

第 2 章 プロジェクト構造

2.1 プロジェクト構造と特長

Lattice Diamond のプロジェクト構造として、以下のような特長があります。

1. 単一ライセンスで複数のデザイン・プロジェクトを同時実行することができます
 - ・ 同一プロジェクトを対象にした複数実行は除きます
 - ・ Synplify-Pro も同様に GUI を複数立ち上げて実行することができます
 - ・ Diamond 3.12 のみに付属する ModelSim Lattice Edition は、異なるシミュレーション・プロジェクトに対して GUI を複数立ち上げることはできますが、シミュレーションの複数実行はできません (Diamond 3.11 までに付属する Active-HDL Lattice Edition も同様です)
2. デザイン・プロジェクトは大きくは ”ストラテジー” および ”インプリメンテーション”、という二つの構成要素で管理されます。本章で詳細に記述します
3. 本ユーザガイドのカバーしていない項目やより詳細については以下をご参照ください
 - ・ スタートページ・ビューからリンクされているドキュメント類
 - ・ オンラインヘルプ (メニュー [Help] → [Lattice Diamond Help])
 - ・ ラティスセミコンダクターのウェブサイト (<https://www.latticesemi.com>)
 - ・ 特定のトピックスについて知りたい場合、GUI 上でマウスポインターを当該アイテムに移動した状態でファンクションキー『F1』を押します。当該項目についてのオンラインヘルプ・ページを直接表示することができます (”コンテキスト・センシティブ・ヘルプ”)

Lattice Diamond では、”プロジェクト” ごとにデザインを構成しますが、主な管理機構は ”ストラテジー (Strategy)” と ”インプリメンテーション (Implementation)” という考え方です。プロジェクトには必ず一つの ”LPF 制約ファイル” が存在し、ユーザが用意する RTL ソースファイルがプロジェクトの入力になります (図 2-1)。

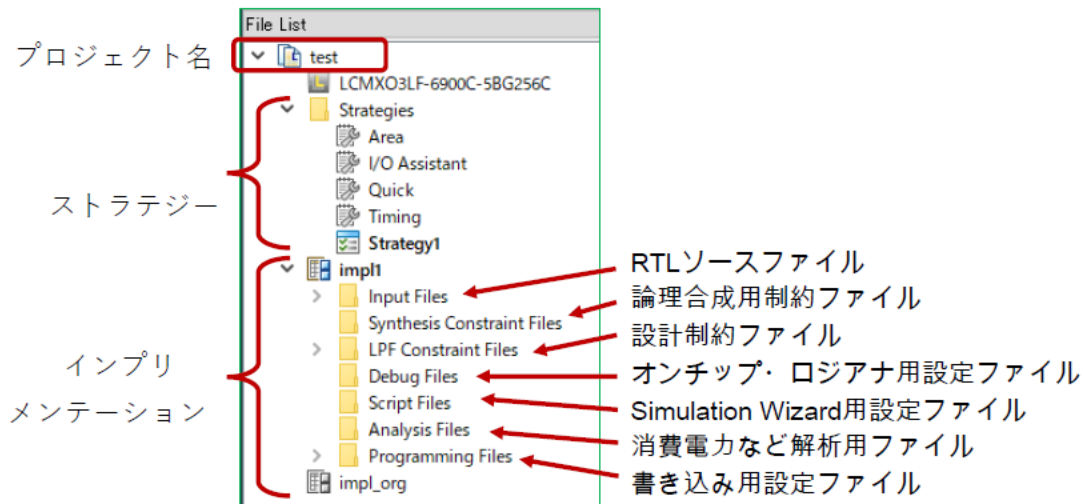
インプリメンテーションはデザインの管理単位であり、デザイン実装に関わる全てのファイルを含んだものです。論理合成や配置配線といった処理プロセスはインプリメンテーション単位で行われます。RTL ソースファイルは [Input Files] セクション、LPF 設計制約ファイルは [LPF Constraint Files] セクション、これ以外にオンチップ・ロジアナ Reveal 用は [Debug Files] セクション、書き込み用ファイルは [Programming Files] セクションに、などそれぞれ表示されます。インプリメンテーションは複数作成することができます。

ストラテジーは各プロセスの制約設定 (Constraints、Preferences) やオプション設定を一元管理する機構です。定義済みストラテジーは五種類あり、プロジェクトとして管理され、またインプリメンテーションごとに使用するストラテジーを 1 つ指定します。複数のインプリメンテーションで同じストラテジーを共用することも、インプリメンテーションごとに異なるストラテジーを割り当てることも可能です (図 2-2)。ストラテジー・オプション個々の詳細についてはプロセス各章の記述をご参照ください。

デザインエントリーは RTL ソースファイル (VHDL/Verilog HDL 記述) や回路図、EDIF のいずれかですが、インプリメンテーションごとに登録 (インポート) します。複数のインプリメンテーションで同じソースファイルを参照して共用することが可能です (図 2-2)。モジュール生成 (第 4 章参照) した場合は、RTL ソースファイルの代わりに、自動生成される定義ファイル (.ipx または .sbx) をインポートします。

註：本 Lattice Diamond 日本語マニュアルは、日本語による理解のため一助として提供しています。作成にあたっては各トピックについて可能な限り正確を期しておりますが、必ずしも網羅的あるいは最新でない可能性や、オリジナル英語版オンラインヘルプや各種ドキュメントと不一致がある可能性があります。疑義が生じた場合は技術サポート担当者にお問い合わせ頂くか、または最新の英語オリジナル・ソースを参照するようにお願い致します。

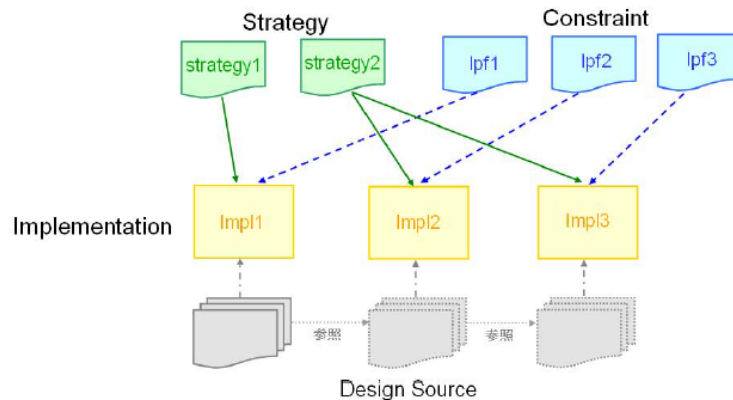
図 2-1. プロジェクト構造



各プロセスの処理で、デザインソース・ファイルと LPF 設計制約ファイル以外は必須ではありません。

なお、本ユーザガイドでは回路図入力や EDIF 入力時の記述を省略していますので、オンラインヘルプをご参照ください。

図 2-2. 複数インプリメンテーション時の各ソース・ファイルの共有例



プロジェクトを構成するファイルリスト・ビュー (File List) 内の各ファイルについて、ボールド字体表示の意味することには二通りあります。[Input Files] セクションのソースファイルについては、トップモジュールとして認識されたファイルがボールド表示になります。ボールド表示が二つ以上あったり、或いは一つもない場合は、その状態を解消する必要があります。

インプリメンテーション下のそれ以外の各セクションと、[Strategies] セクションについては”アクティブ”なファイルがボールド字体表示になります。それらは複数のファイルが登録されることが可能ですが、各プロセスに適用されるアクティブな設定ファイルはそれぞれ一つのみです。この”アクティブ”については第 2.3.3 項、第 2.4.4 項、第 2.5.3 項をそれぞれご参照ください。

2.2 プロジェクト管理

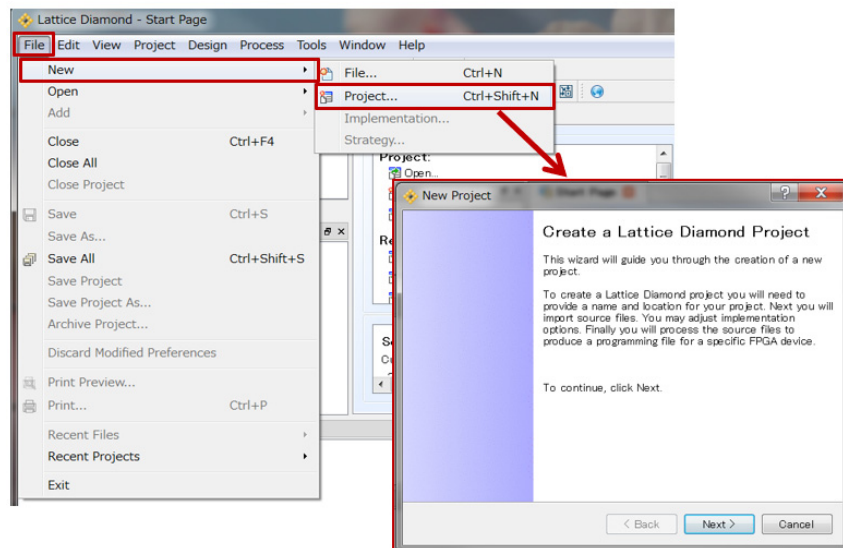
2.2.1 新規プロジェクトの作成

新規プロジェクトの作成は、メニューバーから [File] → [New] → [Project] の順に選択するか、スタートページ・ビュー左上部の [Project] セクションで ”New...” を選択します。これで ”New Project” ウィンドウが表示

されます（図 2-3）ので、その指示に従ってデバイス等を選択すればプロジェクトが生成されます。

プロジェクト生成に伴って”プロジェクト・フォルダー”は自動作成されませんので、フィッティング用の作業フォルダーを事前に作成しておく事を推奨します。

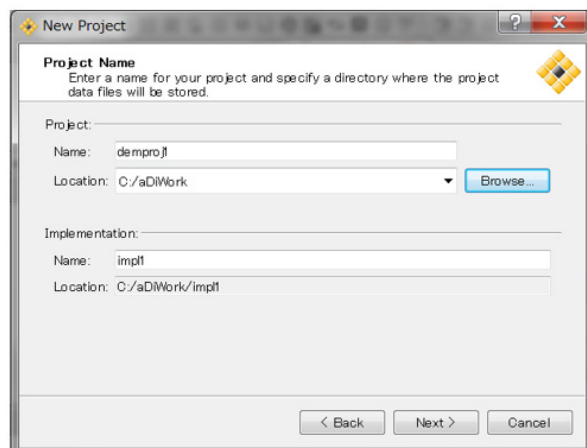
図 2-3. 新規プロジェクトの作成開始



”New Project” ウィンドウの始めは、プロジェクトの生成に必要なフォルダーやデバイスを指定する旨のメッセージが表示されています。そのまま『Next>』ボタンをクリックし、次に表示されるウィンドウでプロジェクト名とプロジェクトのフォルダーパスを設定します（図 2-4）。

Project セクションの「Name」セルにプロジェクト名を入力します。プロジェクト名として使用できる文字は、アルファベット、数字と”_”（アンダースコア）です（先頭の 1 文字目はアルファベットのみ）。使用できない文字はキーを打っても入力されません。

図 2-4. 新規プロジェクトの作成～プロジェクト名とフォルダーパスの設定

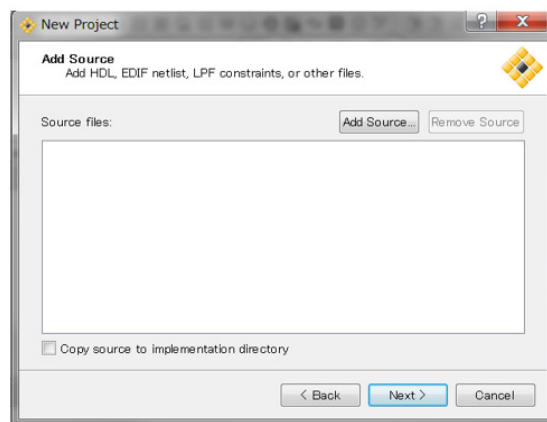


「Location:」セルにはプロジェクトで使用するフォルダーを指定します。直接パスを入力するか、『Browse...』ボタンをクリックして立ち上がるウィンドウ上で適切なフォルダーを選択します。フォルダーパスを直接入力した場合で、指定したフォルダーが存在しないと自動的に新しいフォルダーが生成されます。

Implementation セクションの「Name:」セルにはインプリメンテーション名を入力します。デフォルト名の”impl1”が自動的に入力されていますが、変更できます。「Name:」を変更すると「Location:」セルのパス表示

も自動的に更新されます。設定が完了後、『Next >』ボタンをクリックすると、次はソースをインポートするウィンドウ [Add Source] が表示されます (図 2-5)。

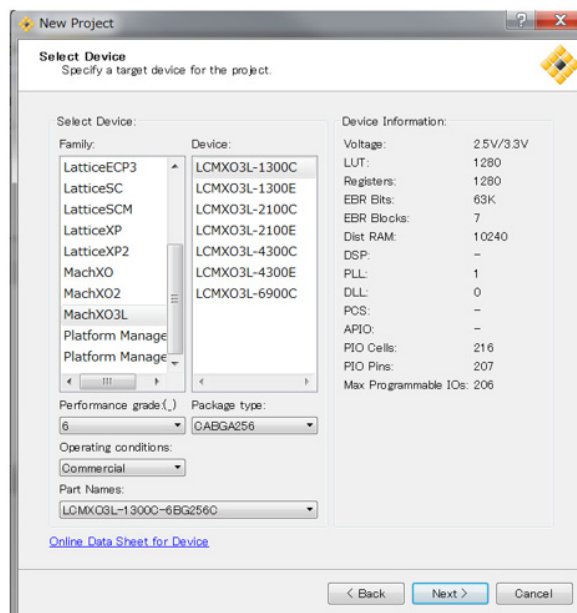
図 2-5. プロジェクトの新規作成～ソースファイルのインポート



『Add Source...』ボタンをクリックすると起動するファイル・ブラウザで、必要なファイル（既存のHDLソースや制約ファイル）を選択します。この際、ウィンドウ左下の「Copy source to...」チェックボックスにチェックが入っていると、選択したファイルがプロジェクトのフォルダーへコピーされ、それがインポートされます。チェックが入っていない場合は選択したファイルがインポートされます。

各ソースファイルはプロジェクト作成後でもインポートできますので、このステップで全てをインポートする必要はありません。完了後『Next >』ボタンをクリックすると、次はターゲット・デバイスの選択ウィンドウが表示されます (図 2-6)。

図 2-6. プロジェクトの新規作成～デバイス選択



このウィンドウで、使用するデバイスやスピードグレード、パッケージを選択します。ライセンス・タイプによって選択できるファミリーに違いがあります。有償版 (Subscription) ライセンスでは全てのファミリーが選択できますが、無償版 (Free) ライセンスでは、SERDES/PCS を内蔵しているデバイスは表示されないため選択できません。デバイスやパッケージはプロジェクト作成後でも変更できますので、確定していない条件については暫定的に適当なものを選択しておき、確定後に変更するようにします。

次に『Next >』ボタンをクリックすると、論理合成ツールの選択ウィンドウが開きます（図 2-7、一部）。デフォルトの論理合成ツールは LSE（Lattice Synthesis Engine）ですが、Synplify Pro を選択する事も可能です。また、プロジェクト生成後でも、変更することが可能です。

図 2-7. 論理合成ツールの選択

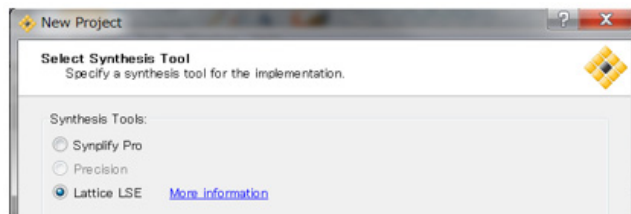
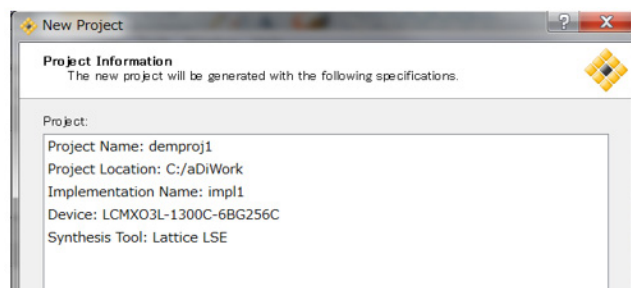


図 2-8. 新規プロジェクトの作成～設定確認ウィンドウ例

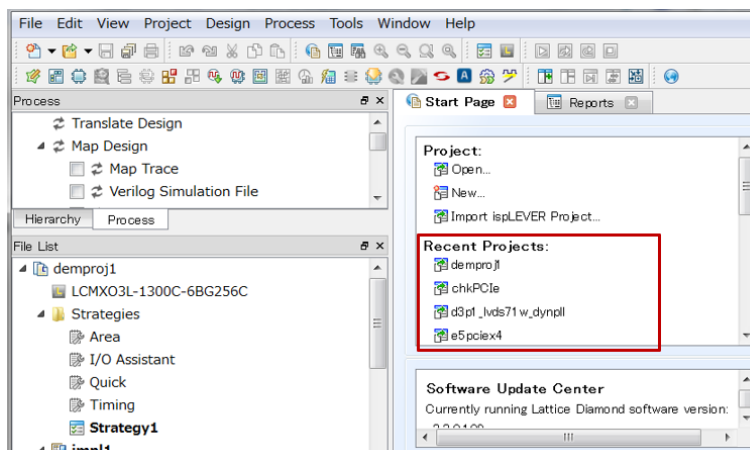


次に『Next >』ボタンをクリックすると、これまでの設定の確認ウィンドウが開きます（図 2-8、一部）。『Finish』ボタン（図 2-8 では未表示のウィンドウ下部）をクリックすれば、設定は完了です。設定内容を変更する場合は、『< Back』ボタンをクリックして適切なウィンドウまで戻り、設定変更を行ってください。作成されたプロジェクト情報（インプリメンテーションやインポートしたファイル等）は、プロジェクト・フォルダーに“<プロジェクト名>.ldf”というファイル名で保存されます。

2.2.2 既存プロジェクトのオープン

既存の Lattice Diamond プロジェクトを開くには、メニューバーから [File] → [open] → [Project...] の順に選択するか、スタートページ・ビュー左上の [Project] セクションで “Open...” をクリックします。また [Recent Projects:] セクションには直近に作業したプロジェクト名が所定数分表示されています（図 2-9 の赤枠）。開きたいプロジェクトがこの中にあれば、そのプロジェクト名をクリックしてもオープンできます。[Recent Projects:] に表示するプロジェクト数は、オプション設定で変更することができます（第 23.3 節参照）。

図 2-9. 直近プロジェクトのオープン



2.2.3 プロジェクトのクローズ

プロジェクトをクローズする場合は、メニューバーから [File] → [Close Project] の順に選択します。他のプロジェクトをオープンするか、Lattice Diamond をシャットダウンしても自動的にクローズされます。メニューの [File] → [Close] や [File] → [Close All] ではプロジェクトはクローズされません。

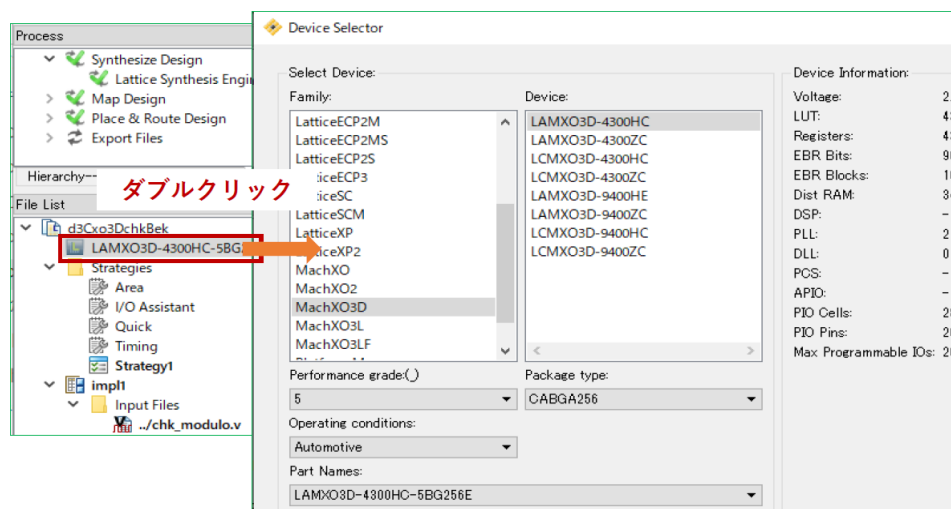
2.2.4 プロジェクト生成後の属性などの変更


新規プロジェクト作成後や既存プロジェクトの再利用などで、プロジェクト設定を変更する必要があることはごく一般的に起きます。

2.2.4.1 ターゲット・デバイスの変更

ターゲットとするデバイスや、デバイスの論理規模（バルク）、或いはパッケージやスピードグレードなど、このような変更の必要がある場合の操作方法是三通りあります。

図 2-10. ターゲットデバイスの変更方法 3




第一の方法は Diamond GUI 上部のアイコンバーから、 をクリックすることで "Device Selector" ウィンドウを表示させます。これはプロジェクト作成時のデバイス選択（図 2-6）と同じ画面です。第二の方法は Diamond のメニューから [Project] → [Device...] と選択することで同じウィンドウを表示させます。第三の方法はファイルリスト・ビュー内でプロジェクト名の下に表示されているデバイス名をダブルクリックすることです（図 2-10）。

ターゲット・デバイスはプロジェクトで一つのみ指定可能です。インプリメンテーション毎に異なる設定はできません。"Device Selector" ウィンドウでの操作はプロジェクト作成時と同様です。変更すると、それまでに処理が完了しているプロセスでも全て論理合成からの再実行になります。

2.2.4.2 論理合成ツールの変更

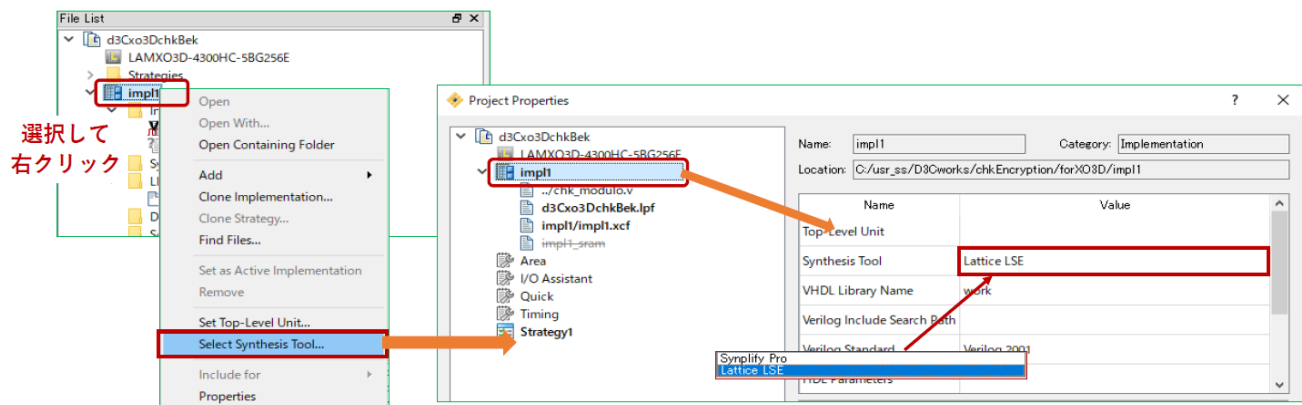
プロジェクト生成時に選択した論理合成ツールを変更する必要がある場合の操作方法も三通りあります。

第一の方法は Diamond GUI 上部のアイコンバーから、 をクリックして "Project Properties" ウィンドウを表示させます。第二の方法は Diamond のメニューから [Project] → [Property Pages] と選択することで同じウィンドウを表示させます。どちらもあらかじめ変更対象のインプリメンテーションを "アクティブ" にしておきます（第 2.3.3 項）。その後表示されるウィンドウの右枠で「Synthesis Tool」のセルをクリックすると現れるプルダウンから選択します。

第三の方法はファイルリスト・ビュー内で "アクティブ" な当該インプリメンテーションを選択した後に右クリックすると表示されるメニューから [Select Synthesis Tool...] を選択します（図 2-11 左）。"Project

Properties” ウィンドウが表示されますので（図 2-11 右）、あとは同様に操作します。

図 2-11. 論理合成ツールの変更操作

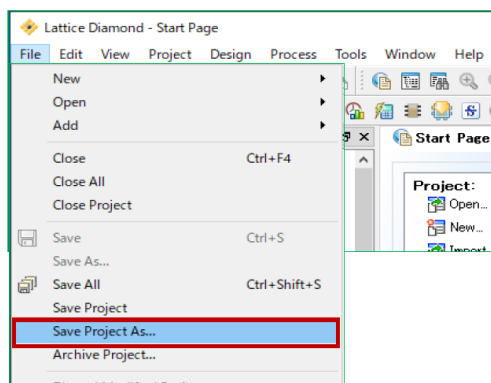


論理合成ツールはインプリメンテーション毎に異なる指定ができます。いずれかのインプリメンテーションで変更しても、他のインプリメンテーションには作用しません。また、変更すると論理合成プロセスからの再実行になります。

2.2.5 プロジェクトの複製

プロジェクトを複製するためには、メニューから [File] → [Save Project As...] を選択して表示されるウィンドウで、保存フォルダーと別名のプロジェクト名称を指定します（図 2-12）。

図 2-12. プロジェクトの複製



2.2.6 アーカイブ・プロジェクトの作成

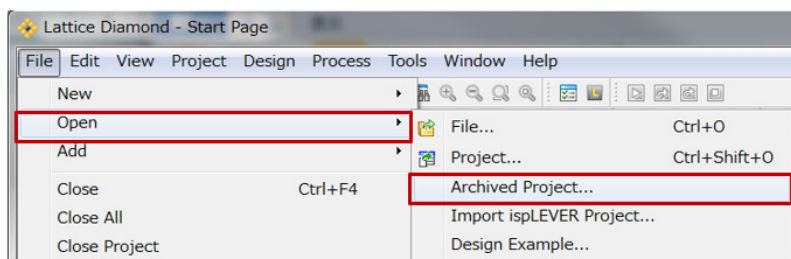
プロジェクトを保存する場合や何らかの都合でフォルダーを移動させる場合等に、プロジェクトのアーカイブを作成することができます。メニューバーから [File] → [Archive Project] を選択すると、「Archive Project」ウィンドウが開きます。このウィンドウでアーカイブ・ファイル（.zip）の保存先と名称を指定します。アーカイブ・ファイルであることが容易に判別できるような名称、例えば「xxx_archv.zip」のようにすることを推奨します。

Diamond の環境設定オプションの一つである「Archive all files under the Project directory when archiving project」がデフォルトでイネーブルされています（第 23.3.1 節参照）。この状態、または「Archive Project」ウィンドウ左下にある「Archive all files under the Project directory.」にチェックが入れると、プロジェクトとの関連に関わらずプロジェクト・フォルダ下にあるフォルダとファイル全てが含まれるアーカイブが作成されます。チェックが入っていない場合で、かつ環境設定オプションがデフォルトからディセーブルに変更さ

れた場合は、プロジェクトの復元に必要な関連フォルダとファイルのみの、サイズの小さいアーカイブ・ファイルが作成されます。FAE のテクニカルサポートにプロジェクトを送付して技術支援を受ける場合などに、後者のアーカイブ機能を用いると、小さいファイルサイズでやりとりが容易になります。

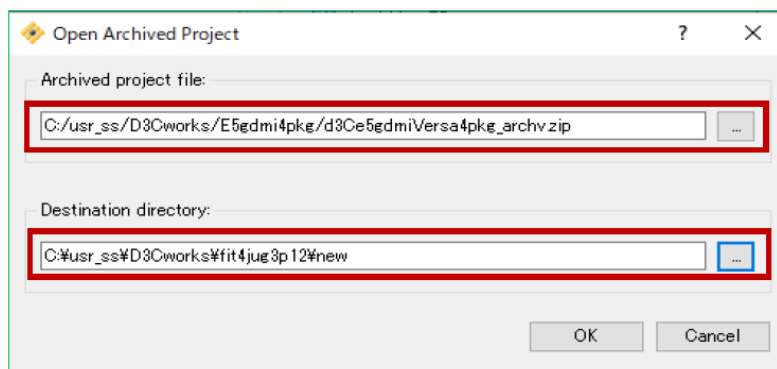
なお、予期しないトラブルを防ぐため、アーカイブ後の .zip ファイルは Windows 対応の解凍ツールを用いて unzip するのではなく、メニューから [File] → [Open] → [Archived Project...] を選択して（図 2-13）復元するようにします。

図 2-13. アーカイブ・プロジェクトの復元操作 1



立ち上がるウィンドウ（図 2-14）で、アーカイブ・ファイルと復元先のフォルダーを「Destination directory」セルにブラウザ後指定して『OK』をクリックします。

図 2-14. アーカイブ・プロジェクトの復元操作 2



指定した復元先のファイル / フォルダ構造は、アーカイブ元のそれとは異なり、ツール所定の様式で展開されます。

なお、当該デザインにモジュール生成ツールで作成した “ROM モジュール” が含まれている場合は注意が必要です。ROM 生成には必ずメモリ初期化用のテキスト・ファイル “xxx.mem” が必要です（第 4.2.3 節参照）。生成された ROM モジュールを再生成する状況がなければ mem ファイルは不要ですが、プロジェクト・フォルダー下に無い場合は、いずれにしろアーカイブ zip には含まれないことに、ご注意ください。

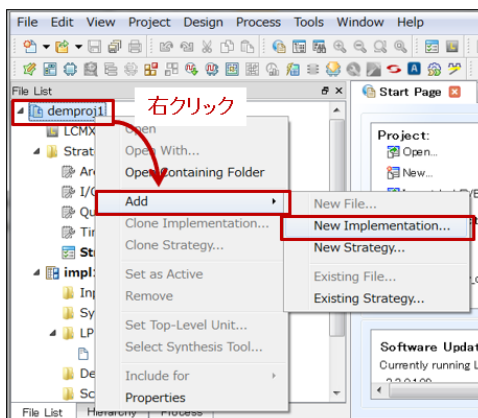
2.3 インプリメンテーションの管理

”インプリメンテーション”とは、前述のとおりデザインの管理単位であり、デザイン実装に関わる全てのファイルを含んだものです。Diamond では、単一プロジェクト内に複数のインプリメンテーションを定義することができます。それぞれのインプリメンテーションで異なるソースファイルや制約ファイル、あるいはオプションを用いて論理合成や配置配線を行うことができます。Diamond では単一プロジェクト内で最適な実装の推敲を行うことが容易です。

2.3.1 新規インプリメンテーションの追加

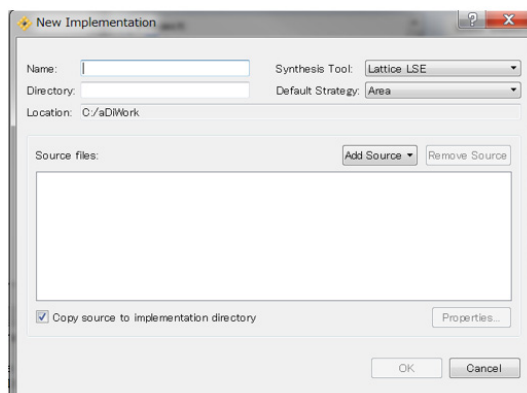
インプリメンテーションは新規プロジェクトを作成した際に必ず 1 つ作成されます（ユーザが命名）。

図 2-15. インプリメンテーションの追加



追加の新規インプリメンテーションが必要な場合、メニューバーから [File] → [New] → [Implementation...] の順に選択するか、ファイルリスト・ビューの一番上に表示されているプロジェクト名を右クリックし [Add] → [New Implementation...] の順に選択します (図 2-15)。これで "New Implementation" ウィンドウが表示されます (図 2-16)。

図 2-16. New Implementation ウィンドウ



「Name:」セルは追加するインプリメンテーションの名称、「Directory:」セルはインプリメンテーションで使用するファイルを格納するフォルダー名を入力します。名称を先に入力するとフォルダー名も自動的に同じものが入力されますが、変更することもできます。

「Default Strategy:」セルはインプリメンテーションに適用するストラテジーを選択します。右側の▼アイコンをクリックすると、プロジェクトに登録されているストラテジーの一覧が表示されますので、適切なものを選択します。インプリメンテーション作成後に変更することも可能です。

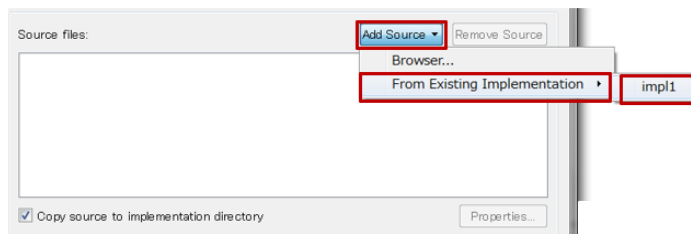
インプリメンテーション作成時にソースファイルのインポートを行うことができます。『Add Source』ボタンをクリックすると、プルダウンメニューで [Browser...] と [From Existing Implementation] が表示されます。ブラウザを使用して既存のファイルを選択する場合は前者を選択します。既存のインプリメンテーションにインポートされているソースを選択する場合は、後者を選択しますが、すると既存のインプリメンテーション名が全て候補としてリストされますので、その中から 1 つを選択します (図 2-17。この図では一つのみ)。

この際、ウィンドウ左下の「Copy source to implementation directory」にチェックが入っていると、選択したソースは新規に作成したインプリメンテーションのフォルダーにコピーされ、それがインポートされます。チェックが入っていないと選択したオリジナルのソースファイルがインポートされます。

コピーしたものをインポートする場合、オリジナルのソースファイルは変更されませんので、他のインプリメンテーションのソースには影響しません。逆にオリジナルのソースファイルをインポートする場合、複

数のインプリメンテーションが同一ソースを参照していれば、更新された際に全てが影響されます。

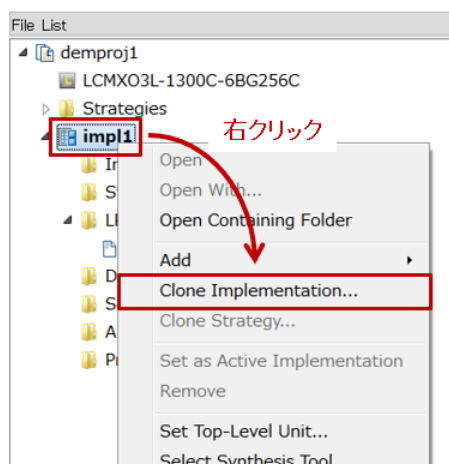
図 2-17. 既存インプリメンテーションからのソースのインポート例



必要なソースを選択後、『OK』ボタンをクリックして設定は完了です。”インプリメンテーション・フォルダー”が自動作成され、各種ファイル / サブフォルダーがその下に置かれます。

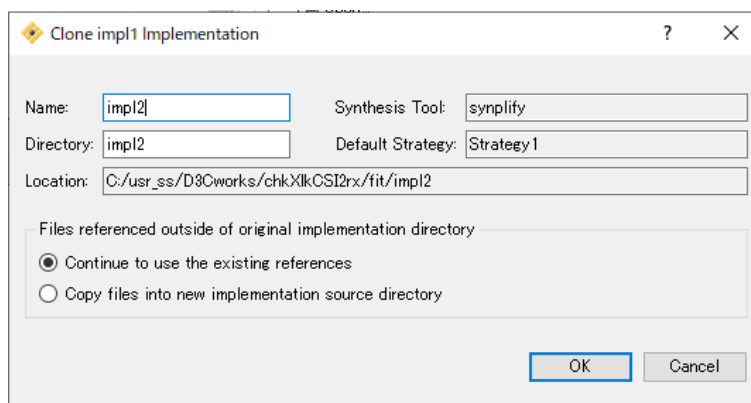
2.3.2 インプリメンテーションの複製（クローン）

図 2-18. インプリメンテーションの複製



Lattice Diamond にはインプリメンテーションの複製（”クローン・インプリメンテーション”）を容易に生成する機能があります。

図 2-19. クローン・インプリメンテーションの設定



複製の元になるインプリメンテーションを選択後、右クリックで[Clone Implementation...]を選びます(図 2-18)。”アクティブ”（次項で記述）でないインプリメンテーションでも選択は有効です。その後表示されるウィンドウ（図 2-19）で複製後の名称を「Name:」セルに入力します。下部でソースファイルを参照するか、

コピーするかの指定も行います。

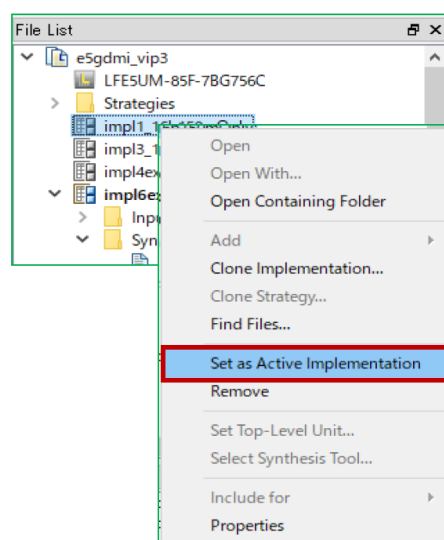
インプリメンテーションがフォルダー一括で複製され、付与した名称にされます。

2.3.3 インプリメンテーションのアクティブ化

複数のインプリメンテーションを持つプロジェクトでは、”アクティブ”なインプリメンテーションは一つだけです。アクティブとは、論理合成以下各処理プロセスの対象ということです。従って意図する処理対象のインプリメンテーションは、初めにアクティブ化してから、またはアクティブであることを確認してからすべての操作を行う必要があります。

ファイルリスト・ビューで、アクティブ化したいインプリメンテーション名を選択後に右クリックし、メニューから [Set as Active Implementation] を選択します。インプリメンテーションがアクティブになったことを示すボールド字体になります。

図 2-20. インプリメンテーションのアクティブ化



2.3.4 インプリメンテーションの削除

不要なインプリメンテーションをプロジェクトから削除するには、まずそのインプリメンテーションを非アクティブ化（他のインプリメンテーションをアクティブ化）します。次にそのインプリメンテーション名を右クリックし、[Remove] を選択するか（図 2-20 で赤枠の下）、またはインプリメンテーションを選択した状態でキーボードの [Delete] キーを押します。

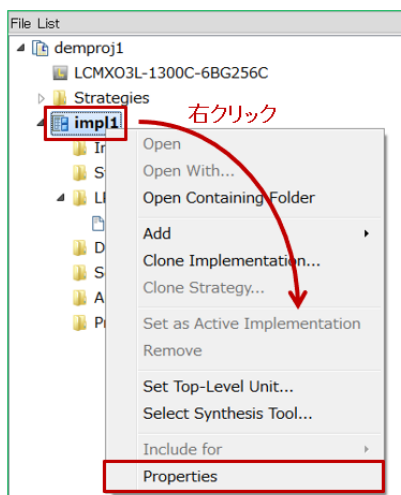
アクティブなインプリメンテーションは削除できません。なお、”削除”はインポート情報の削除であり、ファイルやフォルダー自体は削除されません。

2.3.5 インプリメンテーションのプロパティ設定

プロパティ設定はメニューバーから [Project] → [Property Pages] の順に選択するか、ファイルリスト・ビューでインプリメンテーション名を右クリックして [Properties] を選択します（図 2-21）。これにより”Project Properties”ウィンドウが立ち上がります（図 2-22）。

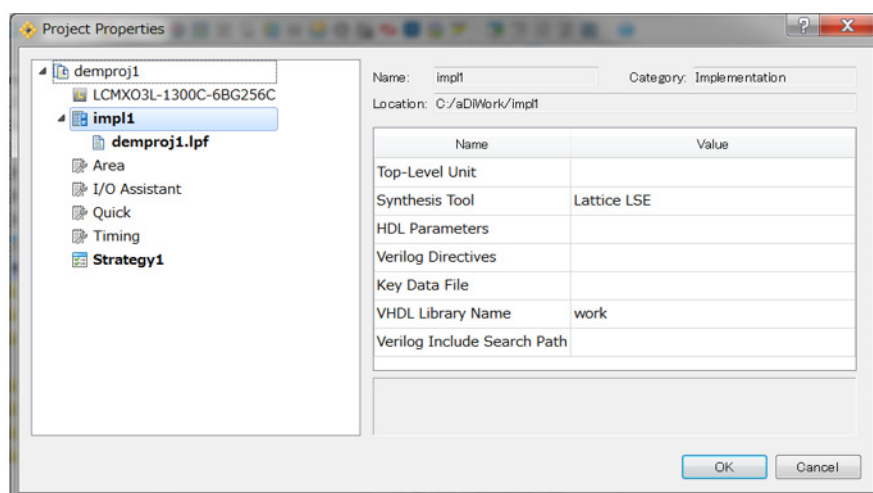
プロパティ設定には、最上位モジュール名、HDL パラメータ、VHDL ファイルのコンパイルライブラリー名、Verilog HDL ソース内でインクルードされているファイルがあるフォルダーのサーチ指定、などがあります。

図 2-21. インプリメンテーションのプロパティ設定



意図するインプリメンテーション名を左枠で選択し、右枠内でそれぞれ設定します。

図 2-22. Project Properties ウィンドウ～インプリメンテーション



非アクティブなインプリメンテーションを選択した場合は、何も表示されません。インプリメンテーションのプロパティとして設定できるのは以下の 4 つの項目です。

Top Level Unit

インポートしたデザインソースの最上位階層のエンティティ (VHDL) / モジュール (Verilog) 名を入力します。ほとんどの場合、最上位階層名はインポートしたソースから自動的に検出します。しかし VHDL/Verilog HDL 混在の場合など、最上位階層が認識されないケースがあります。そのような場合にここで指定します。

Synthesis Tool

選択されている論理合成ツールが表示されますが、変更することができます。

HDL Parameters

HDL ソースにグローバルで作用するパラメータ (generic/VHDL、parameter/Verilog) やコンパイラ指示子 (Verilog) を指定することができます。これ以外のパラメータは HDL ソース内の値が適用されます。設定は以下のように記述します。複数定義する場合はセミコロンで分離します。

記述ルール	パラメータ = 設定値
記述例	g_bus_width=16; comm_id=8;

VHDL Library Name

インプリメンテーションで、デフォルトで使用する VHDL ソースのコンパイル先ライブラリー名の設定です。デフォルトではデザイン全体が work ライブラリーとしてコンパイルされます。VHDL ソースファイル単位でコンパイル・ライブラリー名が異なる場合は、このウィンドウ左枠で当該 VHDL ソースファイルを選択し、右枠の "VHDL Library Name" セルに入力します。

Verilog Include Search Path

Verilog HDL ソース内で相対パス記述でインクルードしているファイルを検索するパスの設定です。インクルードされるファイルが、ソース内にフルパスでファイル名が記述されていたり、プロジェクト内の適切なフォルダーに保存されていたりする場合は、設定する必要はありません。

なお、既知の問題として、Lattice Diamond 3.12 では "Verilog Include Search Path" セルをクリックすると現れるべき『ブラウズボタン』が表示されません。手動で直接フォルダーパスを編集して入力するようにします（さもなくば Diamond 3.11 に戻って "Project Properties" ウィンドウを開き設定する必要があります）。

2.4 ストラテジーの管理

ストラテジーは前述のように各プロセスの制約設定（Constraints、Preferences）やオプション設定を一元管理する機構です。Diamond では、プロジェクト内に複数のストラテジーを持ち、それぞれのストラテジーでは異なるオプション設定を定義することができます。インプリメンテーションごとに "アクティブ" な(有効な) ストラテジーは一つだけ選択できます。

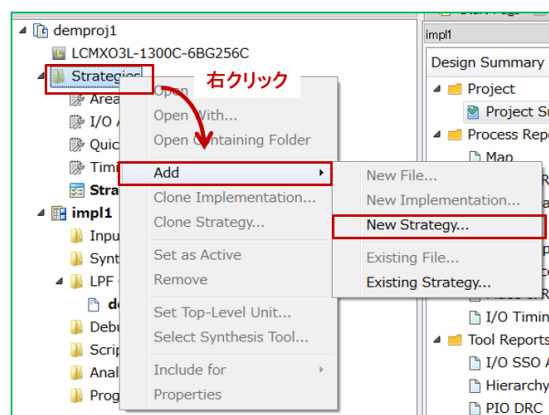
2.4.1 デフォルトのストラテジー

プロジェクトを作成した際に、以下の 5 つのストラテジーが作成されています。Strategy1 以外は編集することができません。

- Area** : 必要なリソース（スライス）数が最小になるように設定されたストラテジー
- I/O Assistant** : I/O Assistant 機能（IO の配置ルールチェック）を実行するためのストラテジー。IO のルールチェックだけの処理用であり、配線処理は行われない
- Quick** : プロセスの処理時間を優先するように設定されたストラテジー
- Timing** : タイミング要求を優先するように設定されたストラテジー（全てデフォルト）
- Strategy1** : Timing と同じ（ユーザーが編集可能）

ストラテジーを追加するには、新たに作成する方法と、既存のストラテジーを複製する方法があります。

図 2-23. ストラテジーの追加

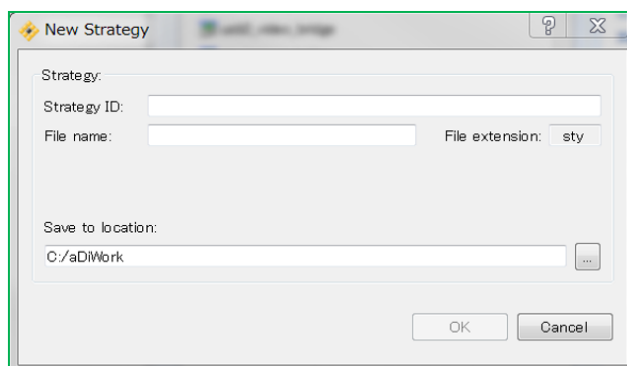


2.4.2 新規ストラテジーの作成

新規作成する場合は、メニューバーから [File] → [New] → [Strategy...] の順に選択するか、ファイルリスト・ビュー内の [Strategies] 表記を選択して右クリックし、[Add] → [New Strategy...] と選択します (図 2-23)。これで、"New Strategy" ウィンドウが表示されます (図 2-24)。

New Strategy ウィンドウの「Strategy ID:」セルにはストラテジー名を入力します。Lattice Diamond 上にはこの名前が表示されます。「File name:」セルにはデフォルトでストラテジー ID と同じ名前が設定されますが、変更することができます。「Save to Location:」セルはストラテジー設定ファイルを保存するフォルダーです。デフォルトではプロジェクトフォルダーが設定されていますが、変更することが可能です。

図 2-24. New Strategy ウィンドウ

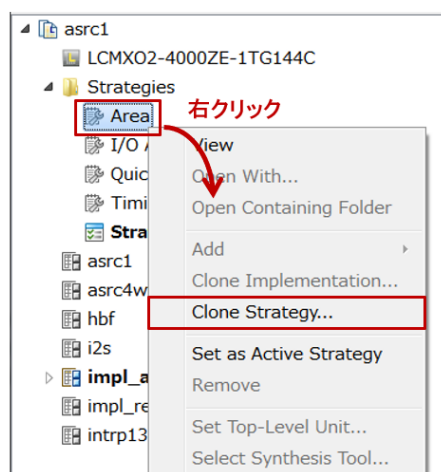


全ての設定完了後、OK ボタンをクリックすると、新規ストラテジー（設定は全てデフォルト）が作成され、「<ストラテジー名>.sty」というファイル名で指定フォルダーに保存されます。

2.4.3 ストラテジーの複製

Lattice Diamond には既存ストラテジーの複製 "クローン (Clone)" を作成する機能があります。複製したいストラテジーを選択して右クリックし、表示されるメニューから [Clone Strategy...] を選択します (図 2-25)。これで、図 2-24 と同様の "Clone *** ウィンドウ" ('***' は複製元のストラテジー名) が立ち上がりますので、新ストラテジー名を入力します。

図 2-25. ストラテジーの複製



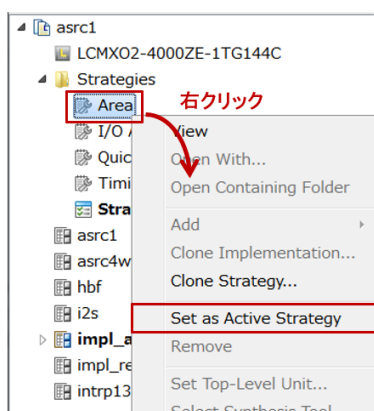
New Strategy ウィンドウの設定は、新規ストラテジーを追加する場合と同じですが、作成されるストラテジーの設定値が複製元と同じになっています。プロジェクト作成時にデフォルトで生成される設定変更できない "Strategy1" 以外のストラテジーでも、複製版は変更できます。

2.4.4 使用するストラテジーの選択（アクティブ化）

インプリメンテーションでは、“アクティブ化”されている1つのストラテジーのみが有効です。このため“ストラテジーのアクティブ化”がすなわち使用するストラテジーの選択になります。

ストラテジーをアクティブ化するには、ファイルリスト・ビューで当該ストラテジーを右クリックして選択後右クリックし、表示されるメニューから [Set as Active Strategy] を選択します（図 2-26）。アクティブ化されたストラテジーは太字で表示され、以後のプロセス処理に適用されます。

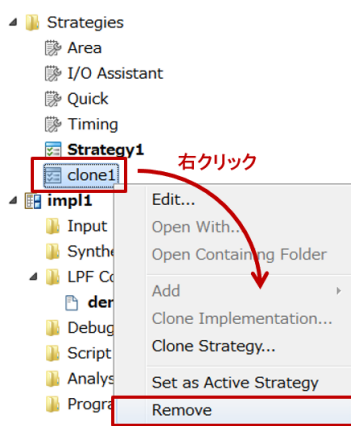
図 2-26. ストラテジーのアクティブ化



2.4.5 ストラテジーの削除

プロジェクトで不要になったストラテジーは削除できます。ファイルリスト・ビューで削除したいストラテジーを右クリックして選択後、表示メニューから [Remove] を選択するか（図 2-27）、キーボードの『Delete』キーを押すと、ストラテジーがプロジェクトから削除されます。”プロジェクトから削除”することはインポート情報の削除であり、ストラテジー・ファイルの削除ではありません。

図 2-27. ストラテジーの削除

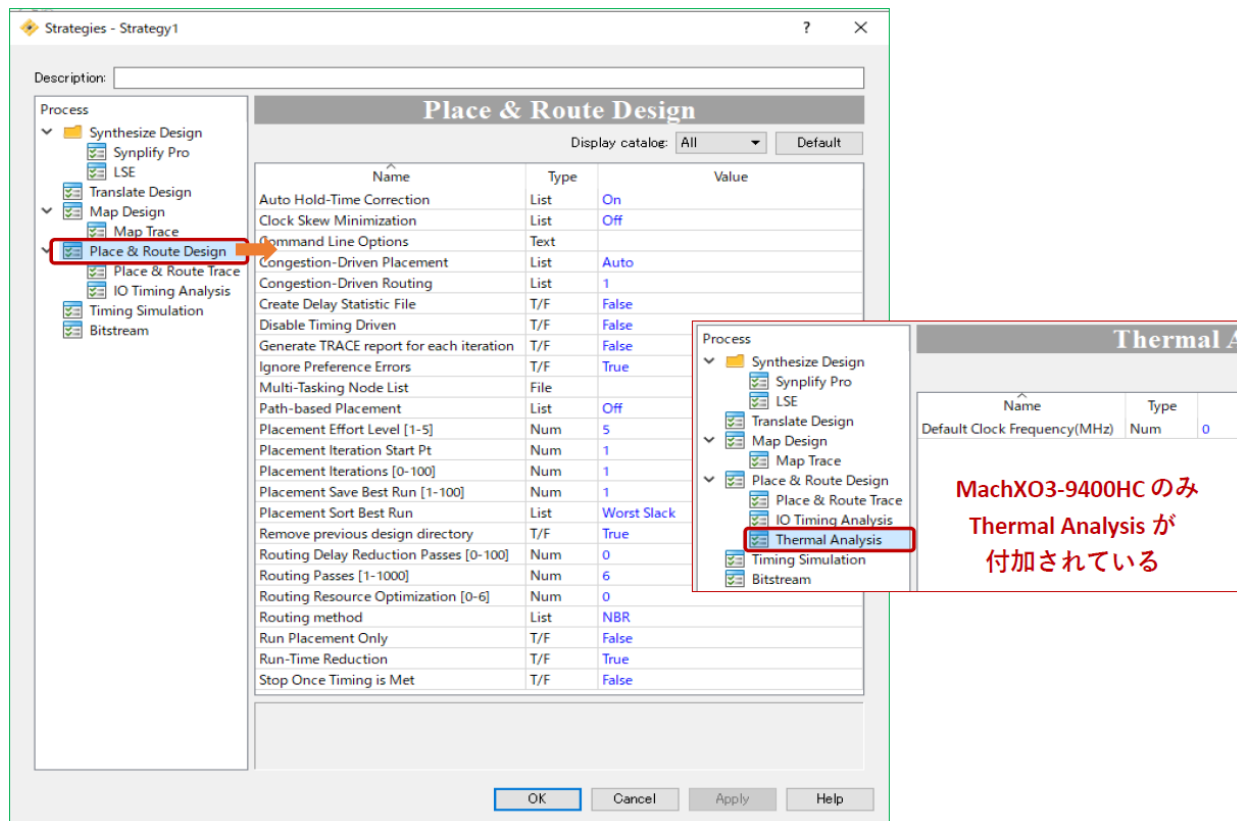


なお、アクティブなストラテジーは削除できません。また、プロジェクト作成時に自動生成されている五つは、“Strategy1”を含めて全て削除できません。

2.4.6 ストラテジー・オプションの編集

ストラテジー・オプション項目を変更・編集するためには、まずファイルリスト・ビューでストラテジー名をダブルクリックするか、または意図するストラテジーを右クリックして選択後、表示されるメニューから [Edit] を選択します。ストラテジーの設定ウィンドウが立ち上がります（図 2-28）。

図 2-28. ストラテジーの設定ウィンドウ (PAR)



このウィンドウで各プロセスの定義済みオプション設定を変更することができます。左枠で意図するプロセスかサブプロセスを選択します。オプション項目の詳細については、各プロセス章をご参照ください。

なお、MachXO3 ファミリーの 9400HC デバイスに限り、PAR サブプロセスとして”Thermal Analysis”が加えられています。消費電力解析が完了していることを担保して、先に進めるようになっています。

2.5 LPF 制約ファイルの管理

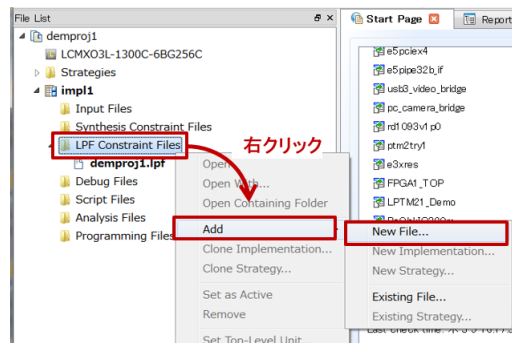
LPF (Logical Preference File) 制約ファイルは設計制約を記述するファイルで、拡張子は .lpf です。Diamond プロジェクトの各インプリメンテーションには 1 つ以上の LPF 制約ファイルが存在できますが、インプリメンテーションごとに”アクティブ”な LPF は一つです。インプリメンテーションのクローン機能を活用し、複数の制約ファイルで異なる制約設定を行い、これらを切り替えて実装した結果を吟味することで、最適な制約を容易に推敲することができます。

新規プロジェクト作成時に既存の LPF をインポートできますが、そうしない場合は自動的に生成されて組み込まれます。名前は”<プロジェクト名>.lpf”になります。デフォルトの内容は、どの実装にも適用される二つの BLOCK 制約が記述されているのみです。ユーザーはこの LPF ファイルを直接テキストエディターで編集しても良いですが、一般的には第 16 章に記述する”スプレッドシート・ビュー”で作成・更新します。

これ以外に新規制約ファイルを追加する場合は、メニューバーから [File] → [New] → [File...] の順に選択するか、ファイルリスト・ビューのインプリメンテーション・セクション下 [LPF Constraint Files] 行を選択後右クリックし、表示されるメニューから [Add] → [New File...] の順に選択します。これで、”New File”ウィンドウが起動します。

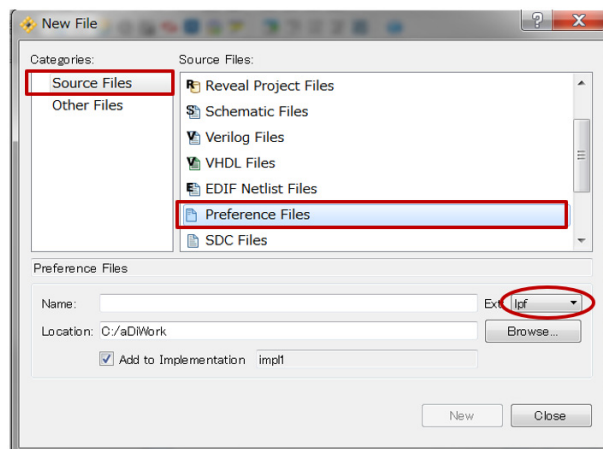
2.5.1 新規 LPF 制約ファイルの作成

図 2-29. 新規 LPF 制約ファイルの作成



Categories 枠で "Source Files"、Source Files 枠で "Preference Files" を選択し、「Name:」セルにファイル名を入力します。「Ext.」部（図 2-30 丸印）で拡張子を選択することもできますが、基本的には "lpf" を選択します。設定完了後に『New』ボタンをクリックすると、新しい LPF 制約ファイルが作成されテキストエディターが立ち上がります。編集後は自動的にインプリメンテーションに取りこまれます。

図 2-30. New File ウィンドウ (LPF 制約ファイルの作成)



新たに作成する LPF 制約ファイルの作業開始時は、完全なブランクです。プロジェクト作成時に自動で生成される LPF デフォルトに含まれる BLOCK 制約も記述されていません。

”LPF 制約ファイルの複製” 機能はありません。マニュアルでファイルや記述内容のコピーを行います。

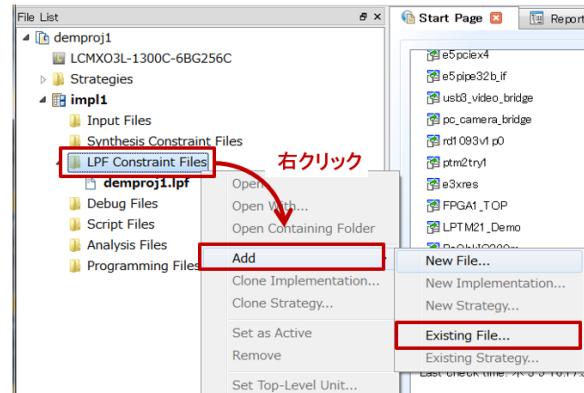
LPF 制約ファイルをインプリメンテーションに追加する場合、新規作成ではなく既存のものを編集したり、別名で複製してインポートした方が容易な場合があります。一般的には前述の通り ”スプレッドシート・ビュー” で作成・更新します。

また、ピン配置指定についての手法例があります。I/O タイミング / 動作周波数やデバイスとしての DRC に問題がないことが確認できている場合、一度 PAR まで処理プロセスを完了すると ”PAD レポート”（拡張子 .pad）が生成されます。その最終部分には LPF 文法に従ったピン配置指定の制約文がまとめられています。PAD レポートの形式は、この部分を LPF にコピー&ペーストすることを想定しています。

2.5.2 既存 LPF 制約ファイルのインポート

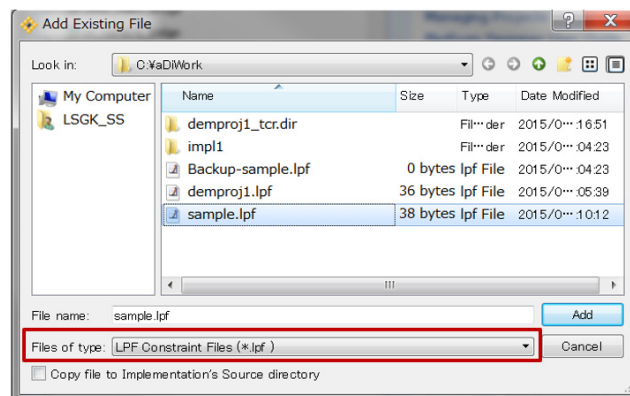
既存の LPF 制約ファイルをインポートする場合は、メニューバーから [File] → [Add] → [Existing File...] と選択するか、ファイルリスト・ビューでインプリメンテーション・セクション下 [LPF Constraint Files] 行を選択後右クリックし、表示されるメニューから [Add] → [Existing File...] の順に選択します（図 2-31）。

図 2-31. 既存 LPF 制約ファイルのインポート



”Add Existing File” ウィンドウ（図 2-32）が表示されます。

図 2-32. インポートする LPF 制約ファイルの選択

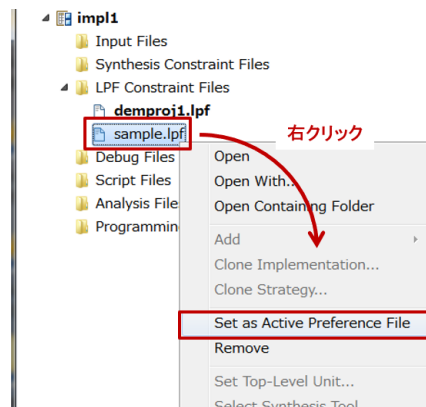


まず ”Files of Type” として [LPF Constraint Files (*.lpf)] を選択します。ブラウズして表示される LPF 制約ファイル候補から適切なファイルを選択し、『Add』ボタンをクリックしてインポートします。

ここで、ウィンドウ左下の [Copy file to Implementation's Source directory] にチェックが入っていると、インプリメンテーション・フォルダーにコピーが作成され、これがインポートされます。チェックが入っていない場合は、選択したファイルがインポートされます。

2.5.3 有効にする LPF 制約ファイルの選択（アクティブ化）

図 2-33. LPF 制約ファイルのアクティブ化

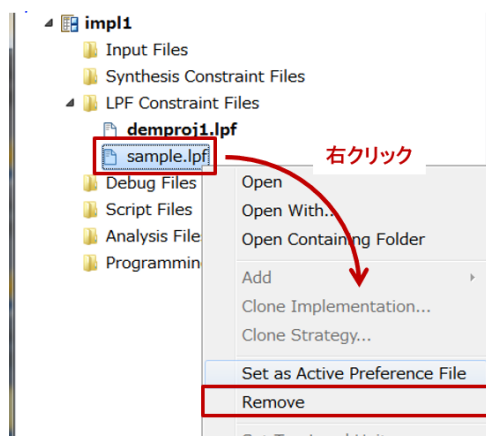


インプリメンテーションに複数の LPF 制約ファイルがインポートされている場合でも、有効なものは “アクティブ” な 1 つのみです。このため “LPF 制約ファイルのアクティブ化” が使用する LPF 制約ファイルの選択になります。アクティブ化するためには、ファイルリスト・ビューでアクティブでない LPF 制約ファイルを選択後右クリックし、表示されるメニューから [Set as Active Preference File] を選択します (図 2-33)。アクティブ化された LPF 制約ファイルは太字で表示され、以後のプロセス処理に適用されます。

2.5.4 LPF 制約ファイルの削除

不要になった LPF 制約ファイルは、プロジェクトから削除することができます。ファイルリスト・ビューで削除したい LPF 制約ファイルを選択後右クリックし、表示されるメニューから [Remove] を選択するか、選択した状態でキーボードの『Delete』キーを押します (図 2-34)。”プロジェクトから削除” はインポート情報の削除であり、ファイル自体は削除されません。

図 2-34. LPF 制約ファイルの削除



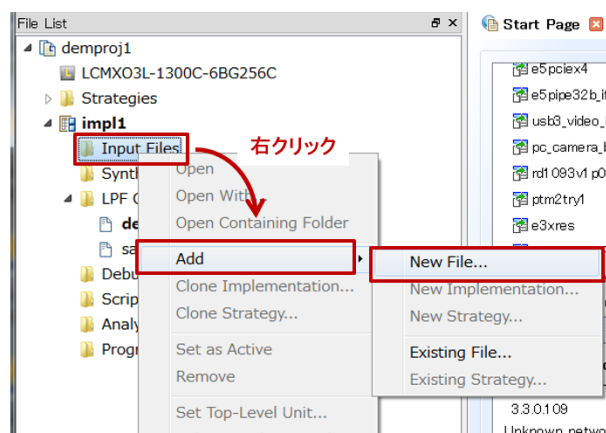
なお、アクティブな LPF 制約ファイルは削除できません (LPF 制約ファイルが 1 つしかない場合を含む)。

2.6 RTL ソースファイルの管理

2.6.1 新規 RTL ソースファイルの作成

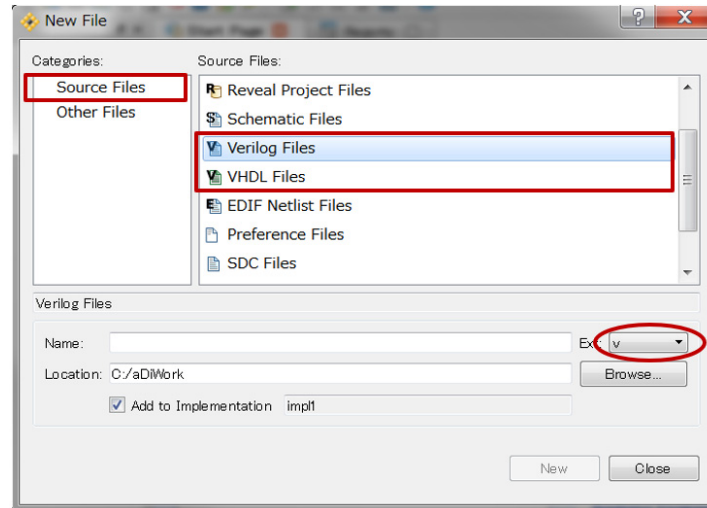
新規 RTL ソースファイルを作成する場合、メニューバーから [File] → [New] → [File...] と選択するか、もしくはファイルリスト・ビューでインプリメンテーション・セクション下の [Input Files] 行を選択後右クリックし、表示されるメニューから [Add] → [New File...] の順に選択します。”New File” ウィンドウが起動します。

図 2-35. 新規 RTL ソースファイルの作成



Categories 枠で "Source Files"、Source Files 枠で "VHDL Files" か "Verilog Files"（または "Schematic Files"）を選択し、「Name:」セルにファイル名を入力します。「Ext.」部（図 2-36 丸印）で拡張子を選択することもできます（これに応じて適切なものが幾つか表示されます）。

図 2-36. ソースファイルの追加 (New File ウィンドウ)

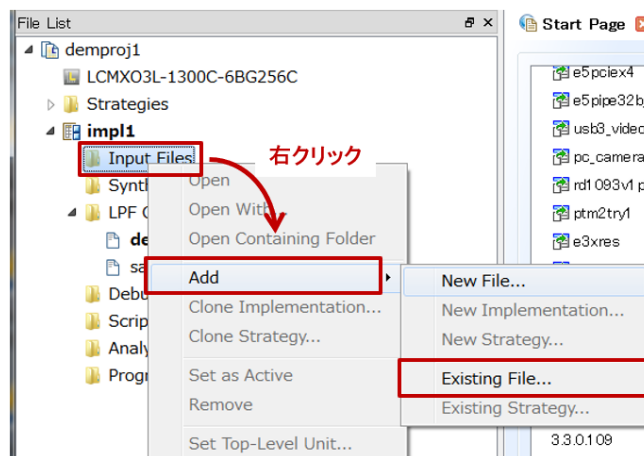


設定完了後に『New』ボタンをクリックするとファイルが作成され、テキストエディター（または回路図エディター）が立ち上がります。なお、「Add to Implementation」ボックスにチェックが入っていると、ファイルは作成した時点でインプリメンテーションにインポートされます。

2.6.2 既存の RTL ソースファイルのインポート

作成済みの RTL ソースファイルをインプリメンテーションにインポートする場合は、メニューバーから [File] → [Add] → [Existing File...] の順に選択するか、もしくはファイルリスト・ビューでインプリメンテーション・セクション下の [Input Files] 行を選択後右クリックし、表示されるメニューから、[Add] → [Existing File...] の順に選択します（図 2-37）。"Add Existing File" ウィンドウが表示されます（図 2-38）。

図 2-37. 既存のソースファイルのインポート

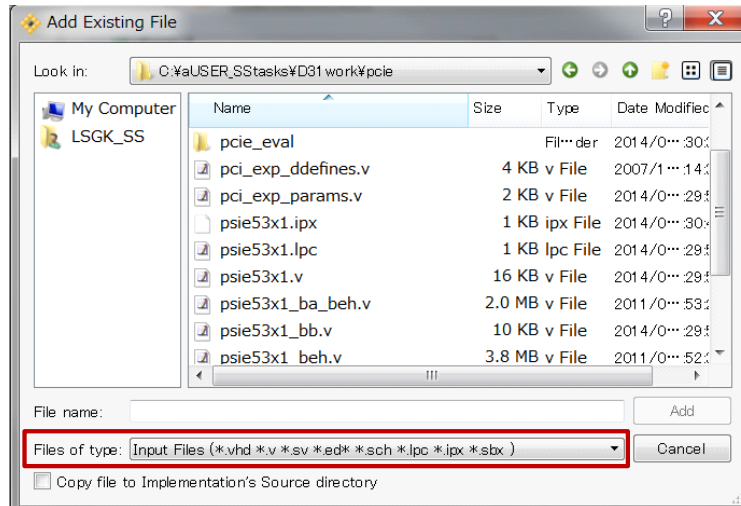


"Add Existing File" ウィンドウでは、まず "Files of Type" として "Input Files (*.vhd *.v ...)" を選択します。これでリストに RTL ソースファイル候補が表示されますので、適切なファイルを選択し『Add』ボタンをクリックします。

ここで、ウィンドウ左下の「Copy file to Implementation's Source directory」にチェックが入っていると、イ

ンプリメンテーション下の ”source フォルダ” にコピーが作成され、これがインポートされます。チェックが入っていない場合は、選択したファイルがインポートされます。

図 2-38. インポートするソースファイルの選択

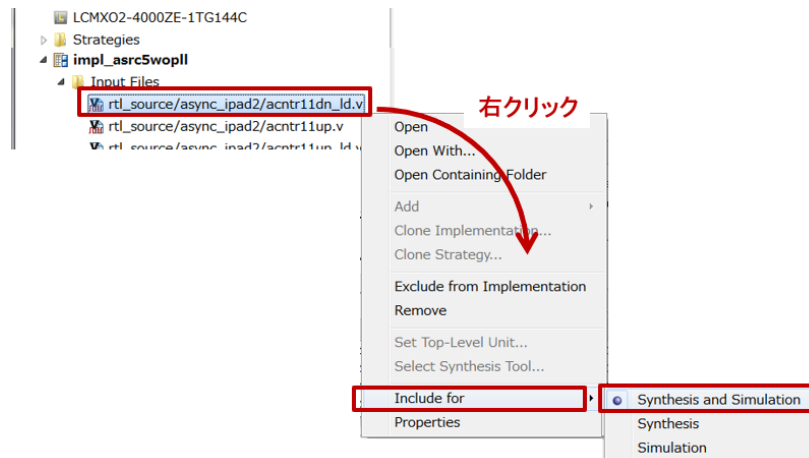


なお、Clarity Designer や IPexpress などのモジュール生成ツール（第 4 章参照）で生成したモジュール記述の場合は、RTL 記述のトップファイル（.v/.vhd）ではなく、それらの定義ファイル（拡張子 “.sbx” または “.ipx”）をインポートします。RTL ファイルをインポートしても構いませんが、その場合は後述の ”Exclude from Implementation” 指定をしておきます。RTL と .sbx/.ipx の両方が有効になっていると、予期しない処理結果になる場合があります。

2.6.3 対象プロセスの指定

インポートした RTL ソースファイルは、デフォルトで ”シミュレーションおよび論理合成” の対象として設定されます。対象とするプロセスを変更することができます。

図 2-39. RTL ソースファイルの対象プロセスを指定



当該 RTL ソースファイルを選択後、右クリックして表示されるメニューから [Include For] を選択するとリストが表示されますので、この中から 1 つを選択します。

[Synthesis] または [Synthesis and Simulation] を選択したソースは、論理合成の対象となります。[Simulation] または [Synthesis and Simulation] を選択したソースは、”Simulation Wizard”（第 10.2 節）で論理シミュレーション用のコンパイルスクリプトが自動作成される時に、デフォルトでコンパイル対象として含まれます。

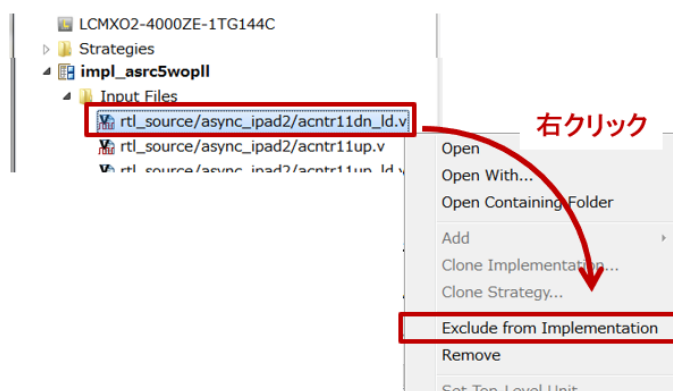
使用例

- ・ テストベンチをインポートして [Simulation] に設定
- ・ IP コアを使用する場合に、シミュレーション用のビヘイビアモデルと合成用のブラックボックスモデル両方をインポートしてそれぞれ [Simulation]、[Synthesis] に設定

2.6.4 使用しないソースファイルの扱い

RTL ソースファイルは、インポートしていてもインプリメンテーションごとに処理の対象から除外することができます。例えば（内部の記述が異なるが）同じ entity/module 名のファイルを複数インポートしておき、インプリメンテーション毎に使用するファイルを切り替えて使用することができます。

図 2-40. RTL ソースファイルの除外



ファイルリスト・ビューの [Input Files] セクションで、処理対象から外したいファイルを選択した後、右クリックして表示されるメニューから [Exclude from Implementation] を選択すると、そのファイルは処理対象から除外されます（図 2-40）。対象から除外されたファイルは、薄い文字で表示されます。

この設定は選択されているインプリメンテーションでのみ有効です。他のインプリメンテーションには反映されません。また "Remove" とは異なり、インポート自体は解除されません。

一度除外したファイルを再び処理対象に戻す場合は、同様に右クリックして表示されるメニューから [Include in Implementation] を選択します。

2.6.5 RTL ソースファイルの削除

不要になった RTL ソースファイルは、プロジェクトから削除できます。[Input Files] セクションで削除したいソースファイルを選択後、右クリックして表示されるメニューから [Remove] を選択するか、選択した状態でキーボードの [Delete] キーを押すと、プロジェクトから削除されます。ファイル自体は削除されません。

また、インポート済みの RTL ソースファイルを移動した場合など、ツールが見つからないと判断すると、当該ファイル名を取り消し線が表示します。削除するか、移動したものを元に複製するか戻すなどの対処をします（任意）。

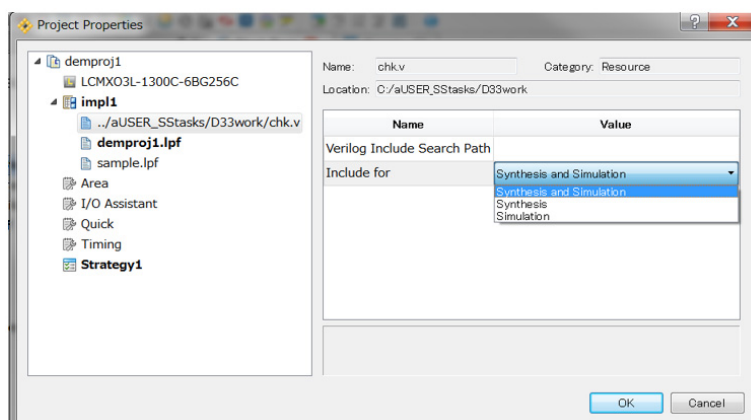
2.6.6 RTL ソースファイルのプロパティ設定

インポート済み RTL ソースファイルのプロパティとして、個別に以下の設定を行うことができます。

- ・ VHDL ソースのコンパイル先ライブラリー名
- ・ Verilog HDL ソース内でインクルードされているファイルの検索パス

第 2.3.5 節と同様の手順で "Project Properties" ウィンドウを表示します。ソースファイルを選択してから表示しても同様です。ウィンドウ左枠でプロパティを設定したいソースファイルを選択すると、ウィンドウ右側に設定可能な項目が表示されます（図 2-41）。

図 2-41. Project Property ウィンドウ～RTL ソースファイルの設定

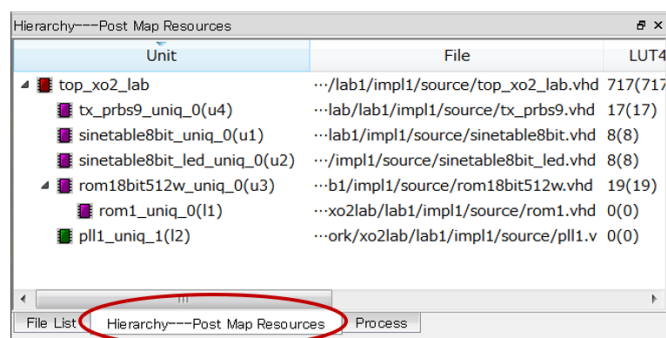


なお、インプリメンテーション作成時に既存のインプリメンテーションにインポートされている RTL ソースを参照した場合、これらのプロパティ設定も反映されます。従ってインプリメンテーション毎に設定しなおす必要はありません。

2.6.7 デザインの階層構造表示

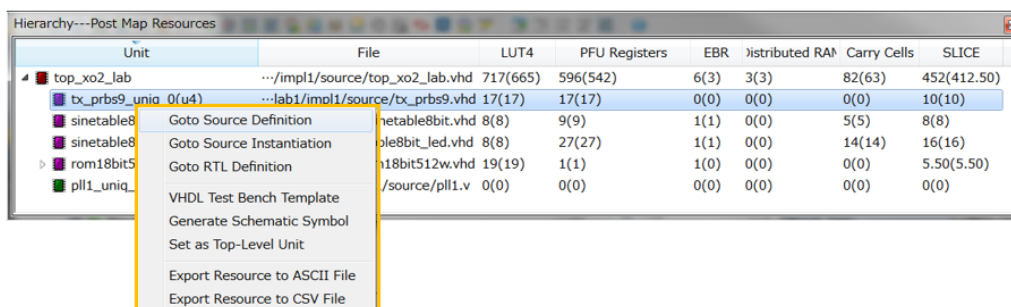
インポートしたファイルは、ファイルリスト・ビューではフラットに表示されるため、ファイルの過不足が直感的にはわかりませんが、”ハイアラーキー（Hierarchy）ビュー”でモジュールの階層構造を確認できます。

図 2-42. ハイアラーキー・ビューによるモジュールの階層構造表示



ハイアラーキー・ビューは論理合成前後で表示内容が変わります。論理合成後は LUT、FF など各リソースの使用数がモジュールごと、階層ごとに表示されます (図 2-42)。また、ソースファイルが VHDL か Verilog かでアイコンの色が異なります。

図 2-43. デタッチしたハイアラーキー・ビュー例でのアクション・メニュー



RTL 記述の想定する規模に合致するかどうかの、おおよその判断が可能です。また、回路規模を目標に収めるための方策を検討するベースになります。

図 2-43 のように特定のモジュールを選択して右クリックすると、メニューが表示されます。[Goto Source Definition / Instantiation] を選択すると、設定しているエディターが開き、モジュール記述やインスタンス箇所にジャンプして表示します。[Export Resource to ASCII / CSV File] ではハイアラーキー・ビューの表示内容を csv ファイルに書き出すことができます。

2.7 プロジェクト管理の注意点

2.7.1 LatticeECP3 SERDES マクロのパラメータファイル

LatticeECP3 PCS (SERDES) マクロの HDL ソースは、GUI ツール IPexpress で生成されますが、同時に動作モード設定等のパラメータを記述したテキストファイル ("autoconfig file"、拡張子 .txt) も生成されます。このパラメータファイルは、**インプリメンテーション・フォルダーに置いておく必要**があります。ECP5 では LatticeECP3 とは異なりますので、留意する必要はありません。

Export Files プロセス実行時にインプリメンテーションフォルダーにテキストファイルが無い場合、ツールは以下のようなメッセージを出力し、マクロはデフォルトのパラメータで処理が行われますので、通常は期待動作になりません。

WARNING – Cannot open configure file: [モジュール名].txt. Use the default setting !

このメッセージが表示された場合、IPexpress で設定したファイルの出力先から該当するテキストファイルをインプリメンテーション・フォルダーにコピーしてください。テキストファイルを File List の Input Files にインポートしても適用されませんので、ご注意ください。

2.7.2 Windows と Linux 間のデータ移転

Windows マシンで作成したプロジェクトを Linux マシンにコピーした場合、HDL ソースや制約ファイル等のインポート情報が Linux 上の Lattice Diamond 上で正しく認識されない場合があります。これは、Windows 上で作成したファイルを (そのまま) Linux 上へコピーした際に、フォントや文字コードの変換が原因でファイル名の大文字／小文字が変更されてしまうことがあるためです。Linux は大文字小文字を区別しますので、ファイル名の大文字／小文字が変換されると正しく認識できなくなります。

回避策は 2 つあります。

1. Linux にコピーしたソースファイル名を全て変更する (拡張子は小文字のまま)
2. Windows からコピーするファイルを tar と gzip で圧縮 (拡張子が tgz or tar.gz) し、コピーした圧縮ファイルを Linux 上で展開する

--- *** ---