5-4 水晶発振回路

■水晶振動子

この節では、やや気難しい回路を題材として、各種の解析を適用します。

また、一度作った回路を部品として再利用する階層設計の方法を練習することも本節の目標です。

*

「水晶発振回路」は、正確な周波数を安定して発生するため、マイコンやFPGA[※18]などの基準クロック、無線通信回路の基準周波数、時間計測などに用いられます。

そして、「水晶発振回路」の周波数を決定している部品が、「水晶振動子」(Xtal、またはクリスタル)です。

「水晶振動子」は、高純度な人工水晶の結晶片を用いて作られます。

また、「水晶振動子」を含む「水晶発振回路」全体を小さなモジュールにした製品が、「水晶発振器」の名称で市販されています。

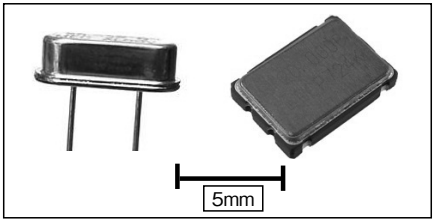

図5-30 「水晶振動子」(左)と「水晶発振器」(右)の外観

「50MEGHz」ぐらいまでの周波数に対応したものが各種市販されていますが、「発振周波数」は固定なので、「分周器」「PLL[※19]」「DLL[※20]」といった回路を用いて、必要な周波数に変換します。

※18「Field Programmable Gate Array」。CADソフトを使って設計した論理回路を、コンピュータから書き込むことができる大規模集積回路。
※19「Phase Locked Loop」の略。基準となる水晶発振回路の分数倍の正確な周波数の高周波信号を作れる。
※20「Delay Locked Loop」の略。基準となる水晶発振回路の分数倍のジッタ(周期揺らぎ)のない高周波クロックを作れる。

マイコンのクロックに使う場合は、発振回路の一部がマイコンに内蔵されているため、メーカー指定の「水晶振動子」と「キャパシタンス」を外付けすれば発振ができ、このため少しの部品追加で※21、「タイマー※22」や「通信ポート」のタイミングなどを高精度化することが可能です。

＊

「水晶振動子」を回路基板に実装して使う場合、「周波数精度」「安定性」「温度依存性」を評価しながら調整する必要があるため、電子回路シミュレータだけで設計できるものではありません。

しかし、「発振回路」は実測が難しい回路なので、試作の前に電子回路シミュレータを用いて、各部品定数に対する回路特性の変化を把握しておきます。

「LTspice」には、「水晶振動子」のシンボルが用意されていますが、モデル・パラメータは設定されていません。

「水晶振動子」のメーカーに依頼してシミュレーション・モデルを入手できる場合もあるようですが、通常は、ネットワーク・アナライザやインピーダン・スアナライザという計測器を用いて、「4素子」または「6素子」からなる等価回路のパラメータを求めることになります。

ここでは、「8MHz水晶振動子」の概算パラメータ値を用いて、シミュレーションを行ないます。

より正確な実測値を使いたい人は、各種「水晶振動子」の実測結果が公開されている、「Radio Experimenter's Blog」のデータを参照してください。

※21 マイコンには、通常、簡易型の「内部クロック発生回路」が搭載されているので、クロックを外部から与える必要がない。しかし、数％程度の周波数誤差があり、通信エラーや時間計測エラーの原因となる。

このため、外部に「水晶振動子」などの外付け部品を付けて、高精度なクロックを発生させる機能が用意されている。

※22 マイコンなどに内蔵されている、時間を計測する回路ブロック。

最初に「水晶振動子」のインピーダンスの周波数特性を調べてみます。

図5-31のように回路図を入力してください。電流源「I1」の向きには注意です。

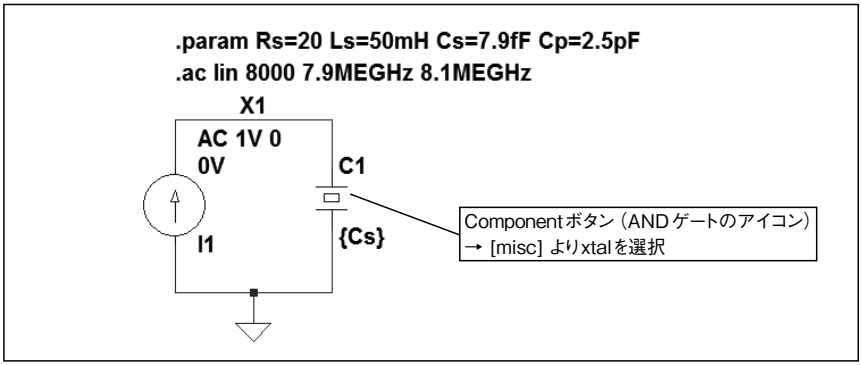

図5-31　「水晶振動子」のインピーダンス測定回路

次に、「水晶振動子」のシンボルを右クリックして、**図5-32**のように変数を設定してください。

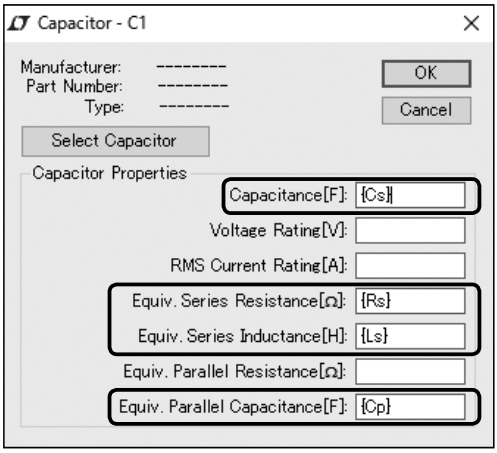

図5-32　「水晶振動子」の設定

変数には、「.param」ディレクティブを用いて、値を代入することにします。

このモデルは「インダクタ」「キャパシタ」「抵抗」からなる等価回路モデルです。

各変数と「水晶等価回路」の対応関係は、**図5-33**に示します。

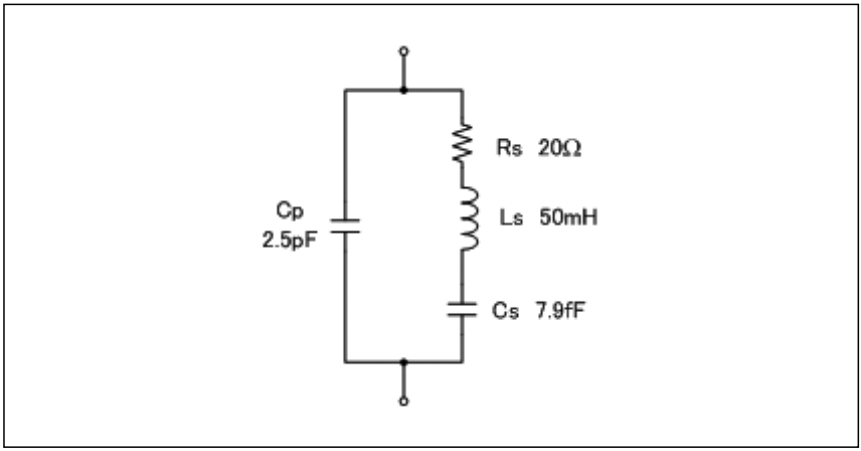

図5-33　「水晶振動子」の4素子等価回路

電流源「I1」は、「DC value = 0V」「AC Amplitude = 1V」「AC Phase = 0」としてください。

このように設定することによって、「V(X1)」が「xtal(C1)」のインピーダンス（インダクタやキャパシタを含む回路の抵抗）の値と同じになります。

「AC Amplitude、AC Phase」は好きな値にして、インピーダンスの定義どおり、「V(X1)/I(X1)」によってインピーダンスを求めてもかまいません。

*

シミュレーションを実行して「V(X1)」をグラフ表示すると、**図5-34**のようになります。

「V(X1)」の振幅は、左側縦軸の目盛り数字をクリックして「Left Vertical Axis」フォームを開き、「Representation」欄で「Logarithmic」を選んでいます（デフォルトでは「Decibel」）。

2つの共振点「fs」(共振：インピーダンスの極小)、fp (反共振：インピーダンスの極大)が、近い周波数に並んで観測されます。

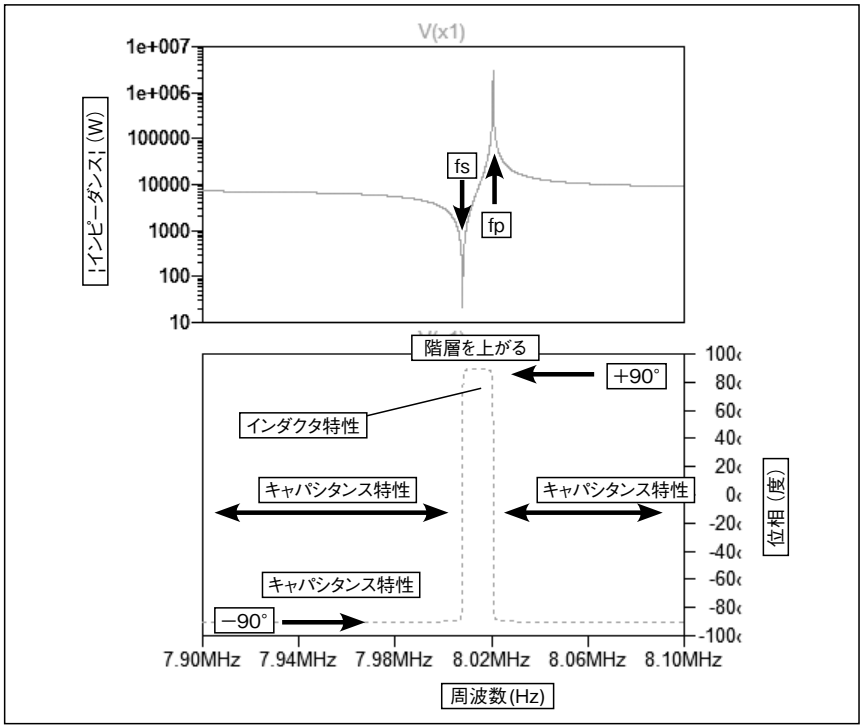

図5-34　「水晶振動子」のインピーダンス

周波数に対する傾きは、低周波側から、「負-正-負」となっています。

また、位相が「−90°」になっているところと、「+90°」のところがあります。

周波数が高くなるほどインピーダンスが増加し、電圧と電流の位相差が「+90°」になるのは、インダクタの性質です。

逆に、周波数が高くなるとインピーダンスが下がり、電圧と電流の位相差が「-90°」になるのは、キャパシタの性質です。

「水晶振動子」は、非常に狭い「fp-fs」間の周波数範囲のときだけインダクタンスと等価で、それ以外の周波数ではキャパシタンスと等価であることが分かります。

■「LC発振回路」の原理

「水晶発振回路」の動作を理解するためには、「LC共振」と「LC発振回路」の原理を知っておく必要があるため、以下で説明します（　知識のある人は、飛ばしてかまいません）。

大学や高専で、工学や理学を専攻した人は、「LC共振回路」の数学的取り扱いに慣れていると思いますが、ここでは、物理的な観点から復習しておきます。

*

抵抗では、「電圧波形」と「電流波形」は同じ形です。

しかし、「インダクタ」と「キャパシタ」は、「電圧波形」と「電流波形」の位相が90度ズレてしまいます。

具体的には、「インダクタ」にsin波の電圧をかけると「−cos波」の電流が流れ、「キャパシタ」にsin波の電圧をかけると、「+cos波」の電流が流れます。

つまり、「インダクタ」と「キャパシタ」に同じ交流電圧を加えて、両者の電流を比較すると逆向きになっています。このため、「インダクタ」と「キャパシタ」を並列に接続すると、お互いに電流を打ち消してしまいます。

「インダクタ」と「キャパシタ」のインピーダンス（交流に対する抵抗値）は周波数によって変化するため、「インダクタ」と「キャパシタ」の電流振幅がちょうど一致するような周波数では、電圧源から両者に流れる電流が完全にキャンセルされて常にゼロになります。

言葉では分かりにくいので、この状況を**図5-35**に示します。

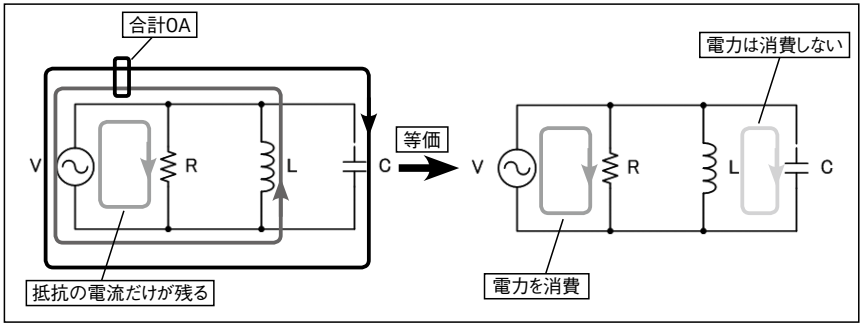

図5-35　「並列共振回路」の共振電流

図5-35のように抵抗「R」、インダクタ「L」、キャパシタ「C」を並列接続した場合、「L」と「C」の電流成分が同じ大きさで逆向きになるような周波数があります。

この周波数では、両方の電流を合計するとゼロになり、交流電圧を加えたのに交流電流が流れないため、「LやCのインピーダンス＝交流電圧/交流電流＝∞」のように見えます。

しかし、実際には、インダクタとキャパシタには電流が流れないのではなく、図5-35右側の図のようにインダクタ「L」とキャパシタ「C」の間では、電流が打ち消されずに流れています。

この状況では、電力を消費する抵抗「R」と電圧源「V」からなる回路部分と、電力を消費しないインダクタ「L」とキャパシタ「C」の回路部分が切り離されているのと同じです。この状態を「反共振」と呼びます。

また、「抵抗」「インダクタ」「キャパシタ」を直列に接続した場合は、インダクタとキャパシタの電圧が打ち消されて、抵抗だけに電圧が加わるように見える周波数があります。この状態を「共振」（または「直列共振」）と呼びます。

「水晶振動子」は、「共振」と「反共振」の両方の特性をもっているので、図5-33のように、「インダクタ」と「キャパシタ」が、直列および並列接続された構造の等価回路で表わすことができます。

＊

「LC発振」を理解するためには、この回路に対するもうひとつの物理的解釈が必要になります。

「抵抗」は、電力（電気エネルギー）を消費して、「ジュール熱」（熱エネルギー）を発生しますが、理想的な「インダクタ」や「キャパシタ」は、電力を消費しません。

「インダクタ」や「キャパシタ」は、「電気エネルギー」を一時的に内部の空間に蓄えるデバイスです。

「インダクタ」は、交流電源から供給された電気エネルギーを「磁気エネルギー」に変換して蓄え、「キャパシタ」は電気エネルギーを「静電エネルギー」に変換して蓄えます。

しかし、交流なので、蓄えられたエネルギーは再び放出されることになります。

「インダクタとキャパシタの共振（または反共振）周波数」では、「インダクタ」と「キャパシタ」で、エネルギーの蓄積と放出の位相がちょうど逆になっているので、互いに蓄えたエネルギーを、送ったり送られたりとキャッチボールをしていると考えられます。

この状況は、「ばね」につけた「重り」が、「運動エネルギー」と「ばねの弾性エネルギー」の変換を繰り返して、振動するのとよく似ています。

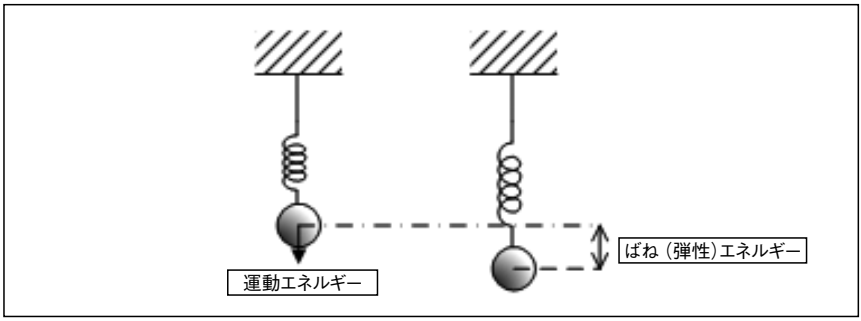

図5-36　「ばね」と「重り」の運動のアナロジー（類推）

「共振」が、「インダクタ」と「キャパシタ」の間だけで起こっているのであれば、一度、「共振状態」になってしまえば、電圧源「V」は必要ないのではないかと思われます。

しかし、残念ながら電圧源「V」を切り離すと、振動は短時間のうちに止まります。

「ばね」に付けた「重り」の振動の場合、外から力を加えずに放っておけば、「空気抵抗」や「ばね材料の発熱」によってエネルギーが少しずつ失われ、振幅が弱まっていきますが、「LC共振」でも、配線の抵抗による「ジュール熱」の発生や、電磁界の外部への放出によって、エネルギーが回路から逃げて、電流は固有振動をしながら減衰していきます。

「LC発振回路」は、電圧源の代わりに、「増幅回路」を使って電流振動の減衰ぶんを増幅によって補うようにした回路です。

「LC共振回路」では、ノード（配線）が2個しかありませんが、増幅回路を接続するためには、「増幅回路の入力端子」（振動のモニタ）、「増幅回路の出力端子」（振動の補給点）、グランドの3つのノード（配線）が必要になるので、「キャパシタ」または「インダクタ」を1個追加し、3端子構成にします。

＊

「インダクタ1個＋キャパシタ2個」で構成したものは、「コルピッツ発振回路[※23]」(Colpitts oscillator) と呼ばれています。

「コルピッツ発振回路」には、**図5-37**のように「非反転増幅回路」（利得が正）を使う方式と、「逆相増幅回路」（利得が負）を使う方式の2種類があります。

※23 逆に、「インダクタ2個＋キャパシタ1個」で構成されたものは、「ハートレー発振回路」(Hartley oscillator) と呼ばれる。

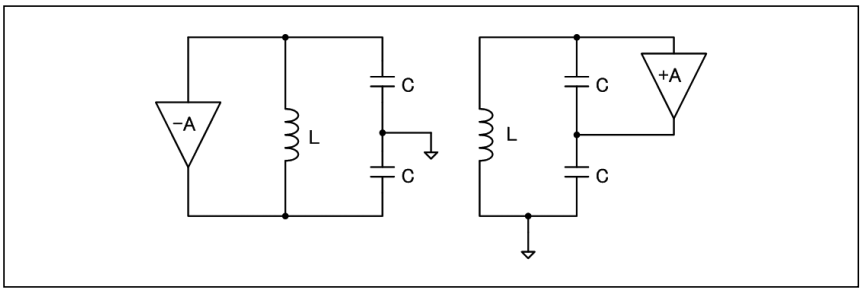

図5-37　「コルピッツ発振回路」の構成

微妙に「増幅回路」の接続ポイントが異なっていますが、元の振動と同じ位相で増幅した信号を戻すように接続されています。

どちらの回路も、「発振周波数」は次式で表わされます。

$$f_{co} = \frac{1}{2\pi}\sqrt{\frac{2}{\mathrm{LC}}}$$

「水晶発振回路」は、「コルピッツ発振回路」のインダクタの代わりに、「水晶振動子」のインダクタ特性を利用しています。

「水晶振動子」は、非常に狭い周波数範囲でインダクタと等価になるため、この周波数範囲で「発振」が起こり、「水晶振動子」の周波数特性によって周波数が決定されます。

「水晶振動子」は本当のインダクタではなく、インダクタと同じ電気的特性をもっているだけなので、「磁気エネルギー」を蓄えるわけではありません。

また、「水晶振動子」は、水晶の「圧電効果※24」によって、結晶内の「Si-O結合」の機械的変形（歪み）を発生させてエネルギーを蓄えることが報告されています。

通常のインダクタとは異なり、「コイル」（銅線を巻いたもの）ではないので、「寄生抵抗」をほとんど含まない理想的な共振器として動作します。

※24 電圧を加えると電気分極により結晶が歪み、結晶が歪むと電気分極により電圧が発生する現象。

■「インバータ」を用いた増幅回路の設計

「水晶発振回路」用の増幅回路としては、「CMOS※25インバータ」(NOT回路)が利用されています。

たとえば、マイコンなどの「水晶振動子」の接続端子にも、「インバータ」が内蔵されています。

「CMOSインバータ」は論理ゲートのひとつですが、アナログ回路としては「プッシュプル増幅回路※26」に相当します。

「LTspice」には「インバータ」などの論理ゲートも部品として用意されていますが、デジタル回路用のモデルになっていて、電圧利得などを設定できません。

そこで、「CMOSインバータ」の内部回路を自分で作ってみます。

「汎用ロジックIC※27」のシミュレーション用モデルを提供しているメーカーもあるようですが、自分で作ったほうが内部パラメータの変更が容易です。

●「MOSFETモデル・パラメータ」の入手

「CMOSインバータ」を作るには、「コンプリメンタリー特性※28」をもつ2種類の「MOSFET」が必要です。

しかし、単体部品として「コンプリメンタリー特性」をもつ「MOSFET」のペアは販売されていないので、半導体メーカーが製造サービスの利用者に提供する「MOSFET」のモデルを使います。

※25「Complementary Metal-Oxide-Semiconductor」の略で、「MOSFET」を基本素子とする集積回路の標準的な製造法の名称。この後に、微細加工寸法の値を記載し、製造技術の世代を示すことが多い。例：CMOS 65nm。

※26 正負逆の特性をもった2個トランジスタを組み合わせて、信号の正方向増幅と負方向増幅を、各トランジスタに行なわせる回路方式。上手く設計すれば、高い電力効率と低歪率を両立できる。

※27 論理ゲート単体の小規模な集積回路。各種論理機能がシリーズとして販売されている。

※28 直流電圧と電流の印可方向を逆にしたときに、等しい特性を示す一組のトランジスタ。たとえば、「OFF-ON」させるのに必要な閾値電圧の絶対値が等しく、正負が逆になっている必要がある。

このモデルは、「MOSFET」から設計するユーザーのためのものなので、設計規則※29の範囲内で、形状や寸法を自由に変更することが可能です。

ただし、実際の製造ラインで測定された精密なデータに基づく「MOSFETモデル・パラメータ」は、半導体メーカーと契約をしないと入手できないため、教育用や技術動向予測のために公開されている「モデル・パラメータ」を利用します。

製造ラインによって「MOSFET」の特性が多少異なりますが、微細加工寸法（製造技術の世代）が同じであれば、大きな違いはありません。

＜ダウンロード先の例＞

・CMOSedu.com（http://cmosedu.com/cmos1/cmosedu_models.txt）
　CMOS 1000nm ～50nmに対応。

・アリゾナ州立大学（http://ptm.asu.edu/latest.html）
　CMOS 180nm ～7nmに対応。

どの製造プロセスを選べばいいのか迷ったら、「電源電圧」で選んでください。

もし、使いたいと思っているマイコンなどの製造プロセスが判明したとしても、「水晶発振回路」は、プロセッサなどの内部回路（コア）とは特性の異なる「IO」（入出力専用の回路）の一種として組み込まれています。

また、「汎用ロジックIC」を使う場合は、内部回路の保護や高性能化のために、インバータ以外の回路が組み込まれているので、正確には使う半導体内の「MOSFET」の特性を特定できません。

しかし、電源電圧によってMOSFETの「閾値電圧」（オンオフが切り替わる電圧）や「Kp」（基準寸法における電流を決定する係数）の値※30が大体決まりますので、電源電圧から製造技術を選択しても、大きく外れることはないと期待できます。

※29 半導体メーカーが、歩留まりや性能の保証を行なうために設けている、各種構造の寸法に関するリール。
※30 これらの値は、電源電圧と密接に関係する「ゲート酸化膜の厚さ」と「不純物量」でほぼ決定され、半導体メーカが正確にコントロールしている。

表5-3　シミュレーション用製造技術の選択

VDD(V)	製造技術
5.0	CMOS 600nm ～ CMOS 1000nm
3.3	CMOS 350nm ～ CMOS 600nm
2.5	CMOS 250nm
1.8	CMOS 180nm

●「集積回路内部のモデル作成」と「MOSFETの最適化」

本書では、「CMOSedu.com」で配布されている「cmosedu_models.txt」というファイルに含まれている、「N_1u(n-ch※31 MOSFET)」と「P_1u(p-ch MOSFET))の「コンプリメンタリー・ペア」を使います。

ダウンロードしたファイルは、回路図データを置くフォルダの中に保存してください。

「水晶発振回路」の消費電力削減の要請から、「発振回路」をスリープ（発振停止）させるために、「インバータ」ではなく「NAND」を使っていることが多いので、ここでも「NANDゲート」を利用します。

「NAND」の2つの入力のうち片方が「HIGH」になっているとき、もう一方の入力が「インバータ」として動作します（**表5-4**）。

※31「n-ch」は「エヌチャネル」、「p-ch」は「ピーチャネル」と呼ぶ。直流電圧、直流電流の向きが逆で動作するコンプリメンタリーのペアである。

表5-4　「NANDゲート」の動作

A	B	Z	動作モード
0	0	1	発振停止(出力=HIGH)
1	0	1	
0	1	1	インバータ動作
1	1	0	(出力＝NOT (A))

図5-38のような「NAND回路」を入力してください。

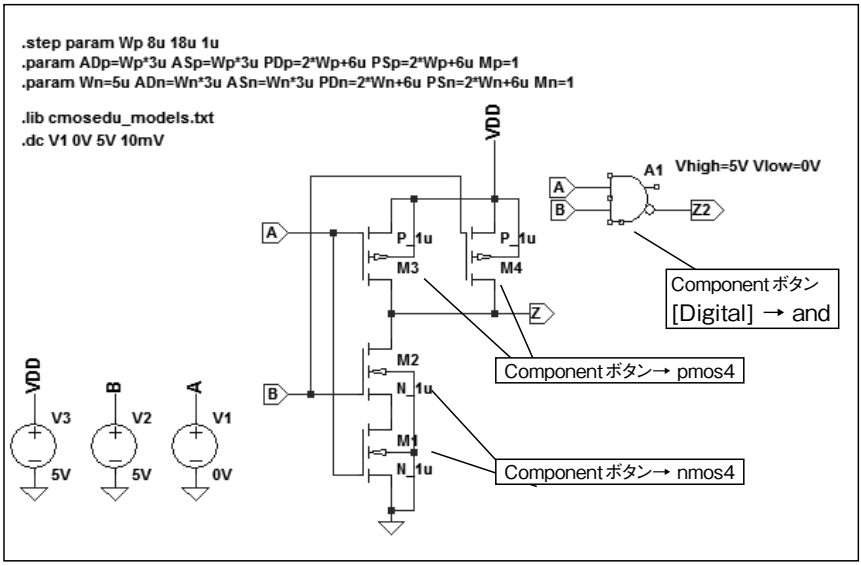

図5-38　NAND回路

集積回路用の「MOSFET」シンボルは、「nmos4(n-ch MOSFET)」と「pmos4(p-ch MOSFET)」です。

右上の「NAND」は、「LTspice」に付属している「NANDゲート」のモデルです。比較のため、こちらも配置しておいてください。

「LTspice」付属の論理ゲートは、「AND」と「NAND」が共通のシンボルになっているので、「and」と名前の付いている論理ゲートをを呼び出してください。

集積回路用の「MOSFET」では、デバイスの型番の代わりに各部の寸法を指定し、この寸法に基づいてデバイス特性が算出されるようになっています。

入力しない寸法パラメータがある場合はデフォルト値が使われ、実際の特性とまったく異なってしまうので、注意してください。

「MOSFET」シンボルを右クリックすると、「Monolithic MOSFET」というフォームが表示されるので、**図5-39**のように入力してください※32。

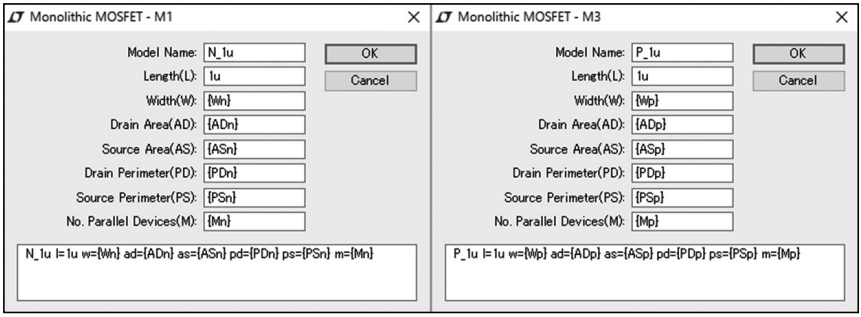

図5-39　「MOSFET」の設定（左：n-ch、右：p-ch）

「Model Name」は、半導体メーカーが定義したモデル名です。モデルファイルをテキスト・エディタで開いて、「.MODEL」ディレクティブの次に書かれている文字列で確認することができます。

「cmosedu_models.txt」のファイルには4種類のモデル名とパラメータのセットが記述されていますが、このうち「N_1u」は、「CMOS 1000nm」(= CMOS 1um)の製造技術で作られる「n-ch MOSFET」、「P_1u」は「N_1u」とコンプリメンタリーな「p-ch MOSFET」です。

※32「MOSFET」の寸法入力は面倒だが、同じ値を使うことが多い。「n-ch」と「p-ch」各1個だけ設定して、シンボルをコピー＆ペーストすれば、寸法も一緒にコピーされる。

表5-5　「MOSFET」の設定パラメータ

パラメータ	名　称	意　味
Length	ゲート長	ドレイン-ソース間の距離
Width	ゲート幅	ドレインとソースの幅
No. Parallel Devices	フィンガー数	MOSFETの並列接続数

図5-39で設定したパラメータのうち、**表5-5**の3つが「MOSFET特性」を決定する最も重要なパラメータです。

その他にもパラメータがありますが、ここでは説明を省略します。詳しくは、文献9の「MOSFETモデル・パラメータ」を参照してください。

「Length」が小さいほど、「gm※33」や動作速度が大きくなり高性能ですが、「製造ばらつき」の影響も大きくなります。

デジタル回路では、製造技術の最小寸法に設定するのが普通です。ここでは、5V電源の論理ゲートとして充分な性能を出せるゲート長「1um」としておきます。

「Width」は、ドレイン-ソース間の電流を流す能力を決定しています。

「Width」を大きくするほど大きな電流を流すことができ、「gm」も大きくなりますが、「Width」を大きくしすぎると、動作速度や雑音性能が下がるため、大体「Length」の「10～20倍」ぐらいまでの値にします。

大きな電流を流したい場合は、「Width」を増やさずに、「No. Parallel Devices」を増やします。

「No. Parallel Devices」は並列接続する数なので、これを「2」に設定すれば、電流がちょうど2倍流れます。

*

次に、ラベル「A1」と表示されている「NANDゲート」のシンボルを、[CTRL]+右クリックして、**図5-40**のように「Value」を設定してください。

「Vhigh」は論理レベルHIGHの電圧、「Vlow」は論理レベルLOWの電圧です。

※33「Transconductance」という 入力電圧変化と出力電流変化の比を表わす。「MOSFET」の増幅能力の指標となっている。

ゲートの閾値は、「Vhigh」と「Vlow」の平均値になります。

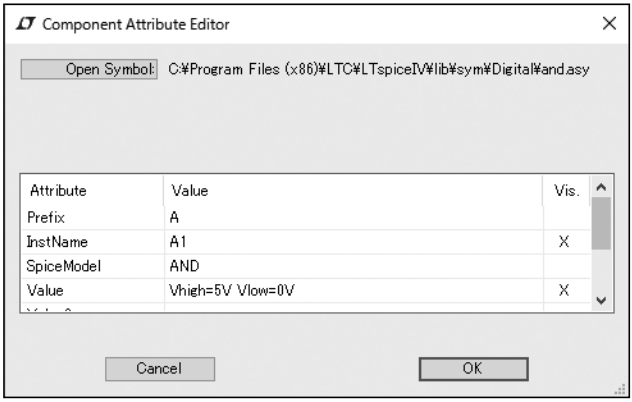

図5-40　「論理ゲート」の設定

この他にも、いくつかのパラメータがあるので、「LTspice」付属の論理ゲートを使う場合は、メニューの「Help」→「Help Topics」から、「A. Special Functions」を参照してください。

*

図5-41に、シミュレーションの実行結果を示します。

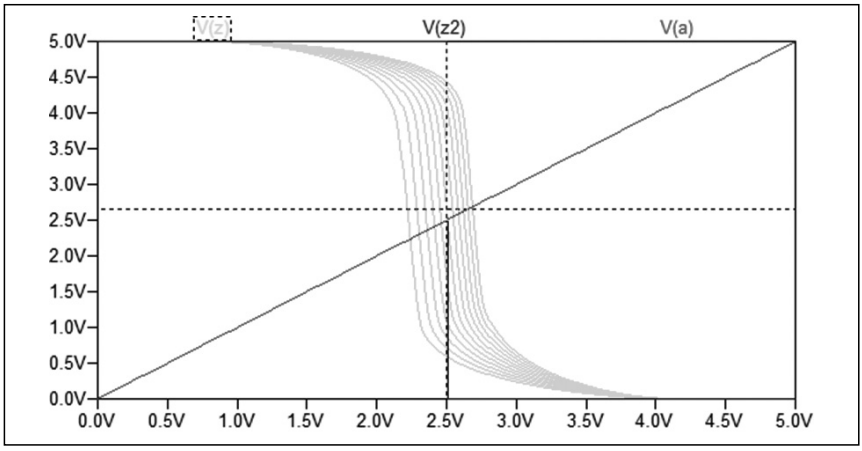

図5-41　「インバータ」の直流伝達特性

「V(Z)」「V(Z2)」「V(A)」をプローブしてください。
グラフは、横軸を「入力電圧」、縦軸を「出力電圧」とした「直流伝達特性」を表わしています。

「V(A)=2.5V」のところで、NANDゲート「A1」の出力「V (Z2)」が急にHIGHからLOWレベルに変化します。
変化点付近を拡大してみると、「10mV」ぐらいの入力変化で、出力が「5V」変化しているので、「電圧利得 = 5V/10mV = 500倍」ぐらいになります。
実際の「論理ゲート」では、このような大きな電圧利得にはなりません。
一方、「MOSFET」で構成したNANDゲートの「V(Z)」の特性は、滑らかな曲線を描いており、最大で20倍程度の電圧利得をもっています。

「Wp」は「p-ch MOSFET」の「Width」であり、この長さが大きいほど、電流が多く流れます。
このため、「n-ch MOSFET」と「p-ch MOSFET」を最適な電流バランスの特性にするには、「Wp」や「Wn」の調整が必要です。
ここでは、「.step」ディレクティブで変数「Wp」を変化させて、特性の変化を調べます。

「入力電圧 = 出力電圧」となる点が、「VDD/2」になるように調整します。
この条件は、論理ゲートの「雑音耐性」を最大にする条件に相当します。
以下の手順で、最適な「Wp」の値を探してみてください。

---

[1] グラフ上部のトレースラベル「V(Z)」をクリック。
[2] グラフ・カーソルを、「横軸 = 2.5V」の位置に移動（マウスのドラッグで正確に移動させるのが難しい場合は、左右矢印キーで移動）。
[3] 上下矢印キーで、「縦軸 = 2.5V」に最も近いトレースにカーソルを移動。
[4] グラフ・カーソルの上を右クリックして、「Cursor Step Information」フォームで、「Wp」の値を読む。

「Wp = 13um」ぐらいで最適となると思います。

「.param」ディレクティブにより、「Wn = 5um」に設定したので、「n-ch MOSFET」と「p-ch MOSFET」のサイズ比は、「Wp/Wn = 2.6」となります。
「Wp/Wn」は、通常「2」～「4」ぐらいの値になります。

■回路の階層化 (hierarchy)

プログラミングでは、何度も実行する必要がある処理内容を「サブルーチン」として定義し、メインルーチンの中で呼び出すことで、理解しやすく記述効率の良いプログラムを作ることが可能になっています。
同様に電子回路の設計でも、複数箇所で使用する汎用性のある回路は、「サブサーキット」として定義し、「サブサーキット」に割り当てたシンボルを使って、より複雑な回路を簡潔に表わす手法が用いられています。

このような方法は、「階層設計」と呼ばれています。
「LTspice」などの電子回路シミュレータでは、トランジスタを用いて設計した実際の回路の他、等価回路で表わされたマクロモデルなども「サブサーキット」として使うことができます。

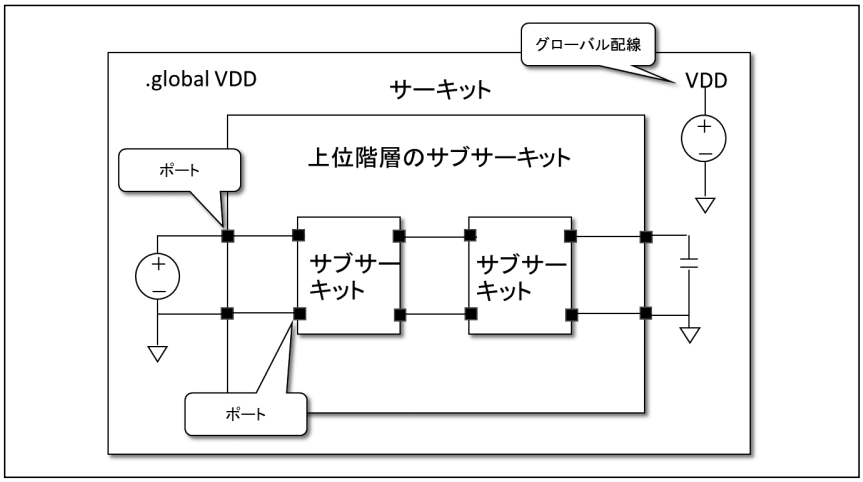

図5-42　「階層化」された回路の構造

●「サブサーキット」の作成

設計した「NAND回路」を「サブサーキット」として利用するため、「NAND」の回路図ファイル（拡張子asc）をコピー[※34]して、図5-43のように編集してください。

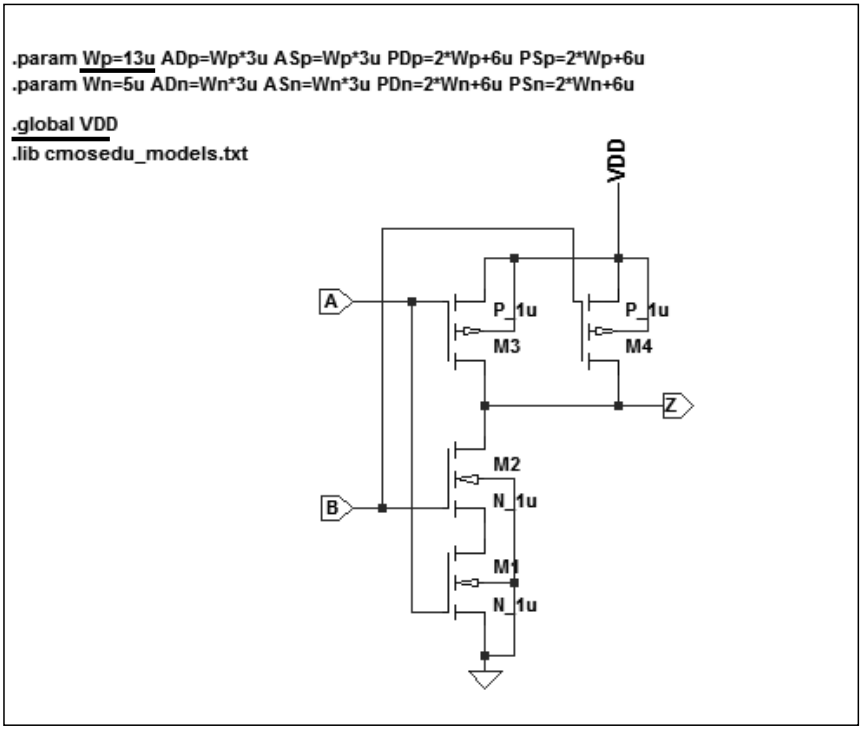

図5-43　NANDの「サブサーキット」

編集箇所は、表5-6のとおりです。
ここでは、「サブサーキット」のファイル名は、「NAND.asc」とします。

※34「LTspice」のツールバー、またはメニューのコピーコマンドは、回路図シート間でも有効。新しい回路図シートに、既存の回路図の必要部分をコピー＆ペーストしてもよい。

表5-6　「NAND回路」の編集箇所リスト

編集箇所	編集内容
論理ゲートA1	シンボルおよび入出力ポートを含む回路の削除。
V1、V2、V3	上位階層から電源を供給するので削除。
.stepディレクティブ	「Wp」の最適値を「.param」で与えるので削除。
.param Wp=13u	「Wp」の最適値の入力を追加。
.param Mp=1	「Mp」は上位階層で指定するため削除。
.param Mn=1	「Mn」は上位階層で指定するため削除。
.dcディレクティブ	解析命令は上位階層で指定するため削除。
.global VDD	電源配線「VDD」をグローバル配線として定義。

「.global」は、指定した配線をグローバル配線として扱うためのディレクティブです。

「サブサーキット」内の配線は「ローカル配線」として定義され、上位階層の回路内の配線に対して「サブサーキット」と同じ配線名を与えても、「サブサーキット」の内部とは接続されません。

したがって、原則として上位階層と配線を接続するためには、「ポート」を通して接続する必要があります（「ポート」の作り方は後で説明します）。

しかし、電源配線のように回路全体で共通の配線を、いちいち「ポート」から接続するのは面倒なので、「.global」で「グローバル配線」に指定しておいて上位階層で同じ名前を付ければ、自動的に「サブサーキット」内の同名の配線と接続が行なわれます。

なお、「グランド」の配線は、自動的に「グローバル配線」として扱われるので、「.global」を指定する必要はありません。

●シンボルの作成

次の手順で、「サブサーキット」のシンボルを作ります。

---

[1] メニューから「File」→「New Symbol」を選択して、「シンボル・エディタ」を起動。

[2] メニューから「Draw」→「Line」、「Draw」→「Arc」などを選び、シンボルを描く（面倒なら、「Draw」→「Rect」で四角形のシンボルを描いても問題ない）。

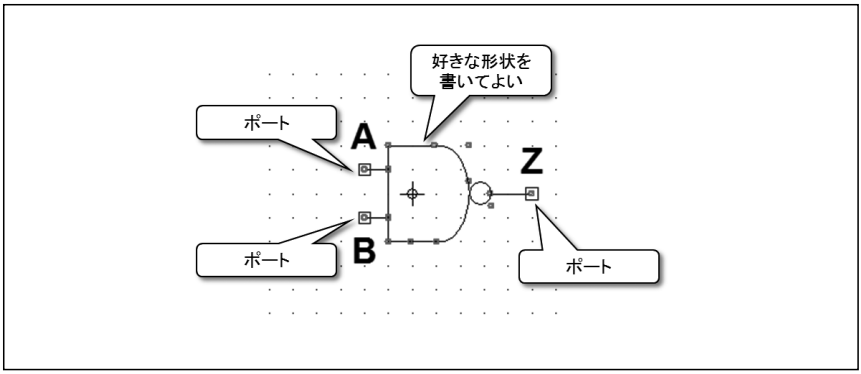

図5-44　シンボル作成画面

[3] メニューから「Edit」→「Add Pin/Port」を選択して、「ポート」を作成。
この際、「Label」は、「サブサーキット」の配線名に合わせる。
また、「TOP」「BOTTOM」「LEFT」「RIGHT」のどれかを選ぶと、ポート名が表示され、「NONE」を選ぶと非表示になる。

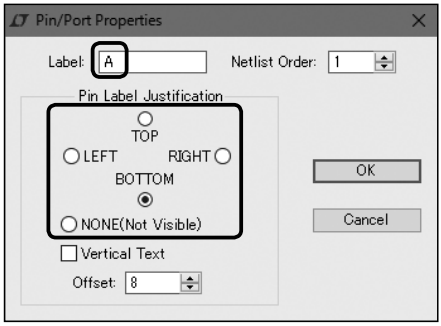

図5-45　「ポート」の設定フォーム

[4] シンボルが完成したら、「NAND.asy」（シンボルファイル）に保存。
保存の際は、「サブサーキット」のファイル名と一致させる必要がある（拡張子のみ異なる）。
また、保存フォルダは、「サブサーキット」のファイルと同じ場所にすること。

●「サブサーキット」の呼び出し

新規に、**図5-46**のような回路図を作ります。

「NAND」のシンボルは、「LTspice」に付属するものではなく、自作したものを呼び出します。

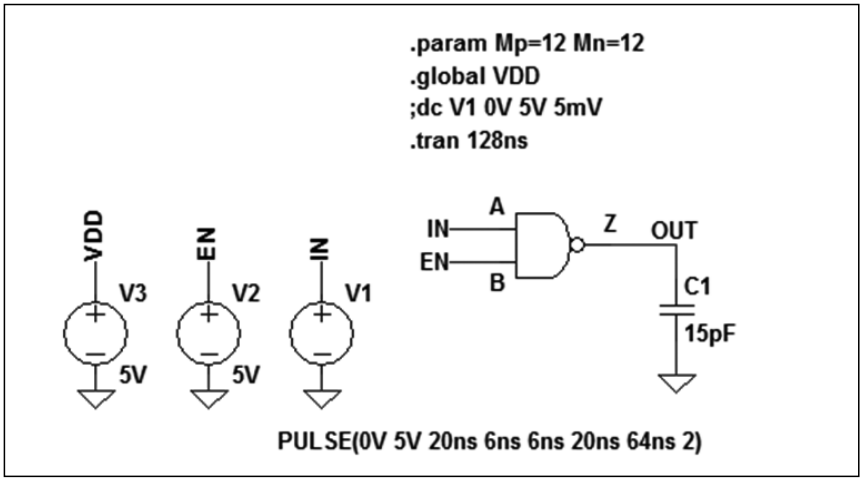

図5-46　「NAND」呼び出し側の回路図

[1] ツールバーから、「Component」ボタン（ANDゲートのアイコン）をクリック。

[2] 「Select Component Symbol」フォームが表示されるので、「Top Directory」プルダウンリストから、「NAND」のシンボルファイルを保存したディレクトリを選択。

[3] シンボルファイル名のリストが表示されるので、「NAND」を選択し、回路図エディタ上に配置して、OKボタンをクリック。

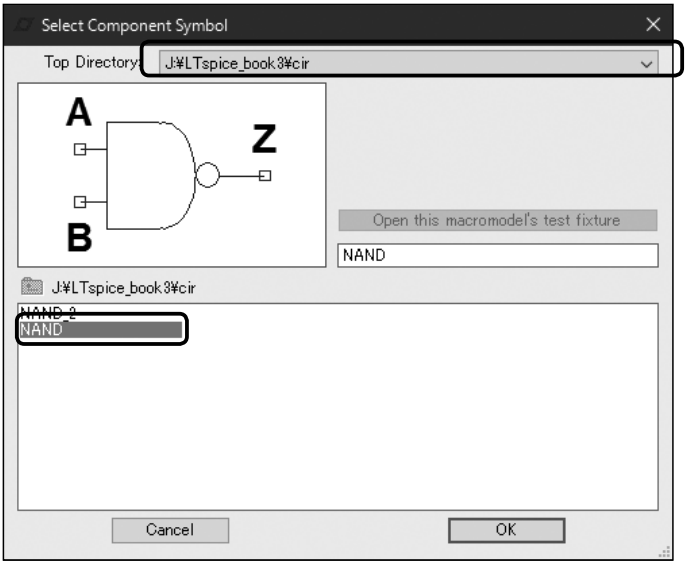

図5-47　「サブサーキット」の呼び出し

「.global」ディレクティブは、「NANDサブサーキット」の内部に書かれているので、「NAND」の呼び出し側で再度書く必要はありませんが、「VDD」が「グローバル配線」であることが、分かりやすいように、呼び出し側の回路図にも記載されています。

シンボルファイル(拡張子 asy)を呼び出すと、同じフォルダ（ディレクトリ)に置かれている、同じファイル名の回路図ファイル(拡張子 asc)が「サブサーキット」化されて、「ネットリスト」に読み込まれます。

次の手順で、確認してみましょう。

---

[1] 回路図エディタのメニューから、「View」→「SPICE Nelist」を選択。
[2] 新しく表示フォームが起動し、「ネットリスト」が表示される。

「.subckt」～「.ends」の部分が、「NAND」の内部回路の「ネットリスト」(回路の接続情報)です。

「M1」～「M4」の、4個の「MOSFET」が使われていることが分かります。

「MOSFET」に流せる電流が大きいほど、「NAND」の出力端子から取り出せる電流も大きくなります。

出力電流を大きくすることによって、出力端子につながれた回路や配線の「キャパシタ」(負荷容量)への充電や放電が速くなり、「伝搬遅延時間※35」が短くなります。

「直流伝達特性」は、MOSFETの「Width」の値によって最適化ずみなので、「n-ch MOSFET」と「p-ch MOSFET」の電流比は変えずに、電流の絶対値を増加させるため、並列接続数「Mp」と「Mn」を増やします。

ただし、「Mp」と「Mn」は同じ値の整数値にしなければいけません。

ここでは、「.param」ディレクティブによって、どちらも「12」に設定しています(この値は、シミュレーションの結果を見て調整する必要があります)。

●「伝搬遅延時間」の測定

図5-46のシミュレーションを実行して、「伝搬遅延時間」を調べてみます。

[1]「V(IN)」と「V(OUT)」を電圧プローブして、入力波形と出力波形をグラフ・ウインドウに表示。

[2]グラフ・ウインドウ上部のトレースラベル「V(IN)」を2回クリック(ダブルクリックではない)して、「グラフ・カーソルの1と2」を表示する。

[3]「グラフ・カーソル1」を、「V(IN)」の立ち上がり波形が「2.5V」を通過する点に移動(少しズレてもかまわない)。

[4]「グラフ・カーソル2」をクリックしてから、グラフ・ウインドウ上部のトレースラベル「V(OUT)」をクリックし、グラフ・カーソルを「V(OUT)」の波形上に移動させる。

[5]「V(OUT)」のグラフ・カーソルを、立ち下がり波形が「2.5V」を通過する点に移動させる。

※35 入力の変化点から出力の変化点までの時間。「CMOSロジック」の場合は、「入力波形がVDD/2を通過する点」から「出力波形がVDD/2を通過する点」までの時間で定義される。

[6] カーソル値ウインドウの「Diff - Horz」欄で時間を読み取る。

図5-48の時間差は、出力の立ち下がり波形（HighからLow）の遅延時間「tpdHL」を表わしています。

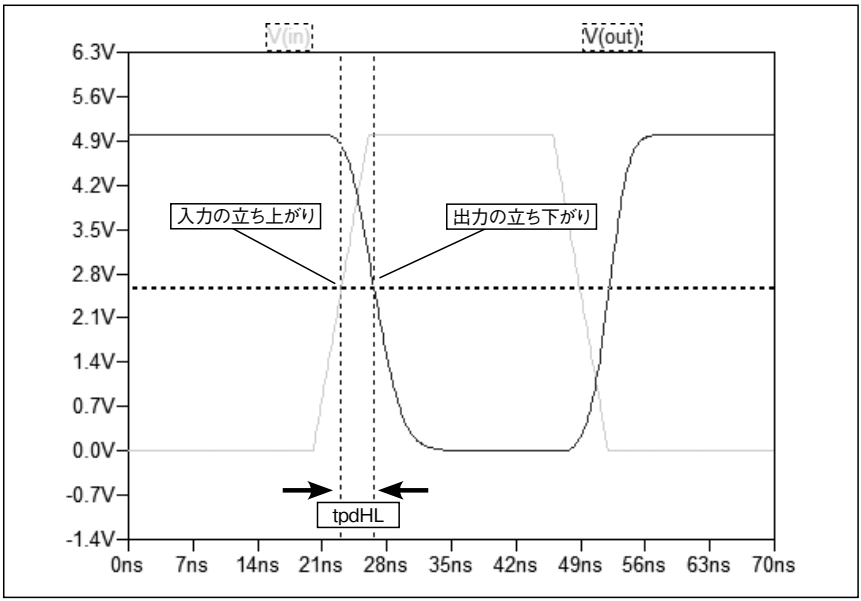

図5-48　「伝搬遅延時間」の測定

出力の立ち上がり波形の遅延時間「tpdLH」も測定してみてください。

「p-ch MOSFET」と「n-ch MOSFET」の「Width」が最適化されていれば、「tpdHL」と「tpdLH」はほぼ等しい値になるはずです。

例として、汎用CMOSロジック「74HCUシリーズ※36」の「伝搬遅延時間」に合わせてみます。

※36 汎用CMOSロジックICのシリーズのうち、本節で作ったような論理回路単体で構成されている品種。最後の「U」は「Unbuffered」（論理回路単体）を指している。より高速なシリーズとして「74VHCU」「74AHCU」などがある。

データシートによると、電源電圧「VDD = 5V」、負荷容量「CL = 15pF」のとき、伝搬遅延時間「tpdHL」と「tpdLH = 5ns ～8ns」ぐらいのものが多いようです。

図5-49の結果では、やや「伝搬遅延時間」が短いので、MOSFETの並列接続数を減らして、「.param Mp=7、Mn=7」にしてみると、「伝搬遅延時間」は「5ns」ぐらいになります。

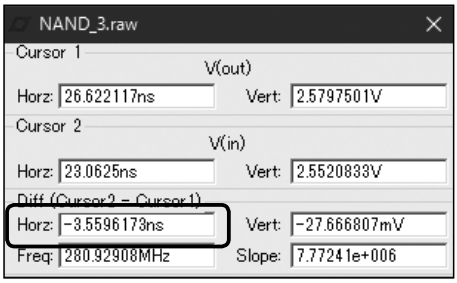

図5-49　2本のグラフカーソル間の差分

より高速なインバータやNANDを使うときは、「Mp」「Mn」を大きくします。論理ゲートのデータシートに「Open Loop Gain」が載っているようなら、値を確認してください。バッファのない1段のゲートの場合は、通常は「20倍」ぐらいです。

図5-41の「直流伝達特性」のシミュレーションで、「-d(V(Z))/d(V(A))」のグラフを作ると、最大値が電圧利得になります。

電圧利得は、MOSFETの「ゲート幅(Width)／ゲート長(Length)」の値を変えて少しは調整が可能です。

大幅に電圧利得を大きくしたい場合は多段構成にしますが、「増幅回路として安定動作させにくくなるので注意してください。

■「水晶発振回路」のシミュレーション

●発振開始特性

図5-50に、典型的な素子値を用いた「水晶発振回路」の例を示します。

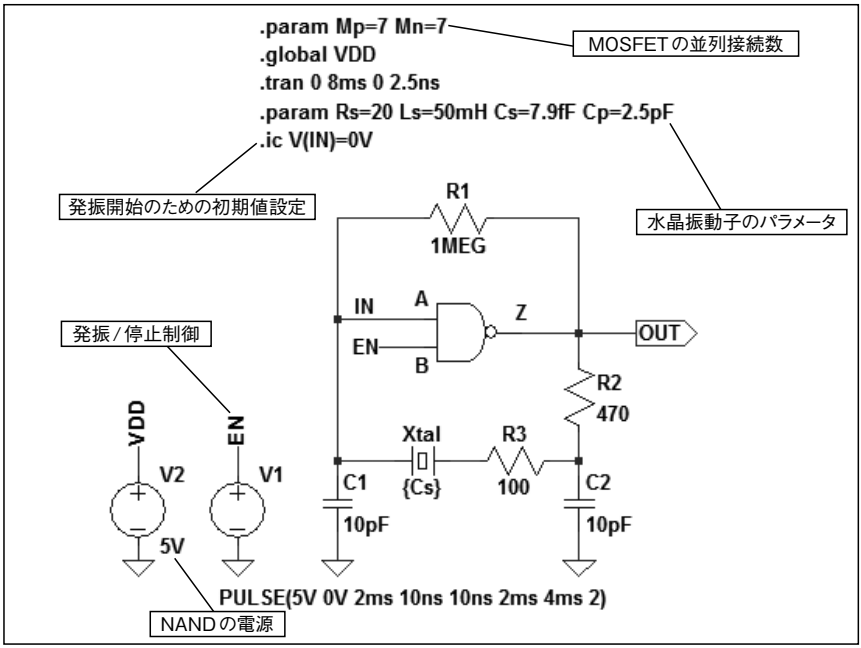

図5-50　水晶発振回路

「NAND」や「インバータ」のような論理ゲートを増幅回路として使うためには、増幅特性が現われる「VDD/2」付近に「直流バイアス電圧」を加える必要があります（**図5-47参照**）。

論理ゲートの入力端子と出力端子を接続し、「入力電圧 = 出力電圧」とすることによって、適切な「直流バイアス電圧」を印可できますが、入力と出力をショートしてしまうと増幅ができないので、抵抗「R1」を挿入しています。

抵抗「R1」は共振特性にも影響を与えるので、発振時に交流電流がほとんど流れないように大きめの抵抗値にしてあります。

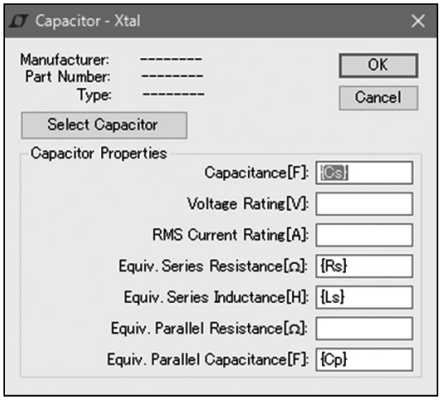

図5-51　水晶振動子の設定

「Xtal」「C1」「C2」が共振回路を形成して、「発振周波数」を決定します。

「R2」は増幅回路(ここではNAND)が、共振回路を励振する強さを調整する抵抗です。

「水晶振動子」を強く励振しすぎると、本来の共振周波数の奇数倍の周波数に現われる、「オーバートーン」と呼ばれる振動モードで共振するため、「水晶振動子」の共振周波数の3倍や5倍の周波数が出ないように「R2」調整します。

ただし、ここで使っている「水晶振動子」の等価回路モデルは、基本振動の共振特性のみをモデル化したものなので、本シミュレーションでは「オーバートーン」での発振は起こりません。

「R3」は、「水晶振動子」に流れる電流を調整する抵抗です。

データシートに記載されている「水晶振動子」の励振レベルを超えないように調整します(励振レベルが高すぎると、異常動作や故障の原因になります)。

「R2」「R3」の値は、増幅回路（ここではNAND）の入力電圧振幅にも影響します。

「V(IN)」が、論理ゲートのデータシートに記載されている入力電圧範囲内になるように「R2」「R3」を調整します。もっとも、回路シミュレータでは、故障はシミュレーションできないので、値を確認するだけです。

「水晶振動子」のパラメータは、**図5-33**と同じ値を使っています。

電圧源「V1」は、「発振モード」と「スリープモード」を切り替えるための制御信号です。**図5-52**のように設定してみてください。

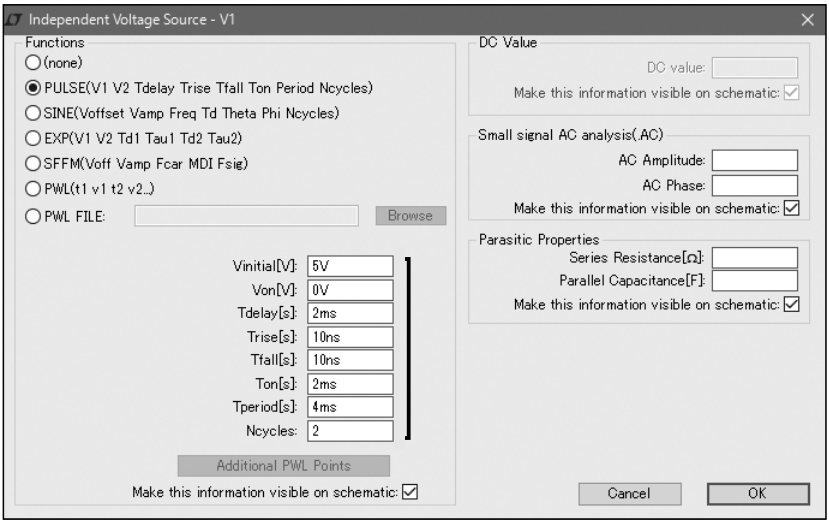

図5-52　「V1」の設定

シミュレーションを実行すると、**図5-53**のような波形が得られます（複数ペインのグラフ作成方法は、**2-2節**を参照）。

NANDの入力側は「sin波」に近い形となり、出力側は「矩形波」に近い形になります。また、「スリープモード」を解除後、振幅が安定するまでに「1ms」ぐらいかかることが分かります。

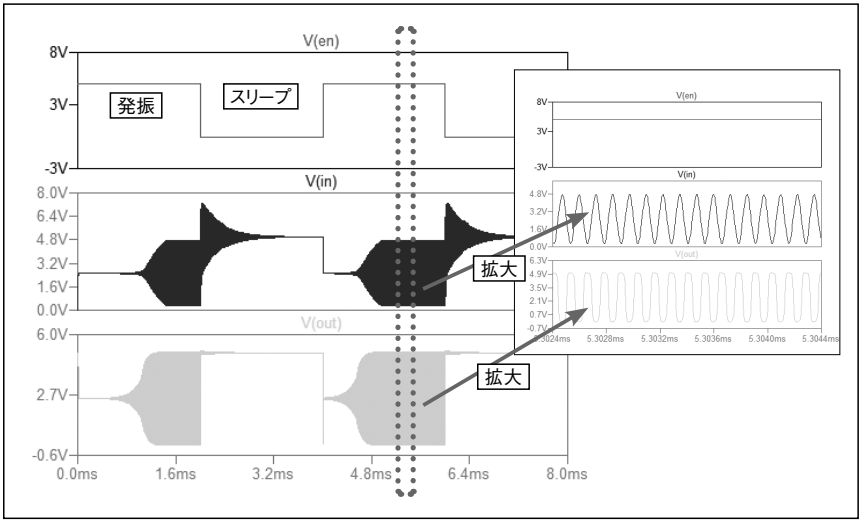

図5-53　「Transient解析」の結果

●「発振周波数」と「励振レベル」の測定

「水晶発振回路」の発振周波数は、「水晶振動子」の特性（規格）によって決定されると説明しましたが、実際には「C1」「C2」の容量値によって発振周波数が少し変化します。

そのため、正確な発振周波数が必要な場合は、「C1」「C2」を調整する必要があります。

＊

「C1」「C2」が発振周波数にどのように影響するのか、**図5-54**の回路でシミュレーションを行ない、「V(OUT)の振幅」「V(IN)の振幅」「励振レベルの振幅」「発振周波数」を調べてみます。

ただし、このシミュレーションには時間が掛かるので、急ぎたい場合はパラメータ・スイープのステップを大きくするなどして、時間を短縮してください。

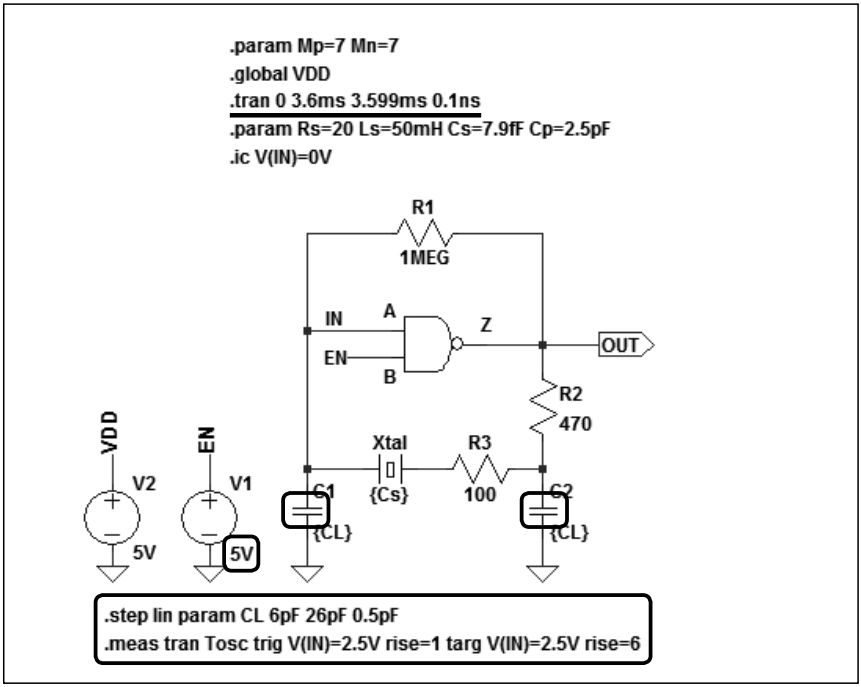

図5-54　「発振振幅」と「励振レベル」の測定

表5-7　「水晶発振回路」の変更箇所リスト

編集箇所	編集内容
V1	右クリックして、「Function欄 = (none)」を選択。 「DC Value = 5V」。
C1, C2	「10pF」を「{CL}」に変更。
.step	図5-54のようにCLのパラメータ・スイープを設定。
.meas	図5-54のように発振周期（5周期）の測定を設定。
.tran	「Stop Time = 3.6ms」「Time to Start Saving Data = 3.599ms」「Maximum Timestepを0.1ns」に設定。

「.tran」の設定では、定常常態になってから8周期（約1us）ぐらいを測定できるように、「Stop Time = 3.6ms」「Time to Start Saving Data = 3.599ms」と設定しています。

「Time to Start Saving Data」は、シミュレーション結果データの保存を開始する時刻なので、この設定によって「0.001ms = 1us」のデータが保存され、その波形に対して「.meas」による測定が行なわれます。

発振回路の「Transient解析」では、発振周期に対してシミュレーションの「タイムステップ」(時間刻み幅)を充分に小さくする必要があります。

「Transient解析」を行なう際、「LTspice」は自動的に「タイムステップ」を調整しながらシミュレーションを行ないますが、発振回路など計算誤差の影響を受けやすい回路の「Transient解析」では、「Maximum Timestep」を必ず設定します。

設定の目安は発振周期の「1/50」～「1/100」ぐらいですが、回路にもよるので、シミュレータ上で発振しない場合は小さい値を設定してみてください。

ここでは、想定される発振周期「1/8MEGHz = 125ns」に対して、「Maximum Timestep = 0.1ns」に設定しています。

高い精度で周波数を測定したいので、特に小さな「Maximum Timestep」を設定していますが、あまり小さな値を設定するとシミュレーションに時間が掛かってしまいます。

[重要]発振回路の「Transient解析」では、発振周波数よりも充分に小さい「Maximum Timestep」を設定する。

「.step」では、「CL」を「6pF」～「26pF」まで、0.5pF刻みでスイープしています。

「CL」の値は発振の安定性にも関係しているので、あまり大きくは変更できません。

「.meas」では、「trig ～ targ」構文を用いて、5周期の時間を測定しています。

「V(OUT)」「V(IN)」のどちらの周波数を測定してもいいのですが、時間変化が急ではない波形のほうが時刻を正確に求めやすいことを考慮して、「V(IN)」を測定しています。

＊

シミュレーションには時間が掛かるので、開始したらしばらく放置してください。

図5-55に、シミュレーション結果の波形の一部を拡大した図を示します。

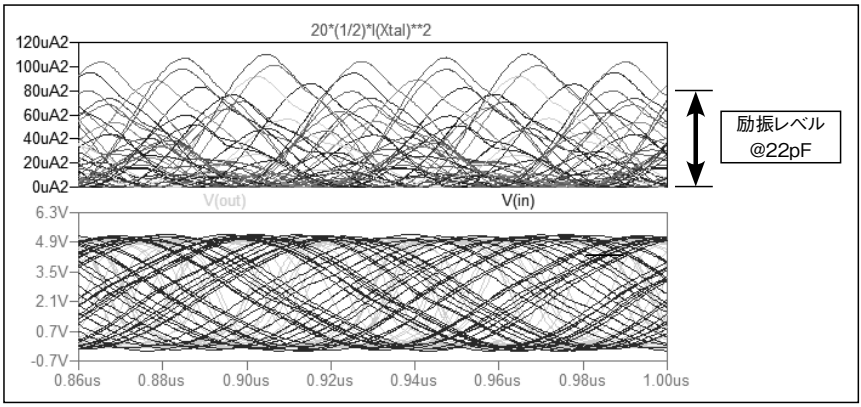

図5-55　「発振振幅」と「励振レベル」

「励振レベル」は、「水晶振動子」内部の消費電力です。

「水晶振動子」が正常動作する「最大励振レベル」がデータシートに記載されているので、これを超えないように「R2」「R3」を調整します。

図5-55上側のグラフでは、「水晶振動子」の消費電力「Rs・(1/2)・I(Xtal)**2」を計算しています（「**」は、2乗の演算を表わします）。

抵抗値の単位は「Ω」ではなく「無次元」として計算しているため、縦軸の単位は「(Ω・A・A) = ワット(W)」ではなく、「uA2」(μA2)のように表示されますが、間違いではありません。

確認が必要なのは、「V(IN)」「V(OUT)」の振幅と、「水晶振動子」の消費電力の最大値です。

上記の「水晶振動子」の消費電力の計算方法の場合、各トレースのピーク値が「励振レベル」(電力の2乗平均)に相当します。

「励振レベル」は「CL」の値によっても変化するので、「CL」を変更したら、「励振レベル」の再調整が必要です。

「C1」「C2」による発振周波数の変化を調べるために、次のような「.meas」ディレクティブを使っています。

```
.meas tran Tosc trig V(IN)=2.5V rise=1 targ V(IN)=2.5V rise=6
```

「V(IN) = 2.5V」を下から上に通過するポイントの、1回目と6回目の間の時間を測定しているので、5周期の時間を測定したことになります。

「Tosc」は、測定結果を保存する変数名です。「rise=6」を「rise=2」に変更して、1周期の時間を測定してもかまいません。

パラメータ・スイープの結果をグラフに表示します。

---

[1] グラフ・ウインドウの上を右クリック。
[2] ポップアップ・メニューから、「View」→「SPICE Error Log」を選択。
[3] 「SPICE Error Log」ウインドウ上で、右クリック。
[4] ポップアップ・メニューから、「Plot .step'ed .meas data」を選択。
[5] 「Tosc」のグラフが表示されるので、グラフ上部のラベル「tosc」を右クリックし、「Expression Editor」で「5/tosc」を入力。
[6] 「5/tosc」(= 発振周波数)のグラフが描かれる。

グラフの作例を、**図5-56**に示します。

「CL」(横軸)が大きくなると、発振周波数が10kHzぐらいの範囲で下がる傾向があることが分かります。

実際の回路では、配線や部品のパッケージなど、いろいろなところに寄生容量があるので、測定しながら調整することになります。

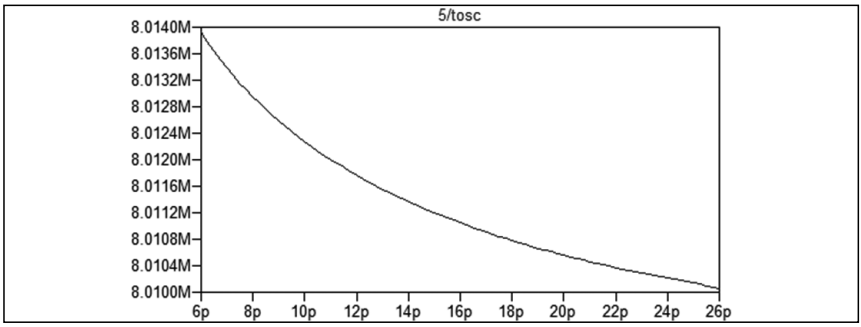

図5-56　「発振周波数」の負荷容量依存性

●発振の温度安定性

環境条件によって「水晶発振回路」が停止したり、振幅が減少したりすると、クロックを使っている回路がすべて停止し、大変困った事態になります。

温度が変わっても正常に動作するか確認するため、発振振幅と周波数の温度特性を調べておきます。

図5-57のように回路を変更してください。

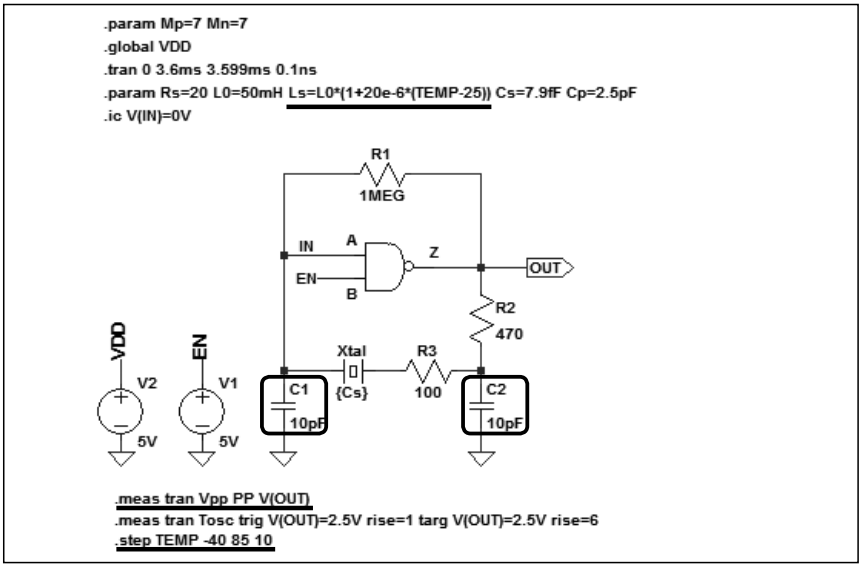

図5-57　温度依存性の測定

また、抵抗に「温度係数」を設定してください。「温度係数」の設定方法は、「反転増幅回路」で説明した方法と同じです。

---

[1] 抵抗のシンボルを、[CTRL] を押しながら右クリック。
[2] 「Component Attribute Editor」が起動するので、「Value2」に「tc=-200-6」を設定。
[3] 「Component Attribute Editor」の OK ボタンをクリック。

NAND を構成している「MOSFET」は、温度特性のモデルが組み込まれているので、設定の必要はありません。

「水晶振動子」の温度依特性は複雑なので、ここでは共振周波数の室温付近の温度係数で近似しておきます。

「水晶振動子」の共振周波数の室温での温度係数は、データシートに記載されています。使っている水晶片の結晶方位によって異なりますが、「$-10 \cdot 10^{-6}$(1/℃)」ぐらいの値です。

「32.768kHz」(計時用)の「水晶振動子」は、室温付近で温度係数が「0(1/℃)」です。共振周波数の温度係数「α」は、等価回路のインダクタの温度係数「β」に換算すると、「$\beta = -2\alpha$」で近似できます。

ここでは、インダクタの温度係数を、「$+20 \cdot 10^{-6}$」として、「.param」ディレクティブに、「L0=50mH」「Ls=L0*(1+20e-6*(TEMP-25))」を設定しておきます。

1次近似なので、室温から離れるほど誤差が大きくなります。

「.meas」では、「V(OUT)のピーク・ツー・ピーク(Peak-to-peak)」と「周波数」を測定しています(「ピーク・ツー・ピーク」は、波形の最大値と最小値の差)。

下記の書式によって測定を行ないます。

.meas 解析種別 結果変数 操作 測定対象の式　測定範囲(trig 〜 targ)

ここでは、「操作」を「PP」(ピーク・ツー・ピーク)としています。

また、「trig ～ targ」構文による「測定範囲」の指定は省略しています。
「測定範囲」を指定しない場合は、波形全体が測定範囲となります。
より正確なピーク・ツー・ピークを求めるためには、1周期だけを指定するほうがいいのですが、振幅が小さい場合は、トリガ (trig ～ targ の条件判定)を適用することが難しくなります。

*

図5-58に、「温度」を横軸としたグラフの作例を示します。

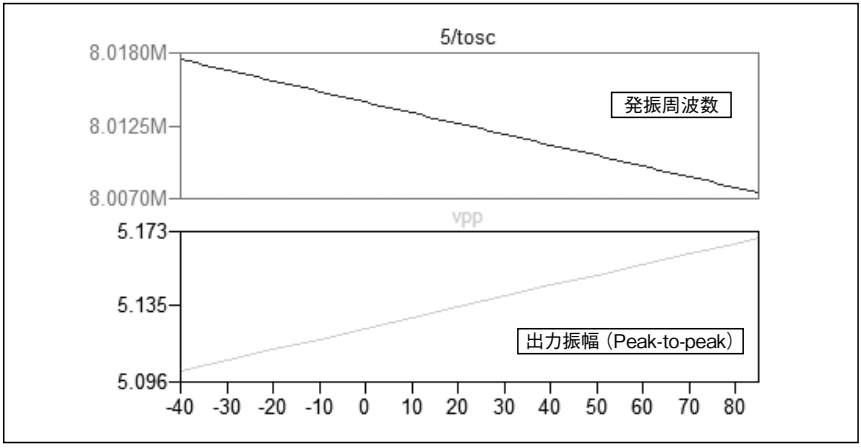

図5-58　「発振周波数」と「振幅」の温度依存性

周波数は「5/tosc」、振幅は「vpp」で表わしています。
「Tosc」は**図5-54**のときの測定と同様に、5周期ぶんの時間になります。

この結果から、「−40℃」～「85℃」の範囲では、「水晶発振回路」が正常動作することが分かります。
また、温度変化によって、振幅が「+0.5mV/℃」程度増加し、周波数が「−80 Hz/℃」程度減少することも分かります。

■「発振余裕度」の評価

「LC発振回路」や「水晶発振回路」は、共振回路の電流振動を減衰させないように、振動波形を増幅してから共振回路に戻すことで、動作を持続させています。

共振回路の「電流振動」(または「電圧振動」)の減衰は、共振回路からのエネルギーの散逸に起因するので、電力を消費する抵抗を共振回路にわざと追加して、発振が停止するまでにどれだけの余裕があるかを評価します。

余裕が少ないと、部品の「特性ばらつき」や「温度変化」によって、発振が停止する可能性があります。

*

図5-59のように、「水晶振動子」と直列に「RM」を挿入してください。

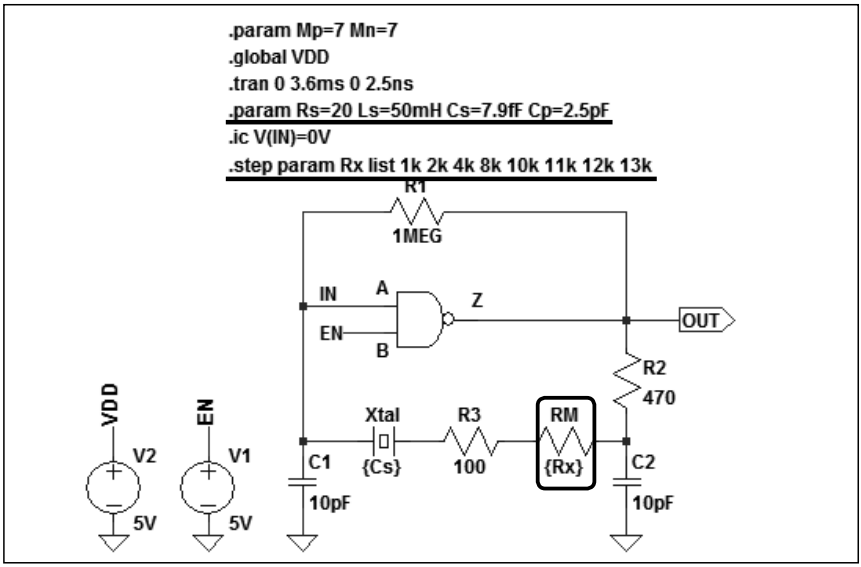

図5-59　「発振余裕度」の測定回路

「R3」と「RM」は、「水晶振動子」に流れる電流によって電力を消費するため、「RM」を大きくしていくと、いずれは発振しなくなります。

「.step」ディレクティブによって、「RM」の抵抗値「Rx」をパラメータ・スイープしています。

「t=0s」から発振が開始される様子を観察するため、「.tran」ディレクティブの設定値も変更してください。

*

シミュレーションが完了したら、「V(OUT)」のグラフを作ってください。

図5-60のように、「Rx」の増加とともに、発振の開始が遅くなり、同時に発振振幅が小さくなっていき、最終的には、実用的な発振開始時間と発振振幅が達成できなくなります。

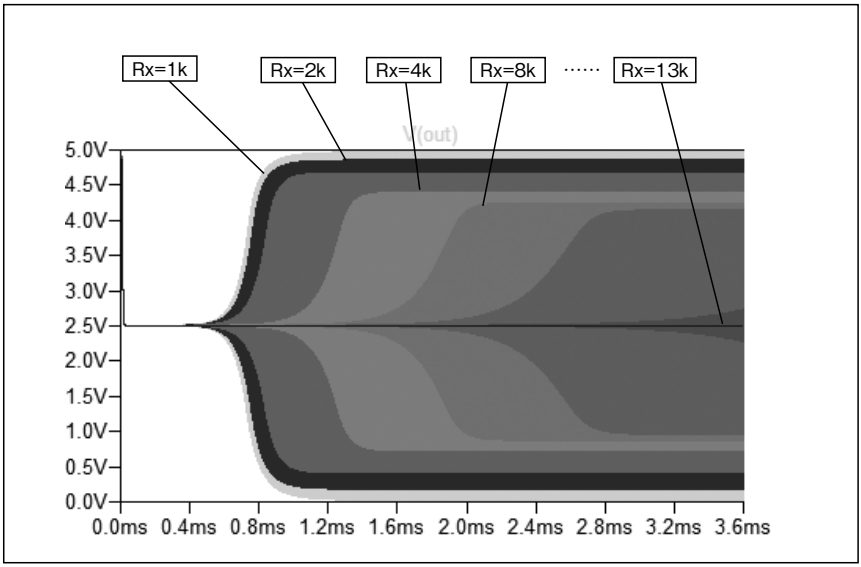

図5-60　「位相余裕度」の測定結果

この例では、「Rx = 13k Ω」ぐらいで発振を開始するのが困難になります。

発振ができない「Rx」の値を「Rx_max」と置くと、次式のように発振余裕度「Mosc」を求めることができます（「Rs」と「Cp」は、「水晶振動子」のパラメータです）。

$$M_{osc} = 1 + \frac{R_{x\_max}}{R_s}\left(1 + C_p \frac{C1+C2}{C1 \cdot C2}\right)^{-2}$$

「Mosc = 1」のとき、共振回路（R1、R2、R3、Rsを含む）の消費電力と増幅回路の電力供給が釣り合い、発振がギリギリ持続します。

「水晶振動子」のメーカーによると、「発振余裕度」は「5～10倍」が必要とされています。

シミュレーション結果によると、「Rx_max = 13kΩ」から、「Mosc = 290」となり、かなり大きい余裕があることになります。

*

実際の発振回路の測定は、計測器や計測用の回路を接続することによって周波数や振幅が変わってしまうので、やや面倒です。

「発振スペクトラム」の測定では、「スペクトラム・アナライザ」という計測器にアンテナを取り付けて、「水晶発振回路」のそばにアンテナを近づければ測定できますが、個人で所有している人はあまりいないと思います。

また、アンテナに増幅回路を付けて、比較的安価な「周波数カウンタ」につないで周波数を測定するという方法もあるようです。

発振振幅をアンテナで測定することは難しいので、オシロスコープに「アクティブ・プローブ※37」を取り付けて、インバータの出力端子に接触させて測定します。

【参考】「C1」と「C2」のバランス

例題の回路では、「C1 = C2」としていましたが、これは回路の動作に必要な条件ではありません。

「C1 = C2」の場合は、共振回路がGNDから見て左右対称となり、「C1」と「C2」の電圧は、同じ電圧振幅でシーソーのようになるはずですが、増幅回路(NANDやインバータ)を接続したため、左右対象な動作をしていません。

※37 寄生容量が小さい高周波測定用のプローブ。電源が必要。

「発振余裕度」が小さい場合は、「C1」と「C2」のバランスを「数pF」ぐらい変更してみてください。

容量を大きくした側にシーソーの支点（GND）が移動し、相対的に容量が小さい側の電圧振幅が増えます。

■「半導体製造サービス」について

本節では、「NANDゲート」内の「MOSFET寸法」と「並列接続数」を調整しましたが、実際にこうした調整をどのようにするのか疑問をもった方もいるかもしれません。

半導体デバイス特性を変更する方法としては、次の2つが考えられます。

①市販の半導体デバイスの型番を変更。
②半導体デバイスの回路を設計して、半導体メーカーに製造発注する。

①で対応できれば手軽ですが、ちょうどいい特性のデバイスが市販されているとは限りませんし、微調整や機能追加もできません。

しかし、②のように、電子回路シミュレータを用いて半導体内部回路を自分で設計し、半導体メーカーに発注すれば、「利得帯域幅積」「ドライブ力（出力電流）」「遅延時間」「入力寄生容量」「トランス・コンダクタンス（入力電圧-出力電流の係数）」などが変更でき、さらに「温度補償」「自己診断」「リダンダンシー※38」などの高度な機能を組み込むことも難しいことではありません。

そして、何より半導体デバイスのモデル・パラメータが入手できないために、シミュレーションができないという事態が起こりません。

「半導体製造サービス」の利用は、一部の企業や研究機関では当たり前に行なわれていますが、まだ一般に普及するまでには至っていません。

しかし、本節で利用したような1umぐらいの加工寸法の製造技術であれば、シャトル方式※39の少量生産を利用して、プリント基板の製造サービスと同程度の費用で製造できるようになりつつあります。

※38 冗長性を持たせること。回路技術分野では、同じ回路を複数用意して、製造不良や障害が発生した場合に代替させる技術を指している。

古い製造技術といっても、小規模なマイクロプロセッサや1GHz以下の高周波回路、アナログディジタル変換器なども搭載可能ですし、汎用目的でなければ、特定の性能を突出させることも可能です。

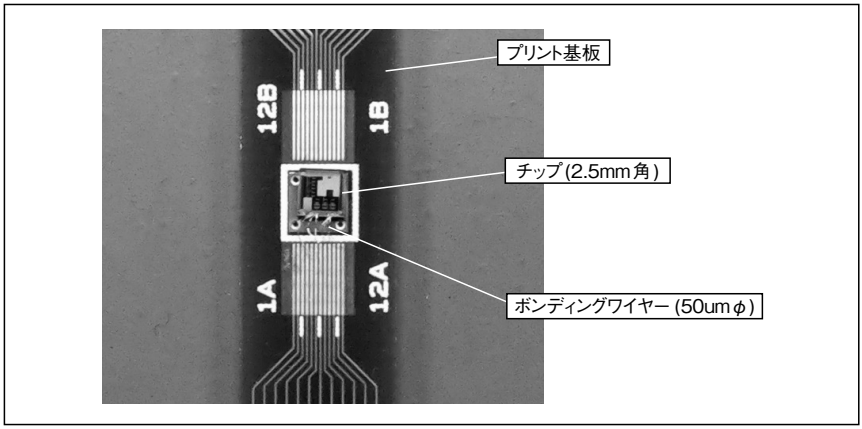

図5-61　プリント基板に直接接合されたカスタムLSI
（2.4GHzコルピッツ発振回路を4基内蔵）

*

近い将来、電子回路シミュレータを片手に、誰でもカスタム半導体を生産できるようになるでしょう。

まだ試行錯誤の段階ですが、半導体の非専門家がフリーのCADソフトを用いて、実際に半導体製造を行なうプロジェクト（**参考文献10**を参照）も行なわれています。

半導体デバイスの設計を含む回路設計では、電子回路シミュレーションが頼りです。

電子回路シミュレータが真価を発揮する時代は、これから始まろうとしています。

※39 定期的に製造依頼を募集し、一枚のウェハを複数のユーザーでシェアして製造を行なう半導体生産の方法。