

Eye Doctor™

シリアルデータ解析ツール

オペレーターズ マニュアル

2007 年 7 月

LeCroy Corporation

700 Chestnut Ridge Road
Chestnut Ridge, NY 10977–6499
Tel: (845) 578 6020、FAX: (845) 578 5985
Internet：www.lecroy.com

© 2007 LeCroy Corporation. All rights reserved.

LeCroy、ActiveDSO、WaveLink、JitterTrack、WavePro、WaveMaster、WaveSurfer、WaveExpert、WaveJet、および WaveRunner は、LeCroy Corporation の登録商標です。その他の製品名またはブランド名は、それぞれの所有者の商標または商標出願中のものです。本書に記載されている情報は、以前のすべての版に優先します。仕様は予告なく変更される場合があります。

ISO 9000 登録品質マネジメントシステムに基づき製造証明書をご覧になるには、www.lecroy.com をご覧ください。



この電子製品は、国や地域によって異なる廃棄およびリサイクルに関する規制の対象となります。

多くの国では、電子機器の廃棄物を通常のゴミ箱に捨てることを禁止しています。

LeCroy 製品の適切な廃棄およびリサイクルに関する詳細については、www.lecroy.com/recycle をご覧ください。

EYEDR-OM-E Rev A
915050-00 Rev A

はじめに **3**

バーチャルプロービング **4**

Eye Doctor Virtual Probe オプション 4

Processing Web でのバーチャルプロービングの設定：概要 5

バーチャルプロービング ダイアログ 7

システム記述 ファイル 7

設計および実装ルール 8

Input/Output 11

ログ 12

Compile ボタン 13

Processing Web におけるバーチャルプロービングの設定：の詳細 14

ヘルパープロセッサ 15

Reframer 15

Sparser 16

Interpolator 17

システム記述ファイル コンパイラ 18

ファイルの概要 18

記述 18

システムの定義 25

ログメッセージ 30

S パラメータ ファイル 36

Mixed モード S パラメータ 38

システム の概要 39

測定構成 40

出力構成 44

イコライザ エミュレーション **50**

Eye Doctor イコライゼーション オプション 50

Processing Web におけるイコライザエミュレーションの設定 50

Basic Equalization レシーバの設定 55

フィードフォワード イコライザ(FFE) の 設定 57

Decision Feedback Equalizer（DFE）の構成 58

Equalization 済み レシーバの接続 59

Equalization レシーバの例 60

空白 ページ

はじめに

Eye Doctor は、LeCroy オシロスコープ内で動作するソフトウェアツールです。このツールには、主に 2 つの機能があります：

- バーチャルプロービング（EYEDR_VP オプションで有効化）
- 理想イコライザ エミュレーション（EYEDR_EQ オプションにより有効化）

バーチャルプロービング機能は、信号プロービングの場面において、以下のような様々な利点をもたらします：

- プローブが接続されている場合とされていない場合の両方で、回路内で発生する波形を確認できるようにすることで、プローブの負荷効果を補正できる
- プローブ接続点以外の場所で発生する波形を捕捉できる機能
- 実際に測定に使用した構成とは異なる回路構成で発生する波形を捕捉する機能

理想イコライザ エミュレーション機能により、以下のような様々なイコライザ構成のエミュレーションが可能になります：

- 線形トランスバーサル、タップ付き遅延線、および線形フィードフォワード イコライザ(FFE)
- クロックリカバリ
- デシジョンフィードバック イコライザ(DFE)

イコライザ エミュレーション機能を使用すると、復元されたデータ、クロック、およびイコライザ処理された波形を確認できます。イコライザは、タップ遅延、タップ数、タップ係数によって指定することも、ブラインド適応を利用して設定を最適化するように自動的にトレーニングすることもできます。

バーチャルプロービング

バーチャルプロービング機能を使用するには、以下の要件のうち少なくとも 1 つを満たしていることを確認してください。

- プローブの負荷およびスルー応答といったプローブの影響を補償する必要があること。
- プローブが回路に接続されていないかのように、プローブ接続時の波形を確認する必要があること。
- プローブを接続した位置以外の回路上の位置における波形を確認する必要があること。
- 測定が行われている回路構成とは異なる回路構成における波形を確認する必要性。

このような要件がある場合、バーチャルプロービング機能が役立ちます。バーチャルプロービングを使用するには、以下が必要です:

- プローブ配置および仮想波形を確認したい構成の両方について、正確かつ完全な回路構成の記述。
- プロービング要素やオシロスコープを含め、システム内の各種回路コンポーネントの挙動に関する、正確かつ完全な測定値、モデル、または仮定。

この情報は、主に以下の 2 種類のファイルを通じてオシロスコープに提供します。

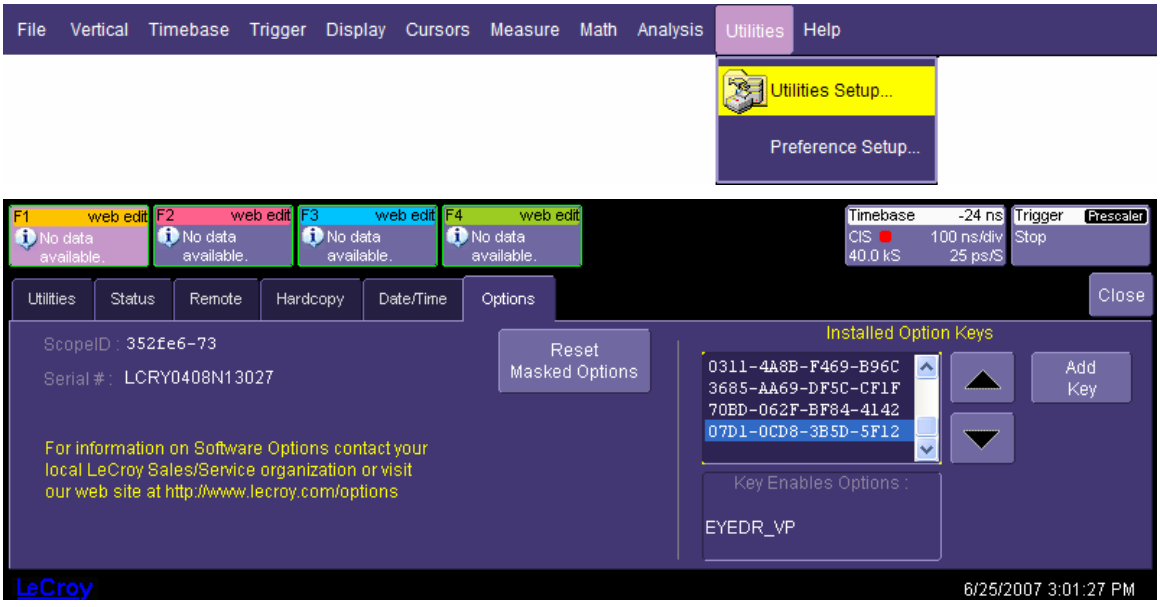
- システム記述ファイル – オシロスコープに対して、様々な回路の記述情報を提供するファイル。
- S パラメータファイル – S パラメータを用いて回路素子の挙動を記述する、Touchstone 形式のファイルです。

もちろん、オシロスコープに EYEDR_VP オプションがインストールされている必要もあります。

以下のセクションでは、バーチャルプロービングの基本的な使用方法を順を追って説明し、その後、操作の詳細について解説します。

Eye Doctor Virtual Probe オプション

Eye Doctor Virtual Probe オプションは、お使いのオシロスコープにすでに搭載されている可能性があります。Support→ About を選択して、ソフトウェアオプションの一覧が表示されるダイアログを開いてください。その一覧にオプションコード **EYEDR_VP** が含まれている場合、お使いのオシロスコープにはすでに Eye Doctor Virtual Probe オプションが搭載されています。含まれていない場合は、**Utility** メニューからオプションのインストールと確認を行います。このオプションには、固有のオプションキーが必要です。



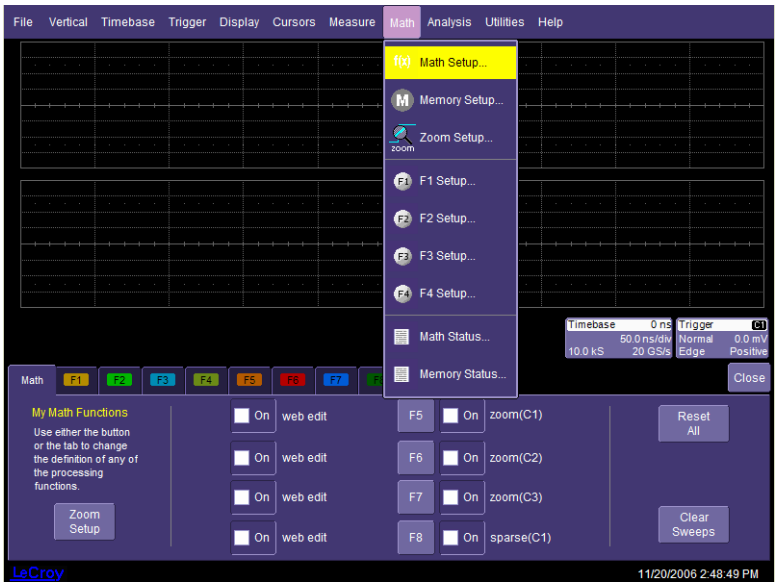
EYEDR_VP オプションのオプションキーを取得するには、オシロスコープの ID およびシリアル番号を LeCroy に提供する必要があります。その後、LeCroy から 16 桁の 16 進数で構成されるオプションキーが提供されます。キーが正しくインストールされると、Installed Option Keys の下に **EYEDR_VP** オプションが表示されます。

オプションキーに加え、お使いのオシロスコープのファームウェアバージョンは 5.0.4 以降である必要があります。現在のファームウェアバージョンは、Utility ダイアログの Status タブで確認できます。

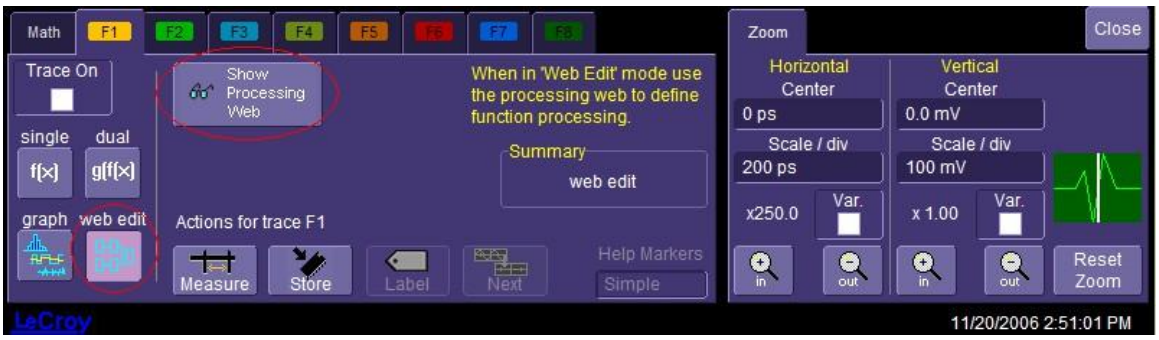
EYEDR_VP オプションを有効にすると、バーチャルプロービング機能と Processing Web エディタが利用可能になります。Processing Web エディタを使用すると、独自のグラフィカルユーザーインターフェースを通じて処理構成を編集することができます。

Processing Web でのバーチャルプロービングの設定：概要

Processing Web エディタには、**Math Setup** メニューからアクセスします。

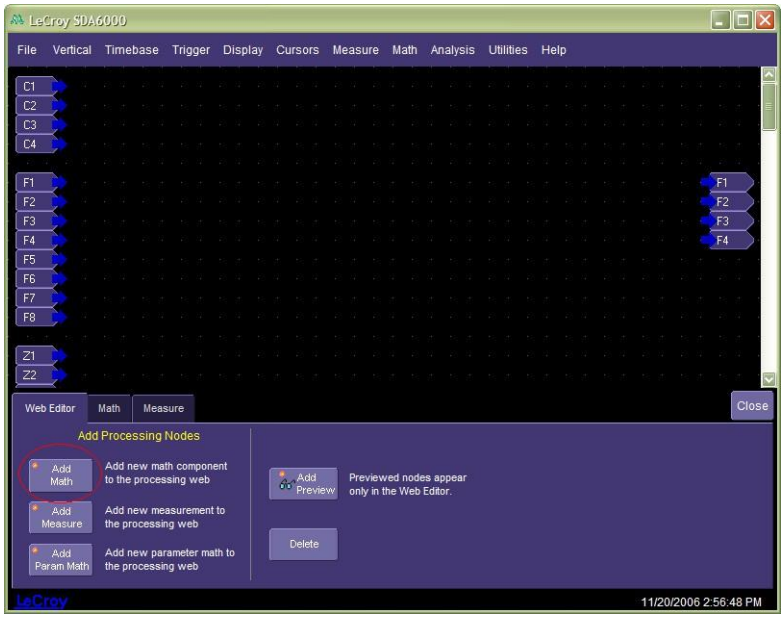


Math メインダイアログには、演算関数の定義方法が表示され、定義可能な各演算関数用のタブが用意されています。任意の演算関数のタブを選択すると、その演算関数を設定できます。

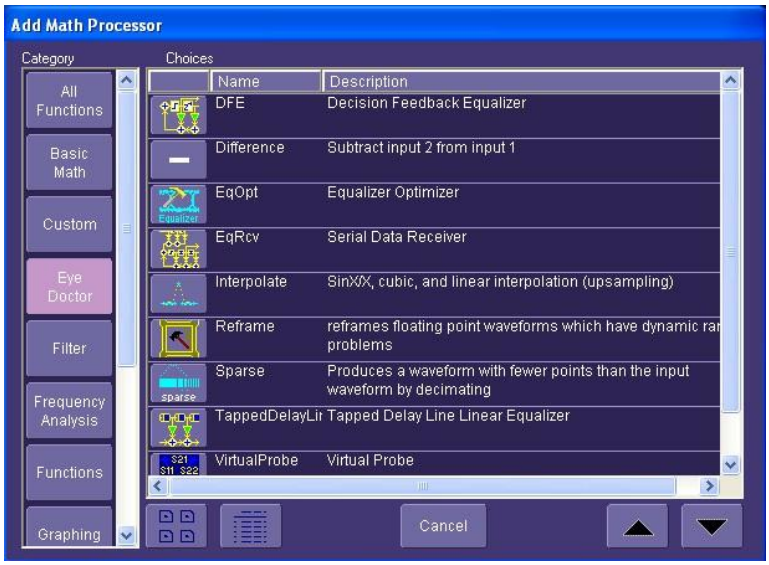


各数式設定ダイアログの左側には、数式機能の設定に関する 4 つのオプションがあります。これらは、single、dual、graph、web edit です。Web edit は、Web edit オプション（EYEDR オプションおよび XWEB オプションによって有効化されます）が有効な場合にのみ利用可能です。**Web edit** が選択されている場合、**Show Processing Web** ボタンから Processing Web にアクセスできます。

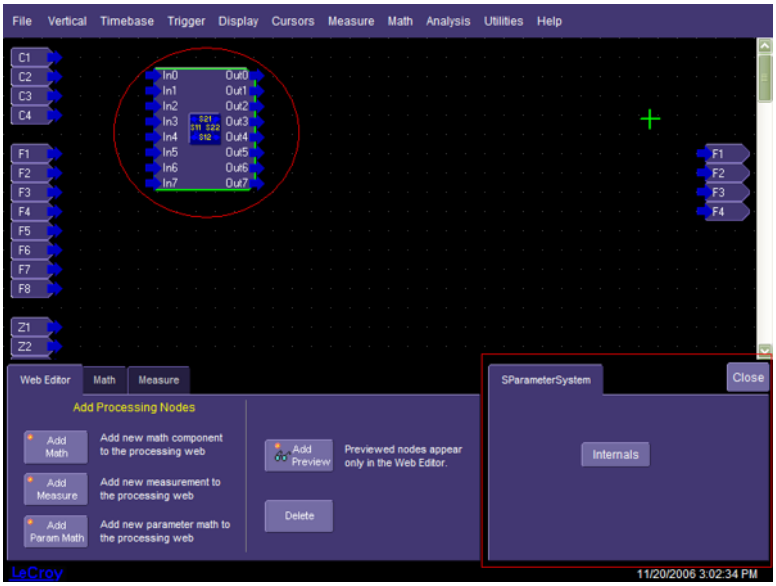
Processing Web が開くと、左側に利用可能なすべての波形ソース、右側に利用可能なすべての波形出力が表示されます。この場合、F1、F2、F3、および F4 は Processing Web edit 波形として定義されているため、これらの関数のアイコンがウェブエディタに表示されます。"Web edit"として定義されていない関数は表示されません。



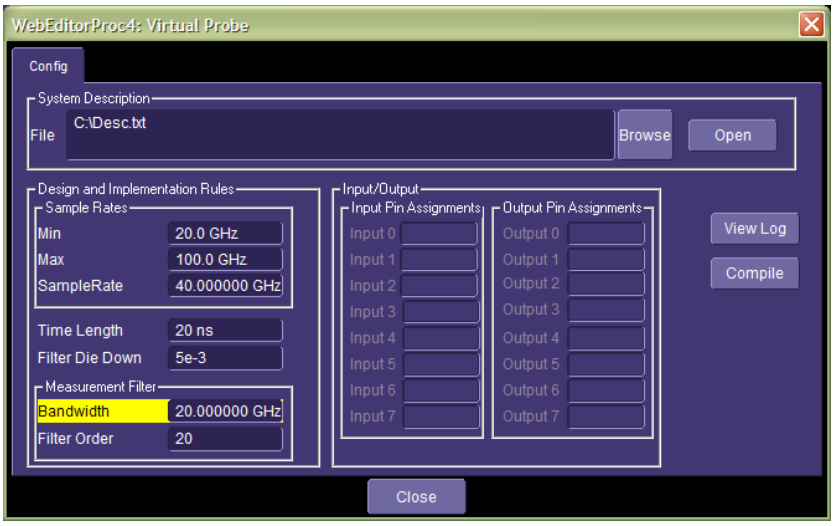
Eye Doctor コンポーネントにアクセスするには、**Add Math** ボタンを選択します。これにより、オシロスコープで使用可能なすべての演算関数が表示されたダイアログが表示されます。**Eye Doctor** カテゴリを選択すると、Eye Doctor のすべての演算関数が表示されます。Web editor 内で使用する場合、これらの関数は**プロセッサ**と呼ばれます。



バーチャルプロービングコンポーネントを使用するには、**VirtualProbe** を選択します。これにより、Processing Web 上にバーチャルプロービングコンポーネント（8 入力・8 出力のプロセッサ）が表示されます。右下隅にあるダイアログの **Settings** ボタンをクリックすると、内部設定ダイアログにアクセスできます。



バーチャルプロービング ダイアログ



バーチャルプロービング ダイアログには、いくつかのセクションがあります：

1. システム記述ファイルの指定
2. 設計および実装に関する規則
3. 入力／出力
4. ログ表示ボタン
5. コンパイルボタン

システム記述 ファイル

この入力ボックスでは、システムの記述に使用するテキストファイルを選択できます。このファイルおよびファイル形式については、以下で詳しく説明します。このファイルは、参照用 S パラメータデータとともに、バーチャルプロービングシステムにおける所望の構成での回路の完全な記述と動作を提供します。また、回路の測定（入力）ノードおよび出力ノードを参照します。

設計および実装ルール

システムの動作を規定する設計および実装ルールには、以下が含まれます。

- バーチャルプロービングが許可される最小および最大サンプリングレート、ならびに想定されるサンプリングレート
- 時間長
- フィルタの減衰量
- 測定用フィルタの帯域幅と次数

動作時、サンプリングレートは、処理コンポーネントの最初の入力ピン（**In0** と表示）に接続された波形のサンプリングレートから導出されます。コンポーネントが接続され、実際の波形が処理されるまでは、システムを事前に設定できるよう、想定されるシステムのサンプリングレートを指定しておくと便利です。これを行うには、サンプリングレートを指定します。ただし、この数値はコンポーネントが実際の波形を処理している際に変化する可能性があります。つまり、システムは動作中に変化するサンプリングレートに適応します。

一方、コンポーネントの動作にとって明らかに不適切なサンプリングレートも存在します。たとえば、10 Gb/s シリアルデータシステムのバーチャルプロービングを目的として、20 GHz までの S パラメータデータがある場合、このコンポーネントが 40 GS/s 未満で動作することは意味をなしません。これは、40 GS/s のサンプリングレートでは 20 GHz のナイキストレートが必要となるためです。したがって、オシロスコープのサンプリングレートの変化に対応するために、最小サンプリングレートは 40 GHz に設定されています。

さらに、最大サンプリングレートが望ましくない状況もあるかもしれません。目標とするサンプリングレートを Min.と Max.の範囲内に収めてください。システムが単一のサンプリングレートでのみ動作するように設計されている場合は、Min.、Max.、および目標サンプリングレートを同じ値に設定してください。

Time Length は、内部の周波数分解能およびインパルス応答の絶対最大長を制御するために使用されます。**Filter Die Down 量**は、時間長によって規定される最大長から、妥当な精度を得るために実際に必要なインパルス応答の部分のみを含むより短い長さにインパルス応答を切り詰め、適切な因果関係を持つインパルス応答を作成するために利用されます。

Measurement Bandwidth と **Filter Oeder** は、オシロスコープおよびプローブシステムの帯域幅を考慮して適用されるフィルタの帯域幅とフィルタ次数を制御するために使用されます。

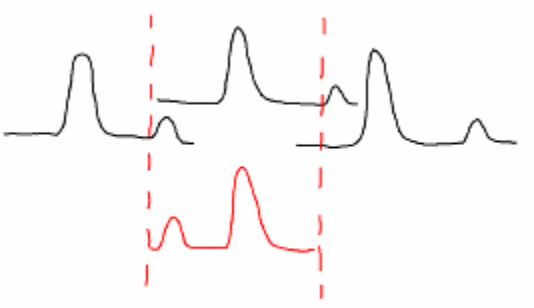
Time Length

時間長は、インパルス応答の絶対的な最大サイズ、ひいては内部の周波数分解能を定義します。すべての S パラメータデータは、所望の最大長を持つインパルス応答を生成できる周波数分解能に合わせるために、内部で補間されます。

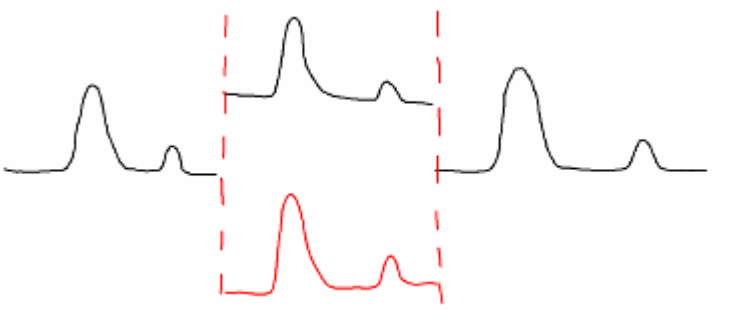
インパルス応答の最大長は、2 つの理由から重要です。1 つは、長いケーブルやバックプレーンを備えた一部のシステムでは、反射およびその影響の伝播に長い時間がかかることです。システムの挙動は S パラメータデータとして提供されるため、バーチャルプロービング機能はこれらの伝播時間を認識していません。したがって、システムに適した数値を指定する必要があります。通常、この数値は、送信源からオシロスコープの入力端子、またはシステム内のレシーバまでの伝搬時間のうち、いずれか大きい方の少なくとも 4～5 倍とします。反射がそれほど大きくない場合は、この数値を小さくすることも可能です。一般的に、システムのコンパイル時間を除けば、この数値をかなり大きく設定しても問題ありません（そのため、デフォルトでは 20 ns に設定されています）。

また、適切な因果関係を維持するためには、時間長を十分に長く設定する必要があります。これは、バーチャルプロービングコンポーネントが FFT 法を用いてシステムのインパルス応答を決定するためです。FFT 法には、逆 FFT（有限インパルス応答（FIR）フィルタを作成するため）を実行する際に、インパルス応答が実際にはパルス列に対する応答であるため、因果関係の問題が生じることがあるという欠点があります。場合によっては、インパルス応答のピークが、実際の応答よりも時間的に早く現れることがありますが、実際にはそれらは応答の後に発生する影響によるものです。時間範囲を長く設定する

ことで、インパルス列内のインパルス間の間隔を広く取り、インパルス応答間の時間的間隔を大きくすることで、こうした可能性を排除することができます。



誤った因果関係の例



正しい因果関係の例

Filter Die Down

Filter Die Down は、インパルス応答の長さを定義します。インパルス応答の長さは、時間 0 から、応答の振幅がこの値を下回る最後の時点までです。これは以下の目的で使用されます。

- 応答における重要でない変動を考慮しないことで、計算量を削減できる長さにインパルス応答を制限すること
- 因果関係の問題を軽減する

インパルス応答の高さはサンプリングレートによって異なるため、指定された値は特定のサンプリングレート（20 GS/s）に正規化されます。つまり、実際に使用されるサンプリングレートが、例えば 200 GS/s の場合、内部で実際に使用されるダイダウン量は、指定された値の 1/10 となります。

内部で計算されたダイダウン量が閾値として使用されます。内部的には、フィルタのインパルス応答が生成された後、システムは時間 0 を起点として時間軸を前後方向にスキャンし、時間 0 からこの閾値を超える最後のポイントまでのインパルス応答を保持します。

以下の図 1 および図 2 に示すインパルス応答を例に挙げます。図 1 は、時間長が 20 ns、ダイダウン量が 0.001 のシステムです。この場合、主要なバンプは 5 ns 付近で終了しますが、18 ns までの部分がすべて含まれます。このインパルス応答を利用するには、タップ数が 400 に近いフィルタが必要となります。また、この例では発生していませんが、18 ns で発生する最後のバンプは、実際には-2 ns から始まる主要なバンプよりも前に発生している可能性があります。

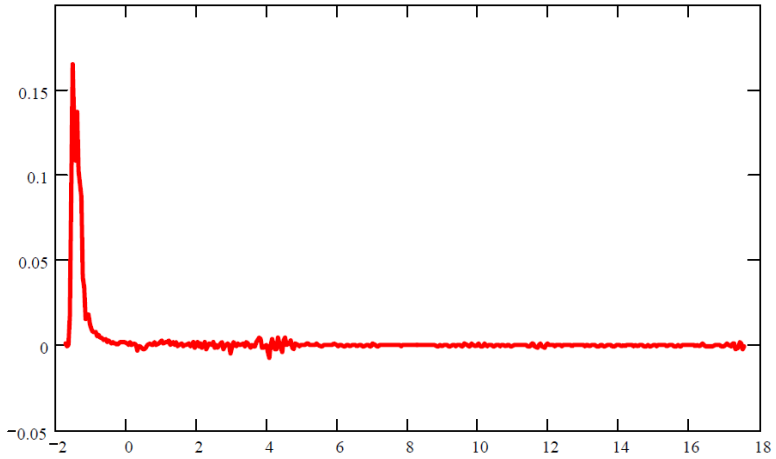


図 1 – 仮想プローブのインパルス応答（Filter Die Down 0.001）

図 2 に示すインパルス応答では、ダイダウン量が 0.003 に設定されています。インパルス応答が約 5 ns（約 120 タップ）に短縮されていることに注目してください。5～18 ns の区間はそれほど重要ではないため、実質的な精度の低下はありません。さらに、このインパルス応答の方が、適切な因果関係をより正確に表している可能性が高くなります。

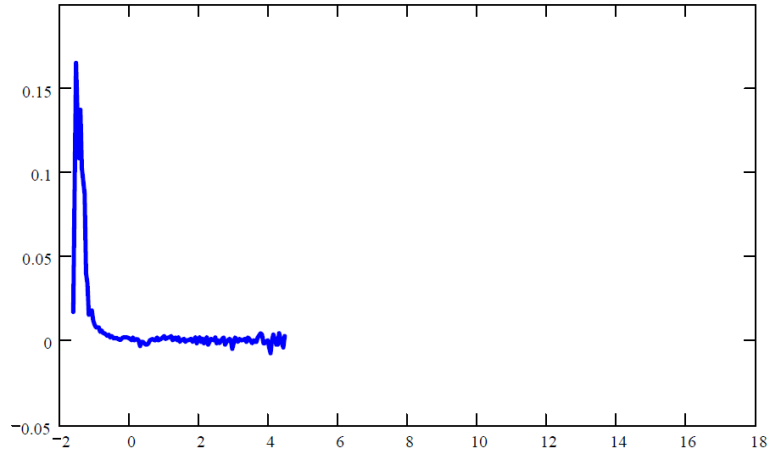


図 2 – 仮想プローブのインパルス応答（Filte Die Down 0.003）

測定帯域幅とフィルタの次数

バーチャルプロービング用の内部フィルタを計算する過程において、特に高周波域で過度なブーストを生じさせるフィルタが生じる可能性があります。多くの場合、これはオシロスコープのチャンネルやプローブ素子の帯域幅が限られている一方で、フィルタの計算が高周波まで及ぶことに起因します。より一般的に言えば、トランスミッタからオシロスコープのチャンネルに至る経路が、バーチャルプロービング ポイントへの経路よりも高周波減衰が大きい（より低い帯域幅）場合、バーチャルプロービング用に生成された内部フィルタには、意図しない高周波増幅が含まれがちになります。

仮想プローブの目的は帯域幅を拡張することではないため、プローブとオシロスコープ チャネルの合計帯域幅を入力することで、これらの影響を軽減することができます。

オシロスコープやプローブの影響に加え、デバイス用に提供されるモデルも帯域幅に制限があることが多く、高周波域の測定精度が低い場合がしばしばあります。このような場合、高周波域の精度を反映した測定帯域幅の値を入力することが有用です。つまり、測定帯域幅は、システムを構成する各種コンポーネントの帯域幅に合わせて適切に設定する必要があります。

測定帯域幅の処理に使用されるフィルタは、一般的に高次フィルタです。これは、所定の次数のバターワースフィルタによって与えられる振幅応答を持つフィルタとして実装されています。位相応答は常に線形。デフォルト値は 20 で、これは非常に

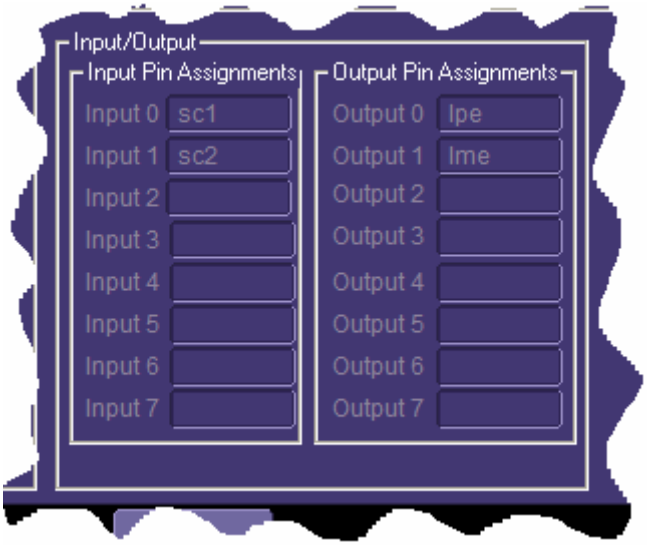
高次なフィルタですが、より鋭いカットオフを得るためには、さらに高次のフィルタを使用することもできます。フィルタの次数が 1 増えるごとに、20 dB/decade（または 6 dB/octave）ずつロールオフが急になります。

Input/Output

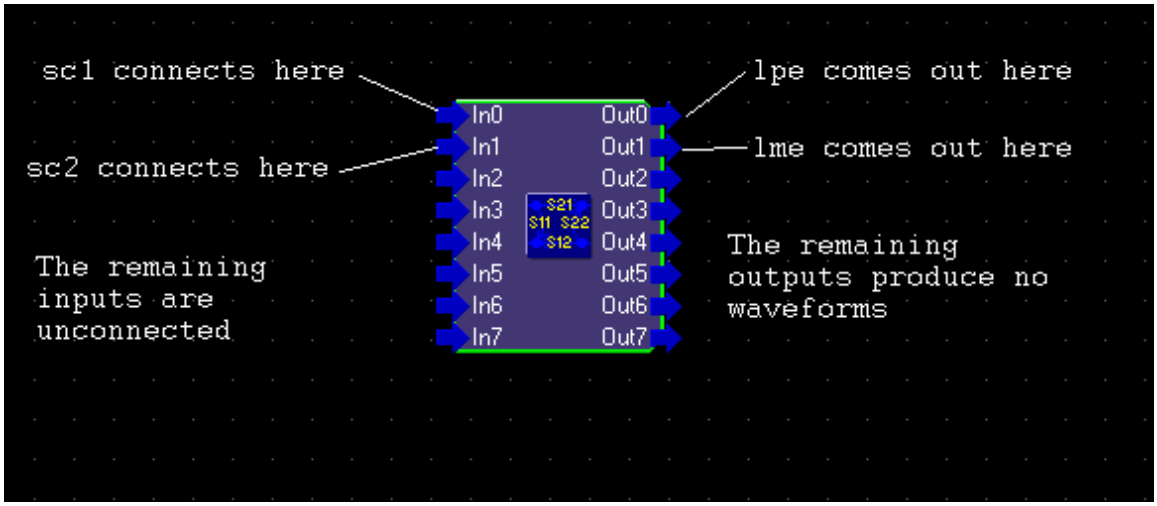
Input/Output ボックスには、Input Pin Assignments と Output Pin Assignments が含まれています。これらのピン割り当ては、コンポーネントの入力ピンおよび出力ピンに割り当てられた、システム記述ファイルで定義された電圧ノードの名前です。ダイアログ内のボックスは読み取り専用であり、指定されたシステム記述ファイルのコンパイルが正常に完了した後に値が入力されます。

ピン割り当てはシステム記述ファイルに依存するため、システム記述ファイルがコンパイルされるまで名前は不明です。ファイルがコンパイルされると、これらの名前が入力され、正しい接続が行われていることを確認するよう促されます。

たとえば、システム記述ファイルに **sc1** および **sc2** というラベルの付いた測定ノード（例えば、オシロスコープのチャンネル 1 および 2 に対応）が記述されており、出力ノードに **lpe** および **lme** というラベルが付けられている場合、コンパイル後に Input/Output ボックスには次のように表示されます：



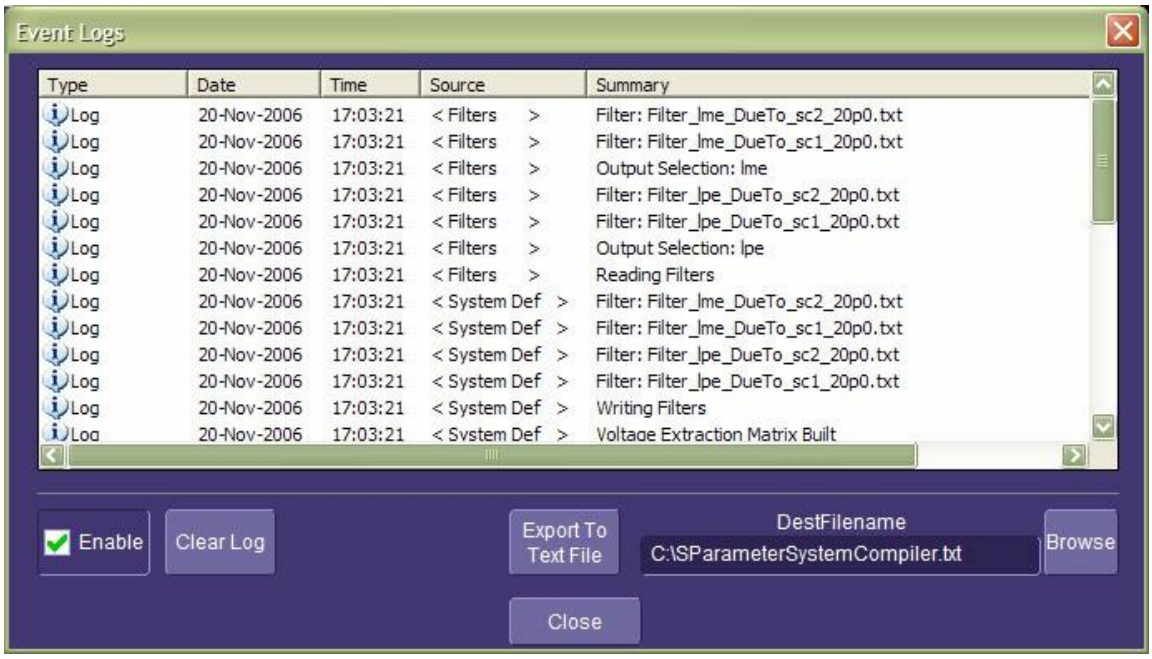
これらのピン割り当てにより、コンポーネントの接続は次のように決定されます：



8 つの入力および出力ピンの割り当ては、8 つの入力および出力ピンを反映していることに注意してください。実際には、通常、どちらか一方のピンのうちごく一部しか使用されません。

ログ

システム記述ファイルのコンパイル時には、システムが正常に動作するために、多くの条件が満たされる必要があります。コンパイル状況を確認するために、ログが用意されています。以下にその例を示します。



ログにはコンパイルの進行状況が逆順で表示され、一番上の行が最後に記録されたメッセージとなります。ログはテキストファイルにエクスポートしたり、消去したりすることができます。テキストファイルからの完全なログ一覧を以下に示します：

```
011.20.2006 17:03:05Informational File: C:\NetList.txt(null)
111.20.2006 17:03:05Informational Devices Counted: 8(null)
211.20.2006 17:03:05Informational Nodes Counted: 8 (null)
311.20.2006 17:03:20Informational 8 Devices Assigned(null)
411.20.2006 17:03:20Informational 8 Nodes Assigned(null)
511.20.2006 17:03:20Informational Device Connections and Nodes Verified(null)
611.20.2006 17:03:20Informational 4 Stimuli Assigned(null)
711.20.2006 17:03:20Informational System Defined(null)
811.20.2006 17:03:20Informational Measures Counted: 2 (null)
911.20.2006 17:03:20Informational Outputs Counted: 2 (null)
1011.20.2006 17:03:20Informational Degrees of Freedom Specified: 2(null)
1111.20.2006 17:03:20Informational With Respect to 4 Stimuli Specified(null)
1211.20.2006 17:03:20Informational Measures, Outputs and StimDefs Assigned(null)
1311.20.2006 17:03:20Informational Stimulus and Node Vector Built(null)
1411.20.2006 17:03:21Informational System Characteristics Matrix Built and Inverted(null)
1511.20.2006 17:03:21Informational System Characteristics Matrix and Stimuli Vector Reduced(null)
1611.20.2006 17:03:21Informational Node Voltage Vector Built(null)
1711.20.2006 17:03:21Informational Voltage Extraction Matrix Built(null)
1811.20.2006 17:03:21Informational Writing Filters(null)
1911.20.2006 17:03:21Informational Filter: Filter_Ipe_DueTo_sc1_20p0.txt(null)
2011.20.2006 17:03:21Informational Filter: Filter_Ipe_DueTo_sc2_20p0.txt(null)
2111.20.2006 17:03:21Informational Filter: Filter_Ime_DueTo_sc1_20p0.txt(null)
2211.20.2006 17:03:21Informational Filter: Filter_Ime_DueTo_sc2_20p0.txt(null)
2311.20.2006 17:03:21Informational Reading Filters(null)
2411.20.2006 17:03:21Informational Output Selection: Ipe(null)
2511.20.2006 17:03:21Informational Filter: Filter_Ipe_DueTo_sc1_20p0.txt(null)
2611.20.2006 17:03:21Informational Filter: Filter_Ipe_DueTo_sc2_20p0.txt(null)
2711.20.2006 17:03:21Informational Output Selection: Ime(null)
2811.20.2006  17:03:21Informational Filter:  Filter_Ime_DueTo_sc1_20p0.txt(null)
2911.20.2006  17:03:21Informational Filter:  Filter_Ime_DueTo_sc2_20p0.txt(null)
```

ログおよびログファイルには、情報、警告、またはエラーのメッセージが含まれています。コンパイルが成功した場合、ログには情報メッセージのみが含まれます。コンパイルが失敗した場合は、1 つ以上のエラーが表示されます。ログファイルには、システム記述ファイルの問題をデバッグするのに役立つ、すべてのログメッセージとその意味の説明が含まれています。

Compile ボタン

Compile ボタンは、波形処理の前にシステムファイルのコンパイルを強制的に実行するために使用されます。コンポーネントを接続する前に、入力および出力のピン割り当て名を確認するにはファイルをコンパイルする必要があるため、この機能は便利です。また、システム上で実際にコンポーネントを実行する前（波形処理が行われるとオシロスコープの制御が難しくなるため）にコンパイルエラーに対処しておく方が効率的です。

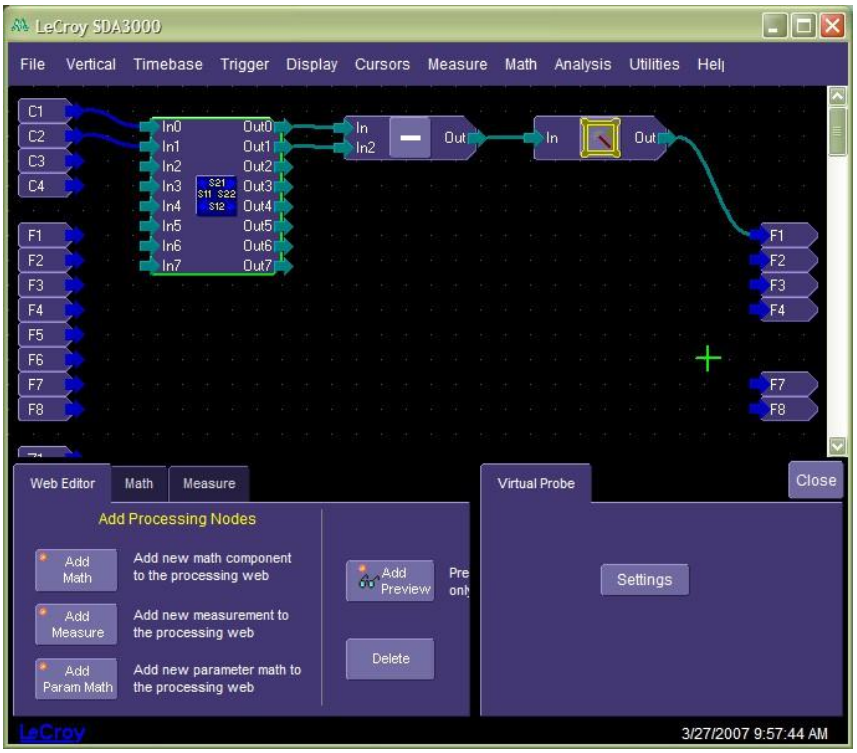
Processing Web におけるバーチャルプロービングの設定：詳細

コンポーネントを Processing Web に配置したら、次の手順を実行します：

1. システム記述ファイルを編集および選択します。
2. 最小、最大、および目標のサンプリングレートを設定し、目標のサンプリングレートが指定された最小値と最大値の範囲内にあることを確認します。
3. Time Length と Filter Die Down 値を選択します。
4. ログを開きます。
5. システムをコンパイルします。
6. ログを確認し、コンパイルが成功したことを確認します。失敗した場合は、システム記述ファイルを修正し、ログをクリアして、コンパイルが成功するまで繰り返しコンパイルし直します。
7. 入力および出力のピン割り当てを確認します。
8. 適切なチャネルおよび演算入出力をコンポーネントに接続し、Processing Web 内でコンポーネントのピンを適切に接続します。
9. オシロスコープを正しいサンプリングレートに設定します。
10. オシロスコープをトリガし、波形が生成・処理されるようにします。

コンポーネントのピンの接続は、ピンを選択し、目的の接続ポイントまでドラッグすることで行います。

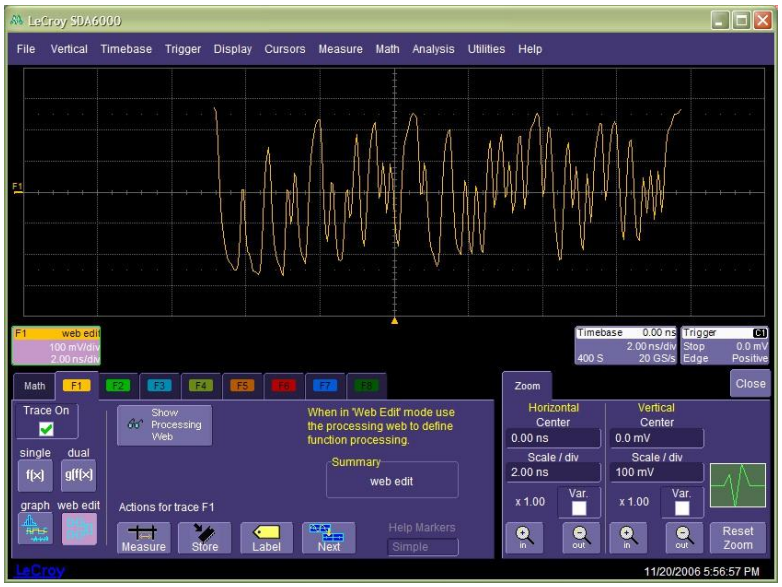
例として、システム記述ファイルに **sc1** および **sc2** というラベルの付いた測定ノード（例えば、オシロスコープのチャンネル 1 および 2 に対応）が記述されており、出力ノードには、負荷における正電圧と負電圧をそれぞれ表す **Ipe** および **Ime** というラベルが付けられているとします。さらに、負荷で生じる差動電圧に関心があるとします。システムは次のように接続されます：



ここでは、オシロスコープのチャンネル 1 と 2 が、それぞれ入力ピン 0 と 1 に接続されています。さらに、差動回路（2 つの波形の差を求める回路）が出力ピン Out0 および Out1 に接続されています。差動回路の出力はリフレーマー回路に接続されており、リフレーマー回路の出力（マニュアルの「リフレーマー」のセクションを参照）は F1 に接続されています。これにより、演算関数 F1 は、オシロスコープのチャンネル 1 および 2 で捕捉した波形を処理することで、負荷における差動電圧に対応する波形を生成するようになります。を生成することを示しています。

（リフレーマーコンポーネントについては、本章の「ヘルパープロセッサ」のセクションで説明しています。）

メニューバーから **Math Setup...**を選択して F1 を有効にし、オシロスコープをトリガすると、以下に示すような波形が表示されます。



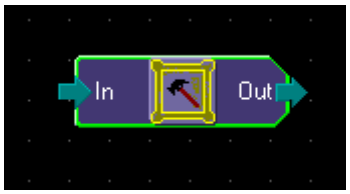
なお、波形には通常、画面の左右両端に隙間が生じます。これは、これらの波形を生成するフィルタが、採用されている FIR フィルタの起動用サンプルを必要とするため、また、仮想プローブコンポーネントによって提供される波形に時間的な遅れまたは進みがあるためです。これらの隙間は、波形を水平方向にズームすることで除去できます。上図は、送信側でプローブされた 10 Gb/s 信号による、受信側での仮想プローブ差動波形を示しています。

ヘルパープロセッサ

浮動小数点波形やサンプリングレートの処理を支援するヘルパープロセッサがいくつかあります。それらは以下の通りです：

- Reframer – 表示および下流の処理のために、浮動小数点波形の上限と下限を処理します。
- Sparser – 波形のサンプリングレートを下げる（デシメーションを行う）
- Interpolator – 波形のサンプリングレートを上げます（アップサンプリング）

Reframer

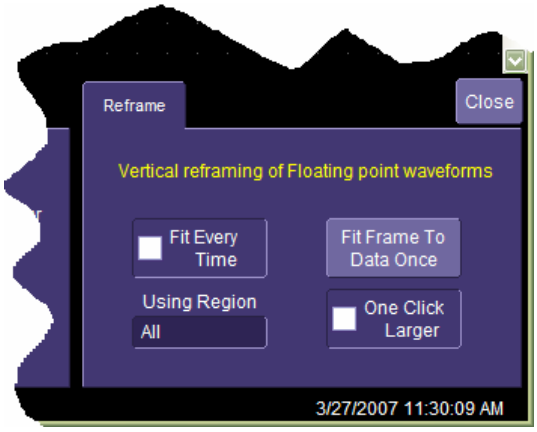


リフレーマーは、浮動小数点波形の垂直方向のリフレーミングを処理します。つまり、波形を画面に表示する必要がある場合や、固定小数点波形プロセッサ用に固定小数点形式に変換する必要がある場合に、波形がどのように解釈されるかを処理します。本質的には、波形に適した上限と下限を設定します。

ダイアログ上のコントロールには、以下の設定項目があります：

- Fit Every Time – 新しい波形ごとに新しいフレームが決定されます。これは、入力波形のすべてのデータポイントが必要となるため非常に処理が遅くなります。これは避けたい状況です。
- Fit Frame To Data Once – 波形を捕捉した後、フレームを一度設定すれば、その後は常にその設定が維持されます。これが最も有用なケースです。システムに波形が用意され、そのサイズが分かっている状態でこのボタンを押すと、波形からフレームが決定されます。

- Using Region – これにより、波形を画面全体（「すべて」）の中央、または画面の上半分もしくは下半分の中央に配置できます。All の設定を推奨します。
- One Click Larger – 波形がフレームの全範囲（垂直スケールの次の 1、2、5 の倍数）に収まるように、垂直フレームが設定されます。Fit Frame To Data Once ボタンを使用した際に、波形が依然としてこの範囲からはみ出す可能性があると思われる場合は、この設定を One Click Larger することで、フレームを少し大きくし、そのような事態に対応することができます。

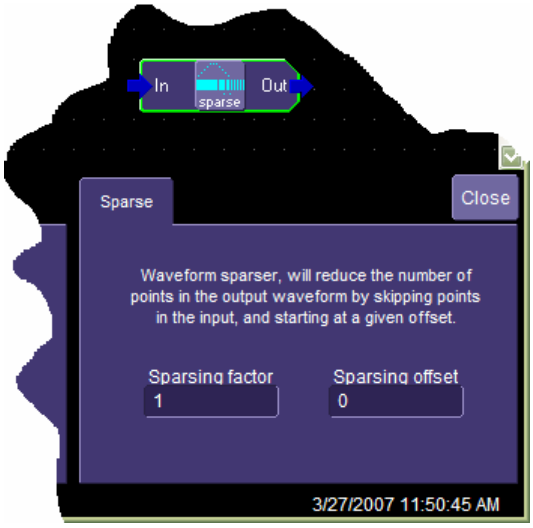


Eye Doctor システムでは、浮動小数点波形を生成するコンポーネントの出力には、リフレーマーを配置する必要があります。これらは、整数波形を表す青色のピンとは対照的に、ターコイズ色の出力ピン（Virtual Probe コンポーネントと同様）で識別されます。

Sparser

Sparser は、波形をダウンサンプリングするコンポーネントです。通常、波形は適切なサンプリングレートで捕捉されるため、これが必要になることはほとんどありませんが、処理速度を向上させたり、特に LeCroy WaveExpert シリーズのオシロスコープで Eye Doctor を使用する際に発生しがちな過度に高いサンプリングレートを低減させたりするために、波形のサンプリングレートを下げるのが有用な場合があります。例えば、WaveExpert は、単位区間あたり 512 ポイントの波形を生成することができます。これを 6 Gb/s の波形に対して実行すると、サンプリングレートが 3 TS/s をわずかに超える波形が生成されます。これは処理には過剰であり、多くの問題を引き起こす可能性があり、中でも処理速度の低下が最も顕著な問題となるでしょう。

バーチャルプロービング コンポーネントの最大サンプリングレートは 500 GS/s に制限されていますが、それでも通常は過剰な性能と言えます。6 Gb/s の波形の場合、通常、約 40 GS/s を超えるサンプリングは必要ありません。つまり、WaveExpert では通常、UI あたり約 8 サンプルで安全にサンプリングできます。これは処理において推奨される設定です。すでに極めて高いサンプリングレートで捕捉された波形がある場合は、Sparser コンポーネントを波形に配置して、サンプルを削除することでサンプリングレートを下げることができます。



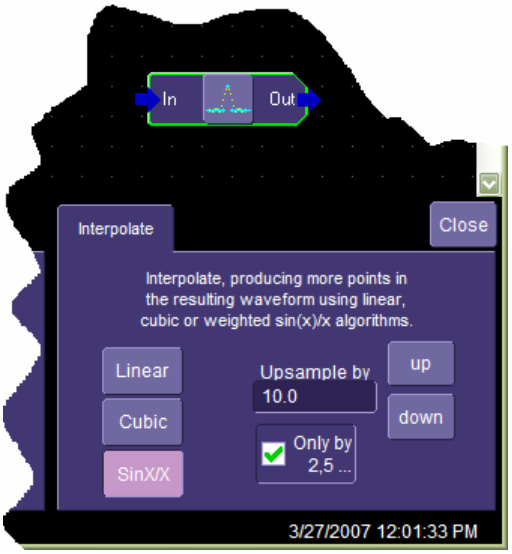
Sparser ダイアログには 2 つのコントロールがあります:

- Sparsing factor – サンプリングレートを低減する倍率。
- Sparsing offset – Sparsing を開始する波形のサンプリングポイントのオフセット。

Sparsing offset はゼロのままにし、Sparsing factor を、サンプリングレートを低減したい量に設定します。上記の例では、UI あたり 512 ポイントで 6 Gb/s の波形が捕捉されています。サンプリングレートを約 40 GS/s に低減するには、Sparser 係数を 76 に設定します。もちろん、この設定を行う際は、エイリアシングが発生しないように注意する必要があります。エイリアシングは、結果としてのサンプリングレートの 1/2 を超える周波数成分（この例では 20 GHz 以上）が存在する場合に発生します。

Interpolator

補間機能は、波形のサンプリングレートを向上させます。これは、新しいポイントを補間することで実現されます。補間は、エイリアシングが発生しない場合（つまり、元の波形成分のすべてが、その波形のサンプリングレートの 1/2 未満の周波数にある場合）に常に有効です。Eye Doctor では、波形をより滑らかにし、見栄えの良いアイパターンを生成するために、この補間機能が役立ちます。



補間機能には、以下のコントロールがあります:

- Linear、Cubic、SinX/X – 補間の方法を制御します。SinX/X を使用してください。
- Upsample by – アップサンプリング倍率（サンプルレートをどの程度引き上げるか）を制御します。

- Only by 2, 5 – Upsample by コントロールの選択単位を制御します。通常、2、5、10、20、50、100 など、1、2、5 の順序で並んだ数値のみ選択可能です。このチェックボックスをオンにすると、任意の整数を選択できるようになります。

システム記述ファイル コンパイラ

ファイルの概要

システム記述ファイルとは、システムを記述したテキストファイルのことです。ファイル内の行のうち、ピリオドで始まる行のみが有効な文を含むものとみなされます。ピリオドで始まらない行は単に無視され、コメントとして使用することができます。

すべての文はピリオドで始まります。それらは以下の通りです。

- **.Device** – デバイス、デバイス名、ポート数、およびその動作（すなわち、S パラメータ）を宣言します。
- **.Node** – 電圧ノード、名前、およびそのノードに関連付けられた 2 つのデバイスの 2 つのポートを宣言します。これにより、2 つのデバイス間のポート間接続が定義されます。
- **.Stim** – 励起および、その励起の発生源となるデバイスとポートを宣言します。
- **.Meas** – 電圧測定ノードを宣言します。これは、結果として生成されるフィルタが動作する電圧ノードを定義します。
- **.Output** – 電圧出力ノードを宣言します。これにより、生成されるフィルタが出力する電圧ノードが定義されます。
- **.StimDef** – 新しい励起および、これらの新しい励起と .Stim ステートメントで宣言された励起との関係を宣言します。

記述

.Device

device ステートメントは、システム内のデバイスを宣言するために使用されます。デバイスは、名前と所定のポート数を備えた（S パラメータで定義される）回路素子とみなされます。

.Device [Name] [NumPorts] [Type]

[Name] は英数字の文字列であり、システム内で宣言される各デバイスごとに一意である必要があります。
[NumPorts] はデバイスのポート数です。
[Type] はデバイスのタイプであり、以下のキーワードのいずれかです：

- File
- IdealThru
- IdealTermination
- IdealOpen
- IdealShort
- IdealPowerSplitter
- SingleEndedToMixedModeConverter
- R
- L
- C

File は、S パラメータファイルを利用する唯一のデバイス タイプです。理想的なものと仮定されるシステムコンポーネントについては、理想タイプを使用できます（ただし、コンポーネントが実際にこれらの理想タイプと一致しない場合、精度が低下します）。IdealTee のような一部の理想タイプは、真に理想タイプとして使用すべきものです（例えば、T 字分岐器は複数のデバイスポートを相互に接続しており、真に理想的です）。S パラメータファイルは、測定済みおよびモデル化されたデバイスに共通するインターフェースです。デバイスの VNA または TDNA 測定を行った場合は、それらを Touchstone 形式の S パラメータファイルとしてエクスポートし、ここで指定します。同様に、マイクロ波シミュレータや電磁界解析ソフトから得られたデバイスのモデルがある場合も、同様に Touchstone 形式の S パラメータファイルをエクスポートして、ここで使用します。ほぼすべてのマイクロ波モデリングおよびシミュレーションパッケージには、このエクスポート機能があります。

ファイル

デバイスタイプ **File** は、デバイス宣言内の file という単語の直後に続くファイル名から、そのデバイスの S パラメータを取得するようシステムに指示します。

ファイル名は二重引用符で囲む必要があります。

ファイルの拡張子は .s[NumPorts]p でなければならず、デバイス宣言で指定されたポート数と同じポート数を持つ S パラメータファイルであることを反映している必要があります。

例：

```
.Device transmissionLine 4 file "tline.s4p"
```

この例では、デバイスの名前は **transmissionLine** で、ポートは 4 つあり、S パラメータは **tline.s4p** というファイルから取得されています。ファイル拡張子の **4** は、宣言されたポート数と一致していることに注意してください。

S パラメータファイルのパスは、常にシステム記述ファイルを含むパスを基準とした相対パスとみなされます。つまり、ファイル名のみが指定されている場合、それらはシステム記述ファイルと同じディレクトリにあるファイルを指します。これにより、設計データを S パラメータファイルと同じディレクトリにまとめて保管でき、ディスク上の移動も容易になります。

IdealThru

デバイスタイプ *IdealThru* は、そのデバイスに対して理想スルーのモデルを使用するようシステムに指示します。理想スルーはポート数が偶数でなければならず、前半のポートは入力ポート、後半のポートは出力ポートとみなされます。理想スルーはすべてのポートで反射係数がゼロであり、1 つのポートにのみ直接接続され、他のすべてのポートから絶縁されています。

例：

```
.Device transmissionLine 4 IdealThru
```

この例では、**transmissionLine** という名前の 4 ポートセクションが作成されます。このセクションの S パラメータは次のようになります。

$$S = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

これは、信号がポート 1 から 3 へ、および 3 から 1 へと、反射損失や相互干渉なしに一意に伝搬することを意味します。同様に、信号はポート 2 から 4 へ、および 4 から 2 へと、反射損失なしに一意に伝搬します。

IdealTermination

デバイスタイプ *IdealTermination* は、システムに対して、理想的かつ完全な終端のモデルを使用するよう指示します。理想的な終端の S パラメータは、すべての周波数においてゼロとなります。

例：

```
.Device termination 2 IdealTermination
```

この例では、**termination** という名前の 2 ポートセクションが作成されます。このセクションの S パラメータは、すべての点で 0 になります。

IdealShort

デバイスタイプ *IdealShort* は、システムに対して、理想的かつ完全な短絡のモデルを使用するよう指示します。理想的な短絡の S パラメータは、ポートの反射係数（-1）を除き、すべての点でゼロになります。

P ポートの理想的な短絡の S パラメータは次の通りです：

$$S = \begin{bmatrix} -1 & 0 & \dots & 0 \\ 0 & -1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -1 \end{bmatrix}$$

例：

.Device termination 1 IdealShort

この例では、**termination** という名前の 2 ポートセクションが作成されます。このセクションの S パラメータは次のようになります。

$$S = [-1]$$

IdealOpen

デバイスタイプ *IdealOpen* は、理想的な完全開放のモデルを使用するようシステムに指示します。理想的な開放の S パラメータは、ポートの反射係数（1）を除き、すべてゼロです。

P ポートの理想的なオープン状態の S パラメータは次のとおりです：

$$S = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

例：

.Device termination 2 IdealOpen

この例では、**termination** という名前の 2 ポートセクションが作成されます。このセクションの S パラメータは次のようになります：

$$S = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

IdealPowerSplitter

デバイスタイプ「IdealPowerSplitter」は、システムに対し、理想的な完全抵抗性電力分配器のモデルを使用するよう指示します。理想的な抵抗性電力分配器とは、すべてのポートで入力に対して完全な整合を示し、あるポートからの電力を他のすべてのポートに均等に分配するデバイスです。このデバイスは対称です。

P ポートの理想電力分配器の S パラメータは以下の通りです：

$$S = \begin{bmatrix} 0 & \frac{1}{P-1} & \dots & \frac{1}{P-1} \\ \frac{1}{P-1} & 0 & \dots & \frac{1}{P-1} \\ \dots & \dots & \dots & \dots \\ \frac{1}{P-1} & \frac{1}{P-1} & \dots & 0 \end{bmatrix}$$

例：

.Device split0 3 IdealPowerSplitter

これにより、**split0** という名前の **3** ポートを持つデバイスが宣言され、その S パラメータは次のように指定されます：

$$S = \frac{1}{2}\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

SingleEndedToMixedModeConverter

デバイスタイプ *SingleEndedToMixedModeConverter* は、シングルエンド信号を Mixed モード信号に変換するデバイスのモデルを使用するようシステムに指示します。このデバイスは、正確に 4 つのポートを持ちます。ポート 1 および 2 はシングルエンドです。ポート 3 および 4 は、それぞれ差動モードおよびコモンモードです。

SingleEndedToMixedModeConverter には、以下の S パラメータがあります：

$$S = \frac{1}{\sqrt{2}} \cdot \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & 1 \\ 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

例：

.Device modeconv 4 SingleEndedToMixedModeConverter

これにより、**modeconv** という名前の SingleEndedToMixedModeConverter が宣言されます。

このコンポーネントにはいくつかの重要な用途があることに注意してください。1 つは、Mixed モードの S パラメータが提供されているデバイスとのインターフェースとして使用できることです。もう 1 つは、Mixed モードをシングルエンドに変換するために使用できることです。 最後に、このデバイスを 2 つ直結で接続すると、差動ノードまたはコモンモードノードとして利用できます。つまり、このデバイスを 2 つ使用し、ピン 3 と 4 を互いに接続すると、その接続ノードを波形出力点として使用でき、差動信号またはコモンモード信号を直接プローブすることが可能になります。この場合、1 つの注意点があります。S パラメータを用いた差動電圧およびコモンモード信号の計算には、私たちが慣れているものとはわずかに異なる式が用いられます。通常、差動電圧は正の電圧から負の電圧を引いたものと捉えられます。さらに、コモンモード電圧は正の電圧と負の電圧の平均値であると捉えられます。Mixed モードの S パラメータでは、この差動電圧は 2 で除算され、コモンモード電圧は 2 倍されます。この複雑さを考慮する必要があるのは、コンバータのコモンモードまたは差動モードのピンを測定ノードまたは出力ノードとして使用する場合に限られます。

R、L、および C

デバイスタイプ R、L、および C は、システムに対して、理想的な抵抗、インダクタ、またはコンデンサのモデルを使用するよう指示します。R の後にはオーム単位の抵抗値が続きます。L の後にはヘンリー単位のインダクタンスが続きます。C の後にはファラッド単位の静電容量が続きます。これらは 1 ポートまたは 2 ポートを持つデバイスであり、1 ポートの場合、デバイスの片側がグランドに接続されていると仮定されます。 複素インピーダンス Z には、以下の S パラメータがあります：

$$S = \frac{\begin{bmatrix} Z & 2 \cdot Z0 \\ 2 \cdot Z0 & Z \end{bmatrix}}{Z + 2 \cdot Z0} \quad \text{（2 ポートの場合）}$$

$$\Gamma = \frac{Z - Z0}{Z + Z0} \quad \text{（1 ポートの場合）}$$

これら 3 つの場合について、この複素インピーダンスは次のように定義される：

$$Z = R \text{（抵抗）}$$

$$Z = j \cdot 2 \cdot \pi \cdot f \cdot L \text{（インダクタ）}$$

$$Z = \frac{1}{j \cdot 2 \cdot \pi \cdot f \cdot C} \text{ (コンデンサ)}$$

例：

.Device termp 2 R 50

これにより、2 本のリード線を持ち、抵抗値が 50 オームの **termp** という名前の抵抗が宣言されます。

注：抵抗デバイスは、0 オームの 1 ポートデバイスを宣言することで、ハードグラウンド接続を行うのにも使用できます。

.Node

ノードとは、あるデバイスのポートと別のデバイスのポートとの接続として定義され、回路内で電圧を持つ一意の点を表します。この電圧は、両方向に伝播する波のスケール調整された和であるとみなされます。

.Node [NodeName] [DeviceName0] [Port0] [DeviceName1] [Port1]...[DeviceNameN][PortN]

[NodeName] は英数字で構成され、システム内で宣言される各ノードごとに一意である必要があります。あるいは、ノード名にこだわらない場合は # を指定することもできます。# を指定した場合、システムコンパイラによってノードに任意の一意な名前が割り当てられます。ノード名はデバイス名と重複してはなりません。重複すると、予期しない動作を引き起こす可能性があります。

[DeviceName0] は最初のデバイスの名前です。
[Port0] は最初のデバイスのポート番号です。
[DeviceName1] は 2 番目のデバイスの名前です。
[Port1] は 2 番目のデバイスのポート番号です。
[DeviceNameN] は N 番目のデバイスの名前です。
[PortN] は N 番目のデバイスのポート番号です。

指定されたすべてのデバイスは、ファイル内のどこかで .Device ステートメントによって定義されている必要があります。また、指定されたすべてのポートは、指定されたデバイス内に存在している必要があります。

例：

.Device D1 2 file “source.s2p”
.Device TLine 4 ファイル “line.s4p”
.Node # D1 2 TLine 3

この例では、.Node コマンドを使用して、**D1** という名前のデバイスのポート **2** から **TLine** という名前のデバイスのポート **3** への接続として定義された、任意の名称の電圧ノードを宣言しています。

.Meas

システム定義の目的は、捕捉した測定波形に基づいて出力波形を生成するフィルタを生成することです。.Meas ステートメントは、測定対象となるノードを宣言します。つまり、システム内の電圧測定点を定義します（すなわち、オシロスコープによって捕捉された波形を定義するノードです）。

.Meas [NodeName]

[NodeName] は、測定対象となるノードの名前です。この名前は、どこかで .Node コマンドを使用して宣言されている必要があります。測定対象のノード名は既知である必要があるため、.Node ステートメントおよび .Meas ステートメント内のノード名は、# として宣言されてはいけません。

システム記述ファイルのコンパイルが完了すると、ダイアログの入力セクション（「入力/出力」を参照）に [NodeName] が表示されます。そこで、仮想プローブコンポーネントの適切な入力を入力波形ソースに接続するよう指示されます。

例：

.Node V1 D1 2 TLine 3
.Meas V1

この例では、**D1** のポート **2** を **Tline** のポート **3** に接続する **V1** という名前のノードがあります。**V1** は測定ノードとして宣言されています。

.Output
システム定義の目的は、捕捉した測定波形に基づいて出力波形を生成するフィルタを生成することです。.Output ステートメントは、生成される（つまり、出力される）ノードを宣言します。したがって、これはシステム内の電圧出力点を定義します（つまり、測定波形に基づいてオシロスコープが生成する波形を定義するノードです）。

.Output [NodeName]

[NodeName] は、出力されるノードの名前です。この名前は、どこかで .Node コマンドを使用して宣言されている必要があります。出力ノード名は既知である必要があるため、.Node ステートメントおよび .Meas ステートメント内のノード名は、# として宣言されてはいけません。

システム記述ファイルのコンパイルが完了すると、[NodeName] がダイアログの出力セクション（Input/Output を参照）に表示され、そこで仮想プローブコンポーネントからの適切な出力を下流のプロセッサに接続するよう指示されます。

例：

```
.Node VP T 1 TLine 2
.Output VP
```

この例では、T のポート **1** と **Tline** のポート **2** を接続する **VP** という名前のノードがあります。**VP** は出力ノード（つまり、このシステムによって波形が提供されるノード）として宣言されています。

.Stim
励起（stimulus）とは、デバイスから発せられる電力波として定義されます。これは、システム内の波の発生源がどこにあるかをバーチャルプロービングコンポーネントに伝えるために使用されます。たとえば、シリアルデータ信号をプロービングする場合、波は通常、どこかのトランスミッタから発せられます。

```
.Stim [Name] [Device] [Port]
```

[Name] は英数字で構成され、システム内で宣言される励起ごとに一意である必要があります。

[Device] は、励起を発するデバイスの名前です。デバイスの名前は、.Device ステートメントを使用してどこかで宣言されている必要があります。

[Port] は、励起を発するデバイスのポート番号です。このポート番号は、どこかで .Device ステートメントを使用して宣言された有効なデバイスポートに対応している必要があります。

例：

```
.Device D1 2 file “source.s2p”
.Stim p D1 1
```

この例では、.Stim 文によって、p という名前の励起が、

D1 という名前のデバイスのポート 1 から、p という名前の励起が発信されることを宣言しています。

.StimDef
.StimDef は、このシステムの中で最も複雑な部分です。

このシステムでは、測定ノード（.Meas ステートメントを使用）、少なくとも 1 つの出力ノード（.Output ステートメントを使用）、および励起（.Stim ステートメントを使用）を宣言します。このシステムの目的は、.Meas ステートメントで宣言されたノードで測定された波形に基づいて、.Output ステートメントで宣言されたノードで波形を生成することです。システム定義コンパイラがこれを実行できるのは、測定ノードの数が適用される励起の数と一致する場合に限られます。多くの場合、これ

は実現できません。その好例として、システムの特定の点で（おそらく差動プローブを用いて）差動電圧が測定されているにもかかわらず、システム内にトランスミッタのプラス側とマイナス側の 2 つの励起を持つ差動ソースが宣言されている場合が挙げられます。

この場合、2 つの自由度（2 つの励起）があるのに対し、測定は 1 つ（差動測定）しかありません。この問題を解決するには、別の測定を追加するか（これは不可能な場合もあります）、あるいは仮定を立てて自由度の数を減らす必要があります。自由度は.StimDef ステートメントで設定されます。

差動測定の場合、ソースのバランスを別途測定するか、あるいはそれがバランスが取れていると仮定する（多くの場合、妥当な仮定である）のが合理的であると思われます。

差動トランスミッタから p と m という 2 つの励起が発信され、VMD という単一の差動測定点があるとします。新しい励起（例えば VDT）を定義し、p を ½ VDT、m を –½ VDT と定めることで、自由度を 2 から 1 に減らすことができ、それによって駆動が平衡状態にあると仮定できます。この状況では、新しい励起 VDT と他の 2 つの他の 2 つの励起 p と m との間に、ある関係性を定義したことになります。この定義された関係性は、システム内に VDT という名前の実際の励起が 1 つしか存在せず（他の励起はこの励起に基づいて制約される）、したがって自由度は 1 つしかないことを意味します。もちろん、結果はこの仮定の妥当性に依存することになります。

この連立方程式を解くために必要な励起と、.Stim ステートメントで宣言された励起との関係は、次のように定義されます：

> .StimDef [F0] [F1] … [FN] defines [S0] [S1] … [SM] as [relS0F0] [relS0F1] … [relS0FN] … [relS1F0] [relS1F1] … [relS1FN] … [relSMF0] [relSMF1] … [relSMFN]

[F0] は新しい励起 **0** の名前です。
[F1] は新しい励起 **1** の名前です。
[FN] は励起 **N** の名前です。
[S0] は、どこかで .Stim ステートメントを用いて宣言された励起 **0** の名前です。
[S1] は、どこかで .Stim ステートメントを用いて宣言された励起 **1** の名前です。
[SM] は、どこかで .Stim ステートメントを用いて宣言された励起 **M** の名前です。
[relS0F0] は、新しい励起 **F0** と定義済み励起 **S0** との関係です。
[relS0F1] は、新しい励起 **F1** と定義済み励起 **S0** との関係です。
[relS0FN] は、新しい励起 **FN** と定義済み励起 **S0** との関係です。
[relS1F0] は、新しい励起 **F0** と定義済み励起 **S1** との関係です。
[relS1F1] は、新しい励起 **F1** と定義済み励起 **S1** との関係です。
[relS1FN] は、新しい励起 **FN** と定義済み励起 **S1** との関係です。
[relSMF0] は、新しい励起 **F0** と定義済み励起 **SM** との関係です。
[relSMF1] は、新しい励起 **F1** と定義済み励起 **SM** との関係です。
[relSMFN] は、新しい励起 **FN** と定義済み励起 **SM** との間の関係です。

この記述は、以下の関係を意味します：

$$\begin{bmatrix} S0 \\ S1 \\ \dots \\ SM \end{bmatrix} = \begin{bmatrix} relS0F0 & relS0F1 & \dots & relS0FN \\ relS1F0 & relS1F1 & \dots & relS1FN \\ \dots & \dots & \dots & \dots \\ relSMF0 & relSMF1 & \dots & relSMFN \end{bmatrix} \cdot \begin{bmatrix} F0 \\ F1 \\ \dots \\ FN \end{bmatrix}$$

例：
あるシステム定義内に、同じ励起によって駆動されると想定される 2 つの同一のソースがあるとします。これら 2 つのソースは 2 つのデバイスとして宣言されるため、励起名は一意である必要があることから、すべての励起名が異なるものになります。例えば：

.Device Source0 2 file “source.s2p”
.Device Source1 2 file “source.s2p”
.Stim pe Source0 1
.Stim me Source0 2
.Stim pm Source1 1
.Stim mm Source1 2

このシステムには 4 つの励起があります。さらに、システム内の 2 つのノードで測定を行うと仮定しましょう。このシステムの自由度を 4 から 2 に減らす必要があります。しかし、ソースは同じ励起を生成しているとすでに述べました。この場合、pe は pm と同一であり、mm は me と同一であると仮定しましょう。p と m という 2 つの新しい励起を定義し、4 つの方程式を定式化します：

Pe=p、pm=p、me=m、mm=m。

行列形式では、次のようになります：

$$\begin{bmatrix} pe \\ pm \\ me \\ mm \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} p \\ m \end{bmatrix}$$

したがって、この関係は次のステートメントで宣言することになります：

.StimDef p m defines pe pm me mm as 1 0 1 0 0 1 0 1

このステートメントは、元の 4 つの自由度（pe、pm、me、mm）を 2 つの自由度（p および m）に削減するため、2 つの測定ノードを使用することが可能になります。

システムの定義

システムを定義するために必要な手順は次のとおりです：

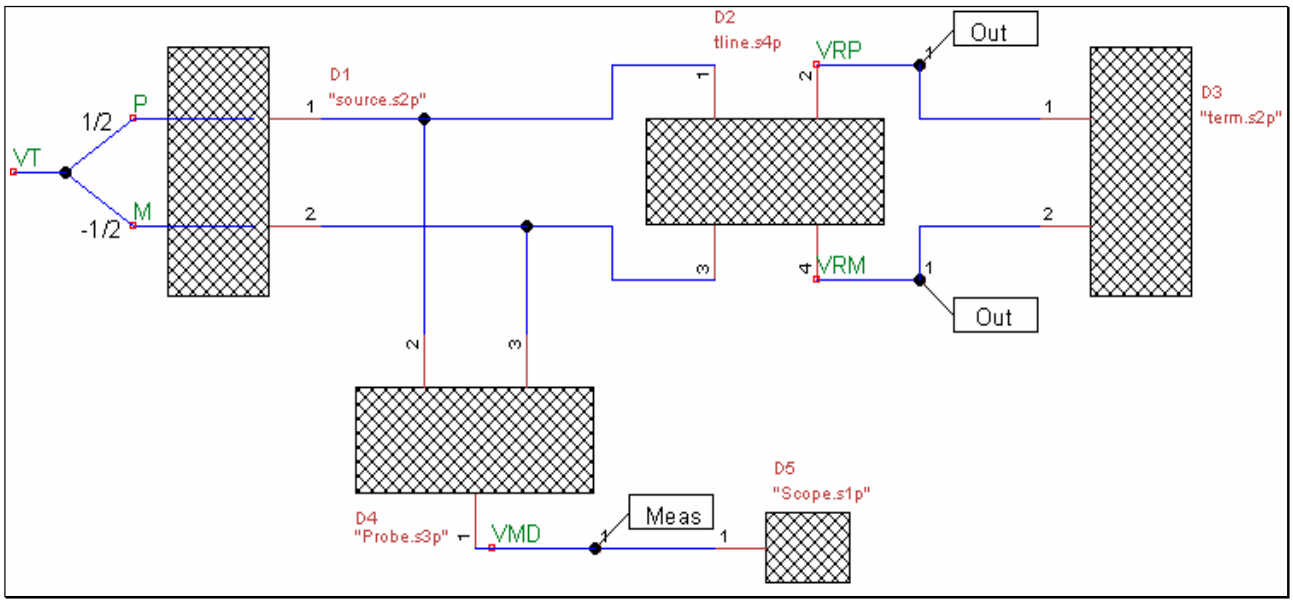
1. システム内のすべてのデバイスに対してボックスを描画します。
2. 描画したすべてのデバイスについて定義を記述します（通常は S パラメータファイルのファイル名）。
3. 描画したデバイスのポート番号にラベルを付けます（ポート番号が S パラメータファイル内のポートと一致していることを確認してください）。
4. デバイスのポートから他のデバイスのポートへと接続線を描画します。
5. デバイス内部からポートへと伸びる矢印として励起を描画し、各励起に一意の名前を付けます。
6. 測定ノードにラベルを付けます。
7. 出力ノードにラベルを付けます。
8. 測定ノードの数に応じた新しい励起を描画し、これらの新しい励起から、先に描画した励起へと重み付きの線を描画します。

このように描画されたシステムは、システム定義ファイルに変換できます。

1. システム図に示されているすべてのデバイスについて、デバイス宣言を記述します。T 字接続部の宣言も必ず含めるようにしてください。
2. すべてのデバイス接続について、ノード宣言を作成してください。ノードが測定対象でも出力先でもない場合は、名前に #を使用してください。
3. デバイスのポートから発信される各励起について、励起宣言を記述してください。
4. 測定対象のノードについて、測定ノード宣言を作成してください。
5. 出力ノードについては、出力ノード宣言を作成してください。
6. 新しい励起と、励起宣言で定義された励起との関係を定義する **stimdef** を作成してください。

例 1

トランスミッタで差動ラインを直接プローブし、システムにレシーバで検出される差動信号を生成させたいとします。システムの図は次のようになるでしょう：



この図を作成するにあたり、前述の手順に従いました：

1. システム内のすべてのデバイスについてボックスを描画しました：
    a. D1 – 差動ソース
    b. D2 – 差動伝送線路
    c. D3 – 差動終端
    d. D4 – プローブ
    e. D5 – オシロスコープ
2. すべてのデバイスの S パラメータを含むファイル名を記述。
3. 描画したデバイスのポート番号にラベルを付けました。なお、伝送線の S パラメータを確認したところ、ポート **1** と **3** が近端、ポート **2** と **4** が遠端を表していることが判明したため、ファイル内の測定値とポート番号が一致するように、シンボルを回転させる必要がありました。
4. デバイスのポートから他のデバイスのポートへの接続線を描画しました。
5. 差動ソースから発せられている励起 **p** および **m** を描画しました。
6. オシロスコープの入力にある測定ノード **VMD** にラベルを付けました。
7. 終端部にある出力ノード **VRP** および **VRM** にラベルを付けました。
8. 励起 **P** を½ **VT**、励起 **M** を–½ **VT** と定義する励起 **VT** を描画しました。

この図をシステム定義ファイルに変換すると、次のようになります：

```
.Device D1 2 File "source.s2p"
.Device D2 4 File "tline.s4p"
.Device D3 2 File "term.s2p"
.Device D4 3 File "Probe.s3p"
.Device D5 1 File "Scope.s1p"
.Node # D1 1 D4 2 D2 1
.Node # D1 2 D4 3 D2 3
.Node VRP D2 2 D3 1
.Output VRP
.Node VRM D2 4 D3 2
.Output VRM
.Node VMD D4 1 D5 1
.Meas VMD
.Stim P D1 1
.Stim M D1 2
.StimDef VT defines P M as 0.5 -0.5
```

このファイルの生成手順は以下の通りです：

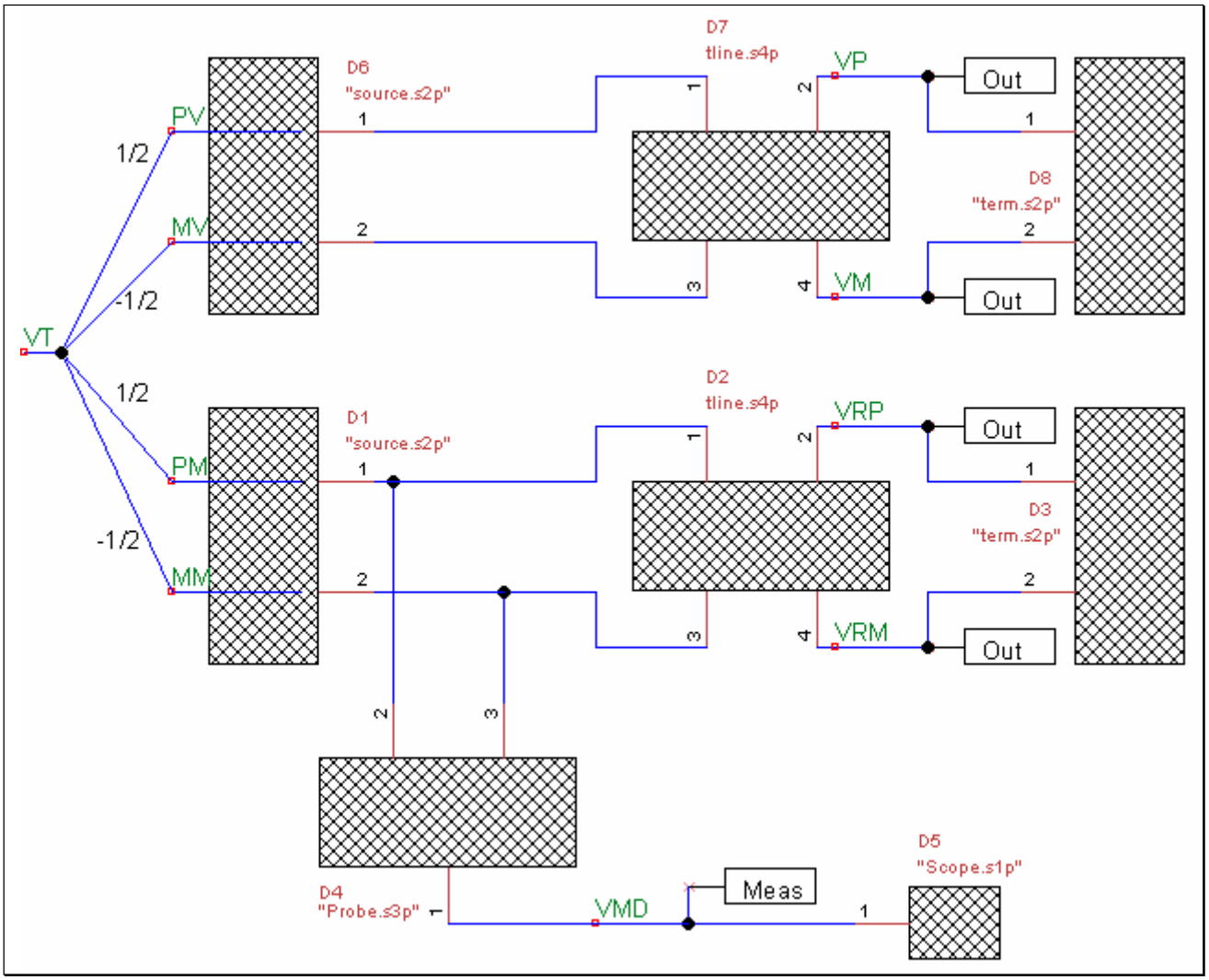
1. システム図に示されているすべてのデバイス（D1～D5）について、デバイス宣言を作成しました。
2. すべてのデバイス接続について、ノード宣言を作成しました。
3. D1 のポート 1 および 2 からそれぞれ発せられる励起 P および M について、励起宣言を作成しました。
4. 測定ノード VMD を宣言しました。
5. 出力ノード VRP および VRM を宣言しました。
6. 以下の関係式を用いて、P および M を定義する励起 VRM を宣言しました：

$$\begin{bmatrix} P \\ M \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{2} \end{bmatrix} \cdot VT$$

例 2

前の例では、トランスミッタにプローブを接続してレシーバを仮想的にプローブするシステムについて説明しました。この例では、プローブを挿入した状態でオシロスコープが受信した波形に基づいて、レシーバ側で出力波形が生成されたことに留意することが重要です。つまり、このような測定によって得られる波形は、プローブが回路に接続されている場合にのみ現れる波形です。プローブが回路に接続されていないときにレシーバで現れる波形を確認したいと仮定しましょう。

そのためには、図を次のように変更します：



この図では、以下の変更を加えていることに注意してください：

1. 仮想プローブを挿入したい条件を示す回路部分を複製しました。
2. 回路の新しい部分に、新しい出力ノードを作成しました。
3. 励起 **P** と **M** の名称を、仮想回路については **PV** と **MV** に、実測回路については **PM** と **MM** に変更しました。
4. すべての励起信号を **VT** に接続し直しました。

なお、この図において、VRP および VRM は以前と同じ波形（プローブが接続されている状態でのレシーバ波形）を生成しますが、VP および VM はプローブが接続されていない状態でレシーバで生成される波形であり、つまりこの測定ではプローブの影響が除去されていることを意味します。結果として得られる波形 VP および VM は、依然として回路にプローブを接続して捕捉した波形から生成されています。

この新しいシステムのシステム記述ファイルは次のようになります：

```
.Device D1 2 File "source.s2p"
.Device D2 4 File "tline.s4p"
.Device D3 2 File "term.s2p"
.Device D4 3 File "Probe.s3p"
.Device D5 1 File "Scope.s1p"
.Device D6 2 File "source.s2p"
.Device D7 4 File "tline.s4p"
.Device D8 2 File "term.s2p"
.Node # D1 1 D4 2 D2 1
.Node # D1 2 D2 3 D4 3
.Node VRP D2 2 D3 1
.Output VRP
.Node VRM D2 4 D3 2
.Output VRM
.Node VMD D4 1 D5 1
.Meas VMD
.Node # D6 1 D7 1
.Node VP D7 2 D8 1
.Output VP
.Node VM D7 4 D8 2
.Output VM
.Node # D7 3 D6 2
.Stim PM D1 1
.Stim MM D1 2
.Stim PV D6 1
.Stim MV D6 2
.StimDef VT defines PM MM PV MV as 0.5 -0.5 0.5 -0.5
```

ログメッセージ

メッセージ	意味
(Error) < File Parsing > File Not Found: [Filename]	システム記述ファイルの名前が見つかりませんでした。 是正措置として、ファイル名の綴りが正しいこと、および指定されたディレクトリにそのファイルが存在することを確認してください。
(Informational) < File Parsing > File: [FileName]	システム記述ファイルの名前が、解析中のファイルとして表示されています。
(Error) < File Parsing > Null Node Name	ノード名のないノード宣言が見つかりました。 宣言されるすべてのノードには、ノード名の後にデバイス名とデバイスポート番号、さらに少なくとも 1 つのデバイス名とデバイスポート番号が続く必要があります。 是正措置としては、ファイルを確認し、ノード名、デバイス名、デバイスポート番号、および少なくともう 1 つのデバイス名とデバイスポート番号が続いていない.Node ステートメントを特定することです。
(Error) < File Parsing > Null Device Port Number	ノード名の後に、デバイス名とデバイスポート番号のペアが続いていないノード宣言が見つかりました。 宣言されたすべてのノードには、ノード名の後にデバイス名とデバイスポート番号、および少なくともう 1 つのデバイス名とデバイスポート番号が続く必要があります。 是正措置としては、ファイルを確認し、.Node ステートメントを特定することです。 ノード宣言行にあるすべてのトークンは解釈されるため、デバイス名とポート番号は必ずペアで指定する必要があることに注意してください。
(Error) < File Parsing > Nodes must connect at least two device ports	宣言されたすべてのノードは、ノード名の後にデバイス名とデバイスポート番号、さらに少なくとも 1 つのデバイス名とデバイスポート番号が続く必要があります。 この基準を満たしていないノード宣言が見つかりました。 修正措置として、ファイルを確認し、.Node ステートメントを見つけ、その後にノード名、デバイス名、デバイスポート番号、および少なくともう 1 つのデバイス名とデバイスポート番号が続いていないものを特定してください。
(Informational) < File Parsing > Devices Counted: [NumDevices]	システム内のデバイス数が表示されます。
(Informational) < File Parsing > Nodes Counted: [NumNodes]	システム内のノード数が表示されます。
(Error) < File Parsing > No Devices	ファイル内にデバイスはありません。 是正措置として、ファイルを確認し、すべてのデバイスが .Device ステートメントを使用して宣言されていることを確認してください。
(Error) < File Parsing > Null Device Name	デバイス名のないデバイス宣言が見つかりました。 .Device ステートメントで宣言されるすべてのデバイスには、名前を指定する必要があります（この名前は、デバイスのポートにノードを割り当てるために必要です）。 是正措置として、ファイルを確認し、すべての .Device ステートメントの直後にデバイス名が指定されていることを確認してください。
(Error) < File Parsing > Number of Ports: [NumPorts] Invalid	ポート数が不正なデバイスが検出されました。

	これは通常、.Device 宣言の構文が正しくない場合、たとえばデバイス名が指定されていない場合や、デバイス名にスペースが含まれている場合に発生します。また、ポート数が 0 または負のポート数が宣言されている場合にも発生する可能性があります。
(Error) < File Parsing > Filename must be surrounded by quotes **FileName Given as: [line]**	S パラメータファイル名が指定されていますが、ファイル名を囲む引用符が見つかりませんでした。 すべての S パラメータファイル名は、引用符で囲む必要があります（ファイル名にスペースが含まれている場合を避けるため）。 修正方法として、すべての S パラメータファイル名を引用符で囲んでください。
(Error) < File Parsing > S-parameter file not read **FileName: [FileName]**	指定された S パラメータ ファイルが読み込まれませんでした。 ファイル名: [ファイル名]これは、ファイルが適切な Touchstone 形式ではない場合に発生します。ただし、Touchstone ファイルの形式には大きなばらつきがあります。 この問題を解決するには、Touchstone 形式に準拠した S パラメータファイルを作成してください。これは、ファイルが適切な Touchstone 形式ではない場合に発生します。ただし、Touchstone ファイルの外観には大きなばらつきがあります。 対処法としては、指定したファイルが指定したディレクトリに実際に存在すること、およびそれが Touchstone 形式のファイルであることを確認してください。それでも問題が解決しない場合は、# で始まる行（例：# MHz S MA R 50.00）と数値以外の情報をすべてファイルから削除してみてください。これで解決するものの、手間がかかる場合（例えば、すべての VNA ファイルの出力を編集する必要がある場合など）は、LeCroy にその旨をお知らせください。そうすれば、当該ファイル形式に適切に対応できるよう検討いたします。 もう 1 つ確認すべき点は、各周波数に対して指定された S パラメータの数がポート数と一致しているかどうかです。たとえば、拡張子が.s4p という拡張子を持つファイルは、4 ポートの S パラメータを定義しています。したがって、各周波数について、16 個の S パラメータが指定されている必要があります（通常、16 個の S パラメータの振幅と位相を表す 32 個の数値）。
(Error) < File Parsing > S-parameter file wrong number of ports **FileName: [FileName]** **Wanted: [NumPortsWanted] - Got: [NumPortsGotten]**	これは、S パラメータファイルのポート数が、デバイスに対して宣言されたポート数と一致していないことを意味します。たとえば、2 ポートのデバイス宣言には、拡張子が .s2p の S パラメータファイルが必要です。 修正方法として、宣言されたデバイスのポート数が正しいこと、および S パラメータファイルは、目的のデバイスポート数と正しく一致しています。S パラメータファイルの形式は拡張子によって示されるため、一般的に、S パラメータファイルの拡張子を変更するだけではこの問題を修正することはできません。
(Error) < File Parsing > IdealThru must have even number of ports	ポート数が偶数でない IdealThru が宣言されています。 Ideal Thru とは、ポートの前半と後半を相互に接続するデバイスです。たとえば、4 ポートの Ideal Thru では、ポート 1 とポート 3、およびポート 2 とポート 4 が接続されます。したがって、ポート数は偶数でなければなりません。 修正方法として、そのデバイスに IdealThru が必要かどうかを確認するか、デバイスのポート数を修正してください。

(Error) < File Parsing > SingleEndedToMixedModeConverter must have 4 ports	ポート数が 4 つではない SingleEndedToMixedModeConverter が宣言されています。 SingleEndedToMixedModeConverter には、2 つのシングルエンドポート（ポート 1 および 2）と、1 つの差動およびコモンモードポート（ポート 3 および 4）があり、したがってポート数は正確に 4 つです。 是正措置として、このデバイスが 4 つのポートを持つように宣言されていることを確認してください。
(Error) < File Parsing > No resistor value provided	抵抗値が指定されていない抵抗が宣言されています。是正措置として、抵抗値を指定してください。
(Error) < File Parsing > Wrong number of resistor ports	抵抗が、1 または 2 以外のポート数で宣言されています。 抵抗は、ポートを 1 つまたは 2 つとして宣言する必要があります。ポートを 1 つとして宣言した場合、抵抗のもう一方の端子はグランドに接続されているものとみなされます。 是正措置として、抵抗は 1 つまたは 2 つのポートを持つように宣言されていることを確認してください。
(Error) < File Parsing > No capacitor value provided	コンデンサが静電容量値なしで宣言されています。 是正措置として、静電容量の値を指定してください。
(Error) < File Parsing > Wrong number of capacitor ports	コンデンサが、1 つまたは 2 つ以外のポート数で宣言されています。 コンデンサは、ポートを 1 つまたは 2 つとして宣言する必要があります。ポートを 1 つとして宣言した場合、コンデンサのもう一方のリード線はグランドに接続されているものとみなされます。 是正措置として、コンデンサを 1 ポートまたは 2 ポートのいずれかで宣言するようにしてください。
(Error) < File Parsing > No inductor value provided	インダクタが、インダクタンス値なしで宣言されています。 是正措置として、インダクタンス値を指定してください。
(Error) < File Parsing > Wrong number of inductor ports	インダクタが、1 または 2 以外のポート数で宣言されています。 インダクタは、ポートを 1 つまたは 2 つとして宣言する必要があります。ポートを 1 つとして宣言した場合、インダクタのもう一方のリード線はグランドに接続されているものとみなされます。 是正措置として、インダクタが 1 つまたは 2 つのポートを持つように宣言されていることを確認してください。
(Error) < File Parsing > Unknown Device Type: [line]	未知のデバイスタイプを持つデバイスが指定されています。 是正措置として、スペルを修正し、既知のデバイスタイプのみが使用されるようにしてください。提供されていないデバイスタイプが必要な場合は、いつでもデバイスの S-パラメータを表す Touchstone 形式のファイルを作成し、そのファイルを指定することができます。
(Informational) < File Parsing > [NumDevices] Devices Assigned	すべてのデバイスに、ポート数、名前、およびタイプが割り当てられ、それらのデバイスの S パラメータが決定されました。
(Error) < File Parsing > Identical Device Name Found	同じ名前のデバイスが 2 つ見つかりました。すべてのデバイスには一意の名前を指定する必要があります。

	是正措置として、名前が重複しているデバイスを見つけ、一意の名前に変更してください。通常、.Node ステートメント内での参照も更新する必要があります。
(Error) < File Parsing > Total Number of Device Ports Must be Even	システム内のすべてのデバイスポートの合計数が偶数ではありません。 システム記述では、すべてのデバイスのすべてのポートが、別のデバイスの 1 つのポートのみに接続されている必要があります。これは、デバイスポートの数が偶数でない限り発生しません。 是正措置として、すべてのデバイスが適切な数のポートを備えていることを確認し、デバイスのすべてのポートを接続することを意図していることを確認してください。デバイスのポートを接続しないことを意図している場合は、タイプ IdealOpen 型の 1 ポートデバイスを宣言し、このデバイスのポートを接続対象のデバイスポートに接続してください。
(Error) < File Parsing > Number of Ports and Nodes don't match	デバイスポートの総数が、宣言されたノード数の 2 倍になっていません。 システム記述では、すべてのデバイスのすべてのポートが、別のデバイスの 1 つのポートにのみ接続されている必要があります。これは、デバイスポートの数が宣言されたノード数の 2 倍でない限り発生しません。 是正措置として、すべてのデバイスが正しい数のポートを持っていること、すべてのデバイスポートを接続するノード宣言がすべて行われていること、およびすべてのデバイスポートを接続する意図があることを確認してください。デバイスポートを未接続にする場合は、1 つの IdealOpen タイプのポートデバイスを設定し、このデバイスのポートを目的のデバイスポートに接続します。
(Error) < File Parsing > Bad Node Connection From [ConnectionFromPortName] [ConnectionFromPortNum] To [ConnectionToPortName][ConnectionToPortNum]	ノード接続が正しくありません。これは、以下のいずれかを意味します。 • .Node ステートメント内のデバイス名のいずれかが見つかりませんでした • .Node ステートメント内のポート番号のいずれかが、デバイスに見つからないポートを参照している 是正措置として、指定されたデバイス名が実際に存在すること、および指定されたポート番号がデバイス内に存在することを確認することです。
(Informational) < File Parsing > [NumNodes]	すべてのノードがデバイスに正常に割り当てられました。
(Error) < File Parsing > Device Port Not Connected Device:[DeviceName] Port:[PortNum]	接続されていないデバイスのポートが検出されました。 是正措置として、表示されたポート番号とデバイス名を確認し、.Node ステートメントでそのデバイス名からそのポート番号への接続が正しく指定されていることを確認してください。デバイスのポートを未接続のままにしておくことはできません。実際にオープンにしておくことが意図されている場合は、1 ポートの ideal open を宣言し、その理想的なオープンに接続してください。
(Error) < File Parsing > Device Port Connection Incorrect Device:[DeviceName] Port:[PortNum] - to node labelled [NodeName]?	これは通常、内部エラーです（つまり、他のすべてが正常に動作していれば発生しないはずです）。これは、ノードの割り当て後、そのノードがデバイスに接続されている状態が見つからないことを意味します。
(Informational) < File Parsing > Device Connections and Nodes Verified	すべてのデバイスのすべてのポートが、ノードに正しく割り当てられています。

(Error) < File Parsing > Stimulus with no name found	.Stim ステートメントが宣言されていますが、名前が指定されていません。 修正方法として、ファイル内の .Stim ステートメントを見つけ、その励起に名前が割り当てられていることを確認してください。
(Error) < File Parsing > Stimulus [StimName] - no device name found	.Stim ステートメントで指定されたデバイスが見つかりませんでした。 是正措置として、指定されたデバイスのスペルを確認してください。
(Error) < File Parsing > Stimulus [StimName] invalid device/port assignment To Device:[DeviceName] Port:[PortNum]	.Stim ステートメントで指定されたデバイスポートが存在しません。 是正措置として、指定されたデバイスのスペルが正しいこと、および参照されているポート番号がデバイス内に存在することを確認してください。
(Error) < File Parsing > No Stimuli assigned	.Stim ステートメントが見つかりませんでした。 修正方法として、.Stim ステートメントを使用して励起を割り当ててください。
(Informational) < File Parsing > [NumStims] Stimuli Assigned	.Stim ステートメントで定義された励起は正常に割り当てられました。
(Informational) < File Parsing > System Defined	システムが定義されました。 これは、印加された励起に関して、さまざまなノードでの電圧の計算を可能にするシステムが存在することを意味します。ただし、これは、出力ノードの電圧が入力ノードの電圧を用いて定義できることを意味するものではありません。
(Error) < File Parsing > System Not Defined	システムを定義できませんでした。 このエラーには、このエラーの原因となった他のエラーが併発しているはずです。是正措置として、それらの他のエラーを修正してください。
(Error) < File Parsing > .Meas declaration with no node name	ノード名を指定していない .Meas ステートメントが検出されました。 すべての .Meas ステートメントは、システム内の有効なノードを指定するノード名で構成されている必要があります。 修正方法として、ノード名が指定されていない .Meas ステートメントを見つけ、有効なノード名が指定されていることを確認してください。
(Error) < File Parsing > .Output declaration with no node name	ノード名の指定がない .Output ステートメントが見つかりました。 すべての .Output ステートメントは、システム内の有効なノードを指定するノード名で構成されている必要があります。(エラー) < ファイルの解析 > ノード名のない .Output 宣言ノード名のない .Output ステートメントが見つかりました。 是正措置としては、ノード名が指定されていない .Output ステートメントを見つけ、有効なノード名が指定されていることを確認することです。
(Error) < File Parsing > Multiple .Stimdef statements found	システム記述ファイルには、.Stimdef ステートメントは 1 つしか存在できません。 是正措置として、.Stimdef ステートメントが 1 つだけあることを確認してください。
(Error) < File Parsing > StimDef with No Actual Stim names	.Stimdef ステートメントと、そのステートメント内のキーワード「defines」の間に、励起の名前が指定されていない .Stimdef ステートメントが見つかりました。 是正措置として、.Stimdef ステートメントに励起を追加してください。

(Error) < File Parsing > Stimdef missing 'defines' keyword	「defines」キーワードを含まない .Stimdef ステートメントが検出されました。 すべての .Stimdef ステートメントには、キーワード「defines」および「as」が含まれている必要があります。 修正方法として、.Stimdef ステートメントが正しい形式で記述されていることを確認してください。
(Error) < File Parsing > StimDef with No Stim names	「defines」と「as」の間に励起名がない .Stimdef ステートメントが見つかりました。
(Error) < File Parsing > Stimdef missing 'as' keyword	「as」キーワードを含まない .Stimdef 文が見つかりました。 すべての .Stimdef ステートメントには、キーワード「defines」および「as」を含める必要があります。 修正措置として、.Stimdef ステートメントが正しい形式で記述されていることを確認してください。
(Error) < File Parsing > Stimdef with incorrect numbers defining stimuli relationship	「as」キーワードを含まない .Stimdef ステートメントが見つかりました。 すべての .Stimdef ステートメントには、キーワード「defines」および「as」が含まれている必要があります。さらに、.Stimdef と「defines」キーワードの間に含まれるキーワードの数は、自由度の数を定義します。「defines」と「as」キーワードの間に含まれるキーワードの数は、励起の数を定義します。「as」キーワードに続く数字は、これらの自由度と励起との関係を定義します。この関係を定義する数字の数は、自由度の数に励起の数を掛けた数と等しくなければなりません。 是正措置として、.Stimdef 文が正しい形式で記述されていることを確認してください。
(Error) < File Parsing > No Measures Found	.Meas ステートメントは見つかりませんでした。 是正措置として、実際の波形捕捉が行われるノードを指定する .Meas ステートメントを挿入してください。
(Error) < File Parsing > No Outputs Found	.Output ステートメントが見つかりませんでした。 修正方法として、波形の出力を表示したいノードを指定する .Output ステートメントを挿入してください。
(Informational) < File Parsing > Measures Counted: [NumMeas]	測定が検出され、割り当てられました。
(Informational) < File Parsing > Outputs Counted: [NumOut]	出力が見つかり、割り当てられました。
(Informational) < File Parsing > Degrees of Freedom Specified: [NumDeg] With Respect to [NumStims]	システムの自由度数は、励起の数に基づいて（.StimDef ステートメントから）決定されています。
(Informational) < File Parsing > Measures, Outputs and StimDefs Assigned	測定ノード、出力ノード、および励起が割り当てられました。
(Informational) < System Def > Stimulus and Node Vector Built	電圧ノード名と励起のベクトルが構築されました。
(Informational) < System Def > System Characteristics Matrix Built and Inverted	各周波数について、指定された S パラメータからシステム特性行列が決定され、逆行列が算出されました。
(Informational) < System Def > System Characteristics Matrix and Stimuli Vector Reduced	各周波数に対する S パラメータ行列が構築され、逆行列が求められた後、励起の数に基づいて、行列は順次簡約されます。これは計算の最適化です。

(Informational) < System Def > Node Voltage Vector Built	反転･縮約されたシステム特性行列に、電圧抽出行列およびノード電圧ベクトルを乗算します。ノード電圧ベクトルは構築されました。
(Informational) < System Def > Voltage Extraction Matrix Built	逆行列化された簡約系特性行列に、電圧抽出行列およびノード電圧ベクトルを乗算しました。電圧抽出行列が構築されました。電圧抽出行列とは、系内の a 波および b 波を実際の電圧に変換する行列です。
(Error) < System Def > Measure Voltage [VoltageName] Not Found	.Meas ステートメントで宣言された電圧ノードが見つかりませんでした。 対処法として、指定されたノード名のスペルが正しいこと、およびそのノードが存在することを確認してください。
(Error) < System Def > Matrix Not Invertible	測定ノードの数が、.StimDef ステートメントで定義された励起の数と一致していません。フィルタを生成するには、.StimDef ステートメントの直後（「defines」という単語の前）に定義された励起の数が、.Meas ノードの数と完全に一致している必要があります。 現在、システムは過制約の場合（測定ノードの数が励起の数を超える場合）を処理できません。 修正方法としては、測定ノードの数を増減させるか、あるいは励起の数を増減させることで必要な条件を満たすようにします。
(Error) < System Def > Output Voltage [VoltageName] Not Found	.Output ステートメントで宣言された電圧ノードが見つかりませんでした。 修正方法としては、指定されたノード名のスペルが正しいこと、およびそのノードが存在することを確認してください。
(Informational) < System Def > Writing Responses	フィルタを定義する周波数応答をディスクに書き込んでいます。
(Informational) < System Def > Response: [ResponseFileName]	特定のフィルタを定義する特定の周波数応答をディスクに書き込みます。
(Informational) < System Def > Writing Filters	フィルタのインパルス応答をディスクに書き込みます。
(Informational) < System Def > Filter: [FilterFileName]	特定のインパルス応答をディスクに書き込みます。
(Informational) < Filters > Reading Filters	ディスクからフィルタを読み込みます。
(Informational) < Filters > Output Selection: [OutputName]	選択された出力に名前を付けます。
(Informational) < Filters > Filter: [FilterFileName]	読み込まれているディスク上のフィルタの名前を指定します。

S パラメータ ファイル

Touchstone フォーマット

Touchstone は、S パラメータを指定するために使用されるファイル形式です。これは、ベクトルネットワークアナライザ（VNA）、時間領域ネットワークアナライザ（TDNA）、およびマイクロ波シミュレータなどの EDA ツールで広く採用されている標準規格です。バーチャルプロービングコンポーネントは、Touchstone ファイルを次のように解釈します。

1. 拡張子は“.s[N]p”という形式でなければなりません。ここで、[N]はデバイスのポート数に対応します。たとえば、2 ポートの S パラメータファイルの拡張子は“.s2p”となります。
2. 1 行あたりの文字数は 2000 文字を超えてはなりません。
3. ‘!’で始まる行はコメント行です。コメントはファイルの先頭に記述する必要があります。周波数や S パラメータの値の記述が始まると、それ以降の行にコメントを記述することはできません。
4. ‘#’で始まる行が少なくとも 1 行ある必要があります。この行には、S パラメータの解釈に役立つトークンが含まれます。有効なトークンは以下の通りです：

a. Hz – 周波数は Hz 単位です。
b. MHz – 周波数は MHz 単位です。
c. GHz – 周波数は GHz 単位です。
d. MA – S パラメータは振幅/角度形式で、振幅は真の振幅、角度は度単位で表されます。
e. RI – S パラメータは実部・虚部形式で表されます。
f. DB – S パラメータは絶対値/位相形式で、絶対値はデシベル単位、位相は度単位で表されます。

5. 上記のトークンが見つからない場合、周波数は MHz 単位、S パラメータは振幅/位相形式であるとみなされます。
6. 各周波数の S パラメータは、[周波数] (S11) (S12) …(S1[N]) (S21) (S22) … (S2[N]) … (S[N]1) (S[N]2) … (S[N][N]) の形式で記載されているものとみなされます。ここで、N はファイル拡張子によって規定されるポート数です。ただし、2 ポートの S パラメータについては、規格に従い [周波数] (S11) (S21) (S12) (S22) の形式で記載されているものとみなされます。
7. すべての S パラメータは 50 オームであると仮定します。S パラメータファイルでは、「#」で始まる行に、通常「R [Z0]」というトークンのペアがあります。ここで、[Z0]は特性インピーダンスを表します。ただし Z0 は 50 と仮定し、これらのトークンは無視します。
8. すべての周波数は、リストの最初の周波数から最後の周波数まで、一定の周波数間隔で並んでいるものとみなします。

S パラメータファイルの解釈および使用に関するその他の注意事項

仮想プローブに提供される S パラメータは、分解能（周波数間隔）と範囲（開始周波数と終了周波数）の両方で異なる、さまざまな周波数スケールを持つ傾向があります。仮想プローブコンポーネントは、すべての S パラメータセットを、システムのサンプリングレート（サンプリングレートの半分が最終周波数となる）および時間長（「時間長」を参照）によって決定される新しい周波数スケールにリサンプリングします。ここで、時間長によって新しい周波数分解能が決まります。このリサンプリングを行う際、分解能が元の分解能よりも高くなったり低くなったりする可能性があり、また周波数範囲によってはデータの外挿が必要になる場合があることを理解しておくことが重要です。周波数範囲が S パラメータデータで提供されている範囲よりも広い場合（つまり、S パラメータがシステムサンプリングレートの半分まで提供されていない場合）、これらの追加で必要なデータポイントにおいて、S パラメータデータはゼロに設定されます。

さらに、VNA は DC までの測定を行わないため、ほとんどの S パラメータデータは DC まで及んでいません。そのため、DC 応答についても、低周波数側の測定点を S パラメータデータで提供されている最初の周波数と同じ値に設定することで外挿されます。

つまり、仮想プローブの有効性を最大限に引き出すためには、いくつかの簡単なルールに従うことが役立ちます。

1. DC 外挿誤差を最小限に抑えるため、VNA が許容する限り低い周波数での S パラメータデータを取得してください。
2. S パラメータデータは、システムのサンプリングレートの半分までの周波数範囲で取得してください。つまり、仮想プローブに 40 GS/s の波形を供給する場合、20 GHz までのデータを提供するのが望ましいです。逆に、システムに供給される信号の周波数成分がそれよりはるかに低い場合は、そもそも高いサンプリングレートでサンプリングする必要はありません。
3. 十分な周波数分解能で S パラメータデータを取得してください。必要な分解能は、対象となる回路素子の電気的長さによって異なります。つまり、長さ 5 ns のケーブルがあり、システム内で反射が発生している場合、最小周波数分解能の 1/5 ns = 200 MHz の 10 分の 1 程度（例えば 20 MHz）でデータを取得するのが適切です。別の言い方をすれば、より実用的な観点から言えば、S パラメータに周波数幅の狭い突起や変動が多数ある場合、これらの突起や変動を捉えるためには、適切な周波数分解能で S パラメータデータを測定する必要があります。通常、回路素子の電気的長さが長くなるにつれて、これらの突起や変動の周波数幅は狭くなります。

Mixed モード S パラメータ

Mixed モード S パラメータは、差動システムの解析を支援するために考案された S パラメータです。差動システムにおいては、シングルエンド S パラメータよりもこれらを検証する方がより有意義です。シングルエンド S パラメータと Mixed モード S パラメータの間には等価性があり、適切に使用すれば、シミュレーションやバーチャルプロービングにおいて（前述のパラメータの検証以外で）Mixed モード S パラメータを使用する必要はありません。とはいえ、利用可能な S パラメータが Mixed モードのものしかない場合もしばしばあります。例えば、多くの VNA は、指示があれば S パラメータを Mixed モードで出力します。さらに、一部のシステムでは Mixed モードパラメータの一部しか測定しないため、ここで少し触れておく価値があります。

差動伝送線路などの 4 ポートデバイスは、シングルエンド S パラメータとして 4×4 行列で表されます。2 本の差動線路が結合していない場合を除き、個々の S パラメータだけでは解析にあまり役立ちません。伝送線が結合している場合、解析を行うにはそれらを Mixed モード S パラメータに変換する必要があります。これはよく知られた式を用いて行われ、その結果として別の 4×4 行列が得られます。この行列は、さらに 2×2 の S パラメータからなる 4 つの象限として考えることもできます。

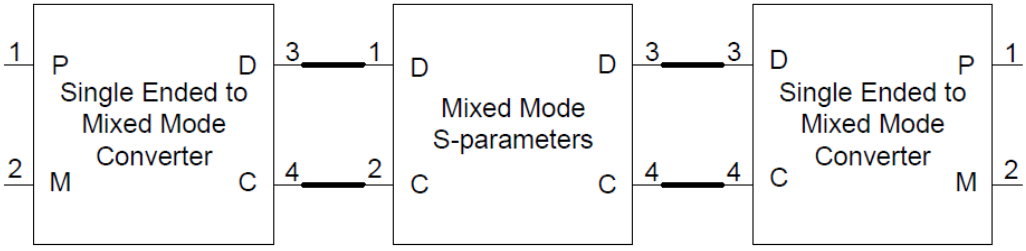
シングルエンド S パラメータでは、S パラメータは 4 つのポートそれぞれにおける入射波と反射波の関係を表すものとみなします。Mixed モード S パラメータでは、シングルエンドネットワークの左端の 2 つのポートと右端の 2 つのポートが、差動信号の正線と負線を表すと仮定します。Mixed モード S パラメータでは、左端の 2 つのポートと右端の 2 つのポートを差動モード成分と共通モード成分と見なし、さらに Mixed モード S パラメータが、入射および反射の差動モード成分と共通モード成分間の関係を表していると捉えます。場合によっては、簡略化が行われ、コモンモードポートを無視（存在しないものと仮定）し、差動モードのみを考慮するとともに、さらに「モード変換」が起こらないと仮定することがあります。この簡略化では、差動ラインを 2 つの差動モードポートを持つ 2 ポートデバイスとして捉えます。この場合の 2 ポート S パラメータは、ネットワークの差動-差動モード応答を表します。

話を進める前に、この仮定は差動線の挙動を調査する際に有用な場合もあるものの、仮想プローブ素子を使用する際には、このような過度な単純化が必要となることはほとんどないことを改めて強調しておきます。したがって、シングルエンドの 4 ポート S パラメータ表現の使用を推奨します。

Mixed モードの S パラメータが提供される場合、いくつかの非常に重要な問題や考慮事項を念頭に置く必要があります。

1. 差動モードの S パラメータの基準インピーダンスは 100 オームと仮定されます（すなわち、完全な差動整合とは、100 オームの差動終端を意味します）。
2. コモンモード S パラメータの基準インピーダンスは 25 オームと仮定されます（すなわち、完全なコモンモード整合とは、25 オームのコモンモード終端を意味します）。
3. 差動モードとコモンモードのポートを確実に識別する必要があります。Mixed モードの S パラメータでは、シングルエンドの S パラメータに比べて、これらを区別するのがより困難です。

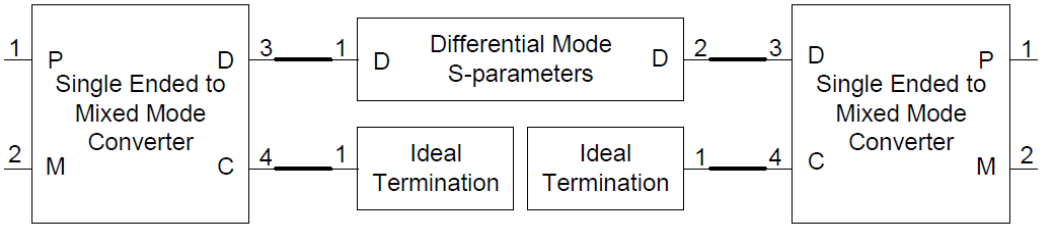
この形式の Mixed モード S パラメータが提供されている場合、シングルエンドから Mixed モードへのコンバータ（SingleEndedToMixedModeConverter を参照）。例えば、「mixedmode.s4p」という名前の 4 ポート Mixed モード S パラメータを含むファイルがある場合、以下のシステム定義ファイルの宣言を使用します：



.Device D1 4 file “mixedmode.s4p”
.Device D2 4 SingleEndedToMixedModeConverter
.Device D3 4 SingleEndedToMixedModeConverter
.Node # D1 1 D2 3
.Node # D1 2 D2 4
.Node # D1 3 D3 3
.Node # D1 4 D3 4

このようにして、デバイスポート D2 1、D2 2、D3 1、および D3 2 は、通常通り仮想プローブで使用可能な 4 ポートのシングルエンドネットワークコンポーネントの周辺部のように見えます。

しかし、ファイル“differentialmode.s2p”には 2 ポートの差動モード S パラメータしか提供されていないと仮定します。このコンポーネントを使用すると、コモンモード信号成分だけでなくモード変換も無視されるため、精度が低下するリスクがあることを理解した上で、次のように使用できます：



.Device D1 2 file “differential.s2p”
.Device D2 4 SingleEndedToMixedModeConverter
.Device D3 4 SingleEndedToMixedModeConverter
.Device D4 1 IdealTermination
.Device D5 1 IdealTermination
.Node # D1 1 D2 3
.Node # D1 2 D3 3
.Node # D2 4 D4 1
.Node # D3 4 D5 1

このようにして、デバイスポート D2 1、D2 2、D3 1、および D3 2 は、通常通り仮想プローブで使用可能な 4 ポートのシングルエンドネットワークコンポーネントの周辺部のように見えます。ただし、ポート 1 および 2 における入射波のコモンモード成分はポート 3 および 4 には伝播せず、その逆も同様であることに留意してください。

Mixed モードの S パラメータは、シミュレーションやバーチャルプロービングにおいてミスを招きやすく、差動ネットワーク素子を 2 ポートの差動モードのみの観点から扱うことは、間違いなく危険です。信号が差動成分のみを含む場合（すなわち、完全に平衡化されたトランスミッタ）であっても、通常、トランスミッタまたはレシーバの終端のいずれかで何らかのモード変換が生じ、コモンモード信号が存在し、それが差動モード信号に何らかの影響を与えることに留意することが重要です。

システム の概要

バーチャルプロービングにおいて最も重要かつ最も抽象的な概念は、解決すべき問題を記述するシステム図です。このセクションでは、後にシステム記述ファイルに変換されるこの図の作成方法について解説します。

この図の作成方法を最も理解しやすくするためには、問題を次の 2 つの要素に分けて考えることが役立ちます。

1. 測定構成 – 波形がシステム内をどのように伝わり、オシロスコープに表示されるかを記述します。
2. 出力構成 – システムから得られる望ましい出力波形を記述します。

測定設定と出力設定の両方が作成されると、それらは関連する励起を通じて結びつけられます。

測定構成

最初の概念は以下の通りです：

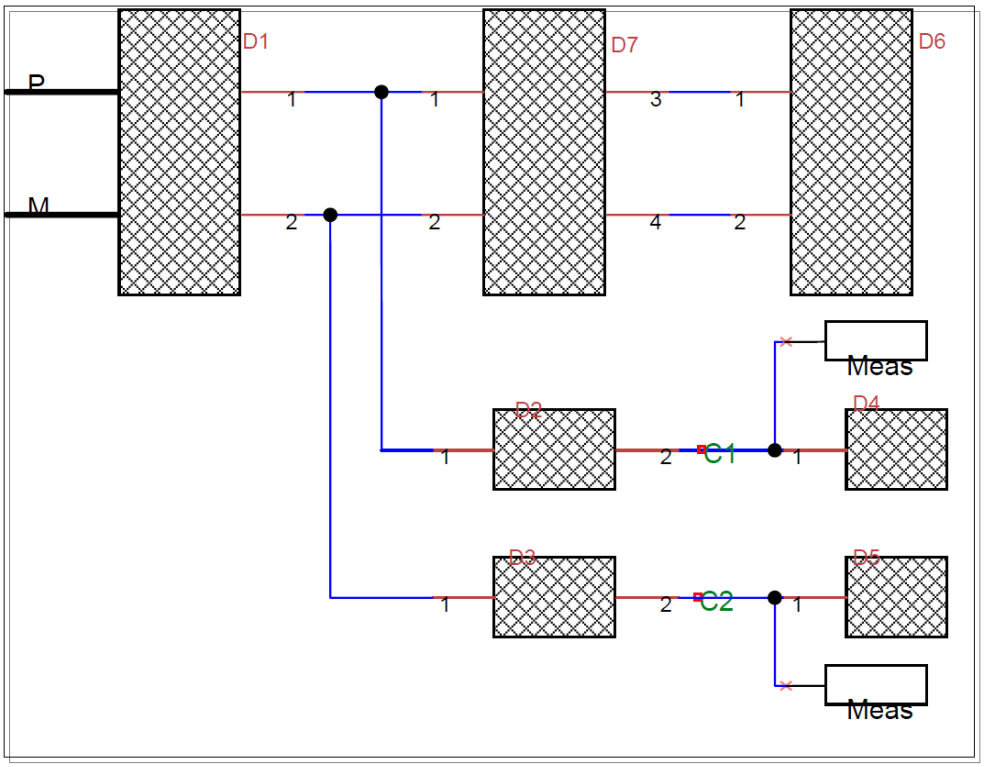
システム図には、測定構成に含まれるすべての回路要素、すなわち回路に信号を供給するすべてのデバイス、および測定に使用されるすべてのオシロスコープチャンネルを含める必要があります。

これは、システム内で信号がどのように生成され、オシロスコープの入力にどのように到達するかを記述するものであるため、絶対的な最小構成となります。つまり、以下のことを意味します：

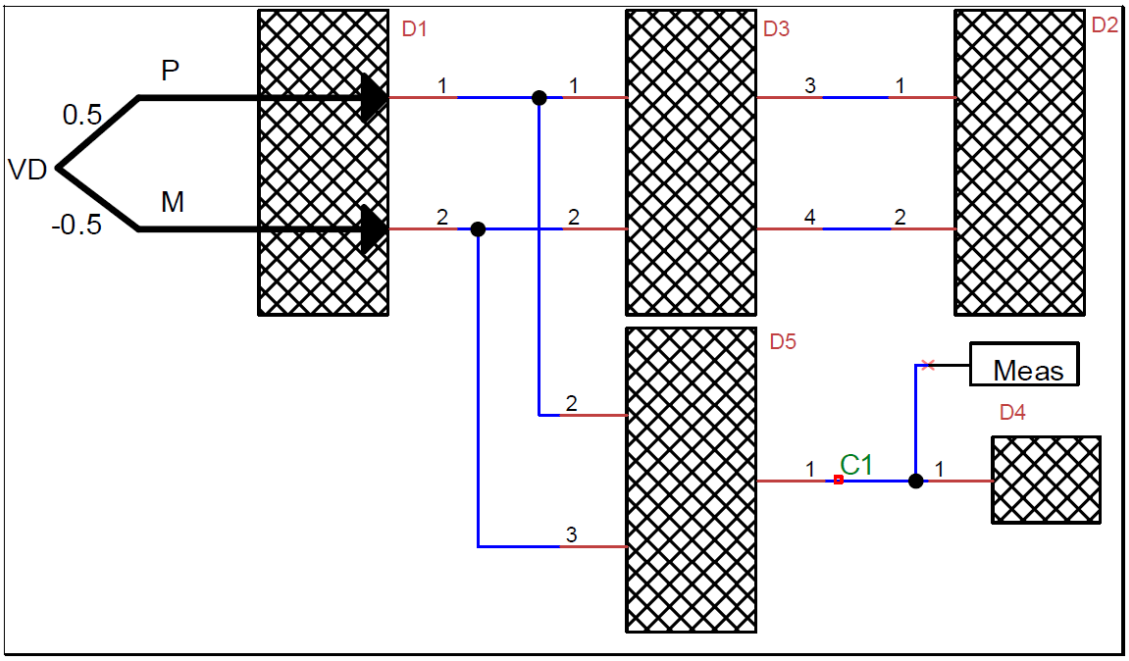
- 信号は、図に記述されていないソースから発生してはなりません。そうしないと、これらの信号は誤って解釈されてしまいます。その例としては、クロストークや電源ノイズなどが挙げられます。
- オシロスコープのチャンネル（オシロスコープの終端を記述する 1 ポートデバイス）を図に示す必要があります。さらに、オシロスコープ接続部のノードにはラベルを付け、システム記述ファイル内の .Meas ステートメントで明示する必要があります。
- 波形が到達し、インターフェースで反射する可能性があるすべての箇所を、図面に示す必要があります。

代表的な例をいくつか示します。これらの構成に関する注意事項は以下の通りです：

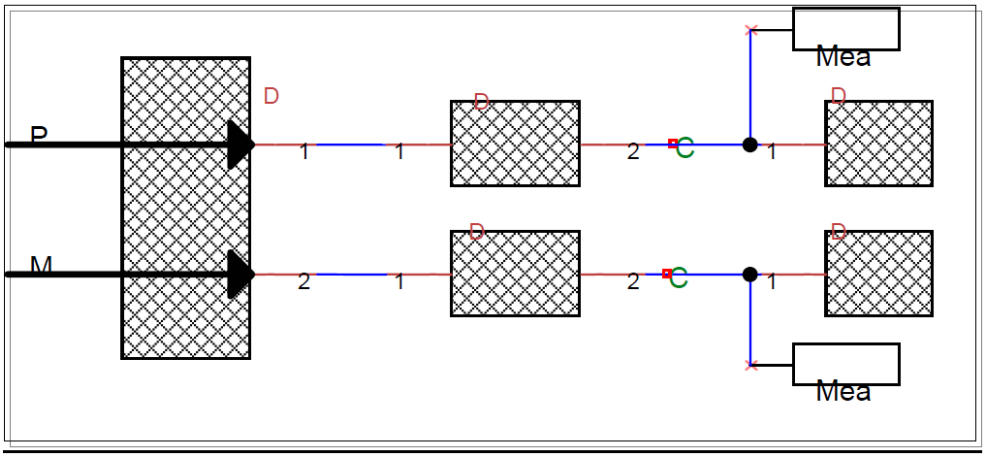
トランスミッタにおけるシングルエンド高インピーダンスプローブ接続



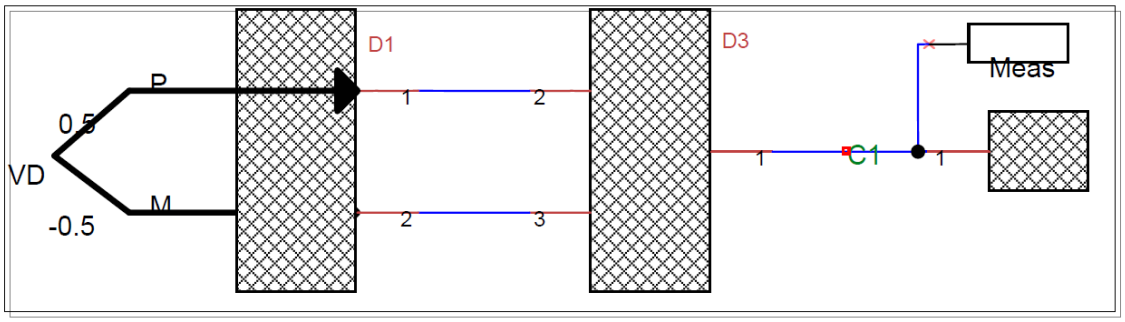
トランスミッタにおける差動高インピーダンスプローブの接続



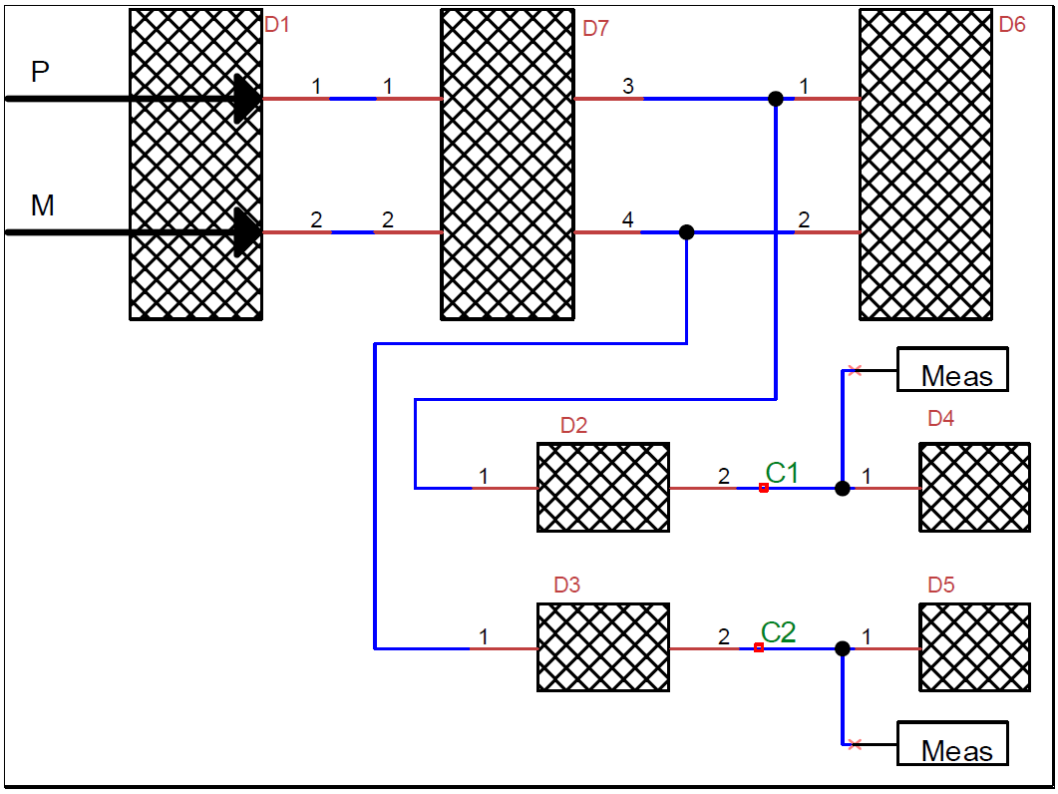
トランスミッタ側のシングルエンド 50 オームプローブ接続



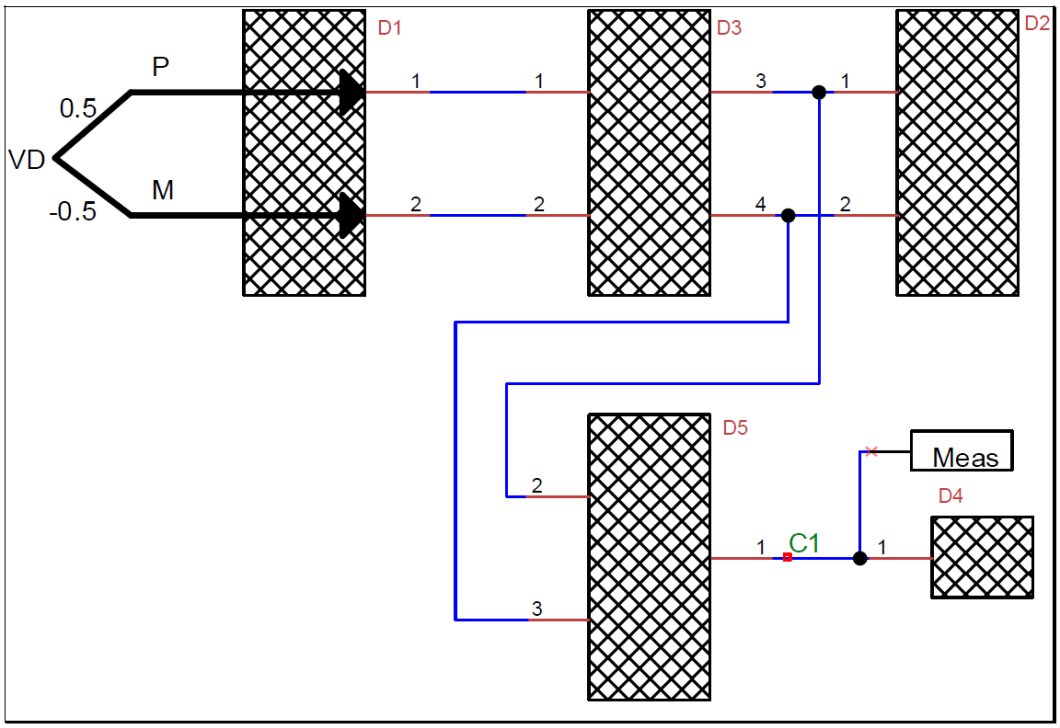
トランスミッタ側の 50 オーム差動プローブ接続



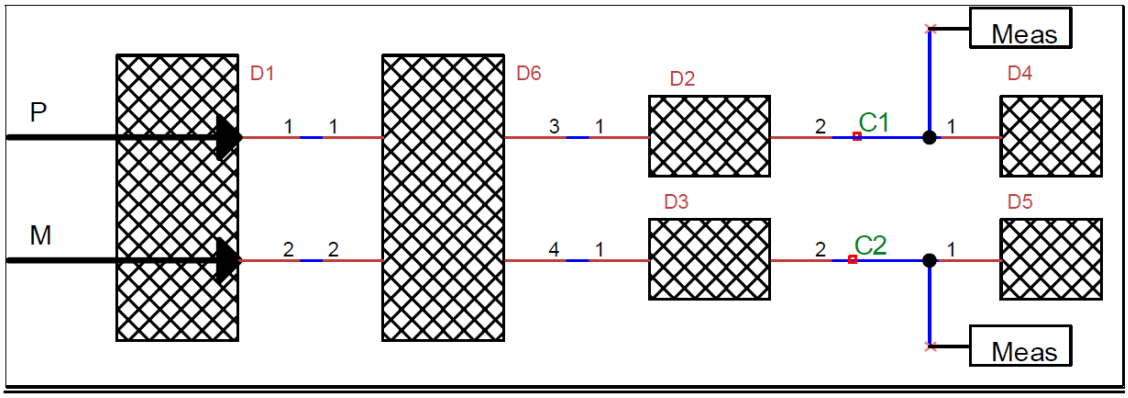
レシーバ側でのシングルエンド高インピーダンスプローブ接続



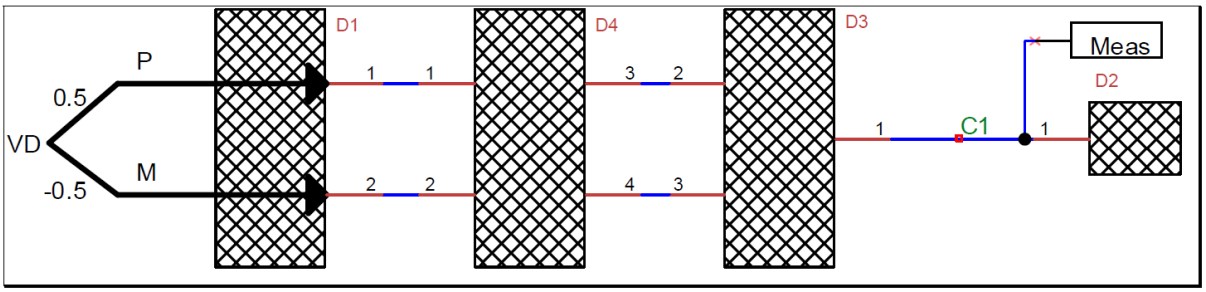
レシーバにおける差動高インピーダンスプローブの接続



レシーバ側のシングルエンド 50 オームプローブ接続



レシーバ側の 50 オーム差動プローブ接続



注：

1. すべての図において、トランスミッタは左側にあり、バックプレーンは中央に 4 ポートデバイスとして表され、終端は右側にあります。
2. いずれの場合も、高インピーダンスプローブはライン上の「T 字分岐」として配置されています（つまり、ドットのあるノードに接続されています）。50 オームのプローブはラインを終端処理しています。
3. オシロスコープチャネルは、1 ポートの素子（終端）であり、オシロスコープのチャネルとプローブ素子の間に位置するノードにはラベルが付けられ、「測定値」として指定されています。このラベルが付けられたノードは、オシロスコープのチャネルポートで発生する電圧であり、オシロスコープが検知する電圧を表します。
4. 励起信号はトランスミッタの両方のデバイスポートから発せられ、それぞれ P および M（プラス信号およびマイナス信号を表す）と指定されます。
5. 差分測定の場合、励起は 2 つありますが、測定ノードは 1 つだけです。したがって、単一の自由度については VD として示され、完全な平衡が仮定されている場合、励起 P および M は VD に関連して示されます（すなわち、P は VD の 1/2、M は VD の -1/2 となります）。これは .Stimdef ステートメントで明示されます（例：.Stimdef VD defines P M as 0.5 -0.5）。
6. これらの測定構成に示されている要素はすべて重要であり、無視することはできません。これらはすべて、測定における回路の一部です。

出力構成

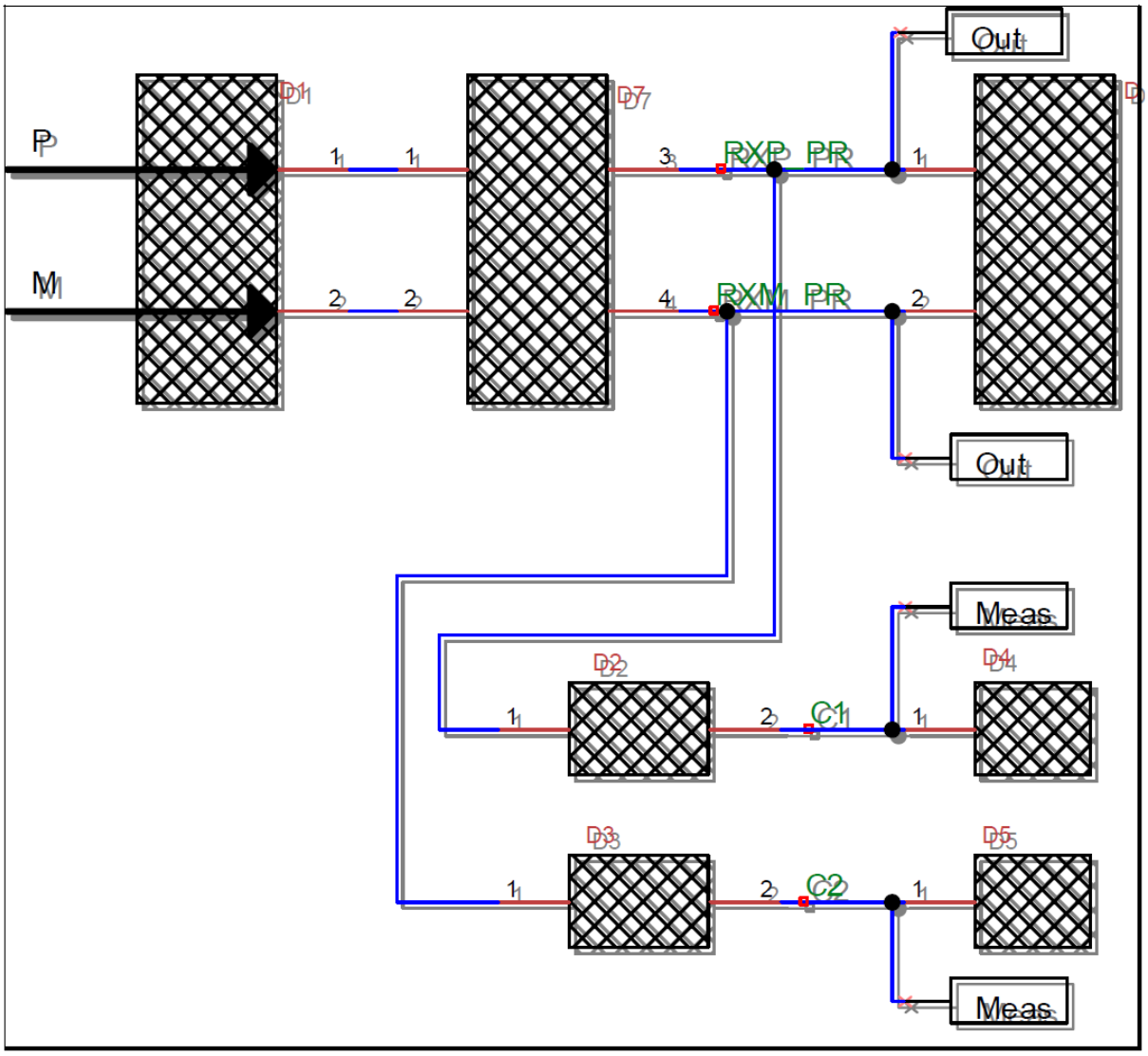
出力構成とは、システムの出力として定義されたノードを含む構成におけるトランスミッタを示した図です。測定構成内に出力構成とノードが見つかった場合、処理は単純明快です。この場合、測定構成内のノードに出力というラベルを付けます。この状況では、対象となる波形は、測定ハードウェアが接続された回路内のものだと考えられます。つまり、バーチャルプローブ コンポーネントは、プローブが接続されている間に回路内に存在する波形を表示します。

出力構成が測定構成と異なるケースは数多くあります。一般的な状況としては、次のようなものがあります。

1. 出力構成が測定構成と同じで、プローブが接続されていないという一点だけが異なる場合です。これは一般的に、高インピーダンスのプローブの場合にのみ当てはまります。この場合、仮想プローブによって生成される波形は、プローブが存在しなかった場合に現れるはずの波形です。これは「プローブ・デエンベディング」と呼ばれます。
2. 出力構成が、完全に異なるハードウェアで構成されている場合です。これには、トランスミッタの入力リターンロス特性の違い、バックプレーンの違い、終端条件の違いなどが含まれます。これは「エミュレーション」と呼ばれます。エミュレーション構成では、ソフトウェアの特性を変更するだけで、異なるハードウェア特性を試すことができます。多くの場合、この構成における特性はシステムシミュレータを通じて生成されます。このようにして、ハードウェアを物理的に製造することなく、その動作を試すことができます。
3. 出力として宣言されたノードは、プローブが接続されている場所とは異なる位置にあります。このようにして、仮想プローブコンポーネントは、プローブ接続点以外の位置で波形を生成します。場合によっては、これらのプローブ接続点にアクセスできないこともあります。

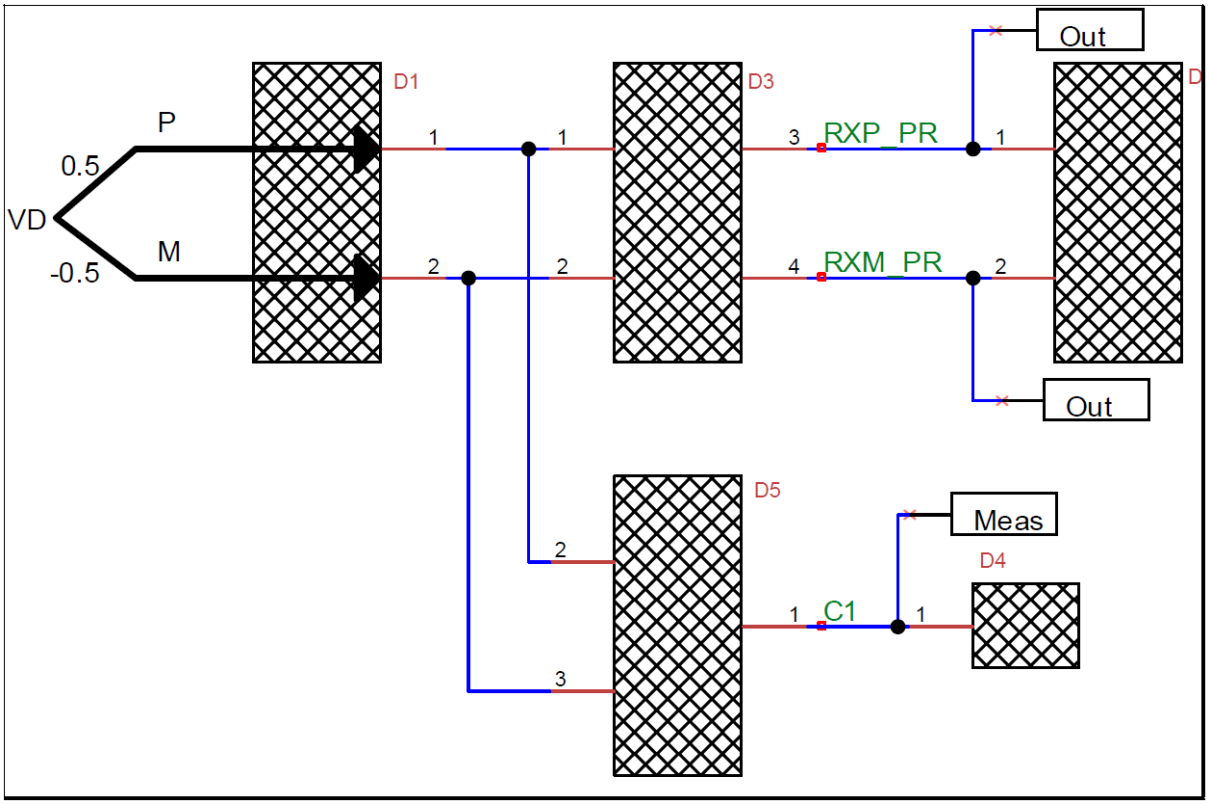
出力構成と測定構成の一致

測定構成内に目的のシステム波形が見つかった場合は、測定構成図内の出力ノードにラベルを付けるだけで済みます。例は以下の通りです：



これは上記の測定構成です（「レシーバにおけるシングルエンド高インピーダンスプローブ接続」を参照）。出力ノード RXP_PR および RXM_PR が追加されています。この状況では、仮想プローブコンポーネントにより、プローブが接続されている間、レシーバで波形が生成されます。この状況は、測定に影響を与える非理想的なプローブ特性がある場合でも、レシーバで正確な波形を確認するのに役立ちます。

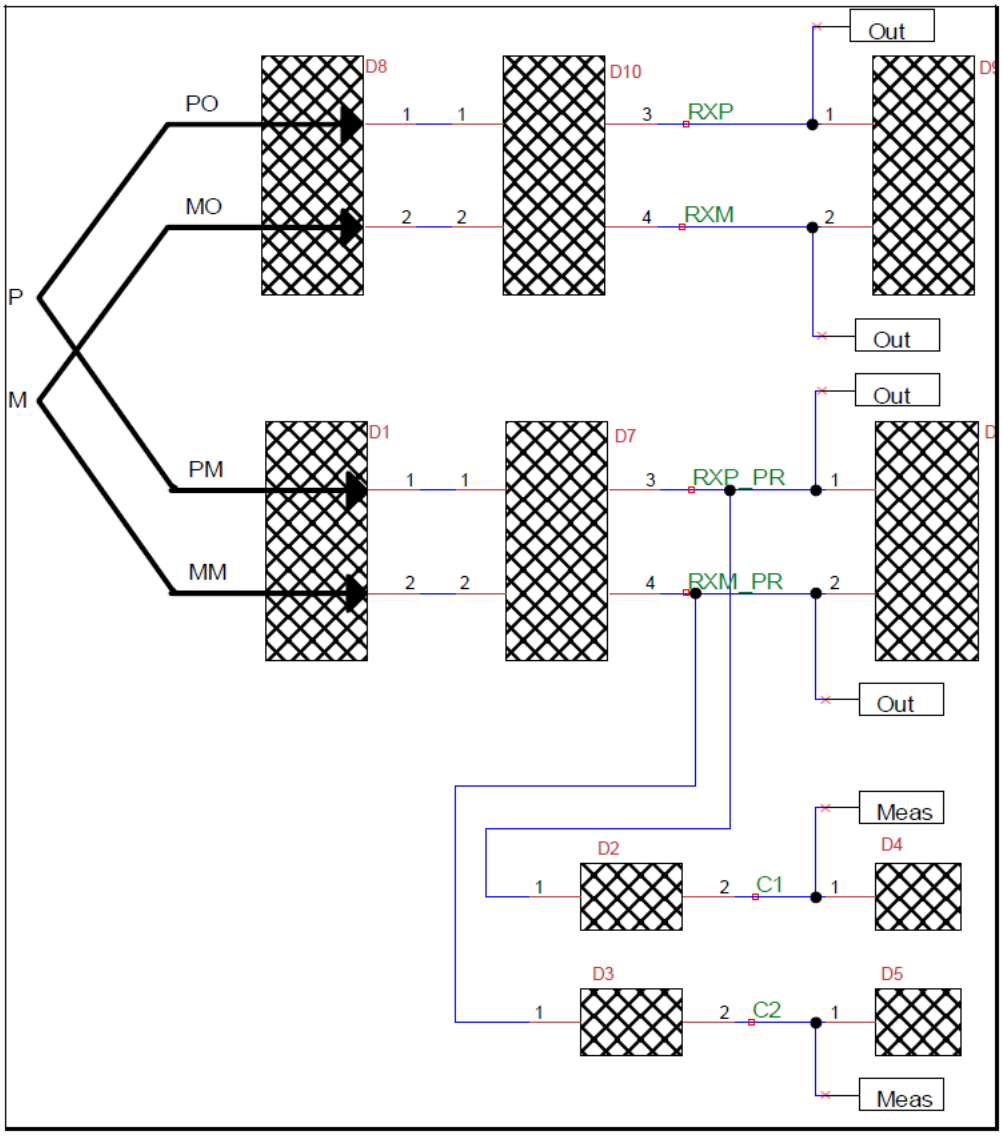
別の例としては、次のようなものがあります。



これは上記の測定構成です（「トランスミッタにおける差動高インピーダンスプローブ接続」を参照）。出力ノード RXP_PR および RXM_PR が追加されています。この状況では、仮想プローブ成分により、トランスミッタにプローブが接続されている間、レシーバ側にも波形が生じます。この状況は、測定に影響を与える非理想的なプローブ特性がある場合でも、レシーバ側の正確な波形を確認するのに役立ちます。

プローブ未接続時の出力構成

前のセクションでは、測定構成と一致する出力構成を示しました。測定構成の出力ノードには、プローブの影響を受けた波形が表示されます。多くの場合、プローブが接続されていない状態での波形を確認したいことがあります。これは、測定構成を、プローブを接続しない状態で単に描き直すことで実現できます。唯一の難点は、励起を、システム内の測定ノードの数と一致する自由度の数に何らかの形で結びつける必要があるという点です。以下に、その実現方法を示す例を示します：

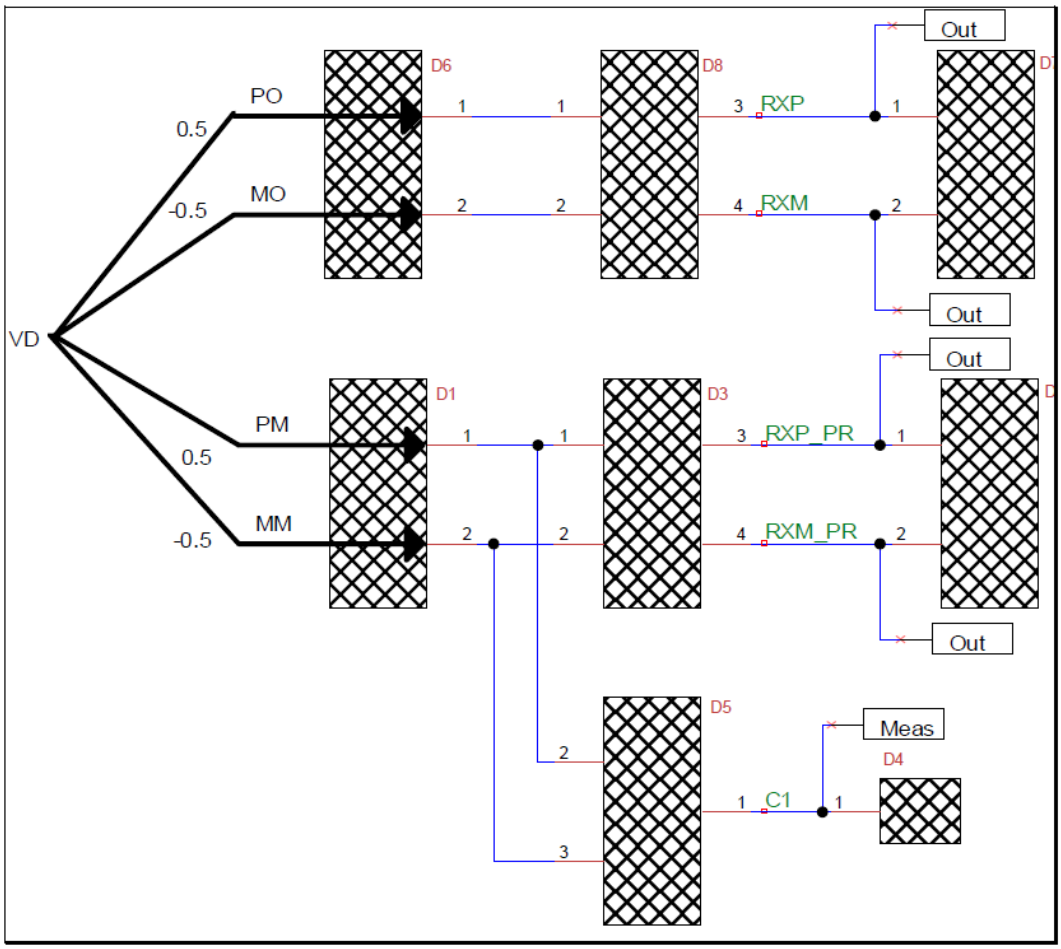


これは、前のセクションの構成であり、回路に出力構成が追加されています。この出力構成は、プローブが接続されていない状態の測定構成の複製です。RXP_PR および RXM_PR に加えて、新しい出力ノード RXP および RXM が追加されました。RXP_PR はプローブが接続された状態でレシーバに現れる波形を表示するのに対し、RXP はプローブが接続されていない場合に現れるはずの波形を表示します。これは主に以下の 2 つの理由から有用です。

1. RXP と RXM は、プローブによる干渉がない場合のシステム性能を示します。つまり、特定の制約条件下では、プローブ D2 および D3 の特性や、オシロスコープのチャネルリターンロス D4 および D5 にかかわらず、RXP と RXM は同じ波形を表示します。これは、測定からプローブの影響を除去するコンプライアンス試験において有用です。
2. RXP および RXM を RXP_PR および RXM_MR と比較することで、測定中にプローブが受信側の波形にどのような影響を与えているかを確認できます。これは、たとえプローブの特性に関する正確な測定データが提供されていたとしても、通常、プローブが測定に与える影響を理解することは非常に難しいため、重要なことです。バーチャルプロービングを使用することで、これを確認することが可能になります。

なお、励起には PO、MO、PM、MM という名前が付けられており、測定ノードの数に合わせて、2 つの自由度 P および M に直接関連付けられています。システム記述ファイルでは、励起を定義した後、.Stimdef ステートメントを使用してこれらを関連付けます（例：.Stimdef P M は、PO、MO、PM、MM を １００１１００１ として定義します）。これは、PO が 1×P+0×M、MO が 0×P+1×M、PM が 1×P+0×M、MM が 0×P+1×M となるためです。

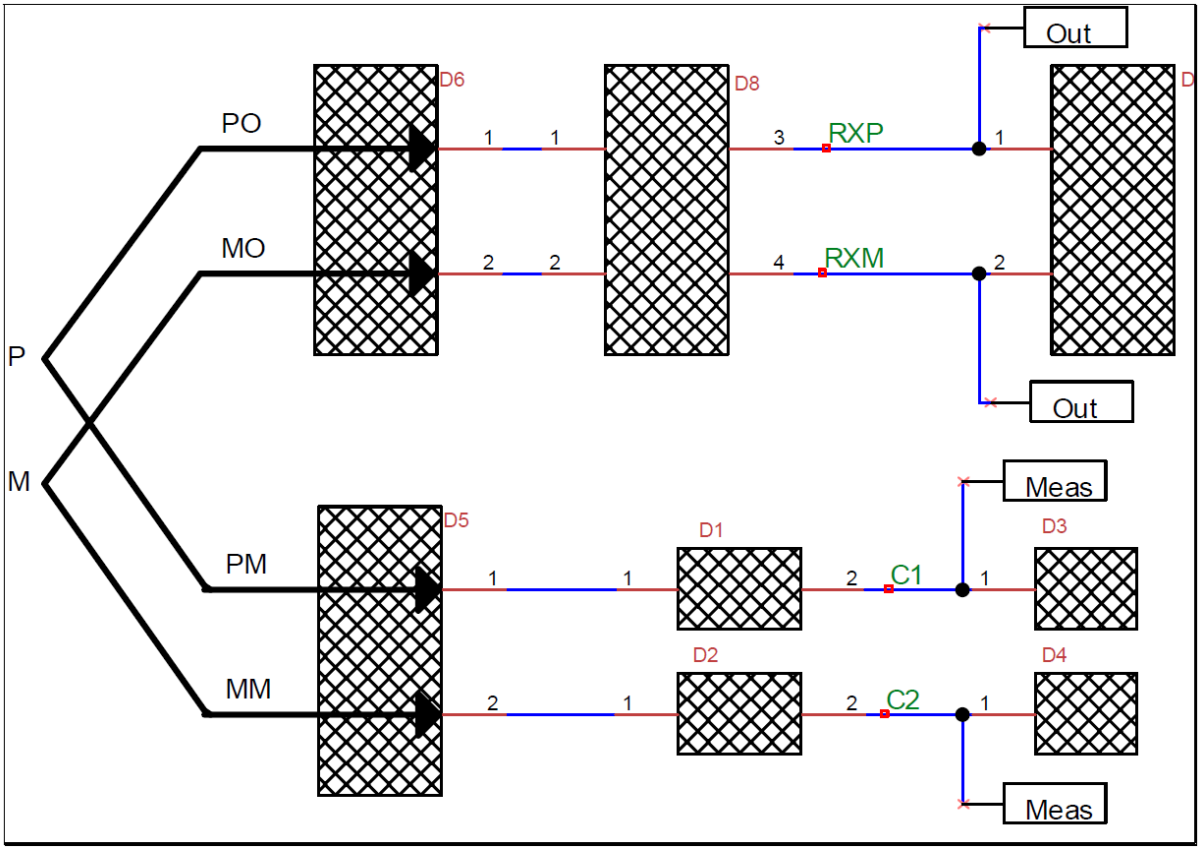
もう 1 つの例:



この構成は、トランスミッタ側で差動プローブを使用している点を除き、前の例と同じです。RXP_PR および RXM_PR に加えて、新しい出力ノード RXP と RXM が追加されています。RXP_PR はプローブが接続された状態でレシーバに現れる波形を示すのに対し、RXP はプローブが接続されていない場合に現れるはずの波形を示します。この場合、励起 PO、MO、PM、MM は単一の自由度 VD に関連しており、PO と PM は VD の 1/2、MO と MM は VD の -1/2 であることに注意してください。したがって、.Stimdef は次のようになります:

.Stimdef VD defines PO MO PM MM as 0.5 -0.5 0.5 -0.5

以下は、ハードウェアが全く異なる場合の出力構成の例です:



この構成では、測定構成が以前に示したもの(「トランスミッタにおけるシングルエンド 50 オームプローブ接続」を参照)であることがわかります。この構成では、トランスミッタがオシロスコープのチャンネル入力に直接接続されています。出力構成は同じトランスミッタで構成されていますが、バックプレーンチャンネルと終端が含まれています。出力ノード RXP および RXM は、このトランスミッタが所定のバックプレーンを所定の終端に駆動した場合に、レシーバの波形がどのように見えるかを示しています。このようにして、実際のトランスミッタ波形を用いて、完全なバックプレーンエミュレーションが実現されます。この構成は、さまざまなチャネルおよび終端条件下でトランスミッタをテストするのに役立ちます。

イコライザ エミュレーション

Eye Doctor のイコライザエミュレーション機能を使用すると、理想的なイコライザをエミュレートすることができます。

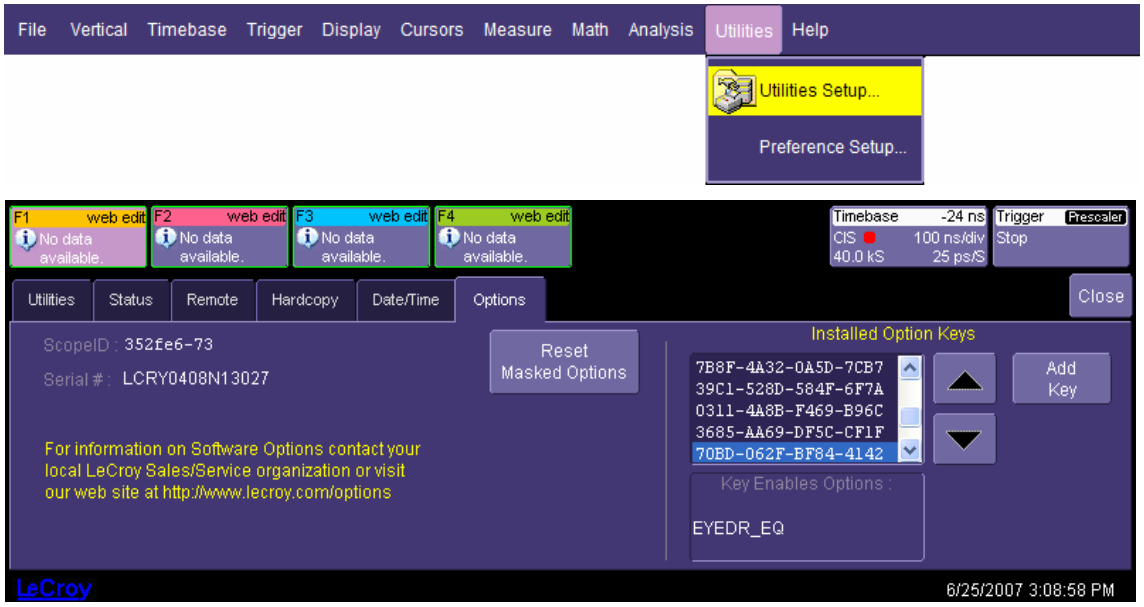
「理想的なイコライザのエミュレーション」機能により、以下のようなさまざまなイコライザ構成のエミュレーションが可能になります:

- 線形トランスバーサル、タップ付き遅延線、および線形フィードフォワード イコライザ(FFE)
- クロックリカバリ
- デシジョンフィードバック イコライザ(DFE)

イコライザエミュレーション機能を使用すると、復元されたデータ、クロック、およびイコライザ処理後の波形を確認できます。イコライザは、タップ遅延、タップ数、タップ係数によって指定することも、ブラインド適応を利用して設定を最適化するよう自動的に学習させることもできます。

Eye Doctor イコライゼーション オプション

Eye Doctor Equalization オプションは、お使いのオシロスコープにすでに搭載されている可能性があります。Support→About を選択して、ソフトウェアオプションの一覧が表示されるダイアログを開いてください。その一覧にオプションコード **EYEDR_EQ** が含まれている場合、お使いのオシロスコープにはすでに Eye Doctor Equalization オプションが搭載されています。含まれていない場合は、**Utilities** メニューからオプションのインストールと確認を行います。このオプションには、固有のオプションキーが必要です。



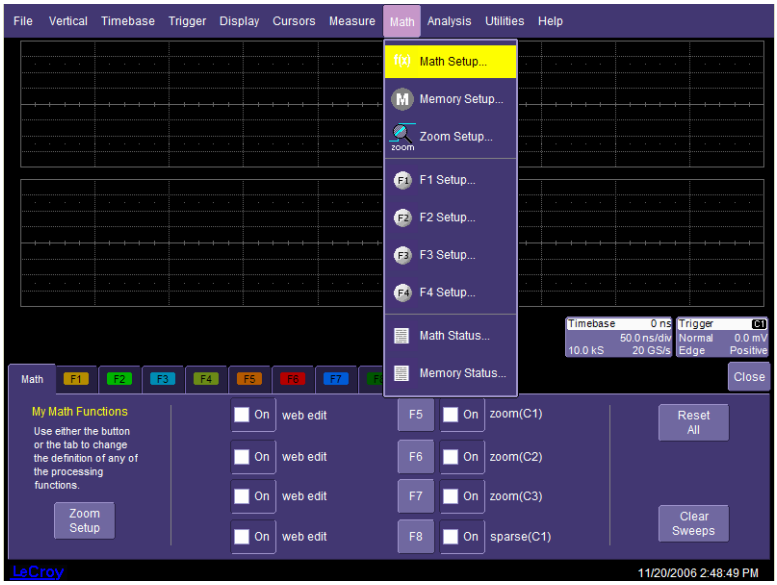
EYEDR_EQ オプションのオプションキーを取得するには、オシロスコープの ID およびシリアル番号を LeCroy に提供する必要があります。その後、LeCroy から 16 桁の 16 進数で構成されるオプションキーが提供されます。キーが正しくインストールされると、Installed Option Keys の下に **EYEDR_EQ** オプションが表示されます。

オプションキーに加え、お使いのオシロスコープのファームウェアバージョンは 5.0.4 以降である必要があります。現在のファームウェアバージョンは、Utilities ダイアログの Status タブで確認できます。

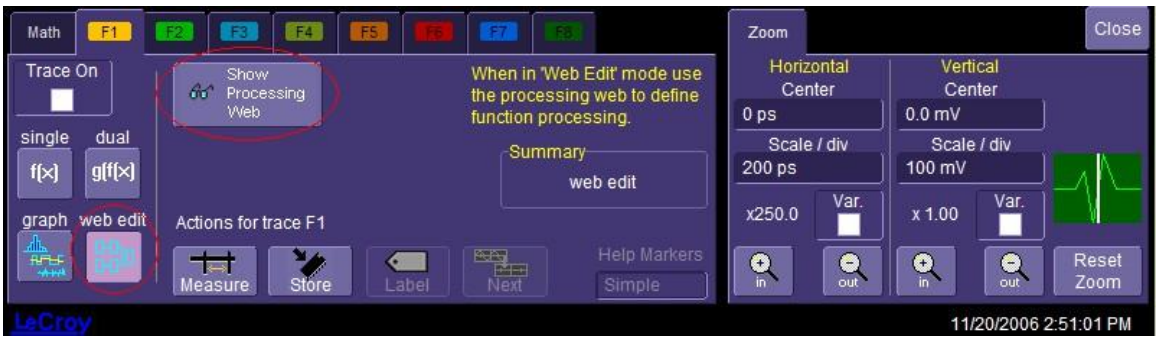
EYEDR_EW オプションを有効にすると、イコライザエミュレーション機能と、Processing Web エディタが利用可能になります。Processing Web エディタを使用すると、独自のグラフィカルユーザーインターフェースを通じて Processing の設定を編集できます。

Processing Web におけるイコライザエミュレーションの設定

Processing Web エディタには、Math Setup ダイアログからアクセスします。

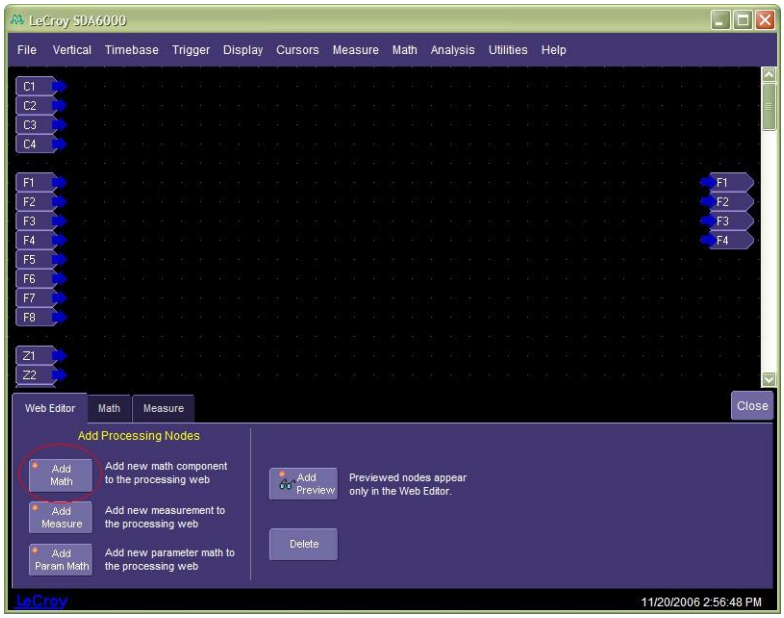


Math メインダイアログには、演算関数の定義方法が表示され、定義可能な各演算関数に対応するタブが用意されています。任意の演算関数のタブを選択すると、その演算関数の設定を行うことができます。

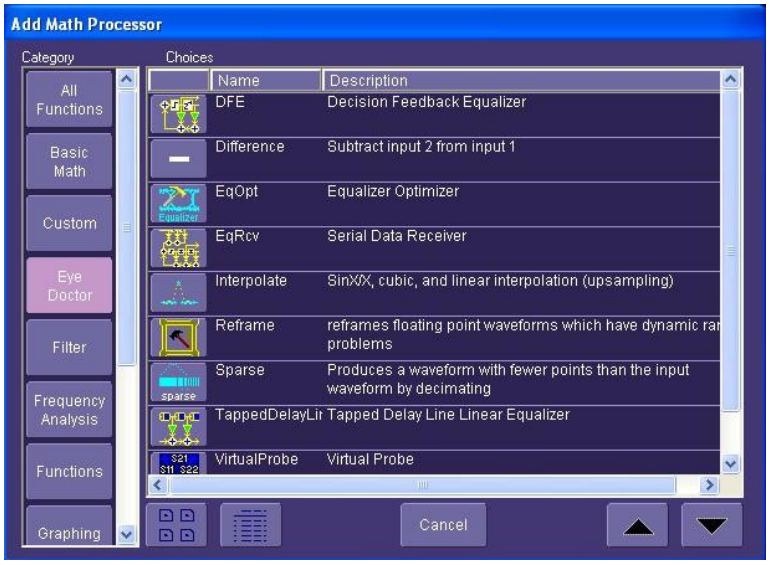


各数式設定ダイアログの左側には、演算関数の設定に関する 4 つのオプションがあります。これらは、Single、Dual、Graph、Web edit です。Web edit は、Web edit オプション（EYEDR によって有効化）が有効になっている場合にのみ利用可能です。Web edit が選択されている場合、Show Processing Web ボタンから Processing Web にアクセスできます。

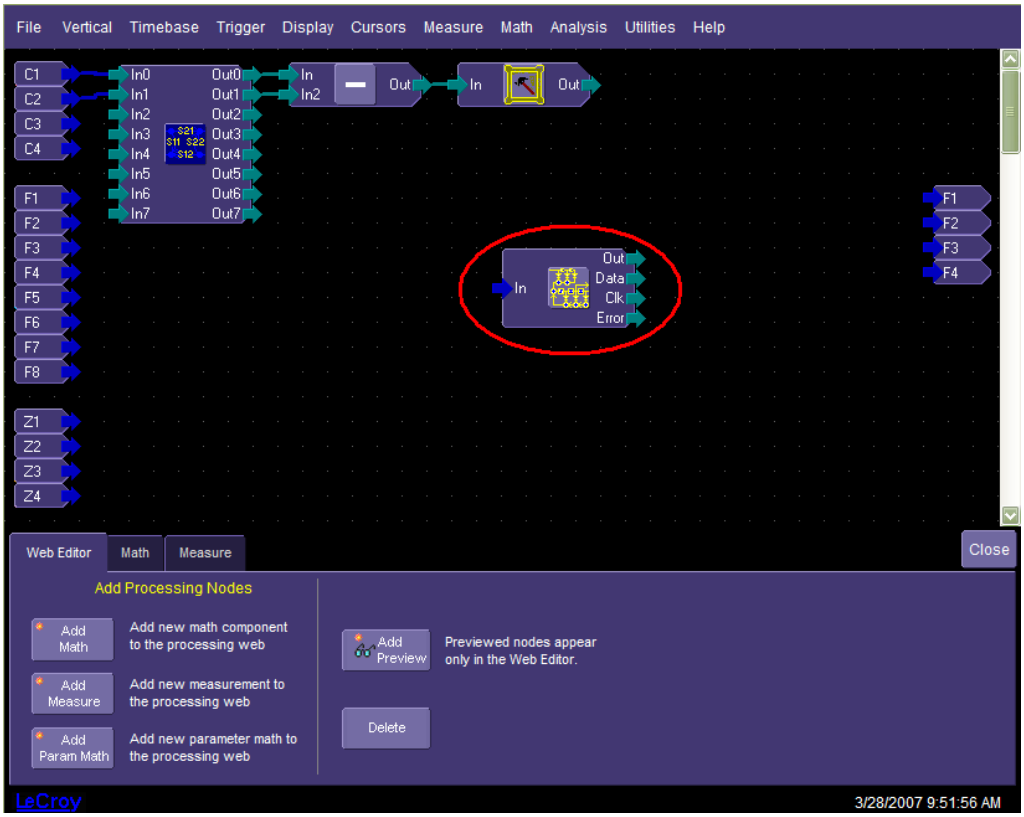
Processing Web が開くと、左側に利用可能なすべての波形ソースが、右側に利用可能なすべての波形出力が表示されます（この例では、F1、F2、F3、F4 が Processing Web edit 用波形として定義されているため、これらのピンが右側に表示されています）。



Eye Doctor コンポーネントにアクセスするには、**Add Math** ボタンを選択します。これにより、オシロスコープで使用可能なすべての演算関数が表示されたダイアログが表示されます。**Eye Doctor** カテゴリを選択すると、すべての Eye Doctor プロセッサが表示されます。



イコライザエミュレーションコンポーネントとして **EqRcv** を選択します。イコライザエミュレーションコンポーネントが Processing Web に表示されます。これは、1 つの入力と 4 つの出力を持つプロセッサです。右下隅には、内部設定ダイアログがあります。



ここでは、イコライザ処理が施されたレシーバコンポーネント（前述の仮想プローブコンポーネントとともに）が表示されています。目標は、仮想プローブコンポーネントによって提供されるレシーバでの波形をイコライズすることです。

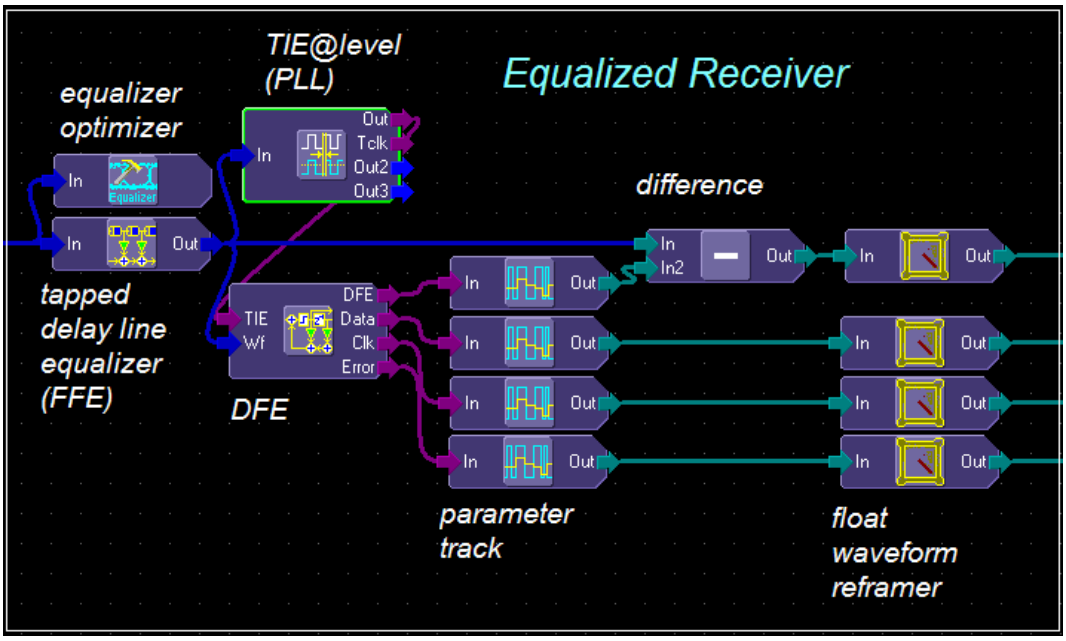
イコライザ受信素子への入力ピンには、アナログシリアルデータ波形が入力されます。これは、理想的には、システム内のレシーバ側で現れる波形です。

4 つの出力ピンは以下の通りです：

- Out – エミュレートされたレシーバのスライサーの入力に現れる、Equalization 処理済みの波形
- Data – エミュレートされたレシーバのスライサーの出力に現れる波形を表す理想的なデータ波形
- Clock – レシーバによって復元されたクロックを表す理想的なクロック波形
- Error – スライサーの入力における波形と、スライサーの出力におけるデータ波形との差（Out – Data）。この信号はノイズ信号を特定するために使用され、ノイズ信号が小さいほどイコライゼーションの精度が高いことを示します。この信号は、外部のユーザー定義によるイコライザ最適化アルゴリズムと組み合わせて使用することができます。

Equalization 済みレシーバは、実際には複数の内部プロセッサで構成されるコンポーネントです。これらのプロセッサはすべて、Processing Web 内で個別に利用可能であり、以下が含まれます:

- イコライザ最適化コンポーネント – 入力波形から最適な FFE および DFE 係数を決定します。
- タップ付き遅延線イコライザ – 線形、フィードフォワード イコライザを提供します。
- タイムインターバル エラー コンポーネント – タップ付き遅延線イコライザ コンポーネントの出力からクロックを抽出する PLL として機能します。
- デシジョンフィードバック イコライザ／レシーバコンポーネント – DFE を行い、復号データおよび抽出されたクロックを決定する役割を果たします。
- DFE 、データ、およびクロックのタイミングとレベルを波形に変換するトラックコンポーネント
- 差分コンポーネント – タップ付き遅延線イコライザの出力から Decision Feedback を差し引き、スライサーの入力でイコライズされた波形を形成します。
- 浮動小数点波形リフレーマー – 出力を適切にフレーミングします。



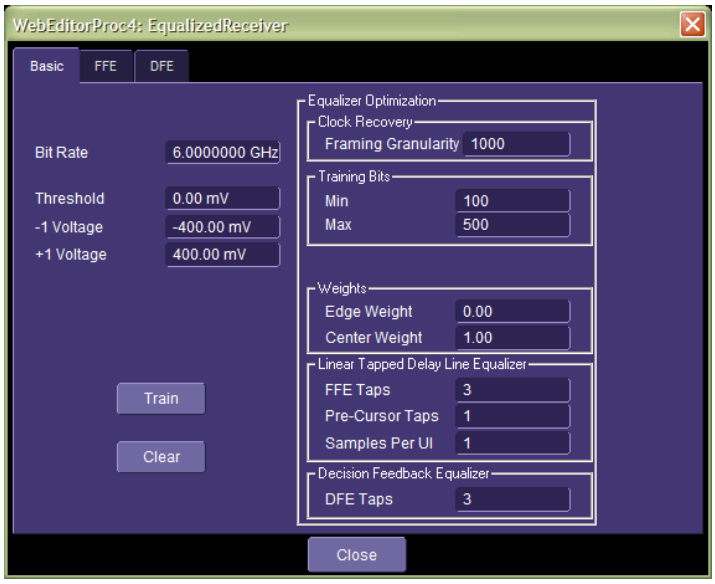
通常、Processing Web 上の個々の部品からイコライゼーション レシーバ コンポーネントを組み立てる必要はありません。

イコライゼーション付きレシーバ コンポーネントの話に戻りますが、このコンポーネントを Processing Web に配置したら、

Settings ボタンをクリックして、構成を設定するポップアップダイアログを開きます。このダイアログには 3 つのタブがあります。

- Basic – ビットレート、しきい値、理想レベルなど、シリアルデータ波形の基本的な特性を設定するダイアログです。また、自動イコライザ最適化のパラメータもここで設定します。
- FFE – 線形、フィードフォワード、タップ付き遅延線イコライザを設定するダイアログです。
- DFE – Decision Feedback Equalizer およびスライサーを設定するダイアログです。

Basic Equalization レシーバの設定



Basic Equalization レシーバの設定ダイアログでは、以下の操作が可能です:

- シリアルデータ信号の基本属性を設定する。
- イコライザの最適化を設定する。

信号の基本的な属性には、**Bit Rate**、ビットの復号を決定する **Threshold**、およびビットの理想的な**-1**(0)および**+1**(1)レベルが含まれます。これらの理想的なレベルは、スライサーによってビット判定が行われる時点で、シリアルデータ波形が保持すべき最適な電圧とみなされます。

イコライザの最適化設定は、イコライザがどのように自動的に最適化されるかを決定します。イコライザは内部クロック回復コンポーネントを利用せずに最適化されるため、フレーム単位の粒度を用いて、最適化のために回復されたクロックの適切な位相を決定する際に必要な階調数が決定されます。この数値は変更しないことをお勧めします。

Training Bits とは、イコライザの最適化に際して入力波形から使用する単位区間の数を指します。この数値は、最適化に使用する波形ポイント数が過大または過小にならないよう、一定の範囲に制限されています。

計算に必要な時間とメモリは、学習ビット数によって異なります:

- ビット数≈ 500:5 秒=メモリ使用量少
- ビット数≈ 500:5 秒=メモリ使用量中
- ビット数≈ 500:2 分=メモリ使用量大

注:トレーニングビット数が大きい場合(つまり、最適化に使用される波形データポイント数がそれに応じて多い場合)、オシロスコープの計算に異常に長い時間がかかり、利用可能なメモリをすべて使い果たしてしまう可能性があります。現在、この状況に対する保護機能は備わっていないため、トレーニングに使用するトレーニングビット数を慎重に検討する必要があります。判断に迷う場合は、**Min** および **Max** ポイントの値をデフォルトのままにしておいてください。

トレーニング中、入力波形ポイント数が Min Training Points 未満の場合、システムはトレーニングを行いません。それ以外の場合は、利用可能な波形ポイント数と指定された Max Training Points のうち、少ない方の値を使用します。

Weights(重み)は最適化の目的を決定します。エッジの配置を最適化したい場合もあれば、意思決定が行われる UI の中心部の電圧レベルを最適化したい場合もあります。また、重みを組み合わせて最適化したい場合もあります。**Edge Weight** と **Center Weight** を使用することで、この重みの配分を選択できます。

内部的には、重みは指定された値を 2 つの重みの合計で割った値に正規化されるため、それぞれに 1.0(最大重み)を割り当てることは、それぞれに 0.5(正規化された値)を割り当てることと同じになります。

Linear Tapped Delay Equalizer(線形タップ遅延線イコライザ)の最適化は、0～20 の範囲から目的のタップ数を選択することで指定します。0 を指定した場合は、FFE が使用されません。タップ数を 1 に指定した場合は、FFE がゲイン／減衰のみを提供し、これが最小設定となります。それ以外の場合、プレカーサータップは、指定されたタップ内でのカーソルの位置を決定します。例えば、4 つのタップが指定され、1 つのプレカーサータップが指定された場合、結果として得られるフィルタは 4 つの係数を持ち、係数 1（2 番目の係数）がカーソルを表します。

Decision Feedback Equalizer（DFE）の最適化は、タップ数を 0 から 20 の間で選択することで指定されます。0 を指定すると、DFE は使用されません。最初のタップ（タップ 0）は、スライサーの出力から直接取得されます。それ以降のタップは、スライサーの出力を遅延させたものから取得されます。

イコライザの最適化設定はいずれも波形の処理には影響を与えず（フィルタの最適化にのみ影響する）、したがって、フィルタの学習が開始されるまでは何の効果も発揮しないことに注意してください。

入力波形に対しイコライザを学習させるには、前述のすべての設定を行い、ダイアログ内の **Train** ボタンをクリックします。

イコライザをクリアして、実際のイコライゼーションを行わないようにするには、ダイアログの **Clear** ボタンを押します。

注:以下の条件が満たされない限り、イコライザのトレーニングは実行されません。

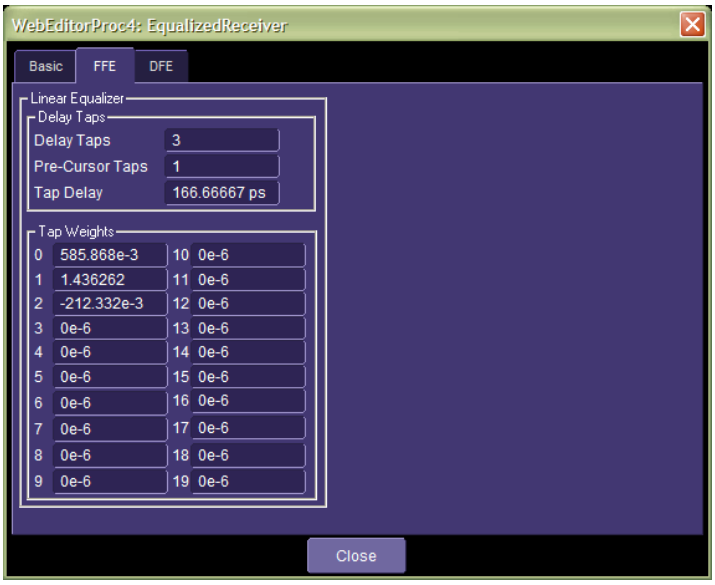
- そのコンポーネントの入力ピンに波形が存在する場合
- コンポーネントの出力ピンの負荷がアクティブである

これが満たされていることを確認するには、イコライザのトレーニングを行う前に、イコライザ処理済みのレシーバからの出力波形を確認してください。

イコライザの最適化の結果、他の設定ダイアログにおける FFE および DFE の構成が更新されます。具体的には:

- FFE の遅延タップ数は、指定されたタップ数に設定されます。
- FFE タップ遅延は、ビットレートおよび UI あたりのサンプル数に応じて設定されます。
- FFE タップの重みは最適に設定されています。
- DFE 遅延タップは、指定されたタップ数に応じて設定されます。
- DFE タップ重みは最適に設定されています。

フィードフォワード イコライザ(FFE) の設定



FFE 設定では、線形タップ付き遅延線イコライザの設定を行います。通常、これらの値はイコライザの最適化の結果として設定されます。このダイアログで結果を表示したり、手動で設定したり、独自の最適化アルゴリズムを使用して設定したりすることができます。

このダイアログページの値は、波形の処理方法を直接決定するものであり、これらの値を変更すると、表示されている波形に即座に反映されます。

FFE の構成には以下が含まれます:

- FFE のディレイタップ数
- プリカーサータップの数
- タップ遅延(通常は単位区間の 1 または 1/2)
- タップ重み – フィルタ係数。

フィルタの出力は常にカーソルタップ(プリカーサータップ数に対応するタップ番号を持つタップ)から取得されます。

Decision Feedback Equalizer（DFE）の構成



DFE 設定では、Decision Feedback Equalizer の設定を行います。通常、これらの値は Equalizer の最適化の結果として設定されます。このダイアログで結果を確認したり、手動で設定したり、独自の最適化アルゴリズムを使用して設定したりすることができます。

このダイアログページの値は、波形の処理方法を直接決定するものであり、これらの値を変更すると、表示されている波形に即座に反映されます。

DFE の設定には以下が含まれます：

- DFE 遅延タップの数
- タップ重み – フィルタ係数

動作中、DFE は以下の処理を行います：

1. クロックの立ち下がりエッジで、スライサーの入力における波形がスライスされます。つまり、その波形が閾値と比較されます。閾値を上回っている場合、電圧は前述の理想的な+1 電圧に設定されます。閾値を下回っている場合は、前述の理想的な-1 電圧に設定されます。
2. スライスされた電圧は、一連の遅延タップ（UI ごとに 1 タップ）に供給されます。
3. クロックの立ち上がりエッジで、遅延タップ内およびその先頭にあるスライスされた電圧値に対して、所定の長さと係数を持つ DFE フィルタが適用されます。
4. DFE フィルタの出力は、差動素子に供給されます。この差動素子は、DFE フィルタの出力を DFE 素子の入力から差し引き、その結果をスライサーに供給します。
5. 手順 1~4 を繰り返します。

注：DFE フィルタは、1 や 0 ではなく、電圧に適用されるフィルタです。

注：通常、実際の DFE は、決定が行われた直後（つまり、波形がスライスされた直後）にフィードバックを適用し、その後の決定に影響を与える時間を確保しようとします。この DFE は、フィードバック信号を瞬時に適用できるという点で理想的です。ただし、ジッタ測定の理由から、結果のアイ図を現実的に分析できるよう、フィードバックはクロックの立ち上がりエッジまで保留されます。

DFE およびスライサーの構成には、さらに 2 つの要素があります：

- クロック調整
- Erasure DFE

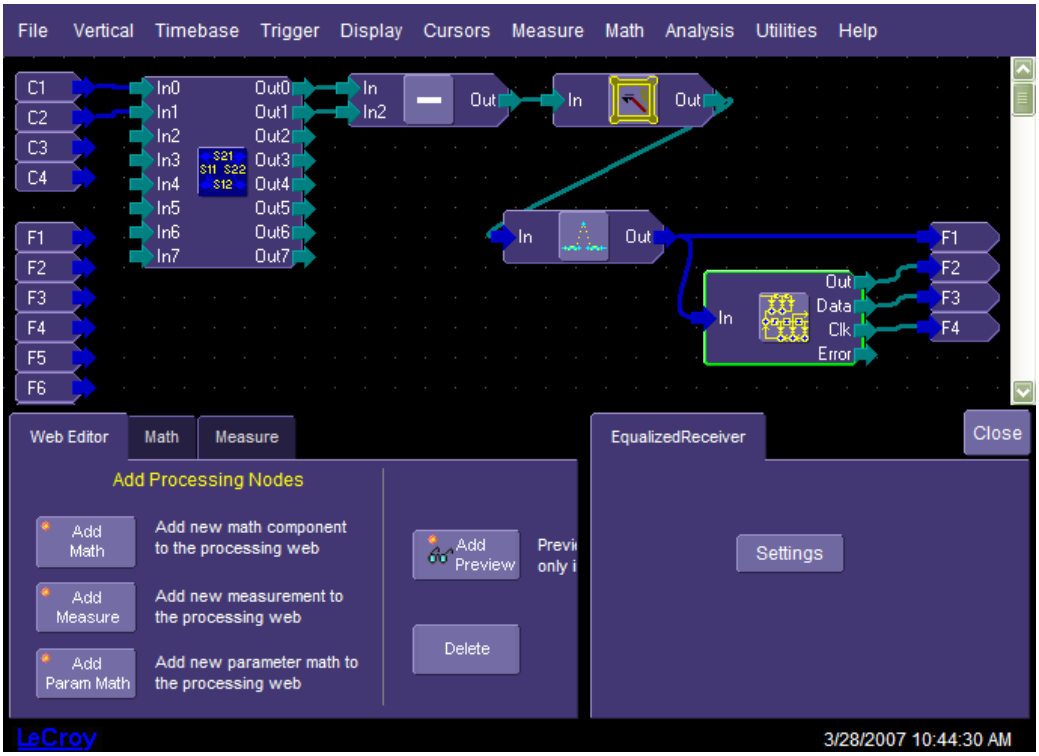
スライサーに入力されるデータに対して、クロックの最適な設定がしばしば存在するため、クロックの配置は調整可能です。クロック調整の仕組みは、正の数値が入力されると、クロックがデータに対して後方に移動し、データがより早くサンプリングされるようになっています。

注:イコライザの最適化では、PLL と同じクロック抽出は行われません。そのため、イコライザのトレーニング後に、同じ最適な結果を得るためにクロックを調整する必要がある場合があります。

DFE はバーストエラー（一度発生すると長期間続くエラー）を起こしやすい傾向があります。DFE がバーストエラーを起こしやすいのは、ビットの正確な復号を支援する DFE の有効性が、ビットを正確に復号する能力に依存しているという、循環的な状況によるものです。適用される DFE の強さによっては、1 ビットのエラーが長期間にわたるエラーの連鎖を引き起こす可能性があります。この状況を改善するために、Erasure DFE が用いられます。Erasure DFE は、しきい値の周囲に帯域を効果的に設定します。スライシングの時点で信号がこの帯域内に収まる場合、これはビット復号における不確実性を示しており、レシーバはビットを復号する義務はあるものの、復号が十分に確実ではなかったため、このビットを Decision Feedback に適用しないことを決定できます。動作において、Erasure DFE が利用され、信号が指定された閾値周辺の電圧デルタの範囲内にある場合、理想的な+1 および-1 の値の平均となる電圧値が DFE の遅延タップに印加され、その結果、そのビットは Decision Feedback に影響を与えないことになります。

注:Erasure DFE を使用する場合、デコードされたデータ波形において、消去は、各ビットの理想的な電圧レベルの中間点に位置するデータビットとして現れます。

Equalization 済み レシーバの接続



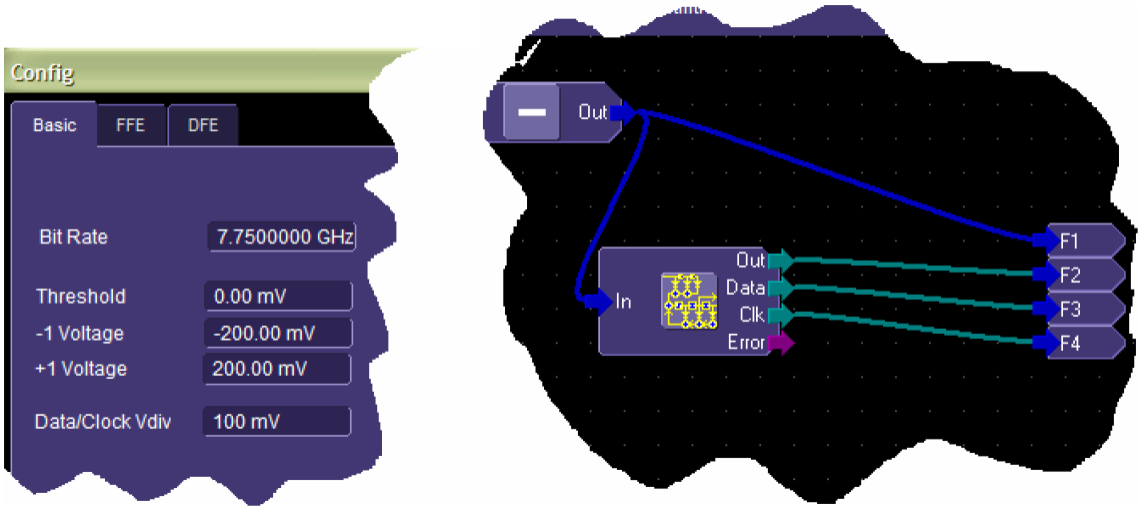
上図は、Equalization レシーバコンポーネントの接続図です。この接続により、Equalization されていない波形が F1 に、Equalization された波形が F2 に到達し、データ波形とクロック波形がそれぞれ F3 および F4 に表示されます。波形をより見やすくするため、アップサンプリング回路を回路内に配置しました。

Equalization レシーバの例

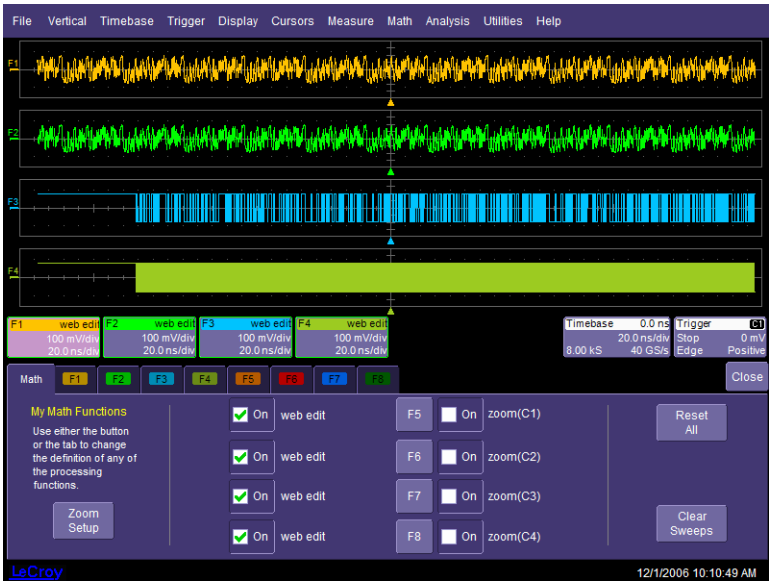
Equalization レシーバに関する残りの情報は、例を用いて説明します。

レシーバでプローブした波形が 7.75 Gb/s NRZ の 127 ビットパターンである状況を想定します。波形の理想的なレベルは 200 mV および -200 mV で、しきい値は 0 であると仮定します。F1 にイコライゼーション処理前の波形、F2 にイコライゼーション処理後の波形、F3 および F4 にデータ波形とクロック波形を表示したいとします。

これにより、イコライゼーション機能付きレシーバの基本設定に示されているような設定になります:



演算波形は、演算設定メニューを使用して有効にします:



上段のレシーバには、イコライゼーション処理前の波形が表示され、その後に同じイコライゼーション処理前の波形（まだイコライザを最適化していないため）、続いてデータ波形、そしてクロック波形が表示されます。

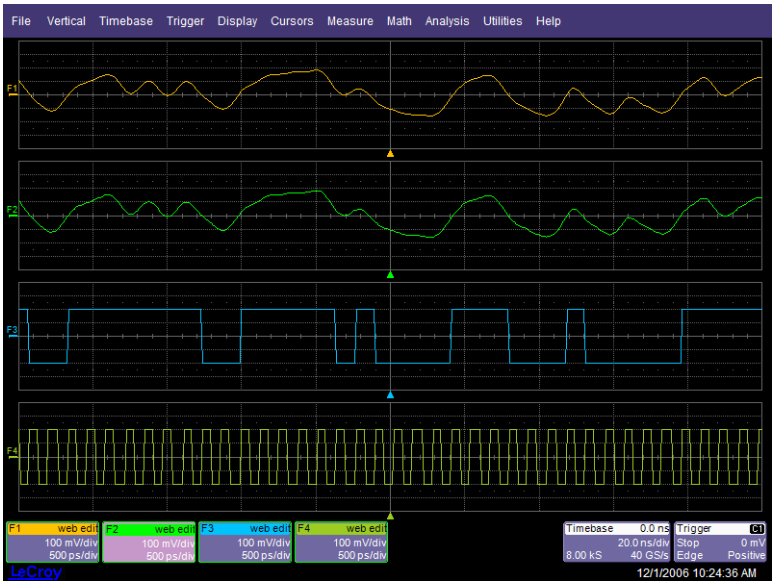
特に注意すべき点は、レシーバでの波形生成におけるフィルタリングの都合上、Equalization されていない波形の左右に隙間があることです。イコライザから出力される波形にも、同様の隙間が生じます。さらに、データ波形とクロック波形には起動時の問題が見られます。ここでは、クロックが安定するまでに約 40 ns かかっています。通常、この起動時の状況は表示されたくないものです。オシロスコープは波形捕捉のたびに連続した波形を受信するわけではないため、起動処理を行う必要があります。この遅延が表示されないようにするには、波形を十分に拡大表示してください。

注: イコライザの起動時の不具合を防ぐため、取得システムのタイムベース設定を調整しないでください。これでは目的を達成できません。代わりにズーム機能を使用してください。

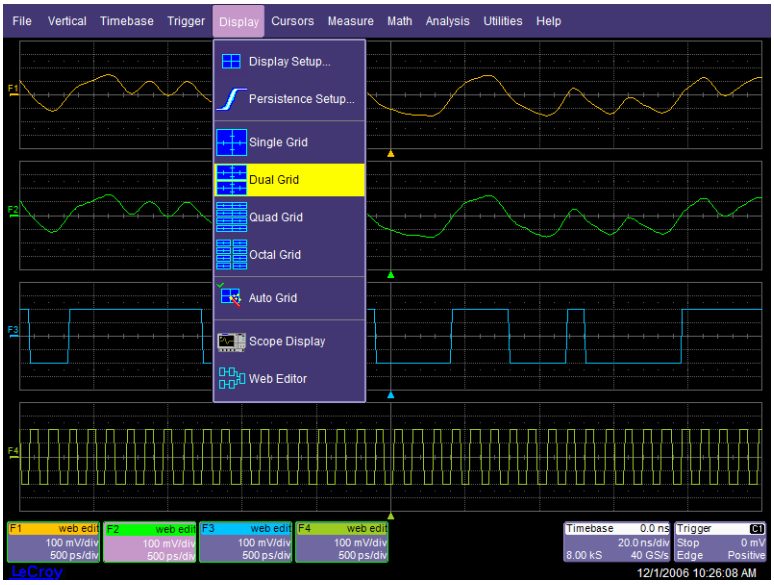
これらの波形はすべて一緒にズームする必要があるため、MultiZoom 機能を使用してトレースのズームを固定してください。

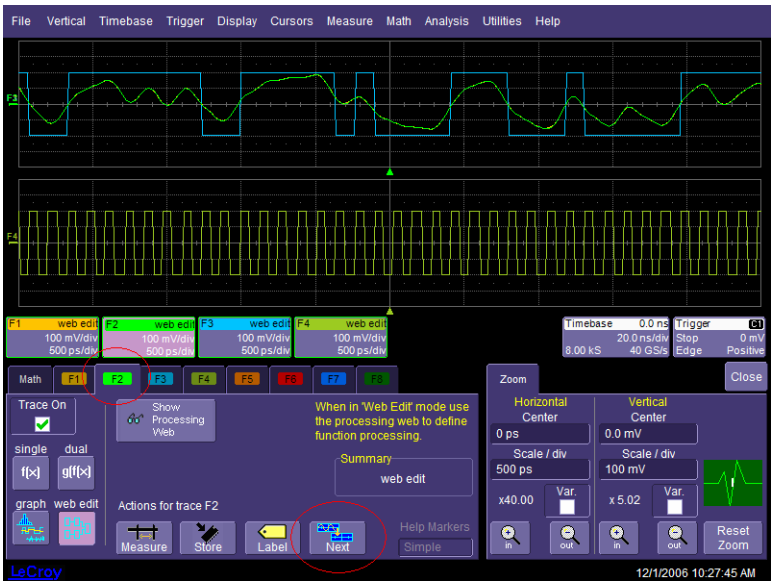


上の図では、Math ドロップダウンメニューから **Zoom Setup...** が選択されています。次に、**MultiZoom** タブを選択し、Multi-Zoom]を **On** にします。マルチズームに表示したいすべての演算トレースを選択します。右側の **Auto-Scroll** コントロールを使用して、ズームされたトレースをスクロールします。



この時点で、イコライゼーション処理前の波形、イコライゼーション処理後の波形、およびデータ波形を互いに重ねて表示すると便利です。これを行うには、メニューバーから **Display** を選択し、**Dual grid** を選択します。演算トレースを選択し、**Next Grid** をタップします：



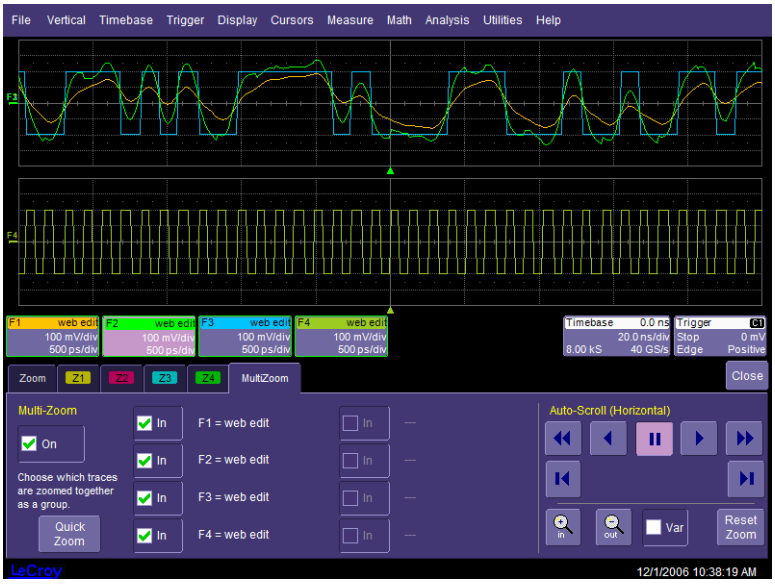


この時点で、すべての波形が上部のグリッドに表示されますが、クロック波形のみは下部に表示されます。デコード済みデータ波形を見ると、多くのビットエラーが発生していることが確認でき、波形の歪みによりデータが明らかに正しくデコードされていないことがわかります。

エッジとセンターを均等に重み付けするようにイコライザの最適化を設定し、3 つの FFE タップ（センタータップ）を UI あたり 2 サンプル、および 2 つの DFE タップを設定します：



Train ボタンを押すと、これらの設定に基づいてイコライザが最適化され、表示されている波形に変化が見られます：



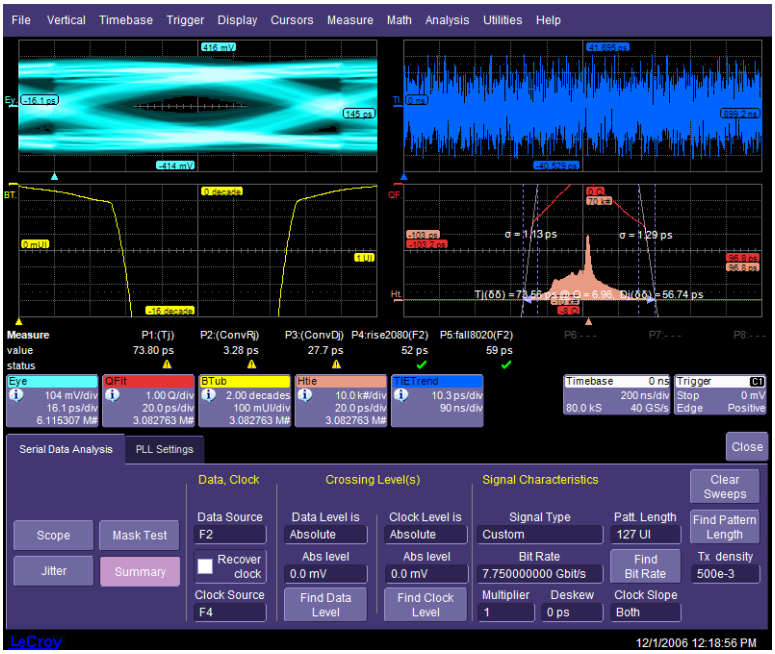
イコライゼーションを施した波形ははるかに良好な性能を示しており、デコードされたビットも正しいことが確認できます。

FFE および DFE タブには、最適化の結果として得られたイコライザの設定が表示されます：



この段階で、イコライザの最適化設定を自由に試してみたり、イコライザのタップや係数の値を自分で指定したりすることができます。

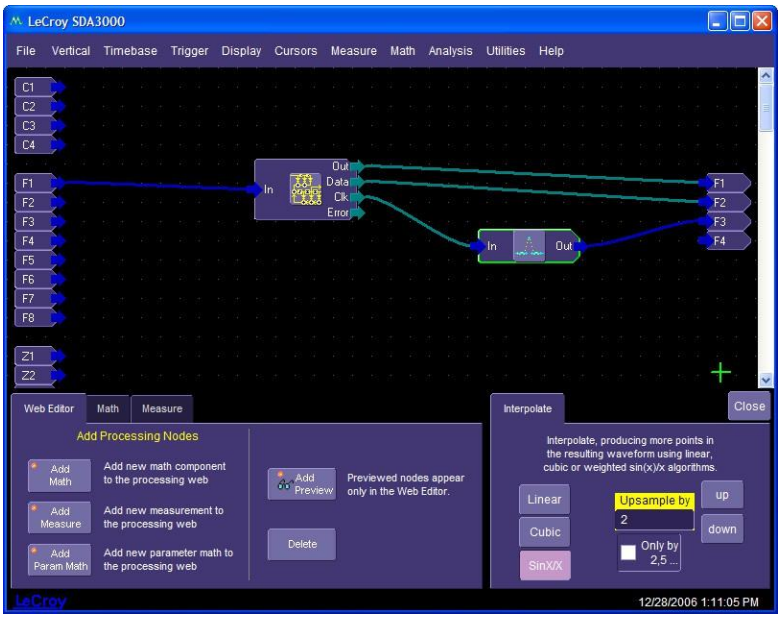
さらに重要なのは、通常の SDA 機能を使ってジッタを分析できることです。これを行うには、波形をズームアウトして（画面上に少なくとも数個のパターンが表示されるように）、表示範囲を広げる必要があります。



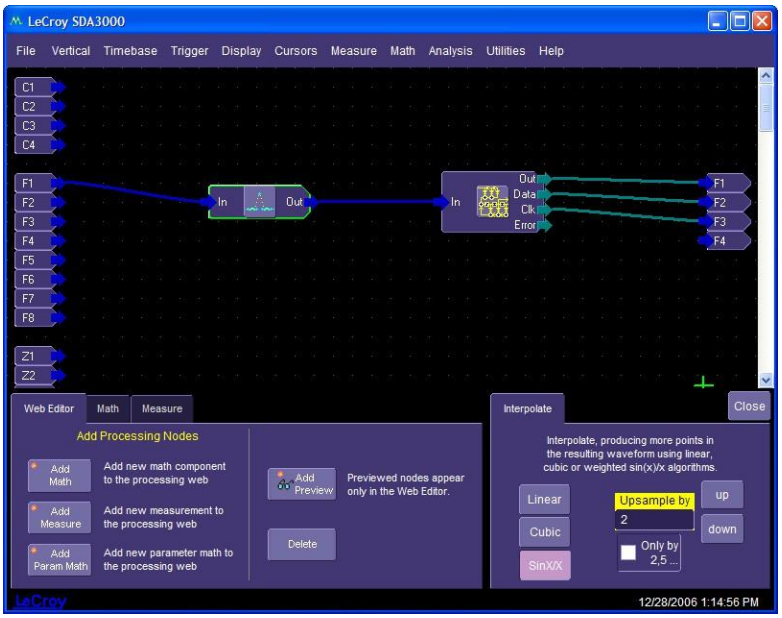
Eye Doctor を SDA 解析ソフトウェアと併用する場合、特に DFE を使用する場合は、**Recover clock** チェックボックスのチェックを外し、Equalization レシーバコンポーネントの出力から提供されるクロックを使用するようにしてください。



Equalization レシーバコンポーネントから復元されたクロックを使用する場合、アップサンプリング機能を使用してクロックの立ち上がり/立ち下がりエッジの時間分解能を向上させると効果的です。これを行うには、**Add Math** を選択し、ポップアップメニューから **Interpolate** を選択します。



あるいは、イコライザーコンポーネントの前にアップサンプリング回路を配置することも有効である（この場合、通常、クロック波形のアップサンプリングは必要ない）。



§ § §