Elektrotehniški vestnik 76(5): 305-310, 2009
Electrotechnical Review: Ljubljana, Slovenija

FPGA implementacija generatorja profila s pomočjo PLL

Robert Horvat, Karel Jezernik

Univerza v Mariboru, FERI, Smetanova ulica 17, 2000 Maribor
E-pošta: robert.horvat@uni-mb.si

Povzetek. V delu je predstavljena primerjava sistema DSP in vezja programirljivih logičnih vrat FPGA. Podrobno je opisan generator profila s pomočjo PLL, ki je realiziran tako na DSP kot na vezju FPGA. Eksperimentalni model je zgrajen s pomočjo Digilentove razvojne plošče Nexys, ki je podrobneje opisana. Prikazana je primerjava rezultatov, dobljenih na sistemu DSP in vezju FPGA.

Ključne besede: dogodkovno vodenje, PLL, DSP sistem, FPGA vezje, VHDL

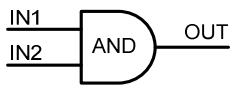
FPGA implementation of PLL – based profile generator

Extended abstract. The paper presents realization of based profile generator on FPGA (field programmable gate arrays). Advantages of FPGA are shows (performs the entire procedures with concurrent operation, higher frequency of activity, very simple implementation of logical function), as also disadvantage (fix-point calculating). Output of based profile generator is function of 2nd order filter, which means that output value is all the time continuous function. Damping and frequency can be set randomly. Simulation and experimental results are shown to validate the strategy. Experimental model is implemented on Nexys Digilent test board.

Keywords: event direction, PLL, DSP system, FPGA circuit, VHDL

1 Uvod

Za implementacijo zelo hitrih sistemov, kjer frekvenca delovanja digitalnega signalnega procesorja (DSP) ne zadostuje več, čedalje pogosteje uporabljamo polja programabilnih logičnih vrat (Field-programmable gate array - FPGA). Prednost vezja FPGA glede na sistem DSP je v vzporednem načinu izvajanja, ki omogoča, da se vsi procesi izvajajo sočasno, torej s številom procesov ne vplivamo na frekvenco delovanja. Pri sistemih DSP (zaporedni način izvajanja) se s kompleksnostjo programa (večje število vrstic) zmanjšuje frekvenca delovanja. Tako lahko z vezjem FPGA dosegamo višje frekvence delovanja [6]. FPGA je elektronsko vezje, sestavljeno iz logičnih vrat, kar je zelo primerno za dogodkovno vodenje. O dogodkovnem vodenju govorimo, ko za spremembo izhodnega stanja ne potrebujemo časa urnega cikla, ampak se to zgodi takoj, ko imamo izpolnjen pogoj za drugo stanje. Primer takšnega vodenja so logična vrata. Pri logičnih vratih (slika 1) se izhodno stanje OUT spremeni glede na vrednosti IN1 in IN2, ne da bi za to potrebovali čas (urn cikel).

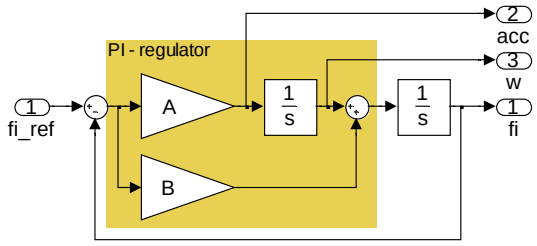


Slika 1: Logična vrata AND
Figure 1. Logic gate AND

Slabost vezja FPGA pa je izvajanje računskih operacij. V nasprotju s sistemom DSP, kjer računamo z enoto CPU, se pri vezju FPGA računa s celimi števili (fix point). Oba sistema je mogoče reprogramirati. Poleg vseh zgoraj opisanih lastnosti je vezje FPGA tudi ekonomsko zanimivo orodje, ki je finančno zelo ugodno, programska oprema pa je brezplačno na voljo na spletu.

Za natančno vodenje mehatronskih mehanizmov, kjer je zahtevana natančno določena trajektorija giba z natančno določeno hitrostjo in pospeškom, potrebujemo poleg čim zmogljivejšega procesorja tudi dobro zasnovan generator profila. Kot osnovni pogoj generatorja profila je nastavitev referenčnega položaja, ki ga dvakrat odvajamo ter tako dobimo profil hitrosti in pospeška. Poznamo preproste primere, kjer kot referenčni položaj uporabimo pravokotni ali trapezni signal. Vendar pa takšna oblika referenčnega signala ustvari hitre spremembe pri referenčni hitrosti in pospešku. Poznamo tudi bolj kompleksne generatorje profila, kot je $\sin^2$ profil, ki generira hitrost in pospešek brez hitrih sprememb, vendar je tudi precej bolj kompleksen in teže izvedljiv z vezjem FPGA. PLL je metoda, ki generira zvezno hitrost in pospešek, kljub temu pa je še vedno dokaj preprosta za realizacijo s pomočjo vezja FPGA. PLL (slika 2) omogoča dvojno odvajanje poljubnega referenčnega signala s pomočjo dveh integratorjev [1], [2].

Prejet 14. julij, 2009
Odobren 6. november, 2009



Slika 2: Blokovna shema PLL - MATLAB
Figure 2. PLL block diagram - MATLAB

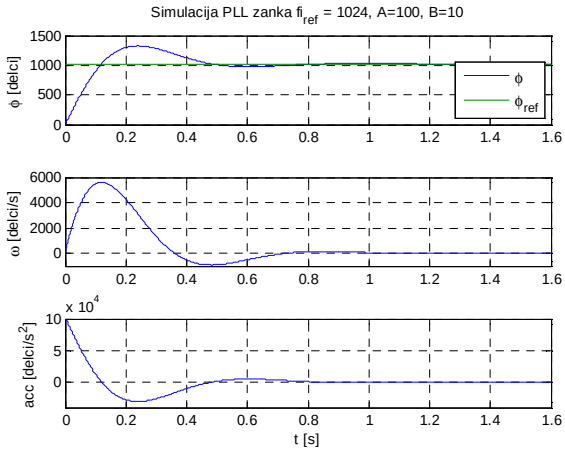
PLL je člen 2. reda s prenosno funkcijo (1),

$$\frac{\varphi(s)}{\varphi_{ref}(s)} = \frac{sB + A}{s^2 + sB + A}, \quad (1)$$

kar pomeni, da lahko s pomočjo ustrezno izbranih vrednosti A in B določimo lastno frekvenco in dušenje izhodnega signala.

2 Izvedba zanke PLL v okolju MATLAB

S pomočjo orodja MATLAB/Simulink je bila izvedena simulacija delovanja PLL (slika 3), s pomočjo katere dobimo referenčne signale, ki jih bomo pozneje primerjali z rezultati, dobljenimi z vezjem FPGA.

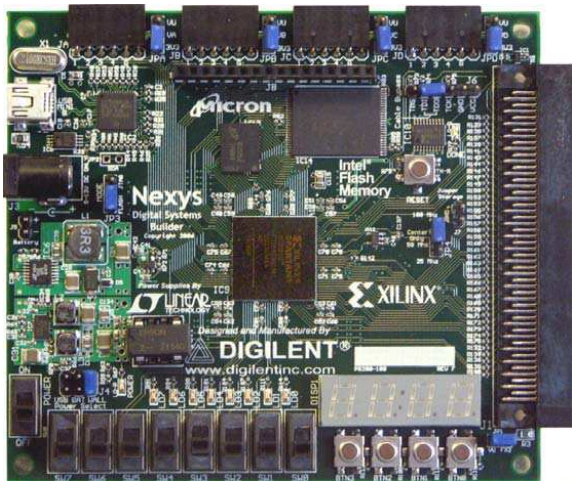


Slika 3: Stopnični odziv PLL– simulacija
Figure 3. PLL step response – Simulation

Ker se izvaja algoritem na vezju FPGA v celoštevilčnem okolju, smo tudi simulacije izračunavali s celimi števili.

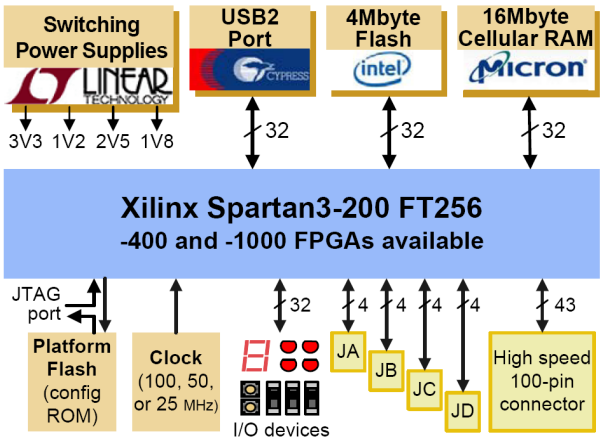
3 Eksperimentalna plošča Nexys DIGILENT

Poskus generatorja profila s pomočjo PLL izvedemo z razvojno ploščo Nexys (slika 4), ki je namenjena izvajanju algoritmov na FPGA in jo lahko programiramo s pomočjo VHDL ((VHSIC (Very High Speed Integrated Circuits) hardware description language)) kode [3].



Slika 4: Plošča Nexys DIGILENT
Figure 4. DIGILENT's Nexys circuit board

Testna plošča [4] vsebuje 100 prosto dostopnih vhodno-izhodnih lokacij, 8 stikal, 8 LED, 4 tipke in 4 sedemsegmentni LED prikazovalnik,…(slika 5). Programiranje plošče Nexys izvajamo preprosto in hitro s pomočjo osebnega računalnika, ki ga s ploščo povežemo prek komunikacije USB.

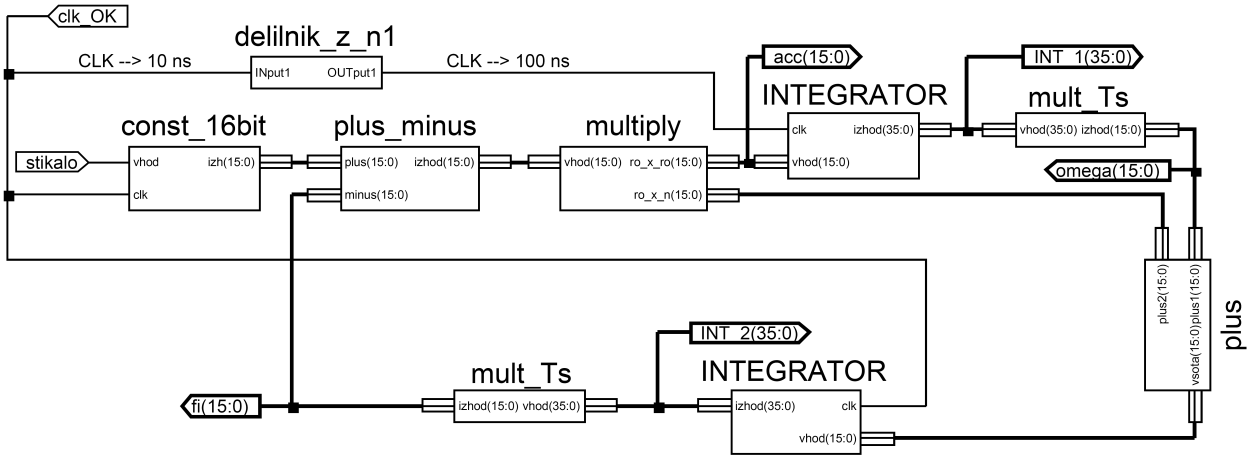


Slika 5: Blokovni diagram Nexys DIGILENT
Figure 5. DIGILENT's Nexys block diagram

Uporabljamo lahko izvedbo vezja z 400.000 ali 1.000.000 logičnimi vrati.

4 Izvedba PLL v okolju ISE

Program ISE je namenjen načrtovanju kode VHDL ali Verilog za programiranje vezja FPGA [5]. Načrtovanje kode VHDL ali Verilog lahko realiziramo s pomočjo diagrama stanj, logičnimi elementi ali neposredno z zapisom kode [6]. V našem primeru je algoritem zapisan v kodi VHDL [7]. Zaradi lažjega testiranja in razumevanja pa je koda razdeljena na manjše sklope (slika 6).



Slika 6: Blokovna shema PLL – ISE
Figure 6. PLL block diagram – ISE

Blokovna shema (slika 6) je sestavljena iz dveh integratorjev, množilnika, seštevalnika in generatorja referenčnega signala, ki bodo podrobneje opisani v nadaljevanju [8]. Odločili smo se za 16-bitno ločljivost podatkov o položaju, hitrosti in pospešku, za kar so prilagojene vse komponente PLL.

4.1 Definicija enot

Če uporabimo PLL za izračun hitrosti in položaja, so vse enote v blokovni shemi definirane glede na izbrano vrednost podatka o referenčnem položaju (2),(3),(4).

PRIMER:

$$2\pi = h07\mathrm{D}0 = 2000$$

- fi_ref(15:0) → 16 bitno predznačeno število [-32767 do 32767].

$$\varphi = h0001 = \frac{2\pi}{2000} = 0,00314\left[\frac{rad}{d}\right] \qquad (2)$$

- acc(15:0) → 16 bitno predznačeno število [-32767 do 32767].

$$acc = h0001 = \frac{2\pi}{2000} = 0.00314\left[\frac{rad/s^2}{d}\right] \qquad (3)$$

- omega(15:0) → 16 bitno predznačeno število [-32767 do 32767].

$$\omega = h0001 = \frac{2\pi}{2000} = 0.00314\left[\frac{rad/s}{d}\right] \qquad (4)$$

4.2 Generator referenčnega položaja – const_16bit

S pomočjo stikala začnemo generirati pravokotne pulze, ki pomenijo referenčni položaj (16-bitna predznačena vrednost). Maksimalna amplitudna razlika je omejena z rezultatom množenja v bloku 'multiply' (4.4).

4.3 Seštevalnik – plus, odštevalnik – plus_minus

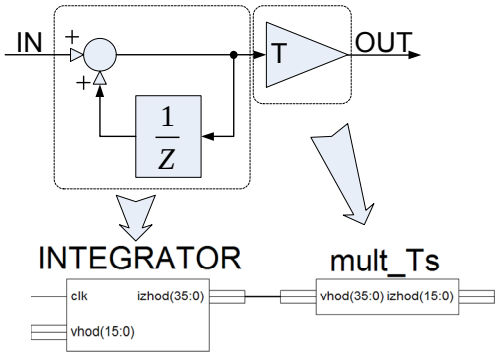
Oba elementa pomenita operacijo seštevanja ali odštevanja 16-bitnega predznačenega števila. Rezultat seštevanja ali odštevanja mora biti znotraj območja 16-bitnega predznačenega števila.

4.4 Množilnik – multiply

Pri množenju dveh 16-bitnih števil dobimo 32-bitni rezultat, od katerega uporabimo spodnjih 16 bitov, kar pomeni, da rezultat množenja vhodnega signala in konstante ne sme presegati maksimalne vrednosti 16-bitnega predznačenega števila.

4.5 Integrator

Integrator je sestavljen iz integracijskega dela in množilnika, ki sta v ISE realizirana posebej (slika 7).



Slika 7: Blokovna shema integratorja - ISE
Figure 7. Integrator block diagram – ISE

Vhod v integrator je 16-bitno predznačeno število, spodnjo in zgornjo omejitev integratorja pa pomeni velikost izhoda integratorja, kjer smo omejeni na 36-bitna predznačena števila. 36-bitno vrednost v integratorju množimo s časom periode urnega takta, ki je manjša od sekunde. Takšno množenje pri celoštevilčnem množenju izvedemo po postopku (5), prikazanem na primeru:

$$
\begin{aligned}
&clk = 100ms \\
&vh = h0300 = 768 \\
&h\text{FFFF} = 65535 = 1\ [s] \\
&h0041 = 65,535 = 1\ [ms] \\
&rezultat = clk \cdot vh = XXXXxxxx \\
&rezultat = 100 \cdot 65,535 \cdot 768 = 5033088 \\
&rezultat = h4\text{CCC80} = h004\text{C}
\end{aligned}
\qquad (5)
$$

Kot rezultat uporabimo samo najvišjih 16-bitov.

5 Prikaz rezultatov

Primerjali smo rezultate, dobljene v simulacijskem okolju Matlab/Simulink, simulacijske rezultate programa ISE (vhdl koda), in eksperimentalne rezultate, dobljene na plošči Nexys Digilent.

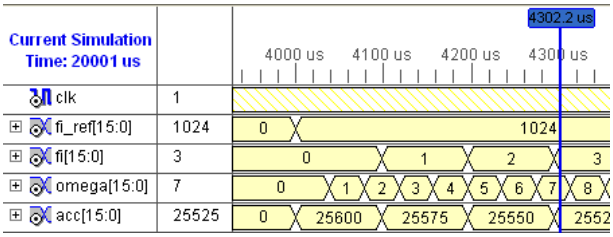
PLL je bila izvedena najprej v okolju MATLAB/Simulink, nato pa še v okolju ISE. Rezultati dobljeni v Simulinku, so referenčni signali, ki smo jih pozneje primerjali z rezultati, dobljenimi v okolju ISE.

5.1 Simulink

V Simulinku sestavimo blokovno shemo (Slika 2), nastavimo potrebne parametre in izrišemo izhodne signale (Slika 3).

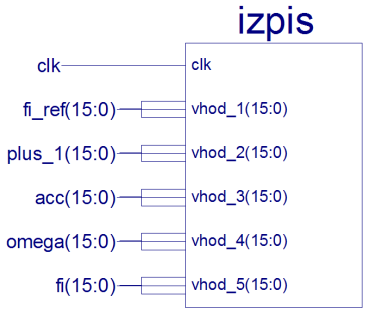
5.2 Simulator ISE

ISE omogoča simulacijo s programom Simulator ISE, vendar pa je rezultat predstavljen le v številčni obliki (slika 8), brez možnosti grafične predstavitve.



Slika 8: Simulacijsko okolje Simulator ISE
Figure 8. ISE Simulator simulation tool

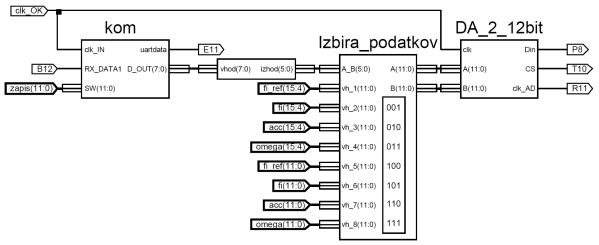
Za prikaz simulacijskih podatkov v grafični obliki je bilo treba zagotoviti zapis podatkov ob vsakem urnem taktu v tabelo. Podatke iz tabele pa lahko potem preprosto s pomočjo orodja Matlab izrišemo in po potrebi tudi skaliramo. Zapis podatkov v tabele zagotovimo z dodatnim blokom (slika 9) v shemi ISE.



Slika 9: Blok za tabeliranje simulacijskih podatkov
Figure 9. Block for editing simulation data

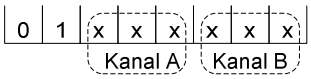
5.3 Poskus

Za prikazovanje rezultatov na plošči Nexys uporabimo dva 12-bitna DA pretvornika. Analogni izhod pa nato opazujemo s pomočjo osciloskopa. Za izvedbo pretvornika DA uporabimo LTC1446, ki omogoča osveževanje podatkov vsake 2µs. To lahko na osciloskopu spremljamo vse podatke le, če je perioda osnovnega urnega takta večja od 2µs. Ker imamo le dva pretvornika DA, spremljati pa želimo več različnih signalov, smo dodali še dodatne bloke. Blok za komunikacijo z računalnikom (RS232), za izbiro podatka in pretvornik DA (slika 10).



Slika 10: Blokovna shema komunikacije PC-Nexys-DA
Figure 10. Communication PC-Nexys-DA block diagram

Za izbiro kanala, ki ga želimo opazovati uporabimo pošiljanje enega znaka (char), ki je sestavljen iz 8 bitov uporabljamo zadnjih šest bitov (Slika 11).

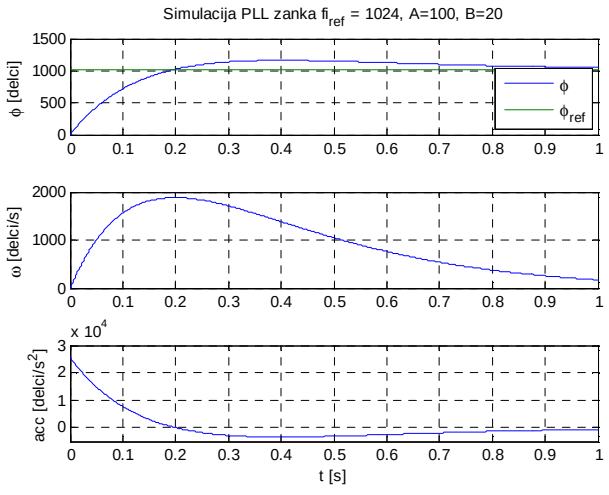


Slika 11: Izbira podatka na vsakem kanalu pretvornika DA
Figure 11. Choosing the data on every DAC channel

6 Rezultati

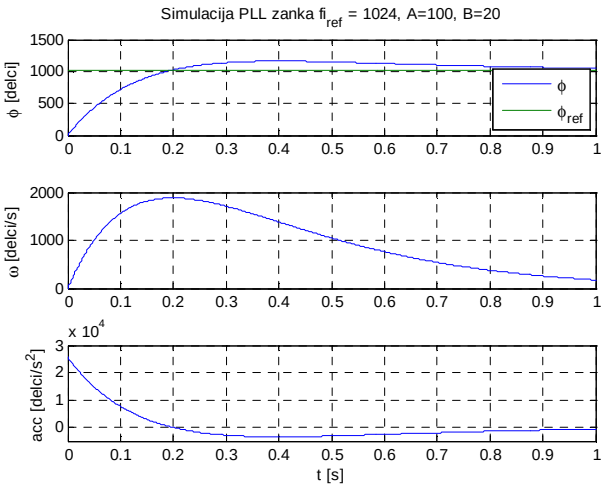
Izvedene so bile primerjave rezultatov simulacije v Matlab/Simulinku, Simulator ISE in rezultati eksperimentalnega modela. Meritve so bile izvedene pri različnih vrednosti A in B, torej pri različnih lastnih frekvencah in različnih vrednostih dušenja.

Stopnični odziv zanke PLL pri nastavitvi parametrov A=25, B=10 in referenčni vrednosti 1024 delcev.



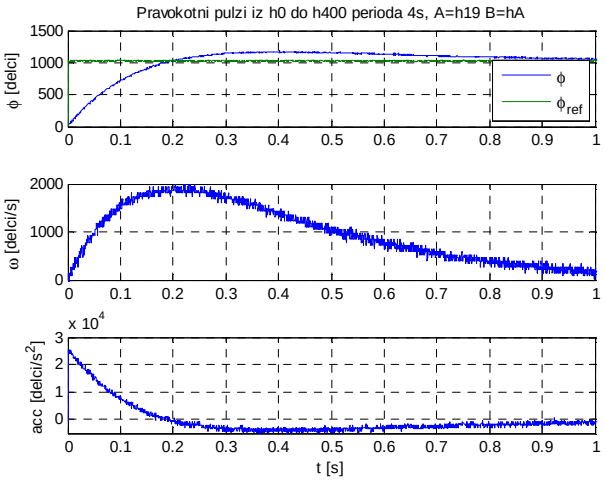
Slika 12: Stopnični odziv – Matlab/Simulink
Figure 12. Step response – Matlab/Simulink

Enake meritve izvedemo tudi v programu ISE Simulator. Rezultati (sliki 12 in 13) se ujemajo, kar pomeni, da PLL deluje tudi v okolju VHDL pravilno.



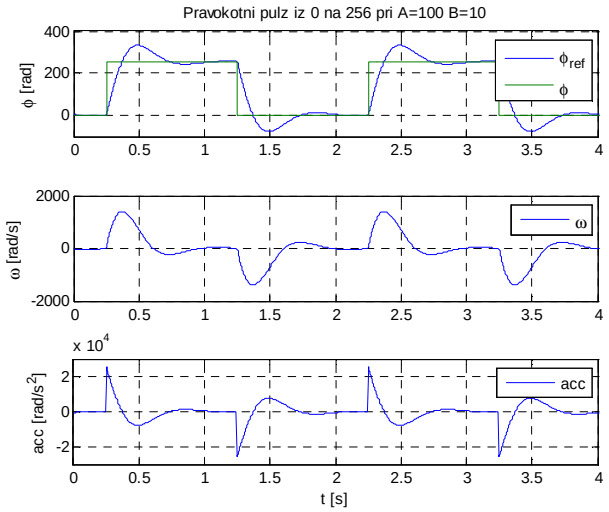
Slika 13: Stopnični odziv – ISE Simulator
Figure 13. Step response – ISE simulator

Enak postopek ponovimo tudi na eksperimentalni plošči, kjer pa je meritev nekoliko manj natančna. Vzrok za manjšo natančnost lahko pripišemo prepočasnemu pretvorniku DA in samemu merilnemu postopku osciloskopa (vedno je prisoten šum). Vsak šum pa na koncu zaradi potrebnega skaliranja še dodatno ojačimo.

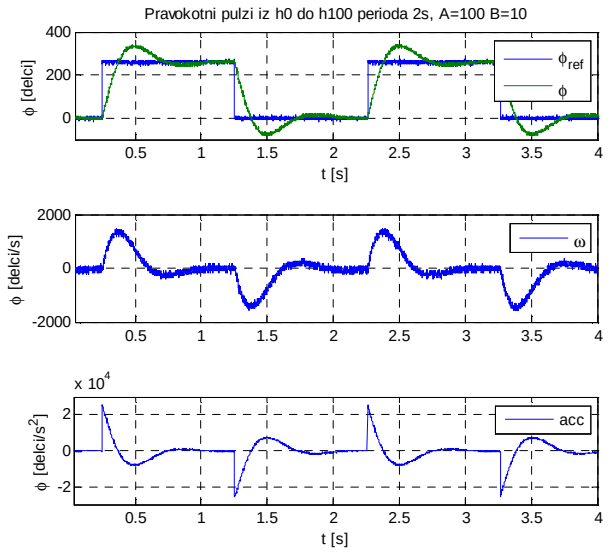


Slika 14: Stopnični odziv – Nexys Digilent
Figure 14. Step response – Nexys Digilent

Primerjali smo tudi odzive zanke PLL pri višji lastni frekvenci in manjšem dušenju. A=100, B=10.



Slika 15: Pravokotni pulzi – ISE Simulator
Figure 15. Pulse response – ISE Simulator



Slika 16: Pravokotni pulzi – Nexys Digilent
Figure 16. Pulse response – Nexys Digilent

7 Sklep

Za mehatronske aplikacije, kjer je zahtevana visoka dinamika, je smotrno izbrati vezje FPGA, ki zagotavlja zelo visoko frekvenco delovanja procesa. Zaradi vzporednega delovanja vseh procesov se lahko le-ti izvajajo sočasno, kar zagotavlja visoko frekvenco delovanja procesa ne glede na to, koliko procesov se izvaja sočasno. Celotno delovanje vezja FPGA temelji na celoštevilčnih vrednostih, kar pomeni določene težave pri izvajanju računskih operacij. Prav tako pa moramo za vse podatke predhodno poznati vse maksimalne vrednosti, kot tudi ali bo vrednost predznačena ali ne. Tako namreč lahko določimo dolžino vektorja za vsak podatek.

Za izvedbo generatorja profila izberemo PLL, ki jo je dokaj preprosto izvesti s pomočjo vezja FPGA, programiranega v VHDL, še vedno pa pomeni generator profila z zveznimi izhodnimi veličinami.

Izkazalo se je, da se lahko sistem, izveden na sistemu DSP uspešno prenese v vezje FPGA, programiran v VHDL, če upoštevamo vse omejitve glede velikosti podatkov, kot tudi možnosti izvajanja računskih operacij.

8 Literatura

[1] Arruda, L.N.; Silva, S.M.; Filho, B.J.C.; PLL structures for utility connected systems, Universidade Federal de Minas Gerais , IEEE 2001

[2] Guan-Chyun Hsieh, James C. Hung, Phase-Locked Loop Techniques-A Survey, National Taiwan Institute of Technology, IEEE 1996

[3] Frank Vahid, Roman Lysecky, VHDL for digital design, University of California, Riverside, University of Arizona, 2007

[4] Digilent Nexys Board Reference Manual, Februar 2007

[5] Andrej Trost, Andrej Žemva, Baldomir Zajc, Modularni FPGA prototipni sistem za obdelavo slik, Univerza v Ljubljani, Fakulteta za elektrotehniko, Elektrotehniški vestnik 69(1):1-6, 2002

[6] B.Bomar, Implementation of a micro programmed control in FPGA, IEEE Trans. Ind. Electron., vol. 49, no. 2, pp. 415-422, Apr. 2002

[7] Pong P. Chu, FPGA prototipyng by vhdl examples, Cleveland State University, 2008

[8] E. Monmasson, M. N. Cirstea, FPGA design methodology for industrial control systems – A review, IEEE Trans. Ind. Electron., vol. 54, no. 4, pp. 1824-1842, Aug. 2007

Robert Horvat je diplomiral leta 2007, zdaj je mladi raziskovalec in podiplomski študent na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru, kjer raziskuje v laboratoriju za robotiko.

Karel Jezernik je doktoriral leta 1976 na Fakulteti za elektrotehniko in računalništvo Univerze v Ljubljani. Na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru je zaposlen kot redni profesor in predstojnik Inštituta za robotiko.