

# INTELLIGENT ELECTRICAL POWER GRIDS

## MODELLING OF POWER SYSTEMS USING OPENMODELICA



LAST UPDATED ON 23-03-2020

### 1. OBJECTIVES AND PREPARATION

This tutorial is meant to give an insight into power system modelling using an open source software based on Modelica language, OpenModelica. The user is assumed to have basic knowledge of power system analysis.

The objective of this assignment is twofold. Firstly, user is asked to create a simple single line diagram of the IEEE 9 bus test system described in [1] without any dynamic models of generators, which serves to give an introduction to power system modelling with OpenModelica. The created system is initialized with values obtained by running power flow on MatPower as demonstrated in the lectures. Secondly, AVR models are added to generator systems. This will show how to control dynamic response of systems in event of disturbances. At the end of this tutorial, the user should be comfortable in creating power system models in OpenModelica, initializing them with MatPower, and run basic time domain simulations.

### 2. STARTING OPENMODELICA

To start OpenModelica, go to start  and select OpenModelica Connection Editor . This opens up the main window of the OpenModelica software (hereafter referred to as OM), as in Figure1. This is main OM window. On the left panel, a list of default libraries available can be seen. On top, below the menu bar, is the tools bar. User can select various functions like Simulation Setup, Simulation Debug, Model Instantiation with single clicks. These functions can also be selected from Menu bar.

OM provides a host of free libraries to model systems, not just electrical, but also thermal, mechanical, control, etc. For developing the IEEE 9 bus system we will use Open iTesla Power System Library (**OpenIPSL**) [2], and few components from a library developed at TU Delft named **DelMod**.

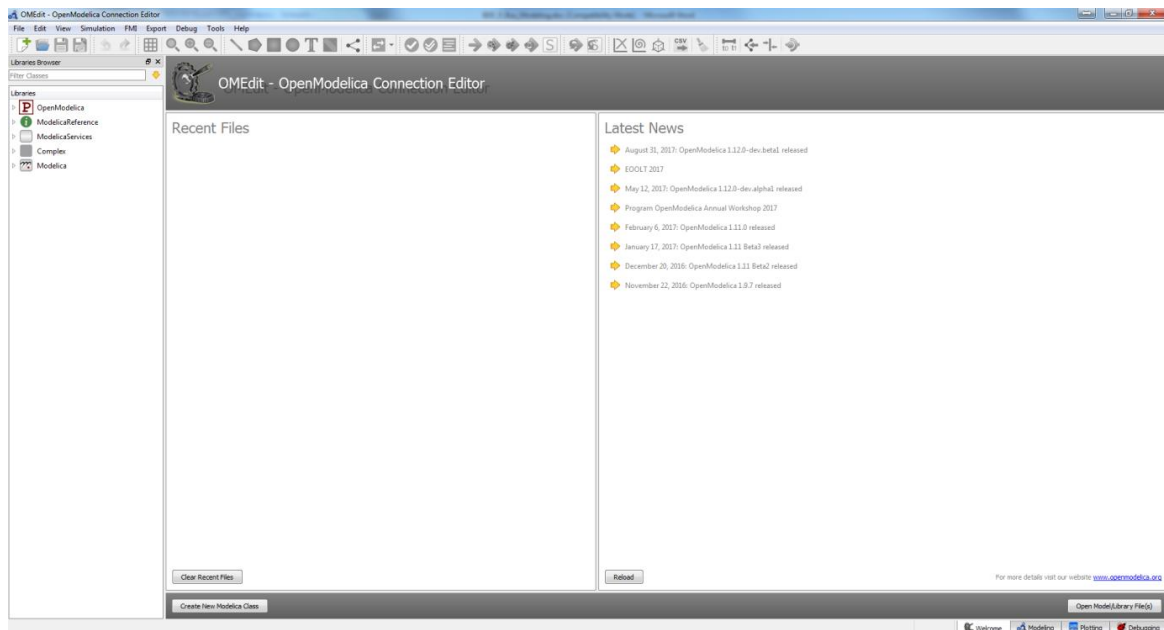


Figure 1: OpenModelica Interface


## Libraries

On the left pane, one can notice there are 5 libraries loaded: OpenModelica, ModelicaReference, ModelicaServices, Complex, and Modelica. These are basic pre-loaded libraries. These contain essential functions and basic models. To load OpenIPSL, select **File>System Libraries> OpenIPSL**. On the left pane, OpenIPSL should be loaded. To load the DelMod library, it needs to be downloaded. DelMod is available simulation companion guide. Once downloaded, load the *package.mo* file using **File>Open Model/Library file(s)**. Now we have the required libraries to model our system.

## 3. TEST SYSTEM

We shall study the IEEE 9 bus test system throughout the lab sessions, as it represents a simple yet realistic example of a multi-machine transmission network. The single line diagram which represents the topology of the IEEE 9 bus test system is shown in Figure 2. This test case consists of 9 buses, 3 generators, 3 two-winding power transformers, 6 lines and 3 loads.



3. Your OM screen may look like that in Figure 3 or not. Figure 3 shows the equation mode. You can easily change your view to the one in figure by selecting Diagram View (by clicking on the  icon) as shown in figure 3. This will change the OM interface to graphical modelling mode as in Figure 4. (Sometimes, OM window will open directly on the graphical mode). This will now be referred to as *network model workspace*.

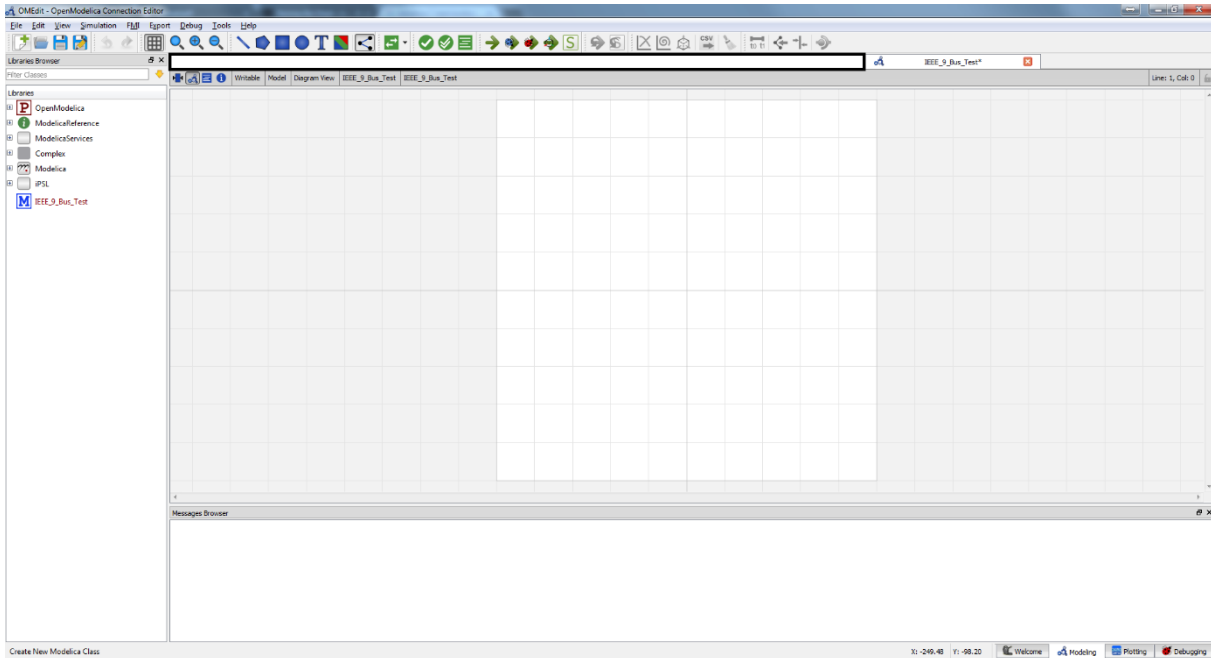


Figure 4: OpenModelica graphical modelling interface

Now that we have created a new project, we need to load components from the library. The OpenIPSL library has models of components that are validated against various other power system analysis softwares like PSSE, PSAT etc. Here, we use models that are validated against PSAT [3].

## Network Creation

The main components to be used for creating the 9 bus network are:

- **Bus:** OpenIPSL → Electrical → Buses → Bus
- **Generators:** OpenIPSL → Electrical → Machines → PSAT → Order4
- **Loads:** OpenIPSL → Electrical → Loads → PSAT → VoltDependant
- **Lines:** OpenIPSL → Electrical → Branches → PwLine
- **Transformer:** OpenIPSL → Electrical → Branches → PSAT → TwoWindingTransformer

Using the components mentioned above, create the network as in Figure 2, without creating generators yet. The data for the system will be filled in later.

**Tip:** Naming the components in a particular fashion is crucial. Follow the golden rule while naming your components: for single terminal element, name of the element followed by bus it is connected to (For ex: Gen1 represents a generator connected to bus 1). For two terminal elements, element name followed by terminal numbers, separated by ‘\_’ (For ex: a line may be named as line\_1\_2 representing a line connected between bus 1 and 2.).The utility of naming your components in such a fashion will be seen soon.


**Note:** Transformer should be connected from **low voltage side to high voltage side**. The sending bus voltage must always correspond to **low voltage side**.

**Tip:** Transformer ratios can be entered as **V1/V2** instead of calculating decimal values.

The model must include a system base card. This is found in **OpenIPSL** → **Electrical** → **SystemBase**. Drag the component into the network workspace and name it as **SysData**. This component sets the frequency and base MVA for the system.

### Modeling generators

One way to model generators in the network is to drag and drop directly from OpenIPSL library into the network model workspace. This technique works best if there are no additional components to be added alongside the generator (such as AVR, PSS, GOV, etc.). If controllers are to be added, they will take up space on the network model workspace as well, crowding the model. Best way to overcome this, is to create a separate model file that can represent a generator and all its associated dynamic models, and then import this model file into the network model workspace.

1. Create a new model file and name it **Gen**. This new Gen.mo file will now on be referred to as generator model workspace.
2. Drag the model **Order4** generator model from OpenIPSL library into the workspace and give it a convenient name such as **Syn1**.
3. Connect  $vf0 \rightarrow vf$  and  $pm0 \rightarrow pm$ .
4. This model needs to interact with the outside world with an electrical pin. Drag the component **pwPin** from OpenIPSL → Interfaces → PwPin to the generator workspace and name it as **p**.
5. Connect **Syn1.p** to **p**.
6. This model can be given an icon as well. Go to the icon mode (next to Diagram Mode, shown in figure 3). Use the options  to create your desired icon for the generator.
7. Save the model. Your model may look like that in Figure 5. This model can now be seen next to the libraries in the Libraries browser tab as **Gen**.

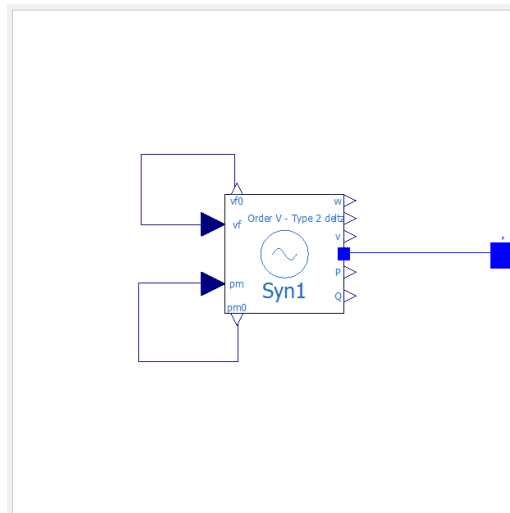


Figure 5: Gen Model

One of the key features in Modelica is model-reuse. Currently if you drag the **Gen** model into the network model workspace, you can connect the generator to the network, but you cannot set any parameters for the machine, by double clicking. To get access to all the machine parameters, we need to make changes to the **Gen** model.

1. Open the **Gen** model, and go to the equations mode. Copy the following code before the **equation** section.

```

extends OpenIPSL.Electrical.Essentials.pfComponent;
parameter Real D = 0 annotation (Dialog(group="Machine parameters"));
parameter Real M = 12.8 annotation (Dialog(group="Machine parameters"));
parameter Real Sn = 100 annotation (Dialog(group="Machine parameters"));
parameter Real T1d0 = 6 annotation (Dialog(group="Machine parameters"));
parameter Real T1q0 = 0.5350 annotation (Dialog(group="Machine parameters"));
parameter Real Vn = 18 annotation (Dialog(group="Machine parameters"));
parameter Real ra = 0 annotation (Dialog(group="Machine parameters"));
parameter Real xd = 0.8958 annotation (Dialog(group="Machine parameters"));
parameter Real xld = 0.1198 annotation (Dialog(group="Machine parameters"));
parameter Real xq = 0.8645 annotation (Dialog(group="Machine parameters"));
parameter Real x1q = 0.1969 annotation (Dialog(group="Machine parameters"));

```

2. Now in the diagram mode, open the machine model, and in the text fields for *Machine Parameters*, enter the parameter names. The machine model dialog should look like in Figure 6. Follow the same convention for *Power Flow Data* tab of the machine. Press OK and save the model.
3. Drag the **Gen** model from libraries pane to network model workspace. Name it as **Gen1**. When you now open the **Gen1** model from network model workspace, it will show you the parameters of the machine you can set. Your dialog should look like that in Figure 7.
4. Connect Gen1 to bus1. Similarly, create two more Gen model instances in the Network

model workspace and name them Gen2 and Gen3. Connect them to bus2 and bus3 respectively. The final network should now look like that in Figure 8.

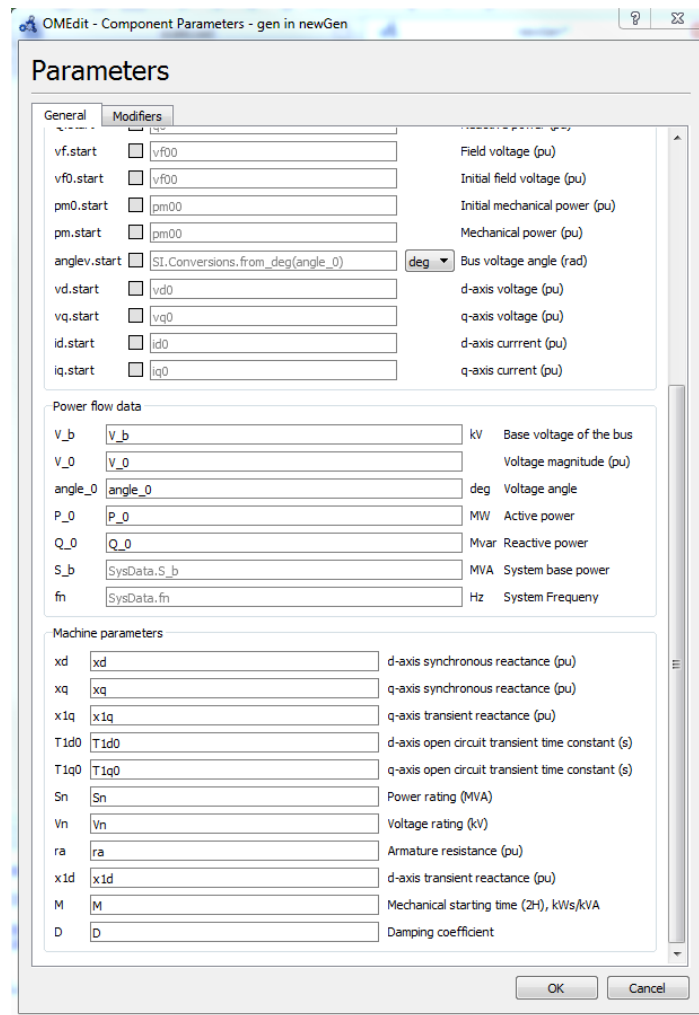


Figure 6: Generator model parameters from Gen model workspace.

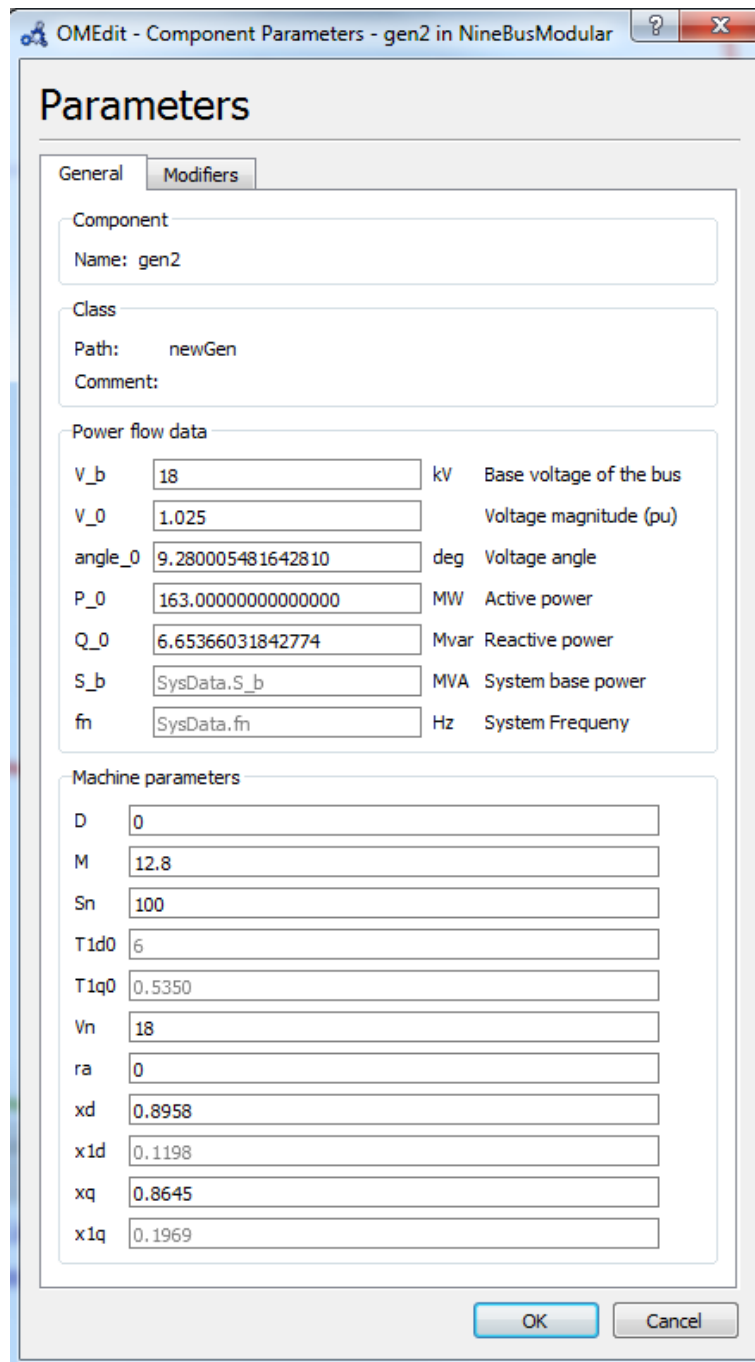


Figure 7: Generator model accessed from network model workspace



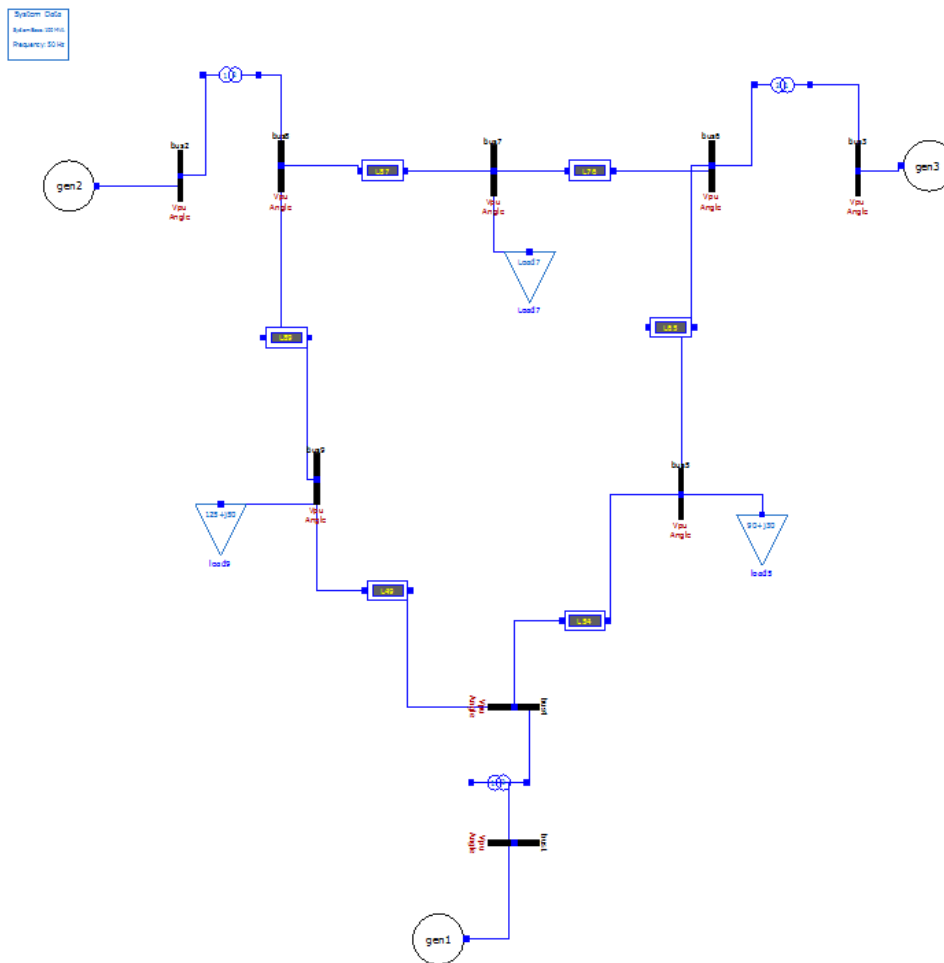


Figure 8: Nine Bus System

Once the model is created, the user needs to input parameter values for the components inside the model. The parameters for all the components are mentioned in tables 1-5. The filling in of data in the system requires two information. The component parameters are taken from the MATPOWER CASE FILE (.m files) (now called MCF). The power flow data is taken from MATPOWER LOAD FLOW RESULTS (now called MLFR) which is obtained by performing load flow to the MCF. In the tables below, data is already filled for each parameter taken from either MCF or MLFR.

Table 1. Transmission line data

Line Name	B (pu)	G (pu)	R (pu)	X (pu)
line_4_5	0.0790	0	0.0170	0.0920
line_5_6	0.1790	0	0.0390	0.1700
line_6_7	0.1045	0	0.0119	0.1008
line_7_8	0.0745	0	0.0085	0.0720
line_8_9	0.1530	0	0.0320	0.1610
Line_9_4	0.0880	0	0.0100	0.0850

**Table 2. Transformer data (on equipment MVA base)**

Name	LV Bus	HV Bus	V_b	V_n	Tap Ratio	R (pu)	X (pu)
XFR1	B1	B4	16.5	16.5	16.5/230.0	0	0.0576
XFR2	B2	B8	18	18	18/230.0	0	0.0625
XFR3	B3	B6	13.8	13.8	13.8/230	0	0.0586

**Table 3. Load data**


Name	Bus Connection	P_0	Q_0	S_n	V_b	V_0	angle_0
Load 5	Bus 5	90	30	100	230	1.012654324017776	-3.687396170157055
Load 7	Bus 7	100	35	100	230	1.015882583627499	0.727536076874302
Load 9	Bus 9	125	50	100	230	0.995630858048295	-3.988805272851462


**Table 4: Generator data**

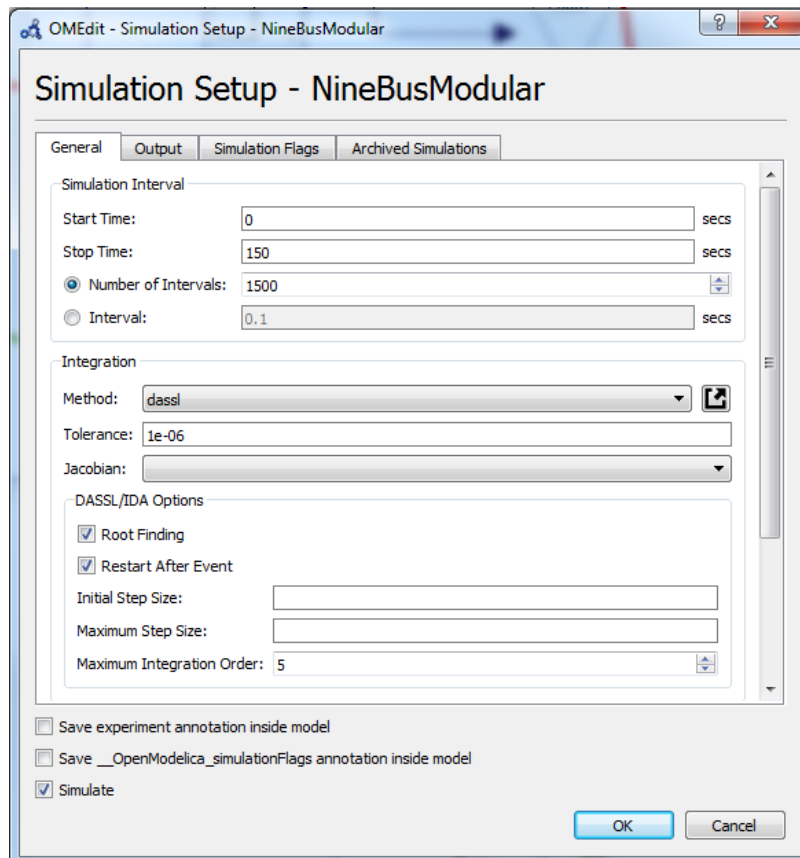
Parameters	Gen1	Gen2	Gen3
Connected to bus	Bus1	Bus2	Bus3
D	0	0	0
M (kWs/kVA)	47.28	12.8	6.02
P_0 (MW)	71.64102147448229	163.000000	85.0000000
Q_0 (MVar)	27.04592353349199	6.65366031842774	-10.85970907098886
Sn (MVA)	100	100	100
Td10 (s)	8.96	6	5.89
Tq10 (s)	0.310	0.5350	0.6
V_0 (pu)	1.04	1.025	1.025
V_b (kV)	16.5	18	13.8
Vn (kV)	16.5	18	13.8
angle_0 (deg)	0	9.280005481642810	4.664751333136778
ra (pu)	0	0	0
xd (pu)	0.1460	0.8958	1.3125
xd1 (pu)	0.0608	0.1198	0.1813
xq (pu)	0.0969	0.8645	1.2578
xq1 (pu)	0.0969	0.1969	0.25

**Note:** The generator buses i.e., Buses 1, 2 and 3 should have base voltage  $V_b$  as per the transformer ratings. These are 16.5, 18.0 and 13.8 kV respectively. The remaining buses are set to a base voltage of 230 kV.

After the system has been initialised, we check the system if there is any unconnected component.

This can be done by clicking on the *Check Model* button  in the tools bar or going to *Simulation* → *Check Model* option. This should give a dialog box with information about the number of variables and equations. If the number of equations and variables is the same, the model is correctly connected. If the number of variables and equations vary, there might be a connection missing. In such a case, recheck model connections.

After the simulation check, simulation can be run. Click on Simulation Setup  in the tools bar. Set the options as shown in *Figure 9* and press OK.



*Figure 9: Simulation Setup Dialog Box*

Once the simulation process is completed, OM switches to the results and plotting tab. Here, the desired variables can be plotted. The plotting window looks like that in *Figure 10*. The variables can be selected from the right pane and plotted.

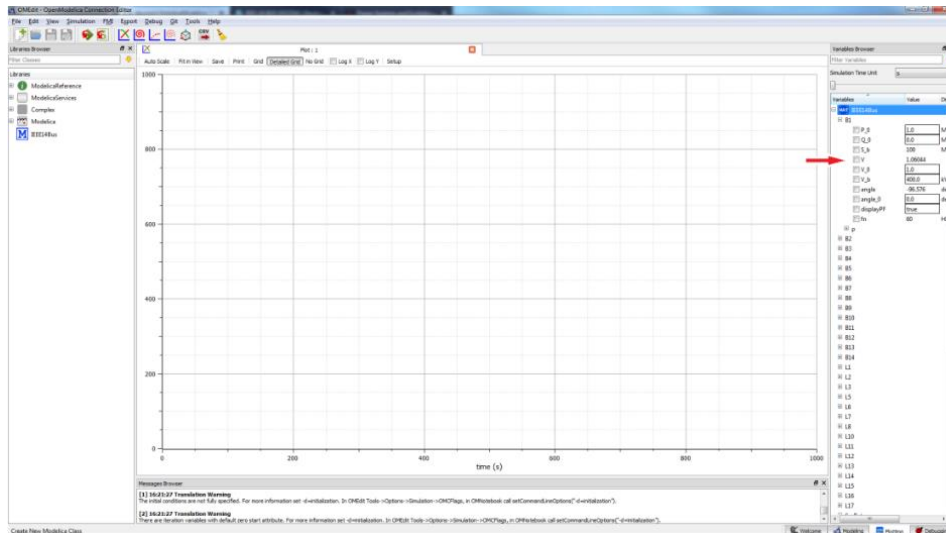


Figure 10: Plotting window

#### 4. WORKFLOW

Until now, a model of the IEEE 9 bus system was constructed and was simulated without any events in the network. The steps below will help you to introduce a load step event in the network and observe system response to such an event.

We will now introduce a load step event and observe what happens.

1. To simulate a load step, we use the *loadmod* from the *delmod* library, instead of *openispl* library. Replace the existing load 7 with the new load *scaleLoad* at bus 7 from the *delmod* library. This is done by selecting **Delmod -> Loads -> scaleLoad**.
2. The parameters and the initialization values for the new load should remain unchanged from the previous model.
3. Now in the graphical view, add a step block next to the load from **modelica -> blocks -> sources -> step**.
4. Connect it to the input of load 7 (new scaled load) and set the height as 0.3. Set offset as 1 and start time as 50. This is shown below in *Figure 11*.

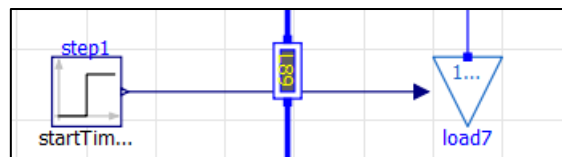



Figure 11: Scaled load

5. Run the simulation with the same conditions as in Figure 9.
  - The Bus voltage variables are (e.g.) *bus1.v* (as per the naming convention adopted by the user). The voltage is expressed in per unit (p.u) i.e., a ratio with respect to the base voltage which is 230kV. Hence, 1 p.u = 230kV, 0.95 p.u = 218.5kV and so on. It is fine to report results in p.u as long as base parameter is mentioned.
  - The generator frequencies are expressed through the rotor speed variable (*w*). **gen1 -> gen -> w (rotor speed)** i.e., *gen1.gen.w*. The actual names might differ so please expand all sub groups to check for the speed parameter.

- This is because rotor speed and frequency are directly related. Again, the rotor speed is also expressed in per unit (p.u). Consequently, 1 p.u can be approximated to be 50 Hz and so on.

**Task: You can save the results as a csv file by clicking on  button.**

- 1. Plot the voltages at bus 1, 2 and 3. Also plot the frequencies of the generators.**
- 2. Analyse the first few seconds of the simulation results from OpenModelica.**

## 5. REFERENCES

- [1] [https://hvdc.ca/uploads/knowledge\\_base/ieee\\_9\\_bus\\_technical\\_note.pdf?t=1460659065](https://hvdc.ca/uploads/knowledge_base/ieee_9_bus_technical_note.pdf?t=1460659065)
- [2] <https://github.com/OpenIPSL/OpenIPSL>
- [3] PSAT Manual. Available at <http://www.eecs.wsu.edu/~ee521/Material/20120927/psat-20080214.pdf>