CIGRE IEEE DLL

CIGRE_IEEE_DLL1

Anton Stepanov, 5/5/2023 12:30:00 PM

1 Description

This device is used to interface EMTP with DLLs created using the IEEE/Cigre DLL Modeling Standard from the Joint IEEE TASS Task Force and CIGRE B4.82 Working Group “Use of real-code in emt models for power system analysis”.
Once a DLL adhering to this standard is created, this device is used to create all the necessary connections (pins) and provide access to the modifiable parameters of the DLL. The interfacing pins (ports) of this device in EMTPWorks are created automatically based on the information contained in the DLL.
This version of the device participates in control system solutions without any time-step delay.
This device has user and manufacturer functionalities: manufacturer license is required to load DLLs into this device, whereas users can change DLL parameters but cannot load new DLLs.
The DLL computations are called based on the time-step value stored in the DLL.

2 Workflow

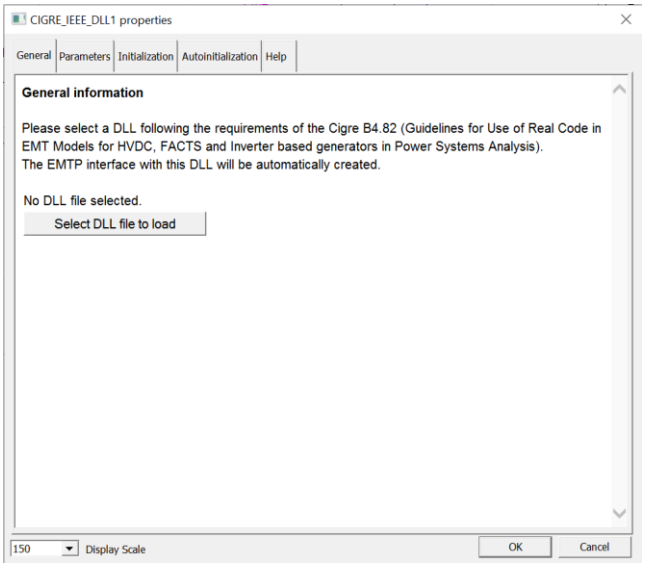
A typical workflow of using the device is as follows:

1. A manufacturer generates a DLL that adheres to the IEEE/CIGRE real code interface standard.
2. Then “loads” the generated DLL into this device. At this point the device is ready and can be distributed to users along with the DLL.
3. Users can tune the device using GUI:
    a. Change DLL parameters
    b. Activate interpolation
    c. Activate model initialization and provide initial output values
    d. Use autoinitialization feature
4. Launch simulation.

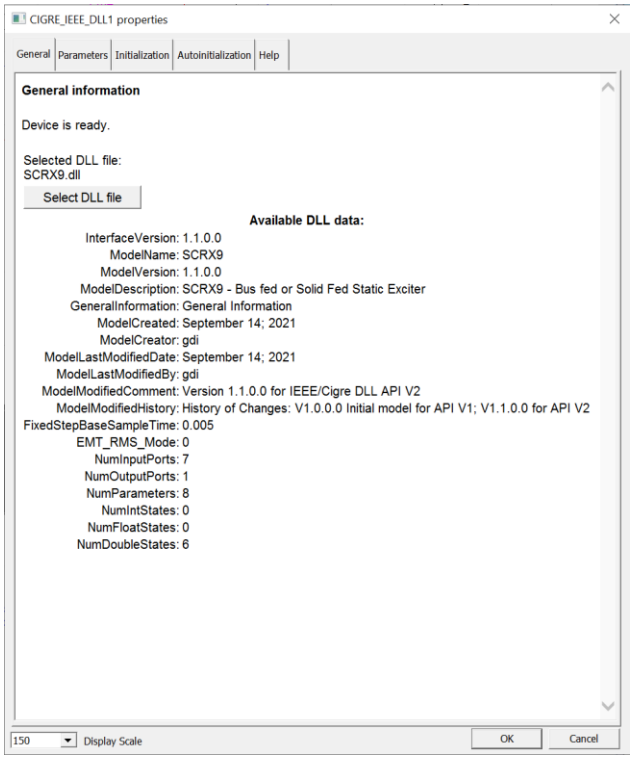
3 Tabs

3.1 General tab

This tab allows to select the DLL to load into the device. To load a DLL, press the “Select DLL file to load” button, **Figure 1**.a. A standard file selection dialog window will open. Navigate to the appropriate DLL file and select it, the device will load it.
Once the DLL has been loaded, this tab will show that this device is ready, i.e. that a DLL has been loaded. It will also show the path to the DLL file and list static information stored in the DLL, such as the model name, version, number of inputs, etc., **Figure 1**.b. The value of FixedStepBaseSampleTime is of particular importance since it will be used as the interval at which the DLL will be called during the simulation. It is advisable to have EMTP simulation time-step lower than FixedStepBaseSampleTime.
It is also possible to provide a new path to the DLL file if its location relative to the EMTP design file has changed after loading. For this, press the “Select DLL file” button and navigate to the new location of the DLL file. It is the user’s responsibility to make sure that the newly selected file is the same as the one used for loading because no verification is made by the device.



a) Before loading a DLL



b) After loading a DLL

Figure 1 General Tab appearance

3.2 Parameters tab

3.2.1 DLL parameters table

Once the DLL is loaded, a table containing all parameters of the DLL is automatically created in the Parameters tab, **Figure 2**. The table lists the names of DLL parameters, their values (default values are taken when a new DLL is loaded into the device), and their units. A tooltip on the name shows the maximum, minimum, and default values of each parameter as well as its description and data type.
The user can enter new values of the parameters within the given limits. If a value entered exceeds the limits of a given parameter, it will be automatically overwritten by the default parameter value.

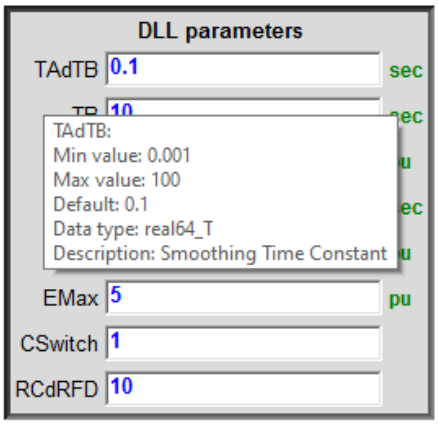


Figure 2 Example of DLL parameters table

3.2.2 Use linear interpolation option

Input signals can be interpolated to match exactly the instants when the model computation should be performed, i.e. the multiples of FixedStepBaseTime. If selected, linear interpolation will be applied to all input signals, which may include logical signals, so select this option with care.

3.2.3 Allow multiple calls to DLL when model sampling time is smaller than EMTP time-step option

Even though it is advisable to have EMTP simulation time-step lower than FixedStepBaseTime, it is not mandatory. If EMTP simulation time-step is larger than FixedStepBaseTime, this option allows to call the DLL several times at each time-point to make sure that the DLL is always called on time.

3.2.4 Copy DLL for this instance option

Some DLLs may use static variables so inherently have difficulties with managing multiple instances of the model. This flag allows to create a unique copy of the DLL for this particular instance so that the DLL would only deal with a single instance of the model. The DLL copy will be deleted when simulation ends.

3.3 Initialization tab

3.3.1 Use model initialization function option

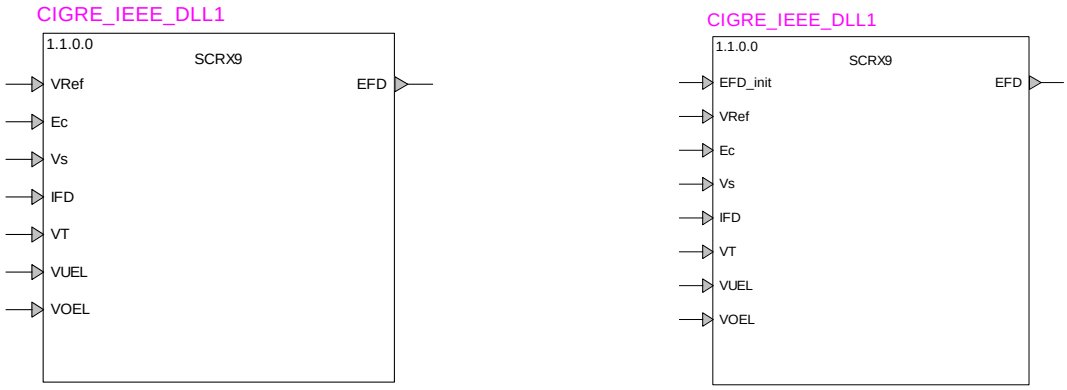
DLLs may provide a function for model initialization at simulation startup. Select this option if such a function should be used.

3.3.2 Initialization release time

This field allows to set the time until which EMTP will call model initialization function provided in the DLL. Initialization release time will only be considered if “Use model initialization function” is activated.

3.3.3 Provide initial output values through the pins option

This device allows two ways for providing initial values of output signals (this is particularly useful for model initialization): by using a table in this device graphical user interface or through the pins. If this checkbox is selected, additional input pins are created automatically, one for each output signal, as shown in **Figure 3**. Otherwise, if the user unchecks the “Provide initial output values through the pins” checkbox, a table containing all output signals is created, **Figure 4**. The table lists the names of all output signals and their initial values. Initial values are set to zero by default.



a) Initial values are provided in the mask **b) Initial values are provided through the pins**

Figure 3 Device symbol depending on how initial output values are provided

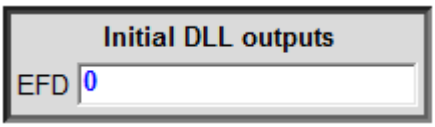


Figure 4 Example of initial DLL outputs table

3.4 Autoinitialization tab

This tab allows to activate the autoinitialization mode, which allows to flat-start the model if model initialization function is not available or is unable to initialize the model perfectly. In this mode, a text file stores all input data for the DLL from time zero until a given time-point when the system converges to a steady state during the first simulation run. For consequent runs, these data points can be supplied to the DLL at the first time-point in a loop to imitate simulation inputs so that at the second time-point the DLL would already be at steady state.

3.4.1 Autoinitialization file selection button

Allows to select the autoinitialization file. Autoinitialization file stores all input data for the DLL. When using autoinitialization data file, make sure that it contains data for all input pins. It is also advisable that autoinitialization data file contain at least one fundamental frequency period of steady-state operation.
It is the user's responsibility to make sure that only one DLL is saving data into the selected data file, since no additional checks are performed by the device.

3.4.2 Parameters table

By default, the autoinitialization option is inactive. The current state is given in the autoinitialization options table, **Figure 5**, and can be one of the three:

1. Do not use autoinitialization
2. Replay data from a text file (text file with autoinitialization data must have already been generated)
3. Record data into a text file

The table also allows to set the time until which data is stored in or retrieved from the autoinitialization file.

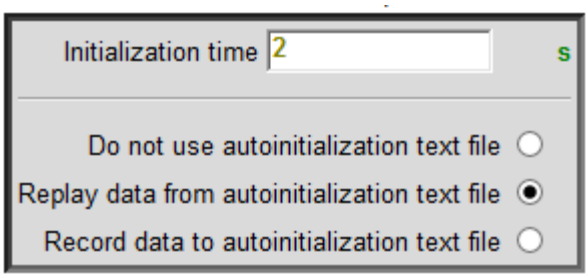


Figure 5 Autoinitialization parameters table

3.4.3 Adjust phase for autoinitialization option

If some of the inputs to this DLL are sinusoidal signals such as AC voltages, their phase may need to be adjusted during autoinitialization if data recoding was performed using a different initial phase. This is the case, for example, when a different transformer connection was used during recording.
If selected, two additional inputs to the DLL appear at the bottom of the input pins list: frequency and phase, **Figure 6**. It should be noted that recording and replaying of the autoinitialization file must be done using the same phase adjustment setting: either active in both cases or inactive in both cases.

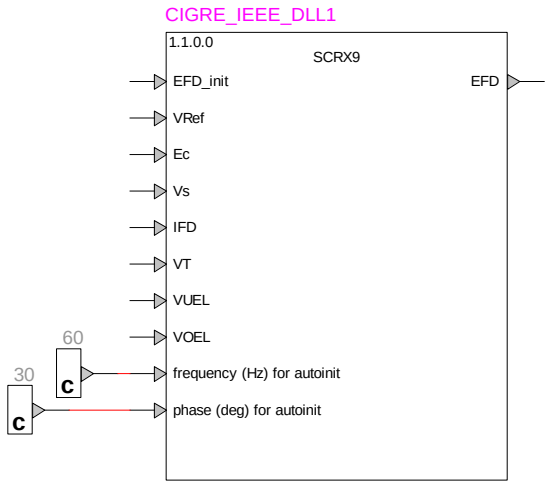


Figure 6 Device symbol if phase adjustment is activated during autoinitialization

3.4.4 Accumulating inputs table

Some inputs to the DLL may be accumulating (increasing over time), such as the simulation time or the count of certain events. If such inputs exist, they can be adjusted when autoinitialization is used: inputs selected in this table will have the last used value from the autoinitialization file added to the EMTP input value.
This table allows to indicate which of the inputs are accumulating, **Figure 7**. By default, no input is accumulating.

Figure 7 Accumulating inputs table

4 Time-domain model

This device serves as an interface to a DLL. It calls the DLL on the multiple of its sampling time. Although iterations may be required by the control system at a given time-point, the DLL is called only once, at the first iteration.