20 < napaka <= 30	POZ3=0.1 * napaka-2 in POZ2=-0.066 * napaka+2
15 < napaka <= 20	POZ2=-0.066 * napaka + 2
5 < napaka <= 15	POZ2=0.1 * napaka - 0,5 in POZ1= -0.1 * napaka + 1,5
3 < napaka <= 5	POZ1=0.2 * napaka
0 < napaka <= 3	POZ1=0.2 * napaka in NULA=-0.333 * napaka + 1
0 => napaka >= -3	NULA=0.333 * napaka +1 in NEG1=-0.2 * napaka
-3 > napaka >= -5	NEG1=-0.2 * napaka
-5 > napaka >= -15	NEG1=0.1 * napaka + 1,5 in NEG2= -0.1 * napaka - 0,5
-15 > napaka >= -20	NEG2=0.066 * napaka + 2
-20 > napaka >= -30	NEG2=0.066 * napaka + 2 in NEG3= -0.1 * napaka-2
napaka <-30	NEG3=1
Za razliko napak smo določili samo dve pripadnostni množici in tri odseke:	
deltanapaka>10	DeltanapakaPOZ=0.013* Deltanapaka + 0.132;
0<deltanapaka<=10	DeltanapakaPOZ= 0.013* Deltanapaka + 0.132; in DeltanapakaNEG=-0.013* Deltanapaka -0.132;
-10<=deltanapaka<=0	DeltanapakaPOZ= 0.013* Deltanapaka + 0.132; in DeltanapakaNEG=-0.013* Deltanapaka -0.132;
deltanapaka<-10	DeltanapakaNEG=-0.013* Deltanapaka -0.132;

Slika 6: Računanje stopnje pripadnosti pogreška in razlike pogreškov za posamezni odsek.

trikotnik svoj vrh. Vrednost stopnje pripadnosti smo računali po enačbi premice (1).

$$y = kx + n \qquad (1)$$

Vrednosti stopnje pripadnosti pogreška in razlike pogreškov za posamezni odsek so prikazane na sliki 5.

Ker smo želeli uporabljati celoštevilsko aritmetiko smo v VHDL-u vrednosti skalirali in jih zato množili z 1000. Primer kode za izračun napake je podan na sliki 6.

```
...
elsif napaka > 20 and napaka <= 30 then
Poz3:=(100*napaka)-2000;
Poz2:=(-66*napaka)+2000;
Poz1:=0;
nula:=0;
neg1:=0;
neg2:=0;
Neg2:=0;
predznak<='0';
...
```

Slika 5: Izračun stopne pripadnosti v VHDL-u.

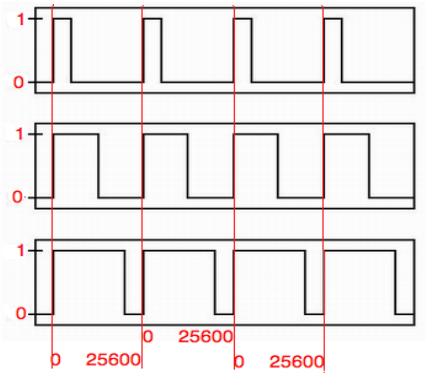
Med množicami smo uporabili sledeča pravila:

- Če je napaka velika negativna je izhod velik negativen.
- Če je napaka srednje negativna je izhod srednje negativen.
- Če je napaka majhna negativna je izhod majhen negativen.
- Če je napaka majhna pozitivna je izhod majhen pozitiven.
- Če je napaka srednje pozitivna je izhod srednje pozitiven.
- Če je napaka velika pozitivna je izhod velik pozitiven.
- Če je napaka nič potem je izhod nič.
- Če je napaka nič in odvod napake pozitivna je izhod mali negativen.
- Če je napaka nič in odvod napake negativna je izhod mali pozitiven.

Za ostrenje mehkih vrednosti smo uporabili poenostavljeno težiščno metodo, enačba 2.

$$y = \frac{\sum_{l=1}^{M} y^{-l} \cdot \mu_B^l\left(x, y^{-l}\right)}{\sum_{l=1}^{M} \mu_B^l\left(x, y^{-l}\right)} \qquad (2)$$

Za P del regulatorja smo ostrenje izvedli tako da smo izračunali vrednosti deljenca in delitelja (slika 7) in ju potem peljali na delilnik.



Slika 7: Delovanje PWM modula 20%, 50% in 80% odprtje.

```
...
deljenec<=(poz3*3600+poz2*500+poz1*50
+nula*1+neg1*50+neg2*500+neg3*3600);
deljitelj<=Poz3+Poz2+poz1+nula+neg1+N
eg2+Neg3;
...
```

Slika 8: Izračun delitelja in deljenca za P-del.

Za I del smo P delu morali prišteti še vrednost pripadnosti mehkim množicam za I del. Za lažje razumevanje je del kode predstavljen v sliki 8.

```
...
if Deltanapaka <0 and Deltanapaka >= -10
then
DeltanapakaPOZ:=13* Deltanapaka+132;
DeltanapakaNEG:=-13* Deltanapaka-132;
Delta:=((DeltanapakaNEG+DeltanapakaPO
Z)*nula);
deljenec<=(poz3*3600+poz2*500+poz1*50
+nula*1+(neg1+Delta)*50+neg2*500+neg3*3
600);
deljitelj<=Poz3+Poz2+poz1+nula+neg1+N
eg2+Neg3+Delta;
...
```

Slika 10: Izračun delitelja in deljenca za I-del.

Deljenec in deljitel smo peljali na delilnik, rezultate delilnika pa nato v PWM modul.

3.1 PWM modul

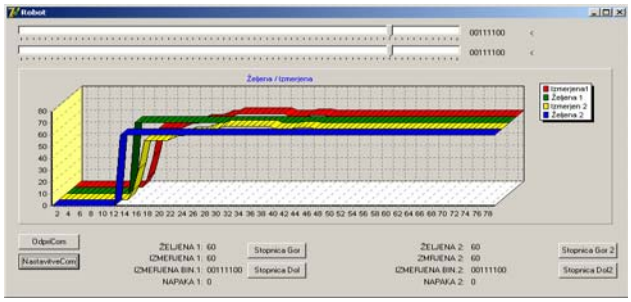
PWM modul, to je modul za pulzno širinsko modulacijo, je bil prav tako napisan v VHDL-u. Modul je bil narejen tako, da je bila frekvenca modulacije 1,95 Khz, saj proizvajalec H-mostiča s katerim smo krmilili motor, priporoča modulacijsko frekvenco okrog 2 Khz. Do frekvence 1,95 smo prišli tako da smo 50Mhz uro delili z 25600.

Pulzno širinsko modulacijo smo nato izvedli tako, da smo določeno število urinih ciklov držali na 1 in preostanek do 25600 na 0. Delovanje PWM-modula prikazuje slika 9.

Čas, ko je bila vrednost PWM-ja na 1 smo določili glede na vrednost, ki smo jo dobili iz mehkega regulatorja. Trenutni vrednosti PWM-ja smo prišteli oziroma odšteli vrednost, ki nam jo je dal mehki regulator.

4 Zaključek in rezultati

Odziv na stopnico je bil odličen glede (Slika 10) na to da smo dobili iz Hallovih sond samo 4 pulze na obrat. Boljših rezultatov z uporabljenimi motorji in senzorji ni možno



Slika 9: Odziv motorjev robota na stopnico.

doseči.

Izmerjena vrednost je od želene odstopala za 0 ali 1 pulz torej za četrtino obrata.

V nadaljevanju dela bi bilo potrebno uporabiti motorje z inkrementalnimi dajalniki, ki imajo večjo število pulzov kot Hallove sonde. Ker se mora oblika in število pripadnostnih funkcij spremiti, če spremenimo regulirani sistem, bi bilo potrebno razmisliti in narediti grafični vmesnik, ki bi avtomatsko sprogramiral regulator in spremenil obliko in število pripadnostnih funkcij za mehčanje.

5 Literatura

[1] http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf