Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK 004.3
DOI: https://doi.org/10.24867/01BE09Popovic

VERIFIKACIJA REKONFIGURABILNE ARHITEKTURE ZA HARDVERSKU AKCELERACIJU PREDIKTIVNIH MODELA MAŠINSKOG UČENJA

VERIFICATION OF A RECONFIGURABLE ARCHITECTURE FOR HARDWARE ACCELERATION OF MACHINE LEARNING PREDICTIVE MODELS

Jasna Popović, Rastislav Struharik, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je predstavljena funkcionalna verifikacija rekonfigurabilne arhitekture za hardversku akceleraciju prediktivnih modela mašinskog učenja - Reconfigurable Machine Learning Classifier (RMLC).*

Ključne reči: *funkcionalna verifikacija, mašinsko učenje, hardverska akceleracija, univerzalna verifikaciona metodologija (UVM), SystemVerilog*

Abstract –*In this paper we present functional verification of a reconfigurable architecture for hardware acceleration of machine learning predictive models - Reconfigurable Machine Learning Classifier (RMLC).*

Keywords: *functional verification, machine learninig, hardware acceleration, universal verification methodology (UVM), SystemVerilog*

1. UVOD

Ovaj rad predstavlja digitalnu verifikaciju rekonfigurabilne arhitekture za hardversku akceleraciju prediktivnih modela mašinskog učenja - *Reconfigurable Machine Learning Classifier* (RMLC). Verifikacija je realizovana u Univerzalnoj Verifikacionoj Metodologiji (*eng. Universal Verification Methodology - UVM*) u kombinaciji sa *SystemVerilog* programskim jezikom.

RMLC se sastoji od većeg broja istovetnih blokova i može se konfigurisati da implementira ortogonalna stabla odluke (eng. *Decision Trees - DT*), neortogonalna stabla odluke, nelinearna stabla odluke, funkcionalna stabla odluke, regresiona stabla odluke, linearne i nelinearne *Support Vector Machine* (SVM) prediktivne modele sa polinomijalnim i radijalnim kernelima, veštačke neuronske mreže zasnovane na radijalnim funkcijama (eng. *Artificial Neural Network* - ANN) i višeslojne perceptrone (eng. *Multilayer Perceptron - MLP*) [1]. Verifikaciono okruženje je projektovano na način koji omogućava jednostavnu konfiguraciju i testiranje navedene arhitekture.

1.1. Motivacija

Skorašnji napredak u informacionoj i komunikacionoj tehnologiji doveo je do eksponencijalnog rasta količine podataka skladištenih u bazama podataka.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Rastislav Struharik, red. prof.

Od početka informacionog doba, skupljanje podataka je postalo jednostavnije, a čuvanje informacija jeftinije. Kako se povećava količina mašinski čitljivih podataka, sposobnost razumevanja i primene ovih informacija ne prati korak sa tim rastom. *Data mining* se javlja kao metoda suočavanja sa ovim eksponencijalnim rastom podataka [2].

Mašinsko učenje predstavlja ključnu stavku prilikom izvlačenja značajnih podataka iz prostora dostupnih podataka koji se skupljaju svaki dan. Jedan od izvora ovih podataka su podaci koji su očitani sa senzora. Više podataka je nastalo u poslednje dve godine nego u čitavoj istoriji ljudske rase [3]. Da bi se ovi podaci analizirali, neophodan je hardver koji omogućava brz protok velike količine informacija [3].

Sposobnost automatskog učenja iz ogromne količine podataka je dovela do neverovatnih dostignuća u naučnim poljima kao što su računarski vid (eng. *comuputer vision*), prepoznavanje govora (eng. *speech recognition*) i procesiranje prirodnog jezika (eng. *natural language processing*). Ovi modeli zahtevaju procesiranje velike količine podataka, i veliku računarsku moć za treniranje. Njihov napredak je ograničen postojećim računarima [4]. Trenutno stanje hardverskog ubrzanja se svodi na klastere grafičkih procesora koji služe kao procesori opšte namene. Snažan protivnik grafičkih procesora su FPGA platforme (eng. *Field Programmable Gate Array*), koji za razliku od grafičkih procesora, imaju fleksibilnu hardversku konfiguraciju, i često pružaju bolje performanse u pogledu energetske efikasnosti.

U okviru procesa dizajna hardvera, postoji više verifikacionih procesa. Ti procesi uključuju funkcionalnu verifikaciju, vremensku (eng. *timing*) verifikaciju, test verifikaciju i proveru ekvivalentnosti. Vremenski najzahtevnija je funkcionalna verifikacija. Funkcionalna verifikacija garantuje da logika u čipu i sistemu radi korektno u svim okolnostima, onako kako je definisano u specifikaciji sistema [5].

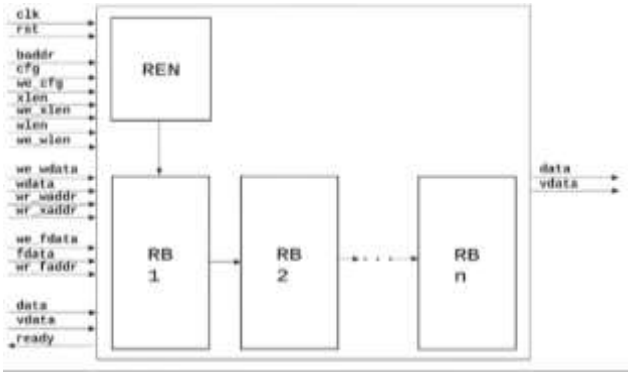
2. OPIS REŠAVANOG PROBLEMA

U ovom poglavlju je dat kratak opis RMLC arhitekture koja se verifikuje. Sadržaj ovog poglavlja se zasniva na radu V. Vranjković [1].

2.1. Detalji arhitekture

Na Slici 1 je prikazana RMLC arhitektura. Arhitekturu čini nekoliko istovetnih blokova koji su međusobno povezani u niz. Ovi blokovi se zovu Reconfigurable

Block (prev. rekonfigurabilan blok - RB) moduli. Broj ovih modula u arhitekturi je tokom dizajna postavljen na 5, i predstavlja parametar arhitekture koji se može podesiti tokom dizajna sistema. Pored RB modula, u arhitekturi se nalazi i Reconfigurable Enable – RE modul, čija je odgovornost da bira koji RB modul se konfiguriše tokom programiranja sistema.

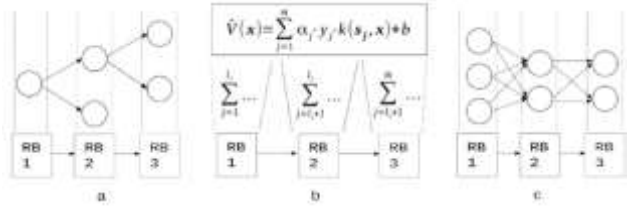


Slika 1. *Blok šema RMLC arhitekture*

RMLC arhitektura implementira tri različita klasifikatora: stabla odluke, SVM, i veštačke neuronske mreže. Kada je RMLC konfigurisan da radi kao stablo odluke, tada će na izlazu arhitekture biti sračunata klasa ulazne instance. Prilikom mapiranja na RB module, svi čvorovi i listovi u jednom nivou stabla su preslikani u jedan RB modul. Kada je RMLC konfigurisan da radi kao SVM, na izlazu klasifikatora će se naći izračunata suma funkcije za predikciju:

$$v(s) = \sum_{i=1}^{m} y_i \alpha_i K(x_i, s) + b \,, \qquad (1)$$

gde je s ulazni podatak. Znak $v(s)$ je klasa ulaznog primera. Prilikom mapiranja, ova suma se deli na delimične sume. Svaka od dobijenih delimičnih suma se potom računa u pojedinačnom RB modulu. I na kraju, kada je RMLC konfigurisan da radi kao veštačka neuronska mreža, na *data* portu će biti izlazne vrednosti za svaki neuron izlaznog sloja, Prilikom mapiranja, svi neuroni jednog skrivenog sloja se preslikavaju u jedan RB modul. Metode mapiranja su prikazane na Slici 2.



Slika 2*: a) mapiranje stabla odluke na RMLC sa tri RB modula, b) mapiranje SVM na RMLC sa tri RB modula, c) mapiranje veštačke neuronske mreže na RMLC sa tri RB modula*

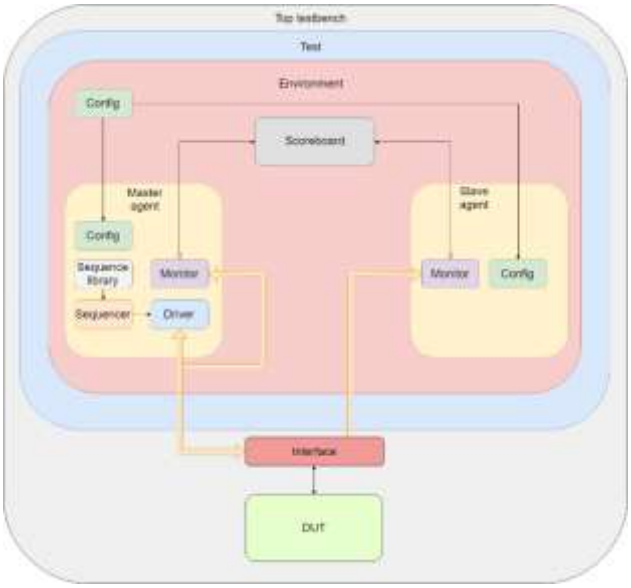
3. IMPLEMENTACIJA

Ovo poglavlje opisuje verifikaciono okruženje koje je implementirano. Opisan je način na koji je realizovan UVM testbench i generator stabala odluke.

3.1. UVM Testbench

Verifikacija je realizovana korišćenjem UVM metodologije i *SystemVerilog* programskog jezika.

Testbench predstavlja *SystemVerilog* modul u kome se instancira DUT, podešava se virtuelni interfejs putem konfiguracione baze podataka, i povezuje se sa instanciranim DUT-om. U ovom modulu se takođe instancira test klasa, i pokreće test pozivom taska run_test(), u okviru initial begin … end bloka, gde su definisani i reset i sinhronizacioni signal (Blok šema testbench-a se nalazi na Slici 3).



Slika 3. *Blok šema realizovanog testbench-a*

Test je *top level* UVM komponenta u okviru UVM t*estbench-a.* Test ima tri glavne funkcije: da instancira verifikaciono okruženje najvišeg nivoa (*eng. top level verification environment*), da konfiguriše verifikaciono okruženje (putem *factory override*-a ili konfiguracione baze podataka), i da primeni stimulus pozivom UVM sekvence kroz okruženje do DUT-a [6]. U ovom testbench-u postoji osnovni test test_base.sv koji instancira verifikaciono okruženje i konfiguraciju najvišeg nivoa. Ovaj osnovni test nasleđuju svi drugi testovi, koji potom drugačije konfigurišu okruženje, ili pokreću neke druge sekvence.

Verifikaciono okruženje se sastoji iz dva agenta, od kojih je jedan aktivan a drugi pasivan agent, konfiguracione komponente i skorborda.

Master agent se sastoji iz tri UVM komponente: drajver, sekvencer i monitor, pored kojih postoje i biblioteka sekvenci, koje se pokreću na sekvenceru, i konfiguracija. Drajver komponenta je aktivna komponenta u agentu, koja je zadužena za pokretanje aktivnosti na interfejsu ka DUT-u. Od sekvencera dobija sekvence transakcija sa podacima koje šalje ka interfejsu. Komunikacija između drajvera i DUT-a se odvija preko virtuelnog interfejsa, što je samo pokazivač na interfejs DUT-a. Sekvence koje sekvencer komponenta prosleđuje drajveru se nalaze u biblioteci sekvenci. Sekvenca koja će se proslediti drajveru se bira na nivou testa. Uloga monitor komponente je da nadgleda interfejs. Ukoliko se pojavi neka aktivnost na interfejsu, monitor komponenta će pokupiti podatke koristeći virtuelni interfejs, i sakupiti ih u jednu transakciju. Tu transakciju potom prosleđuje skorbordu.

Slejv agent (eng. slave) verifikacionog okruženja je definisan kao pasivan agent. Čine ga samo monitor komponenta i konfiguracija. Monitor komponenta nadgleda izlazne signale DUT-a, i ukoliko se validni

podaci pojave na interfejsu, skuplja ih u transakciju i šalje skorbordu.

Skorbord komponenta je osnovni element verifikacionog okruženja posvećen proveri rada DUT-a na funkcionalnom nivou [6]. Skorbord prima transakcije od strane monitor komponente agenata u verifikacionom okruženju preko UVM analysis_port portova. Ovi portovi su specijalizovani TLM portovi čiji se interfejs sastoji iz jedne write() funkcije.

U realizovanom verifikacionom okruženju, skorbord implementira funkcionalnost DUT-a na visokom nivou apstrakcije. Konkretno, implementirane su funkcije za svaku pojedinačnu podržanu konfiguraciju.

Transakcije koje monitor master agenta šalje skorbordu se koriste kao ulaz za implementirane funkcije. Rezultat funkcije se zatim poredi sa transakcijom koju je monitor slejv agenta poslao skorbordu. Ukoliko se rezultati ne poklapaju, šalje se obaveštenje o greški putem uvm_error metode.

3.2. Generator konfiguracije RMLC arhitekture

Kako bi se RMLC arhitektura verifikovala, potrebno je testirati što veći broj slučajeva u kojima bi mogla da se nađe. Ukoliko radi kao DT klasifikator, potrebno je dovesti na ulaze RMLC arhitekture sve moguće konfiguracije stabla odluke. Ukoliko radi kao SVM klasifikator, potrebno je dovesti dovoljno veliki i mali broj SVM vektora, sa manjim ili većim brojem ulaznih instanci. Ukoliko radi kao ANN mreža, potrebno je na ulaze arhitekture dovesti mreže sa manje ili više neurona u skrivenom sloju, sa različitim brojem ulaznih instanci.

Za potrebe verifikacije RMLC arhitekture kada radi kao stablo odluke, razvijen je model stabla odluke na visokom nivou apstrakcije u *SystemVerilog* programskom jeziku. Generisano stablo odluke se potom analizira, i generiše se odgovarajuća konfiguracija za RMLC. Stablo se dalje prosleđuje do scoreboard-a, gde se proverava rad RMLC arhitekture u ovom modu rada.

Za verifikovanje RMLC arhitekture kada radi kao veštačka neuronska mreža nije bilo potrebno razviti poseban model za generisanje konfiguracije RMLC arhitekture. Bilo je neophodno razviti model u okviru scoreboarda, koji će predviđati izlazni rezultat RMLC arhitekture. U tu svrhu, razvijen je jednostavan model koji računa rezultat rada neuroske mreže na nivou sloja neurona. Rezultat ulaznog sloja se prosleđuje do funkcije koja računa rezultat rada neurona iz skrivenog sloja. Rezultat skrivenog sloja se prosleđuje izlaznom sloju, i taj rezultat se poredi sa izlazom DUT-a.

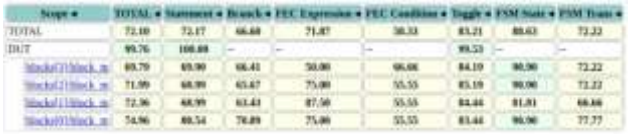
Kada arhitektura radi kao SVM klasifikator nije bilo potrebe za posebnim modelom za generisanje konfiguracije, niti za proračun rezultata. Rad SVM klasifikatora je jednostavno implementiran kao metoda u okviru scoreboard komponente.

4. REZULTATI VERIFIKACIJE

Za potrebe verifikacije razvijen je detaljan verifikacioni plan i test plan. Implementirano je ukupno 10 testova. Testovi se mogu podeliti u tri grupe: testovi gde je DUT konfigurisan kao stablo odluke, testovi gde je DUT konfigurisan kao SVM klasifikator, i testovi gde je DUT konfigurisan kao neuronska mreža sa radijalnim kernelom. Svaka od ove tri grupe ima po jedan ili više direktnih testova koji gađaju specifične slučajeve, i po jedan potpuno nasumičan test.

Kao merilo uspešnosti verifikacije koristili smo metode funkcionalne pokrivenosti i pokrivenosti koda.

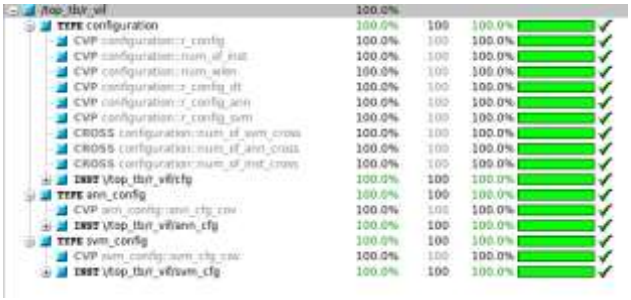
Pokrivenost se definiše kao procenat verifikacionih ciljeva koji su ispunjeni, i koristi se kao mera uspešnosti verifikacije [7]. U širem smislu postoje dva tipa pokrivenosti: pokrivenost koda i funkcionalna pokrivenost. Pokrivenost koda je mera koja označava do kog stepena je kod digitalnog dizajna testiran u datom verifikacionom okruženju. Izveštaj o pokrivenosti koda se automatski generiše od strane alata koji se koristi [7]. Rezultati pokrivenosti koda po instancama dizajna su prikazani na Slici 4.



Slika 4*: Rezultat pokrivenosti koda RMLC arhitekture po instancama dizajna*

Kao što se može zaključiti iz Slike 4, pokrivenost koda nije 100%. Ovakav rezultat je očekivan, s obzirom da su testirane tri konfiguracije, a postoje još tri konfiguracije koje nisu testirane. Samim tim, pokrivenost koda je manja, s obzirom da postoje stanja u koja RMLC neće ulaziti.

Funkcionalna pokrivenost predstavlja metriku koju definiše korisnik, i koja pruža informacije o tome u kojoj meri je funkcionalnost dizajna predviđena specifikacijom upotrebljena prilikom testiranja [7]. Može pružiti informacije o tome da li su se neki interesantni scenariji, krajnji slučajevi ili neki drugi uslovi dizajna desili, i da li su bili provereni od strane verifikacionog okruženja. Na Slici 5 su prikazani rezultati funkcionalne pokrivenosti, koja je 100%.



Slika 5*: Rezultat funkcionalne pokrivenosti*

5. ZAKLJUČAK

U ovom radu je predstavljen proces funkcionalne verifikacije rekonfigurabilne arhitekture za hardversku akceleraciju prediktivnih modela mašinskog učenja. Postupak realizacije verifikacije je započet pravljenjem verifikacionog plana i test plana, zatim je izgrađeno verifikaciono okruženje korišćenjem univerzalne verifikacione metodologije i *SystemVerilog* programskog jezika, i implementirani su testovi koji proveravaju funkcionalnost dizajna. Uspešnost verifikacije je izmerena pokrivenošću koda i funkcionalnom pokrivenošću.

6. LITERATURA

[1] V. Vranjković, "Rekonfigurabilne arhitekture za hardversku akceleraciju prediktivnih modela mašinskog učenja", *Autorski reprint*, 139., 2015.

[2] L. Rokach, O. Maimon, "Series in Machine Perception and Artificial Intelligence – Vol. 69, Data Mining with Decision Trees Theory and Applications", *Worlds Scientific Publishing Co. Pte. Ltd*, 2008.

[3] V. Sze, Y. Chen, J. Emer, A. Suleiman, Z. Zhang, "Hardware for Machine Learning: Challenges and Opportunities*", Massachusetts Institute of Technology Cambridge, MA 02139*, Oktobar 2017.

[4] G. Lacey, G. Taylor, S. Areibi, "Deep Learning on FPGAs: Past, Present, and Future", *University of Guelph 50 Stone Rd E Guelph, Ontario*, Februar 2016.

[5] B. Wile, John C. Goss, W. Roesner, "Comprehensive Functional Verification: The Complete Industry Cycle", *Library of Congress Cataloging-in-Publications Data, ISBN: 0-12-78183-7*, 2005.

[6] "Universal Verification Methodology (UVM) 1.2 User's Guide", *Accellera Systems Initiative (Accellera). Accellera Systems Initiative, 8698 Elk Grove Bldv Suite 1, #114, Elk Grove, CA 95624, USA*, 2011 – 2015.

[7] Mozhikunnath, R., Garg, R. "Cracking Digital VLSI Verification Interview Interview Success", 2016.

Kratka biografija:

Jasna Popović je rođena u Sremskoj Mitrovici 1993. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Embeded sistemi i algoritmi odbranila je 2016.god.

Rastislav Struharik je vanredni profesor na Fakultetu tehničkih nauka. Doktorsku disertaciju pod naslovom "Digitalna elektronska kola za realizaciju stabala odluka" odbranio je dana 18. decembra 2009. godine.