Зборник радова Факултета техничких наука, Нови Сад

UDK: 4.3
DOI: https://doi.org/10.24867/33BE14Maksic

РЈЕШАВАЊЕ ПРОБЛЕМА АУТОМАТИЗАЦИЈЕ ПРЕГЛЕДАЊА VHDL ЗАДАТАКА КРОЗ ИНТЕГРАЦИЈУ SYSTEM VERILOG И PYTHON АЛАТА

SOLVING THE PROBLEM OF AUTOMATING VHDL ASSIGNMENT REVIEW THROUGH THE INTEGRATION OF SYSTEM VERILOG AND PYTHON TOOLS

Миленко Максић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Рад се бави развојем система за аутоматизовано прегледање студентских задатака написаних у VHDL-у, кроз интеграцију SystemVerilog алата за верификацију и Python скрипти за анализу резултата. Циљ система је убрзавање и уједначавање процеса оцјењивања, посебно у условима великог броја студената. SystemVerilog се користи за дефинисање тврдњи и тест сценарија, док Python омогућава аутоматско покретање симулација, парсирање лог фајлова и генерисање извјештаја са бодовима. Систем је примјењен на архиву од 223 студентска рада, а добијени резултати су упоређени са ручним прегледањем. Анализа показује да аутоматизовано оцјењивање може значајно смањити вријеме прегледања и повећати објективност, али и да постоје изазови у праведном бодовању дјелимично тачних рјешења, што је посебно важно у образовном контексту.*

Кључне ријечи: *аутоматизација прегледања, VHDL, SystemVerilog, Python, тврдње, симулација, студентски задаци, дигитални дизајн, верификација хардвера, образовање*

Abstract – *This paper presents a system for automated evaluation of student assignments written in VHDL, integrating SystemVerilog verification tools with Python scripts for result analysis. The goal is to accelerate and standardize the grading process, especially in large academic groups. SystemVerilog is used to define assertion and test scenarios, while Python automates simulation execution, log parsing, and score report generation. The system was applied to a dataset of 223 student submissions, and the results were compared with manual grading. The analysis shows that automated evaluation can significantly reduce grading time and improve objectivity, but also highlights challenges in fairly assessing partially correct solutions – an important consideration in educational environments.*

Keywords: *automated evaluation, VHDL, SystemVerilog, Python, assertions, simulation, student assignments, digital design, hardware verification, education*

НАПОМЕНА:
Овај рад је проистекао из мастер рада чији ментор је био др Небојша Пјевалица, редовни професор.

1. УВОД

Брзи развој интегрисаних кола и дигиталних система значајно је утицао на хардверско инжењерство. Савремени дизајн дигиталних система ослања се на језике за опис хардвера (HDL), као што су VHDL и Verilog, који омогућавају прецизно пројектовање, симулацију и верификацију сложених компоненти. Сложеност система захтијева темељну верификацију, која често траје дуже од самог дизајна. Због тога се користе напредни алати попут SystemVerilog-a, који комбинује објектно оријентисано програмирање и алате за тестирање. Ови алати побољшавају квалитет производа, убрзавају развој и смањују ризик од грешака, што је кључно за индустријску примјену и тржишну конкурентност.

2. ТЕОРИЈСКЕ ОСНОВЕ

Језици за хардверски опис (HDL), као што су VHDL и SystemVerilog, представљају темељ савременог дизајна дигиталних система. Они омогућавају описивање структуре, понашања и временских аспеката хардвера, уз подршку за симулацију и синтезу. VHDL се истиче својом структуром заснованом на ентитетима и архитектурама, што омогућава јасно раздвајање интерфејса и функционалности. Његова примјена обухвата симулацију, прототиповање и документацију, чиме се убрзава развој и смањује ризик од грешака. SystemVerilog, као надоградња Verilog-a, уводи напредне могућности за верификацију, укључујући тврдње (assertions), објектно оријентисано програмирање и подршку за сложене тестне сценарије. Његова интеграција дизајна и тестирања омогућава рано откривање грешака и већу поузданост система. Верификација је кључна фаза у развоју дигиталних система. Поред симулације, све више се користе аутоматизоване методологије и алати као што је SVUnit, који омогућава модуларно тестирање у SystemVerilog-у. SVUnit подржава аутоматизацију, генерисање извјештаја и интеграцију са симулационим окружењима, чиме доприноси ефикаснијем и поузданијем развоју.

3. МОТИВАЦИЈА ПРОБЛЕМА СА КОНЦЕПТОМ РЈЕШЕЊА

Обука студената у домену техничких дисциплина подразумјева објективну провјеру знања кроз задатке у којима се сагледава оспособљеност у самосталном раду на доменским проблемима у ограниченом

времену. Курс „Логичко пројектовање рачунарских система 1“, уводи студенте у основе пројектовања дигиталних система коришћењем VHDL језика, слушају га студенти друге године основних академских студија на три студијска програма. Битна карактеристика курса је релативно велики број студената, што последично доноси веома масовне провјере знања. Ручно оцјењивање великог броја студентских радова представља значајан изазов због сложености задатака и ризика од људских грешака. Као рјешење, предложен је систем за аутоматизовано прегледање, који комбинује Python скрипте за обраду података и SystemVerilog алате за симулацију и верификацију.

Асистенти креирају прилагођене тестне сценарије и тврдње (SVA) како би се осигурала тачна и објективна процјена. Овај приступ омогућава брже, поузданије и конзистентније оцјењивање, уз бољу повратну информацију за студенте, што позитивно утиче на њихово учење и развој.

4. ПРОГРАМСКО РЈЕШЕЊЕ

Основу програмског рјешења чини раније имплементирана инфраструктура за чување студентских задатака, покретање симулација и техничку валидацију, уз коришћење алата Siemens Questa и Quartus. У овом раду додати су модули за обраду добијених извјештаја, систематско бодовање и аутоматизовано формирање резултата, што унапређује објективност и ефикасност оцјењивања.

4.1. Студентска архива

Архива садржи студентске директоријуме са рјешењима постављеног задатка и пратећим лог фајловима. Ови логови настају као резултат аутоматске валидације и детекције грешака као што су комбинационе петље, лечеви и непотпуне листе осјетљивости.

4.2. Аутоматска провјера и валидација студентских рјешења

Скрипта sva.py покреће симулацију у Siemens Questa алату, тестирајући DUT и тестбенч. Резултати се биљеже у лог фајлове run_dut.log и run_tb.log, што омогућава детаљну и објективну процјену сваке компоненте.

4.3. Идентификација комбинационих петљи и лечева

Скрипта qaas.py анализира дизајн у Quartus окружењу, генеришући извјештаје о грешкама као што су лечеви, петље и непотпуне листе. Ови подаци се чувају у top_map_rpt.log, што омогућава бодовање по унапријед дефинисаним критеријумима.

4.4. Креирање извјештаја са коначним резултатима

Процес бодовања се одвија у два корака: парсирање лог фајлова и формирање табеларног приказа бодова по студенту.

4.4.1. Парсирање лог фајлова

Скрипта провјерава постојање логова и анализира тврдње (assertions), број исправних DFF-ова и присуство грешака. Резултати се чувају у JSON фајловима:

- assertions_summary.json
- synth_vho_rst_summary.json
- synth_3problem_summary.json

4.4.2. Сумарно формирање бодова

Бодови се сабирају на основу тачних тврдњи и исправних DFF-ова, а затим се умањују за број детектованих грешака. Коначни резултат представља укупан број бодова по студенту.

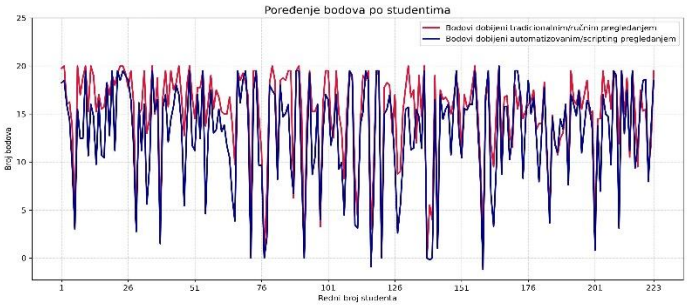
5. РЕЗУЛТАТИ

Програмско рјешење примјењено је на базу од 223 студентска рада, при чему су резултати аутоматске анализе упоређени са оцјенама добијеним путем ручног прегледа. Циљ поређења био је утврђивање тачности, досљедности и мјере разлике између ова два приступа. Машинско бодовање изведено је према два критеријума, базирана на присуству и исправности тврдњи у рјешењима студената.

5.1. Резултати добијени критеријумом 1

Критеријум 1 додјељује бод ако се појединачна тврдња у рјешењу бар једном појављује тачно, без обзира на евентуалне нетачне појаве у остатку кода. На примјер, ако студент у свом дизајну има тврдњу попут ASSERT_sCNT1, која провјерава инкрементацију бројача sCNT на позитивну ивицу сигнала iCLK, бод се додјељује ако је та тврдња најмање једном била тачно симулирана.

- Машинско бодовање је стабилније и досљедније; ручно оцјењивање показује варијације због субјективних оцјена.
- Просјечно оштећење: 2.11 бодова по студенту
- Највеће оштећење: −10.05 бодова
- Највећа добит: +4.00 бода
  Потпуно поклапање резултата код 13 студената

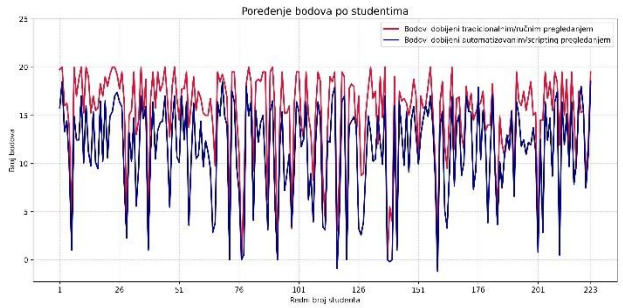


Слика 1 Анализа бодовања – критеријум 1

5.2. Резултати добијени критеријумом 2

Критеријум 2 додјељује бод само ако се тврдња ни једном не појављује нетачно – чак и ако је једном била тачна, бод се укида ако је током других симулација тврдња погрешно активирана. На примјер, за ASSERT_sSIFRA2, која провјерава правилно додавање цифре у шифру приликом уноса, студент губи бод ако се у било ком тренутку током симулације та тврдња испостави као нетачна.

- Строжија оцјена доводи до већег губитка бодова, али с већом прецизношћу у техничкој процјени исправности.
- Просјечно оштећење: 3.70 бодова по студенту
- Највеће оштећење: −13.75 бодова
- Највећа добит: +2.95 бодова
  Потпуно поклапање резултата код 2 студента



Слика Анализа бодовања – критеријум 2

6. ЗАКЉУЧАК

У хардверској индустрији, верификација система се ослања на методе као што су coverage, functional coverage и тврдње (assertions), које омогућавају бинарну процјену исправности – систем је или исправан или није. Овај приступ је ефикасан у техничком контексту, али када се примјењује у образовању, јављају се специфични изазови. Студентска рјешења често нису потпуно исправна, али ни потпуно погрешна, што захтијева флексибилнији и нијансиранији приступ бодовању.

Coverage у индустрији значи да је систем бар једном радио исправно. Functional coverage провјерава да ли су све функционалности тестиране, док тврдње служе за формалну потврду да је систем реаговао у складу са очекивањима у одређеним условима. У академском контексту, овакви критеријуми могу бити недовољни јер не одражавају степен разумјевања студента, нити сложеност његовог рјешења.

Кључна дилема у образовању је: да ли студент заслужује пуне бодове ако је тврдња једном тачна, али у другим случајевима није? У индустрији, то би се сматрало успјехом, али у настави је потребно увести парцијално бодовање које узима у обзир и дјелимичну исправност. На примјер, ако рјешење функционише у већини тест случајева, али садржи мању грешку у једном сценарију, треба размотрити да ли заслужује дио бодова.

Да би се овај проблем ријешио, потребно је развити систем класификације својстава студентских рјешења и дефинисати критеријуме успјеха. То укључује:

- идентификацију критичних својстава која морају бити задовољена,
- одређивање степена исправности (нпр. 90% тачности),
- и формулисање флексибилног модела оцјењивања који уважава различите нивое тачности.

У даљем истраживању, потребно је развити објективан модел бодовања који ће балансирати између индустријских стандарда и академских потреба. Такав модел треба да буде праведан, транспарентан и педагошки оправдан, омогућавајући студентима да добију јасну повратну информацију и подстицај за даље усавршавање.

7. ЛИТЕРАТУРА

[1] „C. Spear, SystemVerilog for Verification: A Guide to Learning the Testbench Language Features,“ 3rd ed. Springer, 2012.

[2] „Н. Пјевалица: Верификација дигиталних интегрисаних кола, System Verilog са основама UVM-а“, Факултет Техничких наука у Новом Саду, 2022, ISBN 9788660224073

[3] „B. Cohen, SystemVerilog Assertions Handbook,“ 3rd ed. VhdlCohen Publishing, 2010.

[4] „D. L. Perry, VHDL: Programming by Example,“ 4th ed. McGraw-Hill, 2002.

[5] „IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language, IEEE Std 1800™-2017,“ IEEE Computer Society, 2017.

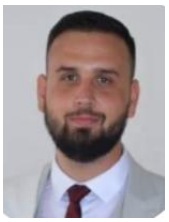
[6] „Intel Corporation, Intel Quartus Prime Pro Edition User Guide: Design Compilation, 2023.“ [На мрежи]. Available: https://www.intel.com

[7] „Siemens Digital Industries Software, QuestaSim User’s Manual, 2023.“ [На мрежи]. Available: https://eda.sw.siemens.com

[8] „AgileSoC Inc., SVUnit User Guide, 2023.“ [На мрежи]. Available: https://github.com/svunit/svunit

[9] „Python Software Foundation, Python 3 Documentation, 2023.“ [На мрежи]. Available: https://docs.python.org/3/

Кратка биографија



Миленко Максић је рођен 13. септембра 1997. године у Бијељини. У школској 2016/17 уписује студијски програм Рачунарство и аутоматика на Факултету техничких наука у Новом Саду. Након завршених основних студија, 2022. године, уписује мастер студије на истом студијском програму, смјер Софтвер за потрошачку електронику.