

注：著作権に関しては意味に相違がある場合は英文を優先します。

# 著作権表示

本和文マニュアルは肥後総合技術研究室が英文マニュアルから作成しています。  
本和文マニュアルを除くすべての権利は**Labcenter Electronics** .にあります。

あなたはどのような状態でも、一度に1つのPC上でソフトウェアを使うライセンスを購入しており、あなたの物としてソフトウェアを所有していません。無許可のコピー、ソフトウェアやドキュメンテーションの貸し付け、また再配布はどんな方法でも著作権侵害となります。ソフトウェア著作権侵害行為は違法となります。

PROSPICEには、バークレー大学が著作権を有するバークレー SPICE3F5 のソースコードが組み込まれています。ソフトウェアに含まれている各製造業社のスパイスモデルは、その各社に著作権があります。

## 警告

あなたはバックアップ目的のために、ソフトウェアのコピーを一つ行えます。しかしながら、あなたはソフトウェアに暗号化されたシリアライゼーション・システムが含まれていることの警告がなされます。ソフトウェアのどのコピーも、あなたへライセンス供給されたマスターディスクに、元をたどることができます。

プロテウスは又、ネットワーク上で、常に特定のライセンスキーを用い、1以上のコピーを妨ぐ為の特別なコードを含んでいます。そのために、複数台で同時に動作させる場合は、それぞれのコピーのためにライセンスキーを購入する必要があります。

## 免責条項

このソフトウェアパッケージ製品と任意の適切な特定目的の製品について、どの種類も無保証です。Labcenter Electronics 社とその従業員あるいは下請け業者のいずれも、パッケージの使用から生じた又、ソフトウェアのエラー、部品ライブラリ、シミュレータモデルあるいはドキュメンテーション、のために生じた、どのような直接的、間接的、あるいは重大な損害、又財務上の損失も責任を持ちません。

ユーザーは特に PROSPICE シミュレータを用いた、シミュレーションによって図面が完成した製品が生産されるとき、それが機能することを検証していないことを思い出させられます。この場合は量産によつて多数の製品を作り出す前に、1度限りのプロトタイプを作ることが常に最も良い方法です。PROSPICE に含まれている、製造業社のスパイスモデルは「現状のまま」を基本として供給されています、そして Labcenter とその製造業社は、いずれもそれらの正確さや機能性についてまったく保証しておりません。



# 目次

|                                 |    |
|---------------------------------|----|
| イントロダクション.....                  | 1  |
| 対話式チュートリアル .....                | 3  |
| 概要 .....                        | 3  |
| 回路図作成 .....                     | 4  |
| コンポーネントの配置: .....               | 4  |
| 移動と回転: .....                    | 4  |
| ズームとスナップング: .....               | 5  |
| 配線接続: .....                     | 5  |
| プログラムの作成.....                   | 6  |
| ソースリスト .....                    | 6  |
| ソースファイルの付加 .....                | 8  |
| プログラムのデバッグ.....                 | 8  |
| 回路のシミュレーション .....               | 8  |
| デバッグモード.....                    | 8  |
| ブレークポイントの設定 .....               | 9  |
| バグを見つける .....                   | 9  |
| 測定の取り方 .....                    | 10 |
| C / C++ でプログラムを書く.....          | 12 |
| サポートしているオブジェクトファイルのフォーマット ..... | 13 |
| アドバンスド・デバッグ.....                | 17 |
| 概要 .....                        | 17 |
| デバッグウインドウ .....                 | 17 |
| 診断構成 .....                      | 18 |
| 診断の構成 .....                     | 18 |
| シミュレーションアドバイザー .....            | 20 |

|                                      |           |
|--------------------------------------|-----------|
| シミュレーションアドバイザーを用いたデバイスのナビゲーション ..... | 20        |
| シミュレーションアドバイザーを伴ったネットのナビゲーション.....   | 21        |
| ハードウェアブレイクポイント .....                 | 24        |
| ハードウェアブレイクポイント .....                 | 24        |
| MPLAB IDEにおける動作 .....                | 25        |
| <b>グラフベース・チュートリアル .....</b>          | <b>27</b> |
| イントロダクション.....                       | 27        |
| 開始 .....                             | 27        |
| ジェネレータ.....                          | 28        |
| プローブ .....                           | 30        |
| グラフ .....                            | 30        |
| シミュレーション.....                        | 32        |
| 測定方法 .....                           | 33        |
| 電流プローブの使用 .....                      | 34        |
| 周波数解析 .....                          | 35        |
| スイープ変数解析 .....                       | 36        |
| ノイズ解析 .....                          | 37        |

# イントロダクション

このガイドの目的はイシスを操作するのに必要なテクニックを習得するのに、適度に複雑な回路を用いながら慣れ親しめるようにしています。チュートリアルは、コンポーネントの配置や配線などの最も簡単な話題から始めて、次により洗練された、新しいライブラリの一部を作成するなどの、編集機能を利用する様に構成しています。

すぐに何かを見たい人の為に、TRAFFIC.DSN は対話式チュートリアルで使用する完成された回路を含んでおり、そして ASIM1TUT.DSN はグラフィック・ベースのチュートリアルで使用する回路を含んでいます。これらの回路は他の 100 個以上のサンプルと共に、デフォルトで以下の SAMPLES ディレクトリにインストールされます。

C:\program files\Labcenter Electronics\Proteus 7 Professional\Samples  
上記の回路は、このチュートリアルのサブディレクトリのパス上に見いだすことができます。

ソフトウェアも含んだ完全なリファレンスマニュアルはまた、イシス「Help」メニューの「ISIS Help」からアクセスできることを知っておくことが重要です。このドキュメンテーションは実践的な入門書としてのソフトウェアを供給します。また、リファレンスマニュアルは、このガイドでカバーされていないより高度な機能の詳細な説明を含んで、ソフトウェア機能の完全な説明内容となっています。

また、web 上のサポートフォーラムから、プロテッスの一般的な事柄についての役に立つ内容やディスカッションを見つけることができます。

ディスカッション: <http://support.labcenter.co.uk>

リファレンスマニュアルを参照した後にもまだ質問や問題がありましたら、最終的に技術サポートのために認可されたディストリビュータに連絡をとるか、もしくは直接メールで、件名にあなたの顧客番号を入れ [support@labcenter.com](mailto:support@labcenter.com) まで送ってください。

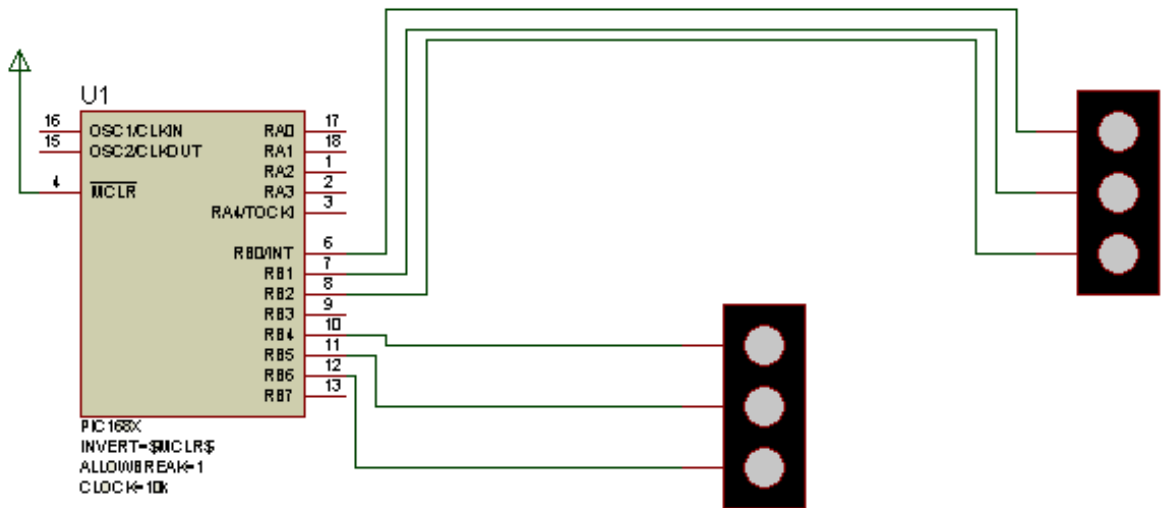


# 対話式チュートリアル

## 概要

このチュートリアルの目的は、単純な接続図の作成を通して、プロテウス VSM を用いてどのように対話形式のシミュレーションを実行するのかを示しています。「Active Components」と ISIS エディタのデバッグ機能を使用する事によって、接続図の基本的なレイアウトと、一般的な回路における処理の流れを習得できるでしょう。これらのトピック(題目)については、ISIS のマニュアルで詳しく述べられています。

私たちがシミュレーションに使用する回路は以下に示すように PIC16F84 マイクロコントローラと接続された2組の信号機です。



走り書きから接続図を描いても、又完成されたバージョンがプロテウスをインストールした「Samples¥Tutorials¥Traffic.DSN」に有りますので、それを用いてもよいでしょう。ISIS の全般的な操作に精通しているユーザーはこの完成されたデザインを用いて、マイクロコントローラ・プログラムのセクションに移るかもしれませんが・・・このデザインファイルは故意にエラーを含んでいることを覚えておいてください - 詳しい内容が書かれていますので読み続けてください。

もしあなたが ISIS に慣れていないのであれば、インターフェースと基本的な使用方法の両方について「ISIS Getting Started Documentation」と、オンラインリファレンスマニュアル(ヘルプメニュー - ISIS Help)で詳しく述べられていますので参照します。次のセクションでこれらの問題に簡単に触れますが、次に進む前にあなたはプログラムに精通しておく必要があるでしょう。

## 回路図作成

### コンポーネントの配置:

では新しい接続図のレイアウト上に、2組の信号機と PIC16F84 を配置して始めましょう。新たにデザインを始めるには、「*Component Icon*」(すべてのアイコンは、それらの使用を手助けするために、テキストによるツールチップと文脈依存型ヘルプの両方を有しています)を選択して、次にライブラリ・ブラウザを起動するために「*Object Selector*」の上にある「P」文字を左クリックします。ライブラリ・ブラウザはエディットウィンドウ上に現われるでしょう(より詳しい情報は、ISIS マニュアルの「Basic Schematic Entry」にあります)。

キーボード上の「P」ボタンを押して、「Key」ワード・フィールドに「Traffic」とタイプしてから、オブジェクトセクターで、接続する信号機を配置する場所でダブルクリックします。PIC16F84A も同様に行います。

図面の中へ「TRAFFIC LIGHTS」と「PIC16F84」の両方を配置入力したのち「Library Browser」を閉じ、オブジェクトセクターで「PIC16F84」を左をクリックします。

(高輝度でこの選択が強調され、コンポーネントのプレビューが画面の右上にある「Overview Window」に現われるでしょう)。次に接続図上にコンポーネントを置くために「Editing Window」上で左クリックします — この方法を繰り返して、接続図上に信号機を2セット配置します。

### 移動と回転:

現在接続図上にビルディング・ブロックを持っていますが、それはまだ理想的な位置に置かれていません。コンポーネントを動かすために、マウスをその上に置いて、左クリック(コンポーネントが明るくなります)します、そして左ボタンを押し続けながら希望する位置まで持ってゆきドラッグ(コンポーネントの外観がマウスポインタの動きに追従して表示されます)してください。左マウスボタンをリリースして外枠を表示している間は、マウスの移動に従ってコンポーネントは指定された位置まで移動するでしょう。この時点ではコンポーネントはまだ明るく表示されていることに注意してください — コンポーネントを通常の状態にもどすためには「*Editing Window*」の任意の何もない場所(空きエリア)をクリックします。

コンポーネントを回転させるのは、マウスを上に入れて右クリックし、そして表示されるコンテキストメニューからローテーションコマンドの1つをクリックします、ここではコンポーネントを90度回転させてます — 必要に応じて繰り返しましょう。コンポーネントを元の状態に戻す為に終了させる為、接続図上の空エリアをクリックします。

接続図をバランス良くするのに(おそらくサンプルに基づいて)、必要に応じてコンポーネントの移動と回転をしましょう。もし問題が発生した場合は ISIS のマニュアルーISIS チュートリアル、を確認するのが良いでしょう。

ここではシミュレーションが目的ですので、接続点に関連した2Dグラフィックスの関係は無視し、シミュレーション回路を作成することに専念します — 興味を持っている人たちのために、ISIS のグラフィックスに関する全ての能力についての説明は「2D Graphics」に書かれています。

### ズーミングとスナップピング:

接続図で配線のために、指定した領域をとズームできることは有効な機能です。中央のマウスボタンを回し、F6 キーを押す、もしくは「Zoom In」アイコンを用いると、マウスの現在位置の周囲がズームされるでしょう、あるいは、「シフトキー」を押し下げつつマウスの左ボタンでボックスをドラッグしても、ドラッグ領域内が拡大表示されるでしょう。再び画像を縮小するには、マウスを「後方へ回転」させるか、F7 キーを押すか、「Zoom Out」アイコンを使用するか、もしくは接続図全体を見ることができるまで画像を徐々に遠ざけ、希望する縮小サイズにして、F8 キーを押すか、もしくはマウスホイールを外側や内側へ回転させて必要な領域にズームさせます。「View Menu」を用いて対応するコマンドにアクセスできます。

ISIS のマニュアルにある「Editing Window」でズーミング(拡大/縮小)とスナップピング(つかむ)の詳しい情報を見つけることができます。

### 配線接続:

回路の配線をする最も簡単な方法は、「Tools Menu」の「Wire Auto Router」オプションを使用することです。これを有効にして下さい(メニューオプションで左ボタンを押し表示させます)。詳しい情報に関しては、ISIS マニュアルの「Wire Auto Router」を見てください。PIC のすべてのピンがはっきりと確認できるまで拡大し、次にピン 6(RB0/INT)の終端上にマウス・ポインタを置きます。マウスカーソルは緑色のペンに変化するはずですが、この表示はピンに配線を接続するのにマウスが正しい位置に置かれている事を示しています。マウスを左クリックして配線を開始しマウスを動かして、信号機セットの内の 1 つの赤信号のピンに接続します。再び緑色のペンカーソルにするのに左クリックして、配線を完了させます。この過程を繰り返し、サンプル回路の両信号機の配線を行います。

配線過程で考慮する価値のある以下の 3 つ程度のポイントがあります:

- あなたはどのモードでも配線することができます — ISIS はあなたが何をしているかをかしこく理解しています。
- 使用可能であるとき、「ワイヤー オートルーター」(自動配線)はおよそ障害物をさけて、接続間の一般的な合理的経路を見いだすでしょう。これは一般的なルールとして、あなたが行う必要がある全ては両方の接続終端で左クリックするのみであり、ISIS 側でそれらの間の配線を行うことを意味しています。
- ISIS は配線している間、「Editing Window」の縁をそっと突くことで自動的に画面を動かすでしょう。これはあなたが最適なレベルに画像を拡大しておいて、目的部品について大体の位置を知っておけば、画面の端を部品が中に入ってくるまで、突くことで画面を移動できる手段があると言う事です。またその代わりにワイヤを配置している間(マウスの中ボタンを使用し)、拡大や縮小で使用する事もできます。

最終的に、我々は動力ターミナルにピン4を針金でとめなければなりません。ターミナルアイコンを選択して、そしてオブジェクトセレクターで「力」を強調します。今適当なスポットを左クリックして、そしてターミナルを置いてください。適切なオリエンテーションを選択して、そして前と比べて同じテクニックを使ってターミナルに4をくぎ付けにするよう電報を打ってください。

配線についての、さらに役立つ情報が、ISIS リファレンスマニュアルの次の場所にあります:

- i** この時点で、回路の完成できているバージョンを読み込むことをお勧めします — これは、もしあなたが作成したバージョンが、私達のものとは異なった部分がある場合に混乱が生じずるのを避けるためです！ 同様に、もし PIC マイクロプロセッサモデルのライブラリを購入していないのであれば、次に進むために事前に準備されたサンプルファイルをロードしなくてはなりません。

## プログラムの作成

### ソースリスト

私達はチュートリアルのもので、PIC で信号機をコントロールすることができるよう、次のプログラムを準備しました。このプログラムは「TL.ASM」と呼ばれるファイルで提供しており、それは「Samples\Tutorials」ディレクトリにあります。

```
LIST      p=16F84 ; PIC16F844 is the target processor

          #include "P16F84.INC" ; Include header file

          CBLOCK 0x10 ; Temporary storage
            state
              11,12
          ENDC

          org      0 ; Start up vector.
          goto    setports ; Go to start up code.

          org      4 ; Interrupt vector.
halt      goto    halt ; Sit in endless loop and do nothing.

setports  clrwf   ; Zero in to W.
          movwf  PORTA ; Ensure PORTA is zero before we enable it.
          movwf  PORTB ; Ensure PORTB is zero before we enable it.
          bsf    STATUS,RP0 ; Select Bank 1
          clrwf ; Mask for all bits as outputs.
          movwf  TRISB ; Set TRISB register.
          bcf    STATUS,RP0 ; Reselect Bank 0.

initialise clrwf ; Initial state.
          movwf  state ; Set it.

loop      call    getmask ; Convert state to bitmask.
          movwf  PORTB ; Write it to port.
          incf  state,W ; Increment state in to W.
          andlw 0x04 ; Wrap it around.
```

```

movwf  state          ; Put it back in to memory.
call   wait           ; Wait :-)
goto   loop           ; And loop :-)

; Function to return bitmask for output port for current state.
; The top nibble contains the bits for one set of lights and the
; lower nibble the bits for the other set. Bit 1 is red, 2 is amber
; and bit three is green. Bit four is not used.
getmask movf  state,W          ; Get state in to W.
addwf  PCL,F          ; Add offset in W to PCL to calc. goto.
retlw  0x41          ; state==0 is Green and Red.
retlw  0x23          ; state==1 is Amber and Red/Amber
retlw  0x14          ; state==3 is Red and Green
retlw  0x32          ; state==4 is Red/Amber and Amber.

; Function using two loops to achieve a delay.
wait   movlw  5
movwf  l1

w1     call   wait2
decfsz l1
goto   w1

return

wait2  clrfsz 12
w2     decfsz 12
goto   w2
return
END

```



実際には、後で述べますが上のコードには手の込んだ誤りがあります…

### ソースファイルの付加

次のステージでは、図面にプログラムを付加しシミュレーションができるようにします。私たちは「Source Menu」にあるコマンドを通してこれを行います。では「Source Menu」に行き、そして「Add/Remove Source Files」からコマンドを選択しましょう。「New」ボタンを押します、「Samples\Tutorials」のディレクトリに行くまで移動して、TLASM ファイルを選択します。クリックしてオープンすると、ファイルはドロップダウン・リストボックスの「Source Code Filename」に表示されるはずで

私たちは、次にファイルのコードを生成するツールを選択する必要があります。この場合は MPASM ツールで十分でしょう。このオプションはドロップダウン・リストボックスで利用可能にしておく必要がありますので、左クリックでよく使う種類のツールとして、それを選択するでしょう。(新しいアセンブラか、コンパイラを使用する場合には、最初に「Define Code Generation Tools」コマンドを用いて登録しておく必要がありますので注意が必要です)。

最終的に、どのファイルをプロセッサで動かすことにするのかを指定する事が必要です。ここでの例では tl.hex(ヘキサファイルは、tl.asm を MPASM でアセンブルする事で生成)となります。このファイルをプロセッサに付加するために、接続図の PIC の部品上で右クリックし、次いで部品上にて左クリックします。これにより「Program File」の領域を含んだ、「Edit Component」ダイアログフォームが現れるでしょう。その中に tl.hex がまだ指定されていない場合は、手動でファイルのパスを入力するか、もしくはファイル名の入力部分の右にある「ホルダ」選択ボタンを通してファイルのある位置まで移動します。そうして動作する様にヘキサファイルを指定した後、対話フォームを抜けるために「OK」ボタンを押します。

私達は今接続図にソースファイルを付加して、そしてどの「Code Generation Tool」を使用するのかを明示しました。ソースコードのコントロールシステムにおいてさらに詳しい説明が プロテウス VSM のオンライン・リファレンスマニュアルの中にあります。

## プログラムのデバッグ

### 回路のシミュレーション

回路をシミュレーションするために、画面下部の左にあるアニメーションパネルの「Play Button」上でマウスを押します。ステータスバーはアニメーションが有効であるときに現れるでしょう。そして、1 つの信号機のライトは緑点灯して、もう片方は赤く点灯する事によって、接続図のピンの論理状態を見ることができ事に気付くはずで

### デバッグモード

回路のデバッグをするために、現在動作しているシミュレーションを停止します。いったんそうすると、CTRL+F12 を押すことでデバッグを開始することができます。2 つのポップアップウィンドウが表示されるでしょう - 1 つは現在のレジスタの維持した値の表示と、もう一つはプログラムのソースコードの状態を表示します。他の多くの情報を表示するウィンドウと共に「Debug Menu」からでも、これらを有効にすることもできます。私達はまた、状態変数における適切な変化がモニターできる「Watch Window」を有効にしたいものです。これらの特徴の十分なディスカッションについてはオンライン・リファレンスマニュアルにて行われています。

今のところ、左端に「Source」ポップアップ通知の赤い矢印が集まります。これは、プログラムカウンタの現在の位置を示す高輝度のラインと共に表示となります。ここにブレークポイントをセットするには「ENTER」キーを押します(ブレークポイントは高輝度のライン位置にいつでも設定されます)。ブレークポイントを削除したい場合は、再び「ENTER」キーを入力する事で、可能ですが、今は設定されたままにしておきましょう。

### ブレークポイントの設定

プログラムを見る方法として、それ自身の繰り返しサイクルにおいて溯って見ることができます。この繰り返しサイクルを開始する前にブレークポイントを設定しておく事が良い考えとなります。マウスによって(アドレス 000E に)高輝度ラインを持ってきて、次に F9 を押すことでこれができます。そしてプログラムを走らせるために F12 を押します。あなたはデジタルブレークポイントに到達したことを示している「ステータスバー」とプログラムカウンタ(PC)のアドレス値をメッセージとして見れるでしょう。これは最初に設定したブレークポイントのアドレスと一致するはずです。

「Debug Menu」にデバッグキーのリストがありますが、多くのパーツはプログラムを通して、シングルステップの F11 から使用するでしょう。F11 キーを押して見てください、そして左側の赤い矢印が次のインストラクションへ移動したことを確認してください。この事は「clrw」インストラクションが実際に実行されてから停止した事です。これは「Registers Window」のWレジスタを見て、ゼロにクリアされたことに気づく事で、これを確認できます。

私たちがいま行う必要があるのは、次のインストラクションを実行する事により発生する結果の予測と、テストして実際に発生する結果を確認する事です。例えば、次のインストラクションは「W」レジスタの内容を PORT A へ移動させます。言い換えれば「Port A」はクリアされなければなりません。このインストラクションを実行し、「Register Window」をチェックしてそうなることを確かめます。このようにして 2 番目のブレークポイントに達するまで連続動作し、両方のポートへ出力できる状態でクリアされ(TRISB レジスタによって決定)、そして状態変数は正しく 0 に設定された事を確認します。

これがファンクション・コールであり、機能の「Stepping Over」(F10 キーを押す事)も持っています、しかしながら完全なチェックの為に、それぞれのインストラクションをステップごとにチェックするのが良いでしょう。次に「F11」を押すと、「getmask」機能によって最初の実行可能なラインまでジャンプするでしょう。ステップしながら前へ進み、私達はムーブ動作が成功して、正しい場所にある「land」により、ルックアップテーブルにゼロのオフセット値を加えました。メインプログラムに戻るときに、予想しているマスクを持っています。さらにシングルステップを進め、そしてポートをマスクすることで接続図に関する正しい結果を見ることができます。再びシングルステップを行えばステートを進めると、W レジスタの値が、正しく1つ増加していることが「Register Window」によって判ります。

それが 3 以上増加されるとき、シングルステップはステートを包むように設計された指示によって、ゼロへと進みます。この動作が有るべき状態で機能しているかを「Watch Window」で見る事ができます。ここでのステートは、次のループが正しく実行されるようマスクして、1 が明確に増加されたステートとなるべきです。

### バグを見つける

さらに詳しい調査によって、3 の代わりに 4 で、アンドを取っている事で、問題が発生しているのがわかります。私たちが欲しい状態は 0、1、2、3 です、そしてこれらのいずれでも、4 とアンドをとる時に 0 となります。これがシミュレーションを走らせるときに信号の状態が変化しない理由です。解決策は単純で、問題の発生しているインス

トラクションの AND をとるステートを4の代わりに3に変えることです。ステートが3まで増加して「Wレジスタ」が4へ増加したらステートは終了し0へ行きます。代替方法としての簡単なケースのテストは、「W」レジスタが4になるとそれをリセットしてゼロにすることです。

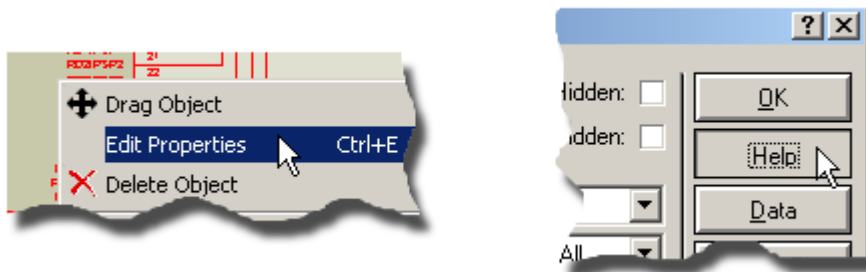
この簡単な例では、プロテウス VSM で利用可能な数多くの追加機能の内の基本的なデバッグ技術について使用しました。これらの機能は次の話題において、ファームウェアが高水準言語で書かれ、さらに詳細に述べられている時でオンライン・リファレンスマニュアルを参照しているとき、この機能のいくらかが見いだせます。

### 測定の取り方

対話型のシミュレーションで測定値を得るために、シミュレーションの前に接続図上に適切な仮想機器を配置し配線します。機器はシミュレーション中において、持続的なフィードバックとバーチャルシステムからの質問に対して応答するでしょう。現在利用可能な機器の全リストを以下に示します。

- オシロスコープ
- ロジックアナライザ
- カウンター / タイマー
- シグナルジェネレータ(信号発生器)
- パターンジェネレーター
- デュアルモードの I2C プロトコルアナライザ
- デュアルモードの SPI プロトコルアナライザ
- TTY / RS232 仮想端末
- DC / AC 電圧形と電流計
- 電圧と電流のプロローブ

それぞれの機器は、コンポーネント(既に配置されている)の上で右クリックすると現れるコンテキストメニューのヘルプファイルがありますので、このコンテキストメニューから「*Edit Properties*」を選択して結果として生じるダイアログフォームから「Help」ボタンを右クリックします。このヘルプファイルは機器のオペレーションにおける全てのモードを説明しており、正しいオペレーションのために有効な参考情報を提供します。



### 仮想機器のヘルプファイルを起動。

すべてのパルス、シグナル及び波形を、接続図上の配線によって明確にすることが重要です。コンポーネント間において、ごまかしや隠されたコミュニケーションは必要ありません。これは設計のある時点における正当性を検証する為に、接続図上の任意の場所にどんな機器でも配置し、ワイヤで接続できることを意味します。

次のイラストでは、とりわけエキサイティングなわけではありませんが、現在の接続図上にロジックアナライザの配置を取り上げます。

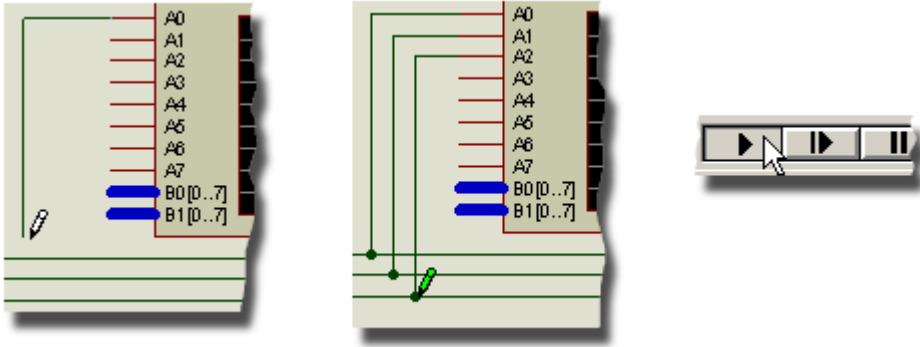
#### 仮想機器の配置と使用

1. 「Instrument」アイコンで選択し、そして「Object Selector」から配置のための機器を選択します。
2. 「Editing Window」で左クリックし配置モードに入って、マウスを希望する場所まで動かして、そして配置を確定するために再び左クリックします。



3. 機器のピン上から線(ワイヤ)を出し、接続図上の必要な線と配線します。

4. 「Animation Control Panel」の「Play」ボタンを押してシミュレーションを開始し、データをモニターします。



もちろんですが、測定やタイミング図のプリント等の為に、カーサーをドラッグ可能です。— もっと多くの情報は、ロジックアナライザのヘルプファイルを参照して下さい。

## C / C++ でプログラムを書く

簡単にするために、上のチュートリアルではアセンブリ言語で書かれたプログラムを中心に置いています。しかしながら実際には今日のほとんどの技術者が、**C/C++** コンパイラを使用してファームウェアを書くでしょう。プロテウスはそのようなプログラムのデバッグは、コンパイラが出力するオブジェクトファイルの出力を用いて完全にサポートしています。実際にはそのようなファイルは、詳しい情報を提供するのでデバッグの機能が通常高められています— 「Advanced Debugging Techniques」のトピックスを、概要やオンライン・リファレンスマニュアルのために見てください。

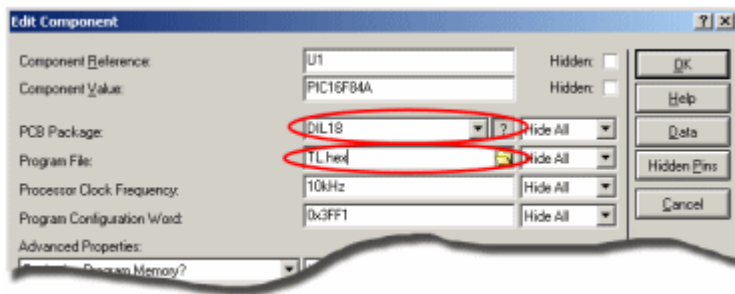
### Proteus VSM を使用してCのプログラムをデバッグする:

1. まずあなたのプロジェクト用に、ISIS 接続図ファイルとプロジェクト・ソースを含んだ一つのフォルダを作ります。
2. コンパイラがオブジェクトファイルを生成できる状態で、ソースファイルをコンパイルします。
3. 接続図上のプロセッサ・コンポーネントをエディットし、クロック周波数の指定及びコンパイラによって生成したオブジェクトファイルをプログラムとして設定します。
4. アニメーションコントロールパネル上の「PAUSE」ボタンを使用してシミュレーションを開始します。

5. 詳細は以前に記述していますが、「Debug」メニューから、ブレークポイント、シングル・ステップ等を設定し、ウインドウでデバッグを開始します。

**i** 追加のデバッグ機能が、オブジェクトファイルで提供された情報(例えば変数ウインドウ、のようなもの)によって利用可能かもしれません。どうぞ以下の「アドバンスデバッグ」テクニクにあるトピックス、あるいは詳細が書かれているオンラインリファレンスマニュアルを見てください。

**i** クロック周波数は常に接続図のコンポーネントで設定します。クロックピンにオシレータを配線すると、意味もなくのCPU資源を多量に費やします。



CPUコンポーネント、クロック周波数指定、そしてプログラム属性の編集

### サポートしているオブジェクトファイルのフォーマット

C / C++ でオブジェクトを書くとき、接続図のコンポーネントに供給するファイルは、ターゲットプロセッサとプログラムを作成するために使用しているコンパイラの種類の方によって決まります。次の表は色々な種類のターゲットプロセッサの為にサポートしているフォーマット(執筆時点)の概要です:

|              | COD | COFF | ELF/DWARF2 | UBROF8 | OMF51 | HEX |
|--------------|-----|------|------------|--------|-------|-----|
| PIC ファミリ     | ✓   | ✓    | ✗          | ✓      | ✗     | ✓   |
| アトメル AVR     | ✗   | ✓    | ✓          | ✓      | ✗     | ✓   |
| フリースケール HC11 | ✗   | ✗    | ✗          | ✓      | ✗     | ✓   |
| ARM7/LPC2000 | ✗   | ✗    | ✓          | ✓      | ✗     | ✓   |
| 8051         | ✗   | ✗    | ✗          | ✓      | ✓     | ✓   |
| ベーシックスタンプ    | N/A | N/A  | N/A        | N/A    | N/A   | N/A |

**i** 物理的なチップをプログラムするのと同じバイナリ(HEX)ファイルを、プロテウスが用いる事が出来る事は、デバッグ情報は使用しないという事実と、プログラムはシミュレーションされますが、デバッグはされないことを意味しています。このため他の利用可能なフォーマットについて特に推奨はしていません。

- ❶ COD シンボリックデバッグのデータ形式における限界は、この結果 VSM デバッグサポートがマシンコードと指定したメモリ位置にステップするのに限られ、ソースレベルでのステップと変数表示がサポートされないことを意味します。標準的なコンパイラでは推奨オプションとして、COFF ファイルを生成するでしょう。
- ❷ ベーシックスタンプのプロセッサでは統合化されたトークナイザが含まれており、必要とされるすべてがプログラムのソースファイル自身で供給されます(コンパイルではありません)。

多数のマイクロコントローラのサンプルデザインが、C / C++ で書かれたソフトウェアで供給されており、前に述べたように動作をします。たいていのコンパイラはコマンドラインから動作させることができ、ISIS ではソースコードコントロールシステムでこれを動作するよう構成し、コンパイラの IDE に慣れるためには、ここでの概要を説明した単純なステップを行うことで維持されます。





# アドバンスド・デバッキング

## 概要

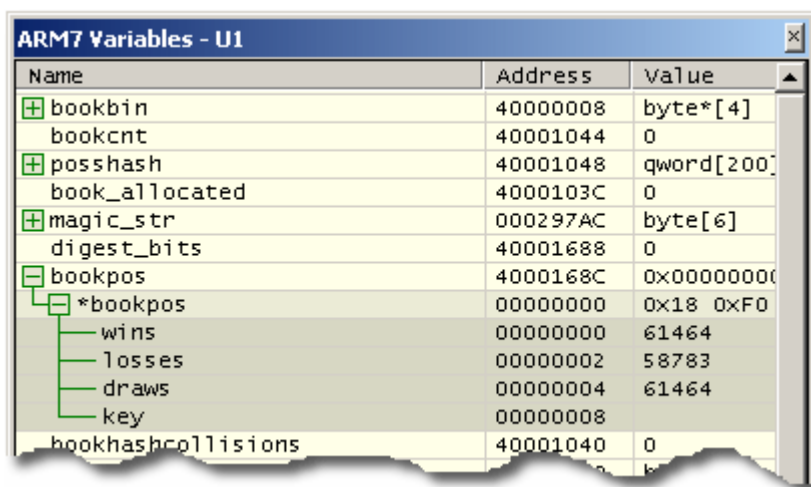
ドキュメントのこの章は、マイクロコントローラのデザインを伴って動作する上で、追加して使うことが可能なデバッキング手法を述べます。前の章で述べた「スタンダード」な手法に加えて、プロテウス VSM はハードウェアやファームウェア設計の双方における誤りを、速やかに発見するのを手伝うための多くの進んだ手法と機能を提供します。これらについて以下に手短かに述べます — これらの手法は、ソフトウェアを伴って前もって提供されているマイクロコントローラのサンプルデザインで試すことが可能です。サンプルディレクトリのデフォルトパスは次です：

```
c:\program files\labcenter electronics\proteus 7 professional\samples\
```

## デバッキングウインドウ

各種のデバッキングウインドウがシミュレーションの間、ISIS の「Debug」メニューからユーザーへ渡されます。これらは単純なメモリダンプからレジスタの表示、スタックのモニター、ウォッチ及び変数のウインドウまであります。特に自走シミュレーションの間可視と、そしてウォッチポイントの条件（もしくは条件付きブレークポイント）指定が、できるウォッチウインドウが特に重要です。

ウォッチウインドウ及び変数ウインドウの両方でスムーズに複合タイプを処理でき、そして SFR ビット情報あるいはポインタ、構造体(struct)、列挙体(enum)情報、を表示するために便利な様拡張されるでしょう。



| Name               | Address  | Value      |
|--------------------|----------|------------|
| bookbin            | 40000008 | byte*[4]   |
| bookcnt            | 40001044 | 0          |
| posshash           | 40001048 | qword[200] |
| book_allocated     | 4000103C | 0          |
| magic_str          | 000297AC | byte[6]    |
| digest_bits        | 40001688 | 0          |
| bookpos            | 4000168C | 0x00000000 |
| *bookpos           | 00000000 | 0x18 0xF0  |
| wins               | 00000000 | 61464      |
| losses             | 00000002 | 58783      |
| draws              | 00000004 | 61464      |
| key                | 00000008 |            |
| bookhashcollisions | 40001040 | 0          |

### 変数ウインドウの複合タイプへの拡張



さらに多くの実用的なコンフィギュレーション例を有した、デバグウインドウについての情報がオンラインリファレンスマニュアル(ISIS ヘルプメニュー — VSM ヘルプ)にあります。

## 診断構成

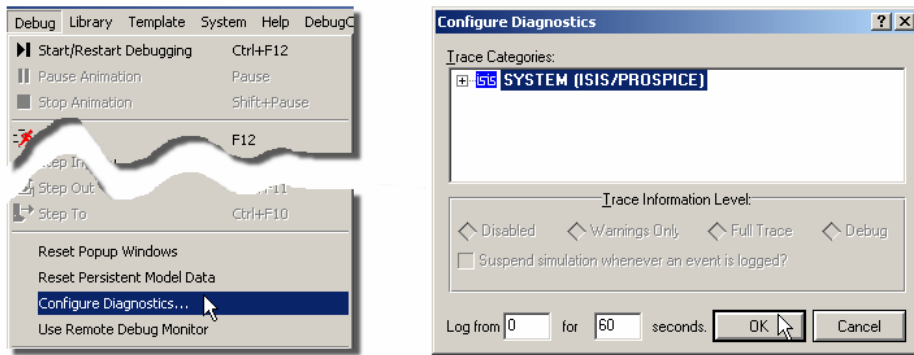
プロテウス VSM はシステムオペレーションの障害検出と確認ができる非常に有効な、広範囲の診断ツールやトレースモードを含んでいます。またその構造はすべての指定した動作がシミュレーションの期間中記録され、そしてシミュレーションアドバイザー上にテキストのレポートフォーマットで表示されます。メッセージのフォーマットはメッセージを記録した時の時間を含んで、メッセージの概要、コンポーネントが発行するメッセージ、そしてオプションとして文脈依存型ヘルプでのさらに詳しい情報を含みます。明らかにする部分の方法を考え、そして内部動作の実行や、システムを通して、データのトレーサビリティを提供する部分を考慮します。

システムレベルシミュレータにおいては、プロテウス VSM が有している診断モードはマイクロプロセッサだけではなく、適切な周辺モデル(液晶ディスプレイ、I2C メモリー、温度制御素子、など)も含んで、これらのトレースモードの細かな設定が、診断の対話フォームで設定することによって可能となります。また各デバイスにおいて見たいメッセージのレベルを制御できます。例えばあなたはマイクロコントローラおよび SPI EEPROM において、オンボード上の SPI 周辺装置との完全なメッセージのトレースを実行したいかもしれませんが、しかし他のシステム部品においては、関係するワーニングメッセージを受けることができるのみです。

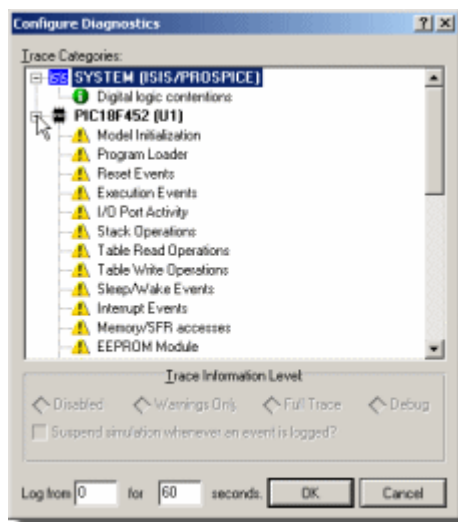
### 診断の構成

診断が有効な状態で、シミュレーションを構成する方法:

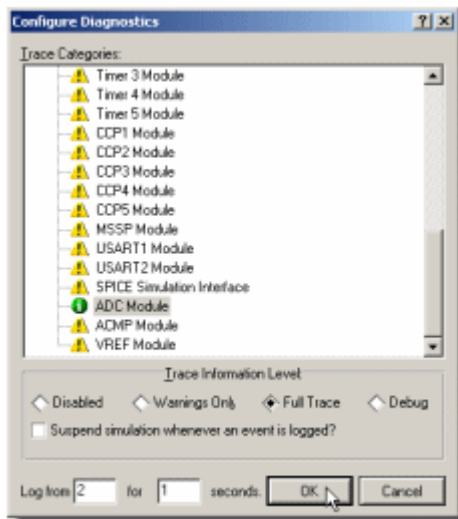
- 「Debug」メニューから「Configure Diagnostics」ダイアログフォームを開始。



- 診断したい項目を見つけて可能にするために、対話形式により、ツリーを拡張します。



- 目的とする項目上で左クリックして選択し、そしてトレースレベルを完全なトレースへと変更します。また、アームタイムと実行時間の両方について、診断がアクティブになる時間間隔を制御するよう設定できます。



- 他の必要とする項目についての処理を繰り返した後、対話形式を終了します。

トレース診断を構成してシミュレーションを走らせる時には、アーム時間において有効になり、指定した期間走るでしょう、そしてシミュレーション・アドバイザー上にすべての結果が表示されるでしょう。

- ① トレース診断を可能にするとシミュレーションにかなりの負荷がかかります。しかしながらそれらは不明瞭な問題を解析するために標準的に用いられ、そしてデバッグ目的で用いられるときには、シミュレーションがリアルタイムで走らない事は問題ではありません。

## シミュレーションアドバイザー

シミュレーションアドバイザーは、シミュレーションが走っている間に生成される、警告と診断メッセージのすべてのエラーの収納場所です。それはアニメーションコントロールパネルに一番近いステータスバーにある ISIS アプリケーションの下に存在します。シミュレーションが走っている間、ステータス表示は常に更新され、ほとんどのサーバータイプの記録されたメッセージ(エラー、警告、トレースメッセージ)とそのようなメッセージの数の両方を表示します。あなたは、シミュレーションの間いつでもシミュレーションアドバイザーを動作させる事ができます。あるいは実際にシミュレーションが走った後(すべてのメッセージは持続的です!)に、ステータスバー上で最小表示となっているアイコン上で、マウスを左クリックすることによって動作できます。

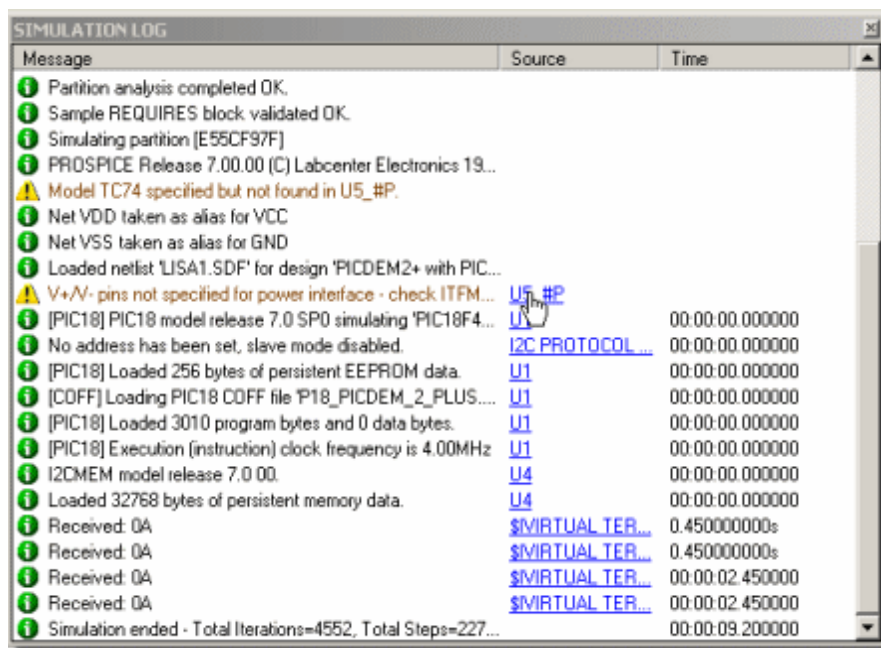


ステータスバーからシミュレーションアドバイザーの起動。

シミュレーションアドバイザーは重要なインディケータと併せて、すべてのメッセージの視覚表示、適切なメッセージと多くのエラーの文脈依存型ヘルプによる、ナビゲーション能力(ネットと接続図上の部品の両方)を提供します。

### シミュレーションアドバイザーを用いたデバイスのナビゲーション

物理的なコンポーネント(システムメッセージ以外のほとんど)に起因した全てのメッセージは、メッセージの右側に関連したソース欄を持ちます。これはメッセージの生成、およびソースリンクでマウスを左クリックするとシミュレーションアドバイザーが最小化し、ソース・コンポーネントのタグ付けとズームが行われ、接続図上にてコンポーネントを表示する役目をします。



シミュレーションアドバイザーにおけるメッセージソースのナビゲート。

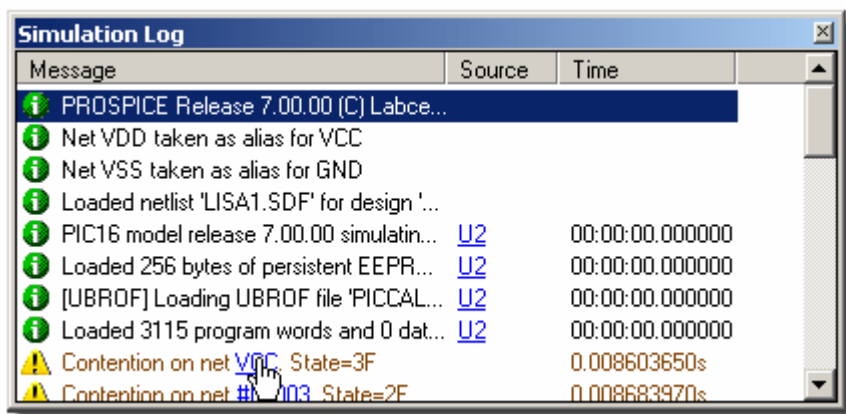
これは極めて有用であり、とても複雑なデザインにおいてデバイス周辺の問題を調べるための、ハードウェアデザインを検証することに役立ちます。

### シミュレーションアドバイザーを伴ったネットのナビゲーション

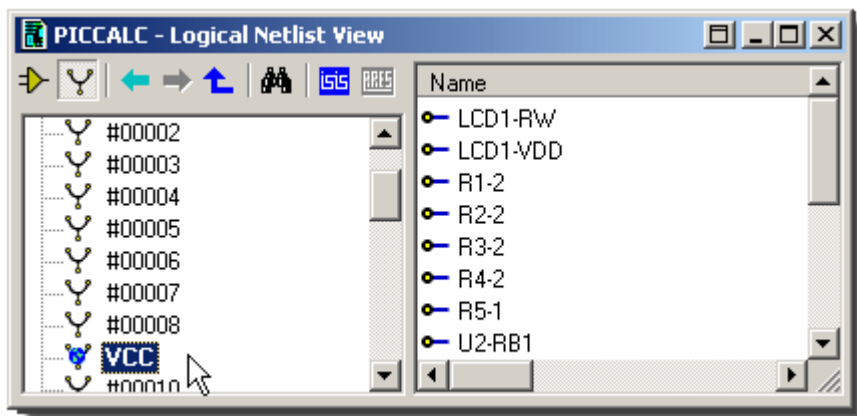
シミュレーションエラーについての最も失望させられるものの1つに、若干の問題(ネットの競合、SPICE 特異行列、等)があり、これらはネット関連と、そして具体的なコンポーネントではないために、接続図上の原因となる回路部分を分離することが極めて扱いにくいこと、があります。シミュレーションアドバイザーは、クリックしてデザインを導くことが適用できるメッセージに関して、「ハイパーリンク」を含むことによって、このタスクを単純化します。

シミュレーションアドバイザーからネットをナビゲートする:

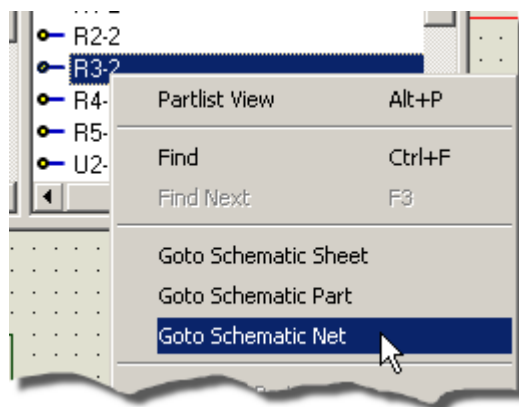
1. シミュレーションアドバイザーに含まれる必要なメッセージの「ネットリンク」をクリックします。



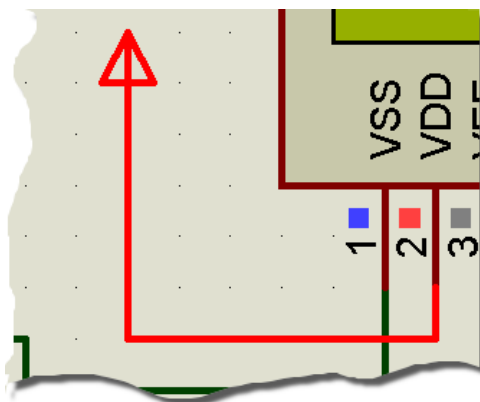
2. ステータスバーにシミュレーションアドバイザーは最小化して戻された後に、ISIS のデザインエクスプローラが起動され、左手パネルにネットのリストを表示し、右手パネルに原因となるネットのすべての接続(ピン)を表示します。



3. これらのピンの1つを右クリックして、結果として生じるコンテキストメニューから「Goto Schematic Net」オプションを選択します。



4. そうすると該当するネットが接続図上で強調表示されます。



❶ この方法はそんなに簡単には動かせませんが、接続図の部分に同等の回路(すでに存在するモデルに部品と一緒に配線する事によって、接続図形式にあるパーツの機能はモデル化される)によってモデル化されます、そしてモデルに存在する問題のネットは、ネットにナビケートすることはできません。そしてこれと、他の寄生的なケースも存在しますが、上で述べたテクニックは、ほとんど全て標準的な状況において動作しますので、そしてそのようなケースにおける強力な分析ツールとなります。

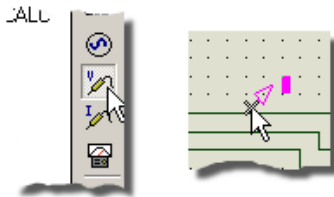
デザインエクスプローラは ISIS チュートリアル、及び ISIS リファレンスマニュアルで、共に詳細が述べられています。

## ハードウェアブレイクポイント

多くのコンポーネントにおいて、特殊な回路状態が発生した時、シミュレーションの一時停止状態を供給します。これらはシングルステップの機能と組み合わせ、特定の状態が発生するまでは通常の回路シミュレーションを用いておき、そして次に何が起こるのかを確認するためにシングルステップを行う事によって特に役に立ちます。

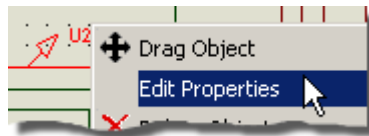
### ハードウェアブレイクポイントのセットアップ:

1. 電圧プローブをブレイクポイントのトリガーを希望する配線(バス)上におきます。

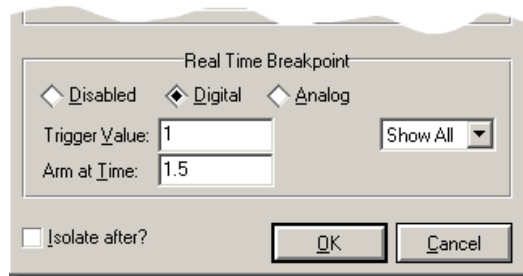


配線にプローブを付加。

2. プローブを右クリックし、表示するダイアログフォームから「Edit Properties」を選択します。



3. 対話形式(ダイアログフォーム)の一番下で、デジタルあるいはアナログと一致したネットのどちらかを選択してプローブを追加し、そしてトリガ値を指定します。デジタルネットと一本の線では、ロジック・ハイあるいはロジック・ローに対応して1あるいは0であり、アナログネットではそれは電圧値となるでしょう。またブレイクポイントが、休止時間後にアクティブになるのであれば、同じくアーム時間を指定することができます。



エディット・プローブ・ダイアログフォームの、トリガー値とアーム時間の設定。

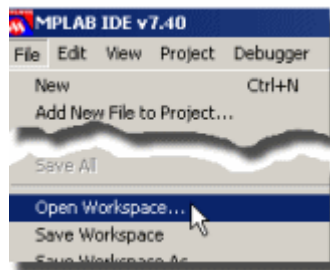
4. 「OK」を押す事でダイアログフォームを終了し、「PLAY」(又は CTRL+F12)でシミュレーションを開始します。

## MPLAB IDE における動作

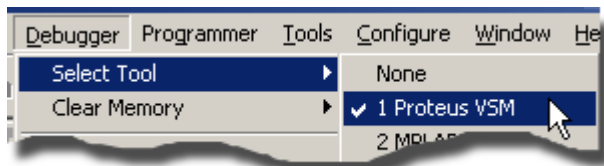
ラプセスター とマイクロチップテクノロジー社は単一の開発環境を組み込み設計に提供するため、幅広く協力して作業しました。MPLAB IDE の中でより便利にデザインを、書いて、テストし、進化させるのに「プロテウス VSM ビュワー」を呼び出すことが可能です。次のシンプルなガイドは MPLAB (V7.4 以降)とプロテウス VSM プロフェッショナル(バージョン7以降)がコンピュータ上にインストールされているものと想定しています。これはビュアーと共に動作する基本的な入門書であることに注意ください — MPLAB とプロテウスを用いた測定、デバッグ、等のさら詳細な考察については、ビュワーのヘルプアイコンから MPLAB のヘルプファイルを起動して確認できます。

### イクスプローラ 16 評価ボードで、プロテウス VSM シミュレーションを読み込み、走らせる:

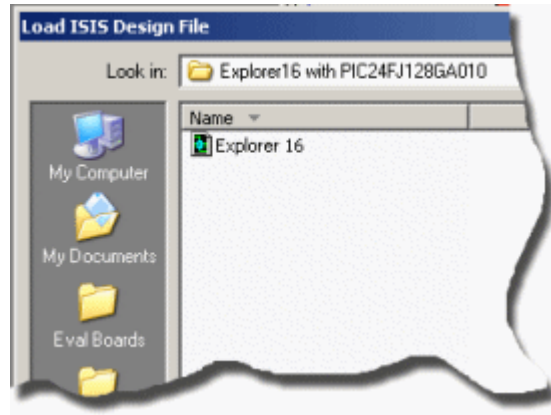
1. MPLAB IDE アプリケーションを起動して、ファイルメニューに行き、そしてオープンワークスペース・コマンドを選択します。そしてプロテウスのインストールされているサンプル・ディレクトリへ導きます。(デフォルトでは、`c:\program files\Labcenter Electronics\Proteus 7Professional\Samples\`) です、次に `VSM MPLAB Viewer\Eval Boards\Explorer16 with PIC24FJ128GA010` ディレクトリに行きます。最後に `PIC24ExplDemo.mcw` ワークスペースを開きます。



2. MPLAB IDE にある「Debugger Menu」に行き、「Select Tool」コマンド、次いで「Proteus VSM」を選択します。この構成で MPLAB でデバッグするためのツールが、プロテウスを使用するように設定されます。



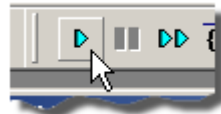
- 3.次に MPLAB にあるプロテウス VSM ビューワーを見ます。ビューワーのオープンアイコンを使って、現れたファイルセレクターから、「Explorer 16」接続図を選択します。



- 4.「Project Menu」から「Build All」を選択し、プロジェクトコードがコンパイルされて、確実にシミュレーションができる状態にします。
- 5.私たちは接続図とプロジェクトがありますので、シミュレーションが開始でできる状態です。MPLAB IDE の上部にある緑のボタンを用いてプロテウスシミュレーションと MPLAB を連結します。



- 6.この時点ではシミュレーションは時間ゼロで停止しています。シミュレーションのスタートは、MPLAB IDE の上部近くの右側にあるプレイボタンをクリックして開始します。これによりプログラムコードは実行され、VSM ビューワーにより、デザインに対するプログラムの結果を表示します。



7. MPLAB の VSM ビューアーを閉じシミュレーションを停止するためには、MPLAB IDE の上部の赤いボタンを使用します。



- i** 注記、PIC24 は大きなパーツなので接続図の2枚目のシートに配置します、そしてプロセッサとの接続はターミナル(端子)を経由して行われます。シートの呼び出しは、接続図の空いている領域で右クリックし必要なシートを選択します。

# グラフベース・チュートリアル

## イントロダクション

このチュートリアルの目的は、簡単なアンプの回路を用いて PROTEUS VSM におけるグラフフィック・ベースのシミュレーションを、どのようにして行なうのかを示すことです。それは以下に従って一步一步進めて行くことで、たやすく理解できる様にしています：

- グラフ、プローブそしてジェネレーターを配置します。
- 実際にシミュレーションを行ないます。
- グラフに結果の表示と、測定した値を表示します。
- 利用可能な分析タイプの調査。

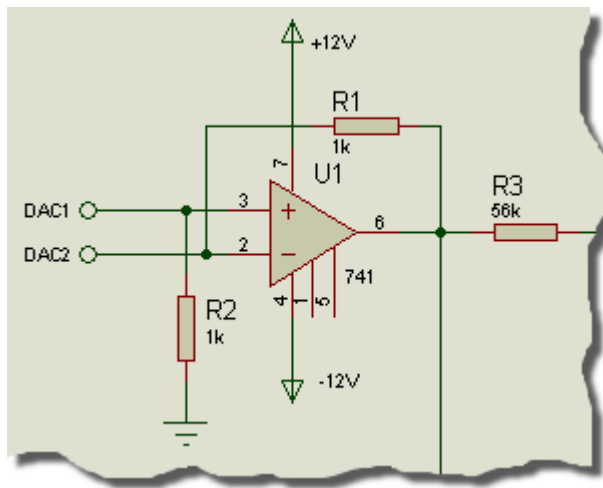
チュートリアルは ISIS の一般的な使用方法ではカバーしていません — それは部品を配置して、配線して、オブジェクトにタグを付けて、等と言う順序です。これは対話型のシミュレーションのチュートリアルにおいてカバーされており、詳細は ISIS マニュアル中に記述されています。あなたがまだ ISIS の使用方法に詳しくないのであれば、このチュートリアルを試みる前に、まず ISIS 精通する必要があります。

あなたが独自の回路でグラフベース・シミュレーションを試される前に、私たちはこのチュートリアルを通して正しく動作できるようになる事を強くおすすめ致します：概念を把握することによって、参考となる各章での題材を吸収することをさらに容易にし、そして長期に渡り、多くの時間の短縮と、フラストレーションを助ける事でしょう。

## 開始

次のページにあるように、シミュレーションしようとしている回路は 741 オペアンプを用いたオーディオアンプです。それは 5 ボルト単一電源で動作する、まれな構成の 741 を示しています。フィードバック抵抗 R3 と R4 で利得はおよそ 10 に設定されています。入力のバイアス部品 R1、R2 と C1 は、信号入力から切り離された非反転入力の仮想グラウンドを構築します。

通常の場合においては、回路の過渡解析を行ないます。この種の分析は、回路において多くの情報が得られ最も有用です。過渡解析でシミュレーションの記述が完成すると、他の分析フォームと比較されるでしょう。



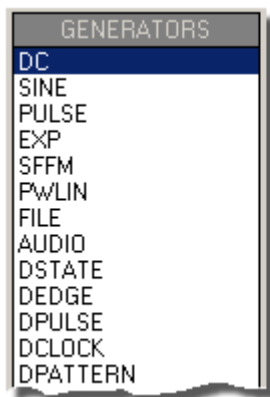
ISIS/TUT.DSN の項目にある画面コピー

もし望む場合は、独自の回路を作成することもできます、あるいはプロテウスのインストールされている中にある "Samples\Tutorials\ASIMTUT1.DSN" からデザインファイルを読みすることもできます。どちらを選択しても、ISIS の実行と、回路を描く事を確実に行ってください。

## ジェネレータ

回路をテストするためには、適当な入力を供給する必要があります。方形波出力を持つ電圧源を、テスト信号の為に用います。ジェネレーターから必要とする信号を発生させます。

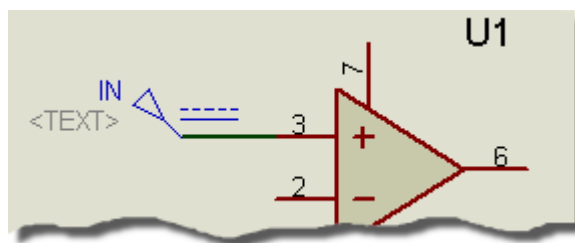
「Generator」アイコンをクリックします — オブジェクトセレクターは利用可能なジェネレーターの種類をリスト表示します。ここでのシミュレーションにはパルスジェネレーターを必要とします。パルスタイプを選択し、エディットウィンドウで「IN」ターミナルの右寄りにマウスを移動して、ワイヤ上で左クリックしてジェネレーターを配置します。



ジェネレータ(機器)は ISIS にある、他のほとんどの機器と似ています。同じ様な手順で配置の前に、ジェネレータの方向を合わせ、配置後に編集、移動、方向の変更または削除を適用します(詳しい情報に関してはオンライン・リファレンスマニュアルの「Generators and Probes」にあります)。

既存のワイヤ上に接続されると、ジェネレーターはシートに付加され通常の配線を行います。もしワイヤー外でジェネレーターを移動させると、ISIS はワイヤを外したいと考えてコンポーネントにとって問題ないようにワイヤーと一緒にドラッグしません。

ジェネレーターがどのように自動的にリファレンスを割り当てするのかに注意しましょう — ターミナル名称は IN です。ジェネレーターをオブジェクトに配線(あるいは直接既存のワイヤーの上に配置)される時はいつも、接続するネット名称が割り当てられます。もしネットが名称を持っていないなら、最も近いコンポーネントのピン名称が初期値として使用されます。

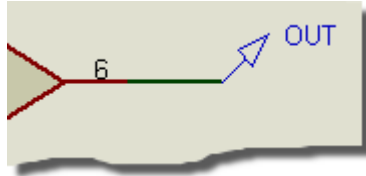


最後に、私達が必要とするパルス波形を決めるためにジェネレーターをエディット(編集)しなくてはなりません。ジェネレーターをエディットするためには、その上で右マウスボタンでタグ付けした後、左マウスボタンをクリックしてエディット・ジェネレーター・ダイアログフォームにアクセスします。「High Voltage」の場所を選択し、値を 10mV にセットします。また同様にパルス幅は 0.5s にセットします。

「OK」ボタンを選択して変更を確定します。この回路は1つのジェネレーターのみ使用していますが、使用する数の制限はありません。

## プローブ

私達の回路はジェネレーターを用いて入力定義されました、次は、必要なモニター位置にプローブを配置します。私達は明らかに出力に関心があります、その後入力をバイアスする事も、プローブの大切なポイントです。もし必要なら、さらに多くのプローブをいつでも測定のコアポイントとなる場所に追加する事ができます、その後シミュレーションを繰り返します。



プローブを配置するには「Voltage Probe」アイコン(もしアクシデントで現在のプローブに、これを選択しなかった場合 - 後で変更する方法を述べています)を左クリックします。プローブをワイヤ上に配置するか、または配置してから配線するか、はジェネレーターと同様な方法に行います。エディットウィンドウで、U1 のピン 3 の左側にマウスを移動し、ピン 3 から R2 に接続されているワイヤ上にて左クリックしプローブを配置します。プローブは直接ピン自体に配置することはできませんので、ワイヤ上に必ず置いてください。それが習得する名称は、最も近くに接続されているデバイスのピン名称である事を知っておいてください。次に2番目のプローブを、接続ドットとターミナル・ピンの間を接続しているワイヤ上にて、「OUT」ターミナルの左側で、左クリックすることにより配置します。

プローブのオブジェクトは、ISIS のジェネレーターやその他のオブジェクトと類似しています; プローブの配置前に、プレビューして方向を決める事や、又配置後にエディット、移動、再回転、あるいは削除(さらに詳しい情報は、プローブの章にあります)する仕方は、同様な手順となります。プローブは、リファレンス・ラベルを変更するために、エディットすることもあります。初期値で割り当てられた名前は、今までのケースの場合は良いのですが、役に立つ情報として、プローブにタグ付けをするときはボディやリファレンスラベルではなく、プローブに関する情報にすることです。

次にシミュレーションのために用意している回路に、結果を表示するグラフを配置する必要があります。

## グラフ

グラフはシミュレーションにおいて重要な役割を果たします: それらは結果を表示する役目だけでなく、実際にどんなシミュレーションが実行されるのかも定義します。1つ以上のグラフを配置して、グラフ上にどんな種類のデータ(デジタル、電圧、インピーダンス、等)を表示することを期待しているかを示すことで、ISIS はどんなタイプ、又はどんなシミュレーションのタイプを実行すべきか、そしてシミュレーションに回路のどんな部品を含める必要があるのかを理解します。過渡分析のためには「Analogue」タイプのグラフを必要とします。それは、デジタル解析から結果を表示するのに使用される「Digital」グラフと、本当に専門化した形式の過渡解析とを区別するために、むしろアナログと呼ばれます。「Mixed」グラフを用いる事で同じ時間軸に対して両方を表示させることができます。

グラフを配置するために、「Graph」アイコンを選択します。「Object Selector」は利用可能なグラフのタイプをリスト表示します。「Analogue」タイプを選択して、エディット・ウィンドウにマウスを動かして、左のマウスボタンを1回クリックした後、適当な大きさになるよう長方形を引き、グラフを置くため再びマウスボタンをクリックします。

ISISにおいて各種のグラフは、微妙な違いはありますが、ほとんど同じ様なオブジェクトとして振る舞います。チュートリアルで適切にそれらの特徴をカバーしていますが、グラフに関する、参照章は、読む価値が十分あります。通常の方法で、マウスボタンでグラフにタグを付けし、そして(左のマウスボタンを使って)グラフのサイズを変える、そして／あるいは、再配置するのに、ハンドルの1ヶ所もしくはグラフの全体をドラッグする、事ができます。

私たちは、ジェネレータとプローブをグラフに追加する必要があります。各ジェネレータには関連してプローブがありますので、入力波形を確認するために、直接ジェネレータにプローブを配置する必要はありません。プローブとジェネレータをグラフに追加するには3つの方法があります:

- 最初の方法は、グラフにプローブ/ジェネレータをタグ付けして、リリースします。— 正確にオブジェクトを配置し直しているように、です。ISISはあなたがプローブ/ジェネレータをグラフの上に置こうとしているのを見つけて、プローブ/ジェネレータを同じ基準にして元の位置に戻し、グラフにトレースを追加します。トレースは、アナロググラフの右もしくは左の軸に関連付けられて、そしてプローブ/ジェネレータは最も近くの軸に追加します。またプローブ/ジェネレータの付加において、新規トレースではいつでも、どの既存のトレースでも下部に付加されます。

プローブ / ジェネレーターをグラフに追加する2番目と3番目の方法は、共にグラフメニューの「Add Trace」コマンドを用います;このコマンドは常に現在のグラフ(1つ以上のグラフがあるときは、現在「Graph」メニューで選択されているものに追加します。)

- 「Add Trace」コマンドが、タグを付けられた、プローブ又はジェネレーターなしで、呼び出される時、「Add Transient Trace」ダイアログ形式が表示されて、図面上にある(他のシートのプローブを含めて)すべてのプローブのリストから選択することができます。
- 1タグ付けされたプローブ/ジェネレータがあれば、「Add Trace」コマンドの呼び出しは、クイック付加となり、現在のグラフにタグ付けされたプローブが迅速に付加されます;「No」オプションを選択すると、「Add Transient Trace」ダイアログフォームは、以前に説明したように呼び出されます。「Yes」オプションを選択すると、現在のグラフに、アルファベット順にすべてのタグ付けされたプローブ/ジェネレータを付加します。

私達はグラフにプローブとジェネレーターを迅速に付加するでしょう。個別にプローブとジェネレーターにタグ付けするか、あるいはさらに速く、回路全体のを覆ったタグボックスでドラッグします — クイック付加の特徴はプローブとジェネレーター以外のタグ付けされたオブジェクトはすべて無視されます。「Graph」メニューから「Add Trace」オプションを選択して、そしてプロンプトに「Yes」と入力してください。トレースがグラフに現れるでしょう(なぜなら1つのグラフのみであり、かつ最後に使用されているので、現在のグラフであると考えます)。現在のところ、トレースは名前(左の軸において)と、空のデータエリア(グラフ本体)から成り立ちます。もしトレースがグラフに現れない場合、ISISがそれらを描くにはあまりにも小さ過ぎるからでしょう。それにタグを付けてからコーナーをドラッグすることによって、それを十分に大きくするよう、グラフのサイズを変えてください。

それが起きると、トレース(アルファベット順に置かれた)は順番に現れます。私達はトレースをシャッフルすることができます。それをするためにはグラフにタグを付けないようにします、そして動かしたい、もしくは編集したいトレ

ースの名前を左クリックします。トレースはそれがタグ付けされたことを示すために高輝度になります。今度は左のマウスボタンを用いて、トレースをドラッグして上方や下方へ、あるいはトレースの編集(マウスを動かさずに左クリック)、そして右ボタンでトレースの削除(即ちグラフから、取り除く)ができます。すべてのトレースのタグを無くすためには、接続図の空きエリアで左マウスボタンをクリックします、トレースラベル上(これはトレースのタグ付けか、削除となるでしょう)では左のマウスボタンをクリックしないでください。

シミュレーションを始める前にセットアップする残りの1つがあります、それはシミュレーションの動作時間を決めるためのものです。ISIS はグラフの X 軸の終点の時間に従って、回路をシミュレーションします、新しいグラフに関して、それは 1 秒が初期値となっています。私達の目的においては入力矩形波は、可聴周波数を考慮した中のかかなり高い値とし、約 10kHz にします。これは1周期が 100us となります。グラフ上で右クリックして表示されるコンテキストメニューから「*Edit Properties*」を選択すると、その結果「*Edit Transient Graph*」対話フォームが表示されます。このフォームは、グラフのタイトル、シミュレーションの開始と終了の時間(これは X 軸の左端と右端に対応しています)の指定、左と右の軸ラベル(これらはデジタルグラフでは表示されません)の指定、そしてシミュレーション動作の一般的な特性を指定します。私達に変更する必要があるのは終了時間ですから、1.00 から 100u(文字通りに 100u とタイプする事もできます—ISIS はこれを 100E-6 に変換します)に修正して、「OK」を選択します。

デザインはこれでシミュレーションのための準備ができました。この時点で、私達のバージョンのデザイン ("Samples\Tutorials\ASIMTUT2.DSN") をロードすることが、実際のシミュレーションで発生する問題の回避や次の章の為に良いでしょう。あるいは自分で作成した回路を使用しながら、問題が起きた場合にだけ、ASIMTUT2.DSN ファイルをロードする事もまた可能です。

## シミュレーション

回路をシミュレートするために、「*Graph*」メニューに必要なすべての「*Simulate*」コマンドを呼び出します(またはキーボードショートカットを使用します:スペースバー)。「*Simulate*」コマンドで回路がシミュレーションされて、現在のグラフ(「*Graph*」メニューでマークしたものが)シミュレーション結果で更新されます。

では動作させましょう。ステータスバーにはシミュレーションの到達した過程が示されています。シミュレーションが完成すると、グラフは新しいデータで書き換えられます。ISIS とシミュレータ・カーネルの現在のバージョンにおいては、グラフの開始時刻は無視されています - シミュレーションの開始時刻はいつも0であり、そして停止時刻に達するか、もしくはシミュレータが静穏状態に達するまで動作します。ESC キーを押すことによってシミュレーションを途中で停止することができます。

シミュレーションアドバイザー(アプリケーション下部のステータスバー横)は最後に行われたシミュレーション動作の情報を保持します。このログは、シミュレーションアドバイザー上でマウスを左クリックすることによって、又は「*Graph*」メニューの「*View Log*」コマンドで、見ることができます。

警告やエラーが報告されない場合は、アナログシミュレーションに関するシミュレーションログはめったに感動的な内容とはなりません、一方間違っている場合には、何が問題であるかの詳細を見つける事ができるでしょう。また場合によっては、シミュレーションログから、グラフ・トレースでは容易に利用できない役に立つ情報が入手できます。



シミュレーション動作後の、シミュレーションアドバイザー表示。

もしあなたが2度「*Simulate*」コマンドを呼び出した場合、何か変であると気付くでしょう - シミュレーションを全く開始しません。これは ISIS のパーティション・マネージメントが、デザインの一部を調べて、特定のグラフに変更があるか、そしてシミュレーションを実行する必要があるか、を判断するほど優れているからです。私たちの単純回路については何も変化していませんので、シミュレーションは全く行われません。もし何らかの理由で常にグラフを再シミュレーションしたい場合は「*Edit Transient Analysis*」ダイアログフォームの「*Always Simulate*」チェック・ボックスにチェックをいれます。実際にシミュレーションされた内容、を知りたい場合は、同じダイアログフォーム上の「*Log Netlist*」チェック・ボックスでチェックできます；これによってシミュレーションログにシミュレータネットリストが含まれます。

これで初回のシミュレーションは完了しました。グラフ上のトレースを見て、その詳細を知ることは難しいです。回路が予測したように機能しているかどうかを調べるためには、若干測定する必要があります・・・

## 測定方法

接続図にあるグラフは、回路と並んであり、最小になっています。タイミングの測定値を得るためには、まずグラフを、最大にしなければなりません。これを行うためには、最初にグラフがタグ付けされていない事を確認してください、次にグラフのタイトルバーでマウスの左ボタンをクリックします；グラフはそれ自体のウインドウで再描画します。表示の上部に沿ってメニューバーは維持されます。この下において、スクリーンの左側にはトレースラベルが表示されて、右側にはトレース自身が表示されます。表示の下部において、左側にはツールバー、右側はステータス領域であり、カーソル時間/ステート情報を表示します。これは新しいグラフであり、私達はまだどの測定値も持っていないので、グラフ上で目に見えるどのカーソルもありません、そしてステータスバーは単にタイトルメッセージのみを表示します。

トレースはそれぞれのラベルと一致するために色分けされます。OUT と U1 (POS IP) のトレースは表示上部に集めており、そして IN トレースラインは下部に沿って配置しています。さらに詳細にトレースの細部を見るために、他の2つと IN トレース を切り離す必要があります。これはトレースラベルを左マウスボタンでドラッグし、画面の右側に持ってくることで達成できます。これで右に Y 軸が現れます、左側のスケールとは分離されています。IN トレースは現時点では、大き過ぎるように見えますが、これは ISIS が右軸を、左軸より詳細なスケールリングに選んだ為です。グラフをはっきりさせるために、IN トレースを完全に取り除くのが、U1(POS IP)がちょうど同じように役に立ちますので、おそらく最も良いでしょう。削除するために2回 IN ラベルを右クリックしてください。そうするとグラフは1つで、左手の Y 軸に戻ります。

私達は2つの量を測定するべきです：

- 回路の電圧利得。
- 出力のおおよその立ち下がり時間。

これらの測定はカーソルを用いて行います。

それぞれのグラフは「Reference(基準)」と「Primary(1次)」の2つの参照用カーソルを持っています。リファレンスカーソルは赤で表示され、プライマリーカーソルは緑で示されます。カーソルは常にトレースにロックされます、波形にカーソルが「乗る」と小さい「X」を表示してカーソルがロックされている事を表します。軸の正確な読みを容易にするために、x 軸と y 軸の両方の小さいマークが、'X'の位置に従って動くでしょう。もしキーボードを用いて動かされると、カーソルは x 軸で次の小さい分割部分に動くでしょう。

リファレンスカーソルを配置して始めましょう。同じキー/操作は、リファレンスとプライマリーカーソルの両方をアクセスするのに用います。どれが実際に影響を受けるかは、キーボードの CTRL キーの使用で選択されます；リファレンスカーソルは 2 つのうちで最も使用が少なく、常に CTRL キー(キーボードの)を押した状態で、アクセスします。カーソルを配置するのに、あなたが必要とするすべてのトレースデータ(トレースラベルではありません - これは別の目的で使用されます)をポイントします、あなたがロックしたいカーソルについては左クリックします。CTRL キーを押しているときは、リファレンスカーソルを配置(又は移動)します；CTRL キーを押していない時は、プライマリーカーソルを配置(又は移動)します。マウスボタン(及びリファレンスカーソルの b キー)が押されている間、カーソルについてドラッグできます。従って、CTRL キーを押し(そして押さえ続け)ながら、グラフの右手側にマウスポインタを移動し、両方のトレース上で、マウスの左ボタンを押します。赤いリファレンスカーソルが現れます。X 軸上の 70u か 80u でカーソル(CTRL キーを押したままで)をドラッグします。そうするとステータスバーのタイトルが取り除かれて、カーソル時間(左に赤色で)と、クエスチョン(右側に)のトレース名称が付いたカーソル電圧を表示するでしょう。それは、私たちが望んでいる OUT のトレースです。

左、と右のカーソルキーを使って、カーソルをX方向に動かすことができ、そしてアップ 及び ダウンのカーソルキーを使って、以前の、あるいは、次のトレースにカーソルをロックすることができます。左(LEFT)、と右(RIGHT)のキーはそれぞれカーソルをx軸の左、あるいは右のリミット位置に移動します。コントロールキーを押下している状態でキーボード上の、左と右の矢印キーを押すと、時間軸上のリファレンスカーソルが、小区分に沿って動きます。

今度は OUT トレース上の 20u と 30u の間にプライマリーカーソルを配置します。CTRL キーを押し続ける必要がないこと以外、前に記述したリファレンスカーソルとまったく同じです。プライマリーカーソルの時間と電圧(緑で)がステータスバーに付加されます。

また 2 個のカーソル位置間の、時間及び電圧の両方の違いも表示されます。電圧差は 100mV 以上あるべきです。入力パルスは正の 10mV で、アンプは 10 倍の電圧利得があります。値が正電圧である事に注意してください、それはプライマリーカーソルはリファレンスカーソルより上にあるからです - 差分の読み出し値は、プライマリーリファレンスです。

また、出力パルスのどちら側の立ち下がりエッジも、カーソルを配置し、相対的な時間の値が測定できます。これはマウスでドラッグするか、カーソルキー(リファレンスカーソルには、CTRL キーを忘れないでください)を使用することで行います。直線のプライマリーカーソルがカーブの右にあり、リファレンスカーソルが立ち下がりエッジの開始の角にあるはずで、あなたは 10us より少し少ない時間の、立ち下がりエッジがあることがわかるでしょう。

## 電流プローブの使用

測定が終わりのましたので、回路に戻ることができます。いつもの方法でグラフウインドウを閉じるか、もしくは急ぐ場合はキーボード上の ESC キーを押すことで閉じることが可能です。私達は次にフィードバックの電流量を調べるのに、R4 の電流を測定します。

電流プローブは電圧プローブと同様に使用されますが、1 つの重要な違いがあります。電流プローブはそれに関連している方向が必要です。電流プローブは実際には、配線を切断してそのギャップの間に挿入することによって動作するので、どの方向を向いているのかを知る必要があります。これは簡単でそれを置く方向で決まります。初期方向(右方向)において電流プローブは、左から右への水平方向の配線における電流値を測定します。垂直な配線の電流を測定するためには、プローブを  $90^\circ$  か  $270^\circ$  に回転する必要があります。不適切な角度にプローブを配置するのは間違いで、シミュレーションを実行した時に、報告されるでしょう。疑問に思う場合はシンボルの矢印を確認ください。これは現在の電流の方向を指しています。

「*Current Probe*」アイコンをクリックし電流プローブを選択します。矢印が下向きになるように、時計回りの「*Rotation*」アイコンを左にクリックしてください。そして右側にある R4 と U1 のピン 6 の間の垂直な配線にプローブを置いてください。タグ付けと、最小となったグラフの右側をドラッグして、グラフの右側にプローブを追加してください。右側は電流プローブのためには良い選択で、電圧プローブと異なり、通常電流の振幅は数桁のスケールで表示する必要があり、それらを詳細に表示する為には個別の軸が必要です。現時点では、トレースは電流プローブのために描かれていません。グラフを再シミュレートする為にスペースバーを押してください、そうすればトレースは現れます。

最小化したグラフからでも、オペアンプに期待する出力波形の帰還ループ電流を見る事ができます。電流はトレースの上、下の領域で  $10\mu\text{A}$  と  $0\mu\text{A}$  間でそれぞれ変化します。お望みならばグラフはより明確なトレースを調べるため最大化されるかもしれませんが。

## 周波数解析

アナログ回路シミュレーションでは過渡解析と同様に、利用できる他のいくつかの分析形式があります。それらは大体同じような仕方にて、グラフ、プローブそしてジェネレーターが使用されます、しかしそれらはこのテーマとは異なった変化となります。私達が考慮する次の分析形式は周波数分析です。周波数分析では、x軸は周波数(対数目盛り)となります、そしてy軸上には測定点(プローブポイント)の大きさと位相が表示されるでしょう。

周波数分析を行なうために、「*Frequency*」グラフが必要となります。「*Graph*」アイコンを左クリックして、オブジェクトセレクトにグラフの種類のリストを再表示し、「*Frequency*」グラフをクリックしてください。そしてマウスの左ボタンでボックスをドラッグして、従来と同様接続図上にグラフを配置します。既にある過渡グラフを取り除く必要は全くありませんが、必要なスペースを確保するために行うかもしれません(右を2度クリックするとグラフは削除されます)。

次にプローブを追加します。電圧プローブを OUT と U1(POS IP)の両方に加え、周波数グラフでは、2つのy軸(左と右)は特別な意味を持っています。左のy軸はプローブ信号の振幅を表示し、そして右のy軸は位相を表示します。両方とも表示するためにはグラフの両サイドにプローブを追加しなくてはなりません。グラフの左で OUT プローブにタグを付けてドラッグし、その後右にそれをドラッグします。各トレースは通常、個別の色を持ちますが、共に同じ名前を持ちます。次に U1(POS IP)プローブをグラフの左側だけにタグ付けして、ドラッグしてください。

振幅と位相値は基準値(リファレンス)を指定しなければなりません。ISIS ではこれは「Reference Generator」を指定することによって行います。リファレンスジェネレータの出力は常に、0dB(1 ボルト)で 0° です。既存のどの種類のジェネレータもリファレンスジェネレータとして指定されるかもしれません。周波数解析では、回路にある他のすべてのジェネレータが無視されます。私達の回路でリファレンスとして IN ジェネレータを指定するには、グラフにおいて単純にそれにタグ付けしドラッグします、それはプローブとして加えるのと同様です。ISIS はそれをジェネレータであると仮定しますので、あなたはリファレンスジェネレータとしてそれを加えて、それを確認しながらステータスラインのメッセージをプリントします。これは確実に行って下さい、さもないとシミュレーションは正確に機能しないでしょう。

私達はグラフの設定について、初期値で選択されている周波数範囲の細かさを、編集する必要はありません。がもし編集する(グラフのポイントを設定して CTRL - Eを押す)のであれば、「Edit Frequency Graph」ダイアログフォームが、過渡の場合とは少し違っているのが見られるでしょう。それらの目的は確定しているもので、軸にラベルを付ける必要はありません、そして表示振幅のプロットを dB もしくは、標準単位にするチェックボックスがあります。このオプションは dB による絶対値表示に設定したままにするのが最も良い方法であり、他の方法では回路からの応答が実際の値とはならないでしょう。

次に、シミュレーションを開始するのにスペースバー(周波数グラフ上にマウスをおいて)を押します。終了した時にグラフのタイトルバーを左クリックして、それを最大にします。最初に OUT の振幅のトレースを考慮すれば、通過帯域利得はちょうど 20dB(期待値)以上であり、そして有効な周波数レンジが、約 50Hz から 20kHz である事が確認できます。カーソルの動作は従来と全く同様に動作します - 上記の説明を確かめるのにカーソルを使用するのが良いでしょう。OUT 位相のトレースは、応答で予想される最大位相歪を示しており、ユニティゲインの周波数においては、グラフ右のすぐにある -90° に落ちます。U1(POS IP)の振幅トレースを調べると、入力バイアス回路のハイパスフィルタ効果を明確に見ることができます。x 軸のスケールは対数であるのに注意してください、そして軸の値を読むには、カーソルを用いるのが最も良い方法です。

## スイープ変数解析

ISIS は、回路パラメータのいくつかを変えることによって、回路がどう影響を受けるかを確認する事が可能です。ここでは 2 つの解析形式が可能 - DC スイープと AC スイープです。「DC Sweep」のグラフは掃引変化に対して連続した動作点の値を表示し、「AC Sweep」グラフは周波数グラフのように、一つの点の振幅と位相の周波数解析値から、連続表示します。

解析におけるこれらの形式は類似しておりますので、私達はただ 1 つのみ考慮します - DCスイープです。入力バイアス抵抗、R1 と R2 は U1 に流れ込む少量の電流の影響を受けています。これらの両方の抵抗値を変えることによって、バイアス点がどのように影響するかを見るために、DCスイープが用いられます。

接続図の空き領域に「DC Sweep」グラフを配置します。次に U1(POS IP)プローブにタグ付けして、グラフの左にそれをドラッグしてください。私たちは、スイープ値を設定する必要があります、これはグラフを編集することによって完了します(そのポイントを指定して、そして CTRL-E を押します)。「Edit DC Sweep Graph」ダイアログフォームは領域で設定した、スイープ変数名、開始と終了の値、およびスイープで取るステップ数、を含んでいます。私達は抵抗値として 100k から 5M の範囲で掃引したいので、100k を開始位置へ、5M を終了位置へ設定します。OK をクリックする事によって更新が行われます。

もちろん、抵抗 R1 と R2 は既にある固定値から、掃引が必要なために変更されています。これを行うためには、R1 を右クリックした後左クリックして編集に入り、「Value」の値を 470k から X に変更します。グラフのダイアログフォームにおいて掃引変数は、X が同様に残されていることに注意しましょう。OK をクリックします、そして R2 も同様に編集を行い、値を X に設定します。

現時点であなたは、それをポイントし、そしてスペースバーを押すことで、グラフをシミュレーションすることができます。そして、グラフを最大にすることによって、バイアスチェーンの抵抗が増加するのに従ってバイアスレベルが減少するのを確認できます。5MΩでそれはかなり異なります。もちろんこれらの抵抗を変えることによって周波数特性にも影響を与えるでしょう。私達は低い周波数に対する影響を確認するために、50Hz での「AC Sweep」解析をするかもしれません。

## ノイズ解析

最後に利用可能な解析のフォームは「Noise」解析です。この解析フォームでは、シミュレータはそれぞれのコンポーネントが発生するサーマルノイズの量を考慮に入れます。そしてこれらの寄与しているすべてのノイズは、回路の各プローブポイントにおいて合成(二乗する)されます。結果はノイズバンド幅と相反してプロットされます。

ノイズ解析には若干重要な特性があります：

- 各々を考慮する時、シミュレーション時間は、回路の電圧プローブ(とジェネレーター)の数に直接比例します。
- 電流プローブはノイズ解析には意味を持たなく、無視されます。
- 非常に多くの情報がシミュレーション・ログファイルとして出力されます。
- PROSPICE は入力と出力の両方のノイズを計算します。前者を行うためには入力のリファレンスを定義しなくてはなりません — これは周波数のリファレンスと同様で、グラフにジェネレーターをドラッグすることによって行います。入力ノイズのプロットは、出力の各ポイントで検出したノイズから、入力の等価雑音を表示します。

私達の回路のノイズ分析を行なうために、まず最初に R1 と R2 を 470kΩ に戻す必要があります。まずこれをします。そして「Noise」グラフのタイプを選択して、そして接続図の使われていない領域に新しいグラフを配置します。私達が本当に必要なのは出力ノイズのみです、そのため OUT 電圧プローブにタグを付けて、そしてグラフにそれをドラッグします。以前と同様のシミュレーション初期値で、私達のニーズに適しています、ただ入力のリファレンスは、入力ジェネレータの IN にセットする必要があります。「Edit Noise Graph」ダイアログフォームに

は、dBs で結果を表示するためのチェック・ボックスがあります。このオプションを使用すると、0dB が 1Vr.m.s.となる事を考慮して下さい。「Cancel」をクリックして、ダイアログフォームを閉じましょう。

従来と同様グラフをシミュレーションしてください。グラフが最大化されるとき、結果の値として、通常この形式のノイズ解析から生じる値は、非常に小さい(このケースの場合は pV)事が確認できます。しかし、あなたはどのようにして回路の雑音源をみつけだせますか? その答えはシミュレーションログにあります。よって CTRL+V を押すことによって、シミュレーションログを見ましょう。下向きの矢印アイコンを使用し、プリントアウトする動作点が下側に過ぎるまで移動させて、開始したテキスト行を見る必要があります。

#### 合計ノイズの分担.....

このリストはノイズを発生するそれぞれの回路素子における個々のノイズの分担(全周波数帯域に渡って)を示します。素子の大部分は、事実上オプアンプの内部にあり、U1\_の識別文字が付加されています。「Edit Noise Graph」ダイアログフォームで「Log Spectral Contributions option」オプションを選択すると、それぞれの点の周波数において、各コンポーネントの分担が示され、より多くのログデータが得られるでしょう。