Genesys 201508 Release Notes

Genesys 2015.08 Release Notes

Release Highlights: Spectrasys Frequency Dependency Support

Spectrasys now supports frequency dependent behavioral models. It enables vendor spreadsheet frequency, temperature, and bias dependent nonlinear parameters such as 1dB compression, IP3 and IP2 to be conveniently used in the RF system simulation without the need to write custom equations. Spectrasys will automatically interpolate the multi-dimensional data defined by a muli-tab spreadsheet or System MDIF file for the component such as a power amplifier, for its correct characteristics at the frequency, temperature and bias used in simulation.

There are two modes of operation: 1) Normal or manual data entry or 2) Dataset or file based. When the simulator runs, spectrums are obviously created at various frequencies. With this feature, the characteristics of these spectrum will be a function of the frequency dependent parameters.

See the Frequency Dependency section for additional information.

Normal Mode

In this mode users can conveniently enter frequencies and parameter values at those frequencies.This entry method also supports mixed parameter fixed and vector values. For example, if there is no frequency dependent data for a particular parameter the simulator will use the fixed value for all frequencies.

See the Supported Models for details on which models are supported for this mode.

Dataset / File Based Mode

When this mode is selected RF amplifier model data will automatically be extracted from a spreadsheet file (File > Import > Sys-Parameter Files > .csv, Excel or File > Import > ADS Files > System MDIF File) that will automatically be imported onto the workspace tree. Multiple independent variables like temperature and power supply voltage can be supported in this mode when using an Excel file with data configured on multiple tabs.

See the Supported Models for details on which models are supported for this mode.

New Frequency Dependent RF Examples

- ADS MDIF Import.wsg
- Basic Mixer Frequency Response.wsg
- Freq Dependent Stats Basics.wsg
- Intermod Distortion vs Freq.wsg
- Interpolated Data.wsg
- L Band Transmitter.wsg
- Multidimensional Data 3D Graphs.wsg
- Multisource PortZ vs Freq.wsg

Release Highlights: MiniCircuits Nonlinear Behavioral Library

A frequency dependent library has been created in conjunction with MiniCircuits. Their library consists of both system level behavioral nonlinear frequency dependent models and passive linear S parameter models. Currently, there are 105 models.

Release Highlights: Analog Devices Nonlinear Behavioral Library

A frequency dependent library has been created in conjunction with Analog Devices, Inc. Their library consist of system level behavioral nonlinear frequency dependent amplifier models. Currently, there are 88 models in the library.

Release Highlights: AM / PM Support for Nonlinear Models

AM to PM can be specified on the amplifier models, nonlinear block model, and the mixer models. There are 3 modes of operation: 1) Off, 2) Point Derivative, 3) Data Points. For the Point Derivative implementation, the user specifies the power level of a an AM / PM derivative and the value of the derivative. For the Data Points implementation a list of phase values are specified along with the power levels of those points. For more information, see Spectrasys AM to PM.

Release Highlights: Import Keysight Sys-Parameter Files

Frequency dependent behavioral model data may be stored and imported via a Keysight Sys-Parameter files can be imported into a dataset. The Sys-Parameters format is very flexible and can be implemented in a spreadsheet, csv, or MDIF files. Multiple sheets are used to represent multidimensional data.

For more information, see Importing Keysight Sys-Parameter Files.

Release Highlights: Import ADS Files

ADS files such as GMDIF, MDIF, S2D, and P2D file import is supported. There are some limitations as Genesys does not have the ADS models that use these files directly. GMDIF and MDIF import support has been optimized around importing behavioral data for simulation so there are some aspects of these files that may not be imported fully. Since Genesys does not have a Data Access Component (DAC), you can import the data into a dataset. Moreover, a new **getinterpdata** function can be used to extract the data of interest that can be used elsewhere in the product.

For more information, see ADS File Import.

Release Highlights: getinterpdata Function

Since Genesys does not have a Data Access Component (DAC) we import the data into a dataset and a new **getinterpdata** function can be used to extract the data of interest that can be used elsewhere in the product. This is powerful multidimensional interpolator that can extract data from a multidimensional dataset.

For more information on getinterpdata, please visit getinterpdata (Engineering Language) or getinterpdata (MATLAB Script).

Release Highlights: Model Ambient Temperature Support

An ambient temperature parameter **Ta** has been added to all models. When this value is empty the analysis temperature is used during simulation otherwise it will use the value specified. When an analysis is ran it will set the global temperature variable called TEMPERATURE. This value is then used by models with empty values. For linear devices the ambient temperature is used to calculate thermal noise. For all non linear devices the ambient temperature is used to calculated all non linear temperature dependent parameters including thermal noise.

Release Highlights: 3D Plots

The following illustrates a few sample MATLAB scripts for creating 3D graphs:

```
[xx,yy] = meshgrid(-1:.1:1,-1:.1:1); % a built-in Matlab Script expression

zz = xx.^2+yy.^2;

graph1 = graph_figure ('First Cartesian graph with wireframe'); % create a graphing object

graph1. add_mesh_data (xx,yy,zz,'salad_bowl_mesh'); % a 3D mesh plot with mesh grid is displayed

graph1. set_title ('Image of a salad bowl mesh'); % Title of the graph
```

graph1. ***set_label_x*** ('X Range'); % label for X-axis

graph1. ***set_label_y*** ('Y Range'); % label for Y-axis

graph1. ***set_label_z*** ('Z Range'); % label for Z-axis

graph2 = ***graph_figure*** ('Cartesian graph without wireframe');

graph2. ***add_surface_data*** (xx,yy,zz,'salad_bowl_surface'); % a 3D surface plot is displayed; explicitly name the trace with the 4th (an optional) argument

graph2. ***set_title*** ('Image of a salad bowl surface'); % Title of the graph can be set after the image is created

```
[theta,phi] = meshgrid(linspace(-pi,pi,88),linspace(-pi/2,pi/2,69));
rho = 0.1 + sin(4*theta).*sin(2*phi);
```

graph3 = ***graph_figure*** ('First spherical graph');

graph3. ***add_spherical_data*** (theta,phi,rho,'sphr_16_lobe'); % a 3D spherical plot is displayed.

graph3. ***set_title*** ('8+8 lobes'); % Title of the graph

[x4,y4,z4]=sph2cart(theta,phi,rho); % a built-in Matlab Script expression

graph4 = ***graph_figure*** ('Spherical graph transformed to Cartesian graph');

graph4. ***add_mesh_data*** (x4,y4,z4); % a 3D mesh plot.

[a5,e5,r5]=cart2sph(x4,y4,z4); % a built-in Matlab Script expression

graph5 = ***graph_figure*** ('Spherical graph back again');

graph5. ***add_spherical_data*** (a5,e5,r5,'eight_lobe'); % a 3D spherical plot.

graph5. ***set_title*** ('8-lobe spherical'); % Title of the graph

graph6 = ***graph_figure*** ('Two images in one');

graph6. ***add_mesh_data*** (x4,y4,z4, 'tr_6a'); % image #1

graph6. ***add_mesh_data*** (xx,yy,zz, 'tr_6b'); % image #2

graph6. ***set_title*** ('Two 3D Shapes'); % Title of the graph

See 3D Graphing for additional information.

Release Highlights: Avago X-Parameter Models (since 2014.03)

Avago components have been provided with required X-Parameter data files in the form of Avago X parameter library. Using the provided X-Parameter files, you can perform non-linear simulations in Genesys. To learn more about using these parameters, watch the X Parameter Library video available in your Genesys installation. These components are available in the Avago XP Parts library. They are extracted with 1-Tone X-Parameter files. Note that you must download the X-parameter files and install them at the standard location. These files are provided as self-extracting zip files. Refer to Using X-Parameter Libraries for detailed instructions on how to download and install the X parameter library.

Release Highlights: MATLAB Script (2014.03 & 2015.08)

MATLAB Script replaces the Math Language expression engine that was syntax compatible with MATLAB. MATLAB Script provides the scripting engine which is the same as MATLAB, and a limited subset of MATLAB functions to be used for equations and parameter expressions. You do not need to buy any license to use MATLAB Script engine and its functions.

Starting in Genesys 2015.08 release, MATLAB Script allows you to use your MATLAB. The supported MATLAB version is 2014a and later. You can use your MATLAB installation (MATLAB 2014a or later) from a workspace equation, a design equation, or the command prompt by switching to “Use MATLAB” in the right click menu. The corresponding MATLAB licenses are required when you use your MATLAB installation. Once you start to use your MATLAB from within Genesys, the corresponding MATLAB licenses will be held for the remainder of the Genesys session, and released on exit.

Starting in Genesys 2015.08 release, the equation debugger supports MATLAB Script equations.

MATLAB Script allows you to use external .m functions from a specified directory. The default MATLAB function directory is $HOME\Genesys\MATLAB. You can change the directory by using Tools > Options... > Directories > MATLAB Functions. In MATLAB Script mode, you can use external .m functions from the specified directory inside Genesys. However, .p functions, MATLAB classdef files and MATLAB script files are not supported in MATLAB Script mode. When switching to use MATLAB, you can use external functions and classdef files (in .m or .p) from the specified directory (Tools > Options... > Directories > MATLAB Functions) and from your MATLAB search path inside Genesys. The following tables show the supported context:

.m File Content	Use MATLAB Script	Use MATLAB 2014a and later
function	supported	supported
classdef	not supported	supported
script	not supported	not supported

.p File Content	Use MATLAB Script	Use MATLAB 2014a and later
function	not supported	supported
classdef	not supported	supported
script	not supported	not supported

File Location	Use MATLAB Script	Use MATLAB 2014a and later
Tools > Options... > Directories > MATLAB Functions	supported	supported
MATLAB search path	not supported	supported

In Genesys, workspace variables, design variables, and dataset variables support the " double precision floating point, 32 bit integer, unitcode strings, boolean, and complex numbers in double precision floating point " datatypes for serialization and deserialization. In MATLAB Script, you can use functions such as double, single, uint8, uint16, uint32, uint64, int8, int16, int32, int64, and logical to cast a variable to a specific datatype and precision. Within a MATLAB Script equation evaluation, the datatype is preserved. However, when a MATLAB Script variable is transferred to Genesys for storage, the variable is converted to one of the supported datatypes that can hold the value without losing the data. Once the variable is converted, it remains in the converted datatype when passing into MATLAB Script again.

In general, a *value* class can be transferred from a retail MATLAB equation (or command prompt) to a Genesys workspace and can be used later in another retail MATLAB equation (or command prompt). An example of the value class is ‘gf’ (Galois field array). On the other hand, a *handle* class has a reference to the actual object, and may not expect to work when transferring across the boundary between retail MATLAB and Genesys. The general guideline is to use handle classes just within a retail MATLAB equation, that is, create the handle class variable in the equation, delete, and clear the handle class variable before the end of the equation. For example:

You can use the 'isvalid' function of the handle class to test whether a variable is a valid handle. For MATLAB 2014b and later, you can use the 'ishghandle' function to test whether a variable is a valid Handle Graphics object.

CAUTION

In general, the performance of the equation system is comparable to the previous releases. You can even observe performance improvement in certain cases, for example, using heavy "for" loops or using computation intensive functions like "conv". However, due to architectural difference in MATLAB Script, you may observe performance degradation in the following conditions:

- The user interface may appear unresponsive when Genesys is running a long and computation-intensive MATLAB Script equation. If the computation is not completed within a few seconds, a cancellation dialog box will be displayed to:
  - indicate Genesys is still running the MATLAB Script equation, and
  - enables you to cancel the equation evaluation.
- For a workspace that has equations set to use retail MATLAB, MATLAB is launched the first time one of these equations is calculated. The launch may take some time, potentially a minute or longer. The user interface may appear unresponsive, and a dialog will be displayed indicating that MATLAB is starting. After retail MATLAB is launched, equation evaluation using MATLAB Script may become noticeably slower due the synchronization needed between the two modes.
- Sweep, optimization, Monte Carlo, and yield simulations may take longer time to complete if there are graph windows open that use MATLAB Script functions like extractsweptdata, sweepplot, histogram, etc. and the graph uses simulation outputs. When the graph window is open, the plot will be recomputed after each analysis. Close the graph windows to improve simulation speed. Note that Genesys uses [Engineering Language] by default, so this performance issue may happen only when MATLAB Script is used in the described scenario.

CAUTION

The following are the features in **MATLAB Script** that behave differently compared to Math Language (supported in releases prior to Genesys 2012.01):

- Tuning variable syntax = ? has been changed to tune.
- There is a set of Genesys specific MATLAB Script functions that do not work in the retail MATLAB mode. See to MATLAB Script Functions in Retail MATLAB Mode.
- Operations on variables with unit and/or independents (see to Defining Variables with Units and Independents) are in general slower than the variables with just primitive data types (For example, double). In general, variables obtained from the datasets by using the using function or the getvariable function has the associated unit and/or independents. To improve the performance, you can cast the variables to the corresponding primitive data types (For example, double(var)). The cast will lose the unit and independents and you can associate unit and independents back using the setdisplayunit and setindep functions.
- Expressions in MATLAB Script equation block without output suppression (semicolon ;), do not generate output in the Command Prompt. However, it is still recommended to use semicolon at the end of expression lines to improve the performance. For maximum performance, use semicolons to terminate MATLAB Script statements.
- Due to MATLAB Script limitation, Genesys can only be installed in a path that contains ASCII characters.
- getunits and setunits have been renamed to getdisplayunit and setdisplayunit. setdisplayunit and setindep do not support class variables.
- The tcpip workspace variable created from previous versions is incompatible with the current tcpip class format. In MATLAB Script mode, the tcpip class uses the MATLAB Script tcpip class implementation. In the retail MATLAB mode, it uses the tcpip from the MATLAB instrument control toolbox. The tcpip object variable stored in the workspace cannot be used interchangeably between MATLAB Script and retail MATLAB.
- The using function (in MATLAB Script) may behave differently for certain cases. See the using function document for details.
- The exist function (in MATLAB Script) no longer supports the checking variable existence in a dataset (the third argument), use the hasvariable function instead.
- The interp1 function (in MATLAB Script) requires the first argument to be a monotonic array.
- The isstring function (in Math Language) is no longer supported in MATLAB Script, use ischar instead.
- In Math Language, the default working directory is the workspace directory, so you may use functions such as fopen and readvector, and use relative path with respect to the workspace directory. However, MATLAB Script does not reset the working directory to the workspace directory. The recommendation is to use getworkspacedir to obtain the workspace directory then use it to construct the full path or use cd to display and change the workspace directory.
- The unsigned 16-bit integer arrays that were interpreted as strings by Math Language, is not compatible with MATLAB Script. For example, the script below works for Math Language but will generate an error in MATLAB Script, with the error message "Error using str2num, requires string or character array input."

  To make this work in MATLAB Script, change the second line as:

- Integers in MATLAB Script (as well as MATLAB) can only be combined with integers of the same class, or scalar doubles. For example, expressions such as:

  do not work. You need to cast the variables to proper data types. For example:

- If the MATLAB Script working directory is in C:\windows\system32, there is an error message like Invalid MEX-file 'C:/windows/system32/version.dll'. When this happens, you can use cd command to manually set the working directory to other locations.

NOTE

The default equation language in Genesys is still Engineering Language , same as the previous releases. You can optionally switch to MATLAB Script or MATLAB in a workspace equation, a design equation, or the command prompt using the right click menu.

Release Highlights: Use Model Update to Equations and Variable Viewers (2014.03 & 2015.08)

The use model of equations has been updated to more closely align with the MATLAB use model.

Workspace tree equations now behave like scripts. The variables defined in them have workspace-wide scope (no matter where in the workspace hierarchy the equation is located) and so they can be used anywhere in the workspace (other tree equations, schematics, design equations, graphs, tables). Workspace tree equation variables are shown in the Workspace Variables viewer. You can enable Workspace Variables viewer from the menu bar (View > Workspace Variables).

A global Command Prompt is now available for equation scripting. The Command Prompt can access all Workspace Variables . The results of command prompt expressions execution are shown immediately in the Command Prompt as well as in the Workspace Variables viewer.

In previous releases, a Workspace tree equation window contained an equation script editor, a variable viewer showing the variables defined in the equations, and a command prompt. Now a workspace tree equation window is just an equation script editor. Variables created from a Workspace tree equation are now shown in the Workspace Variables viewer. The global Command Prompt replaces the local equation window ones compared to the previous releases.

Similarly, a design equation window is now just an equation editor for the design. Design parameters (defined in the Parameters tab of a design) and variables created in a design equation are now shown in the Design Variables viewer. The Design Variables viewer displays the variables of the design that is currently in focus (active window) or the design that was last in focus (if the current active window is not a design). You can enable Design Variables viewer from the menu bar (View > Design Variables).

A design has its own Design Variables scope; the Design Variables (including design parameters and the variables defined in the design equation) are only accessible within the design. The design variables are automatically cleared before the design equation is evaluated.

Specifically, for the MATLAB Script language, a function object is provided to define a MATLAB Script function. A Workspace tree equation or a design equation that uses MATLAB Script can only define a script. However, it can not define functions. For more details, see Using Equations and Functions in a Workspace. Note that the use model of defining and using a function in Engineering Language remains the same as in the previous Genesys releases.

CAUTION

If multiple workspaces are opened at the same time using the same instance of Genesys and these workspaces contain different definitions of the same MATLAB Script function, the behavior is undefined due to function name conflict. In this case, do not allow multiple open workspaces (Tools > Options > General) and reopen the workspace that you want to work on.

Workspace Variables resolve an ambiguous situation where multiple Workspace tree equations can define the same variable with different values. In the previous release, the variable that is going to be used during an evaluation is undefined (assuming the equations that define the same variable are in scope). Now, the variable value is uniquely defined by the last equation that evaluates it and the value is displayed in the Workspace Variables viewer. The tooltip and "Go to definition" context menu can also be used to identify which equation defined the workspace variable.

Following are the tips specifically related to the equation use model update. See Tips for Effective Equation Writing for more details.

1. Ensure that the input and output equations are in separate blocks to prevent circular dependencies.
2. Be aware of variable redefinition, especially when it is redefined unintentionally because workspace tree equations share the same Workspace Variables scope. If it is desirable to define the same workspace variable in different workspace tree equations, consider turning off Auto-calculate f or these workspace tree equations and control their execution manually. If the intent is to define different variable values for different designs, it is recommended to use design equations instead of using multiple workspace tree equations. If the intent is to use multiple workspace tree equations for post processing, ensure the result variables use different names so they can be compared and plotted together.
3. It is no longer recommended to put "clear" at the top of a MATLAB Script equation. Workspace tree equations share the same Workspace Variables scope. Putting "clear" at the top of a workspace tree equation is going to clear **ALL** workspace variables, including the ones defined in other workspace tree equations. Since design variables are always cleared before a design equation is evaluated, there is no need to put "clear" at the top of a MATLAB Script design equation.

A new tutorial video, Equations_EngLang_and_MATLAB_Script, is created for introducing the new equation use model.

CAUTION

All variables have global scope even though they may exist in different equation blocks or located at different positions in the workspace tree. Furthermore, variables are persistent and are not removed until cleared by the user. Renaming a variable is a good example where the old variable name will still persist until cleared manually unless the equations were not executed between the renaming steps.

If you have existing VB scripts that access equation variables through Application.Manager.<Workspace Name>.<Equation Name>.VarBlock, you need to change it to Application.Manager.<Workspace Name>.**WorkspaceVariables**.VarBlock.

NOTE

Release Highlights: Use Model Update to Parameter Evaluation and Units (since 2014.03)

The use model of how a part parameter is evaluated has been updated to resolve the confusion between "Use MKS" and "Use Display" in previous releases. The "Use MKS" and "Use Display" options (provided at the top of the variable viewer frame of an equation window) have been removed from the product. In this release, when a part parameter has **units** and a variable (or an expression) is used to set its value, the following rules are used for parameter evaluation:

- If the variable (or the result of the expression) does not have an associated unit, the unit of the part parameter **is applied** to the value. For example, suppose a variable "var" is 100, and suppose the parameter Value is set to "var" and the parameter Units is set to MHz, the resulting parameter value is 100 MHz (100e6).
- If the variable (or the result of the expression) has an associated unit, the parameter value is simply the MKS value of the variable and the parameter Units **is not used** to interpret the parameter value (in this case, the parameter unit selected is used **ONLY** for display purposes, e.g. to show the parameter value on the schematic). For example, suppose a variable "var" is 100 Hz (MKS value 100 with display unit Hz), and suppose the parameter Value is set to "var" and the parameter Units is set to MHz, the resulting parameter value is 100 Hz (MKS value 100) and is displayed as 100e-6 MHz.

See Using Variables or Design Parameters to Set Part Parameters for more details.

Workspace Compatibility and Workspace Conversion Tool (since 2014.03)

The workspace file format has been changed to improve file integrity due to all backward incompatible use model changes made in this release. The workspaces created using the previous Genesys releases are automatically converted when they are opened. The original workspace is not modified unless you overwrite it by saving (File > Save) the converted workspace.

The workspaces created using Genesys 2014.03 and later are incompatible with 2012.01 and prior releases.

CAUTION

When an old format workspace is loaded and converted you will see an **Information** message in the **Errors** window notifying you that a conversion has happened.

NOTE

If there are any errors encountered during the conversion, a **Conversion Log** note object is created on the workspace tree; this note contains details about the conversion errors encountered.

When loading old XML libraries such as Design, Part, Equation libraries, a dialog box is automatically displayed to convert the old XML library to the new format. You will need to provide a name for the new library and the converted library will be automatically loaded. The old libraries remain unchanged so you can still load them in previous versions. You can also run the conversion tool (see below) outside the product to convert your XML libraries to the new format.

A command line utility program, **WorkspaceConversionTool.exe**, is provided to convert multiple workspaces and XML libraries in a directory. The workspace conversion tool can be found in the **<Genesys_Install_Directory>\bin** directory.

The command line syntax of the conversion tool is like: **WorkspaceConversionTool.exe "SourcePath" "DestinationPath"**.

For example: `WorkspaceConversionTool.exe "C:\Examples\WorkspaceA_For2012_01.wsx" "C:\Examples\Converted\ConvertedWorkspaceA_For2015_08.wsg"`

The conversion tool also works with directories. All workspaces in the source directory and any sub-directories are converted to the destination directory. Non-workspace files in the source directory are copied to the destination directory.

For example: `WorkspaceConversionTool.exe "C:\Examples\MyWorkspaces" "C:\Examples\ConvertedWorkspaces"`

The workspace conversion tool converts the following items to match the new use model in the current product:

1. Convert part parameter units and equations, if necessary, to preserve the same results with the new use model of parameter evaluation. See Use Model Update to Parameter Evaluation and Units.

    a. If a part parameter value is set to a variable (or an expression using a variable) and the variable is defined in a "Use MKS" equation and the variable does not have an associated unit in the original workspace, the part parameter unit will be set to the MKS version of that unit category (for example, from MHz to Hz).
        - If it is not desirable to change part parameter unit, there is a command line option, -ScaleEquationText, in the workspace conversion tool to associate MKS display unit with the variable at the end of the equation that defines the variable. For example, WorkspaceConversionTool.exe C:\tmp\source_directory\workspace1.wsv C:\tmp\destination_directory\workspace1.wsv -ScaleEquationText
    b. If a design equation is set to "Use Display" and there are design parameters that have associated units, additional lines will be inserted at the top of the design equation to strip out design parameter units and adjust design parameter values such that the design behaves the same as before.
2. Convert workspace tree Equation object that defines a Math Language function to a workspace tree Function object that defines a MATLAB Script function. The workspace conversion tool only handles the case where the first valid line in the original equation defines a function. If an equation block defines multiple functions, then only the first function will be converted to a function object. The other function(s) will exist only in the local scope of the converted function.
3. Convert setunits and getunits in Math Language to setdisplayunit and getdisplayunit in MATLAB Script. It also converts the old unit specification in setunits to the new unit specification in setdisplayunit.
4. Convert tuning syntax =? in Math Language to tune in MATLAB Script.
   If there is a "clear" function call at the top of a MATLAB Script equation, the line will be commented out because it is no longer recommended to do so in the current version. Workspace tree equations share the same Workspace Variables. Putting "clear" at the top of a workspace tree equation is going to clear all workspace variables, including the ones defined in other workspace tree equations. Since design variables are always cleared before a design equation is evaluated, there is no need to put "clear" at the top of a MATLAB Scriptdesign equation.
5. Remove old tcpip variables in Math Language from the workspace because the current tcpip class format is incompatible with the previous version. You can rerun the equation to create the tcpip variable for the current version.
6. Convert "isstring" function in Math Language to "ischar" because "isstring" is no longer supported in MATLAB Script.

The workspace conversion cannot preserve the behavior for the following cases and will report conversion errors or warnings after loading the converted workspace.

1. If there are duplicated variables defined in multiple tree equations in the original workspace, only one of them will be displayed in the Workspace Variables viewer. See section "Use Model Update to Equations and Variable Viewers".
2. If there is a variable defined in a workspace tree equation that "Use Display" and the variable is associated with a unit (for example, through setunits or coming from a dataset) and the variable is used to set a part parameter, the behavior could be different than the previous release.
3. If a Math Language equation should be converted to a MATLAB Script function object, but the function name is already in use by another workspace tree object then the function object will not be created. The equation object cannot be renamed to another workspace tree object with the same name.
4. If in Math Language functions like "fopen" and "readvector" use relative file path, make sure the working directory is set properly. You can use 'cd' to display and change the working directory and use 'getworkspacedir' to obtain the workspace directory.

The unit use model has been updated since Genesys 2014.03 and is not compatible with 2012.01 and prior versions. The workspace conversion tool can convert prior workspaces to match the new unit behavior for most cases. However, here are the limitations:

- If the original workspace uses setunits to set the unit of a variable and such variable is used directly or indirectly (through expression) to set a part parameter, the value of the part parameter might be different than previous versions.

Release Highlights: Licensing

- Genesys 2015.08 requires: a) version 2015.05 of the EEsof EDA licensing software and b) version 3.2 codewords to run.
- New date-based version licenses are required for this Genesys 2015.08 release. The new license will also be backward compatible with 2 previous releases of Genesys.
- Genesys does not automatically release licenses after checking out. Starting in Genesys 2014.03, there is a new menu button **Action > Release All Simulation Licenses** for users to manually release all checked out simulation licenses.

Product Improvements and Updates

General

- All RF ports now have a frequency parameter to support frequency dependent port impedances.

Use Model Update to Equation Auto-Calculate

- If the variables defined in an equation are not used anywhere in the workspace (graphs, designs, tables), then the equation will not calculate (execute) when the workspace is opened even if it is set to Auto-calculate.

3D Viewer

- Upgraded to latest ADS 2016.01 3D viewer code. Several defects have been addressed.

Momentum

- Upgraded to the latest ADS 2016.01 Momentum code. Several defects have been addressed.

Layout

- Saving user created (or updated) footprints and layer libraries needing Administrative privileges has been resolved.Starting with **Genesys 2015.08** the default path of the libraries (in **Genesys Global Options**) has been changed from **c:\Program Files\Genesys<**Version**>\lib** to **My Documents\My Models\lib**. If the folder does not exist - it's created automatically on the first run of Genesys 2015.08 and will copy all files from **c:\Program Files\<Genesys Version>\lib** to **My Documents\My Models\lib**. Any following Genesys installations synchronizes the folders only adding files newer than the ones from new Genesys installation folder **c:\Program Files\<Genesys Version>\lib**. An additional benefit of the change is that all new Genesys installations starting from **Genesys 2015.08** will share the same user created footprint and layers libraries and there is no need for any additional actions from the user to copy them as required in previous Genesys releases.

EECAD

- Upgraded to latest ADS 2016.01 EECAD code. Several defects have been addressed.

Monte Carlo, Optimization and Yield

- Updated to use frequency dependent vectors for data. For example, if Monte Carlo analysis was running on a frequency dependent gain vector the perturbation in gain ratio would be applied to every gain value. Therefore each gain value would be perturbed by the same gain ratio.

Testlink

- Testlink has been updated to the latest version and now support Keysight instruments.
- A new Testlink example has been added as Testlink Example.wsg.

Verilog A Compiler

- Verilog A compiler have been updated to the latest version.

Spectrasys

New Non-Linear Digital Step Attenuator Model

A non-linear digital step attenuator part, model, and schematic symbol has been added.

New Limiter Model

A limiter part, model, and schematic symbol have been added. There is a new shipping example \Limiters\Limiter Basics.wsg showing the characteristics of the limiter.

Reference Impedance Standardization

All models that need a reference impedance now use a common parameters called **Zref**. In the past some models used Zo or ParamZ.

The models that were updated are:

- CIRCULATOR
- DELAY
- NonLin
- NonLinHO
- PHASE
- SDATA_NL
- SDATA_NL_HO
- SDATA_NLI

Multisource Model Updates

The Multisource now supports a Frequency Comb source as well as a few Digital Modulation sources. The Multisource will now allow users the ability to mark a signal as an undesired signal. This gives users the ability to create interfering signals from the same Multisource where desired signals are created. The Multisource now supports frequency dependent port impedance.

Mixer S Parameter Accuracy Improvements

The mixer models have a new parameter called Input Port (InPort) that is used to identify the primary input port of the mixer. This information is necessary to improve the S parameter accuracy of the mixer. The linear mixer model now support Conversion Gain in the forward direction and RF to IF Isolation in the reverse direction. This enables higher accuracy load impedances seen on the mixer input port due to varying mixer load impedances.

NOTE

In simulation intensive workspaces like sweeps, monte carlo, and optimization of the a system design the simulation speed can generally be improved by turning off the 'Use Volterra Model' option located on the Calculate tab of the System Analysis. Turning off this option can decrease intermod accuracy.

New DC Power Supply Parameters and Measurements

Current and voltage power supply parameters have been added to the active behavioral models. DC power supply measurements have been added to the main system analysis dataset that show users power supply requirements for the entire design. This includes the calculated DC supply power.

Other Spectrasys Improvements

- Added an SFDR parameter to the ADC that sets the level harmonics generated by the ADC.
- Added ability for ADC spurs to be set either in dBc or dBFS.
- Improved the ADC doc.
- Improved the residual phase noise of the amplifier to only work on desired input spectrum and harmonics created from desired spectrum.
- Fixed many issues with cascaded X parameters models.
- Improved the accuracy of Phase Noise Channel Power (PNCP) measurement.
- Improved the accuracy of Output Saturation Power (EOPSAT and OPSAT) measurements.
- Add noise figure parameter to all SDATA_NL models

New Spectrasys Examples

- AM to PM Basics.wsg
- Limiter Basics
- Monte Carlo Spur Analysis.wsg
- Monte Carlo Sweep.wsg
- Phase Noise Channel Power.wsg
- Quick Sweep Monte Carlo.wsg

Synthesis

- The following new synthesis videos have been created:
  - Passive_Filter_Synthesis
  - Active_Filter_Synthesis
  - Signal_Control_Synthesis
  - Equalized_Network_Synthesis
  - TLine_and_Advanced_TLine_Synthesis
  - Mixer_Synthesis
  - Oscillator_Synthesis
  - PLL_Synthesis
  - WhatIF_Frequency_Planning

  These videos can be found in Genesys by selecting Help > Tutorial Videos.

- Implemented Synthesis Script Verbs to calculate Synthesis objects and get their text messages from Genesys script processor.

Documentation Improvements

- Spectrasys Cascaded Noise Figure
- Spectrasys Filter Models
- Spectrasys Mixer Models

Obsolete Features

Old .wsp and .sch Workspace File Formats (since 2014.03)

Old .wsp and .sch Workspace File Formats The .wsp and .sch workspace file formats (which are over 10 years old) are not supported anymore. To load an old workspace, you must first load it in Genesys 2012.01 and save it as a .wsx file and load that into Genesys 2014.03. If you don't have a copy of Genesys 2012.01 and need to install it, download it from our website.

We have created a simple tool called 'WSP to WSX Converter.exe' that will aid in bulk file conversions. This tool uses an installed version of Genesys 2012.01 to do the conversion. This tool allows the user to point to a .wsp input directory and a .wsx output directory. The tool can be found in the **bin** directory of the install. i.e. C:\Program Files\Genesys 2015.08\bin\WSP to WSX Converter.exe. Documentation for the tool is located in the startup screen of the tool.

File > Send as Email

Under the File menu, the Send as Email option has been obsoleted. Instead, you can simply attach a saved workspace to your email.

Resolved Issues

- Addressed many issues associated with cascaded X parameter models.
- Fixed an issue with the function 'extractsweptdata' so that data can be extracted from nested sweeps
- Resolved several defects for stability
- Improved usability by adding several enhancements
- Resolved an issue of using local design substrates in the design equations

Known Issues

- The graphName parameter for the graph_figure() function cannot have any trailing blank(s). The best practice for graphName specification is to remove both leading and trailing blank(s).
- Antivirus on your system may remove one or more files installed during the installation leading to a corrupt installation.

  Workaround: Disable antivirus and other related programs on your system before installing. You can enable them once the installation is complete.
- When you use the reference info button to view component SData file from vendor libraries, if Genesys pops up SoftPlot, this is because the following file extensions: s2p, s3p, s4p, are associated with SoftPlot. You can manually change the file extension association in Windows to use the desired program to open SData files.
- ADS 2015.01 (and earlier) does not recognize Genesys 2015.08 automatically during ADS-Genesys link. The workaround is to create a "DefaultPath" string value under registry key HKEY_CURRENT_USER\Software\Eagleware\Genesys2015.08_x64\System\ and set the value to the path defined in HKEY_CURRENT_USER\Software\Eagleware\Genesys2015.08\System\DefaultPath.

Supported Platform

- Windows 10 Support: While not officially supported, preliminary tests of Genesys 2015.08 installation on Windows 10 did not reveal any new issues.
- Refer to Geneys 201403 Supported Platforms for the platforms that are officially supported.