Eeschema

リファレンス・マニュアル

著作権

貢献者

Jean-Pierre Charras, Fabrizio Tappero.

日本語翻訳： Silvermoon，Zenyouji，Yoneken，Millo，Nenokuni（順不同、kicad.jp ）

フィードバック

このドキュメントに関するコメントや提案を KiCad メーリングリストに送ってください：

https://launchpad.net/~kicad-developers
http://kicad.jp/ (日本語ユーザコミュニティ)

謝辞

なし

発行日とソフトウエアのバージョン

英語版： 2011 年 11 月 28 日に LibreOffice 3.3.2.により発行されました。
日本語版： 2014 年 10 月 26 日に LibreOffice 3.5.4.により発行されました。

Mac ユーザへの注記

Apple OS X のオペレーティングシステム用の KiCad のサポートは実験的なものです。

目次

1 - Eeschema 入門

2 - Eeschema コマンド全般

目次

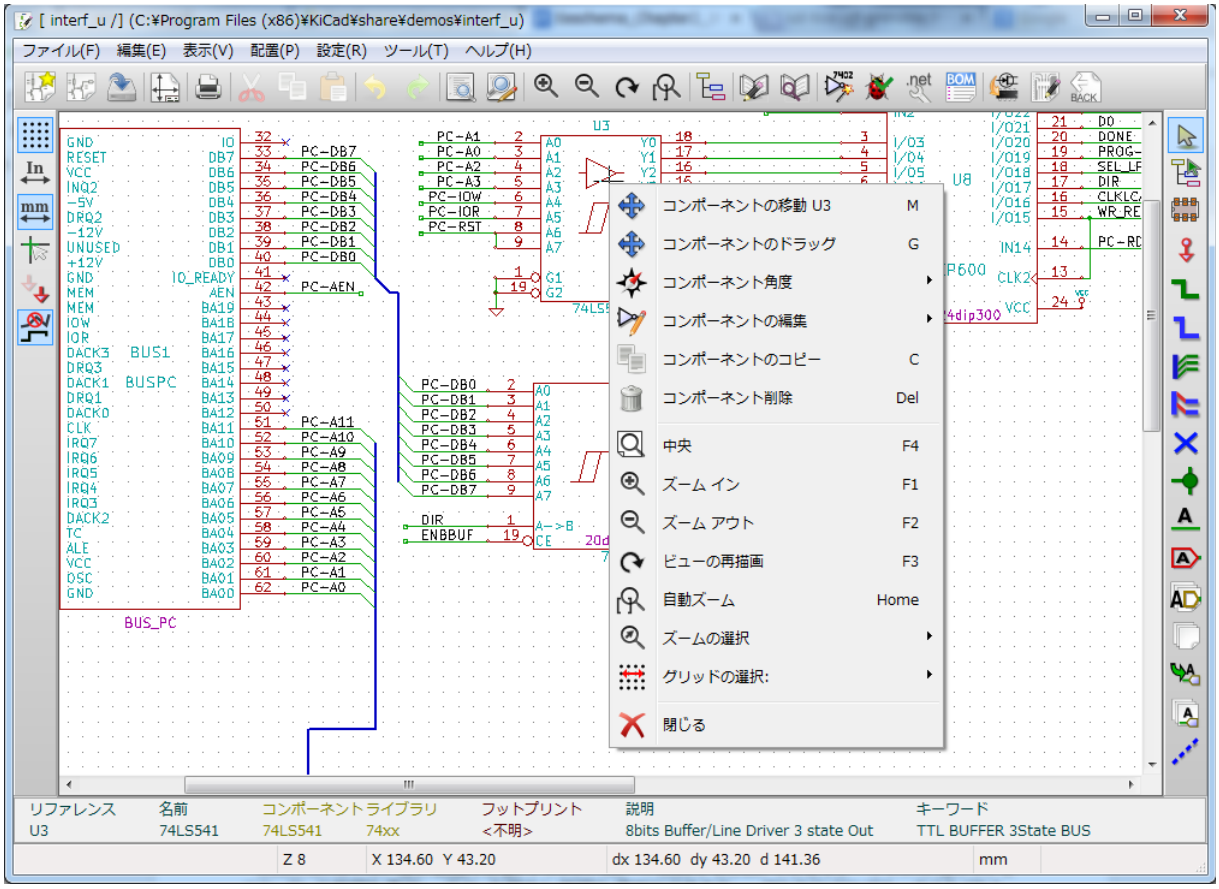
2.1 - Eeschema コマンドへのアクセス

2.2 - コマンド

以下に示す様々な方法でコマンドを起動できます：

- 画面上部のメニューバーをクリックする.
- 画面上部のアイコンをクリックする （一般コマンド）.
- 画面右側のアイコンをクリックする（特別なコマンド、または”ツール”）.
- 画面右側のアイコンをクリックする（特別なコマンド、または”ツール”）.
- マウスボタンをクリックする (重要な補助コマンド). 特に右クリックでは，カーソル下の要素に対応したコンテキストメニューを開きます（ズーム，グリッドと要素の編集）.
- キーボードのファンクションキー （F1，F2，F3，F4，インサートとスペースキー）. 特に “Esc” キーは，多くの進行中のコマンドの中断ができます. ”Insert” キーは，最後に作成された要素の複写ができます.

このように，コマンドへのアクセス可能な方法はたくさんあります.



2.3 - マウスコマンド

基本コマンド

左ボタン

- シングルクリック ：カーソル下のコンポーネントあるいはテキストの特性を表示する.
- ダブルクリック: コンポーネントあるいはテキストを編集 （要素が編集可能な場合）.

右ボタン

- ポップアップメニューを開く.

ブロックの操作

Eeschema では，選択範囲を移動、ドラッグ、コピー、削除することができます． マウスの左ボタンをドラッグして範囲を選択し，ボタンを離すと 範囲の選択ができます． ”Shift”と”Ctrl”キーのうちどちらか 1 つ，もしくは”Shift”と”Ctrl”キーの両方を押しながら選択することで，選択した範囲をコピーするか，それともドラッグするか，あるいは削除するのかが変わります.

コマンドの表:

左マウスボタン	選択範囲を移動.
Shift + 左マウスボタン	選択範囲をコピー.
Ctrl + left mouse button	選択範囲をドラッグ.
Control + Shift + left mouse button	選択範囲を削除.

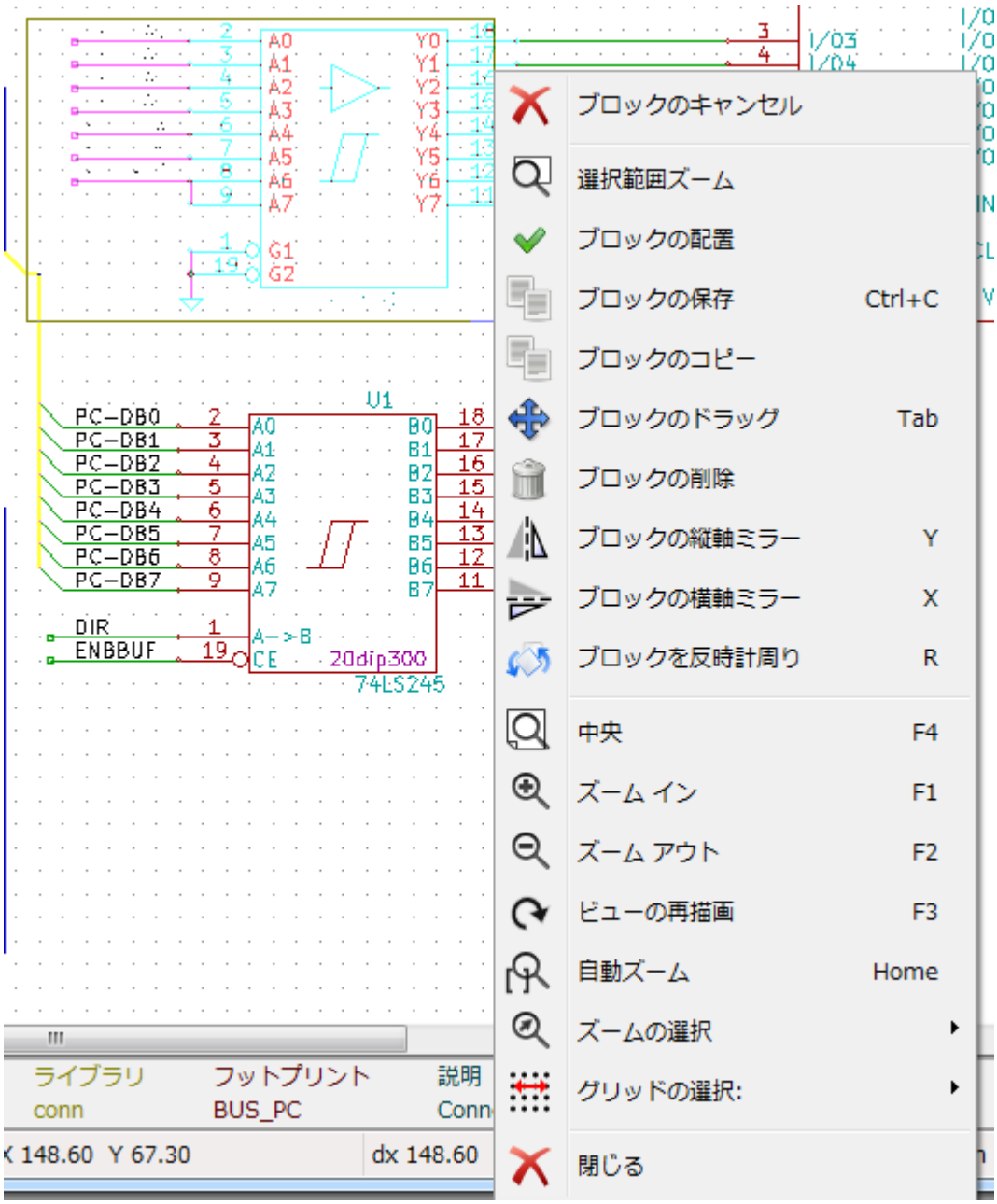
コマンドは，ボタンを離すと実行されます.

選択中には，次のことができます：

- もう一度クリックして要素を置き直す.

- 右ボタンをクリックして取消す.

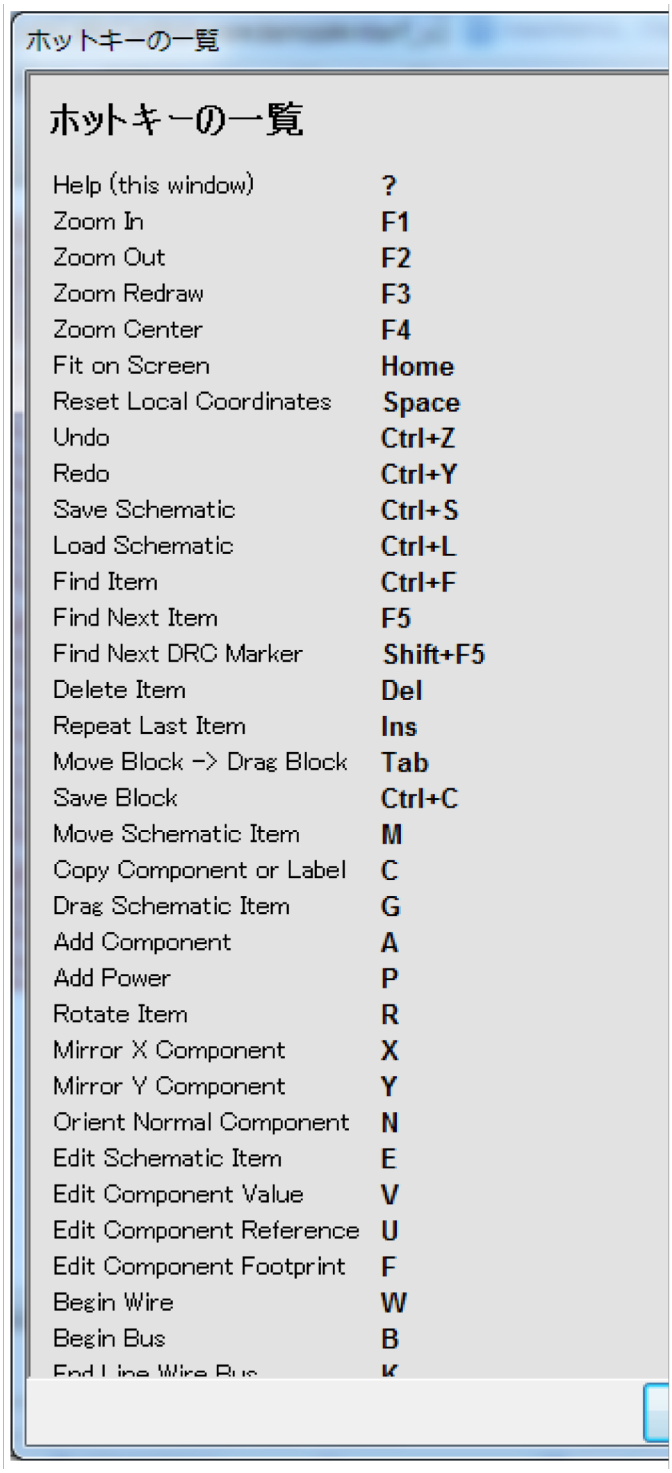
ブロック移動コマンドが実行されているとき，マウスの右ボタンからポップアップメニューを開くと，他のブロックコマンドも選択できます:



2.4 - ホットキー

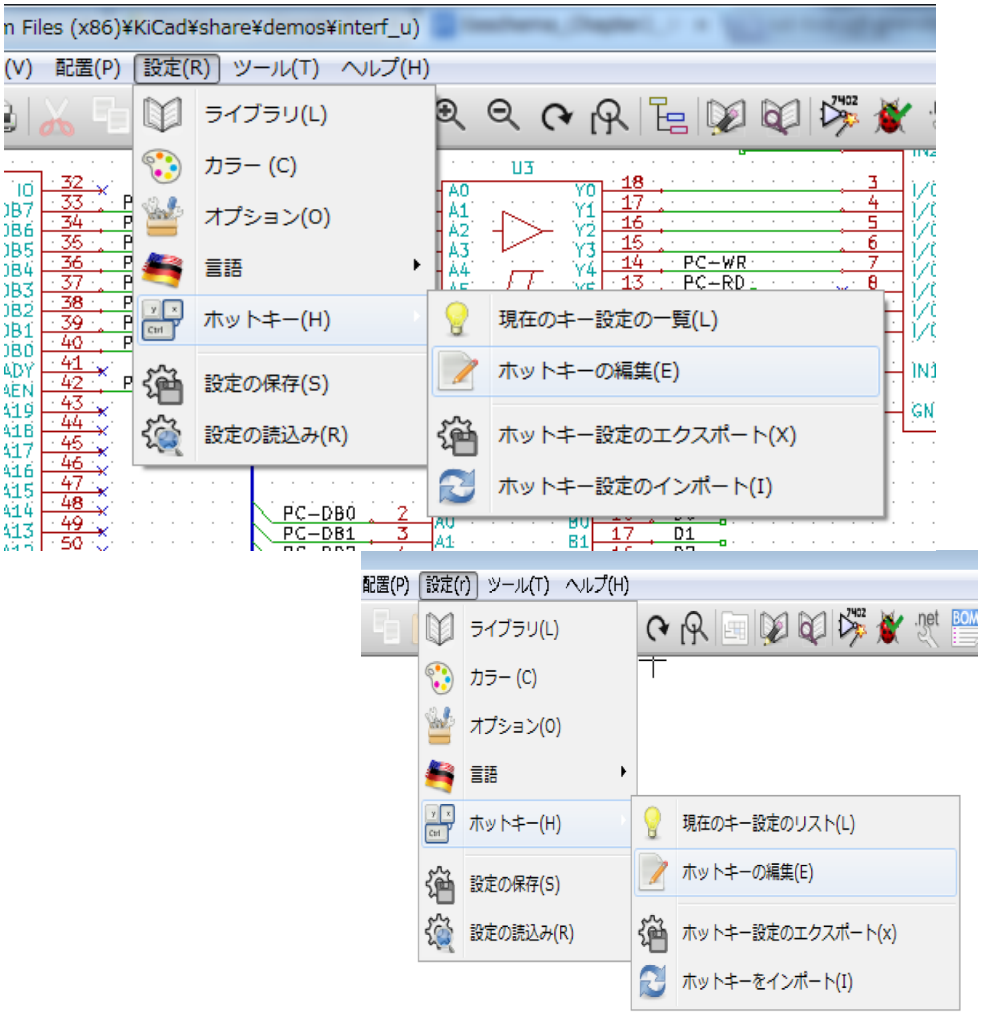
ホットキーは大文字小文字を区別しません.

- “?” キーは現在のホットキーリストを表示する.
- Preference メニュがホットキーを管理する.



デフォルトのホットキーリストはこちら.

ユーザーはホットキーエディタから，全てのホットキーを編集できます.



2.5 - グリッドサイズの選択

Eeschema 上のカーソルは，グリッド表示の有無に関係なく，グリッドごしに動きます．ライブラリ管理メニューでは，常にグリッドが表示されます．

ポップアップメニューあるいは「設定/オプション」メニューから，グリッドサイズを変えることができます． デフォルトのグリッドサイズは 50 mil (0.050 ") あるいは 1,27mm です．

半分のグリッド (20 mil)、あるいは，より細かいグリッド (10 mil)でも作業できます． しかしながら，これは通常の作業では推奨されていません． 半分あるいは細かいグリッドは，特に数百ピンのような大きなピン数のコンポーネントを設計したり，取り扱ったりする場合に使用します．

2.6 - ズームの選択

ズームレベルを変えるには:

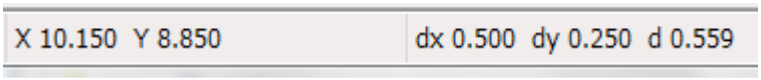
- 右クリックしてポップアップメニューを開き，希望のズームを選ぶ．
- あるいはファンクションキーを使って:
  - F1: ズームイン
  - F2: ズームアウト
  - F3: 再描画
  - カーソル近辺を中央に移動（この操作は，マウスを止めたまま中央ボタンをクリックするのと同じです．）
- Window Zoom: マウスの中ボタンを使ってドラッグする．
  - マウスホイール: ズームイン / ズームアウト
  - SHIFT+マウスホイール: 上/下パン
  - CTRL+マウスホイール: 左/右 パン

2.7 - カーソルの座標表示

表示単位は inch あるいは mm です．しかし Eeschema は内部的には常に 1/1000 inch で扱っています．ウィンドウの下部右側には以下の情報が表示されます:

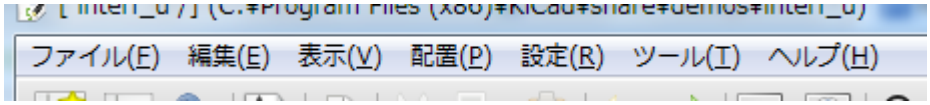
- ズーム倍率
- カーソルの絶対位置
- カーソルの相対位置

相対座標値 (x, y)はスペースキーでリセットされます．リセット後の座標は，リセットした位置が基準となります．



2.8 - トップメニューバー

トップメニューバーでは，回路図やプログラム設定を開いたり，保存したり，ヘルプメニューを開いたりできます．



2.9 - 上部のツールバー

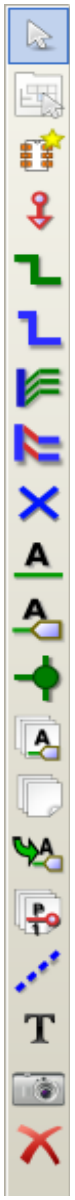
このツールバーから，EeSchema の主な機能へアクセスできます．

	新規回路図の作成.
	回路図を開く.
	回路図全体を保存する（階層も含む）.
	用紙サイズと表題欄の編集.
	プリントメニューを開く.
	ブロック移動の間に選択された要素を削除.
	ブロック移動時に選択された要素をクリップボードにコピー.
	現在のシートで最後に選択した要素，あるいはブロックをコピー.
	取り消し: 最後の変更を取り消す（10 段階まで）.
	やり直し（10 段階まで）.
	コンポーネントのローカライズおよびテキストメニューの呼び出し.

	画面中央近辺をズームイン・アウト.
	画面の再描画とズームの最適化.
	図面階層のツリー構造（サブシートがある場合）と階層のシートの即時選択を表示するナビゲータウィンドウの呼び出し.
	コンポーネントエディタ Libedit (ライブラリコンポーネントの編集、修正、検査)の呼び出し.
	ライブラリの表示 (Viewlib).
	コンポーネントのアノテーション.
	ERC（Electrical Rules Check：電気的接続のチェック）.
	ネットリスト（Pcbnew，Spice フォーマット及びその他フォーマット）の生成.
	BOM（部品表）の生成.
	CVPCB の呼び出し.
	PCBNEW の呼び出し.
	CvPcb から素材ファイル（コンポーネントのフットプリントフィールドを埋めた）をインポート.

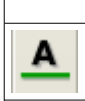
2.10 - 右側のツールバー

このツールバーから，次のツールへアクセスできます:



- コンポーネント，ワイヤ，バス，ジャンクション，ラベル，テキストの配置.
- シートの階層メニューのナビゲーション.
- 階層サブシートと接続シンボルの作成.
- コンポーネントの削除.

これらのツールの詳しい使用法は，「ダイアグラムの作成/編集」で記述しています.
使用法の概要は，以下のとおりです.

	進行中の命令やツールの停止.
	階層ナビゲーション：サブシートのシンボルをクリックしてサブシートを開いたり，サブシートの何もないエリアでクリックすることで階層から上に戻ったりできる.
	コンポーネントの配置.
	電源の配置.
	ワイヤの配置.
	バスの配置.
	バス接続へのワイヤの配置.　これらの要素は装飾的な役割だけなので，配線間の接続に使用すべきではありません.
	バス接続へのバス．これは 2 本のバス同士をつなぐだけです.
	“非接続”シンボル．これは接続されないコンポーネントのピン上に配置します．これは ERC 機能において，ピンが意図的に接続されていないのか，誤って未接続なのかを確認するために使用されます.
	ローカルバスの配置.　異なるワイヤは，同じシート内で同一のラベルを使用すると接続されます．異なるシート間の接続の場合は，グローバルラベルを使用してください.
	グローバルラベルの配置. 異なるシート間であっても，全てのグローバルラベルは接続されます.
	接続点の配置．接続状況がはっきりしない 2 本の交差する配線や，ピンどうしを接続します.
	階層ラベルの配置．シートと，そのシートシンボルを含むルートシート間の接続を配置します.
	階層サブシートシンボルの配置．サイズは変更できます．サブシートのデータを保存するためには，ファイル名を指定する必要があります.
	サブシートからのグローバルラベルのインポート．サブシートのシンボルと接続することができます．グローバルラベルは，もしかして既にサブシートに配置されているかもしれません．こうして作った階層シンボルは，通常のコンポーネントのピンと同様に，必ずワイヤを接続しなければなりません.
	サブシートの接続点を作るグローバルラベルの作成．この機能は上のものとほとんど同じですが，事前にグローバルシンボルを作成しておく必要があります.
	線シンボル．装飾用です．ワイヤのように接続はされません.
	コメントテキストの配置．装飾用.
	ビットマップイメージの挿入.
	選択された要素の削除. いくつかの重なり合った要素が選択された場合には、優先順位は一番小さなものから順になります（ジャンクション，非接続シンボル，配線，バス，テキスト，コンポーネントの順）．これは階層シートにも適用されます．注： ジェネラル ツールバーの“Undelete”機能で最後の削除を取り消すことができます.

2.11 - 左側のツールバー

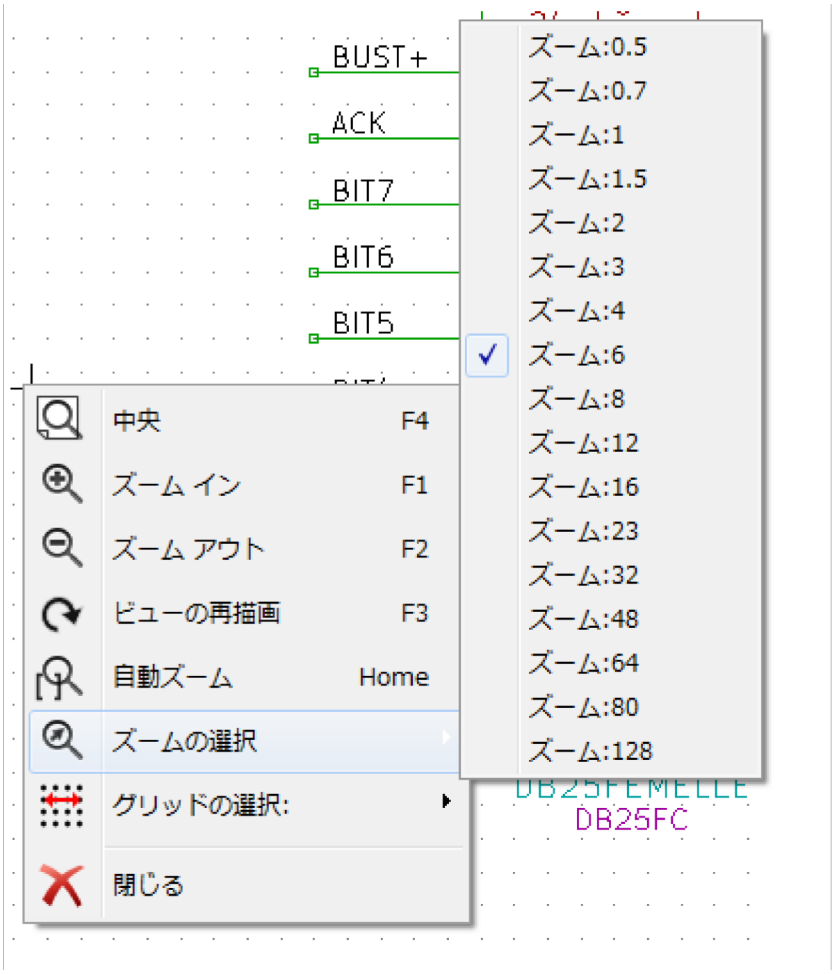
このツールバーは表示オプションを管理します：

- グリッド
- 単位
- カーソル
- 不可視ピン
- ワイヤとバスの許容方向

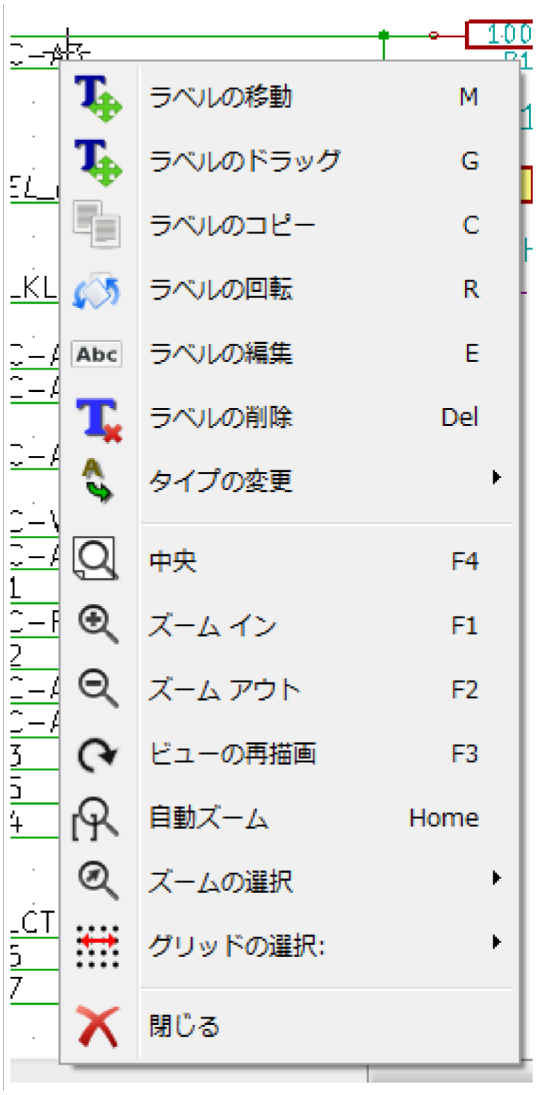
2.12 - ポップアップメニューとクイックエディット

右クリックにより，選択要素に応じたポップアップメニューを開き，下記の機能にアクセスします：

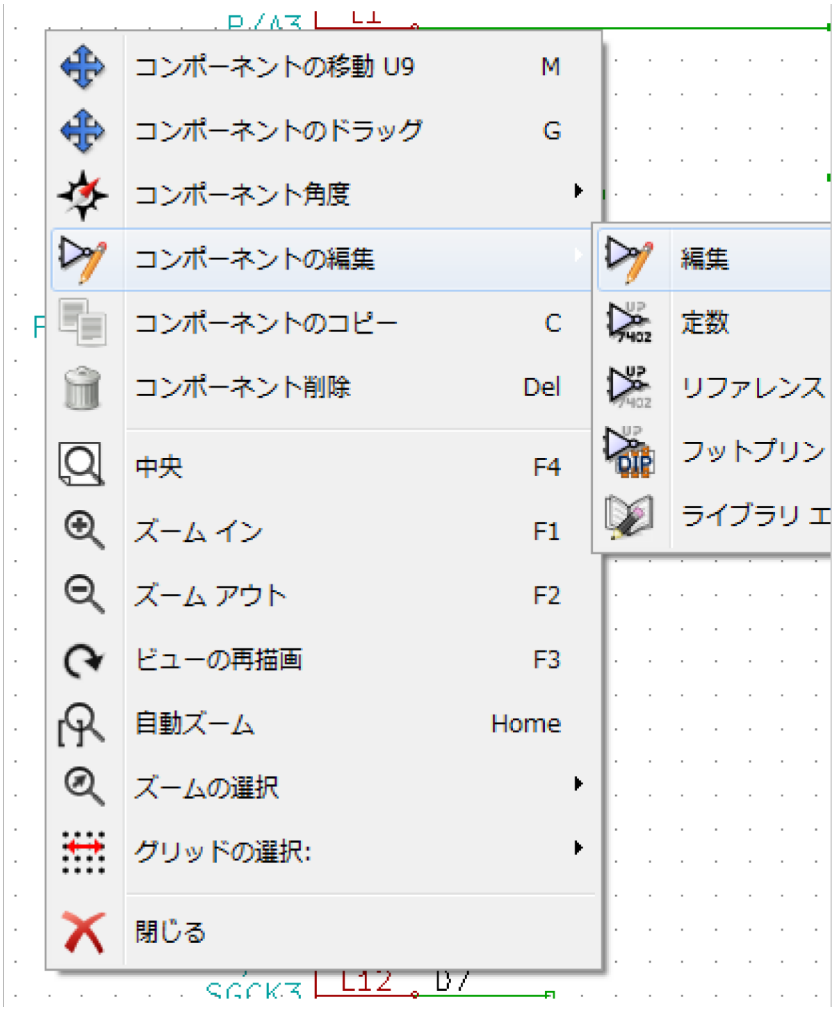
- ズーム倍率
- グリッド調整
- パラメータ編集



要素を選択しないでポップアップメニューの呼出し.


ラベルの移動 M
ラベルのドラッグ G
ラベルのコピー C
ラベルの回転 R
ラベルの編集 E
ラベルの削除 Del
タイプの変更
中央 F4
ズーム イン F1
ズーム アウト F2
ビューの再描画 F3
自動ズーム Home
ズームの選択
グリッドの選択:
閉じる

ラベルの編集.



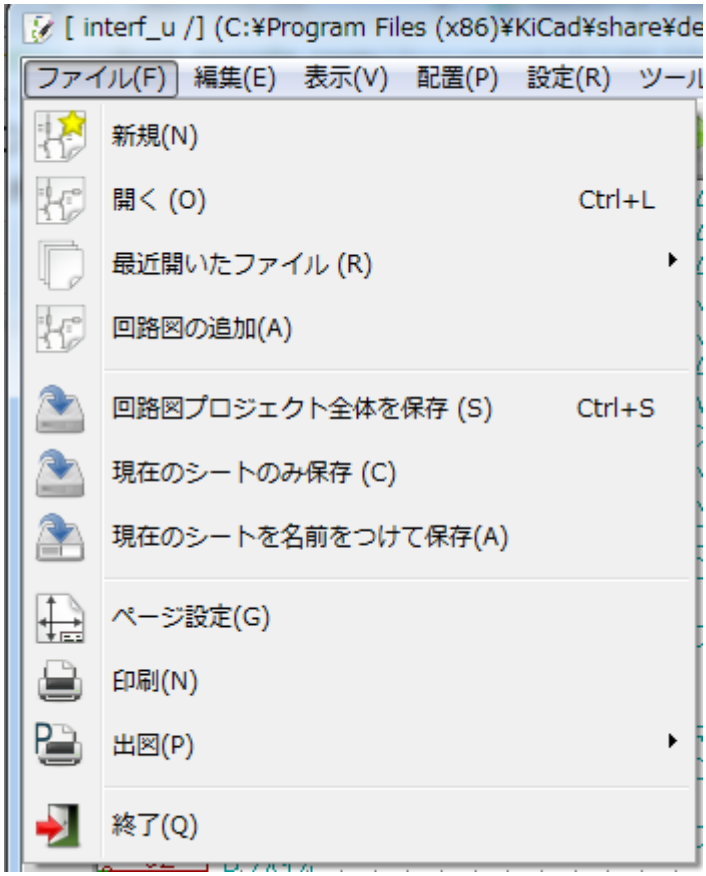
コンポーネントの編集.

3 - メイントップメニュー

目次

3.1 - ファイルメニュー

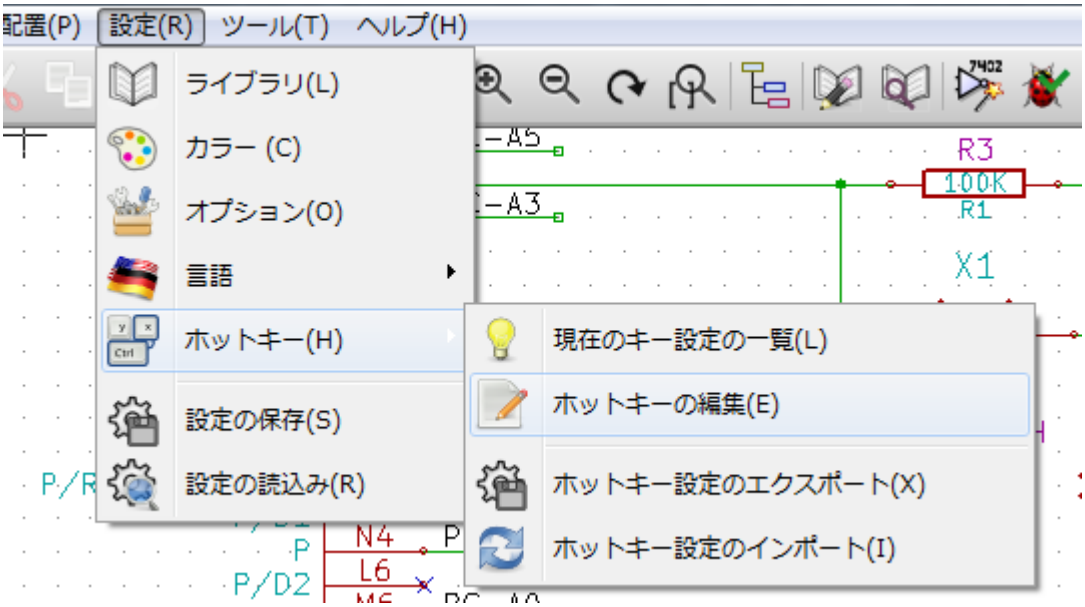
ここでは、"ファイル"メニューがどのように見えるかを確認することができます。



新規	現在の回路図をクリアし，新しい回路図を初期化する.
開く	回路図の階層を開く.
最近開いたファイル	最近開いたファイルのリストから開く.
全ての回路図プロジェクトを保存	現在のシートと全ての階層を保存する.
現在のシートのみ保存	階層の他のシートを除く現在のシートを保存する.
現在のシートを名前をつけて保存...	新しい名前をつけて現在のシートを保存する.
印刷	印刷メニューにアクセスする（「印刷と出図」の章を参照).
出図	Postscript HPGL あるいは SVF フォーマットで出図する （「印刷と出図」の章を参照).
終了	保存せずに終了する.

3.2 - 設定メニュー

設定



ライブラリ	ライブラリとライブラリのパスを選択.
カラー	色を選択.
オプション	表示オプション (単位、グリッドサイズ)を選択.
言語	インタフェースの言語を変更. 主に翻訳者と開発者用.
設定の読込み 設定の保存	設定ファイルの読込みと保存.
ホットキー	ホットキーメニューへのアクセス.

ホットキーサブメニュー

現在のキー設定のリスト	現在のホットキーを表示. ホットキー”？”と同じ.
ホットキーの編集	ホットキーエディタを起動.
ホットキー設定のエクスポート	ホットキー設定ファイルを生成.
ホットキー設定のインポート	以前にエクスポートしたホットキー設定ファイルの読込み.

設定メニュー / ライブラリディレクトリ

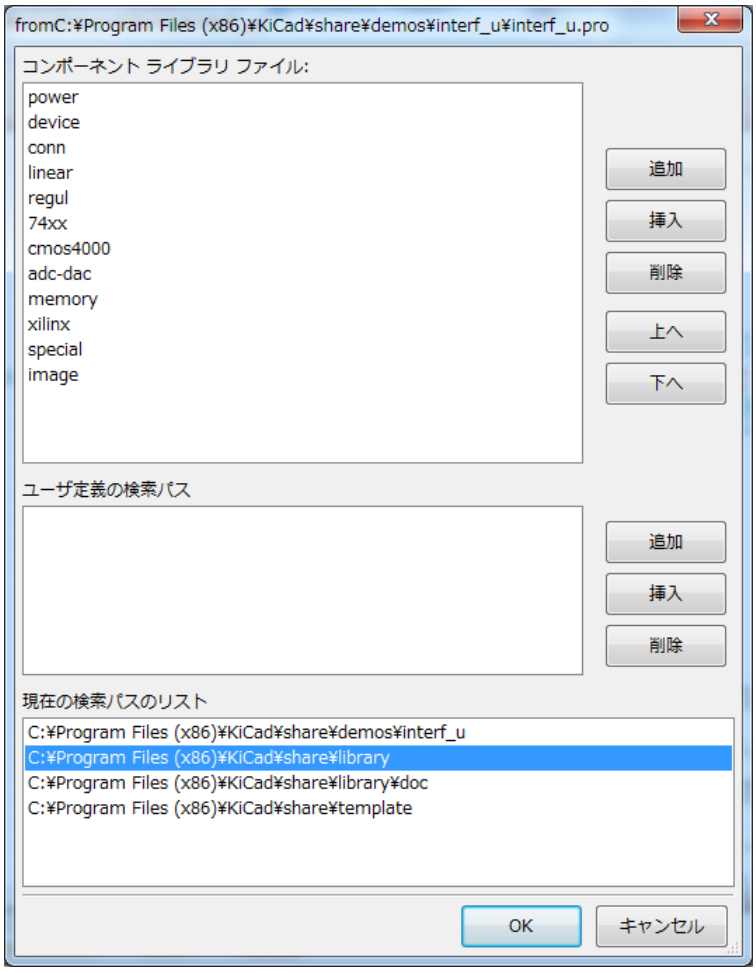
Eeschema の設定は次のものに関係します：

- ライブラリのパス
- ライブラリのリスト

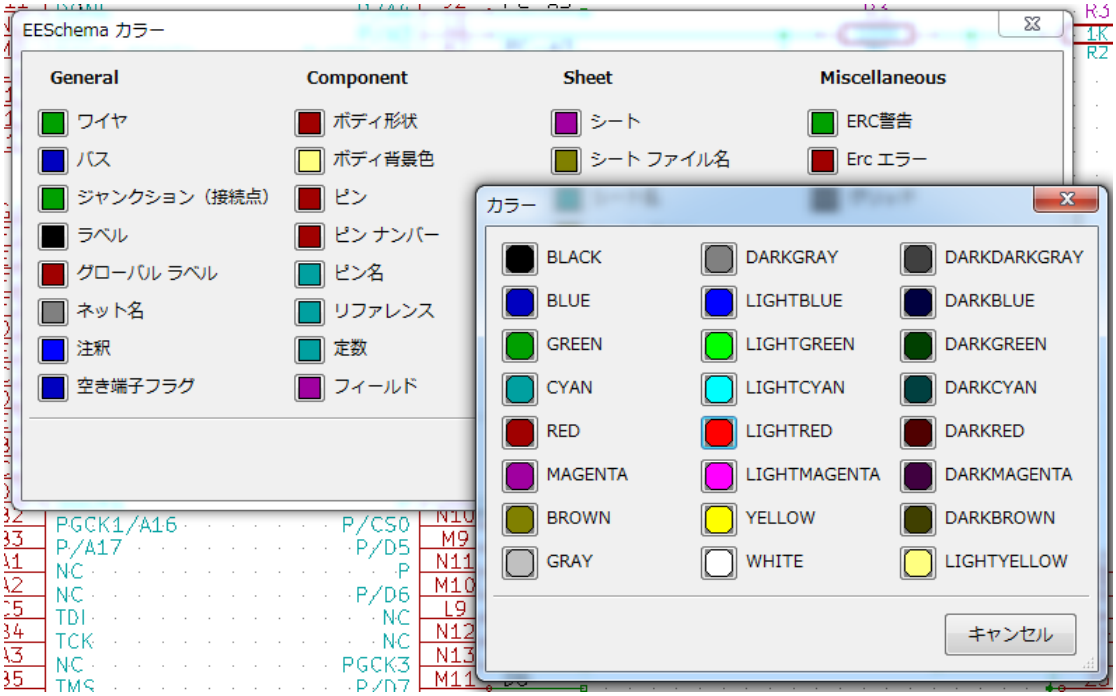
設定パラメータは .pro ファイルに保存されます. 異なるディレクトリでは，異なる設定ファイルを使用することも可能です. EeSchema は次の優先度の順に設定ファイルを検索します.

1. 現在のディレクトリ内の設定ファイル(project>.pro) .
2. Kicad ディレクトリ内の kicad.pro という設定ファイル. このファイルがデフォルト設定となります.
3. ファイルが見つからない場合のデフォルト値。. 少なくともロードするためのライブラリのリストを記入し，設定を保存する必要があります.

4.

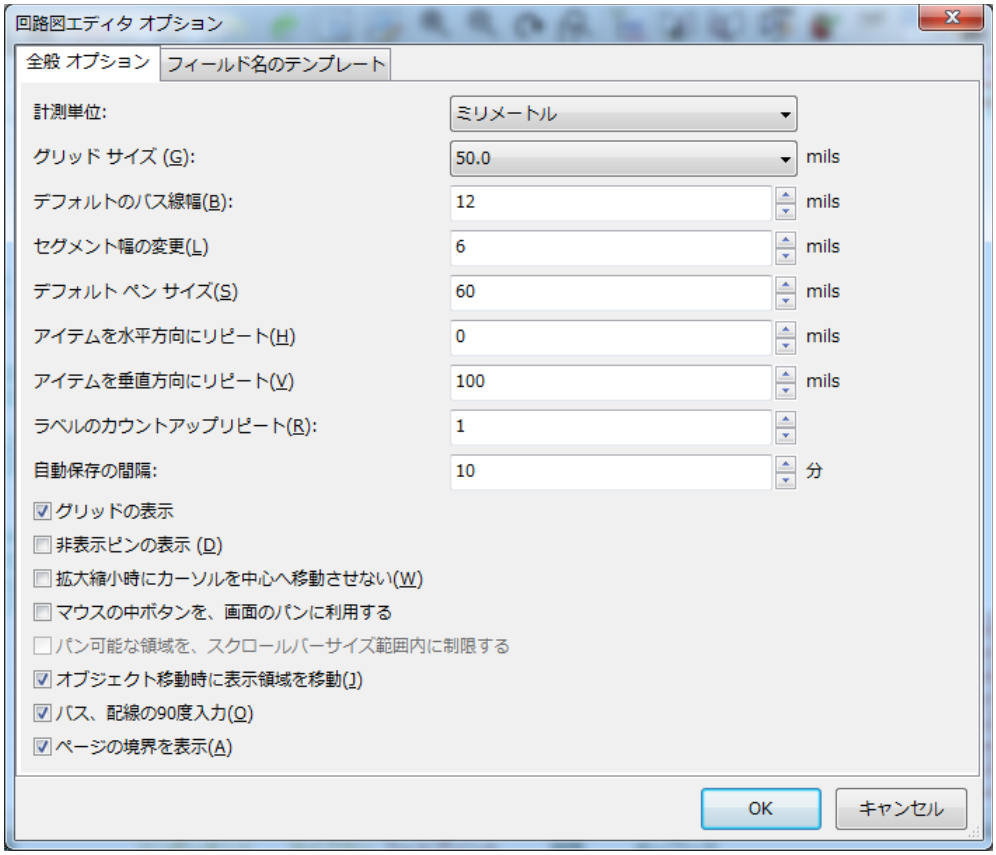


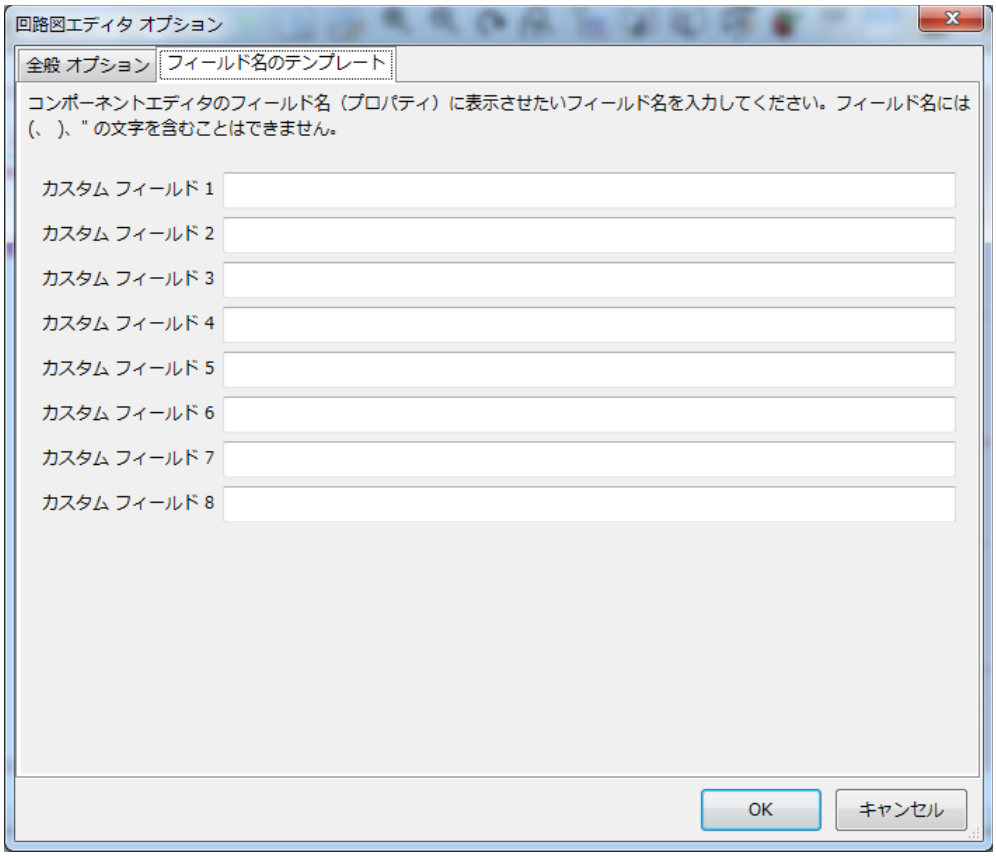
設定メニュー / 色



様々な描画要素の色の選択と、背景色（黒または白のみ）。

設定オプション


回路図エディタ オプション
全般 オプション
フィールド名のテンプレート
計測単位:
ミリメートル
グリッド サイズ (G):
50.0
mils
デフォルトのバス線幅(B):
12
mils
セグメント幅の変更(L)
6
mils
デフォルト ペン サイズ(S)
60
mils
アイテムを水平方向にリピート(H)
0
mils
アイテムを垂直方向にリピート(V)
100
mils
ラベルのカウントアップリピート(R):
1
自動保存の間隔:
10
分
グリッドの表示
非表示ピンの表示 (D)
拡大縮小時にカーソルを中心へ移動させない(W)
マウスの中ボタンを、画面のパンに利用する
パン可能な領域を、スクロールバーサイズ範囲内に制限する
オブジェクト移動時に表示領域を移動(J)
バス、配線の90度入力(O)
ページの境界を表示(A)
OK
キャンセル



計測単位	表示とカーソル座標の単位（インチまたは mm）を選択.
グリッドサイズ	グリッドサイズの選択. 通常グリッド(0.050 インチあるいは 1.27mm)で作業しないといけない．コンポーネント作成には，より細かいグリッドも使用される.
デフォルトの線幅	ペンのサイズは，ペンサイズが指定されていないオブジェクトを描画するために使用される.
デフォルトのテキストサイズ	新しいテキストやラベルが新規作成される時に使用される.
アイテムを水平方向にリピート	要素複写する際の X 軸のシフト値（通常値は 0）. (コンポーネント，ラベル，配線といったアイテムの配置後，インサートキーにより複写される．)
アイテムを垂直方向にリピート	要素複写する際の Y 軸のシフト値（通常値は 0.100 inch あるいは 2.54 mm）.
ラベルのカウントアップリピート	バスの配線のように，一つずつカウントアップする値 （通常は 1 か -1）.
グリッドの表示	チェックがある場合：グリッドを表示.

非表示ピンの表示	非表示ピンの表示. チェックがある場合，電源ピンも表示される.
マウス中ボタンのパン有効化	有効にすると、マウス中ボタンを押した時にカーソル移動に合わせてシート全体が移動する。
マウス中ボタンのパン制限	有効にすると、マウス中ボタンでの表示領域外へのシートは移動できません。
自動パンを有効化	チェックがある場合，配線作業や要素移動の間にウインドウからカーソルが出ると，自動的にウインドウが追従する.
バス、配線の 90 度入力	チェックがある場合，バスや配線は垂直または水平になる. チェックがない場合，バスと配線はどんな傾きにも配置できる.
ページの境界を表示	チェックがある場合，画面上にページの境界が表示される.

言語設定

デフォルトのモードで使用してください．他言語は主にメンテナンス目的で利用されます.

3.3 - ヘルプメニュー

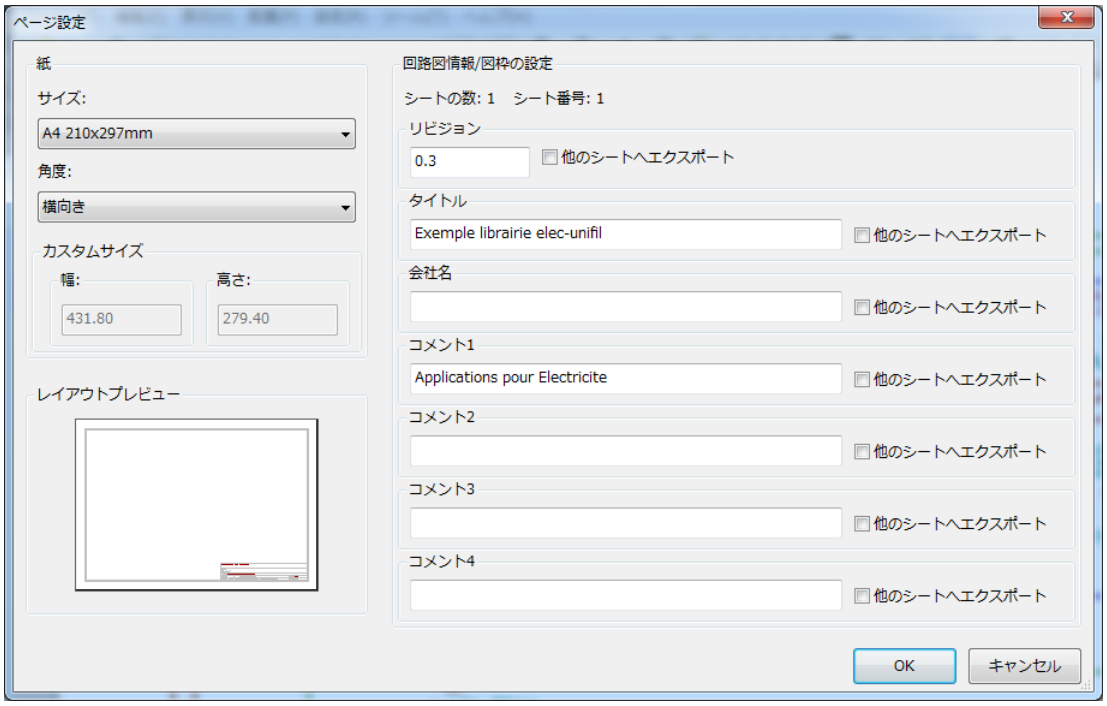
KiCad についての広範囲なチュートリアル用のオンラインヘルプ（この文書）へのアクセスと，Eeschema の現在のバージョンを確認します(Eeschema について).

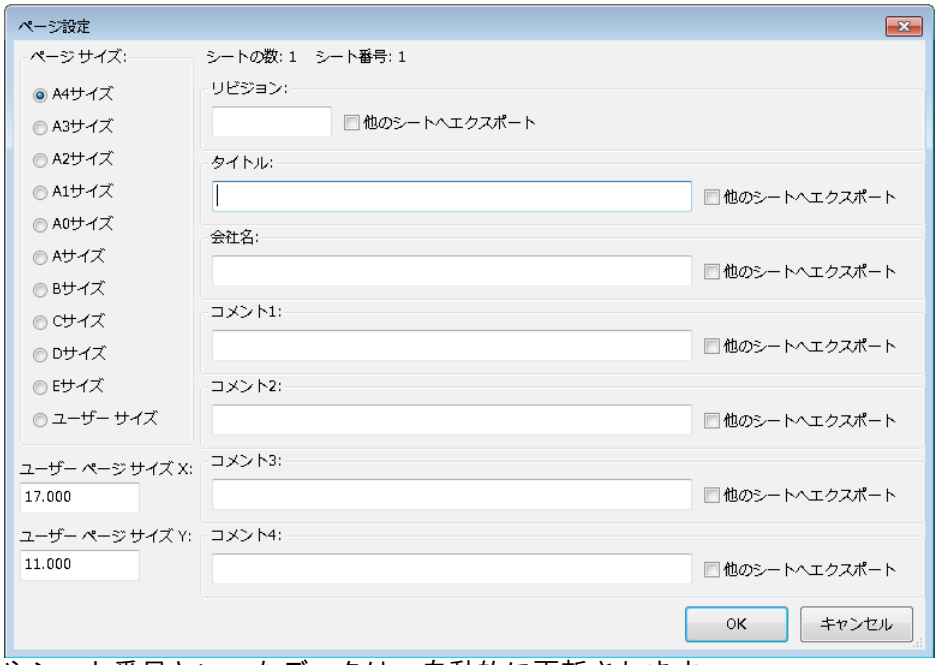
4 - ジェネラルトップツールバー

目次

4.1 - シート管理


アイコンにより、シート管理にアクセスできます。ここでは用紙サイズと右下隅にある表題欄の様々なテキストセクションを定義することができます.



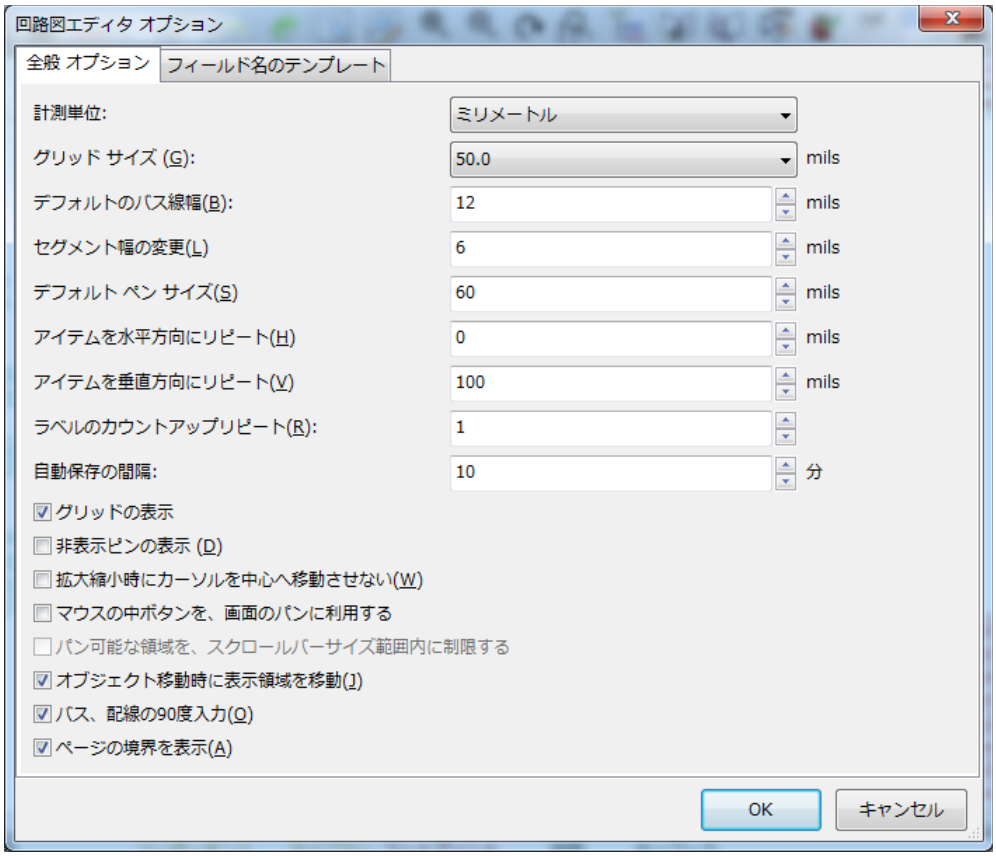


シート総数やシート番号といったデータは，自動的に更新されます.

4.2 - 回路図エディタのオプション

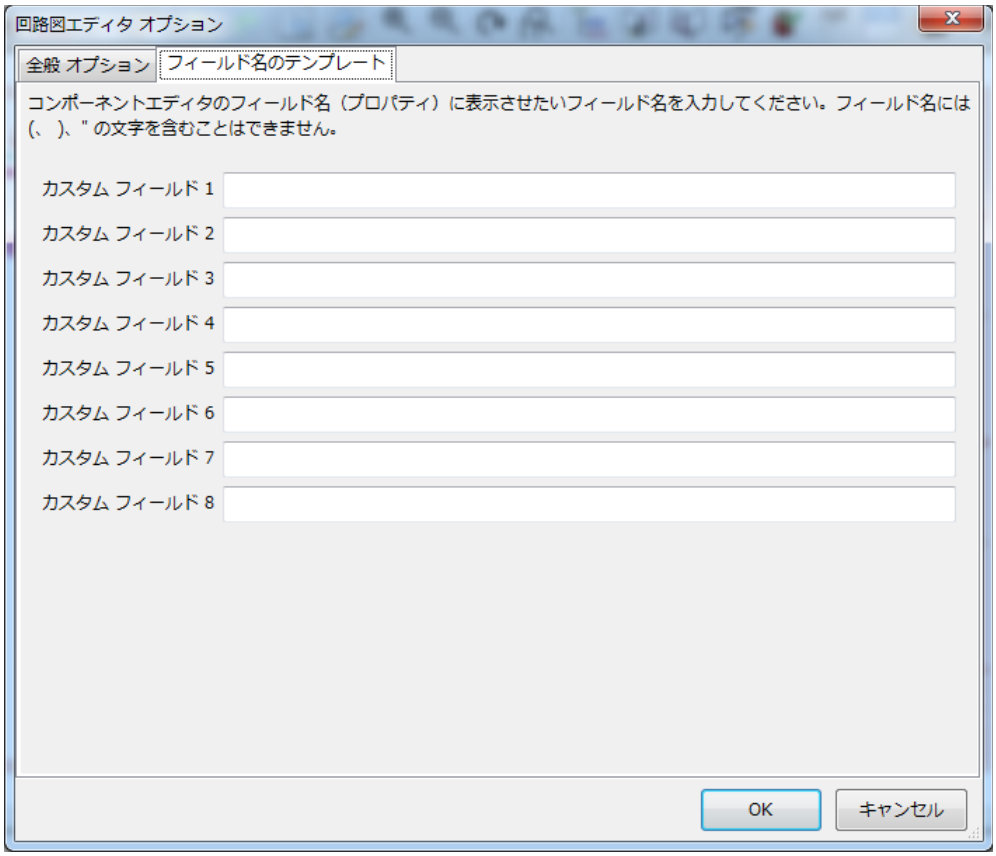
全般オプション

これらのオプションは図面に関係するものです.



フィールド名のテンプレート

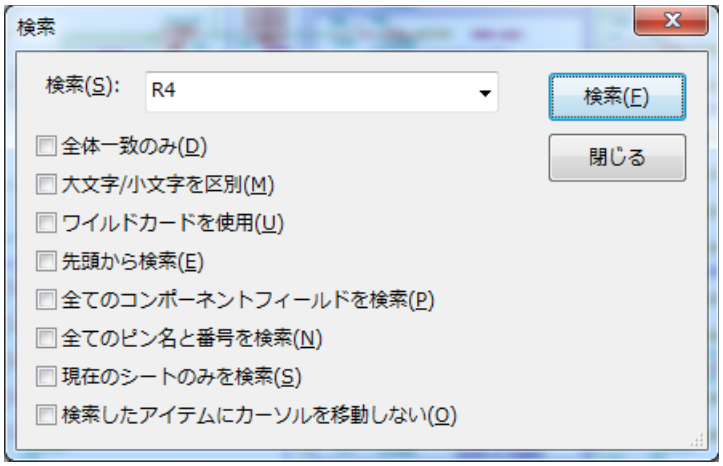
　各々のコンポーネントにあるカスタムフィールド（与えられたコンポーネントが空フィールドのままでも）を定義することができます.



4.3 - 検索ツール

 このアイコンは，検索ツールへのアクセスに使われます．

現在のシートの全体あるいは全体の階層内から，コンポーネント，値，テキスト文字列を検索することができます． 見つかると，関係するサブシートの見つかった要素の上にカーソルが移動します．



4.4 - ネットリストツール

このアイコンから，ネットリストファイルを生成するネットリストツールを呼出します．

このネットリストファイルは，階層全体（通常のオプション），あるいは現在のシートのみ（部分的なネットリストが出力されてしまいますが，このオプションはいくつかのソフトウエアには有用です）に適用できます．

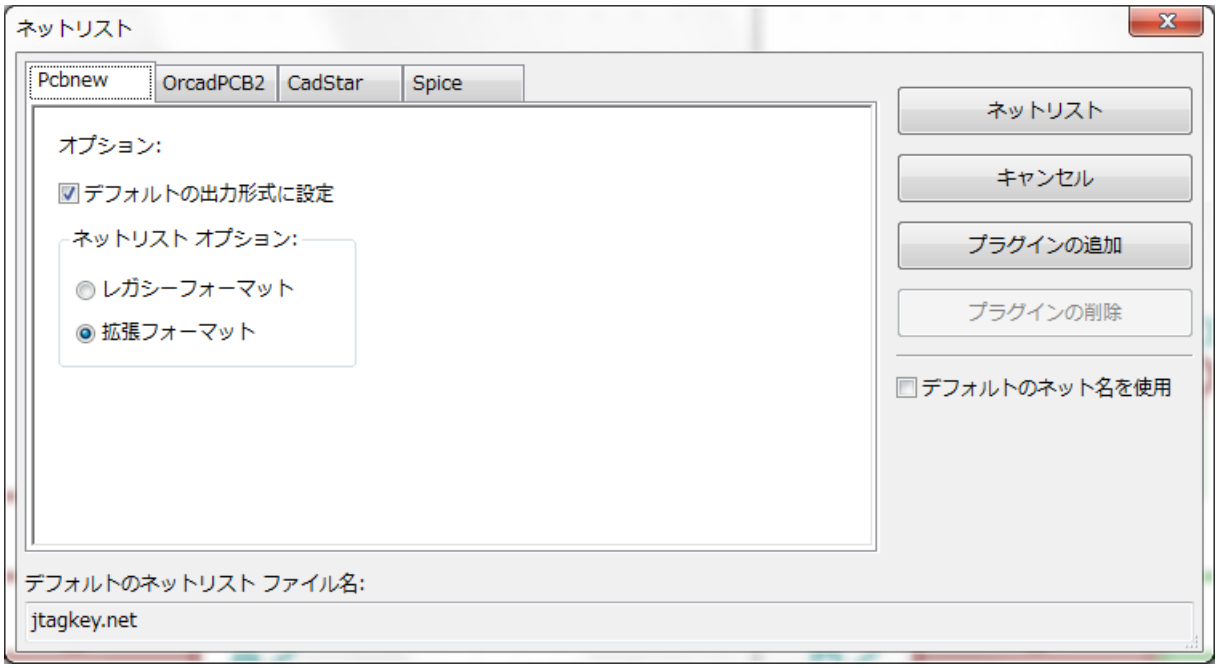
マルチシートの階層において，ローカルラベルはその属するシート内だけで通用します．

したがって，シート 3 のラベル TOTO は，シート 5 のラベル TOTO とは接続されません（意図的にそれらを接続する手立てがされてない場合）．これは，シート番号（アノテーションコマンドによって更新される）がローカルラベルと関連付けられているからです．前の例では，最初の TOTO ラベルは内部では TOTO_3 で，2 番目の TOTO ラベルは内部では TOTO_5 と定義されています．

この関連付けは望めば抑制できますが，望ましくない接続が起こりうることを知っておいてください．

注１：EeSchema にはラベルの長さの制限はありませんが，生成されたネットリストを利用するソフトウェアには制限がある場合があります．

注２：区切られた単語として表示されてしまうため，ラベルの中では空白文字(スペース)を使うべきではありません．それは EeSchema の制限ではなく，多くのネットリストフォーマットのおいて，ラベルは空白文字を含んでいないものだと定義されているからです．



オプション：
デフォルトのフォーマット ：

デフォルトのフォーマットとして Pcbnew を選択するためには，チェックを入れてください．

以下の他のフォーマットも生成できます．

- Orcad PCB2
- CadStar
- Spice, シミュレータ用

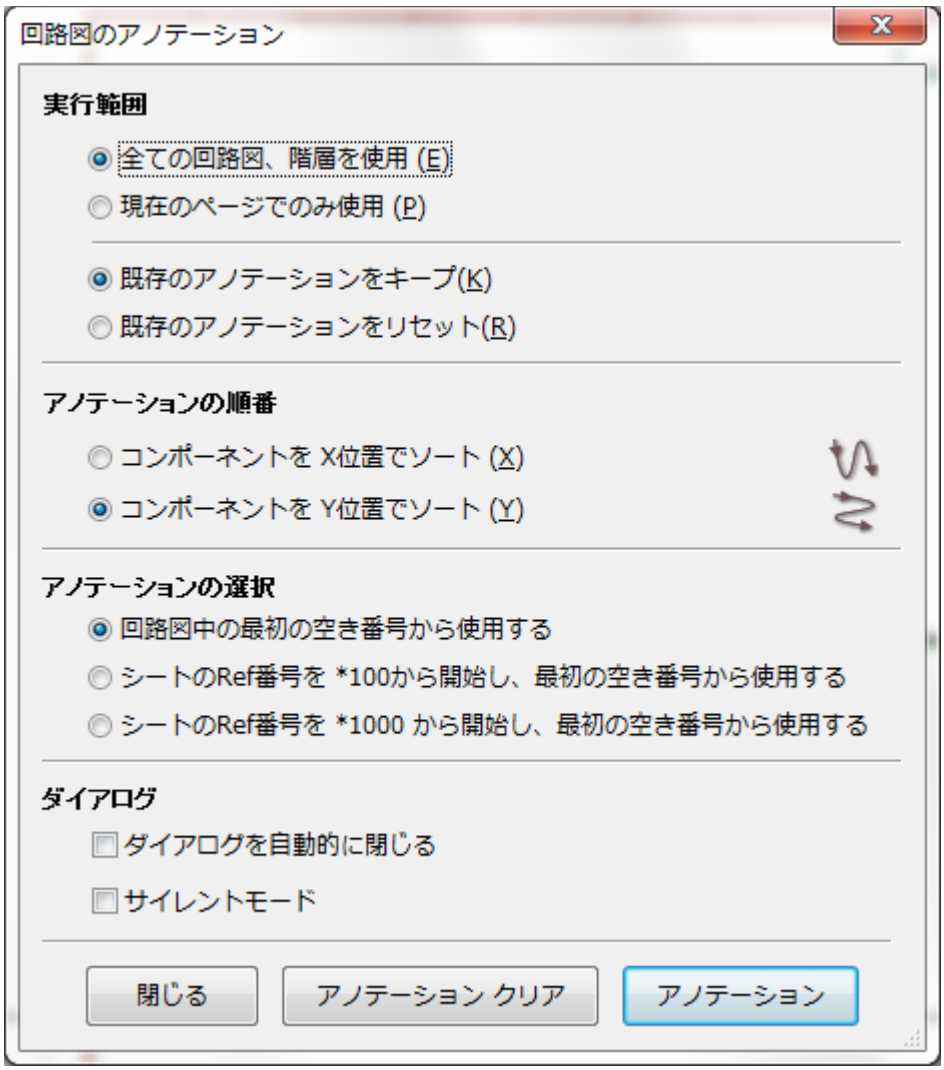
プラグインの追加から，その他のネットリストフォーマットを拡張して生成することもできます（PadsPcb プラグインなど）．

4.5 - アノテーションツール

このアイコンからアノテーションツールを呼出します．このツールは全ての使用されているコンポーネントに対し自動的に名前の割付を行います．

マルチパートコンポーネント(4 ゲート入りの 7400 TTL など）には，マルチパートの接尾辞が割り当てられます．従って，U3 に指定された 7400TTL は，U3A，U3B，U3C，および U3D に分かれます．

全てのコンポーネントに無条件に，あるいは前にアノテートされていない新規のコンポーネントだけにアノテートすることもできます．



スコープ
1) 全ての回路図，階層を使用：全てのシートを再アノテート (通常の選択)
2) 現在のページでのみ使用：現在のシートのみ再アノテート（このオプションは特別な場合のみ使用されます． 例えば，現在のシートの抵抗量を評価する場合など）

3) 既存のアノテーションをキープ：条件付のアノテーション．新しいコンポーネントのみ再アノテート（通常の選択）．

4) 既存のアノテーションをリセット：無条件のアノテーション．全てのコンポーネントを再アノテート（このオプションは重複した表記がある場合に使われる）

順序

コンポーネントへのアノテーション番号設定のためのソートオプション．

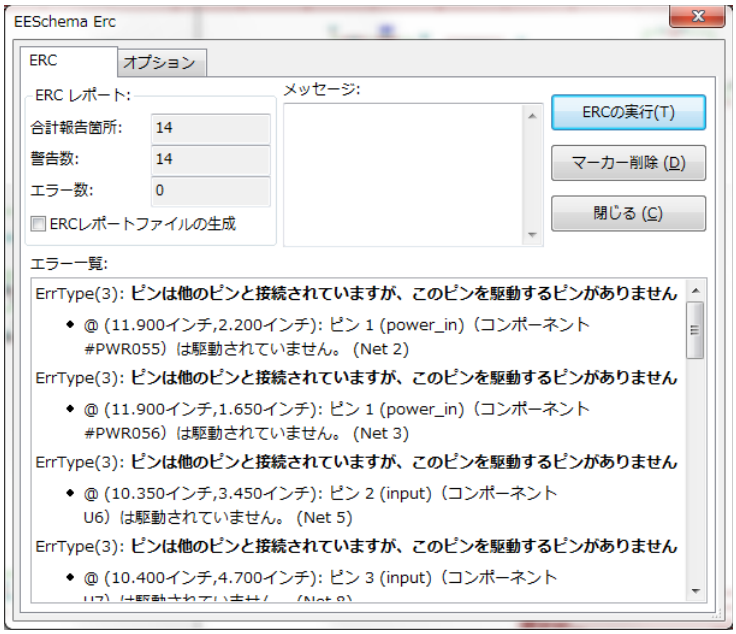
4.6 - ERC（電気的ルールチェック）ツール

 このアイコンから ERC（電気的ルールチェック）ツールを呼出します．

このツールは設計の検証を行い，特に接続忘れや矛盾を検出するのに有用です．

一度 ERC を実行すると，EeSchema はラベルやピン上に問題を目立たせるマーカーを配置します． 診断結果はマーカー上で左クリックすると出てきます， エラーファイルを生成させることもできます．

ERC メインダイアログ



エラーは，Erc ダイアログボックスに表示されます：

•エラーと警告の合計
•エラー数
•警告数

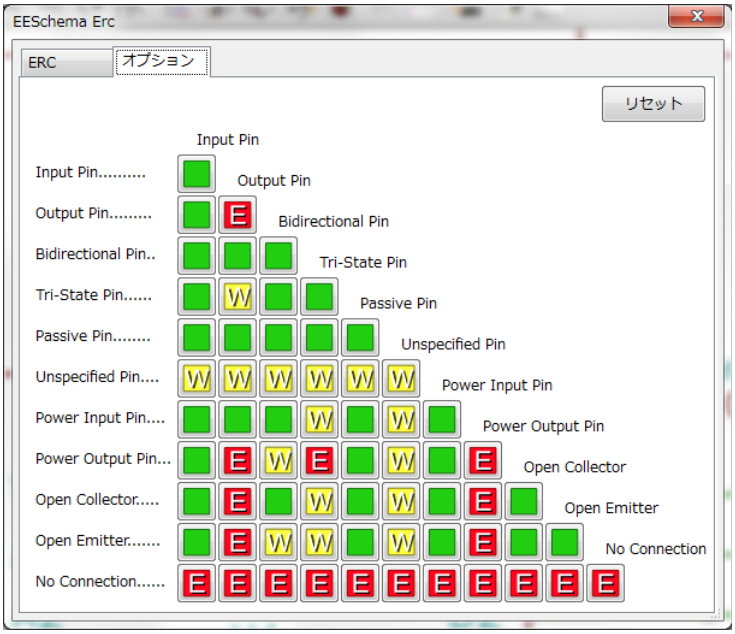
オプション：

•ERC レポートの作成：ERC レポートファイルを生成するには，このオプションをチェックします．
コマンドボタン：

•Erc のテスト：電気的ルールチェックを実行．
•マーカー削除：ERC マーカーを消去．
•閉じる： このダイアログボックスを閉じる．

注：エラーメッセージをクリックすると，回路図の対応するマーカーにジャンプします．

ERC オプションダイアログ



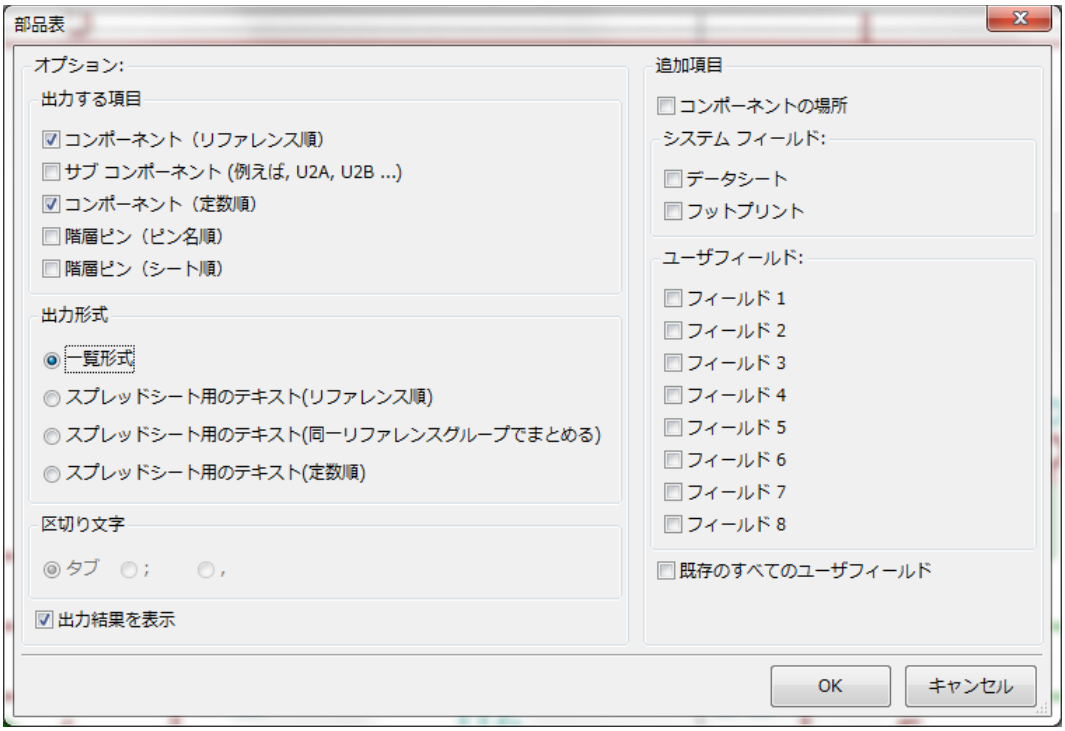
このERC設定ダイアログボックスは，ピン間の接続ルールを確立することができます． それぞれのケースに対して，3つのオプションから選択することができます．

- エラーなし（No error： 緑）
- 警告（Warning： 黄）
- エラー（Error： 赤）

マトリックスのそれぞれの四角上でクリックすることにより，内容を変更できます．

4.7 - BOM（部品表）ツール

このアイコンからBOM（部品表）ツールを呼出します．このメニューから，コンポーネントと階層接続グローバルラベルの一覧表が生成できます．



コンポーネントは次の内容で並べられます．

- リファレンス
- 定数

マルチパートコンポーネントは詳述できます．グローバルラベルは以下のように並べられます．

- アルファベット順の分類
- サブシート

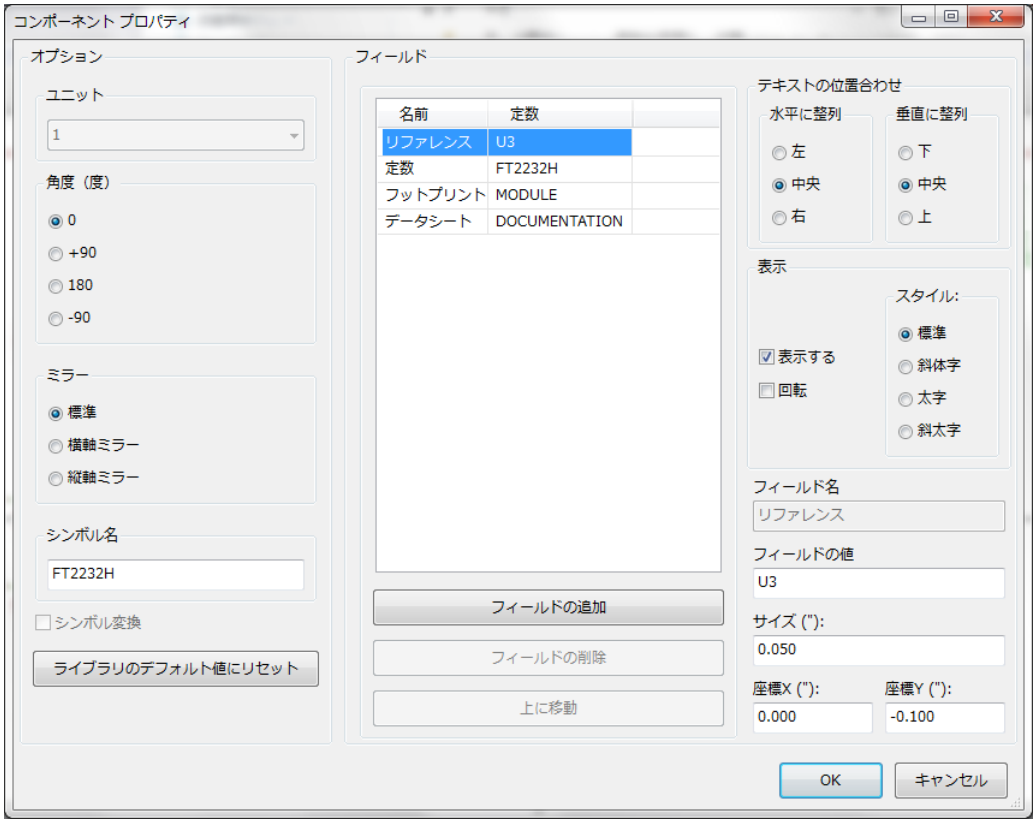
異なる種類の並べ替えを同時に使用できます．オプションは次のとおり．

コンポーネント（リファレンス順）	リファレンス順で部品表をソート．
コンポーネント（定数順）	定数順で部品表をソート．
サブコンポーネント	BOM 上でマルチパートコンポーネントの U2A，U2B といった全てのデバイスを表示．
階層ピン（ピン名順）	アルファベット順に階層ピンをソート．
階層ピン（シート順）	シート番号順に階層ピンをソート．
リスト	印刷用のプレーンテキストファイルを生成．
スプレッドシートインポート用のテキスト	スプレッドシートへ容易にインポートできる，コンマ区切りテキスト形式のファイルを生成．
一行ごとに一つのパーツ	単一の行に同じ定数を持つコンポーネントを組み合わせ，カンマ区切りの参照指定子をリストにした CSV ファイルを生成．
リストブラウザの起動	BOM リストファイル作成後，読み込んで編集するためのエディタを起動．

BOM に使用される一連のコンポーネントプロパティは，次のとおりです．

- Value – 各使用部品のユニーク名
- Footprint – 手入力か，バックアノテートか (下記参照)
- Field1 – 製造業者名
- Field2 – 製造業者の部品番号
- Field3 – ディストリビューターの部品番号

例:



1 行 1 部品の BOM フォーマットを使うためには，同じ値をもつ全てのコンポーネントではなく，回路図上の全てのコンポーネントのプロパティを編集する必要があります．

しかし，両方とも 33k の値で一方が 1/10W，もう一方が 1/4W，あるいは異なるフットプリントをもつ違った部品の場合には，一方を 33k，他方を 33kBig と区別することで，異なる部品としてリスト化します．

ここで生成されるテキストファイルやスプレッドシートを使うことで，部品の管理や調達の手間が簡素化されます．

4.8 - フットプリント割当用インポートツール

アクセス

このアイコンからバックアノテートツールを呼出します．

このツールは回路図の作成や，CvPcb の表，ブラウザツールを使ってのフットプリントの割当ができ，その後回路へのフットプリントのエクスポートバックが可能です．

この機能は，前に CvPcb によって予め作成された.cmp ファイルを読み込み，コンポーネントのフットプリントフィールド（Field3）を初期化します．

これは PcbNew にとっては必須ではありませんが、部品表，ネットリストを作成する際フットプリントフィールドを追加するのは便利です．

この機能は単一のソースファイルにコンポーネントのフットプリント/参照情報を維持し，ネットリストのための余分な.cmp ファイルを作ります．

フットプリントの割当は，EeSchema からネットリストを生成すると確認できます．

Pcbnew に関する注意注意事項

単にコンポーネントへのフットプリント割付けのために .cmp ファイル或いはネットリストどちらを使用するかは、Pcbnew 内部で選択します。

Pcbnew は、.net ファイルに対応する.cmp ファイルが見当たらない場合には，.net ファイルの中にあるコンポーネントプリント/リファレンスを使用します．しかし，.cmp ファイルを使う方が望ましいでしょう．なぜなら，設計者が PcbNew からフットプリントの割当を変える場合には，対応する.cmp ファイルも更新されるかもしれないからです．

5 - 回路図の作成と編集

目次

5.1 - はじめに

回路図は 1 枚のシートのみを使用しても作成可能ですが，規模の大きな回路図の場合は複数のシートで構成することもできます．

回路図が複数のシートから構成される場合を階層構造と呼び，構成する全てのシート（各シートはそれぞれ 1 つのファイルから成っています）が EeSchema のプロジェクトを構成することになります．複数シートで構成されるプロジェクトは"ルート"と呼ばれるメインの回路図と，階層を構成するサブシートから成っています．

プロジェクトを構成するファイルをすべて正しく認識させるためには，これから説明する図面作成のルールに従う必要があります．

以下では，プロジェクトについての説明は，単一シートで構成される回路図と，階層構造の複数シートから構成される回路図の両方の説明をします．また，追加のスペシャルセクションでは，階層構造についてその性質と使い方を解説します．

5.2 - 基本的な検討事項

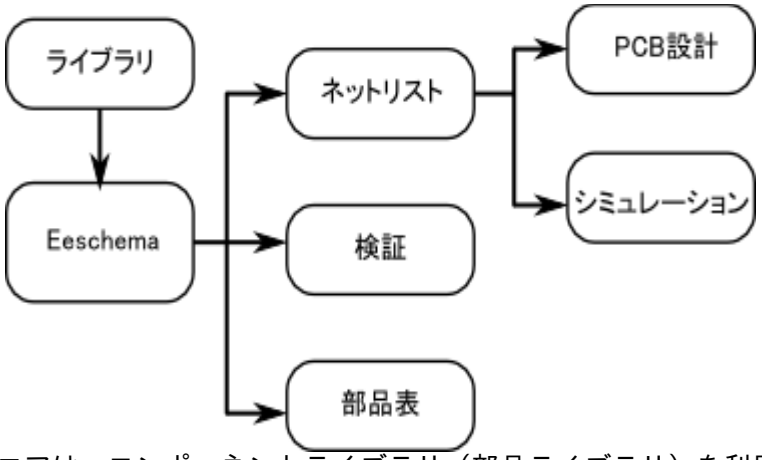
EeSchema を使う回路図設計は，単なる回路図画像の描画に留まりません．この回路図設計は，開発フローのスタートとなります．

- 回路図の誤りや欠落の検出（ERC： Electrical Rules Check）．
- 部品表（BOM： Bill Of Material）の自動生成．
- Pspice などの回路シミュレータのためのネットリスト生成．
- PcbNew を利用したプリント基板設計のためのネットリスト生成．－ 回路図とプリント基板間の整合性チェックは自動的に，リアルタイムで行われます．

これらすべての機能を利用するためには，回路図設計時にルールを守る必要があります．

回路図は主にワイヤ，ラベル，ジャンクション，バス，電源から構成されます．また，回路図を見易くするためにバスエントリ，コメント，破線などのグラフィック要素も配置できます．

5.3 - 開発フロー



回路図設計ソフトウエアは，コンポーネントライブラリ（部品ライブラリ）を利用しています．別の設計ソフトから利用するためには，回路図設計ファイルに加えてネットリストも重要となります．

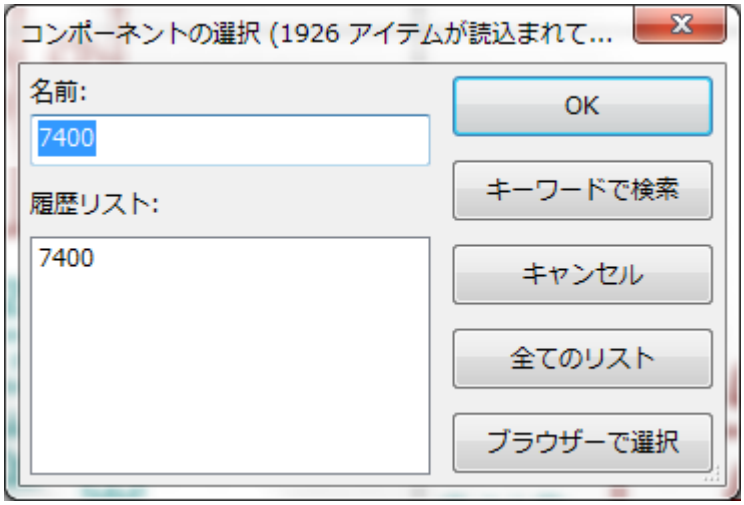
ネットリストは回路図の情報をもとに，コンポーネント間の接続情報を提供するものです．

ネットリストのフォーマットには膨大な種類が存在します．中には Spice 形式のネットリストなど，一般的に多く使われている形式もあります．

5.4 - コンポーネントの編集と配置

コンポーネントの検索と配置

 このアイコンから，回路図にコンポーネントをロードします．描画したい位置をクリックし，そのコンポーネントを配置します．下記の"コンポーネント選択"ダイアログボックスでは，ロードするモジュールの名前を指定することができます．



このダイアログボックスには，最近利用されたコンポーネントも表示されます．

"*"を入力するか，または"すべてのリスト"ボタンをクリックすることで，使用可能な全てのライブラリの一覧を表示します．

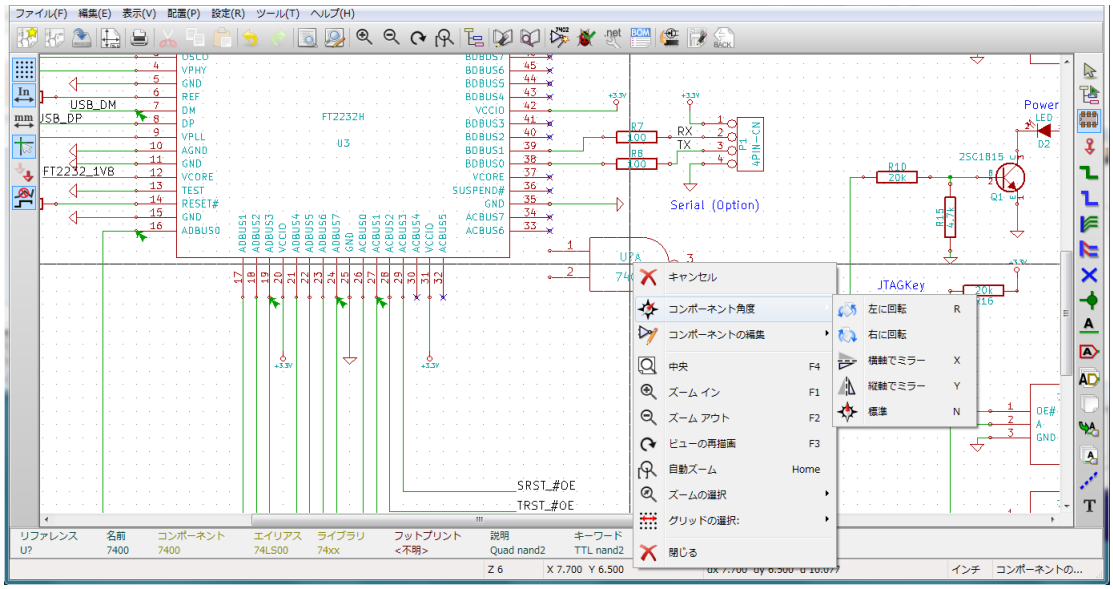
また，条件を指定してリストを絞り込むことができます．例えば，"LM2*"と入力すると，LM2 から始まる名前の全てのコンポーネントが表示されます．

選択したコンポーネントは，配置モードで画面に表示されます．

左クリックでコンポーネントを目的の位置に配置する前に，右クリックで表示されるメニューを利用し，90 度ごとにコンポーネントを回転させたり，あるいは横軸，縦軸でのミラーとすることができます．これらの変更は，部品を配置した後でも簡単に行うことができます．

もし必要なコンポーネントがライブラリに存在しなかった場合でも，同じようなコンポーネントロードし編集することができます．例えば，ライブラリに存在しないコンポーネント"54LS00"を配置したい場合，まず 74LS00 をロードし，74LS00 の名前を 54LS00 へ変更することができます．

配置中のコンポーネントは次のようになります．



電源ポート

電源ポートもコンポーネントのひとつです（"power"ライブラリに分類されています）．よって，これまで説明したコンポーネントの配置と同じ手順で配置することができます．しかし，これら電源ポートは頻繁に配置されるものなので，  アイコンのツールを利用することで簡単に配置することができます．このツールは直接"power"ライブラリを参照するため，毎回"power"ライブラリまでコンポーネントを辿る手間を省くことができます．

配置されたコンポーネントの編集と修正

コンポーネントの編集や修正は，下記の２種類に分類されます：

- コンポーネントそのものの編集（部品の位置や向き，複数部品からなる部品ひとつの選択）．
- コンポーネントのフィールドの修正（参照番号，定数など）．

コンポーネントが配置された直後は，必要に応じてそれらの部品定数を設定します（特に抵抗やコンデンサなど）．しかし，コンポーネントを配置した直後に部品の参照番号の割り当てを行ったり，あるいは 7400 のような複数部品からなるコンポーネント内の部品番号を設定したりすることは無駄になってしまうかもしれません．

これらコンポーネントの参照番号や部品番号は，アノテーション機能を使うことで後から自動的に割り振ることができるのです．

コンポーネントの変更

マウスカーソルをコンポーネント上（定数などのフィールド以外の場所）へ移動させ，以下のいずれかの操作をすることでコンポーネントのプロパティを編集することができます．

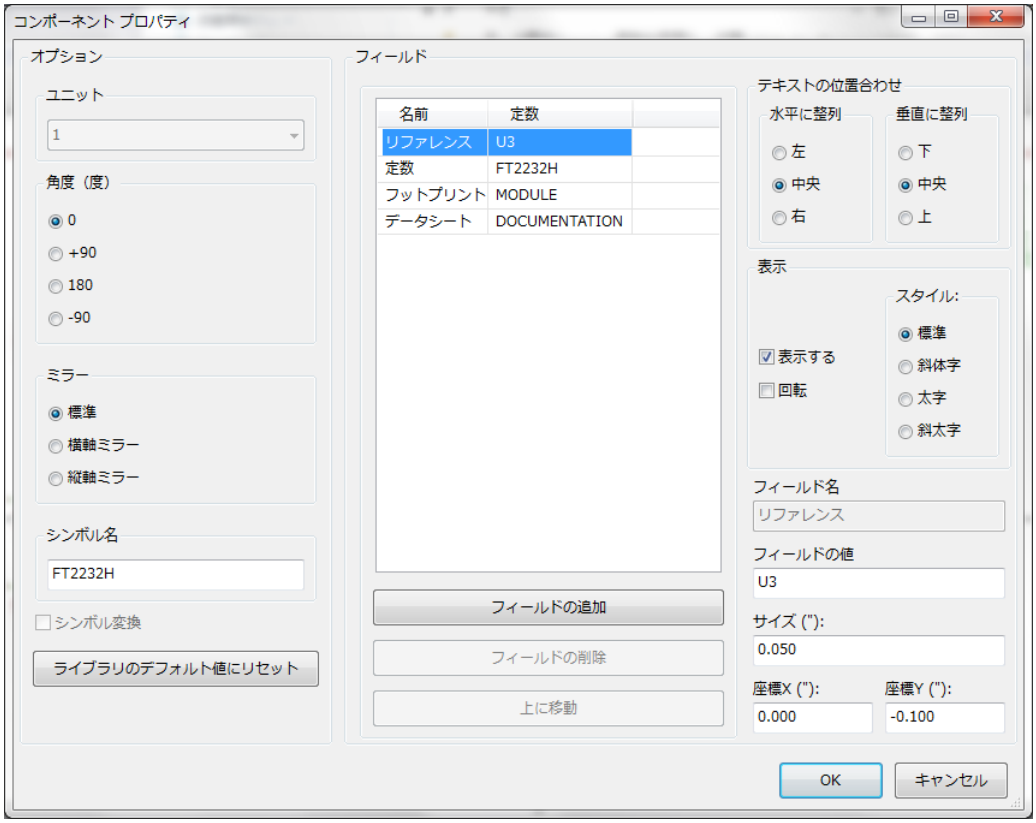
- コンポーネントをダブルクリックし，全てのプロパティを編集することができるダイアログボックスを開く．
- 右クリックしポップアップメニューを出し，表示された編集コマンドを選択する（移動，回転，編集，削除など）．

テキストフィールドの編集

参照記号や定数，位置，向き，フィールドの可視範囲などの設定を変更することができます．以下のいずれかの手順で簡単に変更可能です．

- テキストフィールドをダブルクリックし編集する．
- 右クリックしポップアップメニューをに表示されたコマンドを使用（移動，回転，編集，削除）．

フィールドに関する全てのプロパティを編集する場合や，新たにフィールド項目を作成する場合には，コンポーネントをダブルクリックし，"コンポーネントプロパティ"ダイアログボックスを表示させます．



コンポーネントの角度などの編集や，フィールドの編集，追加，削除を行うことができます．

それぞれのフィールドについて，表示/非表示や表示方向を設定することができます．表示位置に関する設定は，（回転やミラー表示する前の）コンポーネントの配置座標からの相対的な座標で指定されます．

"ライブラリのデフォルト値にリセット"ボタンを利用することで，各フィールドの向きやサイズ，位置が初期化されます．しかし，回路図情報を壊してしまう可能性があるため，フィールドのテキスト内容は変更されません．

5.5 - ワイヤ, バス, ラベル, 電源ポート

はじめに

これらの要素は画面右に縦表示されているツールバーを利用して配置することができます．

このツールバーの機能を以下に示します

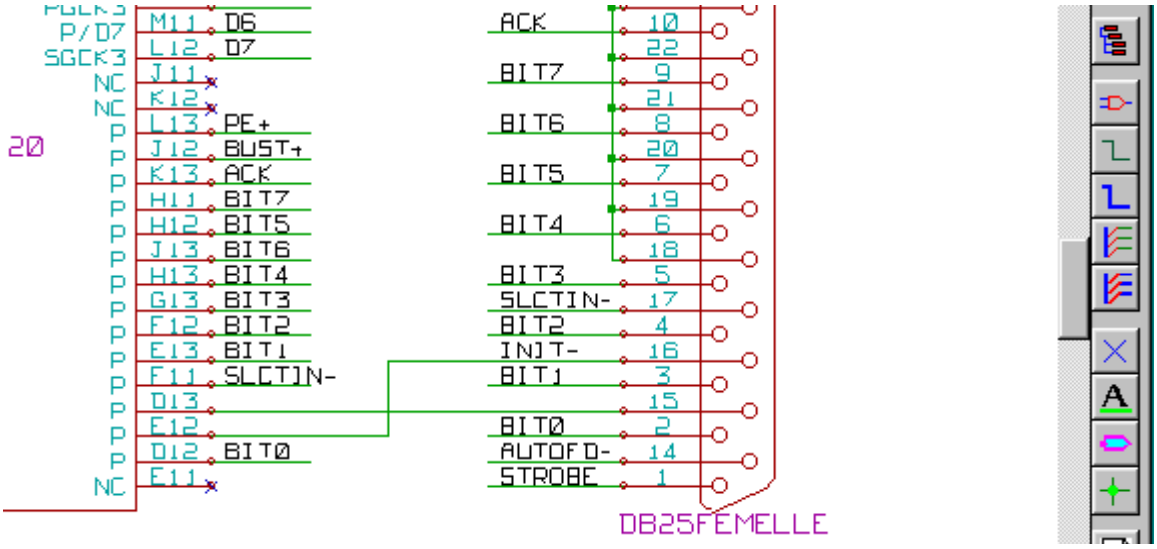
- ワイヤの配置 通常の接続配線．
- バスの配置 回路図を見易くするために，バス配線をまとめラベルを用いて接続．
- ラインかポリゴンを配置．図面を見易くするために用いる．
- ジャンクション（接続点）の配置 ワイヤやバスの交差点で，それぞれの接続を指示．
- バスエントリにワイヤを配置 回路図を見易くするために，バス配線から 1 信号分のワイヤを引き出す．
- ネット名の配置（ローカルラベル） 通常の配線同士の接続に用いる．
- グローバルラベルの配置．シート間での配線接続に用いる．
- 図形テキスト（コメント）を配置 コメント用に用いるテキスト．
- “空き端子フラグを配置” 何も接続しない空き端子やピンに設定するためのシンボル．
- 階層シートの作成 階層シートを作成し，階層間を接続．

接続 （ワイヤとラベル）

接続を確立する方法は 2 つあります．

- ピン間のワイヤ
- ラベル

以下の図はこれら２種類の接続方法を示します．



Note 1:
ラベルが示す点は，ラベル一文字目の左下になります．

この点は，ワイヤに接しているか，あるいはピンの接続位置でなければなりません．

Note 2:
接続を確立するために，ワイヤの端を他のセグメントかピンへ接続します．

もしも配線やピンが重なりあった場合（ワイヤがピンを乗り越えた場合も含みます，但しピンの接続点の場合を除きます），これらは接続されません．しかしラベルの場合は，ラベルが指し示す位置がワイヤのどの位置であってもそのワイヤに接続していることになります．

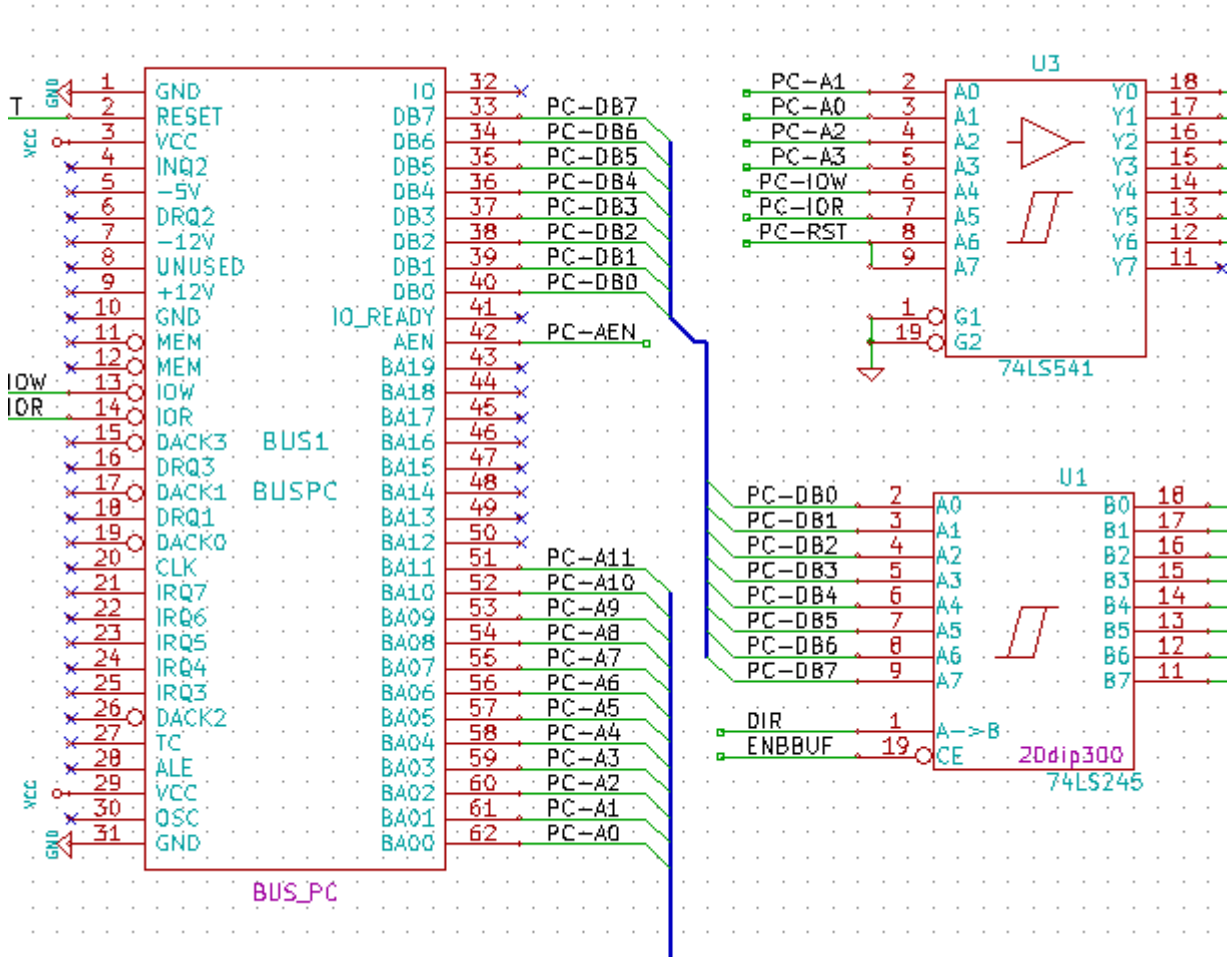
Note 3:
ワイヤ同士を，ワイヤの端以外の場所で接続する必要がある場合は，ワイヤの交差点にジャンクション（接続点）シンボルの配置が必要になります．

以前示した図（ワイヤが DB25FEMALE の 22，21，20，19 ピンに接続されているもの）では，このジャンクション（接続点）シンボルを利用しています．

Note 4:
１つのワイヤに２つの異なるラベルが配置されている場合，これらラベルで示される両方の信号同士が接続されます．どちらか一方のラベルで接続されている他の全ての信号が，このワイヤに接続されるのです．

接続（バス）

以下に示す回路図を見てください．



多くのピン（特にコンポーネント U1，BUS1）がバスへ接続しています．

バスのメンバ

回路図を見易くするために信号の集合体としてバスを利用し，接頭語＋数値というフォーマットで名前を付けています．これはマイクロプロセッサーのバ ス以外にも利用できます．バスに含まれるそれぞれの信号は，バスのメンバとなります．上の図では，PCA0，PCA1，PCA2 は PCA バスのメンバとな ります．

バスそのものを指す場合は，PCA[N..m]のように呼びます．この場合，N と m はバスの最初と最後のワイヤ番号になります．たとえば， PCA バスに 0 から 19 までの 20 本のメンバがある場合，バスの呼び名は PCA[0..19]となります．しかしながら，PCA0，PCA1，PCA2， WRITE，READ のような信号の集合の場合は，バスへ含めることができません．

バスのメンバ同士の接続

ピン同士をバスの同一メンバを用いて接続する場合は，ラベルによって接続しなければなりません．バスは信号の集合体であるため，ピンにバスを直接接続してはいけません．Eeschema はこのような接続を無視します．

上に示したような例では，ピンに接続しているワイヤへ配置されたラベルによってピン間が接続されます．バスワイヤへのバスエントリ部（45 度曲がっているワイヤ部分）を経た接続は Eeschema の回路図としての意味は無く，外観上の見易さを目的としたものとなります．

もしコンポーネントのピン番号が昇順で並んでいるのであれば（メモリやマイクロプロセッサなどで良く見かけます），以下の手順のように繰り返しコマンド（Insert キー）を用いることで非常に速く接続を行う事ができます．：

- 最初のラベルを配置します（例えば PCA0）．
- 繰り返しコマンドを使用してメンバのラベルを配置します．Eeschema は理論的に次のピン位置に相当する縦方向に整列した次のラベルを自動的に生成します（PCA1，PCA2，…）．

- 最初のラベルの下にワイヤを配置します．同様に繰り返しコマンドを利用し，ラベルの下へワイヤを設置していきます．
- 必要に応じて，同じ方法を用いて（最初のエントリを配置し，繰り返しコマンドを使用する）バスエントリを配置して下さい．

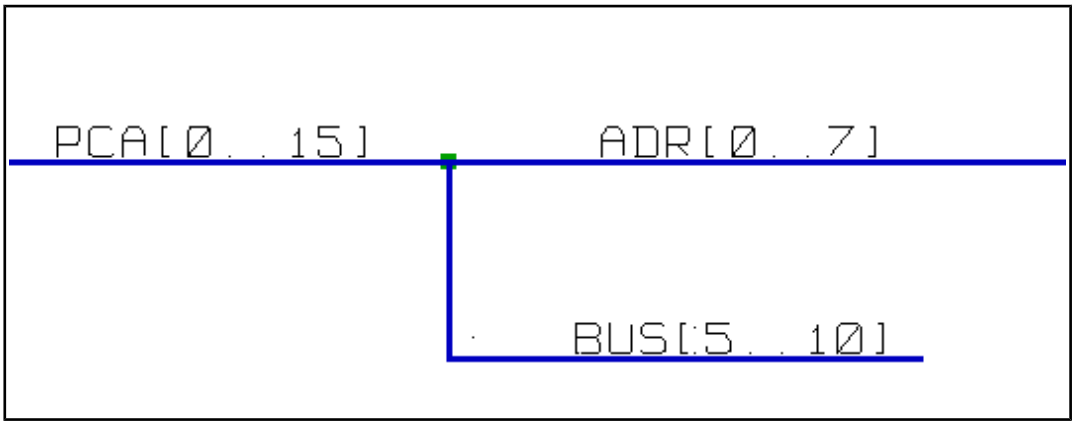
Note:

メニュー"設定"→"オプション"より，繰り返しに関するパラメータを設定することができます：

- アイテムを水平方向にリピート（横方向の間隔）
- アイテムを垂直方向にリピート（縦方向の間隔）
- ラベルのカウントアップリピート（2,3,…のようなインクリメント（加算））

バスのシート間接続

階層構造になっている回路図において，シート間で異なる名前のバス同士を接続する必要がある場合があります．この接続の手順を以下に示します．



バス PCA[0..15]，ADR[0..7]，BUS[5..10]は互いに接続されています．（接続点に注意してください．縦方向のバスが横方向のバスの中央で接続されています．）より正確に言うならば，バスを構成する対応するメンバ同士が接続されます．例えば，PCA0 と ADR0 が接続されています．（同様に，PCA1 と ADR1，・・・，PCA7 と ADR7）さらに，PCA5 と BUS5 と ADR5 も接続されてます（PCA6 と BUS6 と ADR6，PCA7 と BUS7 と ADR7 も同様です）．PCA8 と BUS8 も接続されることとなります（PCA9 と BUS9，PCA10 と BUS10 も同様です）．

一方，この方法では異なる数値を持つメンバ同士は接続されません．異なるバスを，異なる数値同士のメンバを接続したい場合は，接続したいバスのメンバにラベルを設定し，同じワイヤ上にそれら 2 つのラベルを置きます．

電源ポートの接続

コンポーネントに電源ピンがある場合も，他の信号と同様に接続する必要があります．

通常時に電源ピンが見えていないコンポーネント（例えばゲートやフリップフロップ）は少し厄介です．難点は以下の 2 つです：

- 電源ピンが非表示であるため，ワイヤを接続できない．
- 電源ピンの名前が解らない．

これら電源ピンの設定を可視に変更し接続することはあまり良い方法とは言えません．これをしてしまうと，回路図が読みにくくなってしまい，また普通の回路図の慣習に反することになってしまいます．

注:

これらの非表示となっている電源ピンを表示させたい場合は，メインメニューから"設定"→"オプション"をたどるか，左側ツールバー（オプションツールバー）のアイコン  をクリックし，"非表示ピンの表示"オプションをチェックします．

Eeschema は，これら非表示の電源ピンを自動的に接続します：

同じ名前の全ての非表示の電源ピンは，通知無しで自動的に接続されます．

しかし，これらの自動接続も少し手で補う必要があります：

- 他の可視状態の同名ピンへの接続によって，電源ポートへ接続されます．

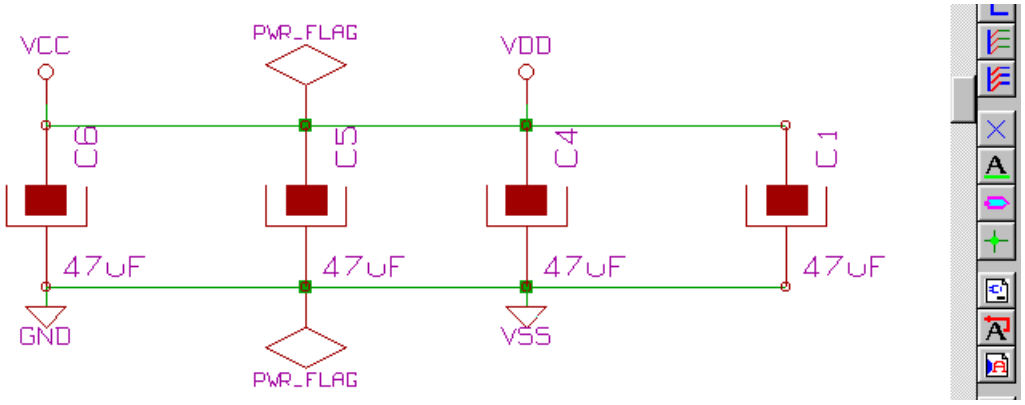
- 異なる名前の非表示電源ピンは，別々のグループとなり相互での接続はされません（例えば，TTL デバイスの"GND"ピンと，MOS デバイスの"VSS"ピンは同じ GND に接続する必要がある場合でも，互いに接続されません）

これらを接続するために，電源ポートシンボルを使います（このためのコンポーネントが準備されており，ライブラリエディタを使用することで新たなシンボルの作成や編集を行う事ができます）．

これらシンボルは，回路図にふさわしい記号と，非表示の電源ピンで構成されています．

ラベルは局所的な接続機能しか無いため，ラベルを利用して電源ピンを接続しないでください．またラベルは非表示の電源ピンは接続できません．（詳細は"hierarchy concepts"の章を参照してください．）

以下の図は，電源ポートの接続例です．



この例では，GND と VSS，VCC と VDD が接続されています．
2 つの PWR_FLGA シンボルがあります．これは，2 つの電源ポート VCC と GND が本当に電源に接続されているかを示すものです．これら 2 つのフラグをつけない場合，ERC は「Warning: power port not powered」という診断結果を出力します．
これらすべてのシンボルは，コンポーネントとして"power"ライブラリに収められています．

空き端子シンボル

この空き端子用シンボルは，ERC チェック時に不要な警告を出さないようにするために利用されるものです．ERC を利用する際に，接続忘れの空きピンの発見を確実にすることができます．

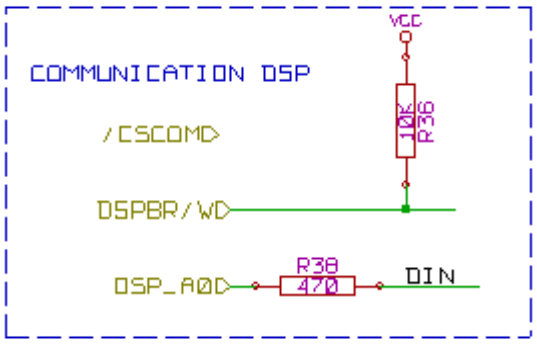
この空き端子シンボル（ツール  ） を配置することで，そのピンを無接続の空き端子とすることができます．しかし，このシンボルは生成するネットリストに影響は与えません．

5.6 - 回路図作成に関する補足

テキストコメント

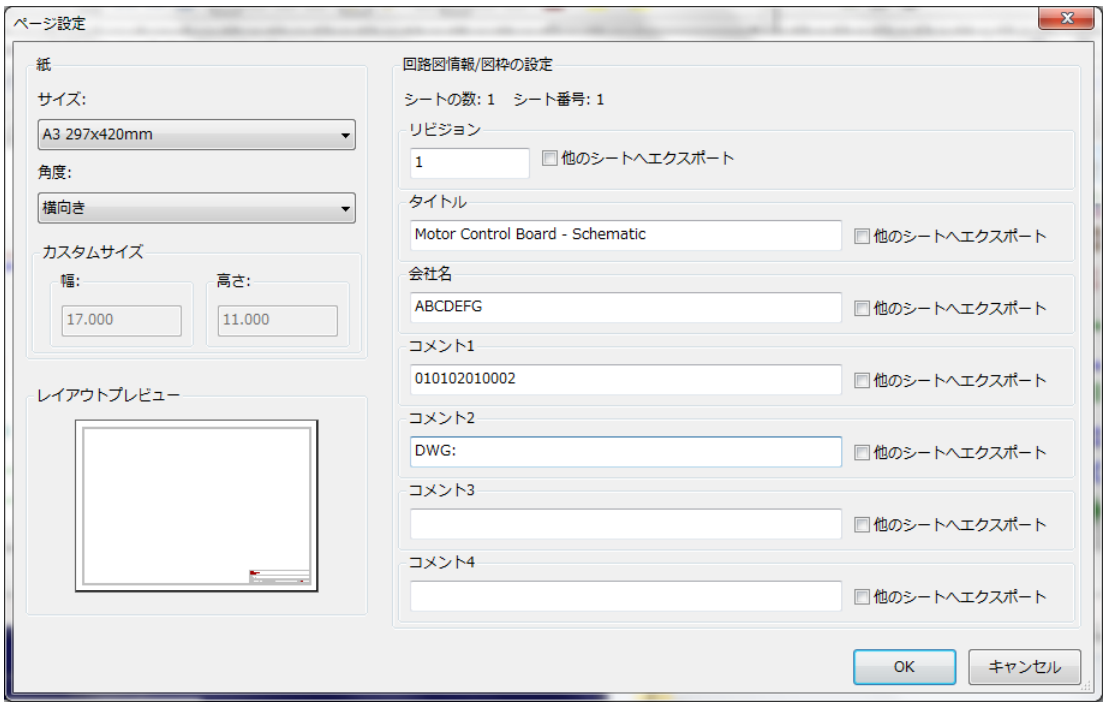
テキストや図枠（フレーム）を配置することで，回路図が見易くなります．図形テキスト（コメント）ツール（  ） と図形ラインかポリゴンを設置ツール（  ） はラベルやワイヤとは違い素子同士の接続を行いません．

図形コメントの例を以下に示します．

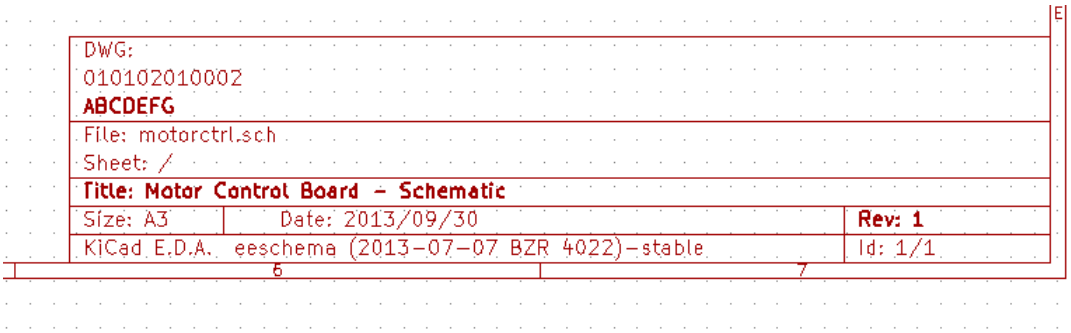


シートの表題欄（タイトルブロック）

表題欄は，  のツールで編集することができます．



完全な表題欄は下記のようになります．



日付やシート番号（Sheet X/Y）は以下のタイミングで自動更新されます．:

- 日付（Date）：そのシートを変更したとき
- シート番号（Sheet Number，階層の時に役立ちます）：アノテーションを実行したとき

6 - 階層回路図

目次

6.1 - はじめに

シート数が２～３枚で済まないようなプロジェクトには，階層的表現を用いるのが一般的によい解決策となります．この種のプロジェクトを管理したい場合，次のことが必要になります：

- 大きなサイズのシートを使用する．その場合，印刷と取り扱いの問題が生じます．
- シートを数枚使用する．これは階層構造に至ります．

この時，完全な回路図は，ルートシートというメインの回路図シートおよび階層を構成するサブシートというものになります．さらに，設計を個別のシートにうまく分割すると可読性が改善されます．

ルートシートからすべてのサブシートを辿ることができなければなりません．Eeschema には，右上のツールバーのアイコン  で使用可能な統合"階層ナビゲーター"があり，階層回路図の管理が非常に簡単です．

階層は２種類あり，これらは共存可能です：１つ目は，すでに開いていて普通に使用するものです．２つ目は，回路図上の従来のコンポーネントのような外観をしたコンポーネントをライブラリ内で作成するというものですが，それは実際にはコンポーネントの内部構造を記述した回路図に対応します．

この２つ目のタイプは集積回路を開発するために使用します．それは，作成中の回路図で機能ライブラリを使用しなければならないからです．

Eeschema は現在この第２のケースには対応していません．

階層は次のようなもです：

- 単一： 任意のシートを一度だけ使用する．
- 複合： 任意のシートを２回以上使用する（複数の実体）．
- 平（Flat）： 単一の階層であるが，シート間の接続は記述されない．

Eeschema はこれら全ての階層を扱うことが可能です．

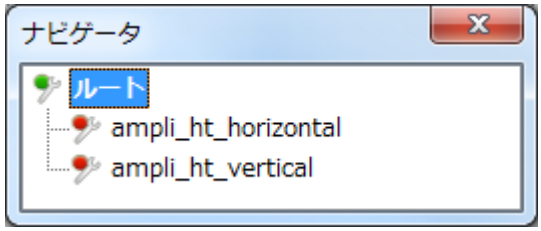
階層回路図の作成は簡単です．階層全体はルート回路図から始まるように管理され，ただ一つの回路図しかないように見えます．

次の２つの重要なステップを理解する必要があります：

- サブシートの作成方法．
- サブシート間の電気的な接続方法．

6.2 - 階層内のナビゲーション

水平ツールバー上の  ボタンでナビゲーターツールが使用できるため，サブシート間のナビゲーションは非常に簡単です．



シート名をクリックすると，そのシートに移動可能になります．すばやく移動するには，シート名を右クリックし，シートに入るを選択します．

右垂直ツールバーの  ツールにより，ルートシートあるいはサブシートに素早く移動可能です．ナビゲーションツールを選択後に以下の操作を行います：

- シート名をクリックしてそのシートに移動する．
- それ以外の場所をクリックしてメインシートに移動する．

6.3 - ローカル，階層およびグローバルラベル

プロパティ

ローカルラベル（  ツール）は，あるシート内のみで信号を接続しています．階層ラベル（  ツール）は，あるシート内のみで信号を接続し，また親シートに配置された階層ピンに接続されています．

グローバルラベル（  ツール）は階層全体に渡って信号を接続しています．非表示の電源ピン（"power in"および"power out"タイプ）は，全階層に渡って互いに接続されているように見えるので，グローバルラベルに似ています．

注

ある階層内で（単一または複合）階層ラベルとグローバルラベルの両方またはそのどちらかを使用可能です．

6.4 - ヘッドラインの階層作成

次のことをする必要があります：

- "シートシンボル"という階層シンボルをルートシート内に配置します．
- ナビゲーターを使用して新規回路図（サブシート）に入り，他の回路図と同様にそれを作成します．
- 新しく作成した回路図（サブシート）にグローバルラベル（HLabels）を配置して２つの回路図間に電気的接続を作成します．また，シートラベル （SheetLabels）という同じ名前を持つラベルをルートシートに配置します．これらのシートラベルはルートシートのシートシンボルや標準的なコン ポーネントピンのような他の回路図要素に接続されます．

6.5 - シートシンボル

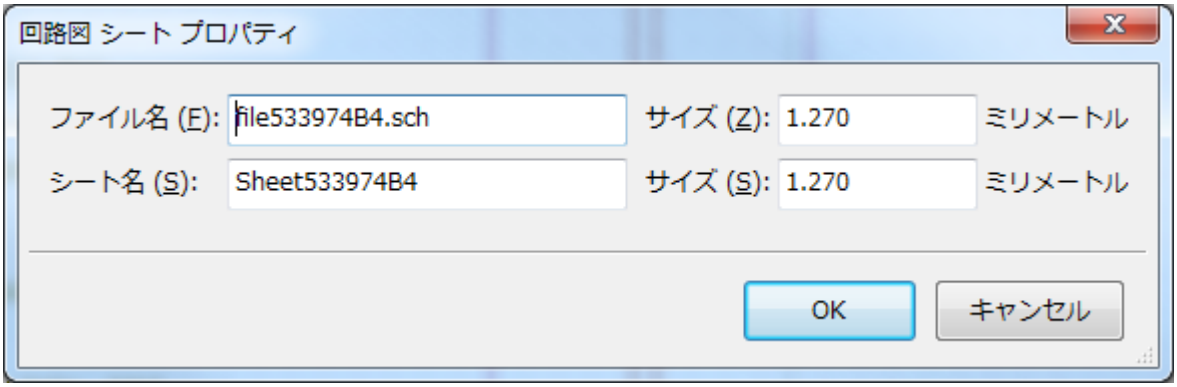
対角上の２点を指定して矩形を作成し，それによりサブシートを表します．

この矩形のサイズは，サブシート内のグローバルラベル（HLabels）に対応した特定のラベルや階層ピンを後で配置可能なものでなければなりません．

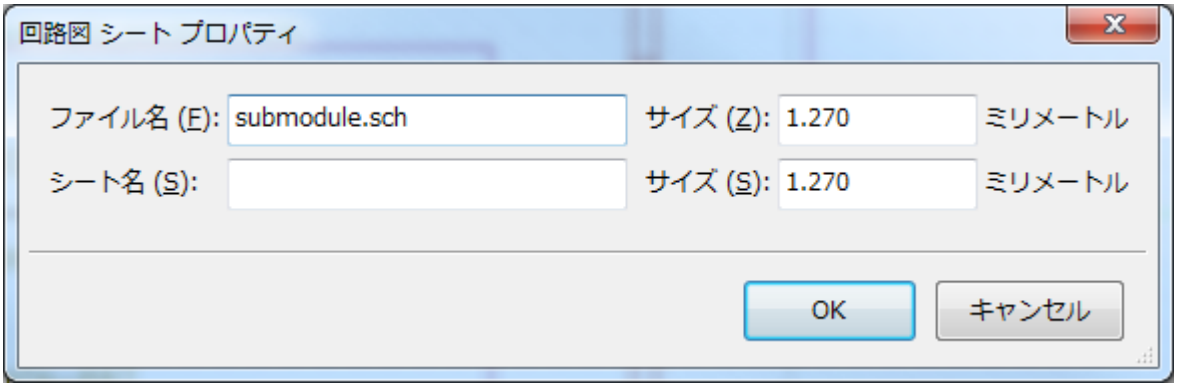
これらのラベルは通常のコンポーネントピンに似ています．  ツールを選択します．

クリックして矩形の左上角を配置します．矩形が十分な大きさとなったら再度クリックして右下角を配置します．

例：



この時，このサブシートのファイル名とシート名の入力が要求されます（階層ナビゲーターを使用し，対応する回路図に移動するために）．



少なくともファイル名の入力が必要です．シート名がない場合，ファイル名がシート名として使用されます（そうするのが普通）．

6.6 - 接続 ― 階層ピン

たった今作成したシンボル用の接続点（階層ピン）をここで作成します．

これらの接続点は通常のコンポーネントピンと似ていますが，ただ１つの接続点で完全なバス接続を行うことが可能です．

それを行うには，次のように２つ方法があります：

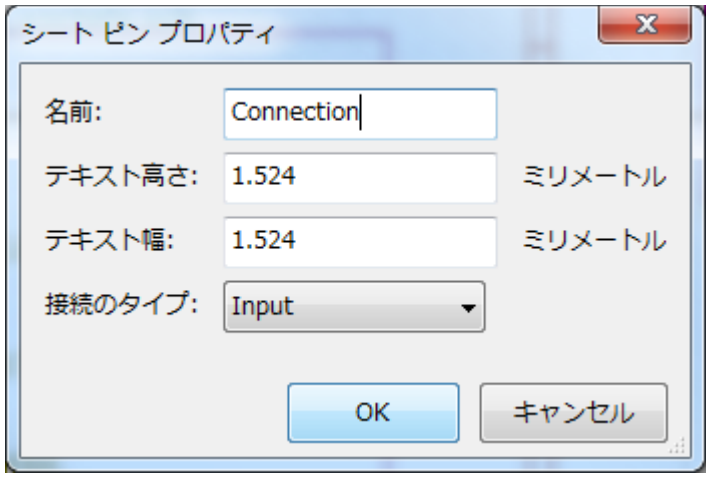
- 必要なピンをサブシート作成前に配置（手動による配置）．
- 必要なピンおよびグローバルラベルをサブシート作成後に配置（半自動配置）．

２つ目の方法が非常に好ましいです．

手動配置:

-  ツールを選択します．
- このピンを配置したい階層シンボルをクリックします．

"CONNECTION"と言う名前の階層ピンを作成する例は以下を参照して下さい．



このピンシート（右クリックしてポップアップメニューの編集を選択します）を編集して，グラフィカルな属性とサイズの定義が可能です．後でそうすることも可能です．

様々なピンシンボルが使用可能です：

- 入力（Input）
- 出力（Output）
- 双方向（BiDir）
- トライステート（Tri State）
- 指定なし（Not Specified）

これらのピンシンボルは単なるグラフィカルな強調で，それ以外の役割はありません．

自動配置:

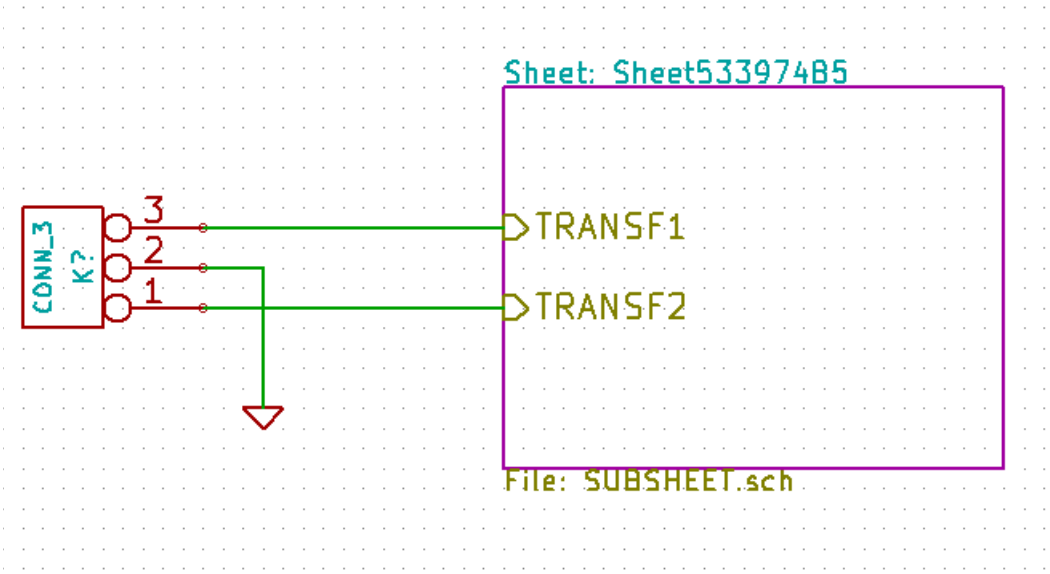
-  ツールを選択します．
- 階層シンボルをクリックし，そこからグローバルラベルに対応するピンをインポートして対応する回路図に配置します．新しいグローバルラベルが存在する場合，つまり，配置済みのピンに対応したものでないなら，階層ピンが現れます．
- このピンを配置したい場所でクリックします．

必要なすべてのピンはエラーなく速やかに配置することが可能です．それらの外観はグローバルラベルと一致しています．

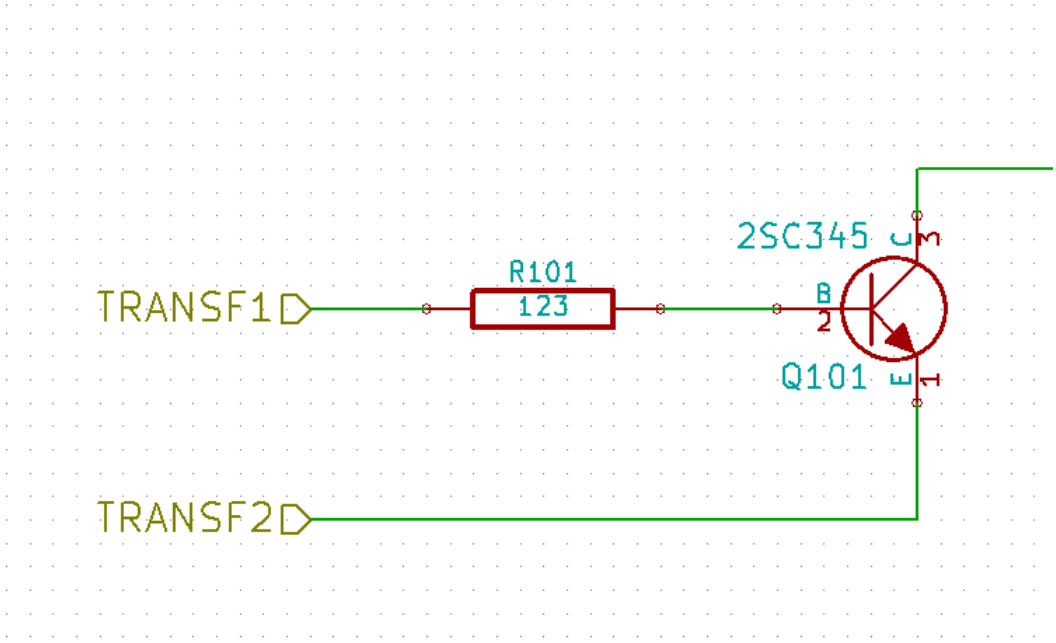
6.7 - 接続 - 階層ラベル

作成したシートシンボルの各ピンはサブシート内の階層ラベルというラベルと一致していなければなりません．階層ラベルはラベルと似ていますが，サブシート およびルートシート間の接続を行います．その２つの相補的なラベル（ピンおよび HLabel）のグラフィカルな表示は似ています．階層ラベルの作成は  ツールで行います．

ルートシートの例は以下を参照して下さい：



ピン TRANSF1 と TRANSF2 がコネクタ JP3 に接続されていることに注意して下さい．
サブシート内でのそれに対応する接続は次のようになります：



２つの階層シート間の接続を成す２つの対応する階層ラベルがあるのがさらにわかります．

注
２つのバスを接続するには，階層ラベルおよび階層ピンを使うことが可能です．この時，既述の構文（Bus [N. .m]）に従います．

ラベル，階層ラベル，グローバルラベルおよび非表示電源ピン

ワイヤによる接続以外に，接続を行う様々な方法について説明します．

単純ラベル

単純ラベルは接続に関してローカルな性質があります．つまり，それが配置されている回路図シートに制限されます．これは次の事実によるためです：

- 各シートにはシート番号が存在する．
- このシート番号はラベルに関連付けられている．

そのため，シート番号 3 にラベル"TOTO"を配置した場合，実際のラベルは"TOTO_3"です．シート番号 1（ルートシート）にラベル"TOTO"を 配置した場合，実際には"TOTO_3"ではなく"TOTO_1"というラベルを配置したことになります．これはシートが 1 つしかない場合でも常にそのよ うになります．

階層ラベル

単純ラベルで言えることは階層ラベルにも当てはまります．

このため，同一シート内で，HLabel の"TOTO"はローカルラベル"TOTO"に接続されていると見なされますが，別のシートの HLabel あるいは"TOTO"というラベルには接続されません．

しかし，HLabel はルートシートに配置された階層シンボル内の対応するシートラベルシンボルに接続されていると見なされます．

非表示電源ピン

非表示の電源ピンは，同一名であるならそれらが互いに接続されていました．このため，"Invisible Power Pin"として宣言されている VCC という名前の全ての電源ピンは，それが置かれているどのシートでもそれらが互いに接続され同電位の VCC を形成しま す．

このことは，あるサブシートに VCC ラベルを配置した場合，そのラベルが VCC ピンには接続されないということを意味します．それは，このラベルが実際には VCC_n であるからです．ここで n とはシート番号です．

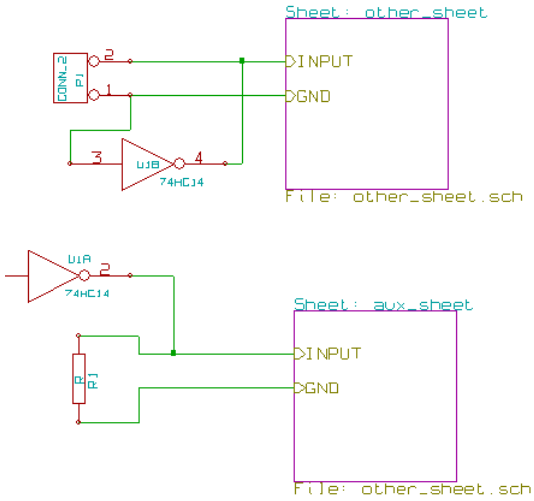
この VCC ラベルを同電位の VCC に実際に接続したいなら，VCC 電源ポートにより非表示電源ピンにそれを明示的に接続する必要があります．

グローバルラベル

同一名のグローバルラベルは階層全体に渡って互いに接続されています．
（vcc ... のような電源ラベルはグローバルラベルです）

6.8 - 複合階層

一例を示します．同じ回路図が 2 回使用されています（2 つの実体）． 2 つのシートのファイル名が同じなので（"other_sheet.sch"）， 2 つのシートは同じ回路図を共有します．しかし，シート名は異なっていなければなりません．



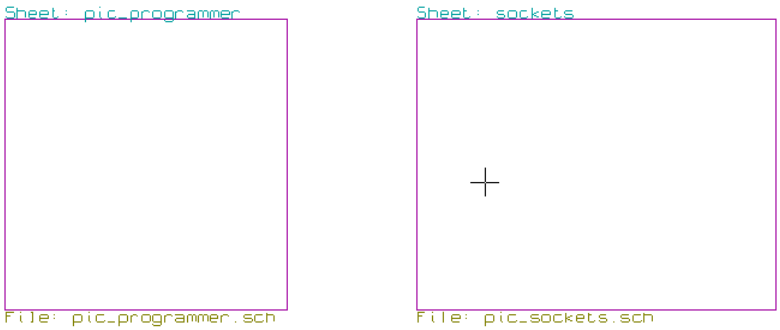
6.9 - 平階層

シート間の接続を作らずに（平（ヒラ）階層），シートを多数使うプロジェクトの作成が可能です．それには次のルールを順守して下さい：

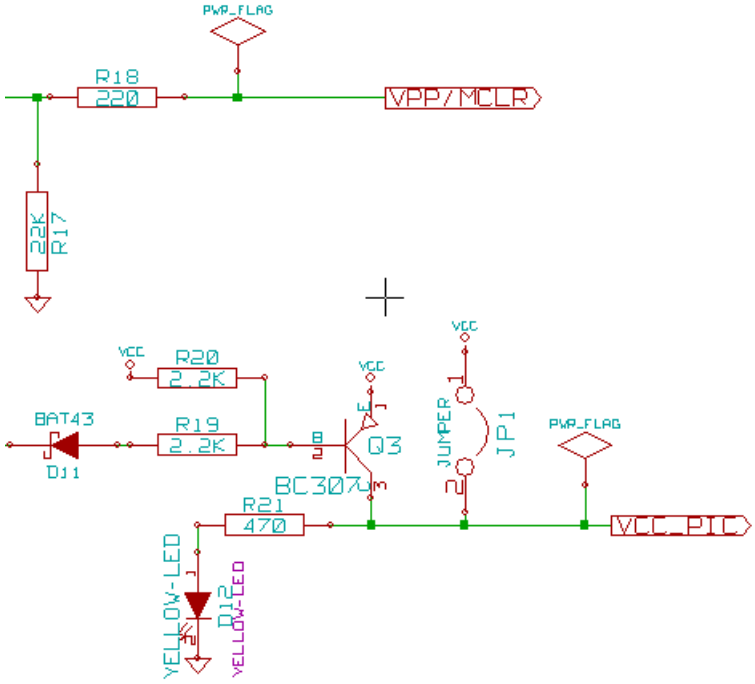
- ルートシートを作成し，他のすべてのシートをそれに含めます．ルートシートはシート間のリンクとして機能します．

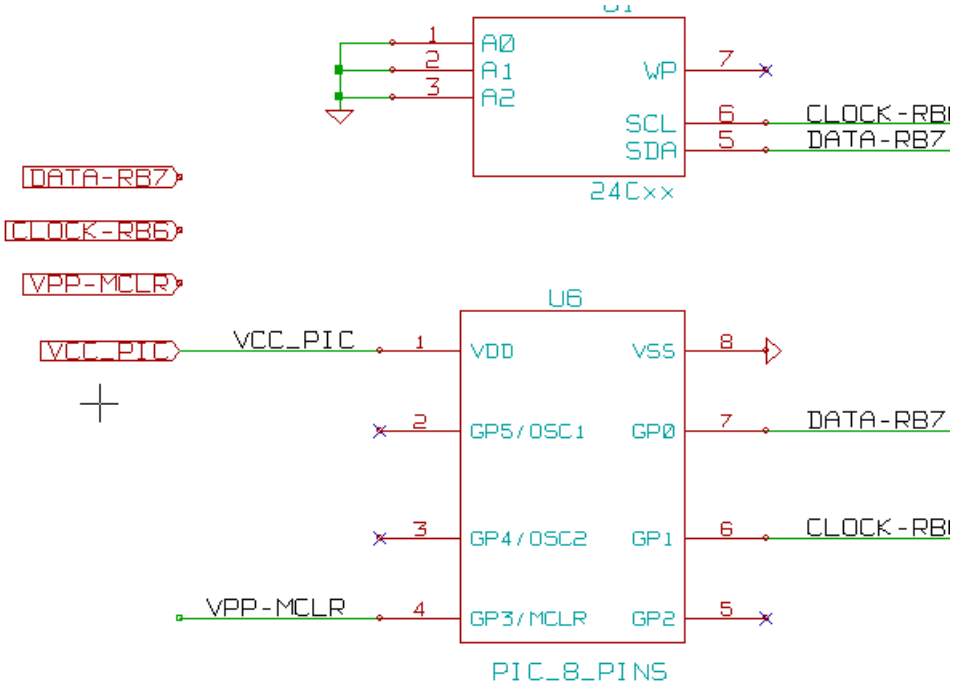
- 明示的な接続はまったく必要ありません．
- シート間のすべての接続には，階層ラベルではなくグローバルラベルを使用します．

ルートシートの例を以下に示します．



２ページあり，それらはグローバルラベルで接続されています．





グローバルラベルを参照	DATA-RB7 CLOCK-RB6 VPP-MCLR

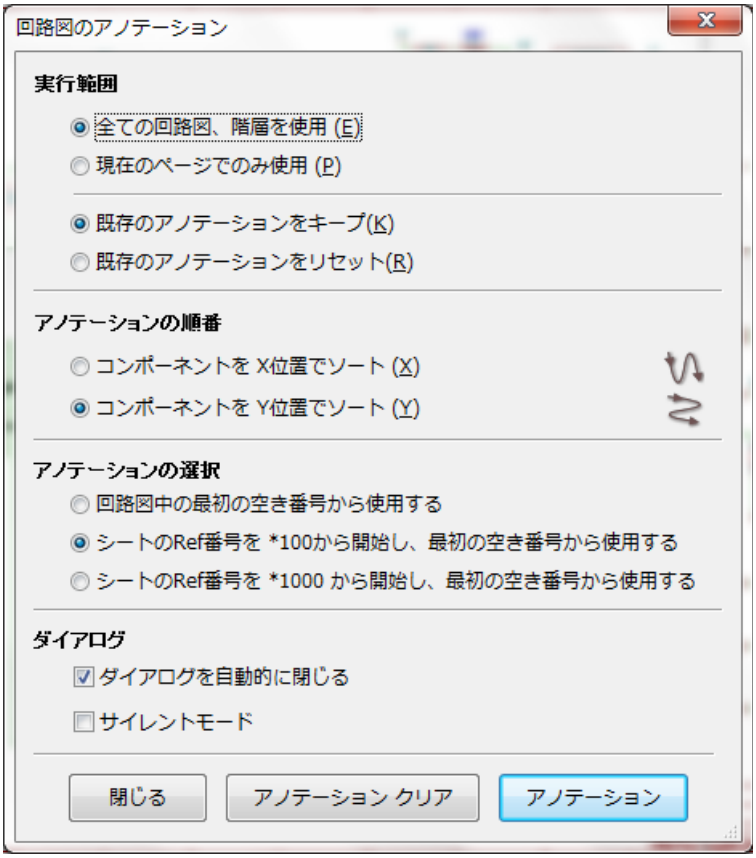
7 - アノテーションツール

目次

7.1 - はじめに

アノテーションツールを用いることで，回路図中のコンポーネントに自動的に参照番号（Ref 番号）を割り当てることができます．複数部品からなるコンポーネントについては，使用パッケージ数が最小になるように部品番号を割り当てます．アノテーションツールは， 

アイコンをクリックすることで利用できます． 以下にアノテーションツールのメインウインドウを示します．



次に示すように，さまざまな利用方法があります．:

- 全てのコンポーネントをアノテート （「既存のアノテーションをリセット（R）」を選択）．
- 新しく追加したコンポーネントのみをアノテート（「既存のアノテーションをキープ（k）」を選択）．
- 全階層をアノテート （「全ての回路図，階層を使用（e）」を選択）．
- 現在のシートのみをアノテート （「現在のページのみ使用（p）」を選択）．

「アノテーションの順番」オプションでは，それぞれのシート内での番号の振り方を指定することができます．

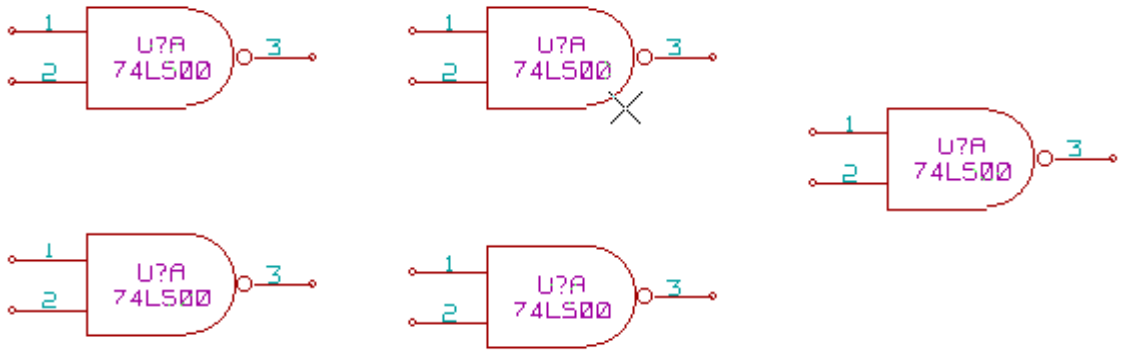
特別な場合を除いて，以前のアノテーション結果を修正しない場合は，プロジェクト全体（全てのシート）と新しいコンポーネントが自動アノテーションの対象となります．

「アノテーションの選択」オプションでは，参照番号の計算方法を指定します．:

- 回路図中の最初の空き番号から使用する：コンポーネントは（各部品記号につき）1 から参照番号が振られます．前回のアノテーションをキープする場合は，使われていない番号から利用されます．
- シートの Ref 番号を*100 から開始し，最初の空き番号から使用する：シート 1 では 101 から，シート 2 では 201 から参照番号が振られます．それぞれ の部品記号（U や R）が 1 シート内で 99 を超えてしまった場合は，継続して 200 以降の参照番号が利用され，シート 2 では 200 番台の最初の空き数字から 参照番号が振られます．
- シートの Ref 番号を*1000 から開始し，最初の空き番号から使用する：シート 1 では 1001 から参照番号が振られ，シート 2 では 2001 から参照番号が振られていきます．

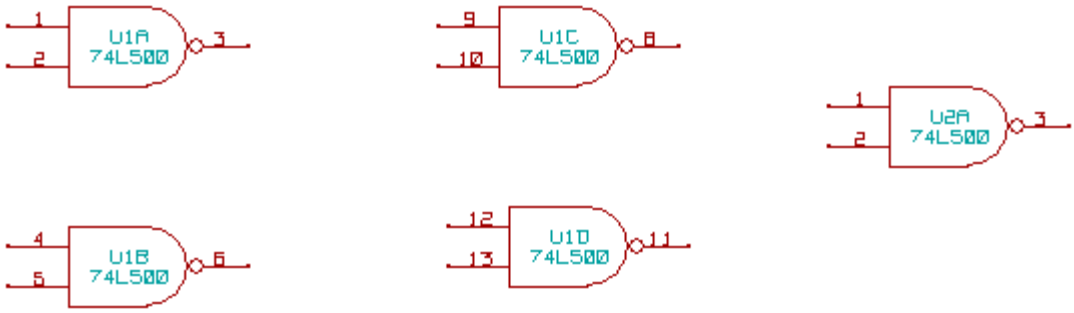
7.2 - 例

アノテーション順序

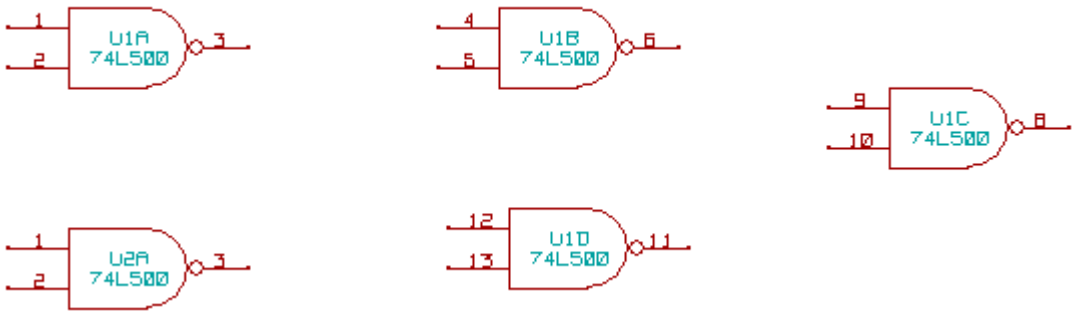


部品配置後，未だ参照番号が振られていない 5 つの素子を例とします．

アノテーションを実行すると，以下のような結果が得られます．



コンポーネントを X 位置でソートした場合：

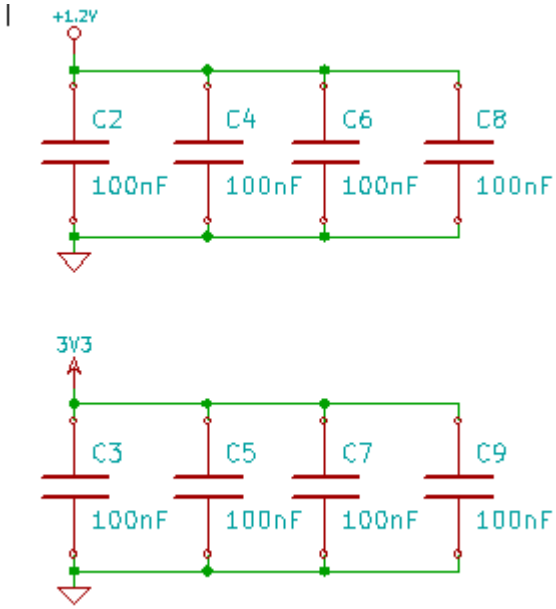


コンポーネントを Y 位置でソートした場合：

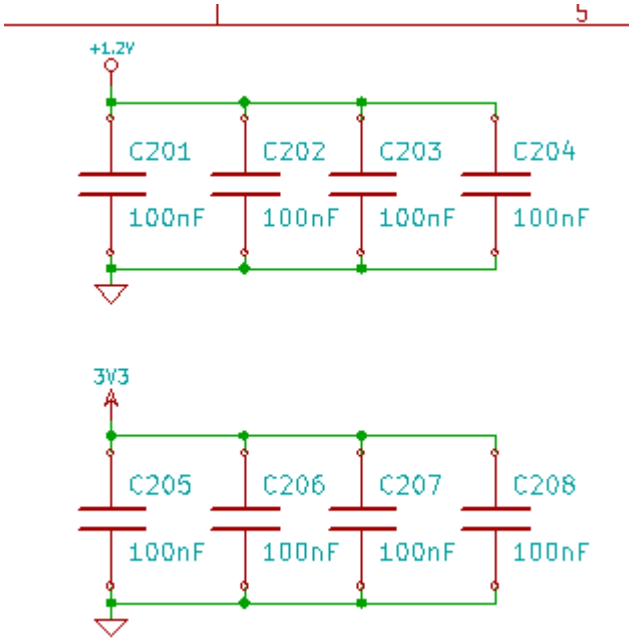
これらアノテーションにより，74LS00 の 4 つのゲートが U1 パッケージにまとめられ，5 番目のゲートが次の U2 へ分類されました．

アノテーションの選択

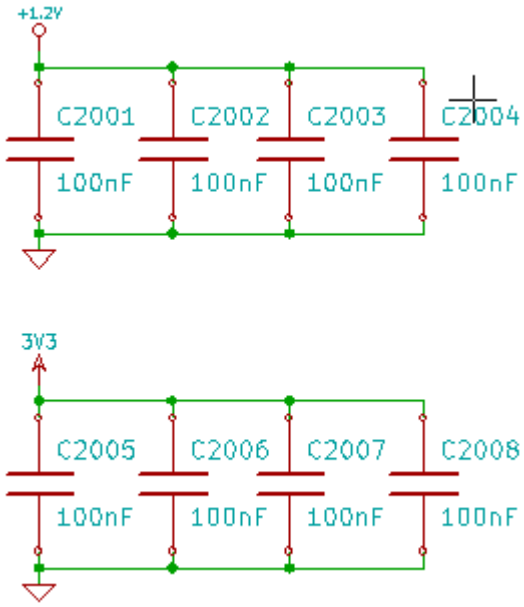
下記の図は，部品をシート 2 に配置し，「回路図中の最初の空き番号から使用する」オプションを利用してアノテーションを行ったものです．



「シートの Ref 番号を*100 から開始し，最初の空き番号から使用する」オプションを利用しアノテーションを行うと，下記のようになります．



「シートの Ref 番号を*1000 から開始し，最初の空き番号から使用する」オプションを利用した場合は，下記のようになります．



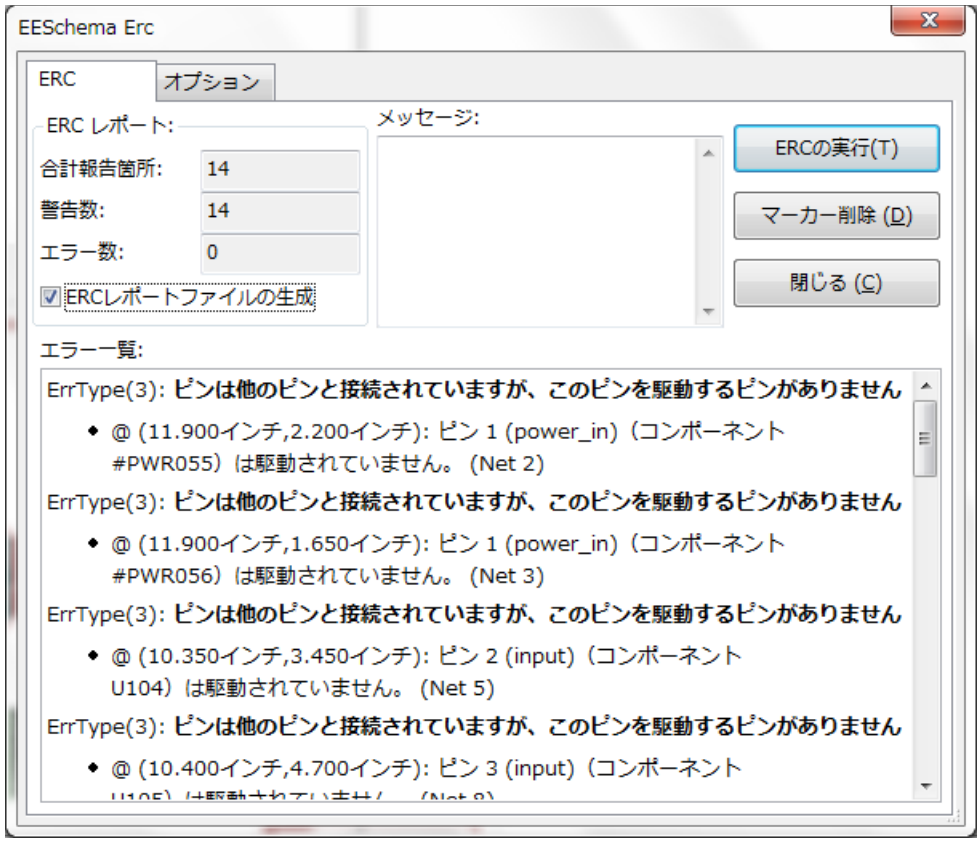
8 - ERC（電気的ルールチェック）による設計検証

目次

8.1 - はじめに

ERC（電気的ルールチェック）ツールは回路図の自動チェックを実行します．ERC は，未接続ピン，未接続の階層シンボル，出力のショートなどのよ うなシート内のすべてのエラーをチェックします．当然ながら，自動チェックは絶対確実なものではありませんし，設計エラーを検出可能なソフトウェアは100%完全ではありません．そのようなチェックは多くの見落としや小さな間違いを検出可能なので非常に便利です．

検出された全てのエラーをチェックし，先に進む前に正常な状態になるよう修正する必要があります．ERC の質はライブラリ作成中に，ピンの電気的なプロパティをどれだけ細かく指定したかに直接関係します．ERC の出力は"エラー"または"警告"として報告されます．



8.2 - ERC の使用法

アイコン  をクリックすると ERC を開始します．

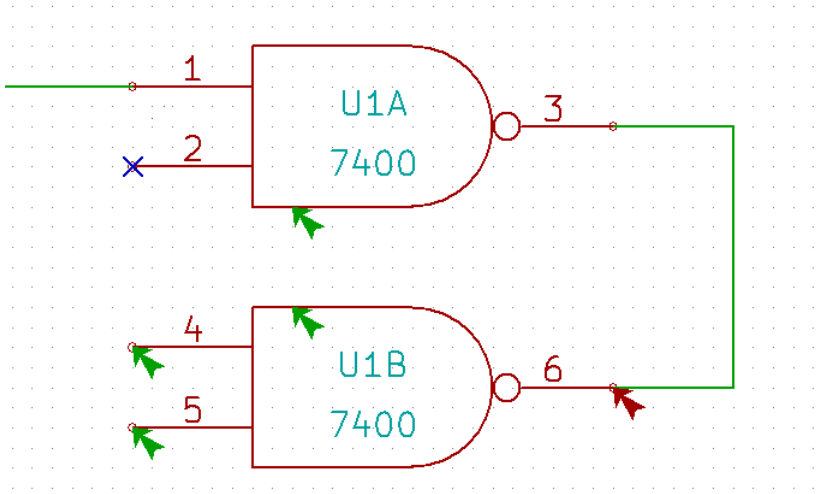
警告は，ERC エラー（ピンまたはラベル）を出力しながら回路図要素上に配置されます．

注:

- このダイアログウィンドウ内でエラーメッセージをクリックすると，回路図内のそれに対応するマーカーに移動することができます．
- 回路図中に表示されたマーカーを右クリックすることで，診断結果のメッセージへアクセスすることができます．

回路図内でマーカーを右クリックしてそれに対応する診断メッセージにアクセスします．

8.3 - ERC の例

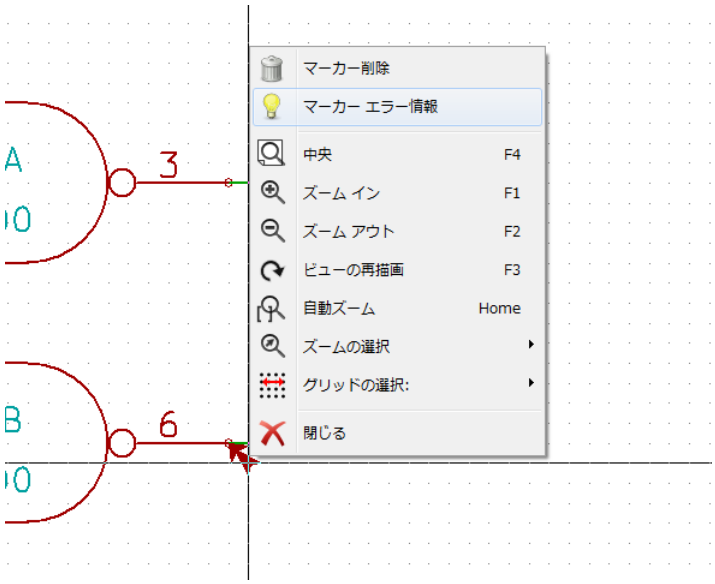


エラーが 5 つ見られます：

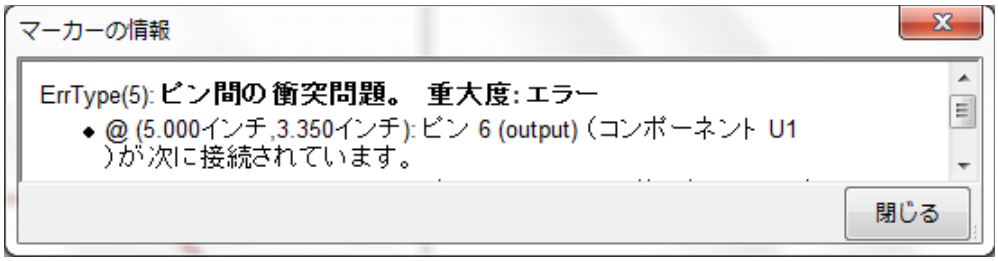
- ２本の出力が誤接続されています（赤の矢印）.
- 入力が２本未接続のままです（緑の矢印）.
- 非表示電源ポートのエラーで，電源フラグがありません（上部に緑の矢印）.

8.4 - 診断結果の表示

マーカーを右クリックして，ポップアップメニューで ERC マーカー診断（diagnostic）ウィンドウが使用可能になります.



そこで，マーカーエラー情報をクリックするとエラーの内容が表示されます.



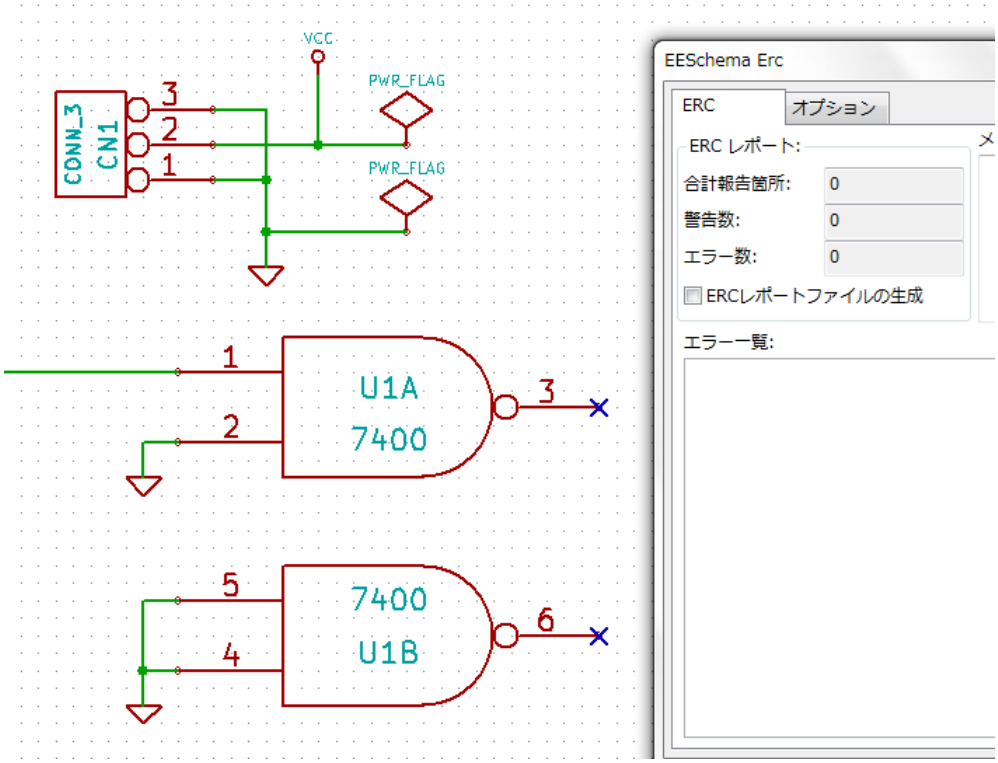
8.5 - 電源および電源フラグ

電源ピンにエラーまたは警告を出すのは一般的です．たとえ，すべて正常のように思われるとしてもで

す．上の例を参照して下さい．大抵の設計では電源はコネクタによって供給されますが，そのコネクタは（電源出力として宣言されているレギュレータ出力のような）電源ではありません．

このため ERC は，電源出力ピンを検出してこの配線を操作するということはせず，電源で駆動されていないものと判断します．

この警告を避けるには，そのような電源ポートに"PWR_FLAG"を配置しなければなりません．次の例を参照して下さい．

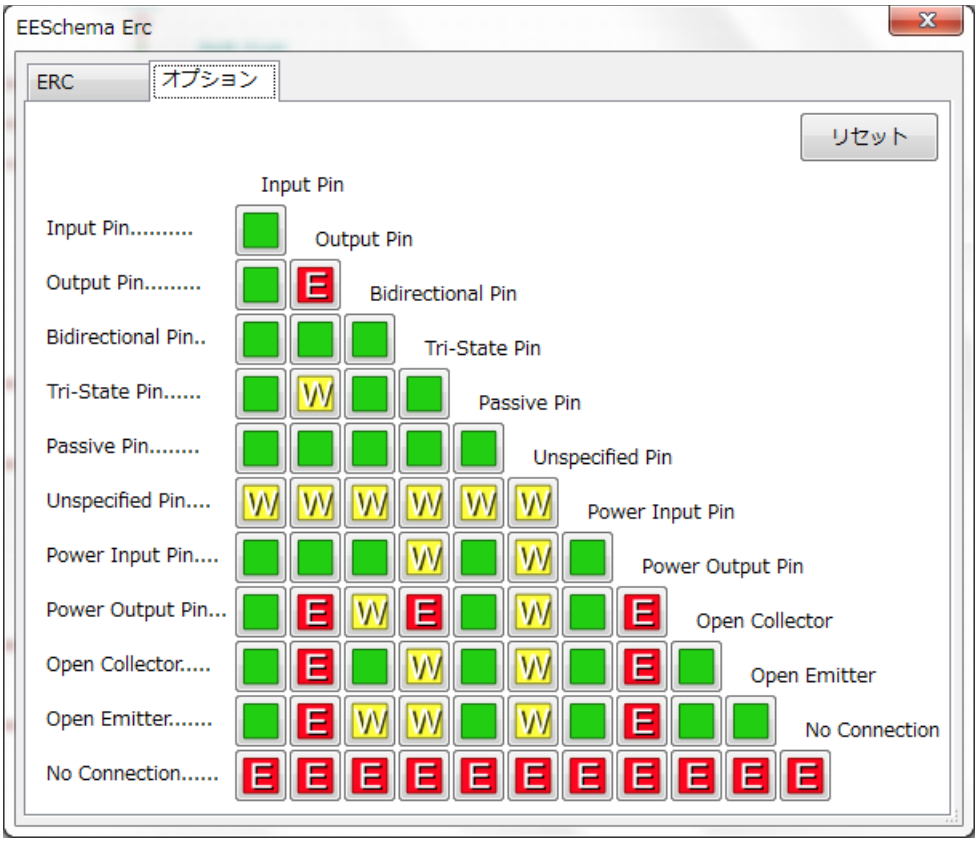


このようにすると，エラーマーカーが消えます．

大抵の場合，PWR_FLAG は GND に接続されていなければなりません．それは普通，レギュレータは電源出力として宣言された出力を持ちますが，グラウ ンドピンは電源出力ではなく（通常の属性は電源入力），その結果グラウンドは PWR_FLAG がなければ電源に接続されたことにはならないからです．

8.6 - ルールの設定

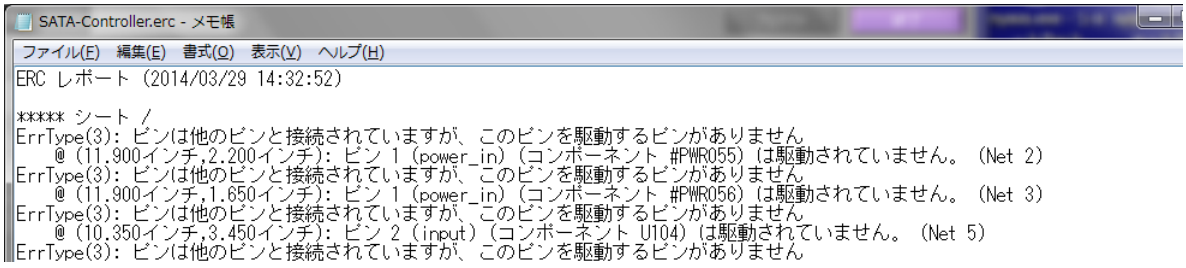
オプションパネルで接続ルールを設定し，エラーおよび警告チェックのための電気的条件を定義します．



マトリクス内の必要な矩形をクリックすると，ノーマル，警告，エラーの選択がサイクリックに切り替わります．それによりルールの変更が可能です．

8.7 - ERC レポートファイル

オプションの ERC レポートの作成にチェックを付けると，ERC レポートファイルの生成と保存が可能です．ERC レポートのファイル拡張子は，.erc です．ERC レポートファイルの例を示します．



9 - ネットリストの作成

目次

9.1 - 概要

ネットリストはコンポーネント間の接続を記述したファイルです．ネットリストのファイルには，次のものが含まれます：

- コンポーネントのリスト．
- 等電位ネットというコンポーネント間の接続のリスト．

さまざまなネットリストのフォーマットが存在します．コンポーネントのリストと等電位リストが２つの別々のファイルであることもあります．回路図入力 （capture）ソフトウェアの使用においては，このネットリストが基本となります．それはネットリストが次のような他の電子系 CAD ソフトウェアとの リンクとなるからです． ：

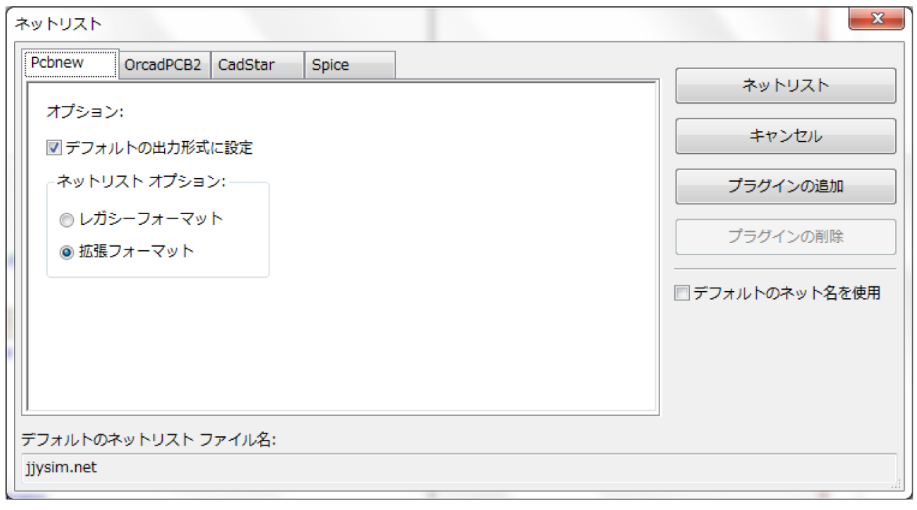
- PCB ソフトウェア．
- 回路および PCB シミュレータ．
- CPLD （および他のプログラマブル IC の）コンパイラ．

Eeschema はネットリストのフォーマットを数種サポートしています．

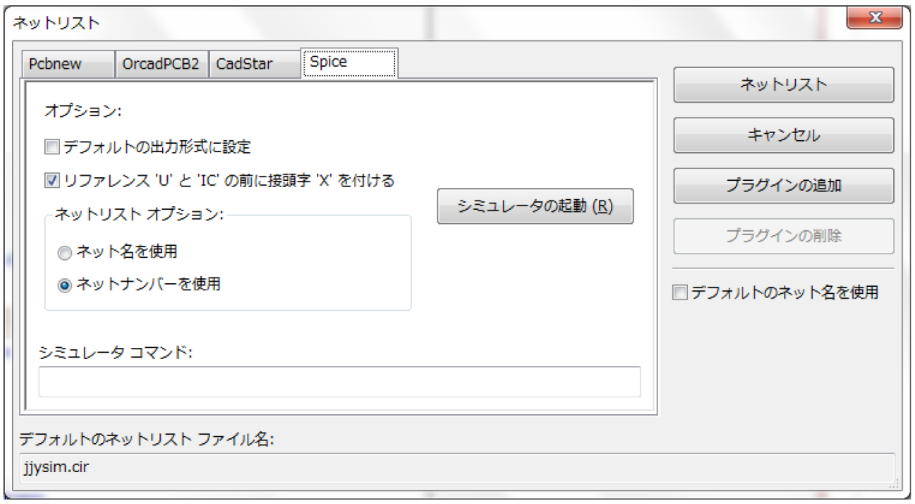
- PCBNEW フォーマット（プリント配線）．
- ORCAD PCB2 フォーマット（プリント配線）．
- CADSTAR フォーマット（プリント配線）．
- 様々なシミュレータ用の Spice フォーマット（Spice フォーマットは他のシミュレータにも使用される）．

9.2 - ネットリストフォーマット

ツールを選択し，ネットリスト作成ダイアログボックスを開きます．



Pcbnew を選択



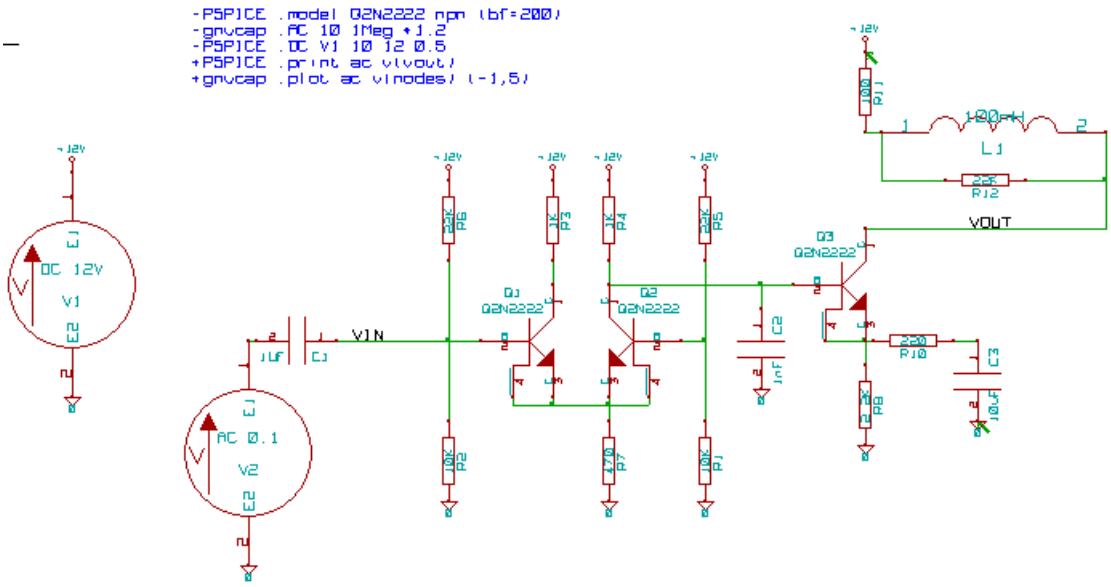
Spice を選択

それぞれのタブで希望するフォーマットを選択できます．Spice フォーマットでは，等電位の名称（その方が読みやすい）か，またはネット番号（Spice の古いバージョンは番号のみ受け付ける）のどちらかでネットリストを生成することが可能です．

注
大きなプロジェクトでは，ネットリストの生成に数分かかることがあります．

9.3 - ネットリストの例

PSPICE ライブラリを使用した回路設計は以下を参照して下さい．



PCBNEW ネットリストファイルの例です．

```
# EESchema Netlist Version 1.0 generee le 21/1/1997-16:51:15
 (
 (32E35B76 $noname C2 1NF {Lib=C}
 (1 0)
 (2 VOUT_1)
)
 (32CFC454 $noname V2 AC_0.1 {Lib=VSOURCE}
 (1 N-000003)
```

```
 (2 0)
)
 (32CFC413 $noname C1 1UF {Lib=C}
 (1 INPUT_1)
 (2 N-000003)
)
 (32CFC337 $noname V1 DC_12V {Lib=VSOURCE}
 (1 +12V)
 (2 0)
)
 (32CFC293 $noname R2 10K {Lib=R}
 (1 INPUT_1)
 (2 0)
)
 (32CFC288 $noname R6 22K {Lib=R}
 (1 +12V)
 (2 INPUT_1)
)
 (32CFC27F $noname R5 22K {Lib=R}
 (1 +12V)
 (2 N-000008)
)
 (32CFC277 $noname R1 10K {Lib=R}
 (1 N-000008)
 (2 0)
)
 (32CFC25A $noname R7 470 {Lib=R}
 (1 EMET_1)
 (2 0)
)
 (32CFC254 $noname R4 1K {Lib=R}
 (1 +12V)
 (2 VOUT_1)
)
 (32CFC24C $noname R3 1K {Lib=R}
 (1 +12V)
 (2 N-000006)
)
 (32CFC230 $noname Q2 Q2N2222 {Lib=NPN}
 (1 VOUT_1)
 (2 N-000008)
 (3 EMET_1)
)
 (32CFC227 $noname Q1 Q2N2222 {Lib=NPN}
 (1 N-000006)
 (2 INPUT_1)
 (3 EMET_1)
)
)
# End
```

PSPICE フォーマットでは，ネットリストは次のようになります.

```
* EESchema Netlist Version 1.1 （Spice format） creation date: 18/6/2008-08:38:03
```

```
.model Q2N2222 npn （bf=200）
.AC 10 1Meg *1.2
.DC V1 10 12 0.5

R12  /VOUT N-000003 22K
R11  +12V N-000003 100
L1  N-000003 /VOUT 100mH
R10  N-000005 N-000004 220
C3  N-000005 0 10uF
C2  N-000009 0 1nF
R8  N-000004 0 2.2K
Q3  /VOUT N-000009 N-000004 N-000004 Q2N2222
V2  N-000008 0 AC 0.1
C1  /VIN N-000008 1UF
V1  +12V 0 DC 12V
R2  /VIN 0 10K
R6  +12V /VIN 22K
R5  +12V N-000012 22K
R1  N-000012 0 10K
R7  N-000007 0 470
R4  +12V N-000009 1K
R3  +12V N-000010 1K
Q2  N-000009 N-000012 N-000007 N-000007 Q2N2222
Q1  N-000010 /VIN N-000007 N-000007 Q2N2222

.print ac v（vout）
.plot ac v（nodes）  （-1,5）

.end
```

9.4 - 注

ネットリスト名の注意事項

ネットリストを使用する多くのソフトウェアツールは，コンポーネント名，ピン名，等電位名（equipotentials）あるいは他の名前に空白（space）の使用を認めません．ラベルあるいはコンポーネントやそのピンの名前と数値欄に空白を使用しないで下さい．

同様に，英数字以外の特殊文字の使用は問題を生じる可能性があります．この制限は Eeschema とは無関係ですが，ネットリストを使用する他のソフトウェアがネットリスト・フォーマットを解釈できなくなる点に関わることに注意して下さい．

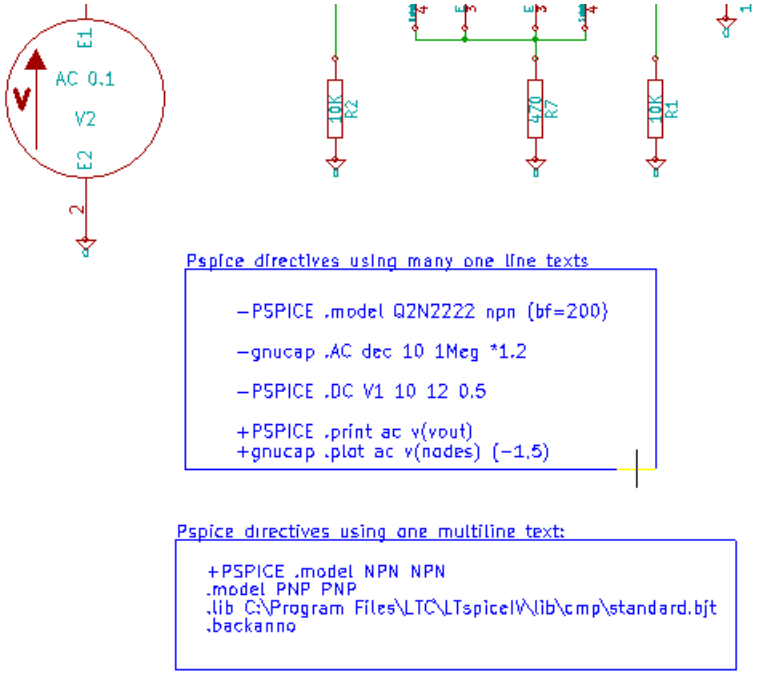
PSPICE ネットリスト

Pspice シミュレーターの場合，ネットリストの中にコマンド行（.PROBE, .AC など）をいくつか含める必要があります．

回路図に含まれる-pspice または-gnucap のキーワードで始まるテキスト行は，ネットリストの先頭に（キーワードがない状態で）挿入されます．

回路図に含まれる+pspice または+gnucap のキーワードで始まるテキスト行は，ネットリストの先頭に（キーワードがない状態で）挿入されます．

１行テキストを複数使用する例，複数行テキストを１つ使用する例です．



例えば：次のようなテキストを入力する場合（ラベルを使用しないこと！）：

-PSPICE .PROBE

.PROBE の行はネットリストに挿入されます.

前述の例ではこの方法でネットリストの先頭に３行，末尾に２行挿入されました.

複数行テキストを使用している場合，+pspice または+gnucap のキーワードは１度だけ必要です：

+PSPICE .model NPN NPN

.model PNP PNP

.lib C:\Program Files\LTC\LTspiceIV\lib\cmp\standard.bjt

.backanno

上の場合，４行生成されます：

.model NPN NPN

.model PNP PNP

.lib C:\Program Files\LTC\LTspiceIV\lib\cmp\standard.bjt

.backanno

また，Pspice の場合，等電位の GND は 0（ゼロ）という名前にしなければならないことに注意して下さい.

9.5 - «プラグイン»を使用する他のフォーマット

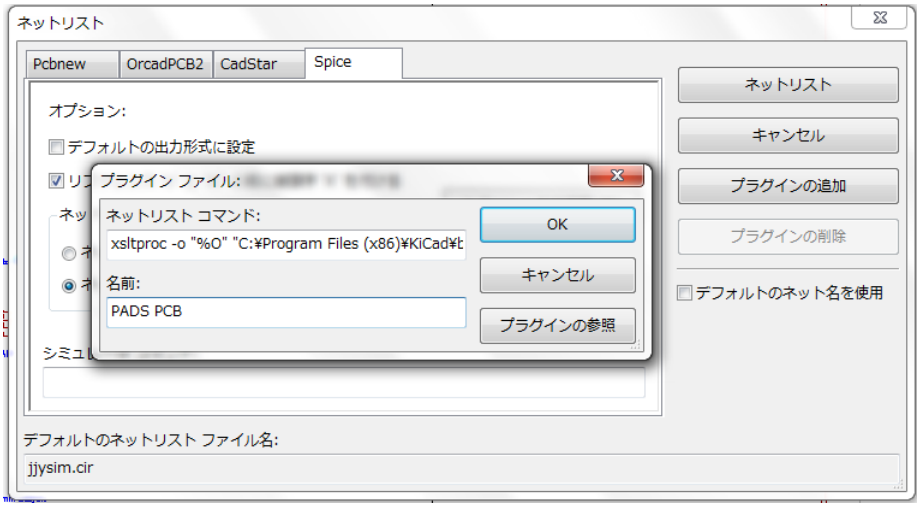
他のネットリスト・フォーマットの場合には，ネットリストコンバーターを追加することが可能です. Eeschema はそれらのコンバーターを自動的に起動します．コンバーターの解説と例は 14 章にあります.

コンバーターはテキストファイル（xsl フォーマット）ですが，Python のような他の言語を使用することが可能です．xsl フォーマットを使用する場 合，ツール（xsltproc.exe あるいは xsltproc）は Eeschema が生成した中間ファイルと，コンバーターファイルを読み込んで，出力 ファイルを生成します. この場合，コンバーターファイル（シートスタイル）は非常に小さく記述が容易です.

ダイアログウィンドウの初期設定

プラグインの追加タブで，新規ネットリスト・プラグインを追加することが可能です．

PadsPcb プラグインのセットアップウィンドウです．



セットアップでは以下が必要です：

- 表題　（例えば：ネットリスト・フォーマットの名前）．
- 起動するプラグイン．

ネットリスト生成時に以下のことを行います：

1. Eeschema は中間ファイル*.tmp を生成します．例えば，test.tmp とします．
2. Eeschema はプラグインを実行し，test.tmp を読み込み，test.net を生成します．

コマンドラインフォーマット

xsltproc.exe を.xsl ファイルの変換ツールとして，ファイル netlist_form_pads-pcb.xsl をコンバーターのシートスタイルとして使用する例です：

```
f:/kicad/bin/xsltproc.exe -o %O.net f:/kicad/bin/plugins/netlist_form_pads-pcb.xsl %I
```

各部の意味は次の通りです：

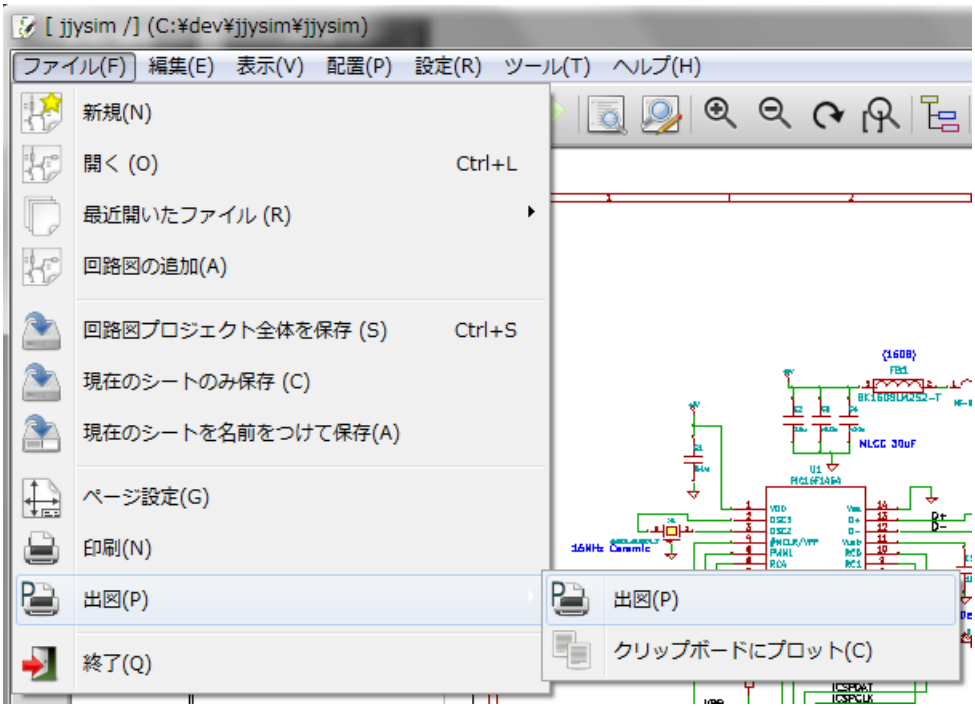
xslproc についてのさらに多くの説明，中間ファイルフォーマットの記述内容，各コンバーターの場合のシートスタイルの例は 14 章を参照して下さい．

10 -　プロットおよび印刷

目次

10.1 - はじめに

ファイルメニューから印刷とプロットの両コマンドの実行が可能です.



サポートしている出力フォーマットは POSTSCRIPT，HPGL，SVG および DXF です．自分のプリンターで直接印刷することも可能です.

10.2 - 共通印刷コマンド

"全てをプロット"は全階層（各シート毎に印刷ファイルを 1 つ生成する）をプロットします.

:ページのプロット： は現在のシートのみ印刷します.

10.3 - HPGL のプロット

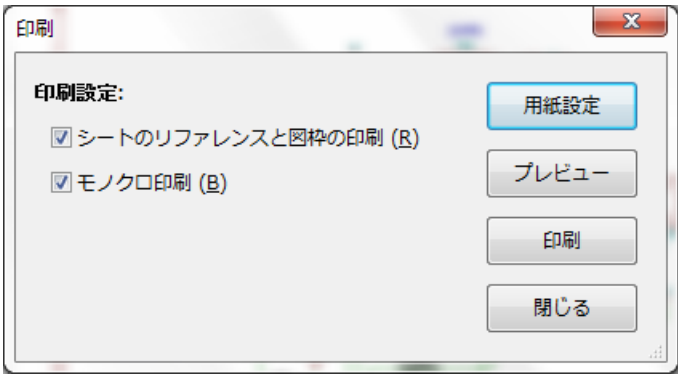
このコマンドにより HPGL ファイルを作成します．このオプションはアイコン 

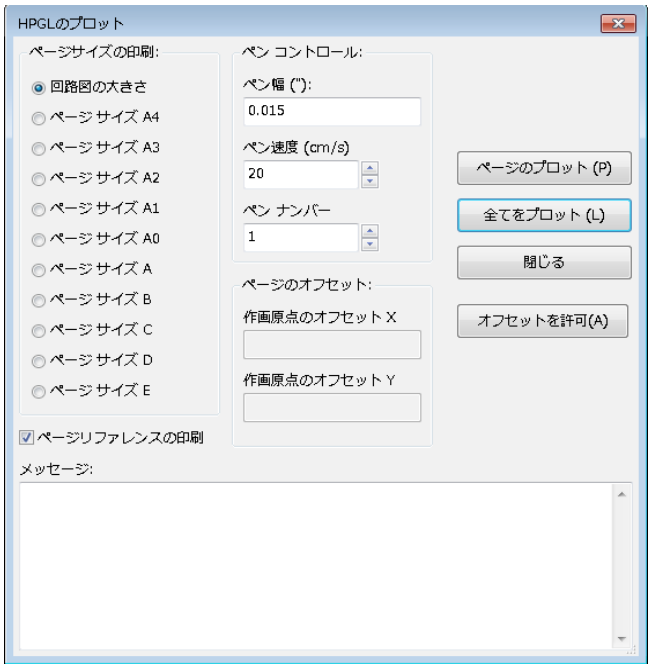
で使用可能です．このフォーマットでは，以下を定義可能です.

- ペンナンバー
- ペン幅（0.001 インチ単位）.
- ペン速度（cm/S）.
- シートサイズ.

- 印刷オフセット.

プロッターのセットアップダイアログ・ウィンドウは次のようなものです.





出力ファイル名はシート名に拡張子.plt を付加したものです.

シートサイズ選択

通常は回路図の大きさにチェックが付きます．この場合，タイトルブロックメニューで定義されているシートサイズが使用され，その時のスケールは 1 になりま す．異なるシートサイズ（A4～A0 あるいは A～E）を選択すると，スケールが自動的に調整されてページにフィットします.

オフセット調整

すべて標準的な寸法である場合，可能な限り正確に描画を中央に配置するようオフセットの調整が可能です．プロッターは原点がシートの中央か左下角にあるので，適切にプロットするために，オフセットを入力（introduce）可能であることが必要です.

一般的に言えることですが,

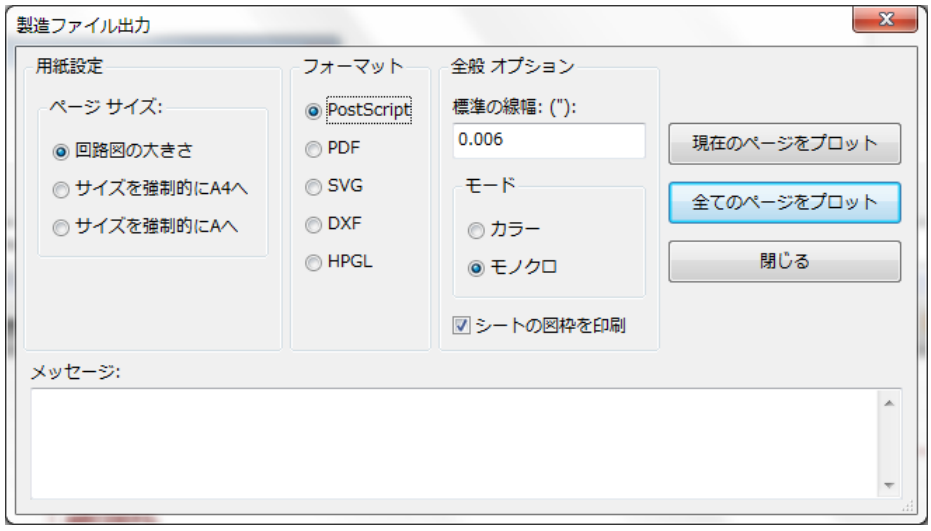
- シートの中央に原点を持つプロッターの場合，オフセットは負の値で，シート寸法の 1/2 に設定しなければなりません.
- シートの左下角に原点を持つプロッターの場合，オフセットは 0 に設定しなければなりません.

オフセットを設定するには次のことを行います.

- シートサイズを選択.
- オフセット X およびオフセット Y を設定.
- オフセットを許可をクリック.

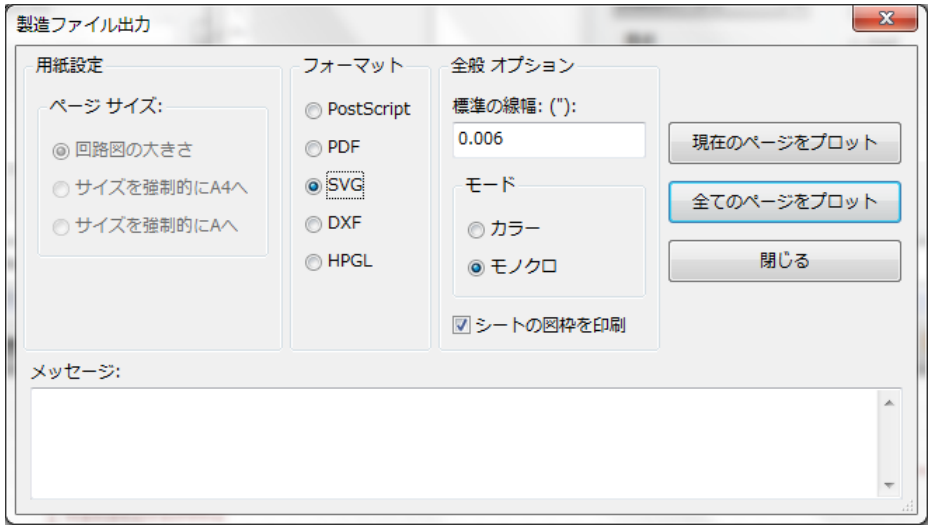
10.4 - Postscript のプロット

このコマンドにより PostScript ファイルを生成します．このオプションはアイコン  .で使用可能です．



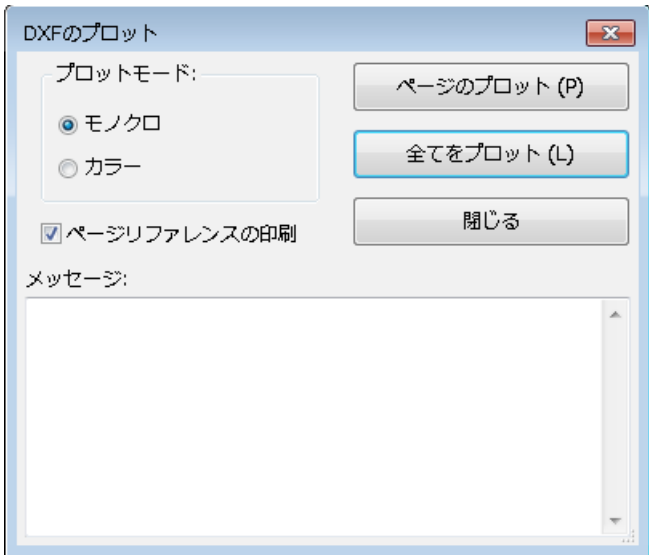
ファイル名はシート名に拡張子.ps を付加したものになります．"ページリファレンスの印刷"オプションを無効にすることが可能です．これは，文書編集ソ フトウェアで図を挿入するためにしばしば使用されるカプセル化（.eps フォーマットの）ポストスクリプトファイルを生成する場合に便利です．メッセージウィンドウは生成されたファイル名を表示します．

10.5 - SVG のプロット



プロットファイルをベクターフォーマットの SVG で生成します．このオプションはアイコン  で使用可能です．ファイル名はシート名に拡張子.svg を付加したものになります．

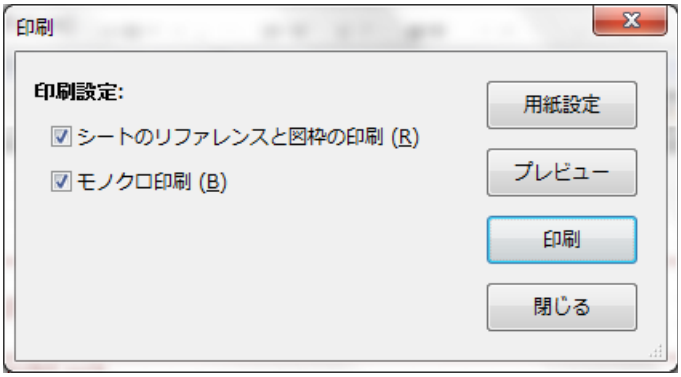
10.6 - DXF のプロット



プロットファイルを DXF フォーマットで生成します．このオプションはアイコン  で使用可能です．ファイル名はシート名に拡張子.dxf を付加したものになります．

10.7 - 紙面に印刷

このコマンド（アイコン  で使用可能）により標準的なプリンター用のデザインファイル（design files）の確認（visualize）や生成が可能です．



"シートのリファレンスとタイトルブロックの印刷"オプションは，シートのリファレンスおよびタイトルブロックの有効無効を切り替えます．

"モノクロ印刷"オプションはモノクロで印刷するように設定します．通常このオプションはモノクロのレーザープリンターを使用する場合に必要です．それは，カラーがハーフトーンで印刷され，非常に読みにくくなることがよくあるからです．

11 - LibEdit - コンポーネント管理

目次

11.1 - ライブラリに関する一般情報

ライブラリ

回路図で使用するコンポーネントはすべてコンポーネントライブラリに保存されています。これらのコンポーネントを管理するシンプルな方法を持つことを可能にするため、ライブラリファイルは、トピック、機能または製造者によってグループ分けされています。

ライブラリ管理メニューによりライブラリを作成、コンポーネントを追加、削除あるいは移動させることができます。ライブラリ管理メニューはまた、ライブラリのすべてのコンポーネントをすぐに表示することができます。

管理メニュー

２つのライブラリ管理メニューが利用可能です。

- ViewLib によりコンポーネントに素早くアクセスしてコンポーネントを見ることができます。

  ViewLib はアイコン  から使用可能です。
- LibEdit. によりすべてのコンポーネントとライブラリを管理することができます。

  LibEdit はアイコン  .から使用可能です。

11.2 - コンポーネントの概要

ライブラリ内のコンポーネントは以下のものから構成されます。

- グラフィカルな型式(design)(ライン、円、テキストフィールド)。
- ERC ツールが使用する電気的なプロパティを記述している通常のグラフィック上の規格(通常の(regular)ピン、クロックピン、反転ピンまたは Low レベルアクティブ)を順守していなければならないピン。
- リファレンス、値、PCB 設計用の対応するモジュール名などのようなテキストフィールド。

コンポーネントはエイリアス、つまりいくつかの名前(例えば 7400 が 74LS00、74HC00、7437 とも言われ、そのためこれらすべてのコンポーネントは回路図の型式が同一である)を持つことが可能です。

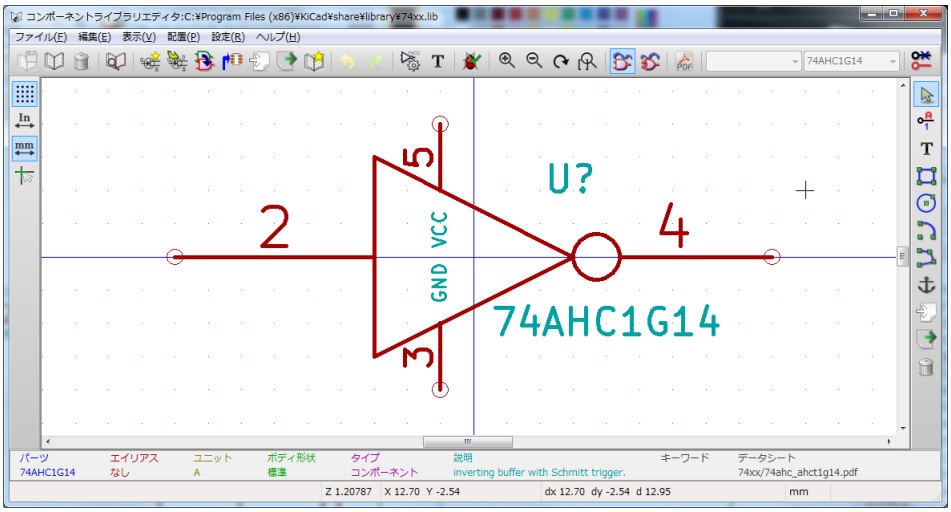
エイリアスの使用は、完全であるがコンパクトにライブラリを作成する非常に興味深い方法で、また短期間でライブラリを構築するための方法を示しています。

コンポーネントを設計するということは以下のことを意味します。

- 通常のプロパティを定義する：ド・モルガン表現または変換表現として知られる可能な２通りの表現を持つ複数パーツを定義します。
- ライン、矩形、円、ポリゴンおよびテキストを使用して設計する。
- ピンを追加する。グラフィック要素、名前、ピン数、電気的プロパティ(入力、出力、３ステート、電源ポートなど)を慎重に定義します。
- 他のコンポーネントの型式とピン配置が同じである場合、エイリアスを追加する。あるいは他のコンポーネントからコンポーネントを作成した場合、エイリアスを削除する。
- 可能なフィールド(モジュール名は PCB 設計ソフトウェアにより使用される)を追加したり、あるいはそれらの可視性を定義する。
- テキストやデータシートの www リンクなどを使用してコンポーネントを記録する。
- 適切な(desired)ライブラリにそれを保存する。

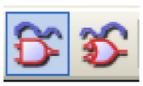
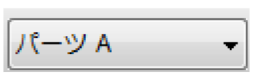
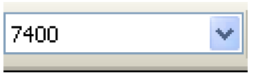
11.3 - 編集用コンポーネントの読み込み

アイコン  をクリックし、コンポーネント編集用の Libedit を開きます。Libedit は以下のように見えます。



Libedit - メインツールバー

	現在のライブラリをハードディスクに保存する。
	現在のライブラリを選択する。

	現在のライブラリ内のコンポーネントを削除する。
	新規コンポーネントを作成する。
	現在のライブラリから編集用にコンポーネントを読み込む。
	読み込んだ現在のコンポーネントから新規コンポーネントを作成する。
	現在のライブラリの現在のコンポーネントを RAM にのみ保存する。 ディスク上のライブラリファイルは変更されない。
	コンポーネントを 1 つインポートする。
	現在のコンポーネントをエクスポートする。
	現在のコンポーネントを使用して新規ライブラリファイルを作成する。
	元に戻す／やり直しコマンド
	コンポーネントのプロパティを編集する。
	コンポーネントのフィールドを編集する：リファレンス、ライブラリでの値／名前および他のフィールド
	表現を表示する：ノーマルまたは変換(ド・モルガン)表現
	関連ドキュメント(が存在する場合)を表示する
	パーツ(多パーツコンポーネントの場合)を選択する
	エイリアス(現在のコンポーネントがエイリアスを持つ場合)を選択する
	ピンを編集する：ピンの形状と位置を個別に編集(多パーツおよびド・モルガン表現の場合)
	

ライブラリの選択および保守

アイコン  で現在のライブラリを選択することが可能です。それにより使用可能なすべてのライブラリを表示し、ライブラリを選択することができます。コンポーネントを読み込んだり保存する場合、このライブラリに対して行われます。コンポーネントのライブラリ名はそのフィールド<<Value>>でもあります。

注

- ライブラリの内容を使用するためには、Eeschema でライブラリを読み込まなければなりません。
- 現在のライブラリの内容は変更後に  をクリックして保存することが可能です。
-  をクリックしてコンポーネントをライブラリから削除することが可能です。

コンポーネントの選択および保存

コンポーネントの編集時には、実際にライブラリ内のコンポーネントに対して作業しているのではなく、ローカルのメモリ内にあるそのコピーに対して作業しています。従って、どのような編集作業も容易に元に戻すことが可能です。また、コンポーネントはローカルのライブラリから、あるいは既存のコンポーネントから作成することもできます。

選択

アイコン  により、利用可能なコンポーネントのリストを表示します。そのコンポーネントは選択および読み込みが可能です。

注

コンポーネントのエイリアスを選択すると、メインのコンポーネントが読み込まれます。Eeschema は実際に読み込んだコンポーネントの名前を常にウィンドウのタイトルに表示します。

- コンポーネントのエイリアスのリストは常に各コンポーネントと共に読み込まれ、このため編集することができます。
- あるエイリアスを編集したい場合、そのエイリアスはツールバーウィンドウ内で選択されていなければなりません： 

リストの最初の項目はルートコンポーネントです。

注

もう一つの方法として、エクスポートコマンド  で前回保存したコンポーネントを、インポートコマンド  により読み込むことができます。

コンポーネントの保存

変更後、コンポーネントを現在のライブラリかまたは新規ライブラリに保存することが可能です。あるいはバックアップファイルにエクスポートことが可能です。

現在のライブラリに保存するには、更新コマンド  を使用します。更新コマンドはローカルメモリ内にコンポーネントを保存するだけであるということを覚えておいて下さい。

コンポーネントを永続的に保存したい場合は、保存アイコン  を使用しなければなりません。それはローカルのハードディスク上のライブラリファイルに変更を加えます。

このコンポーネントで新規ライブラリを作成したい場合、NewLib コマンド  を使用して下さい。そのとき、新規ライブラリ名が必要となります。

注

自分で作成したコンポーネントを検索できるようにしたい場合、そのライブラリを Eeschema のライブラリのリストに追加するのを忘れないで下さい(Eeschema の設定を参照)。

最後に、エクスポートコマンド  を使用して、そのコンポーネントだけを含むファイルを作成することが可能です。このファイルはコンポーネントを一つだけ含む標準ライブラリファイルになり ます。実際、NewLib コマンドとエクスポートコマンドは基本的に同じものです。１つ目のコマンドは、デフォルトラ

イブラリのディレクトリ内にライブラ リを作成するためのデフォルトのオプションです、２番目のコマンドは、ユーザーディレクトリにライブラリを作成するために使用します。

ライブラリ間のコンポーネントの移動

コピー元のライブラリからコピー先のライブラリに簡単にコンポーネントをコピーすることが可能です。それには次のコマンドを使用します

-  ボタンでコピー元ライブラリを選択する。
-  ボタンで移動するコンポーネントを読み込む。そのコンポーネントが表示されます。
-  ボタンでコピー先のライブラリを選択する。
-  ボタンで現在のコンポーネントをローカルメモリに保存する。
-  ボタンでコンポーネントをローカルライブラリ(コピー先のライブラリ)に保存する。

コンポーネントの編集の取り消し

あるコンポーネントに対して作業している時には、その編集したコンポーネントというのは、単にそのライブラリ内の実際のコンポーネントの作業コピー です。このことはメモリ内にそのコンポーネントを保存しない限り、それをただ再読み込みして、行ったすべての変更を取り消ことが可能です。

ローカルのメモリ内にそれをすでに保存し、ハードディスク上のライブラリファイルには保存していない場合には、Eeschema を終了、再起動して、全ての変更を元に戻すことが常に可能です。

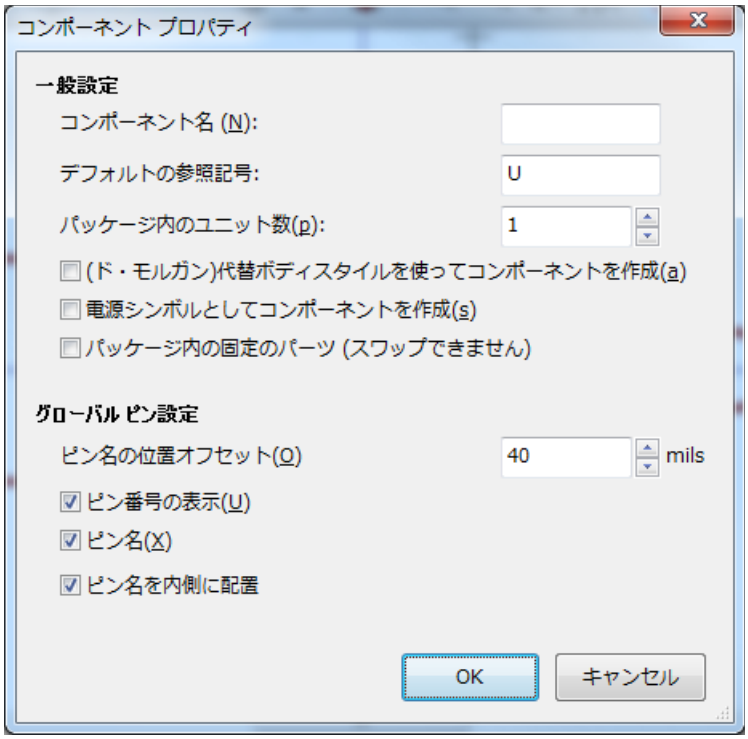
11.4 - ライブラリコンポーネントの作成

新規コンポーネントの作成

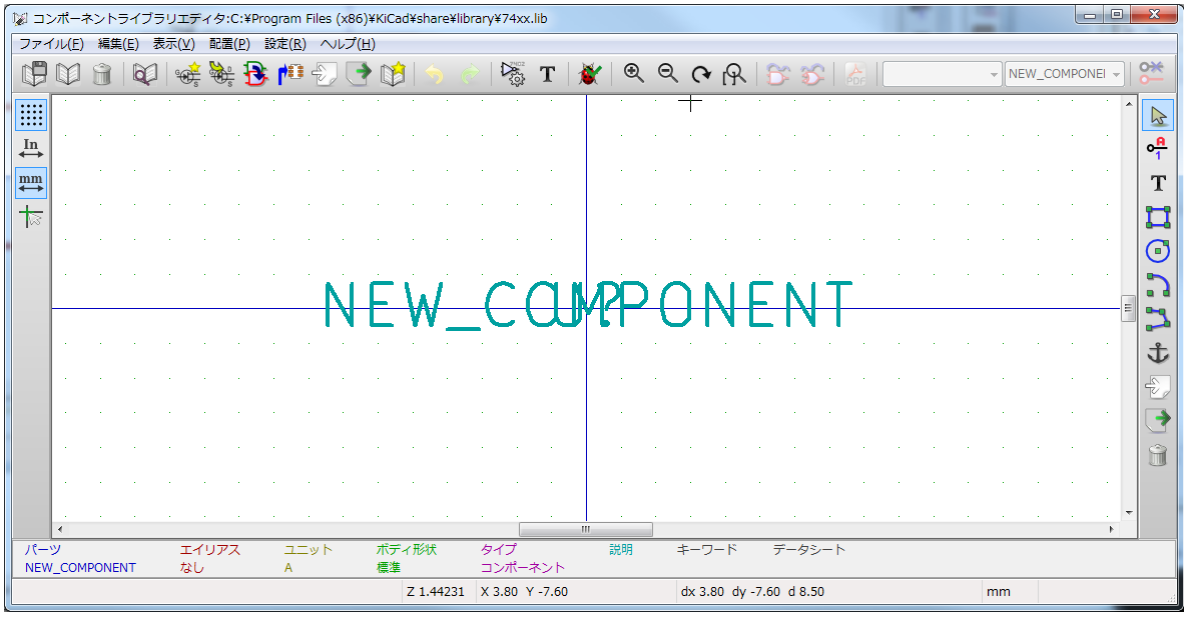
 ボタンの NewPart コマンドを使用して新規コンポーネントの作成が可能です。コンポーネント名(名前は Libedit 用のフィールド値でもあり、回路 図エディターで Value フィールド用のデフォルト値として使用される)、リファレンス(U、IC、R ...)、パッケージ内のパーツ数(例えば、標準コンポーネントの 7400 A は 1 つのパッケージに 4 つのパーツで構成されている)、(標準としてド・モルガンの)変換表現が存在するかどうかの入力が要求されます。

フィールドリファレンスが空欄のままであると、リファレンスはデフォルトの"U"になります。

これらのデータはすべて後で設定することが可能ですが、コンポーネントの作成時に設定する方が望ましいのです。



まず初めに、コンポーネントはこのように見えます。



他のコンポーネントからコンポーネントを作成

作成したいと思っているコンポーネントが KiCad のライブラリにあるものと似ているということがしばしばあります。この場合、既存のコンポーネントを読み込んで変更するのがごく普通です。それを行うステップは以下の通りです。

- 出発点として使うコンポーネントを読み込みます。
- アイコン  をクリックするかまたはその名前を変更(名前を右クリックし、テキスト <<Value>>を編集。新規コンポーネント名を入力するように求められる)します。
- 型となるコンポーネントにエイリアスがある場合、新規コンポーネントからその型と衝突するエイリアスを削除するよう促されます。その答えが NO の場合、新規コンポーネントの作成が中止されます。
- コンポーネント名を変更します。
- 必要に応じて新規コンポーネントを編集します。

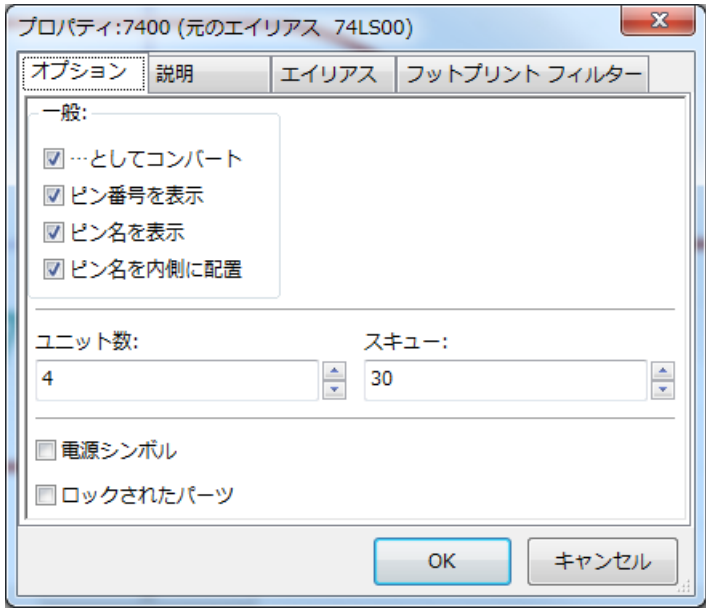
-  ボタンでメモリに新規コンポーネントを保存するか、または  ボタンで新規ライブラリに保存します。あるいは他の既存のライブラリに新規コンポーネントを保存したい場合には、コマンド  でそのライブラリを選択して、その新規パーツを保存します。
-  ボタンのファイル更新コマンドでディスクにライブラリファイルを保存します。

コンポーネントの主要特性の編集

コンポーネントの主要な特性(features)は次のようなものです。

- パッケージあたりのパーツ数。
- 変換表現の存在。
- 関連文書。
- 様々なフィールドの最新情報。

これらの特性はコンポーネント作成時に注意深く追加すべきです。そのほかにも、それらは型となるコンポーネントから引き継がれます。どちらの場合でも、編集コマンド  を使用する必要があります。
編集ウィンドウは次のように見えます。



通常のプロパティを定義するオプションで重要なものは、1) パッケージあたりのパーツ数を定義するユニットの数、2) コンポーネントが２通りの表現があるかどうかです。

ピンを編集したり作成したりする場合、すべてのパーツの対応するピンが一緒に出力され(published)、作成されるので、これら２つのパラメータが正しく設定されることは非常に重要です。

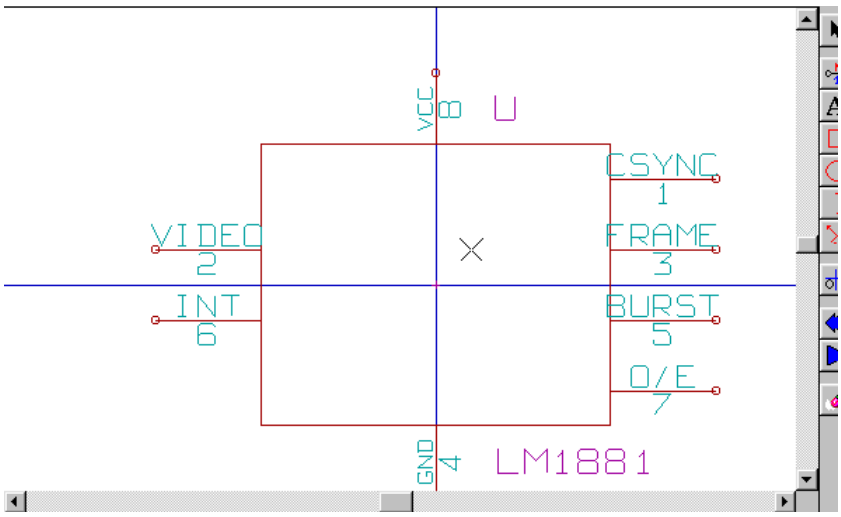
ピンの作成／編集後にパーツ数を増やす場合、この増加に伴う追加の作業が必要です。そうであるにしても、これらのパラメータの変更はいつでも可能です。
グラフィックオプションは次の通りです。

- ピンナンバーを表示
- ピン名を表示

上記でピン番号とピン名のテキストの可視性を定義します。対応するオプションがアクティブの場合、そのテキストが表示されます。

オプション"ピン名を内側に配置"はピン名の位置を定義します；オプションがアクティブの場合、そのテキストはコンポーネント外形線の内側 に表示されます。この場合、ピン名スキューパラメータは内側方向へのテキストの変位を定義します。値は(1/1000 インチ単位で)30~40 が妥当で す。

以下の例は、ピン名を内側に配置オプションにチェックをつけない状態で同じコンポーネントを示しています。ピン名とピン番号の位置に注意して下さい。



多パーツコンポーネント

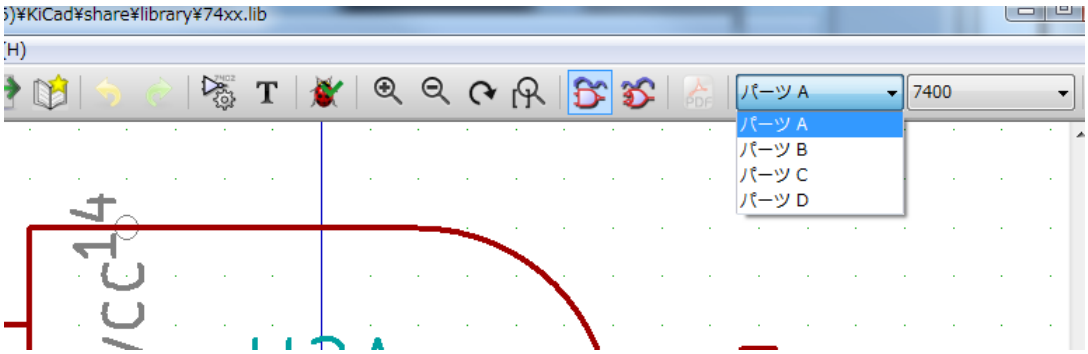
コンポーネント要素の編集中、コンポーネントが複数のパーツまたは表現を持つ場合、このコンポーネントのそれぞれのパーツまたは表現を選択する必要があります。

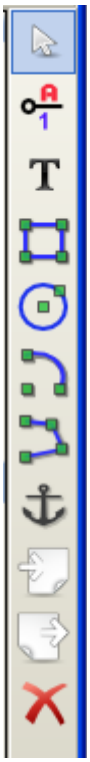
表現の選択の場合、アイコン  かまたはアイコン  をクリックします。

パーツの選択の場合。

11.5 - コンポーネント設計

右側の垂直ツールバーでコンポーネントの全要素を配置することができます。





コンポーネントを編集するために、次のグラフィック要素を使用可能です：

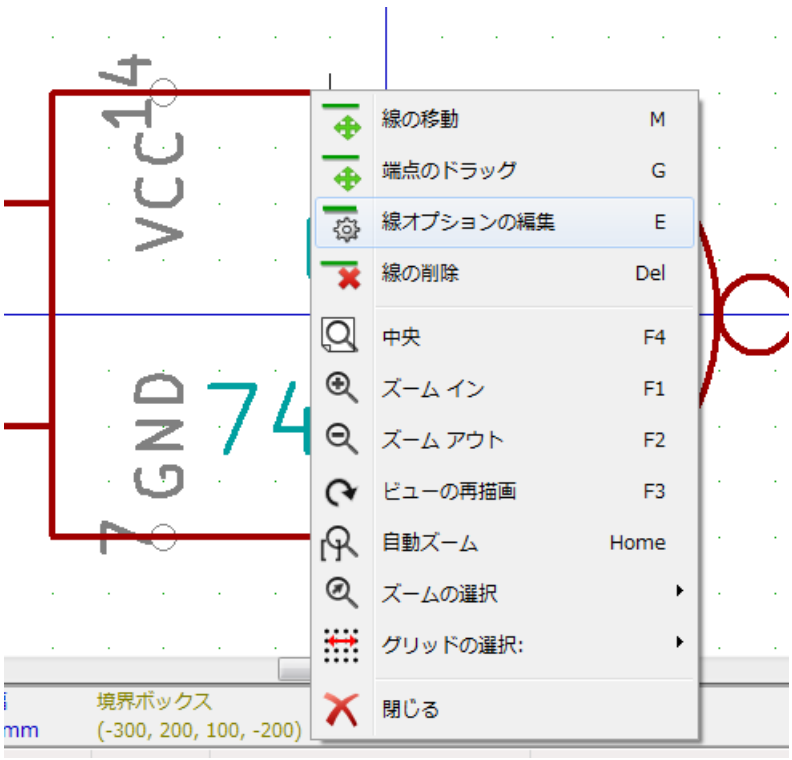
- ライン(およびポリゴン、外形線または塗り潰し)
- 矩形
- 円
- 円弧
- テキスト(フィールドおよびピンのテキスト以外)

ピンとテキストフィールド(値、リファレンス)は、純粋なグラフィック要素ではないので別々に扱われます。

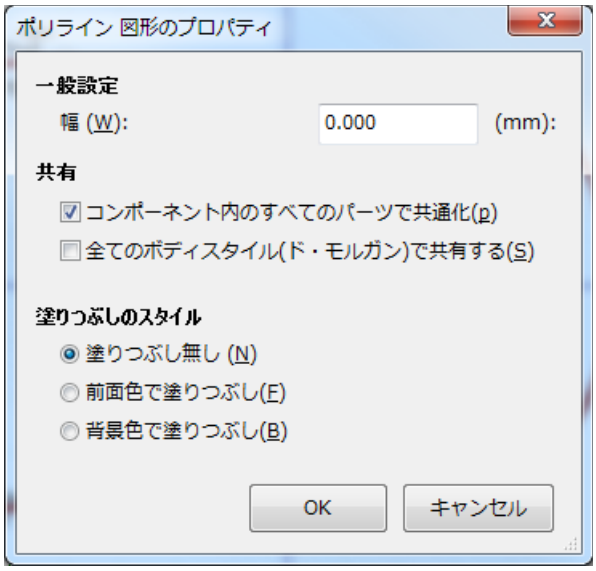
グラフィック要素メンバーシップオプション

個々のグラフィック要素は、表現のタイプ(ノーマルまたは変換)あるいはコンポーネント内の個別のパーツのどちらかに対して、共通(ordinary)かまたは固有である定義することが可能です。

オプションを設定したい(concerned)要素を右クリックするとオプションメニューにアクセスできます。ラインの例を示します。



あるいはこの要素をダブルクリックすると、次のメニューが表示されます。



グラフィック要素の標準的なオプションは次の通りです。

- コンポーネント内の全パーツで共有するにチェックをつける。コンポーネント内の個別のパーツは同じグラフィック表現を持ち、そのためパーツを１つだけ作成するので十分だからです。
- 全てのボディスタイル(ド・モルガン)で共有するにチェックをつけない。２通りの表現を取り入れて、それぞれの表現により異なるグラフィック表現を持つようにするからです。

それから、それぞれのグラフィック表現を作成することが必要です。

"ポリゴン"(ラインが連続的に描画される)タイプの要素では、背景を塗り潰すまたは前景を塗り潰すオプションにより、塗り潰しのポリゴンを生成することができます。

しかし、コンポーネント内の全てのパーツで共有するにチェックを付けないことによって、異なるグラフィックタイプで設計された多パーツコンポーネントの場合(幸運にも稀である)を扱うことができます。

その時は各素を作成しなければなりません。また、オプション"表現に固有"にチェックがついている場合、各パーツに２通りの表現を作成することが必要です。

最後に、最新の IEEE スタンダードで作成したコンポーネントについて、全てのボディスタイル(ド・モルガン)で共有するのオプションにチェックを付ける ことは興味深いことであると言えます。それはグラフィックの本質的要素がノーマルおよび変換表現において同一であるからです。

幾何グラフィック要素

次のツールを使用してそれらの設計が可能です。

- ラインおよびポリゴン、オプションにチェックが付いている場合、外形または塗り潰し。
- 対角線で定義される矩形。
- 中心と半径で定義される円。
- 始点と終点および中心で定義される弧。弧は 0°から 180°まで描画されます。

テキストタイプのグラフィック要素

アイコン  により、テキストを作成がすることができます。テキストは、コンポーネントが反転(mirrored)したとしても常に読むことができます。

11.6 - ピンの作成および編集

アイコン  をクリックしてピンの作成や挿入が可能です。ピンのすべての機能の編集はピンをダブルクリックして行います。もう一つの方法として、右クリックして高速(fast)編集メニューを開きます。

ピンは慎重に作成しなければなりません。それは、どのようなエラーも PCB の設計に影響するからです。すでに配置済みの任意のピンは再編集、削除または移動が可能です。

ピンの概要

ピンはその形状(長さ、グラフィックな外観)、名前および"番号"で定義されますが、番号は常に数字であるとは限りません。PGA ソケットは A12 または AB45 のように文字と数字で定義されます。EEschema ではピン番号を４文字までの英数字で定義します。

ERC ツールを機能させるためには、"電気的"タイプ(入力、出力、３ステート...)も定義されていなければなりません。このタイプがうまく定義されていない場合には、ERC チェックが非効率になります。

重要な注意

- ピン名とピン番号に空白(space)を使用しないで下さい。
- 反転信号のピン名はシンボル"~"で始めます。
- ピン名の文字数を減らしてこの信号のシンボルだけになった場合は、そのピンには名前がないと見なされます。
- "#"で始まるピン名は電源ポート用に予約されています。
- ピン番号は 1～4 文字の英数字から成ります。1、2、...、9999 は有効な数字ですが、A1、B3...(標準的な PGA の表記法)、あるいは Anod、Gnd、Wine なども有効です。

多パーツポンポーネント - ２通りの表現

特に論理ゲート場合、シンボルは２通りの表現(ド・モルガンとして知られる表現)を持つことが可能で、また、IC は数個のパーツを含む(例えば数個の NOR ゲート)場合があり得るということを思い出しましょう。

ある IC の場合、異なるグラフィック要素やピンがいくつか必要かもしれません。

例えば、リレーは異なる構成要素で表現することが可能です。

- コイル
- スイッチ接点 1
- スイッチ接点 2

多パーツ IC および２通りの表現を持つコンポーネントの管理は柔軟です。

ピンは次のようになり得ます。

- それぞれのパーツに共通または固有。
- 両方の表現に共通またはそれぞれの表現に固有。

デフォルトでは、ピンは各パーツのそれぞれの表現に固有です。それは、ピンの数は各パーツで異なり、それらの型式は各表現で異なるからです。

ピンが共通である場合、単に一度作成する必要があります(例えば、電源ピンの場合に)

すべてのパーツでほとんど常に同じ型式の場合もありますが、ノーマルの表現と変換表現の間には違いがあります。

ピン - 基本的なオプション

多パーツおよび／または多重表現を持つコンポーネントはピンの作成と編集の場合に特に厄介です。ピンの大部分が各パーツに固有で(それらのピン番号が各 パーツに固有なので)、また各表現に固有である(それらの形状が各表現に固有なので)限りにおいては、ピンの作成と編集はこのため時間がかかり厄介です。

Eeschema はピンを同時に処理することができます。

デフォルトでは、多パーツコンポーネントや二重表現の場合は、ピン(例えば同じ座標に配置されている全てのピン)を作成、編集(形状と番号を除いて)、削除または移動する時に、パーツおよび表現に対応するすべてのピンに対してこれらの変更がなされます。

- 型式(design)の場合、現在の表現に加えた変更はすべてのパーツに適用されます。
- ピン番号は現在のパーツについて、２つの表現について変更されます。
- 名前は単独で変更されます。

大抵の場合で速やかな変更ができるようにするためにこの依存性を設定しました。

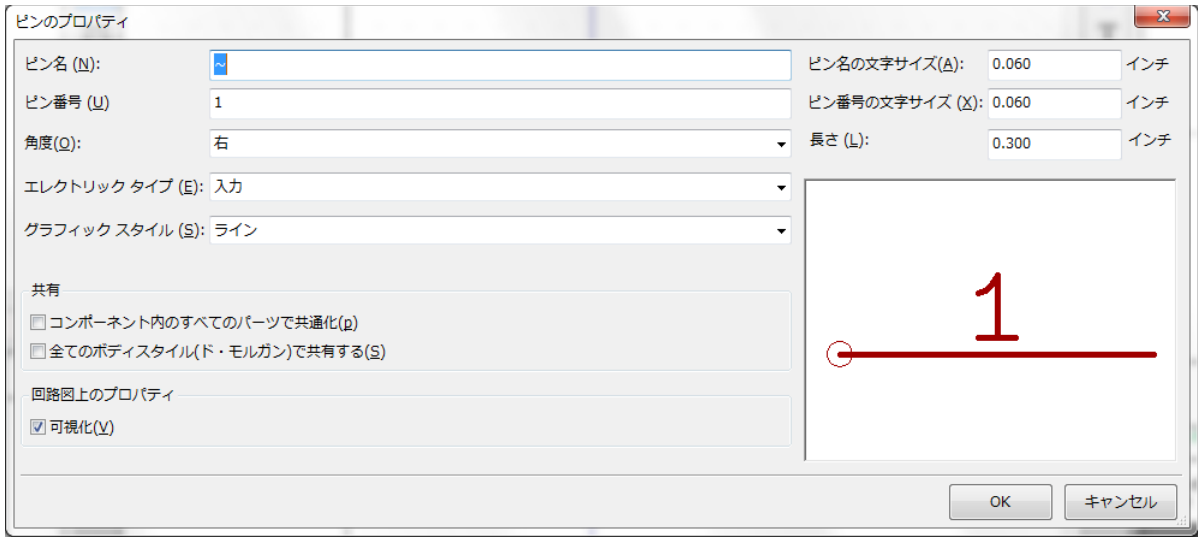
変更に関するこの依存性はオプションメニューで禁止することが可能です。それにより、完全に独立した特性のパーツと表現を持つコンポーネントを作成することができます。

この依存性のオプションは以下のツールで管理されます。

-  ボタンがアクティブでない(ハイライトされていない)場合、編集は全パーツおよび全表現に適用されます。
-  ボタンがアクティブ(ハイライトされている)の場合、編集は現在のパーツおよび表現にのみ(つまり画面上に見えるものに)適用されます。このオプションはほとんど使用されません。

ピン - 特性の定義

ピンプロパティウィンドウでピンの全ての特性を編集することができます。



このメニューはピンを作成したり、あるいは既存のピンをダブルクリックすると、自動的にポップアップします。

このメニューで以下の定義または変更が可能です。

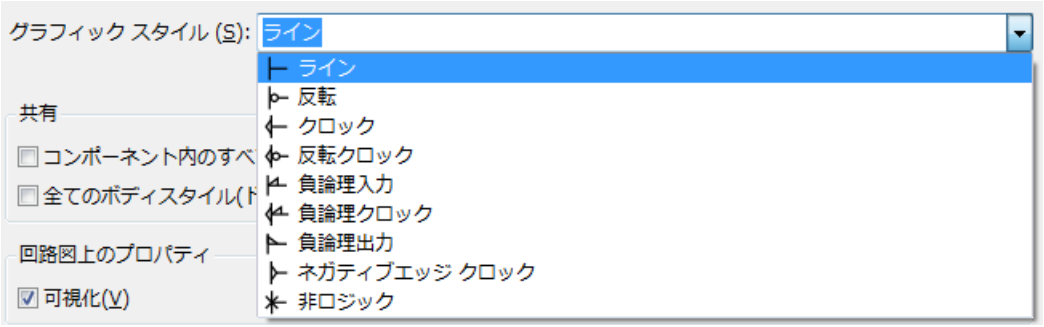
- ピン名とピン名のサイズ。
- ピン番号の番号とサイズ。
- ピン長。
- 電気的なタイプと型式。
- メンバーシップ。ノーマルおよびドモルガン表現に共通。
- 非表示ピン。電源ピンに使用される。

また、次のことを覚えておきしましょう

- 反転信号の場合にはピン名は"~"で始まります。
- 名前が 1 文字だけに減らされると、そのピンには名前がないと見なされます。
- ピン番号は 1～4 文字(英数字)から成ります。-1、2..9999 は有効な数字ですが、A1、B3...(標準的な PGA の表記法)、あるいは Anod、Gnd、V.in なども有効です。

ピン形状

それぞれのピンの形状を下図に示します。



形状の選択は単にグラフィック上の影響があるだけで、ERC チェックあるいはネットリスト機能には何の影響もありません。

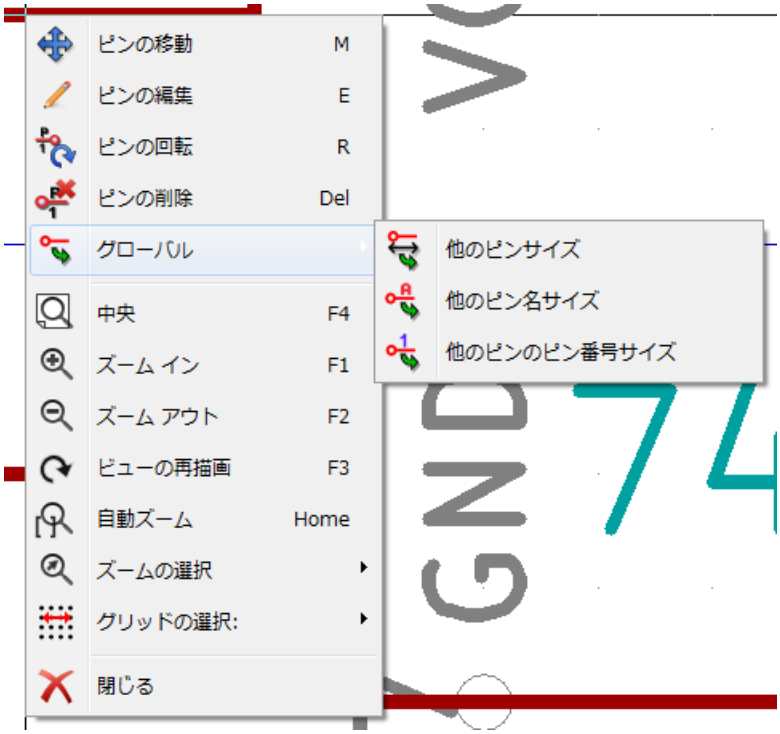
ピン - 電気的タイプ

タイプの選択は ERC ツールに重要です。その選択は IC の入力や出力ピンでは普通に行います。

- BiDi タイプは入力および出力を切り替え可能な双方向ピン(例えばマイクロプロセッサのデータバス)を表します。
- ３ステートタイプは通常の３ステート出力です。
- 受動タイプは抵抗、コネクタなどの受動コンポーネントのピンに使用されます。
- 不特定タイプ(指定なし)は、ERC チェックが関係ない場合に使用することが可能です。
- 電源タイプはコンポーネントの電源ピンに使用されるべきものです。特に、ピンがポートの電源で、"非表示"として宣言されている場合、回路図には表示されません。また、それは自動的に同じタイプの同じ名前(非表示電源ピン)の他のピンに接続されます。
- 電源出力はレギュレータ出力用です。
- オープンエミッタとオープンコレクタタイプも使用可能です。

ピン - グローバル変更

すべてのピンの長さ、あるいはテキストサイズ(名前、パーツ番号)を変更可能です。ポップアップメニューのグローバルコマンドを使用してこれら３つのパラメータの一つを設定します。



変更したいパラメータをクリックし、新しい値を入力します。その値は現在の表現のすべてのコンポーネントピンに適用されます。

ピン - 多パーツコンポーネントおよび二重表現

(7400、7402 などのような)様々なパーツまたは表現は補足的な編集が必要になることがあります。

この補足的な作業は、次の事前策がとられるなら限定的なものになります。

- 一般オプションのパーツごとの編集  のチェックをはずしておかなければなりません。
- 電源ピンは属性共通ユニットおよび common convert active で作成されます(それらは非表示(描画なし)の場合もあります)。

正しい手順は次の通りです。

他のピンが作成された時に、それらは各パーツと各表現用に作成されます。

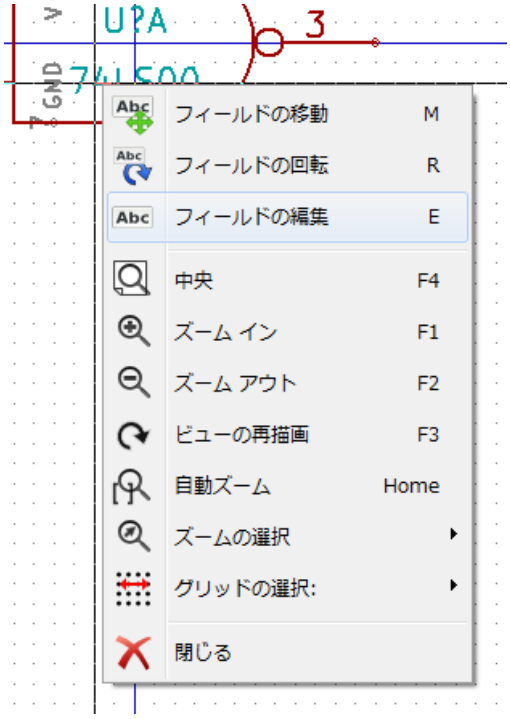
例えば、Eeschema は８つの試料(specimens)の中に 7400 のパーツ A の出力ピンを作成します：パーツ毎に２つ(A、B、C、D の４つパーツがあり、それぞれのパーツについてノーマル表現およびド・モルガンとして知られる変換表現があります)作成します。

しかしながら、Eeschema は最初にパーツ A をノーマル表現で正確に作成します。そのため各パーツには次のことを行う必要があります

- 変換表現を選択して、それぞれのピンの形状と長さを編集する。
- その他のパーツについては、ピン番号を編集する。

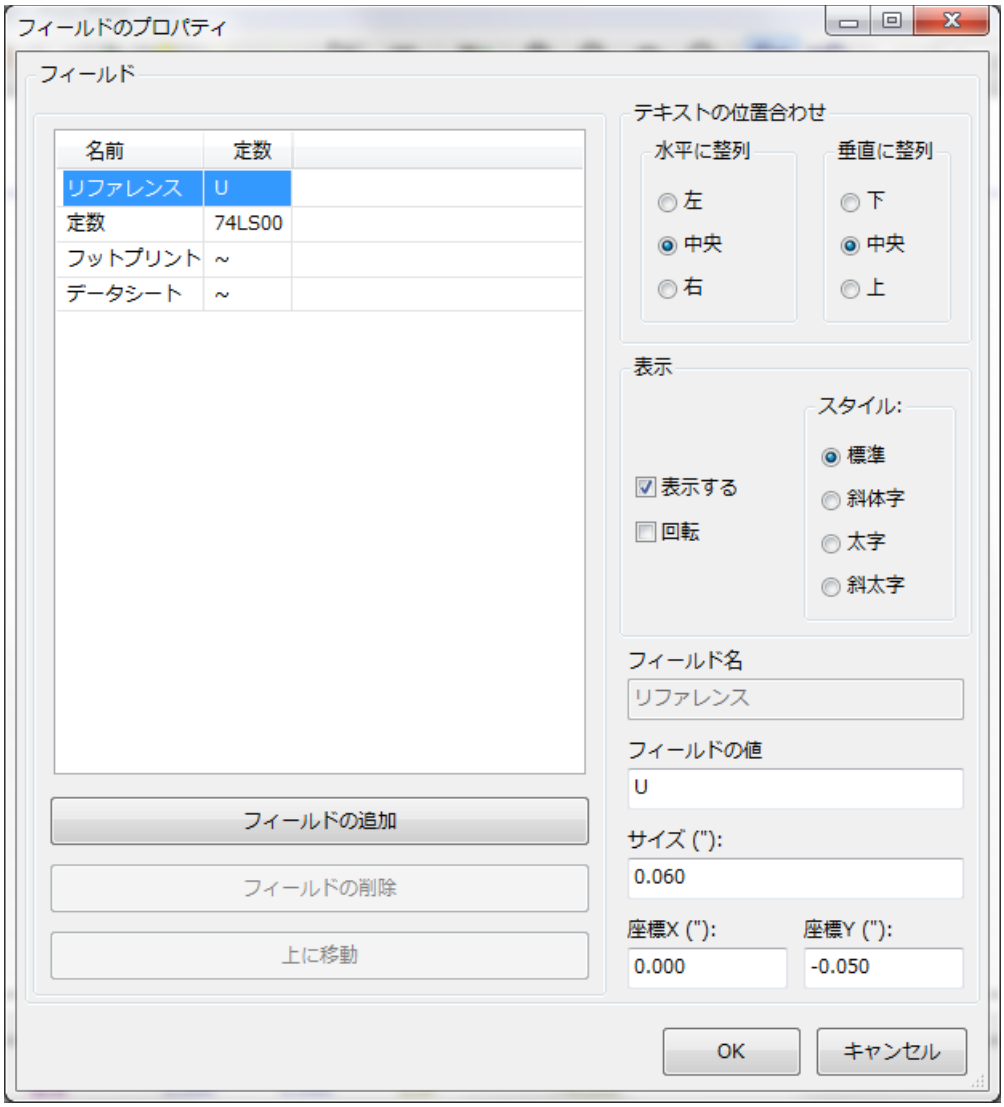
11.7 - フィールドの編集

既存のフィールドの場合、右クリックで高速編集コマンドを使うことが可能です。



より完全な編集をする場合または空のフィールドの場合には、フィールド編集ウィンドウ  を呼び出す必要があります。

そのダイアログウィンドウはこのように見えます。



ここではリファレンスのフィールドが選択されています。フィールドはコンポーネントに関連したテキストの区画(section)で、コンポーネントのグラフィック表示に属するテキストと混同すべきではありません。

これらのフィールドは常に使用可能で、以下の通りです。

- 値
- リファレンス
- 関連するモジュールの名前(PCB 用のフットプリント)
- (主に回路図で使用されることを意図した)ドキュメント・ファイルへのリンク。
- 回路図エディターで定義したテンプレートフィールド(コメント用)

値およびリファレンスのフィールドはコンポーネント作成時に定義され、ここで変更が可能です。モジュール(フットプリント)名を含む(PCB ソフト ウェア用の)ネットリストを直接生成するために、関連するモジュールの名前フィールドを編集することは場合によっては便利であることがあります。

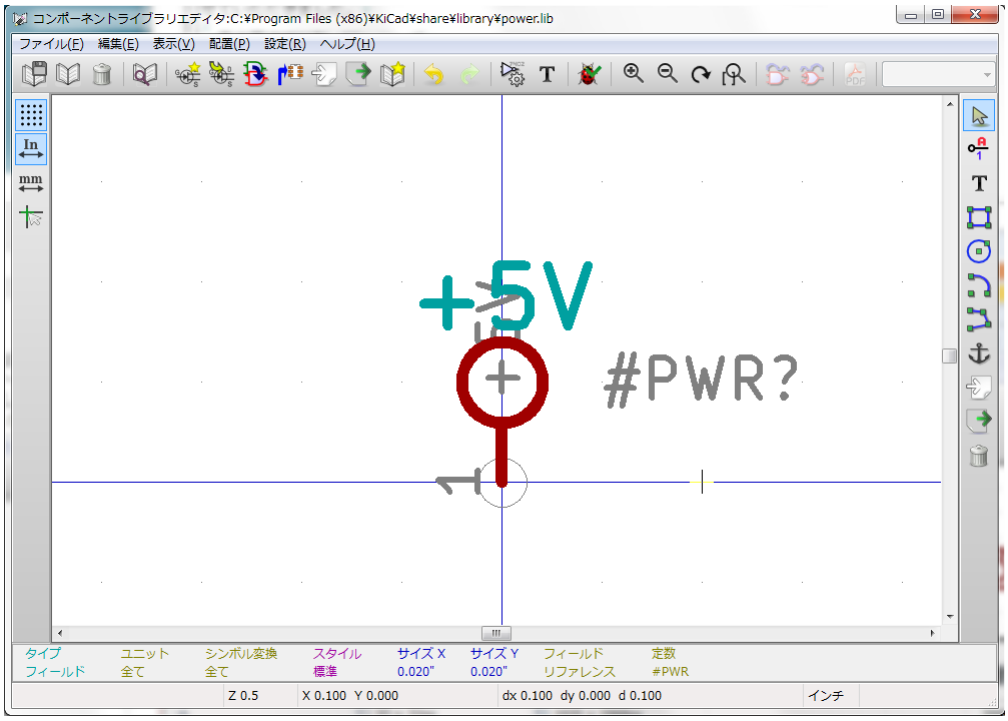
関連する図の名前フィールドは他の電子系 CAD ソフトウェア用に特に有用です。ライブラリの場合、値およびリファレンスフィールドの編集によって、それらのサイズと位置を定義することができます。

重要事項

- 値フィールドのテキストを変更することは、型として使用する既存のコンポーネントを元にして新規コンポーネントを作成することに相当します。実際、ライブラリに保存する場合、この新規コンポーネントは値フィールドに入っている名前を持ちます。
- 非表示のフィールドを編集するには(つまり何もないように見え、そのフィールドが非表示の属性を持っているとしても、LibEdit では表示されるので)、上の一般編集ウィンドウを使用する必要があります。

11.8 - 電源ポートシンボルの作成

電源ポートのシンボルは通常のコンポーネントと同様に作成します。Power.lib のような専用のライブラリにそれらを集めるのが便利かもしれません。 それらはグラフィカルなシンボル(希望する形状)で構成され、"非表示電源"タイプのピンです。電源ポートのシンボルはこのため回路図入力ソフトウェアで 他のすべてのコンポーネントのように扱われます。いくつかの事前策が必須です。
+5V の電源シンボルがあります。



以下のステップに従ってシンボルを作成します。

- "非表示電源"ピンは+ 5V(この名前で+ 5V のネットに接続を行う(establish)ので重要)という名前で、ピン番号が 1(番号は重要ではない)で長さはありません。
- 外形は"ライン"タイプで、明らかにタイプは"電源"で、属性が"非表示"です。
- グラフィックは小さな円で、ピンから円までの線分が作成されます。
- シンボルのアンカーはピン上にあります。
- ピン名のように値を+ 5V にして、このシンボルの値を表示します(デフォルトでピンは非表示なので、その名前は現れません)。
- リファレンスはピン名のように# + 5V(こうすると# + 5V と表示される)にします。リファレンスのテキストは、最初の文字を必ず"#"とします。それ以外の文字は重要ではありません。従来どおり、リファレン スがこのシンボルで始まるすべてのコンポーネントはコンポーネントリストにもネットリストのどちらにも現れません。さらに、シンボルのオプションで、リ ファレンスは非表示として宣言されます。

新規電源ポートシンボルの作成は、他のシンボルを型として使用すると容易にそして速くできます。
単に以下のことをする必要があります

- 型となるシンボルを読み込む。
- 新規電源ポートの名前となるピン名を編集する。
- 値フィールドを編集する(電源ポートの値を表示したい場合、ピン名と同じ名前にする)。
- その新しいコンポーネントを保存する。

12 - LibEdit – 補足

目次

12.1 - 概要

コンポーネントは、次の要素で構成されています。

- グラフィカル表現(幾何学図形、テキスト)
- ピン
- ポストプロセッサで使われる関連テキスト、フィールド：ネットリスト、部品リスト

２つのフィールドが初期化されます: リファレンスと定数
コンポーネントと関連した設計名、関連するフットプリント名、フリーフィールドであるその他フィールドは、一般的には空欄のままにすることができ、回路図エディット中に充填することができます。

しかしながら、コンポーネントに関連したドキュメントを管理することで、研究や、ライブラリの使用、メンテナンスが容易になります。関連付けられたドキュメントは以下で構成されています：

- コメント行
- TTL CMOS NAND2 など空白文字で区切られたキーワード行
- 添付ファイル名(例：アプリケーションノート、pdf ファイル). 添付ファイルのデフォルトディレクトリ:
  kicad/share/library/doc
  見つからない場合:
  kicad/library/doc
  linux の場合:
  /usr/local/kicad/share/library/doc
  /usr/share/kicad/library/doc
  /usr/local/share/kicad/library/doc

キーワードにより、さまざまな選択基準に従って、コンポーネントを検索することができます。コメントとキーワードは、様々なメニューや、ライブラリからコンポーネントを選択する場合に表示されます。

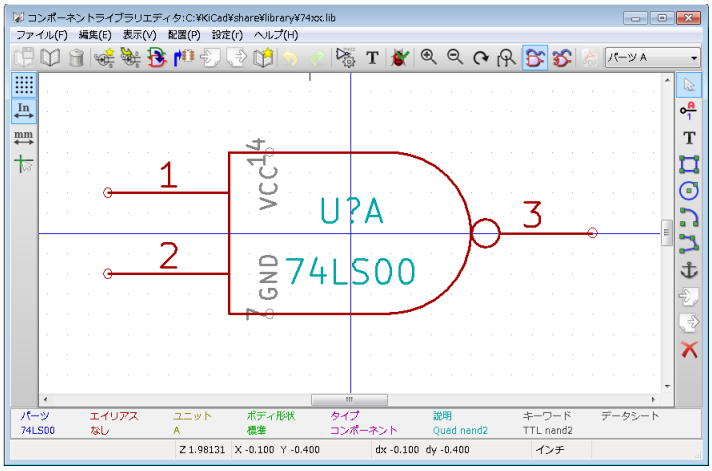
コンポーネントはまた、アンカーポイントを持っています。回転やミラーは相対的に、このアンカーポイントに対して行われ、配置時にはこの点が、基準位置として使用されます。このように、このアンカーを正確に配置することは有用です。

コンポーネントは、エイリアス、つまり等価名(equivalent names)を持つことができます。これにより、作成する必要があるコンポーネントの数をかなり減らすことができます (たとえば、74LS00 は、74000、74HC00、74HCT00... といったエイリアスを持つことができます)。

最終的に、コンポーネントは、その管理を容易にするために、ライブラリ(メーカやテーマ別に分類された)で配布されます。

12.2 - コンポーネントのアンカー位置を決める

アンカーは、座標（0,0）にあり、画面上青い軸で示されています。



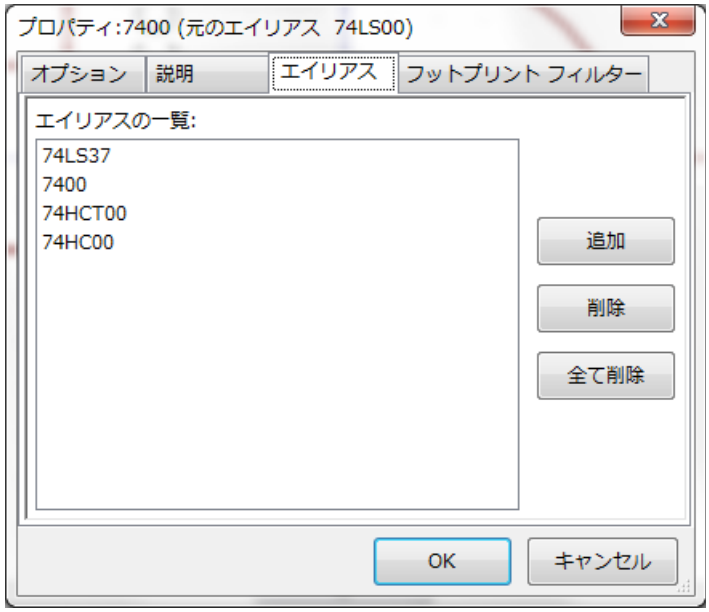
アンカーは、アイコン  を選択し、新しくアンカーを配置したい位置をクリックすることで再配置できます。 図は自動的に新しいアンカーポイント上で再センタリングされます。

12.3 - コンポーネントのエイリアス

エイリアスとは、ライブラリ内での同じコンポーネントに対応した別名のことです。類似したピン配列、表現を持つコンポーネントは、いくつかのエイリアスを持った、１つのコンポーネントで表すことができます。 (例えば 7400 は 74LS00, 74HC00, 74LS37 というエイリアスを持ちます)

エイリアスの使用により素早く完全なライブラリを構築することができます。さらに、kicad により簡単に読み込まれる、これらのライブラリは、さらにコンパクトとなります。

エイリアスのリストを変更するため、  アイコンによりメイン編集ウィンドウを選択し、エイリアスフォルダを選択する必要があります。

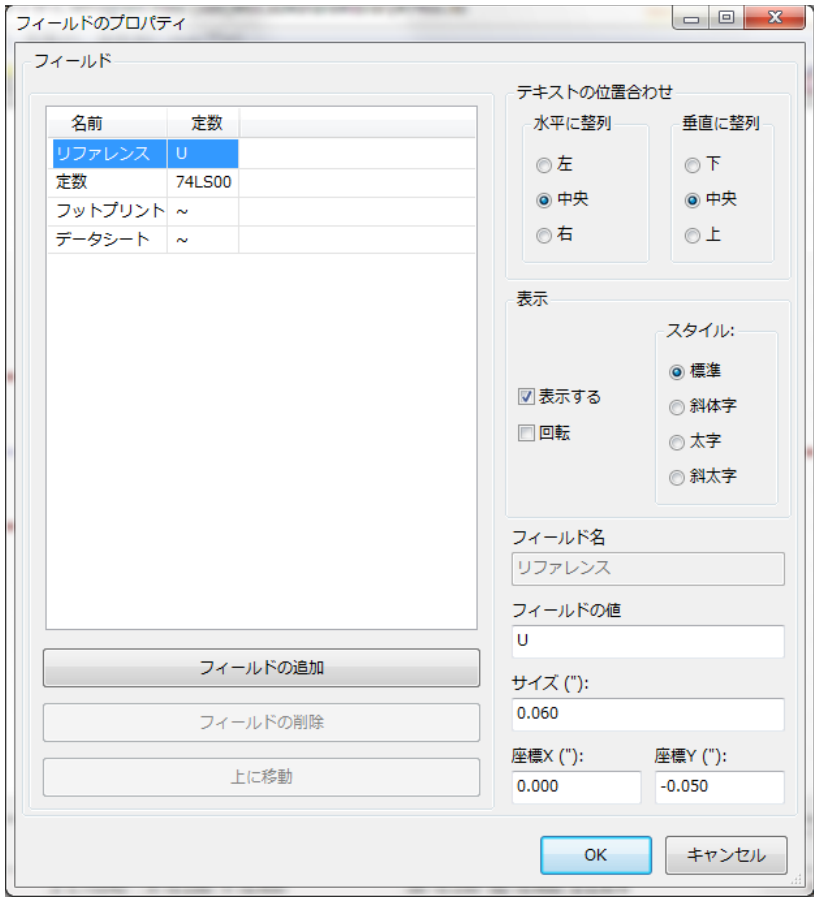


このように希望のエイリアスを追加したり、削除することができます。現在のエイリアスは、編集されているので、明らかに削除することはできません。全てのエイリアスを削除するには、まずルートコンポーネントを選択することが必要です。エイリアスの最初のコンポーネントは、メインツールバーの選択ウィンドウに載っています。

12.4 - コンポーネントのフィールド

フィールドエディタは、  アイコンにより呼び出されます。

４つの特殊フィールド（コンポーネントに付属するテキスト）と、設定可能なユーザー･フィールドがあります。

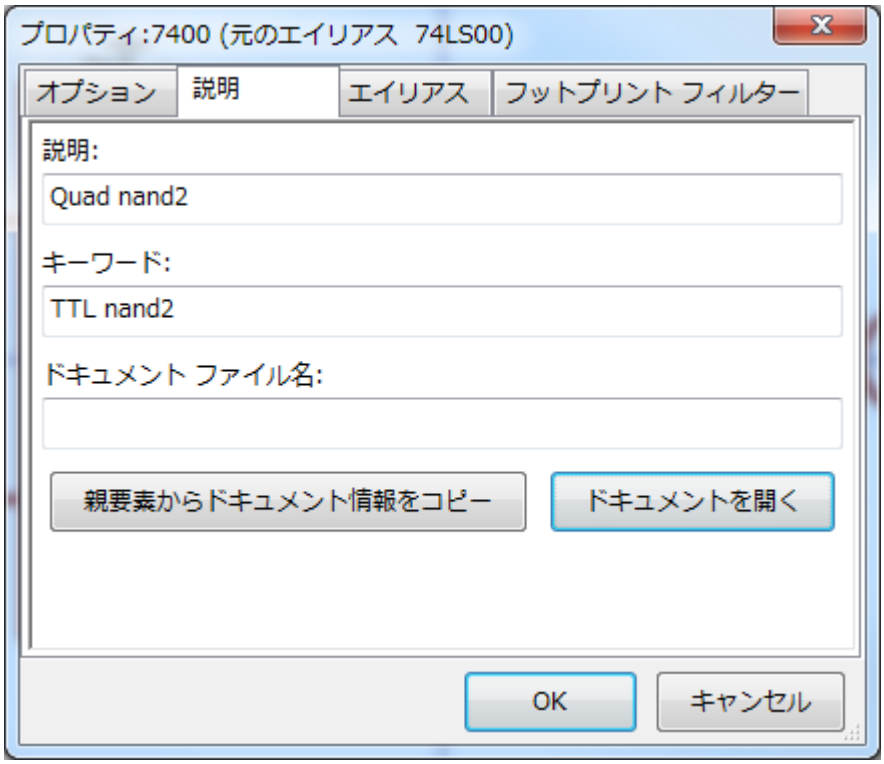


特殊フィールド

- リファレンス(Reference)
- 定数(Value)：ライブラリ内のコンポーネント名であり、回路でのデフォルト値のフィールド
- フットプリント(Footprint)： 基板で使用されるフットプリント名。CvPcb を使ってフットプリントのリストを設定する場合には、あまり有用ではないが、CvPcb が使用されていない場合には必須です。
- シート(Sheet)：執筆時点で使用されていない予約済フィールドです。

12.5 - コンポーネントのドキュメント

ドキュメントの情報を編集するためには、  アイコンを使用し、コンポーネントのメイン編集ウィンドウを呼び出し、ドキュメントフォルダを選択する必要があります。



このドキュメントは、エイリアス間で違った特性となるため、正しいエイリアス、あるいはルートコンポーネントを間違いなく選択してください。"CopyDoc"ボタンをクリックすると、現在編集しているエイリアスに対し、ルートコンポーネントからドキュメントの情報をコピーすることができます。

コンポーネントのキーワード

キーワードにより、特定の選択基準（機能、技術的ファミリなど）に従って、コンポーネントを選択的な方法で検索することができます

EESchema リサーチツールは、大文字と小文字を区別しません。ライブラリで使われる現状もっとも多いキーワードは次のものです。

- CMOS TTL ：ロジックファミリに対して
- AND2 NOR3 XOR2 INV…：ゲートに対して（AND2 = 2 入力 AND ゲート, NOR3 = 3 入力 NOR ゲート）
- JKFF DFF… ：JK あるいは D flip-flop に対して
- ADC, DAC, MUX…OpenCol ：オープンコレクタ出力をもつゲートに対して
  このように回路図エディター内で、NAND2 OpenCol というキーワードでコンポーネントを検索する場合、EESchema は、これら 2 つのキーワードを持つコンポーネントのリストを表示します。

コンポーネントのドキュメント *(Doc)*

ViewLib メニューおよび、ライブラリで表示されたコンポーネントリストからコンポーネントを選択する場合、特に、様々なメニューにはコメント行（およびキーワード）が表示されます。

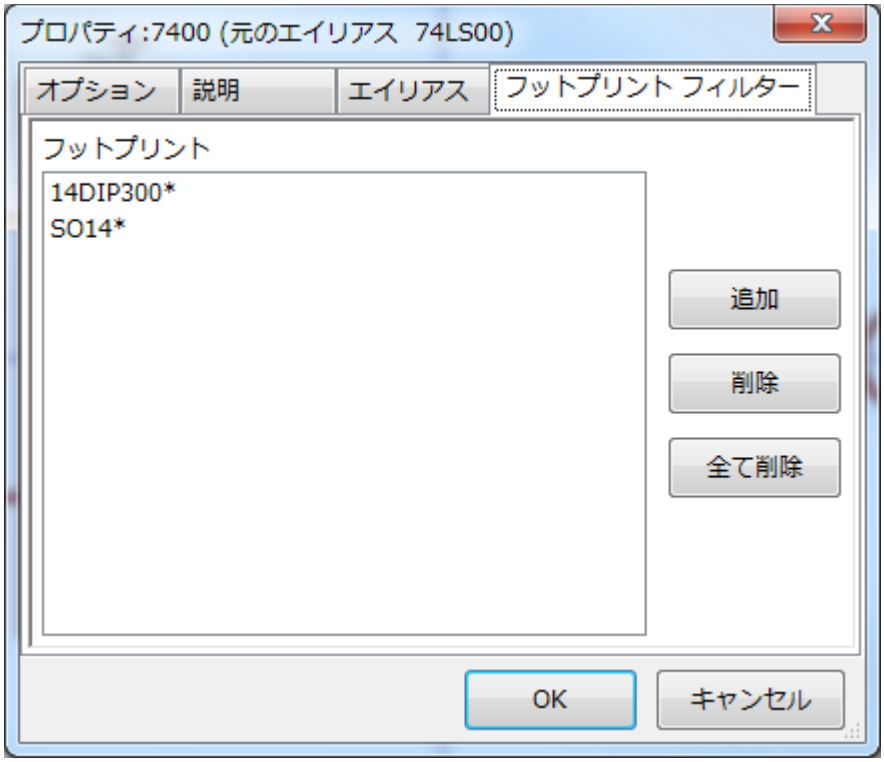
ドキュメントファイルが存在する場合には、回路図エディターソフト内でドキュメントファイルにアクセスが可能であり、コンポーネントを右クリックして表示されるポップアップメニューで利用できます。

関連するドキュメントファイル *(DocFileName)*

利用可能な添付ファイル（ドキュメント、アプリケーション回路）を示します（pdf ファイル、回路図等）

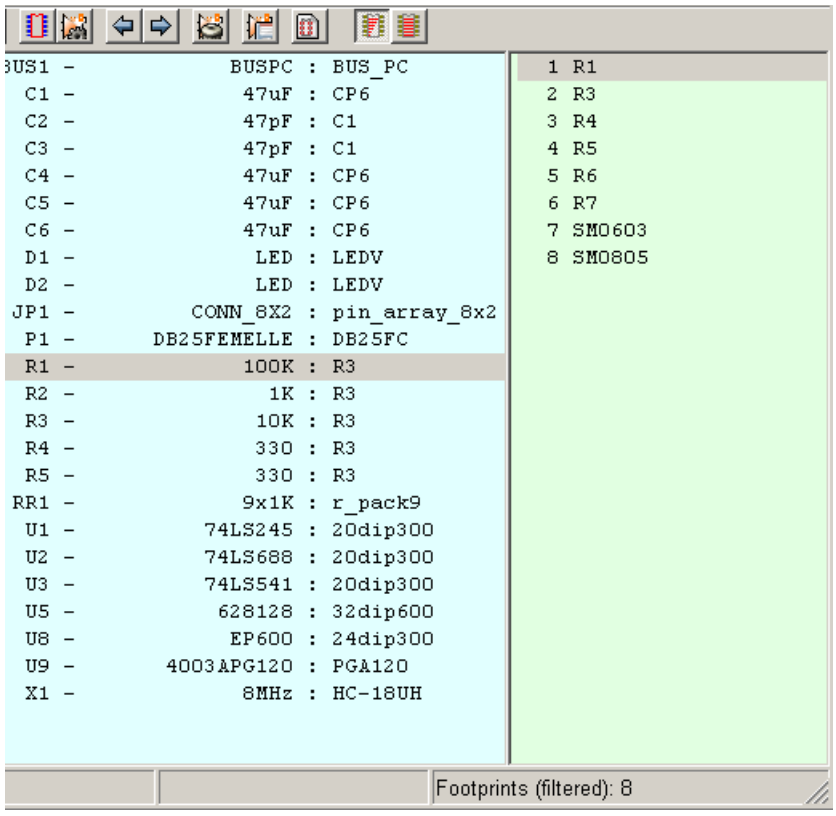
CvPcb のフットプリントフィルタ

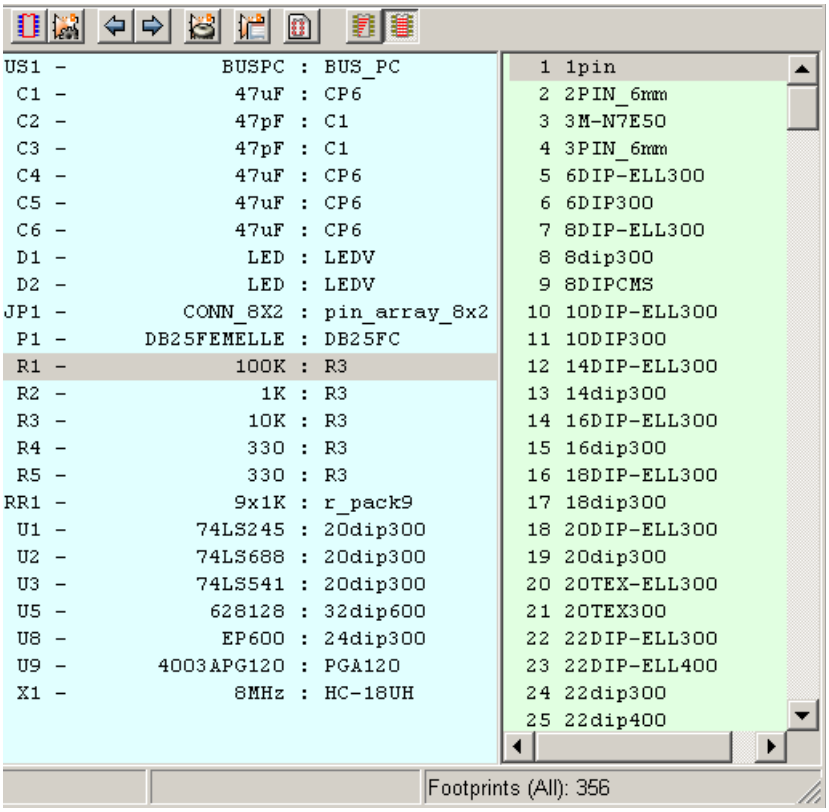
コンポーネントに認めたフットプリントのリストを入力することができます。
このリストは Cvpcb で使われる、認められたフットプリントのみ表示するフィルターの役割をします。空のリストは何もフィルターしません。



ワイルドカード文字が使用できます。
SO14*により、CvPcb は名称が SO14 で始まる全てのフットプリントを示すことができます。
抵抗については R? により R で始まる２文字の全てのフットプリントを示すことができます。

フィルター有り、無しのサンプルを示します。

	フィルター有り

		フィルターなし

12.6 - シンボルライブラリ

頻繁に使用されるシンボルを含んだグラフィックシンボルのライブラリ·ファイルを簡単にコンパイルすることができます。　これはコンポーネント（三角形、AND、OR、ExOR ゲートなどの形状）の作成や、節約と再利用に使うことができます。

これらのファイルは.sym という拡張子を持ち、デフォルトでライブラリディレクトリに保存されます。シンボルは、一般的にそう多くないので、コンポーネントのようにライブラリ内に収集されません。

シンボルの作成、エクスポート

コンポーネントは  ボタンで、シンボルとしてエクスポートすることができます。一般的に 1 つのグラフィックを作成でき、ピンがある場合には、全てのピンを削除することをお勧めします。

シンボルのインポート

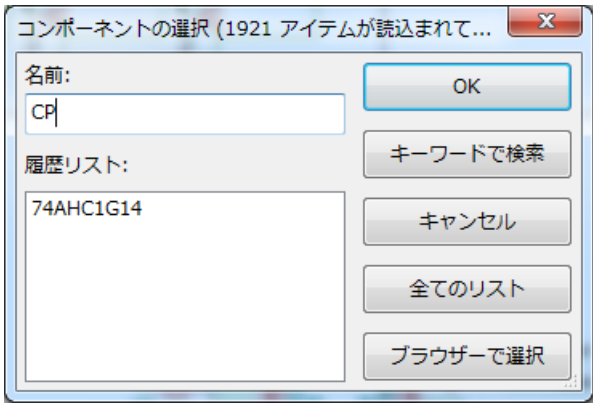
インポートすることで、編集しているコンポーネントにグラフィックを追加することができます。 シンボルは、  ボタンでインポートされます。インポートされたグラフィックは、既存のグラフィックス内で作成されたものとして追加されます。

13 - ライブラリブラウザ

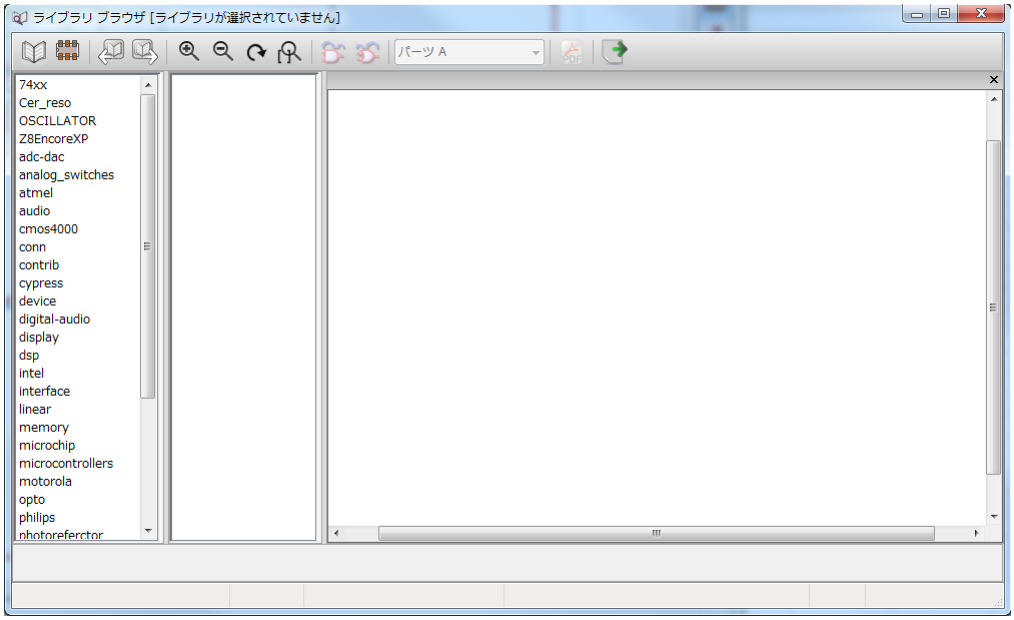
目次

13.1 - はじめに

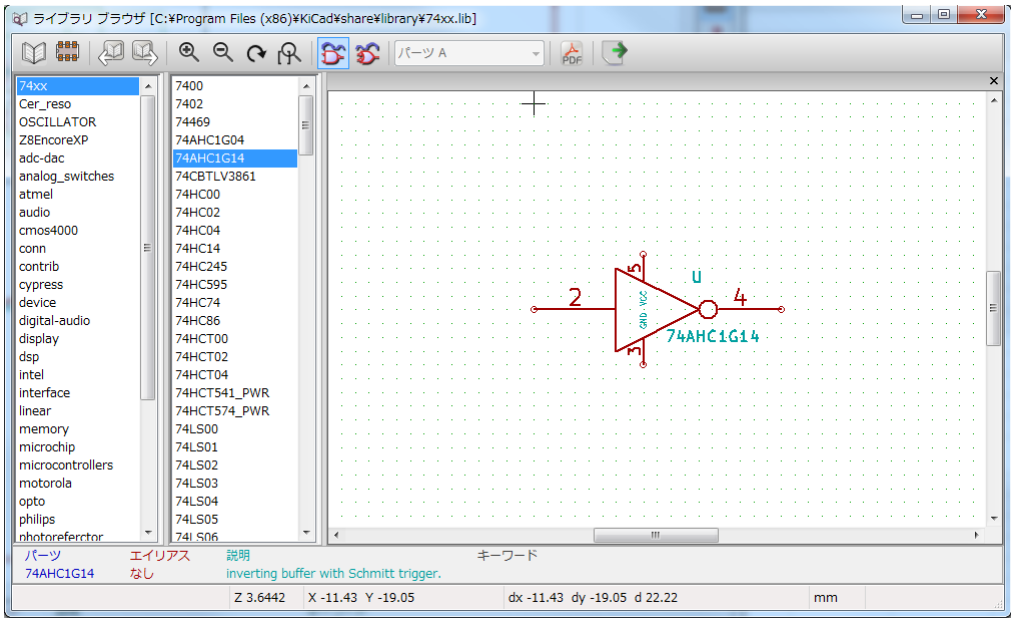
ライブラリブラウザを利用すると、ライブラリの内容をすばやく確認することができます。ライブラリブラウザは、  アイコンか、または右側ツールバーの「コンポーネントの配置」ツールより利用することができます。



13.2 - ライブラリブラウザ - メインウインドウ



ライブラリの内容を確認するには、ウインドウ左側のライブラリを選択する必要があります。利用可能なコンポーネントが２番目のリストに表示されます。

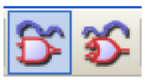
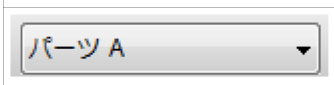


13.3 - ライブラリブラウザ上部ツールバー

ライブラリブラウザの上部に表示されているツールバーを以下に示します。

また、Eeschema の「コンポーネントの配置」ツールより呼びだされた場合のツールバーを以下に示します。

利用可能なコマンドは以下のとおりです。

	一覧より、表示するライブラリを選択します。
	一覧より、表示するコンポーネントを選択します。
	リストで選択されている、１つ前のコンポーネントを表示します。
	リストで選択されている、１つ次のコンポーネントを表示します。
	ズームツールです。
	有効な場合、選択されたコンポーネントの表示を切り替えます(通常パーツまたは"ド・モルガン"変換パーツ)。
	複数部品から構成されるコンポーネントの場合、パーツを選択します。

	一覧より、表示するライブラリを選択します。
	有効な場合、コンポーネントに関連したドキュメントを参照することができます。 Eeschema の「コンポーネントの配置」ツールより呼びだされた場合のみ、この機能が有効になります。
	ライブラリブラウザを閉じ、選択されたコンポーネントを Eeschema 上で配置します。

14 - カスタマイズされたネットリストや BOM の生成

目次

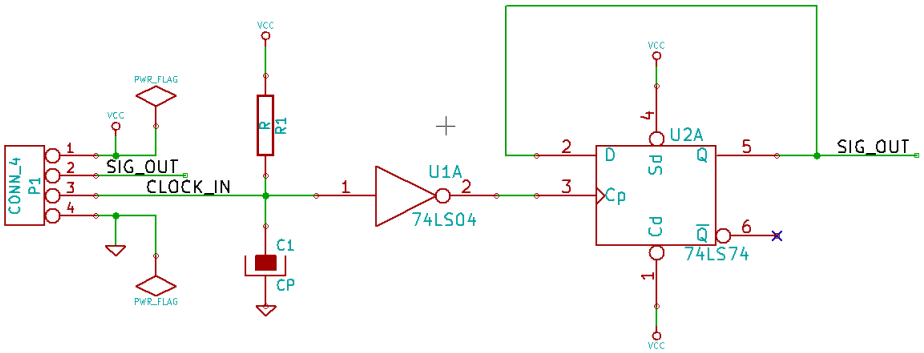
14.1 - 中間ネットリスト

部品表ファイルとネットリストファイルは、Eeschema が生成する中間ネットリストから変換されます。

このファイルは XML フォーマットで書かれており、中間ネットリストと呼ばれています。この中間ネットリストはただのネットリストではありません。部品表やさまざまなレポートを生成するため、設計中の基板に関する大量のデータが含まれているのです。

出力するファイル(部品表かネットリスト)次第で、中間ネットリストの利用される部分が変わってきます。

回路図サンプル



中間ネットリストのサンプル

上記回路図に対応する中間ネットリスト(XML 文法を利用しています)を以下に示します。

```
<?xml version="1.0" encoding="utf-8"?>
<export version="D">
  <design>
    <source>F:\kicad_aux\netlist_test\netlist_test.sch</source>
    <date>29/08/2010 20:35:21</date>
    <tool>eeschema (2010-08-28 BZR 2458)-unstable</tool>
  </design>
  <components>
    <comp ref="P1">
      <value>CONN_4</value>
      <libsource lib="conn" part="CONN_4"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E2141</tstamp>
    </comp>
    <comp ref="U2">
      <value>74LS74</value>
      <libsource lib="74xx" part="74LS74"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E20BA</tstamp>
    </comp>
    <comp ref="U1">
      <value>74LS04</value>
      <libsource lib="74xx" part="74LS04"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E20A6</tstamp>
    </comp>
    <comp ref="C1">
      <value>CP</value>
      <libsource lib="device" part="CP"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E2094</tstamp>
    </comp>
    <comp ref="R1">
```

```
    <value>R</value>
    <libsource lib="device" part="R"/>
    <sheetpath names="/" tstamps="/"/>
    <tstamp>4C6E208A</tstamp>
  </comp>
</components>
<libparts>
  <libpart lib="device" part="C">
    <description>Condensateur non polarise</description>
    <footprints>
      <fp>SM*</fp>
      <fp>C?</fp>
      <fp>C1-1</fp>
    </footprints>
    <fields>
      <field name="Reference">C</field>
      <field name="Value">C</field>
    </fields>
    <pins>
      <pin num="1" name="~" type="passive"/>
      <pin num="2" name="~" type="passive"/>
    </pins>
  </libpart>
  <libpart lib="device" part="R">
    <description>Resistance</description>
    <footprints>
      <fp>R?</fp>
      <fp>SM0603</fp>
      <fp>SM0805</fp>
      <fp>R?-*</fp>
      <fp>SM1206</fp>
    </footprints>
    <fields>
      <field name="Reference">R</field>
      <field name="Value">R</field>
    </fields>
    <pins>
      <pin num="1" name="~" type="passive"/>
      <pin num="2" name="~" type="passive"/>
    </pins>
  </libpart>
  <libpart lib="conn" part="CONN_4">
    <description>Symbole general de connecteur</description>
    <fields>
      <field name="Reference">P</field>
      <field name="Value">CONN_4</field>
    </fields>
    <pins>
      <pin num="1" name="P1" type="passive"/>
      <pin num="2" name="P2" type="passive"/>
      <pin num="3" name="P3" type="passive"/>
      <pin num="4" name="P4" type="passive"/>
    </pins>
  </libpart>
  <libpart lib="74xx" part="74LS04">
    <description>Hex Inverseur</description>
    <fields>
      <field name="Reference">U</field>
      <field name="Value">74LS04</field>
    </fields>
    <pins>
      <pin num="1" name="~" type="input"/>
      <pin num="2" name="~" type="output"/>
      <pin num="3" name="~" type="input"/>
      <pin num="4" name="~" type="output"/>
      <pin num="5" name="~" type="input"/>
      <pin num="6" name="~" type="output"/>
      <pin num="7" name="GND" type="power_in"/>
      <pin num="8" name="~" type="output"/>
      <pin num="9" name="~" type="input"/>
      <pin num="10" name="~" type="output"/>
```

```
      <pin num="11" name="~" type="input"/>
      <pin num="12" name="~" type="output"/>
      <pin num="13" name="~" type="input"/>
      <pin num="14" name="VCC" type="power_in"/>
    </pins>
  </libpart>
  <libpart lib="74xx" part="74LS74">
    <description>Dual D FlipFlop, Set &amp;amp; Reset</description>
    <docs>74xx/74hc_hct74.pdf</docs>
    <fields>
      <field name="Reference">U</field>
      <field name="Value">74LS74</field>
    </fields>
    <pins>
      <pin num="1" name="Cd" type="input"/>
      <pin num="2" name="D" type="input"/>
      <pin num="3" name="Cp" type="input"/>
      <pin num="4" name="Sd" type="input"/>
      <pin num="5" name="Q" type="output"/>
      <pin num="6" name="~Q" type="output"/>
      <pin num="7" name="GND" type="power_in"/>
      <pin num="8" name="~Q" type="output"/>
      <pin num="9" name="Q" type="output"/>
      <pin num="10" name="Sd" type="input"/>
      <pin num="11" name="Cp" type="input"/>
      <pin num="12" name="D" type="input"/>
      <pin num="13" name="Cd" type="input"/>
      <pin num="14" name="VCC" type="power_in"/>
    </pins>
  </libpart>
</libparts>
<libraries>
  <library logical="device">
    <uri>F:\kicad\share\library\device.lib</uri>
  </library>
  <library logical="conn">
    <uri>F:\kicad\share\library\conn.lib</uri>
  </library>
  <library logical="74xx">
    <uri>F:\kicad\share\library\74xx.lib</uri>
  </library>
</libraries>
<nets>
  <net code="1" name="GND">
    <node ref="U1" pin="7"/>
    <node ref="C1" pin="2"/>
    <node ref="U2" pin="7"/>
    <node ref="P1" pin="4"/>
  </net>
  <net code="2" name="VCC">
    <node ref="R1" pin="1"/>
    <node ref="U1" pin="14"/>
    <node ref="U2" pin="4"/>
    <node ref="U2" pin="1"/>
    <node ref="U2" pin="14"/>
    <node ref="P1" pin="1"/>
  </net>
  <net code="3" name="">
    <node ref="U2" pin="6"/>
  </net>
  <net code="4" name="">
    <node ref="U1" pin="2"/>
    <node ref="U2" pin="3"/>
  </net>
  <net code="5" name="/SIG_OUT">
    <node ref="P1" pin="2"/>
    <node ref="U2" pin="5"/>
    <node ref="U2" pin="2"/>
  </net>
  <net code="6" name="/CLOCK_IN">
    <node ref="R1" pin="2"/>
```

```
      <node ref="C1" pin="1"/>
      <node ref="U1" pin="1"/>
      <node ref="P1" pin="3"/>
    </net>
  </nets>
</export>
```

14.2 - 新しいネットリスト形式への変換

部品表ファイルや他形式のファイルへ変換するために、前処理として中間ネットリストファイルの内容を抽出していきます。これはテキスト形式のファイル操作であるため、この中間ネットリストからの情報抽出は Python や XSLT など、XML を入力とできる処理系を利用して簡単にプログラムを書くことができます。

XSLT はそれ自身が XML 言語で書かれる、XML ファイルの処理に最適な言語です。xsltproc と呼ばれるフリーソフトがあり、ダウンロード、インストールすることで得られます。中間ネットリストと入力されたファイルを変換するためのスタイルシートより、結果をファイルへ保存します。xsltproc を使用するためには、XSLT による変換処理のためのスタイルシートが必要となります。これら全ての変換プロセスは、Eeschema により制御され、一度設定しておけば何度でも実行することができます。

14.3 - XSLT のアプローチ

XSL 変換(XSL Transformations : XSLT)に関するドキュメントは、下記より参照することが出来ます：

`http://www.w3.org/TR/xslt`

Pads-Pcb 形式ネットリストファイルの生成

pads-pcb 形式のネットリストは、下記の 2 セクションより構成されています。

- フットプリントの一覧
- ネットリスト：ネット情報によりグループ化された、パッド情報

以下に、中間ネットリストから pads-pcb 形式へ変換するためのスタイルシートを掲載します。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--XSL style sheet to EESCHEMA Generic Netlist Format to PADS netlist format
    Copyright (C) 2010, SoftPLC Corporation.
    GPL v2.

    How to use:
        https://lists.launchpad.net/kicad-developers/msg05157.html
-->

<!DOCTYPE xsl:stylesheet [
  <!ENTITY nl  "&#xd;&#xa;"> <!--new line CR, LF -->
]>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes" indent="no"/>

<xsl:template match="/export">
    <xsl:text>*PADS-PCB*&nl;*PART*&nl;</xsl:text>
    <xsl:apply-templates select="components/comp"/>
    <xsl:text>&nl;*NET*&nl;</xsl:text>
    <xsl:apply-templates select="nets/net"/>
    <xsl:text>*END*&nl;</xsl:text>
</xsl:template>

<!-- for each component -->
<xsl:template match="comp">
    <xsl:text> </xsl:text>
    <xsl:value-of select="@ref"/>
    <xsl:text> </xsl:text>
    <xsl:choose>
        <xsl:when test = "footprint != '' ">
            <xsl:apply-templates select="footprint"/>
        </xsl:when>
        <xsl:otherwise>
```

```
            <xsl:text>unknown</xsl:text>
        </xsl:otherwise>
    </xsl:choose>
    <xsl:text>&nl;</xsl:text>
</xsl:template>

<!-- for each net -->
<xsl:template match="net">
    <!-- nets are output only if there is more than one pin in net -->
    <xsl:if test="count(node)>1">
        <xsl:text>*SIGNAL* </xsl:text>
        <xsl:choose>
            <xsl:when test = "@name != '' ">
                <xsl:value-of select="@name"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:text>N-</xsl:text>
                <xsl:value-of select="@code"/>
            </xsl:otherwise>
        </xsl:choose>
        <xsl:text>&nl;</xsl:text>
        <xsl:apply-templates select="node"/>
    </xsl:if>
</xsl:template>

<!-- for each node -->
<xsl:template match="node">
    <xsl:text> </xsl:text>
    <xsl:value-of select="@ref"/>
    <xsl:text>.</xsl:text>
    <xsl:value-of select="@pin"/>
    <xsl:text>&nl;</xsl:text>
</xsl:template>
</xsl:stylesheet>
```

xsltproc を実行し得られた、pads-pcb の用のネットリストファイルを以下に示します。

```
*PADS-PCB*
*PART*
 P1 unknown
 U2 unknown
 U1 unknown
 C1 unknown
 R1 unknown

*NET*
*SIGNAL* GND
 U1.7
 C1.2
 U2.7
 P1.4
*SIGNAL* VCC
 R1.1
 U1.14
 U2.4
 U2.1
 U2.14
 P1.1
*SIGNAL* N-4
 U1.2
 U2.3
*SIGNAL* /SIG_OUT
 P1.2
```

```
 U2.5
 U2.2
*SIGNAL* /CLOCK_IN
 R1.2
 C1.1
 U1.1
 P1.3
*END*
```

この変換は、次のコマンドラインにより実行することができます：

kicad/bin/xsltproc.exe -o test.net kicad/bin/plugins/netlist_form_pads-pcb.xsl test.tmp

Cadstar 形式のネットリストファイルの生成

Cadstar 形式のネットリストは、下記の 2 セクションで構成されています。

- フットプリントの一覧
- ネットリスト：ネット情報によりグループ化された、パッド情報

以下に変換するためのスタイルシートを掲載します。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--XSL style sheet to EESCHEMA Generic Netlist Format to CADSTAR netlist format
    Copyright (C) 2010, Jean-Pierre Charras.
    Copyright (C) 2010, SoftPLC Corporation.
    GPL v2.

<!DOCTYPE xsl:stylesheet [
  <!ENTITY nl  "&#xd;&#xa;"> <!--new line CR, LF -->
]>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes" indent="no"/>

<!-- Netlist header -->
<xsl:template match="/export">
    <xsl:text>.HEA&nl;</xsl:text>
    <xsl:apply-templates select="design/date"/>  <!-- Generate line .TIM <time>
-->
    <xsl:apply-templates select="design/tool"/>  <!-- Generate line .APP <eeschema
version> -->
    <xsl:apply-templates select="components/comp"/>  <!-- Generate list of
components -->
    <xsl:text>&nl;&nl;</xsl:text>
    <xsl:apply-templates select="nets/net"/>          <!-- Generate list of nets
and connections -->
    <xsl:text>&nl;.END&nl;</xsl:text>
</xsl:template>

 <!-- Generate line .TIM 20/08/2010 10:45:33 -->
<xsl:template match="tool">
    <xsl:text>.APP "</xsl:text>
    <xsl:apply-templates/>
    <xsl:text>"&nl;</xsl:text>
</xsl:template>

 <!-- Generate line .APP "eeschema (2010-08-17 BZR 2450)-unstable" -->
<xsl:template match="date">
    <xsl:text>.TIM </xsl:text>
    <xsl:apply-templates/>
    <xsl:text>&nl;</xsl:text>
</xsl:template>
```

```
<!-- for each component -->
<xsl:template match="comp">
    <xsl:text>.ADD_COM </xsl:text>
    <xsl:value-of select="@ref"/>
    <xsl:text> </xsl:text>
    <xsl:choose>
        <xsl:when test = "value != '' ">
            <xsl:text>"</xsl:text> <xsl:apply-templates select="value"/>
<xsl:text>"</xsl:text>
        </xsl:when>
        <xsl:otherwise>
            <xsl:text>""</xsl:text>
        </xsl:otherwise>
    </xsl:choose>
    <xsl:text>&nl;</xsl:text>
</xsl:template>

<!-- for each net -->
<xsl:template match="net">
    <!-- nets are output only if there is more than one pin in net -->
    <xsl:if test="count(node)>1">
    <xsl:variable name="netname">
        <xsl:text>"</xsl:text>
        <xsl:choose>
            <xsl:when test = "@name != '' ">
                <xsl:value-of select="@name"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:text>N-</xsl:text>
                <xsl:value-of select="@code"/>
        </xsl:otherwise>
        </xsl:choose>
        <xsl:text>"&nl;</xsl:text>
        </xsl:variable>
        <xsl:apply-templates select="node" mode="first"/>
        <xsl:value-of select="$netname"/>
        <xsl:apply-templates select="node" mode="others"/>
    </xsl:if>
</xsl:template>

<!-- for each node -->
<xsl:template match="node" mode="first">
    <xsl:if test="position()=1">
       <xsl:text>.ADD_TER </xsl:text>
    <xsl:value-of select="@ref"/>
    <xsl:text>.</xsl:text>
    <xsl:value-of select="@pin"/>
    <xsl:text> </xsl:text>
    </xsl:if>
</xsl:template>

<xsl:template match="node" mode="others">
    <xsl:choose>
        <xsl:when test='position()=1'>
        </xsl:when>
        <xsl:when test='position()=2'>
           <xsl:text>.TER     </xsl:text>
        </xsl:when>
        <xsl:otherwise>
           <xsl:text>         </xsl:text>
        </xsl:otherwise>
    </xsl:choose>
    <xsl:if test="position()>1">
        <xsl:value-of select="@ref"/>
```

```
        <xsl:text>.</xsl:text>
        <xsl:value-of select="@pin"/>
        <xsl:text>&nl;</xsl:text>
    </xsl:if>
</xsl:template>

</xsl:stylesheet>
```

出力ファイル。

```
.HEA
.TIM 21/08/2010 08:12:08
.APP "eeschema (2010-08-09 BZR 2439)-unstable"
.ADD_COM P1 "CONN_4"
.ADD_COM U2 "74LS74"
.ADD_COM U1 "74LS04"
.ADD_COM C1 "CP"
.ADD_COM R1 "R"


.ADD_TER U1.7 "GND"
.TER     C1.2
         U2.7
         P1.4
.ADD_TER R1.1 "VCC"
.TER     U1.14
         U2.4
         U2.1
         U2.14
         P1.1
.ADD_TER U1.2 "N-4"
.TER     U2.3
.ADD_TER P1.2 "/SIG_OUT"
.TER     U2.5
         U2.2
.ADD_TER R1.2 "/CLOCK_IN"
.TER     C1.1
         U1.1
         P1.3

.END
```

OrcadPCB2 形式ネットリストファイルの生成

このフォーマットは、フットプリントの一覧のみで構成されています。それぞれのフットプリントは接続されるネットの情報を含みます。

変換を行うためのスタイルシートファイルを、以下に示します。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--XSL style sheet to EESCHEMA Generic Netlist Format to CADSTAR netlist format
   Copyright (C) 2010, SoftPLC Corporation.
   GPL v2.

   How to use:
      https://lists.launchpad.net/kicad-developers/msg05157.html
-->

<!DOCTYPE xsl:stylesheet [
  <!ENTITY nl  "&#xd;&#xa;"> <!--new line CR, LF -->
]>
```

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes" indent="no"/>

<!--
   Netlist header
   Creates the entire netlist
   (can be seen as equivalent to main function in C
-->
<xsl:template match="/export">
   <xsl:text>( { EESchema Netlist Version 1.1  </xsl:text>
   <!-- Generate line .TIM <time> -->
   <xsl:apply-templates select="design/date"/>
   <!-- Generate line eeschema version ... -->
   <xsl:apply-templates select="design/tool"/>
   <xsl:text>}&nl;</xsl:text>

   <!-- Generate the list of components -->
   <xsl:apply-templates select="components/comp"/>  <!-- Generate list of components -->

   <!-- end of file -->
   <xsl:text>)&nl;*&nl;</xsl:text>
</xsl:template>

<!--
   Generate id in header like "eeschema (2010-08-17 BZR 2450)-unstable"
-->
<xsl:template match="tool">
   <xsl:apply-templates/>
</xsl:template>

<!--
   Generate date in header like "20/08/2010 10:45:33"
-->
<xsl:template match="date">
   <xsl:apply-templates/>
   <xsl:text>&nl;</xsl:text>
</xsl:template>

<!--
   This template read each component
   (path = /export/components/comp)
   creates lines:
    ( 3EBF7DBD $noname U1 74LS125
     ... pin list ...
     )
   and calls "create_pin_list" template to build the pin list
-->
<xsl:template match="comp">
   <xsl:text> ( </xsl:text>
   <xsl:choose>
      <xsl:when test = "tstamp != '' ">
```

```
        <xsl:apply-templates select="tstamp"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:text>00000000</xsl:text>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text> </xsl:text>
    <xsl:choose>
      <xsl:when test = "footprint != '' ">
        <xsl:apply-templates select="footprint"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:text>$noname</xsl:text>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text> </xsl:text>
    <xsl:value-of select="@ref"/>
    <xsl:text> </xsl:text>
    <xsl:choose>
      <xsl:when test = "value != '' ">
        <xsl:apply-templates select="value"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:text>"~"</xsl:text>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text>&nl;</xsl:text>
    <xsl:call-template name="Search_pin_list" >
      <xsl:with-param name="cmplib_id" select="libsource/@part"/>
      <xsl:with-param name="cmp_ref" select="@ref"/>
    </xsl:call-template>
    <xsl:text> )&nl;</xsl:text>
</xsl:template>

<!--
    This template search for a given lib component description in list
    lib component descriptions are in /export/libparts,
    and each description start at ./libpart
    We search here for the list of pins of the given component
    This template has 2 parameters:
      "cmplib_id" (reference in libparts)
      "cmp_ref"   (schematic reference of the given component)
-->
<xsl:template name="Search_pin_list" >
    <xsl:param name="cmplib_id" select="0" />
    <xsl:param name="cmp_ref" select="0" />
      <xsl:for-each select="/export/libparts/libpart">
        <xsl:if test = "@part = $cmplib_id ">
          <xsl:apply-templates name="build_pin_list" select="pins/pin">
            <xsl:with-param name="cmp_ref" select="$cmp_ref"/>
          </xsl:apply-templates>
        </xsl:if>
```

```
      </xsl:for-each>
</xsl:template>


<!--
   This template writes the pin list of a component
   from the pin list of the library description
   The pin list from library description is something like
        <pins>
          <pin num="1" type="passive"/>
          <pin num="2" type="passive"/>
        </pins>
   Output pin list is ( <pin num> <net name> )
   something like
         ( 1 VCC )
         ( 2 GND )
-->
<xsl:template name="build_pin_list" match="pin">
   <xsl:param name="cmp_ref" select="0" />

   <!-- write pin numner and separator -->
   <xsl:text>  ( </xsl:text>
   <xsl:value-of select="@num"/>
   <xsl:text> </xsl:text>

   <!-- search net name in nets section and write it: -->
   <xsl:variable name="pinNum" select="@num" />
   <xsl:for-each select="/export/nets/net">
      <!-- net name is output only if there is more than one pin in net
           else use "?" as net name, so count items in this net
      -->
      <xsl:variable name="pinCnt" select="count(node)" />
      <xsl:apply-templates name="Search_pin_netname" select="node">
         <xsl:with-param name="cmp_ref" select="$cmp_ref"/>
         <xsl:with-param name="pin_cnt_in_net" select="$pinCnt"/>
         <xsl:with-param name="pin_num"> <xsl:value-of select="$pinNum"/>
         </xsl:with-param>
      </xsl:apply-templates>
   </xsl:for-each>

   <!-- close line -->
   <xsl:text> )&nl;</xsl:text>
</xsl:template>

<!--
   This template writes the pin netname of a given pin of a given component
   from the nets list
   The nets list description is something like
     <nets>
      <net code="1" name="GND">
       <node ref="J1" pin="20"/>
          <node ref="C2" pin="2"/>
```

```
    </net>
    <net code="2" name="">
      <node ref="U2" pin="11"/>
    </net>
  </nets>
  This template has 2 parameters:
    "cmp_ref"   (schematic reference of the given component)
    "pin_num"   (pin number)
-->

<xsl:template name="Search_pin_netname" match="node">
  <xsl:param name="cmp_ref" select="0" />
  <xsl:param name="pin_num" select="0" />
  <xsl:param name="pin_cnt_in_net" select="0" />

  <xsl:if test = "@ref = $cmp_ref ">
    <xsl:if test = "@pin = $pin_num">
    <!-- net name is output only if there is more than one pin in net
        else use "?" as net name
    -->
      <xsl:if test = "$pin_cnt_in_net>1">
        <xsl:choose>
          <!-- if a net has a name, use it,
             else build a name from its net code
          -->
          <xsl:when test = "../@name != '' ">
            <xsl:value-of select="../@name"/>
          </xsl:when>
          <xsl:otherwise>
            <xsl:text>$N-0</xsl:text><xsl:value-of select="../@code"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:if>
      <xsl:if test = "$pin_cnt_in_net &lt;2">
        <xsl:text>?</xsl:text>
      </xsl:if>
    </xsl:if>
  </xsl:if>

</xsl:template>

</xsl:stylesheet>
```

出力ファイル。

```
( { EESchema Netlist Version 1.1  29/08/2010 21:07:51
eeschema (2010-08-28 BZR 2458)-unstable}
 ( 4C6E2141 $noname P1 CONN_4
  (  1 VCC )
  (  2 /SIG_OUT )
  (  3 /CLOCK_IN )
  (  4 GND )
```

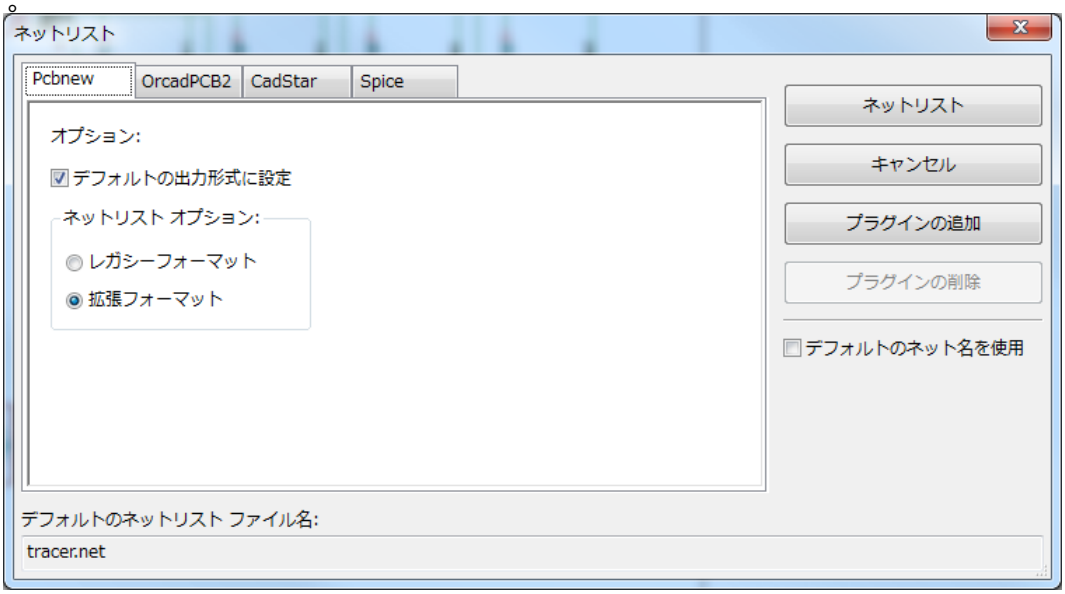
```
 )
 ( 4C6E20BA $noname U2 74LS74
  (  1 VCC )
  (  2 /SIG_OUT )
  (  3 N-04 )
  (  4 VCC )
  (  5 /SIG_OUT )
  (  6 ? )
  (  7 GND )
  (  14 VCC )
 )
 ( 4C6E20A6 $noname U1 74LS04
  (  1 /CLOCK_IN )
  (  2 N-04 )
  (  7 GND )
  (  14 VCC )
 )
 ( 4C6E2094 $noname C1 CP
  (  1 /CLOCK_IN )
  (  2 GND )
 )
 ( 4C6E208A $noname R1 R
  (  1 VCC )
  (  2 /CLOCK_IN )
 )
)
*
```

Eeschema プラグインインタフェイス

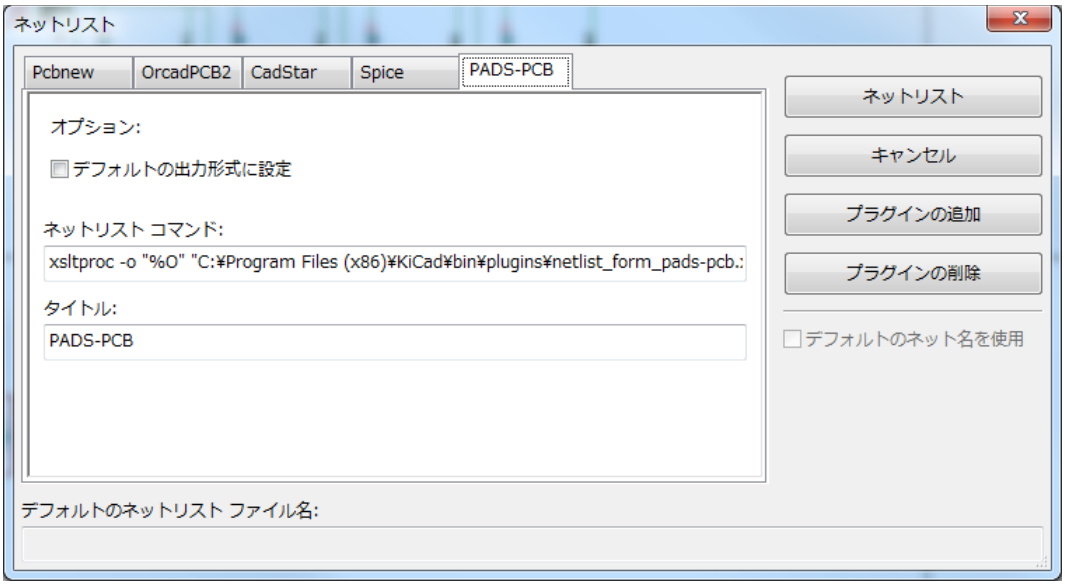
中間ネットリストの変換は Eeschema の中で自動的に実行させることができます。

ダイアログウインドウの初期化

新しいネットリストプラグインをユーザ・インタフェースへ追加するには、「プラグインの追加」タブを利用します。



PadsPcb タブの設定は、下記のように見えます：



プラグインの設定

Eeschema のプラグインの設定ダイアログでは、下記の情報が必要になります：

- タイトル: ネットリストフォーマットの名前など
- 変換を行うためのコマンドライン

ネットリストボタンをクリックすると、次のように実行されます。

1. Eeschema は test.xml のように*.xml の形式で中間ネットリストを生成します。
2. Eeschema は test.xml をプラグインへ入力し、test.net を生成します。

コマンドラインからのネットリストファイル生成

xsltproc.exe を利用し中間ネットリストへスタイルシートを適用する場合、下記のコマンドにより xsltproc.exe が実行されます。

xsltproc.exe -o <output filename> < style-sheet filename> <input XML file to convert>

Windows 環境で Kicad を利用している場合のコマンドラインは以下のようになります。

f:/kicad/bin/xsltproc.exe -o “%O” f:/kicad/bin/plugins/netlist_form_pads-pcb.xsl “%I”

Linux 環境の場合のコマンドを以下に示します。

xsltproc -o “%O” /usr/local/kicad/bin/plugins/netlist_form_pads-pcb.xsl “%I”

netlist_form_pads-pcb.xsl には、適用するスタイルシートのファイル名が入ります。このスタイルシートのパスにスペースが入っている場合には、ダブルクオーテーションで囲むのを忘れないようにしてください。

サポートしているパラメータを下記に示します：

- %B => base filename and path of selected output file, minus path and extension.
- %I => 入力ファイル（中間ネットリストファイル）の完全なパスとファイル名を指定します。
- %O => 出力ファイルの完全なパスとファイル名を指定します。

%I は実際の中間ネットリストファイル名へ置換されます。

%O は実際の出力ファイル名へ置換され、最終的なネットリストファイルとなります。コマンドラインの例を次に示します：

コマンドラインフォーマット：**xsltproc** の例

xsltproc のコマンドラインフォーマットは、下記のようになります：

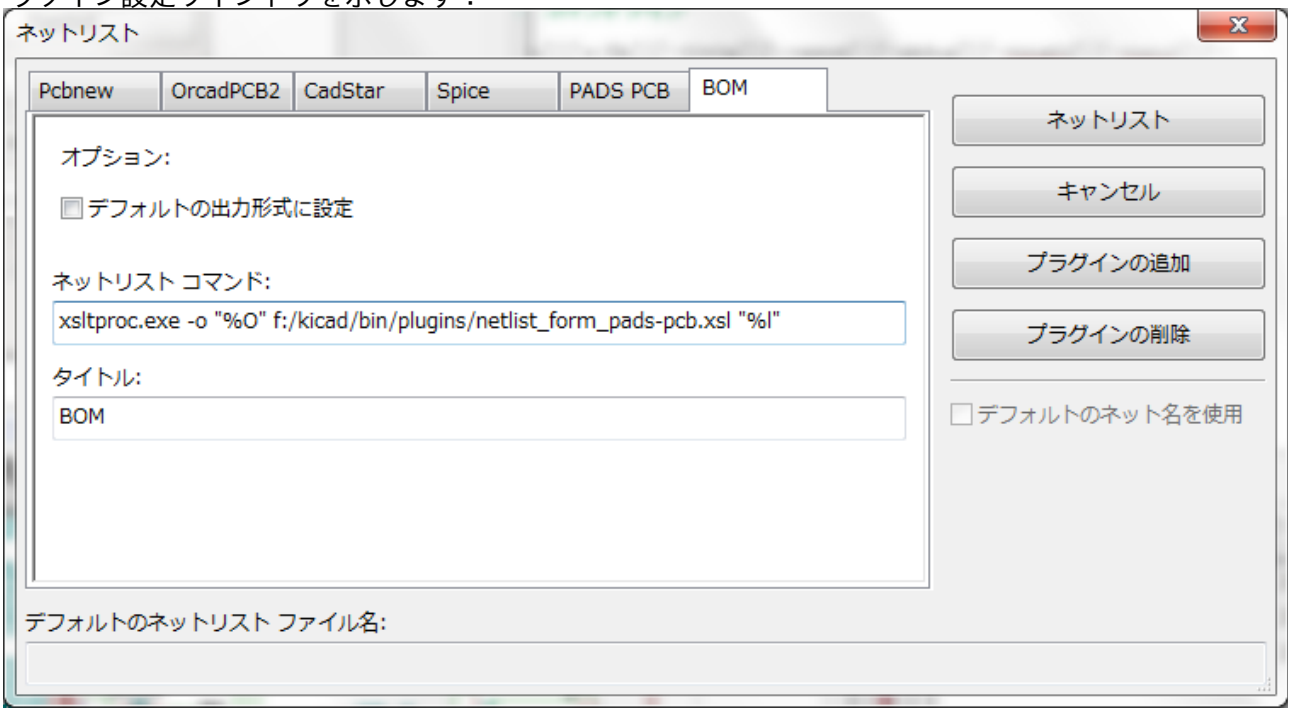
<path or xsltproc> xsltproc <xsltproc parameters>
Wiindows 環境の場合
f:/kicad/bin/xsltproc.exe -o %O f:/kicad/bin/plugins/netlist_form_pads-pcb.xsl %I

Linux 環境の場合
xsltproc -o %O /usr/local/kicad/bin/plugins/netlist_form_pads-pcb.xsl %I

上記は、xsltproc が Windows 環境下で kicad/bin 以下にインストールされていると仮定したものです。

BOM の生成

中間ネットリストファイルは、使用されているコンポーネントのすべての情報を含んでいるため、ここから BOM を生成することができます。以下に BOM を生成サせるための、Windows（Linux）環境下でのプラグイン設定ウインドウを示します：



bom2csv.xsl のパスは、システムによって異なります。現状で最適な BOM を生成する XSLT スタイルシートは、ここでは **bom2csv.xsl** とします。必要に応じて自由に変更することができ、また自身でプラグインを開発する際の参考になります。

14.4 - コマンドラインフォーマット：python スクリプトの例

python を使用した場合のコマンドラインフォーマットは、下記のようになります：
python <script file name> <input filename> <output filename>
Wiindows 環境の場合
python.exe f:/kicad/python/my_python_script.py "%I" "%O"

Linux 環境の場合
python /usr/local/kicad/python/my_python_script.py "%I" "%O"

あなたの PC へ python がインストールされている必要があります。

14.5 - 中間ネットリストファイルの構造

ネットリストファイルの例を次に示します。

```
<?xml version="1.0" encoding="utf-8"?>
<export version="D">
  <design>
    <source>F:\kicad_aux\netlist_test\netlist_test.sch</source>
```

```
    <date>29/08/2010 21:07:51</date>
    <tool>eeschema (2010-08-28 BZR 2458)-unstable</tool>
  </design>
  <components>
    <comp ref="P1">
      <value>CONN_4</value>
      <libsource lib="conn" part="CONN_4"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E2141</tstamp>
    </comp>
    <comp ref="U2">
      <value>74LS74</value>
      <libsource lib="74xx" part="74LS74"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E20BA</tstamp>
    </comp>
    <comp ref="U1">
      <value>74LS04</value>
      <libsource lib="74xx" part="74LS04"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E20A6</tstamp>
    </comp>
    <comp ref="C1">
      <value>CP</value>
      <libsource lib="device" part="CP"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E2094</tstamp>
    <comp ref="R1">
      <value>R</value>
      <libsource lib="device" part="R"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E208A</tstamp>
    </comp>
  </components>
  <libparts/>
  <libraries/>
  <nets>
    <net code="1" name="GND">
      <node ref="U1" pin="7"/>
      <node ref="C1" pin="2"/>
      <node ref="U2" pin="7"/>
      <node ref="P1" pin="4"/>
    </net>
    <net code="2" name="VCC">
      <node ref="R1" pin="1"/>
      <node ref="U1" pin="14"/>
      <node ref="U2" pin="4"/>
      <node ref="U2" pin="1"/>
      <node ref="U2" pin="14"/>
      <node ref="P1" pin="1"/>
    </net>
    <net code="3" name="">
      <node ref="U2" pin="6"/>
    </net>
    <net code="4" name="">
      <node ref="U1" pin="2"/>
      <node ref="U2" pin="3"/>
    </net>
    <net code="5" name="/SIG_OUT">
      <node ref="P1" pin="2"/>
      <node ref="U2" pin="5"/>
      <node ref="U2" pin="2"/>
    </net>
    <net code="6" name="/CLOCK_IN">
      <node ref="R1" pin="2"/>
      <node ref="C1" pin="1"/>
      <node ref="U1" pin="1"/>
      <node ref="P1" pin="3"/>
    </net>
  </nets>
</export>
```

通常のネットリストファイルの構造

中間ネットリストファイルは、次の 5 セクションで構成されています。

- ヘッダーセクション
- コンポーネントセクション
- ライブラリパーツセクション
- ライブラリセクション
- ネットセクション

このファイルは<export>タグで囲まれたものとなります。

```
<export version="D">
 ...
</export>
```

ヘッダーセクション

このヘッダは<design>タグで囲まれます。

```
  <design>
    <source>F:\kicad_aux\netlist_test\netlist_test.sch</source>
    <date>21/08/2010 08:12:08</date>
    <tool>eeschema (2010-08-09 BZR 2439)-unstable</tool>
  </design>
```

このセクションはコメントセクションとして捉えることができます。

コンポーネントセクション

このコンポーネントセクションは<components>タグで囲まれたものとなります。

```
  <components>
    <comp ref="P1">
      <value>CONN_4</value>
      <libsource lib="conn" part="CONN_4"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E2141</tstamp>
    </comp>
  </components>
```

このセクションには、回路図中で使用されているコンポーネントの一覧が含まれます。それぞれのコンポーネントは、次のように記載されます。

```
    <comp ref="P1">
      <value>CONN_4</value>
      <libsource lib="conn" part="CONN_4"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamp>4C6E2141</tstamp>
    </comp>
```

libsource	そのコンポーネントが含まれているライブラリ名
part	ライブラリ中に登録されているコンポーネント名
sheetpath	階層内のシートのパス（回路図階層全体の中で、その回路図シートの位置を明確にするために利用される）
tstamps (time stamps)	回路図ファイルのタイムスタンプ

tstamp (time stamp)	コンポーネントのタイムスタンプ

コンポーネントのタイムスタンプに関する注意

ボード設計時、ネットリストからコンポーネントを識別する際、タイムスタンプ情報はコンポーネント固有の情報となります。
一方で、Kicad はコンポーネントを基板上の対応するフットプリントから識別する方法を用意しています。これは、回路図プロジェクト中のコンポー ネントが再アノテーションされることと、コンポーネントとフットプリント間の結びつき情報を破壊しないために用意されたものです。

タイムスタンプはそれぞれのコンポーネントや回路図プロジェクト内のシートにおいて識別するための独自のものです。しかしながら、複雑な構造体で同じシートが複数回参照される場合などにおいては、同じタイムスタンプを持つコンポーネントが存在することとなってしまいます。

このような複雑な階層構造を持つシートでは、シートのパス情報を利用して個別のタイムスタンプを表現します。「そのシートのパス＋タイムスタンプ」をコンポーネントのタイムスタンプとするのです。

ライブラリパーツセクション

このライブラリパーツセクションは、<libparts>タグで囲まれたものとなり、このセクションは回路図ライブラリの情報を定義するものとなります。このセクションは、次のものを含みます：

- <fp>で定義されるフットプリント名（名前にはワイルドカードが利用されます）
- <fields>で定義されるフィールド
- <pins>で定義されるピン情報

```
  <libparts>
    <libpart lib="device" part="CP">
      <description>Condensateur polarise</description>
      <footprints>
        <fp>CP*</fp>
        <fp>SM*</fp>
      </footprints>
      <fields>
        <field name="Reference">C</field>
        <field name="Valeur">CP</field>
      </fields>
      <pins>
        <pin num="1" name="1" type="passive"/>
        <pin num="2" name="2" type="passive"/>
      </pins>
    </libpart>
  </libparts>
```

`<pin num="1" type="passive"/>` のような行は、ピンの電気的な種類を定義するものです。有効なピンの種類は、次のものがあります。

Input	通常の入力
Output	通常の出力
Bidirectional	入力または出力
Tri-state	バスの入出力
Passive	通常の受動部品のピン
Unspecified	不明な種類
Power input	コンポーネントの電源入力
Power output	レギュレータ IC のような部品の電源出力
Open collector	アナログコンパレータでよくみられるオープンコレクタ
Open emitter	ロジック IC でみられるオープンエミッタ
Not connected	回路図上でオープンとすべきピン

ライブラリセクション

ライブラリセクションは<libraries>タグで囲まれたものとなります。このセクションはプロジェクトから利用されているライブラリ情報を含みます。

```
<libraries>
  <library logical="device">
    <uri>F:\kicad\share\library\device.lib</uri>
  </library>
  <library logical="conn">
    <uri>F:\kicad\share\library\conn.lib</uri>
  </library>
</libraries>
```

ネットセクション

ネットセクションは<nets>タグで囲まれたものとなります。このセクションは、回路図上の接続情報を定義するものです。

```
<nets>
  <net code="1" name="GND">
    <node ref="U1" pin="7"/>
    <node ref="C1" pin="2"/>
    <node ref="U2" pin="7"/>
    <node ref="P1" pin="4"/>
  </net>
  <net code="2" name="VCC">
    <node ref="R1" pin="1"/>
    <node ref="U1" pin="14"/>
    <node ref="U2" pin="4"/>
    <node ref="U2" pin="1"/>
    <node ref="U2" pin="14"/>
    <node ref="P1" pin="1"/>
  </net>
</nets>
```

このセクションでは、回路図上の全てのネットを羅列します。
ネット情報の例を次に示します。

```
    <net code="1" name="GND">
      <node ref="U1" pin="7"/>
      <node ref="C1" pin="2"/>
      <node ref="U2" pin="7"/>
      <node ref="P1" pin="4"/>
    </net>
```

net code	ネットの内部的な識別番号
name	ネット名
node	ネットに接続されるピン

14.6 - xsltproc に関する追加情報

次のページを参照してください： http://xmlsoft.org/XSLT/xsltproc.html

はじめに

xsltproc は XSLT スタイルシートを XML 文書に適用するためのコマンドラインツールです。これは GNOME プロジェクトの一環として開発され、GNOME デスクトップ環境無しでも利用することが可能です。
xsltproc はスタイルシート名と適用するファイル名をオプションとし、コマンドラインより起動されます。標準入力を利用する場合、ファイル名には - 記号を利用します。
If スタイルシートが XML 文書内に指示されている場合、コマンドラインでスタイルシート名を指示する必要はありません。xsltproc は自動的にスタイル シートを検出し利用します。標準では、出力が標準出力となっています。ファイルとして結果を出力したい場合には、 -o オプションを利用します。

コマンドライン

xsltproc [[-V] | [-v] | [-o *file*] | [--timing] | [--repeat] | [--debug] | [--novalid] | [--noout] | [--maxdepth *val*] | [--html] | [--param *name value*] | [--stringparam *name value*] | [--nonet] | [--path *paths*] | [--load-trace] | [--catalogs] | [--xinclude] | [--profile] | [--dumpextensions] | [--nowrite] | [--nomkdir] | [--writesubtree] | [--nodtdattr]] [*stylesheet*] [*file1*] [*file2*] [....]

コマンドラインオプション

-V 又は --version

利用している libxml と libxslt のバージョン情報を表示します。

-v 又は --verbose

xsltproc がスタイルシートとドキュメントを処理する各段階でメッセージを出力します。

-o 又は --output file

<ファイル名>で指定されたファイルへ結果を出力します。「チャンク」などとして知られているように、複数出力したい場合は -o ディレクトリ名/ として指定したディレクトリへファイルを出力させます。この場合、ディレクトリは予め作成しておく必要があります。

--timing

スタイルシートの構文解析、ドキュメントの構文解析、スタイルシートの適用、結果の保存に掛かった時間を表示します。ミリ秒の単位で表示されます。

--repeat

タイミングテストの為に、変換を 20 回繰り返し実行します。

--debug

デバッグの為に、変換されたドキュメントの XML ツリーを出力します。

--novalid

ドキュメントの DTD の読み込みをスキップします。

--noout

結果を出力しません。

--maxdepth <値>

libxslt の無限ループを防ぐため、テンプレートの最大スタック深さを調整します。デフォルトは 500 です。

--html

HTML ファイルを入力ファイルとします。

--param <パラメータ名> <値>

スタイルシート中の、パラメータで指定された<パラメータ名>　および<値>の処理を行いません。パラメータ名と値のペアは、最大 32 個まで指定することができます。値をノードの識別ではなく、文字列として処理したい場合は、--stringparam オプションを利用してください。

--stringparam <パラメータ名> <値>

<パラメータ名>と<値>で指定された値について、ノードの識別ではなく文字列として扱うようにします。

(注：これら文字列は utf-8 エンコードされている必要があります。)

--nonet

DTD のエンティティやドキュメントをインターネットから取得しません。

--path <パス>

DTD やエンティティ、ドキュメントの読み込みに、<パス>で(半角スペースやカンマで区切られた)指定されたファイルのリストを使用します。

--load-trace

処理中に読み込まれた全てのドキュメントを、標準エラー出力へ出力します。

--catalogs

SGML_CATALOG_FILES 内で指定された SGML カタログを外部エンティティの解決に利用します。標準では、xsltproc は XML_CATALOG_FILES で指定された場所を探します。
XML_CATALOG_FILES が定義されていない場合、/etc/xml /catalog を利用します。

--xinclude

Xinclude の仕様に基づき、入力ドキュメントの処理を行います。Xinclude の詳細は、次を参照してください： http://www.w3.org/TR/xinclude/

--profile 又は --norman

スタイルシートのそれぞれのパーツの処理時に、プロファイル情報の詳細を出力します。これはスタイルシートのパフォーマンスを最適化するために利用できます。

--dumpextensions

登録済みの拡張子のリストを標準出力へ出力します。

--nowrite

ファイルやリソースへの書き込みを行いません。

--nomkdir

ディレクトリを作成しません。

--writesubtree <パス>

<パス>で指定されたパス内のファイルのみ書込します。

--nodtdattr

ドキュメント内 DTD の標準アトリビュートを適用しません。

Xsltproc の戻り値

xsltproc はスクリプトからの呼び出し時などに利用しやすいよう、戻り値でステータスを返します。

0: 通常

1: 引数なし

2: パラメータが多すぎる

3: 不明なオプション

4: スタイルシートの構文解析に失敗(parse error)

5: スタイルシート内にエラー

6: ドキュメントのひとつにエラー

7: 未サポートの xsl : 出力メソッド

8: 文字列パラメータがクオートとダブルクオーテーションの両方を含んでいる

9: 内部処理エラー

10: 中断シグナル(CTRL+C など)により処理を終了

11: 出力ファイルに書き込めない

xsltproc に関する追加情報

libxml web ページ: http://www.xmlsoft.org/

W3C XSLT ページ: http://www.w3.org/TR/xslt