Guardian LVSによる実現可能なネットリスト比較検証

1. はじめに

Guardian LVS (Layout versus Schematic) ネットリスト比較ツールによって2つのSPICEネットリストを比較検証できます。通常、比較検証対象のネットリストのうち1つは、回路図エディタで作成された回路図で、回路の論理を表します。 もう1つは、回路の実際の物理レイアウトを表したものです。また、しばしば、同じ回路の2つのネットリストに対して、回路図ネットリスト比較（SVS： Schematic Versus Schematic）を実行し、電気的接続情報の差分をチェックする必要がある場合があります。回路を表すネットリストは、通常のSPICE素子（トランジスタ、ダイオード、抵抗、容量、インダクタなど）や特定のSPICE素子（独立電源、電圧/電流制御電源、伝送線路、スイッチ、結合相互インダクタ、マルチターミナル・ネットワーク、サブサーキット・デバイスなど）として回路を記述します。既知の比較検証ツールは通常のSPICE素子のみを処理しますが、Guardian LVSは通常および特定のSPICE素子の両方を処理できます。本稿は、Guardian LVSが処理する特定SPICE素子の種類とこれらの素子を使用する利点について説明します。

2. Guardian LVSでサポートされている特定SPICE素子の概要

Guardian LVSは、特定SPICE素子（独立電源、電圧/電流制御電源、伝送線路、スイッチ、結合相互インダクタ、マルチターミナル・ネットワーク、サブサーキット・デバイスなど）を処理できます。 [プロジェクト設定]ダイアログの[モデル]設定ページには、素子の名前リストや素子に対して指定可能なオプションが含まれています（図1）。特定SPICE素子のステートメントについては、『Guardian Layout Verificationユーザーズ・マニュアル』を参照してください。これらの素子の一部を次に示します。

- **結合相互インダクタ**

  シンタックス

  Kxxx Laaa Lbbb <Lccc…> kvalue <mname>

  Kxxx: 相互インダクタの名前。先頭文字はKでなければなりません。

  Laaa, Lbbb, Lccc, ...: 2つ以上の結合インダクタの名前。

  kvalue: 相互結合係数。単位なし。kvalueの許容範囲値は、-1.0～+1.0です。この範囲以外の値もGuardian LVSでは受け入れられますが、メッセージ・ファイルに警告としてレポートされます。デフォルト値は0です。

  mname: 磁気コア・モデルの名前。シンプルな結合インダクタ・モデルの代わりに、磁気コア・モデルを使用する場合に指定します。モデル・ステートメントのシンタックスは次のとおりです。

  .MODEL mname CORE

  例

  次の例では、相互インダクタK1がインダクタL1とL2間の結合を定義します。結合係数は0.3です。

```
L1 1 0 0.5m
L2 2 0 0.6m
K1 L1 L2 0.3
```

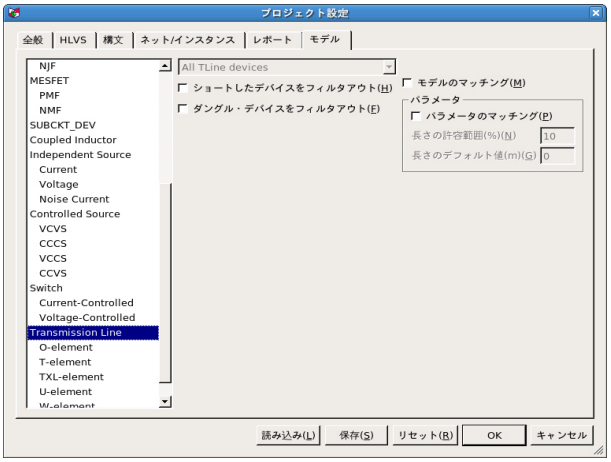

図 1. 特定SPICE 素子

- **独立電流源素子**

  シンタックス

  Ixxx n+ n- <<DC> dcval> <M=val>

  Ixxx: 独立電流源の名前。先頭文字はIでなければなりません。

  n+, n-: 正端子ノードおよび負端子ノードの名前。

  DC: DC電源値のキーワード。

  dcval: DC電流源の値（単位: A）。デフォルト値は0です。

  M: 並列電流源数を表す乗数。デフォルト値は1です。

  例

  次の例は、DC電流値1mAのノード1とノード0間の電流源を示しています。

  I1 1 0 DC 1mA

- **独立電圧源素子**

  シンタックス

  Vxxx n+ n- <<DC> dcval>
  Vxxx: 独立電圧源の名前。先頭文字はVでなければなりません。
  n+, n-: 正端子ノードおよび負端子ノードの名前。
  DC: DC電源値のキーワード。
  dcval: DC電圧源の値。デフォルト値は0です。

  例

  次の例は、DC値5Vのノード1とノード0間に接続した電圧源を示しています。

  V1 1 0 DC 5

Guardian LVSは、電圧/電流制御素子の4つのタイプ（E、F、G、H）をサポートしています。これらの制御素子を使用して、MOS、バイポーラ・トランジスタ、トンネルダイオード、アナログ機能(オペアンプ、コンパレータ、加算器、電圧制御オシレータ、変調器、スイッチ付キャパシタ回路など)をモデリングします。制御素子は、ノード電圧/電流を制御する線形または非線形伝達関数のいずれかで、使用される多項式または折れ線近似線形関数に依存します。Guardian LVSは、電圧/電流制御素子のパラメータをサポートしていませんが、さまざまな伝達関数を認識します。

- **電圧制御電圧源**

  シンタックス

  線形

  Exxx n+ n- <VCVS> nc1+ nc1-

  多項式

  Exxx n+ n- <VCVS> POLY(ndim) nc1+ nc1- nc2+ nc2- ... ncdim+ ncdim-

  折れ線近似線形

  Exxx n+ n- <VCVS> PWL(1) | PPWL(1) | NPWL(1) nc1+ nc1-

  多入力ゲート

  Exxx n+ n- <VCVS> gatetype(k) nc1+ nc1- nc2+ nc2- ... nck+ nck-

  遅延素子

  Exxx n+ n- <VCVS> DELAY nc1+ nc1-

  ビヘイビア

  Exxx n+ n- VOL | VALUE='expression'

  テーブル

  Exxx n+ n- TABLE

  理想オペアンプ

  Exxx n+ n- OPAMP nc1+ nc1-

  変圧器

  Exxx n+ n- TRANSFORMER nc1+ nc1-

  電圧ゲート H(s) (LAPLACE)

  Exxx n+ n- LAPLACE nc1+ nc1-

  電圧ゲート H(s) (POLE)

  Exxx n+ n- POLE nc1+ nc1-

  電圧ゲート H(s) (FREQ)

  Exxx n+ n- FREQ nc1+ nc1-

  電圧ゲイン H(z) (ZTRANS)

  Exxx n+ n- ZTRANS nc1+ nc1-

  Exxx: 電圧制御電圧源の名前。先頭文字はEでなければなりません。

  n+, n-: 正端子ノードおよび負端子ノードの名前。

  VCVS: 電圧制御電圧源のキーワード。ノード名として使用できません。

  nc1+, nc1-...: ペアの正および負の制御ノードの名前。1つのペアは多項式での各次元、もう1つのペアは多入力ゲートでの各ゲートを表します。

  POLY: 多項式関数のパラメータ。ノード名として使用できません。

  ndim: 多項式の次元数。

  PWL (または **NWL**): 折れ線近似線形関数のパラメータ。ノード名として使用できません。

gatetype(k): AND、NAND、OR、NORパラメータは多入力ゲート・デバイスのタイプを設定します。kの値は入力数です。 AND、NAND、OR、NORは、ノード名として使用できません。

DELAY: 遅延素子のパラメータ。ノード名として使用できません。

VOL | VALUE: 出力電圧が式で定義されたことを表すキーワード。

TABLE: 出力電圧がルックアップ・テーブルを用いて検出されることを表すパラメータ。ノード名として使用できません。

OPAMP: 理想オペアンプ素子のパラメータ。ノード名として使用できません。

TRANSFORMER: 理想変圧器のパラメータ。ノード名として使用できません。

LAPLACE: 伝達関数が有理関数形式のラプラス変換関数で記述されることを表すキーワード。ノード名として使用できません。

POLE: 伝達関数がポールとゼロの位置で記述されることを表すキーワード。ノード名として使用できません。

FREQ: 伝達関数が周波数応答テーブルで記述されることを表すキーワード。ノード名として使用できません。

ZTRANS: 伝達関数が有理関数形式のZ変換関数で記述されることを表すキーワード。ノード名として使用できません。

例

1. デバイスがEON、制御ノードが8と６、ノード3と9間に接続された線形VCVS。

```
EON 3 9 8 6
```

2. 制御ノードが1と2、端子n1 n2の線形VCVS。

```
EOUT n1 n2 POLY(1) 1 2
```

3.電圧が2つの電圧v(3,0)とv(4,1)の多項式関数である非線形VCVS。

```
Ea va gnd POLY(2) 3 0 4 1
```

4.電圧がパラメータVOLの後の式から求められる非線形VCVS。

```
Eb 13 0 VOL='10.*log(i(vin)/10m)'
```

5. ラプラス変換関数で記述された伝達関数のVCVS。

```
E1 out grnd LAPLACE in grnd
```

Guardian LVSは、電流制御スイッチおよび電圧制御スイッチをサポートしています。スイッチ素子のパラメータ比較はサポートしていません。

- **電流制御スイッチ**

シンタックス

Wxxx n+ n- vcontrolname mname

Wxxx: 電流制御スイッチの名前。先頭文字はWでなければなりません。

n+, n-: 正端子ノードおよび負端子ノードの名前。

vcontrolname: 制御電流の電圧源の名前。

mname: モデルの名前。モデルの名前は、電流制御スイッチ・モデルを参照する必要があります。モデル・ステートメントのシンタックスは、次のとおりです。

.MODEL mname CSW

例

```
.MODEL WCMOD CSW

W1 n1 n2 VIN WCMOD
```

上記の例では、スイッチW1がノードn1とn2間に配置されています。スイッチの状態は電圧源VINによって制御されます。スイッチはWCMODモデルによって定義されます。

Guardian LVSは、損失(O素子)、無損失(T素子)、 損失(TXL素子)、結合損失(U素子)、結合損失(W素子)の5つの伝送線路をサポートしています。

- **損失 (O素子) 伝送線路**

シンタックス

Oxxx n1 n2 n3 n4 mname <L | LEN | LENGTH=val> <M=val>

Oxxx: 一様に損失のある伝送線路素子の名前。先頭文字はOでなければなりません。

n1, n2: ポート1におけるノード。

n3, n4: ポート2におけるノード。

mname: モデル参照名。

L (または LEN, LENGTH)**:** 伝送線路の物理長 (単位:m)。デフォルトは L=1mです。

M: 並列デバイスを表す乗数。デフォルトはM=1です。

損失伝送線路の各モデルは、モデル・ステートメントで記述されなければなりません。モデル・ステートメントのシンタックスは、次のとおりです。

.MODEL mname LTRA

例

```
.MODEL tlline LTRA
.MODEL connect LTRA
O1 1 0 2 0 tlline l=.6095
ONET 1 2 3 2 connect length=.16
```

3. サブサーキット・デバイスおよびLISAを用いたパラメータ計算

サブサーキット・デバイスはサブサーキットの相似体であり、SPICE素子の代わりに使用できます。サブサーキット・デバイスのインスタンスはサブサーキットのインスタンスによってネットリストに記述され、シンプルなデバイスおよびサブサーキット・インスタンスとして解釈されます。 *.SUBCKT_DEVコマンドを使用して、サブサーキット名に対するサブサーキット・デバイス・タイプを指定します。

***.SUBCKT_DEV sname stype nPins [(Pin_i,..,Pin_j) ⋯]**

- *sname* : サブサーキットの名前。この名前のサブサーキットは、ネットリストに定義できる場合もできない場合もあります。
- *nPins* : サブサーキット・デバイスのピン数。
- *(Pin_i,…,Pin_j)…* : 入れ替え可能なピン・インデックスのグループを指定するオプション・パラメータ。*.SUBCKT_DEVステートメントには、入れ替え可能なピンのグループを複数指定できます。
- stype : サブサーキットのデバイス・タイプ。次のいずれかを指定できます。
  - **M:** MOSトランジスタ
  - **Q:** バイポーラ・トランジスタ
  - **J:** JFET トランジスタ
  - **B または Z:** MESFET トランジスタ
  - **Y:** ユーザ定義素子
  - **X:** サブサーキット
  - **R:** 抵抗
  - **C:** 容量
  - **L:** インダクタ
  - **D:** ダイオード
  - **NMOS:** N型 MOSFET
  - **PMOS:** P型 MOSFET
  - **NPN:** NPN BJT
  - **PNP:** PNP BJT
  - **NJF:** N型 JFET
  - **PJF:** P型 JFET
  - **NMF:** N型 MESFET
  - **PMF**: P型 MESFET
  - **CORE:** 結合インダクタ
  - **I:** 独立電流源
  - **V:** 独立電圧源
  - **N:** 独立ノイズ電流源
  - **VCVS: 電圧制御電圧源**
  - **CCCS:** 電流制御電流源
  - **VCCS: 電圧制御電流源**
  - **CCVS:** 電流制御電圧源
  - **CSW:** 電流制御スイッチ
  - **SW:** 電圧制御スイッチ
  - **LTRA:** 損失 (O素子) 伝送線路
  - **TXL:** 損失 (TXL素子) 伝送線路
  - **T:** 無損失 (T素子) 伝送線路
  - **U:** 結合損失 (U素子) 伝送線路
  - **W:** 結合損失 (W素子) 伝送線路
  - **SP:** マルチターミナル・ネットワーク
  - **VLG:** ユーザ定義Verilog-A素子

  指定するサブサーキット･デバイス･タイプに従って、サブサーキット・デバイス･インスタンスは、サブサーキット定義の中をチェックしないサブサーキット・インスタンス、または指定したタイプのシンプルなデバイスとして解釈されます。サブサーキット・デバイス・タイプがXの場合、サブサーキットsnameのすべてのインスタンスは、サブサーキット・インスタンスとして解釈されます。サブサーキット・デバイス・タイプが*M*、*Q*、*J*、*B*、または*Z*の場合、サブサーキットsnameのインスタンスは、サブサーキット・デバイス・インスタンスまたはシンプルなデバイスとして処理されます。その他のすべてのタイプでは、サブサーキットsnameのインスタンスは、対応するタイプのシンプルなデバイスになります。**.SUBCKT_DEV*ステートメントで指定したサブサーキット・デバイス・タイプ*M*、*Q*、*J*、*B*、または*Z*のサブサーキット・インスタンスをシンプルなデバイスとして認識するには、次の条件を満たす必要があります。
- コメントとして記述されるモデル名は、サブサーキット・インスタンス・ステートメントのパラメータ・セクションに指定する。
- デバイス・タイプは、*.MODEL*ステートメントでモデル名に対して指定する。
- *.SUBCKT_DEVステートメントは、デバイス･タイプに対応するサブサーキット･デバイス･タイプ*M*、*Q*、*J*、*B*、または*Z*のいずれかでこのサブサーキットに対して定義する(たとえば、Mは*PMOS*または*NMOS*に対応)。

例

1. サブサーキットPCHのインスタンスX0とX1は、モデル名PCH1とPCH2である PMOS トランジスタとして認識されます。

```
.MODEL PCH1 PMOS
.MODEL PCH2 PMOS
*.SUBCKT_DEV PCH M 4 (1,3)
X0 1 2 3 4 PCH L=1u W=1u $[PCH1]
X1 3 2 6 4 PCH L=1u W=1u $[PCH2]
```

2. インスタンスX0とX1は、サブサーキット・デバイスとして解釈されます。

```
*.SUBCKT_DEV PCH1 M 4 (1,3)
*.SUBCKT_DEV PCH2 X 3
X0 1 2 3 4 PCH1 L=1u W=1u
X1 a b c PCH2 L=1u W=1u
```

3. インスタンスX0は、モデル名PCHである PMOSトランジスタで、インスタンス X1は、モデル名RMPである抵抗素子です。

```
*.SUBCKT_DEV PCH PMOS 4 (1,3)
*.SUBCKT_DEV RMP R 2
X0 1 2 3 4 PCH L=1u W=1u
X1 a b RMP R=1000
```

Guardian LVSは、サブサーキット・デバイスのパラメータを比較します。デフォルトでは、[プロジェクト設定]ダイアログの[モデル]設定ページでサブサーキット・デバイスに対して[パラメータのマッチング]オプションがオンになっている場合、すべてのパラメータが比較されます。通常、サブサーキット・デバイスには数多くのパラメータが含まれていますが、その一部を比較したい場合があります。この場合、スクリプト・プロシージャを使用して比較するサブサーキット・デバイスのパラメータを決定し、シンプルなサブサーキット・デバイスの並列マージの結果で得た複雑な素子に対してパラメータ計算アルゴリズムを指定します。プロシージャは、LISAスクリプト言語を用いて実装されます。LISAについては、『Language for Interfacing Simucad Applications (LISA) User's Manual』を参照してください。

すべてのスクリプト・プロシージャは、[プロジェクト設定]ダイアログの[全般]設定ページの[LISAスクリプト・ファイル]フィールドで指定される1つのスクリプト・ファイルに含まれます。ユーザは、任意のサブサーキット・デバイス・タイプに対して別のLISAプロシージャを指定できます(図2)。同時に、[モデル]設定ページのリストボックスで指定されていないサブサーキット・デバイスに対して共通のLISAプロシージャを使用できます。少なくとも1つのLISAプロシージャを使用する場合、プロシージャを含むLISAスクリプト・ファイルは[全般]設定ページで指定します。SPICEネットリストのサブサーキット・デバイスのパラメータを処理するには、Guardian LVSインタフェース・プロシージャをLISAスクリプト・プロシージャの実装時に使用します。

GetParameterValues

プロシージャは、並列マージされたサブサーキット・デバイスのパラメータ値のシーケンスを返します。

シンタックス

param_values = GetParameterValues(param_name);

パラメータ

param_name (type STRING) : パラメータの名前。

戻り値

param_values (type SEQUENCE) : パラメータ値のシーケンス。

SetParameterValue

プロシージャは、指定したパラメータの計算値を設定します。

シンタックス

SetParameterValue(param_name, param_value);

パラメータ

param_name (type STRING) : パラメータの名前。

param_value (type FLOAT) : 計算されたパラメータの値。

サブサーキット・デバイスのパラメータ比較のためにLISAスクリプト・プロシージャを使用した例を次に示します。比較するネットリストには、PCHおよびNCHという名前のサブサーキット・デバイスが含まれます。LISAスクリプト・ファイルには、PCHという名前のサブサーキット・デバイスに対してLW_CALC_PCH、NCHという名前のサブサーキット・デバイスに対してLW_CALC_NCHの2つのプロシージャが含まれます。各ネットリスト要素には、L、W、A、Bの4つのパラメータがあります。ただし、一部のパラメータのみ比較されます。つまり、PCH要素では、L、W、Aが比較され、NCH要素では、L、W、Bが比較されます。サブサーキット・デバイス・パラメータの計算および比較結果がParameter Matchレポートに表示されます。

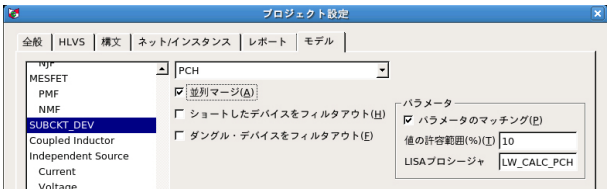

図 2. サブサーキット・デバイス・パラメータに対するLISAプロシージャの指定

例

1つ目のネットリスト:

```
*.SUBCKT_DEV PCH M 4 (1,3)
*.SUBCKT_DEV NCH M 4 (1,3)

X0 n1 IN n2 VDD  PCH L=3u  W=2u A=3
B=20
X1 n1 IN n2 VDD  PCH L=3u  W=2u A=6
B=50
X2 n1 IN n2 VDD  PCH L=3u  W=2u A=3
B=80
X3 net1 IN2 net2 GND AAA
X4 net1 IN2 net2 GND AAA

.SUBCKT AAA d g s b
X0 d g s b  NCH  L=4u W=10.5u A=3
B=30
.ENDS
.END
```

2つ目のネットリスト:

```
X0 n1 IN n2 VDD  PCH L=3u W=6u A=6
B=30
X1 net1 IN2 net2 GND  NCH L=4u  W=6u
A=3 B=60
X2 net1 IN2 net2 GND  NCH  L=4u  W=7u
A=3 B=90
X3 net1 IN2 net2 GND  NCH  L=4u  W=8u
A=3 B=60
.END
```

Parameter Matchレポート:

```
          PARAMETER MATCHES
L1: X:(@)top:X3:X0 .........= ($)  top:X3
         B=0.866025       B=0.866025
         L=4E-006         L=4E-006
         W=2.1E-005       W=2.1E-005

L0: X:($)top:X2 ............= top:X0
         A=6              A=6
         L=3E-006         L=3E-006
         W=6E-006         W=6E-006
(@) - indicates device instances formed
by device reduction, see merge file
($) - indicates device instances formed
by device merging, see merge file
```

LISAスクリプト・ファイル:

```
DEFINE PROCEDURE /REPLACE LW_CALC_PCH
  DO
    BEGIN
      L = GetParameterValues("L");
      W = GetParameterValues("W");
      A = GetParameterValues("A");
      N = L.SIZE;
      I = 0;
      P = 0.0;
      Q = 0.0;
      MAXA = 0.0;
      LOOP
        BEGIN
          I = I + 1;
          P = P + L[I] * W[I];
          Q = Q + W[I] / L[I];
          IF (A[I] GTR MAXA) THEN
(MAXA = A[I]);
          IF (I EQL N) THEN (LEAVE
LOOP);
        END;

      LRES = SQRT(P / Q);
      WRES = SQRT(P * Q);
      SetParameterValue("L", LRES);
      SetParameterValue("W", WRES);
      SetParameterValue("A", MAXA);
    END;

DEFINE PROCEDURE /REPLACE LW_CALC_NCH
  DO
    BEGIN
      L = GetParameterValues("L");
      W = GetParameterValues("W");
      B = GetParameterValues("B");
      N = L.SIZE;
      I = 0;
      P = 0.0;
      Q = 0.0;
      BVAL = 0.0;
      LOOP
        BEGIN
          I = I + 1;
          P = P + L[I] * W[I];
          Q = Q + W[I] / L[I];
          BVAL = BVAL + SIN(B[I]) *
COS(B[I]);
          IF (I EQL N) THEN (LEAVE
LOOP);
        END;

      LRES = SQRT(P / Q);
      WRES = SQRT(P * Q);
      SetParameterValue("B", BVAL);
      SetParameterValue("L", LRES);
      SetParameterValue("W", WRES)
    END;
```

4. まとめ

SVS検証で特定SPICE素子を使用することで、より効率的に回路設計を行うことができます。サブサーキット・デバイス素子は、通常のデバイス（トランジスタ、ダイオード、抵抗など）と比較できるジェネリック・デバイスを表す場合に有用です。LISAスクリプト・プロシージャにより、柔軟なサブサーキット・デバイスのパラメータ計算と比較が実行できます。