

## 第 6 章 論理合成

本章では Lattice Diamond での論理合成の実行方法や、ストラテジ（Strategy）および設計制約の設定方法等について記述します。

### 6.1 論理合成ツール

Lattice Diamond では論理合成ツールとして Synplify Pro と Lattice Synthesis Engine（以下 LSE）が使用できます。Lattice Diamond 2.1 以降のバージョンでは、LSE が MachXO シリーズ（MachXO, MachXO2, MachXO3L）のデフォルトとなっています。LSE は、バージョン 3.3 時点でこれ以外のファミリには対応していません。

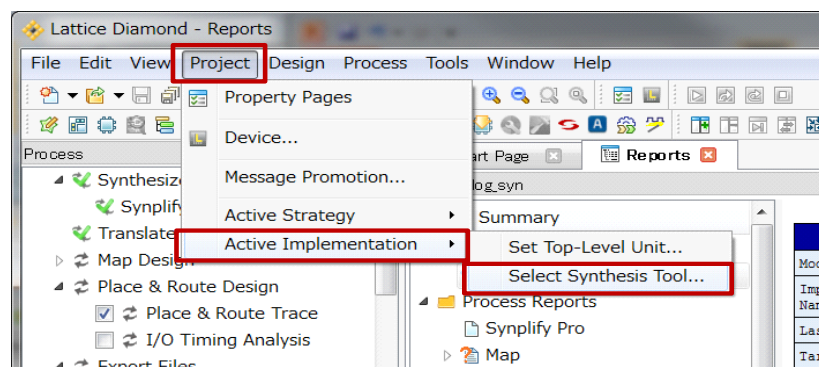
LSE を使用した場合は、論理合成と一緒に [Translate Design] プロセス相当の処理も同時に実行されます。

なお、**Diamond 2.1 および 2.2 からはストラテジ設定オプションの幾つかが新規追加、またはデフォルト指定の変更**が行われていますので、旧バージョンからのアップデートで同一プロジェクトを作業する場合、留意してください。

#### 6.1.1 使用する論理合成ツールの選択

論理合成ツールを選択するには、まず Lattice Diamond のメニューバーから [Project] => [Active Implementation] => [Select Synthesis Tool] を選択します（図 6-1）。

図 6-1. 論理合成ツールの選択



表示される図 6-2 のような Project Properties ウィンドウで Name 欄 Synthesis Tool 項に（デフォルトで）選択されている論理合成ツールが Value 欄に表示されています。Value 欄のどこかをクリック数とプルダウン形式で選択可能なツール名が表示されます。個々の例では Synplify Pro と LSE です。所望のツールを選択後、OK ボタンをクリックします。選択したファミリに対して選択できるツールのみが候補として表示されます。

或いは、プロジェクトやインプリメンテーション生成後に変更することも可能です。二通りの方法があります。第一の方法は図 6-3 のようにインプリメンテーション名を選択後右クリックし、[Select Synthesis Tool...] を選びます。第二の方法は同様にインプリメンテーションを右クリック後 [Properties] を選択します。いずれも図 6-2 とおなじプロパティ・ウィンドウが表示されますので、選択し直します。

図 6-2. 論理合成ツールの選択 (プロパティ・ウィンドウ)

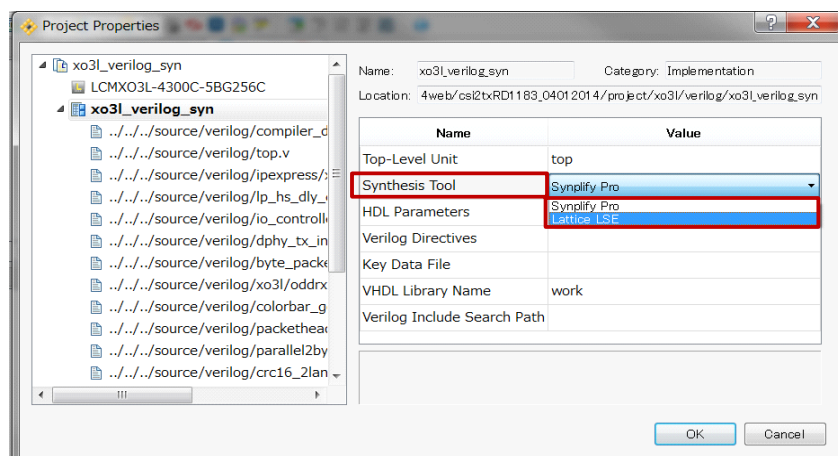
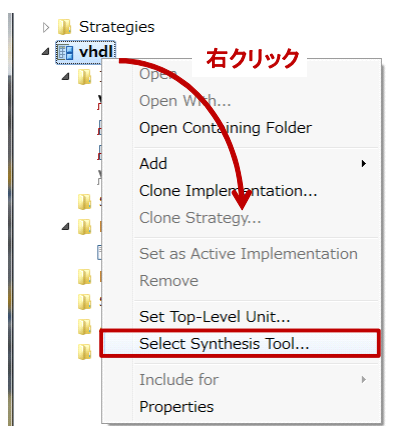


図 6-3. 論理合成ツールの設定変更

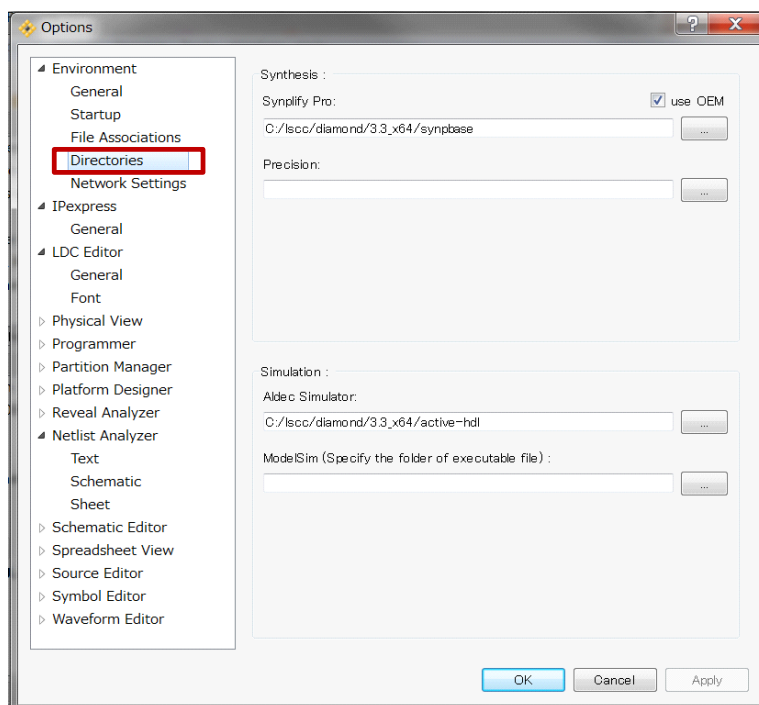


## 6.1.2 バンドル版以外のツールを使用する場合の設定

論理合成ツールの実行パス設定を行うことで、Lattice Diamond にバンドルされている論理合成ツール以外や別バージョンの Synplify Pro を使用できます。Lattice Diamond のメニューバーから [Tools] => [Options...] を選択するとオプション設定ウィンドウが起動します (図 6-4)。ウィンドウ左側のパネルで [Environment] => [Directories] を選択すると、右側はパス設定入力表示になります。ウィンドウ上部に Synplify Pro と Precision のパス設定欄が用意されています。

Synplify Pro のパス設定は、デフォルトではバンドル版のパスが設定されています。Precision のパス設定欄はデフォルトでは空白になっています。必要に応じてこれらのパス設定を変更します。Lattice Diamond にバンドルされている Synplify Pro を選択した場合、ウィンドウ右上の [use OEM] のチェックは外さないでください。これ以外の Synplify Pro の場合は、[use OEM] のチェックを外してください。

図 6-4. Lattice Diamond のオプション設定ウインドウ ～ ツールのパス設定



### 6.1.3 LSE を使用する際の留意事項

LSE にはサポートするデバイスファミリに制限があり、Lattice Diamond 3.3 までは MaxhXO シリーズ (XO, XO2, XO3L) のみが対象です。これら以外のファミリは 3.4 以降順次正式サポートしていきます。ECP5 はバージョン 3.4 からです。

## 6.2 LSE

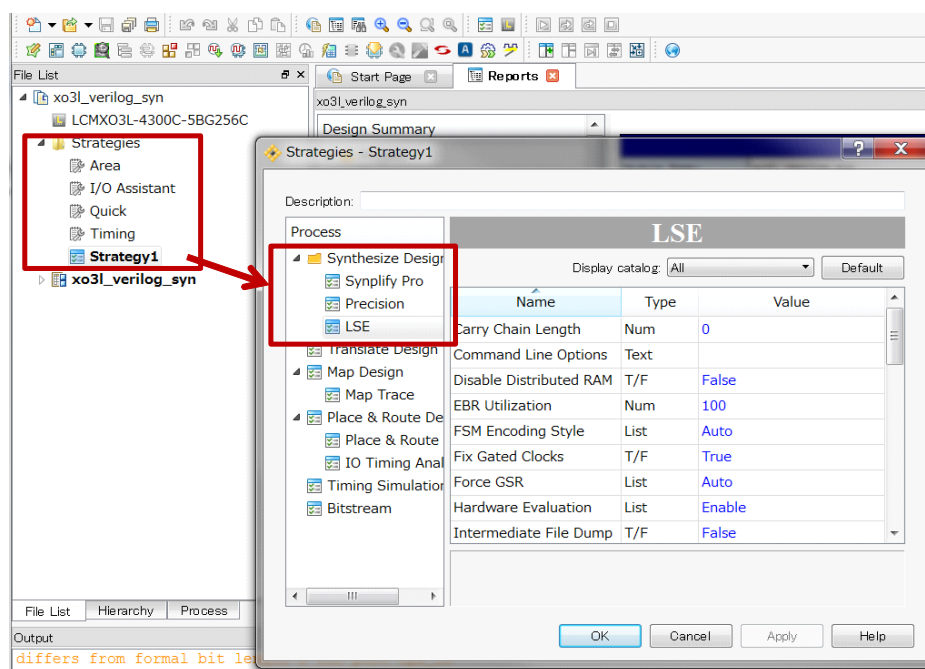
### 6.2.1 ストラテジ設定ウインドウ

LSE のストラテジ・オプションについて記述します。

プロジェクト・ナビゲータ左上に配置されている File List ウインドウに、Project にインポートされているストラテジの一覧が表示されています。複数ある中の太字表示が、インプリメンテーションに適用されます。ストラテジ名をダブルクリックすると、ストラテジ設定ウインドウが開きます。

論理合成ツールとして LSE を指定していれば、ストラテジ設定ウインドウを立ち上げた場合に、図 6-5 のように『LSE』がすで選択されています。

図 6-5. LSE のストラテジ設定ウィンドウ



## 6.2.2 LSE ストラテジ・オプションの詳細

以下にストラテジ・オプション各設定の詳細を説明します。

### Carry Chain Length

パラメータ：数値

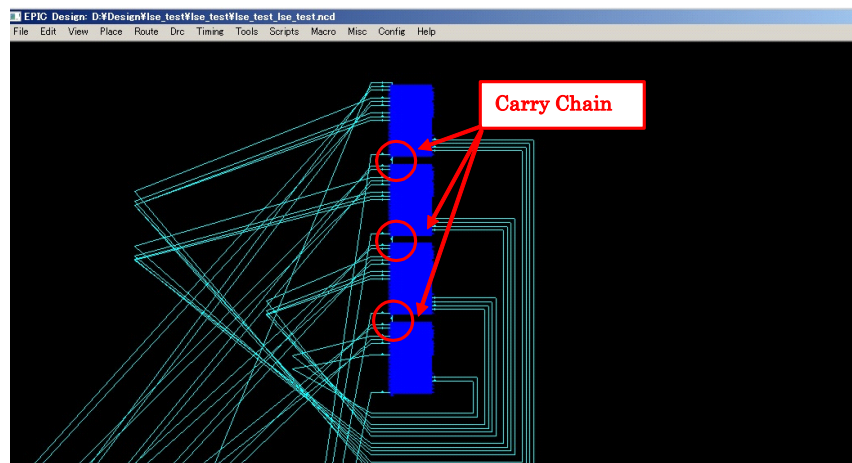
デフォルト値：0

演算回路をインプリする際に使用される桁上がり／桁下がり伝播専用配線である Carry Chain の使用を制限するためのオプションです。

設定値は、SLICE が桁上がり／桁下がりを使用するモードで実現するデータの最大 bit 幅の許容値です。最大で（設定値 + 1）までのカウンタや演算回路が Carry Chain を使用して実現されます（1 ビット目は桁上がりが必要ないため、この設定の対象にならない）。[0] は 0bit ではなく無制限であることを表します。

なお、この設定は [Use Carry Chain] オプションが [True] 設定の場合のみ有効になります。

図 6-6. Carry Chain を使用した SLICE 間接続



### Command Line Options

パラメータ：文字列                      デフォルト値：ブランク

このドキュメントで紹介している GUI のリストに表示されていない特別なオプションを使用する場合に、直接引数等を記述します。

### Disable Distributed RAM

パラメータ：True / False                      デフォルト値：False (Diamond 3.2 から追加)

[True] の場合、RTL 記述から推論してメモリを使用した回路を生成する際に分散メモリ (Distributed Memoy) を使用せず EBR を用います。

### EBR Utilization

パラメータ：数値                      デフォルト値：100

RTL 記述から推論して EBR を使用した回路を生成する際に、ターゲットとする EBR の使用率の上限を設定します。設定した上限を超えた場合、残りはレジスタか分散メモリ (Distributed Memoy) で実現されます。IPexpress で生成した EBR の RTL ソースを使用した場合は、このオプションの適用外です (全て EBR で実現されます)。

### FSM Encoding Style

パラメータ：Auto / One-hot / Gray / Sequential                      デフォルト値 Auto

VHDL の列挙 (enum) 型変数が定義されている場合、その変換形式に関する設定です。Auto (デフォルト) の場合は、ステート数等から適当な構成のステートマシンが構成されます。Auto 以外を選択した場合は、それに沿った構成のステートマシンが構成されます。

### Fix Gated Clocks (Diamond 3.2 から追加)

パラメータ：True / False                      デフォルト値：True

ソース内でクロックゲーティングしているクロック信号を実装に適する構成に変換します。対象クロックは .ldc ファイルに明記されている必要があります。ゲーティング制御用の信号には特定の要件があります。

変換されたクロックと関連するレジスタ情報はログファイル『synthesis.log』に書き出されます。

### Force GSR

パラメータ：Auto / Yes / No                      デフォルト値：Auto

論理合成時の GSR (Global Set/Reset) 挿入に関する設定です。

[Auto] / [Yes] を選択した場合は自動的に GSR バッファが挿入されます (ただし、Set/Reset の構成が条件を満たした場合のみ)。[No] を選択した場合は GSR が使用されません。

GSR はマッピング (Map Design) プロセスでより最適な条件で挿入することができます。特別な理由のない限りこのオプションは [No] を選択することを推奨します。

### Hardware Evaluation

パラメータ：Enable / Disable                      デフォルト値：Enable

IP コアの評価に関する設定です。IP コアをライセンスなしで評価する場合は [Enable] (デフォルト) を選択します。IP ライセンスの入手後の設定はどちらでもかまいません。

### Intermediate File Dump

パラメータ：True / False                      デフォルト値：False

暗号化されたファイル出力可否の設定です。[False] (デフォルト) ではファイルは出力されません。

[True] を設定すると、edif の他に暗号化された Verilog HDL ソースファイルでも出力されます。出力されるファイルの拡張子は [.ve] です。

\*\*\*\*\*-----

- ・ トラブル対応の際などにこの暗号化されたファイルを渡せば、HDL ソースファイルから論理記述を読み取られる心配がなくなります

\*\*\*\*\*-----

### Loop Limit (Diamond 3.2 から追加)

パラメータ：数値

デフォルト値：1950

RTL ソース記述内で用いられる "for" や "while" ループ回数の最大値を与えます。インデックスが変数として与えられている場合のみ適用され、定数の場合は適用されません。余り大きい値を指定すると、スタックオーバーフローを生じる可能性があるため、注意します。

### MUX Style

パラメータ：Auto / L6Mux Multiple / L6Mux Single / PFU MUX デフォルト値：Auto

SLICE 内部のマルチプレクサの使用方針に関する設定です。[Auto] (デフォルト) では LSE が最適と判断したマルチプレクサが使用されます。

[L6Mux Multiple] を選択した場合は、SLICE 内の [L6Mux] (図 6-7 ①) を優先的にカスケード接続して使用します。

[L6Mux Single] を選択した場合は、SLICE 内の [L6Mux] (図 6-7 ①) を優先的に使用しますが、[L6Mux] のカスケード接続は行われません。

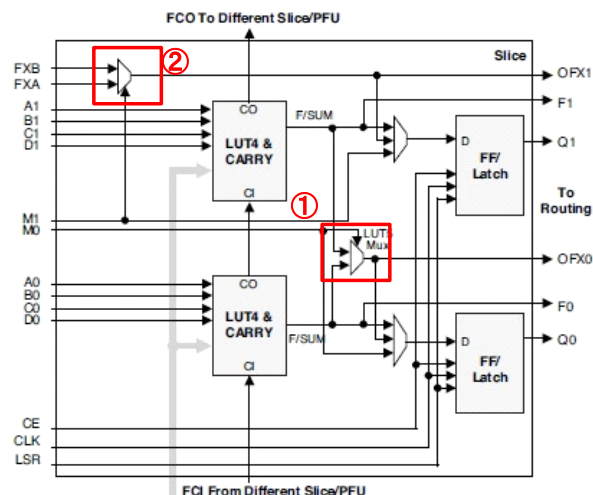
[PFU Mux] を選択した場合は、SLICE 内の [PFU Mux] (図 6-7 ②) を優先的に使用します。

\*\*\*\*\*-----

- ・ 傾向として [L6Mux Multi] < [Auto] < [L6Mux Single] < [PFU Mux] の順に回路規模が大きくなります。ただし、ロジック段数は [PFU Mux] < [L6Mux Single] < [Auto] < [L6Mux Multi] の順に増える傾向があります。どの設定が最適かはデザインに依存します。Auto のままにすることを推奨します。

\*\*\*\*\*-----

図 6-7. MUX Style オプションの影響を受ける SLICE 内のマルチプレクサ



### Macro Search Path

パラメータ：フォルダパス デフォルト値：ブランク

マクロのライブラリパス設定です。デザインに IP コア (ネットリスト提供) や EPIC で生成したマクロを使用している場合、そのネットリストファイル (\*.ngo) またはライブラリファイル (\*.nmc) のあるフォルダパスを記述します。IP やマクロを使用していない場合はブランクのままでも問題ありません。

プロジェクトフォルダ内にファイルが置かれている場合は、ブランクのままで構いません。複数のフォルダにファイルが有る場合は、[ ; ] で区切って全てのパスを記述してください。この際、間にスペースは入れないでください。パスの記述は相対パスにも対応しています。

Synplify Pro を使用した場合、[Translate Design] プロセスのオプションです。

### Max Fanout Limit

パラメータ：最大ファンアウト数      デフォルト値：1000

ファンアウト数の上限の制約設定です。ファンアウトがこの値を超えた場合は、ドライバをコピーしてファンアウトを減らす処理が行われます。ただし、この制約はあくまでガイドラインであり、制約を満たすことにより最適化が妨げられる場合は無視されます。

### Memory Initial Value File Search Path

パラメータ：フォルダパス      デフォルト値：ブランク

EBR（ブロックメモリ）の初期値設定を行う際に、初期値を記述したファイル（\*.mem）が保存されているフォルダパスを指定します。何も指定しない場合は、インプリメンテーション・フォルダが参照されます。

EBR の初期値設定を行わない場合は、設定の必要はありません。また、IPexpress で EBR を生成する際に初期値ファイルを指定している場合は、ここで再度指定する必要はありません。

### Number of Critical Paths

パラメータ：パス数      デフォルト値：3

論理合成後のタイミング検証結果として出力するクリティカルパス数の設定です。レポートは、[TOP モジュール名 ]\_lse.twr ファイルに出力されます。

### Optimization Goal

パラメータ：Area / Balanced / Timing      デフォルト値：Balanced

論理合成で優先するパラメータを決める設定です。[Area]（デフォルト）では、リソース数が最小になるように論理合成が行われます。

[Balanced] を選択した場合は、リソース数とタイミングのバランスを考慮した論理合成が行われます。

[Timing] を選択した場合は、動作周波数が最大となるように論理合成が行われます。

### Output Preference File（Diamond 3.2 から削除）

パラメータ：True / False      デフォルト値：False

論理合成の際に使用したタイミング制約を、lpf（論理合成以降で使用する制約ファイル）に出力させる設定です。[False]（デフォルト）では、lpf ファイルは出力されません。

[True] を選択した場合は、lpf ファイルに制約が出力されます。なお、このファイルは出力されるだけで、そのまま以降の処理で参照されるわけではありません。これを使用する場合は、Lattice Diamond にインポートしてアクティブな制約として設定する必要があります。

### Propagate Constant

パラメータ：True / False      デフォルト値：True

論理合成の際に、固定値または値が固定される回路の最適化に関する設定です。[True]（デフォルト）では、このような回路は最適化され削除されます。

[False] を選択した場合は、回路が最適化されずに残ります。

### RAM Style

パラメータ：Auto / Block RAM / Distributed / Register      デフォルト値：Auto

2次元配列等で RAM が推定される記述の実現方法（リソース）の設定です。[Auto]（デフォルト）では LSE がサイズから適当な方法を選択します。

[Block RAM] を選択した場合は、EBR が使用されます。



[Distributed] を選択した場合は、分散 RAM が使用されます。

[Registers] を選択した場合は、レジスタが使用されます。

なお、シフトレジスタが推定され RAM を使用して構成された場合でもこの設定の影響を受けません。

### ROM Style

パラメータ : Auto / EBR / Logic      デフォルト値 : Auto

2次元配列等で ROM が推定される記述の実現方法（リソース）の設定です。[Auto]（デフォルト）では LSE がサイズから適当な方法を選択します。

[EBR] を選択した場合は、EBR が使用されます。

[Logic] を選択した場合は、ファブリック内のロジックリソースで固定値の配列が構成されます。

### Remove Duplicate Registers

パラメータ : True / False      デフォルト値 : True

同じ機能（論理）を実現するレジスタが複数あった場合の、最適化（削除）に関する設定です。[True]（デフォルト）では、このようなレジスタはリソース数削減のために削除されます。[False] を選択した場合は、削除されずに残ります。

### Remove LOC Properties

パラメータ : On / Off      デフォルト値 : Off

HDL ソース内に記述されたピンアサイン情報に関する設定です。[Off]（デフォルト）では HDL ソース内のピンアサイン情報は以降のプロセスに引き継がれます。ただし、HDL ソースと制約ファイル (\*.lpf) の設定が異なる場合、以降のプロセスでは制約ファイルの設定が優先されます。

[On] を選択した場合、HDL ソース内のピンアサイン情報は以降のプロセスに引き継がれません。

### Resolve Mixed Drivers (Diamond 3.2 から追加)

パラメータ : True / False      デフォルト値 : False

[True] を選択すると、アクティブな信号および VCC（または GND）が同時にドライブしている場合、VCC（または GND）で固定します。

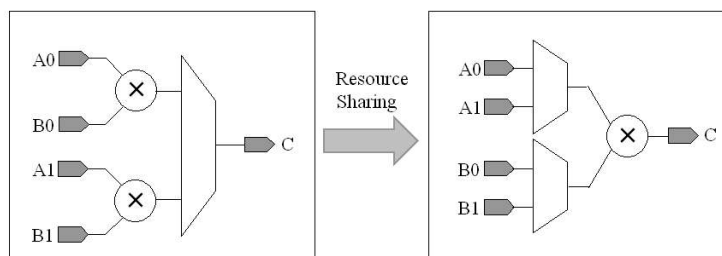
### Resource Sharing

パラメータ : True / False      デフォルト値 : True

リソースシェアリング（リソース共用の最適化処理）に関する設定です。

[True]（デフォルト）を選択した場合、使用するリソースを減らすために、複数ある同一機能の回路を共有させる処理が行われます（図 6-8）。[False] を選択した場合、この処理は行われません。

図 6-8. Resource Sharing 処理の一例



### Target Frequency

パラメータ : 数値（周波数）      デフォルト値 : 200

目標動作周波数の設定です。この設定は [Optimization Goal] オプションが [Area] 設定以外の場合に有効になります。また、この設定はデザイン内の全てのクロックが対象となります。クロック個別に制約を設定する場合は、sdc ファイルに制約を記述してください。



ただし、[Optimization Goal] オプションが [Area] に指定されている場合、本値は無視され、1 MHz を適用します。

### Use Carry Chain

パラメータ：True / False      デフォルト値：True

演算回路実装の際の Carry Chain の使用可否の設定です。[True]（デフォルト）では演算回路の実装に Carry Chain を使用することが許可されます（使用するかどうかは回路構成と [Carry Chain Length] オプションの設定に依存します）。

[False] を選択した場合は、演算回路の実装に Carry Chain が使用されません。

### Use IO insertion

パラメータ：True / False      デフォルト値：True

論理合成の際の IO バッファ挿入に関する設定です。[True]（デフォルト）では論理合成時に最上位階層のポートに IO バッファが挿入されます。

[False] を選択した場合は、IO バッファは挿入されません。

### Use IO Registers

パラメータ：True / False      デフォルト値：True

論理合成時の IO レジスタ使用の可否設定です。[True]（デフォルト）では IO レジスタを使用することが許可されます（使用するかどうかは回路構成に依存します）。

[False] を選択した場合は IO レジスタは使用されません。

### Use LPF Created from SDC in Project (Diamond 3.2 から追加)

パラメータ：True / False      デフォルト値：True

[True]（デフォルト）では制約ファイル .ldc に基づいた .lpf ファイルを生成します。

### VHDL 2008 (Diamond 3.2 から追加)

パラメータ：True / False      デフォルト値：False

VHDL バージョン指定のためのオプションです。[True]（デフォルト）ではプロジェクトの VHDL 言語仕様として IEEE 1076-2008 を選択します。

## 6.2.3 LSE 論理合成制約の設定

### 6.2.3.1 制約ファイルの管理

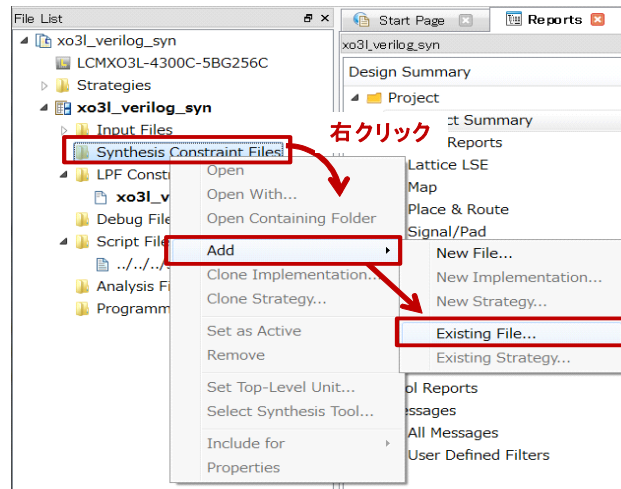
本節では論理合成用制約ファイルの管理方法について説明します。

Lattice Diamond では、論理合成用の制約ファイル (Synthesis Preference File) をインポートすることで論理合成ツールにタイミング制約や論理合成方法の指示を与えることができます。ストラテジ設定はデザイン全体（グローバル）に適用されるオプション式であるのに対して、論理合成制約ファイルを用いることで、例えばクロックネット毎に周波数制約を与える、などといったより詳細なコントロールが可能になります。制約ファイルがなくても論理合成処理自体は支障ありませんが、生成される EDIF の質は異なります。

#### 6.2.3.2 既存の制約ファイルのインポート

プロジェクト作成時には、論理合成制約ファイルは何もインポートされていません。既存の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、File List ウィンドウ内『Synthesis Constraint Files』行を右クリックし（図 6-9）[Add] => [Existing File...] を選択します。

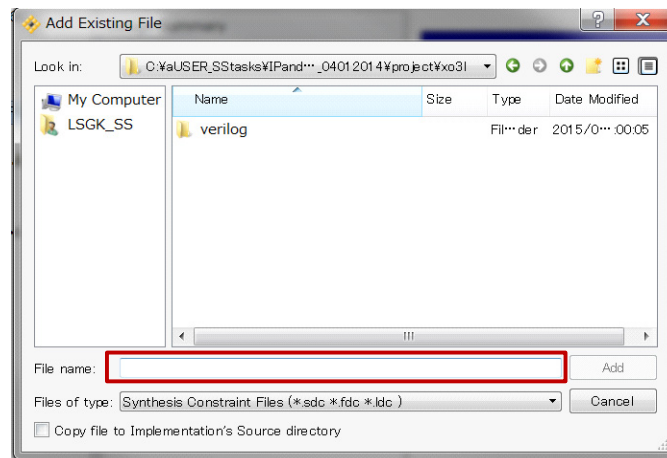
図 6-9. 既存の論理合成制約のインポート



これで、インポートするファイルの選択ウインドウ（図 6-10）が立ち上がり（File of Type は [Synthesis Constraint Files]）、フォルダ内にある制約ファイル候補が表示されるので、ブラウズして適当なファイルを選択後、[Add] ボタンをクリックします。

この際、ウインドウ左下の [Copy file to ...] にチェックが入っていると、インプリメンテーション・フォルダにコピーが作成され、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リンクされてインポートされます。

図 6-10. インポートする制約ファイルの選択

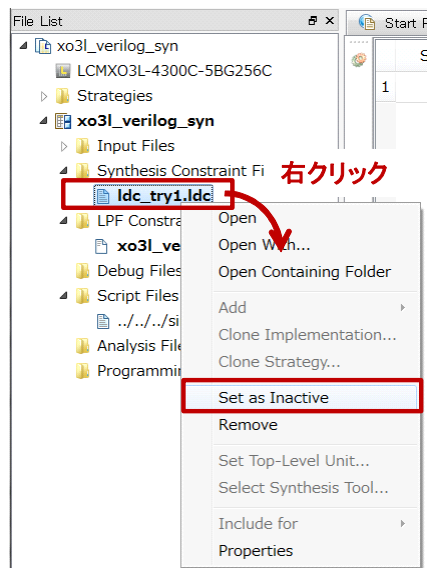


## 6.2.3.3 適用する制約ファイルの選択

インプリメンテーションでは、プロジェクトに複数の論理合成制約をインポートすることができ、デフォルトではインポートした全ての論理合成制約が適用されます。インポートされている論理合成制約を適用しないようにするには、論理合成制約の非アクティブ化を行います。File List ウインドウでアクティブな論理合成制約を右クリックし、[Set as Inactive] を選択します（図 6-11）。これでインプリメンテーションには適用されなくなります。非アクティブにした論理合成制約をアクティブ化するには、同様の手順で [Set as Active LDC]（または SDC）を選択します。

アクティブな論理合成制約は太字で、非アクティブな論理合成制約は細字で表示されます。

図 6-11. 論理合成制約の非アクティブ化



#### 6.2.3.4 論理合成制約の削除

不要になった論理合成制約は、プロジェクトから削除することもできます。図 6-11 と同様に右クリックし [remove] を選択するか、選択した状態でキーボードの [Delete] キーを押すことでプロジェクトから削除されます。「プロジェクトから削除」とは「インポート情報の削除」で、ファイル自体の削除ではありません。

#### 6.2.3.5 LSE 用新規制約ファイルの作成

論理合成制約ファイルを新たに作成して追加します。メニューバーから [File] => [New] => [File...] の順に選択するか、File List ウィンドウ内で『Synthesis Constraint Files』行を右クリックし、[Add]=> [New File...] を選択します。New File Wizard 図 6-13 が起動します。

図 6-12. 新規論理合成制約の追加

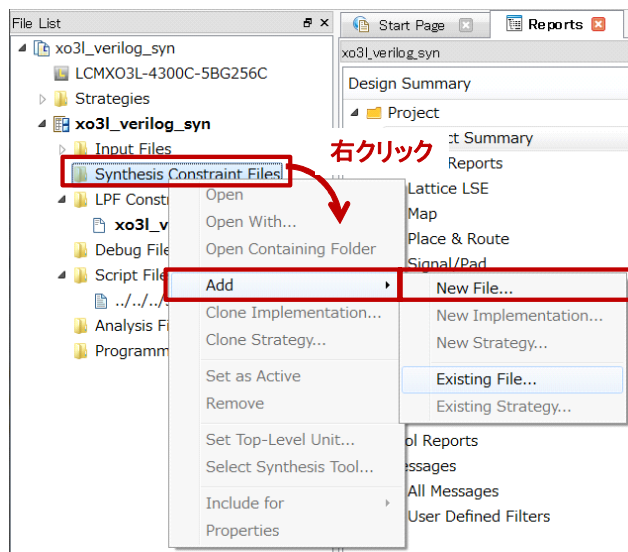
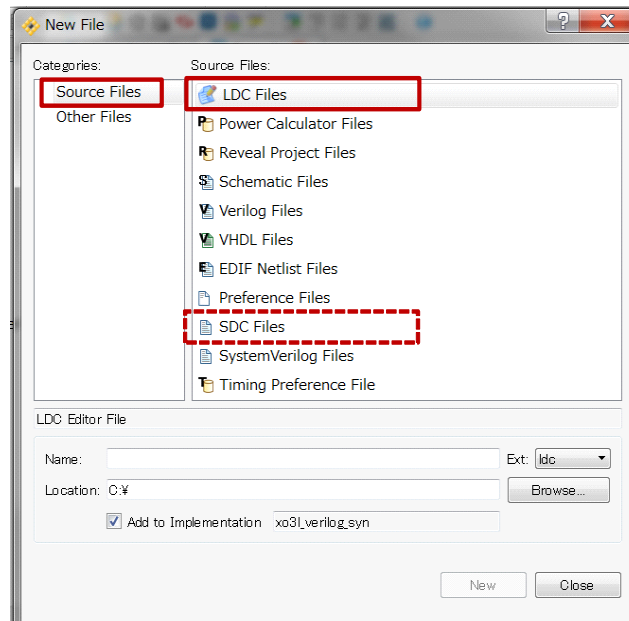


図 6-13. New File Wizard (新規 LDC ファイルの追加)



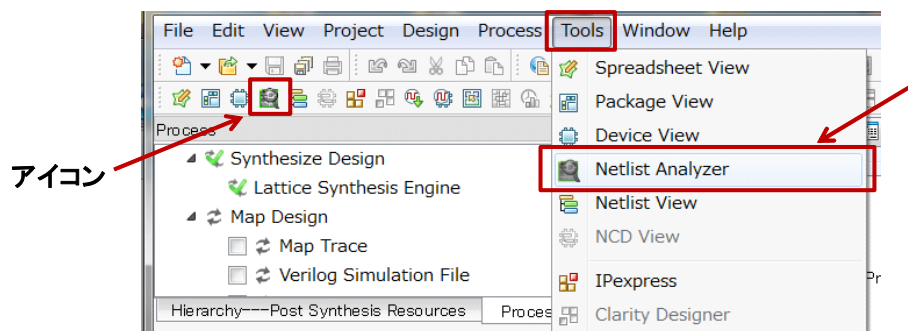
Categories 欄で [Source Files]、Source Files 欄で [LDC Files] または [SDC File] を選択し、Name 欄にファイル名を入力します。適切なフォルダを指定後に New ボタンをクリックすると、新しい論理合成制約ファイルが作成されます。

SDC の場合はテキストエディタが、LDC の場合は LDC Editor (LDC 設定用の GUI ツール) が立ち上がります。

## 6.2.4 Netlist Analyzer

LSE を論理合成ツールとして選択した場合の回路図情報をグラフィカルに閲覧するツールが Netlist Analyzer です。起動するにはアイコンをクリックする (図 6-14、左) か、Tools メニューから選択する (図 6-14、右) ことによって行います。

図 6-14. Netlist Analyzer の起動




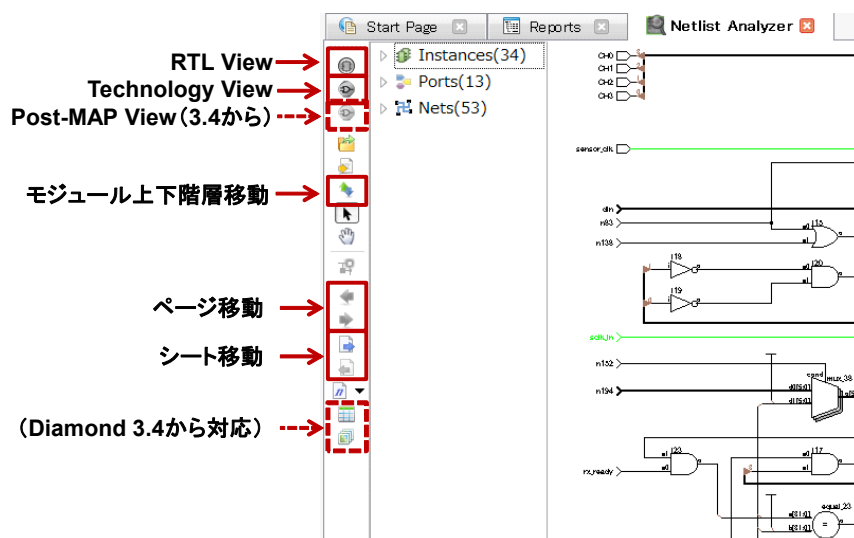
起動後の初期画面は図 6-15 のようになります (一部) ます。枠右上すみの  アイコンをクリックして Netlist Analyzer 表示枠全体をデタッチ (独立に) することを推奨します。実装回路規模が小さい場合を除き、閲覧するには細かすぎるためです。RTL View のスキマティックがデフォルトで表示されます。

図 6-15. Netlist Analyzer 起動後のアイコン表示



アイコン列はウィンドウの左側に並んでいます。上部には RTL View / Netlist View (および Diamond 3.4 から Post-MAP View) 各表示を起動するアイコンがあります。

その下にはモジュール階層の上下移動のために矢印のアイコンがあります。スキマティック表示内で（サブ）モジュールの箱の上にアイコンを移動すると、下位階層がある場合はアイコンが下向き青矢印表示になり、クリックすることで下位層のスキマティック表示になります。上位階層への移動は何も回路素子がない一にアイコンを移動する事で、上向き緑矢印が表示されますのでクリックします。

アイコン列下部にはページやシートの移動用の左右向き矢印アイコンがあります。

(Diamond 3.4 からはさらにその下に表示ウィンドウの様式を変えるアイコンが追加されます。)

アイコン列の右枠はリソース表示ウィンドウです。インスタンス、ポート、ネットの三つに分類されています。

図 6-16. Netlist Analyzer によるリソースリスト確認

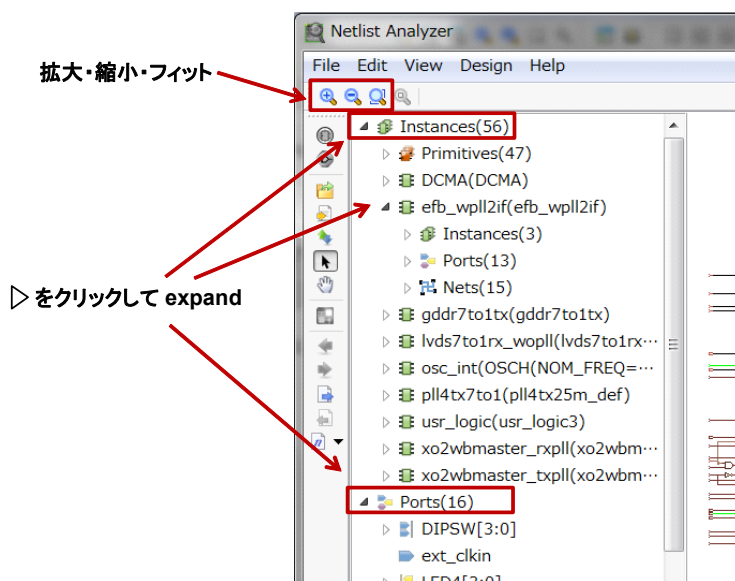
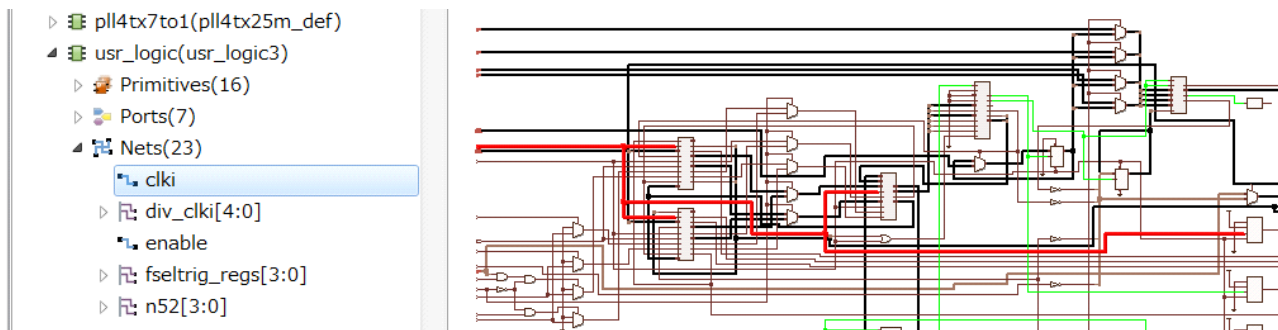


図 6-16 のようにリソース（名）行頭の三角マークをクリックしてエクスパンド（その下位や内容を表示）

します。アルファベット順にいとされています。特のモジュールはポート、ネットをクリックして選択すれば、図 6-17 のようにスキマティック部の該当リソースが赤表示でハイライトされます。

図 6-17. Netlist Analyzer によるハイライト



### オブジェクトのサーチ

特定のリソースやネットを探すときの機能『find』があります。Netlist Analyzer を起動後、スキマティック窓内で回路記号がブランクの領域にマウスを移動して、右クリックし、『Find...』を選択します。

図 6-18. Find ダイアログの起動

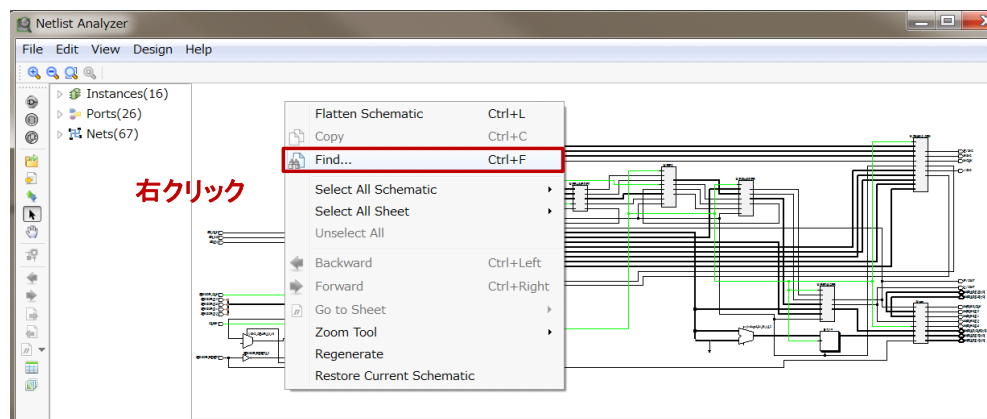
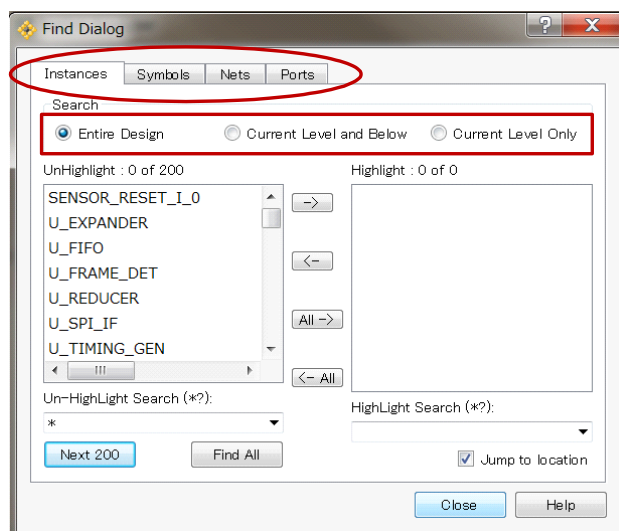


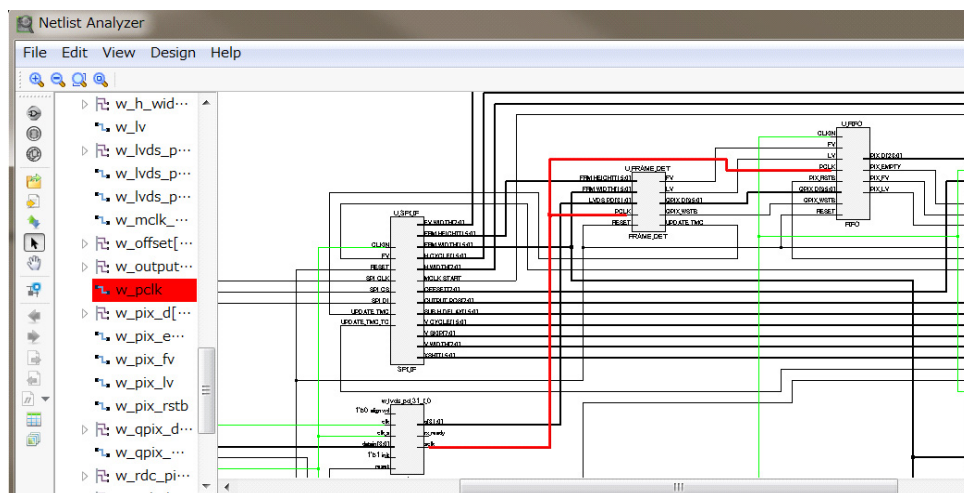
図 6-19. Find Dialog ウィンドウ



表示される図 6-19 のダイアログで、上部タブの『instances』、『Symbols』、『Nets』、『Ports』から意図する対象リソースを選択します。次いでその下 Search セクションでサーチする階層を指定します。デフォルトは『Current Level Only』です。Current Level は本ダイアログを立ち上げた際にマウスクリックした解像度です。

その後、左側に表示される対象候補を選択して矢印『->』などを操作して『Highlight』部に移動させて”Close”をクリックします。各矢印で選択指定と解除を適切に行います。

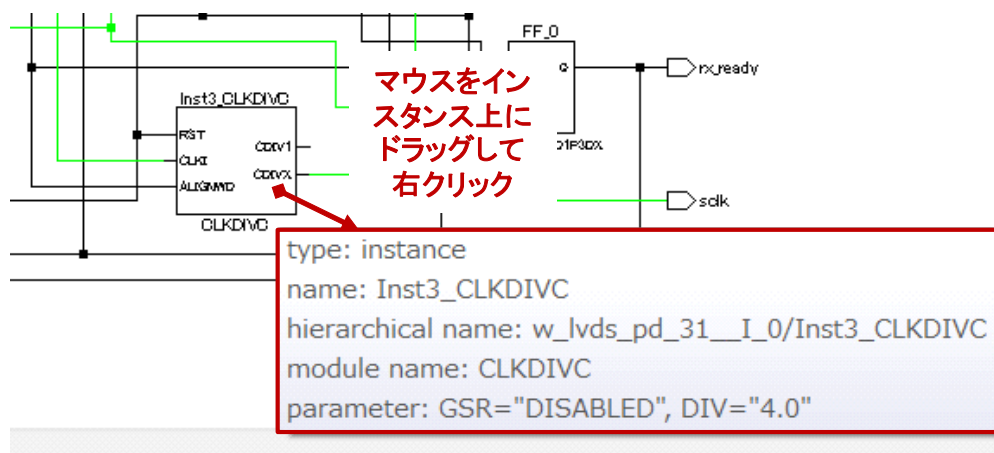
図 6-20. クロックネットワークを Find 指定した後の表示例



### マクロのパラメータ確認

Netlist Analyzer では、デザイン内で私用されている特定マクロのパラメータ設定を確認することができます。『Technology View』を選択して回路図を表示後、意図するマクロの上にマウスを移動させます。そのマクロで定義されているパラメータに対して、どのように設定されているかが表示されます。

図 6-21. マクロ (CLKDIV) の表示例



## 6.2.5 LDC エディタを使用した LSE の制約設定

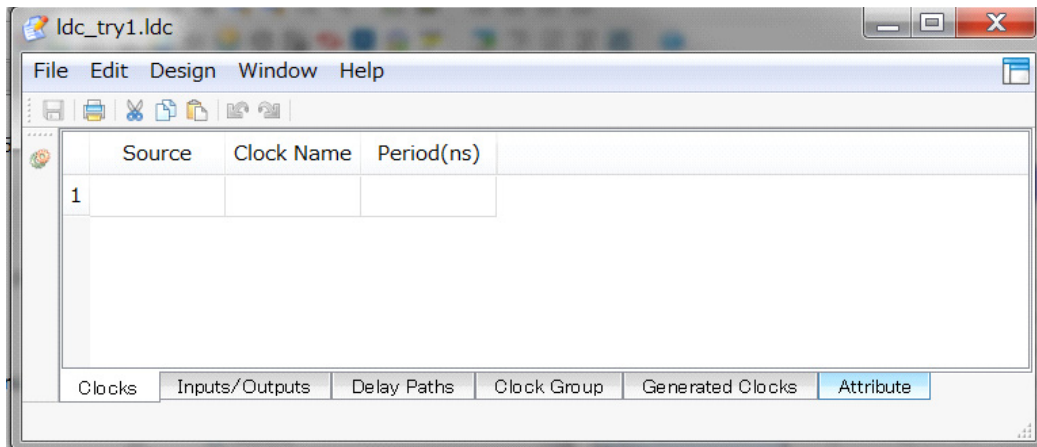
本節では LDC エディタの使い方等について記述します。

**【重要】** Diamond 3.3 時点では、(可能な限り言及していますが) 以降に記述する制約設定が有効に作用しないケースや制約項目があります。意図する通りに動作しているかチェックすることを推奨します。



LSE の場合、論理合成制約ファイルの拡張子は [.ldc] となります。新しい LDC ファイルを作成すると、LDC エディタが起動します。

図 6-22. LDC エディタの初期ウィンドウ



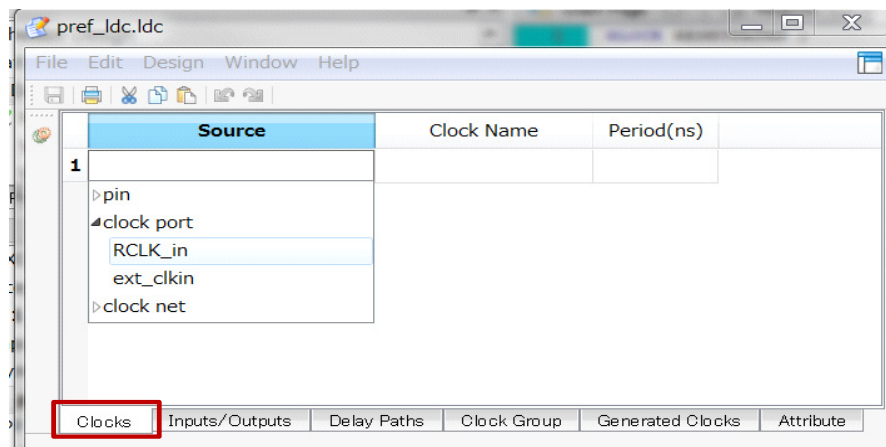
LDC エディタでは、以下の制約設定を行うことができます。

- ・ Clocks タブ： クロックの周期指定
- ・ Inputs/Outputs タブ： 入力ポートの最大入力遅延、出力ポートの最大遅延
- ・ Delay Paths タブ： 特定パスのフォルスパス / マルチサイクルパス / 最大遅延
- ・ Clock Group タブ： 任意のクロック相互関係が非同期、或いは非依存であることを明示
- ・ Generated Clocks タブ： ユーザロジックで生成されたクロックを指定
- ・ Attribute タブ： LSE のサポートする Synplify Lattice Attribute を指定

### 6.2.4.1 クロック周期指定（Clocks タブ）

クロック周期指定は Clocks タブで行います。Source 欄をクリックすると、デザイン内のクロックのリストが表示されます。この中から制約を与えるクロックを選択します。Clock Name 欄には、任意のクロック名 (LSE 内でのみ使用するエイリアス名) を、Period (ns) 欄ではクロック周期を入力します。

図 6-23. LDC Editor によるクロック周期指定



### 6.2.4.2 入力ポートの最大遅延指定（Inputs/Outputs タブ）

入力ポートのタイミング制約として最大遅延（クロックエッジから入力データ確定までの遅延値、図 6-24）を与えることができます。

図 6-24. 入力（最大）遅延

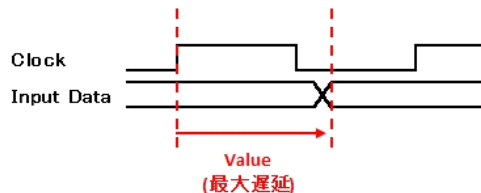
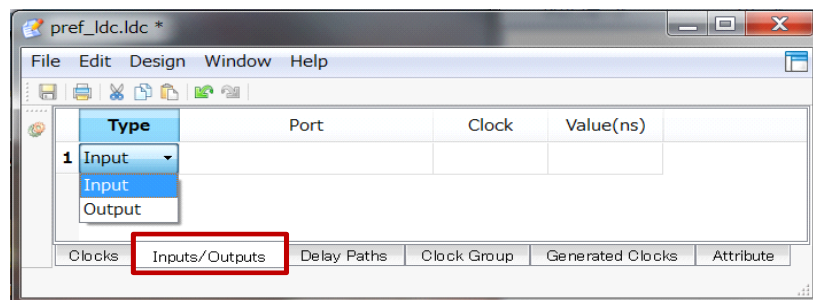


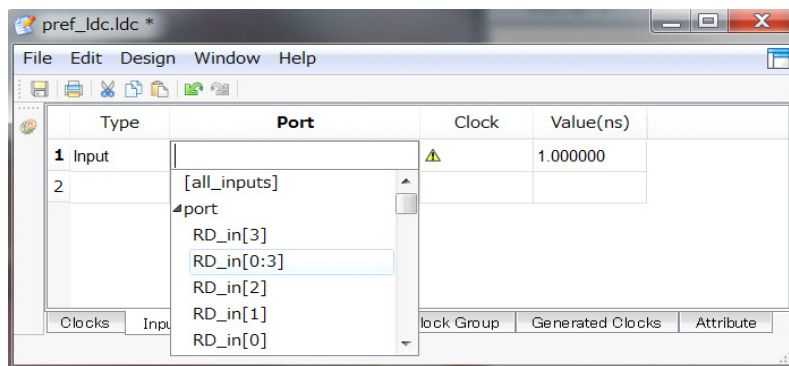
図 6-25. LDC Editor によるポートタイプ（入力）選択



Type 欄で Input を選択します（図 6-25）。

Port 欄をクリックすると入力ポートのリストが、Clock 欄をクリックするとクロックポートのリストが表示されますので（図 6-26）、それぞれ制約の対象を選択し、Value 欄に制約値を入力します（図 6-28）。

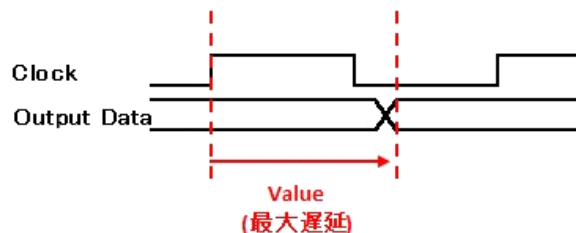
図 6-26. LDC Editor による入力ポート選択



#### 6.2.4.3 出力ポートの最大遅延指定（Inputs/Outputs タブ）

出力ポートへのタイミング制約として最大遅延（クロックエッジから出力データ確定までの遅延、図 6-27）を与えることができます。

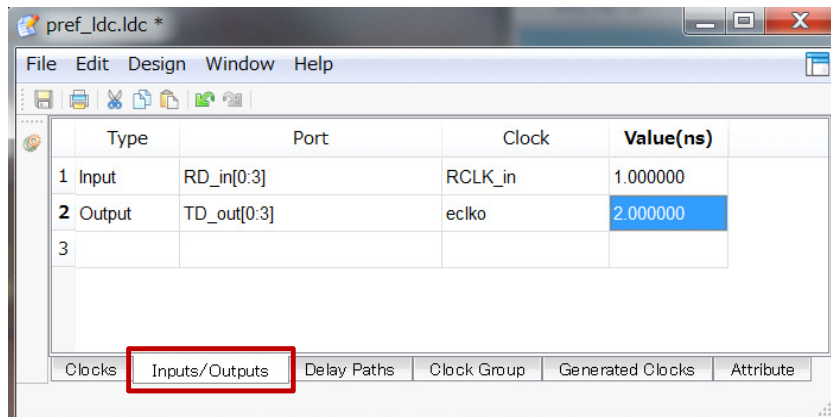
図 6-27. 出力（最大）遅延



Type 欄で Output を選択します (図 6-28)。

Port 欄をクリックすると出力ポートのリストが、Clock 欄をクリックするとクロックポートのリストが表示されるので、それぞれ制約の対象とするものを選択し、Value 欄に制約値を入力します。

図 6-28. LDC Editor による出力最大遅延指定

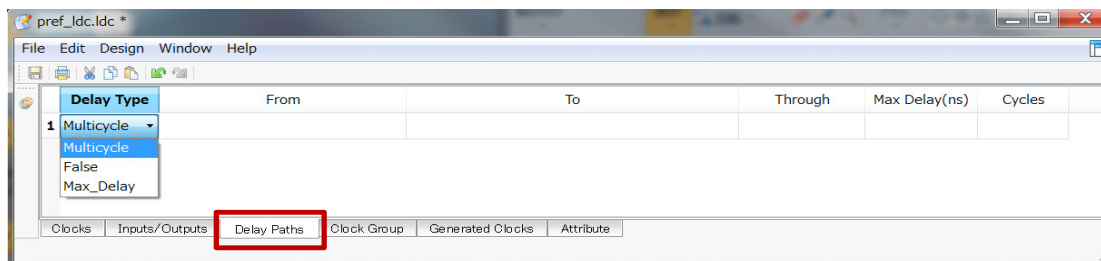


## 6.2.4.4 マルチサイクルパス指定 (Delay Paths タブ)

特定パスへのタイミング制約としてマルチサイクルパス指定 (複数クロックサイクルでのデータ転送) を与えることができます。

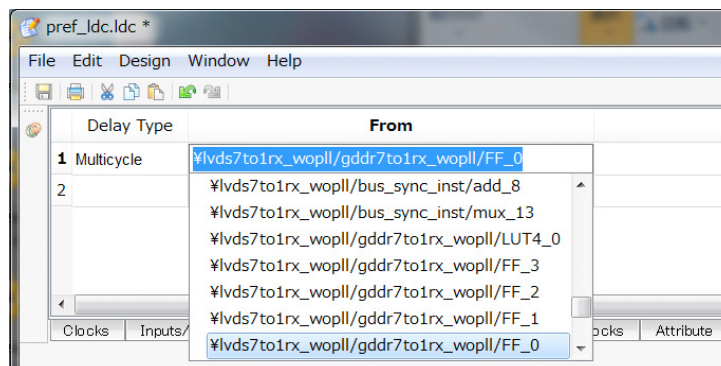
マルチサイクルパス指定には、まず Delay Type 欄で Multicycle を選択します (図 6-29)。

図 6-29. LDC Editor による Delay Paths 内のタイプ選択



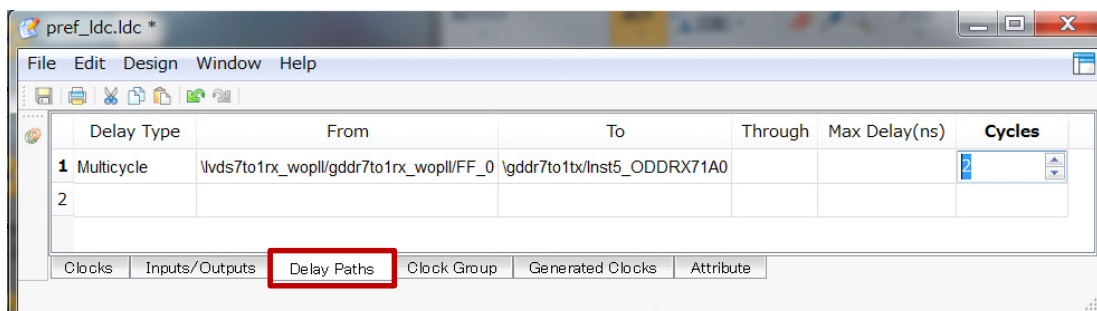
From 欄および To 欄をクリックすると制約の対象となるレジスタやネット名のリストが表示されるのでパスの始点 (From) と終点 (To) を選択します (図 6-30)。

図 6-30. LDC Editor によるマルチサイクル指定 (From)



From / To どちらかは省略することができます。また、指定したパスに複数の経路がある場合は、Through 欄でネット名を選択することで経路を限定することもできます。Cycle 欄には制約値（クロック数）を入力します（図 6-31）。Max delay 欄の入力は不要です。

図 6-31. LDC Editor によるマルチサイクル指定後



#### 6.2.4.5 フォルスパス指定（Delay Paths タブ）

特定パスをタイミング解析の対象から除外する、フォルスパス指定を与えることができます（図 6-32）。

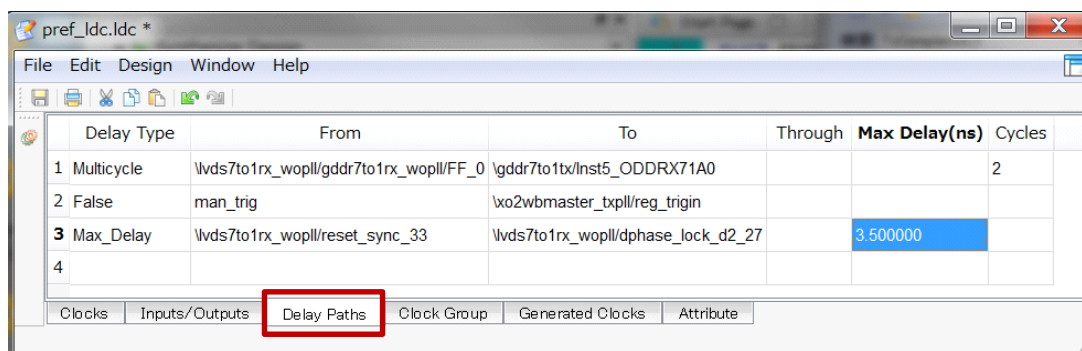
まず Delay Type 欄で False を選択します。From 欄および To 欄をクリックすると制約の対象となるレジスタやネット名のリストが表示されるのでパスの始点（From）と終点（To）を選択します。From / To どちらかは省略することができます。また、指定したパスに複数の経路がある場合は、Through 欄でネット名を選択することで経路を限定することもできます。Cycle 欄および Max delay 欄の入力は不要です。

#### 6.2.4.6 最大パス遅延（Delay Paths タブ）

特定パスに対する最大遅延指定は Delay Paths タブで行います。

Delay Type 欄で Max Delay を選択します。From 欄および To 欄をクリックすると制約の対象となるレジスタやネット名のリストが表示されるのでパスの始点（From）と終点（To）を選択します。From / To どちらかは省略することができます。また、指定したパスに複数の経路がある場合は、Through 欄でネット名を選択することで経路を限定することもできます。Maxdelay 欄には制約値（単位は ns）を入力します（図 6-32）。Cycle 欄の入力は不要です。

図 6-32. LDC Editor による最大パス遅延の指定

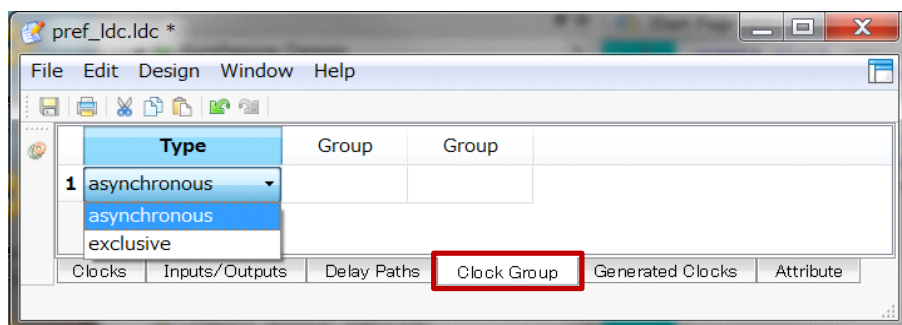


#### 6.2.4.7 クロック相互関係（Clock Group タブ）

特定の 2 クロックが非同期関係、或いは非依存の関係にある場合に明示的に指定することで、タイミング解析が不要なことをツールに指示します。

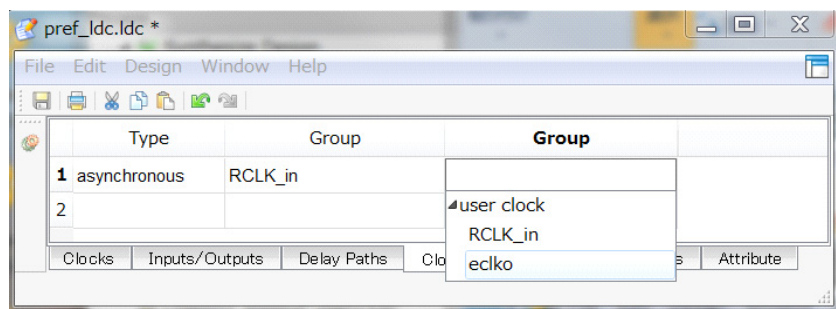
まず Type 欄で関係が非同期（asynchronous）か非依存（exclusive）かを選択します。

図 6-33. LDC エディタによる 2 クロック間の関係の選択



次に二つある Group 欄にそれぞれ意図するクロック（グループ）をクリックすると表示される候補からプルダウン形式で選択します。

図 6-34. LDC エディタによる非同期関係にあるクロックの指定



## 6.2.4.8 ユーザ論理で生成するクロックの指定（Generated Clocks タブ）

ユーザ論理回路で生成する分周クロックなどの指定を行います。

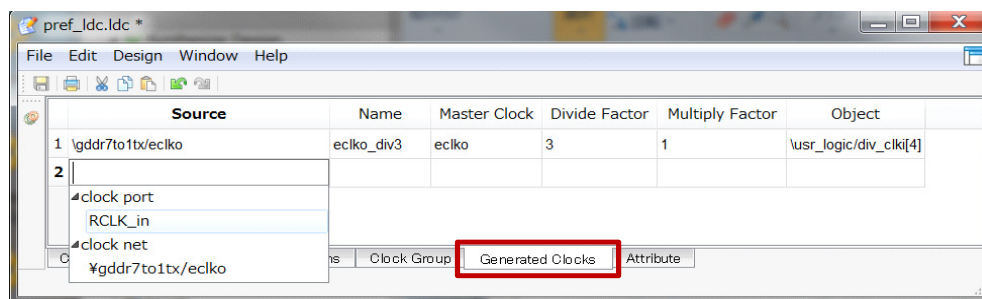
まず Source 欄をクリックすると表示されるクロックの候補から意図する信号を生成するための大元になるピン名を選択します。Name 欄はエイリアス名であり、入力は任意です。

Master Clock 欄は生成クロックの元になっている信号です。

次いで Divide Factor と Multiply Factor を入力します。必ずどちらか一方には入力しなければなりません。値は整数です。

Object 欄で意図するユーザ生成クロックを選択します。

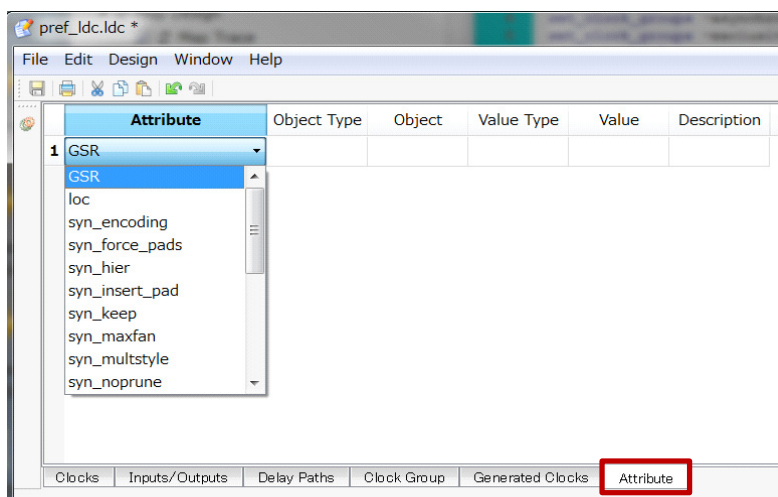
図 6-35. LDC エディタによるユーザ生成クロックの指定





### 6.2.4.9 Synplify Lattice Attribute の指定 (Attribute タブ)

図 6-36. LDC エディタで選択できるアトリビュート (からの選択)



LSE でサポートする Synplify Attribute を指定します。まず Attribute 欄でクリックすると表示される候補から意図するアトリビュートを選択します (図 6-36)。

次いで Object Type 欄でプルダウンから対象オブジェクトのタイプを指定します (図 6-37)。

次に Object 欄で選択したアトリビュートを適用するオブジェクトを、同様にクリックして表示される候補から選択します (Netlist Analyzer GUI からドラッグ&ドロップ形式で指定することも可能です)。候補に表示されないアトリビュートは未サポートです。

Value Type 欄と Value 欄は、選択されたアトリビュートに準じて選択可能な候補が表れますので、それぞれ選択します。Description 欄にはアトリビュートについての説明が表示されます。

図 6-37. アトリビュート適用オブジェクトのタイプ選択

	Attribute	Object Type	Object
1	GSR	module	<Global>
2		net	lvds7to1
3		<any> MultiplierInstance RAMInstance ROMInstance instance module net port register	

### 6.2.5 アトリビュートの RTL 記述例

LSE でサポートするアトリビュート一覧 6.6 節に示しますが、参考として、いくつかの例を示します (Synplify-Pro も同様)。

#### VHDL 例

(定義した信号・ノードを論理圧縮で削除しないで残す)

```
attribute syn_keep      : boolean; -- ワイヤ (ノード) の保持
attribute syn_keep of <signal_name>: signal is true; -- “signal_name” は signal 定義した信号名
```

```
attribute syn_preserve : boolean; -- FF の保持
attribute syn_preserve of <signal_name>: signal is true; -- “signal_name” は signal 定義した信号名

attribute syn_noprune : boolean; -- 出力がないリソース削除の禁止
attribute syn_noprune of <signal_name>: signal is true; -- “signal_name” は signal 定義した信号名
```

(FSM のコーディング方式)

```
attribute syn_encoding : boolean; -- [sequential, binary, oneshot], 他に safe (“default” 記述有効化)
attribute syn_encoding of <fsm_nodes>: signal is “safe, Boolean”; -- “fsm_nodes” は signal 定義した FSM 名
```

### Verilog HDL 例

(I/O タイプ指定、ポート宣言行)

```
input CLKIN /* synthesis IO_TYPE="LVCMOS18" */ ;
output CLKOUT /* synthesis IO_TYPE="LVDS25E" */ ;
```

(定義した信号・ノードを論理圧縮で削除しないで残す。ノード定義行)

```
reg [x:0] <FF_name> /* synthesis syn_preserve=1 */;
wire [y:0] <wire_name> /* synthesis syn_keep=1 */;
```

(FSM のコーディング方式。FSM ノード定義行)

```
reg [z:0] <FF_FSM_name> /* synthesis syn_encoding="safe, binary" */;
```

## 6.3 Synplify Pro

### 6.3.1 ストラテジ設定

特に Synplify Pro に関しては、**Diamond のバージョンによってストラテジ設定オプションの削除や新規追加、またはデフォルト指定の変更が行われています**ので、留意してください。旧バージョンからアップデートする場合は、必ず確認してください。

### 6.3.2 ストラテジ設定ウインドウの起動

起動方法は LSE と同じです (図 6-5 参照)。

### 6.3.3 Synplify Pro ストラテジ・オプション

以下にストラテジ・オプション各設定の詳細を説明します。

#### Allow Duplicate Modules

パラメータ : True / False      デフォルト値 : False

Verilog HDL で同じ名前のモジュールが複数定義されている場合の処理に関する設定です。

[False] (Default) を選択した場合、1 つのデザイン内に同じ名前のモジュールが複数定義されていると、エラーになり論理合成が行われません。[True] を選択した場合、1 つのデザイン内に同じ名前のモジュールが複数定義されていると、最後に定義されたモジュールだけが使用され、他のモジュールは無視されます。

#### Area

パラメータ : True / False      デフォルト値 : MachXO ファミリのみに True、それ以外は False

Area (リソース数) を小さくするためのオプション設定です。



[True] を選択すると Area を最小にするために周波数制約を 1MHz に設定して論理合成を行います。この場合、[Frequency] オプションの設定値は無視されます。sdc ファイルにクロックの周波数（周期）設定がされている場合は、sdc ファイル内の設定が優先されます。

### Arrange VHDL Files

パラメータ：True / False      デフォルト値：True

VHDL ファイルをコンパイルする際に、自動的に最上位モジュールが最後にコンパイルされるようにコンパイル順を変更するためのオプションです。

[True] の場合は、自動的にコンパイル順が調整されます。[False] の場合は、File List ウィンドウで表示されている順に上からコンパイルされます。

### Clock Conversion (Diamond 2.1 以降)

パラメータ：True / False      デフォルト値：True

ゲーテッド・クロックやユーザロジック生成のクロックためのオプションです。従来での Fix Gated Clocks や Fix Generated Clocks オプションと同等の機能です。

[True] の場合は、CE ベースの回路に変換します。[False] では変換しません。

### Command Line Options

パラメータ：文字列      デフォルト値：ブランク

このドキュメントで紹介している GUI のリストに表示されていない特別なオプションを使用する場合に、直接引数等を記述します。設定する場合は、以下のように記述します。

set\_option オプション

記述できるオプションは 1 つだけです。オプションの詳細は Synplify Pro の Reference Manual や User Guide をご覧ください。

### Default Enum Encoding

パラメータ：Default / Onehot / Gray / Sequential      デフォルト値：Default

VHDL の列挙（enum）型変数が定義されている場合、その変換形式に関する設定です。

[Default] を選択した場合、型のメンバ数に応じて自動的に割り振られます。

[Onehot] を選択した場合、各メンバに [onehot]（全 bit 中 1bit のみが '1'）の値が割り当てられます（例：0001, 0010, 0100, 1000）。

[Gray] を選択した場合、各メンバに一度の変化で 1bit のみが変わるような値が割り当てられます（例：000, 001, 011, 111, 110, 010）。

[Sequential] を選択した場合、一度に複数 bit が変化するような値（基本的には binary counter）が割り当てられます。

### Disable IO Insertion

パラメータ：True / False      デフォルト値：False

ネットリストへの I/O バッファ挿入に関する設定です。

[False]（デフォルト）を選択した場合、最上階層のソースのポートに I/O バッファが挿入されます。

[True] を選択した場合は I/O バッファが挿入されません。

I/O バッファの無いネットリストは配置配線ができません。IP コアの生成等の特殊な場合を除き [True] を選択しないで下さい。

### Export Diamond Settings to Synplify Pro GUI (Diamond 2.1 以降)

パラメータ：No / Only on First Launch / Yes      デフォルト値：No

Synplify Pro を Tools メニューから立ち上げて論理合成処理を行う場合に、ストラテジ設定各オプションをエクスポートするかどうかを指定するためのオプションです。

デフォルトの [No] は Diamond 2.0 までの動作と同じです。[Yes] の場合は、Synplify Pro が呼ばれるごとにストラテジ設定を取り込んで起動し、それ以前のセッションで保存された設定があっても無効になります。

[Only on First Launch] では、最初の起動時のみにストラテジ設定を取り込んで起動し、それ以後は直前のセッションでの設定やデフォルトを用いて起動します。

### FSM Encoding

パラメータ : True / False    デフォルト値 : True

ステートマシン用のコンパイラである FSM (Finite State Machine) コンパイラの使用に関する設定です。

[True] (デフォルト) を選択した場合は FSM コンパイラが使用されますが、[False] を選択した場合、FSM コンパイラは使用されないため、ステートマシンに対して最適な論理合成が行われない可能性があります。

### Fanout Guide (Diamond 2.1 以前の名称 : Fanout Limit)

パラメータ : 最大ファンアウト数    デフォルト値 : 1000 (旧値は 100)

ファンアウト数の上限の制約設定です。ファンアウトがこの値を超えた場合は、ドライバをコピーしてファンアウトを減らす処理が行われます。ただし、この制約はあくまでガイドラインであり、制約を満たすことにより最適化が妨げられる場合は無視されます。

なお、Synplify Pro では特定のネットワークに対するファンアウトを制限するために RTL に以下の記述をしても有効です。"value" が意図する上限値です。

Verilog    `[wire | reg] <object 名宣言> /* synthesis syn_maxfan = "value" */ ;`

VHDL    `attribute syn_maxfan of object : objectType is "value" ;`

### Fix Gated Clocks (Diamond 2.1 以降はありません)

パラメータ : 0 / 1 / 2 / 3    デフォルト値 : 3

[Gated Clock Conversion] 処理とそのレポート出力に関する設定です。

[Gated Clock Conversion] は、ソース内では Clock の Gating に使用している信号をレジスタの CE (クロックイネーブル) ポートに接続し、回路の同期化を行う処理です (図 6-38)。

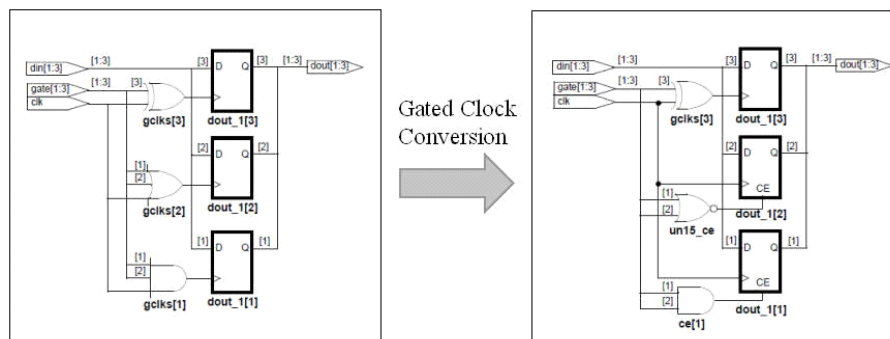
[0] を選択した場合は、[Gated Clock Conversion] 処理が行われません。

[1] を選択した場合は、[Gated Clock Conversion] 処理が行われますが、レポートされません。

[2] を選択した場合は、[Gated Clock Conversion] 処理が行われ、変換できなかったクロックネットに関してのみレポートされます。

[3] を選択した場合は、[Gated Clock Conversion] 処理が行われ、全ての Gated Clock に関する処理結果がレポートされます。

図 6-38. Fixed Gated Clock オプションでのクロック最適化の一例



なお、[0] 以外を選択した場合でも、以下の条件を満たさないリソースに対してはこの [Gated Clock Conversion] が行われません。

1. sdc ファイル内で、define\_clock 指示子により Gating 前のクロックが定義されている場合
2. ゲーテッド・クロックの接続先が、Sequential primitive（基本的にはレジスタのみ）の場合
3. Gating する論理が AND や OR 等の場合（XOR 等は変換されない）

ブロック RAM や DSP ブロックは、論理合成ではブラックボックスとして扱われる (Sequential Primitive ではない) ため、これらに接続されるゲーテッド・クロックに対しては、Conversion 処理は行われません。また、Gating の論理によっては Conversion が行われない場合もあります。

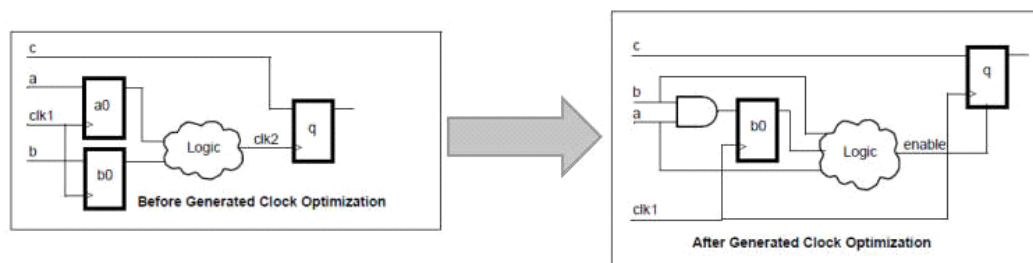
### Fix Generated Clocks (Diamond 2.1 以降はありません)

パラメータ : 0 / 1 / 2 / 3      デフォルト値 : 3

デバイス内のレジスタ出力等から生成されたローカルクロックの最適化処理とそのレポート出力に関する設定です。

この処理では、デバイス内で生成された分周クロック等のローカルクロックを、元のクロックとクロックイネーブルに変換し、回路の同期化を行います (図 6-39)。

図 6-39. Fixed Generated Clocks オプションでのクロック最適化の一例



[0] を選択した場合は、この処理は行われません。

[1] を選択した場合は、この処理は行われますが結果等についてはレポートされません。

[2] を選択した場合は、この処理が行われ変換が行われなかったローカルクロックに関してのみレポートされます。

[3] を選択した場合は、この処理が行われ全てのローカルクロックに関する処理結果がレポートされます。

なお、[0] 以外を選択した場合でも、回路構成によってはこの処理が行われない場合もあります。

一例

分周クロックが階層をまたいで他のモジュールで使用されている。

分周クロックが最上位階層の出力ポートに接続されている。

### Force GSR

パラメータ : Auto / False / True      デフォルト値 : False

論理合成時の GSR (Global Set/Reset) 挿入に関する設定です。

[Auto] / [True] を選択した場合は自動的に GSR バッファが挿入されます (ただし、Set/Reset の構成が条件を満たした場合のみ)。

[False] を選択した場合は GSR が使用されません。

GSR はマッピング (Map Design) プロセスで、より最適な条件で挿入することができます。特別な理由のない限りこのオプションは [False] を選択することを推奨します。

### Frequency (MHz)

パラメータ：周波数 [MHz]                      デフォルト値：なし (Diamond 2.2 以前は 200)

クロックの周波数の制約設定です。

Area 設定が [False] の場合のみ、このオプションで設定した値を周波数制約とする論理合成スクリプトを生成します。この周波数制約は全てのパス（レジスタ-レジスタ間）に対して適用されます。

### Number of Critical Paths

パラメータ：パス数                              デフォルト値：なし (Diamond 2.2 以前は 3)

論理合成のレポート内に出力するクリティカルパス数の設定です。

クリティカルパスは [Number of Start/End point] オプション (デフォルト値 0) が 1 以上の場合のみレポートされますので、クリティカルパスをレポートさせる場合は "Number of Start/End point" を 1 以上に設定してください。

### Number of Start/End points

パラメータ：数値                                  デフォルト値：なし (Diamond 2.2 以前は 0)

論理合成後のタイミング見積もり結果としてレポートするパス数の設定です。

このオプション設定が 1 以上の場合のみ [Critical Path] の詳細がレポートされます。

### Output Netlist Format

パラメータ：None / VHDL / Verilog              デフォルト値：None

論理合成後の HDL ネットリスト出力の設定です。

[None] (デフォルト) を選択した場合、論理合成後のネットリストは EDIF のみが出力され HDL のネットリストは出力されません。

[VHDL] を選択した場合は、EFID と VHDL のネットリスト (\*.vhm) が出力されます。[Verilog] を選択した場合は、EFID と Verilog HDL のネットリスト (\*.vm) が出力されます。

### Output Preference File

パラメータ True / False                          デフォルト値：True (Diamond 2.1 までは False)

Lattice Diamond の制約ファイル出力に関する設定です。

デフォルトの [True] では、論理合成時の制約から必要な制約だけを記述した Lattice Diamond の制約ファイルを出力します。出力ファイル名は "プロジェクト名\_インプリメンテーション名\_synplify.lpf" です。必要に応じて Lattice Diamond にインポートするか、必要な部分のみを使用する制約ファイルにコピーするかして使用してください。

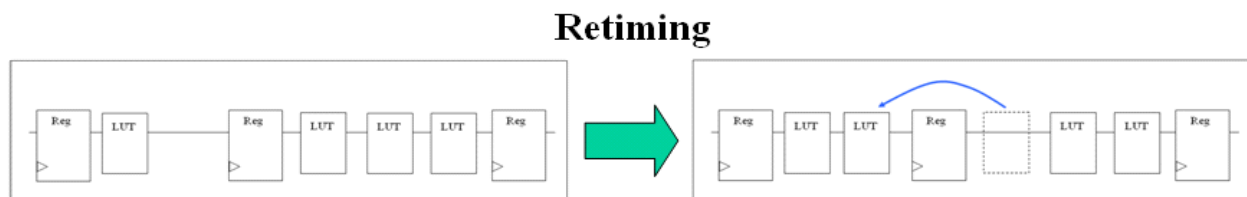
[False] を選択した場合、このファイルは出力されません。

### Pipelining and Retiming

パラメータ：None / Pipelining Only / Pipelining and Retiming      デフォルト値：Pipelining Only (Diamond 2.1 までは None)

論理合成中に行うタイミングの最適化処理 [Retiming] と [Pipelining] の実行に関する設定です。

図 6-40. Retiming 処理の一例



[Retiming] は論理回路を移動させてレジスタ間の負荷を分散させる処理です (図 6-40)。回路全体に対してこの処理が行われます。これを演算回路等に対してのみ行うのが [Pipelining] です。

[None] を選択した場合、Pipelining 処理は行われません。[Pipelining and Retiming] を選択した場合、Pipelining と Retiming 処理が行われます。

[None] を選択した場合、どちらの処理も行われません。

### Push Tristate

パラメータ : True / False

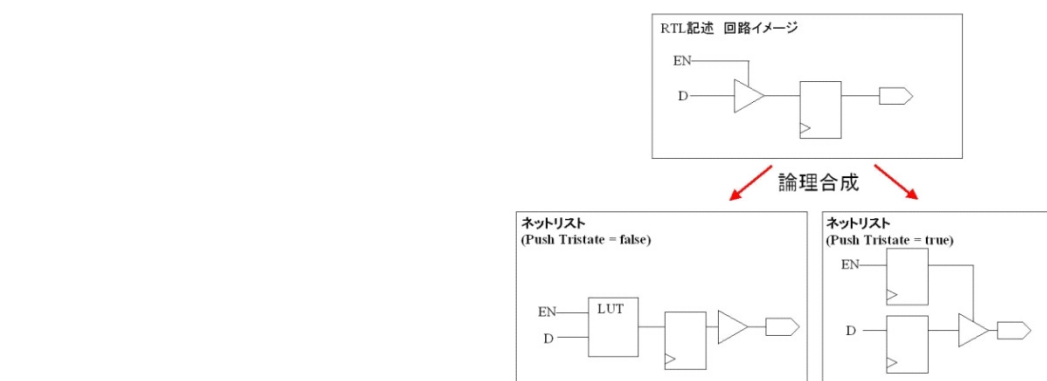
デフォルト値 : True

内部トリステートの論理合成方法に関する設定です。

[True] (デフォルト) を選択した場合、'Z' がレジスタを介してポートに出力されるような記述に対してトリステート制御用レジスタを追加し、IO バッファでトリステート制御を行うように回路を論理合成します (図 6-41)。

[False] を選択した場合は、FPGA 内部で 'Z' が生成されないような回路が論理合成されます。

図 6-41. Push Tristate オプションの効果



### Resolved Mixed Drivers

パラメータ : True / False

デフォルト値 : False (Diamond 2.0 までは True)

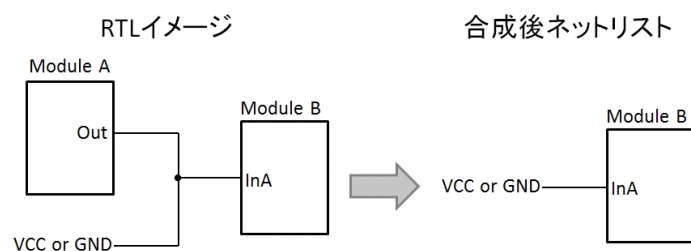
信号の衝突 (Multiple Drive) が発生しているネットの処理に関する設定です。

[True] を選択した場合、RTL 上で図 6-42 のように 1 つのネットが、複数のドライバに接続されており、なおかつその中の 1 つが [1] または [0] 固定の場合、論理合成の際にそのネット固定値として処理されます。複数ドライバによる信号衝突がある旨のメッセージはログに出力されますが、エラーにはなりません。

[False] を選択した場合、このような条件でもエラーで論理合成が中断されます。

なお、どちらを選択した場合でも衝突している全てのドライバが固定値でない場合は、エラーで論理合成が中断されます。

図 6-42. Resolve Mixed Drivers 処理の一例



### Resource Sharing

パラメータ : True / False

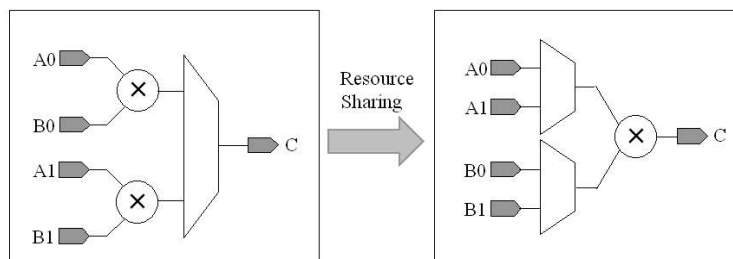
デフォルト値 : True

リソースの最適化処理に関する設定です。

[True] を選択した場合、使用するリソースを減らすために、複数ある同一機能の回路を共有させる処理が行われます (図 6-43)。

[False] を選択した場合この処理は行われません。

図 6-43. Resource Sharing 処理の一例



### Update Compile point Timing Data

パラメータ : True / False

デフォルト値 : False

コンパイルポイント (compile point) オプションを使用した際の振る舞いに関する設定です。階層設計で compile point 指定されている下位モジュールが変更・再度論理合成された場合に、その上位街道のモジュールを再合成するか否かを選択します。[False] (デフォルト) では下位モジュールが再び論理合成されても上位モジュールは再び論理合成されません。

[True] を選択した場合、下位モジュールが再合成されると上位モジュールが再び論理合成されることがあります。ただし、上位モジュールは必ず再合成されるのではなく、下位モジュールの変更によりインターフェイス部分の回路構成とタイミング (遅延) が変化した場合に必要なに応じて行われます。

### Use Clock Period for Unconstrained I/O

パラメータ : True / False

デフォルト値 : False (Diamond 2.1 までは True)

sdc ファイルで制約が与えられていない I/O の制約に関する設定です。

[True] (デフォルト) を選択した場合、制約が与えられていない I/O には "Frequency" オプションで設定されたクロック周波数の 1 周期がセットアップ時間または Clock-to-Output の制約として与えられます。

デフォルトでは制約は与えられません。

### VHDL 2008 (Diamond 2.2 以降)

パラメータ : True / False

デフォルト値 : False

VHDL バージョン指定のためのオプションです。[True] の場合は、プロジェクトの VHDL 言語仕様として IEEE 1076-2008 を選択します。

### Verilog Input

パラメータ : Verilog95 / Verilog2001 / System Verilog

デフォルト値 : Verilog2001

コンパイルする Verilog HDL の記述仕様の選択です。デフォルトの [Verilog2001] では IEE1364-2001 の言語仕様に基づいてコンパイルされます。

[Verilog95] を選択した場合、IEEE1364-1995 の言語仕様に基づいてコンパイルされます。

[System Verilog] は Diamond 2.1 から追加されています。

### 6.3.4 Synplify Pro 制約ファイル生成

バージョン G-2012.09 以降 (Lattice Diamond 3.0 以降) は、"sdc" ファイルによる制約設定が "fdc" (と "adc") を用いる構成に更新されました。従来のレガシー sdc (Synplify Design Constraint) を用いることも可能ですが、新規デザインには推奨されません。本節では Synplify Pro が備える制約エディタ GUI である「SCOPE エディタ」を用いた "Synopsis Design Constraint" としての fdc (FPGA Design Constraints) ファイルの生成方法について、概要のみを記述します。詳細は Synplify Pro のオンライン・マニュアル (user\_guide.pdf, Chapter 5) 等を参照してください。(なお、Synplify Pro には sdc から fdc への変換コマンド "sdc2fdc" がサポートされていますので、こちらにも必要に応じてオンライン・マニュアルをご参照ください。)

#### 6.3.4.1 SCOPE エディタの起動と制約の追加方法

ここでは論理合成時に適用する制約ファイルを新規に生成する方法を記述します。RTL の文法エラー等はすべて解消されているものとします。

まず Lattice Diamond から Synplify Pro を単独で立ち上げます。[File] メニューで [New...] を選択して小ウィンドウを立ち上げます (図 6-44)。「fdc」ファイルの生成のため [FPGA Design Constraints] を選択し、丸印内の [New File Name] を入力して [File Location] をブラウザして指定した後、OK をクリックします。これにより、図 6-45 に示すような制約入力用のワークシート「SCOPE エディタ」が立ち上がります。

図 6-44. SCOPE エディタの起動

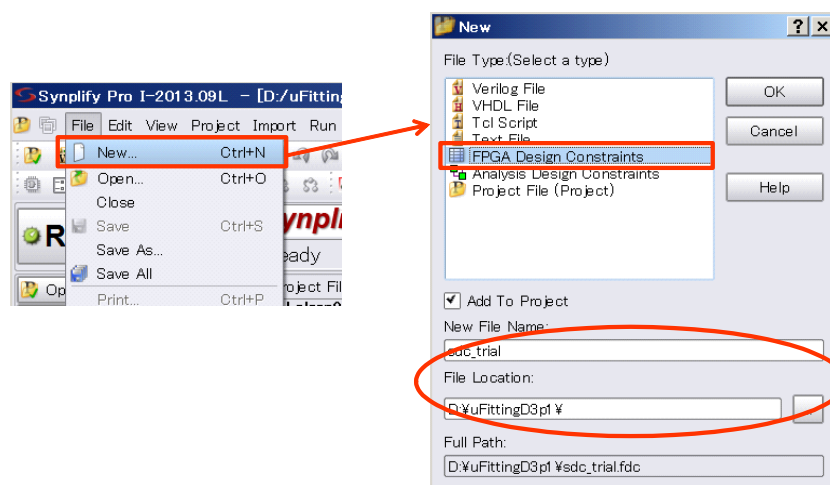
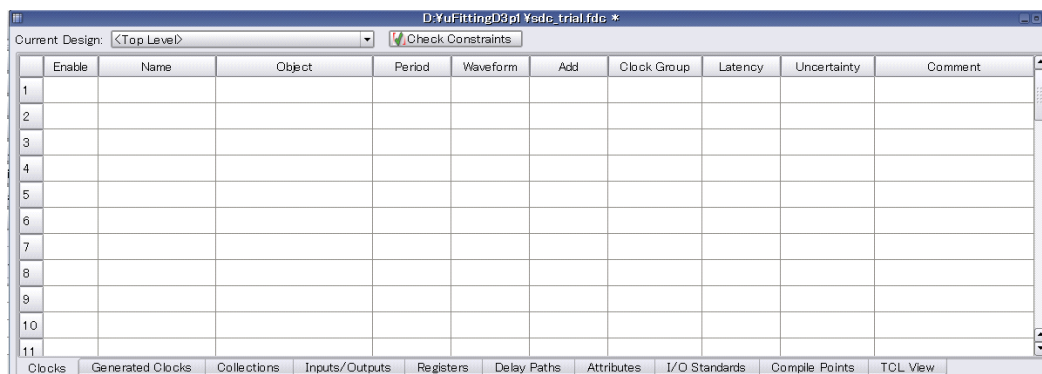


図 6-45. “FPGA Design Constraints”入力用ワークシート GUI



ウィンドウの下部には複数のタブがあります。通常はクロック制約を与える [Clocks] タブや I/O ポートに対する制約の [Inputs/Outputs] タブを選択した後に、それぞれを入力していきます。



習熟していれば本スプレッドシート形式のシートに直接入力して構いませんが、層でない場合の簡単な入力方法があります。クロックネットやポートの「オブジェクト」を GUI 上でドラッグ&ドロップする方法です。手順は以下の通りです。

1. Synplify Pro のトップで左上の大きなボタン [Run] をクリックするか、[Run] メニューから [Compile Only] を選択してコンパイルします (図 6-46、左)。RTL エラーで処理が完了しないと次のステップができませんので、注意します
2. 上部メニューバー内の RTL ビューアイコン (図 6-46 の右図に赤枠で示す) をクリックして、RTL スケマティックを表示させます
3. 図 6-47 のような、右上にある "Restore Down" 印 (どの表示でも構わない) をクリックして、複数のウィンドウ間でマウス移動ができるようにします

図 6-46. コンパイル実行 (左) と RTL ビューの起動 (右)

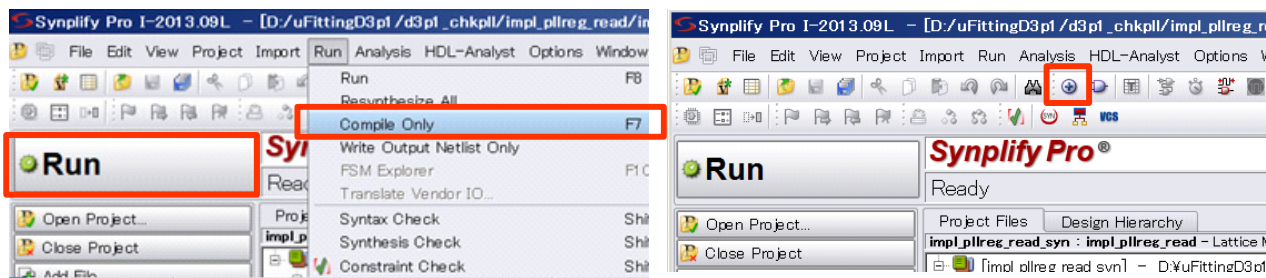


図 6-47. 各ウィンドウの個別 (多重) 表示 "Restore Down"

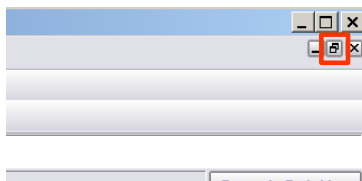
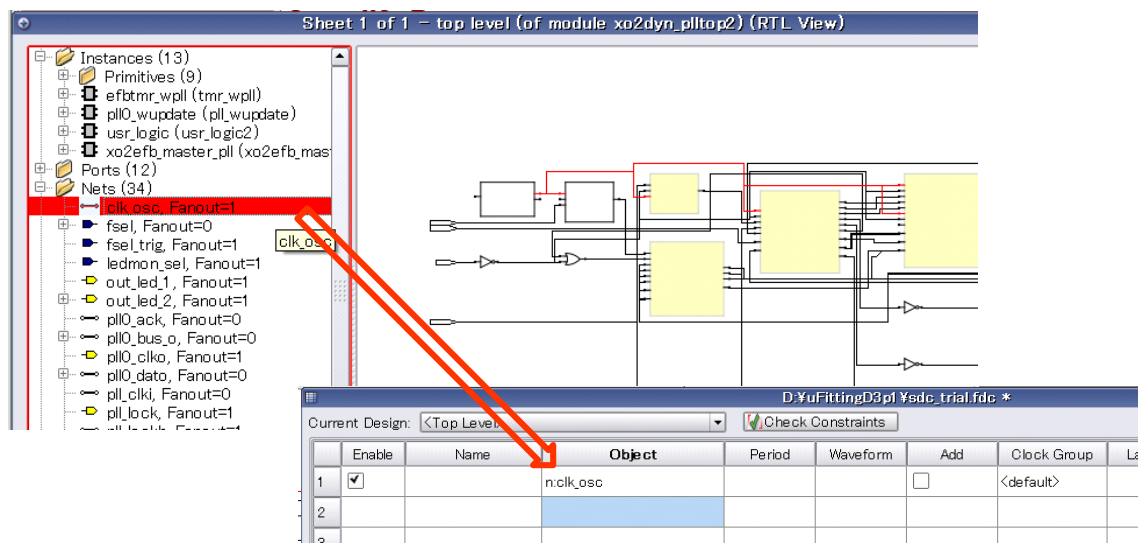


図 6-48. オブジェクトのドラッグ&ドロップ



4. FPGA Design Constraints で Clocks タブを選択します (適当な表示位置に移動しておきます)
5. RTL ビューの左枠で Instances や Nets 項をエクスパンドして意図するクロックネットを選択します (図 6-48、左の赤印が選択された状態)

6. クリックした状態でマウスをドラッグして”Object”セルにドロップします（図 6-48、右）
7. Name や Period（単位は [nsec]）などを入力します
8. RTL ビューの左枠で各階層で必要な分だけ繰り返します

以上がクロック制約の設定概要です。

I/O タイミングについても同様です。異なるのはステップ 4 で選択するタブが ”Inputs/Outputs” であること、ステップ 5 で選択するのがポートであること、そしてステップ 7 で入力するパラメータが異なることです。

なお、この fdc ファイルは、Lattice Diamond のインプリメンテーション内の Synthesis Constraint File セクションで Synplify Pro に渡す論理合成制約ファイルとして取り込み、アクティベートできます。詳細は第 2 章をご確認ください。

また、Synplify Pro では adc（Analysis Design Constraint）ファイルも定義されていますが、Lattice Diamond を用いるフローとしてはその使用を前提としておりません。

### 6.3.4.2 制約ファイルの Synplify Pro 作業プロジェクトへの取り込み

制約ファイル生成時にプロジェクトにインクルードするかどうかを問われます。生成が完了した後、プロジェクト表示ウィンドウ（Project Files タブ）で図 6-49 のように [Logic Constraints (FDC)] 下や [Analysis Design Constraint] にそれぞれファイル名が表示されていれば、意図するとおりの動作です。

ファイル名がリストされていない場合は、図 6-50 左のように GUI トップ左枠内 [Add File...] をクリックすると表れるウィンドウで指定します。この際、[File of types] 項で拡張子として ”\*.fdc” を指定したあとブラウズします（図 6-50、右）。

図 6-49. 制約ファイル fdc のリスト表示

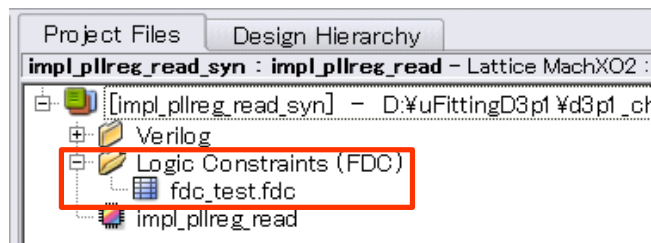
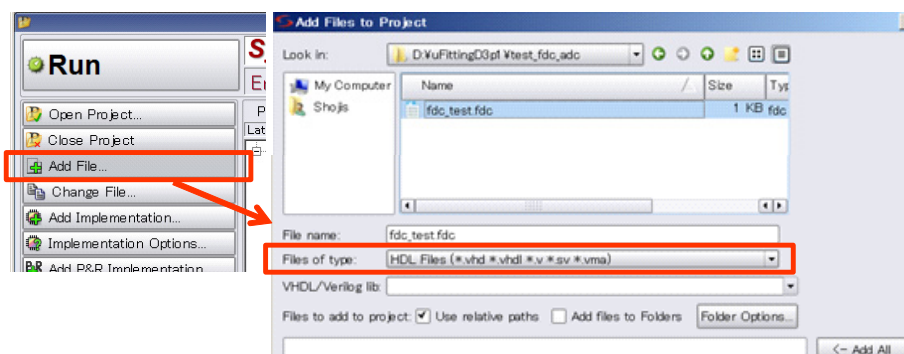
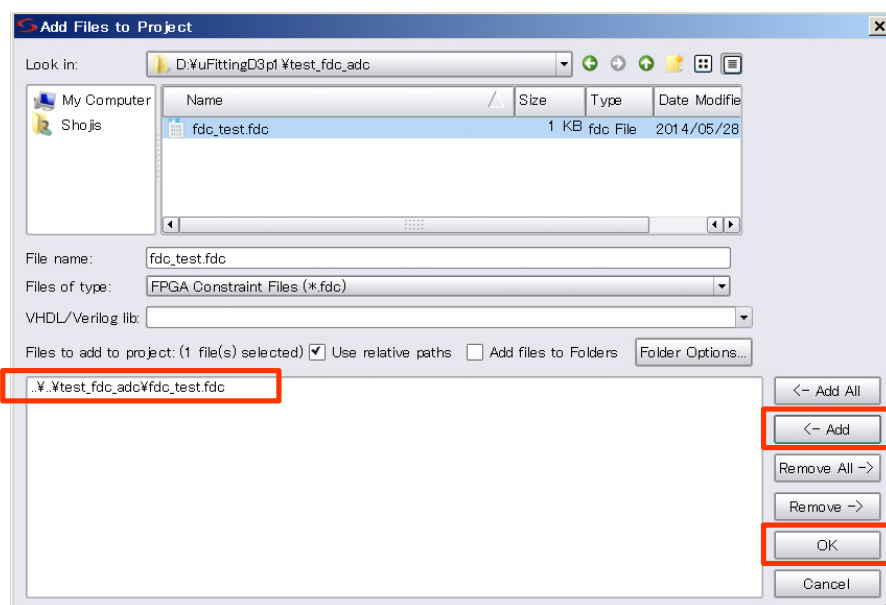


図 6-50. fdc ファイルの取り込み (1) ブラウズ



対象ファイルを選択後右下の [Add] ボタンをクリックし、図 6-51 のように意図する操作が終了後に [OK] をクリックして抜けます。以上によって図 6-49 と同様にプロジェクトに取り込まれたリスト表示にすることができます。

図 6-51. fdc ファイルの取り込み (2) 指定後 Add して OK



## 6.3.5 その他の制約記述

### 6.3.5.1 コンパイルポイントの設定

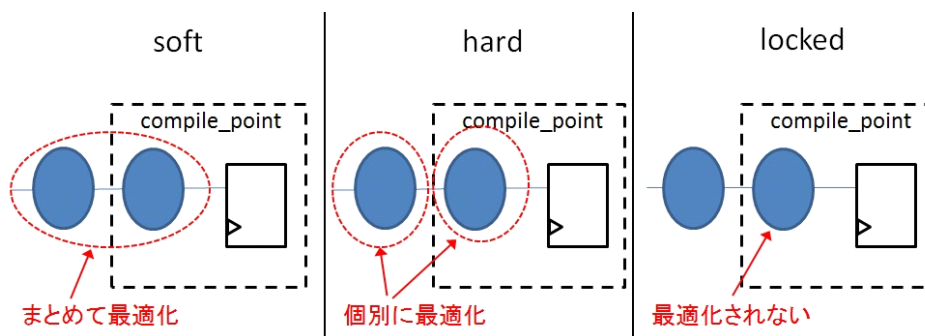
モジュールにコンパイルポイント (Compile point) 設定して論理合成すると、コンパイルポイント毎にデータベースが生成されます。論理合成を行う際にはこのデータベースを参照し、変更されているソースファイルのみの論理合成を行います。このため、論理合成時間の短縮になります。

コンパイルポイントを設定する場合は、sdc ファイル内に、以下のような制約を記述します。

**define\_compile\_point [v: ライブラリ名 . モジュール名] -type { コンパイルポイントタイプ }**

ライブラリ名はデフォルトでは work です。コンパイルポイントタイプは、モジュール間の接続部分の論理合成に関する設定で、[soft][hard][locked] のなかから 1 つを選択します。

図 6-52. コンパイルポイントタイプによる最適化の違い



[soft] を選択した場合は、モジュール間の接続分の回路をまとめて最適化します。[hard] を選択した場合、接続部分の回路をコンパイルポイント毎に個別に最適化を行います。[locked] を選択した場合は、接続部分の最適化は行われません。

### 6.3.5.2 ネットリストの階層展開

デフォルト設定の Synplify Pro では、論理合成後のネットリストをオリジナルの階層を保持したまま出力します。階層を展開したネットリストを出力させる場合は、sdc ファイルに以下のような制約を記述します。

```
define_global_attribute syn_netlist_hierarchy 0
```

この制約は、HDL ソース内に記述することもできます。記述方法については Synplify Pro のヘルプドキュメントを [syn\_netlist\_hierarchy] で検索してください。なお、この設定はネットリストの出力に関する設定で、論理合成は階層ごとに行われます。

### 6.3.6 Synplify Pro ヘルプおよびドキュメント

Synplify Pro の使用方法やオプション設定に関するヘルプドキュメントは、以下のフォルダにインストールされています。

**[Lattice Diamond のインストールフォルダ ]%synbase%\doc**

Diamond 3.3 をデフォルト設定でインストールした場合：

**C:\%lsc%\diamond\3.3%\synbase%\doc**

このフォルダには以下の 3 つのドキュメントがあります。

user\_guide.pdf : 主に全体的なデザインフローについて説明したドキュメント  
 reference.pdf : 主に制約設定や特定機能について詳細に説明したドキュメント  
 release\_notes.pdf : バージョンアップに伴う変更点等について説明したドキュメント

また、Synplify Pro の GUI を立ち上げて [Help] メニューから [online help] 等を使用することもできます。

## 6.4 Synplify Pro から LSE へのコンバージョン

Synplify Pro をプロジェクト / インプリメンテーションの論理合成ツールとして選択していたデザインを LSE に変更する際の留意点などを以下に整理します。

#### ① タイミング制約の確認・見直し（重要）

論理合成ツールが変わると、特にタイミング制約の対象であるクロックネット名が同一にならない場合が少なくありません。同一名でないのは、例えば "clock\_c" のようなサフィックスが異なるとか、明らかに異なる違いの場合が殆どですが、全く異なる場合もあり得ます。これらを全て更新する必要があります。

まず確認することは LPF に既述されている制約が有効かどうか、です。スプレッドシート・ビューで『Timing Preferences』タブを選択すると、有効・無効が比較的容易に判別できます。当該クロックネットを Primary (Secondary)、Edge などに指定している場合（『Clock Resource』タブ）、また I/O タイミング制約の対象クロックとして指定している場合も該当しますので、更新します。

また、MAP Report の "Design Errors/Warnings" 項をチェックすることを推奨します。

#### ② "UGROUP" 指定

オリジナルで指定している UGROUP を一度コメントアウトするなり、無効化して一度 PAR 結果を確認することを推奨します。指定無しでタイミング等を満たせば、そのほうが望ましいと言えます。不満足な結果の場合は、再度指定を元に戻し、実行するようにします。

## 6.5 （付録）ストラテジ・オプションの比較

本節は両論理合成ツールのオプション比較を示します（Ver.3.4 時点）。

LSE		Synplify Pro	
Setting Name	Values	Setting Name	Values
Carry Chain Length	Numerical value	NA	-
Command Line Options	Text	Command Line Options	Text
Disable Distributed RAM	TRUE FALSE	NA	-
EBR Utilization	0 – 100	NA	-
FSM Encoding Style	Auto Binary Gray One-Hot -	Default Enum Encoding	Default - Gray Oneshot Sequential
Fix Gated Clocks	TRUE FALSE	Clock Conversion	TRUE FALSE
Force GSR	Auto No Yes	Force GSR	Auto False True
Hardware Evaluation	Enable Disable	Hardware Evaluation (accessed in “Translate Design” settings)	Enable Disable
Intermediate File Dump	True False	NA	-
Loop Limit	Numerical Value	NA	-
MUX Style	Auto L6Mux Multiple L6Mux Single PFU Mux	NA	-
Macro Search Path	Directory Location (text)	Macro Search Path (accessed in “Translate Design” settings)	Directory Location (text)
Max Fanout Limit	Numerical Value	Fanout Guide	Numerical Value
Memory Initial Value File	File Location & Name (text)	Memory Initial Value Search Path (accessed in “Translate Design” settings)	File Location & Name (text)
Number of Critical Paths	Numerical Value	Number of Critical Paths	Numerical Value
Optimization Goal	Area Balanced Timing	Area	TRUE FALSE (see Step 2 of this document)
Propagate Constants	TRUE FALSE	NA	-
RAM Style	Auto Block_RAM	NA	-

LSE		Synplify Pro	
Setting Name	Values	Setting Name	Values
ROM Style	Distributed	NA	-
	Registers		
	Auto		
	EBR		
Remove Duplicate Registers	Logic	NA	-
	TRUE		
Remove LOC Properties	FALSE	Remove LOC Properties (accessed in "Translate Design" settings)	-
	Off		
	On		Off
Resolve Mixed Drivers	TRUE	Resolve Mixed Drivers	TRUE
	FALSE		
Resource Sharing	TRUE	Resource Sharing	TRUE
	FALSE		
Target Frequency (MHz)	Numerical Value	Frequency (MHz)	Numerical Value
Use Carry Chain	TRUE	NA	-
	FALSE		
Use IO Insertion	TRUE	Disable IO Insertion	FALSE
	FALSE		
Use IO Registers	TRUE	NA	-
	FALSE		
Use LPF Created from Synthesis	TRUE	Use LPF Created from SDC in Project	TRUE
	FALSE		
VHDL 2008	TRUE	VHDL 2008	TRUE
	FALSE		
N/A	-	Allow Duplicate Modules	FALSE
N/A	-	Arrange VHDL Files	TRUE
N/A	-	Export Diamond Settings to Synplify Pro GUI	FALSE
N/A	-	FSM Encoding	No
N/A	-	Number of Start/End Points	Only on First Launch
N/A	-	Output Netlist Format	Yes
N/A	-	Pipelining and Retiming	TRUE
N/A	-	Push Tristates	FALSE
N/A	-	Update Compile Point	TRUE

LSE		Synplify Pro	
Setting Name	Values	Setting Name	Values
N/A	-	Use Clock Period for Unconstrained I/O	TRUE
N/A	-	Verilog Input	FALSE
			System Verilog
			Verilog 2001
			Verilog 95



## 6.6 （付録）LSE サポートの HDL アトリビュート

本節は LSE の対応するアトリビュート一覧を示します。

Name	Description
<a href="#">black_box_pad_pin</a>	Specifies that a pin on a black box is an I/O pad. It is applied to a component, architecture, or module, with a value that specifies the set of pins on the module or entity.
<a href="#">black_box_tri_pins</a>	Specifies that a pin on a black box is a tristate pin. It is applied to a component, architecture, or module, with a value that specifies the set of pins on the module or entity. (could only be for internal fabric based tri-state) – Similar to syn_tristate
<a href="#">full_case</a>	Specifies that a Verilog case statement has covered all possible cases.
<a href="#">loc</a>	Specifies pin, register, and bus locations for Lattice I/Os.
<a href="#">parallel_case</a>	Specifies a parallel multiplexed structure in a Verilog case statement, rather than a priority-encoded structure.
<a href="#">syn_noprune</a>	Controls the automatic removal of instances that have outputs that are not driven.
<a href="#">syn_pipeline</a>	Specifies that registers be moved into multipliers and ROMs in order to improve frequency.
<a href="#">syn_preserve</a>	Prevents sequential optimizations across a flip-flop boundary during optimization, and preserves the signal.
<a href="#">syn_ramstyle</a>	Determines the way in which RAMs are implemented.
<a href="#">syn_replicate</a>	Disables replication.
<a href="#">Syn_romstyle (Lattice)</a>	Determines the way in which ROMs are implemented.
<a href="#">Pragma translate_off/pragma translate_on</a>	Specifies sections of code to exclude from synthesis, such as simulation-specific code.
<a href="#">syn_black_box</a>	Defines a black box for synthesis.
<a href="#">syn_encoding</a>	Specifies the encoding style for state machines.
<a href="#">syn_enum_encoding</a>	Specifies the encoding style for enumerated types (VHDL only).
<a href="#">syn_force_pads</a>	Enables/disables IO insertion on a port level or global level.
<a href="#">syn_hier</a>	Controls the handling of hierarchy boundaries of a module or component during optimization and mapping.
<a href="#">syn_insert_buffer (Lattice)</a>	Inserts a clock buffer according to the vendor-specified value.
<a href="#">syn_insert_pad</a>	Inserts I/O buffers to either ports or nets.
<a href="#">syn_keep</a>	Prevents the internal signal from being removed during synthesis and optimization.
<a href="#">Syn_romstyle Attribute</a>	Determines how ROM architectures are implemented.
<a href="#">syn_sharing</a>	Specifies resource sharing of operators.
<a href="#">syn_srlstyle</a>	Determines how sequential shift components are implemented.
<a href="#">syn_state_machine</a>	Determines if the FSM Compiler extracts a structure as a state machine.
<a href="#">syn_use_carry_chain (Lattice)</a>	Enables/disables carry chain implementation for counter or adder.



Name	Description
<a href="#">syn_useenables</a>	For CPLD technologies, generates register instances with clock enable pins for selective clock-enable inference.
<a href="#">syn_useioff (Lattice)</a>	Packs flip-flops in the I/O ring to improve input/output path timing.
<a href="#">Translate_off/translate_on</a>	Specifies sections of code to exclude from synthesis, such as simulation-specific code.
<a href="#">syn_looplevelimit</a>	Specifies a loop iteration limit for while loops. (similar to loop limit)
<a href="#">syn_maxfan</a>	Sets a fanout limit for an individual input port or register output.
<a href="#">syn_multstyle</a>	Determines how multipliers are implemented.

## 6.7 変更履歴

Ver.	Date	page	内 容
2.3F	June 30/2014	6-12	6.2.2.2 項 Hardware Evaluation、IP ライセンス有りの場合の設定記述を更新
3.1	June 30/2014	6-20 ~ 23	6.5 節「sdc ファイルの記述」に代わって「Synplify Pro の制約ファイル生成」節として SCOPE エディタによる方法の説明
3.3	Mar. /2015	all	GUI ダンプ全面更新。また構成を全面的に変更：6.2 節を LSE、6.3 節を Synplify Pro とし、それぞれについてストラテジ、制約設定、等と節の内容を構成し直し。 6.2.4 Netlist Analyzer 記述追加 6.4 節、LSE へのコンバージョンを追加 6.5 / 6.6 節付録を追加
3.3.1 (3.3, rev1.1)	Apr. 2015	14-15	Netlist Analyzer の『find』機能について、マクロのパラメータ確認について追記
		21	HDL アトリビュート記述例追記

--- \*\*\* ---

