Preizkušanje modelov vezij za digitalno obdelavo signalov

Andrej Trost, Andrej Žemva

Fakulteta za elektrotehniko, Univerza v Ljubljani, Tržaška cesta 25, 1000 Ljubljana
E-mail: andrej.trost@fe.uni-lj.si

Abstract. *Field Programmable Gate Arrays (FPGA) are extensively used for implementation of digital signal processing algorithms due to their performance and programmability. There are several non-commercial and freely available design tools and languages suitable for research, prototyping and educational purpose.*

In the paper, we outline digital signal processing circuit design methodology targeting commonly used FPGA devices with freely available design tools. An efficient development flow requires design of customized test and verification interfaces. The signal generating IP components and hardware verification structures are presented and a testing system case study is evaluated.

1 Uvod

Programirljiva vezja se pogosto uporabljajo za izvedbo algoritmov digitalne obdelave signalov v namenski strojni opremi [1]. Vezja FPGA vsebujejo gradnike kombinacijskih in sekvenčnih vezij ter namenske komponente za digitalno obdelavo signalov DSP [2].

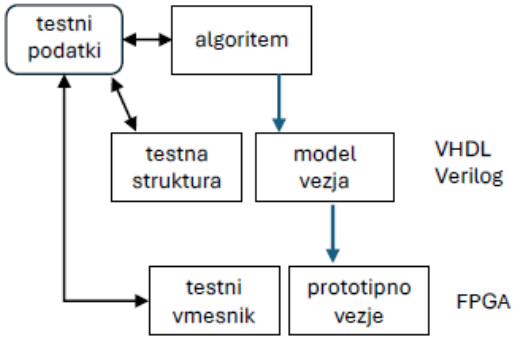
Razvojna orodja za načrtovanje digitalnih vezij v tehnologiji FPGA podpirajo standardna strojno-opisna jezika VHDL in Verilog, ki nista posebej prilagojena za načrtovanje vezij s področja digitalne obdelave signalov. Za učinkovito delo lahko uporabimo opis algoritmov v programskem jeziku in visokonivojsko sintezo [3] ali pa namenska komercialna orodja. Prednost načrtovanja v standardnih jezikih je prosta dostopnost orodij za raziskovalne in učne namene in prenosljivost modela vezja v različne tehnologije.

Algoritme digitalne obdelave signalov je potrebno prilagoditi za optimalno izvedbo v logičnih vezjih. Računske operacije z realnimi števili porabijo veliko logičnih gradnikov, zato jih nadomestimo z operacijami s števili v zapisu fiksne vejice ali celo s celoštevilskimi operacijami [4]. Razgradnja algoritma na osnovne računske operacije omogoča masovno paralelno izvedbo operacij in optimalno izkoristi zmogljivosti namenskega vezja.

V prispevku opisujemo metodologijo načrtovanja vezij za digitalno obdelavo signalov v jezikih VHDL in Verilog s prosto dostopnimi orodji. Načrtovalski potek vključuje testne digitalne komponente, ki so primerne za simulacijo in sintezo vezja. Predstavljene in ovrednotene so komponente, ki generirajo digitalne signal, in testne strukture za preverjanje strojne opreme. Postopek predstavimo in ocenimo na primeru razvoja in testiranja digitalnega sistema za računanje minimuma korelacije dveh harmoničnih signalov.

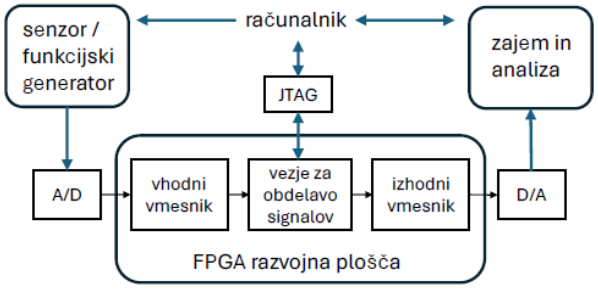
2 Načrtovanje in preizkušanje vezij

Slika 1 prikazuje tipičen potek načrtovanja vezij za obdelavo signalov, ki se začne z razvojem in preizkušanjem algoritmov s testnimi podatki. Prvi korak izvajamo v matematičnih orodjih ali splošnih programskih jezikih, kot sta C++ ali Python. V naslednjem koraku izdelamo model vezja v strojno-opisnem jeziku in verificiramo delovanje na simulaciji s testno strukturo. Testna struktura generira referenčne vhodne signale ali pa kar uporabi enake testne podatke, kot pri razvoju algoritma. Simulacija je časovno zahtevnejši postopek, zato preizkušamo modele vezja z omejenim naborom testnih podatkov.



Slika 1: Potek načrtovanja in preizkušanja namenskih vezij za izvedbo algoritmov digitalne obdelave signalov

Strojna verifikacija vezja poteka na prototipni razvojni plošči z implementiranim vezjem in testnimi vmesniki. Slika 2 prikazuje tipičen potek signalov pri preizkušanju na razvojni plošči z vezjem FPGA. Analogni vmesniki so del razvojne plošče [5] ali pa v obliki razširitvenih modulov. Kontroliran preizkus delovanja izvajamo s funkcijskim generatorjem namesto senzorja in analiziramo izhodni signal. Vmesnik JTAG se uporablja za programiranje in razhroščevanje logičnega vezja na razvojni plošči.



Slika 2: Preizkušanje prototipnih vezij za obdelavo signalov na razvojni plošči z vezjem FPGA

Postopek preizkušanja prototipnih vezij za obdelavo signalov lahko pohitrimo tako, da naredimo generator signalov znotraj vezja FPGA. Testiranje v tem primeru izvajamo na splošno-namenski programirljivi razvojni plošči, saj ne potrebujemo specifičnih vmesnikov za analogne signale. Strojna verifikacija z vezjem FPGA poteka v realnem času, kar omogoča obsežnejše preizkuse v primerjavi s simulacijo.

3 Testne komponente v digitalnem vezju

Pri strojni verifikaciji potrebujemo komponente za generiranje signalov in vmesnike za zajem ter prenos rezultatov na računalnik. Vmesniki za razhroščevanje vezij so vključeni v razvojna orodja, med tem ko je potrebno specifične komponente za generiranje signalov razviti v strojno-opisnem jeziku.

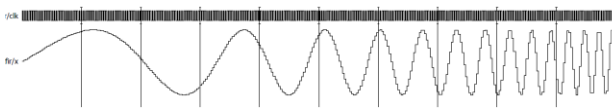
3.1 Digitalni signalni generator

Osnovna testna komponenta je generator harmoničnega signala z nastavljivo frekvenco f in amplitudo A pri podani vzorčni frekvenci f_s:

$$y[i] = A \cdot \sin\left(2\pi\, i \frac{f}{f_s}\right) \qquad (1)$$

Realne vrednosti sinusnega signala predstavimo z dvojiškimi vektorji v zapisu fiksne decimalke ali celih števil. Standardni strojno-opisni jezik Verilog ne podpira trigonometričnih funkcij. Vzorčen sinusni signal zato nadomestimo s tabeliranimi vrednostmi in algoritmom za sprotno računanje vrednosti vzorcev.

Jezik VHDL pozna realna števila in matematično knjižnico s trigonometričnimi funkcijami, ki jih uporabimo za opis generatorja harmoničnega signala v testni strukturi. Za verifikacijo digitalnega sita naredimo generator frekvenčnega preleta sinusnega signala. Primer simulacije frekvenčnega preleta prikazuje slika 3.

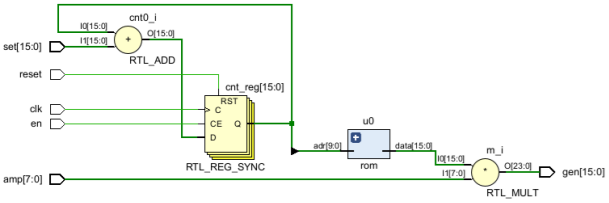


Slika 3: Generiran 8-bitni frekvenčni prelet sinusnega signala

```
-- Generiraj prelet sinusnega signala
singen : process(clk)
 variable a: real := 120.0; -- amplituda
 variable fi: real:= 0.0;
 variable d: real := 0.005; -- korak
 variable si: integer;
begin
 if rising_edge(clk) then
  d := d * 1.01;
  fi := fi + d;
  si := integer(a*sin(MATH_2_PI*fi));
  x <= to_signed(si, 8);     -- kvantizacija
 end if;
end process;
```

Testni proces vsebuje funkcije in stavke, ki niso primerni za sintezo vezja. Digitalni signalni generator naredimo z vezjem numerično krmiljenega oscilatorja (NCO). Sestavljen je iz faznega akumulatorja, pomnilnika ROM z vzorci ene periode signala in amplitudnega množilnika, kot prikazuje slika 4. Izhodna frekvenca je odvisna od frekvence ure f_{clk}, števila bitov akumulatorja N in vhodnega parametra *set*:

$$f = \frac{f_{clk} \cdot set}{2^N} \qquad (2)$$



Slika 4: Zgradba numerično krmiljenega oscilatorja

Model vezja NCO opišemo s parametri, ki določajo kvantizacijo (širino, DSZ) in velikost pomnilnika ROM. V jeziku VHDL modeliramo ROM z zbirko vektorjev in generično zanko za izračun konstantne vsebine:

```
-- generator vsebine sinusne tabele
 g: for i in 0 to romsize-1 generate
 constant sinval: signed (DSZ-1 downto 0) := to_signed(
     integer(A*sin(real(i)*2.0*math_pi/real(romsize))),DSZ);
 begin
   sinrom(i) <= sinval;
 end generate;
```

Tabela 1 prikazuje rezultate sinteze vezja NCO v tehnologijo FPGA Xilinx Artix-7 ob izbranih nastavitvah kvantizacije in velikosti pomnilnika. Najmanjše vezje NCO je v celoti narejeno z vpoglednimi tabelami (LUT) in flip-flopi, večja vezja pa uporabljajo notranje pomnilniške bloke (BRAM) in strojni množilnik (DSP).

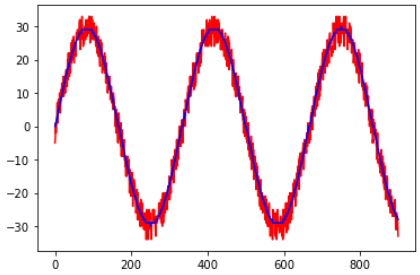
Tabela 1. Poraba FPGA Artix-7 za funkcijski generator

velikost ROM	kvantizacija (bitov)	Poraba gradnikov FPGA DSP; BRAM; LUT; Flip-flopi
64	8	0; 0; 91; 16
256	8	0; 0,5; 107; 16
1k	8	0; 0,5; 109; 16
1k	16	1; 0,5; 53; 16
4k	8	0; 1; 111; 16
4k	16	1; 2; 53; 16

Predstavljena komponenta NCO zahteva zelo malo logičnih gradnikov v sodobnih vezjih FPGA, kjer ostane še veliko prostora za implementacijo vezij za obdelavo signalov. Model generatorja signalov je uporaben tako za simulacijsko testno strukturo kot tudi pri sintezi vezja. Osnovni model NCO je mogoče enostavno nadgraditi na več kanalov, ki so neodvisni ali pa med seboj fazno povezani. Z generatorjem psevdonaključnih vrednosti pa dodamo izhodnemu signalu naključni šum, da dobimo bolj realne testne vzorce, kot prikazuje slika 5.

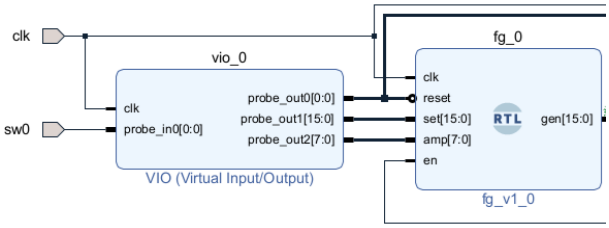
V jeziku Verilog določimo vsebino pomnilnika ROM s konstantami v inicializacijskem bloku ali pa z branjem konstant iz pomnilniške datoteke. Funkcija za branje zahteva konstantno velikost pomnilniške tabele, zato ima

model NCO v Verilogu omejene možnosti parametrizacije. V pomnilniški datoteki lahko shranimo poljubno periodično obliko signala ali pa specifičen testni vektor za preizkušanje vezja za obdelavo signalov.



Slika 5: Vzorci harmoničnega signala z naključnim šumom

3.2 Nastavljanje parametrov



Slika 6: Funkcijski generator priključen na komponento VIO

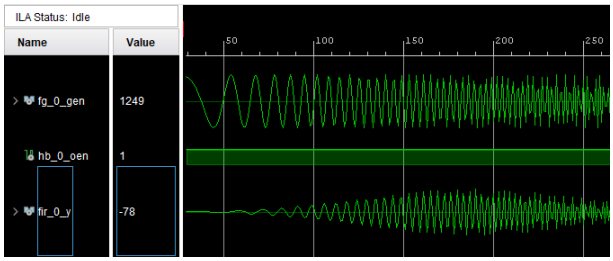
Proces sinteze vezja in tehnološke implementacije je časovno zahteven in vsakokratno prevajanje zaradi spreminjanja parametrov ni smiselno. Za nastavljanje parametrov uporabljamo vmesnike na razvojni plošči ali vgrajene komponente intelektualne lastnine (IP) za razhroščevanje vezja.

Vezja FPGA vsebujejo testni vmesnik JTAG za nalaganje oz. programiranje vezja. Če v vezje vključimo testne IP komponente, omogoča JTAG nastavljanje parametrov in razhroščevanje vezja. Orodje Xilinx Vivado vsebuje IP komponento VIO (angl. Virtual Input/Output), ki ji nastavimo širino ter velikost vhodov in izhodov za nastavljanje parametrov [6]. Na sliki 6 je prikazana blokovna shema testnega sistema za digitalno sito, kjer s komponento VIO nastavljamo parametre funkcijskega generatorja.

3.3 Analizator signalov

Komponenta VIO je primerna le za nastavljanje in opazovanje počasnih signalov, za opazovanje hitrih izhodov v realnem času pa potrebujemo logični analizator. Analizator je komponenta, ki shranjuje vzorce digitalnih signalov v pomnilnik in jih prenaša iz razvojne plošče na računalnik.

Razvojna orodja za vezja FPGA ponujajo analizator signalov v obliki komponent IP. V orodju Vivado je to komponenta ILA (angl. Integrated Logic Analyzer) [7]. ILA vsebuje enega ali več vhodov, prožilno logiko, pomnilnik za shranjevanje signalov in vmesnik JTAG. V razvojnem orodju ima ILA vmesnik za prikazovanje zajetih signalov v obliki časovnega diagrama (slika 7).



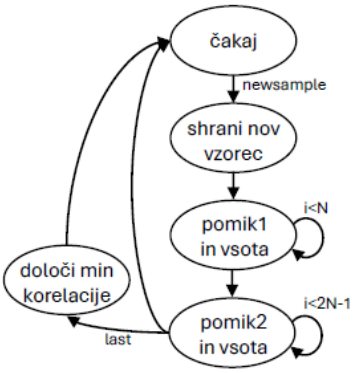
Slika 7: Zajem in prikaz časovnega poteka signalov z vgrajenim logičnim analizatorjem

4 Primer testnega sistema

Uporabo testnih komponent bomo predstavili na primeru modela vezja za iskanje minimuma križne korelacije med avdio signali iz polja mikrofonov. Algoritem išče s korelacijo zamik med sprejetimi signali, kar omogoča določanje smeri izvora zvočnih signalov.

V osnovni izvedbi algoritma predvidevamo na vhodu dva avdio signala, ki ju razdelimo na bloke velikosti 2000 vzorcev. Algoritem računa križno korelacijo med dvema blokoma z zamikanjem do 50 vzorcev v obe smeri. Za preizkus algoritma uporabimo testne harmonične signale in realne vzorce, ki jih s programom v jeziku Python pretvorimo iz zvočnega zapisa WAV v tekstovne datoteke CSV. Datoteke z vzorci zvoka predstavljajo testne podatke za algoritem in simulacijo digitalnega vezja.

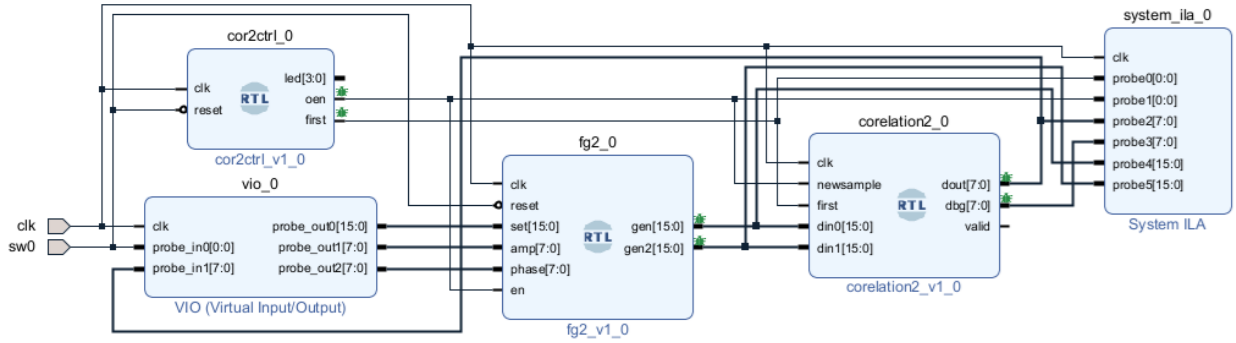
Algoritem križne korelacije je opisan v jeziku C++ in prilagojen za sprotno obdelavo vzorcev zvoka. Slika 8 prikazuje diagram stanj korelacijske funkcije. Ko dobimo nov vzorec vhodnih signalov (newsample), jih shranimo v pomnilnik vrste FIFO (angl. First-In First-Out) globine N=50. Nato izvajamo N iteracij pomikanja prvega pomnilnika in računanja prispevka k vsoti ter pomikanje drugega pomnilnika in prispevka k vsoti.



Slika 8: Postopek računanja korelacije

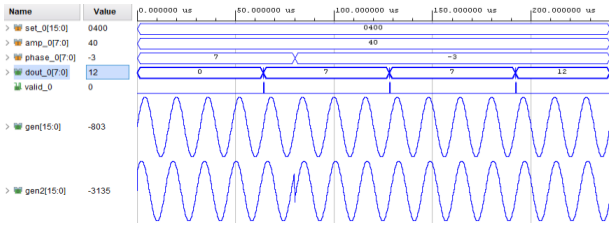
Ko dobimo na vhod zadnji vzorec iz posameznega bloka (signal last), poiščemo minimum skupnih vsot pri posameznih zamikih in vrnemo indeks zamika. Po verifikaciji delovanja algoritma s testnimi podatki naredimo model vezja za korelacijo v jeziku VHDL.

V vezju obdelujemo kvantizirane podatke zapisane v obliki predznačenih dvojiških vektorjev. Vhodni vzorci so 16-bitne vrednosti, korelacijske vsote pa so z eksperimenti določene kot 30-bitna nepredznačena števila. Delovanje vezja preverimo s testno strukturo.



Slika 10: Blokovna shema testnega sistema za računanje korelacije v orodju Vivado

Testna struktura vsebuje generator signalov z dvema fazno zamaknjenima harmoničnima izhodoma. Rezultat algoritma je zaznan fazni zamik po vnaprej določenem številu vzorcev, kot prikazuje simulacija na sliki 9.



Slika 9: Simulacija vezja za določanje minimuma križne korelacije dveh signalov v orodju Vivado

Testni sistem s komponento za računanje korelacije, funkcijskim generatorjem in testnimi komponentami je prikazan na sliki 10. Komponenta VIO nastavlja frekvenco, amplitudo in fazo generiranih signalov ter opazuje rezultat korelacije. Hitre signale zajemamo in spremljamo s komponento ILA. Opazujemo oba vhodna harmonična signala, izhod algoritma, signale za določanje periode vzorcev (oen) in novega izračuna korelacije (first). Na vezju za izvajanje algoritma smo dodali izhod (dbg), na katerem opazujemo notranja stanja med preizkušanjem vezja.

Algoritem za računanje korelacije smo najprej razvili in testirali v jeziku C++, nato pa smo z istimi vhodnimi podatki verificirali model vezja in nazadnje še prototip na razvojni plošči. Za obdelavo 100.000 testnih vzorcev je program potreboval 34 ms, model vezja se je v digitalnem simulatorju izvajal 5,08 s, na razvojni plošči pa poteka obdelava v realnem času.

Preizkus je potekal na programirljivi razvojni plošči Digilent Arty-A7 s frekvenco sistemske ure 100 MHz. Zasedenost vezja FPGA je 10% LUT, 8% flip-flopov, 4% BRAM in 3,3% DSP blokov. S spreminjanjem nastavitev funkcijskega generatorja potrjujemo pravilnost odziva testnega sistema, ki je pripravljen za preizkus z mikrofoni.

5 Zaključek

Standardni strojno-opisni jeziki za modeliranje digitalnih vezij niso posebej prilagojeni za preizkušanje vezij s področja obdelave signalov. V prispevku predstavljamo testne komponente, ki jih potrebujemo za generiranje harmoničnih signalov pri simulaciji takšnih vezij. Če so testne komponente narejene v obliki vezij, primernih za sintezo, kot je NCO, jih lahko vključimo tudi v prototipne sisteme na vezju FPGA. Za nastavljanje parametrov in analizo signalov pa uporabimo komponente IP, ki jih ponuja proizvajalec orodij za vezja FPGA.

Predstavili smo potek načrtovanja in preizkušanja vezij, izdelavo NCO in primer testnega sistema, ki računa minimum korelacije dveh vhodnih signalov.

Zahvala

Delo je bilo sofinancirano iz programa ARIS P2-0415 in projekta: “Spremljanje urbanega hrupa in biodiverzitete za zeleno prihodnost z akustičnim IoT radarjem s klasifikacijo dogodkov na osnovi UI”, ARIS J7-50042.

Literatura

[1] R. Woods, J. McAllister, G. Lightbody, Y. Yi, "FPGA-based Implementation of Signal Processing Systems", John Wiley & Sons, 2017

[2] S. Chen, G. Cai and Z. Huang, "An Enhanced DSP Block Architecture for FPGA Supporting Multi-operands Addition Operation," *2021 IEEE 14th International Conference on ASIC (ASICON)*, Kunming, China, 2021, pp. 1-4

[3] R. Kastner, M. Janarbek, S. Neuendorffer, "Parallel programming for FPGAs", 2018.

[4] A. Przybył, "Fixed-Point Arithmetic Unit with a Scaling Mechanism for FPGA-Based Embedded Systems", Electronics 2021, 10, 1164. https://doi.org/10.3390/electronics10101164

[5] A. Trost, A. Žemva, "Načrtovanje strojne in programske opreme merilne plošče STEMlab", Zbornik enaintridesete mednarodne Elektrotehniške in računalniške konference ERK 2022, 19. - 20. september 2022

[6] LogiCORE IP Virtual Input/Output, LogiCORE IP Product Guide, Xilinx, 2013, https://docs.amd.com/v/u/2.0-English/pg159-vio

[7] Integrated Logic Analyzer v6.2, LogiCORE IP Product Guide, Xilinx, 2016, https://docs.amd.com/v/u/en-US/pg172-ila