強力なスクリプティング機能
数式エディタ・デバッガ活用法-1

・階層構造とスコープ
・デバッガの基本的利用法
のご紹介

EAGLEWARE

はじめに

製品の周波数特性デバッグを行う上で、ネットアナなどで取得した Touchstone,Citi ファイルをシミュレータ上で読み込み、演算させ、所望の値を見つけ出したり、その結果をグラフ表示させたりしたい場合などがあるかと思います。
Genesys の数式エディタ・デバッガ機能を利用することで、このようなことを比較的簡単に実現することが可能になります。

本アプリケーションノートでは、数式を記述するときに特に重要な、階層構造とスコープ、デバッガの基本的利用法、数式のコツなどについてご紹介いたします。

紙面の都合上、基本的な Genesys の操作については触れていません。

サポートされる２つの言語

Genesys の数式では、2008.01 より２つの言語がサポートされております。

1. エンジニアリング言語[EngLang]
   測定値、解析値を扱いやすいように環境が整えられている、言語体系です。
   単位を持たせたり、S パラデータなどの測定値・解析結果の配列変数に独立変数（時間軸解析時は時間、周波数解析時は周波数など）を持たせることができます。ゆえに、グラフへの表示も単位、横軸を気にすることなく簡単に行うことができます。
   S パラメータなどを扱うために必要な関数群（stoy など）が充実しています。

2. 算術言語[MathLang]
   EngLang が測定値、解析値を念頭に設計されているのに対して、MathLang はもっと幅広い学術用途に利用できるように配慮された環境を提供します。
   TCP/IP を利用した計測器⇔Genesys の通信環境が準備され、数学演算関数、DSP のための窓関数なども EngLang より多く、あわせて 200 種類ほど用意されています。

   フレキシブルな演算環境を用意している反面、行列内の要素へアクセスする場合、C/C++のように厳密に要素を指定する必要があるなど、EngLang とは異なる性質があります。

２つの言語により書かれた数式を含む WorkSpace 内で、お互いの値を変数で共有することができますが、EngLang が MathLang の関数、サブルーチンをコールすること、その逆は、文法とパーサーの違いにより許されません。

階層構造とスコープ

ワークスペースツリー内部には、任意のフォルダ＝階層を作成できます。フォルダによる階層構造の中に数式を作成すると、フォルダ内、外、特に上位、下位に変数と関数を参照できる「スコープ」ができます。図１にその概念と、図２にその実例を示します。

図 1 のように、下位の階層から上位の階層の変数、関数を覗くことができますが、その逆はできません。
また、同じ階層の場合、同等に参照することができます。

図 2 は図 1 の構造を実際の数式で表現したものです。フォルダ Bottom 内部の BottomEq が数式本体、BottomEq2 が関数です。

図２の具体的動作

図２は MathLang を利用した場合のスコープについて説明しています。

スコープの概念は、EngLang と MathLang において同じです。

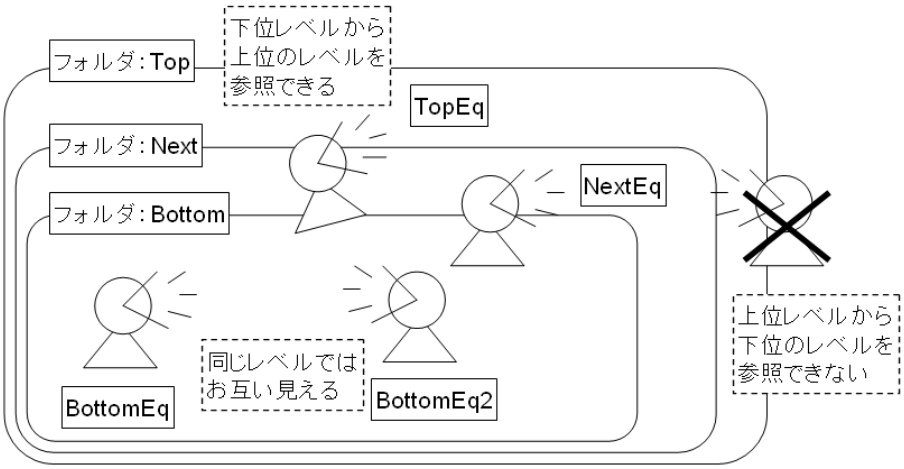

図1　階層構造によるスコープ

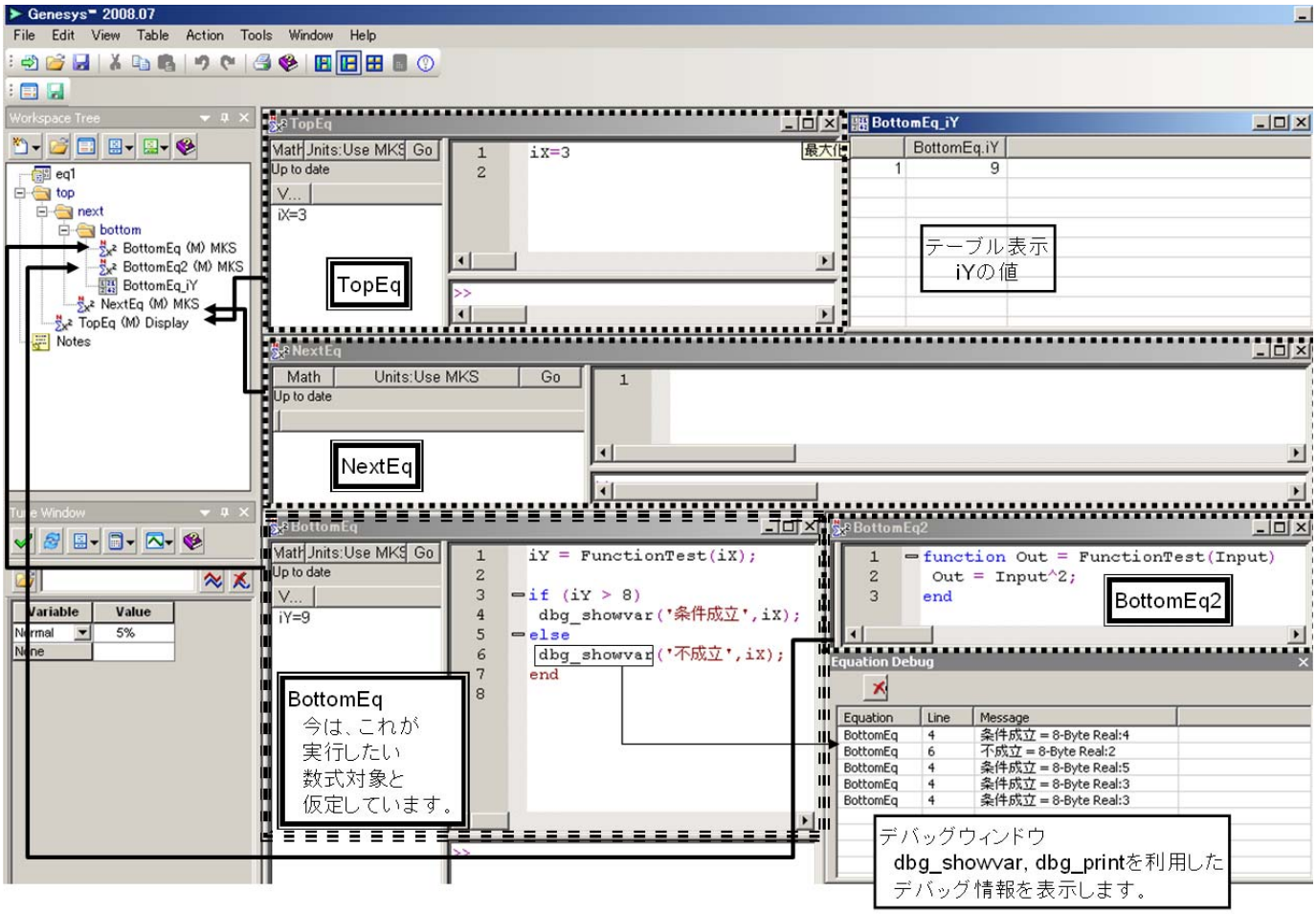

図2　階層構造における数式とその振る舞い

BottomEq 内部で参照される変数 iX は、BottomEq には存在しません。それで、パーサーは階層 Next を参照しようとしますが、そこには iX が存在しないため、その上の階層 Top を参照します。そこには TopEq という数式があり、その中に、iX の定義があります。それを参照し、関数 FunctionTest を呼びます。パーサーは FunctionTest を探しますが、同じフォルダ（スコープ内）の BottomEq2 に FunctionTest のエンティティをみつけ、iX を引数として呼び出します。BottomEq で、FuncitonTest の返却値が変数 iY へ代入されます。

if 文による比較が行われ、iY が 8 よりも大きければ dbg_showvar()関数により、EquationDebug ウィンドウ（図２右下）に変数とその情報を表示させます。また、条件を満たさない場合でも、dbg_showvar()関数で条件を満たさない旨の情報を表示させます。

フォルダ Bottom 内部にあるこれら２つの数式から、フォルダ Next、Top 内の数式を参照できます。

言語の切り替えと単位系

ご紹介した 2 つの言語は図 3 のように切り替えることができます。
また、単位系は 2 種類あり“MKS”“Use Display”があります。

前者はすべて MKS 単位系で数値を扱うと宣言することで、たとえばインダクタンス nH を表記するのには、デフォルトがヘンリーのため、nH にするために e-9 をつける必要があります。

後者は、回路図上で利用されている単位をそのまま利用します。もし、回路で nH が利用されている場合、10 という数式の値を回路に利用すると 10nH になりますが、もしも uH の場合は、数式上はおなじでも、回路では 10uH となります。

数式をライブラリなどにして、再利用を考えている場合は、どんな回路図なども利用可能な MKS 単位が間違えが少なくおすすめします。
一方で、ちょっとした計算、もしくは特定の WorkSpace 内部で利用するのであれば、Use Display の方がシンプルに数式を表現できると思います。

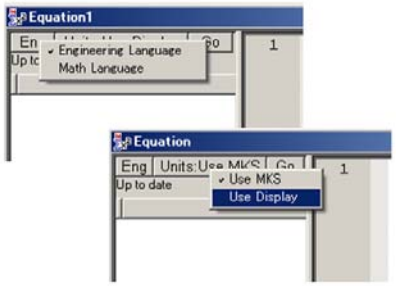

図3　２つの言語・単位系の切り替え

デバッグ方法

ここでは、デバッグの方法について簡単にご紹介します。

制御

図 4 に EngLang を利用した数式のデバッグ環境を示しました。Equation ツールバーには数式の実行方法を制御するボタンが用意され、**ブレークポイント**を置くことで強制的に停めることができます。また、**StepInto**、**StepOver** を利用して、呼び出した関数の中に入り込んだり、関数を超えて次の処理に進んだりと実行行のマーカーを制御できます。

デバッグ時の変数の変更

数式ウィンドウの最下部にあるコマンドラインでは、変数を直接代入（I=50 など）することで、実行中の変数を変更することができます。特に、Loop の時のカウント値を変更させる場合などに有用です。

デバッグ情報の取得

デバッグ時に有益な情報を与えてくれるのが、ウィンドウ左側の変数一覧と、別ウィンドウで表示される EquationDebug ウィンドウです。（EquationDebug ウィンドウがでない場合は、Ctrl+Shift+d もしくはプルダウンメニュー View/Docking Window/Equation Debug Window で表示できます。）

変数一覧では、実行時の変数の値と変数の形式（行列、次元、大きさ、虚数など）が表示されます。

EquationDebug ウィンドウには、前章でご紹介した dbg_print, dbg_showvar などのデバッグ専用コマンドを利用して、現在実行中の内部情報を書き出すことができます。（文法などはマニュアルを参照ください。）

また、図 5 のように、変数を右クリックすると、テーブル、直交座標形式のグラフをすぐに表示することが可能です。この機能により、加工された変数行列が正しいものかどうか等を視覚的に確認することができます。

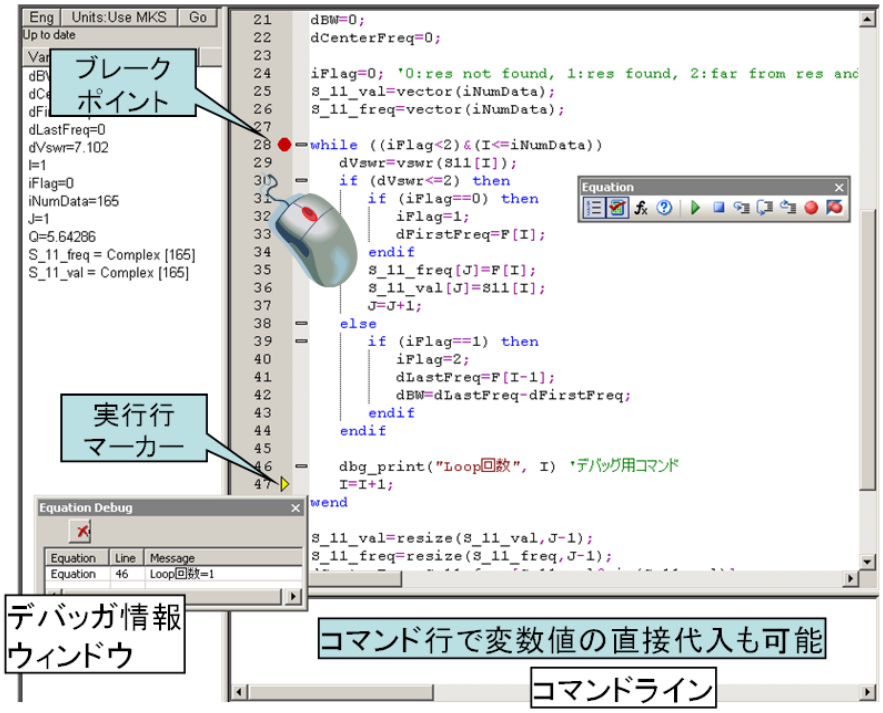

図4　デバッグ環境

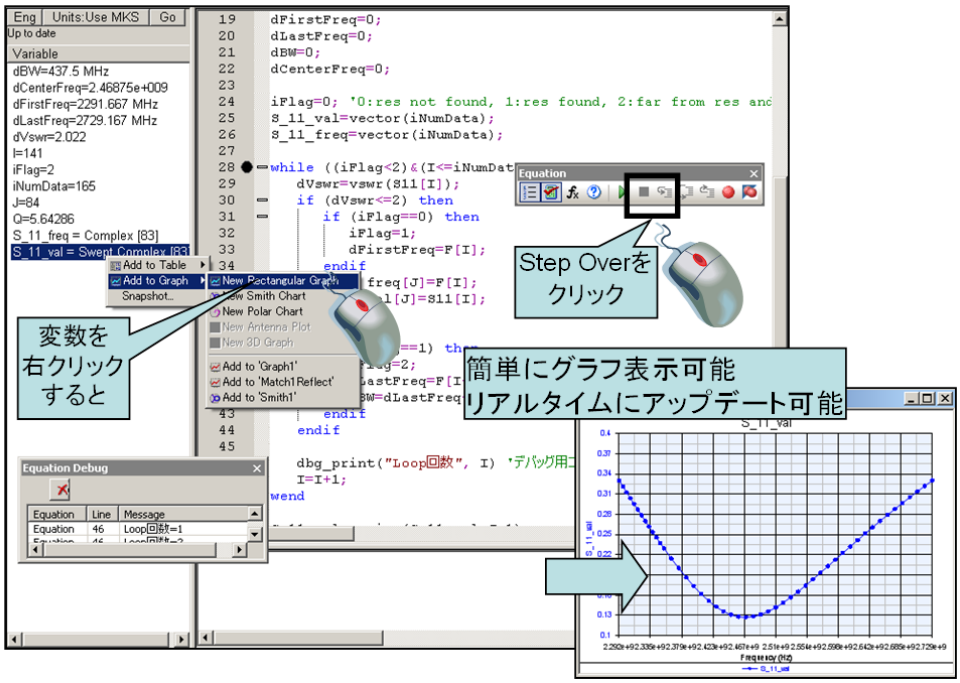

図5　変数のグラフへの表示方法

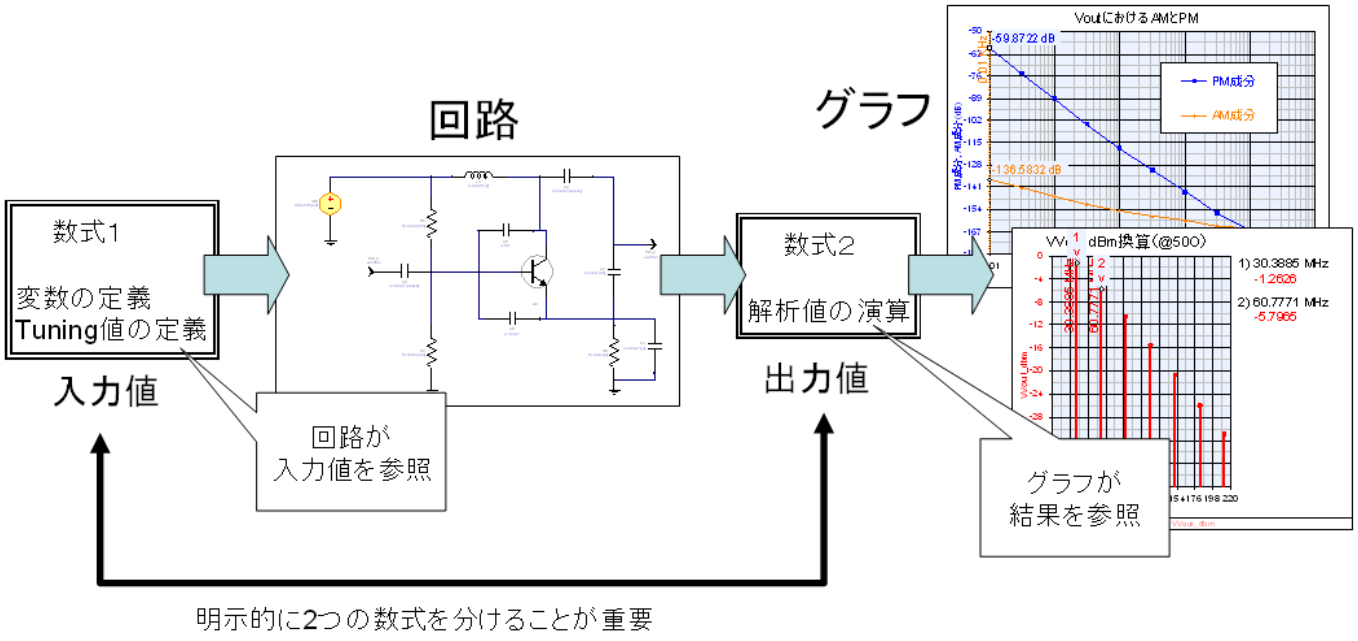

図6　数式を明示的に分ける必要がある場合

数式を活用するコツ

マニュアルでは“Tips for Effective Equation writing”の項に別だしして書かれているように、数式を活用する上で、知っていると便利なコツがいくつかあります。
ここでは、2つほどご紹介します。
是非、マニュアルも参照ください。

数式を明示的に分ける

図 6 に示したように、数式から、回路に対して変数を渡し、解析結果をさらに数式で計算する場合は、パーサーの動作仕様より、“入力”“出力”の数式を明示的に分ける必要があります。
もしも分けて書かれていない場合、動作が極端に遅くなったり、出力する演算結果の値が更新されないなどの誤動作を生じます。

関数への引数の渡し方

複数の引数を関数へ渡すことができます。MathLang の例を図 7 に示します。EngLang、MathLang 共に、外部変数の参照が許されていないため、必要な情報は引数で渡す必要があります。そこで、複数の引数を渡す文法を知っていると、便利です。

まとめ

今回のアプリケーションノートでは、Genesys の便利な数式エディタ・デバッガの機能をご紹介いたしました。
紙面の都合上、全機能をご紹介できませんでしたが、基本的な文法を理解いただくと、S パラ Y パラ変換、キャパシタンスの抽出・・・など、いろいろな活用アイデアが生まれてくると思います。

本アプリケーションノートが、ユーザ様の設計に少しでも貢献できれば幸いです。

```
1  %MathLangの例
2  function ind = ResL(C, F)
3    FHz = 1e6 * F
4    CFarads = 1e-12 * C
5    Omega = 2 * pi * FHz %piは予約語
6    LHenries = 1 / (Omega^2 * CFarads)
7    ind = LHenries * 1e9    %返却値
8  end
```

関数宣言の中で、外部の変数の参照を許されるのは、引数として渡されたものだけです。
ここではC,Fで、返却値はindになります。
MathLangの場合は、[a,b,...]のリストを利用して、複数の値を返却値とすることもできます。

図7　関数へ複数引数の渡し方

参考文献

[1] Genesys ヘルプ,2008.07 version, Agilent Technologies

ワークスペース一覧

本アプリケーションノートで利用している Genesys ワークスペースファイルは弊社の Web よりダウンロードできます。以下の一覧は、ワークスペースファイル名と解析内容との対応を示します。

1. S11 から Q、帯域を求める例

   eq_example.wsx

すべてのワークスペースは、Genesys2008.07 を利用して作成されています。

改訂履歴

初版　2008 年 10 月

アジレント・テクノロジー株式会社
本社 〒192-8510 東京都八王子市高倉町9-1

計測お客様窓口
受付時間 9:00~19:00
(土・日・祭日を除く)
※FAXは24時間受付け

TEL 0120-421-345
(0426-56-7832)
FAX 0120-421-678
(0426-56-7840)
E-mail:contact_japan@agilent.com

Agilent EEsof EDA ホームページ
http://www.agilent.co.jp/find/eda