Global Variables

Jean Mahseredjian, 2016-08-30 10:06:00

1 Introduction

The definition of global variables is useful for applying simulation parameters or user-defined variables in various scripts. The main usage is for masked devices where the programming of the mask is dependent on some simulation parameters or it is needed to define global data from a single location. It is also possible to specify named values that can be used in devices at the top circuit level or

The definition of Global Variables is for advanced usage of the software. It offers many sophisticated options and improper usage can create complex problems that will not be fixed or addressed by the software support team.

2 Usage

The “Design>Utilities>Define Global Variables” command opens the panel shown in Figure 1. A text area is used to enter various global data definition commands. The global data object is named:
oGlobalData
This object fields allow transmitting data to devices with mask scripts.
The checkbox “Apply changes now” forces immediate propagation and recalculation of data in all masked devices using global data and requesting updates.
If “Apply changes now” is turned off then the changes will not be propagated until the user starts the simulation.

The checkbox “Echo on updated device names” is turned on to echo the name of each updated device in the Console window.

In the example of Figure 3 the subcircuit Fault has a mask in which two global variables are used to define switch closing and opening times. The global variables are initially defined in the panel of Figure 2. If the user reopens the panel of Figure 3 and redefines the global variables then the mask of Figure 3 is automatically updated. It is noticed that the mask is set to use global data (“Use Global Data”) and to receive updates when changes in global data occur.
The contents of the text area in Figure 2 are:

```
oGlobalData.ScloseTime=1e-03;
oGlobalData.SopenTime=0.05;
```

To show tclose and topen on the design screen near the device Fault, the user must make visible the ModelData attribute.

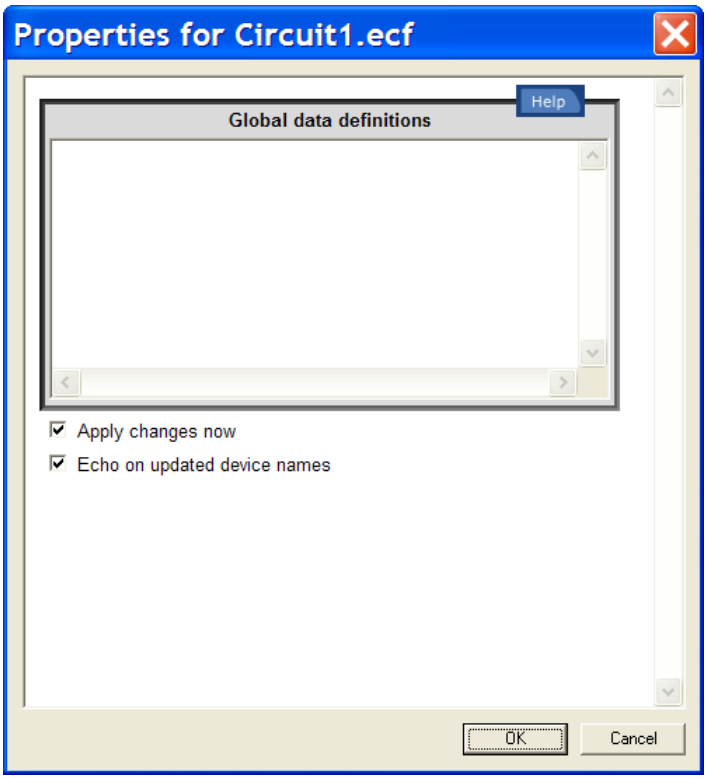

Figure 1 Default Global data definition panel

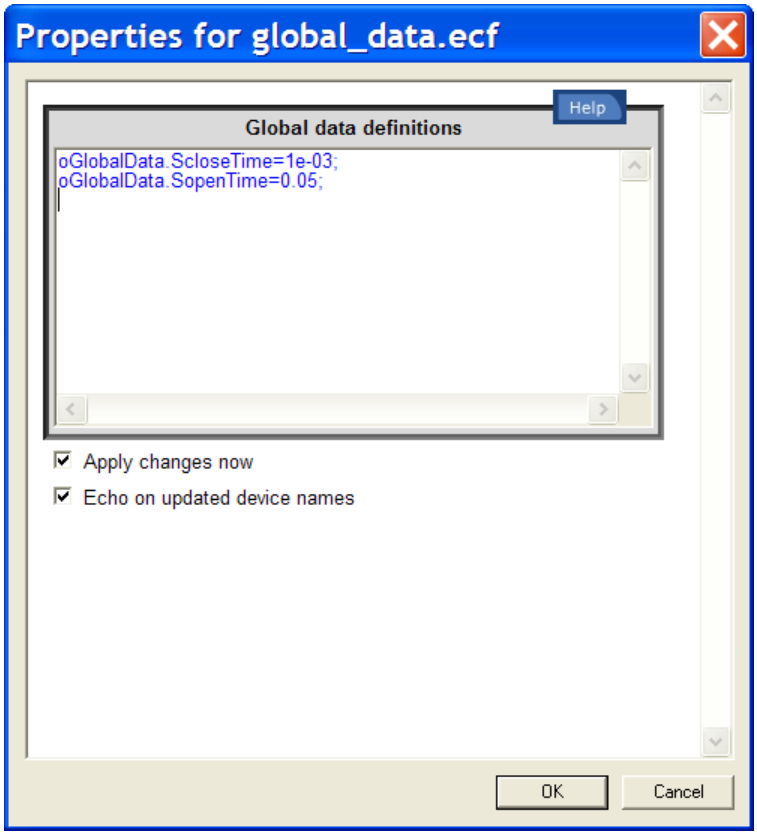

Figure 2 Example of global data definition

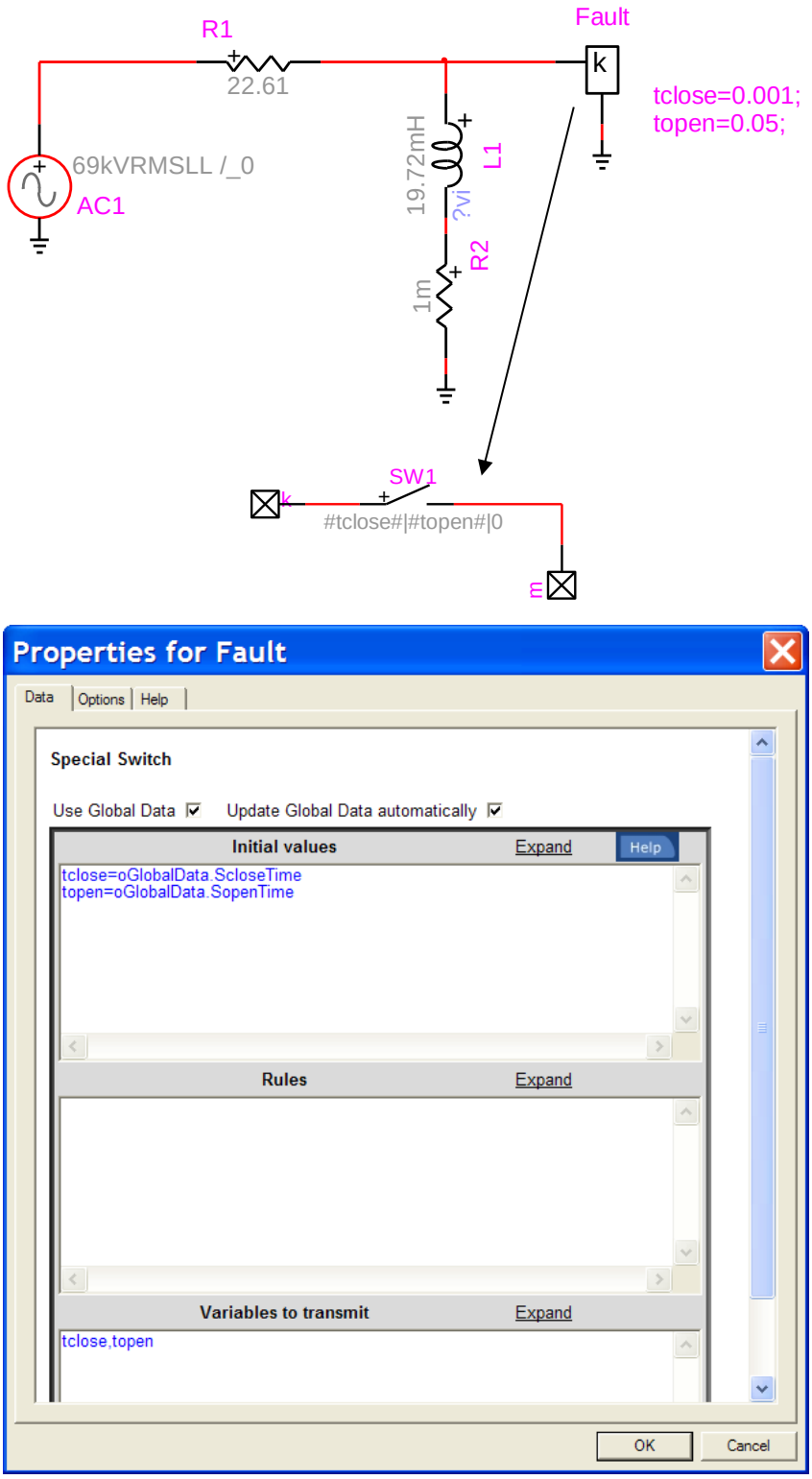

Figure 3 Global data usage example
(see Documentation\Help Files\masking\global_data.ecf)

3 Rules

When the user clicks OK on the global data definition panel of Figure 1, the global data object is redefined and transmitted to all devices requesting and using global data.
The processing of the text area in Figure 1 is based on the JavaScript “eval” function. It means that any function calls, scripts and global data definitions can be entered and evaluated. It is needed to clearly understand how objects in JavaScript are maintained before applying sophisticated programming. If a function call is performed using parseScriptFile then the function file will be located according to standard search rules. The user may add specific locations using the addScriptSearchPath command directly in the text area of Global data definitions. In this example the location “c:\d” is added to the search path:

```
addScriptSearchPath('C:/d')
```

and means that any script file placed into the above folder can be located, parsed and used by a single parseScriptFile command on that script file name. This command can be run several times without increasing the stack of search paths. The command getScriptSearchPaths() can be used to obtain a list of search paths for the current design. It is noticed that this is also a convenient way for initializing the script search paths of a design since the Global data definitions are triggered automatically when the design is opened. Once the search path is established, the scripts can be called from any script function in the design.

If a global data field is defined it remains in the memory of the design. If the definition of a global data field is removed from the Global data definitions it will continue to persist until the design is closed and reopened again. It is possible to use reinitilization procedures through scripting (see section below).

1. The oGlobalData object is unique for each design.
2. The oGlobalData object can be used in various scripts as a standard JavaScript object. It means that in addition to defining variables, it can be used in computations.
3. In addition to data it is allowed to use object methods.
4. In addition to the oGlobalData fields provided by the user in the Global data definitions panel, EMTPWorks maintains a set of predefined variables that can be used in various masks.
5. The devices that are automatically triggered after a change in any global data definition are those carrying the attribute GlobalData equal to 1. Any device with this attribute equal to 1 automatically receives a change in its Status attribute when changes in global data occur. If the device maintains a valid and non-empty Status.Script attribute, then the script named in Status.Script is automatically triggered to perform the required updating actions.
6. Currently the devices with the mask script_black_box.dwj are set to receive and automatically (option) update data using a predefined Status.Script script.
7. When the design is closed, the oGlobalData object is nullified.
8. When the design is reopened again, the oGlobalData object is reinitialized using predefined variables and variables defined in the Global data definitions panel.

4 Reinitializing Global data definitions

As explained above, if a field of oGlobalData is cancelled in Global data definitions, it is not actually eliminated from memory. If, for example, the definition of SopenTime is commented out:

```
oGlobalData.ScloseTime=1e-03;
//oGlobalData.SopenTime=0.05;
```

the field SopenTime still remains in memory until the design is closed and reopened again. This may cause problems since then, after reopening, the variable SopenTime will become undefined and cause an error message. To avoid such problems it is needed to reset definitions explicitly. If, for some reason SopenTime is not needed anymore, then the correct code is:

```
oGlobalData.ScloseTime=1e-03;
delete oGlobalData.SopenTime
```

The delete command will make SopenTime undefined immediately and the user can replace its usage or adjust the design accordingly.

If the user needs to maintain a large collection of data fields or methods defined in Global data definitions, then a better approach is to create a separate user child object by typing the following lines :

```
oGlobalData.user = new UserGlob()
oGlobalData.user.ScloseTime=1e-03;
oGlobalData.user.SopenTime=0.051;

function UserGlob(){
  this.ScloseTime=0; //define at least one field
}
```

This means that now the “Initial values” section of Figure 3 must become:

```
tclose=oGlobalData.user.ScloseTime
topen=oGlobalData.user.SopenTime
```

Having created a single user-defined child object, the child object is now continuously reset and redefined due to the new object definition command. Thus any changes in user object field names are immediately applied in the design.

5 Predefined global variables

The following is a list of predefined variables that can be accessed by the user in masked devices for performing various computation tasks. It is noticed that if the user is familiar with JavaScript and JavaScript extensions in EMTPWorks, then the user can write codes to retrieve an manipulate any other variables available in a design. The following predefined list is provided for convenience. It can be increased by the user.

oGlobalData.Deltat	available from “EMTP>Simulation Options”, integration time-step in seconds
oGlobalData.t_max	available from “EMTP>Simulation Options”, simulation time in seconds
oGlobalData.steadystate	available from “EMTP>Simulation Options”, equal to 1 when the steady-state solution is selected
oGlobalData.timedomain	available from “EMTP>Simulation Options”, equal to 1 when time-domain solution is selected
oGlobalData.LoadFlow	available from “EMTP>Simulation Options”, equal to 1 when the Load-Flow solution is selected
oGlobalData.fscan	available from “EMTP>Simulation Options”, equal to 1 when the Frequency Scan solution is selected
oGlobalData.DefaultFrequency	available from “EMTP>Simulation Options”, Default Power Frequency selection
oGlobalData.confirm_device_updates	Option “Echo on updated device names”. Becomes true when the checkbox is checked.
oGlobalData.Cancel_is_halt	This option allows to use return instead of halt (stop) when starting a scripting sequence which opens a series of devices one-by-one. The default is false, which means that hitting the Cancel button on a device data web does not stop the sequence.
oGlobalData.Cancel_pushed	Sends Cancel button state: true when pushed Cancel for last device data web, false otherwise.
oGlobalData.DataTranmissionOn	is set to true when the device updating process (with global data) is started and becomes false when the device updating process is stopped.
oGlobalData.NV	Object for holding named values.

An example of usage of DefaultFrequency is shown in Figure 4 for the “P 1-phase” periodic meter device which is using the frequency for computing the period of the signal. When the line with the “oGlobalData.DefaultFrequency” is uncommented and with “Use Global Data” and “Update Global Data automatically” are turned on, then this device will always compute using the Default Power Frequency selection in “EMTP>Simulation Options”.

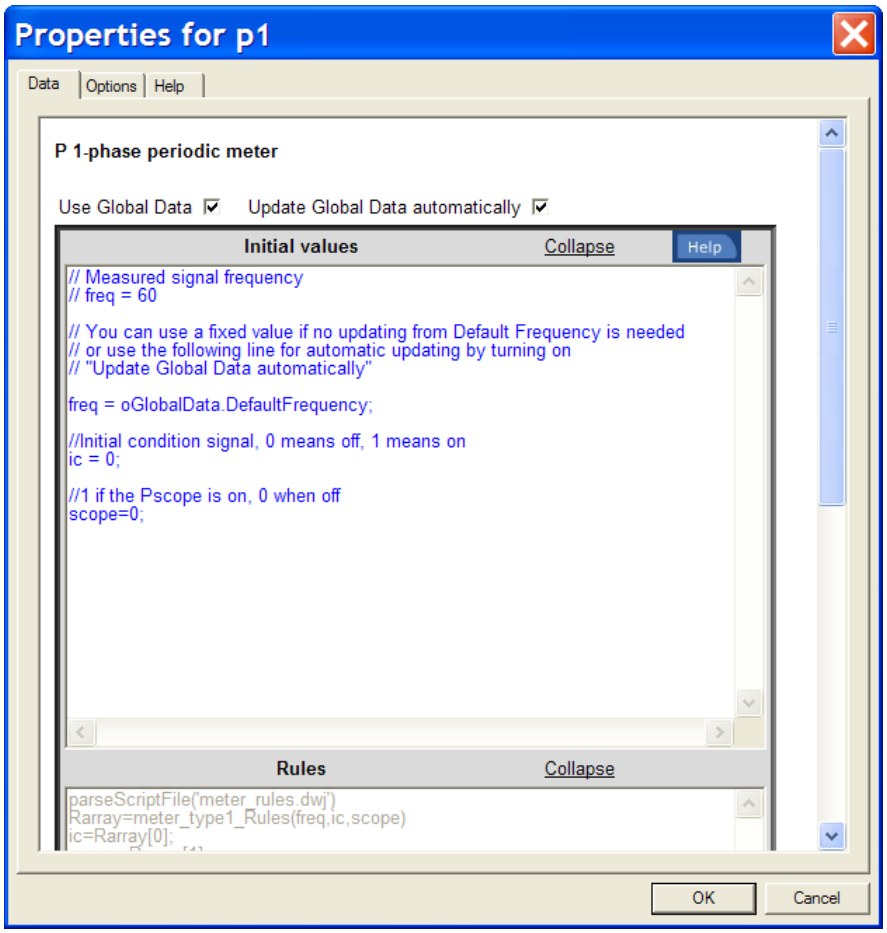

Figure 4 Usage of DefaultFrequency field for automatic setting of meters

6 Using named values at the top circuit level

In Figure 3 the variables tclose and topen are named values. In this example, the named values are determined using the subcircuit mask, but in EMTP it is also possible to define named values for the top level circuit. In the example of Figure 5, the devices L4 and C3 are appearing at the top level (not in a subcircuit) circuit and using the named values L4 and C3.

It is possible to define the variables L4 and C3 using the global data object. In this case, it is achieved by entering the JavaScript lines:

```
oGlobalData.NV.C3=1e-06;
oGlobalData.NV.L4=   19.72e-03;
```

The child object NV signifies “Named Value” and can hold an arbitrary number of user defined fields. The object NV is reinitialized each time the user pushes the OK button.

It is implicit that named values defined using the global data object can be also transmitted into subcircuits using the procedures described above. By default, they are only visible at the top level circuit.

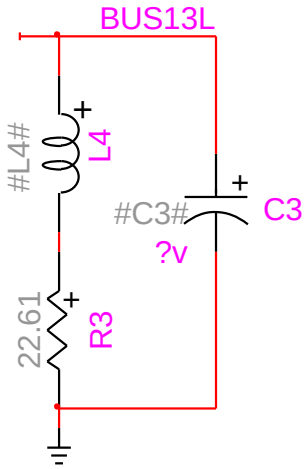

Figure 5 Names value usage at the top level circuit

7 Procedures

The script file used for defining global data object is define_global_data_object.dwj. This script file contains the initialization procedure for oGlobalData: oGlob_initialize.
The Global definition methods (“Design>Define Global Variables”) are programmed in global_data_options_m.dwj.
The device updating procedures are programmed in update_variables_in_black_boxes.dwj. Devices are updated automatically when their attribute field GlobalData is set to 1.

The following list presents EMTPWorks functions that can trigger global data maintenance procedures:

1. Design opening: When a previously saved design is opened, it initializes the global data object using oGlob_initialize and runs user Global data definitions.
2. Setting Simulation Options and saving (click OK):
    a. If the global data object does not exist it is created.
    b. The global data fields related to simulation options are populated.
    c. The Global data definitions are re-evaluated since they may use data from Simulation Options.
    d. All devices are updated to reflect the latest changes in global data. This is a call to Device updating procedures.
3. Define Global Variables:
    a. If the global data object does not exist it is created. This action automatically updates the global data fields related to simulation options.
    b. The Global data definitions are evaluated.
    c. All devices are updated to reflect the latest changes in global data. The updating is started only when “Apply changes now” is selected. This is a call to Device updating procedures.
4. Device updating procedures:
    a. If the global data object does not exist it is created. This action automatically updates the global data fields related to simulation options.
    b. All devices are updated to reflect the latest changes in global data.
5. Start EMTP (all actions related to the generation of the final Netlist): Calls Device updating procedures.
6. Design closing: The global object oGlobalData is nullified.