Razvojni modul DSP2

Darko Hercog, Karel Jezernik
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Smetanova ulica 17, Maribor
darko.hercog@uni-mb.si

EVOLUTION MODULE DSP2

Abstract: The DSP2 board, based on TMS320C32 DSP processor, has been developed at the Institute of Robotic, FERI, University of Maribor. For this board a set of the Simulink blocks was created under the Real-Time Workshop. This blocks enable easy graphical programming of different control algorithms under the Simulink. At the end the C-code is generated from the Simulink model, and downloaded to the DSP2 board.

1 Uvod

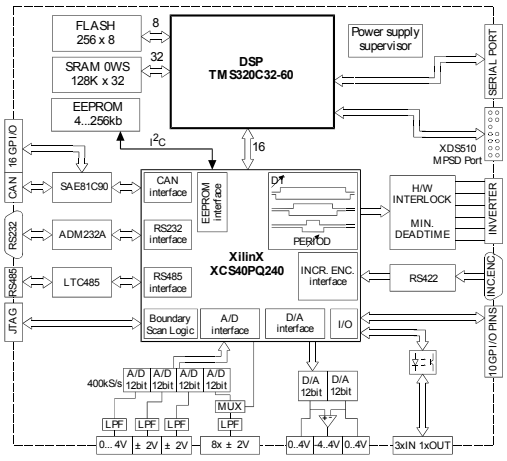
Na Inštitutu za robotiko, fakultete za elektrotehniko računalništvo in informatiko Univerze v Mariboru, je bila razvita DSP2 kartica [1], ki temelji na digitalnem signalnem procesorju (DSP) s plavajočo vejico in se v glavnem uporablja za regulacijo asinhronskih motorjev. Programiranje le-te je do nedavnega potekalo izključno v programskem jeziku C. Ker pa je še vedno preteklo precej časa od zasnove regulacijskega algoritma in simulacije le-tega v simulacijskem programu Simulink, do ralizacije na DSP2 kartici, smo se odločili izdelati DSP2 knjižnico za Simulink, ki omogoča blokovno programiranje omenjene kartice s pomočjo Matlab - Simulinka.

Dandanes na tržišču obstaja kar nekaj kartic z DSP procesorji, ki omogočajo programiranje s pomočjo Matlab-Simulinka, vendar je težko najti takšno, ki bi tako po zmogljivosti, ceni in periferiji ustrezala našim zahtevam. V večini primerov gre za PC kartice, katerih slabost je, da ne zmorejo delovaja brez uporabe PC-ja, še večja šibka točka pa je njihova visoka cena.

V prispevku bo na kratko predstavljena DSP2 kartica oz. razvojni modul DSP2 (EVM-DSP2). Sledi opis Real-Time Workshop-a in postopek generiranja binarne kode iz Simulinkovega modela. Nato je nekaj besed namenjenih DSP2 knjižnici za Simulink, v zadnjem delu prispevka pa je prikazan primer uporabe omenjene knjižnice z izvedbo PID regulatorja.

2 DSP2 kartica

Na Inštitutu za robotiko so se odločili za razvoj lastne kartice, t.i. DSP2 kartice [1], ki temelji na signalnem procesorju s plavajočo vejico TMS320C32 proizvajalca Texas Instrument in Xilinx-ovem FPGA-ju XCS40PQ240 družine Spartan. Kartica se primarno uporablja za regulacijo položaja, hitrosti in navora izmeničnih asinhronskih motorjev, lahko pa se uporablja tudi v splošne namene.



Slika 1: Blokovna shema DSP2 kartice

Na sliki 1 je prikazana blokovna shema DSP2 kartice, kjer so razvidne vse njene kjučne komponente.

Slika 2 prikazuje t.i. razvojni modul DSP2 (EVM-DSP2). Razvojni modul je sestavljen iz dveh kartic in sicer iz že omenjene DSP2 kartice (desna stran slike 2) in dodatne kartice, na kateri se nahaja napajalni del in priključni konektorji (leva stran slike 2). Na priključne konektorje so speljani vsi pomembni vhodno/izhodni signali DSP2 kartice.

Slika 2: Fotografija razvojnega modula DSP2 (EVM-DSP2)

3 Sistemske funkcije in Real-Time Workshop

Sistemske funkcije ali S-funkcije **[6]** razširjajo zmožnosti Simulink-a, saj omogočajo vključitev lastnega bloka v okolje Simulink. Algoritem S-funkcije je možno napisati v Matlab-u ali C programskem jeziku (C-MEX S-funkcije). Struktura S-funkcij je zelo splošna in tako omogoča realizacijo zveznih, diskretnih ali hibridnih sistemov. S S-funkcijo se bloku določi število vhodov, število izhodov, število stanj, čas tipanja, algoritem izračuna izhodne izhodnosti itd. S-funkcije kodirane v Matlabu ali C programskem jeziku, se nato s uporabo Matlab-ovega ukaza *mex* prevedejo kot MEX (Matlab executable) funkcije, in le-te predstavljajo dinamične knjižnice, ki jih Simulink potrebuje med svojim delovanjem. C-mex S-funkcije se vključi v okolje Simulink z uporabo *S-function* bloka s pomočjo urejevalnika dialognih oken (*mask editor*), pa je nato le-temu kreirati statičen, ali dinamičen uporabniški vmesnik. Za vsak blok DSP2 knjižnice [3] (kratek opis se nahaja v prihodnjem poglavju) je bilo potrebno napisati sistemsko funkcijo in kreirati pripadajoč uporabniški vmesnik.

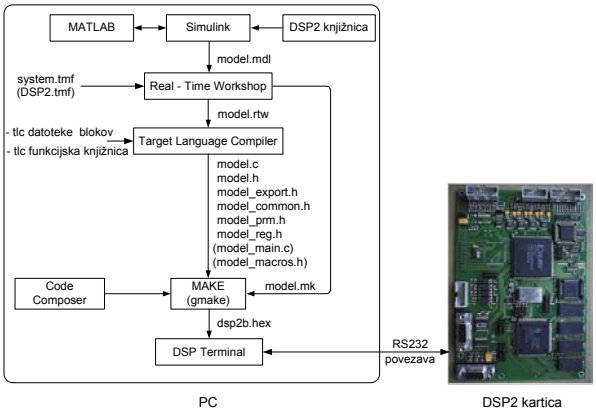
Real-Time Workshop (RTW) **[4]** je dodatek k Simulink-u, ki omogoča avtomatsko generiranje ANSI C ali ADA kode iz Simulinkovega modela. V splošnem RTW generira dve vrsti C kode in sicer:

- splošno C koda
- kodo za vgrajene sisteme

Odločili smo se za generiranje kode za vgrajene sisteme, saj je le-ta precej bolj optimizirana kot splošna C koda, ima pa nekatere omejitve, ki bodo navedene v nadaljevanju članka.

Generiranje C kode iz Simulinkovega modela poteka tako (Slika 3), da RTW na podlagi Simulinkovega modela (shranjen v ASCII datoteki s končnico .mdl) generira RTW datoteko (model.rtw). Le-ta vsebuje vse informacije o modelu, ki so potrebne za generiranje kode, kot so: ime modela, čas tipanja, zaporedje izvajanja posameznih blokov, vrednosti parametrov posameznih blokov, itd. RTW prav tako iz šablonske *makefile* datoteke (template makefile) *system.tmf* (v našem primeru DSP2.tmf) generira *makefile* datoteko *model.mk*, ki je potrebna pri generiranju binarne kode iz generiranih C datotek. V proces generiranja kode se nato vključi ciljni prevajalnik (Target Language Compiler - TLC) **[5]**, ki na podlagi tlc datotek posameznih blokov in tlc funkcijske knjižnice prevede Simulinkov model *model.rtw* v C kodo. Generirana koda se nahaja v več *.c* in *.h* datotekah. Z uporabo *make* programa (uporabljen je bil *gmake* organizacije Free Software Fundation), se na podlagi predhodno generirane *makefile* datoteke *model.mk* in ustreznega prevajalnika izvrši pretvorba generirane C kode v binarno kodo. Izbira prevajalnika je odvisna od sistema, na katerem se bo generirana koda izvajala. V našem primeru je bil uporabljen TI prevajalnik za c3x4x družino signalnih procesorjev, ki je sestavni del Code Composer-ja [8]. Binarna koda se nato s pomočjo DSP Terminala [2] naloži na DSP2 kartico. DSP Terminal, ki je

prav tako bil razvit na Inštitutu za robotiko, zraven naložitve izvršne kode na DSP2 kartico, omogoča tudi sprotno (*online)* spremljanje in spreminjanje vrednosti določenih spremenljivk DSP procesorja.



Slika 3: Potek generiranja izvršne kode

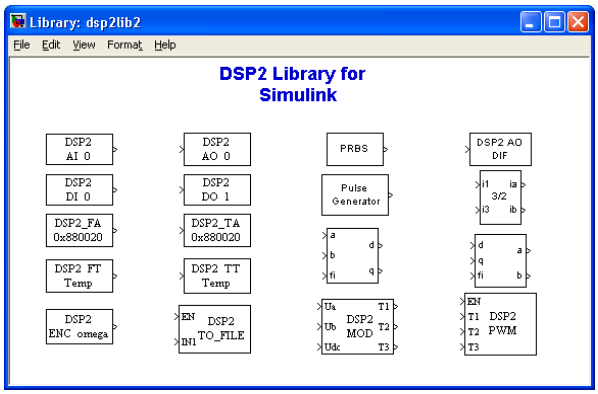
Celoten postopek generiranja binarna kode (dsp2b.hex) se izvede avtomatsko (Slika 3). Čas generiranja pa je odvisen od zmogljivosti uporabljenega osebnega računalnika.

Ker smo se odločili za generiranje kode za vgrajene sisteme, je bilo potrebno za vsak blok DSP2 knjižnice [3], zraven S-funkcije napisati še ustrezno tlc datoteko, ki določa, kako se naj posamezen blok prevede v C kodo. Prav tako je bilo potrebno napisati šablonsko *makefile* datoteko (DSP2.tmf), v kateri je zapisano, kako se naj iz generiranih .c in .h datotek generira binarna koda.

4 DSP2 knjižnica za Simulink

Slika 4 prikazuje DSP2 knjižnico za Simulink. S dvojnim klikom na posamezni blok DSP2 knjižnice, se odpre uporabniški vmesnik bloka, v katerem se lahko nastavijo dodatni parameteri, ki pa so specifični za posamezen blok (npr. amplituda in povprečna vrednost PRBS signala).

DSP2 knjižnica vsebuje sledeče bloke: Analog Input, Analog Output, Digital Input, Digital Output, From Address, To Address, From Terminal, To Terminal, Incremental encoder, To File, Modulator, PRBS, PWM in bloka za transformacijo med koordinatnimi sistemi: *a-b* v *d-q* in *d-q* v *a-b*. Pomen posameznih blokov DSP2 knjižnice je podrobneje pojasnjen v literaturi [3].



Slika 4: DSP2 knjižnica za Simulink

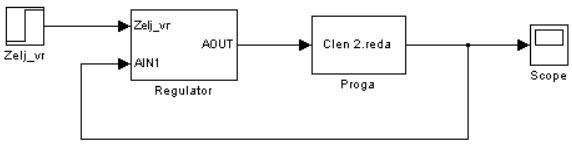
Pomembna lastnost DSP2 knjižnice je zmožnost sprotnega (online) spreminjanja določenih parametrov posameznih blokov Simulinkovega modela. Parametre, ki jih želimo spreminjati v sprotnem načinu je v Simulinku potrebno označiti kot *Tunable Parameters*. Ti parametri so lahko le skalarnega tipa in morajo biti definirani v Matlabovem delovnem področju (workspace). Po generiranju binarne kode in naložitvi le-te na DSP2 kartico, se označeni parametri pojavijo na strani *Parameters* DSP Terminala [2], kjer jim je možno spreminjati vrednosti v sprotnem načinu, oz. medtem, ko se generirana koda izvaja na DSP2 kartici.

Potrebno je omeniti še dve pomembni omejitevi. Ker smo se odločili za generiranje kode za vgrajene sisteme sme Simulinkov model (oz. podsistem, za katerega želimo generirati kodo za DSP2 kartico) vsebovati le diskretne bloke, hkrati pa mora biti določen fiksen korak simulacije. Ta čas se namreč po generiranju kode vpiše v periodni register DSP2 kartice in določa periodo prekinitvenih rutin.

Za opisano knjižnico je napisan tudi instalacijski/deinstalacijski program. Predpogoj za uspešno namestitev na osebni računalnik je predhodna namestitev Matlaba (6.0, 6.1 ali 6.5) s dodatkoma RTW in RTW Embedded Coder ter Code Composer-ja za C3x4x družino signalnih procesorjev.

5 Primer uporabe

V nadaljevanju bo prikazan preprost primer uporabe DSP2 knjižnice za Simulink. Prikazano bo, kako v Simulinku kreirati PID regulator, ki se bo po generiranju kode izvajal na DSP2 kartici. Na sliki 5 je prikazan Simulinkov model zaprtozančnega sistema, ki je sestavljen iz vzbujanja, regulatorja, proge in povratne vezave.



Slika 5: Simulacijska shema

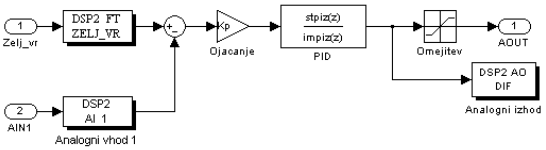
Za progo je bil uporabljen laboratorijski člen 2. reda s sledečo prenosno funkcijo:

$$H_P = \frac{K_1}{sT_1 + 1} \cdot \frac{K_2}{sT_2 + 1} \qquad (1)$$

in sledečimi parametri:

$K_1 = 1,\ T_1 = 20ms,$

$K_2 = 0.2,\ T_2 = 44ms$



Slika 6: Podsistem »Regulator«

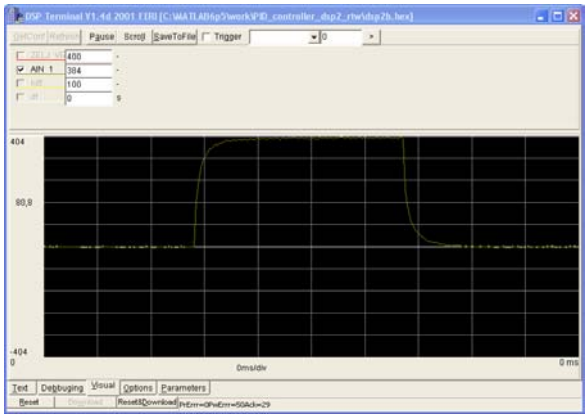
V podsistemu »Regulator« se nahaja algoritem PID regulatorja (Slika 6). Za nastavljanje željene vrednosti je uporabljen blok DSP2_FT (Slika 6) [3]. Blok omogoča (po generiranju kode in naložitvi le-te na DSP2 kartico) nastavljanje željene vrednosti v uporabniškem vmesniku DSP Terminala. Izhodna napetost procesa (člena 2. reda) se zajema preko analognega vhoda 1 (DSP2_AI 1) DSP2 kartice. Razlika med željeno in dejansko vrednostjo je peljana na vhod prenosne funkcije PID regulatorja (izračunana je bila po metodi kompenzacije polov), njegov izhod pa na blok DSP2_AO. Ta blok v simulaciji ne opravlja nobene naloge oz. ima podobno vlogo kot Simulinkov blok *Terminator*. V generirani kodi pa opravlja pisanje vhodnih vrednosti na analogni izhod DSP2 kartice.

V opisanem podsistemu je morda nekoliko nevsakdanje, da imajo izvorni bloki (npr. DSP2_AI) vhod. Potrebno je povdariti, da to velja samo med simulacijo, medtem ko v generirani kodi DSP procesor dejansko odtipa vrednost npr. iz analognega vhoda (v opisanem primeru iz analognega vhoda 1). Takšni »izvorni« bloki imajo v simulaciji vlogo ojačevalnika z ojačanjem 1, saj je algoritem bloka narejen tako, da se vhodna vrednost bloka prepiše na izhod. Prednost takšnega pristopa je v tem, da pri prehodu iz simulacije v delovanje na DSP2 kartici, Simulinkovega modela ni potrebno popolnoma nič spremeniti.

Zaradi že navedenih omejitev je podsistem »Regulator« sestavljen le iz diskretnih blokov, medtem ko je podsistem »Proga« sestavljen iz zveznih Simulinkovih blokov. Ko smo zadovoljni z rezultati simulacije, se preide iz simulacije v delovanje v realnem času na DSP2 kartici v dveh korakih:

1. Najprej se požene generiranje kode za podsistem »Regulator«.
2. Ko se generiranje kode zakjuči, se s pomočjo DSP Terminala generirana koda naloži na DSP2 kartico.

V stanju inicializacije DSP2 kartice se kreira tudi uporabniški vmesnik DSP Terminala. Vsebina uporabniškega vmesnika je odvisna od uporabljenih blokov DSP2 knjižnice v Simulinkovem podsistemu »Regulator«. V ta namen se uporabljata bloka DSP2_FT (*From Terminal*) in DSP2_TT (*To Terminal*). Slednji služi le za prikaz določenih spremenljivk, medtem ko blok DSP2_FT omogoča spreminjanje vrednosti spremenljivk DSP procesorja. Zraven imena spremenljivke se v DSP Terminalu namreč pojavi vnosno polje, ki omogoča vnos novih vrednosti. Po vpisu nove vrednosti se le-ta prenese iz DSP Terminala na DSP2 kartico.

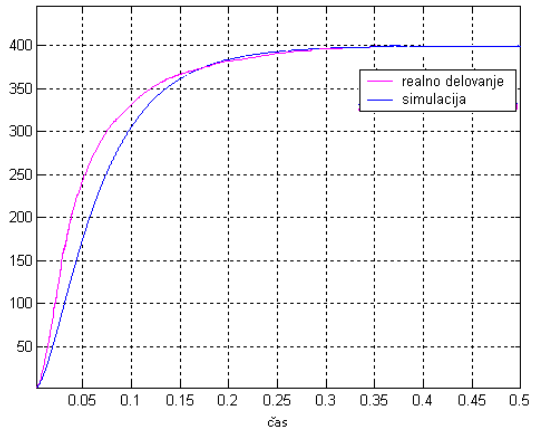


Slika 7: DSP2 Terminal

V opisanem primeru bi lahko bilo ojačanje PID regulatorja upoštevano v prenosni funkciji regulatorja, vendar je namenoma ločeno. Kot je bilo že navedeno, DSP2 knjižnica omogoča sprotno spreminjanje skalarnih vrednosti. Po generiranju kode in naložitvi le-te, se parameter *Kp* (v Simulinku ga je potrebno označiti kot *Tunable parameter*) pojavi na strani *Parameter*s DSP Terminala. Na tak način je možno ojačanje PID regulatorja »fino« nastaviti v sprotnem (*online*) režimu delovanja.

5.1 Rezultati

Na sliki 8 so prikazani rezultati simulacije in realnega delovanja. Rezultati se razlikujejo predvsem zaradi nepopolne identifikacije parametrov člena 2. reda.



Slika 8: Odziv sistema na vzbujanje

S dodatnim nastavljanjem parametrov PID regulatorja in boljše identifikacije sistema, bi bilo moč doseči boljše rezultate, vendar je potrebno povdariti, da je bil osnovni namen primera pokazati uporabo DSP2 knjižnice v realni aplikaciji, in ne optimizacija parametrov PID regulatorja. V prikazanem primeru se je algoritem PID regulatorja na DSP2 karti izvajal približno 100µs, medtem ko je bila perioda prekinitvenih rutin nastavljena na 150µs.

6 Zaključek

V članku je bil predstavljen razvojni modul DSP2, povdarek pa je bil predvsem na Simulinkovem dodatku Real-Time Workshop in DSP2 knjižnici za Simulink.

RTW je odličen dodatek simulacijskemu programu Matlab-Simulink, saj iz Simulinkove sheme na hiter in preprost način generira dobro optimizirano C kodo, katere velikost je primerljiva s ročno kodirano kodo. RTW omogoča hiter prehod iz simulacije v delovanje v realnem času na ciljni strojni opremi. Takšen pristop k načrtovanju sistemov omogoča, da inženirji usmerijo energijo v njihove primerne naloge, torej na načrtovanju sistemov, in ne v kodiranje, kot je bilo do sedaj v stalni praksi. Z avtomatskim generiranjem C kode se je moč izogniti številnim napakam, ki se nehote prikradejo v program pri ročnem kodiranju. Dobra lastnost simulacijskega programa Matlab-Simulink je njegova odprta arhitektura, saj omogoča vključevanje lastnih blokov v okolje Simulink in izgradnjo lastnega ciljnega sistema.

Predstavljen razvojni modul DSP2, s pripadajočo knjižnico za Simulink, se bo na Fakulteti za elektrotehniko računalništvo in informatiko v Mariboru uporabljal pri pedagoškem procesu. Pri poučevanju dinamičnih sistemov je namreč pomembno, da se načrtovanje sistemov vodenja ne zaključi s simulacijo v Matlabu, temveč se načrtan sistem (npr. diskretni regulator) preveri tudi na realnem objektu. Delovanje na realnem sistemu se zaradi neupoštevanja raznih omejitev, nelinearnosti itd. v matematičnem modelu, razlikuje od simulacije izvedene v Matlabu. Tako je za dosego željenega zaprtozančnega odziva potrebna fina nastavitev parametrov regulatorja

v »online« načinu, kar DSP2 knjižnica tudi omogoča. DSP2 knjižnica podpira tudi vključevanje *StateFlow* diagramov v Simulinkov model, kar omogoča zraven regulacijskih algoritmov tudi izvedbo krmilnih krmilni algoritmov na DSP2 kartici.

7 Literatura:

[1] FERI Maribor: *DSP2 Users's Manual,* verzija t3, marec 2001, Inštitut za robotiko, FERI Maribor.

[2] FERI Maribor: *DSP Terminal Users's Manual,* verzija t3, marec 2001, Inštitut za robotiko, FERI Maribor.

[3] FERI Maribor: *DSP2 Library for Simulink User's Manual*, marec 2003, Inštitut za robotiko, FERI Maribor.

[4] The MathWorks, Inc.: *Real-Time Workshop User's guide (rtw_ug.pdf),* verzija 3, januar 1999, www.mathworks.com.

[5] The MathWorks, Inc.: Target Language Compiler Reference Guide (tlc_ref.pdf), verzija 1.2, januar 1999, www.mathworks.com.

[6] The MathWorks, Inc.: Writing S-Functions (sfunctions.pdf), verzija 3, oktober 1998, www.mathworks.com.

[7] The MathWorks, Inc.: Using Simulink (sl_using.pdf), verzija 3, oktober 1999, www.mathworks.com.

[8] Texas Instrument, Inc.: Code Composer User's Guide (SPRU296A.pdf), oktober 1999, www.ti.com.