PLECS
Tutorial

# Creating Custom Components with the PLECS Subsystem Block

サブシステムブロックによるカスタムコンポーネントの作成

-PLECSのマスキングコンセプトと、独自のカスタムコンポーネントモデルを作成する方法についての学習-

Tutorial Version 1.0

KESCO KEISOKU ENGINEERING SYSTEM
計測エンジニアリングシステム株式会社
https://kesco.co.jp

# 1 はじめに

この演習では、マスクした**サブシステム**を使用して、太陽光発電(PV)ストリングのカスタムコンポーネントモデルを作成します。**サブシステム**をマスクすると、サブシステムブロックのカスタム インタフェースを定義して、下層のる回路図を非表示にし、独自のアイコンとダイアログ ボックスを持つ一つのコンポーネントとして表示することができます。PLECSライブラリ内の多くのコンポーネントは、例えばすべての電気機械モデルなどの実際にはマスクされた**サブシステム**です。

このチュートリアルでは、次の内容を学習します:

- PLECSでカスタムコンポーネントを作成する方法
- カスタムコンポーネントにパラメータダイアログを追加する方法
- 動的マスクの定義に使用するスクリプト言語Luaの簡単な紹介

**始める前に:** 演習の各段階で作成したモデルと、参照モデルとを比較して確認します。

# 2 PV電流特性を実装

PVストリング モデルは、同じ電気特性を持つ任意の数の直列接続されたPVモジュールで構成されています。PVモジュールは次のモジュールパラメータによって記述されます:

- 短絡電流$I_{SC}$(A)
- 開放電圧$V_{OC}$(V)
- 最大電力点電流$I_{MPP}$(A)
- 最大電力点電圧$V_{MPP}$(A)

単一のPVモジュールの出力電流特性は、電圧と太陽放射照度(太陽の強さ)の関数です。電流特性は、[1]で示した太陽光発電(PV)モジュールの簡略化されたモデルに基づいて、シミュレーション開始時に**モデル初期化コマンド**から事前に計算されます。PLECSでは、PVストリングは非線形電流源としてモデルリングします。電圧は PVストリングモデル自体からの内部フィードバック信号であり、太陽放射照度はユーザが指定する外部パラメータです。事前に計算されたPV電流特性は、ルックアップテーブルに読み込まれます。

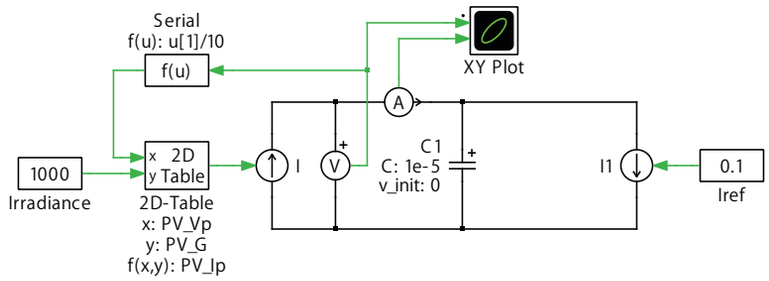
**あなたのタスク:**

**1** モデルを新規作成し、**シミュレーション・パラメータ**(**Ctrl + E**)ウィンドウの**初期化**タブの**モデル初期化コマンド**フィールドに以下のコードスニペットをコピーします。このコードは、一般的なPVパネルのデータシートに記載されている4つの値(開回路電圧、短絡電流、最大電力点(MPP)での電圧、MPPでの電流など)に基づいて、PVモジュールの非線形電流特性を計算します。

```
Voc = 40.7;
Isc = 10.04;
Vmpp = 33.2;
Impp = 9.49;
% Simplified Model of a Photovoltaic Module, A. Bellini et al.
PV_G = 100:100:1000;
PV_Vp = 0:Voc/100:Voc;
Isc = Isc*PV_G/1000;
Impp = Impp*PV_G/1000;
for j = 1:length(PV_G)
    C2 = ((Vmpp/Voc)-1)/log(1-Impp(j)/Isc(j));
    C1 = (1 - Impp(j)/Isc(j))*exp(-Vmpp/(C2*Voc));
    PV_Ip(:,j) = Isc(j).*(1-C1*(exp(PV_Vp/(C2*Voc))-1));
end
```

2 次に、図1に示すモデルを構築します。非線形出力特性を定義するために必要な2Dルックアップ・テーブルブロックは、"制御器ブロック"の"関数&テーブル"にあります。

*図1: PLECSによる電圧および太陽放射照度制御電流源としてのPVストリングモデルの実装*




3 単一のPVモジュールではなく、10モジュールのPVストリングの電圧特性をモデリングするには、**電圧計**と**ルックアップ・テーブル**の間のフィードバックループに**関数**ブロックを配置し、式`u[1]/10`を入力して電圧特性を10倍に拡張します。

4 PV電流の負荷電流に対する状態依存性をなくすために、**電流源(可変)**と並列に**キャパシタ(理想モデル)**を接続する必要があります。そうしないとシミュレーションエラーが発生します。"容量"は`1e-5`Fに設定してください。

5 シミュレーションを実行する前に、**シミュレーションパラメータ**の**ソルバ**タブで次のように設定します:

- 終了時間: `1e-3`秒
- 相対誤差: `1e-6`
- ソルバ: `DOPRI (nonstiff)`

シミュレーションを実行すると、1000W/m$^2$の放射照度の出力IV特性が表示されます。PVモジュールの出力は、この放射照度値が表すピーク太陽光条件に基づいて評価されることに注意してください。ここでは放射照度を0～1000W/m$^2$で使用する必要があります。

この段階では、モデルは参照モデル`custom_components_1.plecs`と同じになっているはずです。

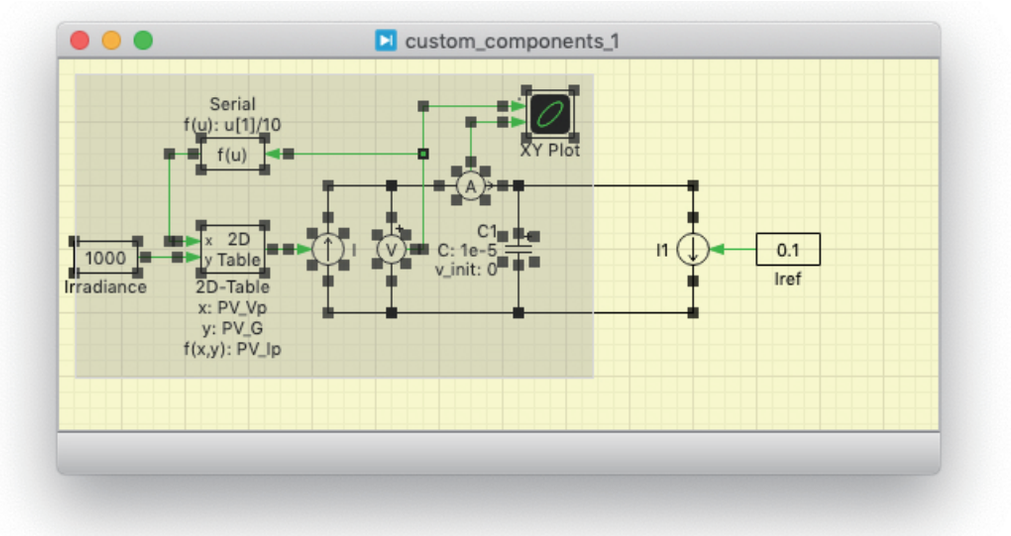
# 3 サブシステムの作成

次のステップは、PVモデルを**サブシステム**に移行させることです。サブシステムは、回路の階層的なモデリングや再利用可能なカスタム コンポーネントの作成に役立ちます。

**あなたのタスク:**

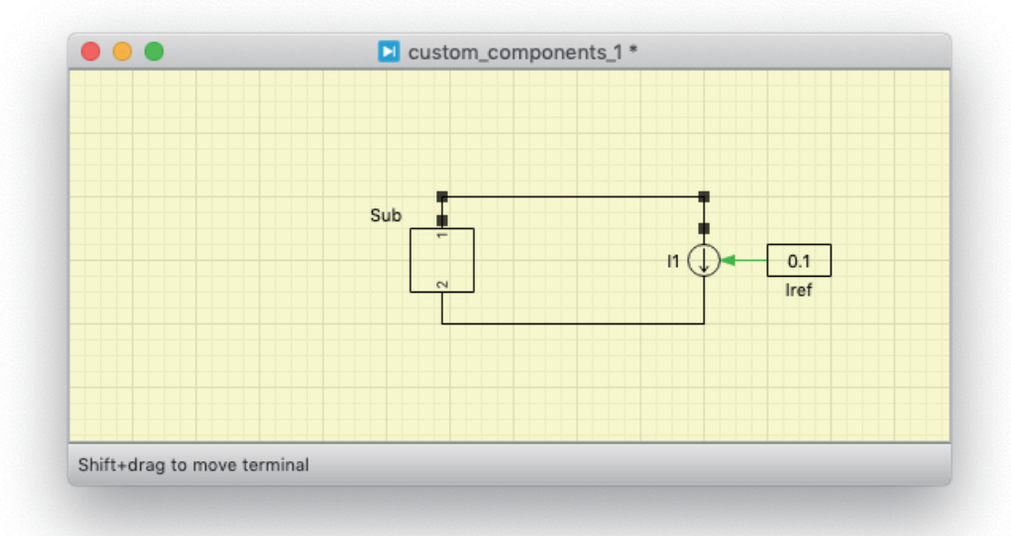
1 図2のように、**サブシステム**に含めるコンポーネントを選択し、右クリックして**サブシステムの作成**を選択するか、**Ctrl + G**を使用します。

*図2: サブシステムに含めるコンポーネントを選択する*




2 サブシステム端子の位置を変更するには、**Shift** キーを押しながら、サブシステムマスク境界のすぐ外側にある接続ワイヤを左クリックします。カーソルが✋に変わり、端子の位置が移動できることを示します。PVサブシステムが図3に示すような位置になるように端子の位置をドラッグして調整してください。次に、サブシステム名を左上隅にドラッグします。

*図3: Shiftキーを押しながら端子をドラッグしてターミナルの位置を変更*

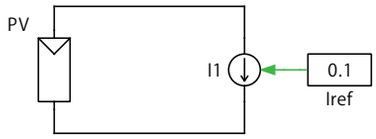



3 カスタムのサブシステム シンボルを作成するには、右クリックして**サブシステム** -> **マスクの作成...**を選択するか、**Ctrl + M**で**サブシステムのマスク エディタ**ウィンドウを開きます。**アイコン**タブで**端子ラベルの非表示**オプションを選択すると、ポートの ラベルがアイコンに表示されなくなります。**言語**でLuaが選択されていることを確認します。次に、**描画コマンド**ウィンドウに次のように入力します:

```
Icon:line({-10, 0, 10},{-20, -7, -20});
```

4 その後、角が描画した線の端点と合うように**サブシステム**のサイズを調整する必要がある場合があります。これを行うには、**サブシステム**を右クリックして**サブシステム** -> リンクの非保護を選択し、**サブシステム**の保護を解除する必要があります。最終的なPVサブシステムのシンボルは、図4のようになります。

*図4: 描画コマンドを使用してサブシステムのマスクをカスタマイズ*




**注意:** サブシステムブロックにマスクアイコンを定義すると、PLECSはブロックとその基礎となる回路図を自動的に保護します。その後、サブシステムブロックのサイズを変更したり、サブ回路図を変更したりできなくなります(最初に保護を解除しない限り)。この保護の目的は、ユーザが意図せずアイコンを変更して使えなくなってしてしまうことを防ぐためです。

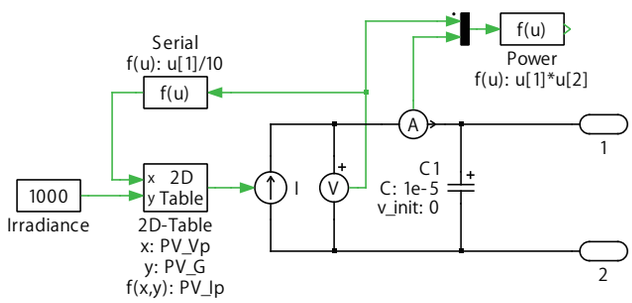
# 4 サブシステムのマスクにプローブ信号を追加

PLECSプローブブロックを使用してカスタム プローブ信号を追加し、サブシステム内のキー値を直接監視できるようになります。サブシステム内で測定または計算された量は、サブシステムマスクのカスタムプローブ信号リストに追加できます。この例では、PVストリングのサブシステムマスクに追加するプローブ信号は、Voltage、Current、およびPowerです。

**あなたのタスク:**

**1** まず、サブシステム内の出力電力を計算して、この信号をプローブ信号のリストに追加できるようにします。マルチプレクサブロックを追加し、電圧信号と電流信号を組み合わせて、関数ブロックを使用して電力を計算します。電力計算機能を追加して修正したサブシステムを図5に示します。

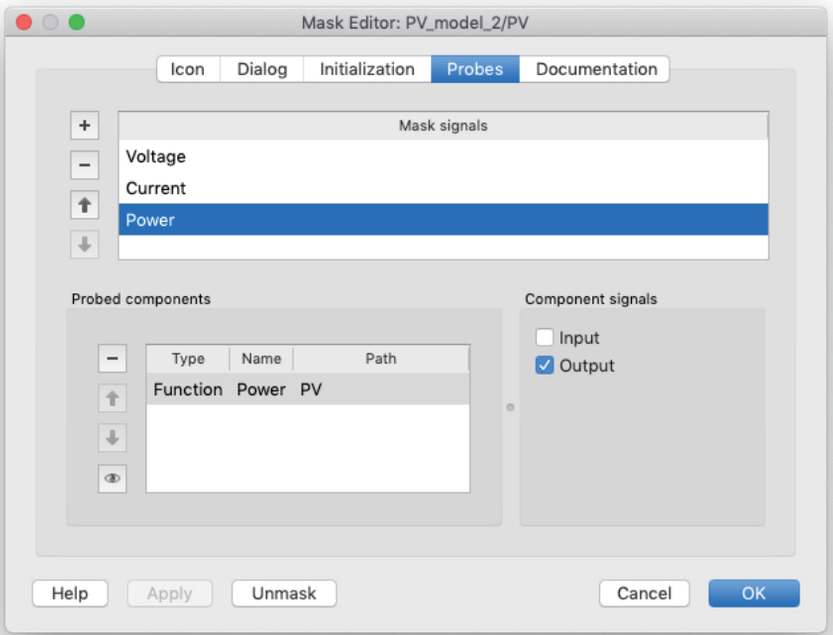
*図5: サブシステムに電力計算を追加して、出力電力をサブシステムのプローブ信号に追加*




**2** プローブ信号リストを作成するには、PVのサブシステムアイコンを右クリックし、**サブシステム -> マスクの編集...(Ctrl+M)**を選択して**マスクエディタ**ウィンドウを開きます。プローブ信号を定義する前に、**文書**タブを選択し、**マスクタイプ**ボックスに`PV panel`と入力してください。これは、PVのサブシステムをPLECSプローブブロックにドラッグしたときにプローブリストに表示される名前です。

**3** **マスクエディタ**ウィンドウウィンドウの**プローブ**タブで、カスタムプローブ信号リストを作成します。追加+ボタンをクリックして、新しいプローブ信号を定義し、これに`Voltage`という名前を付けます。次に、サブシステムから電圧計ブロックを**プローブ・コンポーネント**ボックスにドラッグし、**Component signals**リストで**測定電圧**をチェックします。この手順を繰り返して、電流計と電力を追加します。電力信号を作成するときは、電力を計算した関数ブロックを**プローブ・コンポーネント**ボックスにドラッグし、出力ボックスをチェックする必要があります。この段階では、**マスクエディタ**ウィンドウは図6のようになっているはずです。

*図6: マスクエディタでプローブ信号を作成*




作成したプローブ信号をテストするには、PLESCプローブブロックを回路図の最上位に配置し、PVサブシステムをドラッグします。監視可能な3つの信号のリスト(Voltage、CurrentおよびPower)が表示されます。PLESCプローブブロックを開いてVoltageとCurrentの信号をチェックします。デマルチプレクサブロックを使用して、プローブされた信号をXYプロットブロックに接続します。シミュレーションを再実行すると、結果は以前と同じになるはずです。

この段階では、モデルは参照モデル`custom_components_2.plecs`と同じになっているはずです。

# 5 サブシステムのマスクにパラメータを追加

サブシステムのマスクにダイアログパラメータを追加すると、サブシステムアイコンをダブルクリックしたときに表示されるブロックパラメータダイアログが作成されます。マスクパラメータを使用すると、サブシステムを開かなくても内部パラメータを簡単に調整できます。マスクに初期化コマンドを含めることで、サブシステムのパラメータを自動的に読み込むこともできます。

マスクにパラメータを追加すると、サブシステム内のコンポーネントでグローバル変数の可視性が失われます。マスクされたサブシステムのコンポーネントは、マスクで定義した変数のみが表示されます。これにより、サブシステムは外部変数に依存しないため、サブシステムのモジュール性を維持できます。

**あなたのタスク:**

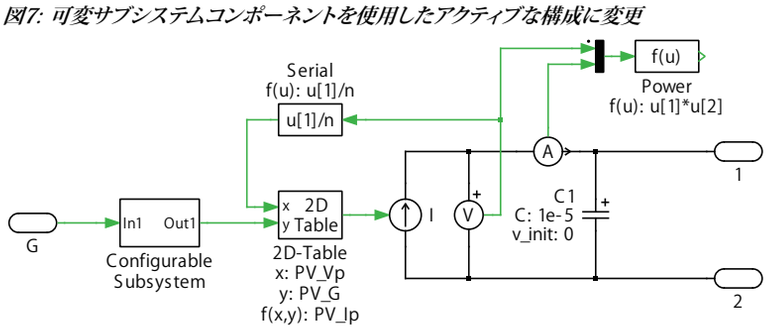
1 **ブロックパラメータ**ダイアログを作成するには、**マスク エディタ**ウィンドウを開き、**ダイアログ**タブで新しいパラメータを追加します。**プロンプト**ボックスにNumber of modulesと入力し、**変数**ボックスでこのパラメータに変数名"n"を付けます。ここで、関数ブロック内の式をu[1]/10からu[1]/nに変更してPVストリングモデルを変更します。

2 2番目のパラメータSolar irradianceを**プロンプト**に追加し、**変数**名を"sun"にします。**可変**ボックスをオンにすると、シミュレーション中にパラメータダイアログに新しい値を入力して変数"sun"をインタラクティブに変更できます。

3 パラメータマスクを完成させるには、**プロンプト**に"Irradiance"を追加し、"sun_combo"という**変数**名を付けます。このパラメータはコンボボックス型です。**コンボボックス値**を、internalとexternalとして、間に文字を入れずに別々の行に定義します。コンボ ボックスパラメータの戻り値は、選択したオプションの1から始まるインデックスを含む文字列です。可変サブシステムコンポーネント内の有効な構成を変更するには、**変数**"sun_combo"を使用します。

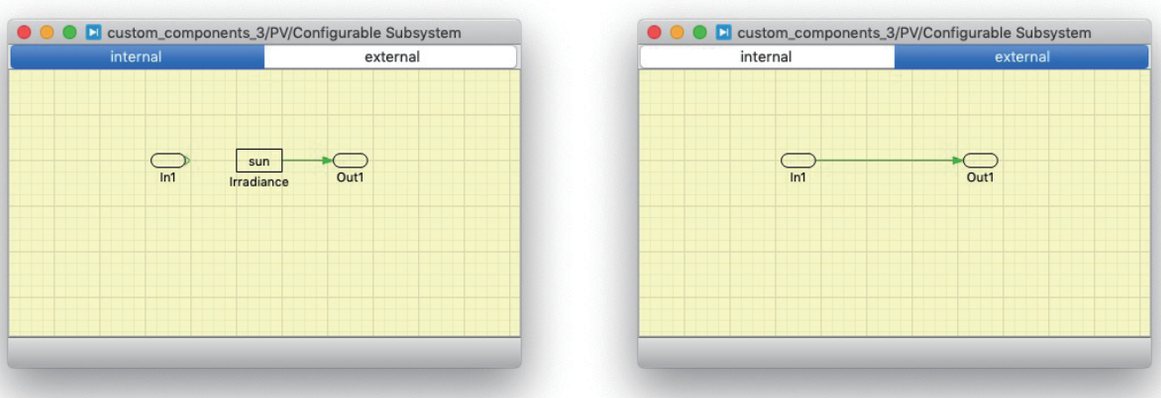
4 サブシステムコンポーネントを回路モデルに追加し、サブシステムを右クリックして**サブシステム -> 可変サブシステムに変換**を選択して、可変サブシステムに変換します。次に図7に示すように接続します。この変更により、太陽放射照度の信号ソースをブロックパラメータで指定した値(internal)か、外部信号(external)を使用するかを選択できるようになります。**入力信号ポート**ブロックに"G"という名前を付けます。この入力ポートは、外部からの放射照度の信号をマスクされたサブシステムに取り込むために使用します。

*図7: 可変サブシステムコンポーネントを使用したアクティブな構成に変更*




5 サブシステムを右クリックして**サブシステム -> サブシステムのモデル表示**を選択するか、**Ctrl + U**を押すと、可変サブシステムブロックの回路図を開くことができます。各構成の回路図には、上部にあるタブからアクセスできます。構成タブをダブルクリックすると、対応する構成の名前(**回路名**)をinternalなどに変更できます。右クリックでタブバーのコンテキストメニューから2つ目の構成を追加します。2番目の**回路名**にexternalという名前を付けます。各サブシステムの構成の回路図を図8に示します。

*図8: 各サブシステム構成の回路図*



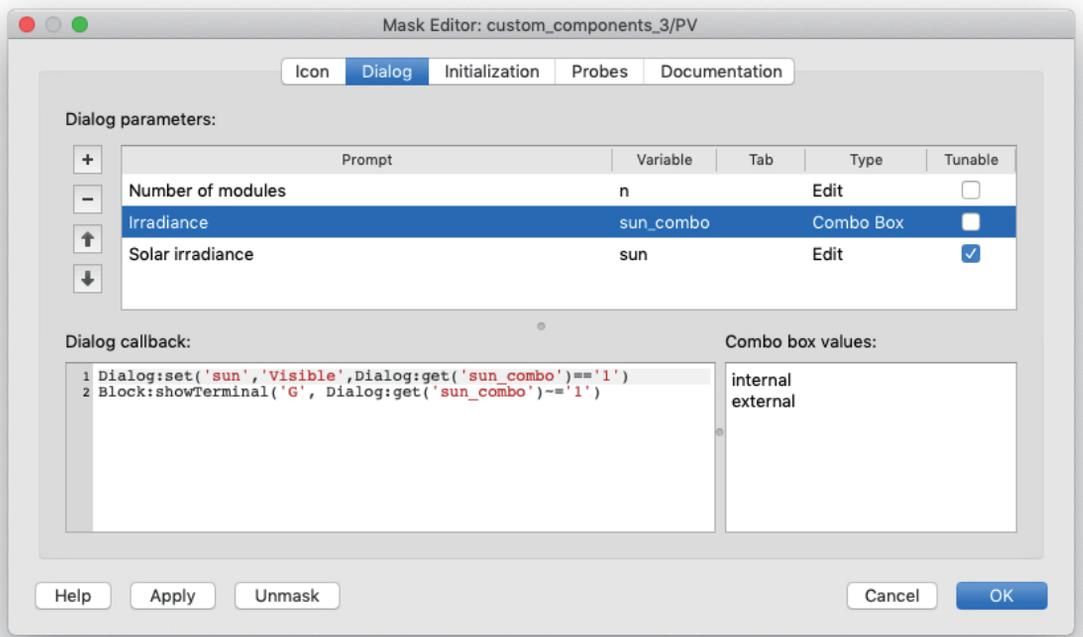
6 可変サブシステムコンポーネントブロックをダブルクリックすると、**ブロックパラメータ**ウィンドウを表示します。**回路名**コンボボックスフィールドの**<参照>**オプションにsun_comboと入力します。これは、**サブシステム**構成がドロップダウンリスト内のハードコーディングされた構成ではなく、参照によって渡されることを意味します。サブシステムのアクティブな構成は、**マスクエディタ**で定義した"sun_combo"の関数として変更できるようになりました。

7 PV panelの**マスクエディタ**ウィンドウの**ダイアログコールバック**セクションに次の2行のコードを追加します。

```
Dialog:set('sun','Visible',Dialog:get('sun_combo')=='1')
Block:showTerminal('G', Dialog:get('sun_combo')~='1')
```

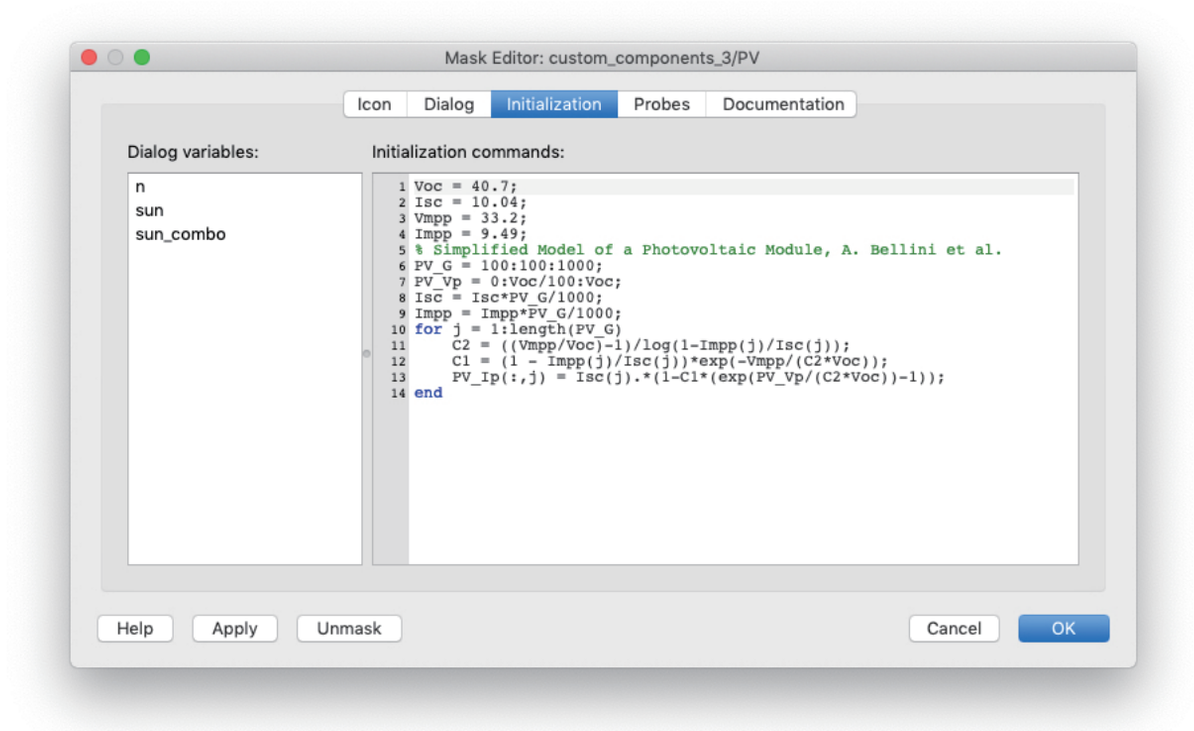
これらのLuaステートメントを使用すると、**Irradiance**パラメータが外部(external)に設定されている場合はSolar irradianceパラメータフィールドを非表示にし、**Irradiance**パラメータが内部(internal)に設定されている場合は信号入力端子"G"を非表示にします。マスクエディタのダイアログパラメータは、図9のようになります。

*図9: マスクエディタウィンドウでのサブシステムダイアログパラメータの作成*




8 サブシステムはマスク変数のみを認識するため、現在の特性の計算はサブシステムのマスク内で実行する必要があります。**シミュレーションパラメータ**の**初期化コマンド**ウィンドウからサブシステムの**初期化コマンド**ウィンドウにコードを移動します。サブシステムの**初期化コマンド**ウィンドウは、図10のようになります。

*図10: マスクしたサブシステムの初期化コマンド*




**9** **サブシステム**のマスクにパラメータを適用すると、**サブシステム**をダブルクリックしても基になる回路図は表示されなくなり、**ブロックパラメータ**ウィンドウが表示されます。"Number of modules"に`10`、"Irradiance"を`internal`、"Solar irradiance"に`1000`を入力します。"Irradiance"パラメータを`external`に変更すると、回路図の最上位レベルに追加の端子が表示されます。この端子には、PLECSライブラリの任意の信号ソース(**ステップ信号**ブロックなど)を接続できます。

この段階では、モデルは参照モデル`custom_components_3.plecs`と同じになっているはずです。

# 6 まとめ

この演習では**サブシステム**の概念を用いて、固有のアイコンとマスクのパラメータを持つカスタムコンポーネントを作成しました。カスタム コンポーネントはトップダウンの設計手法をサポートしており、再利用や構成が容易です。**サブシステム**内で信号をマスクすると、信号の測定も容易になります。

このPVセルコンポーネントのモデルは、温度の変化を考慮して拡張することができ、並列ストリングに接続してインバータシステムの電源として使用することができます。

# 7 Luaの入門

Luaはシンプルでありながら強力なオープンソースのスクリプト言語です。このチュートリアルでは、動的サブシステムマスクを作成するために必要な基本概念について説明します。詳しい情報についてはLuaのウェブサイト[2]を参照してください。

デフォルトでは、Luaはすべての変数を*グローバル*として宣言します。ただし、PLECSは、グローバル変数の作成や変更を禁止する、保護された環境でLuaコードを実行します。したがって、`local`キーワードを使用して変数と関数を明示的に*ローカル*として宣言する必要があります:

```
local x = "a string"
```

## 7.1 マスクアイコンの描画コマンド

Lua言語で使用できるいくつかの描画コマンドを以下に説明します。複数のコマンドを入力すると、コマンドの表示順にグラフィックオブジェクトが描画されます。コマンドの評価中にエラーが発生した場合、PLECSはマスクアイコンに3つの疑問符(`???`)を表示します。

### *Text*

コマンド:

```
Icon:text(x, y, 'text')
```

*text*をアイコンの中央または座標*x*、*y*の中心に表示します。テキストはアイコンと一緒に回転せず、常に左から右に表示します。

### *Line*

コマンド:

```
Icon:line(xvec, yvec)
```

ベクトル*xvec*と*yvec*で指定された線を描画します。両方のベクトルは同じ長さでなければなりません。ベクトルは、`{1, 2, 3}`など中括弧を使用して入力されることに注意してください。

### *Image*

コマンド:

```
Icon:image(xvec, yvec, 'filename')
```

ファイル*filename*から画像を読み取り、マスクアイコンに表示します。*filename*パラメータ名は、絶対パスのファイル名(例: `C:/images/myimage.png`)またはモデルのディレクトリに追加される相対ファイルパス(例: `images/myimage.png`)である必要があります。サポートされている画像形式は、BMP、GIF、JPG、PNGです。

### *パラメータ値のクエリ*

コマンド:

```
Dialog:get('variable')
```

変数*variable*に関連付けられたマスクのパラメータ文字列値を返します。パラメータ値は評価されないことに注意してください。このコマンドを使用すると、マスクのパラメータに応じてマスクアイコンを変更できます。

*関数*

関数は次のように定義されます:

```
local function drawTriangle(x, y)
  Icon:line(Vector{0,8.66,-8.66,0}+x, Vector{-10,5, 5,-10}+y)
end
```

関数定義は、キーワードfunction、*name*(drawTriangle)、*parameters*(x, y)のリスト、ステートメントのリストといった*body*、および終了文字endで構成されます。パラメータは、関数呼び出しで渡される引数の値で初期化されるローカル変数です。上記の例では、中心点*x, y*で三角形を描く関数を定義しています。関数は次のように呼び出すことができます:
ことができます:

```
drawTriangle(10, 0)
```

## 7.2 ダイアログコールバック

*パラメータ値の設定*

コマンド:

```
Dialog:set('variable', 'property', value, ...)
```

変数*variable*に関連付けられたマスクパラメータの1つ以上のプロパティを変更します。EnableまたはVisibleプロパティを指定できます。Enableはパラメータの有効状態を指定します。無効なパラメータはダイアログ内でグレー表示され、変更できません。Visibleは、ダイアログ内でのパラメータの表示/非表示を指定します。

*ターミナルの非表示、表示*

コマンド:

```
Block:showTerminal('name', flag)
```

ブール値の*flag*に応じて、*name*という名前のターミナルを表示または非表示にします。非表示ターミナルのコンパニオンポートは、ターミナルが表示されているが接続されていない場合と同じように機能します。

*ターミナルの移動*

コマンド:

```
Block:moveTerminal('name', x, y)
```

*name*という名前のターミナルを、*unrotated*(回転していない)、*unflipped*(反転していない)ブロックに対する相対座標*x, y*に移動します。ターミナルの回転は変更されないことに注意してください。

# 8 参考文献

[1] A. Bellini, V. Iacovone and C. Cornaro, *Simplified model of a photovoltaic module*, Applied Electronics, Pilsen, pp. 47-51, 2009.

[2] Lua - the programming language, [Online]. Available: http://www.lua.org. [Accessed: Apr. 06, 2020].

改訂履歴:

Tutorial Version 1.0　　初版


**Plexim**

**Pleximへの連絡方法:**

☎ +41 44 533 51 00　Phone

+41 44 533 51 01　Fax

✉ Plexim GmbH　Mail

Technoparkstrasse 1

8005 Zurich

Switzerland

@ info@plexim.com　Email

http://www.plexim.com　Web

**KESCO** KEISOKU ENGINEERING SYSTEM

計測エンジニアリングシステム株式会社

https://kesco.co.jp


*PLECS Tutorial*