第4章　階層回路図の作成法（4bit 加算回路の例）

本章では、4bit 加算回路の作成を通して、NS-Draw での階層回路の作成方法を学びます。

４．１　回路シンボル（インバータ）の作成

まず、C:¥Design¥ns-tools¥example¥TUTORIAL¥CHAPTER_4_5(4BITADDER)のフォルダ下にある、inv1.nsd を開いてください。

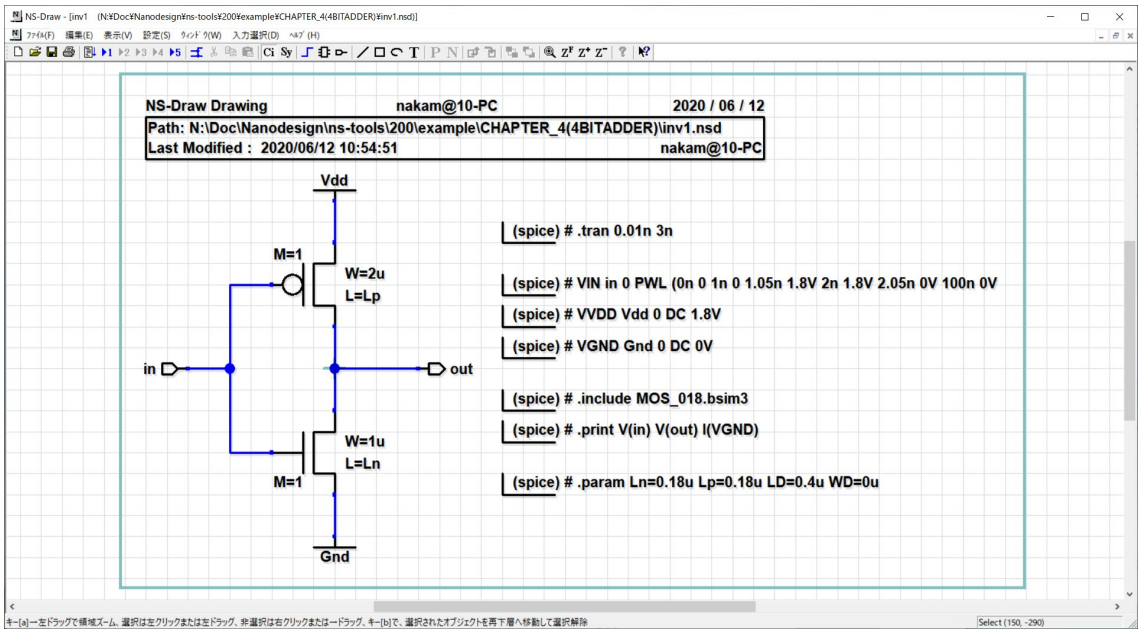

図４．１．１　インバータ回路　inv1.nsd

３章では、インバータ回路(inv0.nsd)を、４端子 MOSFET で設計しましたが、今回は、図４．１．１のように３端子 MOSFET：　nMOSFET(nmos3M.nsd)，pMOSFET(pmos3M.nsd)を使用しています。一般的な CMOS 論理回路では、pMOSFET と nMOSFET の基板端子(sub)はそれぞれ、電源（Vdd)、グランド（Gnd）と固定接続のため、３端子の MOSFET シンボルを用いた方が、回路図が見やすくなります。基板端子の接続については、pMOSFET,nMOSFET のシンボルをクリックして、sub プロパティで、例えば nMOS の場合、図４．１．２ように、グランド(Gnd)を指定してください。(pmos3M,nmos3M の基板端子は、デフォ

ルトで Vdd,Gnd になっています）尚、SPICE シミュレータでは、ネットリスト中で、大文字小文字の区別はされないので、Vdd は、VDD、vdd と表記しても、同一名として扱われます。ns-spice 内部では、素子名、ノード名等、すべて小文字に変換されて処理されます。

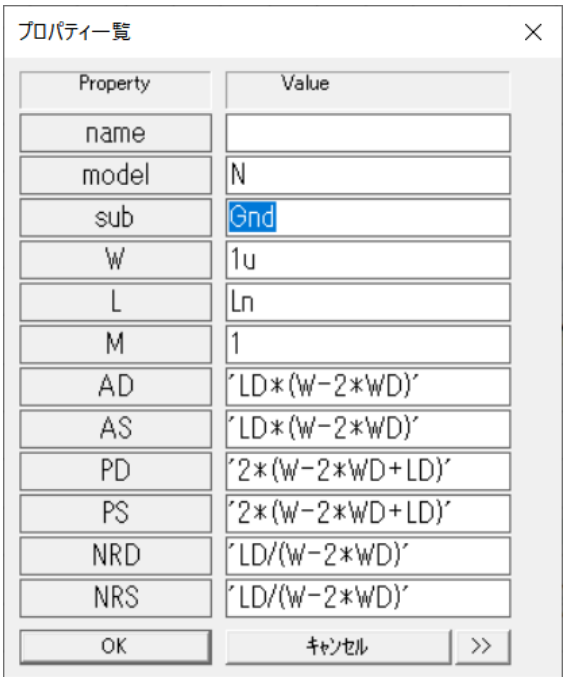

図４．１．２　３端子 MOS の基板端子

ここで、回路図には、回路図自身の図面（回路図：スケマティック）と、そのシンボル表現（シンボル：アイコン）の２種類の図面があります。これから、図４．４．１のインバータ回路図（スケマティック）に対応するシンボル図を作成します。

まず、NS-Draw メニューから、「ウィンドウ(W)」→「シンボル図」（あるいは、ツールバーから、[Sy]ボタン）を選択してみてください。画面が、シンボル画面に切り替わるのですが、このとき、シンボル図が空であるため、次のようなメッセージが出ます。

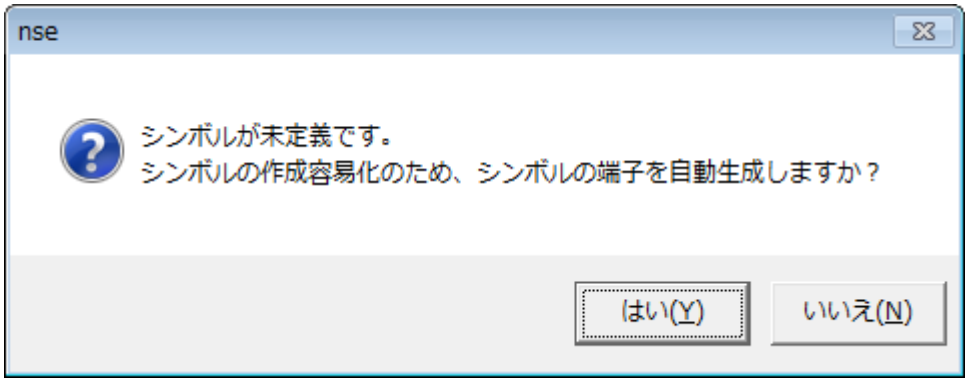

図４．１．３ シンボル未定義メッセージ

ここで「はい」を選択すると、つぎのような画面となります。

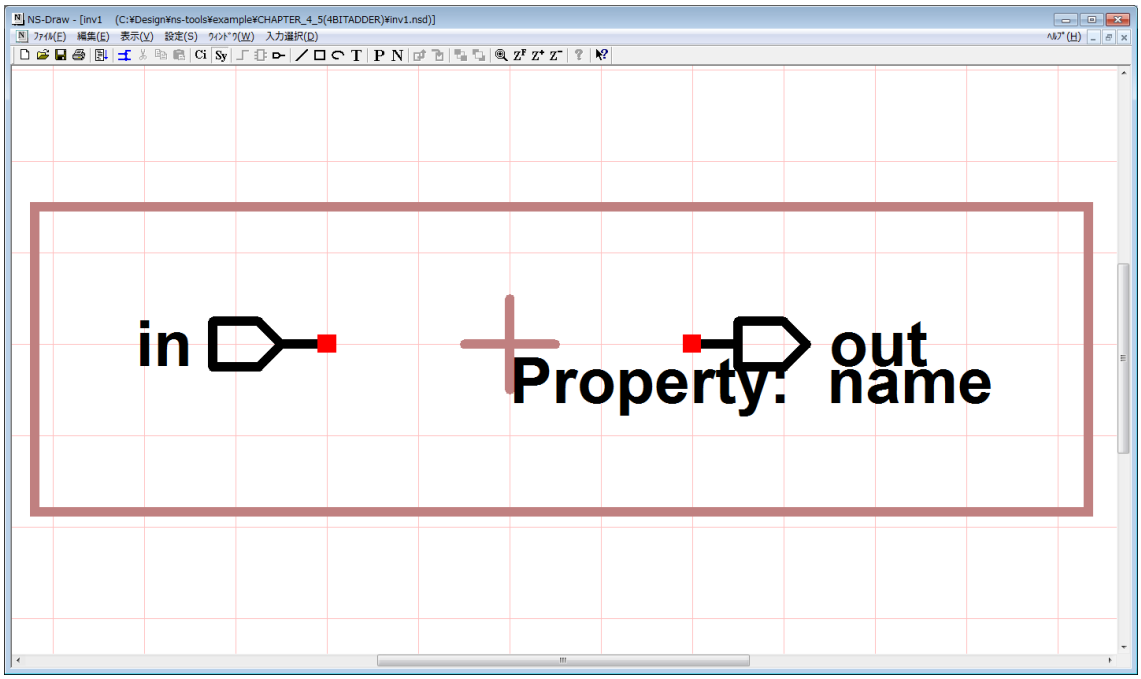

図４．１．４ シンボルの自動生成

in, out の入出力端子、および「Property: name」という文字列が自動生成されています。これを、図４．１．５のように適当な位置に移動してください。

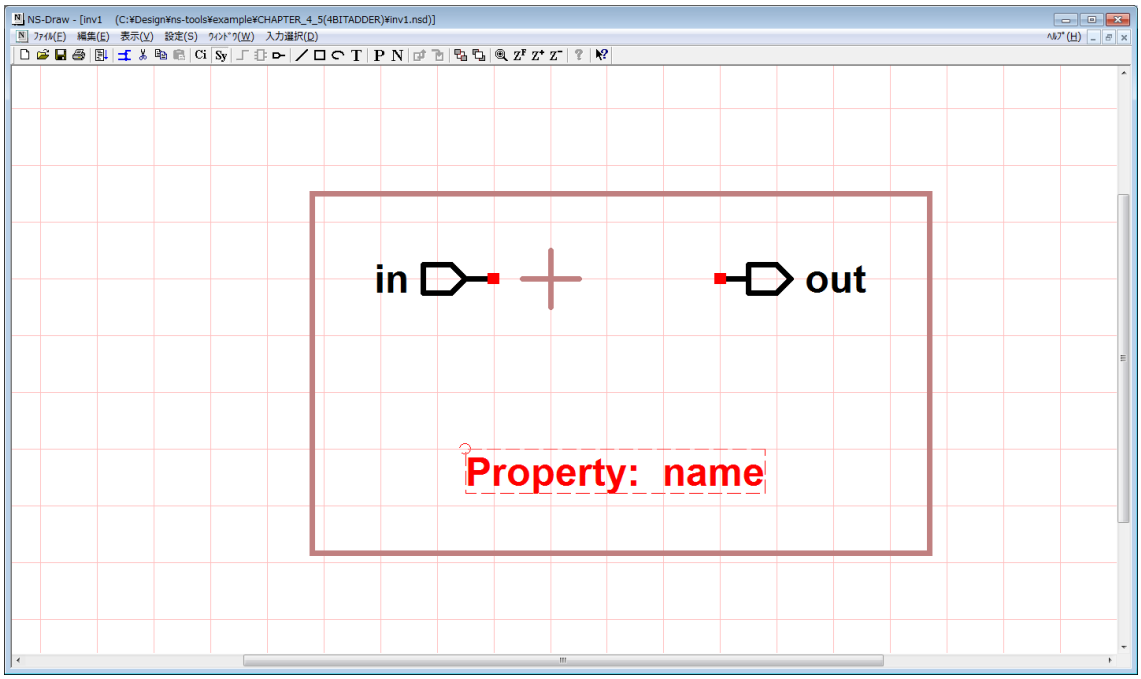

図４．１．５ 端子と Property 位置の移動

次に、ツールバーのほぼ中央にあるライン描画(L)をつかって、三角形の図形を描いてください。配線と同様に、折れ点で左クリックし、最後の点では、ダブルクリックしてください。次に、「円・円弧入力」により、円を追加してください。最終的に図４．１．６のような図面にします。

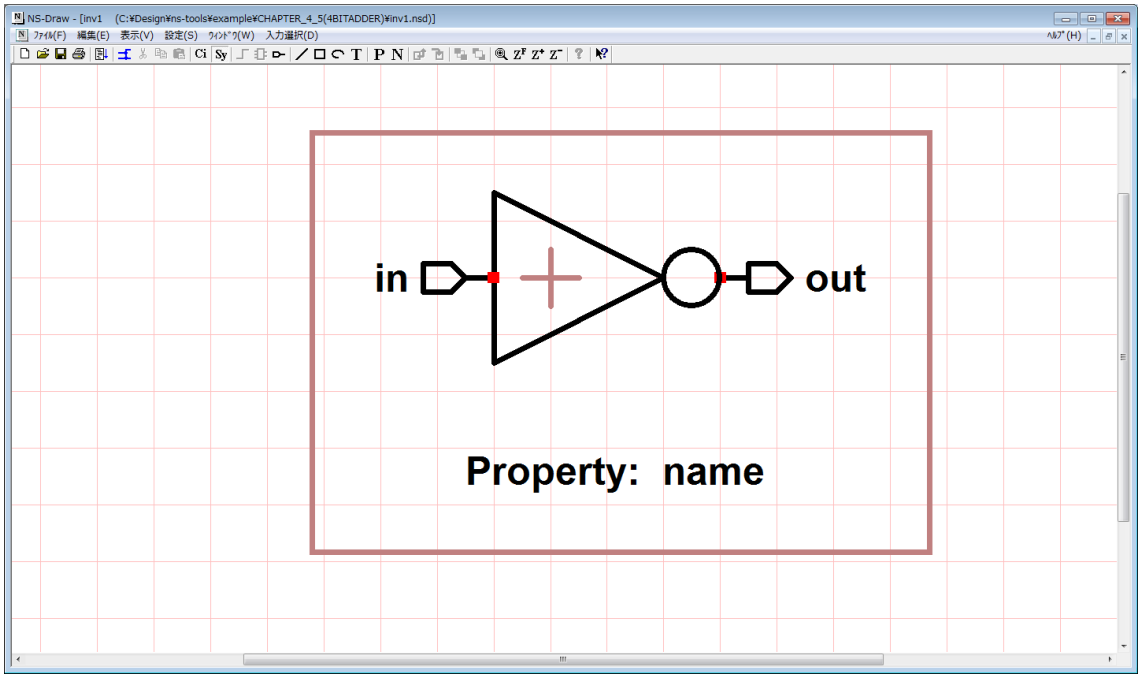

図４．１．６ シンボル図の描画

続いて、「テキスト入力(T)」により、インバータシンボルの名前と W 値、及び、$name を記入します。メニューの「入力選択 (D)」→「テキスト」（もしくはツールバーの「T」）を選ぶと、次のようなウィンドウが表示されます。

図４．１．７ テキスト入力ウィンドウ

ウィンドウの「Text:」欄に所望のテキストを入力します。このインバータ回路の名前である「INV1」と入力してみましょう。その下の「配置基準」欄は、そのテキストを配置したときの原点の位置を示します。ここでは「Center－Center」の位置にチェックを入れてみます（図４．１．８）。

図４．１．８ テキスト入力ウィンドウ（入力後）

「OK」ボタンを押すと、テキスト入力ウィンドウが消え、赤色の「INV1」という文字がマウスについてきますので、適当な位置に置いてクリックしてください。

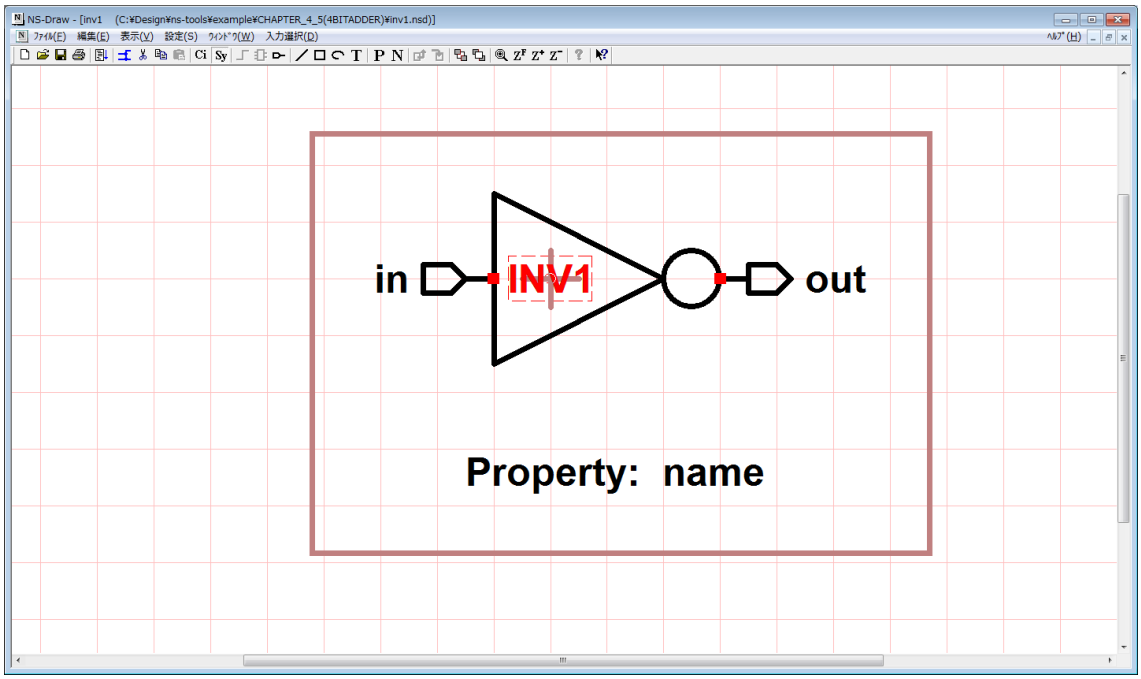

図４．１．９ テキストの配置後

同様にして、$name を図４．１．１０のように記入してください。ここで、テキストの「配置基準」は、とりあえず、「Lower－Left」にしておいてください。$name は、プロパティ：name の値（文字列）を表示するもので、上位の階層で、各インスタンス名（このサブサーキットの名前）と置き換わって表示されます。

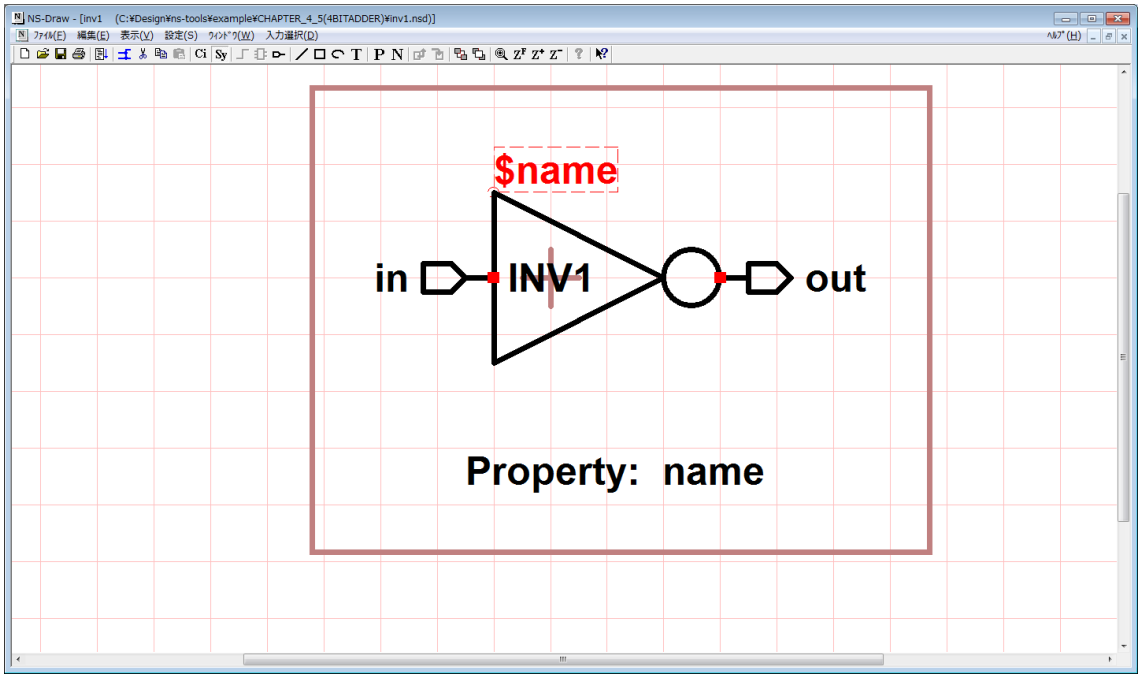

図４．１．１０ 完成したインバータのシンボル図

図形等の編集中に、たとえば、″INV1″という文字を選択しようとした場合に、斜め線のほうが選択されてしまう場合があります。このときは、「編集」→「背面へ移動」を行って、図形の前後関係を入れ替えてください。これは、”b”キーを使っても同様なので、意図しないものが選択された場合は、そのまま b キーを押して、もう一度同じ場所を、クリックして再度選択動作を行ってみてください。

「ウィンドウ(W)」メニューの「回路図」と「シンボル図」を選択するたびに、表示される図面が入れ替わります。これは、k キーや、ツールバーの[Ci]／[Sy]のボタンでも、同様です。シンボルが完成したら、データをセーブしてください。

4．2　2入力 NAND ゲートの作成

次に、インバータと同様に、２入力 nand ゲートを作成します。「ファイル」→「新規作成」から、２入力 nand ゲートの回路図を作成します。図４．２．１　に示すように３端子の MOSFET シンボル： nMOSFET(nmos3M)，pMOSFET(pmos3M)を配置して、W 値は pMOSFET,nMOSFET 共に４um にして、相互の端子接続、および入出力、電源端子の接続を行い、2nand4.nsd と名前をつけてセーブしてください。基板端子は、図４．２．１のように、pMOS は、Vdd，n MOS は Gnd とデフォルト値のままで問題ありません。

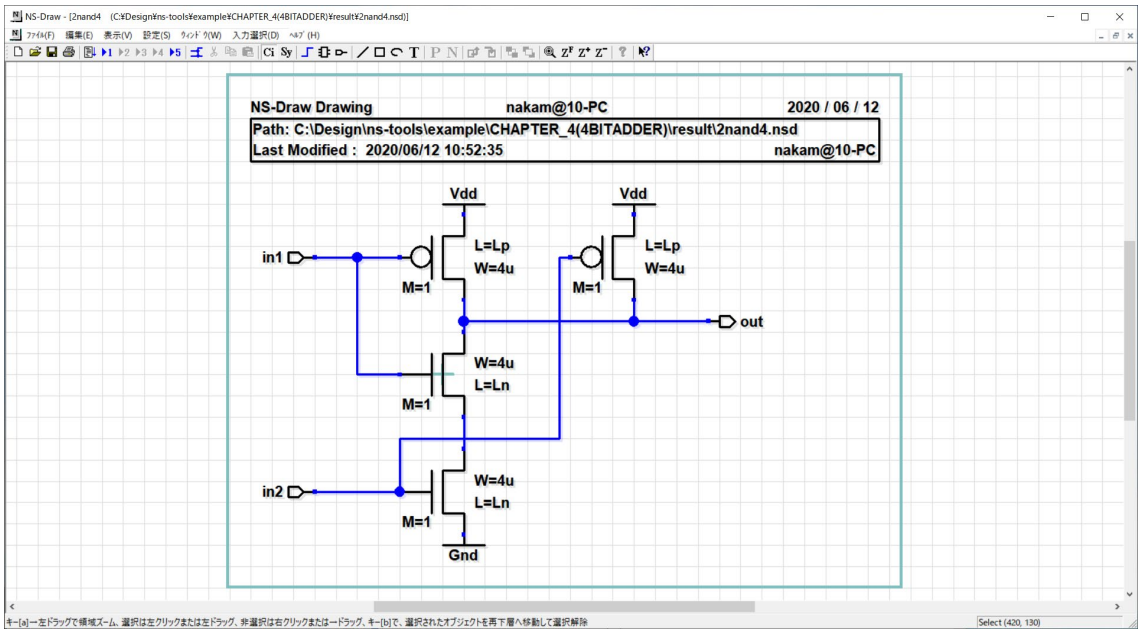

図４．２．１　　２入力 nand トランジスタ接続

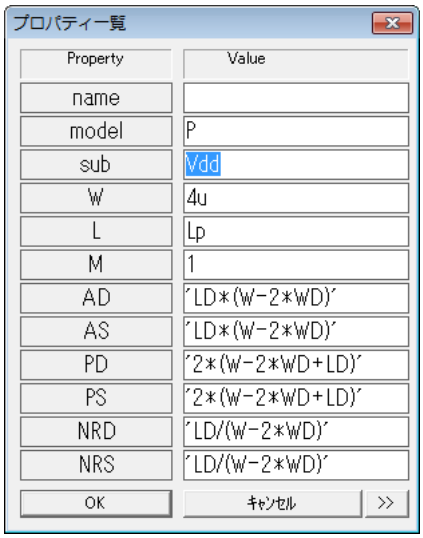

図４．２．２　pMOSFET の基板端子の設定

続いて、シミュレーション用の netlist_line_SPICE.nsd の記述を作成します。ここでは、inv1.nsd に貼り付けた記述を流用することにします。メニューの「ファイル」→「開く(O)」などで inv1.nsd の回路図を開き、回路図中の netlist_line_SPICE.nsd を全てマウスドラッグで選択し、ctrl-C キーでコピーします。

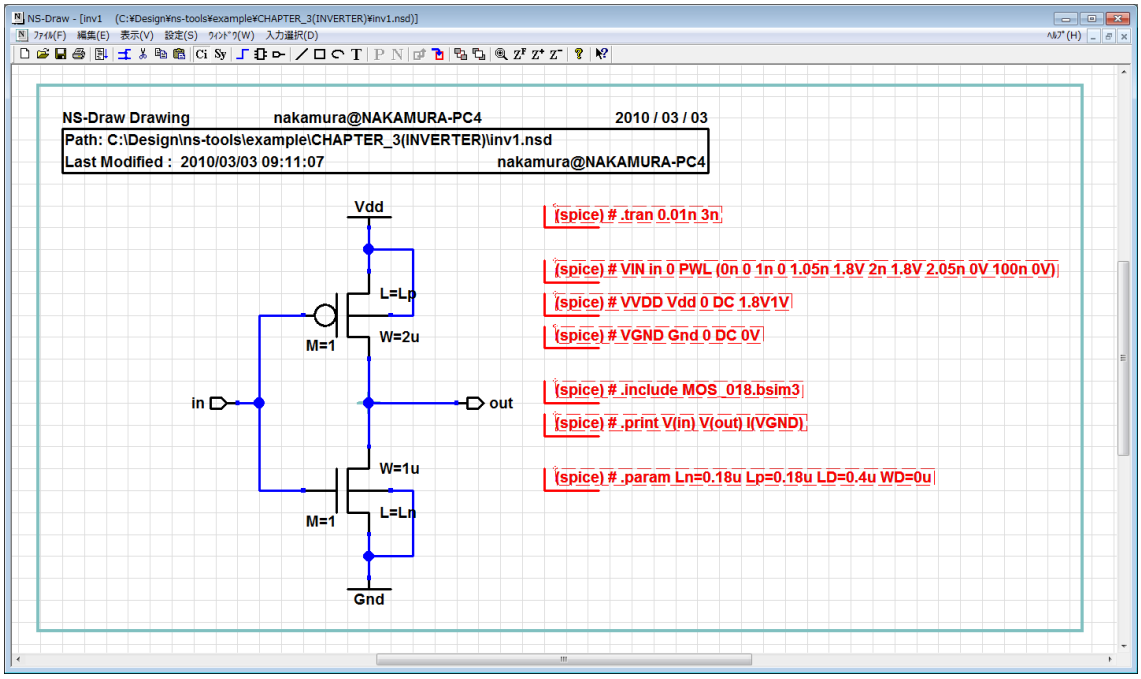

図４．２．３ inv1.nsd のシミュレーション用記述のコピー

ここで、メニューの「ウィンドウ(W)」をマウスクリックすると、下段に現在 NS-Draw で開かれている回路図ファイル名一覧が表示されるので(編集作業中の回路図ファイル名には先頭にチェック印が入っています)、「2nand4.nsd」にマウスカーソルを合わせてクリックすると 2nand4.nsd の画面に戻ります。

2nand4.nsd の画面に戻った後、ctrl-V を押すと、さきほどコピーした inv1.nsd のシミュレーション用記述を、2nand4.nsd の回路図上にペーストすることができます

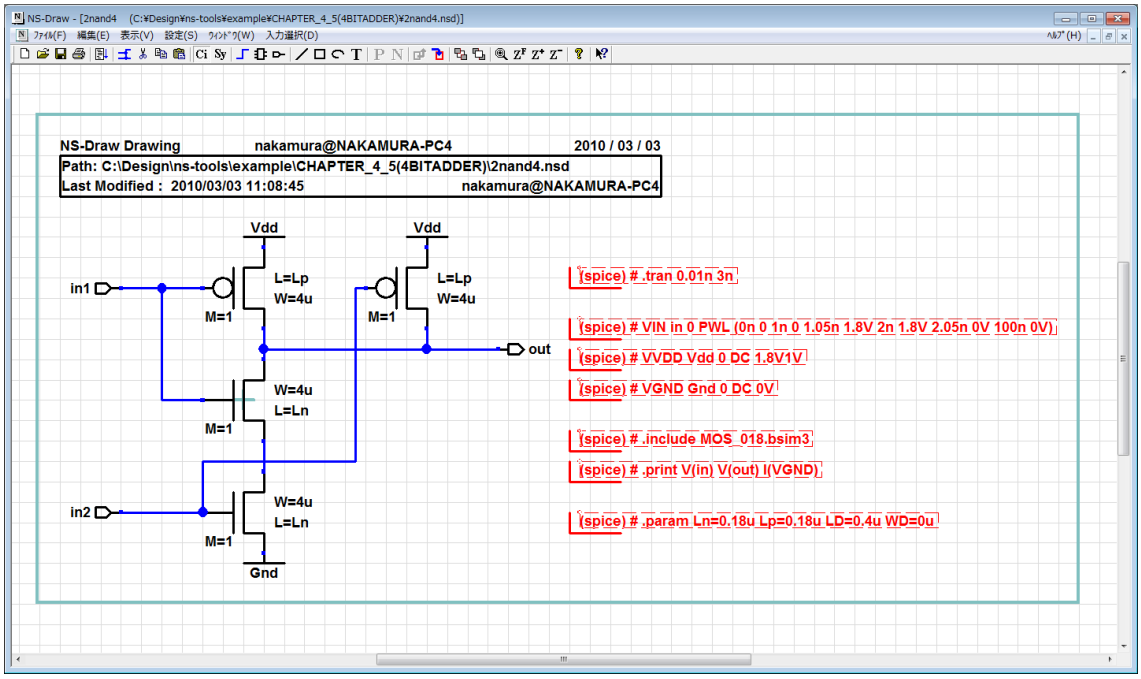

図４．２．４　SPICE シミュレーション用記述の貼り付け

図４．２．４を参考に、ペーストした netlist_line_SPICE 記述を適宜修正・追加して、２入力 nand 回路のシミュレーション用記述を完成させてください。追加・修正箇所は下記の４行分です。各 netlist_line_SPICE.nsd を置く位置は、回路図の中に存在しさえすれば、どのように置いても構いません。各自見やすいように配置してください。

また、今回の回路では、in1, in2 端子から印加する入力信号として、inv1.nsd で用いていた PWL 形式ではなく、繰り返し波形の入力に便利な PULSE 形式で指定してみてください。PULSE 形式は、PULSE(電圧値１　電圧値２　遅延時間　立ち上がり時間　立ち下がり時間　電圧値１となる期間　周期)の順で指定します。例えば図４．２．５内の VIN1 に対する記述では、周期 2nsec、立ち上がり、立ち下がりとも 0.05nsec の、0V から 1.8V の周期波形になります。詳しくは ns-spice または、SPICE3F5 のマニュアルをご参照ください。

```
.tran 0.01n 4n
VIN1 in1 0 PULSE (0V 1.8V 1n 0.05n 0.05n 0.95n 2n)
VIN2 in2 0 PULSE (0V 1.8V 2n 0.05n 0.05n 1.95n 4n)
.print V(in1) V(in2) V(out) I(VGND)
```

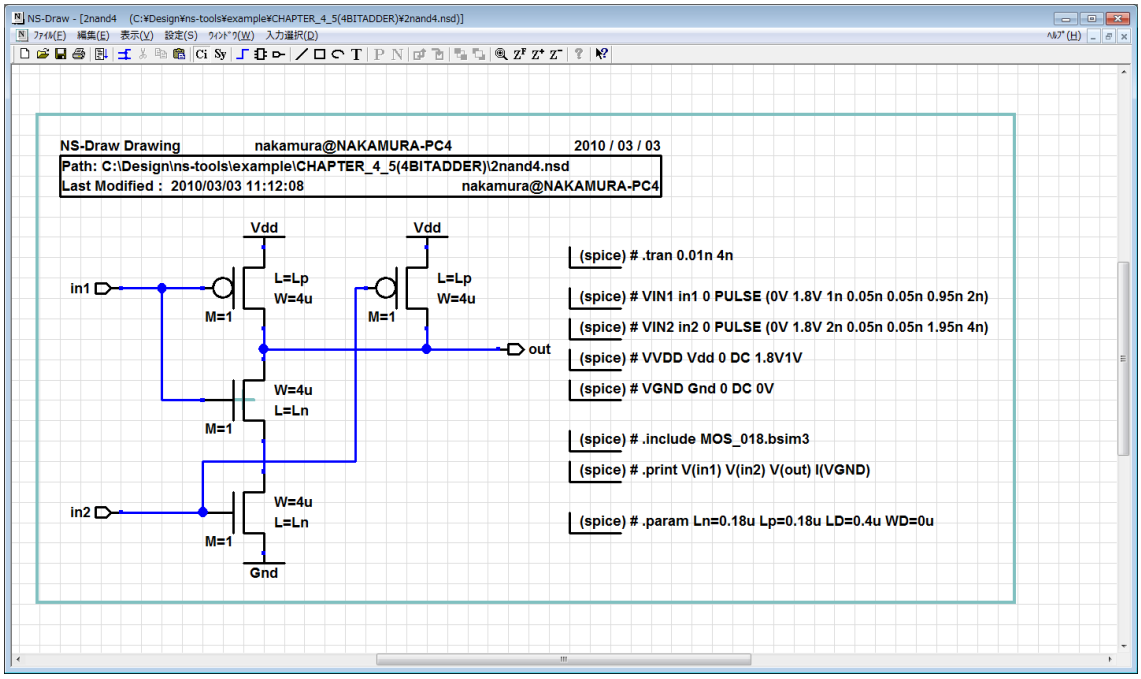

図４．２．５　２入力 nand ゲート（シミュレーション記述完成）

回路図ができたら、シミュレーションを実行してください。
論理ゲートのシミュレーション結果は、波形の数が多くなりがちなので、VS32 で波形を描画後に、メニューから、「描画設定」→「XY 軸設定」を選択します。

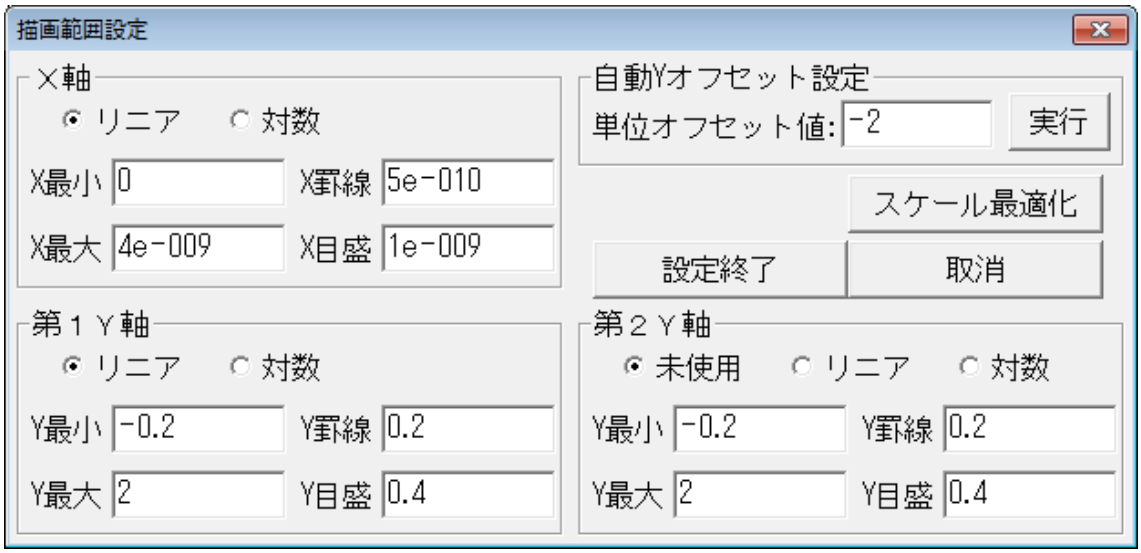

図４．２．６　VS32 の XY 軸設定

自動 Y オフセット設定で、オフセット値として-2 を入力し、「実行」ボタンを押し、

その後、「設定終了」をクリックしますと、図４．２．７のように、波形が一本ずつ、2V ずつ下にずれて表示されます。

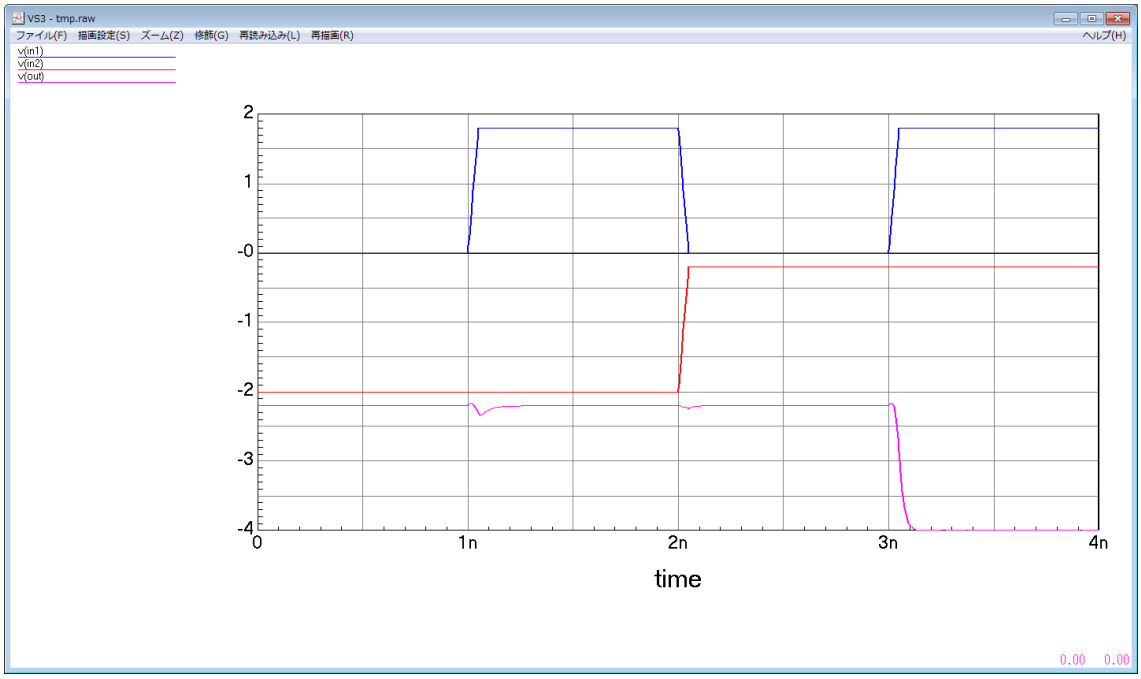

図４．２．７ VS32 による論理ゲートのシミュレーション結果表示

SPICE シミュレーション結果から、２つの入力に対して、出力が nand 論理になっていることを確認してください。

次に、ツールバーの[Si]ボタンにより、図１７のとおりに、シンボル図を作成します。円弧部分は、一度円を入力して、その後円をダブルクリックすることで、円弧として編集できます。in2 側に、”－“印がついていますが、これは、入力端子（in1/in2）の違いが分かるように付けているものです。

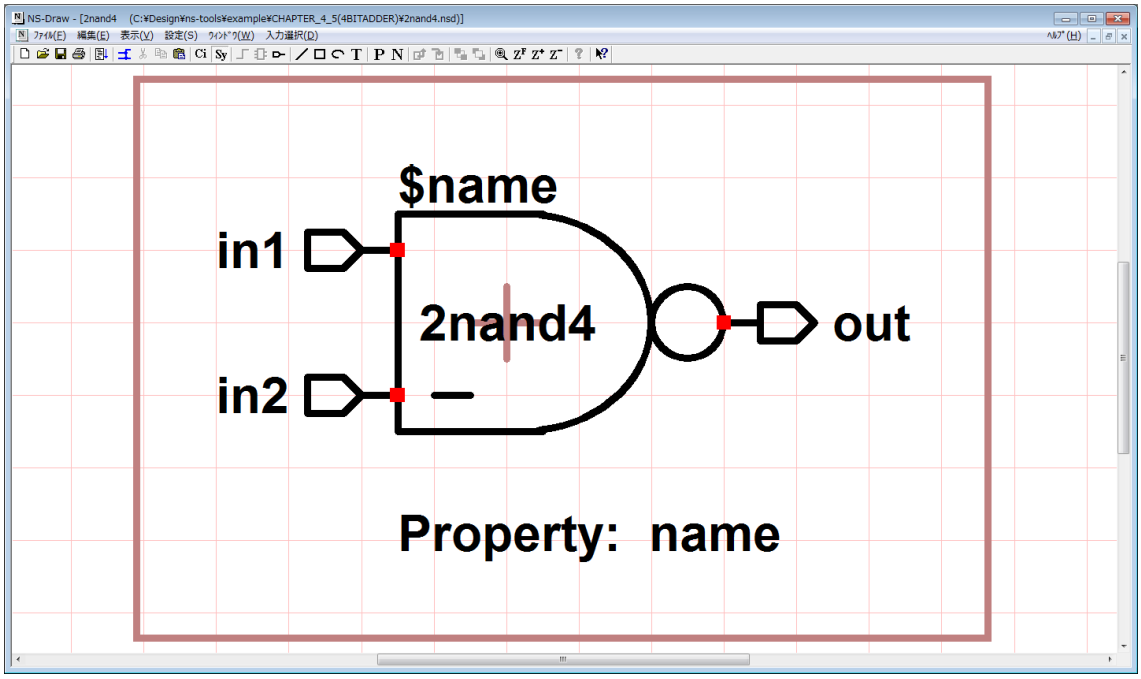

図４．２．８ 2nand4 のシンボル

完成したらセーブしておきます。

４．３ ２入力 XOR ゲートの作成

続いて、2nand4 と inv1 を使って、排他的論理和（XOR）回路を新規作成します。ここで、回路パーツの呼び出し（i キー）において、ライブラリパスとしてカレントディレクトリが追加されており、その内から、先に作成した inv1.nsd や 2nand4.nsd が選択・挿入できるはずです。これを使って、図４．３．１、図４．３．２のように回路図とシンボルを作成してください。各ゲートには、name プロパティが設定できます。各ゲートをダブルクリックして、名前を設定してみてください。ここでは、３つの 2nand4 に na1, na2, na3 と言う名前をつけてみました。ここで設定した名前は、ネットリスト出力時に反映されます。name プロパティを設定していないもの（この回路での２つのインバータ）は、ネットリスト出力時に適当に名前が割り振られます。シンボル図では、特に XOR の円弧の描画が難しいと思います。通常の描画では、端点をグリッドに強制的にあわせられてしまいますが、シフトキーを押しながらマウスを操作すれば、グリッド外の任意の点へ描画や移動ができます。これを利用してください。

回路図が出来たら xor.nsd 等の名前をつけてセーブしてください。その後、回路図内の 2nand4 または inv1 を選択し、ツールバーから「子回路へ降りる(E)」を選択してください。直ちに、下の階層の回路図に移動できます。キーボードの e キーを押しても同じことができます。また、「親回路図へ戻る(Shift-e)」で、上の階層に戻ることができ

ます。キーボードの E（Shift-e）を押しても結構です。この機能によって、階層構造の回路図の間を自由に行き来できます。

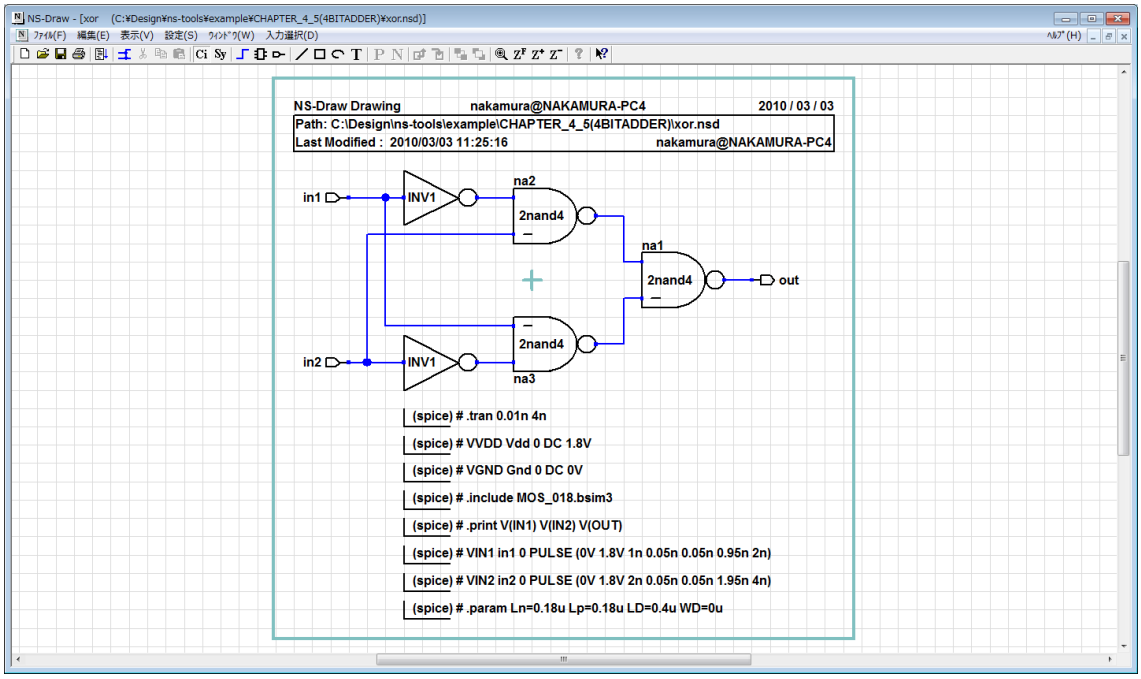

図４．３．１ XOR 回路（回路図）

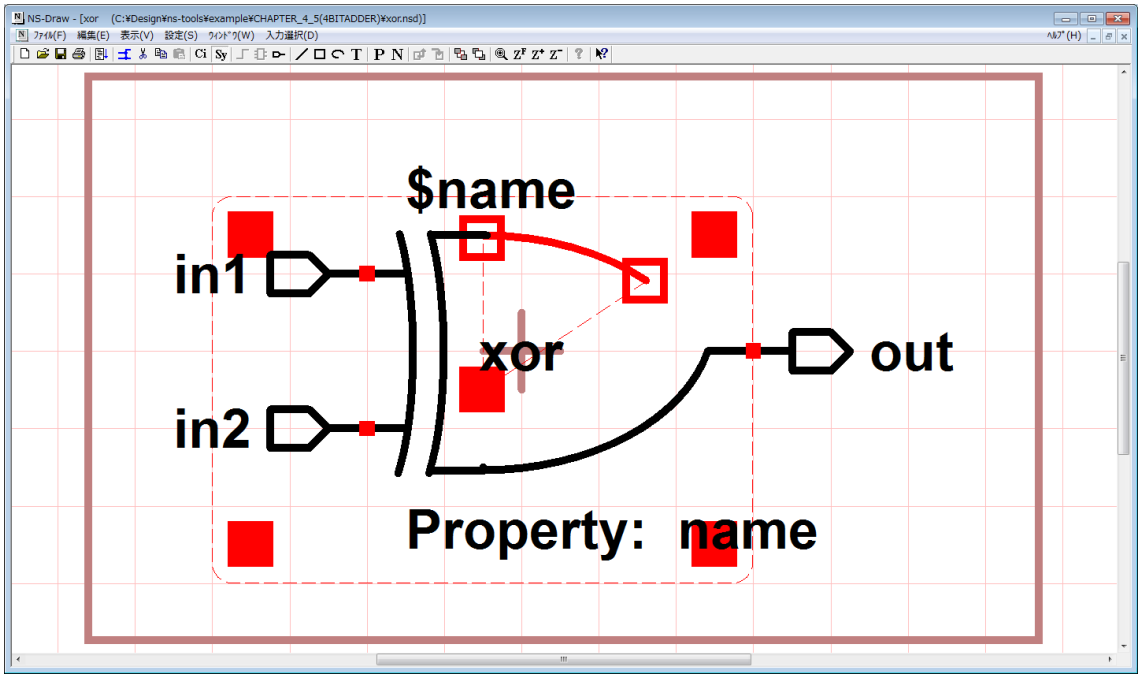

図４．３．２ XOR 回路（シンボル）と円弧の描画

この XOR の回路で、SPICE シミュレーションを行うに際して注意していただきたいこ

とがあります。XOR 回路図（図４．３．１）上では見えていませんが、ここで使われている inv1.nsd や 2nand4.nsd の中には、Ln, Lp, LD, WD といったパラメータが存在しています。例えば、回路図中のどれか１つのインバータ（inv1.nsd）を選択し、キーボードの e キーを押してインバータの中に入ってみてください。この階層では、トランジスタのサイズが、Lp,Ln 等で指定されており、また各トランジスタ内での拡散容量の計算式に、WD,LD のパラメータが使用されています。この階層では、Ln, Lp, LD, WD に対する実際の数値は、図４．３．３に示すように、netlist_line_SPICE.nsd を使った .param 文で指定されていました。

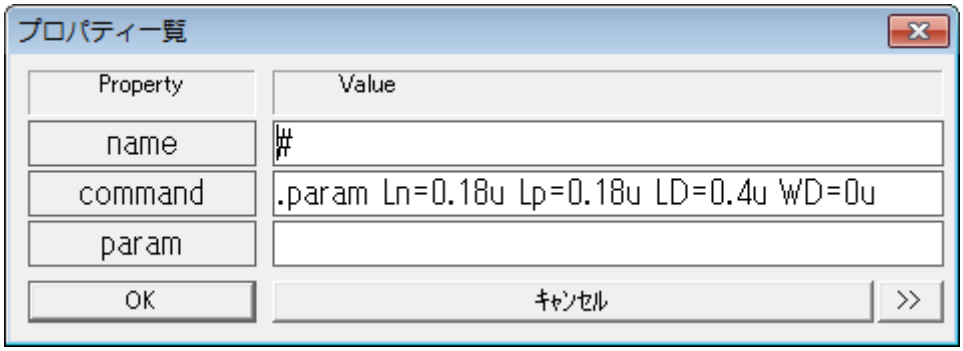

図４．３．３ netlist_line_SPICE.nsd による .param 指定

ここで、netlist_line_SPICE の、Property の name 項に # を記述した場合、この netlist_line_SPICE.nsd の記述は、その階層とトップレベルとして出力されたネットリストのみに埋め込まれます。上位の回路から呼び出された場合には、この行はネットリストには出力されません。よって、内部で Ln, Lp, LD, WD などパラメータを使用している inv1.nsd や 2nand4.nsd を他の回路図に配置した場合は、あらためてその回路図の中で .param 文を使って値を指定しなければなりません。このため、上記の XOR 回路図中には、あらためて .param 文で Ln, Lp, LD, WD を指定する netlist_line_SPICE.nsd が置かれています。

これにより、各階層の回路図単位でもシミュレーションが行え、また上位階層でシミュレーションを行う場合には、下位のシミュレーション記述は無効化され、各階層毎にいつでもシミュレーション可能な回路図を構築することができます。（ちなみに、'#'を'%'に変更すれば、下位の階層の回路図であっても、その行は必ず出力されるようになります。）

以上の点が確認できましたら、キーボードの E（Shift-e）キーを押してもとの XOR 回路に戻り、SPICE シミュレーションを行ってください。XOR 回路のシミュレーション結果を図４．３．４に示します。排他的論理和の動作が正しく行われていることを確認

してください。

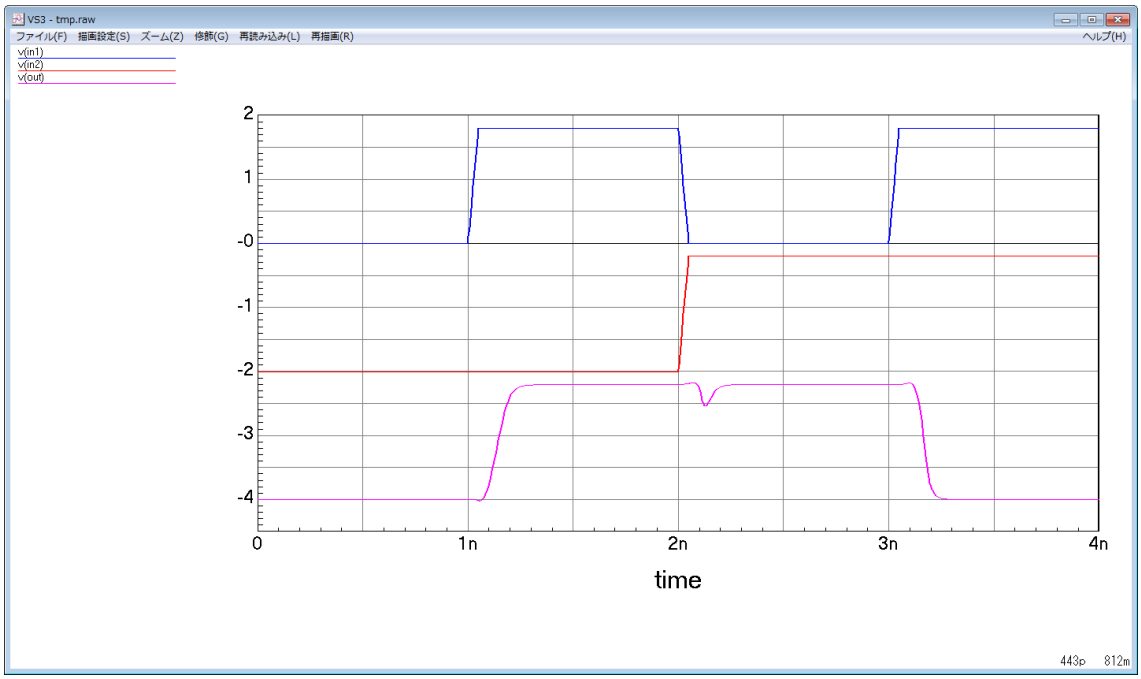

図４．３．４ XOR のシミュレーション結果

このように、回路階層毎にシミュレーションを行って、動作に問題があれば直ちに下層に降りて、各回路ブロックの動作を確認することができます。作成した回路図はシミュレーション記述を埋め込んで個別にシミュレーションして動作を確認して、上位階層の回路を Step-by-Step で組み上げていくことをお勧めします。

４．４ 1bit 全加算回路の作成

次に、これまでと同様に、1bit の全加算回路を、上記の XOR 回路を使って作成します。図４．４．１，図４．４．２，図４．４．３を参考にして、full_adder.nsd を作成、シミュレーションし、保存してください。

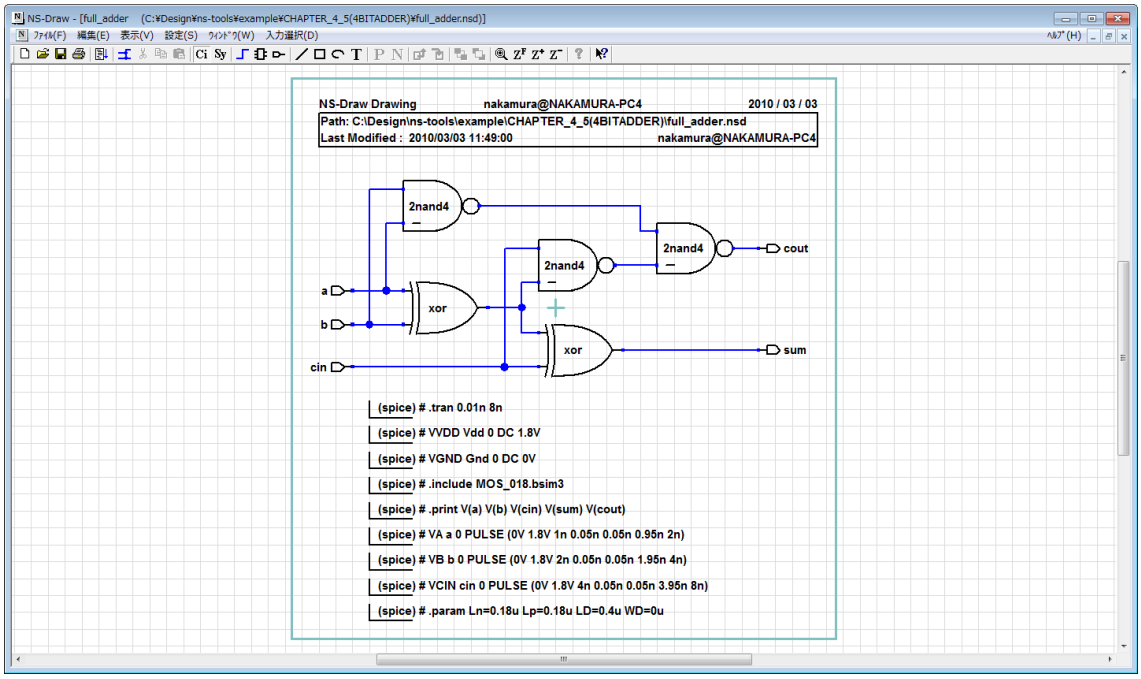

図４．４．１ 1bit 全加算器の回路図

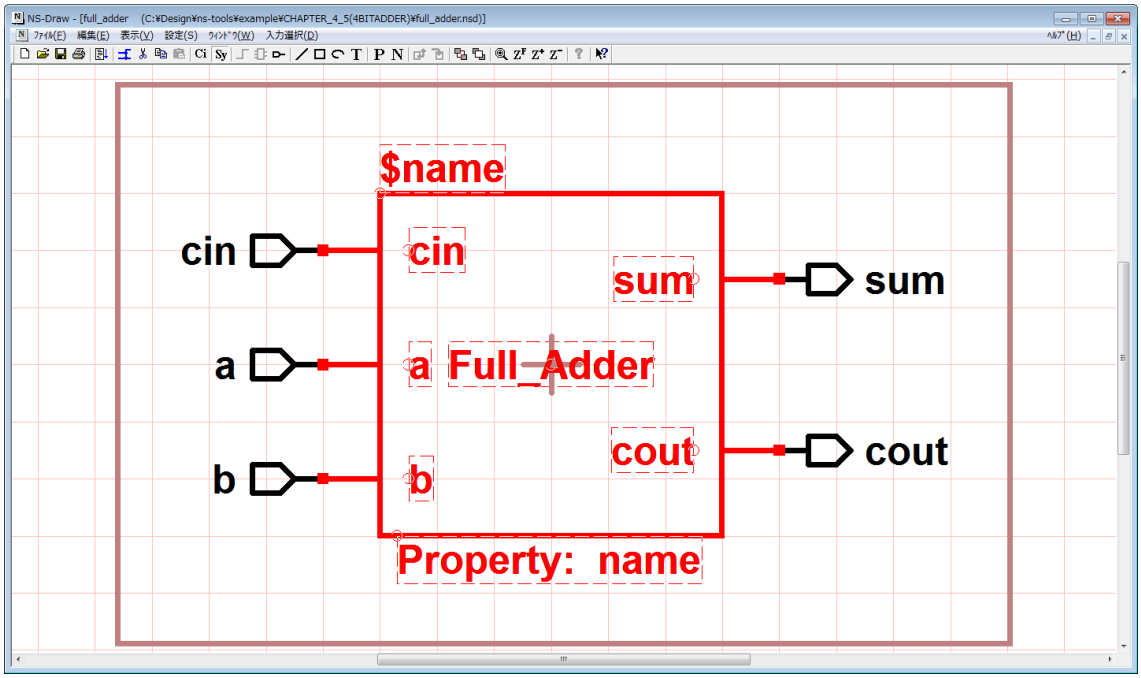

図４．４．２ 1bit 全加算器のシンボル図

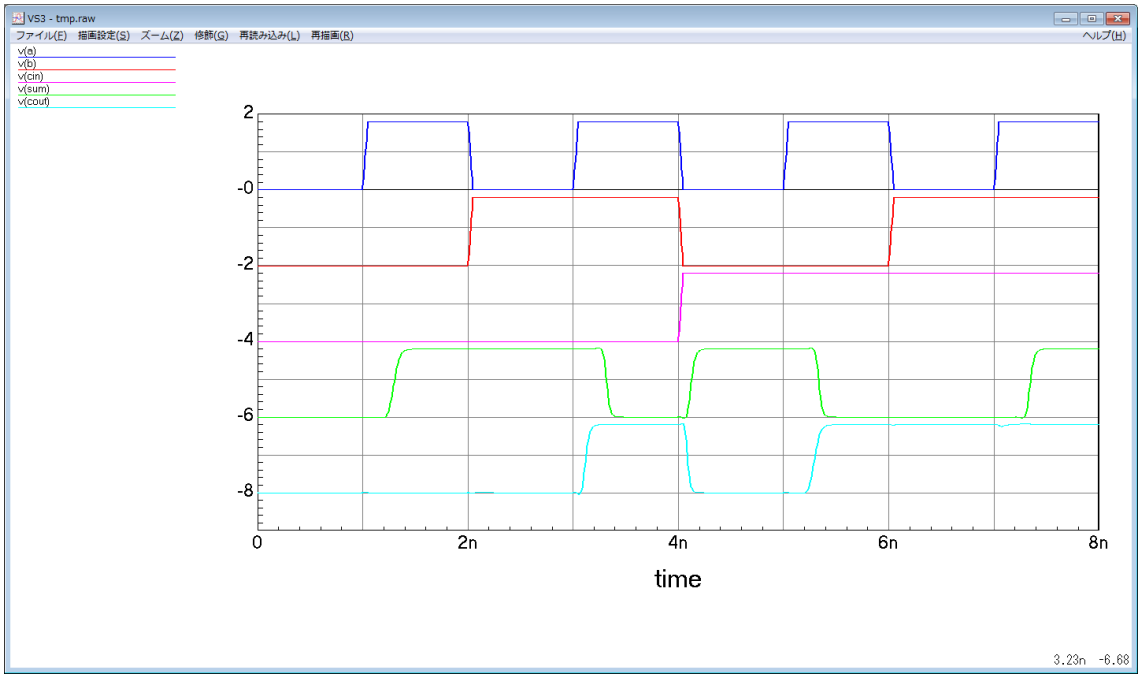

図４．４．３ 1bit 全加算器のシミュレーション結果

図４．４．２のように信号端子名（cin, a, b, sum, cout）をシンボルに記述するような場合、図面上左側にある端子（cin, a, b）のテキストは配置基準を「Center－Left」に、右側にある端子（sum, cout）のテキストの配置基準は「Center－Right」に置くことをお奨めします。シンボル中央にある、名前を示した Full_Adder については「Center－Center」がよいでしょう。こうすることにより、シンボルを回転・反転して配置したときに比較的みやすい位置にテキストが来ると思います（図４．４．４）。回路図上でシンボルを回転したり反転したりして配置したときに、テキストがシンボルに重なったり、逆にテキストがシンボルから離れすぎたりして見づらくならないように、配置基準をうまく設定してください。

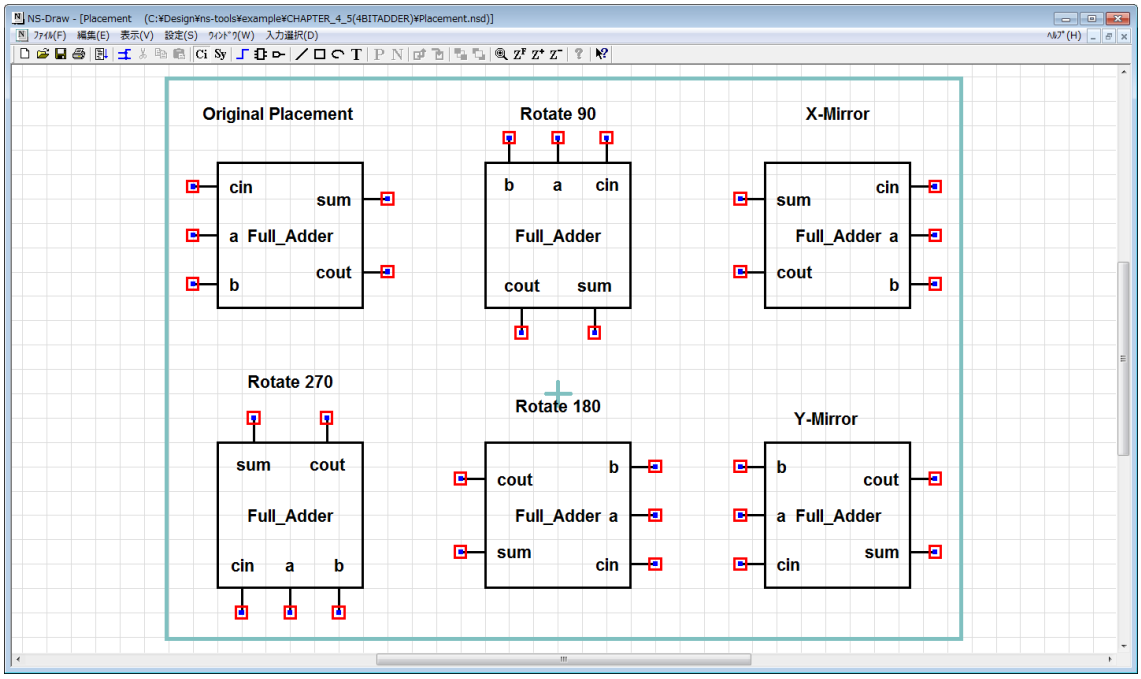

図４．４．４ テキスト配置基準の効果（Placement.nsd）

４．５　4bit 加算回路の作成

1bit 全加算器の動作が問題ない場合は、トップレベルとなる 4bit 加算回路の回路図を作成します。

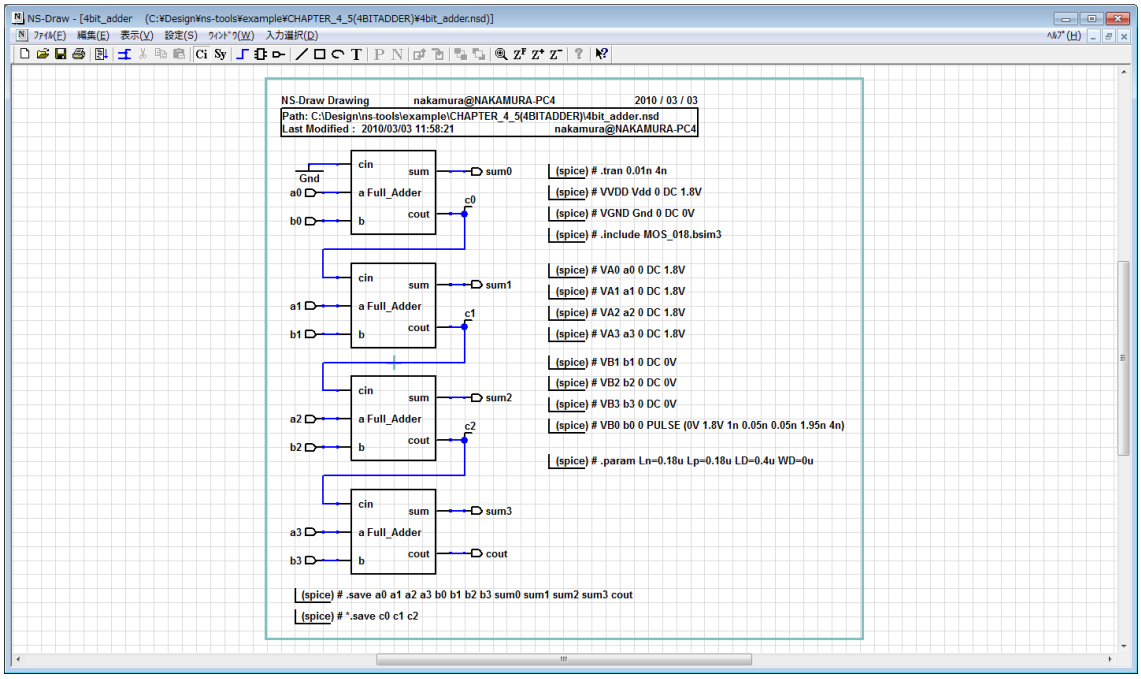

図４．５．１　４bit 加算回路の回路図

回路図内で、c0, c1, c2 等は、配線に名前を指定したものです。NS-Draw では、配線に名前を付けなくても、他の配線と重複しない名前(net_1, net_2 等)が自動的に付加されます。(配線を選択して「ｑ」キーを押すことで、その名前を確認できます。)　配線に名前をしたい場合は、「入力選択」→「端子」で、net_name を選んで配線上に設置し、名前を指定してください。波形を観察する際に、この配線名を指定します。また、ここでは、波形出力のノード名指定として、.print の代わりに.save を使っています。.save は spice3f5 の制御文で、電圧をモニタする場合は、.save のあとにノード名のみを列挙すればよく、V()で囲う必要がありません。

最終的な、４bit 加算回路のシミュレーションを行います。図４．５．１では、入力 a には、4bit の 1111 を設定し、b に 0001 と 0000 を交互に与える場合のシミュレーションになっています。加算結果 (out：5bit 長) として期待されるのは、01111 と、10000 の交互パターンになります。このパターンでは、最下位ビットから、最上位ビットまでの、キャリーの伝播が起こりますので、回路の動作速度を評価することができます。

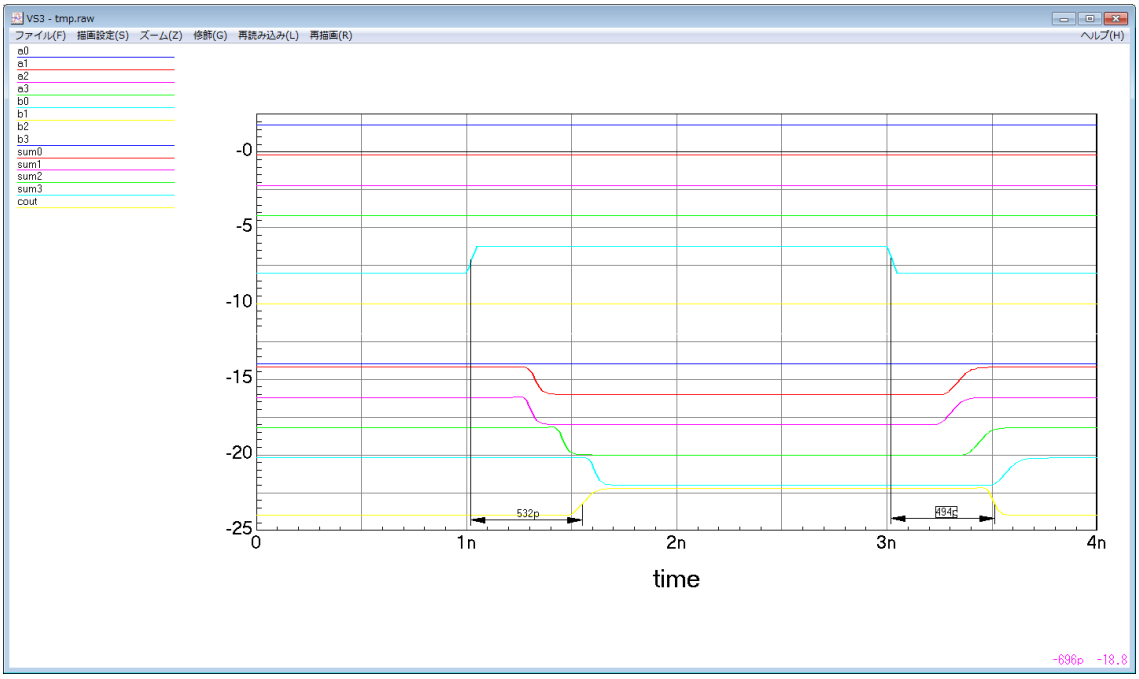

図４．５．２　4bit 加算回路のシミュレーション結果

b0 の最下位ビットの反転に伴って、出力が確定するまでに、500psec 程度の時間がかかっていることがわかります。ちなみに、図４．５．２内の時間線分は、VS32 内の「修飾」→「差分表示入力」で入力できます。

ここで、full_adder セルを選択して、e キーを押して、full_adder セルの中に入ってみてください。この階層でも netlist_line_SPICE や netlist_line_CDL 記述が埋め込まれていますが、先にも述べましたように、プロパティの name 項に # が付けられているものについては、ネットリスト出力時に、トップレベルの回路図中のもののみを出力します。よって、4bit 加算回路のネットリストには、この下の階層のシミュレーション記述は無視され、出力されないことになります。ネットリスト作成後に、ネットリストの中身を見ることで確認してみてください。デフォルトでは netlist_line_SPICE/CDL には # がついています。#の代わりに%を記述すると、その netlist_line_SPICE/CDL が埋め込まれた回路の階層に関わらず、ネットリストにはその記述が埋め込まれることになります。

これにより、各回路階層にシミュレーション記述を残したままで、上位の階層設計ができるようになりました。もし問題が発生しても、直ちに各階層で回路シミュレーションが行えるために、回路のデバッグも容易になります。