

CODEN: LUTFD2/(TFRT-5455)/1-81/(1992)

Modellering och simulering av kraftnät i Omola

Magnus Persson

Institutionen for Reglerteknik
Lunds Tekniska Högskola
Mars 1992

Department of Automatic Control Lund Institute of Technology P.O. Box 118 S-221 00 Lund Sweden		<i>Document name</i> Master Thesis	
		<i>Date of issue</i> March 1992	
		<i>Document Number</i> ISRN LUTFD2/TFRT--5455--SE	
<i>Author(s)</i> Magnus Persson		<i>Supervisor</i> Sven Erik Mattsson, Magnus Akke	
		<i>Sponsoring organisation</i> Sydkraft AB	
<i>Title and subtitle</i> Modellering och simulering av kraftnät i Omola (Modelling and simulation of power systems using Omola)			
<i>Abstract</i> <p>In this thesis the possibilities to use the new universal, equation-based and object-oriented modelling language Omola for modelling and simulation have been investigated with a positive result. The power system engineers need powerful computer-aided tools to be able to design power systems which are stable under any conceivable disturbance. Consequently, modelling and simulation of electric power systems is an important and realistic application. Power system models contain a large number of algebraic equations, which can not be removed analytically. Thus today's general continuous time simulation languages all belong to the CSSL family are not useful, since they assume a model on explicit state form. We need to be able to simulate differential-algebraic systems. Omola supports this requirement. Today special purpose programs are used for simulation of power systems. A drawback of available special-purpose simulation programs is that they are closed and rigid. It is not possible for a user to add a missing component. An object-oriented tool like Omola gives flexibility. In this thesis we focus on transient stability, i.e, modelling the effects of disturbances as losses of generating or transmission facilities, sudden load changes, or momentary faults. Models for generating units, buses, transmission lines and loads have been developed. These model components have been used to implement a 16 machine model of the Scandinavian power grid. Simulations agree very well with results obtained from PSS/E which is a widely used commercial power system simulator.</p>			
<i>Key words</i> Modelling, simulation, power system, transient stability, object-orientation			
<i>Classification system and/or index terms (if any)</i>			
<i>Supplementary bibliographical information</i>			
<i>ISSN and key title</i> 0280-5316			<i>ISBN</i>
<i>Language</i> Swedish	<i>Number of pages</i> 81	<i>Recipient's notes</i>	
<i>Security classification</i>			

Innehåll

Förord	3
1. Inledning	4
2. Modellstruktur	6
3. Komponentmodeller	8
3.1 Resistor och induktor i serie	9
3.2 Transmissionslinjen	10
3.3 Bussen	11
3.4 Lasten	11
3.5 Generatorn	14
3.6 Grafisk representation av delmodellerna	18
4. Initialvärden och störningar	20
4.1 Initialvärden	20
4.2 Modellering av störningar	20
5. Modellering och simulering i OmSim	23
5.1 Definition av modell	23
6. Simulering av Nordel	27
6.1 Avbrott på en ledning	27
6.2 Steg i referensspänningen	29
6.3 Steg i last	29
7. Slutsatser	33
Referenser	34
Appendix A Modeller	36
A1 - Omolamodeller för grundläggande enheter	36
A2 - Omolamodeller för linjer och laster	39
A3 - Omolamodeller för genererande enheter	43
Appendix B En 16-generatormodell av Nordel	46
Appendix C Grafisk representation av Nordel - modellen	78

Förord

Under tiden med mitt examensarbete är det många frågor som har kommit upp, och jag har kommit i kontakt med många personer som har hjälpt mig. Jag vill därför tacka alla på Institutionen för Reglerteknik för kommentarer och råd.

Ett speciellt tack vill jag ge till min handledare Sven Erik Mattsson för hans stöd och hjälp. Jag vill även tacka min handledare på Sydkraft, Magnus Akke, och Alf Larsen för tillhandahållande av simuleringsresultat.

Till slut vill jag tacka Sydkraft för det ekonomiska stödet som jag fått under tiden med examensarbetet.

Magnus Persson
Lund, Mars 1992

1. Inledning

Målet med detta examensarbete är att undersöka om och hur Omola kan användas för att modellera och simulera kraftsystem genom att

1. utveckla modeller för genererande enheter, bussar, transmissionledningar och laster,
2. implementera en modell med sexton generatorer av det nordiska elnätet,
3. använda modellen för att simulera urkoppling av transmissionslinjer, laständringar samt steg i de genererande enheterna
4. jämföra resultaten med simulering gjord på Sydkraft med PSS/E.

Omola är ett universellt, ekvationsbaserat och objekt-orienterat modelleringsspråk, som har utvecklats vid Institutionen för Reglerteknik, LTH (Andersson, 1989, 1990; Mattsson och Andersson, 1990).

Det nordiska nätet försörjer miljontals människor och tusentals industrier med elektricitet. För att kunna göra nätet okänsligt för alla tänkbara störningar behövs kraftfulla datorhjälpmedel. Man måste kunna simulera effekten av olika störningar för att se hur nätet kommer att uppföra sig. För olika typer av störningar behövs olika sorters modeller. Vi kommer här att koncentrera oss på simulering av dynamisk - transient stabilitet och kommer därför att bygga modeller för simulering av brott på ledning, plötsliga laststörningar samt störningar i de genererande enheterna. Dessa störningar har oftast stor amplitud och olinjära modeller behövs, men man kan anta att strömmar och spänningar är sinusformiga. I den tidsskala som man är intresserad av vid studie av transient stabilitet, bestäms beteendet hos de genererande enheterna huvudsakligen av synkrogeneratorerna. De primära energikällorna såsom vattenturbiner och kärnreaktorer har långsammare dynamik. Möjligheterna att initieellt kompensera för transienta störningar genom att reglera de primära energikällorna är små.

Teorin för synkrogeneratorn är känd sedan länge och finns beskriven i läroböcker (t.ex. Bergen, 1986; Elgerd, 1983). Teorin för modellering av större kraftsystem är också välkänd (t.ex. Lindahl, 1971; Andersson och Fouad, 1977; Vournas och Flemming, 1978), men det är ett gap till de simuleringsverktyg som behövs. Det är ett tidsödande arbete att simulera ett stort kraftsystem om all programmering skall göras i Fortran. De stora företagen har utvecklat sina egna simuleringsprogram. Ett av Asea Brown Boveris simuleringsprogram kallas Simpow (Adielson, 1982). Ett amerikanskt program beskrivs av Stagg och El-Abiad (1968). Det mest använda simuleringsprogrammet är PSS/E (Power System Simulator / Engineering) som är utvecklat av det amerikanska företaget, Power Technology Inc. Alla jämförande simuleringar i denna rapport är gjorda med PSS/E. Akke (1989) har utvecklat ett simuleringsverktyg för kraftsystem med linjära modeller i Matlab. Vournas och Hatziargyriou (1991) har utvecklat några datorverktyg för design av kraftsystem.

Dagens generella simuleringsverktyg för tidskontinuerliga modeller är inte användbara då dessa kräver att modellen är skriven på formen $\dot{x} = f(t, x)$. När man gör en modell av ett stort kraftsystem får man också ett stort antal algebraiska ekvationer. Man behöver kunna simulera differential-algebraiska system. Omola uppfyller dessa krav.

Ett objektorienterat modelleringspråk som Omola är mycket flexibelt, användaren kan mycket enkelt addera en ny modell, eller modifiera redan existerande. Detta är mycket svårt att göra i ett specialprogram för simulering av kraftsystem, då dessa är låsta till en viss modell. En annan nackdel med dessa är att de innehåller ofantligt mycket kod för modellering av alla på marknaden förekommande komponenter.

En nackdel med ett generellt verktyg som Omola kan vara att det går snabbare att simulera ett kraftsystem med ett specialprogram. Vi vet emellertid att datorerna blir snabbare och billigare. Framsteg har också gjorts när det gäller att ta fram effektiv kod för numerisk lösning av ekvationssystem. Simulatoren OmSim, som simulerar modellerna i Omola, kontrollerar så att inga singulariteter finns, sorterar ut tidsinvarianta delar samt försöker eliminera algebraiska variabler för numerisk lösning.

Rapporten är disponerad på följande sätt. I kapitel 2 diskuteras den allmänna strukturen av en kraftsystemmodell. De grundläggande modellkomponenterna identifieras och gränssnitten för att beskriva deras hopkoppling specificeras. I kapitel 3 utvecklas modellerna för de olika komponenterna. Även ikoner och grafik tas upp. Initialisering av de olika komponenterna samt modellering av störningar diskuteras i kapitel 4. I kapitel 5 visar vi hur komponentmodellerna kan användas för att göra en modell av ett enkelt kraftsystem, och hur man använder OmSim vid simulering. I kapitel 6 implementeras 16-generatormodellen av det nordiska elnätet Nordel, och resultaten från några simuleringar jämförs med resultaten från PSS/E. Kapitel 7 innehåller slutsatser.

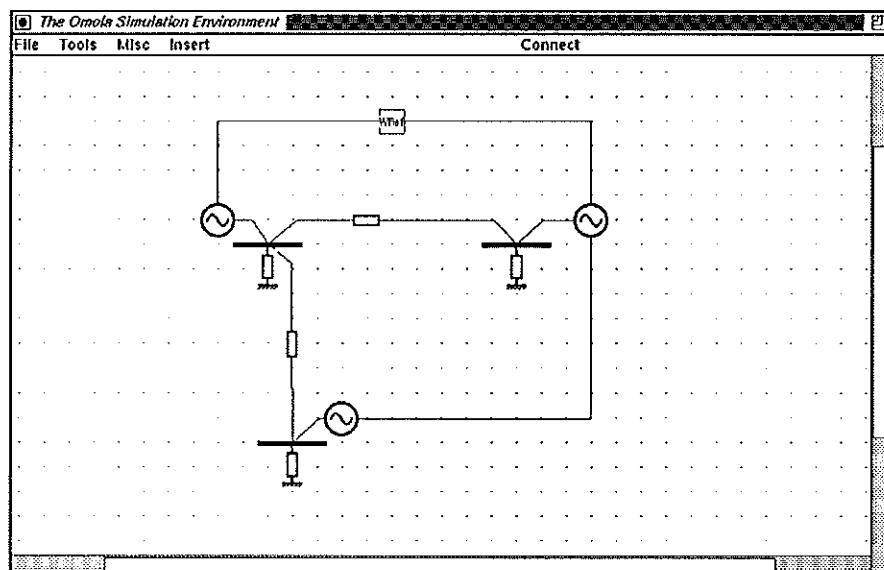
2. Modellstruktur

Ett elnät, även det minsta, är oerhört komplext och innehåller massvis av komponenter. Det är alltså inte möjligt att modellera alla komponenter i detalj. Den vanliga ansatsen för att få modeller av rimlig storlek är att arbeta med aggregerade modeller. Vi kommer att fokusera på en abstraktionsnivå där vi betraktar elnätet som ett transmissionssystem. Vi modellerar till exempel ett kärnkraftverk eller alla genererande enheter inom ett område som en genererande enhet. Figur 2.1 visar ett litet kraftnät. De grundläggande komponenterna är:

1. genererande enheter, symboliserade av cirklar med en sinusvåg
2. bussar, symboliserade av ett tjockt svart, horisontellt streck
3. transmissionslinjer som förbinder bussarna, symboliserade av rektanglar
4. laster, symboliserade av rektangel och jordsymbol.

När man har gjort modeller för dessa komponenter i Omola, kan man definiera en modell av ett kraftsystem genom att rita ett diagram av den typ som visas i figur 2.1.

När de grundläggande modellkomponenterna identifierats är nästa steg i modellstruktureringen att bestämma delmodellernas gränssnitt. I Omola beskriver man växelverkan mellan modeller genom att förbinda terminaler. Terminaler är enkla eller sammansatta variabler och en förbindelse mellan två terminaler betyder huvudsakligen att dessa skall ha samma värde. För att man lätt skall kunna bygga en modell genom att koppla ihop delmodeller krävs enhetliga terminaler. Växelverkan mellan elektriska komponenter beskrivs av Kirchhoffs ström- och spänningslagar. Generellt skulle terminalen hos en transmissionslinje för ett trefassystem innehålla tre spänningar och tre strömmar. Vid symmetriska fel och störningar kan trefassystemet analyseras som tre identiska enfassystem. Kritiska felfall för transient och dynamisk stabilitet är oftast symmetriska. Därför kan analysen utföras på en fas. Om alla sinusformiga variabler representeras av amplitud och fas, behöver inte den



Figur 2.1 Tre generatorer med laster, linjer och impedanser modellerade i Omola

numeriska lösaren lösa upp varje period, den kan i stället ta längre steg och därmed förkortas beräkningstiden. Detta märks speciellt då systemet är nära stationaritet och variationen i amplitud och fas är liten. Tricket är att använda Parks transformation och uttrycka spänningen, V_a , och strömmen, I_a , i fas a som $V_a = V_d \sin \theta + V_q \cos \theta$ och $I_a = I_d \sin \theta + I_q \cos \theta$, där vinkeln θ idealt är sådan att $\theta = \omega_0 t$, med ω_0 som den synkrona vinkelfrekvensen; $\omega_0 = 2\pi f_0$, där f_0 är den synkrona frekvensen, vanligtvis 50 eller 60 Hz.

I Omola definierar vi en terminal med de fyra komponenterna V_d , V_q , I_d och I_q som

```
In_El_Term ISA Base::RecordTerminal WITH
  Vd, Vq ISA VoltageTerminal;
  Id, Iq ISA InCurrentTerminal;
END;
```

Man säger att klassen `In_El_Term` har fyra komponenter och att den är en underklass till den fördefinierade Omolaklassen `Base::RecordTerminal`, som inte har några komponenter eller attribut.

En modellkomponent som skall användas som terminal måste vara av underklassen `Terminal`. I Omola finns två underklasser till `Terminal` fördefinierade: `RecordTerminal` och `SimpleTerminal`. `SimpleTerminal` diskuteras nedan. Underklasser av `RecordTerminal` kan ha komponenter som är `RecordTerminal` eller `SimpleTerminal`.

I fortsättningen kommer vi att använda stora bokstäver till nyckelord i Omola: `ISA`, `WITH`, `END` osv. Omola skiljer inte på stora och små bokstäver. Alla definierade klasser tillhör ett bibliotek. De fördefinierade Omolaklasserna `Model`, `Terminal`, `RecordTerminal`, `SimpleTerminal`, `Parameter` osv. tillhör biblioteket `Base`. Vi kommer att lägga våra modeller i tre bibliotek som kallas `Basic_Parts`, `Impedances` och `Generators`. Klassen `In_El_Term` ovan lägger vi i biblioteket `Basic_Parts`, då denna tillhör baskomponenterna. För att referera till spänningen V_d i `In_El_Term` skriver vi `Basic_Parts::In_El_Term.Vd`. De fyra komponenterna i `In_El_Term` är alla underklasser till `SimpleTerminal` vilken är definierad som

```
Base::SimpleTerminal ISA Base::Terminal WITH
  value      TYPE Real;
  direction  TYPE (Across, In, Out) := 'Across;
  quantity   TYPE String := "electric.voltage";
  unit       TYPE String := "V";
  causality  TYPE (Undefined, Read, Write) := 'Undefined;
  variability TYPE (TimeVarying, Param, Constant, Bound) :=
                                     'TimeVarying;

  initial    TYPE Real := 0.0;
  default    TYPE Real := 0.0;
END;
```

Attributet `value` lagrar värdet på terminalen. Växelverkan mellan delmodeller beskrivs genom att man kopplar ihop terminaler. En hopkoppling mellan två `record-terminaler`, betyder att deras komponenter kopplas samman med varandra. För två terminaler `T1` och `T2` som som är underklasser till `In_El_Term` betyder det att `T1.Vd` kopplas samman med `T2.Vd`. Detsamma sker med `Vq`, `Id` och `Iq`. Hopkoppling av två enkla terminaler kan ha två olika betydelser; Antingen att hopkopplade terminaler skall vara lika eller ha nollsumma. Om man vill att hopkoppling av en terminaltyp skall betyda likhet ger man attributet `direction` värdet `across`. Likhet vill man ha vid hopkoppling av

terminaler som till exempel representerar tryck, temperatur eller spänning. Nollsummering vill man ha för flöden, krafter, moment, elektrisk ström m.m. Dessa har en associerad riktning; är positiv riktning in eller ut ur komponenten? Att en enkel terminal är nollsummeringsterminal anger man genom att sätta `direction` till `in` eller `out` varvid man också definierar positiv riktning. Nollsummering innebär nollsummering med hänsyn tagen till riktning. Man kan därför för varje komponent bestämma vilken riktning man anser vara positiv. Man behöver alltså inte ha globala konventioner som att alla strömmar in i en komponent är positiva. Hopkoppling av två likhetsterminaler T1 och T2, betyder i allmänhet bara $T1 = T2$. Vill vi också specificera kausalitet är det möjligt genom att sätta attributet `causality` lämpligt. Värdet `Read` anger att en koppling på utsidan skall definiera värden, medan `Write` anger motsatsen. I det allmänna fallet behöver man inte ange kausalitet, men om till exempel en koppling representerar en mätning kan det vara lämpligt att ange kausalitet. OmSim kontrollerar då att den givna kausaliteten är konsistent med de andra ekvationerna och kopplingarna. Om så inte är fallet ges felmeddelande. Attributet `unit` anger enheten på värdet som terminalen har. Attributet `quantity` innehåller namnet på storheten som terminalen representerar enligt den internationella standarden ISO 31. När man kopplar ihop två terminaler görs det automatiskt en kontroll att storheterna hos de bägge terminalerna är lika. På detta sätt får man automatisk kontroll av att en spänningsterminal bara kopplas ihop med en spänningsterminal. Med hjälp av attributet `Variability` kan vi definiera terminalen som en parameter eller konstant.

Terminalerna `VoltageTerminal` och `InCurrentTerminal` är bägge underklasser till `SimpleTerminal`. Till exempel

```
VoltageTerminal ISA SimpleTerminal WITH
    quantity := "electric.voltage";
    direction := 'Across;
END;

InCurrentTerminal ISA SimpleTerminal WITH
    quantity := "electric.current";
    direction := 'In;
END;
```

Globala variabler och per-unit systemet

När man bygger modeller av kraftsystem är det mer bekvämt att uttrycka impedanser, strömmar, spänningar och effekt i per-unit (p.u), än i SI-enheterna. Fördelarna med att skala ner parametrar och variabler är att dessa värden kommer att ligga nära varandra numeriskt, detta gör det lättare för den numeriska lösaren, det är också lättare att söka efter fel.

Terminalernas spänningar och strömmar ges i globalt p.u-system. Spännings och strömkomponenter i `In_El_Term` ges i globalt p.u. system. Vi definierar basen för detta som globala konstanter `Sbase`, `Vbase`, och `Ibase` i biblioteket `Basic_Parts`, där vi också definierar en global konstant `w0` för den synkrona vinkelfrekvensen. Omlokoden för detta bibliotek finns i Appendix A1.

3. Komponentmodeller

I detta kapitel skall vi definiera baskomponenter för linjer och laster, där en resistor seriekopplad med en induktor används för att beskriva beteendet. Bussen kommer att definieras. En matematisk modell av synkrogeneratorn kommer att presenteras, liksom dess strömmar och spänningar. Val av referenssystem för att beskriva fasvinklar behandlas, samt enkel reglering av generatorerna. Grafisk representation av delmodellerna tas också upp. Den fullständiga Omolakoden för de olika delmodellerna finns i Appendix A.

3.1 Resistor och induktor i serie

Många elektriska komponenter såsom motstånd, spolar och ledningar har två naturliga kopplingspunkter. Vi definierar därför först en basklass, `TwoPort`, med två terminaler

```
TwoPort ISA Base::Model WITH
  attributes:
    T1 ISA In_El_Term
    T2 ISA In_El_Term
END;
```

`TwoPort` är i princip bara ett skal som består av två terminaler `T1` och `T2`. Detta skal skall vi nu förse med de parametrar, variabler och ekvationer som krävs för att definiera linjen och lasten. Vi skapar därför klassen `RLImpedance` som en underklass till `TwoPort`

```
RLImpedance ISA TwoPort WITH
  parameters:
    R TYPE Discrete Real;
    Xl TYPE Discrete Real;
  variable:
    L TYPE Real;
  equations:
    L = Xl/Basic_parts::W0;
    T1.Id = - T2.Id;
    T1.Iq = - T2.Iq;
    T1.Vd - T2.Vd = R*T1.Id - Xl*T1.Iq + ::alpha*L*dot(T1.Id);
    T1.Vq - T2.Vq = R*T1.Iq + Xl*T1.Id + ::alpha*L*dot(T1.Iq);
END;
```

Variablerna `R` och `Xl` representerar här resistansen och reaktansen hos komponenten. De är här deklarerade som `Discrete` för att markera att de inte kan vara tidskontinuerliga utan kan bara ändra värde vid diskreta tidpunkter. När vi nedan använder denna komponent för att modellera ledningar kommer vi att deklarera om dem till parametrar som kan sättas av användaren. När vi använder komponenten i lastmodellen kommer vi att ge uttryck för `R` och `Xl` i andra parametrar som anger hur mycket effekt lasten konsumerar. Första ekvationen efter `equations` ger en relation mellan reaktansen `Xl` och induktansen `L`. De två efterföljande ekvationerna säger att det flyter samma ström i båda ändar. Minustecknet beror på att positiv riktning är in i båda ändarna.



Figur 3.1 Modell för RLImpedance

De sista ekvationerna beskriver relationen mellan strömmen och spänningsfallet över impedansen. De härledes på följande sätt: Betrakta fasen a . Strömmen i_a från T1 och T2 ges av

$$i_a = i_{d,1} \sin \theta + i_{q,1} \cos \theta$$

Spänningen $v_{a,1}$ vid terminal T1 är

$$v_{a,1} = v_{d,1} \sin \theta + v_{q,1} \cos \theta$$

och $v_{a,2}$ ges analogt. Spänningsfallet över komponenten ges nu av

$$v_{a,1} - v_{a,2} = Ri_a + Ldi_a/dt$$

Om vi nu uttrycker spänningarna $v_{a,1}$ och $v_{a,2}$ samt strömmen i_a i sina d- och q-komponenter fås

$$\begin{aligned} (v_{d,1} - v_{d,2}) \sin \theta + (v_{q,1} - v_{q,2}) \cos \theta = \\ Ld(i_{d,1} \sin \theta + i_{q,1} \cos \theta)/dt + R(i_{d,1} \sin \theta + i_{q,1} \cos \theta) = \\ Ldi_{d,1} \sin \theta/dt + Ldi_{q,1} \cos \theta/dt + Ri_{d,1} \sin \theta + Ri_{q,1} \cos \theta = \\ Ldi_{d,1}/dt \sin \theta + \dot{\theta}Li_{d,1} \cos \theta + Ldi_{q,1}/dt \cos \theta \\ - \dot{\theta}Li_{q,1} \sin \theta + Ri_{d,1} \sin \theta + Ri_{q,1} \cos \theta \end{aligned}$$

Eftersom detta uttryck skall gälla för alla θ , fås

$$v_{d,1} - v_{d,2} = Ri_{d,1} + Ldi_{d,1}/dt - \dot{\theta}Li_{q,1}$$

och

$$v_{q,1} - v_{q,2} = Ri_{q,1} + Ldi_{q,1}/dt + \dot{\theta}Li_{d,1}$$

En vanlig approximation är att man utesluter derivatatermen i ekvationerna ovan och sätter $\dot{\theta} \approx \omega_0$. Om vi inkluderar derivatatermerna i modellen kommer stegsvar att orsaka kraftigt dämpade högfrekventa svängningar, som inte är så viktiga. De gör att modellen bli tidsödande att simulera.

Uttrycket α betecknar en global variabel α , som är antingen 0 eller 1. Om ingenting annat sägs kommer α att vara 0. De två understa ekvationerna betyder att strömmen in i terminal T1, är lika med strömmen in i terminal T2, men med motsatt tecken. Se figur 3.1. Terminalerna är markerade som fyllda rektanglar.

3.2 Transmissionslinjen

Transmissionslinjen överför effekt mellan två kopplingspunkter. Eftersom vi antar att våra spänningar och strömmar är sinusformade kan vi klara oss med en enkel modell. Sett i frekvensplanet behöver vi en modell som är bra för

ω_0 . Skall man ha en modell som är bra för alla frekvenser måste man explicit använda Maxwells ekvationer, som ju är partialekvationer. Vi modellerar en transmissionsledning med en resistor och en induktor i serie. Detta är en vanlig ansats. Vi antar att man genom att direkt mäta upp eller genom att använda Maxwells ekvationer har fått fram värden för R och X_L .

Vi kan nu direkt använda `RLImpedance` för att modellera en transmissionslinje. Dock föredrar vi att med hjälp av `RLImpedance` specificera en ny underklass `Line0`. Detta ger oss en möjlighet att lägga till nya attribut som är av intresse att beräkna vid simulering

```
Line0 ISA RLImpedance WITH
parameters:
  R ISA Parameter;
  Xl ISA Parameter;
variables:
  Ploss, Qloss, Pflow1, Qflow1, Pflow2, Qflow2 TYPE Real;
equations:
  Ploss = R*(sqr(T1.Id)+sqr(T1.Iq))*basic_parts::sbase/1e6;
  Qloss = Xl*(sqr(T1.Id)+sqr(T1.Iq))*basic_parts::sbase/1e6;
  Pflow1 = (T1.Vd*T1.Id+T1.Vq*T1.Iq)*basic_parts::sbase/1e6;
  Qflow1 = (T1.Vq*T1.Id-T1.Vd*T1.Iq)*basic_parts::sbase/1e6;
  Pflow2 = (T2.Vd*T2.Id+T2.Vq*T2.Iq)*basic_parts::sbase/1e6;
  Qflow2 = (T2.Vq*T2.Id-T2.Vd*T2.Iq)*basic_parts::sbase/1e6;
END;
```

Variablerna `Ploss` och `Qloss` är förlusterna i linjen, medan `Pflow1` och `Qflow1` beskriver den effekt som går in i `T1`, och `Pflow2` och `Qflow2` är motsvarande för `T2`. Effekterna är uttryckta i MW och MVA.

3.3 Bussen

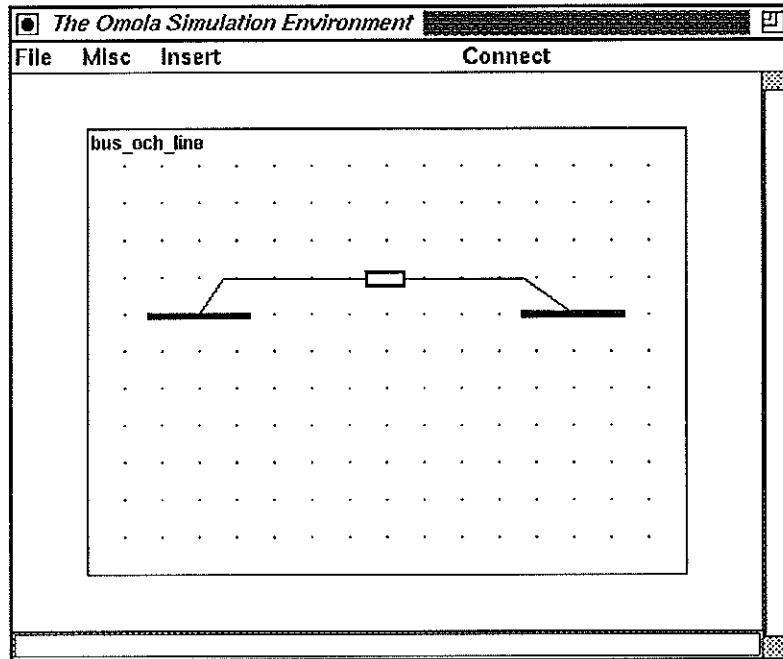
Bussen som tillsammans med linjen utgör nätverket representerar noden där, generatoren, lasten och linjen kopplas samman. Omolakoden för `Bus` är

```
Bus ISA In_El_Term WITH
equations:
  Id = 0;
  Iq = 0;
END;
```

Den enda ekvationen som krävs är, enligt Kirchhoffs strömlag, den ekvation som säger att summan av alla strömmarna in i en `Bus` är noll. I figur 3.2 visas en bild på två bussar ihopkopplade med en linje.

3.4 Lasten

Den tredje och sista passiva komponenten i vårt kraftsystem är lasten. Lasten kan till exempel representera ett antal kommuner som förbrukar en bestämd mängd aktiv- och reaktiv effekt, P och Q . Det är en svår konst att modellera aggregerade laster. Hur de ser ut beror på blandning av belysning, elvärme, elmotorer m.m. Det är inte vår avsikt att gå på djupet här, utan vi modellerar



Figur 3.2 Två bussar och en linje hopkopplade

bara lasterna som enkla RL-länkar. Vi skulle kunna använda `RLImpedance` direkt men föredrar även här att göra en modifiering och skapar en ny klass `Ind_Load0`

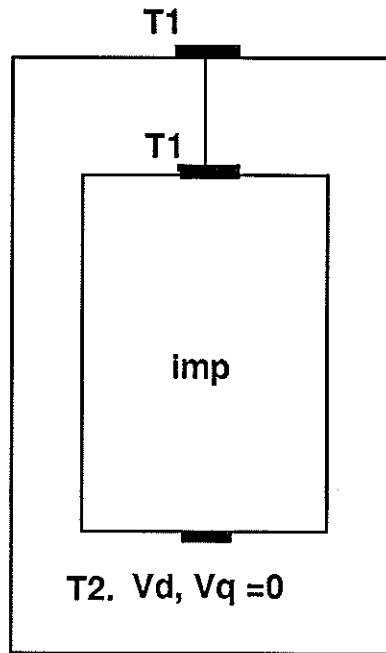
```

Ind_Load0 ISA OnePort WITH
  parameters:
    POSI ISA Parameter;
    QOSI ISA Parameter;
  submodels:
    imp ISA RLImpedance;
  variables:
    P0, Q0 TYPE Discrete Real;
    Ploss, Qloss TYPE Real;
  equations:
    imp.T2.Vd = 0;
    imp.T2.Vq = 0;
    P0 = POSI/basic_parts::Sbase;
    Q0 = QOSI/basic_parts::Sbase;
    Ploss = imp.R*(sqr(T1.Id) + sqr(T1.Iq))
             *basic_parts::sbase/1e6;
    Qloss = imp.Xl*(sqr(T1.Id) + sqr(T1.Iq))
             *basic_parts::sbase/1e6;
  connections:
    T1 AT imp.T1;
END;

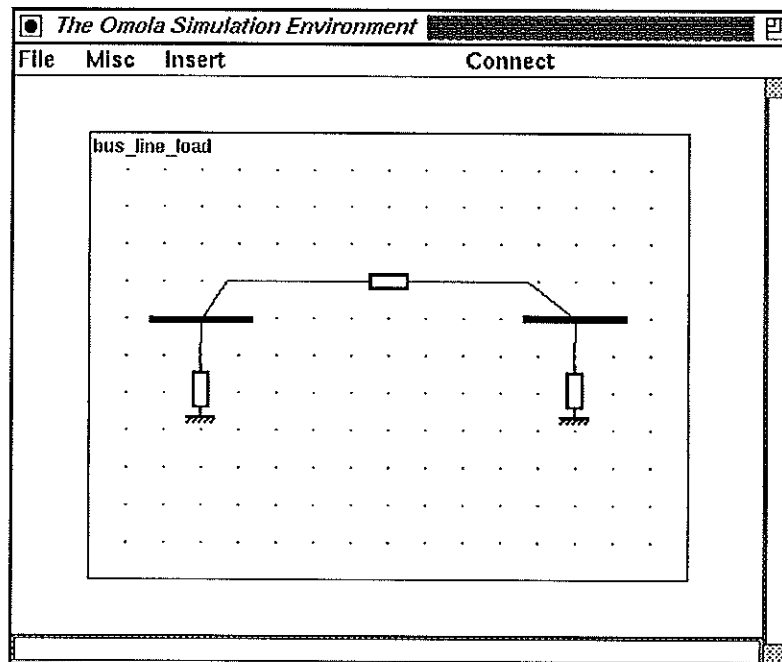
```

Till skillnad från linjen som är en `TwoPort`, är lasten en `OnePort`. Skillnaden är att `OnePort` endast består av en terminal `T1`.

För att få ett bekvämt sätt att definiera hur mycket effekt lasten skall konsumera, har vi infört parametrarna `POSI` och `QOSI`, som i SI-enheterna `W` och `VAR` definierar hur mycket aktiv och reaktiv effekt som konsumeras då lasten

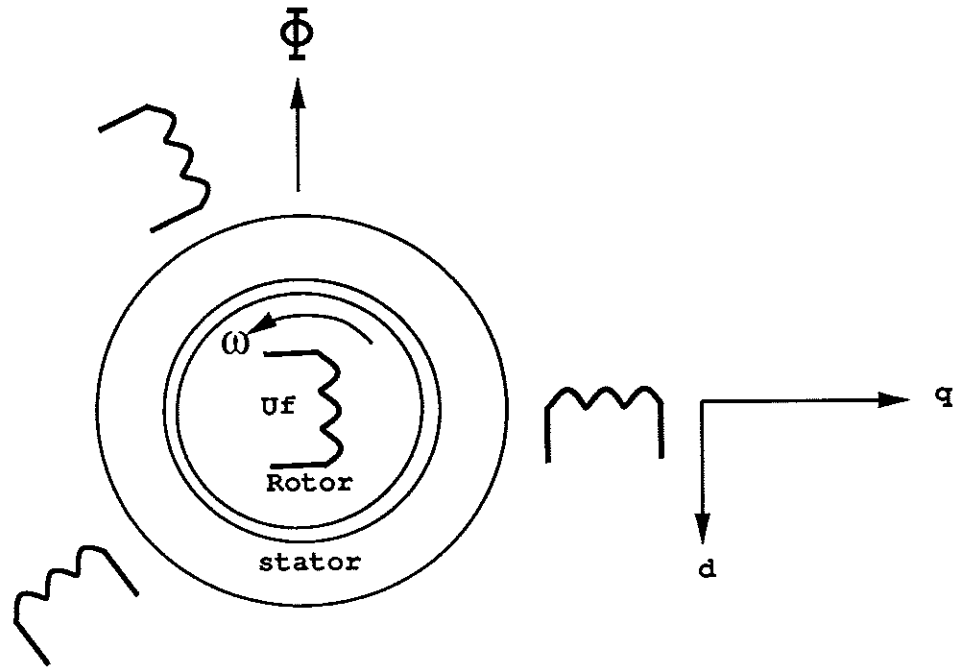


Figur 3.3 Ind_Load0 innehållande imp. Terminalen T2 är jordad



Figur 3.4 Laster hopkopplade med bussar och en linje

kopplas till en spänning som är 1 p.u. Liksom TwoPort är OnePort bara ett skal och vi måste därför introducera en delmodell imp, som har RLImpedances egenskaper. Då lasten bara består av en terminal, T1, jordas T2. Ploss och Qloss beskriver förlusten i lasten i MW och MVar. För att koppla samman de bägge modellerna ansluter vi terminalen T1 på Ind_Load0 med terminalen T1 på imp. Se figur 3.3. I figur 3.4 ser vi hur laster, linjer och bussar kan kopplas ihop. Linjen och lasten finns beskriven i Appendix A2.



Figur 3.5 Principskiss på en trefas synkrongenerator

3.5 Generatoren

I de flesta kraftverk för storskalig elenergiproduktion används en trefas synkrongenerator. Dessa generatorer är ofta långsamt-roterande mångpoliga maskiner. I vissa fall, t.ex gasturbindrivna generatorer, är maskinerna snabbgående med lågt poltal och cylindrisk rotor, s.k "turbo"-maskiner.

Varje synkrongenerator ser principiellt ut som figur 3.5. Den roterande delen kallas rotor och består av ett magnetiskt material och en fältlindning. När fältlindningen ansluts till en spänning u_f , kommer strömmen genom lindningen att ge upphov till ett magnetiskt flöde ϕ , som roterar med rotorns hastighet ω . Detta roterande flöde kommer att inducera en växelspanning i var och en av dom tre statorlindningarna, och vi har en trefasgenerator. När en last ansluts till generatoren, kommer ström att flyta genom statorlindningarna. Denna ström ger upphov till ett flöde som kommer att interferera med flödet från fältlindningen. Det finns synkrongeneratormodeller av olika komplexitet. Vi skall använda en enkel variant som modellerar rotorns rörelse och dynamiken hos fältlindningen.

Svängningsekvationen

Rotorns rörelse beskrivs med svängningsekvationen

$$\frac{2H}{\omega_0} \frac{d^2\theta}{dt^2} + \frac{D}{\omega_0} \frac{d\theta}{dt} = P_m - P_e$$

där vinkeln θ beskriver rotorns elektriska position. Variablerna P_m och P_e är den mekaniska respektive elektriska effekten.

Tröghetskonstanten H representerar rotorns tröghetsmoment och definieras som rotorns kinetiska energi i p.u. vid synkront varvtal. D är en dämpterm som beror på dämplindningen i rotorn.

Elektriska storheter

Statorns emk E'_q längs q-axeln är proportionell mot flödet från fältlindningen, och ges av

$$T'_{d0} \frac{dE'_q}{dt} = U_f - E'_q - (X_d - X'_d)I_d$$

Den genererade aktiva effekten kan skrivas

$$P_e = I_d I_q (X_q - X'_d) + I_q E'_q$$

Motsvarande reaktiva effekt är

$$Q_e = I_d V_d + I_q V_q$$

Generatorns terminalspänning uttryckt i d-q-komponenterna blir

$$V_d = X_q I_q$$

$$V_q = E'_q - X'_d I_d$$

Generatorns terminal-spänningar och -strömmar

En generatormodell behöver en elektrisk terminal så att den kan anslutas till en bus. Spänningarna och strömmarna hos denna terminal måste vi uttrycka i det globala referenssystemet. I generatorn används ett lokalt referenssystem och ett eget per-unit system. Generatorns lokala referenssystem är kopplat till rotorns rörelse, så att q-axeln är ortogonal mot flödet och d-axeln är orienterad mot samma flöde. Detta betyder att d och q-axeln i detta system ligger vinkeln $\delta = \theta - \theta_{ref}$ före det globala systemet. Antag att vinkeln i det globala systemet är θ_{ref} . Spänningen på fas a, v_a är då

$$\begin{aligned} v_a &= v_{d,l} \sin(\theta_{ref} + \delta) + v_{q,l} \cos(\theta_{ref} + \delta) \\ &= v_{d,l} (\cos \delta \sin \theta_{ref} + \sin \delta \cos \theta_{ref}) \\ &\quad + v_{q,l} (-\sin \delta \sin \theta_{ref} + \cos \delta \cos \theta_{ref}) \\ &= (v_{d,l} \cos \delta - v_{q,l} \sin \delta) \sin \theta_{ref} \\ &\quad + (v_{d,l} \sin \delta + v_{q,l} \cos \delta) \cos \theta_{ref} \end{aligned}$$

dvs.

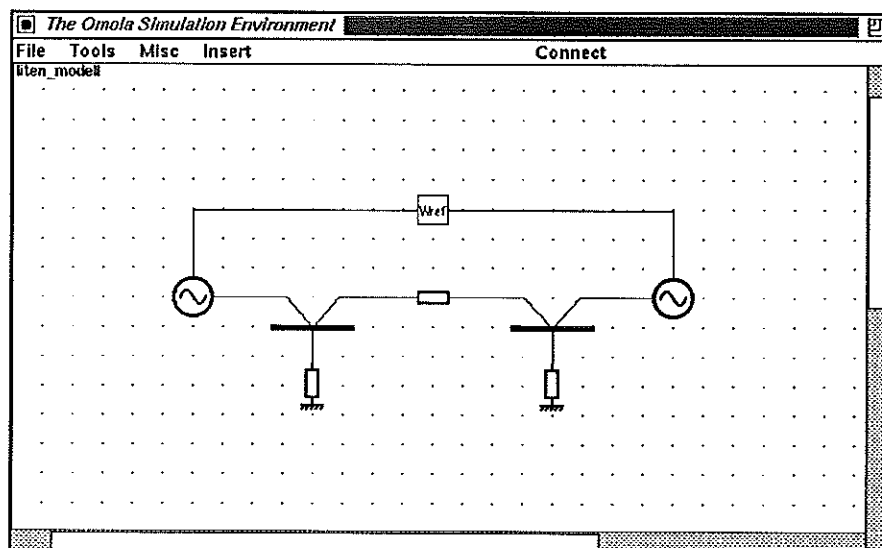
$$\begin{aligned} v_{d,l} &= v_{d,l} \cos \delta - v_{q,l} \sin \delta \\ v_{q,l} &= v_{d,l} \sin \delta + v_{q,l} \cos \delta \end{aligned}$$

Om vi nu också tar hänsyn till att olika p.u. - system används får vi

```
T1.Vd = Vn/Basic_Parts::Vbase*(Vd*cos(Delta) - Vq*sin(Delta));
T1.Vq = Vn/Basic_Parts::Vbase*(Vd*sin(Delta) + Vq*cos(Delta));
T1.Id = In/Basic_Parts::Ibase*(Id*cos(Delta) - Iq*sin(Delta));
T1.Iq = In/Basic_Parts::Ibase*(Id*sin(Delta) + Iq*cos(Delta));
```

Representation av vinklar

I generatormodellen har vi använt två vinklar θ och δ , där θ är rotorns position och $\delta = \theta - \theta_{ref}$ är fasen i förhållande till nätet. Att beräkna δ som $\delta = \theta - \theta_{ref}$ orsakar numeriska problem. Då θ och θ_{ref} är av samma storlek medför



Figur 3.6 Out-Mass-Term från två generatorer hopkopplade i Wref

detta att antalet signifikanta siffror minskar med ökande simuleringstid. Ett bättre sätt är att beräkna δ ur $\dot{\delta} = \omega - \omega_{ref}$ där $\omega = \dot{\theta}$. Vi skriver alltså om svängningsekvationen som

$$\frac{2H}{\omega_0} \frac{d\omega}{dt} + \frac{D}{\omega_0} \omega = P_m - P_e$$

$$\dot{\delta} = \omega - \omega_{ref}$$

Nästa problem är hur man skall definiera ω_{ref} . Det är inte lämpligt att välja $\omega_{ref} = \omega_0$, eftersom generatorerna i kraftsystem inte roterar med exakt ω_0 . Ett vanligt sätt är, att man väljer referenspunkten så att den ligger i medelpunkten av generatorernas rörelseenergi dvs.

$$\omega_{ref} = \frac{\sum H_i \omega_i}{\sum H_i}$$

Här är H_i tröghetskonstanten. För att kunna göra denna uträkning definierar vi först en record terminal Out_Mass_Term

```
Out_Mass_Term ISA RecordTerminal WITH
Components:
  L ISA Out_Angular_MomentumTerminal;
  H ISA Out_Moment_of_InertiaTerminal;
  W ISA AngularTerminal;
END;
```

där $L = H \cdot \dot{\omega}_i$. Dessa Out_Mass_Term från varje generator kopplas nu samman i en nod Wref, Se figur 3.6, som är av klassen InertiaBus. Inertiabus blir i Omolakod

```
Inertiabus ISA Out_Mass_Term WITH
equations:
  W = L/H;
END;
```

där L i Out_Angular_MomentumTerminal har direction := out. Detta out har den trevliga egenskapen att summan av alla L i noden Wref kommer att

bli noll. Man kan jämföra detta med att summan av strömmarna in i en nod skall vara noll. Samma sak gäller för H i Out_Moment_of_InertiaTerminal.

Följande två ekvationer erhålls för exempelvis tre generatorer

$$\begin{aligned}\omega_{ref}.L + L_1 + L_2 + L_3 &= 0 \\ \omega_{ref}.H + H_1 + H_2 + H_3 &= 0\end{aligned}$$

Efter omskrivning erhålls

$$\begin{aligned}\omega_{ref}.L &= -\sum_{i=1}^3 H_i \omega_i \\ \omega_{ref}.H &= -\sum_{i=1}^3 H_i\end{aligned}$$

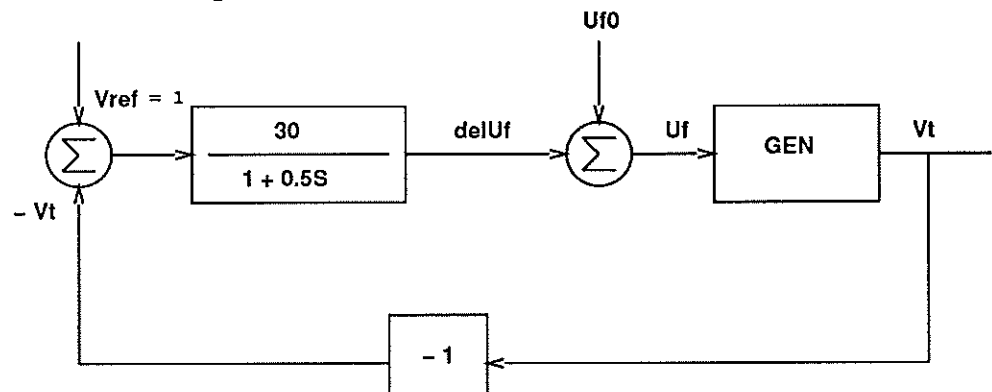
och slutligen

$$\omega_{ref}.W = \frac{\sum_{i=1}^3 H_i \omega_i}{\sum_{i=1}^3 H_i}$$

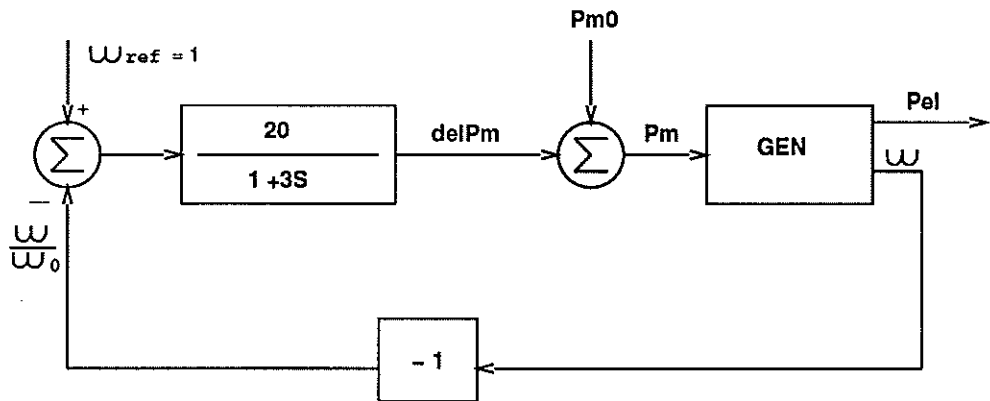
Spännings och effektregulatorer

Varje generator har två insignaler som man kan ändra, samt två utsignaler som det ställs vissa krav på. Insignaler är fältspänningen U_f och den mekaniska effekten P_m , utsignaler är terminalspänningen V_t och den elektriska effekten P_e . Det normala sättet att upprätthålla konstant V_t och P_e är att låta U_f reglera V_t och P_m reglera P_e . Detta har gjorts med två enkla regulatorer.

Figurerna 3.7 och 3.8 visar hur spänningsregulatorn och effektregulatorn är anslutna till generatoren



Figur 3.7 Spänningsregulatorns återkopplingslinga



Figur 3.8 Effektregulatorns återkopplingslinga

3.6 Grafisk representation av delmodellerna

För att kunna definiera nätmodeller grafiskt som i figur 3.6 behöver vi specificera vilka grafiska symboler vi önskar för de olika komponenterna. I Omola kan modellkomponenter ha grafiska attribut. Om vi inte explicit definierar dessa, får vi en rektangel för delmodellerna och små fyllda rektanglar för terminaler.

Om man vill definiera egna symboler eller ikoner använder man först en speciell grafikeditor, bitmap, som finns i X-windows, för att göra en bitmap som man lagrar i en fil.

Antag nu att vi gjort en bitmap som vi lagrat i en fil med namnet "bus". Om vi nu vill använda denna symbol för komponenten Bus gör vi ett tillägg så att den fullständiga modellen för Bus blir

```

Bus ISA In_EL_Term WITH
equations:
  Id = 0;
  Iq = 0;
Graphic ISA Base::Layout WITH
  bitmap TYPE String := "bus";
END;
END;

```

På liknande sätt har ikoner gjorts för generatoren, lasten och linjen. Ikoner har även gjorts för de enskilda simuleringsfallen. Till exempel så har en ikon i form av en strömbrytare gjorts vid simulering av ett avbrott i en ledning.

Förutom att definiera ikoner för olika komponenter är det också av intresse att grafiskt definiera var kopplingspunkterna hos terminalerna skall sitta. Detta gör man lättast genom att använda OmSims grafikeditor. Även denna grafikinformation har en textmässig representation. Till exempel är den fullständiga Omolakoden för TwoPort

```

TwoPort ISA Base::Model WITH
attributes:
  T1 ISA In_EL_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
attributes:
  x_pos := 200

```

```

        Y_pos := 150
        invisible := 1;
    END;
END;
T2 ISA In_El_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 200
        Y_pos := 150
        invisible := 1;
    END;
END;
END;

```

Här beskriver x- och y-positionerna var på TwoPort som de båda terminalerna sitter. De har också gjorts osynliga i den grafiska representationen genom `invisible := 1`.

Vi vill än en gång påpeka att koden ovan för Bus inte är något som användaren normalt skriver in, utan man definierar terminalens position med en grafikeditor, och OmSim genererar sedan automatiskt den textmässiga representationen för att man skall kunna spara modellen på en fil. Det finns också speciella filter som eliminerar alla grafiska attribut så att man kan få en trevligare utskrift av modellen.

4. Initialvärden och störningar

4.1 Initialvärden

Vid simulering av ett stort kraftnät måste alltid vissa av generatorns variabler ges initialvärden, även linjen och lasten skall ges initialvärden. Systemet skulle annars bli instabilt och en fullständig implementering av reglersystemet och uppstartsprocessen skulle behövas. Lösning av systemets algebraiska ekvationer underlättas också med hjälp av initialisering, vilket kortar ner simuleringstiden.

Initialvärdet för systemet kan definieras av spänningen på bussarna och effektproduktionen hos generatorerna under antagande om stationaritet. Bussarnas spänning skulle kunna ges som V_d och V_q , men vi har istället valt att uttrycka dem i belopp och fas eftersom man oftast vill att bussspänningen skall ha beloppet 1 p.u. Se figur 4.1.

Initialiseringen för linjen är lagd i en klass som kallas Line och har Omolakoden

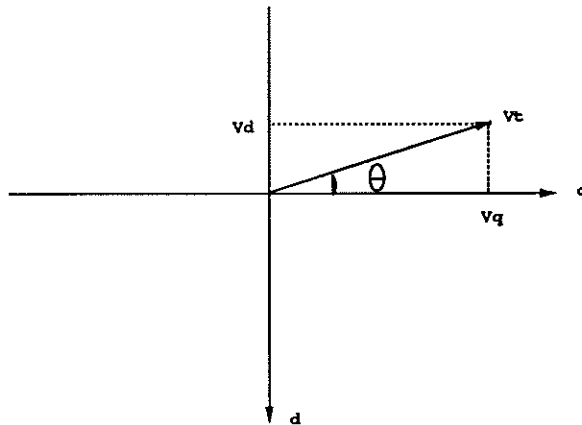
```
Line ISA Line0 WITH
parameters:
  Theta01 ISA Parameter;
  Theta02 ISA Parameter;
  Vt01    ISA Parameter WITH default := 1; END;
  Vt02    ISA Parameter WITH default := 1; END;
Init ISAN Event;
OnEvent Init Do
  new(T1.Vd) := -Vt01*sin(Theta01*Basic_Parts::Pi/180);
  new(T1.Vq) :=  Vt01*cos(Theta01*Basic_Parts::Pi/180);
  new(T2.Vd) := -Vt02*sin(Theta02*Basic_Parts::Pi/180);
  new(T2.Vq) :=  Vt02*cos(Theta01*Basic_Parts::Pi/180);
END;
```

Ovan har initialiseringen deklarerats som en händelse som sker innan simuleringen börjar. Detta görs med raderna Init ISAN Event. OnEvent talar om vad som kommer att hända i händelsen. New operatoren framför T1.Vd beskriver att T1.Vd kommer att få värdet efter likhetstecknet precis innan simuleringens början. På liknande sätt initieras lasten.

För att generatoren i stationaritet skall ligga rätt initialiseras strömmarna I_d och I_q , men även E_q , vinkeln δ och att $\omega = \omega_0$. Startvärden för dessa variabler är tagna ur Akke (1989). Eftersom alla derivator skall vara noll vid $t = 0$, initialiseras också U_f och P_m . En del andra variabler förutom de ovan nämnda har också initialiserats, detta för att påskynda simuleringen.

4.2 Modellering av störningar

Låt oss diskutera hur vi skall modellera en störning som urkoppling av linje. En urkoppling kan modelleras av en brytare som kopplas mellan en transmis-



Figur 4.1 V_d och V_q i d-q-systemet vid $t=0$

sionslinje och en bus. Om vi vill ha avbrott från $t = 1$ skulle vi kunna skriva ekvationerna för brytaren

```
if (Base::Time > 1) then T1.Id else (T1.Vd - T2.Vd) = 0;
if (Base::Time > 1) then T1.Iq else (T1.Vq - T2.Vq) = 0;
```

om vi antar att brytarens terminal som skall kopplas till bussen heter T1. Om $t > 1$ har vi $T1.Id = 0$ och $T1.Iq = 0$ dvs brytaren är öppen. För $t < 1$ har vi $(T1.Vd - T2.Vd) = 0$ och $(T1.Vq - T2.Vq) = 0$ dvs brytaren är sluten. Nackdelen med denna ansats är att modellen är diskontinuerlig vid $t = 1$. Det betyder att den numeriska lösaren får lägga ner mycket arbete på att finna ut var diskontinuiteten finns. Det gör den genom att justera steglängden och den kan då tvingas ner till mycket små steglängder. Vi kan underlätta lösarens arbete genom att definiera urkopplingen som en diskret händelse.

Interrupt ISA TwoPort WITH

Parameter:

FaultTime ISA Parameter WITH default := 1; END;

FaultDuration ISA Parameter WITH default := 100; END;

variables:

fault TYPE DISCRETE integer;

Init, Break, BreakAction ISAN Event;

OnEvent Init DO

schedule(Break, FaultTime);

new(fault) := 0;

END;

OnEvent Break DO

schedule(BreakAction, FaultDuration);

new(fault) := 1;

END;

OnEvent BreakAction DO

new(fault) := 0;

END;

equations:

if (fault == 1) then T1.Id else (T1.Vd - T2.Vd) = 0;

if (fault == 1) then T1.Iq else (T1.Vq - T2.Vq) = 0;

```
T1.Id = - T2.Id;  
T1.Iq = - T2.Iq;  
END;
```

Vid start av simuleringen triggas händelsen `Init` som leder till att satsen `schedule(Break, FaultTime)` utförs. Det innebär att vi har talat om att det sker en händelse vid tiden `FaultTime`. Den numeriska lösaren kommer då inte att försöka simulera förbi `FaultTime`, utan den försöker att simulera precis till `FaultTime`.

Variabler som bara skall ändra värde i samband med händelser men som i övrigt skall vara konstanta deklarerar inte på vanligt sätt. I stället för att deklarerera `fault` i `Interrupt` ovan som `fault ISA variable`, skriver vi `fault TYPE DISCRETE integer`. Nu kommer variabeln `fault` bara att ändra värde i samband med diskreta händelser. När `Init` triggas kommer variabeln `fault` att få värdet noll, `fault` kommer nu att vara noll under tiden fram till `FaultTime`, en sekund ovan. Tittar vi i dom två raderna under `equations` ser vi att då `fault` har värdet noll, kommer spänningsfallet över brytaren att vara noll; den är sluten. Efter `FaultTime` inträffar `Break` och nu får `fault` värdet ett; brytaren är öppen. Tittar vi i ekvationerna igen kommer nu `T1.Id` och `T1.Iq` att vara noll. Detta tillstånd kommer nu att råda under `Faultduration` fram till händelsen `Breakaction`. `Fault` får nu värdet noll igen och brytaren är åter sluten. Liknande händelser används vid simulering av steg i last.

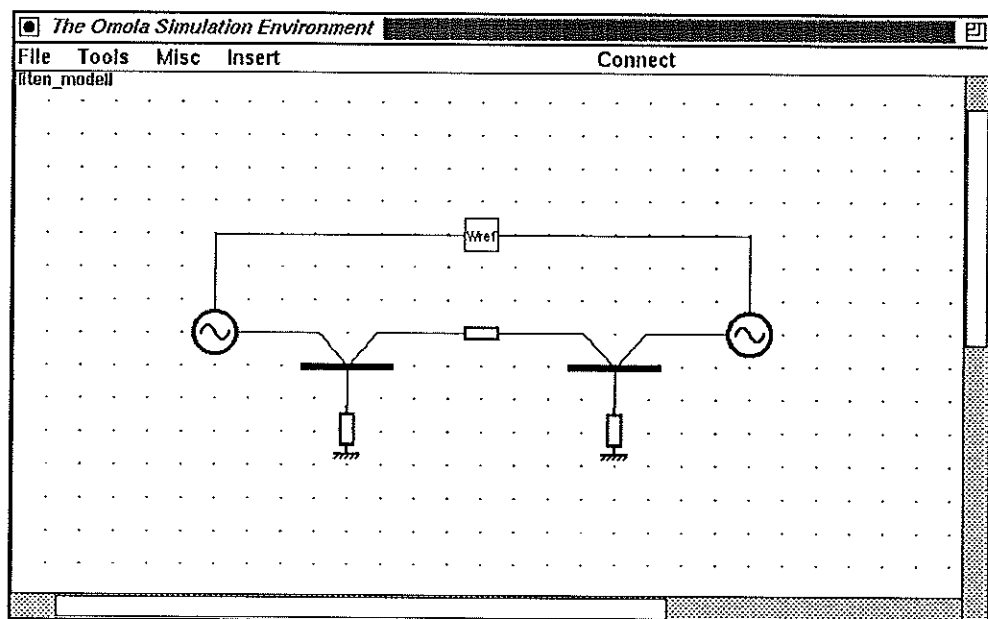
5. Modellering och simulering i OmSim

Alla komponenter i kraftnätet är nu definierade och i detta kapitel skall vi studera hur man skapar en modell enligt figur 5.1, samt handhavandet vid simulering av denna modell.

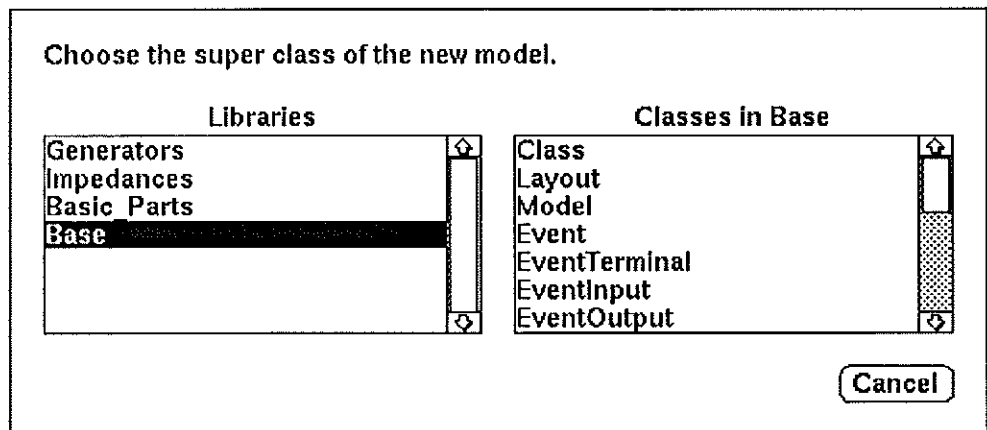
5.1 Definition av modell

De definierade klasserna finns nu lagrade i olika bibliotek beroende på vilken typ av klass de tillhör. Till exempel är alla terminaler definierade i ett bibliotek som kallas `basic_parts`, `Line` och `Ind_Load` i ett bibliotek som kallas `Impedances`, generatoren i `Generators` osv.

Vill vi nu bygga en nätmodell måste vi ladda in dessa bibliotek, och vi måste göra det i nämnd ordning eftersom de beror på varandra. Man kan ladda in modeller på två sätt; genom att räkna upp filerna efter startkommandot `OmSim` eller genom kommando inifrån `OmSim`. När man startar `OmSim` får man upp ett fönster (se figur 5.1) med rullgardinsmenyerna `File`, `Tools`, `Misc`, `Insert` och `Connect`. Fönstret används för grafisk editering. Idag innehåller `OmSim` ingen texteditor, utan sådan editering måste göras utanför `OmSim` med en vanlig texteditor, och man måste sedan ladda in filen genom att använda `Load`-kommandot i `File`-menyn. Om vi vill se vilka bibliotek som laddats in kan vi klicka på `File` och därefter på `New`. Nu kommer det upp ett fönster som innehåller de bibliotek vilka laddades in, plus ett extra fördefinierat som heter `Base`, se figur 5.2. Till höger om biblioteken kan vi se vilka klasser som ligger i respektive bibliotek. Efter att ha klickat på `Model` i biblioteket `Base` får vi upp ett fönster i vilket de komponenter som skall ingå i modellen placeras.



Figur 5.1 En liten modell modellerad i Omola



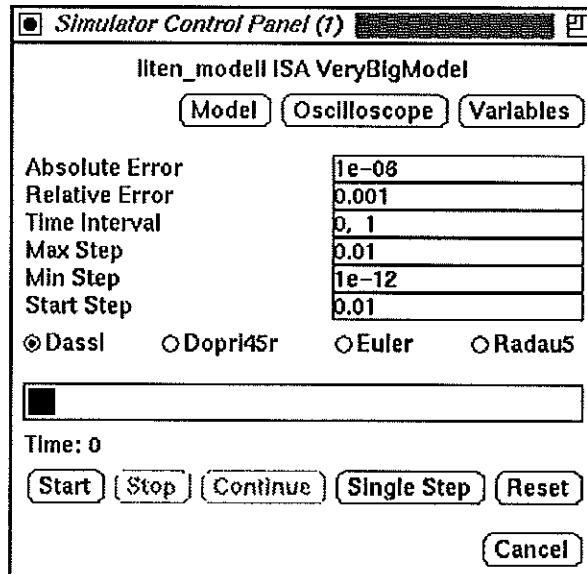
Figur 5.2 Bibliotek och klasser i Omola

Genom att klicka på **Insert** och på biblioteket **Generators**, kommer vi åt de bägge generatorerna som behövs till vår modell. Efter att ha placerat ut dessa klickar vi på biblioteket **Impedances** och kommer nu åt de båda lasterna **Ind_Load** samt linjen **Line**, dessa placeras också ut enligt figur 5.1. För att komma åt **Inertiabus** klickar vi på biblioteket **Basic_Parts** och **Inertiabus**, som också placeras ut. När alla komponenterna är utplacerade ger vi dem ett namn, till exempel **Generator1**, **IndLoad1** och **Line12**, för att markera att linjen ligger mellan **Bus1** och **Bus2**. Därefter kopplas modellen samman genom att vi klickar på **Connect** och därefter på de punkter som skall kopplas ihop. Klickar vi till exempel på **Connect** och därefter på **Generator1** kan vi antingen välja terminal **T1** eller terminal **T2**. Terminal **T1** ansluter vi till **Bus1**, och terminal **T2** till **Wref**, som vi döpt **Inertiabus** till. På liknande sätt förbinds övriga komponenter. När alla förbindelser är gjorda ger vi vår modell ett namn, i detta fallet **liten_modell1**. Därefter sparar vi den genom att klicka på **File** och **Save**. Vi går nu in i texteditorn, till den sparade filen, och lägger in parametrar och variabler på de aktuella komponenterna.

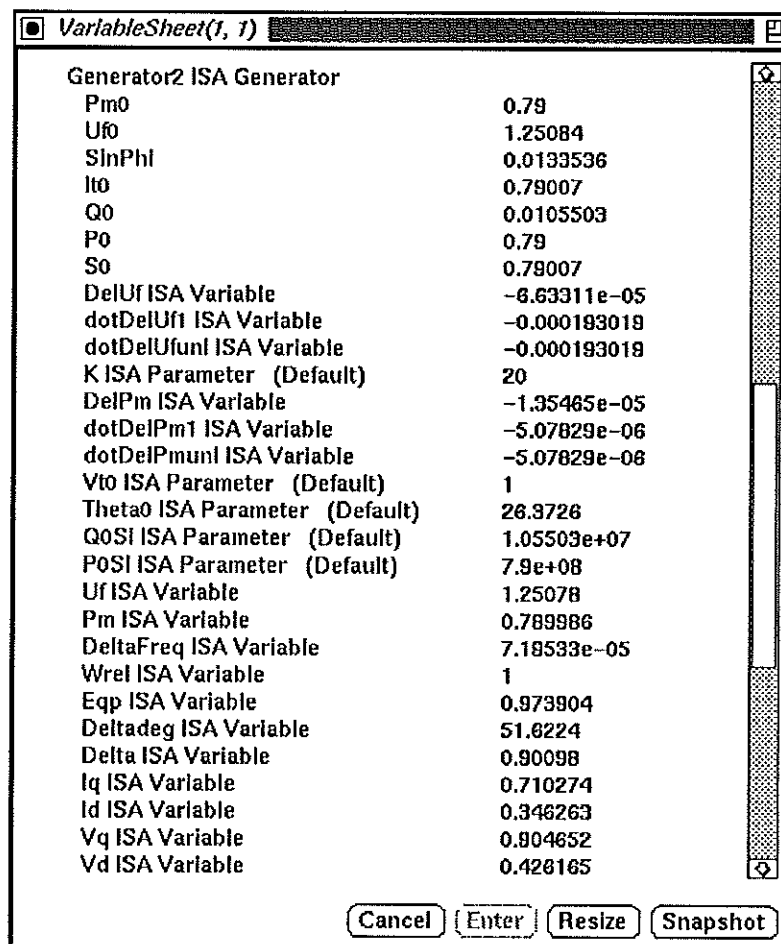
Simulering

För att simulera modellen som vi har gjort, laddar vi ner biblioteken som innan tillsammans med den sparade filen, därefter klickar vi på **Tools** och **Simulate**. Vi får nu upp ett fönster, **Simulator Control Panel**, med bland annat tre knappar överst, se figur 5.3. Genom att klicka på knappen märkt **Model**, får vi återigen upp fönstret med biblioteken. Vi klickar nu på modellen som skall simuleras och nu sker det en kontroll om det är möjligt att koppla ihop modellen som vi har gjort. Det kontrolleras bland annat att terminalerna stämmer dvs. att en spänningsterminal inte är hopkopplad med en strömterminal. Beroende på hur stor modellen är tar detta lite tid. Om allting är rätt hopkopplat kan vi börja simuleringen.

Klickar vi på knappen märkt **Variables** får vi upp ett fönster som kallas **Variable Sheet**, se figur 5.4. Här kan man titta på de variabler som man är intresserad av.

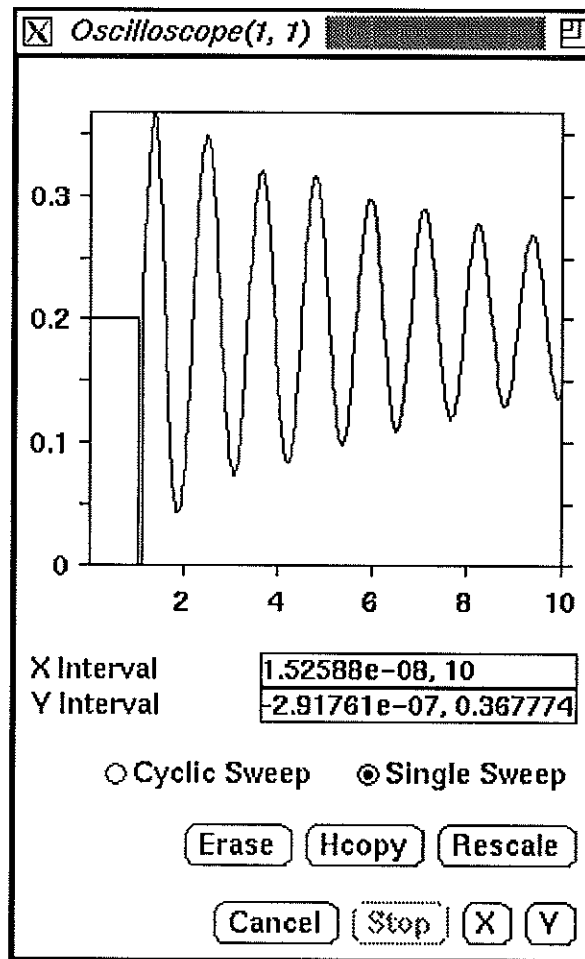


Figur 5.3 Simulatorfönstret med styrknappar och simuleringsparametrar



Figur 5.4 Generators och övriga komponenters variabler

Det går även att plotta variabler i ett Oscilloscope-fönster, vi klickar då på knappen märkt Oscilloscope. Genom att först klicka på den variabel som vi vill titta på i Variable Sheet och sedan på knappen märkt Y i Oscilloscope-fönstret har vi bundit variabeln till Y-axeln. X-axeln är alltid



Figur 5.5 Oscilloskop-fönster

tiden. Möjligheter finns också att lagra variabler på en fil, för att till exempel plotta motsvarande variabler i Matlab. Vi klickar då igen på variabeln och sedan på store. Variabeln lagras nu automatiskt på en fil under simuleringen. Simuleringen startas genom att klicka på knappen märkt Start, och pågår nu i en sekund. Vill vi ändra någon variabel och samtidigt slippa kompilera om allting, gör vi detta genom att klicka på motsvarande variabel märkt default, ändra värdet, klicka på enter och fortsätta simuleringen med Continue. Vill vi simulera i tio sekunder kan vi klicka på Continue nio gånger, ett annat sätt är att ändra Time Interval från 1 till 10 och klicka på Start. Simuleringen pågår nu i tio sekunder, se figur 5.5. Det går också att skala om axlarna om det är något speciellt avsnitt vi är intresserade av. Möjlighet finns även att lägga in fler variabler i samma oscilloskop-fönster, nackdelen med detta är att eftersom alla kurvorna är heldragna kan det vara svårt att urskilja vilken variabel som hör till rätt kurva. Det går också att ändra noggrannheten på simuleringen. När vi vill avsluta simuleringen klickar vi på Misc och Quit.

6. Simulering av Nordel

I detta kapitel skall vi simulera en modell av det Nordiska kraftsystemet som kallas Nordel. Det omfattar Norge, Sverige, Finland och Sjöland. Nordel har en total kapacitet på ungefär 70 GW.

Vi skall här bygga en 16-maskinsmodell av Nordel. Denna modell finns i Akke (1989). I figur 6.1 ser vi en grafisk representation av den Svenska delen av Nordel. Varje generator i modellen består av ett flertal generatorer i verkligheten. Generatorerna i Sverige heter från norr till söder: generator 7, generator 8,..., generator 13. I södra Sverige representeras generatorerna i huvudsak av kärnkraftverken Barsebäck, Oskarshamn, Ringhals och Forsmark. I norra Sverige representeras generatorerna av vattenkraftverk. Appendix C innehåller grafisk representation av Sverige, Danmark, Norge, Finland och Nordel samt Sverige med ett avbrott på en transmissionslinje. Data för modellen finns i Appendix B.

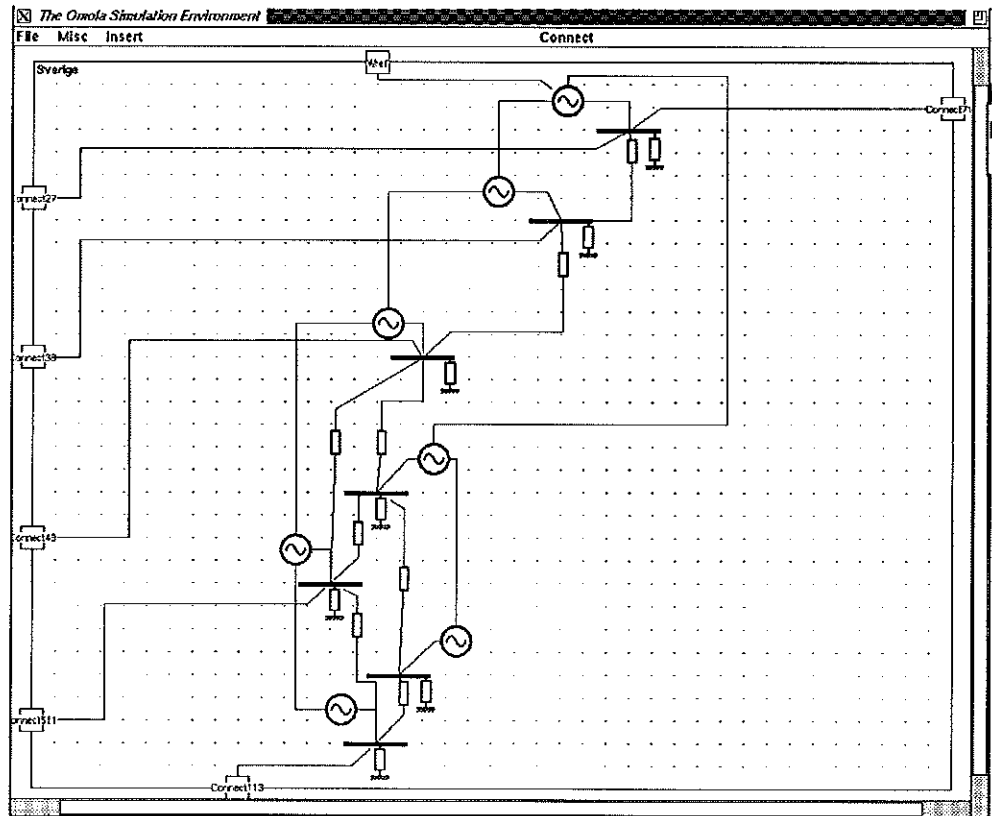
Trots att modellen inte innehåller mer än 16 generatorer är det många parametrar som skall knappas in. Modellen är gjord sådan att alla parametrar direkt har kunnat hämtas ur Akke (1989), parametervärden på lasterna är givna i W och VAR, dessa räknas i modellen automatiskt om till p.u., så att vi får ett R och ett X1. Däremot är motsvarande värden för linjerna redan givna i p.u. Även generatorernas parametrar är givna i p.u.

När modellen byggdes simulerades varje land för sig i stationäritet. Detta medförde inte några större problem. Problem blev det däremot då den slutliga modellen, enligt figur 6.2 skulle simuleras, och resultatet skulle jämföras med motsvarande simuleringar gjorda på Sydkraft med PSS/E. Simuleringarna gjorda i Omola svängde fortare än motsvarande i PSS/E, den stationära punkten stämde inte heller riktigt överens med Akkes på Sydkraft. Vem hade gjort fel? Var det jag?, var det Sydkraft, eller hade bägge gjort fel? Det visade sig efter ungefär 2 veckors felsökning, att $\omega_{ref} = \omega_0$ använts istället för $\omega_{ref} = 1$ p.u. Detta innebar 100π gånger större förstärkning i effektregulatorn. Under denna felsökning hittades dock en del fel i parametervärdena vilket bidrog till att Omola och PSS/E stämde exakt i stationäritet.

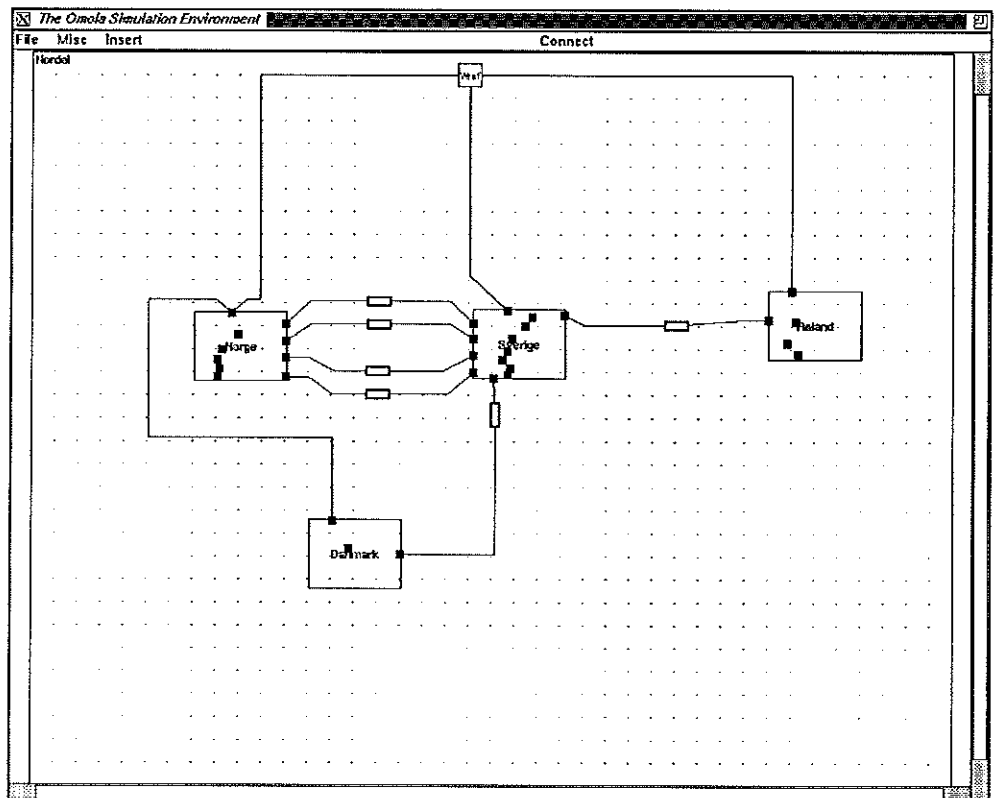
Tre olika störningar har simulerats. Avbrott på en ledning, steg i referensspänningen till en spänningsregulator samt störning i en last. Dessa har också jämförts med liknande simuleringar gjorda av Sydkraft med PSS/E.

6.1 Avbrott på en ledning

Den första störningen är ett avbrott på en ledning mellan generatorerna 10 och 12. Elnätet kan fungera efter avbrottet eftersom det finns alternativa vägar för effekten. I detta fallet betyder det att energiflödet från generator 12 till 10 går över generatorerna 13 och 11. Avbrottet börjar vid $t = 1$ sekund. De aktiva och reaktiva energiflödena för linjerna mellan generatorerna 10-11, 11-13, 13-12 samt 10-12 är plottade i figur 6.3. Här ser man att det svänger med högre frekvens för simuleringarna gjorda i Omola än motsvarande i PSS/E. Det kan bero på att dämpblindningar är modellerade i PSS/E men ej i Omola. I stationärt tillstånd är överensstämmelsen mycket bra mellan de bägge simuleringmodellerna.



Figur 6.1 Delmodell av Sverige i 16-generatorsystemet



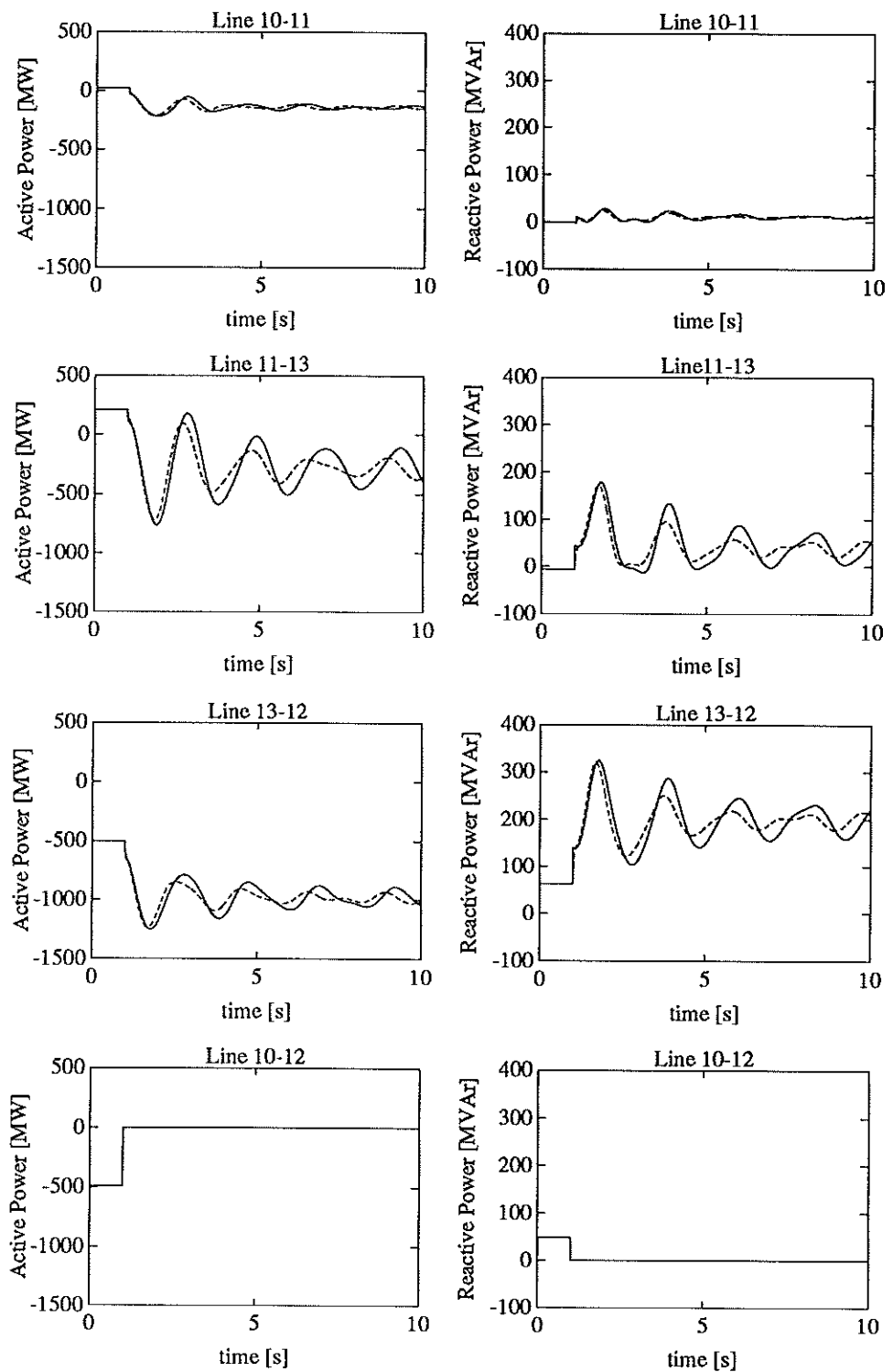
Figur 6.2 16-generatorsystem där Sverige, Danmark, Finland och Norge är hopkopplade

6.2 Steg i referensspänningen

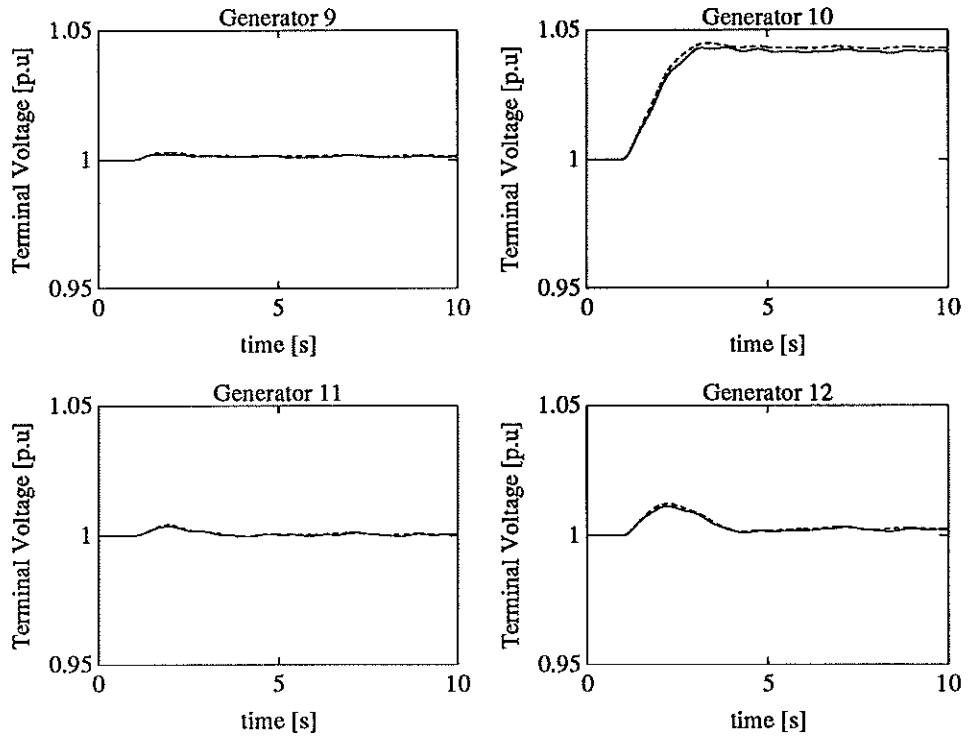
Den andra störningen är ett steg på 5% i referensspänningen till regulatort på generator 10 vid $t = 1$ sekund. I figur 6.4 och 6.5 är V_t och Q_e plottade för generatorerna 9, 10, 11 och 12. Här är överensstämmelsen mycket bra mellan vår modell i Omola och PSS/E.

6.3 Steg i last

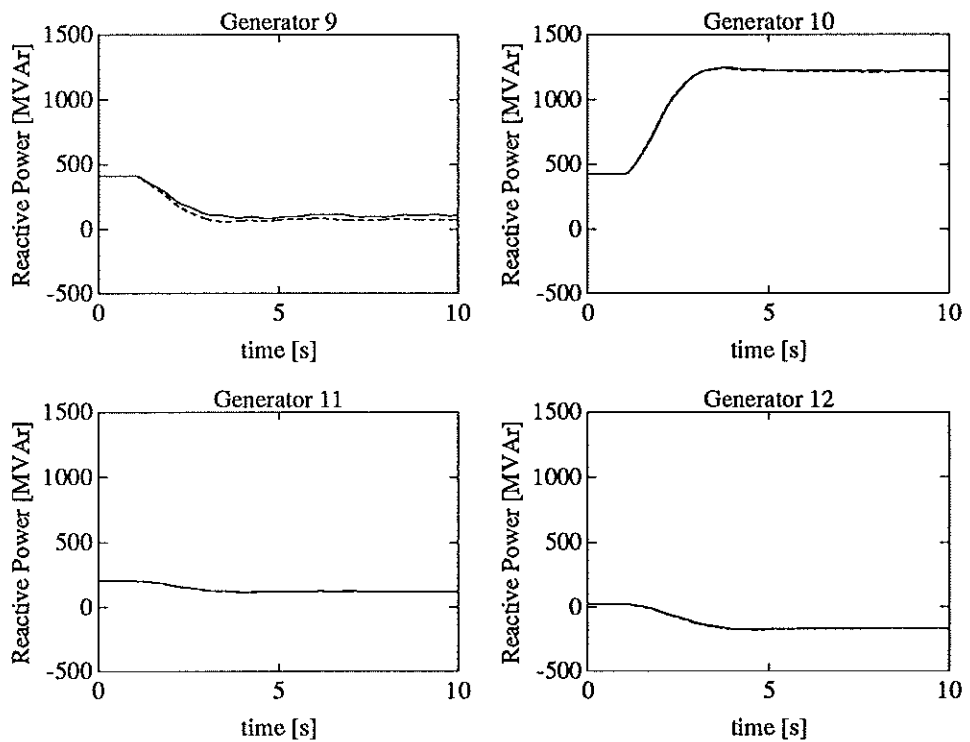
Den tredje störningen är ett steg i lasten med 10% vid generator 10 vid $t = 1$ sekund. I figur 6.6 är P_e plottad för generatorerna 9, 10, 11, och 12 och i figur 6.7 är frekvensavvikelsen plottad för dessa generatorer. Den aktiva effekten stämmer mycket bra mellan de två modellerna. I figur 6.7 svänger simuleringarna gjorda i Omola med högre frekvens än motsvarande i PSS/E, även här kan det bero på att PSS/E har dämpblindningar modellerade.



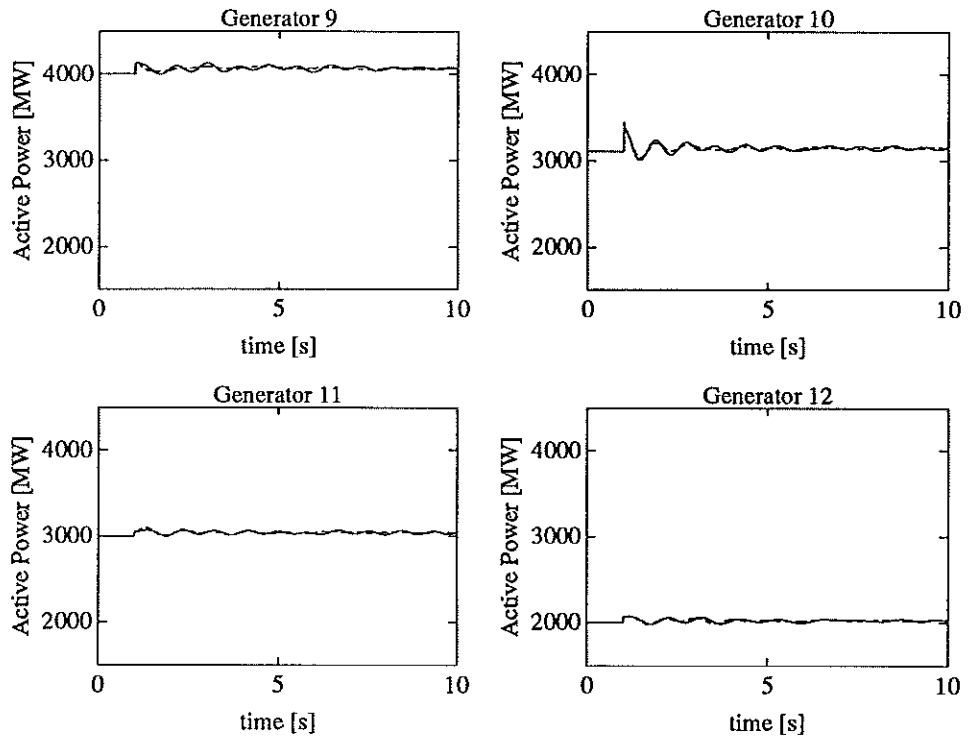
Figur 6.3 Aktivt och reaktivt energiflöde för några linjer, när linjen mellan nod 10 och nod 12 bryts vid $t = 1$. Helledragna linjer är simuleringsresultat från Omsim och streckade är från PSS/E



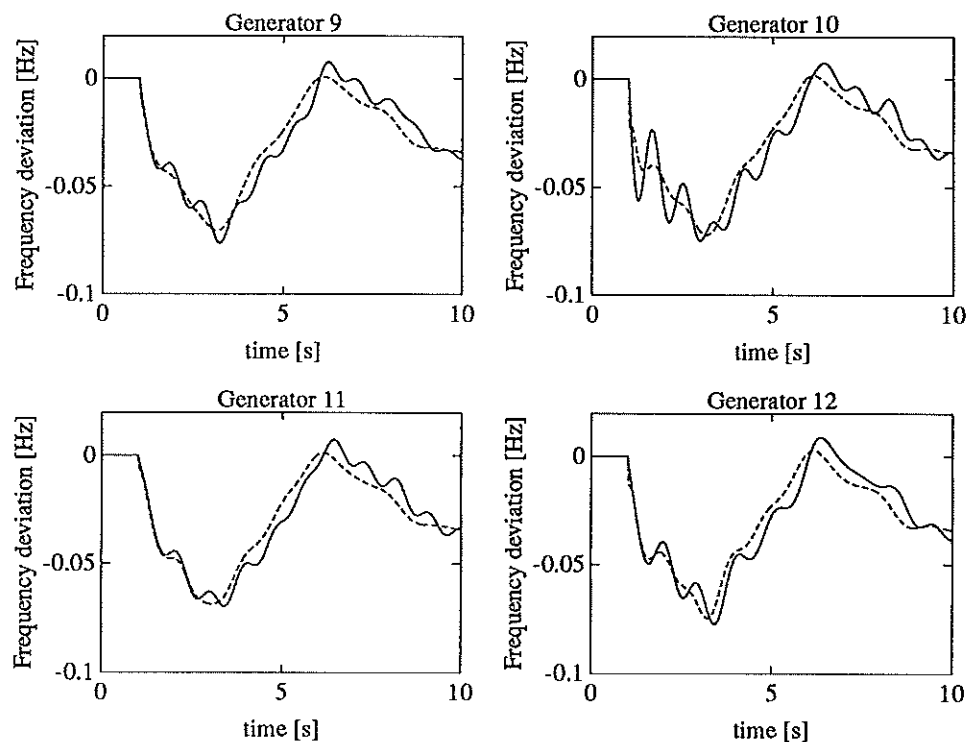
Figur 6.4 Terminals­p­ningen f­r n­gra generatorer, n­r ett steg simuleras till regulatorn p­ generator 10 vid $t = 1$. H­ldragna linjer ­r simuleringsresultat f­rn Omsim och streckade ­r f­rn PSS/E



Figur 6.5 Reaktiva effekten f­r n­gra generatorer, n­r ett steg simuleras till regulatorn p­ generator 10 vid $t = 1$. H­ldragna linjer ­r simuleringsresultat f­rn Omsim och streckade ­r f­rn PSS/E



Figur 6.6 Aktiva effekten för några generatorer, när lasten ökar 10% vid generator 10 vid $t = 1$. Heldragna linjer är simuleringsresultat från Omsim och streckade är från PSS/E



Figur 6.7 Frekvensdeviationen $f - f_0$ för några generatorer, när lasten ökar 10% vid generator 10 vid $t = 1$. Heldragna linjer är simuleringsresultat från Omsim och streckade är från PSS/E

7. Slutsatser

Syftet med detta examensarbete har varit att undersöka om och hur Omola kan användas till att modellera och simulera kraftsystem. Modellen består av genererande enheter, bussar, transmissionsledningar och laster. Dessa komponenter skapades i kapitel 3. Initialisering och modellering av störningar diskuterades i kapitel 4. Omola är ett bekvämt språk att bygga modeller i, och det har varit lätt att lägga till eller ta bort en eller flera delmodeller om det varit nödvändigt.

I kapitel 5 utnyttjades de ovan nämnda komponenterna till att skapa och simulera ett elnät med 2 generatorer. I kapitel 6 utvecklades och simulerades en 16-maskinsmodell (Akke, 1989) för det nordiska elnätet, Nordel. Alla parametervärden som har använts har direkt tagits ur Akke (1989). Har parametrarna givits i SI - enheterna, har automatisk omräkning gjorts till p.u. i Omola. Detta är viktigt eftersom det är många parametervärden att definiera i modellen. Om man tvingas att räkna om parametervärden manuellt ökar felrisken betydligt. Fel av denna typ kan vara svårupptäckta och om man upptäcker att det inte stämmer kan felsökning vara tidsödande. Som vi berättat i kapitel 6 kan det lätt uppstå missförstånd. Vi tog det för självklart att frekvensen in till frekvensregulatorerna skulle vara i rad/s, medan det för Sydkraft som givit oss beskrivningen av regulatorn var självklart att frekvens normerats med ω_0 . Detta ledde till två veckors extra arbete. Men det hade också det goda med sig att vi tvingades att kontrollera allt noga. En del andra små fel hittades också. Resultatet är alltså att vi kan ha större förtroende för modellen.

Simuleringstiden varierar ganska mycket beroende på hur många generatorer man använder i modellen. Simulering av en sju-generatormodell tar ca 2 minuter medan det för en sexton-generatormodell tar ca 9.5 minuter.

I kapitel 6 görs också en jämförelse med liknande simuleringar gjorda på Sydkraft med PSS/E. Av resultaten framgår att överensstämmelsen är mycket bra mellan de bägge simuleringsspråken. Simuleringarna gjorda med Omola oscillerar lite mer än motsvarande för PSS/E på ett par störningar. Detta kan bero på att dämpningarna är modellerade i PSS/E, men ej i Omola. I stationäritet skiljer det ingenting mellan de bägge simuleringsspråken.

Referenser

- ADIELSON, T. (1982): *SIMPOW - A Digital Program System for Static and Dynamic Simulation of Power System*, Paper presented at the international CEPSI conference in Bangkok, November 1982.
- AKKE, M. (1989): "Power System Stabilizers in Multimachine Systems," Licentiate Thesis TFRT-3201, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.
- ANDERSSON, M. (1989): "An Object-Oriented Modelling Environment," in G. Iazeolla, A. Lehman, H.J. van den Herik (Eds.): *Simulation Methodologies, Languages and Architectures and AI and Graphics for Simulation*, 1989 European Simulation Multiconference, Rome, June 7-9, 1989, The Society for Computer Simulation International, pp. 77-82.
- ANDERSSON, M. (1989): "An Object-Oriented Language for Model Representation," *1989 IEEE Control Systems Society Workshop on Computer-Aided Control System Design (CACSD)*, December 16, 1989, Tampa, Florida, pp. 8-15.
- ANDERSSON, M. (1990): *Omola — An Object-Oriented Language for Model Representation*, Licentiate Thesis CODEN: LUTFD2/TFRT-3208, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.
- ANDERSSON, P.M. and A.A. FOUAD (1977): *Power System Control and Stability*, The Iowa State University Press, Ames, Iowa.
- ANON, (1991): *Elmaskinsystem*, CODEN: LUTEDX/TEIE-1001, Department of Industrial Electrical Engineering and Automation, Lund Institute of Technology, Lund, Sweden.
- BERGEN, A.R. (1986): *Power System Analysis*, Prentice-Hall, Englewoods Cliffs, N.J.
- ELGERD, O.I. (1983): *Electric Energy Systems Theory — An introduction*, 2nd ed., Tata McGraw-Hill, New Delhi.
- FITZGERALD, A.E., C. KINGSLEY, JR. and D.D. UMANS (1985): *Electric Machinery*, 4th ed., McGraw-Hill, N.Y., USA.
- LINDAHL, S. (1971): "A State Space Model of a Multimachine Power System," Internal Report TFRT-7118, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.
- MATTSSON S.E. (1984): *Modelling and Control of Large Horizontal Axis Wind PowerPlants*, Doctoral Dissertation TFRT-1026, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.
- MATTSSON, S.E. and M. ANDERSSON (1990): "A kernel for System Representation," in Ü. Jaaksoo and V.I. Utkin: *Automatic Control — 11th Triennial World Congress, Tallinn, 1990*, Vol. V, Pergamon Press, pp. 373-378.

- STAGG, G.W. and A.H. EL-ABIAD (1968): *Computer Methods in Power Systems Analysis*, McGraw-Hill, USA.
- VOURNAS, C.D. and R.J. FLEMING (1978): "Generalization of the Heffron-Phillips Model of a Synchronous Generator," A 78 534-0, presented at the 1978 IEEE Summer PowerMeeting, Los Angeles, Ca.
- VOURNAS, C.D. and N.D. HATZIARGYRIOU (1991): "Computer-Aided Design of Electric Power Transmission Systems," *Preprints of the IFAC Symposium Computer Aided Design in Control Systems*, Swansea, UK, July 1991, pp. 573-578.

Appendix A

Modeller

A1 - Omolamodeller för grundläggande enheter

```
Library Basic_Parts;
Uses Base;
%
%
pi TYPE Real := 4*atan(1);
w0 TYPE Real := ::pi*100; % Nominal angular frequency [rad/s]
Vbase TYPE Real := 4e5; % Global voltage base [V]
Sbase TYPE Real := 1e8; % Global power base [VA]
Ibase TYPE Real := ::Sbase/(::Vbase*sqrt(3)); % Global current base [A]
%
%
VeryBigModel ISA Model WITH
  Graphic ISA Layout WITH
    x_size := 1000;
    y_size := 800;
  END;
END;
%
%
VoltageTerminal ISA SimpleTerminal WITH
  quantity := "electric.voltage";
  unit := "V";
  direction := 'Across;
END;
%
%
In_CurrentTerminal ISA SimpleTerminal WITH
  quantity := "electric.current";
  unit := "A";
  direction := 'In;
END;
%
%
Out_CurrentTerminal ISA SimpleTerminal WITH
  quantity := "electric.current";
  unit := "A";
  direction := 'Out;
END;
%
%
Out_Angular_MomentumTerminal ISA SimpleTerminal WITH
  quantity := "angular.momentum";
  unit := "kg.m2/s";
  direction:= 'out;
END;
%
%
Out_Moment_of_InertiaTerminal ISA SimpleTerminal WITH
  quantity := "moment.of.inertia";
```

```

    unit      := "kg.m2";
    direction:= 'out;
END;
%
%
AngularTerminal ISA SimpleTerminal WITH
quantity := "angular.velocity";
unit      := "rad/s";
direction:= 'Across;
END;
%
%
In_El_Term  ISA RecordTerminal WITH
Components:
    Vd,Vq  ISA VoltageTerminal;
    Id,Iq  ISA In_CurrentTerminal;
END;
%
%
Out_El_Term ISA RecordTerminal WITH
Components:
    Vd,Vq  ISA VoltageTerminal;
    Id,Iq  ISA Out_CurrentTerminal;
END;
%
%
Out_Mass_Term ISA RecordTerminal WITH
Components:
    L ISA Out_Angular_MomentumTerminal;
    H ISA Out_Moment_of_InertiaTerminal;
    W ISA AngularTerminal;
END;
%
%
OnePort ISA Base::Model WITH
attributes:
    T1 ISA In_El_Term WITH
    attributes:
        Graphic ISA Base::Layout WITH
        attributes:
            x_pos := 200;
            y_pos := 300;
            invisible := 1;
        END;
    END;
END;
%
%
TwoPort ISA Base::Model WITH
attributes:
    T1 ISA In_El_Term WITH
    attributes:
        Graphic ISA Base::Layout WITH
        attributes:
            x_pos := 200;
            y_pos := 150;
            invisible := 1;
        END;
    END;
    T2 ISA In_El_Term WITH
    attributes:
        Graphic ISA Base::Layout WITH
        attributes:

```

```

        x_pos := 200;
        y_pos := 150;
        invisible := 1;
    END;
END;
%
%
Bus ISA In_EL_Term WITH      % Node that connects the
equations:                  % generator, the Line and
    Id = 0;                  % the Load
    Iq = 0;
Graphic ISA Base::Layout WITH
    bitmap TYPE String := "bus";
END;
END;
%
%
InertiaBus ISA Out_Mass_Term WITH % Node where the reference
equations:                  % angular frequency is
    W = L/H;                  % calculated
END;

```

A2 - Omolamodeller för linjer och laster

```

Library Impedances;
Uses Base, Basic_Parts;
%
%
%
alpha TYPE Real := 0.0;
% Defines the complexity of the impedanc models below. If alpha = 0, the
% impedance models are algebraic in the d,q components of the voltages and
% currents. You may say that the models are jw-models. If alpha = 1,
% derivations of the d,q components of the voltages and currents may appear
% in the equations.
%
%
%
RLImpedance ISA TwoPort WITH
% A symmetric 3-phase model of a resistor and inductor in series. Uses
% the d,q - representation for voltages and currents. The complexity of the
% model for the inductor part can be controlled by the global variable
% ::alpha. If ::alpha = 0, the derivations of the currents Id and Iq through
% the inductor is neglected. If ::alpha = 1 their contribution to the voltage
% drop over the inductor is included in the model.
parameters:
  R Type Discrete Real;    % Resistance [p.u]
  Xl Type Discrete Real;   % Reactance [p.u]
variable:
  L TYPE Real;
equations:
  L = Xl/Basic_Parts::W0;
  T1.Vd - T2.Vd = R*T1.Id + ::alpha*L*dot(T1.Id) - Xl*T1.Iq;
  T1.Vq - T2.Vq = R*T1.Iq + ::alpha*L*dot(T1.Iq) + Xl*T1.Id;
  T1.Id = - T2.Id;
  T1.Iq = - T2.Iq;
END;
%
%
Line0 ISA RLImpedance WITH
% Models a symmetric 3-phase transmission line by a resistor and inductor
% in series.
parameter:
  R ISA Parameter;        % Resistance [p.u]
  Xl ISA Parameter;       % Reactance [p.u]
variable:
  Ploss, Qloss, Pflow1, Qflow1, Pflow2, Qflow2 TYPE Real;
  % The power losses in the line. Active and reactive
  % energy flows through the terminals in [MW] and [MVar].
equations:
  Ploss = R *(sqr(T1.Id) + sqr(T1.Iq))*basic_parts::Sbase/1e6;
  Qloss = Xl*(sqr(T1.Id) + sqr(T1.Iq))*basic_parts::Sbase/1e6;
  Pflow1 = (T1.Vd*T1.Id + T1.Vq*T1.Iq)*basic_parts::Sbase/1e6;
  Qflow1 = (T1.Vq*T1.Id - T1.Vd*T1.Iq)*basic_parts::Sbase/1e6;
  Pflow2 = (T2.Vd*T2.Id + T2.Vq*T2.Iq)*basic_parts::Sbase/1e6;
  Qflow2 = (T2.Vq*T2.Id - T2.Vd*T2.Iq)*basic_parts::Sbase/1e6;
attributes:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "linev";
  END;
END;
%
%
Line ISA Line0 WITH
% Extension of the model Line0 for a transmission line to allow

```



```

% initialization by defining the voltages Ti.Vd and Ti.Vq for
% both ends (i = 1,2) in terms of amplitude Vt0i and phase Theta0i
% Ti.Vd = -Vt0i*sin(Theta0i)
% Ti.Vq = Vt0i*cos(Theta0i)
% Stationarity is also assumed.
parameters:
  Theta01 ISA Parameter; % [deg]
  Theta02 ISA Parameter; % [deg]
  Vt01 ISA Parameter WITH default := 1.0; END; % [p.u]
  Vt02 ISA Parameter WITH default := 1.0; END; % [p.u]
Init ISAN Event;
OnEvent Init DO
  new(T1.Vd) := -Vt01*sin(Theta01*Basic_Parts::Pi/180);
  new(T1.Vq) := Vt01*cos(Theta01*Basic_Parts::Pi/180);
  new(T2.Vd) := -Vt02*sin(Theta02*Basic_Parts::Pi/180);
  new(T2.Vq) := Vt02*cos(Theta02*Basic_Parts::Pi/180);
  new(T1.Id) := 1/(sqr(R) + sqr(Xl))*(R*(new(T1.Vd)-new(T2.Vd))
    + Xl*(new(T1.Vq)-new(T2.Vq)));
  new(T1.Iq) := 1/(sqr(R) + sqr(Xl))*(-Xl*(new(T1.Vd)-new(T2.Vd))
    + R*(new(T1.Vq)-new(T2.Vq)));

  new(T2.Id) := - new(T1.Id);
  new(T2.Iq) := - new(T1.Iq);
End;
END;
%
%
Ind_Load0 ISA OnePort WITH
% Models a symmetric 3-phase load by a resistor and inductor in series
% connected to a bus. When the bus is 1 p.u, the load is fixed by POSI
% and QOSI.
parameters:
  POSI ISA Parameter; % [W]
  QOSI ISA Parameter; % [VAr]
submodels:
  imp ISA RLImpedance;
variables:
  P0, Q0 TYPE Discrete Real; % Initialization Load [p.u]
  Ploss, Qloss TYPE Real;
  %The power losses in the load. Active and reactive power
  %in [MW] and [MVar]
equations:
  imp.T2.Vd = 0;
  imp.T2.Vq = 0;
  % T2.imp is grounded. Ind_Load0 has only one terminal, T1.
  Ploss = imp.R *(sqr(T1.Id) + sqr(T1.Iq))*basic_parts::Sbase/1e6;
  Qloss = imp.Xl*(sqr(T1.Id) + sqr(T1.Iq))*basic_parts::Sbase/1e6;
connections:
  T1 AT imp.T1;
attributes:
  Graphic ISA Base::Layout WITH
  bitmap TYPE String := "indload";
END;

Init ISAN Event;
OnEvent Init DO
  new(P0) := POSI/basic_parts::Sbase;
  new(Q0) := QOSI/basic_parts::Sbase;
  new(imp.R) := new(P0)/(sqr(new(P0)) + sqr(new(Q0)));
  new(imp.Xl) := new(Q0)/(sqr(new(P0)) + sqr(new(Q0)));
END;
END;
%
%
```

```

Ind_Load ISA Ind_Load0 WITH
% Extension of the model Ind_Load0 for a load to allow
% initialization by defining the voltage T1.Vd and T1.Vq in
% terms of amplitude Vt01 and phase Theta01
% T1.Vd = -Vt01*sin(Theta01)
% T1.Vq = Vt01*cos(Theta01)
% Stationarity is also assumed.
Theta01 ISA Parameter;
Vt01 ISA Parameter WITH default := 1.0; END;
Init ISAN Event;
OnEvent Init DO
    new(T1.Vd) := -Vt01*sin(Theta01*Basic_Parts::Pi/180);
    new(imp.T1.Vd) := new(T1.Vd);
    new(T1.Vq) := Vt01*cos(Theta01*Basic_Parts::Pi/180);
    new(imp.T1.Vq) := new(T1.Vq);
    new(T1.Id) := 1/(sqr(new(imp.R)) + sqr(new(imp.Xl)))*(new(imp.R)
        *(new(T1.Vd)) + new(imp.Xl)*(new(T1.Vq)));
    new(imp.T1.Id) := new(T1.Id);
    new(T1.Iq) := 1/(sqr(new(imp.R)) + sqr(new(imp.Xl)))*(-new(imp.Xl)
        *(new(T1.Vd)) + new(imp.R)*(new(T1.Vq)));
    new(imp.T1.Iq) := new(T1.Iq);
End;
END;
%
%
StepLoad ISA Ind_Load WITH
% StepLoad is used to make a step in the load.
% The step is 10% of nominal load.
Parameter:
    StepTime ISA Parameter WITH default := 1; END; % Starttime of step [s]

Init, Step ISAN Event;

OnEvent Init DO
    schedule(Step, StepTime);
END;

OnEvent Step DO
    new(P0) := (1.1*POSI)/basic_parts::Sbase;
    new(Q0) := (1.1*QOSI)/basic_parts::Sbase;
    new(imp.R) := new(P0)/(sqr(new(P0)) + sqr(new(Q0)));
    new(imp.Xl) := new(Q0)/(sqr(new(P0)) + sqr(new(Q0)));
END;
END;
%
%
Interrupt ISA TwoPort WITH
% Interrupt is used for disconnection of a 3-phase transmission line
Parameter:
    FaultTime ISA Parameter WITH default := 1; END;
    % Starttime of fault [s]
    FaultDuration ISA Parameter WITH default := 100; END;
    % Duration of fault [s]
variables:
    fault TYPE DISCRETE integer;
Init, Break, BreakAction ISAN Event;

OnEvent Init DO
    schedule(Break, FaultTime);
    new(fault) := 0;
END;

OnEvent Break DO

```

```

    new(fault) := 1;
    schedule(BreakAction, FaultDuration);
END;

OnEvent BreakAction DO
    new(fault) := 0;
END;

equations:
    if (fault == 1) then T1.Id else( T1.Vd - T2.Vd) = 0;
    if (fault == 1) then T1.Iq else( T1.Vq - T2.Vq) = 0;
    T1.Id = - T2.Id;
    T1.Iq = - T2.Iq;
Graphic ISA Base::Layout WITH
    bitmap TYPE String := "interrupt";
END;
END;

```

A3 - Omolamodeller för genererande enheter

```

Library Generators;
Uses Base, Basic_Parts;
%
%
%
Generator0 ISA Model WITH
% Models a generating unit as a synchronous generator. The driving
% mechanically torque to the generator is modelled statically.
% The generator is modelled by the swing equation and one electric
% time constant.
terminals:
  T1 ISA Out_El_Term WITH
  attributes:
    Graphic ISA Base::Layout WITH
    attributes:
      x_pos := 200;
      y_pos := 150;
      invisible := 1;
    END;
  END;
  T2 ISA Out_Mass_Term WITH
  attributes:
    Graphic ISA Base::Layout WITH
    attributes:
      x_pos := 200;
      y_pos := 150;
      invisible := 1;
    END;
  END;

parameters:
  % pu units
  In ISA Parameter;          % Base current [A]
  Sn ISA Parameter;         % Base power [VA]
  Vn ISA Parameter;        % Base voltage [V]

  % Generator parameters
  H ISA Parameter;         % Inertia time constant [s]
  D ISA Parameter;         % Speed damping term [p.u]
  TdOp ISA Parameter;      % d-axis transient open circuit time
                           % constant [s]
  Xd ISA Parameter;        % d-axis synchronous reactance [p.u]
  Xq ISA Parameter;        % q-axis synchronous reactance [p.u]
  Xdp ISA Parameter;       % d axis transient reactance [p.u]

variables:
  % Generator variables
  Vref ISA Parameter;      % Reference voltage to voltage regulator [p.u]
  W ISA variable;          % Generator electric angular velocity [rad/s]
  Pe ISA variable;         % Generated electric active power [p.u]
  PeMW ISA variable;       % Generated electric active power [MW]
  Qe ISA variable;         % Generated electric reactive power [p.u]
  QeMVar ISA variable;     % Generated electric reactive power [MVar]
  Vt ISA variable;         % Terminal voltage [p.u]
  Vd ISA variable;         % d-axis component of Vt [p.u]
  Vq ISA variable;         % q-axis component of Vt [p.u]
  Id ISA variable;         % d-axis component of It [p.u]
  Iq ISA variable;         % q-axis component of It [p.u]
  % Vd, Vq, Id, Iq are values in the local generator system. The d,q values
  % in the global system are given by T1.Vd, T1.Vq, T1.Id, T1.Iq.
  Delta ISA variable;      % Angle from reference -

```

```

% to the machine q-axis [rad]
Deltadeg ISA variable; % Delta in degrees
Eqp ISA variable; % Stator emk on q-axis, prop. to -
% flux from field winding [p.u]
Wrel ISA variable; % Relative angular velocity
DeltaFreq ISA variable; % Frequency deviation from f0 = W0/(2*pi) [Hz]
Pm ISA Variable; % Mechanical power [p.u]
Uf ISA variable; % Field voltage [p.u]

equations:
% Generator equations
dot(delta) = W - T2.W;
Wrel = W/Basic_Parts::W0;
DeltaFreq = (W - Basic_Parts::W0)/(2*Basic_Parts::pi);
Deltadeg = 180/Basic_Parts::pi*Delta;
2*H*dot(W) = -D*(W-Basic_Parts::w0) + Basic_Parts::w0*(Pm - Pe);
Td0p*dot(Eqp) = (Uf - Eqp - (Xd - Xdp)*Id);
Pe = Id*Iq*(Xq - Xdp) + Iq*Eqp;
Qe = Vq*Id - Vd*Iq;
PeMW = Pe*Sn/1e6;
QeMVar = Qe*Sn/1e6;
Vq = Eqp - Xdp*Id;
Vd = Xq*Iq;
Vt = sqrt(Vd*Vd + Vq*Vq);
T1.Vd = Vn/Basic_Parts::Vbase*(Vd*cos(Delta) - Vq*sin(Delta));
T1.Vq = Vn/Basic_Parts::Vbase*(Vd*sin(Delta) + Vq*cos(Delta));
T1.Id = In/Basic_Parts::Ibase*(Id*cos(Delta) - Iq*sin(Delta));
T1.Iq = In/Basic_Parts::Ibase*(Id*sin(Delta) + Iq*cos(Delta));
% T1.Vd, T1.Vq, T1.Id, T1.Iq are values in the global system
T2.L = H*W;
T2.H = H;

% Generator parameters
POSI ISA Parameter;
QOSI ISA Parameter;
Theta0 ISA Parameter;
Vt0 ISA Parameter;
% Defines the active- and reactive power that the generator
% produce under assumption of stationarity in [W] and [VAr],
% and the angle Theta0 [deg] of the terminal voltage Vt0 [p.u]

S0, P0, Q0 TYPE DISCRETE Real;
It0 TYPE DISCRETE Real;
SinPhi TYPE DISCRETE Real;
Uf0 TYPE DISCRETE Real; % Nominal Field Voltage [p.u]
Pm0 TYPE DISCRETE Real; % Nominal Mechanical Power [p.u]

Init ISAN Event;
OnEvent Init DO
    new(P0) := POSI/Sn;
    new(Q0) := QOSI/Sn;
    new(S0) := sqrt(sqrt(new(P0)) + sqrt(new(Q0)));
    new(It0) := new(S0)/Vt0;
    new(SinPhi) := new(Q0)/new(S0);
    new(Iq) := new(P0)/sqrt(Vt0*Vt0 + 2*Vt0*new(It0)*Xq*new(SinPhi) +
        sqrt(new(It0)*Xq));

    new(Id) := (if Q0 < 0 then -1 else 1)*sqrt(sqrt(new(It0)) -
        sqrt(new(Iq)));
    new(Eqp) := sqrt(Vt0*Vt0 - sqrt(Xq*new(Iq))) + Xdp*new(Id);
    new(delta) := Theta0*Basic_Parts::pi/180 + asin(Xq*new(Iq)/Vt0);
    new(W) := Basic_Parts::W0;
    new(Uf0) := new(Eqp) + (Xd - Xdp)*new(Id);

```

```

new(Pm0) := new(P0);
new(Vq) := new(Eqp) - Xdp*new(Id);
new(Vd) := Xq*new(Iq);
new(T1.Vd) := Vn/Basic_Parts::Vbase*(new(Vd)*cos(new(Delta)) -
new(Vq)*sin(new(Delta)));
new(T1.Vq) := Vn/Basic_Parts::Vbase*(new(Vd)*sin(new(Delta)) +
new(Vq)*cos(new(Delta)));
new(T1.Id) := In/Basic_Parts::Ibase*(new(Id)*cos(new(Delta)) -
new(Iq)*sin(new(Delta)));
new(T1.Iq) := In/Basic_Parts::Ibase*(new(Id)*sin(new(Delta)) +
new(Iq)*cos(new(Delta)));

END;
attributes:
  Graphic ISA Layout WITH
    bitmap TYPE String := "generator";
  END;
END;
%
%
Generator ISA Generator0 WITH
% Models a power and voltage regulator.
%
% Power regulator variables
  dotDelPmunl, dotDelPmi ISA variable;
  DelPm ISA variable;      % Pm - P0
% Operating parameter
  K ISA Parameter;        % Power regulator gain
% Power regulator equations. The control is limited 0-1 [p.u]
  dotDelPmunl = ((K*(1 - W/Basic_Parts::W0)) - DelPm)/3;
  Pm = Pm0 + DelPm;
  dotDelPmi = if (Pm > 1.0 and dotDelPmunl > 0.0) then 0.0 else dotDelPmunl;
  dot(DelPm) = if (Pm < 0.0 and dotDelPmi < 0.0) then 0.0 else dotDelPmi;
% Voltage regulator variables
  dotDelUfunl, dotDelUf1 ISA variable;
  DelUf ISA variable;      % Field voltage deviation from nominal
                          % value [p.u]
% Voltage regulator equations. The control is limited to 0-10 [p.u]
  dotDelUfunl = ((30*(Vref - Vt)) - DelUf)*2;
  Uf = Uf0 + DelUf;
  dotDelUf1 = if (Uf > 10.0 and dotDelUfunl > 0.0) then 0.0 else
dotDelUfunl;
  dot(DelUf) = if (Uf < 0.0 and dotDelUf1 < 0.0) then 0.0 else dotDelUf1;
END;

```

Appendix B

En 16-generatormodell av Nordel

```
Sverige ISA Basic_Parts::VeryBigModel WITH
attributes:
  Generator13 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 351;
      y_pos := 101;
    END;
  END;
  Ind_Load13 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 394;
      y_pos := 44;
    END;
  END;
  Generator11 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 301;
      y_pos := 276;
    END;
  END;
  Generator12 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 476;
      y_pos := 176;
    END;
  END;
  Generator10 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 451;
      y_pos := 376;
    END;
  END;
  Ind_Load12 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 444;
      y_pos := 119;
    END;
  END;
  Line1213 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
```

```

        x_pos := 419;
        y_pos := 119;
    END;
END;
Ind_Load11 ISA Impedances::Ind_Load WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 344;
        y_pos := 219;
    END;
END;
Line1113 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 368;
        y_pos := 193;
    END;
END;
Line1011 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 369;
        y_pos := 294;
    END;
END;
Line910 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 394;
        y_pos := 394;
    END;
END;
Ind_Load9 ISA Impedances::Ind_Load WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 469;
        y_pos := 469;
    END;
END;
Line911 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 344;
        y_pos := 394;
    END;
END;
Generator9 ISA Generators::Generator WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 401;
        y_pos := 526;
    END;
END;
Generator8 ISA Generators::Generator WITH
attributes:
    Graphic ISA super::Graphic WITH

```



```

attributes:
  x_pos := 521;
  y_pos := 671;
END;
END;
Ind_Load8 ISA Impedances::Ind_Load WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 619;
    y_pos := 619;
  END;
END;
Line78 ISA Impedances::Line WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 666;
    y_pos := 716;
  END;
END;
Generator7 ISA Generators::Generator WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 596;
    y_pos := 771;
  END;
END;
Ind_Load7 ISA Impedances::Ind_Load WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 691;
    y_pos := 716;
  END;
END;
Line89 ISA Impedances::Line WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 591;
    y_pos := 591;
  END;
END;
Line1012 ISA Impedances::Line WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 408;
    y_pos := 183;
  END;
END;
Ind_Load10 ISA Impedances::Ind_Load WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 379;
    y_pos := 254;
  END;
END;
Bus13 ISA Basic_Parts::Bus WITH
attributes:

```

```

Graphic ISA super::Graphic WITH
attributes:
  x_pos := 375;
  y_pos := 50;
END;
END;
Bus11 ISA Basic_Parts::Bus WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 325;
    y_pos := 225;
  END;
END;
Bus12 ISA Basic_Parts::Bus WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 400;
    y_pos := 125;
  END;
END;
Bus10 ISA Basic_Parts::Bus WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 375;
    y_pos := 325;
  END;
END;
Bus9 ISA Basic_Parts::Bus WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 425;
    y_pos := 475;
  END;
END;
Wref ISA Basic_Parts::Out_Mass_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 375;
    y_pos := 800;
  END;
END;
Bus8 ISA Basic_Parts::Bus WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 576;
    y_pos := 626;
  END;
END;
Bus7 ISA Basic_Parts::Bus WITH
attributes:
  Graphic ISA super::Graphic WITH
  attributes:
    x_pos := 650;
    y_pos := 725;
  END;
END;
Connect113 ISA Basic_Parts::In_El_Term WITH

```

```

attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 225;
    y_pos := 1;
  END;
END;
Connect714 ISA Basic_Parts::In_El_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 1000;
    y_pos := 750;
  END;
END;
Connect27 ISA Basic_Parts::In_El_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 1;
    y_pos := 650;
  END;
END;
Connect38 ISA Basic_Parts::In_El_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 1;
    y_pos := 475;
  END;
END;
Connect49 ISA Basic_Parts::In_El_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 1;
    y_pos := 275;
  END;
END;
Connect511 ISA Basic_Parts::In_El_Term WITH
attributes:
  Graphic ISA Base::Layout WITH
  attributes:
    x_pos := 1;
    y_pos := 75;
  END;
END;
C1 ISA Base::Connection WITH
attributes:
  Bus13 AT Ind_Load13.T1;
  bpoints TYPE Matrix [2, 2] := [375.000, 50.0000; 393.000, 62.0000];
END;
C2 ISA Base::Connection WITH
attributes:
  Bus12 AT Generator12.T1;
  bpoints TYPE Matrix [3, 2] := [400.000, 125.000; 452.000,
                                175.000; 475.000, 175.000];
END;
C3 ISA Base::Connection WITH
attributes:
  Bus12 AT Ind_Load12.T1;
  bpoints TYPE Matrix [2, 2] := [400.000, 125.000; 443.000, 137.000];
END;

```

```

C4 ISA Base::Connection WITH
attributes:
  Bus13 AT Line1213.T2;
  bpoints TYPE Matrix [3, 2] := [375.000, 50.0000; 418.000,
                                81.0000; 418.000, 118.000];
END;
C5 ISA Base::Connection WITH
attributes:
  Bus12 AT Line1213.T1;
  bpoints TYPE Matrix [2, 2] := [400.000, 125.000; 418.000, 118.000];
END;
C6 ISA Base::Connection WITH
attributes:
  Bus11 AT Ind_Load11.T1;
  bpoints TYPE Matrix [2, 2] := [325.000, 225.000; 343.000, 237.000];
END;
C7 ISA Base::Connection WITH
attributes:
  Bus11 AT Line1113.T1;
  bpoints TYPE Matrix [3, 2] := [325.000, 225.000; 367.000,
                                225.000; 367.000, 192.000];
END;
C8 ISA Base::Connection WITH
attributes:
  Bus10 AT Line1011.T1;
  bpoints TYPE Matrix [3, 2] := [375.000, 325.000; 368.000,
                                325.000; 368.000, 293.000];
END;
C9 ISA Base::Connection WITH
attributes:
  Bus9 AT Ind_Load9.T1;
  bpoints TYPE Matrix [2, 2] := [425.000, 474.000; 468.000, 487.000];
END;
C10 ISA Base::Connection WITH
attributes:
  Generator13.T2 AT Generator11.T2;
  bpoints TYPE Matrix [3, 2] := [350.000, 100.000; 300.000,
                                100.000; 300.000, 275.000];
END;
C11 ISA Base::Connection WITH
attributes:
  Generator12.T2 AT Generator10.T2;
  bpoints TYPE Matrix [3, 2] := [475.000, 175.000; 475.000,
                                375.000; 450.000, 375.000];
END;
C12 ISA Base::Connection WITH
attributes:
  Bus9 AT Generator9.T1;
  bpoints TYPE Matrix [3, 2] := [425.000, 474.000; 425.000,
                                525.000; 400.000, 525.000];
END;
C13 ISA Base::Connection WITH
attributes:
  Bus8 AT Ind_Load8.T1;
  bpoints TYPE Matrix [2, 2] := [575.000, 625.000; 618.000, 637.000];
END;
C14 ISA Base::Connection WITH
attributes:
  Bus8 AT Generator8.T1;
  bpoints TYPE Matrix [3, 2] := [575.000, 625.000; 575.000,
                                670.000; 520.000, 670.000];
END;
C15 ISA Base::Connection WITH

```

```

attributes:
  Bus8 AT Line78.T2;
  bpoints TYPE Matrix [3, 2] := [575.000, 625.000; 665.000,
                                625.000; 665.000, 715.000];
END;
C16 ISA Base::Connection WITH
attributes:
  Bus7 AT Generator7.T1;
  bpoints TYPE Matrix [3, 2] := [649.000, 724.000; 649.000,
                                770.000; 595.000, 770.000];
END;
C17 ISA Base::Connection WITH
attributes:
  Bus7 AT Ind_Load7.T1;
  bpoints TYPE Matrix [2, 2] := [649.000, 724.000; 690.000, 734.000];
END;
C18 ISA Base::Connection WITH
attributes:
  Generator7.T2 AT Generator8.T2;
  bpoints TYPE Matrix [3, 2] := [595.000, 770.000; 520.000,
                                770.000; 520.000, 670.000];
END;
C19 ISA Base::Connection WITH
attributes:
  Bus9 AT Line911.T1;
  bpoints TYPE Matrix [3, 2] := [425.000, 474.000; 343.000,
                                430.000; 343.000, 393.000];
END;
C20 ISA Base::Connection WITH
attributes:
  Generator10.T2 AT Generator7.T2;
  bpoints TYPE Matrix [6, 2] := [450.000, 375.000; 450.000,
                                413.000; 769.000, 413.000;
                                769.000, 798.000; 595.000,
                                798.000; 595.000, 770.000];
END;
C21 ISA Base::Connection WITH
attributes:
  Bus13 AT Generator13.T1;
  bpoints TYPE Matrix [3, 2] := [375.000, 50.0000; 375.000,
                                100.000; 350.000, 100.000];
END;
C22 ISA Base::Connection WITH
attributes:
  Bus11 AT Generator11.T1;
  bpoints TYPE Matrix [3, 2] := [325.000, 225.000; 325.000,
                                275.000; 300.000, 275.000];
END;
C23 ISA Base::Connection WITH
attributes:
  Bus11 AT Line911.T2;
  bpoints TYPE Matrix [2, 2] := [325.000, 225.000; 343.000, 393.000];
END;
C24 ISA Base::Connection WITH
attributes:
  Bus11 AT Line1011.T2;
  bpoints TYPE Matrix [3, 2] := [325.000, 225.000; 368.000,
                                266.000; 368.000, 293.000];
END;
C25 ISA Base::Connection WITH
attributes:
  Bus10 AT Generator10.T1;
  bpoints TYPE Matrix [3, 2] := [375.000, 325.000; 422.000,

```

```

375.000; 450.000, 375.000];
END;
C26 ISA Base::Connection WITH
attributes:
  Bus10 AT Line910.T2;
  bpoints TYPE Matrix [2, 2] := [375.000, 325.000; 393.000, 393.000];
END;
C27 ISA Base::Connection WITH
attributes:
  Generator11.T2 AT Generator9.T2;
  bpoints TYPE Matrix [3, 2] := [300.000, 275.000; 300.000,
525.000; 400.000, 525.000];
END;
C28 ISA Base::Connection WITH
attributes:
  Generator8.T2 AT Generator9.T2;
  bpoints TYPE Matrix [3, 2] := [520.000, 670.000; 400.000,
670.000; 400.000, 525.000];
END;
C29 ISA Base::Connection WITH
attributes:
  Bus9 AT Line910.T1;
  bpoints TYPE Matrix [4, 2] := [425.000, 474.000; 425.000,
439.000; 393.000, 439.000;
393.000, 393.000];
END;
C30 ISA Base::Connection WITH
attributes:
  Bus13 AT Line1113.T2;
  bpoints TYPE Matrix [4, 2] := [375.000, 50.0000; 375.000,
117.000; 367.000, 117.000;
367.000, 192.000];
END;
C31 ISA Base::Connection WITH
attributes:
  Bus7 AT Connect714;
  bpoints TYPE Matrix [3, 2] := [649.000, 724.000; 696.000,
749.000; 999.000, 749.000];
END;
C32 ISA Base::Connection WITH
attributes:
  Line78.T1 AT Bus7;
  bpoints TYPE Matrix [2, 2] := [665.000, 715.000; 649.000, 724.000];
END;
C33 ISA Base::Connection WITH
attributes:
  Bus9 AT Line89.T2;
  bpoints TYPE Matrix [4, 2] := [425.000, 474.000; 468.000,
516.000; 590.000, 516.000;
590.000, 590.000];
END;
C34 ISA Base::Connection WITH
attributes:
  Line89.T1 AT Bus8;
  bpoints TYPE Matrix [2, 2] := [590.000, 590.000; 575.000, 625.000];
END;
C35 ISA Base::Connection WITH
attributes:
  Bus9 AT Connect49;
  bpoints TYPE Matrix [5, 2] := [425.000, 474.000; 425.000,
506.000; 105.000, 506.000;
105.000, 275.000; 1.00000,
275.000];

```

```

END;
C36 ISA Base::Connection WITH
attributes:
  Generator7.T2 AT Wref;
  bpoints TYPE Matrix [4, 2] := [595.000, 770.000; 565.000,
                                793.000; 375.000, 793.000;
                                375.000, 799.000];

END;
C37 ISA Base::Connection WITH
attributes:
  Bus13 AT Connect113;
  bpoints TYPE Matrix [4, 2] := [375.000, 50.0000; 362.000,
                                25.0000; 225.000, 25.0000;
                                225.000, 1.00000];

END;

C38 ISA Base::Connection WITH
attributes:
  Bus7 AT Connect27;
  bpoints TYPE Matrix [5, 2] := [649.000, 724.000; 625.000,
                                724.000; 51.0000, 724.000;
                                51.0000, 649.000; 1.00000,
                                649.000];

END;
C39 ISA Base::Connection WITH
attributes:
  Bus8 AT Connect38;
  bpoints TYPE Matrix [5, 2] := [575.000, 625.000; 565.000,
                                616.000; 51.0000, 616.000;
                                51.0000, 474.000; 1.00000,
                                474.000];

END;
C40 ISA Base::Connection WITH
attributes:
  Bus11 AT Connect511;
  bpoints TYPE Matrix [5, 2] := [325.000, 225.000; 313.000,
                                202.000; 79.0000, 202.000;
                                79.0000, 75.0000; 1.00000,
                                75.0000];

END;
C41 ISA Base::Connection WITH
attributes:
  Ind_Load10.T1 AT Bus10;
  bpoints TYPE Matrix [2, 2] := [378.000, 272.000; 375.000, 325.000];

END;
C42 ISA Base::Connection WITH
attributes:
  Bus10 AT Line1012.T1;
  bpoints TYPE Matrix [2, 2] := [375.000, 325.000; 407.000, 182.000];

END;
C43 ISA Base::Connection WITH
attributes:
  Bus12 AT Line1012.T2;
  bpoints TYPE Matrix [2, 2] := [400.000, 125.000; 407.000, 182.000];

END;
attribute:
Generator7.Vref.default := 1;
Generator7.Vn.default := 400000;
Generator7.Sn.default := 4e+09;
Generator7.In.default := 10000/sqrt(3);
Generator7.D.default := 0;
Generator7.TdOp.default := 5;
Generator7.H.default := 4;

```

```

Generator7.Xd.default := 1;
Generator7.Xdp.default := 0.2;
Generator7.Xq.default := 0.6;
Generator7.K.default := 20;
Generator7.POSI.default := 2.5e+09;
Generator7.QOSI.default := 9.13297e+06;
Generator7.Theta0.default := 26.9472;
Generator7.Vt0.default := 1;
Ind_Load7.POSI.default := 1.5e+09;
Ind_Load7.QOSI.default := 1e+07;
Ind_Load7.Theta01.default := 26.9472;
Line78.R.default := 0.0005;
Line78.Xl.default := 0.01;
Line78.Theta01.default := 26.9472;
Line78.Theta02.default := 21.2681;
Generator8.Vref.default := 1;
Generator8.Vn.default := 400000;
Generator8.Sn.default := 2.5e+09;
Generator8.In.default := 6250/sqrt(3);
Generator8.D.default := 0;
Generator8.TdOp.default := 5;
Generator8.H.default := 4;
Generator8.Xd.default := 1;
Generator8.Xdp.default := 0.2;
Generator8.Xq.default := 0.6;
Generator8.POSI.default := 1e+09;
Generator8.QOSI.default := 1.40518e+08;
Generator8.Theta0.default := 21.2681;
Generator8.Vt0.default := 1;
Generator8.K.default := 20;
Ind_Load8.POSI.default := 5e+08;
Ind_Load8.QOSI.default := 1e+07;
Ind_Load8.Theta01.default := 21.2681;
Line89.R.default := 0.0005;
Line89.Xl.default := 0.01;
Line89.Theta01.default := 21.2681;
Line89.Theta02.default := 12.9467;
Generator9.Vref.default := 1;
Generator9.Vn.default := 400000;
Generator9.Sn.default := 7e+09;
Generator9.In.default := 17500/sqrt(3);
Generator9.D.default := 0;
Generator9.TdOp.default := 5;
Generator9.H.default := 4;
Generator9.Xd.default := 1;
Generator9.Xdp.default := 0.2;
Generator9.Xq.default := 0.6;
Generator9.K.default := 20;
Generator9.POSI.default := 4e+09;
Generator9.QOSI.default := 4.10264e+08;
Generator9.Theta0.default := 12.9467;
Generator9.Vt0.default := 1;
Ind_Load9.POSI.default := 2e+09;
Ind_Load9.QOSI.default := 1e+07;
Ind_Load9.Theta01.default := 12.9467;
Line910.R.default := 0.0005;
Line910.Xl.default := 0.01;
Line910.Theta01.default := 12.9467;
Line910.Theta02.default := -(7.07345e-18);
Line911.R.default := 0.001;
Line911.Xl.default := 0.02;
Line911.Theta01.default := 12.9467;
Line911.Theta02.default := -(0.678264);

```



```

Generator10.Vref.default := 1;
Generator10.Vn.default := 400000;
Generator10.Sn.default := 4e+09;
Generator10.In.default := 10000/sqrt(3);
Generator10.D.default := 0;
Generator10.TdOp.default := 7;
Generator10.H.default := 5;
Generator10.Xd.default := 2.1;
Generator10.Xdp.default := 0.3;
Generator10.Xq.default := 2;
Generator10.POSI.default := 3.10643e+09;
Generator10.QOSI.default := 4.23423e+08;
Generator10.Theta0.default := -(7.07345e-18);
Generator10.Vt0.default := 1;
Generator10.K.default := 20;
Ind_Load10.POSI.default := 5.8e+09;
Ind_Load10.QOSI.default := 1e+07;
Ind_Load10.Theta01.default := -(7.07345e-18);
Line1011.R.default := 0.003;
Line1011.Xl.default := 0.06;
Line1011.Theta01.default := -(7.07345e-18);
Line1011.Theta02.default := -(0.678264);
Line1012.R.default := 0.001;
Line1012.Xl.default := 0.02;
Line1012.Theta01.default := -(7.07345e-18);
Line1012.Theta02.default := 5.6646;
Generator11.Vref.default := 1;
Generator11.Vn.default := 400000;
Generator11.Sn.default := 4e+09;
Generator11.In.default := 10000/sqrt(3);
Generator11.D.default := 0;
Generator11.TdOp.default := 7;
Generator11.H.default := 5;
Generator11.Xd.default := 2.1;
Generator11.Xdp.default := 0.3;
Generator11.Xq.default := 2;
Generator11.POSI.default := 3e+09;
Generator11.QOSI.default := 2.0524e+08;
Generator11.Theta0.default := -(0.678264);
Generator11.Vt0.default := 1;
Generator11.K.default := 20;
Ind_Load11.POSI.default := 4e+09;
Ind_Load11.QOSI.default := 1e+07;
Ind_Load11.Theta01.default := -(0.678264);
Line1113.R.default := 0.001;
Line1113.Xl.default := 0.02;
Line1113.Theta01.default := -(0.678264);
Line1113.Theta02.default := -(3.06359);
Generator12.Vref.default := 1;
Generator12.Vn.default := 400000;
Generator12.Sn.default := 2.5e+09;
Generator12.In.default := 6250/sqrt(3);
Generator12.D.default := 0;
Generator12.TdOp.default := 7;
Generator12.H.default := 6;
Generator12.Xd.default := 2.1;
Generator12.Xdp.default := 0.3;
Generator12.Xq.default := 2;
Generator12.POSI.default := 2e+09;
Generator12.QOSI.default := 2.30185e+07;
Generator12.Theta0.default := 5.6646;
Generator12.Vt0.default := 1;
Generator12.K.default := 20;

```

```
Ind_Load12.POSI.default := 1e+09;
Ind_Load12.QOSI.default := 1e+07;
Ind_Load12.Theta01.default := 5.6646;
Line1213.R.default := 0.0015;
Line1213.Xl.default := 0.03;
Line1213.Theta01.default := 5.6646;
Line1213.Theta02.default := -(3.06359);
Generator13.Vref.default := 1;
Generator13.Vn.default := 400000;
Generator13.Sn.default := 1.5e+09;
Generator13.In.default := 3750/sqrt(3);
Generator13.D.default := 0;
Generator13.TdOp.default := 7;
Generator13.H.default := 6;
Generator13.Xd.default := 2.1;
Generator13.Xdp.default := 0.3;
Generator13.Xq.default := 2;
Generator13.POSI.default := 1e+09;
Generator13.QOSI.default := 8.79446e+07;
Generator13.Theta0.default := -(3.06359);
Generator13.Vt0.default := 1;
Generator13.K.default := 20;
Ind_Load13.POSI.default := 1.7e+09;
Ind_Load13.QOSI.default := 1e+07;
Ind_Load13.Theta01.default := -(3.06359);
END;
```

```

Danmark ISA Base::Model WITH
attributes:
  Generator1 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 104;
      y_pos := 204;
    END;
  END;
  Ind_Load1 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 177;
      y_pos := 127;
    END;
  END;
  Wref ISA Basic_Parts::Out_Mass_Term WITH
  attributes:
    Graphic ISA Base::Layout WITH
    attributes:
      x_pos := 100;
      y_pos := 300;
    END;
  END;
  T1 ISA Basic_Parts::Bus WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 175;
      y_pos := 175;
    END;
  END;
  Connect113 ISA Basic_Parts::In_El_Term WITH
  attributes:
    Graphic ISA Base::Layout WITH
    attributes:
      x_pos := 400;
      y_pos := 150;
    END;
  END;
  C218 ISA Base::Connection WITH
  attributes:
    T1 AT Ind_Load1.T1;
    bpoints TYPE Matrix [2, 2] := [175.000, 174.000; 176.000, 145.000];
  END;
  C219 ISA Base::Connection WITH
  attributes:
    T1 AT Generator1.T1;
    bpoints TYPE Matrix [3, 2] := [175.000, 174.000; 175.000,
                                   203.000; 103.000, 203.000];
  END;
  C220 ISA Base::Connection WITH
  attributes:
    Wref AT Generator1.T2;
    bpoints TYPE Matrix [3, 2] := [100.000, 300.000; 103.000,
                                   249.000; 103.000, 203.000];
  END;
  C221 ISA Base::Connection WITH
  attributes:
    T1 AT Connect113;
    bpoints TYPE Matrix [3, 2] := [175.000, 174.000; 202.000,

```

```
150.000; 399.000, 150.000];  
END;  
attributes:  
Generator1.Vref.default := 1;  
Generator1.Vn.default := 400000;  
Generator1.Sn.default := 2.5e+09;  
Generator1.In.default := 6250/sqrt(3);  
Generator1.D.default := 0;  
Generator1.Td0p.default := 7;  
Generator1.H.default := 5;  
Generator1.Xd.default := 2.1;  
Generator1.Xdp.default := 0.3;  
Generator1.Xq.default := 2;  
Generator1.K.default := 20;  
Generator1.POSI.default := 1.79e+09;  
Generator1.QOSI.default := 1.05101e+07;  
Generator1.Theta0.default := -(3.17847);  
Generator1.Vt0.default := 1;  
Ind_Load1.POSI.default := 1.8e+09;  
Ind_Load1.QOSI.default := 1e+07;  
Ind_Load1.Theta01.default := -3.17847;  
END;
```

```

Finland ISA Basic_Parts::VeryBigModel WITH
attributes:
  Generator14 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 252;
      y_pos := 502;
    END;
  END;
  Ind_Load14 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 303;
      y_pos := 428;
    END;
  END;
  Line1416 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 328;
      y_pos := 278;
    END;
  END;
  Generator16 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 403;
      y_pos := 103;
    END;
  END;
  Ind_Load16 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 328;
      y_pos := 53;
    END;
  END;
  Line1516 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 278;
      y_pos := 128;
    END;
  END;
  Ind_Load15 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 203;
      y_pos := 178;
    END;
  END;
  Generator15 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 128;

```

```

        y_pos := 228;
    END;
END;
Line1415 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 253;
        y_pos := 328;
    END;
END;
Bus14 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 300;
        y_pos := 450;
    END;
END;
Bus16 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 325;
        y_pos := 75;
    END;
END;
Bus15 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 200;
        y_pos := 200;
    END;
END;
Wref ISA Basic_Parts::Out_Mass_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 250;
        y_pos := 800;
    END;
END;
Connect714 ISA Basic_Parts::In_El_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 1;
        y_pos := 475;
    END;
END;
C202 ISA Base::Connection WITH
attributes:
    Bus16 AT Ind_Load16.T1;
    bpoints TYPE Matrix [2, 2] := [325.000, 75.0000; 327.000, 71.0000];
END;
C203 ISA Base::Connection WITH
attributes:
    Bus16 AT Generator16.T1;
    bpoints TYPE Matrix [3, 2] := [325.000, 75.0000; 354.000,
                                102.000; 402.000, 102.000];
END;
C204 ISA Base::Connection WITH

```

```

attributes:
  Bus16 AT Line1516.T2;
  bpoints TYPE Matrix [3, 2] := [325.000, 75.0000; 277.000,
                                103.000; 277.000, 127.000];
END;
C205 ISA Base::Connection WITH
attributes:
  Bus15 AT Line1516.T1;
  bpoints TYPE Matrix [3, 2] := [200.000, 200.000; 277.000,
                                152.000; 277.000, 127.000];
END;
C206 ISA Base::Connection WITH
attributes:
  Bus15 AT Ind_Load15.T1;
  bpoints TYPE Matrix [2, 2] := [200.000, 200.000; 202.000, 196.000];
END;
C207 ISA Base::Connection WITH
attributes:
  Bus14 AT Ind_Load14.T1;
  bpoints TYPE Matrix [2, 2] := [300.000, 449.000; 302.000, 446.000];
END;
C208 ISA Base::Connection WITH
attributes:
  Bus15 AT Generator15.T1;
  bpoints TYPE Matrix [3, 2] := [200.000, 200.000; 175.000,
                                227.000; 127.000, 227.000];
END;
C209 ISA Base::Connection WITH
attributes:
  Bus15 AT Line1415.T2;
  bpoints TYPE Matrix [3, 2] := [200.000, 200.000; 252.000,
                                230.000; 252.000, 327.000];
END;
C210 ISA Base::Connection WITH
attributes:
  Bus16 AT Line1416.T2;
  bpoints TYPE Matrix [2, 2] := [325.000, 75.0000; 327.000, 277.000];
END;
C211 ISA Base::Connection WITH
attributes:
  Bus14 AT Line1415.T1;
  bpoints TYPE Matrix [3, 2] := [300.000, 449.000; 252.000,
                                449.000; 252.000, 327.000];
END;
C212 ISA Base::Connection WITH
attributes:
  Bus14 AT Line1416.T1;
  bpoints TYPE Matrix [5, 2] := [300.000, 449.000; 300.000,
                                330.000; 300.000, 403.000;
                                327.000, 451.000; 327.000,
                                277.000];
END;
C213 ISA Base::Connection WITH
attributes:
  Generator16.T2 AT Generator15.T2;
  bpoints TYPE Matrix [4, 2] := [402.000, 102.000; 402.000,
                                18.0000; 127.000, 18.0000;
                                127.000, 227.000];
END;
C214 ISA Base::Connection WITH
attributes:
  Bus14 AT Generator14.T1;
  bpoints TYPE Matrix [4, 2] := [300.000, 449.000; 300.000,

```

```

501.000; 301.000, 501.000;
251.000, 501.000];
END;
C215 ISA Base::Connection WITH
attributes:
  Bus14 AT Connect714;
  bpoints TYPE Matrix [3, 2] := [300.000, 449.000; 276.000,
474.000; 1.00000, 474.000];
END;
C216 ISA Base::Connection WITH
attributes:
  Generator14.T2 AT Generator16.T2;
  bpoints TYPE Matrix [4, 2] := [251.000, 501.000; 251.000,
528.000; 402.000, 528.000;
402.000, 102.000];
END;
C217 ISA Base::Connection WITH
attributes:
  Generator14.T2 AT Wref;
  bpoints TYPE Matrix [3, 2] := [251.000, 501.000; 250.000,
649.000; 250.000, 800.000];
END;
attributes:
Generator14.Vref.default := 1;
Generator14.Vn.default := 400000;
Generator14.Sn.default := 2e+09;
Generator14.In.default := 5000/sqrt(3);
Generator14.D.default := 0;
Generator14.TdOp.default := 5;
Generator14.H.default := 3;
Generator14.Xd.default := 1;
Generator14.Xdp.default := 0.2;
Generator14.Xq.default := 0.6;
Generator14.POSI.default := 1e+09;
Generator14.QOSI.default := 1.7213e+07;
Generator14.Theta0.default := 26.9369;
Generator14.Vt0.default := 1;
Generator14.K.default := 20;
Ind_Load14.POSI.default := 1.1e+09;
Ind_Load14.QOSI.default := 9.99998e+06;
Ind_Load14.Theta01.default := 26.9369;
Line1416.R.default := 0.0035;
Line1416.Xl.default := 0.07;
Line1416.Theta01.default := 26.9369;
Line1416.Theta02.default := 27.1829;
Generator16.Vref.default := 1;
Generator16.Vn.default := 400000;
Generator16.Sn.default := 3e+09;
Generator16.In.default := 7500/sqrt(3);
Generator16.D.default := 0;
Generator16.TdOp.default := 7;
Generator16.H.default := 6;
Generator16.Xd.default := 2.1;
Generator16.Xdp.default := 0.3;
Generator16.Xq.default := 2;
Generator16.K.default := 20;
Generator16.POSI.default := 2.5e+09;
Generator16.QOSI.default := 1.72826e+07;
Generator16.Theta0.default := 27.1829;
Generator16.Vt0.default := 1;
Ind_Load16.POSI.default := 2.6e+09;
Ind_Load16.QOSI.default := 1e+07;
Ind_Load16.Theta01.default := 27.1829;

```



```
Line1516.R.default := 0.002;
Line1516.Xl.default := 0.04;
Line1516.Theta01.default := 29.6244;
Line1516.Theta02.default := 27.1829;
Generator15.Vref.default := 1;
Generator15.Vn.default := 400000;
Generator15.Sn.default := 4e+09;
Generator15.In.default := 10000/sqrt(3);
Generator15.D.default := 0;
Generator15.TdOp.default := 7;
Generator15.H.default := 6;
Generator15.Xd.default := 2.1;
Generator15.Xdp.default := 0.3;
Generator15.Xq.default := 2;
Generator15.POSI.default := 3e+09;
Generator15.QOSI.default := 4.46921e+06;
Generator15.Theta0.default := 29.6244;
Generator15.Vt0.default := 1;
Generator15.K.default := 20;
Ind_Load15.POSI.default := 2.8e+09;
Ind_Load15.QOSI.default := 9.99997e+06;
Ind_Load15.Theta01.default := 29.6244;
Line1415.R.default := 0.0025;
Line1415.Xl.default := 0.05;
Line1415.Theta01.default := 26.9369;
Line1415.Theta02.default := 29.6244;
END;
```

```

Norge ISA Basic_Parts::VeryBigModel WITH
attributes:
  Generator2 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 407;
      y_pos := 582;
    END;
  END;
  Generator3 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 232;
      y_pos := 407;
    END;
  END;
  Generator4 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 182;
      y_pos := 282;
    END;
  END;
  Generator5 ISA Generators::Generator WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 207;
      y_pos := 182;
    END;
  END;
  Ind_Load2 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 482;
      y_pos := 532;
    END;
  END;
  Ind_Load3 ISA Impedances::Ind_Load WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 307;
      y_pos := 357;
    END;
  END;
  Line34 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 252;
      y_pos := 327;
    END;
  END;
  Line45 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 282;

```

```

        y_pos := 207;
    END;
END;
Ind_Load5 ISA Impedances::Ind_Load WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 281;
        y_pos := 131;
    END;
END;
Ind_Load4 ISA Impedances::Ind_Load WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 256;
        y_pos := 231;
    END;
END;
Generator6 ISA Generators::Generator WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 178;
        y_pos := 78;
    END;
END;
Ind_Load6 ISA Impedances::Ind_Load WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 253;
        y_pos := 28;
    END;
END;
Line56 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 253;
        y_pos := 103;
    END;
END;
Bus2 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 475;
        y_pos := 550;
    END;
END;
Bus3 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 300;
        y_pos := 375;
    END;
END;
Bus4 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:

```

```

        x_pos := 250;
        y_pos := 250;
    END;
END;
Wref ISA Basic_Parts::Out_Mass_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 400;
        y_pos := 800;
    END;
END;
Bus5 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 275;
        y_pos := 150;
    END;
END;
Bus6 ISA Basic_Parts::Bus WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 251;
        y_pos := 51;
    END;
END;
Connect27 ISA Basic_Parts::In_El_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 1000;
        y_pos := 675;
    END;
END;
Connect38 ISA Basic_Parts::In_El_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 1000;
        y_pos := 475;
    END;
END;
Connect49 ISA Basic_Parts::In_El_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 1000;
        y_pos := 275;
    END;
END;
Connect511 ISA Basic_Parts::In_El_Term WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 1000;
        y_pos := 50;
    END;
END;
C1 ISA Base::Connection WITH
attributes:
    Bus5 AT Line45.T2;

```

```

    bpoints TYPE Matrix [2, 2] := [275.000, 150.000; 281.000, 206.000];
END;
C2 ISA Base::Connection WITH
attributes:
    Bus4 AT Line45.T1;
    bpoints TYPE Matrix [3, 2] := [250.000, 250.000; 281.000,
                                   250.000; 281.000, 206.000];
END;
C3 ISA Base::Connection WITH
attributes:
    Bus5 AT Generator5.T1;
    bpoints TYPE Matrix [3, 2] := [275.000, 150.000; 256.000,
                                   181.000; 206.000, 181.000];
END;
C4 ISA Base::Connection WITH
attributes:
    Bus4 AT Generator4.T1;
    bpoints TYPE Matrix [3, 2] := [250.000, 250.000; 234.000,
                                   281.000; 181.000, 281.000];
END;
C5 ISA Base::Connection WITH
attributes:
    Bus3 AT Ind_Load3.T1;
    bpoints TYPE Matrix [2, 2] := [300.000, 375.000; 306.000, 375.000];
END;
C6 ISA Base::Connection WITH
attributes:
    Bus3 AT Generator3.T1;
    bpoints TYPE Matrix [3, 2] := [300.000, 375.000; 285.000,
                                   406.000; 231.000, 406.000];
END;
C7 ISA Base::Connection WITH
attributes:
    Bus2 AT Generator2.T1;
    bpoints TYPE Matrix [3, 2] := [475.000, 549.000; 458.000,
                                   581.000; 406.000, 581.000];
END;
C8 ISA Base::Connection WITH
attributes:
    Bus2 AT Ind_Load2.T1;
    bpoints TYPE Matrix [2, 2] := [475.000, 549.000; 481.000, 550.000];
END;
C9 ISA Base::Connection WITH
attributes:
    Bus4 AT Ind_Load4.T1;
    bpoints TYPE Matrix [2, 2] := [250.000, 250.000; 255.000, 249.000];
END;
C10 ISA Base::Connection WITH
attributes:
    Bus5 AT Ind_Load5.T1;
    bpoints TYPE Matrix [2, 2] := [275.000, 150.000; 280.000, 149.000];
END;
C11 ISA Base::Connection WITH
attributes:
    Generator5.T2 AT Generator4.T2;
    bpoints TYPE Matrix [4, 2] := [206.000, 181.000; 157.000,
                                   181.000; 157.000, 281.000;
                                   181.000, 281.000];
END;
C12 ISA Base::Connection WITH
attributes:
    Generator4.T2 AT Generator3.T2;
    bpoints TYPE Matrix [4, 2] := [181.000, 281.000; 158.000,

```

```

281.000; 158.000, 406.000;
231.000, 406.000];
END;
C13 ISA Base::Connection WITH
attributes:
  Generator3.T2 AT Generator2.T2;
  bpoints TYPE Matrix [4, 2] := [231.000, 406.000; 160.000,
406.000; 160.000, 581.000;
406.000, 581.000];
END;
C14 ISA Base::Connection WITH
attributes:
  Bus6 AT Generator6.T1;
  bpoints TYPE Matrix [3, 2] := [251.000, 51.0000; 233.000,
77.0000; 177.000, 77.0000];
END;
C15 ISA Base::Connection WITH
attributes:
  Bus6 AT Line56.T2;
  bpoints TYPE Matrix [2, 2] := [251.000, 51.0000; 252.000, 102.000];
END;
C16 ISA Base::Connection WITH
attributes:
  Bus6 AT Ind_Load6.T1;
  bpoints TYPE Matrix [2, 2] := [251.000, 51.0000; 252.000, 46.0000];
END;
C17 ISA Base::Connection WITH
attributes:
  Bus5 AT Line56.T1;
  bpoints TYPE Matrix [3, 2] := [275.000, 150.000; 252.000,
137.000; 252.000, 102.000];
END;
C18 ISA Base::Connection WITH
attributes:
  Generator6.T2 AT Generator5.T2;
  bpoints TYPE Matrix [4, 2] := [177.000, 77.0000; 177.000,
126.000; 206.000, 126.000;
206.000, 181.000];
END;
C19 ISA Base::Connection WITH
attributes:
  Bus4 AT Line34.T2;
  bpoints TYPE Matrix [2, 2] := [250.000, 250.000; 251.000, 326.000];
END;
C20 ISA Base::Connection WITH
attributes:
  Bus3 AT Line34.T1;
  bpoints TYPE Matrix [3, 2] := [300.000, 375.000; 251.000,
350.000; 251.000, 326.000];
END;
C21 ISA Base::Connection WITH
attributes:
  Bus3 AT Connect38;
  bpoints TYPE Matrix [3, 2] := [300.000, 375.000; 300.000,
474.000; 999.000, 474.000];
END;
C22 ISA Base::Connection WITH
attributes:
  Bus4 AT Connect49;
  bpoints TYPE Matrix [3, 2] := [250.000, 250.000; 278.000,
275.000; 999.000, 275.000];
END;
C23 ISA Base::Connection WITH

```

```

attributes:
  Bus2 AT Connect27;
  bpoints TYPE Matrix [3, 2] := [475.000, 549.000; 475.000,
                                674.000; 999.000, 674.000];
END;
C24 ISA Base::Connection WITH
attributes:
  Bus5 AT Connect511;
  bpoints TYPE Matrix [3, 2] := [275.000, 150.000; 428.000,
                                50.0000; 999.000, 50.0000];
END;
C25 ISA Base::Connection WITH
attributes:
  Generator2.T2 AT Wref;
  bpoints TYPE Matrix [3, 2] := [406.000, 581.000; 400.000,
                                627.000; 400.000, 799.000];
END;
attributes:
Generator2.Vref.default := 1;
Generator2.Vn.default := 400000;
Generator2.Sn.default := 1e+09;
Generator2.In.default := 2500/sqrt(3);
Generator2.D.default := 0;
Generator2.Td0p.default := 5;
Generator2.H.default := 3;
Generator2.Xd.default := 1;
Generator2.Xdp.default := 0.2;
Generator2.Xq.default := 0.6;
Generator2.K.default := 20;
Generator2.POSI.default := 7.9e+08;
Generator2.QOSI.default := 1.05503e+07;
Generator2.Theta0.default := 26.3726;
Generator2.Vt0.default := 1;
Ind_Load2.POSI.default := 8e+08;
Ind_Load2.QOSI.default := 1e+07;
Ind_Load2.Theta01.default := 26.3726;
Generator3.Vref.default := 1;
Generator3.Vn.default := 400000;
Generator3.Sn.default := 2.5e+09;
Generator3.In.default := 6250/sqrt(3);
Generator3.D.default := 0;
Generator3.Td0p.default := 5;
Generator3.H.default := 3;
Generator3.Xd.default := 1;
Generator3.Xdp.default := 0.2;
Generator3.Xq.default := 0.6;
Generator3.POSI.default := 1.5e+09;
Generator3.QOSI.default := 1.261e+07;
Generator3.Theta0.default := 17.5743;
Generator3.Vt0.default := 1;
Generator3.K.default := 20;
Ind_Load3.POSI.default := 1.4e+09;
Ind_Load3.QOSI.default := 9.99997e+06;
Ind_Load3.Theta01.default := 17.5743;
Line34.R.default := 0.0035;
Line34.Xl.default := 0.07;
Line34.Theta01.default := 17.5743;
Line34.theta02.default := 12.1251;
Generator4.Vref.default := 1;
Generator4.Vn.default := 400000;
Generator4.Sn.default := 2.5e+09;
Generator4.In.default := 6250/sqrt(3);
Generator4.D.default := 0;

```

```

Generator4.TdOp.default := 5;
Generator4.H.default := 3;
Generator4.Xd.default := 1;
Generator4.Xdp.default := 0.2;
Generator4.Xq.default := 0.6;
Generator4.K.default := 20;
Generator4.POSI.default := 1.79e+09;
Generator4.QOSI.default := 3.14553e+07;
Generator4.Theta0.default := 12.1251;
Generator4.Vt0.default := 1;
Ind_Load4.POSI.default := 1.8e+09;
Ind_Load4.QOSI.default := 1e+07;
Ind_Load4.Theta01.default := 12.1251;
Line45.R.default := 0.008;
Line45.Xl.default := 0.16;
Line45.Theta01.default := 12.1251;
Line45.theta02.default := -0.212956;
Generator5.Vref.default := 1;
Generator5.Vn.default := 400000;
Generator5.Sn.default := 7e+09;
Generator5.In.default := 17500/sqrt(3);
Generator5.D.default := 0;
Generator5.TdOp.default := 5;
Generator5.H.default := 3;
Generator5.Xd.default := 1;
Generator5.Xdp.default := 0.2;
Generator5.Xq.default := 0.6;
Generator5.POSI.default := 4.5e+09;
Generator5.QOSI.default := 1.97229e+08;
Generator5.Theta0.default := -(0.212956);
Generator5.Vt0.default := 1;
Generator5.K.default := 20;
Ind_Load5.POSI.default := 5.5e+09;
Ind_Load5.QOSI.default := 1e+07;
Ind_Load5.Theta01.default := -0.212956;
Generator6.Vref.default := 1;
Generator6.Vn.default := 400000;
Generator6.Sn.default := 7e+09;
Generator6.In.default := 17500/sqrt(3);
Generator6.D.default := 0;
Generator6.TdOp.default := 5;
Generator6.H.default := 3;
Generator6.Xd.default := 1;
Generator6.Xdp.default := 0.2;
Generator6.Xq.default := 0.6;
Generator6.POSI.default := 4.4e+09;
Generator6.QOSI.default := 8.77119e+07;
Generator6.Theta0.default := 15.382;
Generator6.Vt0.default := 1;
Generator6.K.default := 20;
Ind_Load6.POSI.default := 3.5e+09;
Ind_Load6.QOSI.default := 9.99998e+06;
Ind_Load6.Theta01.default := 15.3820;
Line56.R.default := 0.0015;
Line56.Xl.default := 0.03;
Line56.Theta01.default := -0.212956;
Line56.theta02.default := 15.3820;
END;

```



```

Norden1 ISA Basic_Parts::VeryBigModel WITH
attributes:
  Sverige ISA Generators::Sverige WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 532;
      y_pos := 482;
    END;
  END;
  Danmark ISA Generators::Danmark WITH
  attributes:
    Graphic ISA Base::Layout WITH
    attributes:
      x_pos := 354;
      y_pos := 254;
    END;
  END;
  Norge ISA Generators::Norge WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 229;
      y_pos := 479;
    END;
  END;
  Finland ISA Generators::Finland WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 867;
      y_pos := 582;
    END;
  END;
  Line13 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 506;
      y_pos := 406;
    END;
  END;
  Line27 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 379;
      y_pos := 529;
    END;
  END;
  Line38 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 379;
      y_pos := 504;
    END;
  END;
  Line49 ISA Impedances::Line WITH
  attributes:
    Graphic ISA super::Graphic WITH
    attributes:
      x_pos := 379;

```

```

        y_pos := 479;
    END;
END;
Line511 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 427;
        y_pos := 452;
    END;
END;
Line714 ISA Impedances::Line WITH
attributes:
    Graphic ISA super::Graphic WITH
    attributes:
        x_pos := 679;
        y_pos := 579;
    END;
END;
Wref ISA Basic_Parts::InertiaBus WITH
attributes:
    Graphic ISA Base::Layout WITH
    attributes:
        x_pos := 476;
        y_pos := 776;
    END;
END;
C1 ISA Base::Connection WITH
attributes:
    Sverige.Connect113 AT Line113.T1;
    bpoints TYPE Matrix [2, 2] := [503.000, 444.000; 505.000, 405.000];
END;
C2 ISA Base::Connection WITH
attributes:
    Wref AT Finland.Wref;
    bpoints TYPE Matrix [3, 2] := [476.000, 775.000; 831.000,
                                   775.000; 831.000, 618.000];
END;
C3 ISA Base::Connection WITH
attributes:
    Wref AT Sverige.Wref;
    bpoints TYPE Matrix [3, 2] := [476.000, 775.000; 476.000,
                                   556.000; 518.000, 518.000];
END;
C4 ISA Base::Connection WITH
attributes:
    Line27.T1 AT Sverige.Connect27;
    bpoints TYPE Matrix [3, 2] := [378.000, 528.000; 456.000,
                                   528.000; 481.000, 504.000];
END;
C5 ISA Base::Connection WITH
attributes:
    Norge.Connect38 AT Line38.T2;
    bpoints TYPE Matrix [3, 2] := [277.000, 484.000; 305.000,
                                   503.000; 378.000, 503.000];
END;
C6 ISA Base::Connection WITH
attributes:
    Line38.T1 AT Sverige.Connect38;
    bpoints TYPE Matrix [3, 2] := [378.000, 503.000; 456.000,
                                   503.000; 481.000, 487.000];
END;
C7 ISA Base::Connection WITH

```

```

attributes:
  Norge.Connect49 AT Line49.T2;
  bpoints TYPE Matrix [3, 2] := [277.000, 466.000; 303.000,
                                478.000; 378.000, 478.000];
END;
C8 ISA Base::Connection WITH
attributes:
  Line49.T1 AT Sverige.Connect49;
  bpoints TYPE Matrix [3, 2] := [378.000, 478.000; 457.000,
                                469.000; 481.000, 469.000];
END;
C9 ISA Base::Connection WITH
attributes:
  Line113.T2 AT Danmark.Connect113;
  bpoints TYPE Matrix [3, 2] := [505.000, 405.000; 505.000,
                                253.000; 402.000, 253.000];
END;
C10 ISA Base::Connection WITH
attributes:
  Norge.Connect27 AT Line27.T2;
  bpoints TYPE Matrix [3, 2] := [277.000, 503.000; 304.000,
                                528.000; 378.000, 528.000];
END;
C11 ISA Base::Connection WITH
attributes:
  Line714.T1 AT Finland.Connect714;
  bpoints TYPE Matrix [4, 2] := [678.000, 578.000; 727.000,
                                578.000; 727.000, 587.000;
                                806.000, 587.000];
END;
C12 ISA Base::Connection WITH
attributes:
  Line714.T2 AT Sverige.Connect714;
  bpoints TYPE Matrix [4, 2] := [678.000, 578.000; 631.000,
                                578.000; 631.000, 513.000;
                                580.000, 513.000];
END;
C13 ISA Base::Connection WITH
attributes:
  Danmark.Wref AT Norge.Wref;
  bpoints TYPE Matrix [6, 2] := [328.000, 290.000; 328.000,
                                380.000; 127.000, 380.000;
                                127.000, 530.000; 200.000,
                                530.000; 218.000, 515.000];
END;
C14 ISA Base::Connection WITH
attributes:
  Wref AT Norge.Wref;
  bpoints TYPE Matrix [5, 2] := [476.000, 775.000; 234.000,
                                530.000; 248.000, 530.000;
                                218.000, 777.000; 218.000,
                                515.000];
END;
C16 ISA Base::Connection WITH
attributes:
  Line511.T2 AT Sverige.Connect511;
  bpoints TYPE Matrix [2, 2] := [426.000, 451.000; 481.000, 450.000];
END;
C17 ISA Base::Connection WITH
attributes:
  Line511.T1 AT Norge.Connect511;
  bpoints TYPE Matrix [3, 2] := [426.000, 451.000; 303.000,
                                445.000; 277.000, 445.000];

```

```
END;
attributes:
Line113.R.default := 0.001;
Line113.Xl.default := 0.02;
Line113.Theta01.default := -3.06359;
Line113.Theta02.default := -3.17847;
Line27.R.default := 0.005;
Line27.Xl.default := 0.1;
Line27.Theta01.default := 26.9472;
Line27.Theta02.default := 26.3726;
Line38.R.default := 0.009;
Line38.Xl.default := 0.18;
Line38.Theta01.default := 21.2681;
Line38.Theta02.default := 17.5743;
Line49.R.default := 0.008;
Line49.Xl.default := 0.16;
Line49.Theta01.default := 12.9467;
Line49.Theta02.default := 12.1251;
Line511.R.default := 0.0020;
Line511.Xl.default := 0.040;
Line511.Theta01.default := -0.212956;
Line511.Theta02.default := -0.678264;
Line714.R.default := 0.002;
Line714.Xl.default := 0.04;
Line714.Theta01.default := 26.9369;
Line714.Theta02.default := 26.9472;
END;
```

```

DisconLine ISA Basic_Parts::TwoPort WITH
% This model is used to make a break in a 3-phase transmission
% line between bus 10 and line 1012.
% The model contains a line and a interrupt in series.
% Starttime of fault is 1 sek.
Submodels:
    Line1 ISA impedances::Line;

    Interrupt1 ISA impedances::Interrupt;
Connections:
    T1 AT Interrupt1.T1;
    Interrupt1.T2 AT Line1.T1;
    Line1.T2 AT T2;
Parameters:
    R ISA Parameter;
    X1 ISA Parameter;
    Theta01 ISA Parameter;
    Theta02 ISA Parameter;

Equations:
    Line1.R := R;
    Line1.X1 := X1;
    Line1.Theta01 := Theta01;
    Line1.Theta02 := Theta02;
END;

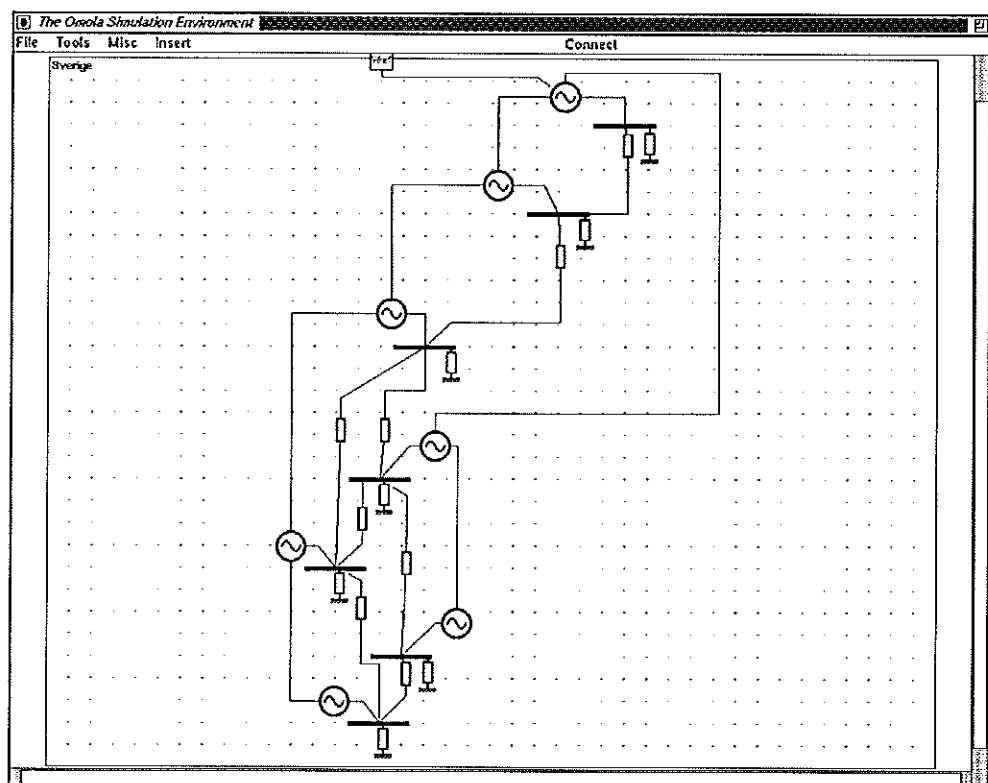
Ex11 ISA Norden1 WITH
    Sverige ISA super::Sverige WITH
        Line1012 ISA DisconLine;
    END;
END;

```

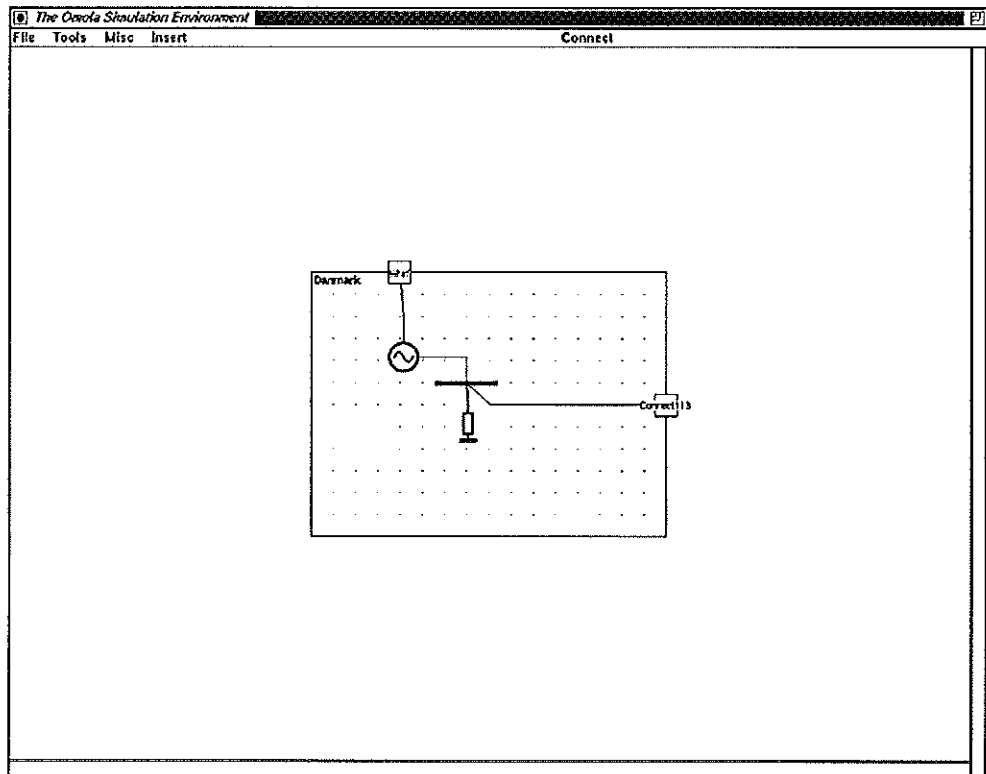
```
Ex13 ISA Norden1 WITH
% This model is used to make a step in the load at bus 10.
% the step is 10% of the initialization load. Starttime of
% step is 1 sek.
  Sverige ISA super::Sverige WITH
    Ind_Load10 ISA impedances::StepLoad;
  END;
END;
```

Appendix C

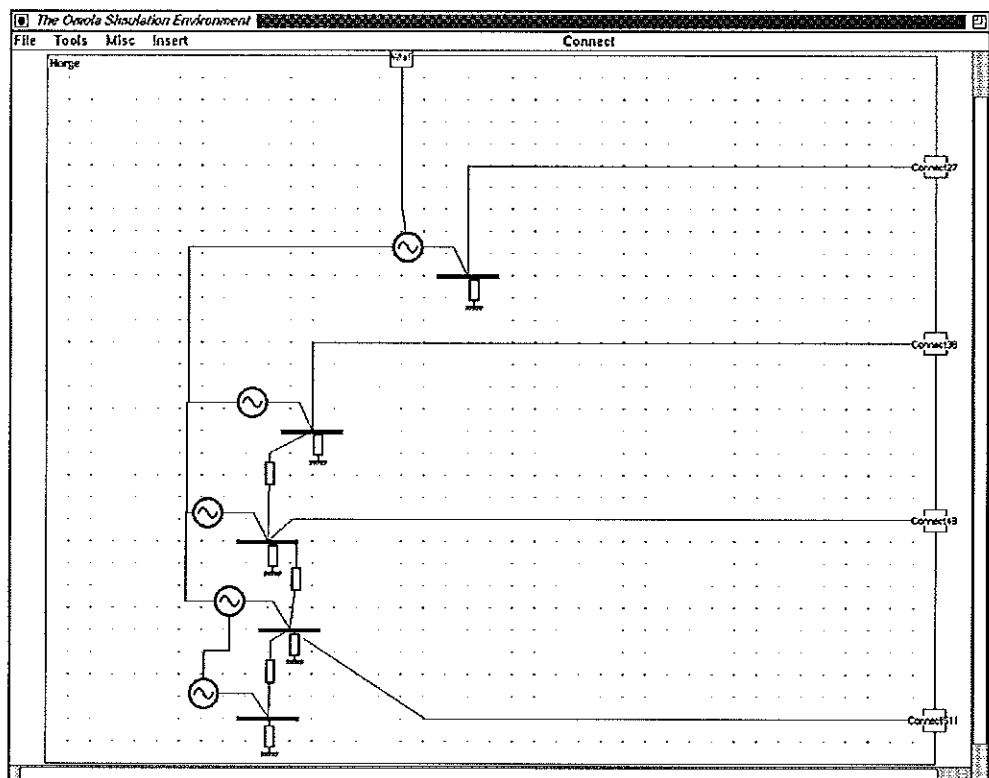
Grafisk representation av Nordel - modellen



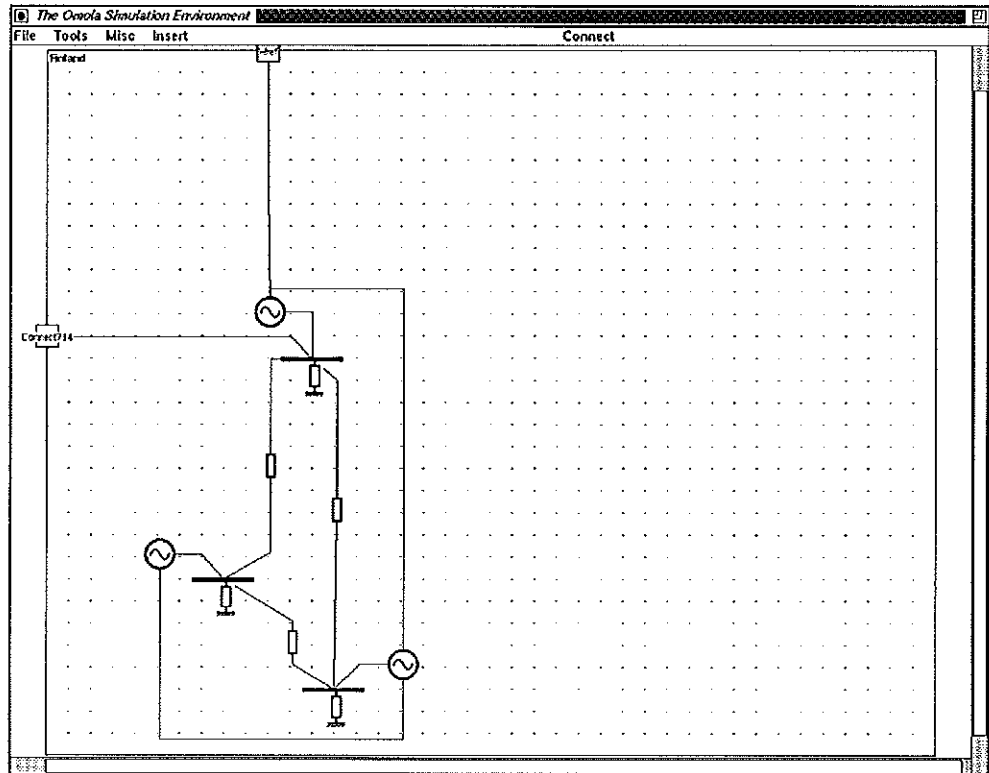
Figur 1. Den svenska delen av 16-maskinmodellen i Omola



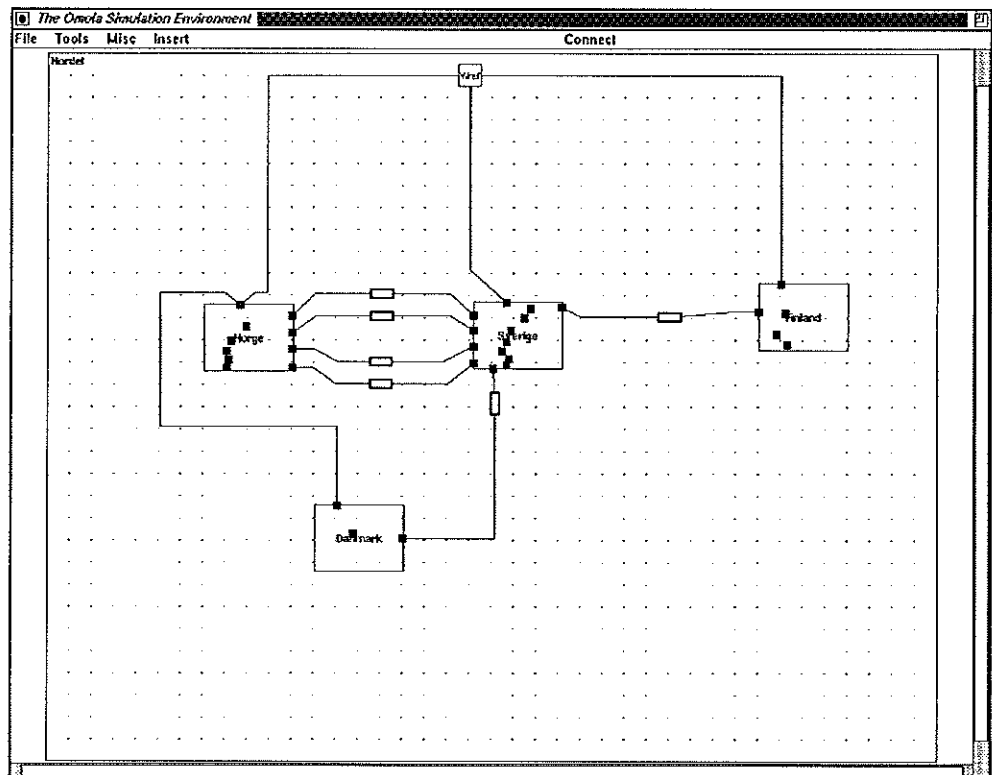
Figur 2. Den danska delen av 16-maskinmodellen i Omola



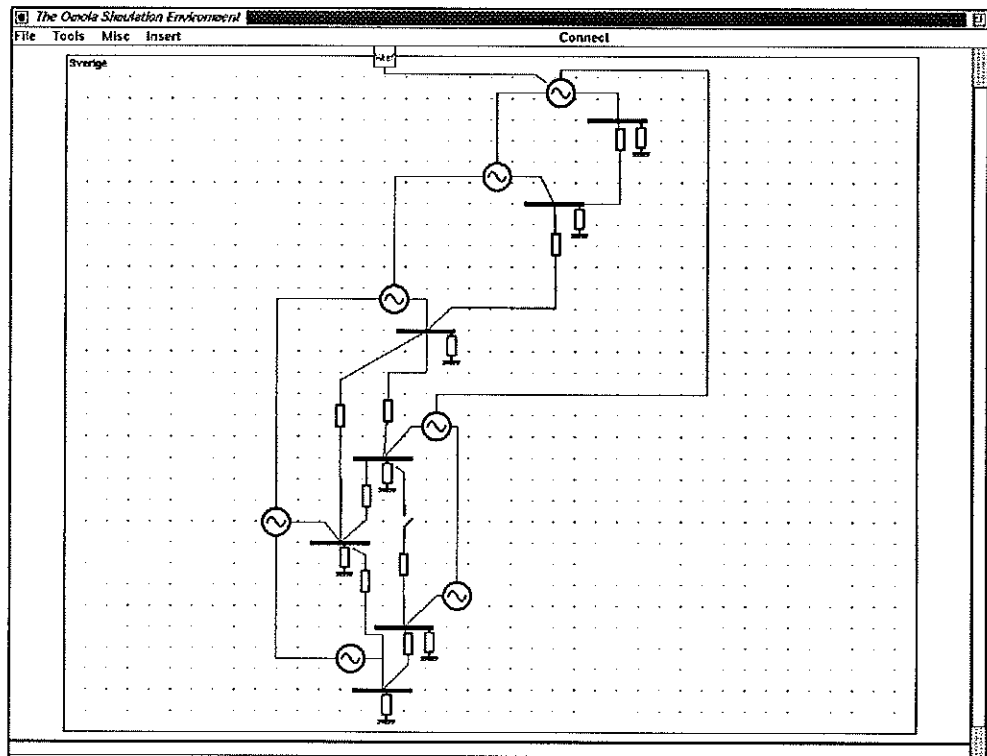
Figur 3. Den norska delen av 16-maskinmodellen i Omola



Figur 4. Den finska delen av 16-maskinmodellen i Omola



Figur 5. Hela 16-maskinmodellen för Nordel i Omola



Figur 6. Den svenska delen av 16-maskinmodellen i Omola med avbrott mellan bus 10 och 12