

線形回路網シミュレーションプログラム (s i m. e x e) マニュアル

1 9 9 2 / 0 8 / 2 5 三浦 高志

2020/04/10 VS 6.0 で 32bit 版にリビルド

```

素子属性          アナログ回路シミュレーションプログラムVer3.0
独立素子 : e      i      r      l      c
従属素子 : k...i to i  v...e to e  q...e to i  w...i to e
ブロック素子 : b      関数素子 : f

コマンド : / + コマンド名
ipart  cpart  dpart  padd  node  parts  free  dir  help
range  point  para  ac  disp  conv  auto  manu
cmode  smode  mode  db  val  mk  nomk  gpos  bmode
cal  save  load  bload  fname  def  dvar  dfnc  end
DOS コマンド : ! + MS-DOS コマンド名
バッチファイル実行 : @ + ファイル名 .... コメント行は % で始める
バッチファイル作成・終了 : #          キー入力待ちは /wait
                                   中断の指示は /exit
計算式で使用可能な因子          巻戻しは /rewind, スキップは /skip
$a      j(式)      ^ (式)  abs  sqrt  arg  log  log10  db
exp      udb      sin  cos  tan  asin  acos  atan
real     image  pow  unpw  hex  khex  mhex

システム変数
f fl fh fs s pi
    
```

```

? /range
データファイル名は ? s21-1
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 500
range 周波数 最高値 ? 1500
range 周波数 ステップ ? 50
e1[1      j 0      ]
f[500      ]
x5 [6.249430e-005 j -1.310620e-005] DB[ -83.8963] arg[ -11.8443]
f[550      ]
x5 [1.264000e-004 j -3.189178e-005] DB[ -77.6970] arg[ -14.1607]
f[600      ]
x5 [2.629061e-004 j -8.073583e-005] DB[ -71.2126] arg[ -17.0712]
f[650      ]
x5 [5.735657e-004 j -2.187432e-004] DB[ -64.2386] arg[ -20.8756]
f[700      ]
x5 [ 0.0013 j -6.586564e-004] DB[ -56.4967] arg[ -26.1066]
f[750      ]
x5 [ 0.0035 j -0.0023 ] DB[ -47.5630] arg[ -33.8111]
f[800      ]
x5 [ 0.0101 j -0.0106 ] DB[ -36.7113] arg[ -46.3547]
f[850      ]
x5 [ 0.0251 j -0.0704 ] DB[ -22.5293] arg[ -70.3469]
f[900      ]
x5 [ -0.5236 j -0.5852 ] DB[ -2.1004] arg[ -131.8198]
f[950      ]
x5 [ 8.9236 j 0.7883 ] DB[ 19.0446] arg[ 5.0485]
f[1,000      ]
x5 [ -0.7610 j 0.6253 ] DB[ -0.1317] arg[ 140.5927]
f[1,050      ]
x5 [ 0.0226 j 0.1132 ] DB[ -18.7576] arg[ 78.7272]
f[1,100      ]
x5 [ 0.0162 j 0.0228 ] DB[ -31.0791] arg[ 54.6583]
    
```

目次

目次.....	2
機能概要	4
使用できる単位.....	5
使用出来る回路素子(素子属性名 と 素子の種類).....	5
登録されているコマンドの種類.....	6
関数電卓の関数名と因子.....	6
シミュレーション用コマンドの解説 (／に続けてコマンド名を入力する).....	7
f a c t o r : 因子	10
システム変数 :	11
各コマンドの詳細説明と操作方法.....	12
(1) /ac	12
(2) /auto	13
(3) /bload.....	13
(4) /bmode.....	15
(5) /cal	16
(6) /cmode	17
(7) /conv	17
(8) /cpart.....	19
(9) /db.....	19
(10) /def.....	19
(11) /dfnc.....	20
(12) /dir.....	20
(13) /disp.....	20
(14) /dpart.....	20
(15) /dvar	21
(16) /end.....	21
(17) /fname	21
(18) /free	22
(19) /gpos	22
(20) /grph.....	22
(21) /help.....	23
(22) /ipart	23
(23) /load.....	25
(24) /manu	25

(25) /mode.....	25
(26) /node.....	26
(27) /padd	26
(28) /para.....	26
(29) /parts.....	28
(30) /point.....	28
(31) /range	28
(32) /rewind.....	30
(33) /save	30
(34) /skip.....	32
(35) /smode.....	32
(36) /val.....	32
電卓としての利用法：	33
サンプルデータファイルの内容.....	34
バッチファイルのサンプル	49
Sim.exe で使用する標準部品として準備してあるブロック素子について.....	64
サンプルバッチファイルの回路図と説明	68
(1) バッチファイル名 s 5. s b で扱う回路図.....	68
(2) バッチファイル名 s 6. s b で扱う回路図.....	77
(3) バッチファイル名 s 7. s b で扱う回路図.....	82
(4) バッチファイル名 s 8. s b で扱う回路図.....	87
(5) バッチファイル名 s 9. s b で扱う回路図.....	91
(6) バッチファイル名 s 10. s b で扱う回路図	93
(7) バッチファイル名 s 11. s b で扱う回路図	96
(8) バッチファイル名 s s 1. s b で扱う回路図	100
(9) バッチファイル名 s s 2. s b で扱う回路図	102
(10) バッチファイル s 20. s b で扱う 3つの回路図	105

機能概要

1. 線形回路網の入力及び定常状態シミュレーションを実行可能
(グラウンドをノード0として、ノードiとノードjに接続されている部品をすべて登録する)
2. 線型回路網のネットリストをファイルとして入出力可能
(ファイル名及び回路網の入出力ノードの番号を同時に記録する)
3. 線形回路網を機能ブロックとしてファイルに入出力が可能
4. 新たな回路網を入力する時に、ファイルとして作成されている機能ブロックをR, L, Cなどの部品と同様に使用することが可能
5. 入力された回路網の回路方程式を作成し、画面表示やファイルに保存が可能
(ファイル名はconv.dとなる)
6. 作成された回路方程式のファイルconv.dは、Mapleに読み込んで、数式処理により伝達関数を求めることが出来る
7. 操作した内容をバッチファイルとしてファイルに保存する事が出来る
8. 操作内容のバッチファイルを読み込みながら、処理を再現する事が出来る
9. バッチファイルから他のバッチファイルにジャンプすることが出来る
10. バッチファイル実行中に入力を受け付け、バッチファイルの巻き戻し、バッチファイルの最後へのジャンプ、バッチファイルの実行停止などを指定出来る
11. 複素数電卓として使用する事が出来る
(演算処理に名前をつけて、手続きを登録する事が出来る)
12. プログラムに対する入力は、小文字でも大文字でも入力可能
13. ネットリスト入力の際に素子の値は、通常の電子工学で使用されている単位が使用出来る。

抵抗が1Kオームのときは 1k (あるいは1K) と入力が可能。

同様に1Mオームなら、1m (1M) 、

1ミリオームは0.001または1/1k と入力する。

1000Mオームなら、1g (1G)

コンデンサで0.1uFなら、0.1u (0.1U)

0.001uFなら、1n (1N)

120pFなら、120p (120P)

使用できる単位

k	...	10^3
m	...	10^6
g	...	10^9
u	...	10^{-6}
n	...	10^{-9}
p	...	10^{-12}

14。コマンド `/i p a r t`, `/p a d d`, `/p a r a` 及び `/a c` において、素子の値などを数式で入力することが可能。

例えば、規格表にトランジスタのコレクタ抵抗がアドミタンスで記載されていた場合 (50 u S となっていた時)

$1/50\text{ u}$ と入力すればよい

また、 $r e * (1 + k)$ の様に、変数名も使用可能である

使用出来る回路素子(素子属性名 と 素子の種類)

e	独立電圧源	i	独立電流源
r	抵抗素子	c	コンデンサ素子
l	インダクタンス素子		
v	電圧従属電圧源	q	電圧従属電流源
w	電流従属電圧源	k	電流従属電流源
b	ブロックとして登録された素子		
f	関数(数式)として登録された素子(現在は対応していない)		

登録されているコマンドの種類

(1) シミュレーション用コマンド： / + コマンド名

ipart	cpart	dpart	padd	node	parts	free	help	
range	point	para	ac	grph	disp	conv	auto	manu
cmode	smode	mode	db	val	mk	nomk	gpos	bmode
cal	save	load	bload	fname	def	dvar	dfnc	end

注意： grph コマンドは VS 6.0 でビルドした時に削除した。2020/04/10

(2) DOS コマンドの実行： ! + MS-DOS コマンド名

(3) バッチファイルの作成及び終了：

キー入力待ちは /wait

コメント行は %% で始める

(4) バッチファイルの実行： @ + ファイル名

バッチファイルの巻き戻しは /rewind

バッチファイルの最後にジャンプは /skip

バッチファイルの終了は /exit

関数電卓の関数名と因子

\$a	j(式)	^(式)	abs	sqrt	arg	log	log10	db
exp	udb	sin	cos	tan	asin	acos	atan	
real	image	pow	unpw	hex	khex	mhex		

システム変数

f	fl	fh	fs	s	pi
---	----	----	----	---	----

シミュレーション用コマンドの解説（／に続けてコマンド名を入力する）

- (1) `i p a r t` 回路情報（ネットリスト）を初期化して新規に回路情報を入力する。 `/ c p a r t`に続けて、`/ p a d d`を行なうのと同等の機能
- (2) `c p a r t` 回路情報を初期化する
- (3) `d p a r t` 回路情報を表示する
- (4) `p a d d` 回路情報に素子を追加する
- (5) `n o d e` 回路情報で使用されている最大ノード番号を表示する
- (6) `p a r t s` 回路情報で使用されている素子の総数を表示する
- (7) `f r e e` 使用可能なメモリの残量を表示する（あまり意味がない）
- (8) `h e l p` ヘルプ画面を表示する（コマンド名などの確認が出来る）
- (8) `r a n g e` 周波数特性を計算して、指定された周波数ステップごとに指定ノードの電圧値を表示する
- (9) `p o i n t` システム変数 `f` の値における指定ノードの電圧を計算する
- (10) `p a r a` 素子名を指定して、素子値を変化させたときの指定ノードの電圧を計算する。周波数は `f` の値で固定される。
すべての独立電圧源及び独立電流源は与えられた値として計算される。
- (11) `a c` `/ p a r a`によって動作点を決定後使用する
入力信号名と値を決定し、変化させたい素子名と変化の範囲を入力する。回路中の他の独立電圧源及び独立電流源を0として指定ノードの電圧を計算する
- (12) `g r p h` 指定されたデータファイルを読み込んで、周波数特性などのグラフを表示する → 削除 2020/04/10

- (13) `d i s p` 計算するノード番号を指定する
- (14) `c o n v` 出力先を指定すると、回路情報から数式による係数行列（回路方程式の係数）を作成して出力する
- (15) `s a v e` 回路情報にファイル名と接続ノード番号をつけて保存する
- (16) `l o a d` 指定されたファイルから回路情報を読み込む
- (17) `b l o a d` 機能ブロックのファイル（ブロック素子）を展開する
- (18) `f n a m e` / `l o a d` で読み込んだファイル名を表示する
- (19) `d e f` 関数電卓で使う手続きを定義する、手続き名は15文字まで
- (20) `d v a r` 使用されている変数名（`a 3 b` 等も可）と値を表示する
- (21) `d f n c` 定義されている手続き名と式を表示する
- (22) `e n d` `s i m. e x e` を終了してMS-DOSへ戻る
- (23) `c m o d e` モード設定クリア、計算は値で行い、ファイルを作らない
- (24) `s m o d e` モード設定セット、計算はdBで行い、ファイルを作る
- (25) `m o d e` 現在のモードの設定状態を表示する
- (26) `d b` 計算モードをdBにする
- (27) `v a l` 計算モードを値にする
- (28) `m k` ファイルを作成にするモードに切り替える
- (29) `n o m k` ファイルを作らないモードに切り替える

(30) `m k` `/range, /para, /ac` において、計算結果をファイルとして保存するモードに切り替える。
作成されたファイルは `/grph` でグラフ表示に使われる。

! に続けてMS-DOSコマンドを入力して実行する

`! d i r` や `! t y p e` の様に入力する、`v z` エディタなどプログラムサイズの小さなものは実行可能だが、大きなものは実行出来ない

を入力してバッチファイルを作成する

を入力すると、作成するバッチファイル名を聞いてくるので、入力する。次に、コマンドラインから # を入力するまでの間に実行したコマンドがファイルとして保存される。拡張子 `. s b` が自動的に付けられる。例えば、周波数特性を計算するときなど、入力信号源名、値、最低周波数、最高周波数、周波数ステップなど決まっている入力を何度も繰り返す時などはこれらをバッチファイルにしておくとう便利である。
バッチファイル作成中に `/wait` と入力しておくとう、
`@ +` ファイル名で実行する時に処理を一時停止させる事が出来る。
`/exit` と入力しておくとう、バッチファイルの実行を中断するので、バッチファイルの実行を1命令ごとに確認したいときに使用できる。
バッチファイルはネスティングが10レベルまで可能になっている。
ネスティングしたバッチファイルを作成するには、
コマンドでネスティングしていないバッチファイルを必要な個数作成した後、`s i m. e x e` を終了して、エディタでファイルを修正する。`s i m. e x e` の中から、`! v z b a t x` のようにエディタを起動してこれを行なうことも可能である。

@ に続けて、バッチファイル名を入力してバッチファイルを実行する

コマンドラインから `@ b a t 1` のように入力すると、`b a t 1. s b` をロードしてきて
キーボード入力の代わりに、ファイルから1行づつ読んできて、自動実行する。
実行が速過ぎる場合には、エディタを使用して適当な数コマンドごとに、
`/wait` を挿入しておくとう確認しながら実行が可能となる。

f a c t o r : 因子

(1) \$ に続けて関数名 (\$ a など) / d e f で定義した手続き (式) を実行して表示する

(2) j に続けて変数名 (j b 2 など) または、j (式) j は虚数単位の意味

(3) ~ に続けて変数名 (~ c など) または、~ (式) ~ は共役複素数の意味

(3) a b s (式) 絶対値 (式は複素数でも可)

(4) s p r t (式) 平方根 (式は複素数でも可)

(5) a r g (式) 式が $a + j b$ という値になる時の
偏角 ($\text{atan}(b/a)$), 単位は度 (degree)

(6) l o g (式) 自然対数 (式は複素数でも可)

(7) l o g 1 0 (式) 常用対数 (式は複素数でも可)

(8) d b (式) 式の値をデシベル値に変換する

(9) e x p (式) 自然対数の底によるべき乗 (複素数対応)

(10) u d b (式) 式の値をデシベル値から倍率に変換する

(11) s i n (式) 式の値 (単位は d e g r e e) の正弦

(12) c o s (式) 式の値の余弦
 $\text{c o s } (45) = 0.707$

(13) t a n (式) 式の値の正接

(14) a s i n (式) 式の値の逆正弦

(15) a c o s (式) 式の値の逆余弦

(16) a t a n (式)	式の値の逆正接 $a t a n (1) = 45$
(17) r e a l (式)	式の値の実数部
(18) i m a g e (式)	式の値の虚数部
(19) p o w (式1、式2)	式1の式2乗 $p o w (2, 3) = 8$
(20) u n p w (b、c)	$b = c ^ d$ となる、dを求める。
(21) h e x (式)	式の値を16進数から10進数に変換する
(22) k h e x (式)	式の値をキロ単位の16進数から10進数に変換する
(23) m h e x (式)	式の値をメガ単位の16進数から10進数に変換する

システム変数：

f	周波数特性などの計算に使用する周波数値
f l	／ r a n g e で使用する周波数範囲の最低値
f h	／ r a n g e で使用する周波数範囲の最高値
f s	／ r a n g e で使用する周波数の上昇ステップ値
s	$j 2 * p i * f$ を格納するための変数 (角周波数)
p i	円周率の値 3. 1 4 1 5 9 6

各コマンドの詳細説明と操作方法

(1) /ac

システム変数 f で指定される周波数において、指定の素子を希望の範囲で変化させた時の交流信号の振幅を計算する。入力信号源と指定された素子以外の電圧源は 0 として交流成分だけの計算を行なう。ファイルを作成するモードの場合 (/mk) には指定されたノードの電圧値の絶対値をファイルにセーブする。

計算ステップについては、マイナスも可であり、

a /auto モードかつ 最高値/最低値の比が 10 以下の時は、
入力されたステップ値で計算を行なう。

b /manu モードまたは 最高値/最低値の比が 10 を超える時は、
/gps で設定された計算ポイント数で最低値から最高値まで
等比級数で増加する様にステップ値を自動的に変化させる。

/para と /ac を交互に実行し、直流動作点と交流利得の初期設定を行ない、
次に /range によって使用する全周波数範囲における特性を計算する。

```
B? /ac
ac 入力信号源名は ? e1
値は ? a

値を変化させる素子名は ? r1
ac r1 最低値 ? 3k
ac r1 最高値 ? 4k
ac r1 ステップ ? 100
e1[0.2209 j 0]
r1[3000] f[1,000]
x5 [-44.2770 j -0.8588] abs[ 44.2853] arg[ -178.8888]
x6 [-44.2769 j -0.8611] abs[ 44.2853] arg[ -178.8858]
r1[3100] f[1,000]
x5 [-45.0898 j -0.8746] abs[ 45.0983] arg[ -178.8888]
x6 [-45.0897 j -0.8769] abs[ 45.0983] arg[ -178.8858]
r1[3200] f[1,000]
x5 [-45.8794 j -0.8900] abs[ 45.8880] arg[ -178.8887]
x6 [-45.8793 j -0.8923] abs[ 45.8880] arg[ -178.8858]
r1[3300] f[1,000]
x5 [-46.6467 j -0.9049] abs[ 46.6555] arg[ -178.8886]
x6 [-46.6467 j -0.9072] abs[ 46.6555] arg[ -178.8859]
r1[3400] f[1,000]
x5 [-47.3927 j -0.9195] abs[ 47.4017] arg[ -178.8886]
x6 [-47.3927 j -0.9217] abs[ 47.4017] arg[ -178.8859]
r1[3500] f[1,000]
x5 [-48.1183 j -0.9336] abs[ 48.1274] arg[ -178.8885]
x6 [-48.1183 j -0.9358] abs[ 48.1274] arg[ -178.8859]
r1[3600] f[1,000]
x5 [-48.8243 j -0.9473] abs[ 48.8335] arg[ -178.8884]
x6 [-48.8243 j -0.9495] abs[ 48.8335] arg[ -178.8859]
r1[3700] f[1,000]
x5 [-49.5115 j -0.9607] abs[ 49.5208] arg[ -178.8884]
x6 [-49.5114 j -0.9628] abs[ 49.5208] arg[ -178.8859]
r1[3800] f[1,000]
x5 [-50.1806 j -0.9737] abs[ 50.1900] arg[ -178.8883]
x6 [-50.1805 j -0.9758] abs[ 50.1900] arg[ -178.8859]
r1[3900] f[1,000]
x5 [-50.8322 j -0.9864] abs[ 50.8418] arg[ -178.8883]
x6 [-50.8322 j -0.9885] abs[ 50.8418] arg[ -178.8860]
r1[4000] f[1,000]
x5 [-51.4672 j -0.9988] abs[ 51.4769] arg[ -178.8883]
x6 [-51.4672 j -1.0008] abs[ 51.4769] arg[ -178.8860]
```

/mk の場合

```
? /ac
データファイル名は ? test
ac 入力信号源名は ? e1
値は ? 1

値を変化させる素子名は ? r1
ac r1 最低値 ? 3k
ac r1 最高値 ? 4k
ac r1 ステップ ? 100
e1[1] j 0 ]
r1[3000] f[1,000] ]
x5 [-452.5988 j -8.8132] abs[ 452.6846] arg[ -178.8845]
x6 [-452.5988 j -8.8132] abs[ 452.6846] arg[ -178.8844]
r1[3100] f[1,000] ]
x5 [-452.5988 j -8.8124] abs[ 452.6846] arg[ -178.8846]
x6 [-452.5988 j -8.8125] abs[ 452.6846] arg[ -178.8845]
r1[3200] f[1,000] ]
x5 [-452.5988 j -8.8117] abs[ 452.6846] arg[ -178.8846]
x6 [-452.5988 j -8.8117] abs[ 452.6846] arg[ -178.8846]
r1[3300] f[1,000] ]
x5 [-452.5989 j -8.8110] abs[ 452.6846] arg[ -178.8847]
x6 [-452.5989 j -8.8110] abs[ 452.6846] arg[ -178.8847]
r1[3400] f[1,000] ]
x5 [-452.5989 j -8.8103] abs[ 452.6846] arg[ -178.8848]
x6 [-452.5989 j -8.8104] abs[ 452.6846] arg[ -178.8848]
r1[3500] f[1,000] ]
x5 [-452.5989 j -8.8097] abs[ 452.6846] arg[ -178.8849]
x6 [-452.5989 j -8.8098] abs[ 452.6846] arg[ -178.8849]
r1[3600] f[1,000] ]
x5 [-452.5989 j -8.8092] abs[ 452.6846] arg[ -178.8850]
x6 [-452.5989 j -8.8092] abs[ 452.6846] arg[ -178.8850]
r1[3700] f[1,000] ]
x5 [-452.5989 j -8.8086] abs[ 452.6846] arg[ -178.8850]
x6 [-452.5989 j -8.8087] abs[ 452.6846] arg[ -178.8850]
r1[3800] f[1,000] ]
x5 [-452.5989 j -8.8081] abs[ 452.6846] arg[ -178.8851]
x6 [-452.5989 j -8.8082] abs[ 452.6846] arg[ -178.8851]
r1[3900] f[1,000] ]
x5 [-452.5989 j -8.8076] abs[ 452.6846] arg[ -178.8852]
x6 [-452.5989 j -8.8077] abs[ 452.6846] arg[ -178.8852]
r1[4000] f[1,000] ]
x5 [-452.5989 j -8.8072] abs[ 452.6846] arg[ -178.8852]
x6 [-452.5989 j -8.8072] abs[ 452.6846] arg[ -178.8852]
```

(2) /auto

上記シミュレーションコマンド/range, /para, /ac で計算ステップを最高値／最低値の比の値によって自動的に切り替えるモードに設定する。

比の値が10以下であれば、入力されたステップ値で計算を行ない、10を超えていれば、等比級数で増加する様なステップ値で計算を行なう。

(3) /bload

ブロック素子ファイルを読み込んで解凍する

回路情報の中にブロック素子が使用されている時には、/bloadによって

回路を展開しなければ/rangeなどで計算を行なうことができない。

ブロック素子のネスティングは許されている。

```
? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r1 ] value[ 1e+004 j 0 ]
left[ 0] right[ 3] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[ 4] parts[ ce ] value[ 0.001 j 0 ]
left[ 0] right[ 4] parts[ re ] value[ 2200 j 0 ]
left[ 0] right[ 6] parts[ r1 ] value[ 1000 j 0 ]
left[ 1] right[ 2] parts[ cc ] value[ 0.001 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 5e+004 j 0 ]
left[ 2] right[ 5] parts[ b1 ] value[ 0 j 0 ]
@ func trn:2,4,5
left[ 3] right[ 5] parts[ rc ] value[ 3800 j 0 ]
left[ 5] right[ 6] parts[ cc ] value[ 0.001 j 0 ]
最大の素子番号 = 11
最大のノード番号 = 6
未解凍のブロックを 1 個含んでいます
?
```

```
? /bload
bname b1 sfnc trn:2,4,5

func trn:2,4,5 name trn.cir blk b1
@ func trn:1,2,3

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
i[ 3] j[ 0] 部品名[e2] 値[24 j 0]
i[10] j[ 8] 部品名[ebb1] 値[0.6 j 0]
独立電圧源素子の個数 3
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
1 2 3 4 5 6

表示ノード番号 (0 なら全て、-1 ならこのまま) ? 5
ノード番号 (0 なら終了) ?

使用した式の個数 0、密度 0.000 %
使用した式のサイズ 463
最大の素子番号 = 16
最大のノード番号 = 10
```

/bload 後の接続情報

```
? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r1 ] value[ 1e+004 j 0 ]
left[ 0] right[ 3] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[ 4] parts[ ce ] value[ 0.001 j 0 ]
left[ 0] right[ 4] parts[ re ] value[ 2200 j 0 ]
left[ 0] right[ 6] parts[ rl ] value[ 1000 j 0 ]
left[ 1] right[ 2] parts[ cc ] value[ 0.001 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 5e+004 j 0 ]
left[ 2] right[ 10] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 3] right[ 5] parts[ rc ] value[ 3800 j 0 ]
left[ 4] right[ 7] parts[ rbb1 ] value[ 0.1 j 0 ]
left[ 5] right[ 6] parts[ cc ] value[ 0.001 j 0 ]
left[ 5] right[ 9] parts[ rbb1 ] value[ 0.1 j 0 ]
left[ 7] right[ 8] parts[ reb1 ] value[ 26 j 0 ]
left[ 7] right[ 9] parts[ kb1 ] value[ 100 j 0 ]
@ master[reb1 ] left[ 7] right[ 8]
left[ 8] right[ 10] parts[ ebb1 ] value[ 0.6 j 0 ]
最大の素子番号 = 16
最大のノード番号 = 10
? _
```

ブロック素子を読み込むと、最大のノード番号が 6 から 10 増加した。

(4) /bmode

回路リストあるいは回路リストのファイルでブロックを使用している時に、ブロック名を ? で始めると、/bload でブロックを解凍する時に、キーボードから任意のブロックファイル名を入力可能となっている。

ブロック名入力でリターンだけを入力すると、回路リストに入力されているブロック名を使用する。

/bmode コマンドを入力して、1 を入力すると、? で始めていないブロックファイル名であっても、キーボードから入力するようになる。/bmode で 0 を入力した場合は? で始まるブロック名だけがキーボードからファイル名の再指定が可能となる。

/ipart, /padd で、ブロック素子を入力する時場合には、接続情報を入力するが、

trn : 1, 2, 3 と通常は入力する所を、
? : 1, 2, 3 あるいは ?trn : 1, 2, 3 と入力すると、

実際にロードするブロックファイル名をキーボードから入力出来るようになる。

／b m o d e で 1 を入力してあれば、? がなくても、すべてのブロックファイル名をキーボードから入力して、目的に合ったブロックファイル名を指定出来るようになる。

(5) /cal

/range, /para, /ac, /point などのシミュレーションコマンドにおいて、/disp では指定された表示ノードの電圧値をそのままあるいはデシベル値として計算している。

従って、抵抗 R 1 に流れる電流値を計算してグラフ表示することが出来ない。同様に、

抵抗 R 1 で消費される電力値のグラフ表示も出来ない。

このような場合には、コマンド/calにより、上記の要求が満たされる。

もし、単純に電圧値ではなく電流や電力を計算したい時には次の様に設定を行う。

1 /c a l

現在のモードが表示される

ここで 1 を入力する

2 /d i s p

現在の計算式が表示される（最初は何も表示されない）

ここで、リターンキーを入力する

1 番目の計算式を入力して下さい と表示されるので式を入力する
ノードの電圧値は x に ノード番号をつけて、x 3 の様に入力する。

$(x 5 - x 4) / r 2$ 抵抗 R 2 がノード 4 と 5 の間に接続
されていて、抵抗に流れる電流を計算する時

d b (x 8) ノード 8 の電圧をデシベル値に換算したい時
a b s (x 8) ノード 8 の電圧値の絶対値を計算したい時
x 5 - x 4 ノード 4 と ノード 5 の電位差を計算したい時

計算式には、 $\sin(\cdot)$ 、 $\log(\cdot)$ 、などの `sim.exe` で定義されているすべての関数を使用することが出来る。

- 3 `/cal` のモード設定を 1 にすると、
現在設定されている `/disp` で設定した表示ノード番号は無効となり、
`/db` や `/val` も無視される。

`/cal` のモードを 0 に戻すと、
`/disp` で設定した表示ノード番号は再び有効となり、
`/db` や `/val` も有効となる。
`/ac` と `/range` では、ノード電圧の絶対値が使われる。

(6) `/cmode`

mode をクリアする

```
? /cmode
データファイルを作成しない..... /mk または /nomk で切り替える
値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /auto (自動切り替え) です
..... /auto または /manu で切り替える
等比級数のポイント数は 32 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
1 2 3 4 5 6
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

(7) `/conv`

`/save` は回路の接続情報を保存するが、`/conv` は接続情報を回路方程式の係数行列に変換して出力する。回路の各ノード間の抵抗情報が文字式の状態で保存されている。

この係数行列のデータを数式処理ソフト Maple に入力すると、回路方程式を解くことが出来る。

```
? /conv
係数行列の出力先を選択して下さい
プリンター P , ファイル(conv.d) F , 画面 C f
```

出力された係数行列ファイル Maple 用

```
restart: with(linalg):
siki:=matrix(10,10,0):e:=vector(10,0):
siki[ 1, 1] := 1 ;
e[1] := +e1 ;
siki[ 2, 1] := -s*cc ;
```

```

siki[ 2, 2] := +1/r1+s*cc+1/r2+1/r1b1 ;
siki[ 2, 3] := -1/r2 ;
siki[ 2, 10] := -1/r1b1 ;
siki[ 3, 3] := 1 ;
e[3] := +e2 ;
siki[ 4, 4] := +s*ce+1/re+1/rbb1 ;
siki[ 4, 7] := -1/rbb1 ;
siki[ 5, 3] := -1/rc ;
siki[ 5, 5] := +1/rc+s*cc+1/rbb1 ;
siki[ 5, 6] := -s*cc ;
siki[ 5, 9] := -1/rbb1 ;
siki[ 6, 5] := -s*cc ;
siki[ 6, 6] := +1/r1+s*cc ;
siki[ 7, 4] := -1/rbb1 ;
siki[ 7, 7] := +1/rbb1+1/reb1+kb1*1/reb1 ;
siki[ 7, 8] := -1/reb1-kb1*1/reb1 ;
siki[ 8, 2] := -1/r1b1 ;
siki[ 8, 7] := -1/reb1 ;
siki[ 8, 8] := +1/reb1 ;
siki[ 8, 10] := +1/r1b1 ;
siki[ 9, 5] := -1/rbb1 ;
siki[ 9, 7] := -kb1*1/reb1 ;
siki[ 9, 8] := +kb1*1/reb1 ;
siki[ 9, 9] := +1/rbb1 ;
siki[ 10, 8] := -1 ;
siki[ 10, 10] := 1 ;
e[10] := +ebb1 ;
x:=linsolve(siki,e);
e1 := 1 ;
r1 := 10000 ;
re := 2200 ;
ce := 0.001 ;
r1 := 1000 ;
e2 := 24 ;
cc := 0.001 ;
r2 := 50000 ;

```

```
rc := 3800 ;
r1b1 := 0.1 ;
rbb1 := 0.1 ;
reb1 := 26 ;
kb1 := 100 ;
ebb1 := 0.6 ;
;end;
```

変換終了

使用した式の個数 30、 密度 30.000 %

使用した式のサイズ 463

/dpart による接続情報

```
? /dpart
left[ 0] right[ 1] parts[ e1          ] value[      1 j 0          ]
left[ 0] right[ 2] parts[ r1          ] value[ 1e+004 j 0          ]
left[ 0] right[ 3] parts[ e2          ] value[      24 j 0          ]
left[ 0] right[ 4] parts[ ce          ] value[    0.001 j 0          ]
left[ 0] right[ 4] parts[ re          ] value[    2200 j 0          ]
left[ 0] right[ 6] parts[ r1         ] value[    1000 j 0          ]
left[ 1] right[ 2] parts[ cc          ] value[    0.001 j 0          ]
left[ 2] right[ 3] parts[ r2          ] value[ 5e+004 j 0          ]
left[ 2] right[10] parts[ r1b1        ] value[      0.1 j 0          ]
left[ 3] right[ 5] parts[ rc          ] value[    3800 j 0          ]
left[ 4] right[ 7] parts[ rbb1        ] value[      0.1 j 0          ]
left[ 5] right[ 6] parts[ cc          ] value[    0.001 j 0          ]
left[ 5] right[ 9] parts[ rbb1        ] value[      0.1 j 0          ]
left[ 7] right[ 8] parts[ reb1        ] value[      26 j 0          ]
left[ 7] right[ 9] parts[ kb1         ] value[     100 j 0          ]
@ master[reb1      ] left[ 7] right[ 8]
left[ 8] right[10] parts[ ebb1        ] value[      0.6 j 0          ]
最大の 素子番号 = 16
最大のノード番号 = 10
```

(8) /cpart

回路の接続情報をクリアする。

(9) /db

デシベルで計算するモードにする。

(10) /def

自作の関数を定義する。

```
? /def
関数定義: 関数名を英数字15文字以内で入力して下さい ? a2b2
fnc(a2b2) = a*a+b*b ... 関数を入力して下さい。リターンならこのままです
a*a+b*b
fnc(a2b2) = a*a+b*b
? a=2
      2 j 0
? b=3
      3 j 0
? $a2b2
     13 j 0
```

(11) /dfnc

定義された関数名とその内容を表示する。

```
? /dfnc
No.  0 a          b*3
No.  1 a2b2       a*a+b*b
```

変数や自作の関数などは、/save で保存しておく、/load で読み込むことが出来る。

(12) /dir

/load, /load, @マクロファイル名のコマンドにおいて、
.cir及び.sbfのファイルが存在するディフォルトディレクトリを
設定、変更する。

(13) /disp

電圧を計算して表示するノードを指定する

表示ノード番号 (0 なら全て) ? と表示されるので、

/ipart, /paddの最後で行なったように、表示ノード番号を入力する。

```
? /disp
独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
i[ 3] j[ 0] 部品名[e2] 値[24 j 0]
i[10] j[ 8] 部品名[ebb1] 値[0.6 j 0]
独立電圧源素子の個数 3
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
1 2 3 4 5 6 7 8 9 10
表示ノード番号 (0 なら全て、-1 ならこのまま) ? 6
ノード番号 (0 なら終了) ?
? /range
データファイル名は ? test
ac 入力信号源名は ? e1
値は ? 1
range 周波数 最低値 ? 1000
range 周波数 最高値 ? 1000
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x6 [ -452.5992 j -8.7920 ] abs[ 452.6846] arg[ -178.8871]
```

(14) /dpart

回路の接続情報を表示する。

(15) /dvar

sim システムに記憶されているすべての変数を表示する。

```
? /dvar
No. 0 f 0 0
No. 1 fl 0 0
No. 2 fh 0 0
No. 3 fs 0 0
No. 4 s 0 0
No. 5 pi 3.1416 0
No. 6 r1b 357.4800 0
No. 7 b1 0 0
No. 8 r1a 81.215 0
No. 9 c 1.674000e-008 0
No. 10 r2 280.968 0
No. 11 v1b1 1,000,000 0
No. 12 r1b1 1,000,000 0
No. 13 b 3 0
No. 14 a 2 0
```

(16) /end

sim.exe を終了する。

(17) /fname

現在読み込んでいる接続情報ファイル名を表示する。

```
? /load
読み込み
ファイル名は ? bpf11
@ func @bpf11:0,1,4
node 4
独立電圧源素子の個数 0
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況

表示ノード番号 (0 なら全て、-1 ならこのまま) ?
未解凍のブロックがあります。 解凍しますか (y/n) y
bname b1 sfnc ic1:0,3,0,4

func ic1:0,3,0,4 name ic1.cir blk b1
@ func ic1:0,1,2,3

従属電圧源素子の接続ノード
i[ 4] j[ 0] 部品名[v1b1] 値[1e+006 j 0]
独立電圧源素子の個数 0
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
1 2 3 4

表示ノード番号 (0 なら全て、-1 ならこのまま) ?

使用した式の個数 0、密度 0.000 %
使用した式のサイズ 202
最大の 素子番号 = 7
最大のノード番号 = 4
最大の 素子番号 = 7
最大のノード番号 = 4
? /fname
現在読み込んでいるファイル名は = bpf11.cir
```

(18) /free

未定義

(19) /gpos

上記シミュレーションコマンド/range, /para, /ac でステップ値を等比級数で増加させる時に最低値から最高値までいくつのポイントで行なうかを設定する。

最低のポイント数は5、最高ポイント数は300である。

(20) /grph

このコマンドは削除したので、使えない。

周波数特性などのグラフを表示する。

/range, /para, /ac で作成した周波数特性などの計算ファイルが必要。

周波数など横軸の最高値／最低値の比が10倍を超える時には、自動的に横軸を対数目盛りに変更して分かり易いグラフ表示を行なう。

データファイル名は ? と表示されるので、データファイル名を拡張子を付けずに入力する。rcと入力すると rc.d がオープンされる。

最低周波数 0 最高周波数 1000 のように、周波数範囲が表示される。
出力値 最低値 -16.07 最高値 0 のように、出力値の範囲が表示される。

基準とする出力値は ? と表示されるので、
希望の出力値を入力する。例えば、0。

縦方向の位置は (0から399) ? と表示されるので、
上で入力した出力値が画面でどの高さに表示したいかを入力する。
一番上の位置が0で、一番下の位置が399になる。例えば、0。

画面の縦方向全体に対する出力の振幅は ? と表示されるので、
画面の上から下までに対応する出力値の変化幅を入力する。例えば、20。

ハードコピーを取りますか ? (y/n) では、yかn
(現在のプログラムでは、ドライバが古すぎて印刷は出来ないと思われる)
再度違う条件で表示しますか ? (y/n) でも、yかn
別のデータファイルでグラフを表示しますか ? (y/n) で、
nを入力すると sim. exe のプロンプトに戻る。

(21) /help

コマンドなどの一覧表示

(22) /ipart

メモリ上のネットリストをクリアして、新規に回路データの入力を行なう。計算できる最大の部品数はブロック素子の解凍後で約200、ノード数は60までとなっている。

```
B? /ipart

left[ 0] right ? 1
素子名は ? e1
値は ? 1

left[ 0] right ? 3
素子名は ? re
値は ? 1k

left[ 0] right ? 6
素子名は ? rl
値は ? 10k

left[ 0] right ? 5
素子名は ? e2
値は ? 12

left[ 0] right ? /

left[ 1] right ? 2
素子名は ? cc
値は ? 100u

left[ 1] right ? /

left[ 2] right ? 5
素子名は ? rb
値は ? 100k

left[ 2] right ? 4
素子名は ? bl
ファイル名と接続ノードを入力して下さい(ic1:0, 1, 2 の様に)
trn:2, 3, 4

left[ 2] right ? /

left[ 3] right ? /

left[ 4] right ? 5
素子名は ? rc
値は ? 10k

left[ 4] right ? 6
素子名は ? cc
素子名 cc は既に使用されています。
同じ素子名を使用しますか? (y/n) y

left[ 4] right ? *

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
i[ 5] j[ 0] 部品名[e2] 値[12 j 0]
独立電圧源素子の個数 2
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況

表示ノード番号 (0 なら全て、-1 ならこのまま) ?
未解凍のブロックがあります。 解凍しますか (y/n)
```

l e f t [0] r i g h t ? と表示されるので、ノード0と接続されている素子で最小のノード番号を入力する。通常は1から入力する。

素子名は ? と表示されるので、上で指定したノード間に接続される素子名を入力する。

素子名は素子属性記号に続けて英数字を書く。（e 1 , c 2 , b 3 のように）

既に、部品リストに登録されている素子名を入力すると、同じ素子名を使用するかどうかを確認してくるので、y または n で決定する。

素子属性がブロック（bで始まる素子）の時には、ブロック名と接続ノードを聞いてくるので、 b p f 1 1 : 0 , 1 , 2 の様に入力する。

素子属性が従属素子（k , v , q , wで始まる素子）の時には、マスター素子名を聞いてくる。入力したマスター素子名が既に他の従属素子で使用中の場合には違うマスター素子名の入力を求めてくる。マスター素子の属性はr , l , cである。値は ? と表示されるので、素子の値を入力する。

（1 0 0 0 , 1 e 3 , 1 e - 6 などのように）

再び

l e f t [0] r i g h t ? と表示されるが、ノード0に接続される素子がない時には / を入力すると、

l e f t [1] r i g h t ? と左ノード番号が1だけ進んで、右ノード番号の入力になる。上と同様に接続される素子が無い時は / を入力する。

全ての素子を入力したら、r i g h t ? で * を入力する。

最大ノード番号 = 2 のように、入力された最大のノード番号を表示する。

表示ノード番号 （0 なら全て） ? と表示される。

これは、／ r a n g e などで周波数特性を計算するときに画面に表示するノード番号の入力を要求している場面である。例えば、2と入力すると、ノード番号（0 なら終了） ? と表示される。

これは、他にも画面に計算結果を表示したいノードがあればノード番号を入力する様に要求している場面である。他になければ、0を入力する。

(23) /load

回路情報ファイルの読み込み

ファイル名は ? と表示されるのでファイル名を拡張子を付けずに入力する

```
? /load
読み込み
ファイル名は ? test
@ func @test:0,1,2
node 10

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
i[ 3] j[ 0] 部品名[e2] 値[24 j 0]
i[10] j[ 8] 部品名[ebb1] 値[0.6 j 0]
独立電圧源素子の個数 3
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況

表示ノード番号 (0 なら全て、-1 ならこのまま) ? 5
ノード番号 (0 なら終了) ?
```

(24) /manu

上記シミュレーションコマンド/range, /para, /ac で計算ステップを常に等比級数で増加するようにステップ値を変化させる。

(25) /mode

mode の現在の状態を表示する。

```
? /mode
データファイルを作成する..... /mk または /nomk で切り替える
デシベル値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /auto (自動切り替え) です
..... /auto または /manu で切り替える
等比級数のポイント数は 32 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
1 2 3 4 5 6
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

/nomk すると、

```
? /mode
データファイルを作成しない..... /mk または /nomk で切り替える
デシベル値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /auto (自動切り替え) です
..... /auto または /manu で切り替える
等比級数のポイント数は 32 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
1 2 3 4 5 6
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

(26) /node

回路の最大のノード番号を表示する。

(27) /padd

現在入力されているネットリストに回路を追加する時に使用する。

現在のネットリストを表示した後、

l e f t [0] r i g h t ? と表示されて、/ i p a r t の素子の接続情報の入力状態になる。

l e f t [0] に接続するものが無ければ、/exit を入力すると、l e f t [1] に進むので、希望する左ノードまで進んだら、右ノード番号を入力して、素子データを入力する。追加する素子が無くなったら、*を入力すると追加が終了する。

(28) /para

システム変数 f で指定される周波数において、指定の素子を希望の範囲で変化

させたときの電圧を計算する。全ての電圧源は入力された値を使用して計算する。

回路の直流動作点の確認などで利用する。

```
B? /para
値を変化させる素子名は    ?    r1
para r1 最低値 ? 10k
para r1 最高値 ? 50k
para r1 ステップ ? 5k
r1[1e+004    ] f[0            ]
x5 [    18.3965 j 0            ] abs[    18.3965] arg[            0]
r1[1.5e+004    ] f[0            ]
x5 [    15.9725 j 0            ] abs[    15.9725] arg[            0]
r1[2e+004    ] f[0            ]
x5 [    13.9472 j 0            ] abs[    13.9472] arg[            0]
r1[2.5e+004    ] f[0            ]
x5 [    12.2295 j 0            ] abs[    12.2295] arg[            0]
r1[3e+004    ] f[0            ]
x5 [    10.7544 j 0            ] abs[    10.7544] arg[            0]
r1[3.5e+004    ] f[0            ]
x5 [    9.4739 j 0            ] abs[    9.4739] arg[            0]
r1[4e+004    ] f[0            ]
x5 [    8.3518 j 0            ] abs[    8.3518] arg[            0]
r1[4.5e+004    ] f[0            ]
x5 [    7.3605 j 0            ] abs[    7.3605] arg[            0]
r1[5e+004    ] f[0            ]
x5 [    6.4783 j 0            ] abs[    6.4783] arg[            0]
B? キー入力待ち？
```

値を変化させる素子名は ? と表示されるので、変化させたい素子名を入力する。

para 最低値 ? と表示されるので、最低値を入力する。

para 最高値 ? と表示されるので、最高値を入力する。

para ステップ ? と表示されるので、変化ステップ値を入力する。マイナスも可。

(最高値 - 最低値) / ステップ の値が 300 以内になる

ように設定する。この値が 300 を超えた場合には、/gpos で設定された計算回数を使用して、変化ステップは自動的に決定される。

a /auto モードかつ 最高値/最低値の比が 10 以下の時は、

入力されたステップ値で計算を行なう。

b /manu モードまたは 最高値/最低値の比が 10 を超える時は、

/gpos で設定された計算ポイント数で最低値から最高値まで等比級数で増加する様にステップ値を自動的に変化させる。

/mk, /val のモードが設定されている時には表示ノードの電圧値の実数部を保存する。これは、/para が主に動作点の確認や決定に利用するために絶対値でグラフを表示しても意味がないからである。

/para で /mk, /db のモードでは表示ノードの電圧値の絶対値をファイルに保存する。

/mk の場合

```
? /para
データファイル名は ? test
値を変化させる素子名は ? r1
para r1 最低値 ? 10 k
para r1 最高値 ? 50k
para r1 ステップ ? 5k
r1[1e+004 ] f[0 ]
x5 [ 18.3965 j 0 ] abs[ 18.3965] arg[ 0]
r1[1.5e+004 ] f[0 ]
x5 [ 15.9725 j 0 ] abs[ 15.9725] arg[ 0]
r1[2e+004 ] f[0 ]
x5 [ 13.9472 j 0 ] abs[ 13.9472] arg[ 0]
r1[2.5e+004 ] f[0 ]
x5 [ 12.2295 j 0 ] abs[ 12.2295] arg[ 0]
r1[3e+004 ] f[0 ]
x5 [ 10.7544 j 0 ] abs[ 10.7544] arg[ 0]
r1[3.5e+004 ] f[0 ]
x5 [ 9.4739 j 0 ] abs[ 9.4739] arg[ 0]
r1[4e+004 ] f[0 ]
x5 [ 8.3518 j 0 ] abs[ 8.3518] arg[ 0]
r1[4.5e+004 ] f[0 ]
x5 [ 7.3605 j 0 ] abs[ 7.3605] arg[ 0]
r1[5e+004 ] f[0 ]
x5 [ 6.4783 j 0 ] abs[ 6.4783] arg[ 0]
?
```

(29) /parts

最大の素子番号、最大のノード番号を表示する。

未解凍のブロックの個数を表示する。

(30) /point

現在のシステム変数 f が示す周波数における出力値を計算する。

`sim.exe` のコマンドラインから `? f=1000`

などのように周波数を入力してから、`? /point` とコマンドを入力すると、表示ノードの電圧を表示する。`/mk`、`/nomk` に関係なく画面表示のみ。

(31) /range

指定された入力信号源以外の独立電圧源と独立電流源は仮に 0 に設定され、周波数特性の計算を行う。ファイルを作成するモードの場合には指定されたノードの電圧値の絶対値をファイルにセーブする。

```
B? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1000
range 周波数 最高値 ? 1000
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x5 [ -94.6500 j -1.8267 ] abs[ 94.6677] arg[ -178.8944]
x6 [ -94.6498 j -1.8417 ] abs[ 94.6677] arg[ -178.8852]
```

`/mk` の場合

```
? /range
データファイル名は ? test
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1000
range 周波数 最高値 ? 1000
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x5 [ -94.6500 j -1.8267 ] abs[ 94.6677] arg[ -178.8944]
x6 [ -94.6498 j -1.8417 ] abs[ 94.6677] arg[ -178.8852]
```

a c 入力信号源名は ? と表示されるので、

素子属性が e で入力信号源として計算したい素子名を入力する。この時、他の独立電圧源及び独立電流源の値は 0 として交流シミュレーションを実行する。

通常は e 1 と入力する。

値は ? と表示されるので、上で指定した電圧源の電圧を入力する。

通常はゲインが分かり易いように 1 と入力する。実際の出力振幅が知りたい時には入力電圧をそのまま入力する。

最低周波数 ? と表示されるので、最低の周波数値を入力する。

最高周波数 ? と表示されるので、最高の周波数値を入力する。

ステップ ? と表示されるので、周波数の上昇ステップ値を入力する。

(最高周波数 - 最低周波数) / ステップ の値が 300 以内になるように設定する。この値が 300 を超えた場合には、/gpos で設定された計算回数を使用して、周波数ステップは自動的に決定される。

a /auto モードかつ 最高値/最低値の比が 10 以下の時は、入力されたステップ値で計算を行なう。

b /manu モードまたは 最高値/最低値の比が 10 を超える時は、/gpos で設定された計算ポイント数で最低値から最高値まで等比級数で増加する様にステップ値を自動的に変化させる。

周波数特性を計算して、/disp、/padd、/ipart において最後に指定された表示ノードの電圧値と位相を画面表示する。

*** オプション機能 ***

/smode、/cmode、/db、/val、/mk 及び /nomk によりファイルを作成するかどうか、計算を値で行なうか db で行なうかの切り替えが出来る。

/mk によりデータファイルを作るモードに切り替えてある場合には計算結果をファイルとして保存することができる。このモードの時には、a c 入力信号源名 ? の前に、保存するためのファイル名の入力を求められる。(lpf1, rc のように 8 キャラクタ以内で、拡張子を付けずに入力する)

/mode により計算モードとファイルモードがどのように設定されているかを確認できる。

(32) /rewind

バッチファイル実行時に、バッチファイルに記述された `/wait` コマンドにより、キー入力待ちになっている時に `/rewind` とキー入力すると、そのバッチファイルの先頭まで戻ることが出来る。

(33) /save

回路情報をファイルにして保存する。

ファイル名は `?` と表示されるので、ファイル名を入力する。

(`lowpass` のように拡張子を付けずに入力すると、自動的に `lowpass.cir` として保存する)

既に、同名のファイルが登録されている時には、

1. . 重書き、他のキーならファイル名を再入力 `?`

と表示されるので、重ね書きするときは1を、他のファイル名に変更するときには0などを入力する。

外部ノード (1, 2, 3 の様に)

と表示されるので、他の回路の中で、この回路をブロック素子という部品として使用するときには接続する必要があるノード番号をカンマで区切って入力する。

素子の接続情報

```
? /dpart
left[ 0] right[ 1] parts[ e1          ] value[      1 j 0          ]
left[ 0] right[ 2] parts[ r1          ] value[ 2. 6e+004 j 0          ]
left[ 0] right[ 3] parts[ e2          ] value[      24 j 0          ]
left[ 0] right[ 4] parts[ ce          ] value[    0. 001 j 0          ]
left[ 0] right[ 4] parts[ re          ] value[    2200 j 0          ]
left[ 0] right[ 6] parts[ r1          ] value[ 1e+006 j 0          ]
left[ 1] right[ 2] parts[ cc          ] value[    0. 001 j 0          ]
left[ 2] right[ 3] parts[ r2          ] value[ 5e+004 j 0          ]
left[ 2] right[10] parts[ r1b1        ] value[     800 j 0          ]
left[ 3] right[ 5] parts[ rc          ] value[    3800 j 0          ]
left[ 4] right[ 7] parts[ rbb1        ] value[     0. 1 j 0          ]
left[ 5] right[ 6] parts[ cc          ] value[    0. 001 j 0          ]
left[ 5] right[ 9] parts[ rbb1        ] value[     0. 1 j 0          ]
left[ 7] right[ 8] parts[ reb1        ] value[     26 j 0          ]
left[ 7] right[ 9] parts[ kb1         ] value[    100 j 0          ]
@ master[reb1      ] left[ 7] right[ 8]
left[ 8] right[10] parts[ ebb1        ] value[     0. 6 j 0          ]
最大の素子番号 = 16
最大のノード番号 = 10
```

回路情報の保存

```
? /save  
セーブ  
ファイル名は ? test  
外部ノード (1, 2, 3 の様に)  
0, 1, 2
```

保存された接続情報

@test:0, 1, 2

10

0 1 @e @e1 @ 0 1 0 @

0 2 @r @r1 @ 0 26000 0 @

0 3 @e @e2 @ 0 24 0 @

0 4 @c @ce @ 0 0.001 0 @

0 4 @r @re @ 0 2200 0 @

0 6 @r @r1 @ 0 1e+006 0 @

1 2 @c @cc @ 0 0.001 0 @

2 3 @r @r2 @ 0 50000 0 @

2 5 @* @b1 @ 1 0 0 @trn:2, 4, 5

2 10 @r @r1b1 @ 0 800 0 @

3 5 @r @rc @ 0 3800 0 @

4 7 @r @rbb1 @ 0 0.1 0 @

5 6 @c @cc @ 0 0.001 0 @

5 9 @r @rbb1 @ 0 0.1 0 @

7 8 @r @reb1 @ 0 26 0 @

7 9 @k @kb1 @reb1 0 100 0 @

8 10 @e @ebb1 @ 0 0.6 0 @

fnc

(34) /skip

バッチファイル実行時に、バッチファイルに記述された / w a i t コマンドにより、キー入力待ちになっている時に / s k i p とキー入力すると、そのバッチファイルを最後までスキップすることが出来る。

(35) /smode

mode をセットする

```
? /smode
データファイルを作成する..... /mk または /nomk で切り替える
デシベル値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /auto (自動切り替え) です
..... /auto または /manu で切り替える
等比級数のポイント数は 32 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
1 2 3 4 5 6
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

(36) /val

値で計算するモードにする。

電卓としての利用法：

変数を回路データの変数と合わせて、200まで使用できる。また、ユーザ関数（手続き）を60個まで定義して使用できる。変数名、関数名は15文字まで有効である。

1. 変数に値を代入する。変数名＝値（または式）

? a = 100

? a = 1 + j 3 (変数は全て複素数として扱う)

? a b c = a * b / c などのように入力する

? > 変数名（直前の計算結果が変数に代入される）

2. 関数（手続き）の定義：

/ d e f

関数定義： 関数名をアルファベット15文字以内で入力して下さい？

と表示されるので、関数名をアルファベット15文字以内で入力する。

例えば、nyuryoku1

f n c (nyuryoku) = . . . リターン ならこのまま

と表示されるので式を入力する。既に、同名の関数が定義されている時には

. . . の所に式が表示される。

a 1 * \$ b - c のように、関数定義されている他の関数を利用することも可能
この場合、関数名に\$をつける。

関数を定義したら、\$nyuryokuを入力すると、定義した計算が行われる。

3. 計算：

? 100 + 30

? a 1 / (b - c)

? \$ a * (b + c)

? s i n (a)

? p o w (2, b) のように、f a c t o rに含まれる全ての因子を使用可

定義済みの変数名と値は / d v a r で、定義済みの関数名（手続き名）と式は

/ d f n c で確認ができる。

変数と関数は回路データをセーブする時に、同時にファイルにセーブされる。

sim.exe の操作練習

まず、sim.exe を起動して、/ipart コマンドによって回路図の入力を練習します。

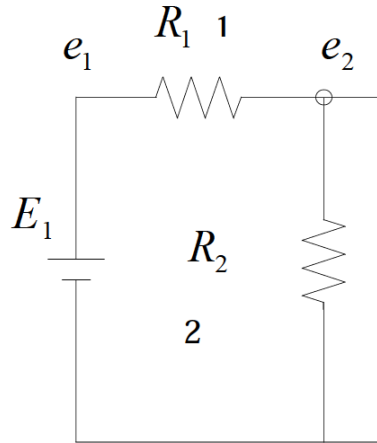


図 ex-1

図で e_1 , e_2 と表示されているのは、対応するノードの電圧を表します。そして、 e_1 のノード番号は 1、 e_2 のノード番号は 2、電源 E_1 のマイナス側はノード番号 0 とします。

図では、抵抗 R_1 の右側に 1 が見えますが、これは抵抗値を表します。回路図の入力時には好きな値を入力して構いません。 R_2 の値も同様です。

```
? /ipart
left[ 0] right ? 1
素子名は ? e1
値は ? 1

left[ 0] right ? 2
素子名は ? r2
値は ? 2

left[ 0] right ? /

left[ 1] right ? 2
素子名は ? r1
値は ? 1

left[ 1] right ? *

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
独立電圧源素子の個数 1
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況

表示ノード番号 (0 なら全て、-1 ならこのまま) ? 2
ノード番号 (0 なら終了) ?
```

入力した回路データは/dpartによって確認できます。

```
? /dpart
left[ 0] right[ 1] parts[ e1          ] value[          1 j 0          ]
left[ 0] right[ 2] parts[ r2          ] value[          2 j 0          ]
left[ 1] right[ 2] parts[ r1          ] value[          1 j 0          ]
最大の 素子番号 = 3
最大のノード番号 = 2
```

2つの抵抗の値を変更して、確認します。

```
? r1=1k
1,000 j 0

? r2=2k
2,000 j 0

? /dpart
left[ 0] right[ 1] parts[ e1          ] value[          1 j 0          ]
left[ 0] right[ 2] parts[ r2          ] value[        2000 j 0          ]
left[ 1] right[ 2] parts[ r1          ] value[        1000 j 0          ]
```

このように、コマンドラインから簡単に素子値を変更することが出来ます。

ノード2の電圧を調べてみます。

E1は直流電源として、f=0に設定してから、/pointを実行します。

```
? f=0
0 j 0

? /point
f[0          ]
x2 [ 0.6667 j 0          ] abs[ 0.6667] arg[ 0]
```

回路図を入力した時に、表示ノード番号を2に指定したのでx2の値が表示される。

次に、r1の値を変化するとe2がどのように変化するかを調べてみます。

r1を10から10kまで、5点の計算を行います。

```
? /gpos
MANU ON または 最高/最低の比が10以上の時に、
シミュレーションで計算するポイント数を入力して下さい
最低 5 以上、最大 300 以下の数字を入力して下さい
現在の設定は 32 です
5
gpos = 5 に設定しました
? /manu
? /para
値を変化させる素子名は ? r1
para r1 最低値 ? 10
para r1 最高値 ? 10k
para r1 ステップ ? 1k
r1[10          ] f[0          ]
x2 [ 0.9950 j 0          ] abs[ 0.9950] arg[ 0]
r1[56.23       ] f[0          ]
x2 [ 0.9727 j 0          ] abs[ 0.9727] arg[ 0]
r1[316.2       ] f[0          ]
x2 [ 0.8635 j 0          ] abs[ 0.8635] arg[ 0]
r1[1778        ] f[0          ]
x2 [ 0.5293 j 0          ] abs[ 0.5293] arg[ 0]
r1[1e+004      ] f[0          ]
x2 [ 0.1667 j 0          ] abs[ 0.1667] arg[ 0]
```

r1 を変化させる範囲が広いので、計算するポイント数を 5 個に限定して表示画面に収まるように指定します。/gpos コマンドで 5 を入力してから、/manu コマンドを入力します。/auto の状態の場合には、値を変化させる最大値と最小値の倍率が 10 を超える場合には、自動的に最大 32 ポイントの計算を行います。

次に、/para コマンドを実行します。シミュレーションのコマンドを実行しても、素子の値は元のまま変化しません。

```
? /dpart
left[ 0] right[ 1] parts[ e1] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r2] value[ 2000 j 0 ]
left[ 1] right[ 2] parts[ r1] value[ 1000 j 0 ]
```

次は、回路に部品を追加してみます。コマンド/padd を使用します。

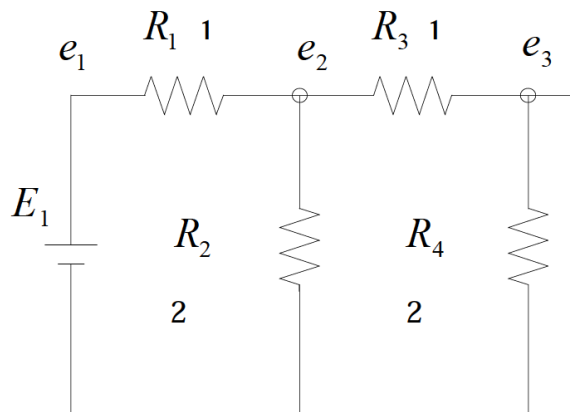


図 ex-2

/padd コマンドを入力すると、現在の回路情報が表示されて、次に左ノード 0 に対する追加素子のノード番号の入力が求められます。

```
? /padd
left[ 0] right[ 1] parts[ e1] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r2] value[ 2000 j 0 ]
left[ 1] right[ 2] parts[ r1] value[ 1000 j 0 ]
最大の素子番号 = 3
最大のノード番号 = 2
```

```
left[ 0] right ? 3
素子名は ? r4
値は ? 2k

left[ 0] right ? /
left[ 1] right ? /

left[ 2] right ? 3
素子名は ? r3
値は ? 1k

left[ 2] right ? *
```

左ノード番号を次に進めるには、/を入力します。

追加する素子が無くなったら、*を入力します。

素子の追加を終了すると、表示ノードの指定になります。

```
独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
独立電圧源素子の個数 1
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
2
表示ノード番号 (0 なら全て、-1 ならこのまま) ? 0
```

表示ノード番号を変更するには、/disp コマンドを使用します。
ノード 2 と 3 を指定します。

```
? /disp

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
独立電圧源素子の個数 1
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
1 2 3
表示ノード番号 (0 なら全て、-1 ならこのまま) ? 2
ノード番号 (0 なら終了) ? 3
ノード番号 (0 なら終了) ?
```

sim.exe の現在設定されているモードを確認するには、/mode を使います。

```
? /mode
データファイルを作成しない..... /mk または /nomk で切り替える
値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /manu (等比級数) です
..... /auto または /manu で切り替える
等比級数のポイント数は 5 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
2 3
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

モードをクリアするには、/cmode を使用します。

```
? /cmode
データファイルを作成しない..... /mk または /nomk で切り替える
値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /auto (自動切り替え) です
..... /auto または /manu で切り替える
等比級数のポイント数は 32 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
2 3
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

モードをセットするには、/smode を使います。

```
? /smode
データファイルを作成する..... /mk または /nomk で切り替える
デシベル値で計算するモード..... /db または /val で切り替える
計算ステップのモードは /auto (自動切り替え) です
..... /auto または /manu で切り替える
等比級数のポイント数は 32 です..... /gpos で設定する
シミュレーションにおける 表示ノード設定状況
2 3
ブロック名を回路リストから得るモード..... /bmode で設定する
シミュレーション・コマンドの計算を ノード電圧のみで計算するモード
..... /cal で設定する
```

個別のモードだけを切り替えるには、/mk, /nomk, /db, /val, /auto, /manu, /gpos, /bmode, /cal, /disp などのコマンドを利用します。ここでは、/val モードに設定します。

部品を追加した回路情報を確認します。

```
? /dpart
left[ 0] right[ 1] parts[ e1          ] value[      1 j 0      ]
left[ 0] right[ 2] parts[ r2          ] value[    2000 j 0      ]
left[ 0] right[ 3] parts[ r4          ] value[    2000 j 0      ]
left[ 1] right[ 2] parts[ r1          ] value[    1000 j 0      ]
left[ 2] right[ 3] parts[ r3          ] value[    1000 j 0      ]
```

r1 を 1k から 10k まで変化させると、e2 と e3 がどのように変化するかを確認します。

```
? /auto
? /gpos
MANU ON または 最高/最低の比が 10 以上の時に、
シミュレーションで計算するポイント数を入力して下さい
最低 5 以上、最大 300 以下の数字を入力して下さい
現在の設定は 32 です
10
? /para
データファイル名は ? test
値を変化させる素子名は ? r1
para r1 最低値 ? 1k
para r1 最高値 ? 10k
para r1 ステップ ? 1k
r1[1000] ] f[0 ]
x2 [ 0.5455 j 0 ] abs[ 0.5455] arg[ 0]
x3 [ 0.3636 j 0 ] abs[ 0.3636] arg[ 0]
r1[2000] ] f[0 ]
x2 [ 0.3750 j 0 ] abs[ 0.3750] arg[ 0]
x3 [ 0.2500 j 0 ] abs[ 0.2500] arg[ 0]
r1[3000] ] f[0 ]
x2 [ 0.2857 j 0 ] abs[ 0.2857] arg[ 0]
x3 [ 0.1905 j 0 ] abs[ 0.1905] arg[ 0]
r1[4000] ] f[0 ]
x2 [ 0.2308 j 0 ] abs[ 0.2308] arg[ 0]
x3 [ 0.1538 j 0 ] abs[ 0.1538] arg[ 0]
r1[5000] ] f[0 ]
x2 [ 0.1935 j 0 ] abs[ 0.1935] arg[ 0]
x3 [ 0.1290 j 0 ] abs[ 0.1290] arg[ 0]
r1[6000] ] f[0 ]
x2 [ 0.1667 j 0 ] abs[ 0.1667] arg[ 0]
x3 [ 0.1111 j 0 ] abs[ 0.1111] arg[ 0]
r1[7000] ] f[0 ]
x2 [ 0.1463 j 0 ] abs[ 0.1463] arg[ 0]
x3 [ 0.0976 j 0 ] abs[ 0.0976] arg[ 0]
r1[8000] ] f[0 ]
x2 [ 0.1304 j 0 ] abs[ 0.1304] arg[ 0]
x3 [ 0.0870 j 0 ] abs[ 0.0870] arg[ 0]
r1[9000] ] f[0 ]
x2 [ 0.1176 j 0 ] abs[ 0.1176] arg[ 0]
x3 [ 0.0784 j 0 ] abs[ 0.0784] arg[ 0]
r1[1e+004] ] f[0 ]
x2 [ 0.1071 j 0 ] abs[ 0.1071] arg[ 0]
x3 [ 0.0714 j 0 ] abs[ 0.0714] arg[ 0]
```

/mk のモードだったので、計算結果が test.d に保存されています。
計算モードが/val だったので、数値（電圧値）で記録されます。
/mk ならデシベル値で記録されます。

test.d の内容

2 10	2 は表示ノードの個数、10 は計算するポイント数
1000	値を変化させる素子の値
2 0.545455	ノード 2 の計算値
3 0.363636	ノード 3 の計算値
0 0	表示ノードの終わり
2000	
2 0.375	
3 0.25	
0 0	
3000	
2 0.285714	
3 0.190476	
0 0	
4000	
2 0.230769	
3 0.153846	
0 0	
5000	
2 0.193548	
3 0.129032	
0 0	
6000	
2 0.166667	
3 0.111111	
0 0	
7000	
2 0.146341	
3 0.097561	
0 0	
8000	
2 0.130435	
3 0.0869565	
0 0	
9000	
2 0.117647	
3 0.0784314	
0 0	
10000	
2 0.107143	
3 0.0714286	
0 0	

新しい回路図を入力します。

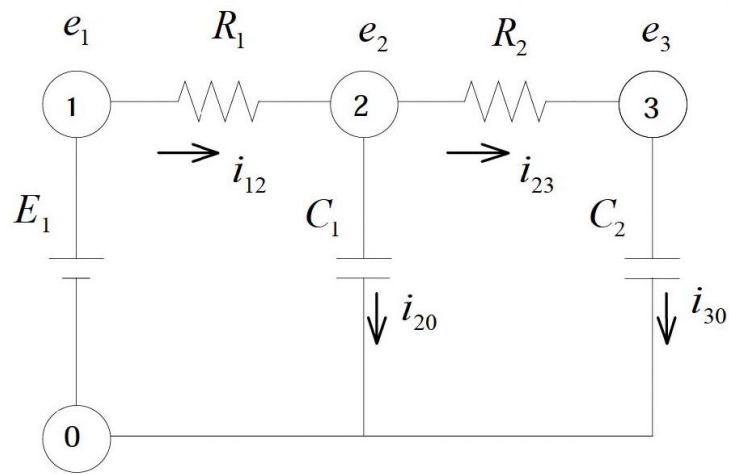


図 ex-3

```
? /ipart
left[ 0] right ? 1
素子名は ? e1
値は ? 1

left[ 0] right ? 2
素子名は ? c1
値は ? 0.01u

left[ 0] right ? 3
素子名は ? c2
値は ? 0.02u

left[ 0] right ? /

left[ 1] right ? 2
素子名は ? r1
値は ? 1k

left[ 1] right ? /

left[ 2] right ? 3
素子名は ? r2
値は ? 1k

left[ 2] right ? *
```

```
? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ c1 ] value[ 1e-008 j 0 ]
left[ 0] right[ 3] parts[ c2 ] value[ 2e-008 j 0 ]
left[ 1] right[ 2] parts[ r1 ] value[ 1000 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 1000 j 0 ]
```


この回路では C 素子を使用されているので、入力信号源 E1 の周波数を変化させた場合のノード 2 と 3 の電圧値を確認する。/range コマンドを使用する。

```
? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 0
range 周波数 最高値 ? 10k
range 周波数 ステップ ? 1k
e1[1 j 0 ]
f[0 ]
x2 [ 1 j 0 ] abs[ 1] arg[ 0]
x3 [ 1 j 0 ] abs[ 1] arg[ 0]
f[1,000 ]
x2 [ 0.9526 j -0.1750 ] abs[ 0.9685] arg[ -10.4084]
x3 [ 0.9161 j -0.2901 ] abs[ 0.9609] arg[ -17.5709]
f[2,000 ]
x2 [ 0.8452 j -0.2889 ] abs[ 0.8932] arg[ -18.8681]
x3 [ 0.7267 j -0.4715 ] abs[ 0.8663] arg[ -32.9759]
f[3,000 ]
x2 [ 0.7334 j -0.3382 ] abs[ 0.8076] arg[ -24.7585]
x3 [ 0.5305 j -0.5382 ] abs[ 0.7557] arg[ -45.4145]
f[4,000 ]
x2 [ 0.6426 j -0.3490 ] abs[ 0.7313] arg[ -28.5047]
x3 [ 0.3730 j -0.5365 ] abs[ 0.6534] arg[ -55.1913]
f[5,000 ]
x2 [ 0.5751 j -0.3428 ] abs[ 0.6695] arg[ -30.7931]
x3 [ 0.2579 j -0.5048 ] abs[ 0.5669] arg[ -62.9350]
f[6,000 ]
x2 [ 0.5257 j -0.3309 ] abs[ 0.6211] arg[ -32.1916]
x3 [ 0.1761 j -0.4637 ] abs[ 0.4960] arg[ -69.2072]
f[7,000 ]
x2 [ 0.4888 j -0.3185 ] abs[ 0.5834] arg[ -33.0852]
x3 [ 0.1176 j -0.4219 ] abs[ 0.4380] arg[ -74.4216]
f[8,000 ]
x2 [ 0.4605 j -0.3073 ] abs[ 0.5536] arg[ -33.7134]
x3 [ 0.0754 j -0.3830 ] abs[ 0.3904] arg[ -78.8651]
f[9,000 ]
x2 [ 0.4380 j -0.2978 ] abs[ 0.5296] arg[ -34.2179]
x3 [ 0.0444 j -0.3480 ] abs[ 0.3508] arg[ -82.7350]
f[10,000 ]
x2 [ 0.4194 j -0.2902 ] abs[ 0.5101] arg[ -34.6798]
x3 [ 0.0212 j -0.3169 ] abs[ 0.3176] arg[ -86.1679]
```

ノード 2 と 3 の電圧値は周波数が高くなるにしたがって減衰して、ローパスフィルタの特性を示すことが分かる。

周波数 f を 1k に固定して、C2 を 0.02u から 0.1u まで変化させるとどのように変化するかを調べる。/ac コマンドを使用する。

```

? /ac
交流シミュレーションを行ないませんが、現在の周波数は 0.000000 です
周波数を変更しますか (y/n) y
周波数は? 1000
ac 入力信号源名は ? e1
値は ? 1

値を変化させる素子名は ? c2
ac c2 最低値 ? 0.02u
ac c2 最高値 ? 0.1u
ac c2 ステップ ? 0.02u
e1[1      ] j 0      ]
c2[2e-008  ] f[1,000  ]
x2 [ 0.9526 j -0.1750 ] abs[ 0.9685] arg[ -10.4084]
x3 [ 0.9161 j -0.2901 ] abs[ 0.9609] arg[ -17.5709]
c2[4e-008  ] f[1,000  ]
x2 [ 0.8742 j -0.2469 ] abs[ 0.9084] arg[ -15.7721]
x3 [ 0.7639 j -0.4389 ] abs[ 0.8810] arg[ -29.8799]
c2[6e-008  ] f[1,000  ]
x2 [ 0.7926 j -0.2769 ] abs[ 0.8396] arg[ -19.2609]
x3 [ 0.6025 j -0.5041 ] abs[ 0.7856] arg[ -39.9169]
c2[8e-008  ] f[1,000  ]
x2 [ 0.7241 j -0.2797 ] abs[ 0.7763] arg[ -21.1168]
x3 [ 0.4659 j -0.5138 ] abs[ 0.6936] arg[ -47.8034]
c2[1e-007  ] f[1,000  ]
x2 [ 0.6719 j -0.2688 ] abs[ 0.7236] arg[ -21.8049]
x3 [ 0.3606 j -0.4954 ] abs[ 0.6127] arg[ -53.9469]

```

C2 の素子値が大きくなるほど、信号が減衰することが確認できます。

次は、ブロック素子を利用する回路を入力します。

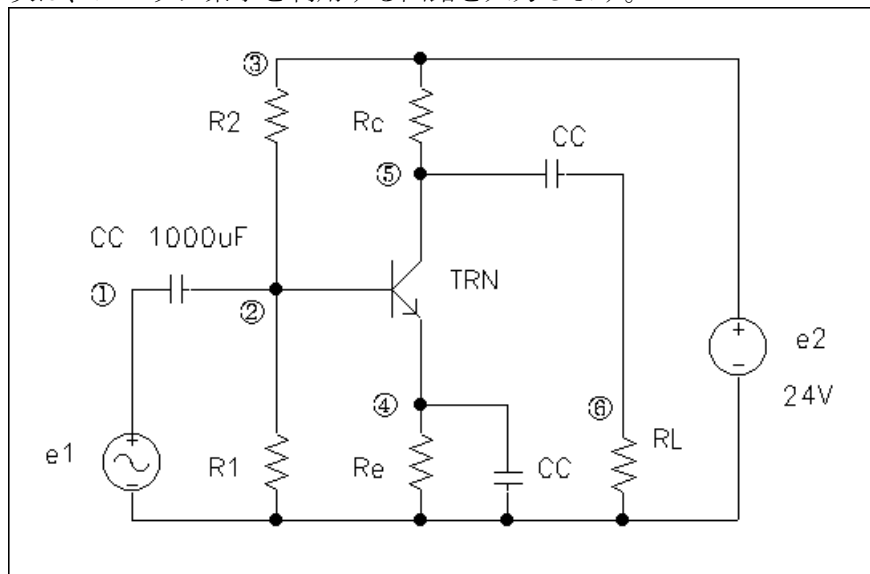


図 ex4

```

B? /ipart

left[ 0] right ? 1
素子名は ? e1
値は ? 1

left[ 0] right ? 2
素子名は ? r1
値は ? 10k

left[ 0] right ? 4
素子名は ? re
値は ? 2.2k

left[ 0] right ? 4
素子名は ? ce
値は ? 1000u

left[ 0] right ? 6
素子名は ? rl
値は ? 1k

left[ 0] right ? 3
素子名は ? e2
値は ? 24

left[ 0] right ? /

left[ 1] right ? 2
素子名は ? cc
値は ? 1000u

left[ 1] right ? /

left[ 2] right ? 3
素子名は ? r2
値は ? 50k

left[ 2] right ? 5
素子名は ? b1
ファイル名と接続ノードを入力して下さい(ic1:0,1,2 の様に)
trn2:2,4,5

left[ 2] right ? /

left[ 3] right ? 5
素子名は ? rc
値は ? 3.8k

left[ 3] right ? /

left[ 4] right ? /

left[ 5] right ? 6
素子名は ? cc
素子名 cc は既に使用されています。
同じ素子名を使用しますか？ (y/n) y

left[ 5] right ? *

```

```
? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r1 ] value[ 1e+004 j 0 ]
left[ 0] right[ 3] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[ 4] parts[ ce ] value[ 0.001 j 0 ]
left[ 0] right[ 4] parts[ re ] value[ 2200 j 0 ]
left[ 0] right[ 6] parts[ rl ] value[ 1000 j 0 ]
left[ 1] right[ 2] parts[ cc ] value[ 0.001 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 5e+004 j 0 ]
left[ 2] right[ 5] parts[ b1 ] value[ 0 j 0 ]
@ func trn:2,4,5
left[ 3] right[ 5] parts[ rc ] value[ 3800 j 0 ]
left[ 5] right[ 6] parts[ cc ] value[ 0.001 j 0 ]
```

未解凍のブロックをロードします。

```
? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r1 ] value[ 1e+004 j 0 ]
left[ 0] right[ 3] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[ 4] parts[ ce ] value[ 0.001 j 0 ]
left[ 0] right[ 4] parts[ re ] value[ 2200 j 0 ]
left[ 0] right[ 6] parts[ rl ] value[ 1000 j 0 ]
left[ 1] right[ 2] parts[ cc ] value[ 0.001 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 5e+004 j 0 ]
left[ 2] right[ 10] parts[ rl b1 ] value[ 300 j 0 ]
left[ 3] right[ 5] parts[ rc ] value[ 3800 j 0 ]
left[ 4] right[ 7] parts[ rbb1 ] value[ 0.1 j 0 ]
left[ 5] right[ 6] parts[ cc ] value[ 0.001 j 0 ]
left[ 5] right[ 9] parts[ rbb1 ] value[ 0.1 j 0 ]
left[ 7] right[ 8] parts[ reb1 ] value[ 26 j 0 ]
left[ 7] right[ 9] parts[ kb1 ] value[ 50 j 0 ]
@ master[reb1 ] left[ 7] right[ 8]
left[ 8] right[ 10] parts[ ebb1 ] value[ 0.7 j 0 ]
```

ノード 5 が E2 の半分程度の電圧になるように R2 を調整します。

現在の電圧を調べます。

```
? /point
f[0
x5 [ 18.8124 j 0 ] abs[ 18.8124] arg[ 0]
```

現在は R2=50K で、E2 の半分よりだいぶ高い電圧です。R2 を小さくするとコレクタ電流が増加して e5 が下がるので、/para で R2 を 10k から 20K の間で変化させてみます。

```
r2[1.9e+004 ] f[0
x5 [ 11.9125 j 0 ] abs[ 11.9125] arg[ 0]
r2[2e+004 ] f[0
x5 [ 12.3639 j 0 ] abs[ 12.3639] arg[ 0]
```

R2=20K に決定します。

```
? r2=20k
20,000 j 0
```

/range で、ゲインを測定します。

```
? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1k
range 周波数 最高値 ? 1k
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x5 [ -119.4764 j -2.9742 ] abs[ 119.5135] arg[ -178.5740]
```

次は、オペアンプを含む回路図を入力します。

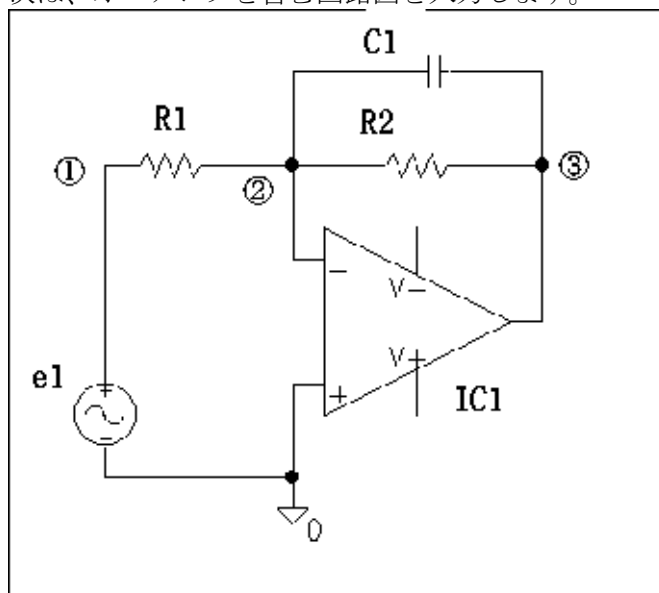


図 ex5

```

/ipart
left[ 0] right ? 1
素子名は ? e1
値は ? 1

left[ 0] right ? /

left[ 1] right ? 2
素子名は ? r1
値は ? 1k

left[ 1] right ? /

left[ 2] right ? 3
素子名は ? b1
ファイル名と接続ノードを入力して下さい(ic1:0,1,2 の様に)
ic1:0,2,0,3

left[ 2] right ? 3
素子名は ? r2
値は ? 10k

left[ 2] right ? 3
素子名は ? c1
値は ? 0.1u

left[ 2] right ? *

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
独立電圧源素子の個数 1
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況

表示ノード番号 (0 なら全て、-1 ならこのまま) ?
未解凍のブロックがあります。 解凍しますか (y/n) y
bname b1 sfnc ic1:0,2,0,3

```

```

? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 3] parts[ v1b1 ] value[ 1e+006 j 0 ]
@ master[r1b1 ] left[ 2] right[ 0]
left[ 1] right[ 2] parts[ r1 ] value[ 1000 j 0 ]
left[ 2] right[ 0] parts[ r1b1 ] value[ 1e+006 j 0 ]
left[ 2] right[ 3] parts[ c1 ] value[ 1e-007 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 1e+004 j 0 ]

```

この回路もローパスフィルタ特性を示します。`/range` を使って確認してください。
`r1`, `r2`, `c1` の値を変えた時の周波数特性の変化なども調べてください。

以上で、`sim.exe` の基本的な操作練習は終わりです。

サンプルデータファイルの内容

ic1.cir: sim.exeで使用する、標準オペアンプのブロック素子
0はグラウンド、1はマイナス入力、2はプラス入力、3は出力のノードを表わす

```
@ic1:0,1,2,3
3
0 3 @v @v1 @r1 0 100000 0 @
1 2 @r @r1 @ 0 1000000 0 @
fnc
```

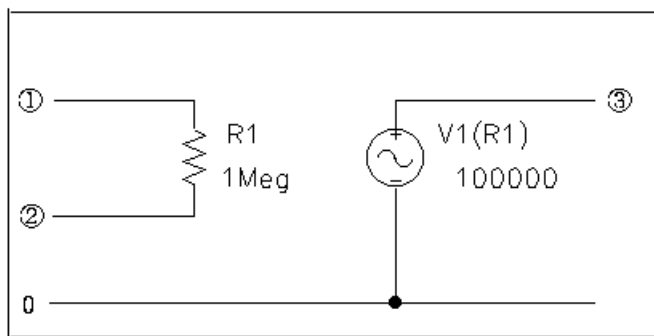


図 ic1.cir の等価回路

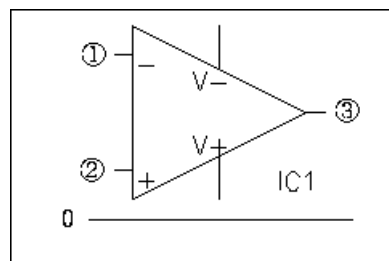


図 一般回路図での表記法

bpf11.cir: ic1を使用して作成した、1次バンドパスフィルターのブロック素子
0はグラウンド、1は信号の入力、4は出力のノードを表わす。
ブロック素子として定義してあるので、入力信号の電圧源は含んでいない。

```
@bpf11:0,1,4
4
0 2 @r @r1b @ 0 357.48 0 @
0 3 @b @b1 @ 1 0 0 @ic1:0,3,0,4
1 2 @r @r1a @ 0 81215 0 @
2 3 @c @c @ 0 1.674e-08 0 @
2 4 @c @c @ 0 1.674e-08 0 @
3 4 @r @r2 @ 0 280968 0 @
fnc
```

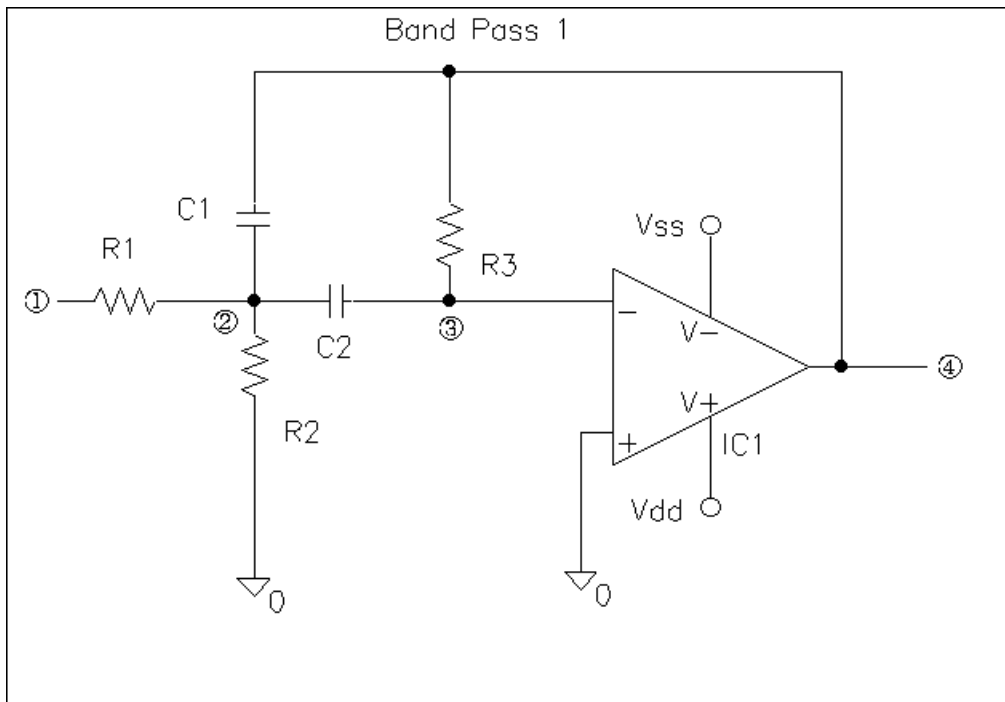
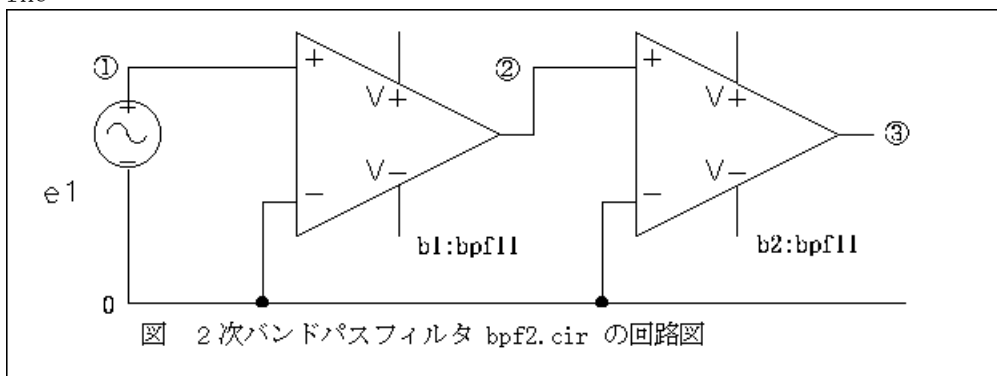


図 バンドパスフィルタの基本となる、1次バンドパスフィルタ回路 bpf11.cir

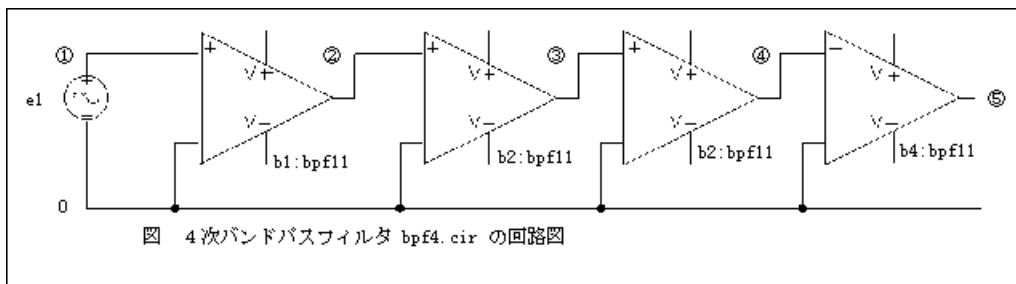
bpf2.cir: b p f 1 1 を使用して作成した、2次バンドパスフィルタのネットリスト
 ブロック素子として定義された、bpf11.cir を2個使用して作成した2次のバンドパス
 フィルタの回路。0はグランド、1は信号の入力、3は出力のノードを表わす。
 信号源電圧源 e1 がノード1とノード0に接続されている。

```
@bpf2:0,1,3
3
0 1 @e @e1 @ 0 1 0 @
0 2 @b @b1 @ 1 0 0 @bpf11:0,1,2
0 3 @b @b2 @ 2 0 0 @bpf11:0,2,3
fnc
```



bpf4.cir: b p f 1 1 を使用して作成した、4 次バンドパスフィルターのネットリスト
 ブロック素子として定義された、bpf11.cir を 2 個使用して作成した 2 次のバンドパス
 フィルタの回路。0 はグランド、1 は信号の入力、5 は出力のノードを表わす。
 信号源電圧源 e1 がノード 1 とノード 0 に接続されている。

```
@bpf4:0,1,5
5
0 1 @e @e1 @ 0 1 0 @
0 2 @b @b1 @ 1 0 0 @bpf11:0,1,2
0 3 @b @b2 @ 2 0 0 @bpf11:0,2,3
0 4 @b @b3 @ 3 0 0 @bpf11:0,3,4
0 5 @b @b4 @ 4 0 0 @bpf11:0,4,5
fnc
```



バッチファイルのサンプル

b a t 1 . s b , b a t 2 . s b , b a t 3 . s b はバッチファイルの
 サンプルである。これらは、s i m . e x e の # コマンドで作られたバッチ
 ファイルをエディタを使用して、コメントを付加したり、/ w a i t コマンド
 を追加したものである。

b a t 1 . s b :

```
% バッチファイルによる自動実行のテスト
% 2 次のバンドパスフィルタの回路データ bpf2
% を読み込んで、解凍せずに load を終了後 bload
% で解凍して、回路リストを表示する。
/load
% 回路データのファイル名
```

```

bpf2
% 表示ノード指定しない

% 解凍しない

% 解凍しない

% 解凍のために bload を実行する
/bload
% 表示ノード指定しない

% キャパシタ cb2 の値を変更する
cb2=1.514e-8
% 回路リストを表示して、キー入力待ち
/dpart
/wait
% 次のバッチファイルを実行して、キー入力待ち
@bat2
/wait

```

b a t 2 . s b :

```

% バッチファイル名 bat2
% 表示ノードを指定して、周波数特性を計算する
%
% 表示ノードを指定
/disp
3

% 周波数特性を計算して、データファイル bpf2b を作る
% モードをセット
/smode
/range
bpf2b
% 入力信号源名と値

```

```

e1
1
% 最低周波数、最高周波数、周波数ステップ
500
1500
50
% キー入力待ち
/wait
% 次のバッチファイルを実行する
@bat3
% キー入力待ち bat2 終了
/wait

bat3.s b :
% バッチファイル名 bat3
% bpf2 の接続行列を出力する
/save
bpf2-b
0, 1, 3

% 係数行列を出力する
/conv
% ハードコピーするか? 判断を入力する
f
! copy conv.d bpf2.coe
! del conv.d

bpf2 の接続行列
@bpf2-b:0, 1, 3
7
0 1 @e @e1 @ 0 1 0 @
0 2 @* @b1 @ 1 0 0 @bpf11:0, 1, 2
0 2 @v @v1b1b1 @r1b1b1 0 1e+006 0 @
0 3 @* @b2 @ 2 0 0 @bpf11:0, 2, 3
0 3 @v @v1b1b2 @r1b1b2 0 1e+006 0 @
0 4 @r @r1bb1 @ 0 357.48 0 @

```

```

0 5 @* @b1b1 @ 3 0 0 @ic1:0,5,0,2
0 6 @r @r1bb2 @ 0 357.48 0 @
0 7 @* @b1b2 @ 4 0 0 @ic1:0,7,0,3
1 4 @r @r1ab1 @ 0 81215 0 @
2 6 @r @r1ab2 @ 0 81215 0 @
4 2 @c @cb1 @ 0 1.674e-008 0 @
4 5 @c @cb1 @ 0 1.674e-008 0 @
5 0 @r @r1b1b1 @ 0 1e+006 0 @
5 2 @r @r2b1 @ 0 280968 0 @
6 3 @c @cb2 @ 0 1.514e-008 0 @
6 7 @c @cb2 @ 0 1.514e-008 0 @
7 0 @r @r1b1b2 @ 0 1e+006 0 @
7 3 @r @r2b2 @ 0 280968 0 @
fnc

```

bpf2 の係数行列

```

restart: with(linalg):
siki:=matrix(7,7,0):e:=vector(7,0):
siki[ 1, 1] := 1 ;
e[1] := +e1 ;
siki[ 2, 2] := 1 ;
siki[ 2, 5] := -v1b1b1 ;
siki[ 3, 3] := 1 ;
siki[ 3, 7] := -v1b1b2 ;
siki[ 4, 1] := -1/r1ab1 ;
siki[ 4, 2] := -s*cb1 ;
siki[ 4, 4] := +1/r1bb1+1/r1ab1+s*cb1+s*cb1 ;
siki[ 4, 5] := -s*cb1 ;
siki[ 5, 2] := -1/r2b1 ;
siki[ 5, 4] := -s*cb1 ;
siki[ 5, 5] := +s*cb1+1/r1b1b1+1/r2b1 ;
siki[ 6, 2] := -1/r1ab2 ;
siki[ 6, 3] := -s*cb2 ;
siki[ 6, 6] := +1/r1bb2+1/r1ab2+s*cb2+s*cb2 ;
siki[ 6, 7] := -s*cb2 ;
siki[ 7, 3] := -1/r2b2 ;

```

```
siki[ 7, 6] := -s*cb2 ;
siki[ 7, 7] := +s*cb2+1/r1b1b2+1/r2b2 ;
x:=linsolve(siki,e);
```

```
e1 := 1 ;
r1bb1 := 357.48 ;
r1ab1 := 81215 ;
cb1 := 1.674e-008 ;
r2b1 := 280968 ;
r1bb2 := 357.48 ;
r1ab2 := 81215 ;
cb2 := 1.514e-008 ;
r2b2 := 280968 ;
v1b1b1 := 1e+006 ;
r1b1b1 := 1e+006 ;
v1b1b2 := 1e+006 ;
r1b1b2 := 1e+006 ;
x3 := -0.0216264 ;
```

```
;end;
```

変換終了

使用した式の個数 20、 密度 40.816 %

使用した式のサイズ 360

上記の係数行列のデータを

```
restart: with(linalg):から x:=linsolve(siki,e);
```

の行までを数式処理ソフト Maple に入力して実行すれば

回路方程式を文字式のままで解くことができます。

s1.sb、s2.sbなどのバッチファイルを実行すると、回路を設計するためにどのようにコマンドを利用すれば良いのかを知ることが出来ます。グラフコマンドが使えないので、バッチファイルを実行すると、計算結果がs1-1.dなどのファイルとして保存されます。

トランジスタを使用した回路の設計練習用にサンプル回路を準備してある。

全てのサンプル回路は、回路リストの形ではなくバッチファイルの形で準備してある。これは、設計の各ステップごとにどのコマンドをどのように使用すれば回路の設計が出来るかを示すためである。

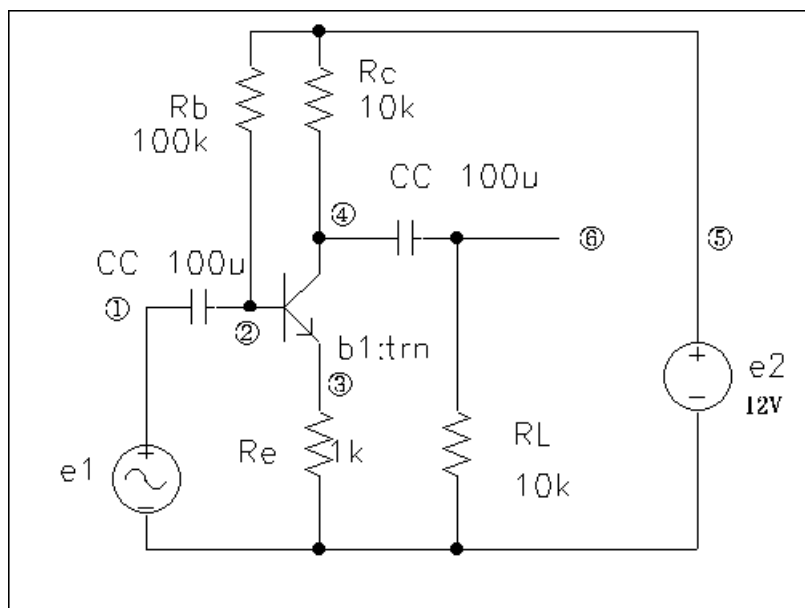


図 s1.sb で設計する回路図

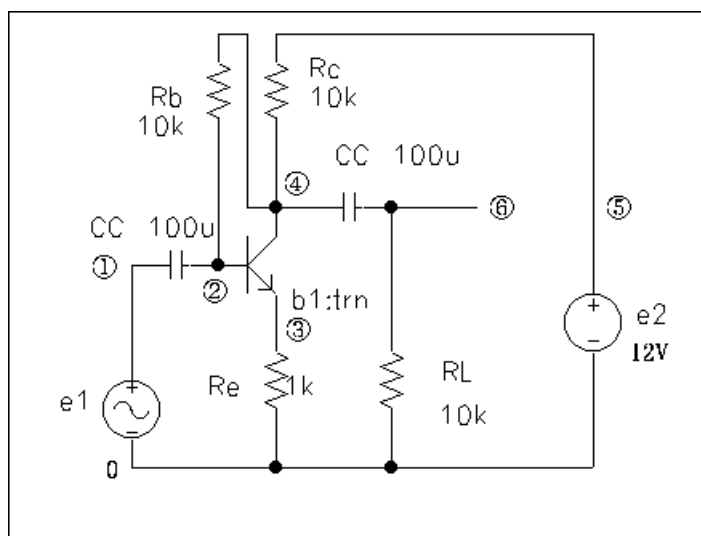


図 s2.sb で設計する回路図

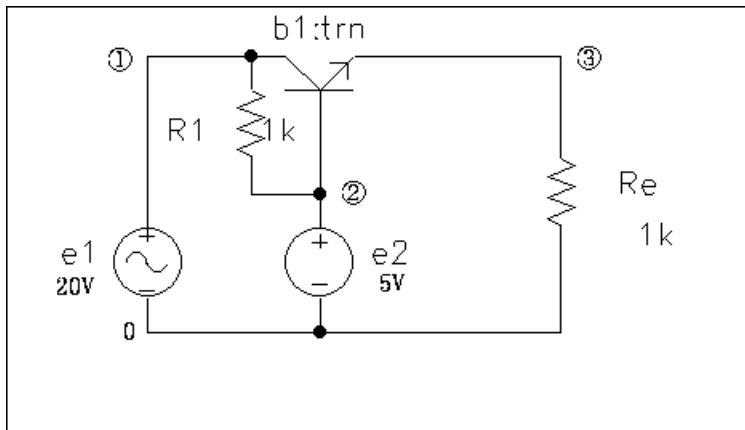


図 s3. sb で設計する回路図

1. サンプル回路の 1 と 2 はエミッタ接地の増幅回路である。サンプル回路の 3 はエミッタホロワの増幅回路である。
2. サンプル回路の 1 と 2 の設計目標は
 1. 無信号時のノード 4 の動作点を電源電圧（12V）の半分の 6V にする。
 2. 周波数 1KHz でのゲインを -10 倍とする。
 3. コレクタ抵抗と負荷抵抗は 10K とする。
 4. カップリングコンデンサは 100uF とする。

従って、上記の目標を達成するために変更可能な素子は R_b と R_e の 2 種類となる。ここで、 R_b は直流動作点を決定する素子であり、 R_e はゲインを決定する素子である。しかし、これらは相互に若干の影響を与えるので数回交互に調整を行なう必要がある。

ここでは、まず 2. のゲインの目標達成のために R_e を変化させてゲインが -10 となる R_e を探す所から始めている。この時、コマンドラインから $f = 1000$ と入力して周波数の設定を行なっている。また、計算結果を表示するノードは 4 を指定している。

ゲインの確認には、コマンド `/ac` を使用する。
 入力信号源は `e1` であり、値は `1` と設定する。
 変化させる素子名は `Re` であり、最低値は `100`、最高値は `1000`、ステップは `100` と指定するとシミュレーションが開始される。
 この時、以下の様に数値が表示される。

```

/ac
データファイル名は ? sl-1
ac 入力信号源名は ? el
値は ? 1

値を変化させる素子名は ? re
ac re 最低値 ? 100
ac re 最高値 ? 1000
ac re ステップ ? 100
·[22me1[1 j 0 ]·[0m
re[100 ] f[1,000 ]
x4 [ -49.3282 j -0.0046 ] abs[ 49.3282] arg[ -179.9947]
re[200 ] f[1,000 ]
x4 [ -24.7082 j -3.702932e-004] abs[ 24.7082] arg[ -179.9991]
re[300 ] f[1,000 ]
x4 [ -16.4820 j 1.845695e-004] abs[ 16.4820] arg[ 179.9994]
re[400 ] f[1,000 ]
x4 [ -12.3652 j 3.005037e-004] abs[ 12.3652] arg[ 179.9986]
re[500 ] f[1,000 ]
x4 [ -9.8939 j 3.182741e-004] abs[ 9.8939] arg[ 179.9982]
re[600 ] f[1,000 ]
x4 [ -8.2459 j 3.085159e-004] abs[ 8.2459] arg[ 179.9979]
re[700 ] f[1,000 ]
x4 [ -7.0685 j 2.909557e-004] abs[ 7.0685] arg[ 179.9976]
re[800 ] f[1,000 ]
x4 [ -6.1853 j 2.719908e-004] abs[ 6.1853] arg[ 179.9975]
re[900 ] f[1,000 ]
x4 [ -5.4984 j 2.538051e-004] abs[ 5.4984] arg[ 179.9974]
re[1000] f[1,000 ]
x4 [ -4.9487 j 2.370918e-004] abs[ 4.9487] arg[ 179.9973]
B?

```

ここで、注目するのは -49.3281 の位置である。

これが、-10 に出来るだけ近づく様な Re を探すことである。

```

値を変化させる素子名は ? re
ac re 最低値 ? 400
ac re 最高値 ? 500
ac re ステップ ? 10
el[1 j 0 ]
re[400 ] f[1,000 ]
x4 [ -12.3652 j 3.005037e-004] abs[ 12.3652] arg[ 179.9986]
re[410 ] f[1,000 ]
x4 [ -12.0638 j 3.047518e-004] abs[ 12.0638] arg[ 179.9986]
re[420 ] f[1,000 ]
x4 [ -11.7768 j 3.082603e-004] abs[ 11.7768] arg[ 179.9985]
re[430 ] f[1,000 ]
x4 [ -11.5032 j 3.111173e-004] abs[ 11.5032] arg[ 179.9985]
re[440 ] f[1,000 ]
x4 [ -11.2420 j 3.134000e-004] abs[ 11.2420] arg[ 179.9984]
re[450 ] f[1,000 ]
x4 [ -10.9923 j 3.151753e-004] abs[ 10.9923] arg[ 179.9984]
re[460 ] f[1,000 ]
x4 [ -10.7536 j 3.165023e-004] abs[ 10.7536] arg[ 179.9983]
re[470 ] f[1,000 ]
x4 [ -10.5249 j 3.174327e-004] abs[ 10.5249] arg[ 179.9983]
re[480 ] f[1,000 ]
x4 [ -10.3058 j 3.180122e-004] abs[ 10.3058] arg[ 179.9982]
re[490 ] f[1,000 ]
x4 [ -10.0957 j 3.182807e-004] abs[ 10.0957] arg[ 179.9982]
re[500 ] f[1,000 ]
x4 [ -9.8939 j 3.182741e-004] abs[ 9.8939] arg[ 179.9982]
B? とりあえず re=490 として、次に直流動作点を決定する
ノード4 が 6v となるように rb を決定する
キー入力待ち? ■

```



```

B? re=490
    490 j 0
B? /dpart
left[ 0] right[ 1] parts[ e1      ] value[      1 j 0      ]
left[ 0] right[ 3] parts[ re       ] value[     490 j 0      ]
left[ 0] right[ 5] parts[ e2       ] value[      12 j 0      ]
left[ 0] right[ 6] parts[ rl       ] value[    1e+004 j 0      ]
left[ 1] right[ 2] parts[ cc       ] value[    0.0001 j 0      ]
left[ 2] right[ 5] parts[ rb       ] value[    1e+005 j 0      ]
left[ 2] right[10] parts[ rlb1     ] value[      0.1 j 0      ]
left[ 3] right[ 7] parts[ rbb1     ] value[      0.1 j 0      ]
left[ 4] right[ 5] parts[ rc       ] value[    1e+004 j 0      ]
left[ 4] right[ 6] parts[ cc       ] value[    0.0001 j 0      ]
left[ 4] right[ 9] parts[ rbb1     ] value[      0.1 j 0      ]
left[ 7] right[ 8] parts[ reb1     ] value[      26 j 0      ]
left[ 7] right[ 9] parts[ kb1      ] value[     100 j 0      ]
@ master[reb1      ] left[ 7] right[ 8]
left[ 8] right[10] parts[ ebb1     ] value[      0.6 j 0      ]
最大の素子番号 = 14
最大のノード番号 = 10
B? 周波数を 0 とする
キー入力待ち?

```

また、直流動作点の決定においてはコマンド `/para` を使用して、素子名 `Rb` を変化させるが、このときには先程の位置の数字が出来るだけ 6 に近づく `Rb` を探すことになる。`/para` を使用する時には、`f=0` と設定することにも注意すべき点である。

```

B? /para
データファイル名は ? s1-3
値を変化させる素子名は ? rb
para rb 最低値 ? 10k
para rb 最高値 ? 100k
para rb ステップ ? 10k
rb[1e+004 ] f[0      ]
x4 [ -179.5123 j 0      ] abs[ 179.5123] arg[      180]
rb[2e+004 ] f[0      ]
x4 [ -151.9670 j 0      ] abs[ 151.9670] arg[      180]
rb[3e+004 ] f[0      ]
x4 [ -131.3490 j 0      ] abs[ 131.3490] arg[      180]
rb[4e+004 ] f[0      ]
x4 [ -115.3370 j 0      ] abs[ 115.3370] arg[      180]
rb[5e+004 ] f[0      ]
x4 [ -102.5427 j 0      ] abs[ 102.5427] arg[      180]
rb[6e+004 ] f[0      ]
x4 [  -92.0847 j 0      ] abs[  92.0847] arg[      180]
rb[7e+004 ] f[0      ]
x4 [  -83.3766 j 0      ] abs[  83.3766] arg[      180]
rb[8e+004 ] f[0      ]
x4 [  -76.0131 j 0      ] abs[  76.0131] arg[      180]
rb[9e+004 ] f[0      ]
x4 [  -69.7051 j 0      ] abs[  69.7051] arg[      180]
rb[1e+005 ] f[0      ]
x4 [  -64.2408 j 0      ] abs[   64.2408] arg[      180]
B? ノード4の値がすべて、マイナスなので rb をさらに大きくして調べる
rb 100k から 1M まで調べる
x4の動作点電圧の変化をs1-4.dに出力する
キー入力待ち?

```

```

データファイル名は ? s1-65
値を変化させる素子名は ? rb
para rb 最低値 ? 1m
para rb 最高値 ? 2m
para rb ステップ ? 100k
rb[1e+006 ] f[0 ]
x4 [ 1.1380 j 0 ] abs[ 1.1380] arg[ 0]
rb[1.1e+006 ] f[0 ]
x4 [ 2.0829 j 0 ] abs[ 2.0829] arg[ 0]
rb[1.2e+006 ] f[0 ]
x4 [ 2.8765 j 0 ] abs[ 2.8765] arg[ 0]
rb[1.3e+006 ] f[0 ]
x4 [ 3.5526 j 0 ] abs[ 3.5526] arg[ 0]
rb[1.4e+006 ] f[0 ]
x4 [ 4.1354 j 0 ] abs[ 4.1354] arg[ 0]
rb[1.5e+006 ] f[0 ]
x4 [ 4.6429 j 0 ] abs[ 4.6429] arg[ 0]
rb[1.6e+006 ] f[0 ]
x4 [ 5.0889 j 0 ] abs[ 5.0889] arg[ 0]
rb[1.7e+006 ] f[0 ]
x4 [ 5.4840 j 0 ] abs[ 5.4840] arg[ 0]
rb[1.8e+006 ] f[0 ]
x4 [ 5.8363 j 0 ] abs[ 5.8363] arg[ 0]
rb[1.9e+006 ] f[0 ]
x4 [ 6.1524 j 0 ] abs[ 6.1524] arg[ 0]
rb[2e+006 ] f[0 ]
x4 [ 6.4377 j 0 ] abs[ 6.4377] arg[ 0]
B? rb が 1.8M と 1.9M の間をさらに調べる
x4の動作点電圧の変化をs1-7.dに出力する
キー入力待ち？
B? /para
データファイル名は ? s1-7
値を変化させる素子名は ? rb
para rb 最低値 ? 1.8m
para rb 最高値 ? 1.9m
para rb ステップ ? 10k
rb[1.8e+006 ] f[0 ]
x4 [ 5.8363 j 0 ] abs[ 5.8363] arg[ 0]
rb[1.81e+006 ] f[0 ]
x4 [ 5.8694 j 0 ] abs[ 5.8694] arg[ 0]
rb[1.82e+006 ] f[0 ]
x4 [ 5.9022 j 0 ] abs[ 5.9022] arg[ 0]
rb[1.83e+006 ] f[0 ]
x4 [ 5.9346 j 0 ] abs[ 5.9346] arg[ 0]
rb[1.84e+006 ] f[0 ]
x4 [ 5.9667 j 0 ] abs[ 5.9667] arg[ 0]
rb[1.85e+006 ] f[0 ]
x4 [ 5.9985 j 0 ] abs[ 5.9985] arg[ 0]
rb[1.86e+006 ] f[0 ]
x4 [ 6.0299 j 0 ] abs[ 6.0299] arg[ 0]
rb[1.87e+006 ] f[0 ]
x4 [ 6.0610 j 0 ] abs[ 6.0610] arg[ 0]
rb[1.88e+006 ] f[0 ]
x4 [ 6.0918 j 0 ] abs[ 6.0918] arg[ 0]
rb[1.89e+006 ] f[0 ]
x4 [ 6.1223 j 0 ] abs[ 6.1223] arg[ 0]
rb[1.9e+006 ] f[0 ]
x4 [ 6.1524 j 0 ] abs[ 6.1524] arg[ 0]
B? rb=1.85M と決定する
キー入力待ち？

```

s1.sb を実行すると、

1. 直流動作点は $R_b=1.85M$ によって 6 V になり、
2. 交流ゲインは $R_e=495$ オームによって、 -10 倍となることが分かる。

s2. sb を実行すると、

```
ac 入力信号源名は ? e1
値は ? 1

値を変化させる素子名は ? re
ac re 最低値 ? 100
ac re 最高値 ? 1000
ac re ステップ ? 100
e1[1      j 0      ]
re[100    ] f[1,000 ]
x4 [ -32.5511 j -0.1772 ] abs[ 32.5516] arg[ -179.6881]
re[200    ] f[1,000 ]
x4 [ -16.1387 j -0.0444 ] abs[ 16.1387] arg[ -179.8422]
re[300    ] f[1,000 ]
x4 [ -10.6546 j -0.0198 ] abs[ 10.6546] arg[ -179.8938]
re[400    ] f[1,000 ]
x4 [ -7.9101 j -0.0111 ] abs[ 7.9101] arg[ -179.9195]
re[500    ] f[1,000 ]
x4 [ -6.2626 j -0.0071 ] abs[ 6.2626] arg[ -179.9350]
re[600    ] f[1,000 ]
x4 [ -5.1639 j -0.0049 ] abs[ 5.1639] arg[ -179.9453]
re[700    ] f[1,000 ]
x4 [ -4.3790 j -0.0036 ] abs[ 4.3790] arg[ -179.9527]
re[800    ] f[1,000 ]
x4 [ -3.7902 j -0.0028 ] abs[ 3.7902] arg[ -179.9582]
re[900    ] f[1,000 ]
x4 [ -3.3322 j -0.0022 ] abs[ 3.3322] arg[ -179.9625]
re[1000   ] f[1,000 ]
x4 [ -2.9658 j -0.0018 ] abs[ 2.9658] arg[ -179.9660]
B? re: 300 から 400 の間でゲインが -10 となる
```

```
ac re 最低値 ? 300
ac re 最高値 ? 400
ac re ステップ ? 10
e1[1      j 0      ]
re[300    ] f[1,000 ]
x4 [ -10.6546 j -0.0198 ] abs[ 10.6546] arg[ -179.8938]
re[310    ] f[1,000 ]
x4 [ -10.3006 j -0.0185 ] abs[ 10.3006] arg[ -179.8971]
re[320    ] f[1,000 ]
x4 [ -9.9686 j -0.0174 ] abs[ 9.9686] arg[ -179.9002]
re[330    ] f[1,000 ]
x4 [ -9.6568 j -0.0163 ] abs[ 9.6568] arg[ -179.9031]
re[340    ] f[1,000 ]
x4 [ -9.3633 j -0.0154 ] abs[ 9.3633] arg[ -179.9059]
re[350    ] f[1,000 ]
x4 [ -9.0865 j -0.0145 ] abs[ 9.0865] arg[ -179.9085]
re[360    ] f[1,000 ]
x4 [ -8.8251 j -0.0137 ] abs[ 8.8251] arg[ -179.9109]
re[370    ] f[1,000 ]
x4 [ -8.5778 j -0.0130 ] abs[ 8.5778] arg[ -179.9133]
re[380    ] f[1,000 ]
x4 [ -8.3435 j -0.0123 ] abs[ 8.3436] arg[ -179.9155]
re[390    ] f[1,000 ]
x4 [ -8.1213 j -0.0117 ] abs[ 8.1213] arg[ -179.9176]
re[400    ] f[1,000 ]
x4 [ -7.9101 j -0.0111 ] abs[ 7.9101] arg[ -179.9195]
B? とりあえず re=310 として、次に直流動作点を決定する
ノード4 が 6v となるように rb を決定する
キー入力待ち? ■
```

```

データファイル名は ? s2-3
値を変化させる素子名は ? rb
para rb 最低値 ? 10k
para rb 最高値 ? 100k
para rb ステップ ? 10k
rb[1e+004 ] f[0 ]
x4 [ 1.0483 j 0 ] abs[ 1.0483] arg[ 0]
rb[2e+004 ] f[0 ]
x4 [ 1.1515 j 0 ] abs[ 1.1515] arg[ 0]
rb[3e+004 ] f[0 ]
x4 [ 1.2528 j 0 ] abs[ 1.2528] arg[ 0]
rb[4e+004 ] f[0 ]
x4 [ 1.3522 j 0 ] abs[ 1.3522] arg[ 0]
rb[5e+004 ] f[0 ]
x4 [ 1.4497 j 0 ] abs[ 1.4497] arg[ 0]
rb[6e+004 ] f[0 ]
x4 [ 1.5455 j 0 ] abs[ 1.5455] arg[ 0]
rb[7e+004 ] f[0 ]
x4 [ 1.6396 j 0 ] abs[ 1.6396] arg[ 0]
rb[8e+004 ] f[0 ]
x4 [ 1.7320 j 0 ] abs[ 1.7320] arg[ 0]
rb[9e+004 ] f[0 ]
x4 [ 1.8227 j 0 ] abs[ 1.8227] arg[ 0]
rb[1e+005 ] f[0 ]
x4 [ 1.9119 j 0 ] abs[ 1.9119] arg[ 0]
B? ノード4の値がすべて、6V 以下なので rb をさらに大きくして調べる
rb 100k から 1M まで調べる
s2-4を出力する
キー入力待ち？

```

```

B? /para
データファイル名は ? s2-4
値を変化させる素子名は ? rb
para rb 最低値 ? 100k
para rb 最高値 ? 1m
para rb ステップ ? 100k
rb[1e+005 ] f[0 ]
x4 [ 1.9119 j 0 ] abs[ 1.9119] arg[ 0]
rb[2e+005 ] f[0 ]
x4 [ 2.7246 j 0 ] abs[ 2.7246] arg[ 0]
rb[3e+005 ] f[0 ]
x4 [ 3.4161 j 0 ] abs[ 3.4161] arg[ 0]
rb[4e+005 ] f[0 ]
x4 [ 4.0116 j 0 ] abs[ 4.0116] arg[ 0]
rb[5e+005 ] f[0 ]
x4 [ 4.5299 j 0 ] abs[ 4.5299] arg[ 0]
rb[6e+005 ] f[0 ]
x4 [ 4.9850 j 0 ] abs[ 4.9850] arg[ 0]
rb[7e+005 ] f[0 ]
x4 [ 5.3879 j 0 ] abs[ 5.3879] arg[ 0]
rb[8e+005 ] f[0 ]
x4 [ 5.7470 j 0 ] abs[ 5.7470] arg[ 0]
rb[9e+005 ] f[0 ]
x4 [ 6.0691 j 0 ] abs[ 6.0691] arg[ 0]
rb[1e+006 ] f[0 ]
x4 [ 6.3596 j 0 ] abs[ 6.3596] arg[ 0]
B? rb が 800K と 900K の間で、ノード4が 6v となるので、
とりあえず rb=800K と決定する
キー入力待ち？

```

```

値を変化させる素子名は ? re
ac re 最低値 ? 400
ac re 最高値 ? 500
ac re ステップ ? 10
el[1] j 0
re[400] f[1,000]
x4 [-12.2821 j 1.633487e-004] abs[ 12.2821] arg[ 179.9992]
re[410] f[1,000]
x4 [-11.9827 j 1.779979e-004] abs[ 11.9827] arg[ 179.9991]
re[420] f[1,000]
x4 [-11.6975 j 1.910844e-004] abs[ 11.6975] arg[ 179.9991]
re[430] f[1,000]
x4 [-11.4255 j 2.027757e-004] abs[ 11.4255] arg[ 179.9990]
re[440] f[1,000]
x4 [-11.1659 j 2.132196e-004] abs[ 11.1659] arg[ 179.9989]
re[450] f[1,000]
x4 [-10.9179 j 2.225459e-004] abs[ 10.9179] arg[ 179.9988]
re[460] f[1,000]
x4 [-10.6806 j 2.308688e-004] abs[ 10.6806] arg[ 179.9988]
re[470] f[1,000]
x4 [-10.4534 j 2.382899e-004] abs[ 10.4534] arg[ 179.9987]
re[480] f[1,000]
x4 [-10.2356 j 2.448990e-004] abs[ 10.2356] arg[ 179.9986]
re[490] f[1,000]
x4 [-10.0267 j 2.507757e-004] abs[ 10.0267] arg[ 179.9986]
re[500] f[1,000]
x4 [-9.8262 j 2.559915e-004] abs[ 9.8262] arg[ 179.9985]
B? re=490 に決定する
キー入力待ち? _

```

```

値を変化させる素子名は ? re
ac re 最低値 ? 490
ac re 最高値 ? 500
ac re ステップ ? 1
el[1] j 0
re[490] f[1,000]
x4 [-10.0267 j 2.507757e-004] abs[ 10.0267] arg[ 179.9986]
re[491] f[1,000]
x4 [-10.0063 j 2.513260e-004] abs[ 10.0063] arg[ 179.9986]
re[492] f[1,000]
x4 [-9.9860 j 2.518697e-004] abs[ 9.9860] arg[ 179.9986]
re[493] f[1,000]
x4 [-9.9657 j 2.524069e-004] abs[ 9.9657] arg[ 179.9985]
re[494] f[1,000]
x4 [-9.9456 j 2.529376e-004] abs[ 9.9456] arg[ 179.9985]
re[495] f[1,000]
x4 [-9.9255 j 2.534621e-004] abs[ 9.9255] arg[ 179.9985]
re[496] f[1,000]
x4 [-9.9055 j 2.539802e-004] abs[ 9.9055] arg[ 179.9985]
re[497] f[1,000]
x4 [-9.8855 j 2.544923e-004] abs[ 9.8855] arg[ 179.9985]
re[498] f[1,000]
x4 [-9.8657 j 2.549980e-004] abs[ 9.8657] arg[ 179.9985]
re[499] f[1,000]
x4 [-9.8459 j 2.554977e-004] abs[ 9.8459] arg[ 179.9985]
re[500] f[1,000]
x4 [-9.8262 j 2.559915e-004] abs[ 9.8262] arg[ 179.9985]
B? 最終決定 re=491 とする
キー入力待ち?

```

```

B? /para
データファイル名は ? s2-8
値を変化させる素子名は ? rb
para rb 最低値 ? 800k
para rb 最高値 ? 900k
para rb ステップ ? 10k
rb[8e+005 ] f[0 ]
x4 [ 5.8084 j 0 ] abs[ 5.8084] arg[ 0]
rb[8.1e+005 ] f[0 ]
x4 [ 5.8416 j 0 ] abs[ 5.8416] arg[ 0]
rb[8.2e+005 ] f[0 ]
x4 [ 5.8743 j 0 ] abs[ 5.8743] arg[ 0]
rb[8.3e+005 ] f[0 ]
x4 [ 5.9067 j 0 ] abs[ 5.9067] arg[ 0]
rb[8.4e+005 ] f[0 ]
x4 [ 5.9388 j 0 ] abs[ 5.9388] arg[ 0]
rb[8.5e+005 ] f[0 ]
x4 [ 5.9706 j 0 ] abs[ 5.9706] arg[ 0]
rb[8.6e+005 ] f[0 ]
x4 [ 6.0020 j 0 ] abs[ 6.0020] arg[ 0]
rb[8.7e+005 ] f[0 ]
x4 [ 6.0330 j 0 ] abs[ 6.0330] arg[ 0]
rb[8.8e+005 ] f[0 ]
x4 [ 6.0638 j 0 ] abs[ 6.0638] arg[ 0]
rb[8.9e+005 ] f[0 ]
x4 [ 6.0943 j 0 ] abs[ 6.0943] arg[ 0]
rb[9e+005 ] f[0 ]
x4 [ 6.1244 j 0 ] abs[ 6.1244] arg[ 0]
B? 最終決定 rb=860K とする
rb=860k

```

1. 直流動作点は $R_b=860K$ によって $6V$ になり、
2. 交流ゲインは $R_e=491$ オームによって、 -10 倍となることが分かる。

s3. sb を実行すると、

```

B? /para
データファイル名は ? s3-1
値を変化させる素子名は ? e1
para e1 最低値 ? 5
para e1 最高値 ? 20
para e1 ステップ ? 5
e1[5 ] f[0 ]
x3 [ 4.3984 j 0 ] abs[ 4.3984] arg[ 0]
e1[10 ] f[0 ]
x3 [ 4.3984 j 0 ] abs[ 4.3984] arg[ 0]
e1[15 ] f[0 ]
x3 [ 4.3984 j 0 ] abs[ 4.3984] arg[ 0]
e1[20 ] f[0 ]
x3 [ 4.3984 j 0 ] abs[ 4.3984] arg[ 0]
B? e1 が変化しても、出力電圧はほとんど変化しないことが分かる
次は、e2 を変化させてみる 1 から 10 V
s3-2を出力する
キー入力待ち? _

```

```

B? /para
データファイル名は ? s3-2
値を変化させる素子名は ? e2
para e2 最低値 ? 1
para e2 最高値 ? 10
para e2 ステップ ? 1
e2[1] [ 0.3999 j 0 ] ] abs[ 0.3999] arg[ 0]
x3 [ 0.3999 j 0 ] ] abs[ 0.3999] arg[ 0]
e2[2] [ 1.3995 j 0 ] ] abs[ 1.3995] arg[ 0]
x3 [ 1.3995 j 0 ] ] abs[ 1.3995] arg[ 0]
e2[3] [ 2.3991 j 0 ] ] abs[ 2.3991] arg[ 0]
x3 [ 2.3991 j 0 ] ] abs[ 2.3991] arg[ 0]
e2[4] [ 3.3988 j 0 ] ] abs[ 3.3988] arg[ 0]
x3 [ 3.3988 j 0 ] ] abs[ 3.3988] arg[ 0]
e2[5] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
e2[6] [ 5.3981 j 0 ] ] abs[ 5.3981] arg[ 0]
x3 [ 5.3981 j 0 ] ] abs[ 5.3981] arg[ 0]
e2[7] [ 6.3977 j 0 ] ] abs[ 6.3977] arg[ 0]
x3 [ 6.3977 j 0 ] ] abs[ 6.3977] arg[ 0]
e2[8] [ 7.3973 j 0 ] ] abs[ 7.3973] arg[ 0]
x3 [ 7.3973 j 0 ] ] abs[ 7.3973] arg[ 0]
e2[9] [ 8.3970 j 0 ] ] abs[ 8.3970] arg[ 0]
x3 [ 8.3970 j 0 ] ] abs[ 8.3970] arg[ 0]
e2[10] [ 9.3966 j 0 ] ] abs[ 9.3966] arg[ 0]
x3 [ 9.3966 j 0 ] ] abs[ 9.3966] arg[ 0]
B? 出力電圧は e2=0.6 V になっていることが分かる
負荷抵抗 re を変化させてみる 100 から 1k
s3-3を出力する
キー入力待ち? _

/para
データファイル名は ? s3-6
値を変化させる素子名は ? r1
para r1 最低値 ? 1k
para r1 最高値 ? 10k
para r1 ステップ ? 1k
r1[1000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[2000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[3000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[4000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[5000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[6000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[7000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[8000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[9000] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
r1[1e+004] [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
x3 [ 4.3984 j 0 ] ] abs[ 4.3984] arg[ 0]
B? 以上で、この回路では e2 の値が出力電圧を決定していることが分かる。
この回路形式は、安定化電源の基本形として利用されている
また、この回路は出力インピーダンスが低いのでバッファアンプとして使われる
1 番のパッチファイル s3.sb を終了します。
? _

```

1. エミッタホロワ回路では、e1 の電圧が変化しても、エミッタ電圧はほとんど変化しない事が分かる。
2. エミッタホロワ回路では、エミッタ電圧はベース電圧 -0.6 V 程度になることが分かる。

Sim.exe で使用する標準部品として準備してあるブロック素子について

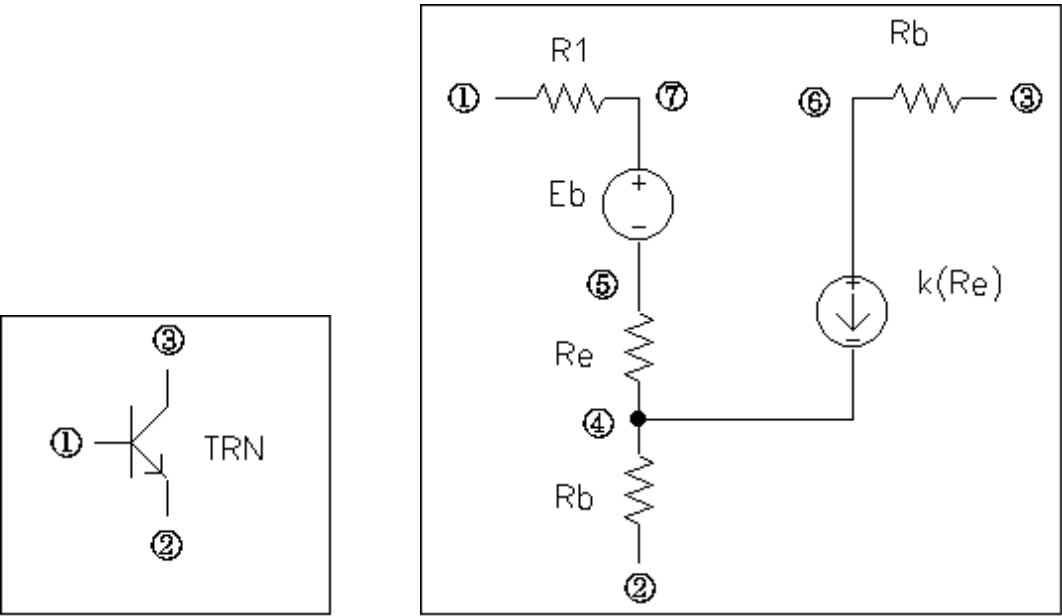


図 npn トランジスタの回路図記号と等価回路 TRN. CIR, TRN2. CIR

npn トランジスタは2種類のブロック素子を準備してある。ブロック素子名（ファイル名）はそれぞれ trn. cir と trn2. cir である。これらは、等価回路の素子の値が異なるだけである。

素子名	trn. cir	trn2. cir
R 1	0. 1	1 0 0
R e	2 6	2 6
R b	0. 1	0. 1
k (R e)	1 0 0	5 0
E b	0. 6	0. 7

R 1 はトランジスタの規格表では h_{ie} に相当します。k は同様に h_{fe} （または β ）に相当します。 $h_{ie} = 26\text{mV} / (I_c / h_{fe})$ により求められます。

例えば、 $I_c=10\text{mA}$, $h_{fe}=100$ なら $R1 = h_{ie} = 260$ オームとなります。

この等価回路では正確にダイオードの整流機能がシミュレーション出来ないため、実際のトランジスタの動作とは異なります。すなわち、ベース・エミッタ間が逆バイアスとなる入力電圧に対して、通常はベース電流はほとんど流れませんがこの等価回路では逆方向にも入力電圧に比例したベース電流が流れてしまいます。従って、適正な直流バイアス点を決定し小信号の入力に対してのシミュレーションにのみ用いるべきです。

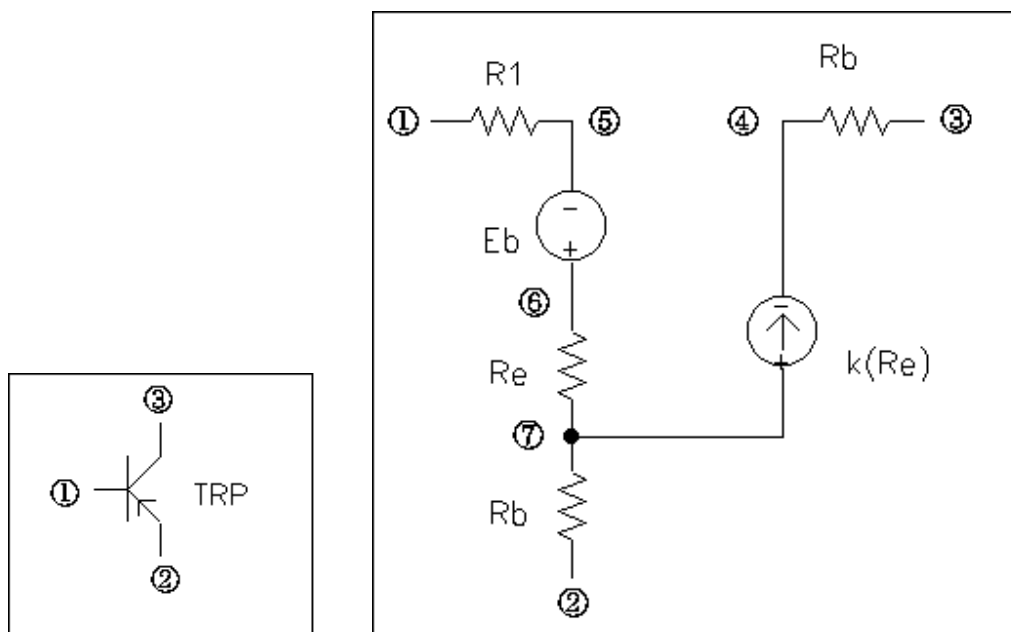


図 pnp トランジスタの回路図記号と等価回路 TRP. CIR, TRP2. CIR

pnp トランジスタは2種類のブロック素子を準備してある。ブロック素子名（ファイル名）はそれぞれ `trp.cir` と `trp2.cir` である。これらは、等価回路の素子の値が異なるだけである。

素子名	<code>trp.cir</code>	<code>trp2.cir</code>
R 1	0. 1	1 0 0
R e	2 6	2 6
R b	0. 1	0. 1
k (R e)	1 0 0	5 0
E b	0. 6	0. 7

R 1 はトランジスタの規格表では h_{ie} に相当します。k は同様に h_{fe} （または β ）に相当します。 $h_{ie} = 26\text{mV} / (I_c / h_{fe})$ により求められます。

例えば、 $I_c=10\text{mA}$, $h_{fe}=100$ なら $R1 = h_{ie} = 260$ オームとなります。

この等価回路では正確にダイオードの整流機能がシミュレーション出来ないため、実際のトランジスタの動作とは異なります。すなわち、ベース・エミッタ間が逆バイアスとなる入力電圧に対して、通常はベース電流はほとんど流れませんがこの等価回路では逆方向にも入力電圧に比例したベース電流が流れてしまいます。従って、適正な直流バイアス点を決定し小信号の入力に対してのシミュレーションにのみ用いるべきです。

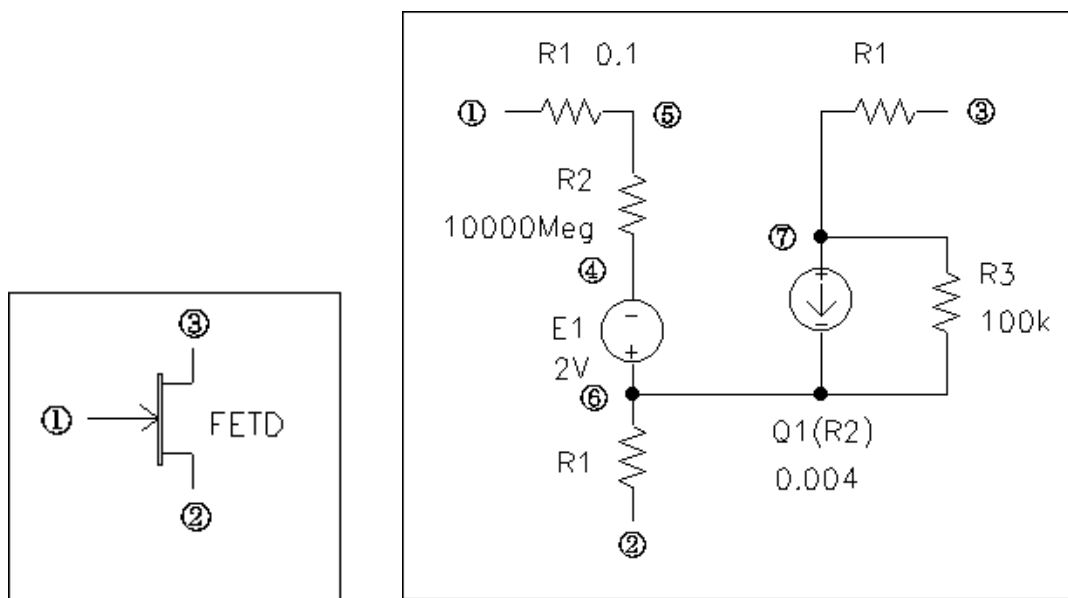


図 n型デプリーションタイプFETの回路図記号と等価回路 FETD.CIR

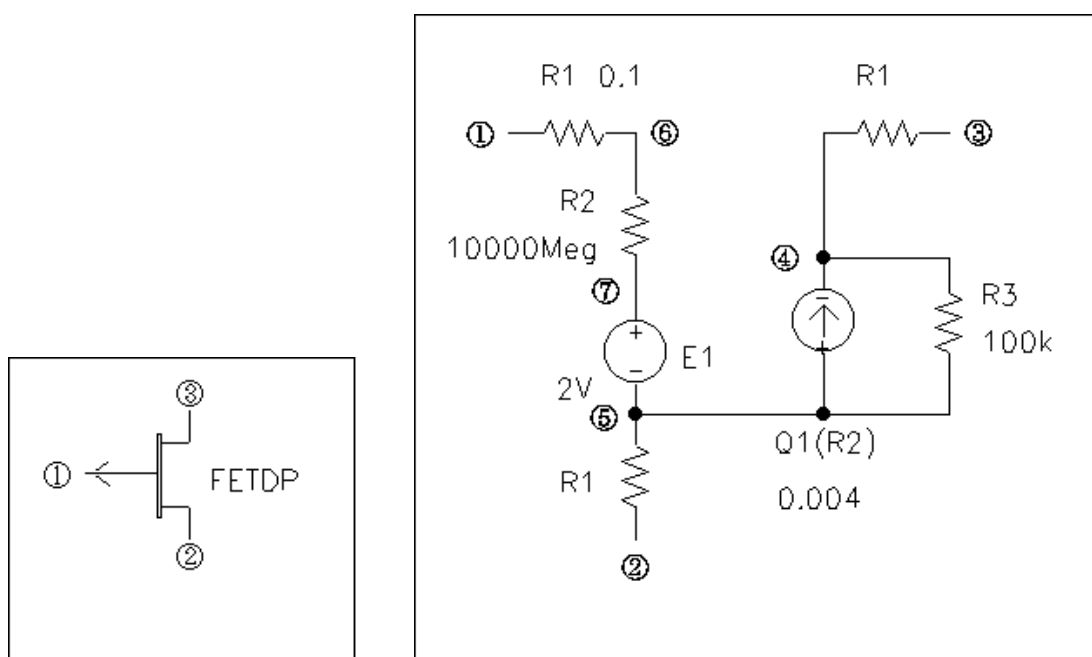


図 p型デプリーションタイプFETの回路図記号と等価回路 FETDP.CIR

実際にシミュレーションを行うときには、素子の規格表を参考にして、E1，R3，Q1の値を適正な値に変更してから使用する。
また、Q1の値については、動作点に合った値に設定する必要がある。

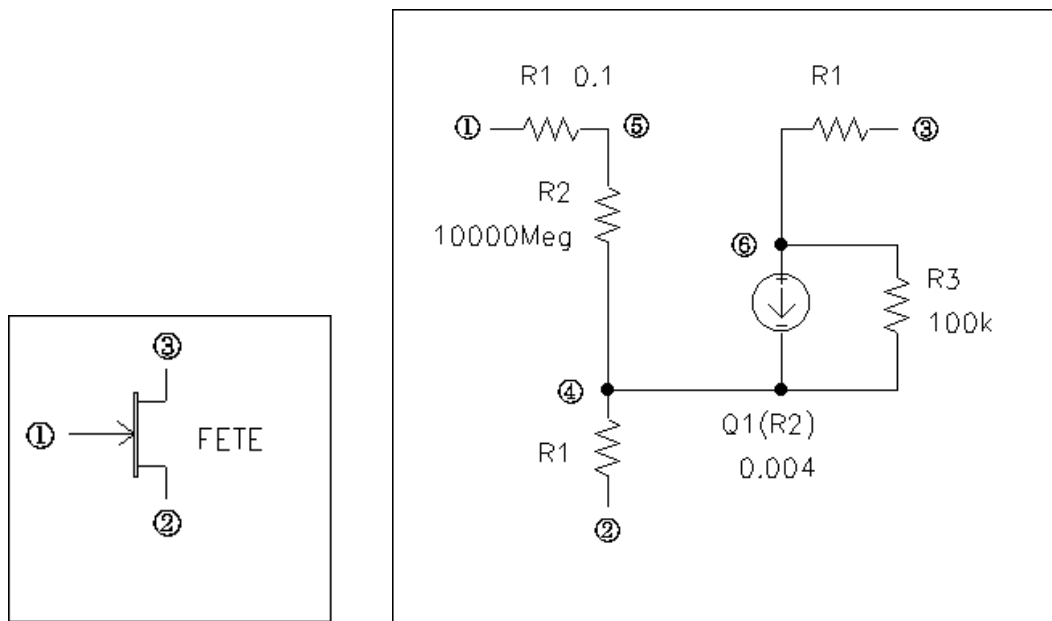


図 n型エンハンスメントタイプFETの回路図記号と等価回路 FETE. CIR

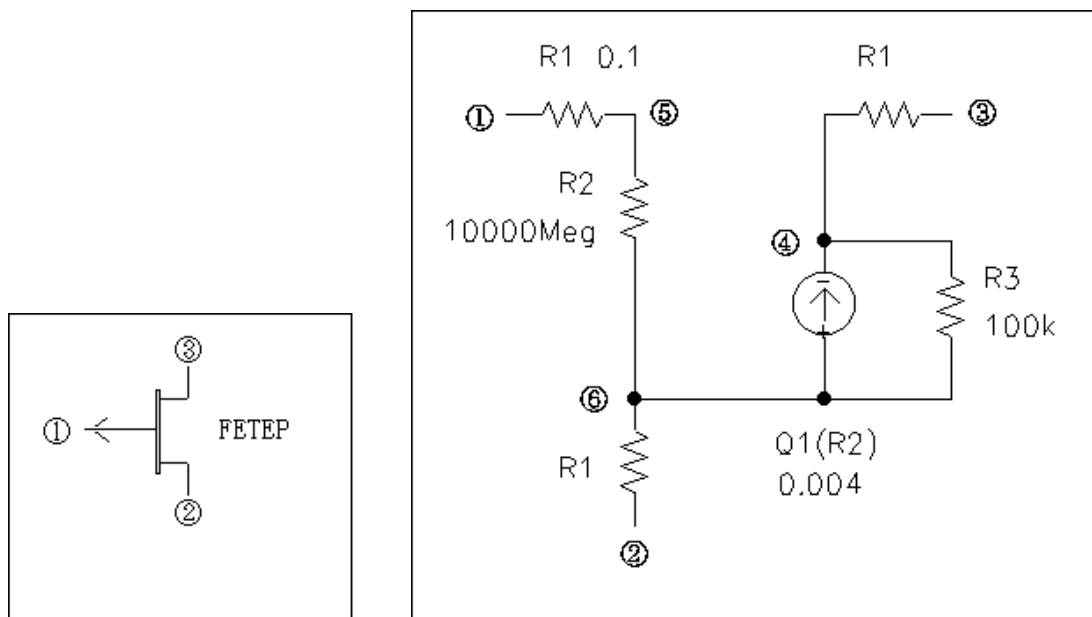


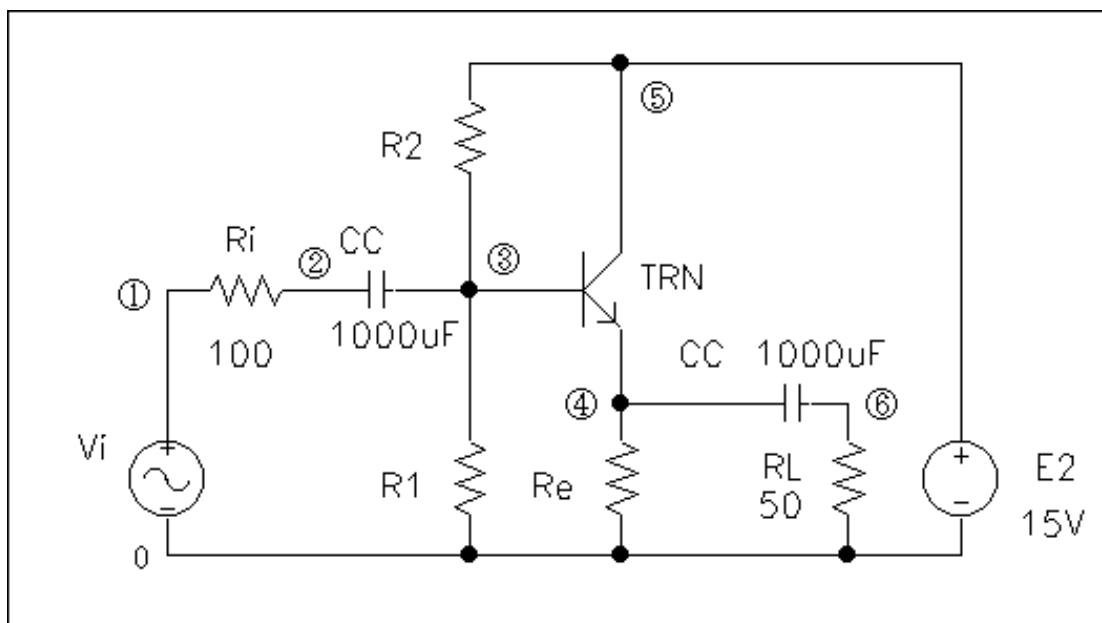
図 p型エンハンスメントタイプFETの回路図記号と等価回路 FETEP. CIR

上で示した他に、多数のトランジスタ、FET、オペアンプ、BPF回路、HPF回路、LC回路などの部品回路(*.cir)が準備されている。

サンプルバッチファイルの回路図と説明

(1) バッチファイル名 s 5. s b で扱う回路図

エミッタホロワ増幅回路



設計目標

1. 付加抵抗 R_L の両端に発生する出力電圧の入力信号に対する比率 (電圧利得) は最低でも 0.9 を確保する。(エミッタホロワ回路の電圧ゲインは最大でも 1 である)
2. 入力信号の振幅は、 -4 V から $+4\text{ V}$ までとする。
3. 入力信号源抵抗は $R_i = 100$ オームとする。
4. 負荷抵抗は $R_L = 50$ オームとする。
5. npn 型トランジスタ TRN の h_{fe} (等価回路では、 k_{b1} となる) は 100 から 200 までのばらつきがあるものとする。
6. $V_{ce(sat)} = 1\text{ V}$ とする。
7. カップリングコンデンサ CC は入力信号周波数の範囲に対して無視出来るインピーダンスとする。

s 5. s b の概略説明

1. ノード4の直流動作点を決定する。

$V_{ce}(sat) < V_{ce}$ 及び $0 < x_4$ から $x_4 = 7$ とする

```
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[3000 ] f[0 ]
x4 [ 5.2906 j 0 ] abs[ 5.2906 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[4000 ] f[0 ]
x4 [ 5.9195 j 0 ] abs[ 5.9195 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[5000 ] f[0 ]
x4 [ 6.3658 j 0 ] abs[ 6.3658 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[6000 ] f[0 ]
x4 [ 6.6989 j 0 ] abs[ 6.6989 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[7000 ] f[0 ]
x4 [ 6.9571 j 0 ] abs[ 6.9571 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[8000 ] f[0 ]
x4 [ 7.1630 j 0 ] abs[ 7.1630 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[9000 ] f[0 ]
x4 [ 7.3311 j 0 ] abs[ 7.3311 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
r1[1e+004 ] f[0 ]
x4 [ 7.4709 j 0 ] abs[ 7.4709 ] arg[ 0 ]
x6 [ 0 j 0 ] abs[ 0 ] arg[ 0 ]
B?
r1 = 8K とします。この時、 x4 = 7.1630V
利得を確認します。
```

2. R 1, R 2 の決定

a. 入力インピーダンスによる条件

b. ノード4の直流動作点による条件

により、とりあえず $R_2 = 3\text{ K}$ とする。次に、/para を使用してノード4が 7 V となる

R_1 を求める。 $R_1 = 8\text{ K}$ が得られる。

3. ゲインを確認する

```
利得を確認します。
キー入力待ち？
B? r1=8k
   8,000 j 0
B? f=1000
   1,000 j 0
B? /ac
ac 入力信号源名は ? e1
値は ? 1
値を変化させる素子名は ? kb1
ac kb1 最低値 ? 100
ac kb1 最高値 ? 200
ac kb1 ステップ ? 50
e1[1 ] f[1,000 ]
kb1[100 ] f[1,000 ]
x4 [ 0.9087 j 4.139658e-005 ] abs[ 0.9087 ] arg[ 0.0026 ]
x6 [ 0.9087 j 0.0029 ] abs[ 0.9087 ] arg[ 0.1850 ]
kb1[150 ] f[1,000 ]
x4 [ 0.9227 j 4.731972e-005 ] abs[ 0.9227 ] arg[ 0.0029 ]
x6 [ 0.9227 j 0.0030 ] abs[ 0.9227 ] arg[ 0.1853 ]
kb1[200 ] f[1,000 ]
x4 [ 0.9299 j 5.042787e-005 ] abs[ 0.9299 ] arg[ 0.0031 ]
x6 [ 0.9299 j 0.0030 ] abs[ 0.9299 ] arg[ 0.1855 ]
B? キー入力待ち？
```

h f e = 1 0 0 の時に、あまり余裕がないので、ノード4の直流動作点を8 Vにあげてみる。

/para を使用してR 1 を求めると、R 1 = 1 5 Kとなり、今度はh f e = 1 0 0 のときにも余裕がある。

```
B? /para
値を変化させる素子名は ? r1
para r1 最低値 ? 8k
para r1 最高値 ? 15k
para r1 ステップ ? 1k
r1[8000 ] f[0 ]
x4 [ 7.1630 j 0 ] abs[ 7.1630] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[9000 ] f[0 ]
x4 [ 7.3311 j 0 ] abs[ 7.3311] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[1e+004 ] f[0 ]
x4 [ 7.4709 j 0 ] abs[ 7.4709] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[1.1e+004 ] f[0 ]
x4 [ 7.5890 j 0 ] abs[ 7.5890] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[1.2e+004 ] f[0 ]
x4 [ 7.6901 j 0 ] abs[ 7.6901] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[1.3e+004 ] f[0 ]
x4 [ 7.7777 j 0 ] abs[ 7.7777] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[1.4e+004 ] f[0 ]
x4 [ 7.8542 j 0 ] abs[ 7.8542] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
r1[1.5e+004 ] f[0 ]
x4 [ 7.9216 j 0 ] abs[ 7.9216] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
B? キー入力待ち?
B?
r1 = 15K とします。この時の利得を確認します。
キー入力待ち?
B? r1=15k
15,000 j 0
B? f=1000
1,000 j 0
B? /ac
ac 入力信号源名は ? e1
値は ? 1
値を変化させる素子名は ? kb1
ac kb1 最低値 ? 100
ac kb1 最高値 ? 200
ac kb1 ステップ ? 50
e1[1 ] f[1,000 ]
kb1[100 ] f[1,000 ]
x4 [ 0.9136 j 3.409550e-005] abs[ 0.9136] arg[ 0.0021]
x6 [ 0.9136 j 0.0029 ] abs[ 0.9136] arg[ 0.1845]
kb1[150 ] f[1,000 ]
x4 [ 0.9278 j 3.984743e-005] abs[ 0.9278] arg[ 0.0025]
x6 [ 0.9278 j 0.0030 ] abs[ 0.9278] arg[ 0.1848]
kb1[200 ] f[1,000 ]
x4 [ 0.9350 j 4.286738e-005] abs[ 0.9350] arg[ 0.0026]
x6 [ 0.9350 j 0.0030 ] abs[ 0.9350] arg[ 0.1850]
B? キー入力待ち?

```

4. 出力インピーダンスの確認

負荷抵抗 $R_L = 1\text{ M}$ （無負荷の状態）として、ノード6の出力電圧が1 Vとなる入力信号の電圧を求める。これをaとする。

/acによって、入力信号の電圧がaの時に、ノード6の出力電圧が0.5 V（半分）になる

R_L の値を求める。これが出力インピーダンスの値である。

$Z_o = 1.3\text{ オーム}$ が得られる。

エミッタホロワの出力インピーダンスはかなり小さいことが分かる。

```

B? /ac
ac 入力信号源名は ? e1
値は ? 1

値を変化させる素子名は ? e1
ac e1 最低値 ? 1
ac e1 最高値 ? 1
ac e1 ステップ ? 1
e1[1] j 0
e1[1] f[1,000]
x4 [ 0.9370 j 8.396168e-005] abs[ 0.9370] arg[ 0.0051]
x6 [ 0.9370 j 8.411081e-005] abs[ 0.9370] arg[ 0.0051]
B? キー入力待ち？

B?
この状態で x6 = 1 となる e1 を求めます
a=1/x6
1.0673 j -9.580561e-005

B? /ac
ac 入力信号源名は ? e1
値は ? a

値を変化させる素子名は ? r1
ac r1 最低値 ? 1m
ac r1 最高値 ? 1m
ac r1 ステップ ? 1
e1[1.067 j -9.581e-005]
r1[1e+006] f[1,000]
x4 [ 1 j -1.591549e-007] abs[ 1] arg[-9.118907e-006]
x6 [ 1 j 0] abs[ 1] arg[ 0]
B?
予定通りに x6 = 1 となりました。
キー入力待ち？

B?
次に、r1 を変化させて x6 = 0.5 となる、r1 を求めます
エミッタホロワの出力インピーダンスは re/10 以下と考えられます

キー入力待ち？

B? /ac
ac 入力信号源名は ? e1
値は ? a

値を変化させる素子名は ? r1
ac r1 最低値 ? 1
ac r1 最高値 ? 5
ac r1 ステップ ? 1
e1[1.067 j -9.581e-005]
r1[1] f[1,000]
x4 [ 0.4419 j -0.0387] abs[ 0.4436] arg[ -5.0101]
x6 [ 0.4370 j 0.0308] abs[ 0.4381] arg[ 4.0330]
r1[2] f[1,000]
x4 [ 0.6112 j -0.0186] abs[ 0.6115] arg[ -1.7452]
x6 [ 0.6089 j 0.0298] abs[ 0.6096] arg[ 2.8047]
r1[3] f[1,000]
x4 [ 0.7018 j -0.0109] abs[ 0.7019] arg[ -0.8872]
x6 [ 0.7004 j 0.0263] abs[ 0.7009] arg[ 2.1496]
r1[4] f[1,000]
x4 [ 0.7582 j -0.0071] abs[ 0.7583] arg[ -0.5360]
x6 [ 0.7573 j 0.0230] abs[ 0.7577] arg[ 1.7425]
r1[5] f[1,000]
x4 [ 0.7967 j -0.0050] abs[ 0.7967] arg[ -0.3581]
x6 [ 0.7960 j 0.0204] abs[ 0.7963] arg[ 1.4651]
B? キー入力待ち？

```

```

B? /ac
ac 入力信号源名は ? e1
値は ? a

値を変化させる素子名は ? r1
ac r1 最低値 ? 1
ac r1 最高値 ? 2
ac r1 ステップ ? 0.1
e1[1.067 j -9.581e-005]
r1[1 j f[1,000] ]
x4 [ 0.4419 j -0.0387 ] abs[ 0.4436] arg[ -5.0101]
x6 [ 0.4370 j 0.0308 ] abs[ 0.4381] arg[ 4.0330]
r1[1.1 j f[1,000] ]
x4 [ 0.4652 j -0.0355 ] abs[ 0.4665] arg[ -4.3690]
x6 [ 0.4607 j 0.0311 ] abs[ 0.4617] arg[ 3.8638]
r1[1.2 j f[1,000] ]
x4 [ 0.4866 j -0.0327 ] abs[ 0.4877] arg[ -3.8467]
x6 [ 0.4824 j 0.0313 ] abs[ 0.4834] arg[ 3.7083]
r1[1.3 j f[1,000] ]
x4 [ 0.5064 j -0.0302 ] abs[ 0.5073] arg[ -3.4150]
x6 [ 0.5025 j 0.0313 ] abs[ 0.5035] arg[ 3.5648]
r1[1.4 j f[1,000] ]
x4 [ 0.5247 j -0.0280 ] abs[ 0.5254] arg[ -3.0537]
x6 [ 0.5211 j 0.0313 ] abs[ 0.5220] arg[ 3.4319]
r1[1.5 j f[1,000] ]
x4 [ 0.5417 j -0.0260 ] abs[ 0.5423] arg[ -2.7480]
x6 [ 0.5384 j 0.0311 ] abs[ 0.5393] arg[ 3.3086]
r1[1.6 j f[1,000] ]
x4 [ 0.5575 j -0.0242 ] abs[ 0.5580] arg[ -2.4868]
x6 [ 0.5544 j 0.0309 ] abs[ 0.5553] arg[ 3.1939]
r1[1.7 j f[1,000] ]
x4 [ 0.5723 j -0.0226 ] abs[ 0.5727] arg[ -2.2617]
x6 [ 0.5694 j 0.0307 ] abs[ 0.5702] arg[ 3.0868]
r1[1.8 j f[1,000] ]
x4 [ 0.5861 j -0.0211 ] abs[ 0.5865] arg[ -2.0663]
x6 [ 0.5834 j 0.0304 ] abs[ 0.5842] arg[ 2.9866]
r1[1.9 j f[1,000] ]
x4 [ 0.5991 j -0.0198 ] abs[ 0.5994] arg[ -1.8954]
x6 [ 0.5965 j 0.0301 ] abs[ 0.5973] arg[ 2.8928]
B? キー入力待ち? _

```

5. 入力インピーダンスの確認

$R_i = 0.001$ オームとして、ノード6の出力電圧が1Vとなる入力信号の電圧aを求め
る。/acによって、入力信号の電圧がaの時に、ノード6の出力電圧が0.5V(半分)
になる R_i の値を求める。これが入力インピーダンスの値である。

$Z_i = 1250$ オームが得られる。

```

次に、入力インピーダンスを求めます
ri = 0.001 オームの時の利得が半分になる ri の値が入力インピーダンスです
r1 = 50 に戻します

r1=50
50 j 0

B? ri = 0.001 として、x6 = 1V となる、e1 を求めます

キー入力待ち?

B? ri=0.001
0.0010 j 0

B? /ac
ac 入力信号源名は ? e1
値は ? 1

値を変化させる素子名は ? e1
ac e1 最低値 ? 1
ac e1 最高値 ? 1
ac e1 ステップ ? 1
e1[1 j 0]
e1[1 j f[1,000] ]
x4 [ 0.9859 j 1.018468e-004] abs[ 0.9859] arg[ 0.0059]
x6 [ 0.9859 j 0.0032 ] abs[ 0.9859] arg[ 0.1883]
B? a=1/x6
1.0143 j -0.0033

```



```

/ac
ac 入力信号源名は ? e1
値は ? a

値を変化させる素子名は ? ri
ac ri 最低値 ? 1000
ac ri 最高値 ? 1500
ac ri ステップ ? 50
e1[1.014 j -0.003334 ]
ri[1000 ] f[1,000 ]
x4 [ 0.5585 j -0.0020 ] abs[ 0.5585] arg[ -0.2052]
x6 [ 0.5585 j -2.221239e-004] abs[ 0.5585] arg[ -0.0228]
ri[1050 ] f[1,000 ]
x4 [ 0.5465 j -0.0020 ] abs[ 0.5465] arg[ -0.2058]
x6 [ 0.5465 j -2.232647e-004] abs[ 0.5465] arg[ -0.0234]
ri[1100 ] f[1,000 ]
x4 [ 0.5349 j -0.0019 ] abs[ 0.5349] arg[ -0.2064]
x6 [ 0.5349 j -2.241114e-004] abs[ 0.5349] arg[ -0.0240]
ri[1150 ] f[1,000 ]
x4 [ 0.5238 j -0.0019 ] abs[ 0.5238] arg[ -0.2070]
x6 [ 0.5238 j -2.246974e-004] abs[ 0.5238] arg[ -0.0246]
ri[1200 ] f[1,000 ]
x4 [ 0.5132 j -0.0019 ] abs[ 0.5132] arg[ -0.2075]
x6 [ 0.5132 j -2.250520e-004] abs[ 0.5132] arg[ -0.0251]
ri[1250 ] f[1,000 ]
x4 [ 0.5030 j -0.0018 ] abs[ 0.5030] arg[ -0.2080]
x6 [ 0.5030 j -2.252010e-004] abs[ 0.5030] arg[ -0.0257]
ri[1300 ] f[1,000 ]
x4 [ 0.4932 j -0.0018 ] abs[ 0.4932] arg[ -0.2085]
x6 [ 0.4932 j -2.251675e-004] abs[ 0.4932] arg[ -0.0262]
ri[1350 ] f[1,000 ]
x4 [ 0.4838 j -0.0018 ] abs[ 0.4838] arg[ -0.2090]
x6 [ 0.4838 j -2.249719e-004] abs[ 0.4838] arg[ -0.0266]
ri[1400 ] f[1,000 ]
x4 [ 0.4747 j -0.0017 ] abs[ 0.4747] arg[ -0.2095]
x6 [ 0.4747 j -2.246323e-004] abs[ 0.4747] arg[ -0.0271]
ri[1450 ] f[1,000 ]
x4 [ 0.4660 j -0.0017 ] abs[ 0.4660] arg[ -0.2099]
x6 [ 0.4660 j -2.241649e-004] abs[ 0.4660] arg[ -0.0276]
ri[1500 ] f[1,000 ]
x4 [ 0.4575 j -0.0017 ] abs[ 0.4575] arg[ -0.2104]
x6 [ 0.4575 j -2.235841e-004] abs[ 0.4575] arg[ -0.0280]
B? キー入力待ち? _

```

```

B?
ri = 1250 オームで x6 は大体 0.5V となります
従って、入力インピーダンス Zi = 1250 オーム と求められました。

次に、抵抗器とトランジスタの消費電力を計算します

```

6. 各抵抗器とトランジスタの消費電力を計算し使用する抵抗器の定格を決定する。
- R 1 は $1 / 16 \text{ W}$ 、R 2 は $1 / 8 \text{ W}$ 、R e は 5 W となる。トランジスタのコレクタ損失は 1.2 W となり、周囲温度 70 度まで使用するとしたら、熱抵抗が 25 度/W 以下の放熱器が必要である。R L には交流信号のみが現れるので、実効値で消費電力を計算して、R L には 1 W の抵抗器を使用する。

```

B?
まず、ri=100 に戻して、直流動作点を再計算します
ri=100
100 j 0

B? f=0
0 j 0

```

```

B? 表示ノードを3、4及び6とします
/disg

独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
i[ 5] j[ 0] 部品名[e2] 値[15 j 0]
i[10] j[ 8] 部品名[ebb1] 値[0.6 j 0]
独立電圧源素子の個数 3
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
4 6

表示ノード番号 (0 なら全て、-1 ならこのまま) ? 3
ノード番号 (0 なら終了) ? 4
ノード番号 (0 なら終了) ? 6
ノード番号 (0 なら終了) ?
B? /point
f[0
x3 [ 8.5784 j 0 ] abs[ 8.5784] arg[ 0]
x4 [ 7.9216 j 0 ] abs[ 7.9216] arg[ 0]
x6 [ 0 j 0 ] abs[ 0] arg[ 0]
B? キー入力待ち?

B?
r2 の両端の電圧は、v2 = e2 - x3
r2 の消費電力は、p2 = v2 * v2 / r2

v2=(e2-x3)
6.4216 j 0

B? p2=v2*v2/r2
0.0137 j 0

B? 1/p2
72.7504 j 0

B?
従って、約10倍の余裕をもって r2 は 1/8W の抵抗器が使用できます

B?
同様に、r1 については

v1=x3
8.5784 j 0

B? p1=v1*v1/r1
0.0049 j 0

B? 1/p1
203.8347 j 0

B?
同様に、10倍の余裕をもって r1 は 1/16W の抵抗器が使用できます

```

```

B?
re については
ve=x4
7.9216 j 0

B? pe=ve*ve/re
1.2550 j 0

B? re については、4 倍の余裕をもって、5W の抵抗器を使用して下さい
キー入力待ち？

B?
trn の消費電力は、主にコレクタ損失ですから
Vce = (e2 - x4), Ie = x4/re , Pc = Vce * Ie

vce=e2-x4
7.0784 j 0

B? ie=x4/50
0.1584 j 0

B? pc=vce*ie
1.1214 j 0

B? コレクタ損失は、約1.2W となります

接合温度の限界を 120 度として、周囲温度 70 度まで使用するには
コレクタ損失を多めに 2W として計算すると、
(120 - 70)/2 = 25 度/W 以下の放熱器を使用する必要があります

(120-70)/2
25 j 0

```

```

B?

rl については、-4V から +4V までの振幅の正弦波がかかるとすると、
e1 の実効値は e1 = 4/sqrt(2)
この値の e1 における出力電圧を求める

f=1000
1,000 j 0

B? /ac
ac 入力信号源名は ? e1
値は ? 4/sqrt(2)

値を変化させる素子名は ? rl
ac rl 最低値 ? 50
ac rl 最高値 ? 50
ac rl ステップ ? 50
e1[2.828 j 0 ]
rl[50 j 0 ] f[1,000 ]
x3 [ 2.6212 j 1.567832e-004] abs[ 2.6212] arg[ 0.0034]
x4 [ 2.5842 j 9.643665e-005] abs[ 2.5842] arg[ 0.0021]
x6 [ 2.5842 j 0.0083 ] abs[ 2.5842] arg[ 0.1845]
B? キー入力待ち？

```

```

B?
r1 にかかる電圧は x6 だから、消費電力 Pl は  $Pl = x6 * x6 / r1$ 
pl=x6*x6/r1
0.1336 j 8.602222e-004
B? 1/pl
7.4872 j -0.0482
B? 従って、r1 については、約 7 倍の余裕をもって、1W の抵抗器が使用できます

以上で、エミッタホロワ増幅回路の設計を終わります

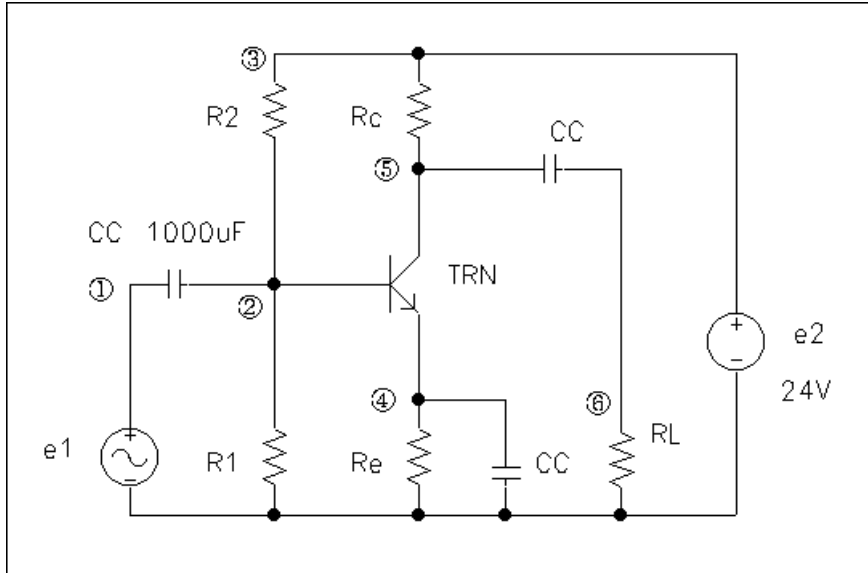
1 番のバッチファイル s5.sb を終了します。
? _

```

以上の様に、/para で直流動作点を満足する抵抗値を決定し、/ac によって交流ゲインを満足する抵抗値を決定することが出来る。また、負荷抵抗を開放状態（ $R_L = 1\text{ M}$ 等の非常に大きな値）にしたときの交流ゲインが半分になる負荷抵抗の値が出力インピーダンスであり、入力抵抗 R_i がほとんどゼロのときの交流ゲインが半分になる入力抵抗の値が入力インピーダンスである。この方法は色々な回路の解析に使用出来るので覚えておくと便利である。

(2) バッチファイル名 s 6. s b で扱う回路図

エミッタ接地増幅回路



設計の指針

1. ノード5の直流動作点を電源電圧 e_2 の半分に設定すると、出力のダイナミックレンジが最大になるので、12Vに設定する。

/point によって、現在のノード5の電圧を確認すると、18.3965Vになっており12Vまで下げる必要がある。ノード5の電圧を下げるには、TRNのコレクタ電流を増加すればよい。これはノード2の電圧を上げることを意味している。

このためにはR2を小さくするか、R1を大きくすればよい。ここでは、入力インピーダンスが下がりすぎないようにR1を大きくする方法を選ぶ。/para によってR1を変化させて確認すると、ノード5が12Vになるのは $R1 = 26\text{ K}$ の時である事が分かる。

2. 入力信号の周波数が1000Hzの時のゲインを調べてみる。現在のゲインは-1820であることがわかる。しかし、この値は現実的な値ではない。trn.cirのr1は0.1オームとなっているが、トランジスタの規格表では通常のトランジスタでは800から1200オームの値が載っているので、 $r1 = 800$ に変更する。このためには、ブロック素子が解凍されたときの素子名であるr1b1を800に設定する。

3. もう一度、直流動作点を確認する。——> あまり変化はない。

4. もう一度、ゲインを確認する。——> -100倍となり、妥当な値である。

エミッタ接地増幅回路のゲインはエミッタホロワに比べると非常に大きく、入力と出力の位相が逆転することがわかる。

5. サンプルs 5. s bの方法を使って、出力インピーダンスを調べる。

$Z_o = 3.8 \text{ K}$ オームが得られる。

サンプル s 5. s b のエミッタホロワ増幅回路の出力インピーダンスは 1.3 オーム であったことと比較すると、エミッタ接地増幅回路の出力インピーダンスはかなり高いことがわかる。このことは、負荷抵抗が変化すると出力電圧もかなり変化することを意味している。負荷の状態が変動する場合にも、安定した大きなゲインを希望するなら、エミッタ接地増幅回路の後ろにエミッタホロワ増幅回路を接続することが良い方法である。

また、エミッタ接地増幅回路では位相が反転するので、これを 2 段接続すると、もとに戻ることも覚えておくと良い。

```
B? /ipart
```

```
left[ 0] right ? 1  
素子名は ? e1  
値は ? 1
```

```
left[ 0] right ? 2  
素子名は ? r1  
値は ? 10k
```

```
left[ 0] right ? 4  
素子名は ? re  
値は ? 2.2k
```

```
left[ 0] right ? 4  
素子名は ? ce  
値は ? 1000u
```

```
left[ 0] right ? 6  
素子名は ? r1  
値は ? 1k
```

```
left[ 0] right ? 3  
素子名は ? e2  
値は ? 24
```

```
left[ 0] right ? /
```

```
left[ 1] right ? 2  
素子名は ? cc  
値は ? 1000u
```

```
left[ 1] right ? /
```

```
left[ 2] right ? 3  
素子名は ? r2  
値は ? 50k
```

```
left[ 2] right ? 5
```

```
素子名は ? b1  
ファイル名と接続ノードを入力して下さい(icl:0,1,2 の様に)  
trn:2,4,5
```

```
left[ 2] right ? /
```

```
left[ 3] right ? 5  
素子名は ? rc  
値は ? 3.8k
```

```
left[ 3] right ? /
```

```
left[ 4] right ? /
```

```
left[ 5] right ? 6  
素子名は ? cc  
素子名 cc は既に使用されています。  
同じ素子名を使用しますか? (y/n) y
```

```
left[ 5] right ? *
```

独立電圧源素子の接続ノード

```
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
```

```
i[ 3] j[ 0] 部品名[e2] 値[24 j 0]
```

独立電圧源素子の個数 2

独立電流源素子の個数 0

シミュレーションにおける 表示ノード設定状況

表示ノード番号 (0 なら全て、-1 ならこのまま) ? キー入力待ち?

```

/dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r1 ] value[ 1e+004 j 0 ]
left[ 0] right[ 3] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[ 4] parts[ ce ] value[ 0.001 j 0 ]
left[ 0] right[ 4] parts[ re ] value[ 2200 j 0 ]
left[ 0] right[ 6] parts[ r1 ] value[ 1000 j 0 ]
left[ 1] right[ 2] parts[ cc ] value[ 0.001 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 5e+004 j 0 ]
left[ 2] right[ 10] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 3] right[ 5] parts[ rc ] value[ 3800 j 0 ]
left[ 4] right[ 7] parts[ rbb1 ] value[ 0.1 j 0 ]
left[ 5] right[ 6] parts[ cc ] value[ 0.001 j 0 ]
left[ 5] right[ 9] parts[ rbb1 ] value[ 0.1 j 0 ]
left[ 7] right[ 8] parts[ reb1 ] value[ 26 j 0 ]
left[ 7] right[ 9] parts[ kb1 ] value[ 100 j 0 ]
@ master[reb1 ] left[ 7] right[ 8]
left[ 8] right[ 10] parts[ ebb1 ] value[ 0.6 j 0 ]
最大の素子番号 = 16
最大のノード番号 = 10

```

```

B? 動作点確認
/point
f[0 ]
x5 [ 18.3965 j 0 ] abs[ 18.3965] arg[ 0]
B? ノード5の動作点を  $x5 = 24/2 = 12V$  に設定する
現在の動作点は  $x5 = 18.3965 > 12$  なので、
R1 を増加させてバイアス電流を増加させれば、 $x5$  を下げることができる
/para を使って調べて見る。

```

```

B? /para
値を変化させる素子名は ? r1
para r1 最低値 ? 10k
para r1 最高値 ? 50k
para r1 ステップ ? 5k
r1[1e+004 ] f[0 ]
x5 [ 18.3965 j 0 ] abs[ 18.3965] arg[ 0]
r1[1.5e+004 ] f[0 ]
x5 [ 15.9725 j 0 ] abs[ 15.9725] arg[ 0]
r1[2e+004 ] f[0 ]
x5 [ 13.9472 j 0 ] abs[ 13.9472] arg[ 0]
r1[2.5e+004 ] f[0 ]
x5 [ 12.2295 j 0 ] abs[ 12.2295] arg[ 0]
r1[3e+004 ] f[0 ]
x5 [ 10.7544 j 0 ] abs[ 10.7544] arg[ 0]
r1[3.5e+004 ] f[0 ]
x5 [ 9.4739 j 0 ] abs[ 9.4739] arg[ 0]
r1[4e+004 ] f[0 ]
x5 [ 8.3518 j 0 ] abs[ 8.3518] arg[ 0]
r1[4.5e+004 ] f[0 ]
x5 [ 7.3605 j 0 ] abs[ 7.3605] arg[ 0]
r1[5e+004 ] f[0 ]
x5 [ 6.4783 j 0 ] abs[ 6.4783] arg[ 0]

```

```

B? R1 が 25K から 30K の間で x5 = 12 となるので、詳しく調べる

/para
値を変化させる素子名は ? r1
para r1 最低値 ? 25k
para r1 最高値 ? 30k
para r1 ステップ ? 1k
r1[2.5e+004 ] f[0 ]
x5 [ 12.2295 j 0 ] abs[ 12.2295] arg[ 0]
r1[2.6e+004 ] f[0 ]
x5 [ 11.9168 j 0 ] abs[ 11.9168] arg[ 0]
r1[2.7e+004 ] f[0 ]
x5 [ 11.6134 j 0 ] abs[ 11.6134] arg[ 0]
r1[2.8e+004 ] f[0 ]
x5 [ 11.3187 j 0 ] abs[ 11.3187] arg[ 0]
r1[2.9e+004 ] f[0 ]
x5 [ 11.0325 j 0 ] abs[ 11.0325] arg[ 0]
r1[3e+004 ] f[0 ]
x5 [ 10.7544 j 0 ] abs[ 10.7544] arg[ 0]
B? /point
f[0 ]
x5 [ 11.9168 j 0 ] abs[ 11.9168] arg[ 0]
B? f = 1000Hz における、ゲインを調べて見る
表示ノードを 5 から 6 に変更する

キー入力待ち？

B? f=1000
1,000 j 0

B? /disp
独立電圧源素子の接続ノード
i[ 1] j[ 0] 部品名[e1] 値[1 j 0]
i[ 3] j[ 0] 部品名[e2] 値[24 j 0]
i[10] j[ 8] 部品名[ebb1] 値[0.6 j 0]
独立電圧源素子の個数 3
独立電流源素子の個数 0
シミュレーションにおける 表示ノード設定状況
5

表示ノード番号 (0 なら全て、-1 ならこのまま) ? 6
ノード番号 (0 なら終了) ?
B? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1000
range 周波数 最高値 ? 1000
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x6 [-1.820.6864 j -816.5491 ] abs[1.995.4076] arg[ -155.8445]
B? 現在のゲインは -1820 となっていますが、これは現実と比べて大き過ぎます。
回路ブロック trn.cir の r1 = 0.1 となっていますが、
トランジスタの規格表には 800 から 1200 の値が hie として記載されています。
ここでは、r1=800 とします
trn.cir は b1 という名称のブロックとして使用されていますから
r1=800 とするには、 r1b1=800 と入力する必要があります
r1b1=800
800 j 0

```



```

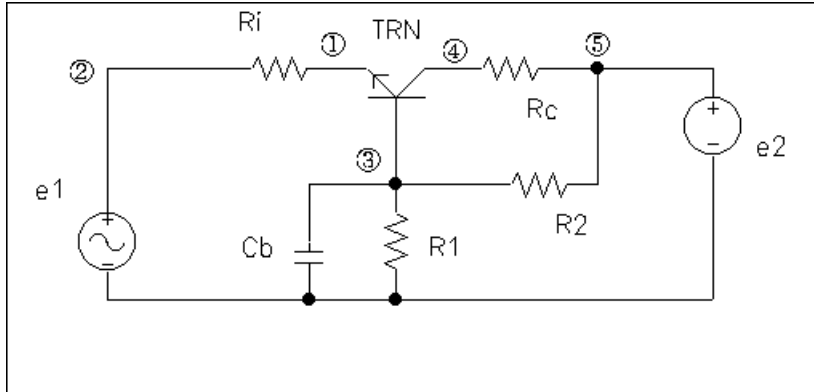
B? /ac
ac 入力信号源名は ? el
値は ? a

値を変化させる素子名は ? rl
ac rl 最低値 ? 3k
ac rl 最高値 ? 4k
ac rl ステップ ? 100
el[0.2209 j 0 ]
rl[3000 ] f[1,000 ]
x5 [ -44.2770 j -0.8588 ] abs[ 44.2853] arg[ -178.8888]
x6 [ -44.2769 j -0.8611 ] abs[ 44.2853] arg[ -178.8858]
rl[3100 ] f[1,000 ]
x5 [ -45.0898 j -0.8746 ] abs[ 45.0983] arg[ -178.8888]
x6 [ -45.0897 j -0.8769 ] abs[ 45.0983] arg[ -178.8858]
rl[3200 ] f[1,000 ]
x5 [ -45.8794 j -0.8900 ] abs[ 45.8880] arg[ -178.8887]
x6 [ -45.8793 j -0.8923 ] abs[ 45.8880] arg[ -178.8858]
rl[3300 ] f[1,000 ]
x5 [ -46.6467 j -0.9049 ] abs[ 46.6555] arg[ -178.8886]
x6 [ -46.6467 j -0.9072 ] abs[ 46.6555] arg[ -178.8859]
rl[3400 ] f[1,000 ]
x5 [ -47.3927 j -0.9195 ] abs[ 47.4017] arg[ -178.8886]
x6 [ -47.3927 j -0.9217 ] abs[ 47.4017] arg[ -178.8859]
rl[3500 ] f[1,000 ]
x5 [ -48.1183 j -0.9336 ] abs[ 48.1274] arg[ -178.8885]
x6 [ -48.1183 j -0.9358 ] abs[ 48.1274] arg[ -178.8859]
rl[3600 ] f[1,000 ]
x5 [ -48.8243 j -0.9473 ] abs[ 48.8335] arg[ -178.8884]
x6 [ -48.8243 j -0.9495 ] abs[ 48.8335] arg[ -178.8859]
rl[3700 ] f[1,000 ]
x5 [ -49.5115 j -0.9607 ] abs[ 49.5208] arg[ -178.8884]
x6 [ -49.5114 j -0.9628 ] abs[ 49.5208] arg[ -178.8859]
rl[3800 ] f[1,000 ]
x5 [ -50.1806 j -0.9737 ] abs[ 50.1900] arg[ -178.8883]
x6 [ -50.1805 j -0.9758 ] abs[ 50.1900] arg[ -178.8859]
rl[3900 ] f[1,000 ]
x5 [ -50.8322 j -0.9864 ] abs[ 50.8418] arg[ -178.8883]
x6 [ -50.8322 j -0.9885 ] abs[ 50.8418] arg[ -178.8860]
rl[4000 ] f[1,000 ]
x5 [ -51.4672 j -0.9988 ] abs[ 51.4769] arg[ -178.8883]
x6 [ -51.4672 j -1.0008 ] abs[ 51.4769] arg[ -178.8860]
B? RL = 3.8K で出力が -50 となります。出力インピーダンスは  $Z_o = 3.8K$  です
サンプル s5 の出力インピーダンスは  $Z_o = 1.3$  オームでしたから、
エミッタ接地増幅回路の出力インピーダンスはかなり高いことが分かります。
これは、負荷抵抗の値が出力電圧の値に大きな影響を与えるという事です。
以上で、エミッタ接地増幅回路の設計を終了します。

```

(3) バッチファイル名 s7. sb で扱う回路図

ベース接地増幅回路



設計の指針

1. ノード4の直流動作点を電源電圧 e_2 の半分に設定する。

/para を使用して、 R_2 を 10 K から 150 K まで変化させてノード4が 12 V になる R_2 の値を求める。

今回は、/mk により計算データをファイルに作成し、それを/grph によってグラフを表示して、 $x_4 = 12$ となる R_2 の値を調べる。 R_2 の目盛りは対数目盛りになるので注意する事。 ——> $R_2 = 106.35\text{ K}$ が得られる。

2. 入力信号の周波数が 1000 Hz の時のゲインを調べる。

サンプル s6. sb と同様に $r_{1b1} = 800$ としてゲインを調べる。ゲインは 50 であることが分かる。従って、入力と出力の位相は同じである。ゲインはエミッタ接地よりも小さいが、エミッタホロワよりも大きい事が分かる。

3. $f = 0$ として、直流動作点を再度確認する。 ——> あまり変化なし。

4. 出力インピーダンスを調べる。

/padd を使用して2個の部品を追加する。ノード4からコンデンサ C_C を通して負荷抵抗 $R_L = 1\text{ M}$ を接続する。これは無負荷時のゲインを確認するときの負荷抵抗の値である。この状態で、出力電圧が 100 (1でも構わない) となる e_1 を求める。

次に、/ac を使用して、入力信号の値を今求めた値を入力して、 R_L を 1 K から 100 K まで変化させて出力電圧が半分になる R_L の値を見つける。この時にもグラフを表示して確認すると分かりやすい。

$Z_o = 4950$ オームが得られる。この値はエミッタ接地の場合と同程度以上の値であり、出力電圧が負荷抵抗により影響されやすいことを示している。

5. 次に、入力インピーダンスを調べる。

まず、 $R_i = 0.001$ オームとしたときの出力電圧が 100 (1でもよい) となる信号入力電圧を求める。

/ac を使用して、この入力信号電圧の時に、 R_i を 0.1 オームから 1 K オームまで変化させて出力電圧が先程の半分になる R_i の値を見つける。ここでも、グラフ使用。

$Z_i = 8.28$ が得られる。

従って、ベース接地増幅回路では、入力インピーダンスが非常に小さいことが分かる。このことは、信号源インピーダンスの高い入力に接続するためには、まずエミッタホロワを信号に接続してからベース接地増幅回路に接続した方が良いことを意味している。エミッタ接地増幅回路と同様に負荷変動によるゲイン変動を防ぎたいければ出力側にもエミッタ接地増幅回路が必要である。この構成では入力と出力の位相が同じであることが特徴である。

```
B? /dpart
left[ 0] right[ 2] parts[ e1          ] value[      1 j 0 ]
left[ 0] right[ 3] parts[ cb          ] value[    0.001 j 0 ]
left[ 0] right[ 3] parts[ r1          ] value[    1e+004 j 0 ]
left[ 0] right[ 5] parts[ e2          ] value[      24 j 0 ]
left[ 1] right[ 2] parts[ ri          ] value[     100 j 0 ]
left[ 1] right[ 6] parts[ rbb1         ] value[      0.1 j 0 ]
left[ 3] right[ 5] parts[ r2          ] value[    3e+004 j 0 ]
left[ 3] right[ 9] parts[ rib1         ] value[      0.1 j 0 ]
left[ 4] right[ 5] parts[ rc          ] value[     5000 j 0 ]
left[ 4] right[ 8] parts[ rbb1         ] value[      0.1 j 0 ]
left[ 6] right[ 7] parts[ reb1         ] value[      26 j 0 ]
left[ 6] right[ 8] parts[ kb1          ] value[     100 j 0 ]
@ master[reb1      ] left[ 6] right[ 7]
left[ 7] right[ 9] parts[ ebb1         ] value[      0.6 j 0 ]
最大の素子番号 = 13
最大のノード番号 = 9
B? 今入力した、ベース接地増幅回路の回路リストです
```

```
B? /para
データファイル名は ? test
値を変化させる素子名は ? r2
para r2 最低値 ? 20k
para r2 最高値 ? 160k
para r2 ステップ ? 20k
r2[2e+004 ] f[0 ]
x1 [ 4.8470 j 0 ] abs[ 4.8470] arg[ 0]
x4 [ -166.4437 j 0 ] abs[ 166.4437] arg[ 180]
r2[4e+004 ] f[0 ]
x1 [ 2.7821 j 0 ] abs[ 2.7821] arg[ 0]
x4 [ -64.2213 j 0 ] abs[ 64.2213] arg[ 180]
r2[6e+004 ] f[0 ]
x1 [ 1.9872 j 0 ] abs[ 1.9872] arg[ 0]
x4 [ -24.8723 j 0 ] abs[ 24.8723] arg[ 180]
r2[8e+004 ] f[0 ]
x1 [ 1.5663 j 0 ] abs[ 1.5663] arg[ 0]
x4 [ -4.0332 j 0 ] abs[ 4.0332] arg[ 180]
r2[1e+005 ] f[0 ]
x1 [ 1.3056 j 0 ] abs[ 1.3056] arg[ 0]
x4 [ 8.8698 j 0 ] abs[ 8.8698] arg[ 0]
r2[1.2e+005 ] f[0 ]
x1 [ 1.1284 j 0 ] abs[ 1.1284] arg[ 0]
x4 [ 17.6450 j 0 ] abs[ 17.6450] arg[ 0]
r2[1.4e+005 ] f[0 ]
x1 [ 1 j 0 ] abs[ 1] arg[ 0]
x4 [ 24 j 0 ] abs[ 24] arg[ 0]
r2[1.6e+005 ] f[0 ]
x1 [ 0.9027 j 0 ] abs[ 0.9027] arg[ 0]
x4 [ 28.8147 j 0 ] abs[ 28.8147] arg[ 0]
B? R2 が 100K から 120K の間で x4 = 12V となっています
```

```

B?
画面の数字でも分かり易くするには、
/gpos
5
/manu

のようにモードを設定して、計算するポイントを極力減らして
/para, /ac, /range などのコマンドを実行する方法があります。

キー入力待ち？

B? /gpos
MANU ON または 最高/最低の比が10以上の時に、
シミュレーションで計算するポイント数を入力して下さい
最低 5 以上、最大 300 以下の数字を入力して下さい
現在の設定は 32 です
5
gpos = 5 に設定しました
B? /manu
B? /para
データファイル名は ? test
値を変化させる素子名は ? r2
para r2 最低値 ? 10k
para r2 最高値 ? 150k
para r2 ステップ ? 10k
r2[1e+004 ] f[0 ]
x1 [ 7.9397 j 0 ] abs[ 7.9397] arg[ 0]
x4 [ -319.5473 j 0 ] abs[ 319.5473] arg[ 180]
r2[1.968e+004] f[0 ]
x1 [ 4.9072 j 0 ] abs[ 4.9072] arg[ 0]
x4 [ -169.4250 j 0 ] abs[ 169.4250] arg[ 180]
r2[3.873e+004] f[0 ]
x1 [ 2.8571 j 0 ] abs[ 2.8571] arg[ 0]
x4 [ -67.9349 j 0 ] abs[ 67.9349] arg[ 180]
r2[7.622e+004] f[0 ]
x1 [ 1.6299 j 0 ] abs[ 1.6299] arg[ 0]
x4 [ -7.1856 j 0 ] abs[ 7.1856] arg[ 180]
r2[1.5e+005 ] f[0 ]
x1 [ 0.9482 j 0 ] abs[ 0.9482] arg[ 0]
x4 [ 26.5626 j 0 ] abs[ 26.5626] arg[ 0]
B? /para
データファイル名は ? test
値を変化させる素子名は ? r2
para r2 最低値 ? 100k
para r2 最高値 ? 110k
para r2 ステップ ? 1k
r2[1e+005 ] f[0 ]
x1 [ 1.3056 j 0 ] abs[ 1.3056] arg[ 0]
x4 [ 8.8698 j 0 ] abs[ 8.8698] arg[ 0]
r2[1.01e+005 ] f[0 ]
x1 [ 1.2952 j 0 ] abs[ 1.2952] arg[ 0]
x4 [ 9.3872 j 0 ] abs[ 9.3872] arg[ 0]
r2[1.02e+005 ] f[0 ]
x1 [ 1.2849 j 0 ] abs[ 1.2849] arg[ 0]
x4 [ 9.8949 j 0 ] abs[ 9.8949] arg[ 0]
r2[1.03e+005 ] f[0 ]
x1 [ 1.2749 j 0 ] abs[ 1.2749] arg[ 0]
x4 [ 10.3932 j 0 ] abs[ 10.3932] arg[ 0]
r2[1.04e+005 ] f[0 ]
x1 [ 1.2650 j 0 ] abs[ 1.2650] arg[ 0]
x4 [ 10.8824 j 0 ] abs[ 10.8824] arg[ 0]
r2[1.05e+005 ] f[0 ]
x1 [ 1.2553 j 0 ] abs[ 1.2553] arg[ 0]
x4 [ 11.3627 j 0 ] abs[ 11.3627] arg[ 0]
r2[1.06e+005 ] f[0 ]
x1 [ 1.2457 j 0 ] abs[ 1.2457] arg[ 0]
x4 [ 11.8343 j 0 ] abs[ 11.8343] arg[ 0]
r2[1.07e+005 ] f[0 ]
x1 [ 1.2364 j 0 ] abs[ 1.2364] arg[ 0]
x4 [ 12.2975 j 0 ] abs[ 12.2975] arg[ 0]
r2[1.08e+005 ] f[0 ]
x1 [ 1.2272 j 0 ] abs[ 1.2272] arg[ 0]
x4 [ 12.7526 j 0 ] abs[ 12.7526] arg[ 0]
r2[1.09e+005 ] f[0 ]
x1 [ 1.2182 j 0 ] abs[ 1.2182] arg[ 0]
x4 [ 13.1996 j 0 ] abs[ 13.1996] arg[ 0]
r2[1.1e+005 ] f[0 ]
x1 [ 1.2093 j 0 ] abs[ 1.2093] arg[ 0]
x4 [ 13.6388 j 0 ] abs[ 13.6388] arg[ 0]
B? R2 が 106K から 107K の間で x4 = 12V となっています

```

```

r2[1.063e+005] f[0
x1 [ 1.2429 j 0 ] abs[ 1.2429] arg[ 0]
x4 [ 11.9742 j 0 ] abs[ 11.9742] arg[ 0]
r2[1.064e+005] f[0
x1 [ 1.2425 j 0 ] abs[ 1.2425] arg[ 0]
x4 [ 11.9974 j 0 ] abs[ 11.9974] arg[ 0]
r2[1.064e+005] f[0
x1 [ 1.2420 j 0 ] abs[ 1.2420] arg[ 0]
x4 [ 12.0206 j 0 ] abs[ 12.0206] arg[ 0]
B? 大体、R2 = 106.35k で x4 = 12V となっていますから、この値に決定します。
r2=106.35k
106,350 j 0

B? 周波数 f = 1000Hz における、ゲインを確認します
B? ゲインを確認します
/range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1000
range 周波数 最高値 ? 1000
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x1 [ 0.0765 j -1.344054e-005] abs[ 0.0765] arg[ -0.0101]
x4 [ 45.7201 j 6.653730e-004] abs[ 45.7201] arg[8.338353e-004]
B? ゲインは 1 割程低下しています

次は、f=0 としてから、動作点を確認します
キー入力待ち？

B? f=0
0 j 0

B? /point
f[0 ]
x1 [ 1.2328 j 0 ] abs[ 1.2328] arg[ 0]
x4 [ 12.4756 j 0 ] abs[ 12.4756] arg[ 0]
B? 動作点は 3% 程高くなっていますが、あまり大きな変動ではありません。
B? 次は、出カインピーダンスを求めます
現在は無負荷の状態ですから、交流的に負荷抵抗を追加します。
ノード 0 と 10 の間に RL=1M と ノード 4 と 10 の間に cc=1000uF を追加します
キー入力待ち？

B? /padd
left[ 0] right ? 10
素子名は ? rl
値は ? 1m

left[ 0] right ? /
left[ 1] right ? /
left[ 2] right ? /
left[ 3] right ? /
left[ 4] right ? 10
素子名は ? cc
値は ? 1000u

left[ 4] right ? *

```

```

B? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1000
range 周波数 最高値 ? 1000
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1,000 ]
x4 [ 45.4927 j 6.620266e-004] abs[ 45.4927] arg[8.337898e-004]
x10 [ 45.4927 j 6.692670e-004] abs[ 45.4927] arg[8.429087e-004]
B? a=e1/x10*100 の値の e1 で、x10 の電圧が 100 となるはずです。

```

```

rl[4281 ] f[1,000 ]
x4 [ 46.3591 j -9.357424e-004] abs[ 46.3591] arg[ -0.0012]
x10 [ 46.3591 j 7.876171e-004] abs[ 46.3591] arg[9.734263e-004]
rl[5456 ] f[1,000 ]
x4 [ 52.4396 j -7.398792e-004] abs[ 52.4396] arg[-8.083958e-004]
x10 [ 52.4396 j 7.899306e-004] abs[ 52.4396] arg[8.630822e-004]

```

```

rl[4900 ] f[1,000 ]
x4 [ 49.7424 j -8.238691e-004] abs[ 49.7424] arg[-9.489731e-004]
x10 [ 49.7424 j 7.917947e-004] abs[ 49.7424] arg[9.120282e-004]
rl[5000 ] f[1,000 ]
x4 [ 50.2500 j -8.077113e-004] abs[ 50.2500] arg[-9.209641e-004]
x10 [ 50.2500 j 7.917959e-004] abs[ 50.2500] arg[9.028172e-004]

```

B? RL = 4950 で、x10 = 50 となります。
出力インピーダンスは Zo = 4950 オームです

```

ri[8.2 ] f[1,000 ]
x4 [ 50.2434 j -0.0048 ] abs[ 50.2434] arg[ -0.0054]
x10 [ 50.2434 j -0.0048 ] abs[ 50.2434] arg[ -0.0054]
ri[8.3 ] f[1,000 ]
x4 [ 49.9403 j -0.0048 ] abs[ 49.9403] arg[ -0.0055]
x10 [ 49.9403 j -0.0048 ] abs[ 49.9403] arg[ -0.0055]

```

B? Ri = 8.28 で x10 = 50 となっています

以上のことからベース接地増幅回路の特徴をまとめると、

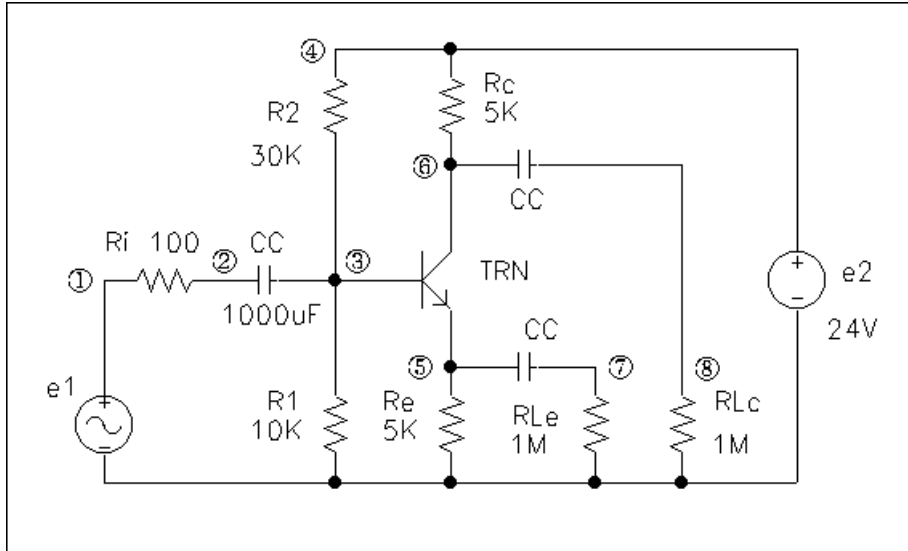
- 1 入力インピーダンスが非常に低い
- 2 出力インピーダンスが非常に高い
- 3 ゲインは中程度
- 4 入力と出力の位相が同じ（エミッタ接地だけが位相が反転する）

ということが分かりました。

以上でベース接地増幅回路の設計を終了します

(4) バッチファイル名 s8. sb で扱う回路図

反転増幅回路 トランジスタのコレクタとエミッタの両方から出力を得る回路



設計の指針

1. ノード6の直流動作点を電源電圧 e_2 の半分に設定する。

現在の電圧は、18.7321Vなので、`/para` を使用して R_2 を1Kから30Kまで変化させノード6が12Vになる R_2 の値を調べる。ここでも、グラフ使用。

$x_6 = 12$ となる時に同時に x_5 も12となり、トランジスタの $V_{ce} = 0$ になってしまうので、 $x_6 = 14.4$ Vになる R_2 をさがすことにする。

$R_2 = 13$ Kで $x_6 = 14.4$ V, $x_5 = 9.8$ Vとなる。

従って、 $V_{ce} = x_6 - x_5 = 4.6 > V_{ce(sat)}$ となり動作が保証される。

2. ノード7及びノード8のそれぞれのゲインを確認する。

入力信号の周波数を1000Hzとして、ゲインを確認する。

ノード7のゲインは0.9824、ノード8のゲインは-0.9726であることが分かる。

したがって、コレクタ側は入力と位相が反転しエミッタ側は入力と同位相の出力が得られ、ゲインはどちらも約1である。しかし、コレクタ側の方が少しだけゲインが小さい。

3. コレクタ側とエミッタ側のゲインの絶対値を等しくする。

`/ac` を使用して、 R_c を変化させて、 x_7 と x_8 の出力が同じになる R_c の値を求める。

$R_c = 5.075$ Kで等しくなることが分かる。

4. 出力インピーダンスを調べる。コレクタ側とエミッタ側のそれぞれの出力インピーダンスを調べる。

コレクタ側の出力インピーダンス $Z_{oc} = 5\text{ K}$

エミッタ側の出力インピーダンス $Z_{oe} = 1.5\text{ オーム}$

負荷抵抗が 1 M の時の出力電圧を 1 として、負荷抵抗を 100 オーム から 1 M まで変化させたときの出力電圧を V_{ac} を使用して確認すると下表が得られる。

負荷抵抗	コレクタ側出力	エミッタ側出力
1	0.0002	0.7311
10	0.002	0.9653
100	0.0194	0.9964
1 K	0.16	0.9996
10 K	0.667	1.0
100 K	0.9565	1.0
1 M	1.0	1.0

従って、エミッタ側では負荷抵抗が 100 オーム 以上の範囲で変動しても出力電圧には影響が現れないが、コレクタ側にはエミッタホロワを接続するなど負荷変動の影響がコレクタ側に伝わらないための工夫が必要であることが分かる。

反転増幅器では直流動作点を適正に選び $V_{ce} > V_{ce(sat)}$ を満足するように設計しなければならない。次に、コレクタ側の出力はエミッタホロワに接続して負荷変動がゲインに影響しないための工夫が必要である。さらに、コレクタ抵抗 R_c を調整してコレクタ側とエミッタ側のゲインが等しくなるように設計する必要がある。

5. エミッタ側の負荷抵抗の変動があっても、エミッタ側のゲインはあまり影響を受けない事が分かった。この時に、コレクタ側のゲインはどうなっているのかを調べる。

エミッタ側の負荷抵抗を 100 オーム から 10 K まで変動させると、エミッタ側のゲインは変化しないが、コレクタ側のゲインは約 -33 から 1 まで大きく変化している。従って、エミッタ側の負荷抵抗も変動しないようにエミッタホロワのようなバッファを接続しておいた方が良いと思われる。

6. 負荷抵抗が固定値であることが保証されている場合には、コレクタ側もエミッタ側もエミッタホロワなどを接続しなくてもよい。


```

B? ノード6の動作点を 12V に設定します。
一度現在の動作点を確認します。
f=0
/point
f=0
0 j 0

B? /point
f[0 ]
x5 [ 5.3206 j 0 ] abs[ 5.3206] arg[ 0]
x6 [ 18.7321 j 0 ] abs[ 18.7321] arg[ 0]
B? 目標の電圧より x6 が高いので、R2 を減らしてバイアス電流を増加させます。
/para
test
r2
1k
30k
1k

```

```

r2[9655 ] f[0 ]
x5 [ 11.4980 j 0 ] abs[ 11.4980] arg[ 0]
x6 [ 12.6158 j 0 ] abs[ 12.6158] arg[ 0]
r2[1.409e+004] f[0 ]
x5 [ 9.2553 j 0 ] abs[ 9.2553] arg[ 0]
x6 [ 14.8363 j 0 ] abs[ 14.8363] arg[ 0]
r2[2.056e+004] f[0 ]
x5 [ 7.1579 j 0 ] abs[ 7.1579] arg[ 0]
x6 [ 16.9130 j 0 ] abs[ 16.9130] arg[ 0]
x6=12V で x5 も 12V となり Vce=0となるので、x6=14.4V に設定します
= 10K から 20K の間で x6 = 14.4V となります。

```

```

r2[1.3e+004 ] f[0 ]
x5 [ 9.7252 j 0 ] abs[ 9.7252] arg[ 0]
x6 [ 14.3711 j 0 ] abs[ 14.3711] arg[ 0]
R2 = 13K で x6=14.4V になります、この時 x5=9.8V となっています。
って、Vce = x6 - x5 = 4.6V となります。

```

```

B? /ac
データファイル名は ? test
ac 入力信号源名は ? e1
値は ? 1

値を変化させる素子名は ? rlc
ac rlc 最低値 ? 1m
ac rlc 最高値 ? 1m
ac rlc ステップ ? 1
e1[1 j 0 ]
rlc[1e+006 ] f[1,000 ]
x7 [ 0.9824 j 2.763706e-005] abs[ 0.9824] arg[ 0.0016]
x8 [ -0.9726 j -2.736343e-005] abs[ 0.9726] arg[ -179.9984]
B? エミッタ、コレクタそれぞれの負荷抵抗が 1M の時
ノード8の利得が、ノード7の利得より 1% 低い

RLc=RLe=10K に変更して、交流利得を確認してみます。

```

```
rc[5070] f[1,000]
x7 [ 0.9822 j 4.325543e-005] abs[ 0.9822] arg[ 0.0025]
x8 [-0.9815 j -4.317665e-005] abs[ 0.9815] arg[-179.9975]
rc[5080] f[1,000]
x7 [ 0.9822 j 4.325543e-005] abs[ 0.9822] arg[ 0.0025]
x8 [-0.9828 j -4.322624e-005] abs[ 0.9828] arg[-179.9975]
```

Rc=5075 で、x7 と x8 の利得が等しくなっている

```
rlc[4642] f[1,000]
x7 [-0.9951 j -7.349554e-012] abs[ 0.9951] arg[-180]
x8 [ 0.4801 j 7.788229e-006] abs[ 0.4801] arg[9.294158e-004]
rlc[1e+004] f[1,000]
x7 [-0.9951 j -7.349554e-012] abs[ 0.9951] arg[-180]
x8 [ 0.6667 j 6.933314e-006] abs[ 0.6667] arg[5.958300e-004]
```

コレクタ側の出カインピーダンスは $Z_{oc}=5K$ です。

コレクタ側の出カインピーダンスは $Z_{oc}=5K$ と求められました。
 負荷抵抗 R_L が $1K$ から $1M$ まで変化すると、 x_8 は 0.16 から 1 まで変化します。

B? $x_8=1$ を基準に考えると、 16% まで利得が下がる訳です。

```
rle[1] f[1,000]
x7 [ 0.4270 j 0.0294] abs[ 0.4280] arg[ 3.9432]
x8 [-2,135.0760 j -147.1056] abs[2,140.1377] arg[-176.0586]
rle[2.783] f[1,000]
x7 [ 0.6754 j 0.0264] abs[ 0.6759] arg[ 2.2369]
x8 [-1,214.1842 j -47.3879] abs[1,215.1085] arg[-177.7650]
```

$R_{Le}=1.5$ 位で $x_7=0.5$ となっています。

B? エミッタ側の出カインピーダンスは $Z_{oe}=1.5$ です。
 コレクタ側の出カインピーダンスは $Z_{oc}=5K$ でしたから、
 コレクタ側に比較してかなり低い値です。これは、負荷抵抗の大きさで
 出力電圧があまり変化しないことを意味しています。
 逆に、コレクタ側では負荷抵抗の大きさによって出力電圧が大きく変化します。

R_{Le} が 1 から $10K$ まで変化しても x_7 は 0.47 から 1 までしか変化していません。

```
ri[3594] f[1,000]
x7 [ 0.6073 j -6.848726e-006] abs[ 0.6073] arg[-6.460924e-004]
x8 [-0.6073 j 6.880442e-006] abs[ 0.6073] arg[ 179.9994]
ri[1e+004] f[1,000]
x7 [ 0.3573 j -6.594672e-006] abs[ 0.3573] arg[-0.0011]
x8 [-0.3573 j 6.613203e-006] abs[ 0.3573] arg[ 179.9989]
```

B? $R_i=5.5K$ で $x_7=0.5$ となります。入力インピーダンスは $Z_i=5.5K$ です。

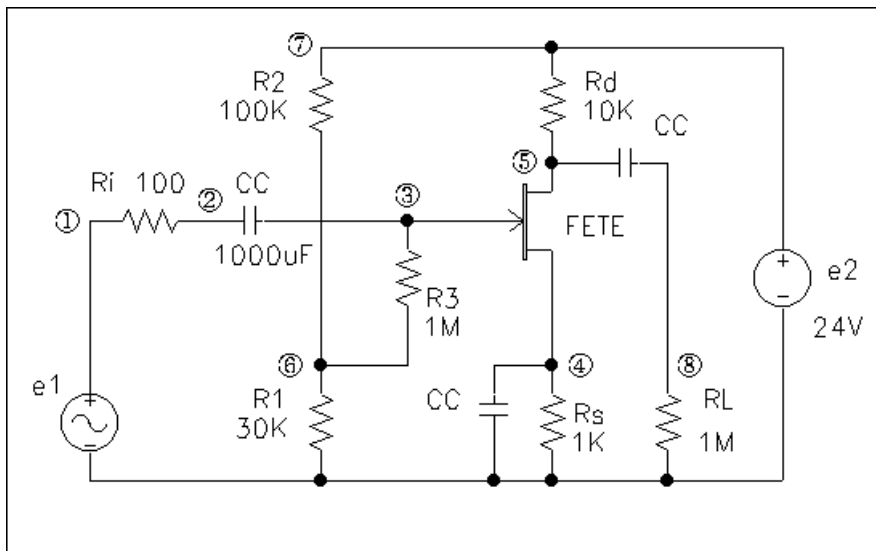
$R_i=5.5K$ で $x_7=0.5$ となります。入力インピーダンスは $Z_i=5.5K$ です。

転増幅回路の特性は、
 カインピーダンスはエミッタホロワ回路の特性、
 カインピーダンスは、エミッタ側はエミッタホロワ回路の特性
 コレクタ側はエミッタ接地回路の特性となっています。

上で、反転増幅回路の設計を終了します。

(5) バッチファイル名 s 9. s b で扱う回路図

ソース接地増幅回路 (N型エンハンスメントタイプ F E T を使用)



設計の指針

1. N型エンハンスメントタイプのF E Tでは、ゲート電圧をソース電圧よりも高くしなければドレイン電流が流れない。
2. この回路形式はトランジスタのエミッタ接地増幅回路に対応するものである。
3. ノード5の直流動作点を電源電圧 e_2 の半分に設定する。

/point を使用して、現在のノード5の電圧を確認すると、負の値になっているので、/para を使用して、 R_1 を1 Kから30 Kまで変化させて適当な値を見つける。

$R_1 = 6.5 K$ でノード5が12 Vになる。

4. 入力信号の周波数が1000 Hzの時のゲインを確認する。ゲインは-19.038と分かる。ゲインはエミッタ接地増幅回路よりも小さい。出力の位相が入力と逆になっていることもエミッタ接地増幅回路と同じである。
5. 出力インピーダンスを求める。 $Z_o = 9 K$ が得られる。F E Tのソース接地増幅回路では、トランジスタのエミッタ接地増幅回路よりも出力インピーダンスが高いことがわかる。出力にエミッタホロワのようなバッファ回路が必要である。

これには、F E Tのソースホロワ増幅回路に対応する。

6. 入力インピーダンスを求める。 $Z_i = 1 M$ が得られる。

この値は R_3 の値であり、 $R_3 = 10 M$ なら入力インピーダンスも10 Mとなる。

入力インピーダンスはエミッタホロワよりも高いので、信号源インピーダンスの高い応用においても使用する事が可能である。

B? ノード5の動作点を電源電圧 e2 の 1/2 の 12V に設定します。
現在の動作点を確認します。

```
/point  
f[0  
x5 [ -19.8192 j 0 ] abs[ 19.8192] arg[ 180]
```

B? ノード5が負の電圧になっています。
これは、ドレイン電流が大き過ぎる事を表わしています。
ドレイン電流が大きいということは、ノード6の電圧が高いということです。
/para で R1 を 1K から 30K まで変化させて調べましょう。

```
r1[6135 ] f[0  
x5 [ 12.6724 j 0 ] abs[ 12.6724] arg[ 0]  
r1[7696 ] f[0  
x5 [ 10.1064 j 0 ] abs[ 10.1064] arg[ 0]
```

B? R1=6.5K で動作点が 12V になります。

```
r1=6.5k  
6,500 j 0  
  
B? /point  
f[0  
x5 [ 12.0654 j 0 ] abs[ 12.0654] arg[ 0]
```

B? /range

データファイル名は ? test
ac 入力信号源名は ? e1
値は ? 1

```
range 周波数 最低値 ? 1k  
range 周波数 最高値 ? 1k  
range 周波数 ステップ ? 1  
e1[1 j 0 ]  
f[1,000 ]  
x5 [ -19.0384 j -0.0114 ] abs[ 19.0384] arg[ -179.9656]
```

B? ゲインは -19.0384 と求められました。
- 記号は出力の位相が反転することを表わしています。

```
r1[8577 ] f[1,000 ]  
x5 [ -0.4899 j -2.941220e-004] abs[ 0.4899] arg[ -179.9656]  
r1[1.166e+004] f[1,000 ]  
x5 [ -0.5670 j -3.399397e-004] abs[ 0.5670] arg[ -179.9656]  
B? RL=9K で出力が 0.5 となるので、出力インピーダンスは 9K です。
```

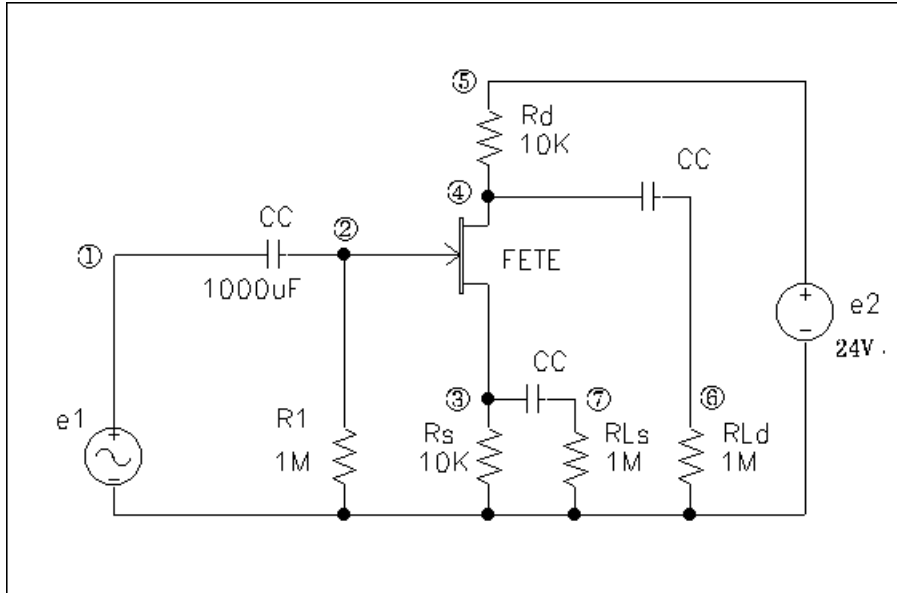
```
ri[1e+006 ] f[1,000 ]  
x5 [ -0.5015 j -2.911442e-004] abs[ 0.5015] arg[ -179.9667]  
ri[2.154e+006] f[1,000 ]  
x5 [ -0.3183 j -1.847832e-004] abs[ 0.3183] arg[ -179.9667]
```

B? 入力インピーダンスは 1M です

FET のソース接地増幅回路は、トランジスタのエミッタ接地増幅回路に似ていますが
ゲインが低めです。
出力インピーダンスは同等ですが、入力インピーダンスはかなり高い事が分かります
以上で、ソース接地増幅回路の解析を終了します。

(6) バッチファイル名 s10. sb で扱う回路図

反転増幅回路 (N型デプリーションタイプFETを使用)



設計の指針

1. N型デプリーションタイプのFETでは、 $V_{gs} = 0V$ で最もドレイン電流が流れる。ソース電圧がゲート電圧よりも高くなるに従って、ドレイン電流が減少する。
2. この回路形式はトランジスタの反転増幅回路に対応するものである。
3. ノード4の電圧を電源電圧の半分に設定する。

/point を使用して、現在の電圧を調べると、

ノード3の電圧は1.64Vであり、ノード4の電圧は22.358Vが得られる。

このままでは、出力電圧の振幅があまり得られないので調整をする。

FETのパラメータ e1b1 を現在の2Vから6Vに変更して確認すると、

ノード3の電圧は5.8834V、ノード4の電圧は18.117Vとなる。これなら、ソース側は5.8834Vを中心にプラス・マイナス5V程度、ドレイン側は18.117Vを中心にプラス・マイナス5V程度の出力が得られるようになる。

4. ノード3とノード4のゲインを求める。

ノード3のゲインは0.09706、ノード4のゲインは-0.9706となる。

トランジスタとは異なり、ゲート電流が流れないためノード3とノード4のゲインは絶対値が等しい。また、ソース側は非反転、ドレイン側は反転出力になっている。

5. 出力インピーダンスを求める。

ドレイン側の出力インピーダンス $Z_{od} = 9.8K$ が得られる。

ソース側の出力インピーダンス $Z_{os} = 270$ オームが得られる。

ソース側の出力インピーダンスはトランジスタの場合のエミッタ側に比べて、出力インピ

ードンスは高めの値である。

これは、F E T の g_m をトランジスタの h_{fe} に換算すると約 10 分の 1 程度しかないためである。出力インピーダンスを低くするには g_m の大きな F E T を選ぶこと。

6. トランジスタの反転増幅回路の場合と同様に、ソース側の負荷抵抗の変動がドレイン側に影響を及ぼすかどうかを確認する。

ソース側の負荷抵抗を 100 オームから 10 K まで変動させたとき、

負荷抵抗が 1 K 以上であれば、ソース側のゲインはほとんど変化しないが、

ドレイン側のゲインは約 -1.5 から -1 まで大きく変動している。これも、トランジスタの反転増幅回路と同じ傾向である。

7. 負荷抵抗が固定値であればドレイン側もソース側もとくにソースホロワ等のバッファは必要ないが、負荷抵抗の変動が予想される場合にはバッファを考慮すべきである。

```
B? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 2] parts[ r1 ] value[ 1e+006 j 0 ]
left[ 0] right[ 3] parts[ rs ] value[ 1e+004 j 0 ]
left[ 0] right[ 5] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[ 6] parts[ rld ] value[ 1e+006 j 0 ]
left[ 0] right[ 7] parts[ rls ] value[ 1e+006 j 0 ]
left[ 1] right[ 2] parts[ cc ] value[ 0.001 j 0 ]
left[ 2] right[ 9] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 3] right[ 7] parts[ cc ] value[ 0.001 j 0 ]
left[ 3] right[ 10] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 4] right[ 5] parts[ rd ] value[ 1e+004 j 0 ]
left[ 4] right[ 6] parts[ cc ] value[ 0.001 j 0 ]
left[ 4] right[ 11] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 8] right[ 9] parts[ r2b1 ] value[ 1e+010 j 0 ]
left[ 8] right[ 10] parts[ e1b1 ] value[ 2 j 0 ]
left[ 10] right[ 11] parts[ q1b1 ] value[ 0.004 j 0 ]
@ master[r2b1 ] left[ 8] right[ 9]
left[ 10] right[ 11] parts[ r3b1 ] value[ 1e+005 j 0 ]
最大の素子番号 = 17
最大のノード番号 = 11
```

B? まず、ノード 3 と 4 の動作点を確認します。

/point

f[0]

```
x3 [ 2 j 0 ] abs[ 2 ] arg[ 0 ]
x4 [ 22 j 0 ] abs[ 22 ] arg[ 0 ]
```

B? ノード 3、4 の動作点は電源 e2 及びグランドから 2V しか余裕がありません
従って、使用可能な出力電圧の振幅は 2Vp-p 程度となります。

もっと大振幅の出力まで、使用する必要がある場合には、
R1 に正のバイアス電圧を加える必要があります。

```

rld[6310      ] f[1,000      ]
x3 [ 1.0015 j 1.740584e-007] abs[ 1.0015] arg[9.958068e-006]
x4 [ -0.3913 j 5.983376e-006] abs[ 0.3913] arg[ 179.9991]
rld[1e+004    ] f[1,000      ]
x3 [ 1.0012 j 1.690814e-007] abs[ 1.0012] arg[9.676016e-006]
x4 [ -0.5056 j 3.937318e-006] abs[ 0.5056] arg[ 179.9996]

```

B? 出力インピーダンスは約 9.8K です。

```

rls[251.2     ] f[1,000      ]
x3 [ 0.4850 j -1.582220e-004] abs[ 0.4850] arg[ -0.0187]
x4 [ -19.5977 j -0.0057     ] abs[ 19.5977] arg[ -179.9833]
rls[398.1     ] f[1,000      ]
x3 [ 0.5989 j -9.598095e-005] abs[ 0.5989] arg[ -0.0092]
x4 [ -15.4865 j -0.0035     ] abs[ 15.4865] arg[ -179.9872]

```

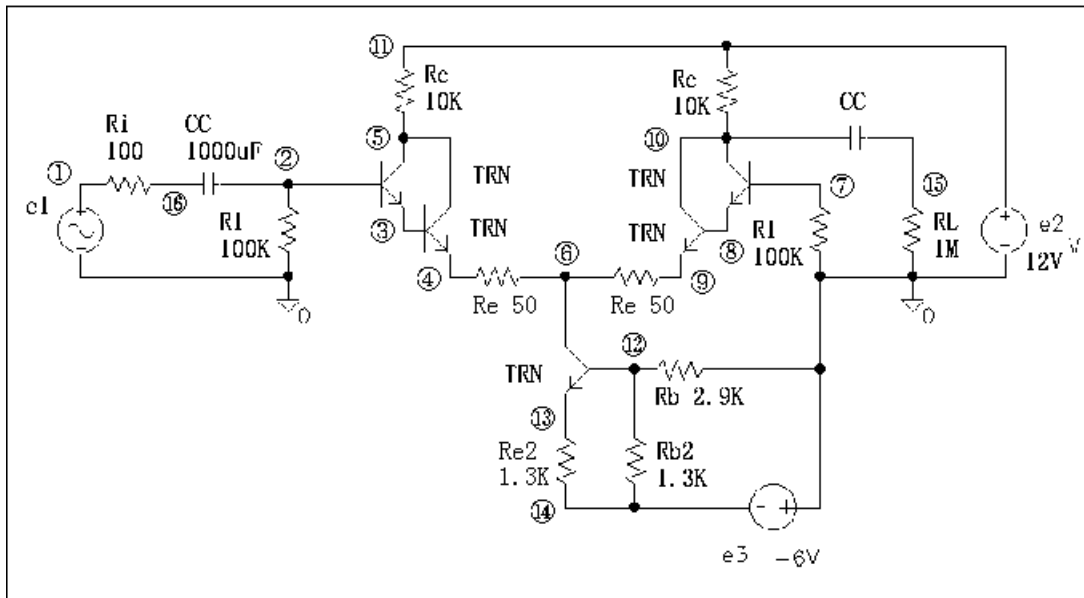
B? ノード3の出力インピーダンスは 270 オームです。

FET による位相反転回路では、
 ドレインとソースにおけるゲインが等しくなる。
 これはトランジスタと違って、ゲート電流が無いためである。
 ドレイン側の出力インピーダンスはトランジスタ回路と同等であるが、
 ソース側の出力インピーダンスはトランジスタ回路よりも高い
 これは FET の g_m をトランジスタの h_{fe} に換算すると
 約 10 分の 1 程度しかないためである。

以上で、FET 位相反転回路の解析を終了します。

(7) バッチファイル名 s11. sb で扱う回路図

トランジスタのダーリントン接続により入力インピーダンスを高めた差動増幅回路



設計の指針

1. ノード5とノード10の直流動作点を電源電圧e2の半分に設定する。

回路図の下側にあるトランジスタ回路は、一般的には定電流回路といわれており、 R_{e2} によって定電流の値が決定されている。この値を変えればノード5とノード10の電圧を変更する事が出来る。

/paraを使用して、 R_{e2} を1Kから2Kまで変化させて、ノード5とノード10の電圧が6Vになる R_{e2} の値を求める。ここでも、グラフ機能を利用する。

$R_{e2} = 1K$ で希望の電圧になることが分かる。

2. R_c に流れる電流を計算する。
(R_c の両端の電圧) / R_c で求められるから、 x_5 を求める。

f=0

/point

$(e_2 - x_5) / R_c = 0.6mA$ が得られる。

3. 入力信号の周波数が1000Hzの時のゲインを調べる。

ゲインは約90倍である。ノード5は位相反転出力、ノード10は非反転である。

4. 入力インピーダンスを求める。

$Z_i = 95K$ が得られる。これは、 R_1 の値とほとんど同じである。

R_1 の右側のインピーダンスを Z_{ii} としてこの値を計算すると、

$95K = 100K * Z_{ii} / (100K + Z_{ii})$ より、

$Z_{ii} = 1.9M$ が得られる。従って、 R_1 を1Mに増加すれば入力インピーダンスを65

0 Kに増加することが可能であることがわかる。

$$1\text{ M} * 1.9\text{ M} / (1\text{ M} + 1.9\text{ M}) = 650\text{ K}$$

しかし、この場合にはトランジスタのベース電流が減少するためにゲインが約50に下がってしまう。

```
B? /dpart
left[ 0] right[ 1] parts[ e1          ] value[      1 j 0 ]
left[ 0] right[ 2] parts[ r1          ] value[ 1e+005 j 0 ]
left[ 0] right[ 7] parts[ r1          ] value[ 1e+005 j 0 ]
left[ 0] right[11] parts[ e2          ] value[     12 j 0 ]
left[ 0] right[12] parts[ rb          ] value[    2900 j 0 ]
left[ 0] right[14] parts[ e3          ] value[      -6 j 0 ]
left[ 0] right[15] parts[ r1          ] value[ 1e+006 j 0 ]
left[ 1] right[16] parts[ ri          ] value[     100 j 0 ]
left[ 2] right[16] parts[ cc          ] value[    0.001 j 0 ]
left[ 2] right[20] parts[ r1b1        ] value[     0.1 j 0 ]
left[ 3] right[17] parts[ rbb1        ] value[     0.1 j 0 ]
left[ 3] right[24] parts[ r1b2        ] value[     0.1 j 0 ]
left[ 4] right[ 6] parts[ re          ] value[     50 j 0 ]
left[ 4] right[21] parts[ rbb2        ] value[     0.1 j 0 ]
left[ 5] right[11] parts[ rc          ] value[ 1e+004 j 0 ]
left[ 5] right[19] parts[ rbb1        ] value[     0.1 j 0 ]
left[ 5] right[23] parts[ rbb2        ] value[     0.1 j 0 ]
left[ 6] right[ 9] parts[ re          ] value[     50 j 0 ]
left[ 6] right[35] parts[ rbb5        ] value[     0.1 j 0 ]
left[ 7] right[28] parts[ r1b3        ] value[     0.1 j 0 ]
left[ 8] right[25] parts[ rbb3        ] value[     0.1 j 0 ]
left[ 8] right[32] parts[ r1b4        ] value[     0.1 j 0 ]
left[ 9] right[29] parts[ rbb4        ] value[     0.1 j 0 ]
left[10] right[11] parts[ rc          ] value[ 1e+004 j 0 ]
left[10] right[15] parts[ cc          ] value[    0.001 j 0 ]
left[10] right[27] parts[ rbb3        ] value[     0.1 j 0 ]
left[10] right[31] parts[ rbb4        ] value[     0.1 j 0 ]
left[12] right[14] parts[ rb2         ] value[    1300 j 0 ]
left[12] right[36] parts[ r1b5        ] value[     0.1 j 0 ]
left[13] right[14] parts[ re2         ] value[    1300 j 0 ]
left[13] right[33] parts[ rbb5        ] value[     0.1 j 0 ]
left[17] right[18] parts[ reb1        ] value[     26 j 0 ]
left[17] right[19] parts[ kb1         ] value[     100 j 0 ]
@ master[reb1      ] left[17] right[18]
left[18] right[20] parts[ ebb1        ] value[     0.6 j 0 ]
left[21] right[22] parts[ reb2        ] value[     26 j 0 ]
left[21] right[23] parts[ kb2         ] value[     100 j 0 ]
@ master[reb2      ] left[21] right[22]
left[22] right[24] parts[ ebb2        ] value[     0.6 j 0 ]
left[25] right[26] parts[ reb3        ] value[     26 j 0 ]
left[25] right[27] parts[ kb3         ] value[     100 j 0 ]
@ master[reb3      ] left[25] right[26]
left[26] right[28] parts[ ebb3        ] value[     0.6 j 0 ]
left[29] right[30] parts[ reb4        ] value[     26 j 0 ]
left[29] right[31] parts[ kb4         ] value[     100 j 0 ]
@ master[reb4      ] left[29] right[30]
left[30] right[32] parts[ ebb4        ] value[     0.6 j 0 ]
left[33] right[34] parts[ reb5        ] value[     26 j 0 ]
left[33] right[35] parts[ kb5         ] value[     100 j 0 ]
@ master[reb5      ] left[33] right[34]
left[34] right[36] parts[ ebb5        ] value[     0.6 j 0 ]
最大の素子番号 = 46
最大のノード番号 = 36
```

B? ノード 5、10 の直流動作点を確認します。
キー入力待ち？

B? /point

```
f[0]
x5 [ 7.2470 j 0 ] abs[ 7.2470] arg[ 0]
x10 [ 7.2470 j 0 ] abs[ 7.2470] arg[ 0]
```

B? Re2 を変更して、動作点を 6V に設定します

B? /para

データファイル名は ? test

値を変化させる素子名は ? re2

para re2 最低値 ? 1k

para re2 最高値 ? 2k

para re2 ステップ ? 0.2k

```
re2[1000] f[0]
x5 [ 5.8341 j 0 ] abs[ 5.8341] arg[ 0]
x10 [ 5.8341 j 0 ] abs[ 5.8341] arg[ 0]
re2[1200] f[0]
x5 [ 6.8539 j 0 ] abs[ 6.8539] arg[ 0]
x10 [ 6.8539 j 0 ] abs[ 6.8539] arg[ 0]
re2[1400] f[0]
x5 [ 7.5843 j 0 ] abs[ 7.5843] arg[ 0]
x10 [ 7.5843 j 0 ] abs[ 7.5843] arg[ 0]
re2[1600] f[0]
x5 [ 8.1331 j 0 ] abs[ 8.1331] arg[ 0]
x10 [ 8.1331 j 0 ] abs[ 8.1331] arg[ 0]
re2[1800] f[0]
x5 [ 8.5605 j 0 ] abs[ 8.5605] arg[ 0]
x10 [ 8.5605 j 0 ] abs[ 8.5605] arg[ 0]
re2[2000] f[0]
x5 [ 8.9029 j 0 ] abs[ 8.9029] arg[ 0]
x10 [ 8.9029 j 0 ] abs[ 8.9029] arg[ 0]
```

B? Re=1K とします。

B?

今決定した回路定数における、Rc に流れる電流を計算します。

/point

```
f[0]
x5 [ 5.8341 j 0 ] abs[ 5.8341] arg[ 0]
x10 [ 5.8341 j 0 ] abs[ 5.8341] arg[ 0]
```

B? Rc に流れる電流 = (Rc の両端の電圧)/Rc で求められます。

(e2-x5)/rc

6.165861e-004 j 0

B? 電流は約 0.6mA です。

```

B? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1k
range 周波数 最高値 ? 1k
range 周波数 ステップ ? 1
e1[1      ] j 0      ]
f[1,000    ]
x5 [ -90.3685 j -1.564120e-004] abs[ 90.3685] arg[ -179.9999]
x10 [ 89.4737 j 1.547225e-004] abs[ 89.4737] arg[9.907875e-005]
B? ゲインは約 90 倍です。ノード5は位相反転、ノード10は非反転です。

```

Ri を 1k から 1M まで変化させて、x10=0.5 となる Ri の値を求めます。

```

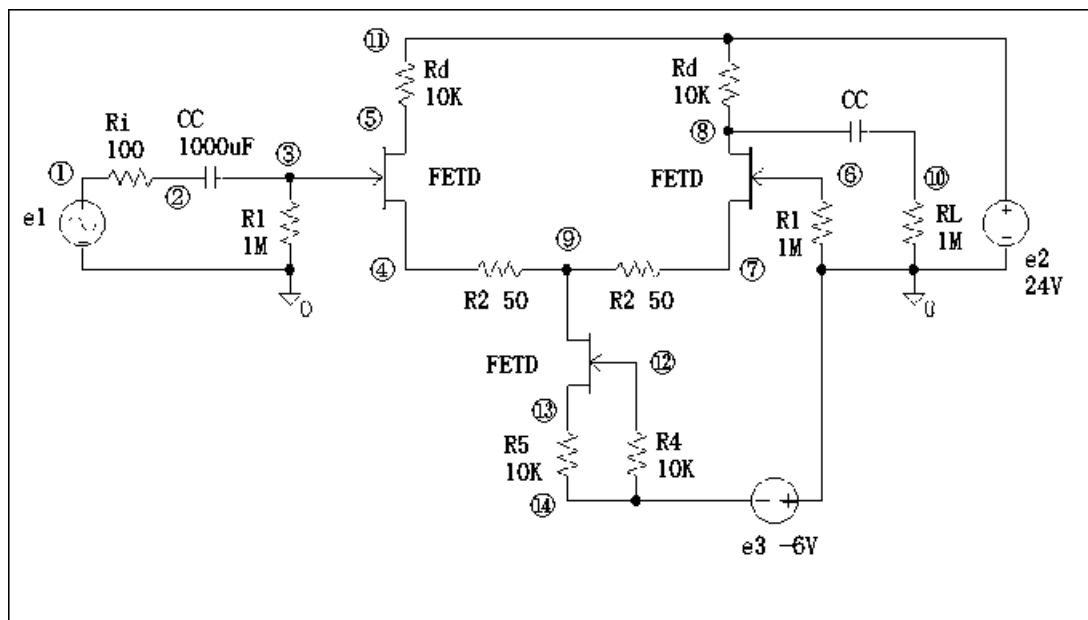
ri[4.642e+004] f[1,000    ]
x5 [ -0.6710 j -7.723226e-007] abs[ 0.6710] arg[ -179.9999]
x10 [ 0.6643 j 7.636302e-007] abs[ 0.6643] arg[6.586223e-005]
ri[1e+005] f[1,000    ]
x5 [ -0.4836 j -4.011454e-007] abs[ 0.4836] arg[ -180]
x10 [ 0.4788 j 3.964200e-007] abs[ 0.4788] arg[4.744088e-005]

```

Ri = 93.9K で、X10 = 0.5 になるので、入力インピーダンスは 93.9K です。

N型デプリーションタイプのFETを使用した差動増幅回路

N型デプリーションタイプのFETを使用した差動増幅回路



1. ノード5とノード8の直流動作点を電源電圧 e_2 の半分に設定する。

R 5によって定電流の値が決定されている。この値を変えればノード5とノード8の電圧を変更する事が出来る。

$R_5 = 600$ オームで希望の電圧になることが分かる。

トランジスタのダーリントン接続の回路に比べると、ゲインはかなり小さい。

$Z_i = 1/M$ が得られる。これは、 R_1 の値と同じである。

4. ノード5側にもノード8側と同じように負荷抵抗を接続すると、ゲインも等しくなる。

```

B? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 3] parts[ r1 ] value[ 1e+006 j 0 ]
left[ 0] right[ 6] parts[ r1 ] value[ 1e+006 j 0 ]
left[ 0] right[10] parts[ r1 ] value[ 1e+006 j 0 ]
left[ 0] right[11] parts[ e2 ] value[ 24 j 0 ]
left[ 0] right[14] parts[ e3 ] value[ -12 j 0 ]
left[ 1] right[ 2] parts[ ri ] value[ 100 j 0 ]
left[ 2] right[ 3] parts[ cc ] value[ 0.001 j 0 ]
left[ 3] right[16] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 4] right[ 9] parts[ r2 ] value[ 50 j 0 ]
left[ 4] right[17] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 5] right[11] parts[ r3 ] value[ 1e+004 j 0 ]
left[ 5] right[18] parts[ r1b1 ] value[ 0.1 j 0 ]
left[ 6] right[20] parts[ r1b2 ] value[ 0.1 j 0 ]
left[ 7] right[ 9] parts[ r2 ] value[ 50 j 0 ]
left[ 7] right[21] parts[ r1b2 ] value[ 0.1 j 0 ]
left[ 8] right[10] parts[ cc ] value[ 0.001 j 0 ]
left[ 8] right[11] parts[ r3 ] value[ 1e+004 j 0 ]
left[ 8] right[22] parts[ r1b2 ] value[ 0.1 j 0 ]
left[ 9] right[26] parts[ r1b3 ] value[ 0.1 j 0 ]
left[12] right[14] parts[ r4 ] value[ 1e+006 j 0 ]
left[12] right[24] parts[ r1b3 ] value[ 0.1 j 0 ]
left[13] right[14] parts[ r5 ] value[ 1000 j 0 ]
left[13] right[25] parts[ r1b3 ] value[ 0.1 j 0 ]
left[15] right[16] parts[ r2b1 ] value[ 1e+010 j 0 ]
left[15] right[17] parts[ e1b1 ] value[ 2 j 0 ]
left[17] right[18] parts[ q1b1 ] value[ 0.004 j 0 ]
@ master[r2b1 ] left[15] right[16]
left[17] right[18] parts[ r3b1 ] value[ 1e+005 j 0 ]
left[19] right[20] parts[ r2b2 ] value[ 1e+010 j 0 ]
left[19] right[21] parts[ e1b2 ] value[ 2 j 0 ]
left[21] right[22] parts[ r3b2 ] value[ 1e+005 j 0 ]
left[21] right[22] parts[ q1b2 ] value[ 0.004 j 0 ]
@ master[r2b2 ] left[19] right[20]
left[23] right[24] parts[ r2b3 ] value[ 1e+010 j 0 ]
left[23] right[25] parts[ e1b3 ] value[ 2 j 0 ]
left[25] right[26] parts[ q1b3 ] value[ 0.004 j 0 ]
@ master[r2b3 ] left[23] right[24]
left[25] right[26] parts[ r3b3 ] value[ 1e+005 j 0 ]
最大の素子番号 = 36
最大のノード番号 = 26
B? ノード5と8の現在の動作点を確認します

```

```

B? /point
f[0]
x5 [ 15.8791 j 0 ] abs[ 15.8791] arg[ 0]
x8 [ 15.8791 j 0 ] abs[ 15.8791] arg[ 0]
B? R5 を変化させて、動作点が 12V となる値を求めます

```

```

/para
データファイル名は ? test
値を変化させる素子名は ? r5
para r5 最低値 ? 100
para r5 最高値 ? 10k
para r5 ステップ ? 100

```

```

r5[464.2 ] f[0 ]
x5 [ 9.7852 j 0 ] abs[ 9.7852] arg[ 0]
x8 [ 9.7852 j 0 ] abs[ 9.7852] arg[ 0]
r5[774.3 ] f[0 ]
x5 [ 14.0892 j 0 ] abs[ 14.0892] arg[ 0]
x8 [ 14.0892 j 0 ] abs[ 14.0892] arg[ 0]

```

B? R5=600 位で目標の動作点となります。

```

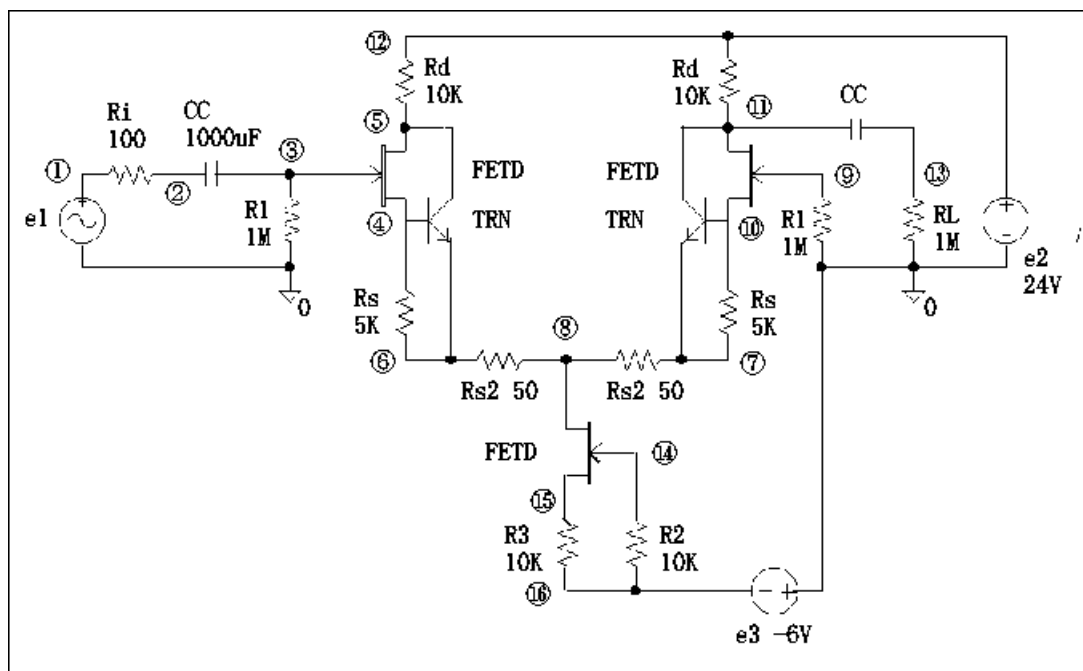
B? ゲインを確認します
/range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1k
range 周波数 最高値 ? 1k
range 周波数 ステップ ? 1
e1[1 j 0 ]
f[1.000 ]
x5 [ -15.3849 j -2.449362e-006] abs[ 15.3849] arg[ -180]
x8 [ 15.2181 j 2.398985e-006] abs[ 15.2181] arg[9.032112e-006]
B? ゲインは約 15 倍ですから、ダーリントン接続による回路よりかなり低い値です。

```

(9) バッチファイル名 ss2.sb で扱う回路図

FETとトランジスタを組み合わせた差動増幅回路



設計の指針

1. ノード5とノード11の直流動作点を電源電圧 e_2 の半分に設定する。

回路図の下側にあるFET回路は、一般的には定電流回路といわれており、

R3によって定電流の値が決定されている。この値を変えればノード5とノード11の電圧を変更する事が出来る。

/paraを使用して、R3を100オームから10Kまで変化させて、ノード5とノード11の電圧が12Vになる R_{e2} の値を求める。

ここでも、グラフ機能を利用する。

$R3 = 600$ オームで希望の電圧になることが分かる。

2. 入力信号の周波数が1000Hzの時のゲインを調べる。

ゲインは約6.3倍である。ノード5は位相反転出力、ノード11は非反転である。

トランジスタのダーリントン接続の場合とFETのみの場合の中間のゲインが得られる。

トランジスタの場合に $R1 = 1M$ とした場合よりもゲインは大きい。

3. 入力インピーダンスを求める。

$Z_i = 1M$ が得られる。これは、 $R1$ の値と同じである。

$R1 = 10M$ に増加すれば入力インピーダンスを10Mに増加することが可能であることがわかる。

この場合にはトランジスタの回路とは異なり、ゲインに影響はない。

4. ノード5側にもノード11側と同じように負荷抵抗を接続すればゲインを等しくする事が出来る。

left[0]	right[1]	parts[e1	value[1 j 0
left[0]	right[3]	parts[r1	value[1e+006 j 0
left[0]	right[9]	parts[r1	value[1e+006 j 0
left[0]	right[12]	parts[e2	value[24 j 0
left[0]	right[13]	parts[r1	value[1e+006 j 0
left[0]	right[16]	parts[e3	value[-12 j 0
left[1]	right[2]	parts[ri	value[100 j 0
left[2]	right[3]	parts[cc	value[0.001 j 0
left[3]	right[18]	parts[r1b1	value[0.1 j 0
left[4]	right[6]	parts[rs	value[5000 j 0
left[4]	right[19]	parts[r1b1	value[0.1 j 0
left[4]	right[24]	parts[r1b2	value[0.1 j 0
left[5]	right[12]	parts[rd	value[1e+004 j 0
left[5]	right[20]	parts[r1b1	value[0.1 j 0
left[5]	right[23]	parts[rbb2	value[0.1 j 0
left[6]	right[8]	parts[rs2	value[50 j 0
left[6]	right[21]	parts[rbb2	value[0.1 j 0
left[7]	right[8]	parts[rs2	value[50 j 0
left[7]	right[10]	parts[rs	value[5000 j 0
left[7]	right[25]	parts[rbb4	value[0.1 j 0
left[8]	right[32]	parts[r1b5	value[0.1 j 0
left[9]	right[34]	parts[r1b3	value[0.1 j 0
left[10]	right[28]	parts[r1b4	value[0.1 j 0
left[10]	right[35]	parts[r1b3	value[0.1 j 0
left[11]	right[12]	parts[rd	value[1e+004 j 0
left[11]	right[13]	parts[cc	value[0.001 j 0


```

left[ 11] right[ 27] parts[ rbb4 ] value[ 0.1 j 0 ]
left[ 11] right[ 36] parts[ r1b3 ] value[ 0.1 j 0 ]
left[ 14] right[ 16] parts[ r2 ] value[ 1e+004 j 0 ]
left[ 14] right[ 30] parts[ r1b5 ] value[ 0.1 j 0 ]
left[ 15] right[ 16] parts[ r3 ] value[ 600 j 0 ]
left[ 15] right[ 31] parts[ r1b5 ] value[ 0.1 j 0 ]
left[ 17] right[ 18] parts[ r2b1 ] value[ 1e+010 j 0 ]
left[ 17] right[ 19] parts[ e1b1 ] value[ 2 j 0 ]
left[ 19] right[ 20] parts[ q1b1 ] value[ 0.004 j 0 ]
@ master[r2b1 ] left[ 17] right[ 18]
left[ 19] right[ 20] parts[ r3b1 ] value[ 1e+005 j 0 ]
left[ 21] right[ 22] parts[ reb2 ] value[ 26 j 0 ]
left[ 21] right[ 23] parts[ kb2 ] value[ 100 j 0 ]
@ master[reb2 ] left[ 21] right[ 22]
left[ 22] right[ 24] parts[ ebb2 ] value[ 0.6 j 0 ]
left[ 25] right[ 26] parts[ reb4 ] value[ 26 j 0 ]
left[ 25] right[ 27] parts[ kb4 ] value[ 100 j 0 ]
@ master[reb4 ] left[ 25] right[ 26]
left[ 26] right[ 28] parts[ ebb4 ] value[ 0.6 j 0 ]
left[ 29] right[ 30] parts[ r2b5 ] value[ 1e+010 j 0 ]
left[ 29] right[ 31] parts[ e1b5 ] value[ 2 j 0 ]
left[ 31] right[ 32] parts[ r3b5 ] value[ 1e+005 j 0 ]
left[ 31] right[ 32] parts[ q1b5 ] value[ 0.004 j 0 ]
@ master[r2b5 ] left[ 29] right[ 30]
left[ 33] right[ 34] parts[ r2b3 ] value[ 1e+010 j 0 ]
left[ 33] right[ 35] parts[ e1b3 ] value[ 2 j 0 ]
left[ 35] right[ 36] parts[ q1b3 ] value[ 0.004 j 0 ]
@ master[r2b3 ] left[ 33] right[ 34]
left[ 35] right[ 36] parts[ r3b3 ] value[ 1e+005 j 0 ]
最大の素子番号 = 50
最大のノード番号 = 36

```

```

B? ノード5と11の現在の動作点を確認します
f=0
0 j 0

B? /point
f[0 ]
x5 [ 23.0105 j 0 ] abs[ 23.0105] arg[ 0]
x11 [ 23.0105 j 0 ] abs[ 23.0105] arg[ 0]
B? R3 を変化させて、動作点が 12V となる値を求めます。

```

```

r3[464.2 ] f[0 ]
x5 [ 9.7896 j 0 ] abs[ 9.7896] arg[ 0]
x11 [ 9.7896 j 0 ] abs[ 9.7896] arg[ 0]
r3[774.3 ] f[0 ]
x5 [ 14.0937 j 0 ] abs[ 14.0937] arg[ 0]
x11 [ 14.0937 j 0 ] abs[ 14.0937] arg[ 0]

```

```

B? R3=600 で目標の動作点となります
R3=600 に設定します

```



```

B? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 1k
range 周波数 最高値 ? 1k
range 周波数 ステップ ? 1
e1[1 j 0]
f[1,000]
x5 [ -64.2221 j -1.023649e-005] abs[ 64.2221] arg[ -180]
x11 [ 63.5718 j 1.003277e-005] abs[ 63.5718] arg[9.042308e-006]
B? ゲインは、6.3倍となり FET のみより大きく、
トランジスタのダーリントン接続よりも小さい値が得られました。

```

(10) バッチファイル s20. sb で扱う3つの回路図

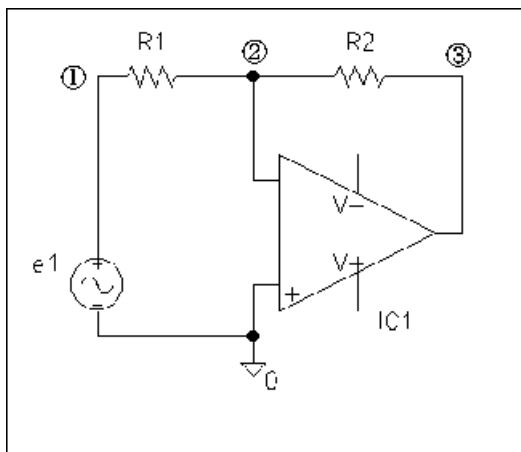


図 s20_1 反転増幅回路

設計の指針

1. 理想オペアンプでは、

a. 入力インピーダンスは無限大

b. ゲインも無限大

c. 周波数特性も無限大

の特性を仮定している。

しかし、実際に市販されているオペアンプではそれらはすべて有限の値となっているため理想の性能からは誤差が生じる。

2. 理想オペアンプでは、a. と b. の仮定からノード2とノード0の電位は等しくなる。

このためノード2はグラウンドのレベルと等しくなるため、仮想グラウンドと呼ばれる。

3. また、この仮定から R_1 に流れる電流と R_2 に流れる電流が等しくなる。従って、ノード1とノード3の位相は逆になり、ゲインは $-R_2/R_1$ となる。
4. 試しに、 $e_1 = 2$ とすれば、ノード3の電圧は -20 V になるはずである。
5. この等価回路で使用しているオペアンプの入力インピーダンスが 1 M であり、ゲインが 1000000 のために、ノード2は 0 V にならず、 -10 uV 程度になっている。
また、ゲインも抵抗の比率に比べて $1/1000000$ の誤差を含んでいる。
6. 試しに、オペアンプのゲインを 1000 と 1000000 に変更してみると、
ゲイン 1000 の時、ノード2は -0.0101 V 、ゲインは $1/88.5$ の誤差、
ゲイン 1000000 の時は、ノード2は -0.1 mV 、ゲインは $1/9091$ の誤差となる。
ゲインを 1000000 に戻して、
7. 試しに、オペアンプの入力インピーダンスを 100 K と 10 M に変更してみると、
 100 K の時も 10 M の時も、 1 M の場合と変わらないので、誤差はゲインによって大きく左右されるといっても良い。
8. 反転回路の入力インピーダンスは R_1 の値そのものである。

```
B? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 3] parts[ v1b1 ] value[ 1e+006 j 0 ]
@ master[rlb1] left[ 2] right[ 0]
left[ 1] right[ 2] parts[ r1 ] value[ 1000 j 0 ]
left[ 2] right[ 0] parts[ r1b1 ] value[ 1e+006 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 1e+004 j 0 ]
最大の素子番号 = 5
最大のノード番号 = 3
B? 最初に直流におけるゲインを確認します。
```

```
B? /range
ac 入力信号源名は ? e1
値は ? 1

range 周波数 最低値 ? 0
range 周波数 最高値 ? 0
range 周波数 ステップ ? 1
e1[1] j 0 ]
f[0] ]
x1 [ 1 j 0 ] abs[ 1 ] arg[ 0 ]
x2 [-1.000011e-005 j 0 ] abs[ 1.000011e-005 ] arg[ 180 ]
x3 [-10.0001 j 0 ] abs[ 10.0001 ] arg[ -180 ]
B? ノード2の電圧はほとんど 0V となっています。
オペアンプの非反転入力ノードがグランドレベルの時に、
オペアンプの反転入力ノードのことを、仮想グランドと呼ぶことがあります。

また、ノード3の電圧はほとんど -10V となっています。
この値は、  $-R_2/R_1$  の比率に等しくなっています。

試しに、 $e_1=2$  とすれば、ノード3は  $-20$  となるはずです。
```

```

e1[2      j 0      ]
f[0      ]
x1 [      2 j 0      ] abs[      2] arg[      0]
x2 [-2.000022e-005 j 0] abs[2.000022e-005] arg[      180]
x3 [-20.0002 j 0      ] abs[      20.0002] arg[      -180]
B? 予想どおり、-20V となっています。

```

B? 入力インピーダンスは、 $Z_{in} = e1 / (R1 \text{ に流れる電流})$ で求められます。

```

e1=2
      2 j 0

```

```

B? i1=(x1-x2)/r1
      0.0020 j 0

```

```

B? zin=e1/i1
      999.9900 j 0

```

B? 入力インピーダンスはほとんど、 $R1$ と等しくなっています。

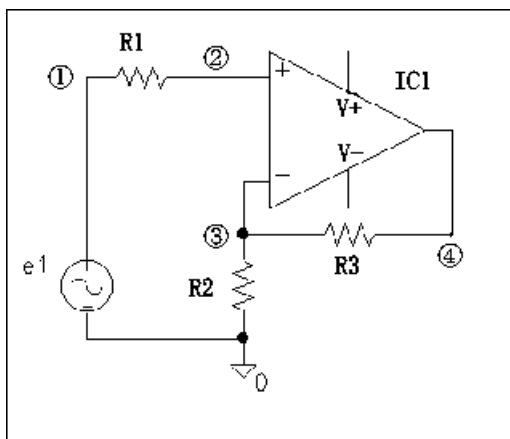


図 s20_2 非反転増幅回路

設計の指針

1. 非反転回路では、理想オペアンプの仮定 a. と b. からノード 2 とノード 3 及びノード 1 の電位は等しくなる。
2. 従って、ノード 4 の電圧はノード 1 の電圧に $R3 / R2$ を掛けた値にノード 1 の電圧を加えた値に等しくなる。よって非反転回路のゲインは、 $(1 + R3 / R2)$ となる。
3. 非反転回路の入力インピーダンス Z_i を求める。

$Z_i = e1 / (R1 \text{ に流れる電流値})$

$R1 \text{ に流れる電流値} = (\text{ノード 1 の電位} - \text{ノード 2 の電位}) / R1 = 1 \text{ pA}$

より、 $Z_i = 90000 \text{ M}$ となり、非常に入力インピーダンスが高いことが分かる。

```

B? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 3] parts[ r2 ] value[ 1000 j 0 ]
left[ 0] right[ 4] parts[ v1b1 ] value[ 1e+006 j 0 ]
@ master[r1b1 ] left[ 3] right[ 2]
left[ 1] right[ 2] parts[ r1 ] value[ 1e+006 j 0 ]
left[ 3] right[ 2] parts[ r1b1 ] value[ 1e+006 j 0 ]
left[ 3] right[ 4] parts[ r3 ] value[ 1e+004 j 0 ]
最大の素子番号 = 6
最大のノード番号 = 4
B? 先程と同様に、直流におけるゲインを調べます。

```

```

e1[1 j 0 ]
f[0 ]
x1 [ 1 j 0 ] abs[ 1] arg[ 0]
x2 [ 1 j 0 ] abs[ 1] arg[ 0]
x3 [ 1 j 0 ] abs[ 1] arg[ 0]
x4 [ 11.0002 j 0 ] abs[ 11.0002] arg[ 0]
B? 今度は、ノード1、2及び3が全て同じ電圧となりました。
これは、入力インピーダンスが十分高くて R1 にはほとんど
電流が流れないことを表わしています。

```

```

B? また、オペアンプの電圧増幅率が非常に高いために
ノード2と3の電圧がほとんど同じ電圧になっています。

ノード4の電圧は 11V となっていますがこれは
(R2+R3)/R2 の比率と等しくなっています。

入力インピーダンスは  $Z_{in} = e1 / (R1 \text{ に流れる電流})$  で求められます。
i1=abs(x1-x2)/r1
1.100024e-011 j 0

B? zin=e1/i1
90,907,089,999.2307 j 0

B? 入力インピーダンスは約90,000M オームと求められました。

```

s20_1 の R1 または R2 にコンデンサやインダクタを組み合わせると周波数によって、ゲインが変化する回路を構成することができます。

s20_2 では、R2 または R3 にコンデンサやインダクタを組み合わせる事で、周波数によって、ゲインが変化する回路を構成することができます。

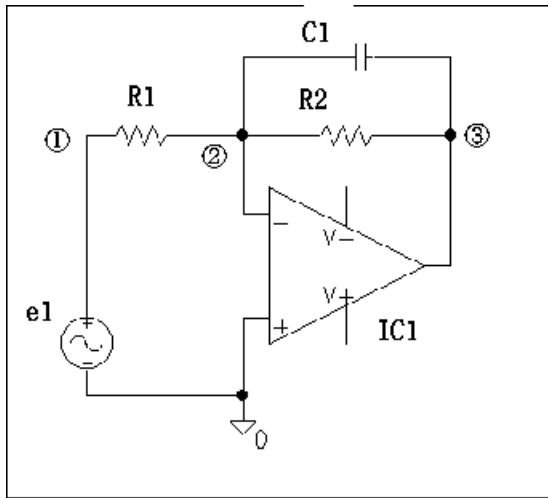


図 s20_3 ローパスフィルタ

設計の指針

1. 図 20-1 の R 1 または R 2 にコンデンサやインダクタを組み合わせるとゲインが周波数によって変化ようになる。
2. 図 20-2 では、R 2 または R 3 にコンデンサやインダクタを組み合わせるとゲインが周波数によって変化ようになる。
3. 図 20-3 は一つの例として、R 2 と並列にコンデンサを接続した回路である。この回路ではゲインは－（C 1 と R 2 の並列インピーダンス）／R 1 と表される。従って、周波数が高くなるに従ってゲインは小さくなり、一般的にはローパスフィルタと呼ばれる特性となる。周波数特性を確認するには、グラフ機能を用いるのが最も直感的に理解が出来る。
4. 角周波数を ω とすると、ゲイン G は、

$$G = -\frac{R_2}{R_1} \frac{1}{1 + j\omega C_1 R_2}, \quad \therefore |G|_{\omega} = \frac{R_2}{R_1} \frac{1}{\sqrt{1 + (\omega C_1 R_2)^2}}$$

と表される。周波数が 0（直流）のときのゲインを $G_0 = \frac{R_2}{R_1}$ とすると、

$\omega_c C_1 R_2 = 1$ の時に、 $|G|_{\omega_c} = \frac{G_0}{\sqrt{2}} \rightarrow -3dB$ となる。

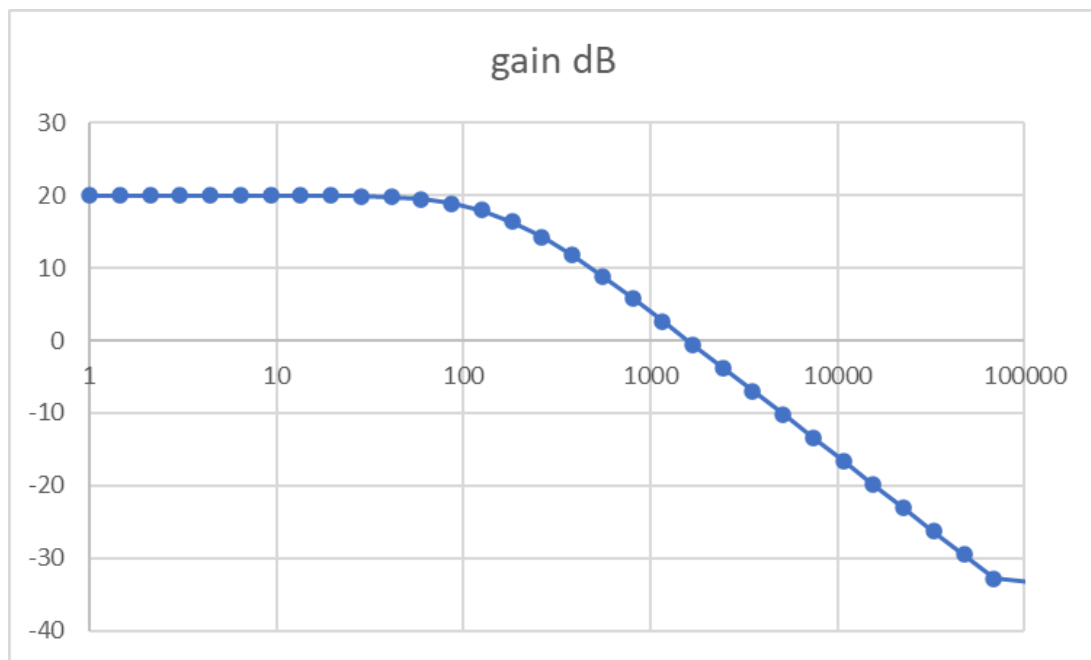
このことから、 $\omega_c = \frac{1}{C_1 R_2} \rightarrow f_c = \frac{1}{2\pi C_1 R_2}$ をカットオフ周波数と呼ぶ。

```

B? /dpart
left[ 0] right[ 1] parts[ e1 ] value[ 1 j 0 ]
left[ 0] right[ 3] parts[ vlb1 ] value[ 1e+006 j 0 ]
@ master[rlb1 ] left[ 2] right[ 0]
left[ 1] right[ 2] parts[ r1 ] value[ 1000 j 0 ]
left[ 2] right[ 0] parts[ rlb1 ] value[ 1e+006 j 0 ]
left[ 2] right[ 3] parts[ c1 ] value[ 1e-007 j 0 ]
left[ 2] right[ 3] parts[ r2 ] value[ 1e+004 j 0 ]
最大の素子番号 = 6
最大のノード番号 = 3
range 周波数 最低値 ? 1
range 周波数 最高値 ? 100k
range 周波数 ステップ ? 1
el[1 ] j 0 ]
f[1 ]
x3 [ -9.9997 j 0.0628 ] DB[ 19.9999] arg[ 179.6400]
f[1.4497 ]
x3 [ -9.9993 j 0.0911 ] DB[ 19.9997] arg[ 179.4781]
f[2.1017 ]
x3 [ -9.9984 j 0.1320 ] DB[ 19.9993] arg[ 179.2434]
f[3.0470 ]
x3 [ -9.9964 j 0.1914 ] DB[ 19.9985] arg[ 178.9032]
f[4.4173 ]
x3 [ -9.9924 j 0.2773 ] DB[ 19.9968] arg[ 178.4101]
f[6.4040 ]
x3 [ -9.9839 j 0.4017 ] DB[ 19.9931] arg[ 177.6958]
f[9.2841 ]
x3 [ -9.9662 j 0.5814 ] DB[ 19.9853] arg[ 176.6615]
f[13.4596 ]
x3 [ -9.9291 j 0.8397 ] DB[ 19.9691] arg[ 175.1660]
f[19.5129 ]
x3 [ -9.8520 j 1.2079 ] DB[ 19.9353] arg[ 173.0102]
f[28.2887 ]
x3 [ -9.6938 j 1.7230 ] DB[ 19.8650] arg[ 169.9212]
f[41.0113 ]
x3 [ -9.3774 j 2.4164 ] DB[ 19.7209] arg[ 165.5501]
f[59.4557 ]
x3 [ -8.7754 j 3.2783 ] DB[ 19.4327] arg[ 159.5156]
f[86.1954 ]
x3 [ -7.7321 j 4.1876 ] DB[ 18.8830] arg[ 151.5606]
f[124.9609 ]
x3 [ -6.1864 j 4.8573 ] DB[ 17.9144] arg[ 141.8624]
f[181.1609 ]
x3 [ -4.3561 j 4.9584 ] DB[ 16.3910] arg[ 131.2999]
f[262.6364 ]
x3 [ -2.6859 j 4.4323 ] DB[ 14.2909] arg[ 121.2152]
f[380.7546 ]
x3 [ -1.4873 j 3.5583 ] DB[ 11.7242] arg[ 112.6847]
f[551.9954 ]
x3 [ -0.7675 j 2.6620 ] DB[ 8.8509] arg[ 106.0835]
f[800.2502 ]
x3 [ -0.3805 j 1.9131 ] DB[ 5.8034] arg[ 101.2482]
f[1,160.1553]
x3 [ -0.1847 j 1.3465 ] DB[ 2.6651] arg[ 97.8112]
f[1,681.9243]
x3 [ -0.0887 j 0.9379 ] DB[ -0.5184] arg[ 95.4056]
f[2,438.3541]
x3 [ -0.0424 j 0.6499 ] DB[ -3.7240] arg[ 93.7344]
f[3,534.9811]
x3 [ -0.0202 j 0.4493 ] DB[ -6.9401] arg[ 92.5779]
f[5,124.8059]
x3 [ -0.0096 j 0.3103 ] DB[ -10.1613] arg[ 91.7788]
f[7,429.6395]
x3 [ -0.0046 j 0.2141 ] DB[ -13.3849] arg[ 91.2272]
f[10,771.0506]
x3 [ -0.0022 j 0.1477 ] DB[ -16.6097] arg[ 90.8465]
f[15,615.2301]
x3 [ -0.0010 j 0.1019 ] DB[ -19.8350] arg[ 90.5839]
f[22,638.0341]
x3 [-4.942395e-004 j 0.0703 ] DB[ -23.0606] arg[ 90.4028]
f[32,819.2787]
x3 [-2.351625e-004 j 0.0485 ] DB[ -26.2863] arg[ 90.2778]
f[47,579.4431]
x3 [-1.118904e-004 j 0.0335 ] DB[ -29.5120] arg[ 90.1917]
f[68,977.8538]
x3 [-5.323714e-005 j 0.0231 ] DB[ -32.7378] arg[ 90.1322]
f[100,000 ]
x3 [-2.533000e-005 j 0.0159 ] DB[ -35.9636] arg[ 90.0912]
B? ある周波数を起点にゲインが減衰するローパスフィルタ
の一般的な周波数特性を示しています。

```

上の周波数とゲインのデータをエクセルでグラフを作成した。



低い周波数ではゲインは 20dB 程度だが、100Hz を越えるあたりからゲインが低下している。ローパスフィルタの特性を示している。