Utveckla fordonselektronik i virtuell hårdvara

EXPERTARTIKEL

Observerbar, styrbar, deterministisk, skriptbar – det är en utvecklingsmiljö med virtuella prototyper

Av Marc Serughetti, Synopsys

Marc Serughetti är affärsutvecklingschef på Synopsys med ansvar för bland annat strategi, utveckling och driftsättning av prototypningsteknik i vertikala tillämpningsmarknader. Han har drygt 20 års erfarenhet av programutvecklingsverktyg, mjukvaru-IP (operativsystem, mellanvara och tillämpningsprogram) och tillämpningar inom konsument, fordon, industri och nätverk. Innan Synopsys ledde han produktmarknadsförings- och affärsutvecklingsgrupper på CoWare, Wind River och Integrated Systems.

Dagens fordon kan innehålla tiotals miljoner rader källkod. Kodvolymen växer explosionsartat och ställs dessutom inför nya utmaningar, som övergången till multikärnor, AUTOSAR-appar, integration och test av mjukvara och av funktionssäkerhet – och detta är endast några av de utmaningar som är direkt kopplade till den växande kodvolymen.

Utvecklare har under många år kunnat luta sig mot väletablerade konstruktionsmetoder på den funktionella nivån, som MIL (model-in-the-loop) och SIL (software-in-the-loop). Det som dessa saknar är att de inte tar hänsyn till den underliggande hårdvaran.

I OCH MED DEN VÄXANDE komplexiteten kan utvecklarna inte längre vänta på att ECU-hårdvaran ska finnas fysiskt tillgänglig innan de påbörjar sitt arbete. Det krävs ny, bättre teknik som tillåter tidigt utveckling, integrering och test och dessutom skapar ökad produktivitet och håller associerade kostnader nere.

Den här texten handlar om hur man med hjälp av simuleringsteknik kan realisera en VHIL-miljö (virtual-hardware-in-the-loop) långt innan den fysiska hårdvaran existerar.

I en VHIL kombineras simulering av digital hårdvara med hjälp av virtuella utvecklingsmiljöer för styrkretsar, med simulering av annat, som analog hårdvara och mekaniska system.

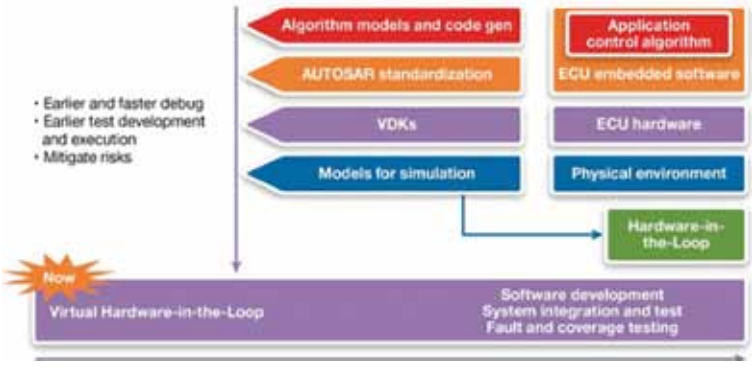
VHIL-miljöer är en bas för tidig mjukvaruutveckling, systemintegrering, prestandavalidering, felinjektion, kodtäckningstest enligt ISO 26262, och regressionstest. VHIL gör sammantaget att utveckling kan påbörjas tidigare, att testtäckning blir bättre och att defekter identifieras tidigare. Detta ger i slutänden robustare produkter av högre kvalitet till lägre kostnader och på kortare utvecklingstid.

DEN ÖKANDE KODVOLYMEN i fordonen har negativa effekter på utvecklingskostnader, men också på varumärken. Om du vill kan du snabbt googla upp den senaste historien om återkallade fordon och aktuella branschdiskussioner kring problemet.

IEEE Spectrum hade exempelvis följande rubrik ”Ytterligare 936000 fordon återkallas på grund av fel i mjukvara och elsystem”. Artikeln berättar att mjukvarufelet gjorde att ett kullager kunde skadas om föraren växlade för snabbt mellan lägena neutral, drive och reverse när hen exempelvis fastnat i snö. Det skadade kullagret kunde betyda att motorn slutade driva eller ställa till problem med att lägga i parkeringsbromsen. En programuppdatering gjorde så att växlingen skedde mjukare.

ALLA BILTILLVERKARE BEKYMRAS över den växande kodvolymen. Mjukvaruutveckling har blivit fordonsföretagens största utmaning. Det illustreras av följande fakta:

- Fordon innehåller 10–100 miljoner kodrader
- Nya funktioner, leveranstider och kvalitet – allt hänger på mjukvara
- Det integrerade fordonssystemet innehåller 50–100 styrkretsar, över 150 distribuerade funktioner och mjukvara från olika leverantörer, konstruerade av utvecklingsgrupper som består av folk från olika företag och som är spridda över hela jorden.



Med virtuella prototyper kan du tidigarelägga mjukvaruutveckligen.

VAD GÄLLER personsäkerhetskritiska tillämpningar är det avgörande att identifiera nya metoder som kan förbättra mjukvarans kvalitet och pålitlighet, och dessutom kan skalas upp till ▶

den nya komplexiteten. Samtidigt som man håller kostnaderna för mjukvarutest nere.

Stora förändringar i arbetssätt har skett vad gäller utveckling av ECU:er (electronic control units) under de senaste 20 åren. Några av intressanta områden har varit:

- Modellering och simulering
- Kodgenerering
- AUTOSAR

EN ECU BESTÅR AV en hårdvaruplattform (digital och analog), en mjukvaruinfrastruktur (operativsystem, kommunikation) och en styralgoritm.

Validering och verifiering av ECU:er hänger ihop. De måste ske i samma omgivning som ECU:n interagerar med. Modellering och simulering blev en patentlösning i mitten av 90-talet. Modellerna för algoritmer och miljö kunde simuleras tillsammans.

Verktyg som Matlab och Simulink spelade en central roll för funktionell simulering av algoritmer. Behovet drevs fram av den ökande komplexiteten i de utvecklade systemen. Sådana miljöer kallas MIL (model-in-the-loop).

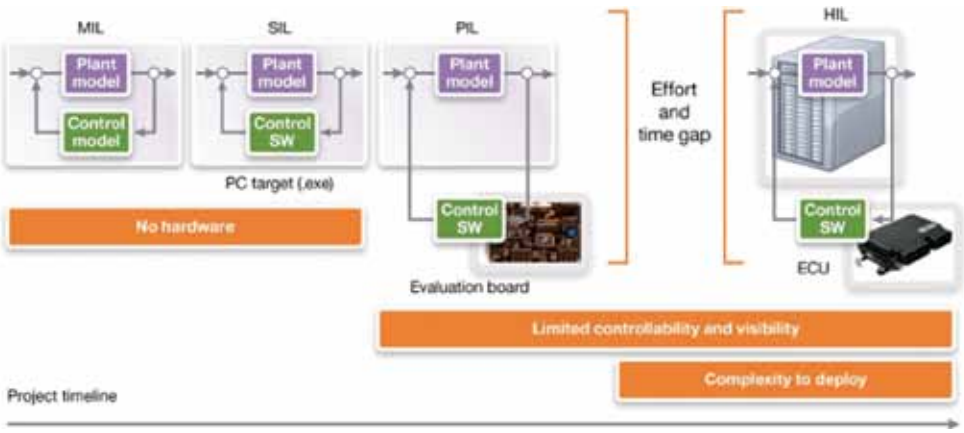
Efter modellering och simulering gjorde nästa koncept sitt insteg: kodgenerering. Med utgångspunkt från att utvecklaren hade modeller, var idén att börja ”kompilera” dessa modeller från grafiska representationer av algoritmerna, till högnivåspråk som C. Den genererade koden kunde sedan köras på en PC, på särskild hårdvara eller på en inbyggnadsplattform. Man kunde generera kod för styralgoritmerna eller miljömodellerna.

BRUKET AV KODGENERATORER gav upphov till flera sorters utvecklingsplattformar, bland dem:

- SIL (Software-in-the-Loop): styralgoritmen kompileras på en värd-PC och exekveras tillsammans med miljömodellen. SIL fokuserar på att koden är korrekt och att den exekverar snabbt.
- PIL (Processor-in-the-Loop): styralgoritmen körs på ett utvärderingskort. Därmed kan mjukvaran köras på samma arkitektur som målplattformen.
- HIL (Hardware-in-the-Loop): den hårdvara som kör programmet används tillsammans med kod som genereras för miljömodellen och exekveras på en dedikerad hårdvaruplattform.

TILL SLUT DÖK för några år sedan också AUTOSAR upp som standardarkitektur för mjukvara till fordonselektronik, med stöd från ledande OEM:er och tier-1-företag.

AUTOSAR-simulatorer har använts för utveckling av inbyggd programvara. De kompileras dock för exekvering på en värd-PC och beaktar inte styrkretshårdvaran.



In-the-loop-metodernas begränsningar.

Fokus ligger på tidig validering av konstruktion och den genererade koden med målet att slippa konstruktionsiterationer senare i processen.

Ovan beskrivna angreppssätt har gett ECU-utvecklare en lång rad användbara metoder, men det finns fortfarande luckor som inte adresseras.

- MIL och PIL tar inte hänsyn till de underliggande hårdvaruplattformarna, som ökar i komplexitet och funktionalitet – tänk till exempel på nya fordonselektronik-MCU:er som är multikärnor.
- HIL fungerar bara om ECU-hårdvaran finns tillgänglig. Det är ett stort steg både i tid och ansträngning mellan det att PIL och HIL finns tillgängliga.

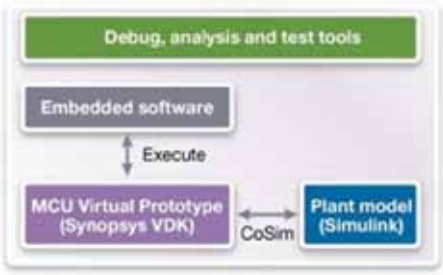
Det är uppenbart att modellering och simulering har haft stort värde. Det ser man om man tittar på utvecklingsprocessen som helhet. Koncepten har dock bara applicerats på styralgoritmen och miljömodellen. Inte på hårdvaruplattformen.

Orsaken är att hårdvarusimulering till dags dato inte har kunnat leverera den exekveringsprestanda som utvecklare förväntar sig.

Dessutom finns dessa simuleringsmiljöer typiskt hos halvledarföretagen och är inte så lätta att komma åt för de utvecklare som finns på OEM:er och tier-1-företag.

Virtuella prototyper har under de senaste åren visat sig vara en nyckel till att kunna börja arbeta med mjukvara tidigt i utvecklingsprocessen.

En virtuell prototyp är en modell av digital hårdvara på en hög beskrivningsnivå. Den skapas i språket SystemC (IEEE 1666) som bland annat stöder TLM (Transaction-level modeling).



Virtuell Hardware-in-the-Loop på en PC i Simulink.

Modellen körs på en vanlig skrivbords-PC under Linux eller Windows och den kan med hög prestanda emulera hårdvaruplattformar som styrkretsar och ECU:er (electronic control units), inklusive nätverk av ECU:er.

VIRTUELLA PROTOTYPER exekverar omodifierad binärkod som körs på hårdvaruplattformen. **En utvecklingsmiljö baserad på virtuell hårdvara har flera fördelar.** Med virtuella prototyper får du:

- Observerbarhet
- Styrbarhet
- Determinism
- Styrbarhet via skriptprogram
- Och de är icke-störande

Man använder ett separat parallellt utvecklingsverktyg för ovanstående.

Exempelvis kan utvecklare använda virtuella prototyper för att när som helst stoppa systemexekveringen (även i en heterogen multikärna). Därefter kan de läsa och modifiera värden, spåra exekveringen av program till hårdvara, och köra skriptprogram.

En programutvecklingsmiljö som använder en virtuell prototyp som målplattform för hårdvaran kallas för en VDK (Virtual Development Kit). VDK:er är lätta att underhålla, driftsätta och att ordna i arkiv, även när man arbetar i internationella utvecklingsgrupper.

Från VDK till Virtuell HIL-miljö

Virtuella prototyper ger dig teknik att kringgå begränsningarna i dagens utvecklingsmetoder. En VHIL-miljö (virtuell Hardware-in-the-Loop-miljö) utgörs av kombinationen av en VDK som kör program och exekveras tillsammans med en simulering av miljön. Den kan börja användas tidigare, kan driftsättas enkelt och utgör en mer effektiv utvecklingsplattform.

Framför allt gör den det möjligt för ut-

vecklaren att i en simulatormiljö integrera ECU:s sista nyckelelement – den fysiska hårdvaran. Generellt leder användandet av en VHIL till minskade utvecklingskostnader, förbättrad mjukvarukvalitet och mer robusta system.

FÖRDELARNA ERHÅLLS inom framför fyra områden:

- Tidig access: utvecklare får möjlighet att påbörja mjukvaran tidigt och tidigarelägga test och utförande.
- Produktivitet: HIL-miljön skapar synlighet och kontrollerbarhet vilket gör det möjligt att identifiera och åtgärda problem effektivare.
- Större tillgänglighet: ingenjörer jorden runt i alla tidszoner kan snabbt att komma åt miljön från sitt eget skrivbord. Den kan också användas i regressionstester dygnet runt.
- Enklare att driftsätta och underhålla: det går att från ett centralt håll sätta upp och underhålla miljön. Miljöerna är enkla att organisera i arkiv, kan distribueras via FTP eller från en serverfarm och är enkla att rekonfigurera.

STYRKRETSARNA BLIR ALLT KOMPLEXARE och för att slippa omkonstruktioner och kvalitetsproblem är det viktigt att det går att validera och verifiera programvara så tidigt som möjligt.

Du som är verksam inom fordonselektronik kan ta itu med många användningsfall när du får tillgång till en VDK och en virtuell HIL-miljö.

Det vanligast användningsfallet är before-hardware-availability. Mjukvaruutvecklaren kan börja koda och testa komplexa drivrutiner, multikärnor och AUTOSAR-mjukvara. I detta skede gör de utökade avlusningsmöjligheterna som en VDK erbjuder en signifikant nytta – den försäkrar att kodstacken redan inledningsvis är av hög kvalitet.

När VDK:n väl är integrerad med en virtuell HIL kan en lång rad användningsfall adresseras.

Det första är att systemintegrering och test kan göras virtuellt och utvecklare kan tidigarelägga utvecklingen av testet och dess exekvering. Det är allmänt vedertaget att om du hittar fel i konstruktionsprocessen tidigt, så reducerar du signifikant kostnaden för att åtgärda dessa fel senare.

Tidig felinjektion med hjälp av en VDK.

Det andra användningsfallet gäller kodtäckning och felinjektion. En stor fördel med en virtuell HIL-miljö är att fel kan injiceras var som helst.

Också systemets tillstånd kan modifieras, och även permanenta fel kan injiceras. Också kombinationer av extremvärden (corner cases) kan testas och valideras. Återigen: att tidigt förstå vilka fel som ska korrigeras och vilka modifieringar som ska inkluderas, är en stor fördel.

Ett sista exempel är regressionstest. Det är lätt att instantiera en virtuell miljö i en serverfarm och därefter över natten tidigt validera multipla mjukvarustackar representerande olika fordon eller olika fordonskonfigurationer.

VI HAR GETT EN LITEN HISTORIK över utvecklingsverktyg för mjukvara till ECU:er.

Nya utvecklingstrender som komplexare styrkretsar, säkerhetscertifiering och systemkomplexitet – driver fordons-OEM:er och leveranörer att vidareutveckla sin utvecklingsprocess.

Virtuella prototyper och VDK:er gör att modellering och simulering kan tas till hårdvaran och gör det därmed möjligt att skapa en virtuell miljö som innehåller programvara, ECU-hårdvara och miljön som kontrolleras.

Följande frågeställningar förtjänar belysning när en företagsledning överväger att övergå till en virtuell approach till sin fordonselektronik.

Bred driftsättning av VDK:er sker förmodligen som en del av en strategi att förbättra utvecklingsprocessen. Det har gjorts analyser av OEM:er och tier-one-leverantörer som visar att beslutsfaktorer från kvalitativa analyser ofta baseras på en önskan att motverka utvecklingsrisker, medan beslutsfaktorer från kvantitativ analys huvudsakligen handlar om att öka konstruktörernas produktivitet.

För företag som är nya till detta angreppssätt, kan ett pilotprojekt vara värt att överväga för att etablera intern erfarenhet. Ett pilotprojekt är som mest effektivt när det är väldefinierat och har som mål att exekveras parallellt med ett produktionsprojekt. Man bör också utnyttja den breda kunskap som finns inom området och som sprids på konferenser och branschevenemang.

AVSLUTNINGSVIS, när det handlar om att driftsätta virtuell teknik behöver man stöd av en leverantör med en mångfald kompetenser, bland dem erfarenhet av simulering och verktyg för verktygsteknik för hårdvara, väletablerade samarbeten inom leverantörskedjan för IP och halvledare inom fordonselektronik, en global närvaro för lokal support och driftsättningsstöd samt en finansiellt stabil återförsäljare med en investeringsapproach som stöder långsiktiga projekt inom fordonselektronik och industriell tillväxt. ■

REFERENSER

"Honda Recalls 936 000 More Vehicles for Electrical and Software Fixes", IEEE Spectrum Robert Charette/Wed, September 07, 2011 http://spectrum.ieee.org/riskfactor/green-tech/advanced-cars/honda-recalls-936000-more-vehicles-for-electrical-and-software-fixes

ISO 26262, Wikipedia.org - http://en.wikipedia.org/wiki/ISO_26262

Synopsys: Software Testing for Safety-Critical Systems, Victor Reyes.

Synopsys: Automotive Technical Bulletin Virtual Design and Verification Solutions for e-Mobility David W. Smith

Ito, Y., Sugure, Y., and Oho, S., "Virtual Validation Cloud : An Electronic System Validation Platform Based on Cloud Computing Resources," SAE Int. J. Passeng. Cars - Electron. Electr. Syst. 5(1):198-208, 2012, doi:10.4271/2012-01-0501.

Synopsys: Using VDKs for Automotive Systems Development – Quantitative and Qualitative Analysis of the Return on Investment from VDKs.