**Facial recognition**

*Dokumentacija za predmet: Funkcionalna verifikacija hardvera*

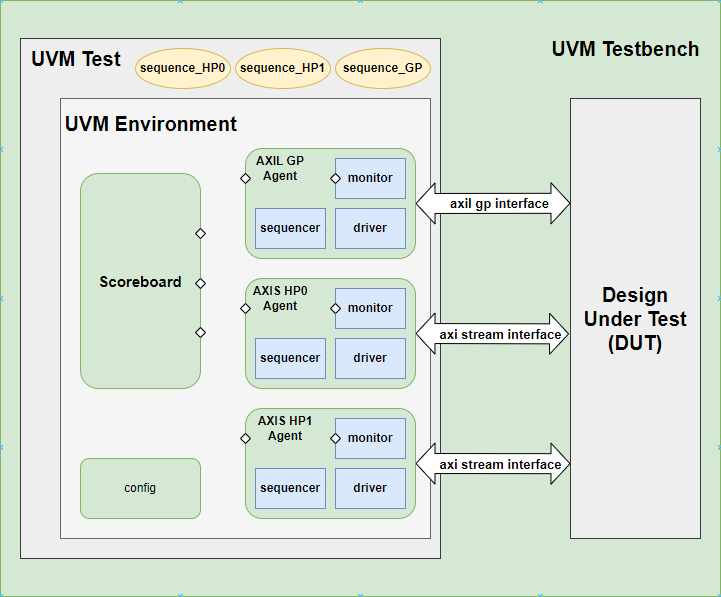
1. Opis algoritma

Ideja projekta je prepoznavanje ljudskog lica na fotografiji. Fotografija se prosleđuje programu, a on iscrtava okvir oko detektovanog lica. Centralni dio algoritma predstavlja filtriranje slike (po x-osi i y-osi) koje podrazumijeva množenje manjih dijelova slike sa filterima. Naime, dimenzije filtera su 3x3, te se svaki komad slike dimenzija 3x3 množi sa njime. Šablon se ponavlja sve dok se ne prođe kroz čitavu sliku pomjerajući se prvo po x-osi za 1 piksel, a nakon obrađenog reda za 1 piksel po y-osi. Ostatak algoritma sprovodi niz transformacija nad slikom čiji je krajnji rezultat skup HOG vektora. Zatim je potrebno pronaći skup HOG vektora na dijelu originalne slike koji je sličan skupu HOG vektora šablona slike. Algoritam ekstrahuje HOG šablona slike kojoj se mijenjaju dimenzije počevši od minimalne dimenzije pa do trećine minimalne dimenzije originalne slike da bi se omogućilo prepoznavanje lica bez obzira na njegovu veličinu na originalnoj slici. Sa različitim veličinama HOG-a šablona slike algoritam prolazi kroz originalnu sliku i pamti one koje su iznad određenog praga (threshold).

*Slika 1.1 Originalna slika Slika 1.2 Šablon slika Slika 1.3 Izlazna slika*

1. Verifikaciono okruženje



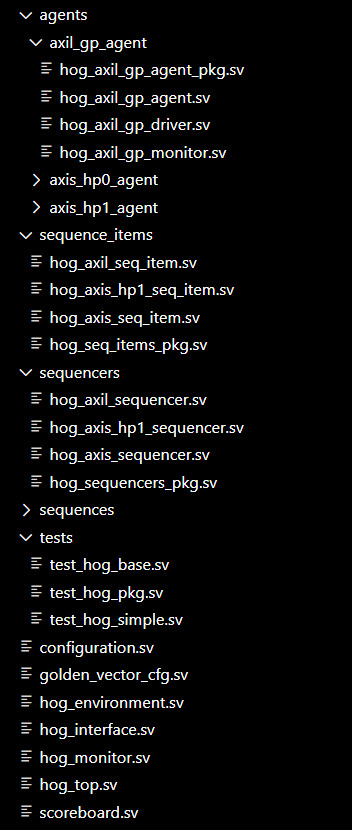
*Slika 2.1 Verifikaciono okruženje*

Verifikaciono okruženje sadrži sljedeće komponente:

* Aktivan agent
* Scoreboard
* Config (konfiguracioni paket)
* Sequence
* Test
* Enviroment
* Top

Uz ove komponente tu se nalazi i DUT (design under test) koji se verifikuje. Da bi se sistem pokrenuo koriste se 2 različita AXI protokola. AXI Lite za parametre slike i AXI Stream koji popunjava BRAM blokove pikselima koji treba da budu obrađeni.

Hijerarhija fajlova prikazana je na *Slici 2.2.*

**

*Slika 2.2 Hijerarhija okruženja*

1. Komponente verifikacionog okruženja
   1. Top

U top fajlu su ubačeni uv makroi i naš test aket. Interfejs verifikacionog okruženja je povezan sa interfejsom dizajna koji se verifikuje.

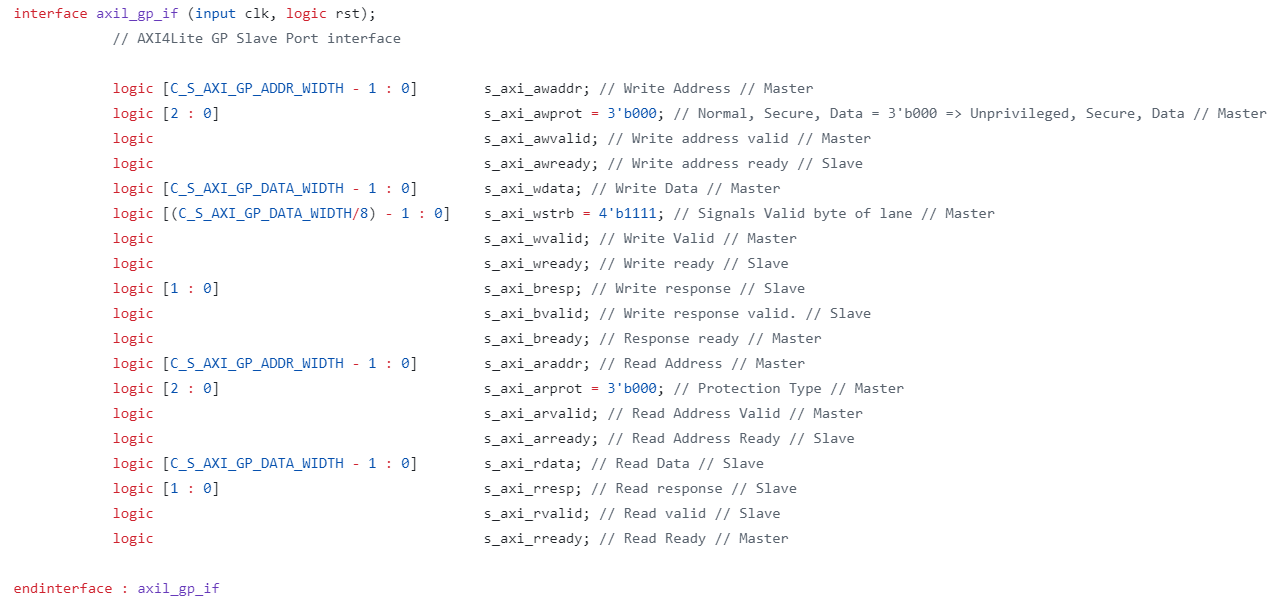
U intial petlji proslijeđen je interfejs enviroment-a preko uvm\_config\_db, generisan taktni signal i reset.

* 1. Enviroment

Dodjela virtuelnog interfejsa agentima, konfiguraciji i scoreboard-u kao i kreiranje ovih komponenata. Vrši se povezivanje monitora i scoreboarda.

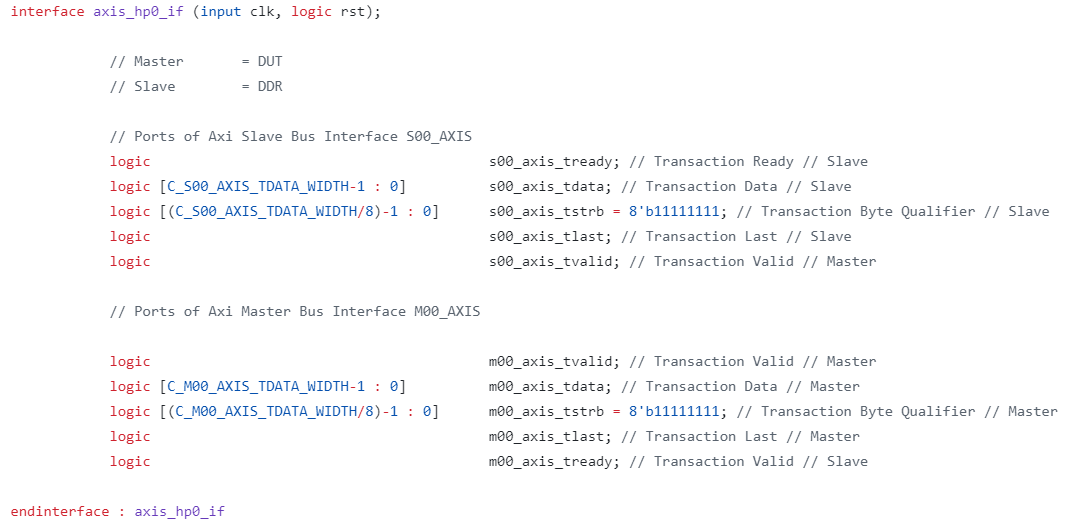
* 1. Interface

Unutar interface fajla dati su AXI Lite parametri za širinu podatka i adresu (redom 32 i 5) i AXI Stream parametri za širinu podataka 64 bita. Na *Slici 3.3.1* prikazani su AXI Lite signali.

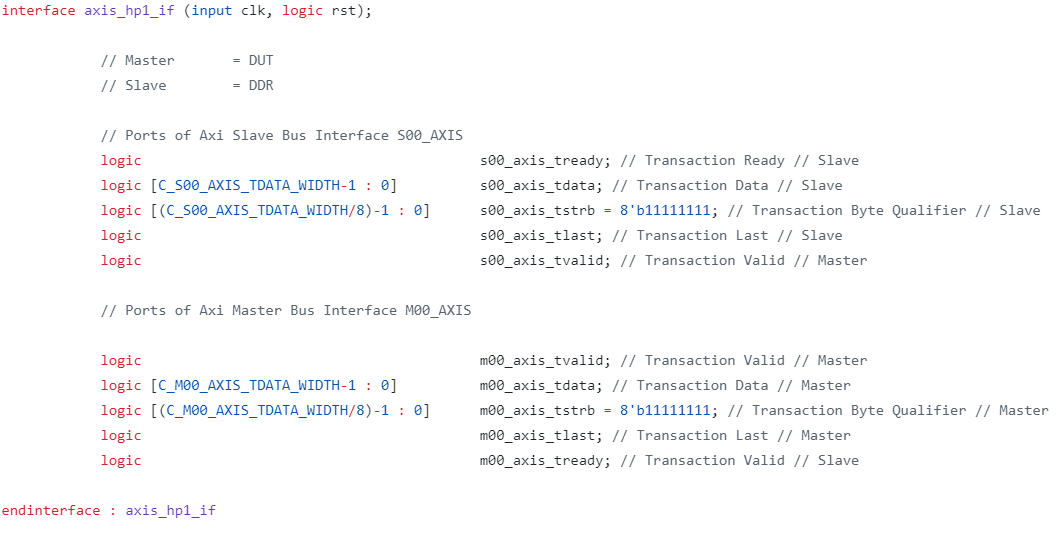


*Slika 3.3.1 AXI Lite signali*

Na *Slici 3.3.2* i *Slici 3.3.3* prikazani su AXI Stream signali, redom za HP0 i HP1 port.



*Slika 3.3.2 AXI Stream signali za HP0*

**

*Slika 3.3.3 AXI Stream signali za HP1*

* 1. Test

**Base test**

**Simple test**

* 1. Sequence

U ovom poglavlju će biti rečeno nešto o base sequence-u i simple sequence-u.

**Base sequence** služi kao osnovna struktura za druge sekvence koje će naslijediti njegove osobine. Base sequence će biti proširena od strane druge sekvence u kojoj će se definisati konkretni testovi u metodi body(). Te sekvence neće morati da podižu i spuštaju objections, već će se kontrola vršiti iz base sequence-a.

**Simple sequence** zapravo nasljeđuje **base sequence.** Pomoću njega se generiše sekvenca koji će biti proslijeđena za pokretanje i rad sistema. U modulu za generisanje sekvence AXI Lite protokola nalaze se potrebne adrese sahe registra (start, height, width, width\_2, width\_4, rows\_num, cycle\_num\_out, cycle\_num\_in, effective\_row\_limit, bram\_height) kao i parametri za sliku. Kada je sve spremno podiže se *start* signal i zatim se AXI Stream sekvence popunjava BRAM blokove ulaznim podacima. Zatim se čeka na *ready*, a kada se on podigne na 1 kraj obrade.

* 1. Agent

Unutar agenta uključujemo driver za kontrolu transakcija, monitor za nadgledanje izlaza sistema, sekvencer za povezivanje sequence-a i driver-a i sequence item. U ovom modulu se povezuju sequence item i driver preko sekvencera.

* 1. Sequencer

Sekvencer služi kao komponenta koja kontroliše tok stimulusa ka driver-u. Njegova osnovna uloga je da koordiniše generisanje i slanje transakcija iz sekvenci do drajvera.

* 1. Sequence item

Unutar sequence item-a nalaze se neophodne informacije za driver i seqence. On sadrži sve signale interfejsa koji će putem driver-a biti proslijeđeni na DUT.

* 1. Driver

Driver AXI Lite-a i AXI Stream-a sadrži kod za upis i čitanje transakcija. Preko AXI Lite driver-a vrši se upis svih parametara slike i čitanje ready signala, dok se preko AXI Stream-a upisuju ulazni podaci u DUT i čitaju obrađeni.

* 1. Monitor

Monitor je zadužen da prati i nadgleda signale na izlazu sistema. On će u kombinaciji sa scoreboard-om koji koristi metodu zlatnih vektora utvrditi da li DUT radi ispravno. U monitoru koji je namijenjen za AXI Lite transakcije prate se adrese na koje se upisuje i sa kojih se čita kao i podaci koji se salju odnosno primaju. U monitoru koji je namijenjen AXI Stream transkacijama prate se podaci koje smo poslali ka DUT-u odnosno oni koje smo primili od DUT-a.

* 1. Scoreboard

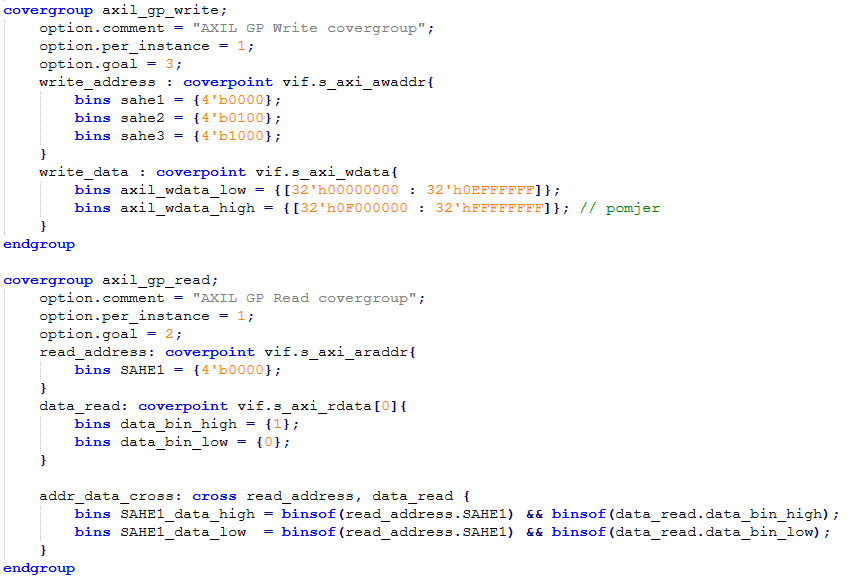
Za provjeru rezultata korišćena je metoda zlatnih vektora. Scoreboard poredi vrijednosti koje je monitor zabilježio sa vrijednostima koje se nalaze u fajlu golden vector-a. Fajl golden vector-a se dobija iz virtuelne platforme. Koriste se assert naredbe da bi se upredili očekivani rezultati sa stvarnim rezultatima, ukoliko dođe do nepoklapanja, biće prijavljena greška uz pomoć *uvm\_error* poruke.

* 1. Config

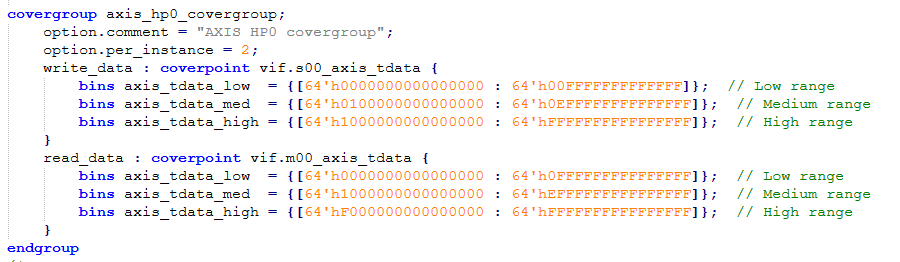
U konfiguracionom fajlu se omogućava učitavanje para ulaz-izlaz golden vector-a.

1. Coverage

Coverage se prikuplja u monitorima. Kod AXI Lite protokola prikupljamo informacije da li se desio upis u svaki registar i da li se desilo čitanje iz registra u kome se nalazi *ready* bit i koliki je opseg podataka pokriven prilikom čitanja i upisa u registre. Takođe, koe AXI Stream protokola provjerava se koliki je opseg podataka pokriven. Na *Slici 4.1* prikazan je covergroup za AXI Lite, dok je na *Slici 4.2* prikazan covergroup za AXI Stream. Kod AXI Lite transakcija zadat je tačan goal za prikipljanje jer znamo da treba da se dese 3 transakcije upisa i 2 transakcije čitanjanja.

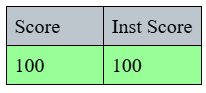


*Slika 4.1 Covergroup za AXI Lite*



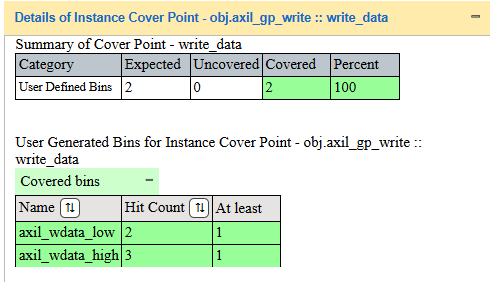
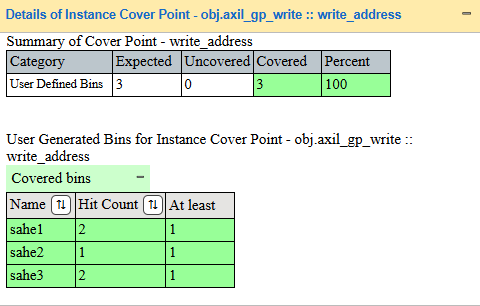
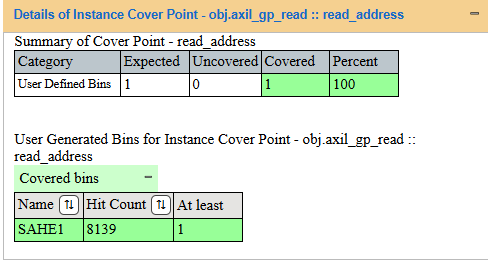
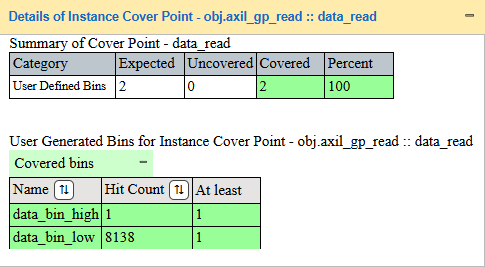
*Slika 4.2 Covergroup za AXI Stream*

Na *Slici 4.3* prikazan je ukupan coverage.



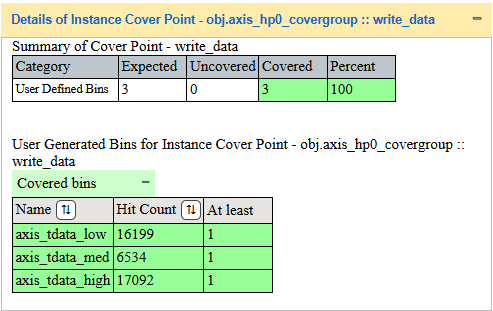
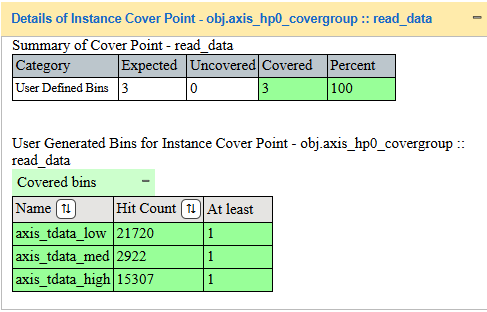
*Slika 4.3 Ukupan coverage*

Na *Slici 4.4* prikazani su rezultati pokrivenosti za AXI Lite protokol.



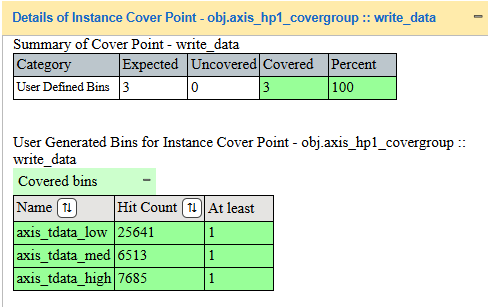
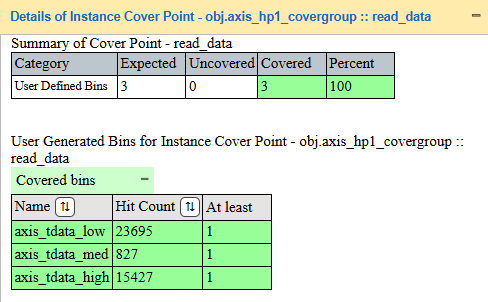
*Slika 4.4 coverage za AXI Lite*

Na *Slici 4.5* prikazan je coverage za AXI Stream protokol na HP0 portu.



*Slika 4.5 coverage za AXI Stream na HP0*

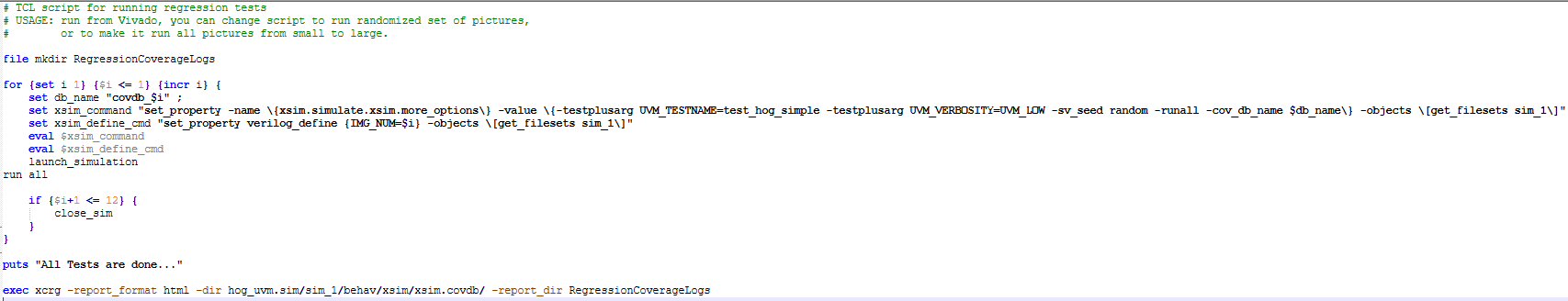
Na *Slici 4.6* prikazan je coverage za AXI Stream protokol na HP1 portu.



*Slika 4.6 coverage za AXI Stream na HP1*

1. Regresija

Regresija se svodi na puštanje više simulacija pri čemu redoslijed slika može da se randomizuje ili da se redom puštaju. Na *Slici 5.1* prikazan je kod za puštanje testova.



*Slika 5.1 Regresija*

1. Verifikacioni plan

Verifikacioni plan sadrži sljedeće korake:

* Provjera funkcionalnosti reset-a sistema. Ova funkcionalnost se provjerava na početku simulacije u trenutku kada se čitav sistem resetuje.
* Provjera funkcionalnosti AXI Lite interfejsa.
* Provjera AXI Lite transakcija upisa (covergroup za axi write)
* Provjera AXI Lite transakcija čitanja (covergroup za axi read)
* Provjera da li se desio upis u sve registre AXI Lite protokola
* Provjera prolaska kroz sva stanja
  + Provjera da li je sistem prošao kroz sva stanja u FSM-u se utvrđuje putem waveform-a.
* Provjera funkcionalnosti sistema koja se vrši u “scoreboard” komponenti.