

Överbrygga kvalitetsgapet

Verifiering är nödvändigt, men hur avgör man att en verifieringsmiljö är bra nog. George Bakewell, produktmarknadschef på SpringSoft, beskriver här hur mutationsbaserade testmetoder kan effektivisera arbetet.

Avancerade halvledarkonstruktioner verifieras i sofistikerade och mångskiftande verifieringsmiljöer, vars komplexitet är jämförbar med eller överträffar den hos själva konstruktionerna.

Framsteg har gjorts inom generering av stimuli och metoder för mätning av täckningen. Men trots detta berättar inte dagens verktyg för ingenjörerna ”hur bra” testbänken är på att föra effekterna av buggar vidare till observerbara punkter, eller på att upptäcka felaktiga beteenden som indikerar att det finns buggar.

Detta medför att beslut om vad man skall fokusera verifieringsarbetet mot, hur man metodiskt kan förbättra miljön, om denna är tillräckligt robust för att fånga in de mest potentiella buggarna samt slutligen när verifieringen är ”klar”, ofta baseras på ofullständiga data eller ren finger-

toppskänsla.

Denna artikel handlar om hur man kan använda mutationsbaserade testmetoder för att mäta och driva fram förbättringar inom alla kvalitetsaspekter på funktionell verifiering i simuleringsbaserade miljöer som en lösning av dessa problem.

BEFINTLIGA METODER

Funktionell verifiering förbrukar en väsentlig del av den tid och de resurser som tilldelas ett typiskt designprojekt. I takt med att chipen fortsätter att växa i storlek och komplexitet blir konstruktörerna i allt högre grad beroende av speciella verifieringsteam för att kunna garantera att systemen fullt ut följer specifikationerna.

Verifieringsingenjörerna har tillgång till en uppsättning specialanpassade verktyg och metoder för att kunna automatisera verifieringen och förbättra kvali-

teten. Trots detta är funktionella logikfel fortfarande en viktig orsak till projektfördröjningar och ”re-spins”.

En viktig orsak till detta är att två viktiga aspekter på verifieringsmiljöns kvalitet, nämligen möjligheten att låta följderna av en bugg föras vidare till en observerbar punkt liksom möjligheten att observera de felaktiga följderna och därmed upptäcka buggen, inte kan analyseras eller mätas.

Befintliga metoder som kodtäckning och funktionell täckning ignorerar i hög grad dessa aspekter. Det gör att funktionella fel kan komma undan i verifieringsprocessen, trots mycket goda täckningsresultat.

KODTÄCKNING

Kodtäckning är i sig ett enkelt mått på möjligheten för ett antal stimuli att aktivera logiken i konstruktionen. ”Aktivera” innebär

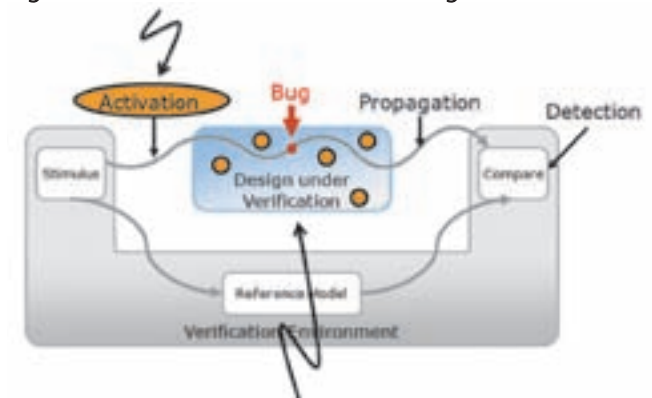
här bl a att exekvera varenda rad, växla vartenda signaltillstånd och gå igenom varenda signalväg och liknande aktiviteter.

Detta är visserligen ett nödvändigt villkor – man kan inte hitta en bugg om man inte ”berör” den kod som hör till buggen. Men det är hel säkert inte ett tillräckligt villkor för att man skall kunna upptäcka alla eller ens de

flesta problemen i en konstruktion.

Kodtäckningen säger ingenting om möjligheten för verifieringsmiljön att föra vidare effekten av en bugg som upptäckts, och inte heller att ifall den förts vidare upptäcka att den finns. Verifieringsingenjörerna måste därför acceptera att även om kodtäckningen ger intressanta data, så är den ett dåligt

Kodtäckning mäter aktiveringen, men säger ingenting om vidarebefordran och detektering



Funktionell täckning kontrollerar "viktiga" funktionspunkter, men är subjektiv och inkomplett

Fig 1. Kodtäckning och funktionell täckning ger bara begränsad information.

mått på den totala kvaliteten hos verifieringsmiljön.

FUNKTIONELL TÄCKNING

Funktionell täckning är generellt mer intressant, och i sig mer användbar. Enkelt uttryckt ger den en möjlighet att avgöra om man har täckt in alla viktiga funktionsområden. "Viktiga" definieras här på olika sätt, t ex som alla operationstillstånd, alla funktionella sekvenser eller något liknande.

Problemet är att funktionell täckning per definition är subjektiv och i sig icke komplett. De funktionsområden (funktionella "täckningspunkter") som skall kontrolleras definieras av ett antal ingenjörer, och de baseras vanligen på en designspecifikation. Detta är ett dokument som noggrant beskriver hur konstruktionen är tänkt att fungera. Men det ger inte någon ingående information om hur konstruktionen inte skall fungera.

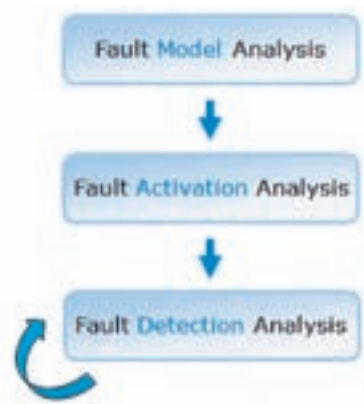
Om specifikationen täckte in alla tänkbara buggar som skulle kunna uppstå i en konstruktion, och beskrev hur dessa skulle visa sig funktionellt, då skulle det vara enkelt att översätta denna lista till en uppsättning funktionella täckningspunkter som ingående skall kontrolleras under verifieringen.

Men nu är det inte på detta vis. Så även om funktionell täckning ger användbara och nödvändiga data måste man ha någon metod som visar om man verifierar den funktionalitet som anges i specifikationen – som kodtäckning. Funktionell täckning är ett dåligt mått på den totala kvaliteten hos verifieringsmiljön.

MUTATIONSBASERADE METODER

Mutationsbaserade testmetoder kan vara den nyckel som ger oss möjlighet att komma förbi alla bristerna hos befintliga verktyg och överbygga "kvalitetsgapet" i funktionell verifiering.

Mutationsbaserad testning började användas under det tidiga 70-talet inom mjukvaruforskning. Till en början var syftet att



Lös och iterera problemen i takt med att de hittas

Fig 2. De tre stegen i en process med funktionell kvalificering.

vid mjukvarutestning få fram de bästa möjliga testuppsättningarna.

En "mutation" är en konstgjord modifiering av testprogrammet som införs av en feloperator och som ändrar beteendet hos det testade programmet. Testuppsättningen modifieras sedan för att kunna hitta denna beteendändring.

När testuppsättningen upptäckt alla införda mutationer ("dödat alla mutanter") sägs testuppsättningen vara "mutationsadekvat". Många teoretiska konstruktioner och hypoteser har definierats för att stödja giltigheten hos mutationsbaserad testning.

Inom området verifiering av digital logik är den grundläggande principen att injicera fel i en konstruktion – för att kontrollera kvaliteten hos vissa delar av verifieringsmiljön – välkänd av verifieringsingenjörerna. Metoden används ibland när man inte fullt ut litar på testbänken och det inte finns något annat sätt att få återkoppling.

I dessa fall av "för hand framställt", mutationsbaserad testning begränsas kontrollen till ett mycket specifikt område hos verifieringsmiljön som har intresse för verifieringsingenjören. Att utöka denna manuella ansats till mer än en liten mängd kod skulle vara opraktiskt.

Genom att automatisera mutationsbaserade metoder får man ett objektiva och omfattande sätt att utvärdera, mäta och förbättra kvaliteten hos funktionella verifieringsmiljöer för komplexa konstruktioner. Intelligent användning ger en mutationsbaserad ansats detaljerad information om verifieringsmiljöns möjligheter att aktivera, vidareföra och detektera buggar. Den hittar också viktiga svagheter och hål som passerat utan att upptäckas av klassiska, konventionella metoder.

Analysen av fel som inte förs vidare, eller som inte upptäcks av verifieringsmiljön, pekar på brister i stimuli, observerbarhet och de "checkers" som används för att detektera icke förväntat beteende och påvisa förekomsten av designbuggar.

FUNKTIONELL KVALIFICERING

Användandet av mutationsbaserad teknologi för att mäta kvaliteten

Analysera konstruktionen för att bestämma var felen skall injiceras

Analysera verifieringsmiljön genom att simulera testerna en gång

Mät möjligheten att detektera buggar genom att injicera fel och simulera

- Skriv instrumenterad RTL
- Identifiera icke detekterbara fel
- Identifiera icke aktiverade fel
- Korrelera fel och tester
- Identifiera icke detekterade fel
- Rapportera information om felen för åtgärdande av dem

hos och upptäcka svagheter i verifieringen av digitala logikkonstruktioner kallas "funktionell kvalificering". Processen infördes av Certess år 2006 med Certitude Functional Qualification System, som nu ingår i SpringSofts produktfamilj för Verification Enhancement. I praktiken omfattar metoden den trestegs process som visas i fig 2.

Det första steget i denna process är en fas med statistisk analys, som bestämmer var fel kan injiceras och skriver ut en ny version av den RTL-kod som möjliggör denna injicering. Därefter följer ett simuleringssteg som bestämmer den uppsättning fel som inte aktiveras av något test, och som för de fel som aktiveras korrelerar vart och ett av dem med det test som aktiverar felet.

Slutligen kommer en detekteringsprocess som injicerar felen ett i taget, kör de relevanta (korrelerade) testerna för att avgöra om ett visst fel detekterats, och som ifall ett fel inte upptäckts ger återkoppling till användaren för att hjälpa denne att åstadkomma en fix (lägga till en checker som saknas, utveckla ett testscenario som fattas, etc).

Komplexiteten hos dagens konstruktioner, liksom det antal test som behövs för att verifiera dem, kräver extra automatisering och intelligens. Annars kommer kombinationen av möjliga fel och tillhörande test att överbelasta processen och göra den opraktisk.

Det är troligt att vissa fel inte kan detekteras, antingen på grund av redundant logik eller "död kod" som inte har någon väg till någon utgång. Om man kan hitta och eliminera dessa fel under den statiska analysfasen kan man spara in avsevärd simuleringsstid under de efterföljande aktiverings- och detekteringsstegen.

På liknande sätt kan information om ett visst test som samlats in under aktiveringsfasen – erforderlig simuleringsstid, antal aktiverade fel o dyl – med stor fördel användas för att göra detekteringsfasen mer effektiv.

Metodologiska överväganden är också mycket viktiga när det gäller att göra teknologi för funktionell kvalificering praktiskt användbar för dagens halvledarkonstruktioner. Forskning har visat

att vissa klasser av fel har större sannolikhet att exponera betydande svagheter än andra. Så genom att tidigt hitta och kvalificera dessa fel kan man snabbt få återkoppling om det totala hälsotillståndet hos miljön.

Att lösa problem som relateras till dessa högprioriterade fel redan när de upptäcks ger omedelbara och avsevärda förbättringar av verifieringsmiljön. Det ger också motivation för att sätta in partiell funktionell kvalificering tidigt i verifieringsprocessen.

Inkrementell användning – dvs att köra funktionell kvalificering, lösa problem och sedan köra igen – ger också möjligheter att samla in kvalificeringsdata över tiden. Fel som hittas tidigt i processen behöver ju inte kvalificeras igen senare, så länge som den tillhörande RTL-koden inte ändras.

Forskning har också visat att om man löser "stora problem" tidigt, så löser man ofta samtidigt många "mindre" eller mer subtila problem som härrör från samma saknade checkers eller testscenarier. Så en inkrementell ansats gör den totala processen mer effektiv.

Slutligen måste sägas att komplett och flexibel tillgång till alla analysresultat är en nödvändighet. Intuitiva rapporter behövs t ex för att visa var fel har injicerats i HDL-koden, liksom för att ge felstatus och information om varje givet fel.

Det är mycket viktigt att system för funktionell kvalificering intimt integreras med befintliga kommersiella simulatorer och avancerade avbuggningssystem. De måste också vara fullt kompatibla med dagens verifieringsmetoder, t ex framtvängad slumpvis generering av stimuli (constrained random stimulus generation) och påståendebaserad verifiering (assertion-based verification).

KOMPLEXT

Verifiering av dagens avancerade halvledarkonstruktioner är en ytterst komplex process, och de tillhörande verifieringsmiljöerna är ofta än mer komplexa än själva konstruktionerna. Det finns otaliga tillfällen för problem som låter RTL-buggar slinka genom processen.

De mest uppenbara problemen är felande eller felaktiga

checkers som inte klarar av att detektera ett icke korrekt beteende, liksom otillräckliga testscenarier som går igenom hela konstruktionen men inte för vidare effekterna av dolda buggar.

Misstag i de "wrapper scripts" som startar och hanterar verifieringsprocessen kan vara än mer bekymmersamma, eftersom de kan låta stora mängder tester markeras som "passed" när de i själva verket skulle markeras som "failed". Detta riskerar att maskera allvarliga RTL-buggar, eftersom ingenjörerna inte är benägna att undersöka och avbugga tester som markerats som "passed".

Dagens metoder – som kodtäckning och funktionell täckning – ger visserligen intressanta och ibland nödvändiga data om verifieringens status och förlopp. Men de kan inte på något adekvat sätt hantera dessa problem. Deras inkompleta och subjektiva natur lämnar alltför stort utrymme för fel och ouppmärksamhet.

Mutationsbaserade metoder ger å andra sidan en både vittomfattande och objektiv värdering av verifieringsmiljöns kvalitet. Den funktionella kvalificeringsprocess som finns inbyggd i dessa metoder mäter miljöns förmåga att aktivera logik som är relaterad till potentiella buggar, vidarebefordra effekten av dessa buggar till en observerbar utgång samt detektera det felaktiga beteendet och därmed förekomsten av buggarna.

På så sätt hittar den allvarliga hål i verifieringsmiljön som kan leda till att verkliga RTL-buggar kan klara sig genom processen. Den ger också vägledning om hur man kan fylla igen dessa hål. Med automation och intelligent operation i kombination med rätt metodik blir användandet av funktionell kvalificering både praktiskt och tillgängligt under hela verifieringsprocessen. ■ ■ ■

George Bakewell,
produktmarknadschef,
SpringSoft

Om författaren

George Bakewell, produktmarknadschef på SpringSoft, har produkt- och teknisk ansvar för sofistikerade system för förbättrad verifiering. Han samarbetar nära med produktutvecklingsteam angående utveckling av nya krav och specifikationer, har varit talare på tekniska konferenser, deltagit i organisationer för industristandardisering samt lett djupgående kurser och applikationsworkshops. Georg har över 20 års erfarenhet från EDA-industrin och har en BA-examen i Electronic Engineering och Computer Science från University of Colorado.

Trådlös kommunikation

Ny bok av Krister Andreasson (359 sid) beskriver principer och kretskopplingar för trådlös kommunikation. Uppdaterad med de senaste systemen! Se exempel på www.mikrovagspridning.se.

Bok 950:- exkl moms. Beställ från:

Richardson Electronics
stefan@rell.com
tel 08 564 70590