



Kan AI verkligen granska ett kretsschema?

Varför räcker inte generativa AI-modeller för att granska elektronikkonstruktioner? Vad krävs för att AI:n ska kunna hitta rätt information, koppla slutsatser till sina källor och ge ett resultat som faktiskt går att verifiera?

Ett konstruktionsfel i en digital isolator tog sig hela vägen till hårdvaran. Isolatorns gränssnitt matades med 5 V medan den anslutna bussen arbetade med 3,3 V-logik. Resultatet blev att signalerna inte nådde den logiknivå som komponenten behövde för att säkert tolka dem som höga. Orsaken gick att härleda direkt ur schemat och databladet, men ingen hade jämfört rätt parametrar innan kortet beställdes.

Många sena konstruktionsfel är just sådana: inte subtila, utan vanliga fel som ingen hann leta efter eller samband som låg utspridda över flera dokument. Generativ AI verkar därför vara en naturlig lösning. Den kan läsa stora informationsmängder och resonera om samband, men är samtidigt tränad att ge ett svar även när underlaget är ofullständigt.

FRÅGAN ÄR INTE BARA om AI kan hitta fel, utan om slutsatsen går att verifiera. En bättre informerad gissning är fortfarande en gissning. Först när AI:n kan visa sitt underlag går det att använda den för konstruktionsgranskning.



AI-baserad konstruktionsgranskning är inte främst en fråga om större modeller. Det är en fråga om systemarkitektur: hur konstruktionen representeras, hur rätt underlag väljs ut och hur varje tekniskt påstående kan spåras tillbaka till sin källa.

Elektronikutveckling har redan flera lager av automatiserad kontroll. Electrical Rule Check, ERC, hittar exempelvis oanslutna pinnar, drivkonflikter och inkonsekvenser i schemat. Design Rule Check, DRC, verifierar att layouten följer definierade regler för geometri och tillverkningsbarhet.

Verktygen är nödvändiga, men de verifierar främst konsistens och regeluppfyllelse. En konstruktionsgranskning måste dessutom avgöra om konstruktionen faktiskt gör det konstruktören avsåg. Ett schema kan vara elektriskt konsekvent och ändå innehålla fel komponentvariant, en underdimensionerad matning eller en signal i fel spänningsdomän.

Det är därför en erfaren granskare inte bara läser vad som är anslutet till vad. Granskaren försöker förstå driftsfallen, kontrollera antaganden och jämföra den faktiska kopplingen med datablad och konstruktionsavsikt.

Av Daniel Rhodin, E-Sharp

Daniel Rhodin är grundare av E-Sharp, som utvecklar AI-baserade verktyg för verifierbar granskning av elektronikkonstruktioner.



NÄR EN KONSTRUKTIONSGRANSKNING missar ett fel är det sällan ett bevis på att granskarna saknar kunskap. Det är oftare ett skalningsproblem. Ingen människa kan hålla varje komponentvariant, pinnfunktion, spänningsdomän, fotnot och elektriskt gränsvärde aktivt i huvudet samtidigt. Under tidspress riktas uppmärksamheten mot det som verkar viktigast, medan vardagliga fel i mindre uppmärksammade delar av konstruktionen kan passera.

Maskinen är bättre på det uttömmande, medan människan är bättre på det svårbedömda. Potentialen ligger därför inte i att ersätta ingenjörens omdöme, utan i att låta systemet ställa samma frågor till varje relevant nät, matning och komponent och därefter föra de verkligt svåra fallen till en människa.

DET ÄR FRESTANDE att exportera ett schema, bifoga några datablad och fråga: "Ser du några fel?" I små demonstrationer kan det ge im-

ponerande resultat. I verklig produktutveckling uppstår tre grundläggande problem.

1. Hela konstruktionen ryms inte i ett kontextfönster

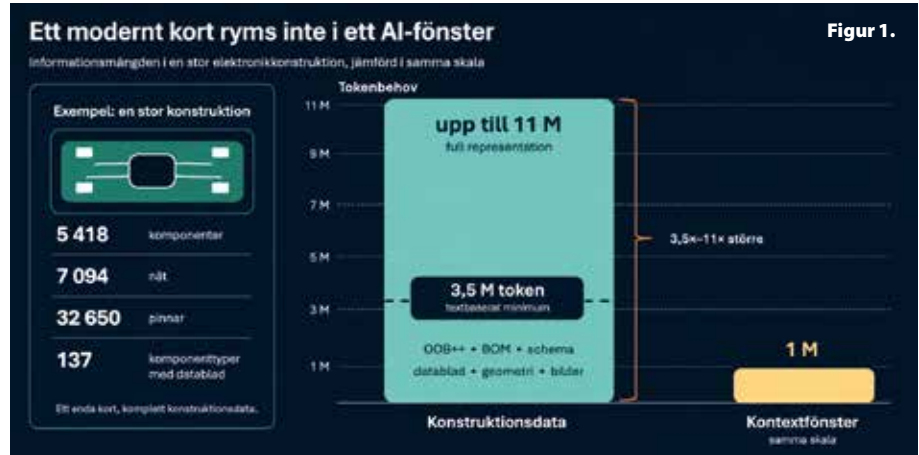
Ett kontextfönster är den informationsmängd som språkmodellen kan ha tillgänglig samtidigt när den skapar sitt svar. När fönstret fylls med schema, stycklista, konstruktionsdata och datablad konkurrerar allt material om modellens uppmärksamhet.

Som figur 1 visar kan underlaget för ett större kort motsvara flera gånger ett kontextfönster på en miljon token. Den svåra delen är sällan att läsa mer data. Den svåra delen är att hitta rätt data.

En modell som får hela schemat, alla datablad och all kringinformation måste fortfarande förstå vilken tabellrad, fotnot och komponentvariant som är relevant för just den signal eller matning som granskas.

2. Datablad är inte vanlig text

Datablad är strukturerade tekniska dokument. Betydelsen ligger ofta i kombinationen av rubrik, tabell, rad, kolumn, enhet, testvillkor och fotnot. En maximal inspanning kan exempelvis vara ett absolut gränsvärde, ett rekommenderat arbetsområde eller ett villkor för en viss variant, kapsel eller temperatur.



Figur 1.

Problemet är att en vanlig språkmodell inte nödvändigtvis ser tabellen på samma sätt som en ingenjör. Vid PDF-extraktion presenteras innehållet ofta som en sekvens av ord och tal. Om rad- och kolumnstrukturen går förlorad försvinner också relationen mellan värdet och villkoret som gör värdet giltigt.

Modellen kan då hitta ett tal som verkligen finns i dokumentet men koppla det till fel parameter och ändå formulera en fullt trovärdig slutsats.

För teknisk granskning räcker det därför inte att göra dokumentet sökbar. Systemet måste kunna bevara eller återskapa tabell-

strukturen, läsa en sida visuellt när textutdraget är tvetydigt och visa exakt var informationen kommer ifrån.

3. Generativa språkmodeller är optimerade för att svara

I en konstruktionsgranskning är ett trovärdigt påhitt farligare än ett uteblivet svar. En språkmodell kan återge ett vanligt pinnamn, ett typiskt gränsvärde eller en standardkoppling trots att den aktuella komponenten avviker.

Problemet är inte att modellens minne alltid är fel, utan att det är rätt tillräckligt



ofta för att inge förtroende. De sällsynta avvikelserna är samtidigt precis de detaljer som konstruktionsgranskningen ska fånga: ett variantsuffix som ändrar uppstarts-beteendet, en pinne som flyttats mellan kapslingar eller ett absolut gränsvärde som är 5,5 V i stället för 6 V.

Minnet får därför aldrig behandlas som bevis. Systemet måste skilja mellan det som modellen känner igen och det som verifierats i den aktuella konstruktionens källor.

I STÄLLET FÖR ATT SKICKA KONSTRUKTIONEN som en odifferentierad dokumenthög till en språkmodell kan systemet bygga upp en strukturerad modell av komponenter, nät, pinnar, varianter och relationer. Därmed går det att ställa konkreta frågor om exempelvis en spänningsdomän, en matningsväg eller en viss komponentvariant.

Databladet måste på motsvarande sätt behandlas som tekniska källor. Det räcker inte att hitta ett ord eller ett tal. Systemet behöver kunna visa vilken tabell, sida eller passage som stöder slutsatsen.

När en kontroll görs kan bara den relevanta delen av konstruktionen och de dokumentpassager som behövs hämtas. Språkmodellen får då ett avgränsat problem med definierade objekt och spårbara källor, i stället för att söka genom hela underlaget eller fylla luckor med generell kunskap.

DEN VIKTIGASTE PRINCIPEN för verifierbar AI-granskning är att systemet ska citera eller avstå. Om en slutsats kräver ett värde ur ett datablad ska källan kunna visas. Om underlaget inte räcker ska systemet inte presentera antagandet som ett faktum.

Resonemang behövs för att koppla samman konstruktionsdata, regler och dokumentation, men det måste gå att skilja mellan observerade fakta, härledda slutsatser och sådant som fortfarande är okänt.



Principen kan få systemet att verka mindre imponerande i en demonstration. I praktisk ingenjörssamhet är den en styrka. Ett synligt kunskapsgap kan hanteras. Ett dolt antagande riskerar att bli en del av konstruktionen.

Ett fungerande granskningssystem måste inte bara redovisa vad det har hittat, utan också vad det hade möjlighet att kontrollera. I Probe beskrivs täckningen som täckt, partiell eller blind. Begreppen säger inte om kortet är bra, utan om underlaget räcker för att genomföra den aktuella kontrollen.

En kontroll av en matningsspänning kan vara täckt när schemat, den beställningsbara komponentvarianten och relevanta databladsvillkor finns tillgängliga. Den kan vara partiell när komponenten är identifierad men ett avgörande villkor i databladet ännu inte har kunnat läsas säkert. Den är blind om systemet exempelvis inte kan avgöra vilken variant som faktiskt är monterad.

Ett tomt resultat betyder alltså inte automatiskt att granskningen inte hittade något fel. Det kan också betyda att granskningen saknade synlighet, och den skillnaden måste vara explicit.

EN REGELTRÄFF är en startpunkt, inte ett färdigt fynd. Regler är bra på att hitta definierade mönster, medan AI:n kan använda konstruktionsdata och dokumentation för att sätta träffen i sitt tekniska sammanhang.

Vid granskningen av ett av våra kort genererade systemet 16 regelträffar. Efter analys återstod sex relevanta observationer och fyra verkliga fel, varav två blockerande. Tre flaggningar från samma regel visade sig vara falska positiva. Den fjärde ledde till ett verkligt fel som kunde identifieras genom konstruktionsdata och komponentens datablad.

Regeln hade alltså inte formulerat hela fyndet, men den hade riktat granskningen till rätt område.

En flaggning är inte ett svar. Värde ligger i att öka täckningen, rikta ingenjörens uppmärksamhet och göra resonemanget granskningsbart. Falska positiva träffar är hanterbara när systemet visar varför de uppstod. Falsk säkerhet är betydligt farligare.

FÖR ATT AI-BASERAD konstruktionsgranskning ska fungera i praktiken krävs minst fem saker:

- En domänmodell som representerar komponenter, nät, pinnar, varianter och deras relationer.
- Selektiv hämtning av relevant konstruktionsdata och dokumentation, i stället för att allt skickas in samtidigt.
- Källor som bevarar tillräcklig struktur för att tabeller, villkor och fotnoter ska kunna tolkas korrekt.
- En strikt princip för att citera eller avstå, så att antaganden inte maskeras som verifierade fakta.
- Synlig täckning, så att användaren kan se var systemet har fullständigt, partiellt eller otillräckligt underlag.

Det handlar alltså mindre om hur stor språkmodellen är och mer om vilket underlag den får, hur informationen struktureras och om slutsatserna går att kontrollera.

MÄNNISKAN BESTÄMMER FORTFARANDE. AI-baserad granskning ersätter inte mänsklig expertis. Systemets uppgift är att flytta arbetet från ostrukturerad genomläsning till riktad teknisk bedömning: lyfta fram misstänkta samband, visa relevant stöd och markera kunskapsluckor.

Ingenjören fattar fortfarande beslutet.

För teknisk granskning är det mindre viktigt hur mänskligt AI-svaret låter. Det viktiga är om systemet kan visa varifrån ett värde kommer, skilja en komponentvariant från en annan och redovisa när underlaget inte räcker.

Det mest värdefulla svaret är därför inte alltid ett fynd. Ibland är det en exakt källa och ibland ett tydligt besked om att frågan inte går att avgöra. Först när den skillnaden blir synlig går AI från avancerad gissningsmaskin till praktiskt ingenjörssverktyg. ■

FAKTA

Vad är Probe?

Probe är ett granskningssystem för elektronikdesign utvecklat av E-Sharp AB, som bygger upp en strukturerad modell av scheman, BOM-data, tillverkningsunderlag och datablad. Istället för att skicka hela konstruktionen till en språkmodell hämtar Probe endast den del av underlaget som är relevant för den aktuella frågan och kopplar varje slutsats till sin källa. Systemet används för att identifiera feltyper som ofta ligger utanför ERC- och DRC-kontrollernas räckvidd.