電子回路シミュレータSPICE

情報工学科
天野英晴

このスライドは演習をやるための最短な道のりを示しています。詳細はマニュアルを参照ください。

電子回路シミュレータ

- 論理シミュレーション
  - LとHのみを扱う
  - 遅延、電力は見積もり可能
  - 大規模回路の設計検証
  - Verilog HDL→計算機構成同演習
- 電子回路シミュレーション
  - アナログ的な電圧、電流の変化をシミュレーション
  - アナログ/ディジタル両方に使える
  - 遅延、電力を精密にシミュレーション可能
  - 実行時間が大きいため、大規模な回路のシミュレーションは困難

情報工学科で重要なのは、論理回路の動作をLとH（0と1）で解析する論理シミュレーションだと思います。これでも遅延、電力を見積もることができ、大規模回路の設計検証には欠かせません。これは計算機構成同演習、コンピュータアーキテクチャB、VLSI設計論で時間を掛けて紹介します。これに対して電子回路シミュレーションは、電圧、電流のアナログ的な変化をシミュレーションします。このため、アナログ、ディジタルの両方で使うことができ、遅延や電力を精密にシミュレーションできます。しかし、実行時間が大きいため、大規模集積回路をまるごとシミュレーションすることはできません。

SPICE

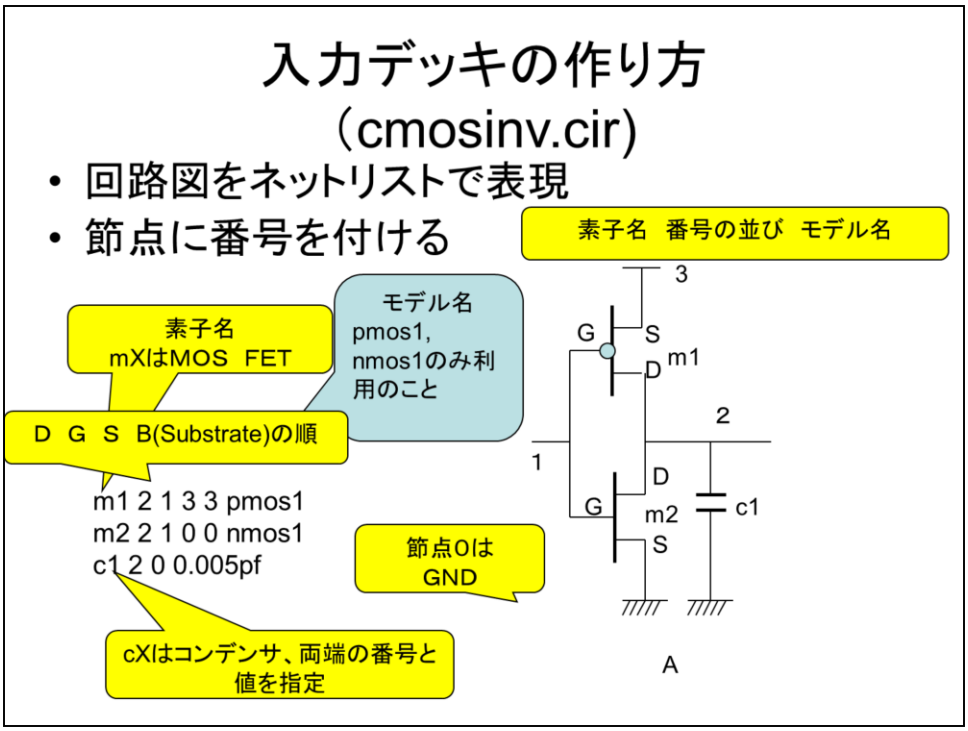
- 1980年代にUCB（カリフォルニア大学Berkeley校）で開発された
- 改良が続けられて世界中で利用されている
- 様々な版がある
  - 半導体のセル設計等：hspice
  - 高速シミュレーション：hsim
  - PC用：PSPICE
  - 今回はフリーソフトのngspiceを利用
    (http://www.ngspice.sourceforge.net)
- 基本的にはバッチ処理で、入力デッキを作ってシミュレータに掛けて、結果を後に見る

SPICEは1980年代にUCBで開発された由緒正しい電子回路シミュレータで、改良に改良がくわえられ、今でも世界中で用いられています。目的によってさまざまな版があり、プロ仕様のhspiceは普通に使うと高額なライセンス料が必要です。（教育目的でVDECのライセンスを使えば安いです。電子工学科はこれで演習をやっています。）今回はフリーソフトのngspiceを利用します。SPIECEの概念は80年代のソフトの考え方なので、はっきり言って古臭いです。基本的な考え方はバッチ処理であり、あらかじめ入力デッキというファイルを作ってやって、これをシミュレータに入力して、出て来た結果を後で見て解析します。

SPICE入力デッキ

- 回路記述、モデル記述、シミュレーション制御の3つに分かれる。
- 先頭に＊を付けるとコメント
- フリーフォーマットではない。行を折り返す場合は次の行の先頭に＋を付ける
- 一行の長さに制限があるものがあるので注意
  - 何といっても昔の考え方なんで、、
  - 今でもSPICEはFORTRANで書かれている
- 変数には約束事が多いので注意

ではSPICEの入力デッキを見てみましょう。例題のCMOSインバータ（cmosinv.cir)を見てみましょう。回路記述、モデル記述、シミュレーション制御の3つのパートからできています。先頭の＊を付けるとコメントで、行を折り返す場合は次の行の先頭に＋を付けます。また、一行の長さに制限のある場合もあるので気を付けましょう。この辺、なんといってもSPICEは昔のソフトです。変数名にも約束事が多いので注意してください。情報工学科で習うモダンでクールなプログラミング環境とは違う、ということを認識してください。



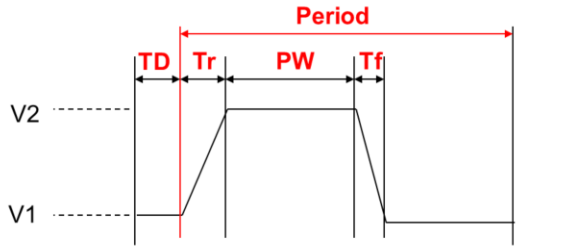
まず回路の記述には、節点記述法を使います。節点は0がGND以外は、適当に(いい加減)に番号を振ります。これに素子を接続していきますが、素子は「素子名　端子の番号の並び　モデル名」の順番で定義します。MOS FETは、素子名の先頭をmにします。後は何でもいいのですが、ここでは番号を順に振っていて、ここではm1はpMOSです。次に端子の番号を順に書いていきますが、MOS FETではDrain Gate Source Base(Substrate)の順です。BaseはpMOSでは電源、nMOSではGNDに繋ぎます。m1の場合、2 1 3 3になります。最後はモデル名です。ここでは後程pmos1とnmos1の二種類のモデルを定義しますが、それ以外は使いません。**なので、ここでpmos2とかnmos2とか書いてはいけません。**このモデル名はコンデンサや抵抗では省略されます。他の素子についてはマニュアルを参照してください。全ての素子をこのルールで定義すると、右に示す回路図ができあがります。

電源、入力の指定

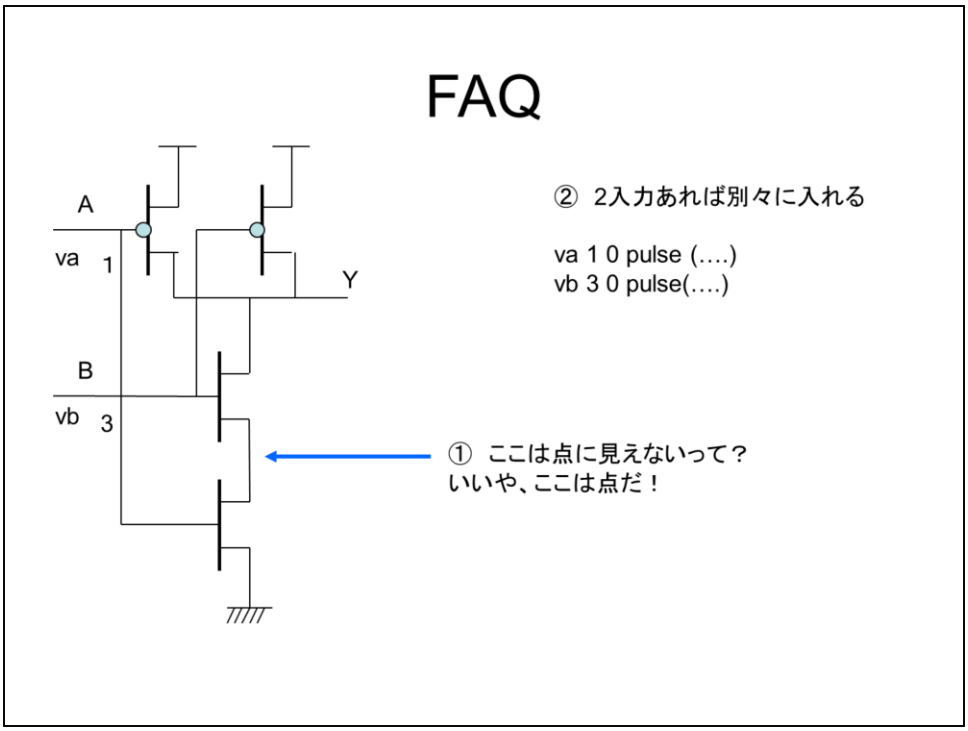
vXは電圧源

vcc 3 0 5v
vin 1 0 pulse(**0 5** 1ns 1ns 1ns 40ns 80ns)

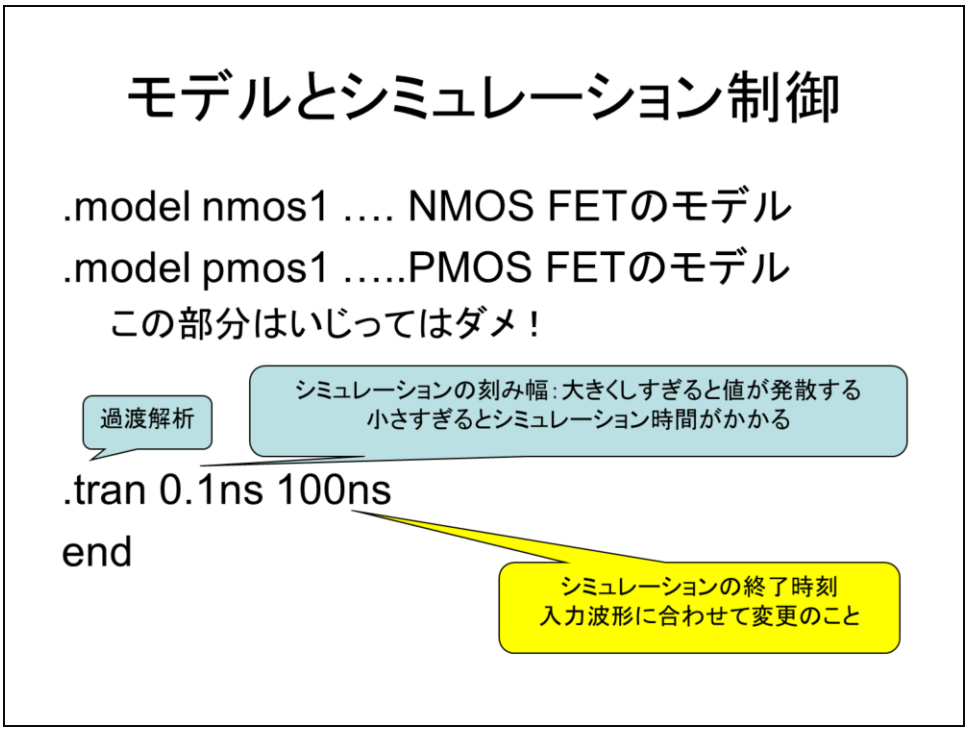
pulse(**V1 V2 TD Tr Tf PW Period**)



先頭にvを付けると電圧源を表します。固定電源の場合、電圧源の両端の端子番号を書き、その後に掛ける電圧を指定します。入力信号も電圧源として定義されますが、ここではpulseなど変化する入力を定義することができます。この演習で使うpulseは図のように電圧の時間変化を表します。



図の①のような場所は点に見えないぞ！という質問がありますが、これは二つのFETがくっついているのでどう見ても点です。ちゃんと番号を与えなければならないです。2つ入力があるときの入力電圧の与え方がわからない、という質問も良く受けるのですが、別々に定義してやればいいので簡単です。



次にモデルを定義する部分がありますが、ここはいじってはいけません。最後の.tranはシミュレーションの時間的な刻み幅と終了時刻を示す部分です。刻み幅は、大きすぎると値が発散するのですが、小さいとシミュレーション時間が掛かります。ここでは0.1ns固定でいいと思います。終了時刻はシミュレーションの入力波形に応じて変化させてください。

ngspice の起動とシミュレーション

ngspice cmosinv.cir

ngspice-> run

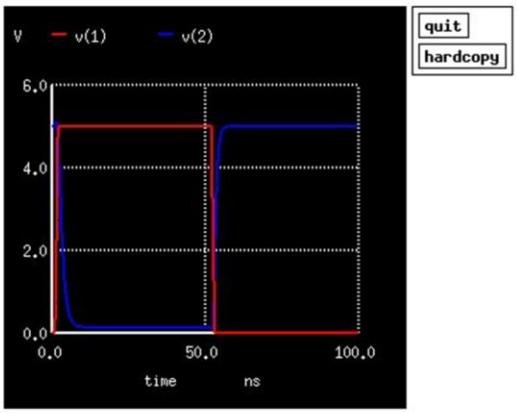
ngspice->plot v(1) v(2)

左クリックで、座標が表示される

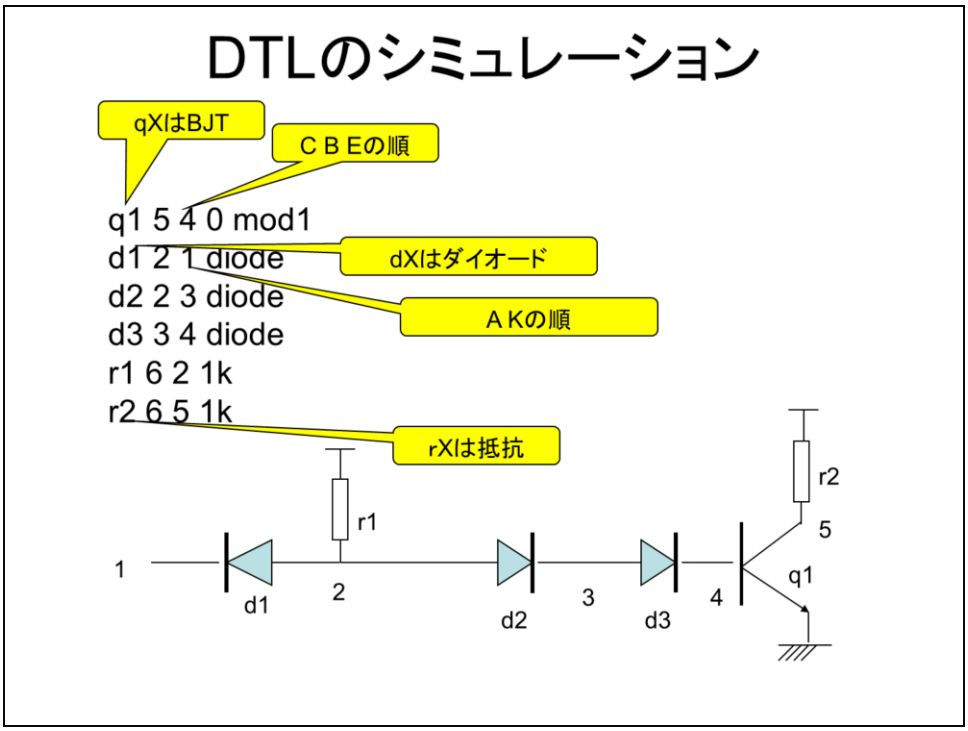
右クリック→ドラッグで拡大画面が表示される

他の機能はhelpで表示される

抜ける時はquit

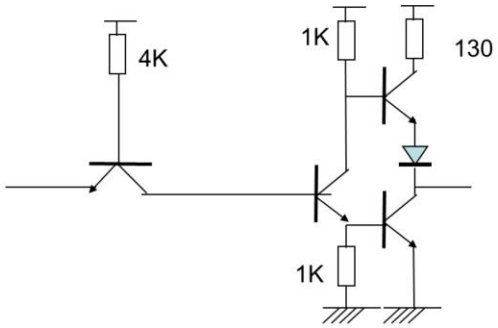


ncspice 入力デッキで立ち上がります。runで実行し、終わったらplot v(節点番号）で、見たい節点の番号を表示できます。この波形は左クリックで座標が表示され、右クリック→ドラッグで拡大画面が表示されますので、適当に遊んでみて使いかたを覚えてください。helpで他の機能を表示できます。抜ける時はquitと打ち込んでください。



dtlinv.cirはDTLのシミュレーション用入力デッキです。今度はダイオード、抵抗、BJTが使われます。BJTは戦闘がqで始まり、Collecter Base Emitterの順です。やや複雑ですが、問題ないと思います。

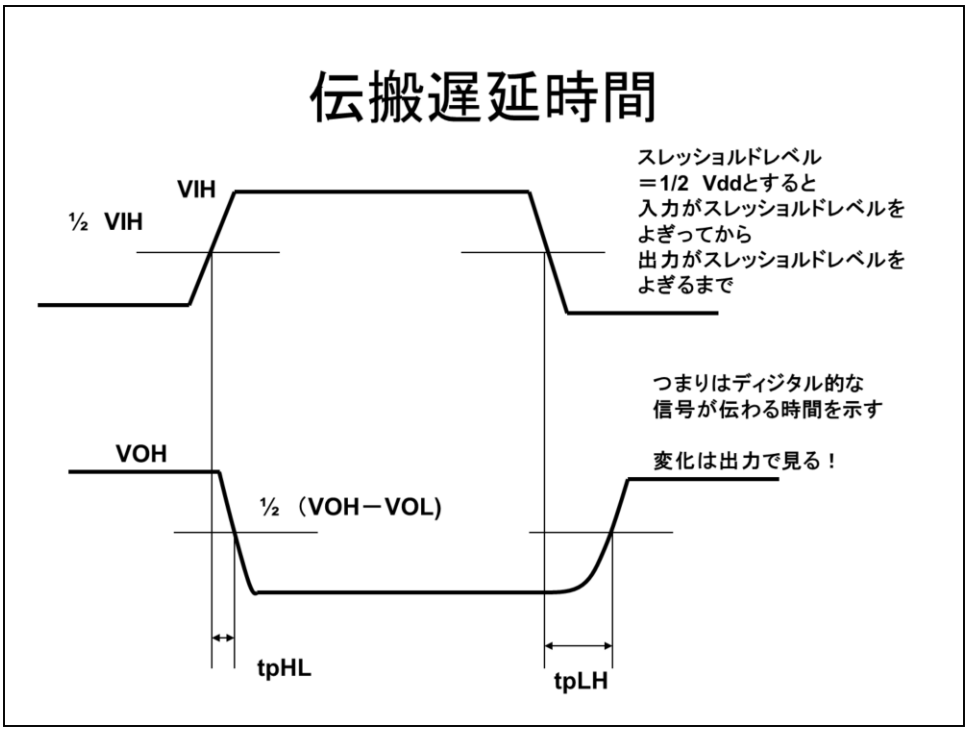
1入力のTTL



演習

- CMOSのNAND回路を構成し、シミュレーションして伝搬遅延時間（tpHL, tpLH)のおおよそを測定せよ。負荷容量は例題のNOT回路と同様とせよ
  - 2つの入力を時間的にずらして入れることに注意！
- マニュアル中のTTLのNOT回路（1入力のNAND)を構成し、シミュレーションせよ。CMOSのNAND同様、伝搬遅延時間（tpHL, tpLH)のおおよそを測定せよ。
  - 伝搬遅延時間については「規格表」の回を参照のこと
- 提出期限：8月1日、測定値のみ提出のこと。テキストデータの形式は問わないが、一般的に読めること。
- 提出先：keio.jp

では演習をやってみましょう。



ディジタル回路の動特性は、伝搬遅延時間で表されます。出力の安定レベルをVOL、VOHとしてます。立下り伝搬遅延時間tpHLは入力が（VOH-VOL）/2をよぎってから、これに反応して出力がVOHから（VOH-VOL)/2をよぎるまでの時間です。CMOSの場合、VOH=VDD、VOL=GNDと考えて良いので、入力がVDD/2をよぎってから、出力がVDDからVDD/2に変化するまでの時間と考えて良いです。立ち上がり伝搬遅延時間tpLHは、入力がVDD/2をよぎってから、出力が0VからVDD/2に変化するまでの時間です。VDD/2をスレッショルドレベルと考えて良いので、この値は入力がスレッショルドレベルをよぎってから、出力がスレッショルドレベルをよぎるまで、つまり、ディジタル的な信号の伝わる時間を示します。pはpropagation delayの頭文字です。変化の方向は出力で見ることに注意してください。tpHLとtpLHは同じと見なせる場合もありますが、素子によってはかなり違う場合もあります。