Spice3f5 マニュアル

日本語訳 by Ayumi's Lab.

2002 年 2 月 25 日

目 次

1 はじめに

Spice は，非線形直流解析，非線形過渡解析，線形交流解析用の汎用回路シミュレーションプログラムです．抵抗，コンデンサ，インダクタ，相互インダクタ，独立電圧源，独立電流源，4 種類の従属電源，無損失・有損失伝送線路 (2 つの異なった実装があります)，スイッチ，一様分布 RC 線，5 つのもっとも一般的な半導体素子 (ダイオード，バイポーラ接合型トランジスタ (BJT)，JFET, MESFET, MOSFET) によって構成された回路を解析します．

Spice3 バージョンは，Spice 2G.6 を元にしています．Spice3 は，新しい特徴を取り込むように開発されましたが，Spice2 プログラムで広く使われている機能やモデルもサポートしています．

Spice は，半導体素子のモデルを内蔵しており，利用者は必要最小限のモデルのパラメータ値を指定すればよいのです．BJT のモデルは，Gummel and Poon の積分電荷モデルに基づいていますが，Gummel-Poon のパラメータが指定されない場合，モデルはより単純な Ebers-Moll モデルになります．どちらの場合でも，電荷蓄積効果，電気抵抗，出力コンダクタンスの電流依存性を反映できます．ダイオードのモデルは，接合型ダイオードおよびショットキーバリアダイオードのどちらにも使えます．JFET モデルは，Shichman and Hodges の FET モデルに基づいています．6 つの MOSFET モデルが実装されています：MOS1 は I-V 特性の 2 乗則によって記述され，MOS2 [1] は解析的モデル，MOS3 [1] は半経験的モデル，MOS6 [2] は短チャネル区間で精度の高い単純な解析的モデル，MOS4 [3, 4] と MOS5 [5] は，BSIM (Berkeley Short-channel IGFET Model) と BSIM2 です．MOS2, MOS3, MOS4 は，チャネル長調整，subthreshold conduction, scattering-limited velocity saturation, small-size effects, charge-controlled capacitances のような副次的な効果も扱います．

1.1 解析の種類

1.1.1 直流解析

Spice の直流解析部は，インダクタを短絡し，コンデンサを開放して，回路の直流動作点を決定します．直流解析のオプションは，制御行 `.DC`, `.TF`, `.OP` で指定します．直流解析は，過渡解析を行なう場合に過渡現象の初期状態を決定するため，また交流小信号解析を行なう場合に非線形素子の線形化小信号モデルを決定するために，それらの解析に先立って自動的に行なわれます．必要なら，直流解析の一部として，直流小信号に対する伝達関数の値 (入力に対する出力の比)，入力抵抗，出力抵抗も計算できます．直流解析は，直流伝達特性を描くのにも使えます．つまり，指定された独立電源をユーザが指定した範囲に渡って変化させ，電源の一連の値に対する出力変数の値を保存します．

1.1.2 交流小信号解析

Spice の交流小信号解析部は，出力変数の周波数特性を計算します．プログラムは，最初に回路の直流動作点を計算し，回路中のすべての非線形素子について線形化した小信号モデルを求めます．続いて，ユーザが指定した周波数の範囲に渡って，得られた線形回路を解析します．通常，交流小信号解析の結果は，伝達関数 (電圧利得，伝達インピーダンスなど) です．回路に交流入力が 1 つしかない場合，出力変数の値がそのまま入力に対する伝達関数となるように，入力の振幅を 1 とし，位相を 0 に設定すると便利です．

1.1.3 過渡解析

Spice の過渡解析部は，出力変数の過渡現象を，ユーザが指定した時間間隔について，時間の関数として計算します．初期状態は，直流解析によって自動的に決定されます．時間に依存しないすべての電圧・電流源 (たとえば電源) は，それらの DC 値に設定されます．過渡解析の時間間隔は，制御行 .TRAN で設定します．

1.1.4 極・ゼロ解析

Spice の極・ゼロ解析部は，小信号交流伝達関数の極とゼロを計算します．プログラムは，最初に直流動作点を計算し，回路のすべての非線形素子について線形化した小信号モデルを求めます．この線形化した回路を使って，伝達関数の極とゼロを見つけます．

2 種類の伝達関数を解析できます．一つは，(出力電圧)/(入力電圧) という形式で，もう一つは，(出力電圧)/(入力電流) という形式です．これらの 2 種類の伝達関数の形式は，すべてのケースをカバーし，入力/出力インピーダンスや電圧利得などの関数の極/ゼロを見つけることができます．入力と出力の端子は，ノードのペア 2 つで指定します．

極・ゼロ解析は，抵抗，コンデンサ，インダクタ，線形従属電源，独立電源，BJT, MOSFET, JFET, ダイオードに対して行なえます．伝送線路はサポートされていません．

解析には，数値的な局所最適化を用いています．回路が大きい場合，時間が相当かかったり，すべての極・ゼロを見つけることができない可能性もあります．回路によっては，この手法は迷子になり，とてつもなく多くの極・ゼロを見つけてしまうことがあります．

1.1.5 小信号歪み解析

Spice の歪み解析部は，小入力信号に対する定常高調波歪みおよび混変調出力を計算します．回路の入力として単一周波数の信号が指定された場合，回路中のすべての点について，2 次および 3 次の高調波が複素数で計算されます．2 種類の周波数の入力が回路に加えられた場合，入力周波数の和と差，および低い周波数と高い周波数の第 2 高調波との差について，回路変数の複素数値を計算します．

歪み解析は，次の非線形素子に対して行なえます：ダイオード，BJT, JFET, MOSFET (レベル 1, 2, 3, 4/BSIM1, 5/BSIM2, 6), MESFET．歪み解析では，すべての線形素子を使えます．回路中にスイッチがある場合，歪みの計算のための小さな励振によってスイッチの状態が変わらなければ，解析結果は正しいです．

1.1.6 感応度解析

Spice3 は，モデルのパラメータを含むすべての回路変数に関して，出力変数の直流動作点の感応度または交流小信号感応度のいずれかを計算します．Spice は，各素子の各パラメータを独立に変化させることによって生じる出力変数 (ノードの電圧または枝の電流) の差を計算します．この手法は数値的な近似ですから，極端に敏感なパラメータの 2 次の効果を反映したり，0 でない非常に小さな感応度を表わすことができなかったりすることがあります．さらに，各変数は，その値に対する率で変化させられるので，値が 0 のパラメータは解析されません (これはデータの量が非常に増えるのを防ぐ利益があります)．

1.1.7 ノイズ解析

Spice のノイズ解析部は，与えられた回路について，素子によって生成されたノイズの解析を行ないます．入力と出力の端子が指定されると，この解析は，各素子 (および素子内の各ノイズ生成源) の出力端子電圧への貢献度を計算します．また，指定された入力に関して，出力ノイズと等価な入力ノイズも計算します．これらの解析は，指定された範囲内のすべての周波数について行なわれます．計算されたノイズの大きさは，回路変数を定常ガウス確率過程と見なしたスペクトル密度に対応します．

スペクトル密度を計算した後，指定された周波数範囲にわたってこれらの値を積分して (この周波数範囲の) 総ノイズ電圧/電流にします．この計算された値は，回路変数を定常ガウス確率過程と見なした分散に対応します．

1.2 異なった温度での解析

Spice へのすべての入力は，27°C で測定されたと仮定されます．この温度は，制御行 `.OPTION` の `TNOM` パラメータで変更できます．温度の影響をモデル化している任意の素子については，モデル自身の `TNOM` パラメータを指定することによって，温度の値を個別に指定できます．回路のシミュレーションは，制御行 `.OPTION` で `TEMP` パラメータを指定して変更しない限り，27°C で実行されます．個々の素子について，それぞれに `TEMP` パラメータを指定すれば，回路の温度を変更することができます．

温度依存性は，抵抗，ダイオード，JFET, BJT, レベル 1, 2, 3 の MOSFET でサポートされています．BSIM MOSFET (レベル 4, 5) には，モデルのすべてのパラメータを Spice に入力する前に調整するという，別の温度依存性のしくみがあります．BSIM の温度調整の詳細については，[6] および [7] を参照してください．

BJT やダイオードのモデル方程式の指数項には，温度が陽に現われています．さらに，飽和電流にも温度依存性が含まれています．BJT の飽和電流の温度依存性は，次の式によって決定されます．

$$I_S(T_1) = I_S(T_0)\left(\frac{T_1}{T_0}\right)^{XTI} \exp\left(\frac{E_g q(T_1 T_0)}{k(T_1 - T_0)}\right)$$

ここで，k はボルツマン定数，q は電荷，E_g はエネルギーギャップ (モデルのパラメータ)，XTI は飽和電流温度指数 (モデルのパラメータで，通常は 3) です．

順方向および逆方向のベータの温度依存性は，次の式によります．

$$B(T_1) = B(T_0)\left(\frac{T_1}{T_0}\right)^{XTB}$$

ここで，T_1 および T_0 は絶対温度，XTB はユーザが指定するモデルのパラメータです．ベータに対する温度の影響は，B_F, I_{SE}, B_R, I_{SC} の値 (それぞれ Spice のモデルパラメータ `BF`, `ISE`, `BR`, `ISC`) を適切に調整することによって反映されます．

接合型ダイオードモデルの飽和電流の温度依存性は，次の式によって決定されます．

$$I_S(T_1) = I_S(T_0)\left(\frac{T_1}{T_0}\right)^{\frac{XTI}{N}} \exp\left(\frac{E_g q(T_1 T_0)}{Nk(T_1 - T_0)}\right)$$

ここで，N はエミッション定数 (モデルのパラメータ)，その他の記号の意味は上記と同じです．ショットキーバリアダイオードの飽和電流の温度指数 `XTI` は，通常 2 です．

すべての素子モデルについて，温度は接合電位の値 U (Spice では PHI) に陽に現われます．温度依存性は，次の式によって決定されます．

$$U(T) = \frac{kT}{q} \log_e\left(\frac{N_a N_d}{N_i(T)^2}\right)$$

ここで，k はボルツマン定数，q は電荷，N_a はアクセプタ不純物濃度，N_d はドナー不純物濃度，N_i は固有キャリア濃度，E_g はエネルギーギャップです．

MOSFET モデルでは，表面移動性の値 M_0 (すなわち UO) に温度が陽に現われます．温度依存性は，次の式で決定されます．

$$M_0(T) = \frac{M_0(T_0)}{\left(\frac{T}{T_0}\right)^{1.5}}$$

抵抗の温度による影響は，次の式でモデル化されています．

$$R(T) = R(T_0)[1 + TC_1(T - T_0) + TC_2(T - T_0)^2]$$

ここで，T は回路の温度，T_0 は名目温度，TC_1, TC_2 は 1 次および 2 次の温度係数です．

1.3 収束

直流動作点および過渡解析の解は，繰り返し計算によって求めます．以下の 2 つの条件が両方成立した時に，繰り返し計算が終了します．

1. 非線形の枝の電流が許容値 (0.1%または 1 pA (1×10^{-12} A) のいずれか大きな方) 内に収束した．
2. ノードの電圧が許容値 (0.1%または 1 μV (1×10^{-6} V) のいずれか大きな方) 内に収束した．

Spice で使われているアルゴリズムは，とても信頼できるものですが，解に収束するとは限りません．このような場合，プログラムはそのジョブを中止します．

直流解析が収束しない場合，主に回路のつなぎ方，要素の値，モデルのパラメータ値の指定に誤りがあります．循環的な (再生式の) スイッチング回路すなわち正帰還のある回路は，帰還回路内のいくつかの素子に OFF オプションを使うか，または回路が希望の状態に収束するようにするための制御行 .NODESET オプションを使うかしない限り，直流解析ではおそらく収束しないでしょう．

2 回路の記述

2.1 全体の構成と慣例

解析する回路を Spice に指定するには，回路のつなぎ方と要素の値を定義する**要素行**と，モデルのパラメータを指定したり実行の制御を定義する**制御行**とで記述します．入力ファイルの最初の行はタイトルであり，最後の行は "`.END`" でなければなりません．その他の行の順序は任意です (もちろん，継続行は元になる行の直後になければなりません).

回路の各要素は，要素の名前，要素がつながる回路のノード，要素の電気的特性を決定するパラメータの値からなる要素行で指定します．要素名の最初の文字により要素の種類を指定します．Spice の要素の種類の形式は，後ほど説明します．文字列 "*XXXXXXX*", "*YYYYYYY*", "*ZZZZZZZ*" は，任意の英数字列を表わします．例えば，抵抗の名前は，文字 `R` で始まる 1 文字以上の文字列です．したがって，`R`, `R1`, `RSE`, `ROUT`, `R3AC2ZY` は，すべて正しい抵抗の名前です．素子の種類各々についての詳細は，次の節で説明します.

1 行の中のフィールドは，1 個以上の空白，1 個のカンマ，1 個の等号 ('='), 括弧で区切られます．余分な空白は無視されます．次の行の 1 桁目を '+' で始めると，行を継続できます．Spice は，2 桁目以降から読み込みを続けます.

名前のフィールドは，文字 (A から Z) で始め，区切文字を含んではいけません.

数値のフィールドは，整数フィールド (`12`, `-44`)，浮動小数点数フィールド (`3.14159`)，整数または浮動小数点数のあとに整数の指数が続くもの (`1e-14`, `2.65e3`)，整数または浮動小数点数のあとに以下の尺度が続くもののいずれかです.

`T` $= 10^{12}$	`G` $= 10^{9}$	`Meg` $= 10^{6}$	`K` $= 10^{3}$	`mil` $= 25.4^{-6}$
`m` $= 10^{-3}$	`u` $= 10^{-6}$	`n` $= 10^{-9}$	`p` $= 10^{-12}$	`f` $= 10^{-15}$

数値の直後に続く尺度でない文字は無視されます．尺度の直後に続く文字も無視されます．したがって，`10`, `10V`, `10Volts`, `10Hz` は，すべて同じ数値になり，`M`, `MA`, `MSec`, `MMhos` は，すべて同じ尺度になります．`1000`, `1000.0`, `1000Hz`, `1e3`, `1.0e3`, `1KHz`, `1K` は，すべて同じ数値となることに注意してください.

ノードの名前は，任意の文字列です．基準 (グラウンド) ノードは，かならず '`0`' です．Spice3 ではノードの名前は数値としてではなく文字列として扱われるので，'`0`' と '`00`' は異なったノードになりますが，Spice2 では同じノードになります．回路には，電圧源とインダクタのみのループ，または電流源とコンデンサのみのループを作ってはなりません．回路の各ノードは，グラウンドと直流的につながっていなければなりません．非終端伝送線路を許している伝送線路のノードおよび MOSFET のサブストレートノード (内部的に 2 箇所とつながっている) を除いては，各ノードは 2 つ以上の素子とつながっていなければなりません.

2.2 タイトル行，コメント行，.END 行

2.2.1 タイトル行

例:

```
POWER AMPLIFIER CIRCUIT
TEST OF CAM CELL
```

入力ファイルの最初の行はタイトル行です．この行は，出力の各部分のヘッダとして，そのまま表示されます．

2.2.2 .END 行

例:

```
.END
```

入力ファイルの最後の行は，“.End” 行でなければなりません．ピリオドも名前の一部であることに注意してください．

2.2.3 コメント

一般形:

```
*〈任意のコメント〉
```

例:

```
* RF=1K      Gain should be 100
* Check open-loop gain and phase margin
```

最初の桁のアスタリスクは，この行がコメント行であることを表わします．コメント行は，回路の記述の任意の場所に置くことができます．Spice3 は，行頭が空白の場合もコメント行として扱います．

2.3 素子のモデル

一般形:

```
.MODEL MNAME TYPE(PNAME1=PVAL1 PNAME2=PVAL2 ...)
```

例:

```
.MODEL MOD1 NPN (BF=50 IS=1E-13 VBF=50)
```

最も単純な回路の要素は，典型的にはいくつかのパラメータ値を必要とするだけです．しかし，Spice に含まれている素子によっては (特に半導体素子)，数多くのパラメータ値を必要とするものもあります．回路の中では，同じパラメータで定義した素子を複数使うことがよくあります．こういった理由により，独立した .MODE 行で素子モデルの一連のパラメータを定義して，一意のモデル名を割り当てます．そして，Spice の素子要素行では，モデル名を参照します．

これらの複雑な素子の場合，素子要素行は素子名，素子がつながるノード，素子のモデル名からなります．さらに，ある素子については，幾何的な係数，初期条件といったオプションのパラメータを指定することもできます (詳細についてはトランジスタとダイオードの 3.4節を参照してください)．

上記の *MNAME* はモデル名で，*TYPE* は次の 15 種類のうちの 1 つです．

R	半導体抵抗モデル
C	半導体コンデンサモデル
SW	電圧制御スイッチ
CSW	電流制御スイッチ
URC	一様分布 RC モデル
LTRA	有損失伝送線路モデル
D	ダイオードモデル
NPN	NPN BJT モデル
PNP	PNP BJT モデル
NJF	N チャネル JFET モデル
PJF	P チャネル JFET モデル
NMOS	N チャネル MOSFET モデル
PMOS	P チャネル MOSFET モデル
NMF	N チャネル MESFET モデル
PMF	P チャネル MESFET モデル

パラメータの値は，パラメータ名の後ろに等号を続け，パラメータ値を続けて指定します．値が指定されていないモデルのパラメータには，各モデルごとに以下で与えられるデフォルト値が設定されます．モデル，モデルのパラメータ，デフォルト値は，素子要素行の記述とともに次の節に載っています．

2.4 サブ回路

Spice の要素からなるサブ回路を定義して，素子モデルと同様な方法で参照することができます．サブ回路は，入力ファイル中で，要素行の集まりによって定義します．プログラムは，サブ回路が参照された場所に一連の要素を挿入します．サブ回路の大きさや複雑さに制限はありません．サブ回路は，他のサブ回路を含むことができます．サブ回路の使い方の例は，付録 Aにあります．

2.4.1 .SUBCKT 行

一般形:

.SUBCKT *subnam N1* 〈*N2 N3 ...*〉

例:

```
.SUBCKT OPAMP 1 2 3 4
```

回路の定義は，.SUBCKT 行で始まります．*SUBNAM* はサブ回路の名前で，*N1*, *N2*, ... は外部ノードで，0 以外を指定します．.SUBCKT に続く要素行の集まりによってサブ回路を定義します．サブ回路の定義の最後の行は .ENDS 行です (2.4.2節を参照のこと)．サブ回路の定義の中に制御行が現われてはなりません．それ以外のもの (別のサブ回路の定義，素子モデル，サブ回路の呼出し (2.4.3節を参照のこと) など) はサブ回路の定義に入れることができます．サブ回路の定義の一部となっている素子モデルや別のサブ回路の定義は，局所的なものです (つまり，そのようなモデルや定義は，サブ回路の定義の外側には見えません)．.SUBCKT 行に現われない要素のノードも，0 (グラウンド) を除いては局所的です．グラウンドは常に大域的です．

2.4.2 .ENDS 行

一般形:

.ENDS 〈*SUBNAM*〉

例:

```
.ENDS OPAMP
```

サブ回路の定義の最後の行は，.ENDS 行でなければなりません．サブ回路の名前が含まれている場合，そのサブ回路の定義が終わることを表わします．省略されている場合は，すべてのサブ回路の定義が終わります．名前は，サブ回路の定義が入れ子になっている場合のみ必要です．

2.4.3 サブ回路の呼出し

一般形:

X*YYYYYYY N1* 〈*N2 N3 ...*〉 *SUBNAM*

例:

```
X1 2 4 17 3 1 MULTI
```

Spice では，文字 'X' で始まる仮想要素を指定することにより，サブ回路を使用できます．その後には，サブ回路を展開する時に使う回路のノードを指定します．

2.5 ファイルを結合する: .INCLUDE 行

一般形:

.INCLUDE *filename*

例:

```
.INCLUDE /users/spice/common/wattmeter.cir
```

特に共通のモデルやサブ回路など，回路の記述の一部をさまざまな入力ファイルで再利用することがよくあります．Spice の任意の入力ファイルで .include 行を使うと，他のファイルの内容があたかもその場所にコピーされたようになります．ファイル名に関しては，オペレーティングシステムによる制限以外に，Spice による制限はありません．

3 回路の要素とモデル

不等号 (‘⟨’ ‘⟩’) で囲まれているデータフィールドは，オプションです．明示されている区切り文字 (括弧，等号など) は，オプションですが，区切り文字の存在を表わしています．将来の実装では，その区切り文字を必要とすることもあるでしょう．ここで示した区切り文字の付け方に一貫して従っていれば，入力ファイルが読みやすくなるでしょう．Spice は，枝の電圧および電流に関して (電流は電圧が下がる方向に流れるといった) 一貫した表わし方を用います．

3.1 基本的な素子

3.1.1 抵抗

一般形:

R*XXXXXXX N1 N2 VALUE*

例:

```
R1 1 2 100
RC1 12 17 1K
```

N1 と *N2* は，要素のノードです．*VALUE* は抵抗値 (Ω) で，正でも負でもかまいませんが，0 ではいけません．

3.1.2 半導体抵抗

一般形:

R*XXXXXXX N1 N2* ⟨*VALUE*⟩ ⟨*MNAME*⟩ ⟨L=*LENGTH*⟩ ⟨W=*WIDTH*⟩ ⟨TEMP=*T*⟩

例:

```
RLOAD 2 10 10K
RMOD 3 7 RMODEL L=10u W=1u
```

これは第 3.1.1節で示されている抵抗をさらに一般的にしたもので，温度の影響をモデル化しており，幾何的な情報とプロセスの仕様から実際の抵抗値を計算します．*VALUE* が指定されていれば，幾何的な情報に優先して抵抗値が定義されます．*MNAME* が指定されていれば，モデル *MNAME* の中のプロセスの情報と，与えられた *LENGTH* と *WIDTH* から抵抗値が計算されます．*VALUE* が指定されていない場合，*MNAME* と *LENGTH* を指定しなければなりません．*WIDTH* が指定されていない場合，モデルで与えられているデフォルトの幅が使われます．(オプションの) TEMP は，素子が動作する温度で .OPTION 制御行の温度設定よりも優先します．

3.1.3 半導体抵抗モデル (R)

この抵抗モデルは，幾何的な情報から抵抗値を計算し，温度の修正を行なえるような，プロセスに関連した素子のデータによって構成されます．使用できるパラメータは以下のとおりです．

名前	パラメータ	単位	省略値	例
TC1	1 次の温度係数	Z/°C	0.0	—
TC2	2 次の温度係数	Z/°C^2	0.0	—
RSH	シート抵抗	Z/[]	—	50
DEFW	省略時の幅	*m*	1e-6	2e-6
NARROW	エッチングによる側面の狭まり	*m*	0.0	1e-7
TNOM	パラメータ測定時の温度	°C	27	50

シート抵抗は狭まりのパラメータおよび抵抗素子の L と W と共に使われ，次の式により名目抵抗が決まります．

$$R = RSH\frac{L - NARROW}{W - NARROW}$$

DEFW は，素子の W が指定されていない場合の省略時の値として使われます．RSH または L のいずれかが指定されていない場合，標準の省略時の抵抗値 1kZ が使われます．TNOM は，.OPTIONS 制御行で与えられる回路全体の値を上書きするために使われ，このモデルのパラメータが別な温度で測定されたことを表わします．名目抵抗が計算された後，次の式により温度の影響を反映します．

$$R(T) = R(T_0)[1 + TC_1(T - T_0) + TC_2(T - T_0)^2]$$

3.1.4 コンデンサ

一般形:

C*XXXXXXX N+ N- VALUE* ⟨IC=*INCOND*⟩

例:

```
CBYP 13 0 1UF
COSC 17 23 10U IC=3V
```

N+ と *N-* は，それぞれ正と負のノードです．*VALUE* はファラド単位の静電容量です．

(オプションの) 初期状態は，当初 (時刻 0) のコンデンサの電圧 (単位 V) です．初期状態がもし指定されていても，.TRAN 制御行に UIC オプションが指定されている場合「のみ」適用されます．

3.1.5 半導体コンデンサ

一般形:

C*XXXXXXX N1 N2* ⟨*VALUE*⟩ ⟨*MNAME*⟩ ⟨L=*LENGTH*⟩ ⟨W=*WIDTH*⟩ ⟨IC=*VAL*⟩

例:

```
CLOAD 2 10 10P
CMOD 3 7 CMODEL L=10u W=1u
```

これは第 3.1.4節で説明したコンデンサの，より一般的な形式で，実際のコンデンサの容量を，厳密に幾何的な情報およびプロセスの仕様から計算できます．*VALUE* が指定された場合，それに

より静電容量が定義されます．*MNAME* が指定された場合，モデル *MNAME* のプロセスの情報と，指定された *LENGTH* と *WIDTH* から静電容量が計算されます．*VALUE* が指定されていない場合，*MNAME* および *LENGTH* を指定しなければなりません．*WIDTH* が指定されていない場合，モデルの省略時の幅が使われます．*VALUE* または *MNAME*, *LENGTH*, *WIDTH* のどちらかを指定しますが，両方指定してはなりません．

3.1.6 半導体コンデンサモデル (C)

コンデンサモデルは，厳密に幾何的な情報から静電容量を計算するのに使われる情報からなります．

名前	パラメータ	単位	省略時の値	例
CJ	ジャンクション底部容量	F/m^2	—	5e-5
CJSW	ジャンクション側壁容量	F/m	—	2e-11
DEFW	省略時素子幅	m	1e-6	2e-6
NARROW	側面のエッチングによる狭まり	m	0.0	1e-7

コンデンサの容量は次の式で計算されます．

$$CAP = C_J(LENGTH - NARROW)(WIDTH - NARROW) + 2C_{JSW}(LENGTH + WIDTH - 2NARROW)$$

3.1.7 インダクタ

一般形:

L*YYYYYYY* *N+* *N-* *VALUE* ⟨IC=*INCOND*⟩

例:

```
LLINK 42 69 1UH
LSHUNT 23 51 10U IC=15.7MA
```

N+ と *N-* は，それぞれ正と負のノードです．*VALUE* はヘンリー単位のインダクタンスです．

(オプションの) 初期条件は，当初 (時刻 0) に *N+* からインダクタを通って *N-* へ流れる電流 (単位 A) の大きさです．初期状態がもし指定されていても，.TRAN 制御行に UIC オプションが指定されている場合「のみ」適用されます．

3.1.8 相互インダクタ

一般形:

K*XXXXXXX* L*YYYYYYY* L*ZZZZZZZ* *VALUE*

例:

```
K43 LAA LBB 0.999
KXFRMR L1 L2 0.87
```

LYYYYYYY と LZZZZZZZ は 2 つの結合されたインダクタの名前で，*VALUE* は結合係数 *K* です．結合係数は 0 を超え 1 以下の値です.「ドット」の規約に従い，各インダクタの最初のノードにドットが付きます.

3.1.9 スイッチ

一般形:

SXXXXXXX *N+ N- NC+ NC- MODEL* ⟨ON⟩⟨OFF⟩
WYYYYYYY *N+ N- VNAM MODEL* ⟨ON⟩⟨OFF⟩

例:

```
s1 1 2 3 4 switch1 ON
s2 5 6 3 0 sm2 off
Switch1 1 2 10 0 smodel1
w1 1 2 vclock switchmod1
W2 3 0 vramp sm1 ON
wreset 5 6 vclck lossyswitch OFF
```

ノード 1 および 2 は，スイッチの端子がつながるノードです．モデル名は必須で，初期条件はオプションです．電圧制御スイッチでは，ノード 3 と 4 は，それぞれ正および負の制御ノードです．電流制御スイッチでは，制御電流は指定された電圧源を流れる電流です．制御電流の正の方向は，正のノードから電圧源を通り負のノードへと流れる方向です.

3.1.10 スイッチモデル (SW/CSW)

スイッチモデルは，Spice で使えるほとんど理想的なスイッチです．このスイッチは，抵抗が 0 から無限大へと変化するのではなく，常に有限の正の値をとるという意味で，まったく理想的であるとは言えません．開閉時の抵抗値を適切に選択することにより，他の回路の要素と比較して実質的に 0 と無限大にみなせます．使えるパラメータは以下のとおりです.

名前	パラメータ	単位	省略時の値	スイッチ
VT	threshold 電圧	V	0.0	S
IT	threshold 電流	A	0.0	W
VH	ヒステリシス電圧	V	0.0	S
IH	ヒステリシス電流	A	0.0	W
RON	閉抵抗	Z	1.0	両方
ROFF	開抵抗	Z	1/GMIN*	両方

*(GMIN の説明については 4.1節の .OPTIONS 制御行を参照のこと．省略時の開抵抗の値は $10^{12}\,\Omega$ になります．)

スイッチのような高度に非線形な理想的な要素を使うと，回路のノードの電圧に大きな不連続性が生じる場合があります．スイッチの状態が変化することによって引き起こされる急速な変化は，数値の丸め誤差や許容誤差の問題を起こし，間違った結果や時間ステップ (による解析) が困難な

状況へとつながります．スイッチを使う場合，以下のステップを踏むことにより，この状況を改善できます．

まず，理想的なスイッチのインピーダンスを回路の他の要素と比べて無視できるくらいにわずかに高くあるいは低く設定するのが懸命です．すべての場合において「理想的」に近いインピーダンスをスイッチに設定すると，上述した非連続性の問題を悪化させます．もちろん，MOSFET のような実際の素子をモデル化する場合，モデル化する素子の大きさに応じた現実的なレベルにオン抵抗を調整すべきです．

スイッチに広範囲のオン/オフ抵抗 (ROFF/RON $> 10^{12}$) を設定しなければならない場合，.OPTIONS 制御行を使って TRTOL を省略時の値 7.0 以下に設定して，過渡解析時の許容誤差を下げるべきです．コンデンサのまわりにスイッチを置いたとき，オプション CHGTOL も下げるべきです．これらの 2 つのオプションの推奨値は，それぞれ 1.0 と 1e-16 です．こららの変更により，回路の急速な変化による誤りが起きないよう，Spice3 がスイッチの接点をより注意深く扱うように指示します．

3.2 電圧源，電流源

3.2.1 独立電圧源，電流源

一般形:

V*XXXXXXX* *N+* *N-* ⟨⟨DC⟩ *DC/TRAN_VALUE*⟩ ⟨AC ⟨*ACMAG* ⟨*ACPHASE*⟩⟩⟩
+ ⟨DISTOF1 ⟨*F1MAG* ⟨*F1PHASE*⟩⟩⟩ ⟨DISTOF2 ⟨*F2MAG* ⟨*F2PHASE*⟩⟩⟩
I*YYYYYYY* *N+* *N-* ⟨⟨DC⟩ *DC/TRAN_VALUE*⟩ ⟨AC ⟨*ACMAG* ⟨*ACPHASE*⟩⟩⟩
+ ⟨DISTOF1 ⟨*F1MAG* ⟨*F1PHASE*⟩⟩⟩ ⟨DISTOF2 ⟨*F2MAG* ⟨*F2PHASE*⟩⟩⟩

例:

```
VCC 10 0 DC 6
VIN 13 2 0.001 AC 1 SIN(0 1 1MEG)
ISRC 23 21 AC 0.333 45.0 SFFM(0 1 10K 5 1K)
VMEAS 12 9
VCARRIER 1 0 DISTOF1 0.1 -90.0
VMODULATOR 2 0 DISTOF2 0.01
IIN1 1 5 AC 1 DISTOF1 DISTOF2 0.001
```

N+ と *N-* は，それぞれ正および負のノードです．電圧源は，必ずしも接地する必要はありません．正の電流の向きは，正のノードから電圧源を通り負の端子へ流れる方向です．正の値の電流源では，*N+* ノードから，電流源を通り *N-* 端子へと電流が流れます．電圧源は，回路を動作させるために使われるだけでなく，Spice の「電流計」としても使われます．すなわち，電流を測定するには値を 0 にした電圧源を回路に挿入します．そのようにしても，電圧源は回路をショートしていると見なされるため，回路の動作にまったく影響しません．

DC/TRAN_VALUE は，直流解析および過渡解析の場合の電源の値です．直流解析，過渡解析用の値が 0 の場合，この値を省略することができます．電圧源，電流源の値が時刻により変化しない (電源など) 場合，値の前に DC を付けても構いません．

ACMAG は交流の振幅，*ACPHASE* は交流の位相です．電源は交流解析時にこの値に設定されます．キーワード AC の後ろで *ACMAG* が省略された場合，1 が仮定されます．*ACPHASE* が省略され

た場合，0 が仮定されます．この電源が交流小信号の入力でない場合は，キーワード AC と交流の値を省略します．

DISTOF1 と DISTOF2 は，独立電源にそれぞれ周波数が *F1* および *F2* の歪みの入力があることを指定するキーワードです (4.3.3節の .DISTO 制御行の説明を参照のこと)．キーワードに続けて，オプションで振幅と位相を指定します．振幅と位相の省略時の値は，それぞれ 1.0 と 0.0 です．

過渡解析用に，時刻に依存する値を電源に指定することができます．時刻に依存する値を指定した場合，直流解析では時刻 0 の値が使われます．パルス，指数，正弦波，折れ線，単一周波数の FM といった，時刻に依存した関数が 5 つあります．電圧源，電流源の値 (振幅) 以外のパラメータが省略された，または 0 に設定された場合，示された省略時の値が仮定されます．(*TSTEP* は表示用の増分であり，*TSTOP* は最終の時刻です (説明は 3.3節の .TRAN 制御行を参照のこと)．)

パルス　一般形:

PULSE(*V1 V2 TD TR TF PW PER*)

例:

```
VIN 3 0 PULSE(-1 1 2NS 2NS 2NS 50NS 100NS)
```

パラメータ	省略時の値	単位
V1 (初期値)		V または A
V2 (パルス時の振幅)		V または A
TD (遅延時間)	0.0	秒
TR (立ち上がり時間)	TSTEP	秒
TF (立ち下がり時間)	TSTEP	秒
PW (パルスの幅)	TSTOP	秒
PER (周期)	TSTOP	秒

パルスの波形は次のようになります．

時刻	値
0	*V1*
TD	*V1*
TD + *TR*	*V2*
TD + *TR* + *PW*	*V2*
TD + *TR* + *PW* + *TF*	*V1*
TSTOP	*V1*

中間の値は，線形補間によって決まります．

正弦波　一般形:

SIN(*VO VA FREQ TD THETA*)

例:

```
VIN 3 0 SIN(0 1 100MEG 1NS 1E10)
```

パラメータ	省略時の値	単位
VO (オフセット)		V または A
VA (振幅)		V または A
FREQ (周波数)	1/TSTOP	Hz
TD (遅延)	0.0	秒
THETA (減衰係数)	0.0	1/秒

波形は次のようになります.

時刻	値
0 から *TD*	V_O
TD から TSTOP	$V_O + V_A e^{(t-TD)THETA} \sin(2\pi FREQ(t+TD))$

指数　一般形:

EXP(*V1　V2　TD1　TAU1　TD2　TAU2*)

例:

```
VIN 3 0 EXP(-4 -1 2NS 30NS 60NS 40NS)
```

パラメータ	省略時の値	単位
V1 (初期値)		V または A
V2 (パルス時の値)		V または A
TD1 (立ち上がり遅延時間)	0.0	秒
TAU1 (立ち上がり時定数)	TSTEP	秒
TD2 (立ち下がり遅延時間)	*TD1* + TSTEP	秒
TAU2 (立ち下がり時定数)	TSTEP	秒

波形は次のようになります.

時刻	値
0 から *TD1*	V_1
TD1 から *TD2*	$V_1 + (V_2 - V_1)\left(1 - e^{\frac{-(t-TD_1)}{TAU_1}}\right)$
TD2 から TSTOP	$V_1 + (V_2 - V_1)\left(1 - e^{\frac{-(t-TD_1)}{TAU_1}}\right) + (V_1 - V_2)\left(1 - e^{\frac{-(t-TD_2)}{TAU_2}}\right)$

折れ線　一般形:

PWL(*T1　V1* ⟨*T2　V2　T3　V3　T4　V4　...*⟩)

例:

```
VCLOCK 7 5 PWL(0 -7 10NS -7 11NS -3 17NS -3 18NS -7 50NS -7)
```

値の組 (*Ti*, *Vi*) は，時刻 = *Ti* における電源の値を *Vi* (V または A) と指定します．中間の時刻における値は，入力された値を線形補間したものになります.

単一周波数 FM　一般形:

SFFM(*VO VA FC MDI FS*)

例:

```
V1 12 0 SFFM(0 1M 20K 5 1K)
```

パラメータ	省略時の値	単位
VO (オフセット)		V または A
VA (振幅)		V または A
FC (搬送波周波数)	1/TSTOP	Hz
MDI (変調指数)		
FS (信号周波数)	1/TSTOP	Hz

波形は次の式になります.

$$V(t) = V_O + V_A \sin\Big(2\pi F_C t + MDI \sin(2\pi F_S t)\Big)$$

3.2.2　線形依存電圧源，電流源

Spice では，次の 4 つの式で表わされる線形依存電圧源，電流源を使えます.

$$i = gv \qquad v = ev \qquad i = fi \qquad v = hi$$

ここで，g, e, f, h は，それぞれ相互コンダクタンス，電圧増幅度，電流増幅度，相互抵抗を表わす定数です.

電圧制御線形電流源　一般形:

G*XXXXXXX N+ N- NC+ NC- VALUE*

例:

```
G1 2 0 5 0 0.1MMHO
```

N+ および *N-* は，それぞれ正と負のノードです．電流は，正のノードから電圧源を通って，負のノードへと流れます．*NC+* および *NC-* は，それぞれ正と負の制御ノードです．*VALUE* は相互コンダクタンス (単位 ℧，今は S) です.

電圧制御線形電圧源　一般形:

E*XXXXXXX N+ N- NC+ NC- VALUE*

例:

```
E1 2 3 14 1 2.0
```

N+ および *N-* は，それぞれ正と負のノードです．*NC+* および *NC-* は，それぞれ正と負の制御ノードです．*VALUE* は電圧増幅率です.

電流制御線形電流源　一般形:

F*XXXXXXX N+ N- VNAM VALUE*

例:

```
F1 13 5 VSENS 5
```

N+ および *N-* は，それぞれ正と負のノードです．電流は，正のノードから電流源を通って，負のノードへと流れます．*VNAM* は，制御電流が流れている電圧源の名前です．制御電流の正の方向は，*VNAM* の正のノードから電圧源を通り負のノードへ流れる方向です．*VALUE* は電流増幅率です．

電流制御線形電圧源　一般形:

H*XXXXXXX N+ N- VNAM VALUE*

例:

```
HX 5 17 VZ 0.5K
```

N+ および *N-* は，それぞれ正と負のノードです．*VNAM* は，制御電流が流れている電圧源の名前です．制御電流の正の方向は，*VNAM* の正のノードから電圧源を通り負のノードへ流れる方向です．*VALUE* は相互抵抗 (単位は Ω) です．

3.2.3　非線形依存電源

一般形:

B*XXXXXXX N+ N-* ⟨I=*EXPR*⟩ ⟨V=*EXPR*⟩

例:

```
B1 0 1 I=cos(v(1))+sin(v(2))
B1 0 1 V=ln(cos(log(v(1,2)^2)))-v(3)^4+v(2)^v(1)
B1 3 4 I=17
B1 3 4 V=exp(pi^i(vdd))
```

N+ は正のノード，*N-* は負のノードです．V, I パラメータの値により，それぞれ電圧源の電圧および電流源の電流が決まります．I が指定されている場合，この素子は電流源となり，V が指定されている場合，この素子は電圧源となります．これらのパラメータのうち 1 つのみを指定します．

非線形電源の小信号交流の振る舞いは，直流動作点におけるその電源の微係数を比例定数とした線形電源と同じです．

V および I に与える式は，システム中の電圧，および電圧源を通る電流の任意の関数にすることができます．次の実数変数の関数が定義されています．

abs	asinh	cosh	sin
acos	atan	exp	sinh
acosh	atanh	ln	sqrt
asin	cos	log	tan

関数 u は単位ステップ関数であり，引数が 0 以上のとき値が 1 になり，引数が 0 未満のとき値が 0 になります．関数 uramp は単位ステップ関数の積分で，引数を x とすると，x が 0 以下の場合に値が 0 となり，x が 0 を超える場合に値が x となります．これらの 2 つの関数は，部分部分の非線形関数を合成する際に役立ちますが，収束が悪くなることがあります．

次の標準的な演算子が定義されています．

```
+   -   *   /   ^   単項 -
```

log, ln, sqrt の引数が 0 未満の場合，引数の絶対値が使われます．除数あるいは log, ln の引数が 0 となった場合，エラーが生じます．偏微分の関数の引数が定義域を外れると，その他の問題が生じることもあります．

時間を式に入れるには，定電流源の電流をコンデンサに積分し，その電圧を使います (コンデンサの初期電圧を忘れずに設定しておくこと)．非線形の抵抗，コンデンサ，インダクタは，非線形依存電源によって合成できます．非線形抵抗は自明です．非線形コンデンサおよびインダクタは，対応する線形素子を使い，非線形依存電源によって生じる変数の変化により実現できます．次のサブ回路は，非線形コンデンサを実現します．

```
.Subckt nlcap   pos neg
* Bx: f(input voltage) を計算する
Bx   1    0    v = f(v(pos,neg))
* Cx: 線形コンデンサ
Cx   2    0    1
* Vx: コンデンサに流れ込む電流を計測する電流計
Vx   2    1    DC 0Volts
* Cx に流れる電流を回路に反映する
Fx   pos  neg  Vx 1
.ends
```

非線形インダクタも同様です．

3.3 伝送線路

3.3.1 無損失伝送線路

一般形:

T*XXXXXXX N1 N2 N3 N4* Z0=*VALUE* ⟨TD=*VALUE*⟩ ⟨F=*FREQ* ⟨NL=*NRMLEN*⟩⟩
\+ ⟨IC=*V1*, *I1*, *V2*, *I2*⟩

例:

```
T1 1 0 2 0 Z0=50 TD=10NS
```

N1 および *N2* はポート 1 のノード，*N3* および *N4* はポート 2 のノードです．Z0 は特性インピーダンスです．伝送線路の長さは 2 つの形式のどちらかで指定します．伝送遅延 TD は，直接 (例えば TD=10ns) 指定できます．または，周波数 F と，周波数 F における伝送線路内の波長によって基準化した電気長 NL を指定することもできます．周波数が指定され，NL が省略された場合，0.25 が

仮定されます (すなわち，周波数の 1/4 波長となります)．線路長を表わすどちらの形式もオプションですが，どちらか一方を指定する必要があります．

この要素は，1 つの伝送モードのみをモデル化しています．実際の回路で 4 つのノードすべてが異なっている場合，2 つのモードが励起されることがあります．そのような状況をシミュレートするには，2 つの伝送線路が必要になります．(より詳しい説明は付録 Aの例を参照のこと．)

(オプションの) 初期条件の指定は，伝送線路の各ポートの電圧と電流により行ないます．初期条件が指定された場合，.TRAN 制御行で UIC オプションが指定されている場合「のみ」適用されます．

注意：損失が 0 の有損失伝送線路 (下記参照) は，実現の詳細のため，無損失伝送線路よりも正確です．

3.3.2 有損失伝送線路

一般形:

O*XXXXXXX N1 N2 N3 N4 MNAME*

例:

```
O23 1 0 2 0 LOSSYMOD
OCONNECT 10 5 20 5 INTERCONNECT
```

これは 2 ポートの単一導体有損失伝損線路の畳み込みモデルです．*N1* と *N2* はポート 1 のノード，*N3* と *N4* はポート 2 のノードです．注意：損失が 0 の有損失伝送線路は，実現の詳細のため，無損失伝送線路よりも正確です．

3.3.3 有損失伝送線路モデル (LTRA)

一様 RLC/RC/LC/RG 伝送線路モデル (以後 LTRA モデルと呼ぶ) は，一様定数分布伝送線路をモデル化しています．RC および LC は，さらに URC および TRA モデルによりモデル化されています．しかし，さらに新しい LTRA モデルは，他のモデルと比べると通常より速く，より正確です．LTRA モデルの動作は，伝送線路のインパルス応答の入力に関する畳み込みに基づいています ([8] を参照のこと)．

LTRA モデルは，いくつかのパラメータをとり，必須のパラメータとオプションのパラメータがあります．

名前	パラメータ	単位/種別	省略時の値	例
R	抵抗/長さ	Z/unit	0.0	0.2
L	インダクタンス/長さ	H/unit	0.0	9.13e-9
G	コンダクタンス/長さ	Ʊ/unit	0.0	0.0
C	静電容量/長さ	F/unit	0.0	3.65e-12
LEN	伝送線の長さ		なし	1.0
REL	ブレークポイントの制御	任意	1	0.5
ABS	ブレークポイントの制御		1	5
NOSTEPLIMIT	線路遅延未満に時間ステップを制限しない	フラグ	not set	set
NOCONTROL	複雑な時間ステップの制御を行なわない	フラグ	not set	set
LININTERP	線形補間を使う	フラグ	not set	set
MIXEDINTERP	2 次補間に失敗したら線形補間を使う	フラグ	not set	set
COMPACTREL	ヒストリ圧縮のための特別な reltol		RELTOL	1.0e-3
COMPACTABS	ヒストリ圧縮のための特別な abstol		ABSTOL	1.0e-9
TRUNCNR	時間ステップの制御に Newton-Raphson 法を使う	フラグ	not set	set
TRUNCDONTCUT	インパルス応答のエラーを低くするために時間ステップを制御しない	フラグ	not set	set

これまでに次の形式の伝送線路が実現されています：RLC (直列損失のみの一様伝送線路)，RC (一様 RC 伝送線路)，LC (無損失伝送線路)，RG (分布直列抵抗並列コンダクタンスのみ)．その他の組み合わせはエラーとなり，試すべきではありません．伝送線路の長さ LEN を指定しなければなりません．

NOSTEPLIMIT は，RLC の場合に時間ステップを線路の遅延未満に制限するという省略時の制約を解除するフラグです．NOCONTROL は，RLC および RC の場合に畳み込みエラーの基準に基づく時間ステップの省略時の制限を解除するフラグです．これによりシミュレーションが速くなりますが，結果の精度が悪くなる場合があります．LININTERP は，遅延された信号を計算するのに，省略時の 2 次補間ではなく線形補間を使うフラグです．MIXEDINTERP は，2 次補間が適切かどうか判断する基準を使い，適切でない場合は線形補間を使うよう指示するフラグです．2 次補間が適切な場合は 2 次補間が使われます．TRUNCDONTCUT は，インパルス応答関連の量を実際に計算するときのエラーを押さえるために省略時に行なわれる時間ステップの短縮化をやめるフラグです．COMPACTREL および COMPACTABS は，畳み込みのために保存されている過去の値のヒストリを制御する量です．これらに大きな値を指定すると，精度は低くなりますが，通常シミュレーションの速度が速くなります．これらの値は，.OPTIONS の節 (4.1) で説明されている TRYTOCOMPACT オプションと共に使います．TRUNCNR は，時間ステップ制御ルーチンで，適切な時間ステップを決めるのに Newton-Raphson 法を使うよう指示するフラグです．省略時には，前の時間ステップを半分にして試してみる方法を使います．REL および ABS は，ブレークポイントの設定を制御する量です．

シミュレーションのスピードを速くするために，もっとも試す価値があるオプションは，REL です．省略時の値 1 は，精度の観点からは通常安全ですが，ときどき計算時間が長くなることがあります．2 を超える値は，すべてのブレークポイントを排除し，精度の観点からは安全ではないことを念頭に置きつつ，回路の残りの部分の性質に身を任せることになります．回路に鋭い非連続性が

ないと予測されれば，通常ブレークポイントをすべて取り除いて構いません．0 から 1 の値は，通常必要ありませんが，多くのブレークポイントを設定するために使います．

.OPTIONS 制御行で TRYTOCOMPACT が指定されている場合，COMPACTREL も試してみるとよいでしょう．有効な値の範囲は 0 から 1 です．大きな値を指定すると，シミュレーションの精度は低くなりますが，スピードが速くなる場合もあります．TRYTOCOMPACT が .OPTIONS 制御行に指定されていない場合，ヒストリの圧縮は行なわれず，精度は高くなります．NOCONTROL, TRUNCDONTCUT, NOSTEPLIMIT も精度を犠牲にしてスピードを速くする傾向があります．

3.3.4 一様分布 RC 線路

一般形:

U*XXXXXXX N1 N2 N3 MNAME* L=*LEN* ⟨N=*LUMPS*⟩

例:

```
U1 1 2 0 URCMOD L=50U
URC2 1 12 2 UMODL l=1MIL N=6
```

N1 および *N2* は，RC 伝送線路が結んでいる 2 つのノードで，*N3* はコンデンサがつながるノードです．*MNAME* はモデルの名前，*LEN* は RC 伝送線路の長さ (単位は *m*) です．*LUMPS* は，RC 伝送線路をモデル化するのに使う部分の個数です (このパラメータが省略された場合の動作については，モデルの説明を参照のこと)．

3.3.5 一様分布 RC 線路モデル (URC)

URC モデルは，L. Gertzberrg が 1974 年に提案したモデルから導いたものです．このモデルは，URC 伝送線路をサブ回路的に展開して，内部的に生成されたノードを持つひとかたまりの RC 部分が複数集まって構成されるネットワークにより構成されます．RC 部分は，URC 伝送線路の中央に向かって，公比が K の等比数列になっています．使われる部分の数は，URC 伝送線路素子で指定されなければ，次の式で決定されます．

$$N = \frac{\log\left(F_{max}\frac{R}{L}\frac{C}{L}2\pi L^2\left(\frac{K-1}{K}\right)^2\right)}{\log K}$$

URC 伝送線路は，ISPERL パラメータの値に非 0 を指定しなければ，抵抗とコンデンサの部分によってのみ構成されます．非 0 の値を指定した場合，コンデンサの代わりに，ゼロバイアス時の接合容量が置き換える前の静電容量と等価で，伝送線路 1 m あたりの飽和電流が ISPERL A の逆バイアスのダイオードが使われます．オプションの RSPERL は 1 m メートルあたりの直列抵抗 (単位 Ω) を指定します．

	名前	パラメータ	単位	省略時の値	例	面積
1	K	伝播定数	—	2.0	1.2	—
2	FMAX	最大周波数	Hz	1.0G	6.5Meg	—
3	RPERL	単位長あたりの抵抗	Z/*m*	1000	10	—
4	CPERL	単位長あたりの静電容量	F/*m*	1.0e-15	1pF	—
5	ISPERL	単位長あたりの飽和電流	A/*m*	0	—	—
6	RSPERL	単位長あたりのダイオード抵抗	Z/*m*	0	—	—

3.4 トランジスタ，ダイオード

ダイオード，BJT, FET, MES-FET で使われる面積係数は，指定されたモデルの等価並列素子の個数を決定します．これに影響されるパラメータには，後述するモデルの記述において「面積」の列にアスタリスクを付けてあります．チャネルおよびドレイン，ソースの拡散に関するいくつかの幾何的な係数は，MOSFET の素子行で指定できます．

ある素子には，2 つの形式で初期状態を指定できます．第一の形式は，2 つ以上安定な状態がある回路の直流の収束を向上させます．素子が OFF であると指定された場合，その素子の端子の電圧が 0 であるとして直流動作点が決定されます．収束したら，その端子の電圧の値を正しく求めるために繰り返しを続けます．回路に直流安定状態が 2 つ以上ある場合，OFF オプションを使って，望ましい状態に対応する解を得られます．実際に素子が導通している場合に素子を OFF と指定すると，プログラムは (解が収束すると仮定して) 正しい解を得ようとしますが，プログラムは 2 つ以上の別々の解に自力で収束しなければならないので，より多くの繰り返しが必要になります．.NODESET 制御行は OFF オプションと同様の目的に使えます．.NODESET オプションは使うのが簡単で，収束を助けるための好ましい手段です．

初期条件の 2 番目の形式は，過渡解析と共に指定するものです．これは，上述の収束を助ける方法とは異なり，本当の「初期条件」です．初期条件の詳しい説明は，.IC 制御行 (4.2.2節) と .TRAN 制御行 (4.3.9節) の説明を参照のこと．

3.4.1 接合型ダイオード

一般形:

D*XXXXXXX* *N+* *N-* *MNAME* ⟨*AREA*⟩ ⟨OFF⟩ ⟨IC=*VD*⟩ ⟨TEMP=*T*⟩

例:

```
DBRIDGE 2 10 DIODE1
DCLMP 3 7 DMOD 3.0 IC=0.2
```

N+ および *N+* は，それぞれ正と負のノードです．*MNAME* はモデルの名前，*AREA* は面積係数，OFF はこの素子に対して直流解析をする際の (オプションの) 初期条件です．面積係数が省略された場合，1.0 という値が仮定されます．IC=*VD* という (オプションの) 初期条件の指定は，静止動作点とは異なる状態から過渡解析を始めるために，.TRAN 制御行の UIC オプションと共に使います．(オプションの) TEMP 値は，この素子が動作する温度で，.OPTION 制御行の温度の指定よりも優先します．

3.4.2 ダイオードモデル (D)

ダイオードの直流特性は，パラメータ IS と N によって決定されます．抵抗 RS のパラメータもあります．電荷蓄積効果は，通過時間 TT と，パラメータ CJO, VJ, M によって定まる非線形の空乏層容量としてモデル化されています．飽和電流の温度依存性は，エネルギーパラメータ EG と飽和電流指数パラメータ XTI によって定義されます．これらのパラメータが測定された名目温度は TNOM で，このパラメータの省略時の値は .OPTIONS 制御行で指定された回路全体の値です．逆方向降伏は，逆方向のダイオードの電流が指数的に増加するようにモデル化されており，パラメータ BV および IBV (両者とも正の値) によって決定されます．

	名前	パラメータ	単位	省略時の値	例	面積
1	IS	飽和電流	A	1.0e-14	1.0e-14	*
2	RS	抵抗	Z	0	10	*
3	N	エミッション係数	—	1	1.0	
4	TT	通過時間	sec	0	0.1ns	
5	CJO	ゼロバイアス接合容量	F	0	2pF	*
6	VJ	接合電位	V	1	0.6	
7	M	勾配係数	—	0.5	0.5	
8	EG	活性化エネルギー	eV	1.11	1.11 Si	
					0.69 Sbd	
					0.67 Ge	
9	XTI	飽和電流温度指数	—	3.0	3.0 jn	
					2.0 Sbd	
10	KF	フリッカ雑音係数	—	0		
11	AF	フリッカ雑音指数	—	1		
12	FC	順方向バイアス空乏層容量式の係数	—	0.5		
13	BV	逆方向降伏電圧	V	∞	40.0	
14	IBV	降伏電圧時の電流	A	1.0e-3		
15	TNOM	パラメータ測定温度	°C	27	50	

3.4.3 バイポーラ接合型トランジスタ (BJTs)

一般形:

Q*XXXXXXX NC NB NE* ⟨*NS*⟩ *MNAME* ⟨*AREA*⟩ ⟨OFF⟩ ⟨IC=*VBE*, *VCE*⟩ ⟨TEMP=*T*⟩

例:

```
Q23 10 24 13 QMOD IC=0.6, 5.0
Q50A 11 26 4 20 MOD1
```

NC, NB, NE は，それぞれコレクタ，ベース，エミッタのノードです．*NS* は (オプションの) サブストレートノードです．指定されていない場合，グラウンドが使われます．*MNAME* はモデルの名前，*AREA* は面積係数，OFF はこの素子に対して DC 解析をする際の (オプションの) 初期条件です．面積係数が省略された場合，1.0 という値が仮定されます．IC=*VBE*, *VCE* という (オプションの) 初期条件の指定は，静止動作点とは異なる状態から過渡解析を始めるために，.TRAN 制御行の UIC

オプションと共に使います．過渡解析の初期条件を設定するよりよい方法については，.IC 制御行の説明 (4.2.2節) を参照のこと．(オプションの) TEMP 値は，この素子が動作する温度で，.OPTION 制御行の温度の指定よりも優先します．

3.4.4 バイポーラ接合型トランジスタモデル (NPN/PNP)

Spice のバイポーラ接合型トランジスタモデルは，Gummel and Poon の積分電荷制御モデルの改良です．この改良 Gummel-Poon モデルは，元のモデルを拡張して，バイアスレベルが高いときのいくつかの効果を取り入れています．このモデルは，いくつかのパラメータが指定されていない場合，より単純な Ebers-Moll モデルに自動的に簡略化されます．改良 Gummel-Poon モデルで使われるパラメータ名は，プログラムの利用者によりわかりやすいように，また物理的あるいは回路設計的な思考を反映するように選ばれています．

直流モデルは，順方向電流増幅率を決定するパラメータ IS, BF, NF, ISE, IKF, NE と，逆方向電流増幅率特性を決定するパラメータ IS, BR, NR, ISC, IKR, NC と，順領域および逆領域の出力コンダクタンスを決定する VAF と VAR により定義されます．抵抗 RB, RC, RE というパラメータもあり，RB は電流に依存することもできます．ベースの電荷蓄積は，順方向および逆方向の通過時間 TF, TR (順方向通過時間 TF は必要ならバイアスに依存するようにできる) と，B-E 接合は CJE, VJE, MJE によって，B-C 接合は CJC, VJC, MJC によって，C-S (コレクタ-サブストレート) 接合は CJS, VJS, MJS によって決定される非線形の空乏層容量によってモデル化されています．飽和電流 IS の温度依存性は，エネルギーギャップ EG と飽和電流指数 XTI によって決定されます．さらに，新しいモデルでは，ベース電流の温度依存性は，ベータ温度指数 XTB によりモデル化されています．指定された値は，温度 TNOM で測定されたと仮定されます．これは .OPTIONS 制御行で指定できますが，.MODEL 行で指定すればそちらが優先します．

改良 Gummel-Poon モデルで使用される BJT のパラメータを以下に示します．Spice2 以前のバージョンのパラメータ名も受け付けます．

改良 Gummel-Poon BJT パラメータ

	名前	パラメータ	単位	省略時の値	例	面積
1	IS	トランスポート飽和電流	A	1.0e-16	1.0e-15	*
2	BF	理想最大順方向ベータ	—	100	100	
3	NF	順方向電流エミッション係数	—	1.0	1	
4	VAF	順方向アーリー電圧	V	∞	200	
5	IKF	高電流順方向ベータ降下点	A	∞	0.01	*
6	ISE	B-E 漏洩飽和電流	A	0	1.0e-13	*
7	NE	B-E 漏洩エミッション係数	—	1.5	2	
8	BR	理想最大逆方向ベータ	—	1	0.1	
9	NR	逆方向電流エミッション係数	—	1	1	
10	VAR	逆方向アーリー電圧	V	∞	200	
11	IKR	高電流逆方向ベータ降下点	A	∞	0.01	*
12	ISC	B-C 漏洩飽和電流	A	0	1.0e-13	*
13	NC	B-C 漏洩エミッション係数	—	2	1.5	
14	RB	ゼロバイアスベース抵抗	Z	0	100	*
15	IRB	ベース抵抗が最小値との中間になる電流	A	∞	0.1	*

	名前	パラメータ	単位	省略時の値	例	面積
16	RBM	高電流時の最小ベース抵抗	Z	RB	10	*
17	RE	エミッタ抵抗	Z	0	1	*
18	RC	コレクタ抵抗	Z	0	10	*
19	CJE	B-E ゼロバイアス空乏層容量	F	0	2pF	*
20	VJE	B-E 固有電位	V	0.75	0.6	
21	MJE	B-E 接合指数係数	—	0.33	0.33	
22	TF	理想順方向通過時間	sec	0	0.1ns	
23	XTF	TF のバイアス依存係数	—	0		
24	VTF	TF の V_{BC} 依存を表わす電圧	V	∞		
25	ITF	TF に与える高電流の影響パラメータ	A	0		*
26	PTF	周波数 = 1.0/(TF × 2π) Hz における追加位相	deg	0		
27	CJC	B-C ゼロバイアス空乏層容量	F	0	2pF	*
28	VJC	B-C 固有電位	V	0.75	0.5	
29	MJC	B-C 接合指数係数	—	0.33	0.5	
30	XCJC	内部ベースノードにつながる B-C 空乏層容量の割合	—	1		
31	TR	理想逆方向通過時間	sec	0	10ns	
32	CJS	ゼロバイアスコレクタ-サブストレート容量	F	0	2pF	*
33	VJS	サブストレート接合固有電位	V	0.75		
34	MJS	サブストレート接合指数係数	—	0	0.5	
35	XTB	順方向および逆方向ベータ温度指数	—	0		
36	EG	IS の温度効果に使われるエネルギーギャップ	eV	1.11		
37	XTI	IS の温度効果の指数	—	3		
38	KF	フリッカ雑音係数	—	0		
39	AF	フリッカ雑音指数	—	1		
40	FC	順バイアス空乏層容量式の係数	—	0.5		
41	TNOM	パラメータ測定温度	°C	27	50	

3.4.5 接合型 FET (JFETs)

一般形:

JXXXXXXX *ND NG NS MNAME* ⟨*AREA*⟩ ⟨OFF⟩ ⟨IC=*VDS*, *VGS*⟩ ⟨TEMP=*T*⟩

例:

```
J1 7 2 3 JM1 OFF
```

ND, *NG*, *NS* は，それぞれドレイン，ゲート，ソースのノードです．*MNAME* はモデルの名前，*AREA* は面積係数，OFF はこの素子に対して直流解析をする際の (オプションの) 初期条件です．面

積係数が省略された場合，1.0 という値が仮定されます．IC=*VDS*，*VGS* という (オプションの) 初期条件の指定は，静止動作点とは異なる状態から過渡解析を始めるために，.TRAN 制御行の UIC オプションと共に使います．過渡解析の初期条件を設定するよりよい方法については，.IC 制御行の説明 (4.2.2節) を参照のこと．(オプションの) TEMP 値は，この素子が動作する温度で，.OPTION 制御行の温度の指定よりも優先します．

3.4.6 接合型 FET モデル (NJF/PJF)

この JFET モデルは，Shichman and Hodges の FET モデルから導いたものです．直流特性は，ゲート電圧に対するドレイン電流の変化を決定するパラメータ VTO, BETA と，出力コンダクタンスを決定する LAMBDA と，2 つのゲート接合の飽和電流 IS により定義されます．抵抗 RD と RS というパラメータもあります．電荷蓄積は，2 つのゲート接合について，非線形の空乏層容量によってモデル化されています．空乏層容量は，接合電圧の −1/2 乗で変化し，パラメータ CGS, CGD, PB で定義されます．

Spice3f およびそれ以降のバージョンでは，調整パラメータ B が追加されました．詳細は [9] を参照のこと．

	名前	パラメータ	単位	省略時の値	例	面積
1	VTO	しきい値電圧 (V_{TO})	V	−2.0	−2.0	
2	BETA	相互コンダクタンスパラメータ (B)	A/V^2	1.0e-4	1.0e-3	*
3	LAMBDA	チャネル長調整パラメータ (L)	1/V	0	1.0e-4	
4	RD	ドレイン抵抗	Z	0	100	*
5	RS	ソース抵抗	Z	0	100	*
6	CGS	ゼロバイアス G-S 接合容量 (C_{gs})	F	0	5pF	*
7	CGD	ゼロバイアス G-D 接合容量 (C_{gd})	F	0	1pF	*
8	PB	ゲート接合電位	V	1	0.6	
9	IS	ゲート接合飽和電流 (I_S)	A	1.0e-14	1.0e-14	*
10	B	ドーピング広がりパラメータ	—	1	1.1	
11	KF	フリッカ雑音係数	—	0		
12	AF	フリッカ雑音指数	—	1		
13	FC	順方向バイアス空乏層容量式の係数	—	0.5		
14	TNOM	パラメータ測定温度	°C	27	50	

3.4.7 MOSFET

一般形:

M*XXXXXXX ND NG NS NB MNAME* ⟨L=*VAL*⟩ ⟨W=*VAL*⟩ ⟨AD=*VAL*⟩ ⟨AS=*VAL*⟩
+ ⟨PD=*VAL*⟩ ⟨PS=*VAL*⟩ ⟨NRD=*VAL*⟩ ⟨NRS=*VAL*⟩ ⟨OFF⟩
+ ⟨IC=*VDS*, *VGS*, *VBS*⟩ ⟨TEMP=*T*⟩

例:

```
M1 24 2 0 20 TYPE1
M31 2 17 6 10 MODM L=5U W=2U
M1 2 9 3 0 MOD1 L=10U W=5U AD=100P AS=100P PD=40U PS=40U
```

ND, *NG*, *NS*, *NB* は，それぞれドレイン，ゲート，ソース，バルク (サブストレート) ノードです．*MNAME* はモデルの名前です．L, W は，チャネルの長さと幅です (単位 *m*)．AD および AS は，ドレインとソースの拡散領域の面積です (単位 m^2)．U という接尾子は μm ($10^{-6}\,m$) で，p は μm^2 ($10^{-12}\,m^2$) です．L, W, AD, AS のいずれかが指定されなかった場合，省略時の値が使われます．省略時の値を使うと，入力ファイルの表現が簡単になるだけでなく，素子の寸法を変更する際の編集も簡単になります．PD および PS は，ドレインとソースの接合の周囲の長さです (単位 *m*)．NRD および NRS は，ドレインおよびソースの拡散の等価的な「ます」の数を指定します．これらの値は，各トランジスタの寄生直列ドレイン，ソース抵抗を正確に表わすため，.MODEL 制御行のシート抵抗 RSH に掛かります．PD, PS の省略時の値は 0.0 で，NRD, NRS の省略時の値は 1.0 です．OFF はこの素子に対して直流解析をする際の (オプションの) 初期条件です．IC=*VDS*，*VGS*，*VBS* という (オプションの) 初期条件の指定は，静止動作点とは異なる状態から過渡解析を始めるために，.TRAN 制御行の UIC オプションと共に使います．過渡解析の初期条件を設定するよりよい方法については，.IC 制御行の説明 (4.2.2節) を参照のこと．(オプションの) TEMP 値は，この素子が動作する温度で，.OPTION 制御行の温度の指定よりも優先します．

温度の指定は，レベル 1, 2, 3, 6 の MOSFET「のみ」有効で，レベル 4, 5 (BSIM) 素子には使えません．

3.4.8 MOSFET モデル (NMOS/PMOS)

Spice は，I-V 特性の定式化が異なる 4 つの MOSFET 素子モデルを提供します．変数 LEVEL により使うモデルを指定します．

LEVEL=1 → Shichman-Hodges
LEVEL=2 → MOS2 ([1] で記述されている)
LEVEL=3 → MOS3, 半経験的モデル ([1] を参照)
LEVEL=4 → BSIM ([3] で記述されている)
LEVEL=5 → 新しい BSIM (BSIM2; [5] で記述されている)
LEVEL=6 → MOS6 ([2] で記述されている)

レベル 1 からレベル 3 の MOSFET の直流特性は，素子のパラメータ VTO, KP, LAMBDA, PHI, GAMMA により定義されています．これらのパラメータは，プロセスのパラメータ (NSUB, TOX など) が与えられている場合，Spice によって計算されますが，利用者が指定した値が優先されます．エンハンスメント型の N-チャネル (P-チャネル) 素子の場合，VTO は正 (負) で，デプレッション型の場合 VTO は負 (正) です．電荷蓄積は，重複容量を表わす 3 つの定容量 CGSO, CGDO, CGBO と，ゲート，ソース，ドレイン，バルク領域に分布する非線形の酸化物薄膜容量と，底部と周辺部に別れた 2 つのサブストレート接合の非線形の空乏層容量 (それぞれ接合電圧の MJ 乗，MJSW 乗で変化する) によりモデル化され，パラメータ CBD, CBS, CJ, CJSW, MJ, MJSW, PB によって決定されます．電荷蓄積効果は，Meyer が提案した不連続線形電圧依存容量モデルによりモデル化されています．LEVEL=1 モデルでは，酸化物薄膜の電荷蓄積効果は，少し異なる形で扱われます．これらの電圧依存の容量は，入力の記述に TOX が指定された場合のみ含まれ，Meyer の式を用いて表わされます．

接合を表わすパラメータがいくつか重複しています．たとえば，逆方向電流は IS (単位 A) または JS (単位 A/m^2) のいずれかで入力できます．前者は絶対的な値であり，後者は AD または AS が掛けられて，それぞれドレイン，ソース接合の逆方向電流となります．面積は省略時の値にすることができ，この素子の行で入力された AD, AS と接合の特徴を常に関連づけるのは意味がないので，

この手法が選ばれました．同様な考え方は，ゼロバイアス接合容量 CBD, CBS (単位 F)，と CJ (単位 F/m^2) にも当てはまります．寄生直列ドレイン，ソース抵抗は，RD, RS (単位 Ω)，または RSH (単位 Ω/ます) で指定できます．後者は，この素子の行で入力されたますの数 NRD, NRS 倍されます．

KAPPA パラメータに関するレベル 3 モデルの不連続性が発見されました ([10] を参照のこと)．提供された修正は，Spice3f2 およびそれ以降のバージョンで実装されています．この修正によりパラメータのあてはめが影響される可能性があるため，古い実装を使うオプション BADMOS3 を設定することができます (シミュレーション変数と .OPTIONS 行の 4.1節を参照のこと)．Spice のレベル 1, 2, 3, 6 のパラメータは，

	名前	パラメータ	単位	省略時の値	例
1	LEVEL	モデルの番号	—	1	
2	VTO	ゼロバイアスしきい値電圧 (V_{TO})	V	0.0	1.0
3	KP	相互コンダクタンスパラメータ	A/V^2	2.0e-5	3.1e-5
4	GAMMA	バルクしきい値パラメータ	$V^{1/2}$	0.0	0.37
5	PHI	表面電位 (U)	V	0.6	0.65
6	LAMBDA	チャネル長調整 (MOS1 と MOS2 のみ) (L)	1/V	0.0	0.02
7	RD	ドレイン抵抗	Z	0.0	1.0
8	RS	ソース抵抗	Z	0.0	1.0
9	CBD	ゼロバイアス B-D 接合容量	F	0.0	20fF
10	CBS	ゼロバイアス B-S 接合容量	F	0.0	20fF
11	IS	バルク接合飽和電流 (I_S)	A	1.0e-14	1.0e-15
12	PB	バルク接合電位	V	0.8	0.87
13	CGSO	チャネル幅 1 m あたりのゲート-ソース重複容量	F/m	0.0	4.0e-11
14	CGDO	チャネル幅 1 m あたりのゲート-ドレイン重複容量	F/m	0.0	4.0e-11
15	CGBO	チャネル幅 1 m あたりのゲート-バルク重複容量	F/m	0.0	2.0e-10
16	RSH	ドレインとソースの拡散シート抵抗	Z/[]	0.0	10.0
17	CJ	接合領域 1 m^2 あたりのゼロバイアスバルク接合底部容量	F/m^2	0.0	2.0e-4
18	MJ	バルク接合底部勾配係数	—	0.5	0.5
19	CJSW	接合周囲 1 m あたりのゼロバイアスバルク接合側壁容量	F/m	0.0	1.0e-9
20	MJSW	バルク接合側壁勾配係数	—	0.50 (レベル 1) 0.33 (レベル 2, 3)	
21	JS	接合領域 1 m^2 あたりのバルク接合飽和電流	A/m^2		1.0e-8
22	TOX	酸化膜の厚さ	m	1.0e-7	1.0e-7
23	NSUB	サブストレートのドーピング	1/cm^3	0.0	4.0e15
24	NSS	表面準位密度	1/cm^2	0.0	1.0e10
25	NFS	高速表面準位密度	1/cm^2	0.0	1.0e10

	名前	パラメータ	単位	省略時の値	例
26	TPG	ゲートの材質 +1 サブストレートと逆 −1 サブストレートと同じ 0 Al ゲート	—	1.0	
27	XJ	冶金的な接合の深さ	*m*	0.0	1M
28	LD	横方向拡散	*m*	0.0	0.8M
29	UO	表面移動度	cm^2/Vs	600	700
30	UCRIT	移動度低下の臨界電界 (MOS2 のみ)	V/*cm*	1.0e4	1.0e4
31	UEXP	移動度低下の臨界電界係数 (MOS2 のみ)	—	0.0	0.1
32	UTRA	横方向電界係数 (移動度)(MOS2 にはない)	—	0.0	0.3
33	VMAX	キャリアの最大ドリフト速度	*m*/s	0.0	5.0e4
34	NEFF	総チャネル電荷 (固定および移動) 係数 (MOS2 のみ)	—	1.0	5.0
35	KF	フリッカノイズ係数	—	0.0	1.0e-26
36	AF	フリッカノイズ指数	—	1.0	1.2
37	FC	順方向バイアス空乏層容量式の係数	—	0.5	
38	DELTA	しきい値電圧の幅効果 (MOS2, MOS3)	—	0.0	1.0
39	THETA	移動度調整 (MOS3 のみ)	1/V	0.0	0.1
40	ETA	静電フィードバック (MOS3 のみ)	—	0.0	1.0
41	KAPPA	飽和電界係数 (MOS3 のみ)	—	0.2	0.5
42	TNOM	パラメータ測定温度	°C	27	50

レベル 4 およびレベル 5 (BSIM1, BSIM2) のパラメータは，プロセスの仕様からすべて得られ，自動的に生成されます．J. Pierret [4] は,「プロセス」のファイルを生成する手段を説明しています．Spice3 と共に提供されているプログラム Proc2Mod は，このファイルを変換して，Spice の入力ファイルに取り込むのに適した一連の BSIM1 の `.MODEL` 行にします．以下で l/w の列に * が付けられたパラメータには，長さと幅に依存する対応するパラメータが存在します．たとえば，`VFB` は単位が V の基本的なパラメータで，`LVFB` と `WVFB` というパラメータが存在し，その単位は V-*m* です．次式,

$$L_{effective} = L_{input} - DL$$

および

$$W_{effective} = W_{input} - DW$$

のように指定された実際の素子のパラメータを評価するために，次の式,

$$P = P_0 + \frac{P_L}{L_{effective}} + \frac{P_W}{W_{effective}}$$

が使われます．

Spice の他のモデルとは異なり，BSIM モデルはすべてのパラメータを提供するプロセス設計システムと共に使うように設計されているので，パラメータに省略時の値はなく，一つでも指定を忘れるとエラーとなります．パラメータの組の例とプロセスファイルのフォーマットに関しては，Spice2 実装ノート [3] を参照のこと．

BSIM2 に関しては，参考文献 [5] を参照のこと．

Spice BSIM (レベル 4) パラメータ

名前	パラメータ	単位	l/w
VFB	フラットバンド電圧	V	*
PHI	表面逆転電位	V	*
K1	本体効果係数	$V^{1/2}$	*
K2	ドレイン/ソース空乏層電荷共有係数	—	*
ETA	ゼロバイアスドレイン誘導バリア低下係数	—	*
MUZ	ゼロバイアス移動度	cm^2/V-s	
DL	チャネルの短化	mm	
DW	チャネルの狭化	mm	
U0	ゼロバイアス横方向電界移動勾配係数	V^{-1}	*
U1	ゼロバイアス速度飽和係数	mm/V	*
X2MZ	$V_{ds} = 0$ のときのサブストレートバイアスに対する移動度の感応度	cm^2/V^2-s	*
X2E	サブストレートバイアスに対するドレイン誘導バリア低下効果の感応度	V^{-1}	*
X3E	$V_{ds} = V_{dd}$ のときのドレインバイアスに対するドレイン誘導バリア低下効果の感応度	V^{-1}	*
X2U0	サブストレートバイアスに対する横方向電界移動度勾配効果の感応度	V^{-2}	*
X2U1	サブストレートバイアスに対する速度飽和効果の感応度	mmV^{-2}	*
MUS	ゼロサブストレートバイアスおよび $V_{ds} = V_{dd}$ のときの移動度	cm^2/V^2-s	
X2MS	$V_{ds} = V_{dd}$ のときのサブストレートバイアスに対する移動度の感応度	cm^2/V^2-s	*
X3MS	$V_{ds} = V_{dd}$ のときのドレインバイアスに対する移動度の感応度	cm^2/V^2-s	*
X3U1	$V_{ds} = V_{dd}$ のときのドレインバイアスに対する速度飽和の感応度	mmV^{-2}	*
TOX	ゲート酸化物の厚さ	mm	
TEMP	パラメータ測定時の温度	°C	
VDD	測定バイアスの範囲	V	
CGDO	チャネル幅 1 m あたりのゲート-ドレイン重複容量	F/m	
CGSO	チャネル幅 1 m あたりのゲート-ソース重複容量	F/m	
CGBO	チャネル長 1 m あたりのゲート-バルク重複容量	F/m	
XPART	ゲート酸化物容量電荷モデルフラグ	—	
N0	ゼロバイアスしきい値下傾斜係数	—	*
NB	サブストレートバイアスに対するしきい値下傾斜の感応度	—	*
ND	ドレインバイアスに対するしきい値下傾斜の感応	—	*
RSH	ドレインとソースの拡散シート抵抗	Z/[]	
JS	ソースドレイン接合電流密度	A/m^2	
PB	ソースドレイン接合の固有電位	V	
MJ	ソースドレイン接合の勾配係数	—	
PBSW	ソースドレイン接合側壁の固有電位	V	
MJSW	ソースドレイン接合側壁の勾配係数	—	
CJ	単位面積あたりのソースドレイン接合容量	F/m^2	
CJSW	単位長あたりのソースドレイン接合側壁容量	F/m	
WDF	ソースドレイン接合の省略時の幅	m	
DELL	ソースドレイン接合長の減少	m	

XPART=0 は，飽和時にドレイン/ソースの電荷を 40/60 に分割する指定で，XPART=1 は，ドレイン/ソースの電荷を 0/100 に分割する指定です．

3.4.9 MESFET

一般形:

ZXXXXXXX *ND NG NS MNAME* ⟨*AREA*⟩ ⟨OFF⟩ ⟨IC=*VDS*，*VGS*⟩

例:

```
Z1 7 2 3 ZM1 OFF
```

ND, *NG*, *NS* は，それぞれドレイン，ゲート，ソースのノードです．*MNAME* はモデルの名前，*AREA* は面積係数，OFF はこの素子に対して直流解析をする際の (オプションの) 初期条件です．面積係数が省略された場合，1.0 という値が仮定されます．IC=*VDS*，*VGS* という (オプションの) 初期条件の指定は，静止動作点とは異なる状態から過渡解析を始めるために，.TRAN 制御行の UIC オプションと共に使います．過渡解析の初期条件を設定するよりよい方法については，.IC 制御行の説明 (4.2.2節) を参照のこと．

3.4.10 MESFET モデル (NMF/PMF)

この MESFET モデルは，[11] で記述されている Statz et al. の GaAs FET モデルから導いたものです．直流特性は，ゲート電圧に対するドレイン電流の変化を決定するパラメータ VTO, B, BETA と，飽和電圧を決定する ALPHA と，出力コンダクタンスを決定する LAMBDA により定義されます．これは，次の式によって定義されます．

$$I_d = \begin{cases} \dfrac{B(V_{gs}-V_T)^2}{1+b(V_{gs}-V_T)}\left(1-\left(1-A\dfrac{V_{ds}}{3}\right)^3\right)(1+LV_{ds}) & \text{for } 0 < V_{ds} < \dfrac{3}{A} \\ \dfrac{B(V_{gs}-V_T)^2}{1+b(V_{gs}-V_T)}(1+LV_{ds}) & \text{for } V_{ds} > \dfrac{3}{A} \end{cases}$$

2 つの抵抗 RD と RS のパラメータもあります．電荷蓄積は，ゲート-ドレイン電圧およびゲート-ソース電圧の関数としての総ゲート容量によりモデル化されており，パラメータ CGS, CGD, PB により定義されます．

	名前	パラメータ	単位	省略時の値	例	面積
1	VTO	ピンチオフ電圧	V	−2.0	−2.0	
2	BETA	相互コンダクタンスパラメータ	A/V^2	1.0e-4	1.0e-3	*
3	B	ドーピング広がり拡張パラメータ	1/V	0.3	0.3	*
4	ALPHA	飽和電圧パラメータ	1/V	2	2	*
5	LAMBDA	チャネル長調整パラメータ	1/V	0	1.0e-4	
6	RD	ドレイン抵抗	Z	0	100	*
7	RS	ソース抵抗	Z	0	100	*
8	CGS	ゼロバイアス G-S 接合容量	F	0	5pF	*
9	CGD	ゼロバイアス G-D 接合容量	F	0	1pF	*
10	PB	ゲート接合電位	V	1	0.6	
11	KF	フリッカ雑音係数	—	0		
12	AF	フリッカ雑音指数	—	1		
13	FC	順バイアス空乏層容量式の係数	—	0.5		

4 解析と出力の制御

以下のコマンド行は，回路記述ファイルの中で解析やグラフ作成を指示するためのものです．会話型のコマンド解釈系 (第 5節で詳しく説明されている) にも同様のコマンドがあります．入力ファイルで解析やグラフ (あるいは表) の作成を指示する方法は，バッチ実行で役立ちます．`-b` オプションが指定されるか，省略時の入力がファイルからリダイレクトされている場合に，バッチモードに入ります．バッチモードでは，入力ファイルの制御行により指定された解析 (たとえば "`.ac`", "`.tran`" など) が，("`.control`" 行が存在する場合を除いて; 会話型コマンド解釈系の節を参照のこと) 即座に実行されます．`-r`「生ファイル」オプションが指定された場合，すべての生成されたデータは Spice3 生ファイルに書き出されます．生ファイルは Spice3 の会話型モードまたは Nutmeg によって読み込まれます．詳しくは，第 5節を参照のこと．この場合，内部の素子の変数の値を記録するために `.SAVE` 行 (4.4.1を参照) を使います (付録 B を参照のこと).

生ファイルが指定されていない場合，次に説明する `.PRINT`, `.PLOT`, `.FOUR` 制御行によって，グラフ (ラインプリンタ形式) と表を表示します．`.PRINT`, `.PLOT`, `.FOUR` 行は，Spice2 との互換性をとるためのものです.

4.1 シミュレータの変数 (.OPTIONS)

Spice3 にはシミュレーション用のさまざまな変数があり，精度や速度を制御したり，ある素子の省略時の値を制御するために変更できます．これらのパラメータは，"`set`" コマンド (5.3.39節で説明します) または "`.OPTIONS`" 行により変更できます.

一般形:

`.OPTIONS` *OPT1 OPT2* `...` (または `OPT=`*OPTVAL* `...`)

例:

```
.OPTIONS RELTOL=.005 TRTOL=8
```

`.OPTIONS` 行を使うと，プログラムの制御を初期状態に戻し，特定のシミュレーションの目的のためのオプションを設定できます．Nutmeg 用の追加のオプションも指定でき，そのオプションは Nutmeg が入力ファイルを読んだときに効果を発揮します．"`set`" コマンドによって指定された Nutmeg 用のオプションは，あたかも `.OPTIONS` 行で指定されたかのように Spice3 にも渡されます．`.OPTIONS` 行によって設定できるパラメータと '`set`' コマンドの書式については，会話型のコマンド解釈系に関する 5.3.39節を参照のこと．以下のオプションは，任意の順序で任意に組み合わせることができます．(以下の) '*x*' は，ある正の数を表わします.

オプション	効果
`ABSTOL=`*x*	電流の絶対誤差の許容範囲を再設定します．省略時の値は 1 pA.
`BADMOS3`	MOS3 モデルで "`kappa`" の不連続性がある古いバージョンを使います.
`CHGTOL=`*x*	電荷の許容範囲を再設定します．省略時の値は `1.0e-14`.
`DEFAD=`*x*	MOS ドレイン拡散面積の値を再設定します．省略時の値は 0.0.
`DEFAS=`*x*	MOS ソース拡散面積の値を再設定します．省略時の値は 0.0.
`DEFL=`*x*	MOS チャネル長の値を再設定します．省略時の値は 100.0 μm.

オプション	効果
DEFW=*x*	MOS チャネル幅の値を再設定します．省略時の値は 100.0μm.
GMIN=*x*	許容できる最小のコンダクタンス GMIN の値を再設定します．省略時の値は 1.0e-12.
ITL1=*x*	直流動作点の繰り返し回数の制限を再設定します．省略時の値は 100.
ITL2=*x*	直流伝達関数の繰り返し回数の制限を再設定します．省略時の値は 50.
ITL3=*x*	過渡解析の繰り返し回数の下限を再設定します．省略時の値は 4．(注意：Spice3 では実装されていません).
ITL4=*x*	過渡解析の各時点の繰り返し回数の制限を再設定します．省略時の値は 10.
ITL5=*x*	過渡解析の総繰り返し回数の制限を再設定します．省略時の値は 5000. ITL5=0 と設定すると，このチェックを省きます．(注意：Spice3 では実装されていません).
KEEPOPINFO	交流，歪み，極-ゼロ解析のいずれかを行なったときに，動作点の情報を保持します．これは回路が大きな場合に (冗長な) ".OP" 解析を行ないたくない場合に特に有用です.
METHOD=name	Spice が使用する数値積分手法を設定します．使用できる名前は，"Gear" または "trapezoidal" (または単に "trap") です．省略時の値は trapezoidal.
PIVREL=*x*	最大の列の値と受け入れ可能なピボットの値の相対比率を再設定します．省略時の値は 1.0e-3．数値計算のピボット選択アルゴリズムでは，受け入れ可能な最小のピボット値が EPSREL=AMAX1(PIVREL*MAXVAL, PIVTOL) によって決定されます．ここで MAXVAL はピボットの対象となる列の最大の要素です (部分ピボッティング).
PIVTOL=*x*	ピボットになることができる行列の要素の絶対最小値を再設定します．省略時の値は 1.0e-13.
RELTOL=*x*	相対誤差の許容値を再設定します．省略時の値は 0.001 (0.1%).
TEMP=*x*	回路の動作温度を再設定します．省略時の値は摂氏 27 度 (300 ケルビン). TEMP は，温度に依存する任意の素子で指定した値が優先されます.
TNOM=*x*	素子のパラメータを測定した名目温度を再設定します．省略時の値は摂氏 27 度 (300 ケルビン)．TNOM は，温度に依存する任意の素子で指定した値が優先されます.
TRTOL=*x*	過渡解析の誤差の許容値を再設定します．省略時の値は 7.0．このパラメータは Spice が実際の打ち切り誤差を大きく見積る係数の推定値です.
TRYTOCOMPACT	LTRA モデルのみに適用されます．指定されると，シミュレータは LTRA 伝送線路の入力電圧と電流の過去のヒストリを圧縮しようとします.
VNTOL=*x*	絶対電圧誤差の許容値を再設定します．省略時の値は 1μV.

さらに，Spice2 エミュレーションモードでは，次のオプションが働きます.

オプション	効果
ACCT	課金情報と計算時間の統計を表示します.
LIST	入力データの要約を表示します.
NOMOD	モデルパラメータの表示を抑止します.
NOPAGE	改ページを抑止します.
NODE	ノードの表を表示します.
OPTS	オプションの値を表示します.

4.2 初期条件

4.2.1 .NODESET: 初期ノード電圧の推測値を指定する

一般形:

.NODESET V(*NODNUM*)=*VAL* V(*NODNUM*)=*VAL* ...

例:

```
.NODESET V(12)=4.5 V(4)=2.23
```

.NODESET 行は，指定したノードを指定した電圧に保つという予備的な試みを行なうことにより，直流解析や過渡解析の初期解を見つけやすくします．その後この制約は緩和され，本当の解に向かって繰り返しを続けます．双安定または不安定な回路で収束するために .NODESET 行が必要になることがあります．一般には，この行が必要とされることはありません.

4.2.2 .IC: 初期条件を設定する

一般形:

.IC V(*NODNUM*)=*VAL* V(*NODNUM*)=*VAL* ...

例:

```
.IC V(11)=5 V(4)=-5 V(2)=2.2
```

.IC 行は，過渡解析の初期条件を設定します．.TRAN 制御行に UIC パラメータが指定されているかどうかによって，2 通りに解釈されます．また，この行を .NODESET 行と混同してはいけません．.NODESET 行は直流の収束を助けるだけで，(多安定の回路を除いては) 最終的なバイアスの解には影響しません．2 つの解釈は，以下の通りです.

1. .TRAN 行で UIC パラメータが指定されている場合，.IC 制御行で指定されたノードの電圧は，コンデンサ，ダイオード，BJT, JFET, MOSFET の初期条件を計算するために使われます．これは，各素子の行に IC=... パラメータを指定したのと等価ですが，それよりはかなり便利です．IC=... パラメータを併用してもよく，.IC の値よりも優先されます．過渡解析を行なう前に直流バイアスの解 (過渡解析の初期解) を求めないので，素子の初期状態を計算するのに使われる直流源の電圧を .IC 制御行ですべて指定するよう注意を払ってください.

2. .TRAN 制御行で UIC パラメータが指定されていない場合，過渡解析を行なう前に直流バイアスの解 (過渡解析の初期解) を求めます．この場合，.IC 制御行で指定されたノードの電圧は，バイアスの解を求める間，指定した初期値に保たれます．過渡解析の間は，ノード電圧に関するこの制約はなくなります．これは，Spice によって一貫した直流解が計算されるので，より望ましい方法です．

4.3 解析

4.3.1 .AC: 小信号交流解析

一般形:

.AC DEC *ND FSTART FSTOP*
.AC OCT *NO FSTART FSTOP*
.AC LIN *NP FSTART FSTOP*

例:

```
.AC DEC 10 1 10K
.AC DEC 10 1K 100MEG
.AC LIN 100 1 100HZ
```

DEC は 10 倍の変化を表わし，*ND* は 10 倍あたりの点の数です．OCT はオクターブの変化を表わし，*NO* はオクターブあたりの点の数です．LIN はリニアな変化を表わし，*NP* は点の数です．*FSTART* は開始周波数，*FSTOP* は最終周波数です．この行が入力ファイルに存在すると，Spice は指定された範囲について回路の交流解析を行ないます．この解析に意味があるようにするためには，少なくとも一つの独立電源に AC 値を指定する必要があります．

4.3.2 .DC: 直流伝達関数

一般形:

.DC *SRCNAM VSTART VSTOP VINCR* [*SRC2 START2 STOP2 INCR2*]

例:

```
.DC VIN 0.25 5.0 0.25
.DC VDS 0 10 .5 VGS 0 5 1
.DC VCE 0 10 .25 IB 0 10U 1U
```

DC 行は，直流伝達特性の電源と，その掃引の範囲を定義します (コンデンサは開放され，インダクタは短絡されます)．*SRCNAM* は独立電圧源または電流源の名前です．*VSTART*, *VSTOP*, *VINCR* は，それぞれ開始値，最終値，増分です．最初の例では，電圧源 VIN の値が，0.25 V から 5 V まで，0.25 V きざみで増えていきます．オプションで 2 番目の電源 (SRC2) を掃引パラメータに指定できます．この場合，第 2 の電源のそれぞれの値について，最初の電源が掃引されます．このオプションは，半導体素子の出力特性を得るのに役立ちます．付録 Aの 2 番目の回路例の説明を参照のこと．

4.3.3 .DISTO: 歪み解析

一般形:

```
.DISTO DEC ND FSTART FSTOP ⟨F2OVERF1⟩
.DISTO OCT NO FSTART FSTOP ⟨F2OVERF1⟩
.DISTO LIN NP FSTART FSTOP ⟨F2OVERF1⟩
```

例:

```
.DISTO DEC 10 1kHz 100Mhz
.DISTO DEC 10 1kHz 100Mhz 0.9
```

.DIST 行は，回路の小信号歪み解析を行ないます．多次元のテイラー級数を使って動作点の非直線性を表わして，多次元の Volterra 級数解析を行ないます．級数展開では 3 次までの項を使います．

オプションのパラメータ *F2OVERF1* が指定されていない場合，.DISTO は高調波解析を行ないます．すなわち，単一の入力周波数 F_1 に関する回路の歪みを解析します．入力周波数は，.AC コマンドとちょうど同じように，.DISTO コマンドの引数で指定された範囲で掃引されます．この周波数の入力は，1 つ以上の入力電源に現われてもよく，振幅と位相は入力となる電源の行に DISTOF1 キーワードの引数として指定します (独立電源の説明 (3.2.1節) を参照のこと)．(DISTOF2 キーワードの引数は，この場合関係ありません)．この解析によって，入力周波数 F_1 を掃引した場合の，すべてのノード電圧と枝電流の高調波 $2F_1, 3F_1$ の交流値の情報が生成されます．(1 という値は，入力の基本波では $\cos(2\pi F_1 t)$ と等価であるという慣習により，(複素数の歪み出力としての) 1 は，$2F_1$ においては $\cos(2\pi(2F_1)t)$ を意味し，$3F_1$ においては $\cos(2\pi(3F_1)t)$ を意味します．) Nutmeg のコマンドで歪みの成分 ($2F_1$ または $3F_1$) を選択して，表示やグラフの作成ができます．(通常，おもに高調波成分の振幅に関心があるので，交流歪み値の振幅を調べることになるでしょう)．これらの値は，実際の高調波成分の交流値であり，HD_2 や HD_3 とは異なります．HD_2 や HD_3 を求めるには，.AC 行から得られる対応する F_1 の交流値で割る必要があります．この除算は Nutmeg のコマンドでできます．

オプションの *F2OVERF1* パラメータを指定する場合は，0.0 を超え 1.0 未満の実数を指定します．この場合，.DISTO はスペクトル解析を行ないます．2 つの異なった周波数 F_1 と F_2 の正弦波を回路に入力したとみなします．F_1 は .DISTO 制御行にしたがって .AC 制御行と同じように掃引されます．F_1 が掃引される間，F_2 は $F2OVERF1 \times FSTART$ という単一の周波数に固定されます．回路の中の各独立電源は，F_1, F_2 という 2 種類の (重ね合わされた) 正弦波による歪み解析用の入力となります．F_1 成分の振幅と位相は，電源の入力行の DISTOF1 キーワードの引数で指定します (独立電源の説明 (3.2.1節) を参照のこと)．F_2 成分の振幅と位相は，電源の入力行の DISTOF2 キーワードの引数で指定します．この解析によって，入力周波数 F_1 を掃引した場合の，すべてのノード電圧と枝電流の混変調出力周波数 $F_1 + F_2, F_1 - F_2, 2F_1 - F_2$ の交流値の情報が描かれます．関心のある混変調出力を setplot コマンドを使って選択し，print および plot コマンドを使って表示することができます．高調波解析の場合と同様に，結果は混変調周波数の実際の交流電圧や電流であり，IM パラメータを得るには，.AC の値で標準化する必要があります．

独立電源の記述に DISTOF1 または DISTOF2 キーワードがない場合，その電源は対応する周波数の入力がないと仮定されます．振幅と位相の省略時の値は，それぞれ 1.0 と 0.0 です．位相は，度で指定します．

正確な結果を得るために，理想的には *F2OVERF1* を無理数とすべきですが，実際には不可能なので，その値を分数で表わしたときの分母ができるだけ大きくなるよう (最低 3 以上) にしてくだ

さい (すなわち，*F2OVERF1* を分数 A/B で表わしたとき，A と B に共通の因数がなく，B ができるだけ大きくなるようにしてください．*F2OVERF1* は 1 未満なので，$A < B$ となります．)．なぜそうすべきかを説明するために，*F2OVERF1* が 49/100 の場合と 1/2 の場合を考えてみます．スペクトル解析では，出力される周波数は $F_1 + F_2, F_1 - F_2, 2F_1 - F_2$ です．後者の場合，$F_1 - F_2 = F_2$ となり，同じ周波数に強い基本波の F_2 成分があるため，$F_1 - F_2$ 成分は誤りとなってしまいます．同様に，後者の場合，$F_1 + F_2 = 2F_1 - F_2$ であり，それぞれの結果も誤りとなってしまいます．*F2OVERF1* = 49/100 の場合，$F_1 - F_2 = 51/100F_1 \neq 49/100F_1 = F_2$ なので，この問題は生じません．この場合，2 つの周波数成分は，F_2 と $F_1 - F_2$ という非常に近いところにあります．Volterra 級数を使う利点のひとつは，たとえば過渡解析の場合とは異なり，記号で表わされた混合周波数 (すなわち $nF_1 + mF_2$) における歪みを計算するので，歪みの成分がたとえ非常に近くにあっても，歪みの大きさを正確に求めることができることです．欠点は，もちろん 2 つの混合周波数が一致した場合に，結果が結合されて表示されないことです (これはおそらく後処理によって可能です)．現状では，関心のある利用者は，混合周波数の値を自分で調べて，必要なら混合周波数が一致した歪みを加算してください．

4.3.4 .NOISE: ノイズ解析

一般形:

.NOISE V(*OUTPUT* ⟨,*REF*⟩) *SRC* (DEC|LIN|OCT) *PTS FSTART FSTOP*
+ ⟨*PTS_PER_SUMMARY*⟩

例:

```
.NOISE V(5) VIN DEC 10 1kHZ 100Mhz
.NOISE V(5,3) V1 OCT 8 1.0 1.0e6 1
```

.NOISE 行は，回路のノイズ解析を行ないます．*OUTPUT* は総出力ノイズを求めるノードです．*REF* が指定されている場合，ノイズの電圧は V(*OUTPUT*) − V(*REF*) となります．省略時には，*REF* はグラウンドであると仮定されます．*SRC* は，入力ノイズを参照する独立電源の名前です．*PTS*, *FSTART*, *FSTOP* は，.AC と同様なパラメータで，グラフを描く周波数の範囲を指定します．*PTS_PER_SUMMARY* はオプションの整数で，指定された場合，各ノイズ生成素子の寄与の情報が，*PTS_PER_SUMMARY* 周波数点ごとに生成されます．

.NOISE 制御行は 2 つのグラフを生成します．一つはノイズスペクトル密度曲線で，もう一つは指定した周波数範囲の総積分ノイズです．すべてのノイズ電圧/電流は 2 乗の単位です (スペクトル密度では V^2/Hz および A^2/Hz，積分ノイズでは V^2 および A^2)．

4.3.5 .OP: 動作点解析

一般形:

.OP

入力ファイルにこの行を含めると，インダクタを短絡し，コンデンサを開放して，回路の直流動作点を決定するよう Spice に指示します．直流解析は，過渡解析の初期条件を決定するために過渡解析の前に，また，非線形素子の線形化小信号モデルを決定するために，交流小信号解析，ノイズ解析，極-ゼロ解析の前に，自動的に行なわれます (前述の KEEPOPINFO 変数 (5.5節) を参照のこと)．

4.3.6 .PZ: 極-ゼロ解析

一般形:

.PZ *NODE1 NODE2 NODE3 NODE4* CUR POL
.PZ *NODE1 NODE2 NODE3 NODE4* CUR ZER
.PZ *NODE1 NODE2 NODE3 NODE4* CUR PZ
.PZ *NODE1 NODE2 NODE3 NODE4* VOL POL
.PZ *NODE1 NODE2 NODE3 NODE4* VOL ZER
.PZ *NODE1 NODE2 NODE3 NODE4* VOL PZ

例:

```
.PZ 1 0 3 0 CUR POL
.PZ 2 3 5 0 VOL ZER
.PZ 4 1 4 1 CUR PZ
```

CUR は (出力電圧)/(入力電流) 型の伝達関数を表わし，VOL は (出力電圧)/(入力電圧) 型の伝達関数を表わします．POL は極解析のみを表わし，ZER はゼロ解析のみを表わし，PZ は極-ゼロ解析を表わします．この機能は，主に極またはゼロを見つける計算が収束しない場合に，少なくとも一方を出力するために用意されています．*NODE1* および *NODE2* は，2 つの入力ノードで，*NODE3* および *NODE4* は，2 つの出力ノードです．したがって，出力と入力のポートと伝達関数の種類に関して完全な自由があります．

会話モードでは，コマンドの構文は同じですが，最初のフィールドは .PZ ではなく PZ です．結果を表示するには，コマンド 'print all' を使います．

4.3.7 .SENS: 直流または小信号交流感応度解析

一般形:

.SENS *OUTVAR*
.SENS *OUTVAR* AC DEC *ND FSTART FSTOP*
.SENS *OUTVAR* AC OCT *NO FSTART FSTOP*
.SENS *OUTVAR* AC LIN *NP FSTART FSTOP*

例:

```
.SENS V(1,OUT)
.SENS V(OUT) AC DEC 10 100 100k
.SENS I(VTEST)
```

.SENS 解析を指定すると，0 はでないすべての素子パラメータに対する *OUTVAR* の感応度が計算されます．*OUTVAR* は回路の変数 (ノードの電圧または電圧源の枝電流) です．最初の形式は，*OUTVAR* の直流動作点の値の感応度を計算します．2 番目の形式は，*OUTVAR* の交流値の感応度を計算します．交流感応度で指定するパラメータは，交流解析のパラメータと同じです (上述の .AC を参照のこと)．出力される値の次元は，(入力 1 パーセントの変化あたりの出力の変化率ではなく) 入力 1 単位の変化あたりの出力の変化です．

4.3.8 .TF: 伝達関数解析

一般形:

.TF *OUTVAR INSRC*

例:

```
.TF V(5, 3) VIN
.TF I(VLOAD) VIN
```

TF 行は，直流小信号解析の小信号出力と入力を定義します．*OUTVAR* は小信号出力変数で，*INSRC* は小信号入力電源です．この行があると，Spice は伝達関数 (出力/入力) の直流小信号値，入力抵抗，出力抵抗を計算します．最初の例では，Spice は VIN に対する V(5, 3) の比と，VIN の小信号入力抵抗と，ノード 5 と 3 の間で測った小信号出力抵抗を計算します．

4.3.9 .TRAN: 過渡解析

一般形:

.TRAN *TSTEP TSTOP* ⟨*TSTART* ⟨*TMAX*⟩⟩

例:

```
.TRAN 1NS 100NS
.TRAN 1NS 1000NS 500NS
.TRAN 10NS 1US
```

TSTEP は，表示やグラフの増分です．*TSTEP* は，後処理系と共に使うために提案された計算間隔として扱われます．*TSTOP* は最終の時刻で，*TSTART* は最初の時刻です．*TSTART* が省略されている場合，0 と仮定されます．過渡解析は，常に時刻 0 から始まります．区間 ⟨0, *TSTART*⟩ では，(安定状態に達するまで) 回路は解析されますが，出力は保存されません．区間 ⟨*TSTART*, *TSTOP*⟩ では，回路が解析され，出力が保存されます．*TMAX* は，Spice が用いる最大のステップの大きさです．省略時の値として，プログラムは *TSTEP* または (*TSTOP* – *TSTART*)/50.0 のいずれか小さいほうを選びます．*TMAX* は，計算の間隔を表示の増分 *TSTEP* よりも小さくしたい場合に有用です．

UIC (初期条件を使う) は，オプションのキーワードで，過渡解析に先立って Spice が安定動作点の解を求めないように指示するものです．このキーワードが指定された場合，Spice はさまざまな要素の IC=... によって指定された値を過渡解析の初期条件として使い，解析を進めます．.IC 制御行が指定された場合，その素子の初期条件を計算するために .IC 行のノードの電圧が使われます．UIC が指定されていない場合の .IC 制御行の解釈については，.IC の説明 (4.2.2節) を参照のこと．

4.4 バッチ出力

4.4.1 .SAVE 行

一般形:

`.SAVE` ベクトル ベクトル ベクトル ...

例:

```
.SAVE i(vin) input output
.SAVE @m1[id]
```

後に Spice3 または Nutmeg で使えるように，`.SAVE` 行でリストされたベクトルを生ファイルに記録します (Nutmeg はシミュレート機能のない Spice3 のデータ解析部です)．標準的な書式でベクトルの名前を指定できます．`.SAVE` 行が指定されていない場合，標準のベクトルの組 (ノードの電圧，電圧源の枝電流) が保存されます．`.SAVE` 行が指定されている場合，指定されたベクトルのみが保存されます．素子の内部データに関しては，付録 B を参照のこと．生ファイルの使い方については，会話型コマンド解釈系の 5節も参照のこと．

4.4.2 .PRINT 行

一般形:

`.PRINT` *PRTYPE OV1* ⟨*OV2 ... OV8*⟩

例:

```
.PRINT TRAN V(4) I(VIN)
.PRINT DC V(2) I(VSRC) V(23, 17)
.PRINT AC VM(4, 2) VR(7) VP(8, 3)
```

`.PRINT` 行は，1 つから 8 つの出力変数からなる表の内容を定義します．*PRTYPE* は，出力させたい分析の種類 (`DC`, `AC`, `TRAN`, `NOISE`, `DISTO`) です．電圧や電流の出力変数の書式は，`print` コマンドの 5.3.28節で説明されているものと同じです．Spice2 は，出力変数を次の書式に制限しています (この制限は Spice3 にはありません).

`V(`*N1* ⟨`,`*N2*⟩`)` ノード *N1* と *N2* の電位差を指定します．*N2* (とその前のカンマ) が指定されていない場合，グラウンド (`0`) が仮定されます．詳しくは `print` コマンドの 5.3.28節を参照のこと．Spice2 との互換性のため，交流解析については，`V(`*N1*`,`*N2*`)` の "`V`" を次の文字に置き換えることによって，次の 5 つの値にアクセスできます．

`VR`	実部
`VI`	虚部
`VM`	振幅
`VP`	位相
`VDB`	$20\log_{10}$(振幅)

`I(V`*XXXXXXX*`)` 独立電圧源 `V`*XXXXXXX* を流れている電流を指定します．正の電流の向きは，正のノードから電圧源を通り負のノードへと流れる方向です．交流解析については，電圧の出力の場合で説明したように，文字 `I` を対応する文字で置き換えることができます．

ノイズ解析と歪み解析の出力変数の一般形は，他の解析の形式とは異なります．
各解析種別について，`.PRINT` 行の数に制限はありません．

4.4.3 .PLOT 行

一般形:

.PLOT *PLTYPE OV1* ⟨(*PLO1*, *PHI1*)⟩ ⟨*OV2* ⟨(*PLO2*, *PHI2*)⟩ ... *OV8*⟩

例:

```
.PLOT DC V(4) V(5) V(1)
.PLOT TRAN V(17, 5) (2, 5) I(VIN) V(17) (1, 9)
.PLOT AC VM(5) VM(31, 24) VDB(5) VP(5)
.PLOT DISTO HD2 HD3(R) SIM2
.PLOT TRAN V(5, 3) V(4) (0, 5) V(7) (0, 10)
```

.PLOT 行は，1 つから 8 つの出力変数からなる 1 つのグラフの内容を定義します．*PLTYPE* は，出力させたい分析の種類 (DC, AC, TRAN, NOISE, DISTO) です．*OVI* の構文は，.PRINT 行や会話モードの plot コマンドと同じです．

2 つ以上の軌跡が重なったところは，文字 'X' で示されます．

1 つのグラフに 2 つ以上の変数が現われる場合，最初に指定された変数はグラフに描かれるだけでなく，表でも出力されます．すべての変数の表出力が必要な場合は，.PRINT 行も指定してください．

各解析種別について，.PLOT 行の数に制限はありません．

4.4.4 .FOUR: 過渡解析の出力のフーリエ解析

一般形:

.FOUR *FREQ OV1* ⟨*OV2 OV3* ...⟩

例:

```
.FOUR 100K  V(5)
```

.FOUR (フーリエ) 行は，過渡解析の一部としてフーリエ解析を行なうよう指示します．*FREQ* は基本波の周波数，*OV1*, *OV2*, *OV3* は解析する出力変数です．フーリエ解析は，区間 ⟨TSTOP−*period*, TSTOP⟩ に対して行なわれます．ここで，TSTOP は過渡解析で指定された最終時刻で，*period* は基本波の周波数の 1 周期です．直流成分と最初の 9 個の高調波が得られます．最大の精度を得るには，TMAX (.TRAN 行 (4.3.9節) を参照のこと) を *period*/100.0 (Q が高い回路ではそれ以下) に設定します．

5 会話型の解釈系

Spice3 は，シミュレータと，データ解析およびグラフ作成用のフロントエンドによって構成されています．フロントエンドは，Nutmeg という名前の分離した「スタンドアロン」プログラムとして実行できます．

Nutmeg は，`spice -r` または会話型の Spice3 セッションの `write` コマンドによって生成された「生」データファイルを読み込みます．Nutmeg または会話型の Spice3 では，シミュレーションのデータをグラフにしてグラフィック端末やワークステーションのディスプレイに表示することができます．会話型の Spice3 フロントエンドで使えるほとんどのコマンドは，Nutmeg でも使えます．そうではない Spice のみに有効なコマンドは，アスタリスク (“*”) を付けてあります．生ファイルは，Spice2 が標準出力に書き出すデータとは違います．それと同じデータは，Spice3 に `-b` コマンド行オプションを指定すると生成されます．

Spice と Nutmeg は，環境変数 `DISPLAY` を見つけると，グラフ表示に X Window System を使います．そうでなければ，グラフィック端末独立インターフェイス (MFB) が使われます．ワークステーションで X を使っている場合，`DISPLAY` 変数を常に設定すべきです．Spice3 や Nutmeg を実行しているシステム以外にグラフィクスを表示したい場合，`DISPLAY` は *machine*`:0.0` という形式にします．詳細については，X Window System の適切なドキュメントを参照のこと．

コマンドの形式

`spice` [`-n`] [`-t` *term*] [`-r` *rawfile*] [`-b`] [`-i`] [*inputfile* `...`]
`nutmeg` [`-`] [`-n`] [`-t` *term*] [*datafile* `...`]

オプションは，

- ファイルが指定されていない場合に省略時のデータファイル (“`rawspice.raw`”) を読み込まない．Nutmeg のみ．

`-n` (**または** `-N`) 起動時に “`.spiceinit`” ファイルを取り込まない．通常，Spice および Nutmeg は，このファイルをカレントディレクトリで探し，見つからなかったら利用者のホームディレクトリで探します．

`-t` *term* (**または** `-T` *term*) MFB 名が *term* の端末で動作します．

`-b` (**または** `-B`) バッチモードで動作します．Spice3 は標準の入力ファイル (すなわちキーボード) を読むか，指定された入力ファイルを読み，指定された解析を行ないます．出力は，Spice2 のようなラインプリンタによるプロット (ascii プロット) か，Spice の生ファイルです．詳細については，第 4節および第 4.4節を参照のこと．入力ソースファイルが端末ではない (すなわち IO リダイレクション “<” が使われている) 場合，Spice3 は自動的にバッチモードになります (`-i` で会話型にできます)．このオプションは Spice3 のみ有効です．

`-s` (**または** `-S`) サーバーモードで動作します．これはバッチモードに似ていますが，一時的な生ファイルが使われ，シミュレーションが終わると 1 個の “`@`” のみの行に続いて標準出力に書き出されます．このモードは Spice デーモンによって使われます．このオプションは Spice3 のみ有効です．

`-i` (**または** `-I`) 会話型モードで動作します．これは，標準入力が端末ではないが会話型モードが望ましいときに有用です．標準入力が端末ではない場合，コマンド補完は利用できません．このオプションは Spice3 のみ有効です．

`-r` *rawfile* (**または** `-P` *rawfile*) シミュレーションの結果を格納するファイルとして *rawfile* を使います．このオプションは Spice3 のみ有効です．

これ以降の Spice への引数は，Spice3 の入力ファイルであると解釈され，そのファイルが読み込まれ，(メモリ中に) 保存されます (バッチモードで動作している場合は，即座に実行されます)．Spice3 は，ほとんどの Spice2 の入力ファイルを受け付け，`.plot`, `.four`, `.print` 行で指定された ascii プロットを出力し，フーリエ解析を行ない，ノードの表示を行ないます．`.width` 行で `out` パラメータが指定された場合，その効果は `set width = ...` と同じです．しかし，Spice3 の ascii プロットは複数の範囲を扱えないので，`.plot` 行のベクトルの範囲が異なっていると，Spice2 で得られるよりも情報がかなり減ってしまいます．また，Spice3 の出力は，上述の行で要求されたデータのみ表示されるという点で，Spice2 ほど冗舌ではありません．

Nutmeg では，これ以降の引数は，Nutmeg に読み込まれるバイナリまたはテキストフォーマット (**convert**(**1**) を参照のこと) のデータファイルとして解釈されます．ファイルがバイナリ形式の場合，一部のみが完了していることもあります (シミュレーションが終了する前に Spice2 の出力を確認する際に有用です)．1 つのファイルに，さまざまな解析のデータセットが任意の数含まれていても構いません．

5.1 式，関数，定数

Spice および Nutmeg の時刻，電圧などのデータの形式は，ベクトルです．各ベクトルには型があり，ベクトルの型に応じた演算を行なったり，代数的に結合したりできます．ベクトルは通常データファイルを読み込んだとき (後述の `load` コマンド (5.3.25節) を参照のこと) に作成されます．`let` コマンドでも作成できます．

式は，ベクトルとスカラー (スカラーは長さが 1 のベクトル) と次の演算からなる代数式です．

```
+   -   *   /   ^   %
```

`%` は剰余演算子で，カンマ演算子には 2 つの意味があります：利用者が定義する関数の引数リストに現われた場合には，引数を区切る働きをします．それ以外は，項 `x , y` は $x + j(y)$ を意味します．

論理演算子 `&` (論理積), `|` (論理和), `!` (否定) と，比較演算子 <, >, >=, <=, =, <> (等しくない) もあります．代数式で使用した場合，C と同じように働いて，0 または 1 という値を生成します．比較演算子には次の別名があります．

```
gt    >
lt    <
ge    >=
le    <=
ne    <>
eq    =
and   &
or    |
not   !
```

これらは，< や > が IO リダイレクションと混同される (これはほとんど常にそうですが) ときに有用です.

次の関数を使えます.

`mag(`*vector*`)`	ベクトルの振幅
`ph(`*vector*`)`	ベクトルの位相
`j(`*vector*`)`	ベクトルを i ($\sqrt{-1}$) 倍する
`real(`*vector*`)`	ベクトルの実部
`imag(`*vector*`)`	ベクトルの虚部
`db(`*vector*`)`	$20\log_{10}$(`mag(`*vector*`)`)
`log(`*vector*`)`	ベクトルの常用対数 (底が 10)
`ln(`*vector*`)`	ベクトルの自然対数 (底が e)
`exp(`*vector*`)`	ベクトルの指数
`abs(`*vector*`)`	ベクトルの絶対値
`sqrt(`*vector*`)`	ベクトルの平方根
`sin(`*vector*`)`	ベクトルの正弦
`cos(`*vector*`)`	ベクトルの余弦
`tan(`*vector*`)`	ベクトルの正接
`atan(`*vector*`)`	ベクトルの逆正接
`norm(`*vector*`)`	ベクトルを 1 に正規化したもの (すなわち最大の要素の大きさを 1 とする)
`rnd(`*vector*`)`	各要素が，0 から各要素の絶対値の間の整数乱数であるベクトル
`mean(`*vector*`)`	ベクトルの要素の平均値からなるスカラー (長さが 1 のベクトル)
`vector(`*number*`)`	要素が $0, 1, \ldots,$ *number* -1 の長さが *number* のベクトル．*number* がベクトルの場合，最初の要素が使われ，整数でない場合は大きさの切り捨てが使われます.
`length(`*vector*`)`	ベクトルの長さ
`interpolate(`*plot*`.`*vector*`)`	現在のグラフのスケールに合わせて，指定されたベクトルを補間したもの．この関数は，`polydegree` により多項式補間の次数を決定します.
`deriv(`*vector*`)`	与えられたベクトルの微分値を計算します．多項式により補間した値を数値微分したもので，満足の行く結果が得られないことがあります (特に微分を繰り返した場合)．この実装では，ベクトルのスケールの実部に対してのみ微分を計算します.

vector は，すでに定義されたベクトルの名前または浮動小数点数 (スカラー) です．*number* は，`14.6Meg`, `-1.231e-4` などの Spice が受け付ける任意の書式で指定します．科学記法または `MEG`, `G` などの省略記法のいずれも使えますが，両方を指定してはいけません．Spice と同様に，数値の後ろにアルファベットを続けても構いません.

expr`[`*num*`]` という記法は，*expr* の *num* 番目の要素を意味します．多次元のベクトルでは，1 次元減ったベクトルが返されます．多次元のベクトルでは，*expr*`[`*m*`][`*n*`]` という記法は，*m* 番目のサ

ブベクトルの n 番目の要素を返します．ベクトルのある範囲を取り出すには，*expr*[*lower*，*upper*] という形式を使います．

「現在のプロット」(後述の `setplot` コマンド (5.3.41節) を参照) ではないプロットの中のベクトルを参照するには，*plotname*.*vecname* という記法を使います．

プロット名またはベクトル名として，ワイルドカード `all` を使えます．プロット名が `all` の場合，すべてのプロットの一致するベクトルが指定され，ベクトル名が `all` の場合，指定されたプロットのすべてのベクトルが参照されます．ワイルドカードを含む式に 2 項の演算を使うことはできません．例えば，`all + all` が何を表わすのかは明らかでないからです．式の例は，

```
cos(TIME) + db(v(3))
sin(cos(log([1 2 3 4 5 6 7 8 9 10])))
TIME * rnd(v(9)) - 15 * cos(vin#branch) ^ [7.9e5 8]
not ((ac3.FREQ[32] & tran1.TIME[10]) gt 3)
```

Spice のベクトルの名前は，`@`*name*[*param*] という名前でも構いません．ここで，*name* は素子の実体またはモデルの名前です．これは，素子あるいはモデルの *param* パラメータの値を表わします．どのパラメータが使えるかについての詳細は，付録 B を参照のこと．値は長さが 1 のベクトルです．この機能は `show` コマンドでも使用でき，便利なようにコマンドスクリプトの変数でも使用できます．

Nutmeg には定義済みの定数がいくつかあります．

`pi`	π (3.14159...)
`e`	自然対数の底 (2.71828...)
`c`	光速 (299,792,500m/sec)
`i`	-1 の平方根
`kelvin`	絶対 0 度 ($-273.15\,^{\circ}$C)
`echarge`	電子の電荷 ($1.6021918 \times 10^{-19}$ C)
`boltz`	Boltzman 定数 ($1.3806226 \times 10^{-23}$)
`planck`	Planck 定数 ($h = 6.626200 \times 10^{-34}$)

これらはすべて MKS 単位系です．これらの名前と衝突する変数名を使った場合は，変数が優先します．

5.2 コマンドの解釈

コマンドとして文字がタイプされ，その名前の組み込みコマンドが存在しない場合，ファイルを見つけるために `sourcepath` にリストされているディレクトリを順に探します．ファイルが見つかると，それはコマンドファイルとして (あたかも `source` されたように) 読み込まれます．ファイルが読み込まれる前に，コマンド行のファイル名に続くワードの数とワードのリストが，変数 `argc` と `argv` に設定されます．ファイルが済んだら，これらの変数はなくなります．コマンドファイルが他のコマンドファイルを呼び出すときには，`argv` と `argc` は変更されてしまうので，保存しておかなければなりません．また，局所変数がないため，コマンドファイルは再入可能ではありません．(もちろん，明示的にスタックを操作することもできますが...) このようにして，シェルスクリプトに似た Nutmeg と Spice3 用のスクリプトを書くことができます．

スクリプトが Spice3 でうまく動作するためには，スクリプトを空行 (捨てられてしまうので他の内容でも構いません) で始め，その後に `.control` が続かなければなりません．これは `source` コマ

ンドが回路の入力とコマンドファイルの実行の両方に用いられていることから生じる不幸な結果です．また，この機能により，回路ファイルの名前をコマンドとして単にタイプすれば，自動的に実行されます．回路の中で指定されている解析を行なわずに，コマンドは即座に実行されます (スクリプトが実行される前に解析を行いたい場合には，スクリプトに run コマンドを入れておきます)．

さまざまなコマンドスクリプトが /usr/local/lib/spice/scripts (またはあなたのマシンの任意のパス) にインストールされており，省略時の sourcepath は，このディレクトリを含んでいます．したがって，これらのコマンドファイルを (ほとんど) 組み込みコマンドのように使えます．

5.3 コマンド

5.3.1 Ac*: 交流小信号周波数応答解析を行う

一般形:

ac (DEC|OCT|LIN) *N Fstart Fstop*

交流解析を行います．詳しくは，このマニュアルの 4.3.1節を参照のこと．

5.3.2 Alias: コマンドの別名を作成する

一般形:

alias [*word*] [*text ...*]

word を *text* の別名にします．C シェルの別名のように，ヒストリ置換を使うこともできます．

5.3.3 Alter*: 素子やモデルのパラメータを変更する

一般形:

alter *device value*
alter *device parameter value* [*parameter value*]

Alter は，素子の値，または素子やモデルの指定されたパラメータを変更します．最初の形式は，1 つの主要な値しかない素子 (抵抗，コンデンサなど) で使います．2 番目の形式は，より複雑な素子 (BJT など) に使います．名前に “#” が含まれている場合，2 番目の形式でモデルのパラメータを変更できます．

ベクトルを値として指定するには，ベクトルを “[” で始め，ベクトルの値を続け，“]” で終えます．各値と “[”, “]” の間には，空白を入れて下さい．

5.3.4 Asciiplot: 旧式の文字プロットで値をグラフにする

一般形:

asciiplot *plotargs*

ベクトルのラインプリンタグラフを作成します．グラフは標準出力に出力されます．ファイルに出力するには，`asciiplot` *args* `...` > *file* を使います．`set` のオプション `width`, `height`, `nobreak` により，グラフの幅と高さと，改ページをするかどうかが決まります．`asciiplot` は x 軸に単純な線形補間を使用しているので，x 軸が単調増加でないもの (たとえば `sin(TIME)`) の ascii グラフを作成しようとすると，問題が生じることがあります．

5.3.5 Aspice: 非同期に Spice を実行する

一般形:

`aspice` *input-file* [*output-file*]

Spice3 を動作させ，終了したら結果のデータを読み込みます．生データは，一時ファイルに残っています．*output-file* が指定されている場合，診断出力はそのファイルに出力され，そうでない場合は捨てられます．

5.3.6 Bug: バグレポートをメールする

一般形:

`bug`

バグレポートを送ります．問題の要約と，動作させている OS の名前とバージョン番号と，Spice のバージョンと，当該の入力ファイルを送って下さい．(`GUBADDR` を定義していると，メールはそこに送られます．)

5.3.7 Cd: ディレクトリを変更する

一般形:

`cd` [*directory*]

現在の作業ディレクトリを *directory* に変更します．指定がなければ，利用者のホームディレクトリに変更します．

5.3.8 Destroy: データセットを削除する

一般形:

`destroy` [*plotnames* | `all`]

指定されたシミュレーションのデータを保持しているメモリーを解放します．

5.3.9 Dc*: 直流掃引解析を行う

一般形:

dc *Source-Name Vstart Vstop Vincr* [*Source2 Vstart2 Vstop2 Vincr2*]

直流伝達特性分析を行います．詳細については，このマニュアルの 4.3.2節を参照のこと．

5.3.10 Define: 関数を定義する

一般形:

define *function*(*arg1*, *arg2*, ...) *expression*

名前が *function* で，引数が *arg1*, *arg2*, ... の関数を *expression* で定義します．式の中で引数を使えます．後に関数が使われるときに，仮引数は与えられた引数で置き換えられて解釈されます．*expression* がない場合，*function* の定義が表示され，define に引数がない場合は，現在有効なすべての関数定義が表示されます．

次の関数は有用です．

```
define max(x,y) (x > y) * x + (x <= y) * y
define min(x,y) (x < y) * x + (x >= y) * y
```

5.3.11 Delete*: トレースやブレークポイントを解除する

一般形:

delete [*debug-number* ...]

指定されたブレークポイントやトレースを削除します．*debug-number* は status コマンドで示されたものです (status > file とした場合，*debug-number* は表示されません)．

5.3.12 Diff: ベクトルを比較する

一般形:

diff *plot1 plot2* [*vec* ...]

指定されたプロットのすべてのベクトルまたは指定されたベクトルを比較します．2 つのプロットの中に異なったベクトルがある，またはベクトルの値に明らかに異なっていると，その差異が報告されます．明らかな差異を決定するのに，変数 diff_abstol, diff_reltol, diff_vntol が使われます．

5.3.13 Display: 既知のベクトルとその型を表示する

一般形:

display [*varname* ...]

現在定義されているベクトルまたは指定されたベクトルの要約を表示します．変数 nosort が設定されていなければ，ベクトルは名前順に並べられます．表示される情報は，ベクトルの名前，長さ，型，実数か複素数か，です．さらに，一つのベクトルには [scale] というラベルが付くこともあります．vs 引数なしで plot のようなコマンドが与えられたとき，scale が x 軸として使われます．scale は，生ファイルや新しいプロットの中で，常に最初のベクトルになります．scale を未定義にする (すなわち let TIME=[]) と，残りのベクトルの一つが新しいスケールになります (どれになるかは不定です).

5.3.14 Echo: 文章を表示する

一般形:

echo [*text* ...]

与えられた文章を画面に表示します.

5.3.15 Edit*: 現在の回路を編集する

一般形:

edit [*file*]

現在の Spice3 入力ファイルをファイルに出力し，そのファイルに対してエディタを呼び出して，利用者がそれを変更し，ファイルを読み戻して現在のファイルと置き換えます．ファイル名が指定されている場合，そのファイルを編集して読み込み，その回路を現在の回路とします.

5.3.16 Fourier: フーリエ変換を行う

一般形:

fourier *fundamental_frequency* [*value* ...]

指定された各値について，基本波の第 10 高調波まで (変数 nfreqs が設定されている場合はそこまで，5.5参照) のフーリエ解析を行います．出力は，.FOUR 行と同様です．*value* は任意の有効な式です．*value* は固定の間隔の，変数 fourgridsize で指定された個数のグリッド (指定されていない場合は 200) に補間されます．補間は，変数 polydegree が設定されていればその次元で，設定されていなければ 1 次で行われます．polydegree が 0 の場合，補間は行われません．しかし，時間軸が単調増加でない場合，このようにすると誤った結果が得られます.

5.3.17 Hardcopy: グラフを印刷用のファイルに保存する

一般形:

hardcopy *file plotargs*

plot と同様ですが，グラフを含んだ *file* というファイルを作成します．このファイルは **plot(5)** 形式のイメージで，**plot(1)** プログラム，または lpr に -g フラグを付けることにより印刷されます.

5.3.18 Help: Spice3 コマンドの要約を表示する

一般形:

`help` [`all`] [*command* `...`]

ヘルプを表示します．引数 `all` が指定された場合，入力可能なすべてに関する短い説明が表示されます．*commands* が指定された場合，そのコマンドの説明が表示されます．それ以外の場合は，主要なコマンドのみが表示されます．

5.3.19 History: これまで入力されたコマンドを表示する

一般形:

`history` [*number*]

ヒストリ，すなわちキーボードから入力された最新の *number* 個のコマンドを表示します．Spice3 バージョン 3a7 とそれ以前のバージョンでは，すべてのコマンド (ファイルから読んだものを含めて) が保存されます．

5.3.20 Iplot*: シミュレーション実行中にグラフを作成する

一般形:

`iplot` [*node* `...`]

Spice3 のシミュレーション実行中に，*node* の値をグラフ化します．`iplot` コマンドは，過渡解析シミュレーションで問題を起こしている点を見つけるのに使えます．

5.3.21 Jobs: 計算中の非同期の Spice ジョブを表示する

一般形:

`jobs`

現在実行中の非同期の Spice3 ジョブについて報告します．Nutmeg は，コマンドを実行するたびにジョブが完了したかどうか調べます．ジョブが終了していると，データが読み込まれ，利用可能になります．

5.3.22 Let: 値をベクトルに代入する

一般形:

`let` *name* `=` *expr*

上述した式 *expr* で指定された値で *name* という新しいベクトルを作成します．式が `[]` (長さが 0 のベクトル) の場合，ベクトルは未定義になります．名前に添字を付ける (たとえば `name[0]`) ことにより，ベクトルの個々の要素を変更できます．引数がない場合は，`let` は `display` と同じ動作をします．

5.3.23 Linearize*: 線形なスケールに補間する

一般形:

```
linearize vec ...
```

現在のプロットのすべてのベクトル，または引数で指定されたベクトルのみで新しいプロットを作成します．新しいベクトルは，現在アクティブな過渡解析の tstep, tstart, tstop の値によって決定される線形の時間スケールで補間されます．現在読み込まれている入力ファイルに過渡解析が含まれていて (または，最後の reset の後で会話型で tran コマンドを実行していてもよい)，現在のプロットが過渡解析のものでなければなりません．このコマンドは，Spice3 が過渡解析の結果を Spice2 のようには出力しないため必要となります．

5.3.24 Listing*: 現在の回路のリストを表示する

一般形:

```
listing [logical] [physical] [deck] [expand]
```

logical 引数が指定されている場合，リストはすべての継続行が 1 行につなげられ，physical 引数が指定されている場合，各行はファイルと同じように表示されます．省略時は，logical です．deck リスティングは physical と同じですが，行番号を付けずに，入力ファイルをそのまま再現します (ただし大文字小文字は保存されません)．expand が指定された場合，すべてのサブ回路が展開されて回路が表示されます．

5.3.25 Load: 生ファイルのデータを読み込む

一般形:

```
load [filename ...]
```

バイナリまたはテキスト形式の生ファイルのデータを，指定された *file* から読み込みます．省略時のファイル名は rawspice.raw，または -r フラグが指定されていればその引数のファイル名です．

5.3.26 Op*: 動作点解析を行う

一般形:

```
op
```

動作点解析を行います．詳細については，このマニュアルの 4.3.5節を参照のこと．

5.3.27 Plot: 値をグラフで表示する

一般形:

plot *exprs* [ylimit *ylo yhi*] [xlimit *xlo xhi*] [xindices *xilo xihi*]
[xcompress *comp*] [xdelta *xdel*] [ydelta *ydel*] [xlog] [ylog] [loglog]
[vs *xname*] [xlabel *word*] [ylabel *word*] [title *word*] [samep]
[linear]

指定された *exprs* を画面に (グラフィック端末を使っている場合) グラフで表示します．xlimit と ylimit 引数は，それぞれ x 軸，y 軸の下限と上限を決定します．xindices 引数は，グラフに表示される点の範囲を決定します．*xilo* 番目から *xihi* 番目の間の点がグラフに表示されます．xcompress 引数は，*comp* 点ごとに 1 点をグラフに表示することを指定します．xdelta または ydelta パラメータは，それぞれ x 軸，y 軸のグリッドの間隔を指定します．パラメータ名は，それぞれ xl, yl, xind, xcomp, xdel, ydel のように省略できます．

xname 引数は，x 軸のスケールとして使われる式です．xlog または ylog が指定されていると，それぞれ x 軸または y 軸が対数軸になります (loglog は両方を指定したのと同じです)．xlabel, ylabel 引数は，指定されたラベルをそれぞれ x 軸，y 軸に使います．

samep が指定された場合，(xname 以外の) 他のパラメータの値は，コマンド行で再定義されなければ，前の plot, hardcopy, asciiplot コマンドのものが使われます．

title 引数は，グラフの下部のグラフ名の場所に使われます．

linear キーワードは，省略時に対数軸となるグラフ (交流解析の出力など) の軸をリニアにするのに使います．

最後に，キーワード polar は極座標プロットを生成します．スミスチャートを生成するには，キーワード smith を使います．データが変換されることに注意してください．スミスチャートでは，関数 $(x-1)/(x+1)$ で変換されたデータを見ることになります．スミスチャートのグリッドがあるがスミス変換を行わない極座標プロットを作成するには，キーワード smithgrid を使います．

5.3.28 Print: 値を表示する

一般形:

print [col] [line] *expr* ...

式 *expr* で表されたベクトルを表示します．col 引数がある場合，指定されたベクトルを並べて表示します．line が指定された場合，ベクトルは横に表示されます．指定されたベクトルの長さがすべて 1 の場合は line が省略時の動作で，それ以外の場合は col が省略時の動作です．オプション width, length, nobreak がこのコマンドに有効です (asciiplot (5.3.4節) を参照のこと)．式が all の場合，すべての利用可能なベクトルが表示されます．したがって，print col all > file は，すべてのベクトルを Spice2 形式でファイルに出力します．変数 noprintscale が真でなければ，scale ベクトル (時間，周波数) が常に最初の列になります．

5.3.29 Quit: Spice3 または Nutmeg を終了する

一般形:

quit

Nutmeg または Spice を終了します．

5.3.30 Rehash: 内部ハッシュ表を初期化する

一般形:

```
rehash
```

UNIX コマンド探すときに使われる内部ハッシュ表を再計算して，利用者のコマンドサーチパスにあるすべての UNIX コマンドをコマンド補完で利用可能にします．これは，unixcom を最初に設定しなければ役に立ちません (5.5節を参照のこと).

5.3.31 Reset*: 解析を初期化する

一般形:

```
reset
```

(ブレークポイントの後や 1 つ以上の解析をすでに行った後で) 回路の中間データを破棄し，入力ファイルを読み直します．Spice-3e とそれ以前のバージョンでは，これは run コマンドにより自動的に行われます.

5.3.32 Reshape: ベクトルの次元を変更する

一般形:

reshape *vector vector* ...
または
reshape *vector vector* ... [*dimension*, *dimension*, ...]
または
reshape *vector vector* ... [*dimension*][*dimension*] ...

このコマンドは，ベクトルまたはベクトルの集合の次元を変更します．最後の次元は，指定しなくてもよく，自動的に埋められます．次元の指定がなければ，最初のベクトルの次元が他のベクトルにコピーされます．'dimensions of x were inconsistent' という形式のエラーメッセージは，無視して構いません.

5.3.33 Resume*: stop の後でシミュレーションを再開する

一般形:

```
resume
```

stop または割り込み (ctrl-C) の後で，シミュレーションを再開します.

5.3.34 Rspice: リモートで Spice を実行する

一般形:

`rspice` *input_file*

input_file を入力ファイルとして，または引数がない場合は現在の回路を入力として，Spice3 をリモートで実行します．Nutmeg または Spice3 はジョブが終了するのを待ち，リモートジョブからの出力を利用者の標準出力へと渡します．ジョブが終了した時，`aspice` のようにデータが読み込まれます．変数 `rhost` が設定されている場合，Nutmeg は省略時のリモート Spice3 サーバーマシンではなくそのホストに接続します．このコマンドは，“`rsh`” コマンドを使うので，“`.rhosts`” ファイルやその他の同等な方法で認証される必要があります．“`rsh`” は「リモートシェル」プログラムのことなので，あなたのシステムでは “`remsh`” であるかもしれないことに注意して下さい．省略時の “`rsh`” という名前以外を使うには，変数 `remote_shell` を設定します．変数 `rprogram` が設定されている場合，`rspice` は，これをリモートシステムで動作させるプログラムへのパス名として使います．

`rspice` は，“`alter`” や “`altermod`” コマンドで変更された要素を認識しません．

5.3.35 Run*: 入力ファイルの解析を実行する

一般形:

`run` [*rawfile*]

入力ファイルで指定されたシミュレーションを実行します．制御行 `.ac`, `.op`, `.tran`, `.dc` のいずれかがある場合，それが実行されます．*rawfile* が指定されていればそこにも出力されますが，会話的に出力を利用することもできます．Spice-3e とそれ以前のバージョンでは，入力ファイルは再度読み込まれ，`set` や `alter` コマンドの影響は元に戻ります．この作用はもはやありません．

5.3.36 Rusage: 資源の使用量

一般形:

`rusage` [*resource* `...`]

資源の使用量の統計を表示します．資源の名前が指定された場合，その資源の使用量のみを表示します．ほとんどの資源で，回路が読み込まれている必要があります．現在，有効な資源は，

`elapsed` 最後の `rusage elapsed` の呼び出しからの経過時間

`faults` ページフォルトとコンテキストスイッチの数 (BSD のみ)

`space` 使用しているデータスペース

`time` これまでに使用した CPU 時間

`temp` 動作温度

`tnom` 素子のパラメータを測定した温度

`equations` 回路の方程式の数

`time` 総解析時間

totiter 総繰り返し回数

accept 受け入れられた時点の数

rejected 拒絶された時点の数

loadtime 回路行列と RHS を読み込むのにかかった時間

reordertime 行列並べ替えの時間

lutime L-U 分解の時間

solvetime 逆行列計算時間

trantime 過渡解析の時間

tranpoints 過渡解析の時点の数

traniter 過渡解析の繰り返しの数

trancuriter 最後の時点の過渡解析の繰り返しの数*

tranlutime 過渡解析の L-U 分解の時間

transolvetime 過渡解析の逆行列計算時間

everything 上記のすべて

* まちがって「1 点あたりの過渡解析の繰り返し」と表示されています.

5.3.37 Save*: 出力の組を保存する

一般形:

```
save [all | output ...]
.save [all | output ...]
```

出力の組を保存して，残りは捨てます．save コマンドでノードが指定されている場合，それは run コマンドのあとで現在のプロットに現れているか，Spice がバッチモードで起動された場合，生ファイルに存在している必要があります．ノードがトレースされているかグラフに描かれている (5.3.52節参照) 場合も保存されます．過去との互換性のため，save がない場合，すべてが保存されます.

キーワード “all” が save に現れた場合，リストされた値に加え，すべての省略時の値 (ノードの電圧と電圧源の電流) も保存されます.

5.3.38 Sens*: 感応度分析を行なう

一般形:

```
sens output_variable
sens output_variable ac (DEC|OCT|LIN) N Fstart Fstop
```

感応度分析を行ないます．*output_variable* は，ノードの電圧 (たとえば “`v(1)`” や “`v(A,out)`” または電圧源を流れる電流 (たとえば “`i(vtest)`”) のいずれかです．最初の形式は直流感応度を計算し，2 番目の形式は交流感応度を計算します．出力値の次元は，(入力の変化 1 パーセントあたりの出力の変化率ではなく) 入力の変化 1 単位あたりの出力の変化です．

5.3.39 Set: 変数の値を設定する

一般形:

```
set [word]
set [word = value] ...
```

value があれば，*word* の値を *value* に設定します．任意の *word* に任意の値 (数値または文字列) を設定できます．*value* が指定されていない場合，値は論理値の「真」になります．

`$`*word* と書くことによって *word* の値をコマンドに入れることができます．変数に括弧で囲まれた値のリスト (値とは空白で分けなければなりません) を設定した場合，変数の値はそのリストになります．

Nutmeg で使われる変数は，5.5節でリストされています．

5.3.40 Setcirc*: 現在の回路を変更する

一般形:

```
setcirc [circuit_name]
```

現在の回路とは，後述するシミュレーションコマンドで使われる回路です．回路が `source` コマンド (5.3.47節参照) で読み込まれると，その回路が現在の回路になります．

5.3.41 Setplot: ベクトルの現在の組を切り替える

一般形:

```
setplot [plotname]
```

指定された名前のプロットを現在のプロットに設定します．名前が指定されていない場合は，利用者にメニューで尋ねます．(プロットには読み込まれた順序で `tran1` や `op2` といった名前が付けられます．これらの名前は `setplot` や `display` コマンドで示され，5.3.12節の `diff` で使われます．) “`New plot`” を選択した場合，現在のプロットは，まったくベクトルが定義されていないものになります．

「プロット」という用語は，Spice の実行の結果であるベクトルのグループを表わしています．2 つ以上のファイルが読み込まれている場合，または 1 つのファイルに 2 つ以上のプロットがある場合，Nutmeg はそれらを分離して管理し，現在のプロットのベクトルのみを表示します．

5.3.42 Settype: ベクトルの型を設定する

一般形:

`settype` *type vector* `...`

指定されたベクトルの型を *type* に変更します．型名は，`sconvert` のマニュアルページで説明されています．

5.3.43 Shell: コマンドインタプリタを呼び出す

一般形:

`shell` [*command*]

OS のコマンドインタプリタを呼び出します．指定された *command* を実行するか，会話型のシェルを呼び出します．

5.3.44 Shift: リスト変数を変更する

一般形:

`shift` [*varname*] [*number*]

varname がリスト変数の名前なら，*number* 要素分左にシフトします (すなわち，左側の *number* 個の要素が取り除かれます)．省略時の *varname* は `argv` で，省略時の *number* は 1 です．

5.3.45 Show*: 素子の状態をリストする

一般形:

`show` *devices* [`:` *parameters*]，`...`

古い形式:

`show -v @`*device*[`[`*name*`]`]

`show` コマンドは，指定された素子の動作条件を要約した表 (Spice2 の動作点要約と同じような) を表示します．*device* が指定されていない場合，標準の素子の組が表示されます．*device* が 1 文字の場合，その種類の素子すべてが表示されます．*device* がサブ回路名 (“`:`” で始まり “`:`” で終わる) の場合，そのサブ回路の素子のみが表示されます (サブ回路の素子を再帰的に取り出すには，名前を “`::`” で終えます)．2 番目および 3 番目の形式は，サブ回路から指定された種類の素子を選ぶのに (“*letter*`:`*subcircuit*`:`” や “*letter*`:`*subcircuit*`::`” のように) 組み合わせて使えます．素子の完全な名前を指定すれば，その素子のみを表示します．“*#modelname*” または “`:`*subcircuit#modelname*” または “*letter*`:`*subcircuit#modelname*” という形式を使うことによって，モデルにより素子を選択することもできます．

parameters が指定されていない場合，標準的なパラメータの組が表示されます．*parameters* のリストに “+” が含まれている場合，指定されたパラメータに加え，標準のパラメータの組が表示されます．

devices と *parameters* について，ワード “all” は自明な働きをします．*device* リストと *parameter* リストを分ける “:” には，空白が必要です．

(“-v” を伴った)「古い形式」では，データを古いより冗舌な Spice3f 以前の形式で表示します．

5.3.46 Showmod*: モデルのパラメータの値を表示する

一般形:

showmod *models* [: *parameters*], ...

showmod コマンドは，show コマンド (上述) と同じように動作しますが，モデルのパラメータの値を表示します．*model* の形式は，素子の種類を指定する 1 文字, “*letter*:*subckt*:”, “*modelname*”, “:*subckt*:*modelname*”, “*letter*:*subcircuit*:*modelname*” です．

5.3.47 Source: Spice3 入力ファイルを読み込む

一般形:

source *file*

Spice3 では，Spice3 入力ファイル *file* を読み込みます．ファイルには，.control と .endc で囲んで Nutmeg と Spice3 のコマンドを入れることができます．これらのコマンドは，回路が読み込まれた直後に実行されます．したがって，ac は，対応する .ac 行と同様に動作します．すべての入力ファイルの 1 行目はタイトル行であるとして，解釈はされませんが回路の名前として保存されます．この規約の例外は，.spiceinit です．したがって，Spice3 コマンドスクリプトは空行で始めねばならず，“*#” を続けると，制御行と解釈されます．これにより，以前のバージョンでは無視されるコマンドを Spice3 の入力ファイルに埋め込むことができます．

Nutmeg では，ファイル *filename* からコマンドを読み込みます．文字 * で始まる行は，コメントとして扱われ，無視されます．

5.3.48 Status*: ブレークポイントの情報を表示する

一般形:

status

現在有効なトレースとブレークポイントをすべて表示します．

5.3.49 Step*: 時点をいくつか実行する

一般形:

step [*number*]

1 回または *number* 回繰り返し計算を行ない，停止します．

5.3.50 Stop*: ブレークポイントを設定する

一般形:

`stop` [`after` *n*] [`when` *value cond value*] `...`

ブレークポイントを設定します．引数 `after` *n* は，繰り返しを *n* 回行なった後で停止することを意味し，引数 `when` *value cond value* は，最初の *value* と 2 番目の *value* の関係が指定された *cond* の場合に停止します．*cond* は，

`eq` または =	等しい
`ne` または <>	等しくない
`gt` または >	より大きい
`lt` または <	より小さい
`ge` または >=	以上
`le` または <=	以下

のいずれかです．

関係演算子と衝突するので，`stop` コマンドでは IO リダイレクションはできません (いずれにせよ出力は生成されません)．*value* は実行中の回路のノード名または実数です．たとえば `stop after 4 when v(1) > 4 when v(2) < 2` のように 2 つ以上の条件が指定された場合，2 つの条件の論理和を意味します．

5.3.51 Tf*: 伝達関数解析を行なう

一般形:

`tf` *output_node input_source*

`tf` コマンドは，伝達関数解析を行ない，指定された出力ノードと入力電源間の伝達関数 (出力/入力)，出力抵抗，入力抵抗を返します．この解析は，小信号直流 (ゆっくりと変化する) 入力を仮定しています．

5.3.52 Trace*: ノードをトレースする

一般形:

`trace` [*node* `...`]

解析の 1 ステップごとに，指定されたノードの値が表示されます．同時に複数のトレースを有効にできます．すべての解析でトレースを使えるわけではありません．トレースを解除するには，`delete` コマンドを使います．

5.3.53 Tran*: 過渡解析を行なう

一般形:

`tran` *Tstep Tstop* [*Tstart* [*Tmax*]] [`UIC`]

過渡解析を行ないます．詳細は，このマニュアルの 4.3.9節を参照のこと．

5.3.54 Transpose: 多次元のデータセットの要素を交換する

一般形:

`transpose` *vector vector* `...`

このコマンドは，多次元のベクトルを転置します．2 つの可変電源による直流伝達特性以外の Spice3 のどの解析も，多次元のベクトルを生成しません．1 次元のベクトルを 2 次元のベクトルにするには，“`reshape`” コマンドを使います．さらに，標準のスケールはグラフを描くのに不適切になります．2 番目の電源に対応するベクトルの最初の部分のみに対してグラフを描く必要があります．たとえば (MOS トランジスタの伝達特性を生成する回路の例),

```
spice3 > dc vgg 0 5 1 vdd 0 5 1
spice3 > plot i(vdd)
spice3 > reshape all [6,6]
spice3 > transpose i(vdd) v(drain)
spice3 > plot i(vdd) vs v(drain)[0]
```

5.3.55 Unalias: 別名を解除する

一般形:

`unalias` [*word* `...`]

word に設定されている別名があれば，それを解除します.

5.3.56 Undefine: 関数定義を解除する

一般形:

`undefine` *function*

指定された利用者定義の関数の定義を解除します.

5.3.57 Unset: 変数をクリアする

一般形:

`unset` [*word* `...`]

指定された変数 (*word*) の値をクリアします.

5.3.58 Version: Spice のバージョンを表示する

一般形:

`version` [*version_id*]

実行中の Nutmeg のバージョンを表示します．引数が指定されている場合，引数が現在の Spice のバージョンと一致するかどうか調べます．(これはおもに生ファイルの `Command:` 行で使われます．)

5.3.59 Where: 問題があるノードや素子を特定する

一般形:

```
where
```

過渡解析や動作点解析を行なっているときに，収束しない原因となっている最後のノードまたは素子の名前が記録されます．回路を調べたり，問題を修正したり，バグレポートを作成したりできるよう，`where` はこの情報を表示します．このコマンドは，シミュレーション実行中にも，シミュレータが解析を諦めた後でも使うことができます．過渡解析では，解析の進行を監視するのに `iplot` コマンドを使うことができます．解析が急激に遅くなったり止ったりしたときに，(ctrl-C で) シミュレータに割り込みをかけ，`where` を発行します．1 つのノードまたは素子のみが表示されますが，問題が 2 つ以上のノードで起こっていることもあります．

5.3.60 Write: データをファイルに書き出す

一般形:

`write` [*file*] [*exprs*]

指定された式を *file* に書き出します．

ベクトルは，まずプロットごとにまとめられて書き出されます (すなわち，式のリストに，あるプロットからの 3 つのベクトルと，別のプロットからの 2 つのベクトルが含まれている場合，2 つのプロットが出力され，最初のプロットには 3 つのベクトルが，もう一つのプロットには 2 つのベクトルが入ります)．さらに，ベクトルのスケールが指定されていない場合，スケールが自動的に書き出されます．

省略時の形式はテキストですが，`set filetype` コマンドにより変更できます．省略時のファイル名は `rawspice.raw`，またはコマンド行の `-r` フラグの引数で指定されたものです．省略時の式は `all` です．

5.3.61 Xgraph: グラフ表示に xgraph(1) を使う

一般形:

`xgraph` *file* [*exprs*] [*plot_options*]

Spice3/Nutmeg の `xgraph` コマンドは，`plot` コマンドのようにデータをグラフ化しますが，X11 で良く使われているグラフ作成プログラム **xgraph** を使います．

file が “`temp`” または “`tmp`” の場合，グラフを作成している間データを保持するのに一時ファイルが使われます．利用可能なオプションについては，`plot` コマンド (5.3.27節) を参照のこと．極座標およびスミスチャート以外のオプションを利用できます．

5.4 制御構造

5.4.1 While End

一般形:

```
while condition
    statement
    ...
end
```

任意の代数式 *condition* が真の間，*statement* を実行します.

5.4.2 Repeat End

一般形:

```
repeat [number]
    statement
    ...
end
```

statement を *number* 回実行します．引数がない場合は，永久に実行します.

5.4.3 Dowhile End

一般形:

```
dowhile condition
    statement
    ...
end
```

`while` と同様ですが，*statement* が実行された後で *condition* を調べます.

5.4.4 Foreach End

一般形:

```
foreach var value ...
    statement
    ...
end
```

各 *value* について 1 回ずつ，変数 *var* の値が現在の *value* の値に設定されて *statement* が実行されます．(*var* は $*var* という記法でアクセスできます—5.6節参照).

5.4.5 If Then Else

一般形:

```
if condition
   statement
   ...
else
   statement
   ...
end
```

condition が 0 以外の場合，最初の *statement* の組が実行され，*condition* が 0 の場合，2 番目の *statement* の組が実行されます．`else` と 2 番目の *statement* の組は，省略することもできます．

5.4.6 Label

一般形:

```
label word
```

`goto` *word* という文が現われたとき，制御がこの点に移されます．それ以外の場合は，なにもしません．

5.4.7 Goto

一般形:

```
goto word
```

`label` *word* という文がこのブロックの中および外側のブロックにある場合，制御がそこに移ります．ラベルがトップレベルにある場合，`goto` 文より前になくてはなりません (すなわち，前方への `goto` は，ブロックの中だけで使えます)．

5.4.8 Continue

一般形:

```
continue
```

この文を取り囲む `while`, `dowhile`, `foreach` ブロックがある場合，制御は条件のテストまたは (`foreach` の場合) 次の値に移ります．そうでない場合は，エラーとなります．

5.4.9 Break

一般形:

```
break
```

この文を取り囲む while, dowhile, foreach ブロックがある場合，制御はブロックの外側に移ります．そうでない場合は，エラーとなります．

もちろん，制御構造は入れ子にできます．ブロックに入ったときに入力が端末の場合，プロンプトは現在入っているブロックの数に対応した個数の '>' になります．現在の制御構造は，デバッグ用のコマンド cdump で調べることができます．

5.5 変数

"set" コマンドで変数を設定することにより，Nutmeg と Spice3 の動作を変えることができます．以下で言及する変数に加え，".OPTIONS" の 4.1節で説明したオプションを Spice3 の set コマンドで設定して，シミュレータの振る舞いを変えることができます．

set コマンドで変更できる Nutmeg に有効な変数は，

diff_abstol diff コマンドで使われる絶対許容範囲．

appendwrite write が使われたときにファイルが存在している場合，追加書き込みを行ないます．

color*N* これらの変数は X がカラーディスプレイで動作しているときに色数を決定します．*N* は 0 から 15 です．色 0 は背景色，色 1 はグリッドと文字の色，色 2 から 15 はベクトルのグラフ表示に順に使われます．色の変数の値は，/usr/lib/rgb.txt にある色の名前でなければなりません．

combplot 点をつないだグラフではなく，x 座標の各点から垂直線を伸ばしたグラフを作成します．このオプションは，後述の plottype オプションの中で指定することもできます．

cpdebug cshpar デバッグ情報を表示します (-DCPDEBUG フラグを付けてコンパイルされていなければなりません)．現在のバージョンではサポートされていません．

debug 設定されていると，大量のデバッグ情報を出力します (-DFTEDEBUG フラグを付けてコンパイルされていなければなりません)．現在のバージョンではサポートされていません．

device グラフィクスデバイスの名前 (/dev/tty??)．この変数が設定されていない場合，利用者の端末が使われます．他のモニターにグラフを出力したい場合，device と term の両方を設定しなければならないでしょう．(device がファイルの名前に設定されている場合，Nutmeg はグラフィック制御コードをそのファイルにダンプします—これはグラフを保存するのに有用です．)

echo 各コマンドを実行前に表示します．

filetype この値は ascii または binary のいずれかで，ファイルのフォーマットを決定します．省略時の値は ascii です．

fourgridsize フーリエ解析を行なうときに何点で補間するかを指定します．

gridsize この変数が整数に設定されていると，グラフを描く際に y 軸をその数で等分します．そうでなければ，現在のスケールが使われます (これは等間隔の点とは限りません)．現在のスケールが厳密に単調増加でなければ，このオプションの効果はありません．

hcopydev これが設定されている場合，hardcopy コマンドを実行すると，作成されたファイルは自動的に lpr -P*hcopydev file* というコマンドで *hcopydev* というプリンタに出力されます．

hcopyfont この変数は，ハードコピーでグラフを出力する際のフォント名を指定します．値は装置によって異なります．

hcopyfontsize これは，ハードコピーでグラフを出力する際の，フォントのスケーリングファクターです．

hcopydevtype この変数は，hardcopy コマンドで使うプリンタ出力の種類を指定します．hcopydevtype が設定されていない場合，**plot(5)** 形式が仮定されます．標準の配付では，別の出力形式としては postscript を認識します．hcopydev と共に使われた場合，hcopydevtype にはそのプリンタでサポートされている形式を指定しなければなりません．

height asciiplot と print col のページの長さ．

history ヒストリリストに保存するイベントの数．

lprplot5 これは，**plot(5)** 形式のグラフをプリンタやプロッタに送るために使われるコマンドを指定する際に使われる，**printf(3s)** 形式の書式文字列です．与えられる最初のパラメータはプリンタ名で，2 番目のパラメータはグラフが入っているファイル名です．どちらのパラメータも文字列です．不適切な書式文字列を指定すると，当然ですが Spice3 が異常終了します．

lprps これは，PostScript 形式のグラフをプリンタやプロッタに送るために使われるコマンドを指定する際に使われる，**printf(3s)** 形式の書式文字列です．与えられる最初のパラメータはプリンタ名で，2 番目のパラメータはグラフが入っているファイル名です．どちらのパラメータも文字列です．不適切な書式文字列を指定すると，当然ですが Spice3 が異常終了します．

nfreqs fourier コマンドで計算する周波数の数 (省略時の値は 10).

nobreak asciiplot と print col で改ページしません．

noasciiplotvalue asciiplot を実行する際に，グラフに表示する最初のベクトルの値を左側に表示しません．

noclobber IO リダイレクションをする際に，既存のファイルを上書きしません．

noglob ワイルドカード文字 '*', '?', '[', ']' を展開しません．

nogrid グラフを描くときにグリッドを表示しません (ただし軸にラベルは付けます).

nomoremode nomoremode が設定されていない場合，大量のデータが画面に表示される (たとえば print や asciiplot コマンド) ときに，画面 1 面ごとに出力が止まり，リターンが押されると表示を再開します．nomoremode が設定されている場合，画面ごとのチェックは行なわれず，画面からスクロールされ消えていきます．

nonomatch noglob が設定されていない場合で，ワイルドカードを含む表現が一致しない場合，文句を言わずにワイルドカード文字をそのまま使います．

nosort 変数名をソートせずに表示します．

noprintscale print col コマンドで，最も左側にスケールを表示しません．

numdgt データの表を表示する (fourier, print col) ときの桁数．省略時の精度は 6 桁です．倍精度は約 16 桁なので，numdgt は 16 以下に設定すべきです．負の数値を表示する場合，表の幅を一定にするため，1 桁少ない桁数で表示します．

plottype normal, comb, point:*chars* のいずれかです．normal は，省略時の値で，点を結んだグラフを作成します．comb は櫛状のグラフを作成します (上述の combplot 変数の説明を参照のこと)．point は各点を個別に打ちます．*chars* はグラフに描く各ベクトルに使う文字のリストです．省略された場合は，標準の文字の組が使われます．

polydegree plot コマンドがデータをフィットさせる多項式の次数．polydegree が N の場合，Nutmeg は N 点ごとに N 次の多項式で補間を行ない，両端の点の間に 10 点を描きます．点が単調増加でない場合，補間がうまく行くまで曲線を回転させて次元を減少させます．

polysteps 曲線のあてはめを行なう際に，2 点の間で補間する点の数．省略時の値は 10.

program 実行中のプログラムの名前 (argv[0]).

prompt プロンプト．文字 '!' は現在のイベント番号に置きかわります．

rawfile 作成される生ファイルの省略時の名前．

diff_reltol diff コマンドで使われる相対許容範囲．

remote_shell rspice を実行する際に使われる名前 (省略時の値は "rsh").

rhost リモートの Spice3 を実行するマシン (5.3.34節の rspice コマンドの説明を参照のこと).

rprogram rspice コマンドで使われるリモートプログラムの名前．

slowplot 各グラフとグラフの間で停止し，利用者がリターンをタイプするのを待ちます．

sourcepath source コマンドを実行するときに，ファイルを探すディレクトリのリスト．省略時の値は，カレントディレクトリと標準 Spice ライブラリ (/usr/local/lib/spice または Spice3 のソースで #define LIBPATH されたもの) です．

spicepath aspice コマンドで使うプログラム．省略時の値は，/cad/bin/spice.

term 現在の端末の mfb 名．

units この値が degrees の場合，すべての三角関数はラジアンではなく度を使います．

unixcom コマンドが定義されていない場合，それを UNIX のコマンドとして実行します．このオプションを設定すると，5.3.30節の rehash コマンドを行なったのと同じ効果があります．これは Nutmeg をログインシェルとして使いたい人に有用です．

verbose 冗舌になります．echo と debug/cpdebug の中間です．

diff_vntol diff で使われる電圧の絶対許容誤差．

width asciiplot と print col のページの幅．

x11linearcs X11 の実装には，円弧の描画がうまくないものがあります．このオプションを設定すると，Spice3 は直線を使って曲線を近似してグラフを描きます．

xbrushheight X が動作している場合に使われるブラシの高さ．

xbrushwidth X が動作している場合に使われるブラシの幅．

xfont データをグラフ化したりラベルを入力する際に使われる X のフォントの名前．可変幅のフォントを使うと，グラフがきれいに見えなくなることもあります．

Spice3 は使うが Nutmeg は使わない変数がいくつかあります．

editor edit コマンドで使うエディタの名前．

modelcard モデルカードの名前．

noaskquit 保留中の回路があるか，あるいは保存されていないプロットがあるかをチェックしません．通常，このような場合，Spice3 は利用者に警告します．

nobjthack BJT にノードが 4 つあると仮定します．

noparse 入力ファイルを読み込んだときに，解釈しないようにします (デバッグに有用です)．もちろん，解釈されていなければ実行できません．

nosubckt サブ回路を展開しません．

renumber .include があったとき，番号を付け直します．

subend サブ回路を終えるカード．

subinvoke サブ回路を呼び出す文字 (通常は 'x')．

substart サブ回路を始めるカード．

5.6 その他

入力ファイルにサブ回路がある場合，Spice3 はサブ回路の実体を展開します．サブ回路は .subckt と .ends，または変数 substart と subend の任意の値で区切られています．サブ回路の実体は，素子の種類を 'x' と指定することにより作成されます．すなわち，素子の行を次のように書いた場合に実体が作成されます．

```
xname node1 node2 ... subcktname
```

ここでノードは .subckt 行の仮引数を置き換えるノード名です．仮引数でないすべてのノードとサブ回路内の素子の名前には，実体に指定された名前と ':' が前に付きます．何重にも入れ子になったサブ回路の場合，ノードや素子の名前は *subckt1*:*subckt2*:...:*name* のようになります．変数 subinvoke が設定されていると，'x' ではなくそれがサブ回路の実体を指定する文字として使われます．

Nutmeg は，メモリを使い果たしそうかときどきチェックし，利用者にそのことを知らせます (これは Spice のフロントエンドでより有用です)．

C シェル形式の引用 "" および '' と，逆クォート置換を使えます．単一引用符の中では，置換 (ヒストリ置換など) は行なわれません．二重引用符の中では，複数の単語が 1 つの単語として扱われ，置換が行なわれます．逆引用符で囲まれたテキストは，そのテキストをシェルのコマンドとして実行した結果により置き換えられます．

Tenex 形式 (4.3 C シェルの 'set filec') の，コマンド，ファイル名，キーワードの補完もできます．行の 2 文字目以降で EOF (ctrl-D) をタイプすると，コマンドや指定可能な引数が表示されます (ctrl-D だけをタイプすると Nutmeg を終了します)．ESC をタイプすると，Nutmeg はすでに利用者がタイプしたもので補完しようとします．すべてのコマンドのリストを見たいときは，空白 ctrl-D とタイプします．

変数の値をコマンドの中で使うには，値を使いたいところで $*varname* とします．特別な変数 $$ と $< は，それぞれプログラムのプロセス ID と，変数が評価されたときまでに端末から読み込まれた行数を表わします．変数の名前が $&*word* という形式の場合，*word* はベクトル (5.1節参照) として扱われ，その値はその変数の値からとられます．$*foo* が有効な変数で，型がリストの場合，式 $*foo*[*low*-*high*] は要素の範囲を表わします．上限の添字または下限の添字のいずれかを指定しないこともでき，$*foo*[*len*-0] とすれば逆順のリストを得ることができます．$?*foo* という記法は，変数 *foo* が定義されている場合 1 となり，そうでない場合は 0 となります．$#*foo* は *foo* がリストならばその要素の数に，数値や文字列の場合 1 に，論理変数の場合に 0 となります．

C シェルのヒストリ置換と同様なヒストリ置換を使えます．詳細については，C シェルのマニュアルページを参照のこと．

文字 ~, {, } は，C シェルでの働きと同じ，すなわちホームディレクトリと選択肢の展開を行ないます．ワイルドカード文字 *, ?, [,] も使えますが，まず noglob を未設定にする必要があります．こうすると数式を入力するのが困難になるので，ワイルドカードの展開が終わったら，noglob を再設定すべきでしょう．パターン [^abc] は，a, b, c 以外の文字にマッチします．

IO リダイレクションも可能です．記号 >, >>, >&, >>&, < は，C シェルと同じ働きをします．

セミコロンで区切って，複数のコマンドを 1 行でタイプすることもできます．

標準 (通常 ~cad/lib/mfbcap) 以外の mfbcap ファイルを使いたい場合，Nutmeg または Spice を開始する前に，環境変数 Spice_MFBCAP を設定する必要があります．-m オプションや mfbcap 変数はもはや動作しません．

X を使っている場合，開いているウィンドウの任意の場所にカーソルをおいて文字をキーボードからタイプすると，その文字はウィンドウのその場所に描かれます．ウィンドウは **xpr(1)** プログラムでプリンタに送ることができます．

Nutmeg は，他の OS 同様 VAX/VMS で動作させることができます．コマンド補完や *, ?, [,] の展開，逆クォート置換，シェルコマンドなどのいくつかの機能は動作しません．

システムによっては，表示の途中に -more- というプロンプトが出て，UNIX でやるのと同様に，任意のキーでなくリターンで反応しなければならないこともあります．

5.7 バグ

ラベル入力機能は，原始的なものです．ラベルを入力するときは，ゆっくりタイプしてください．Nutmeg は 1 秒ごとに入力をチェックしているので，文字が速く到着すると混乱してしまいます．

X でグラフのウィンドウを作成してから色を再定義して，ウィンドウを再表示すると，正しい色で再表示されません．

次のような別名を定義したとき，

```
alias pdb plot db( '!:1' - '!:2' )
```

このように，引数リストの置換が起きないように注意深くクォートしなければなりません．引数全体をクォートすると，正しく動作しません．

利用者定義の関数で，引数を *plot.vec* 構文の名前の一部としては使えません．たとえば，

```
define check(v(1)) cos(tran1.v(1))
```

は動作しません．

plot all all としたり，コマンドで 1 つのグラフにワイルドカードの参照を 2 回使うと，その結果は予測できません．

asciiplot コマンドは，対数軸や delta キーワードを扱えません．

MFB で認識される端末の名前は，/etc/termcap とは異なる場合があります．端末タイプを再設定するには，

```
set term = termname
```

というコマンドを使います．ここで，*termname* は mfbcap の中にある名前です．

hardcopy コマンドは，**plot(5)** 形式を理解するプログラムを持っていなければ，**plot** コマンドのない VMS や他のシステムでは無意味です．

Spice3 は Spice2 の .plot 行で使われるすべての記法を理解し，vp(1) を ph(v(1)) に翻訳します．しかし，これらの名前の中に空白があると，動作しません．したがって，v(1, 2) や (-.5, .5) は認識されません．

BJT には，3 つまたは 4 つのノードがありますが，これはサブ回路の展開ルーチンがどの名前を変えるのかを決めるのを困難にしています．4 番目のパラメータがモデル名として宣言された場合，3 つのノードがあると仮定され，そうでなければそれはノードと解釈されます．これを無効にするには，変数 "nobjthack" を設定します．こうすると (少なくともサブ回路の展開の目的に関しては) BJT に必ず 4 つのノードがあると解釈されます．

@*name*[*param*] 記法は，trace, iplot などではまだ動作しません．

(.spiceinit ファイル以外の) コマンドファイルの最初の行は，コメントとしなければなりません．そうしないと，Spice が空の回路を作ってしまうことがあります．

コマンド行で指定されたファイルは，.spiceinit の前に読み込まれます．

参考文献

[1] A. Vladimirescu and S. Liu. The simulation of MOS integrated circuits using SPICE2. *ERL Memo No. ERL M80/7, Electronics Research Laboratory, University of California, Berkeley*, October 1980.

[2] T. Sakurai and A. R. Newton. A simple MOSFET model for circuit analysis and its application to CMOS gate delay analysis and series-connected MOSFET structure. *ERL Memo No. ERL M90/19, Electronics Research Laboratory, University of California, Berkeley*, March 1990.

[3] B. J. Sheu, D. L. Scharfetter, and P. K. Ko. SPICE2 implementation of BSIM. *ERL Memo No. ERL M85/42, Electronics Research Laboratory, University of California, Berkeley*, May 1985.

[4] J. R. Pierret. A MOS parameter extraction program for the BSIM model. *ERL Memo Nos. ERL M84/99 and M84/100, Electronics Research Laboratory, University of California, Berkeley*, November 1984.

[5] Min-Chie Jeng. Design and modeling of deep-submicrometer MOSFETSs. *ERL Memo Nos. ERL M90/90, Electronics Research Laboratory, University of California, Berkeley*, October 1990.

[6] Soyeon Park. Analysis and SPICE implementation of high temperature effects on MOSFET. Master's thesis, University of California, Berkeley, December 1986.

[7] Clement Szeto. Simulator of temperature effects in MOSFETs (STEIM). Master's thesis, University of California, Berkeley, May 1988.

[8] J. S. Roychowdhury and D. O. Pederson. Efficient transient simulation of lossy interconnect. In *Proc. of the 28th ACM/IEEE Design Automation Conference*, San Francisco, June 17–21 1991.

[9] A. E. Parker and D. J. Skellern. An improved FET model for computer simulators. *IEEE Trans CAD*, 9(5):551–553, May 1990.

[10] Simulation and modeling. In R. Saleh and A. Yang, editors, *IEEE Circuits and Devices*, volume 8(3), pages 7–8 and 49. IEEE, May 1992.

[11] H. Statz et al. GaAs FET device and circuit simulation in SPICE. *IEEE Transactions on Electron Devices*, 34(2):160–169, February 1987.

A　回路の例

A.1　回路 1: 差動ペア

以下の入力ファイルは，単純な差動ペアの動作点を決定します．さらに，1 Hz から 100 MHz の範囲に渡って交流小信号応答が計算されます．

```
SIMPLE DIFFERENTIAL PAIR
VCC  7  0    12
VEE  8  0    -12
VIN  1  0    AC 1
RS1  1  2    1K
RS2  6  0    1K
Q1   3  2  4 MOD1
Q2   5  6  4 MOD1
RC1  7  3    10K
RC2  7  5    10K
RE   4  8    10K
.MODEL MOD1 NPN BF=50 VAF=50 IS=1.E-12 RB=100 CJC=.5PF TF=.6NS
.TF V(5) VIN
.AC DEC 10 1 100MEG
.END
```

A.2　回路 2: MOSFET の特性

以下の入力ファイルは，MOSFET 素子の出力特性を，V_{DS} については 0–10 V の範囲，V_{GS} については 0–5 V の範囲で計算します．

```
MOS OUTPUT CHARACTERISTICS
.OPTIONS NODE NOPAGE
VDS  3  0
VGS  2  0
M1   1  2  0  0 MOD1 L=4U W=6U AD=10P AS=10P
* VIDS MEASURES ID, WE COULD HAVE USED VDS, BUT ID WOULD BE NEGATIVE
VIDS 3  1
.MODEL MOD1 NMOS VTO=-2 NSUB=1.0E15 UO=550
.DC VDS 0 10 .5 VGS 0 5 1
.END
```

A.3　回路 3: RTL インバータ

以下の入力ファイルは，単純な RTL インバータの直流伝達特性と過渡パルス応答を決定します．入力は，0 V から 5 V へのパルスで，遅延時間，立ち上がり時間，立ち下がり時間は 2 ns，パルス幅は 30 ns です．過渡解析の区間は 0 から 100 ns で，1 ナノ秒ごとに表示されます．

```
SIMPLE RTL INVERTER
VCC  4  0    5
VIN  1  0    PULSE 0 5 2NS 2NS 2NS 30NS
RB   1  2    10K
Q1   3  2  0 Q1
RC   3  4    1K
.MODEL Q1 NPN BF 20 RB 100 TF .1NS CJC 2PF
.DC VIN 0 5 0.1
.TRAN 1NS 100NS
.END
```

A.4　回路 4: 4 ビット 2 進加算器

以下の入力ファイルは，全体の回路の一部を記述するサブ回路を使って，4 ビット 2 進加算器をシミュレートします.

```
ADDER - 4 BIT ALL-NAND-GATE BINARY ADDER

*** SUBCIRCUIT DEFINITIONS
.SUBCKT NAND 1 2 3 4
*   NODES:  INPUT(2), OUTPUT, VCC
Q1        9  5  1 QMOD
D1CLAMP   0  1    DMOD
Q2        9  5  2 QMOD
D2CLAMP   0  2    DMOD
RB        4  5    4K
R1        4  6    1.6K
Q3        6  9  8 QMOD
R2        8  0    1K
RC        4  7    130
Q4        7  6 10 QMOD
DVBEDROP 10  3    DMOD
Q5        3  8  0 QMOD
.ENDS NAND

.SUBCKT ONEBIT 1 2 3 4 5 6
*   NODES:  INPUT(2), CARRY-IN, OUTPUT, CARRY-OUT, VCC
X1   1  2  7  6   NAND
X2   1  7  8  6   NAND
X3   2  7  9  6   NAND
X4   8  9 10  6   NAND
X5   3 10 11  6   NAND
X6   3 11 12  6   NAND
X7  10 11 13  6   NAND
X8  12 13  4  6   NAND
X9  11  7  5  6   NAND
.ENDS ONEBIT

.SUBCKT TWOBIT 1 2 3 4 5 6 7 8 9
*   NODES:  INPUT - BIT0(2) / BIT1(2), OUTPUT - BIT0 / BIT1,
*           CARRY-IN, CARRY-OUT, VCC
X1   1  2  7  5 10  9   ONEBIT
X2   3  4 10  6  8  9   ONEBIT
.ENDS TWOBIT

.SUBCKT FOURBIT 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
*   NODES:  INPUT - BIT0(2) / BIT1(2) / BIT2(2) / BIT3(2),
*           OUTPUT - BIT0 / BIT1 / BIT2 / BIT3, CARRY-IN, CARRY-OUT, VCC
X1   1  2  3  4  9 10 13 16 15   TWOBIT
X2   5  6  7  8 11 12 16 14 15   TWOBIT
.ENDS FOURBIT

*** DEFINE NOMINAL CIRCUIT
.MODEL DMOD D
.MODEL QMOD NPN(BF=75 RB=100 CJE=1PF CJC=3PF)
VCC   99  0   DC 5V
VIN1A  1  0   PULSE(0 3 0 10NS 10NS   10NS   50NS)
VIN1B  2  0   PULSE(0 3 0 10NS 10NS   20NS  100NS)
VIN2A  3  0   PULSE(0 3 0 10NS 10NS   40NS  200NS)
```

```
VIN2B  4  0   PULSE(0 3 0 10NS 10NS   80NS  400NS)
VIN3A  5  0   PULSE(0 3 0 10NS 10NS  160NS  800NS)
VIN3B  6  0   PULSE(0 3 0 10NS 10NS  320NS 1600NS)
VIN4A  7  0   PULSE(0 3 0 10NS 10NS  640NS 3200NS)
VIN4B  8  0   PULSE(0 3 0 10NS 10NS 1280NS 6400NS)
X1     1  2  3  4  5  6  7  8  9 10 11 12  0 13 99 FOURBIT
RBIT0  9  0   1K
RBIT1 10  0   1K
RBIT2 11  0   1K
RBIT3 12  0   1K
RCOUT 13  0   1K

*** (FOR THOSE WITH MONEY (AND MEMORY) TO BURN)
.TRAN 1NS 6400NS
.END
```

A.5 回路 5: 伝送線路インバータ

以下の入力ファイルは，伝送線路インバータをシミュレートします．2 つの伝送モードが励起されるので，2 つの伝送線路の要素が必要です．同軸線路の場合，最初の線路 (T1) は，シールドに対する内部導体をモデル化し，2 番目の線路 (T2) は，外界に対するシールドをモデル化します．

```
TRANSMISSION-LINE INVERTER
V1   1  0         PULSE(0 1 0 0.1N)
R1   1  2         50
X1   2  0  0  4   TLINE
R2   4  0         50

.SUBCKT TLINE 1 2 3 4
T1   1  2  3  4   Z0=50 TD=1.5NS
T2   2  0  4  0   Z0=100 TD=1NS
.ENDS TLINE

.TRAN 0.1NS 20NS
.END
```