

Modellbaserad utveckling på alla nivåer



Modellbaserad utveckling av inbyggda system är populärt, men många företag bara utnyttjar bara en liten del av de möjliga fördelarna. Jonas Cornelsen, Henry Feng och Mats Roslund från Fengco Real Time Control AB beskriver här hur gränssnitten mellan de olika stegen från algoritmutveckling till testning kan förenklas.

De flesta företag jobbar modellbaserat idag i olika grader, som t ex att bygga funktioner i Matlab/Simulink. Detta är ett mycket bra steg men för att få de riktigt stora fördelarna krävs också att dessa modeller kan användas i de övriga utvecklingsstegen. Ett väl fungerande koncept leder till stora besparingar av både tid och resurser och därmed också pengar.

Modellbaserad utveckling är ett vedertaget begrepp idag som många företag idag säger sig arbeta efter. Vad är då modellbaserad utveckling och vilka fördelar har det? Den frågan har antagligen väldigt många olika svar och antalet fördelar beror naturligtvis på om man använder metoden i alla utvecklingssteg - från algoritmutveckling till testning och kalibrering av den slutliga produkten. Så här skulle vi vilja förklara modellbaserad utveckling:

- De algoritmer och funktioner som används utvecklas i ett grafiskt utvecklingsprogram med simuleringsmöjlighet (t ex Simulink och StateFlow). Fördelarna är att man får en bra överblick över funktionerna samt att det är betydligt lättare att göra modifieringar jämfört med det traditionella arbetssättet.

- När man går mellan de olika utvecklingsstegen (funktionsdesign, generering av kod till styrenhet, testning etc) skall man kunna återanvända arbete man gjort i tidigare steg. Till exempel skall man kunna automatgenerera kod av de modeller som man tidigare har utvecklat och verifierat. Om man vill testa en helt ny algoritm eller om man vill lägga till en ny funktion i en existerande styrenhet så ska gammalt arbete kunna återanvändas. Istället för att bygga

allt från början på nytt, uppdateras bara modellen med nya algoritmer och funktioner. I och med de starka kopplingarna mellan de olika utvecklingsstegen kan man återanvända mycket av det arbete som tidigare gjorts och behöver endast lägga ned tid på modellutveckling. Resten är automatik. Det är i detta sammanhang de absolut största fördelarna ligger.

- De algoritmer som utvecklats kan senare återanvändas i andra projekt och man kan på så sätt bygga upp ett bibliotek av algoritmer och funktioner som sedan kan sätta ihop för önskad funktionalitet.

- De tester man gör skall vara repeterbara dvs om man upptäcker ett fel vid en körning så skall man kunna återskapa exakt samma fel. Detta får man lättast genom att automatisera testsekvenserna. Detta har dessutom fördelarna att det är lätt att dokumentera och versionshantera de tester som utförts.

- Att jobba modellbaserat fullt ut innebär att man lämnar det traditionella sättet med Word-dokument som skickas mellan olika avdelningar. Man låter istället de entydiga modellerna bli den gemensamma plattformen som alla avdelningar jobbar efter. Det innebär en stor besparing i tid och möda för kommunikation om specifikationer och överlämnande av delresultat mellan avdelningar.

V-CYKELN

Den utvecklingsprocess som här används är indelad i fem olika steg;

- 1) Design av algoritmer och funktioner med Simulink och StateFlow
- 2) Snabb test av algoritmerna på prototyp hårdvara och styrenhet.
- 3) Automatisk kodgenerering och programmering av styrenhet
- 4) Test av styrenhet med Hardware-In-the-Loop simulator
- 5) Kalibrering av styrenheten på slutgiltiga platsen.

MODELLFRAMTAGNING

Med ett modelleringsprogram t ex Simulink och StateFlow är det enkelt att bygga upp och testa sina algoritmer och funktioner. Avancerade simuleringsmöjligheter gör att man kan testa olika testfall och se om sina funktioner uppfyller alla krav. För att simuleringen skall vara pålitlig bör man också ha en modell över de objekt

De olika utvecklingsstegen i modellbaserad utveckling kan delas in i fem olika steg - V-cykeln.

som styrenheten i slutänden skall styra. Då får man en sluten loop som allmänt kallas MIL-simulering (Model-In-the-Loop).

PRAKTISK VERIFIERING

Nästa steg är att testa algoritmerna mot i verkliga processen, vilket innebär att alla sensorer, ställdon och kommunikationsbussar kopplas in. Dessa konfigureras väldigt enkelt genom att lägga in I/O-block direkt i sin modell. Kod genereras automatiskt och laddas ner på ett väldigt kraftfull processorkort som kommer att agera som en prototypstyrenhet. Detta sätt kallas Fullpass. Alla parametrar och variabler är mät- och kalibrerbara från programmet ControlDesk.

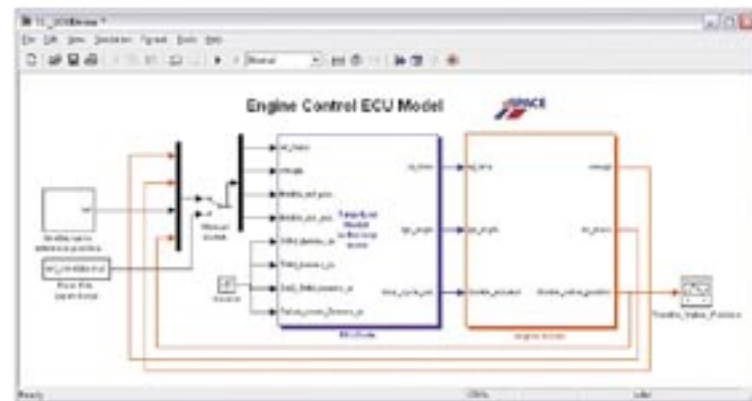
Fördelarna här är att det går väldigt snabbt att koppla upp sig mot hårdvara med full tillgång till I/O-signaler. Dessutom är det exakt den modell som man tagit fram i tidigare stadium som exekveras utan att man har skrivit en enda rad kod för hand. Detta steg kallas allmänt för RCP (Rapid Control Prototyping). Om det finns en färdig styrenhet med sensorer och drivsteg och dess styrfunktion ska modifieras, kan snabbtest utföras tillsammans med prototyp hårdvara som endast innehåller ersättningsfunktioner i styrenheten. Detta sätt kallas Bypass.

AUTOMATISK KODGENERERING

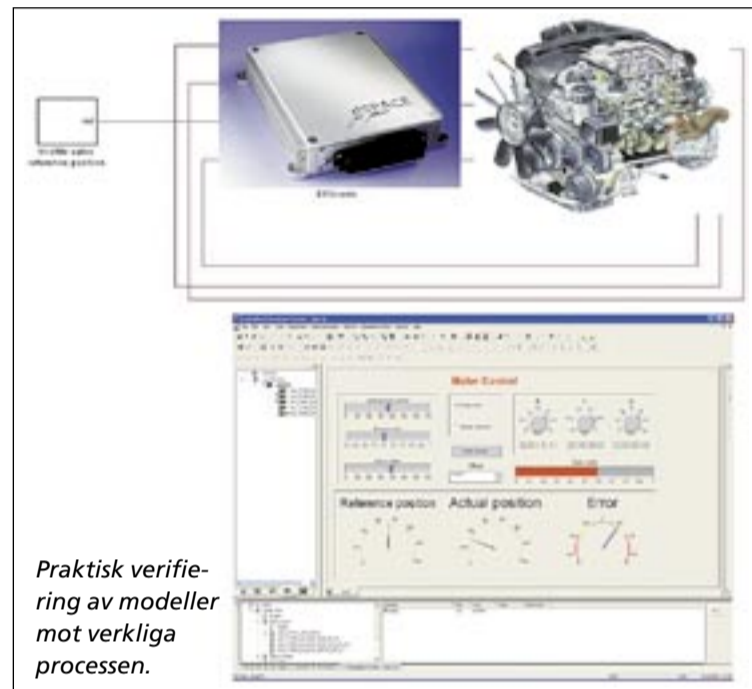
Produktionskod för styrenheten kan automatisk genereras direkt från modellerna med programmet TargetLink. Många simuleringsalternativ finns som gör det enkelt att kontrollera att koden och modellen ger samma resultat. Stöd finns inbyggt för OSEK realtidsoperativsystem samt för att generera ASAM2-filer.

Fördelarna med automatisk kodgenerering är många men de viktigaste är att modell och kod är alltid konsistenta samt att det går snabbare att genomföra modifieringar. Koden som kommer ut är dessutom lika effektiv som handskrivna kod och är dessutom väl dokumenterad.

Vid detta skede har man en programmerad styrenhet som man skulle kunna plugga in och



Med ett modelleringsprogram t ex Simulink och StateFlow är det enkelt att bygga upp och testa sina algoritmer och funktioner.



Praktisk verifiering av modeller mot verkliga processen.

testköra. Naturligtvis måste den testas noggrant och se så att den uppfyller de krav som man ställt på den. Detta kan man göra på två sätt; i testbänk mot verklig hårdvara eller med en HIL-simulator (Hardware-In-the-Loop).

En HIL-simulator kör en mjukvarumodell av t ex en riktig motor och skickar ut korrekta sensorsignaler och CAN-meddelande till ECU'n. På detta sätt tror styrenheten att den är inkopplad mot en riktig motor och kommer att agera därefter. Fördelarna med denna testning är att man på ett effektivt och säkert sätt kan testa hur styrenheten agerar vid kortslutning, glapp eller felaktiga signaler - (dvs, diagnostik) utan att man riskerar att förstöra en dyrbar motor.

En annan fördel är att testerna kan automatiseras och t ex köras nattetid och sedan utvärderas på dagtid. Man kan naturligtvis också testa nätverket och distribuerade funktioner genom att koppla ihop ett antal styrenheter. Alla tester blir dessutom repeterbara och återanvändbara i andra projekt.

KALIBRERING

Kalibrering kan ske i det sista steget, men kan redan utföras i alla tidigare faser tack vare gemensam parameteruppsättning i alla faser. Genom att använda den ASAP2-fil som automatiskt genereras vid kodgenereringen kan kalibreringsprogrammet CalDesk enkelt komma åt och ändra

variabler direkt på styrenheten i realtid. Man kan komma åt informationen i styrenheten på tre olika sätt; genom debugporten, över CAN samt direkt från adress/databussen. Vilket alternativ som är att rekommendera beror på applikationen samt vilken processor man använder. CalDesk kan även hantera mätmoduler från t ex Ipetronik och ICM samt fungera som en CANalyzer - parallellt med kalibreringen.

ITERATIV PROCESS

Modellbaserad utveckling är ett mycket effektivt sätt att arbeta på, men det finns många olika nivåer på modellbaserad utveckling. Det viktigaste är inte varje utvecklingssteg för sig utan hur dessa stegen är förbundna med varandra. Eftersom utvecklingen av en ny produkt oftast är en iterativ process, innebär detta många småändringar hela tiden. Om varje ändring i princip innebär att mycket arbete måste göras om så inser man snart de stora fördelarna om alla steg grundar sig på samma bas som t ex Simulink och StateFlow. I så fall görs de modifieringar som behövs i modellen och man kan i princip återuppta arbetet där man slutade. Det innebär att man genom att använda modellbaserad utveckling i hela V-cykeln kan spara både mycket tid och pengar.

JONAS CORNELSEN, HENRY FENG, MATS ROSLUND, FENGCO REAL TIME CONTROL AB
www.fengco.se