



LatticeECP2/M sysCLOCK PLL/DLL 設計と使用ガイド

はじめに

このユーザーズガイドはLatticeECP2M™ とLatticeECP2™ で利用できるクロックリソースとデバイス・アーキテクチャについて説明します。PLLや、DLL、クロック分周器などと共に、プライマリクロック、セカンダリクロック、エッジクロック、およびクロック要素についての詳細を明らかにします。

各パッケージごとのPLL数とDLL数を表10-1と表10-2に示します。

表10-1 PLL とDLL の個数 ～ LatticeECP2 ファミリ

デバイス	記 述	ECP2-6	ECP2-12	ECP2-20	ECP2-35	ECP2-50	ECP2-70
SPLL 数	標準PLL (GPLLのサブセット)	0	0	0	0	2	4
GPLL 数	汎用PLL	2	2	2	2	2	2
DLL 数	汎用DLL	2	2	2	2	2	2
DQSDLL 数	DDR用途DLL	2	2	2	2	2	2

表10-2 PLL とDLL の個数 ～ LatticeECP2M ファミリ

デバイス	記 述	ECP2M-20	ECP2M-35	ECP2M-50	ECP2M-70	ECP2M-100
SPLL 数	標準PLL (GPLLのサブセット)	6	6	6	6	6
GPLL 数	汎用PLL	2	2	2	2	2
DLL 数	汎用DLL	2	2	2	2	2
DQSDLL 数	DDR用途DLL	2	2	2	2	2
SERDES	4チャンネルSERDESクワッド	1	1	2	4	4

クロック/コントロール分配ネットワーク

LatticeECP2/Mファミリはグローバルクロックの分配をクワドラント・ベースのプライマリクロック8本とフレキシブルなセカンダリクロックの形式で提供します。またデバイスは上下左右の各辺に2本のエッジクロックを提供します。他のクロックソースにはクロック入力ピン、内部ノード、PLL、DLL、スレーブ遅延線 (Slave Delay Line)、およびクロック分周器(CLKDIV)が含まれます。

LatticeECP2/M トップレベル・ビュー

図10-1はLatticeECP2-50デバイスのプライマリクロック構造を示します。

図10-1 LatticeECP2-50 クロック構造

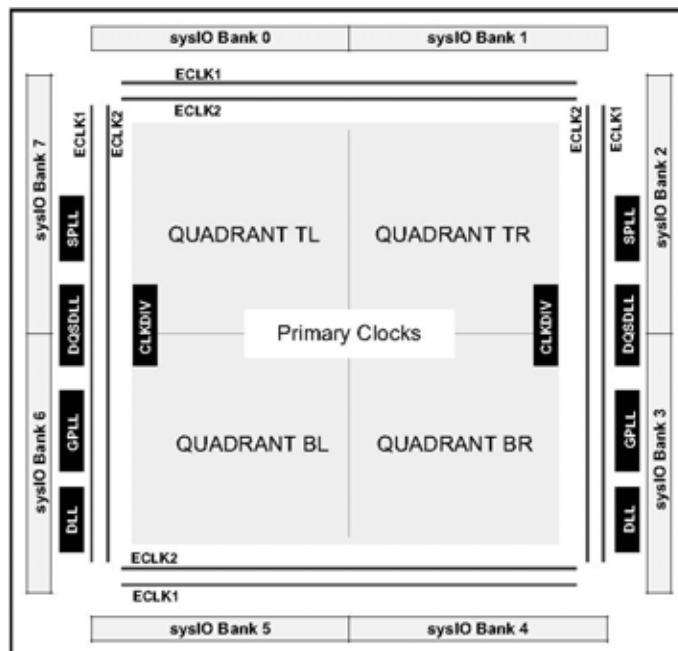
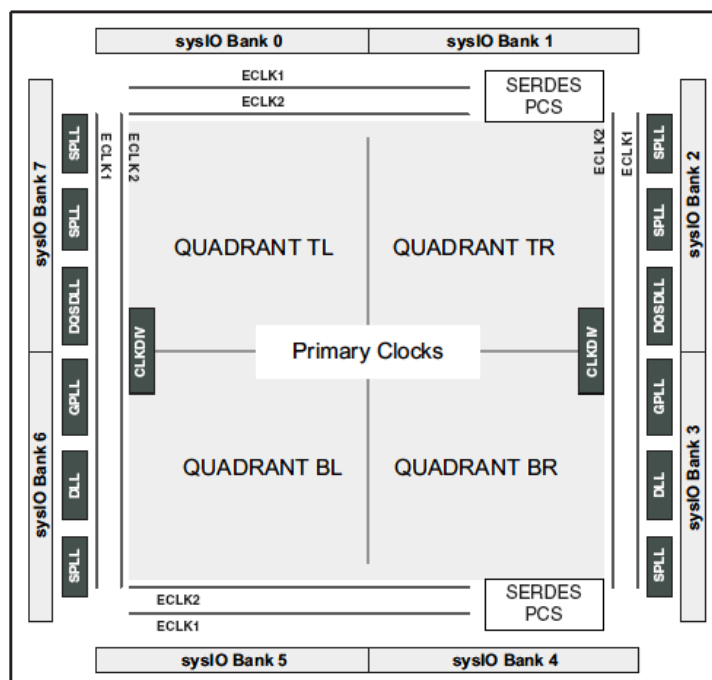


図10-2はLatticeECP2M-50デバイスのプライマリクロック構造について図示します。図には2つのSERDESブロックが示されています。上下各辺のエッジクロックは、それらがSERDESブロックとの境界に達するところで終わります。LatticeECP2Mファミリの他のメンバーでは、SERDESブロック数が異なることを除けば、同様の構造になっています。

図10-2 LatticeECP2M-50 クロック構造



プライマリクロック

各クワドラントでは最大8本のプライマリクロックを受け付けます。これらのうち2本のクロックは、ダイナミック・クロック選択(DCS)機能を提供します。DCSを用いないプライマリクロック6本を “Primary Pure” として、DCSを用いるクロック2本を “Primary DCS” として、プリマップ・デザインプランナ (Design Planner, Pre-MAP) で指定することができます。プライマリクロックのソースは以下の通りです。

- ・ PLL出力
- ・ DLL出力
- ・ CLKDIV出力
- ・ 専用のクロックピン
- ・ 内部ノード
- ・ SERDES TX_H_CLK (LatticeECP2Mのみ)

セカンダリクロック

LatticeECP2/Mセカンダリクロックはフレキシブルな領域ベースのクロックリソースです。各領域は4つの独立したクロック入力を持つことができますが、領域のリソースとしては、プライマリクロックで定義するクワドラント境界を超えることができます。

リージョンあたり8つのセカンダリクロック・マルチプレクサがあります。各マルチプレクサには、4つの異なるソースからの入力があります。これらの3本は内部ノードから来ています。4番目の入力はプライマリクロック・ピンから来ます。入力ソースは必ずしもマルチプレクサと同じリージョンに配置されるというわけではありません。この構造はセカンダリクロックのグローバルな使用を可能にします。

セカンダリクロックのソースは以下の通りです。

- ・ 専用のクロックピン
- ・ 内部ノード

エッジクロック

LatticeECP2/Mには1辺あたり2本のエッジクロックがあります。これらクロックはインジェクション (注入) 遅延とスキューが小さく、I/Oレジスタのクロックとして用いられます。エッジクロック(ECLK)リソースは、高速I/Oインターフェイスのためにファンアウト能力が高く設計されています。詳細な接続性情報について付録Bを参照してください。エッジクロックのソースは以下の通りです。

- ・ 左右辺バンクのエッジクロック
 - 専用のクロックピン
 - PLL出力
 - DLL出力
 - 内部ノード
- ・ 上下辺バンクのエッジクロック
 - 専用のクロックピン
 - 内部ノード

ECLKはセカンダリクロック・リソースと汎用配線リソースを直接ドライブすることができます。これはまた、ECLKクロックのソースを、汎用配線を通して同時にプライマリ・クロックネットに接続することができることを意味します。図10-3はセカンダリクロックとエッジクロック構造について説明しています。

図10-3 LatticeECP2-50のセカンダリクロックとエッジクロック

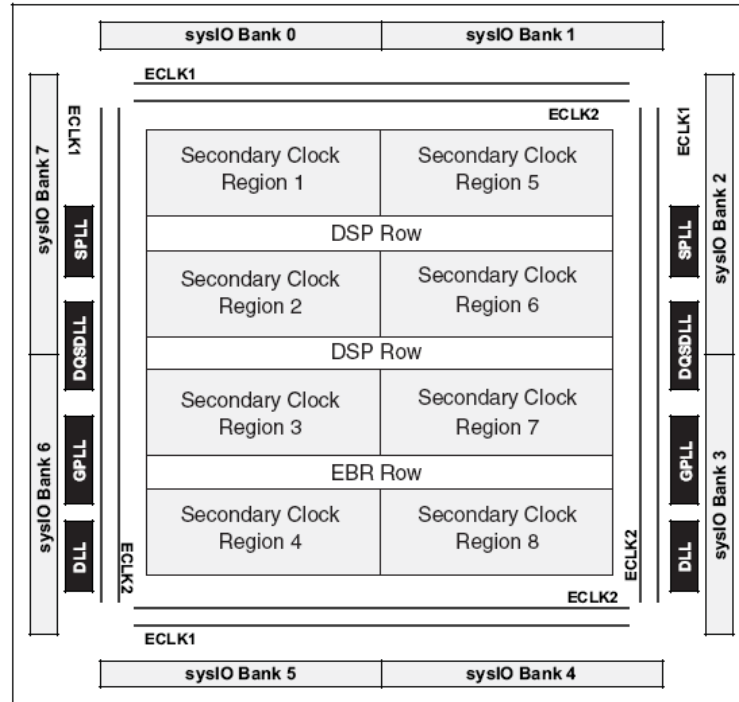
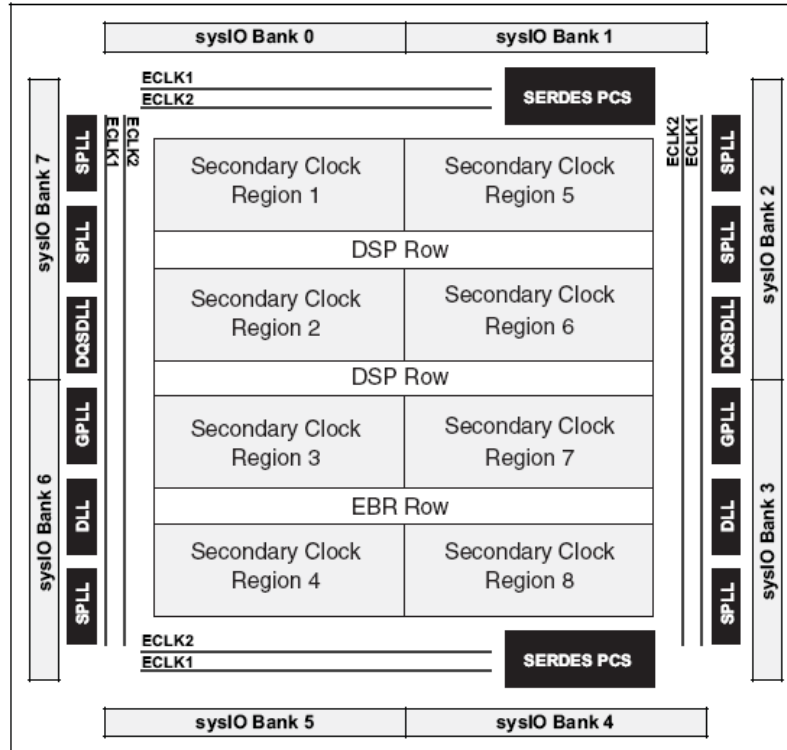


図10-4 LatticeECP2M-50のセカンダリクロックとエッジクロック



プライマリクロックについての注記

PLL性能を最適化するためにはフィードバック・ソースとしてCLKOPを用いなければなりません。

ほとんどの設計者がPLLをクロックツリー注入除去(Injection Removal)モードで用いますが、CLKOPはプライマリクロックに割り当てられるべきです。ユーザが別の方法を指定しない限り、ソフトウェアは自動的にこのように設定します。

CLKOPはCLK0からCLK5のみに配線することができ、CLKOS / CLKOKは全てのプライマリクロック(CLK0からCLK7)に配線することができます。

CLK6かCLK7がプライマリクロックとして用いられて、DCSへの入力が1本のクロックしかないとき、DCSはソフトウェアによってバッファモードとして割り当てられます。より詳細については、このドキュメントのDCSセクションを参照してください。

デザインツールでのクロック指定

必要に応じて、特定のクロックソースを分配するために、ユーザはプライマリクロック、セカンダリクロック、もしくはエッジクロックとして指定することができます。図10-5はプリマップ・デザインプランナでどのようなようになるかを示します。付録Cで議論するように、これ以外にプリファレンス・ファイルを用いることもできます。

Primary-PureとPrimary-DCS

プライマリ・クロックネットをPrimary_Pure (CLK0～CLK5)かPrimary_DCS (CLK6とCLK7)のどちらかに割り当てることができます。

グローバル・プライマリクロックとクワドラント・プライマリクロック

グローバル・プライマリクロック

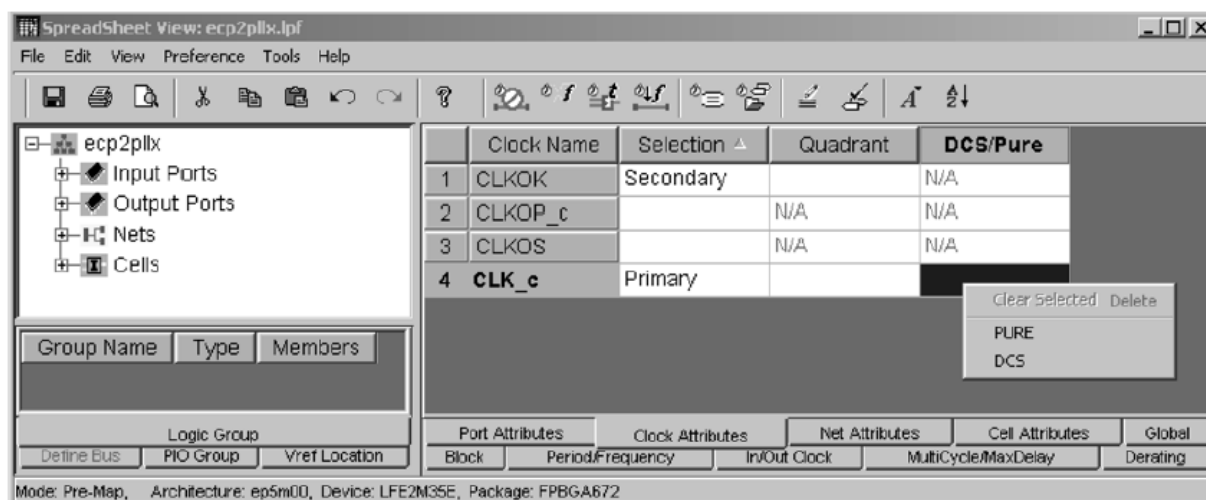
或るプライマリクロックがクワドラント・クロックとして割り当てられない場合、ソフトウェアはそれをグローバルクロックであると想定します。

6本のグローバルなPrimary_Pureクロックと2本のグローバルなPrimary_DCSクロックが利用できます。

クワドラント・プライマリクロック

どのプライマリクロックもクワドラント・クロックに割り当てることができます。クロックは単一のクワドラントか、(斜めの位置関係にない)隣接している2つのクワドラントに割り当てることができます。

図10-5 デザインプランナ (プリマップ) におけるクロック制約 (旧プリファレンスエディタ)



クワドラント・クロックネットが使用されるとき、各クロックがドライブするレジスタを、配線の問題なしにそのクワドラント内に配置することができることを、ユーザは確実にしなければなりません。

クワドラント・プライマリクロック機構では、プライマリクロック・ソースがすべて利用できる場合、プライマリクロックの最大数は32です。

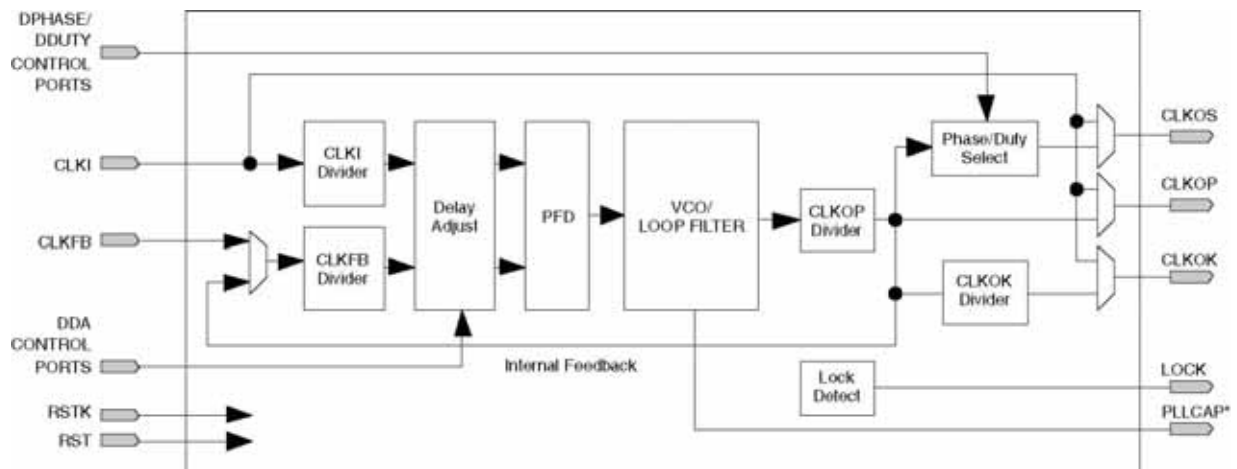
エッジクロックについての注記

デザインプランナ(プリマップ)でのエッジクロックの選択はispLEVERバージョン6.1以降で利用できます。クロックネットワーク・ダイアグラムについては付録Aを参照してください。

sysCLOCK PLL

LatticeECP2/M PLLはクロック注入遅延の除去や、周波数合成、位相 / デューティ比調整、或いはダイナミック遅延調整などの機能を提供します。図10-6はPLLのブロック図を示します。

図10-6 PLL ブロック図



機能記述

PLL分周器と遅延ブロック

入力クロック (CLKI) 分周器

CLKI分周器は、PLLブロックへの入力クロック周波数を制御するのに用いられます。分周器設定値は出力クロックの除数に直接該当します。入力分周器の入出力は、デバイス・データシートで指定された入出力周波数範囲の中になければなりません。

フィードバック・ループ (CLKFB) 分周器

CLKFB分周器は、フィードバック信号を分周するのに用いられます。実質的にこれは出力クロックを n 倍します。というのは分周されたフィードバックはPLLブロックへの入力周波数に合わせるためにスピードアップしなければならないためです。分周されたフィードバック周波数が入力周波数と等しくなるまで、PLLブロックは出力周波数を大きくします。デバイス・データシートで規定される入出力周波数範囲の中にフィードバック分周器の入出力が収まる必要があります。

遅延調整

遅延調整回路はプログラマブル・クロック遅延を提供し、130ps (nominal)増分のステップで合計1.04nsの遅れか進みを与えることができます。時間遅延設定には許容範囲があります。詳細に関してはデバイス・データシートを参照してください。このモード下では、CLKOP、CLKOS、およびCLKOKが同様に影響されます。遅延調整には、2つの動作モードがあります。

- スタティック遅延調整；このモードでは、ユーザの選択した遅延がパワーアップ時に構成されます。この機能はipsLEVERバージョン6.1からサポートされます（現在、遅延は0に固定されています）。
- ダイナミック遅延調整(DDA)；このモードでは、簡易なバスが、遅延を構成するのに用いられます。バス信号はFPGA内の回路で利用できます。

出力クロック (CLKOP) 分周器

CLKOP分周器は二つの目的を果たしていて、VCO出力のデューティサイクルを50%にすること、そしてジッタを最小にするためにVCO周波数を640MHzから1280MHzの範囲に収めることです。CLKOSが使用されているかどうかに関係なく、CLKOP分周器の値は同じです。

CLKOK 分周器

CLKOK分周器出力はグローバル・クロックネットに供給できます。これはPLLのCLKOP信号を分周値で分周し、より低い周波数のクロックを生成します。

位相調整とデューティサイクル選択

ユーザにはCLKOSの位相とデューティサイクルをプログラムするオプションがあります。位相は22.5°ステップで調整ができ、デューティサイクル精度はクロック周期の1/16です。しかし1/16と15/16デューティ比は、最小パルス幅の違反を避けるためにサポートされません。

ダイナミック位相調整 (DPHASダイナミック)とダイナミック・デューティサイクル選択 (DDUTY)

ユーザはダイナミックモードで位相調整とデューティサイクル選択を制御することができます。ダイナミックモードが選択されるとこの両方が指定できます。どちらか一方の機能のみが用いられる場合、他の制御入力を所望の固定ロジックレベルに設定することができます。

外付けコンデンサ

低い周波数の入力クロックに対応するために、外付けコンデンサを用いるオプションがあります。詳細はこのドキュメントの”オプションの外付けコンデンサ”セクションを参照してください。

PLL入力と出力

CLKI 入力

CLKI信号はPLLの基準クロックです。PLLが正しく動作するためには、（入力は）データシートの仕様に準拠しなければなりません。CLKIは、専用のデュアルファンクション・ピン、もしくは配線から引き出すことができます。

RST 入力

PLLリセットは2つの条件で発生します。パワーアップ時に、コンフィグレーション・ブロックからの内部パワーアップ・リセット信号がPLLをリセットします。PLLモジュールの一部にはユーザ制御のPLLリセット信号RSTがあり、内部的に生成されたリセット機能やピンからの入力ドライブすることができます。このRST信号は内部のすべてのPLLカウンタ、(M分周器を含む)フリップフロップ、およびチャージポンプをリセットします。M分周器リセットはM分周器出力を入力クロックと同期させます。RSTが非アクティブになると、PLLはロックイン・プロセスを開始し、PLLがロックを完了するには t_{LOCK} 時間がかかるでしょう。図10-7はRST入力のタイミングを示します。RSTはHighアクティブです。

RST信号はオプションです。

図10-7 RST入力タイミング図

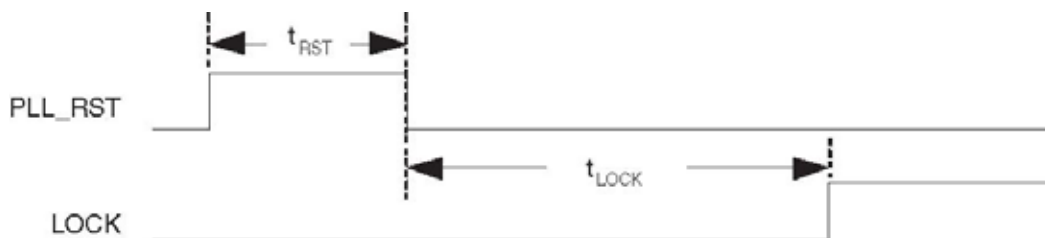
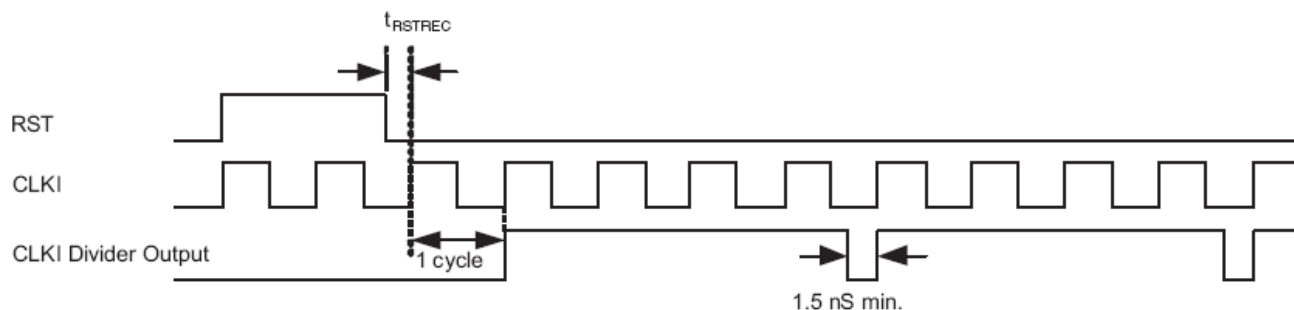


図10-8はRST入力とCLKI分周器出力とのタイミング関係を示します。

図10-8 RST入力とCLKI分周器出力とのタイミング関係 (CLKI_DIV=4の例)

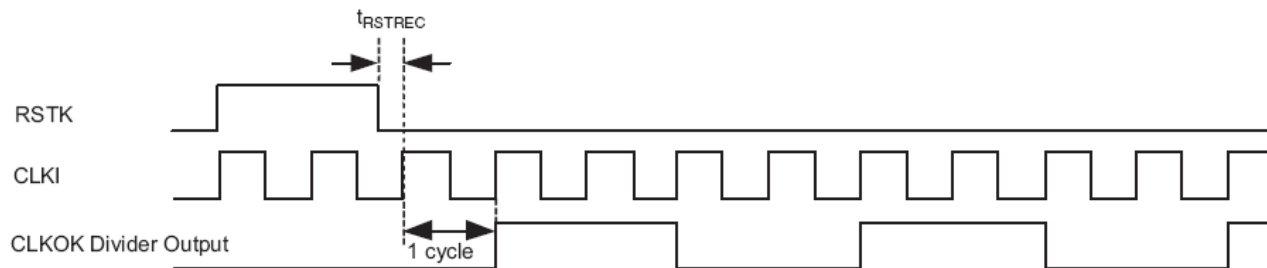


RSTK入力

RSTKはK分周器のためのリセット入力です。K分周器リセットは、K分周器出力クロックを入力クロックと同期させるのに用いられます。LatticeECP2/MはI/Oセルでオプションのギアボックスを、出力と入力の両方に持っています。K分周器リセットはギアボックス実装の役立ちます。RSTKはHighアクティブです。

図10-9はRSTK入力とCLKOK分周器出力とのタイミング関係を示します。

図10-9 RSTK入力とCLKOK分周器出力とのタイミング関係 (CLKOK_DIV=4の例)



CLKFB入力

フィードバック分周器を介して加えられるPLLへのフィードバック信号として与えることができるのは、プライマリ・クロックネット(CLKOP)、所望のピン、CLKOP分周器から直接(内部フィードバック)、または汎用配線からです。外部フィードバックは、ボードレベルのクロック・アライメントとしての補償を可能にします。

CLKOP出力

sysCLOCK PLLの主出力CLKOPはプライマリクロックとエッジクロックとして選択できる信号です。このクロック信号はCLK_OUTピンで利用できます。

位相とデューティサイクル選択できるCLKOS出力

sysCLOCK PLLの補助クロック出力CLKOSも、プライマリクロックとエッジクロックとして選択できる信号です。CLKOSは位相シフト、そして/または、デューティサイクル調整が必要なときに使用されます。プログラマブル位相シフトは22.5度の増分で異なる位相を得ることができます。デューティ選択機能はクロック周期の1/16刻みでのデューティ選択を提供します。この機能はダイナミック・コントロールモードでもサポートされます。

低い周波数のCLKOK出力

より低い周波数が必要なとき、CLKOKが利用できます。これもプライマリクロックとして選択できる信号です。

ダイナミック遅延コントロール/ダイナミック位相調整/ダイナミック・デューティ比

これらの機能に関する詳細については本ドキュメントの後ろで説明されます。これら機能のための入出力ポートは表10-3で示されています。

表10-3 ダイナミック遅延調整、ダイナミック位相およびデューティ比調整用のポート

パラメータ	I/O	記 述
DDAMODE	I	DDA(Dynamic Delay Adjust; ダイナミック遅延調整)モード。"1"; ピン制御(ダイナミック)、"0"; ヒューズ制御(スタティック)
DDAIZR	I	DDA遅延ゼロ。"1"; 遅延=0、"0"; 遅延=ON
ZZAILAG	I	DDA進み・遅れ。"1"; 進み、"0"; 遅れ。
DDAIDEL[2:0]	I	DDA遅延ステップサイズ
DPAMODE	I	DPA(Dynamic Phase Adjust/Duty Cycle Select; ダイナミック位相調整/デューティ比選択)モード。"1"; ピン制御 (ダイナミック)、"0"; フューズ制御 (スタティック)
DPHASE[3:0]	I	DPA 位相調整入力
DDUTY[3:0]	I	DPA デューティサイクル選択入力

LOCK出力

LOCK出力はPLLのステータスに関する情報を提供します。デバイスがパワーアップされた後、入力クロックが有効になると、PLLは仕様にあるロック時間内にロックを達成します。ロックがいったん達成されると、PLLロック信号がアサートされます。PLLが動作中に、入力クロックかフィードバック信号が無効になると、PLLはロックを失います。PLLを基準クロックと再同期させるにはPLL_RSTを与えることを推奨します。LOCK信号はリセット信号などとしてFPGA内ユーザロジックで利用できます。

PLLCAP

このポートはソフトウェア・モジュールに含まれていません。代わりに、これはデバイスのPLLCAPピンに直接配線されています。詳細はこのドキュメントの“オプションの外付けコンデンサ” セクションを参照してください。

PLLアトリビュート（属性）

PLLはいくつかのアトリビュートを利用しますが、これはソース中の制約によってPLLのコンフィグレーションができます。以下のセクションはこれらのアトリビュートとそれらの使い方を述べます。

FIN

入力周波数は分周器設定に基づきますが、仕様にある周波数範囲の値です。

CLKI_DIV, CLKFB_DIV, CLKOP_DIV, CLKOK_DIV

これらの分周器は各出力クロックの周波数を決定します。ユーザは無効な組み合わせを入力することはできません。入力周波数、分周器、およびPLL仕様で決定されます。

注；LatticeECP/EC、LatticeXP、MacoXOデバイスにおけるPLLと異なって、CLKOSが使用されているかどうかに関係なく、CLKOP 分周値は同じです。CLKOP_DIV値は、CLKI_DIVとCLKFB_DIV値と共に、FINとCLKOP_FREQの規定された範囲に基づいて、その中で f_{VCO} を最大にするように計算されます。これらの値の設定は、出力クロック・デューティ比ができるだけ50%に近くなるように計算されます。

Frequency_Pin_CLKI, Frequency_Pin_CLKOP, Frequency_Pin_CLKOK

これらの出力クロック周波数は分周器の値を決定します。

CLKOP周波数公差

必要な出力周波数が達成できない場合、クロック出力の周波数公差を入力しても良いでしょう。

FDEL

本フィーチャはispLEVERバージョン6.1以降で利用できます。

PHASEADJ

PHASEADJアトリビュートは、CLKOS出力用に粗い(Coarse)位相シフトを選択するのに用いられます。位相調整は22.5度の増分でプログラマブルです。

DUTY (デューティサイクル)

DUTYアトリビュートは、CLKOS出力のデューティサイクルを選択するのに用いられます。デューティサイクルは1/16周期の増分でプログラマブルです。2/16～14/16がサポートされます。1/16と15/16のデューティ比は、最小パルス幅のバイオレーションを避けるためにサポートされません。

FB_MODE

CLKFB分周器をドライブするフィードバック・ソースとしては3つあり、それらは内部、クロックツリー、そして外部フィードバックです。クロックツリー・フィードバックがデフォルトで用いられます。内部フィードバックは、フィードバックパス遅延を最小にするために、クロックツリーの前のCLKOP分周器出力から取り出します。ユーザクロック・フィードバックは専用ピン、クロックピンまたはユーザの指定した内部ロジックからドライブされます。

DELAY_CNTL

このアトリビュートは、遅延調整モードを選択するように設計されます。アトリビュートが "DYNAMIC" に設定されると、遅延制御はDDAMODEピンの入力ロジックに従って、ダイナミックとスタティックの間で切り替わります。アトリビュートが "STATIC" に設定されると、このモードでダイナミック遅延入力は無視されます。

PHASE / DUTY_CNTL

このアトリビュートは、位相調整/デューティ比選択モードを選択するように設けられています。アトリビュートが "DYNAMIC" に設定される場合、位相調整/デューティ比選択制御は、DPAMODEピンの入力論理によってダイナミックとスタティックの間で切り替わります。アトリビュートが "STATIC" に設定される場合、ダイナミック位相調整/デューティ比選択入力はこのモードでは無視されます。

CLKOS / CLKOKの選択

デザインで用いられるときだけ、ユーザはこれらの出力クロックを選択します。

CLKOP / CLKOS / CLKOK BYPASS

セットされるとそれぞれのバイパスがイネーブルされます。CLKIは直接対応する出力クロックに配線されます。

RESET/RSTKの選択

デザイン内で用いるときだけ、ユーザはこれらのリセット信号を選択します。

LatticeECP2/M PLLモジュール

ユーザがIPexpressを用いてPLLモジュールを生成すると、モジュールはPLLライブラリ要素とモジュールに必要な付加的周辺ロジックから成るラッパーが得られます。図10-10は典型的なPLLモジュール図を示します。モジュールポート名はライブラリ要素によっては、幾つかのケースで異なる場合があります。モジュールのポート名は、IPexpressウィンドウか、或いは生成されたモジュールのソースコードでわかります。これらは、ユーザのデザインで接続されるポートです。またIPexpressは、インスタンス化のテンプレートファイルも作成し、デザイン内でPLLモジュールをどのようにインスタンスするかを示します。ユーザは生成されたソースコード、または<filename>.lpcファイルをプロジェクト内にインポートすることができます。

図10-10 LatticeECP2/M、IPexpressで生成される典型的なPLLモジュール

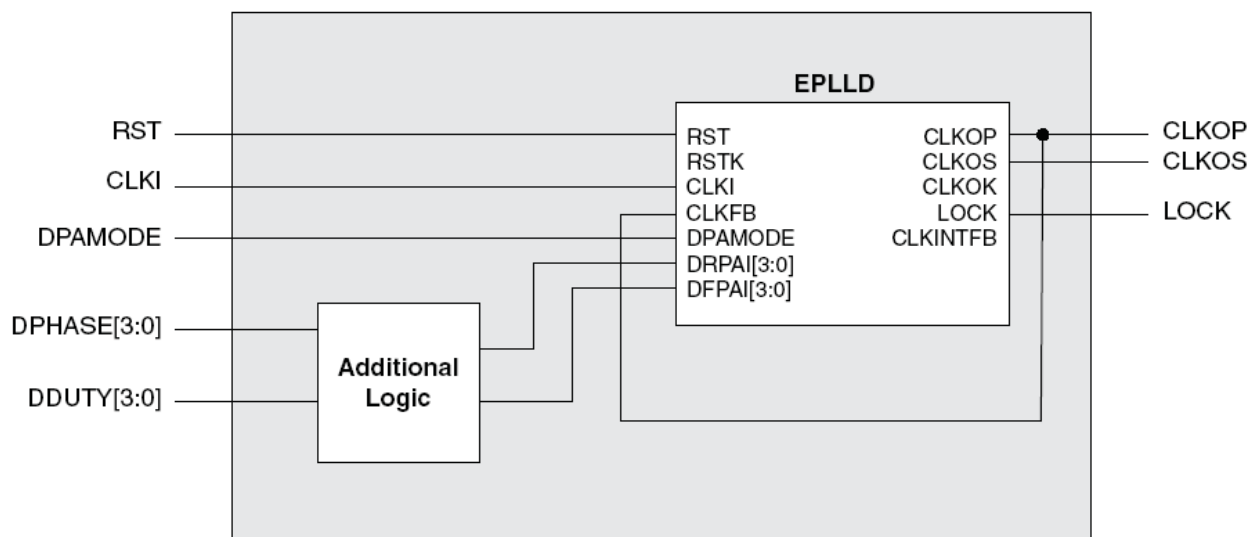


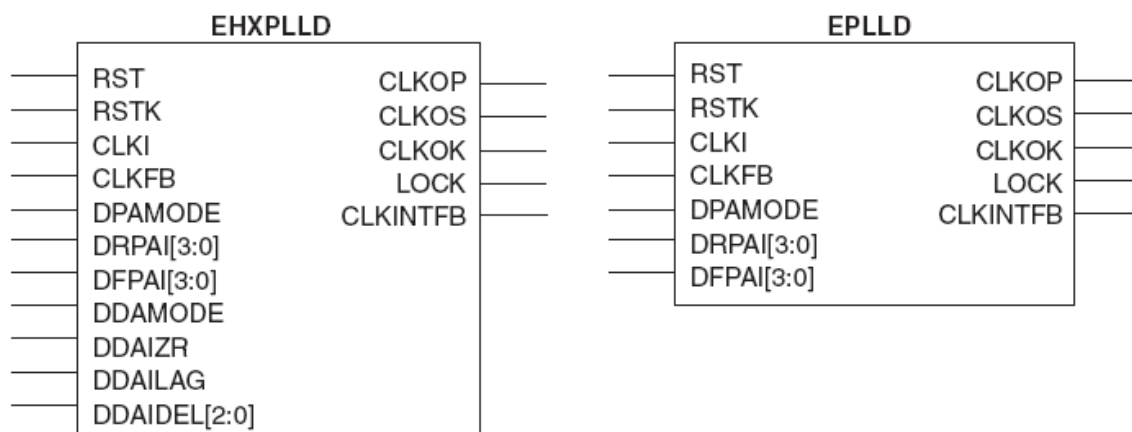
図10-10に示すPLLモジュールは、ユーザがPLLリセット信号、PLLロック信号、ダイナミック位相/デューティサイクル各ポートと共に、CLKOPとCLKOSポートを用いることを指定した例を示します。また、ソフトウェアがCLKOP信号をプライマリ・クロックツリーを介してCLKFBポートに接続するように、CLKOPフィードバックを指定しています。ユーザはCLKI、RST、DPAMODE、DPHASE[3:0]、DDUTY[3:0]、CLKOP、CLKOS、およびLOCK信号をそれぞれデザイン内の信号と接続します。

LatticeECP2/M PLLライブラリ定義

すべてのLatticeECP2/MデバイスがフルフィーチャのPLLであるGPLL（General Purpose PLL；汎用PLL）を2つサポートします。さらに、より大きいデバイスの幾つかには、GPLL機能のサブセットを持つSPLL(Standard PLL；標準PLL)が2つ～4つあります。

2つのPLLライブラリ要素がLatticeECP2/M PLL実装のために定義されています。図10-11はLatticeECP2/M PLLライブラリのシンボルを示します。GPLLはEPLLDかEHXPLLDのどちらかとして構成でき、SPLLはEPLLDとしてのみ構成できます。

図10-11 LatticeECP2/M PLL ライブラリ・シンボル



ダイナミック遅延調整DDA (EHXPLDのみ)

ダイナミック遅延調整はDDAMODE入力で制御されます。DDAMODE入力が 1 に設定されるとき、遅延制御は入力、DDAIZR、DDAILAG、およびDDAIDEL(2:0)によって行われます。このモードにおいてはアトリビュートDELAY_CNTLを“DYNAMIC”に設定しなければなりません。表10-4はアトリビュート/入力設定に基づく遅延調整値を示します。

このモードで、PLLは位相の突然の変化のためロックがはずれるかもしれません。PLLを再ロックさせるためにはRSTをアサートしなければなりません。RSTのネゲート時に、PLLはロックイン・プロセスを開始し、PLLがロックを完了するには t_{LOCK} 時間がかかるでしょう。

表10-4 遅延調整

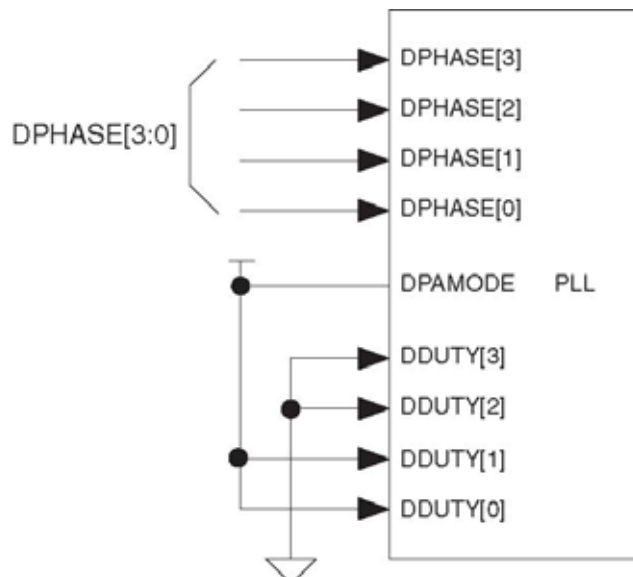
DDAMODE = 1: ダイナミック遅延調整			遅延 1 $t_{DLY} = 130ps$ (公称値)	DDAMODE = 0
DDAIZR	DDAILAG	DDAIDEL[2:0]		等価 FDEL 値
0	1	111	進み 8 T_{DLY}	-8
0	1	110	進み 7 T_{DLY}	-7
0	1	101	進み 6 T_{DLY}	-6
0	1	100	進み 5 T_{DLY}	-5
0	1	011	進み 4 T_{DLY}	-4
0	1	010	進み 3 T_{DLY}	-3
0	1	001	進み 2 T_{DLY}	-2
0	1	000	進み 1 T_{DLY}	-1
1	Don't Care	Don't Care	遅延なし	0
0	0	000	遅れ 1 T_{DLY}	1
0	0	001	遅れ 2 T_{DLY}	2
0	0	010	遅れ 3 T_{DLY}	3
0	0	011	遅れ 4 T_{DLY}	4
0	0	100	遅れ 5 T_{DLY}	5
0	0	101	遅れ 6 T_{DLY}	6
0	0	110	遅れ 7 T_{DLY}	7
0	0	111	遅れ 8 T_{DLY}	8

ダイナミック位相 / デューティ・モード

このモードはダイナミック位相調整とダイナミック・デューティ選択の両方を同時に設定します。これらは2つのモード、"Dynamic Phase and Dynamic Duty" および "Dynamic Phase and 50% Duty" です。

固定デューティ比があるダイナミック位相調整を用いるには、単に必要なデューティ比の値に相当する DDUTY[3:0] 入力を設定してください。図10-12は回路例について示します。この例は、ユーザが必要なデューティサイクルが3/16であると想定しています。

図10-12 デューティサイクルを3/16に固定するダイナミック位相調整セットアップ例



ダイナミック位相調整/デューティサイクル選択

位相調整設定は表10-5で説明されています。

表10-5 ダイナミック位相調整の設定

DPHASE[3:0]	スタティックモードでの等価な PHASEADJ
0000	0
0001	22.5
0010	45.0
0011	67.5
0100	90.0
0101	112.5
0110	135.0
0111	157.5
1000	180.0
1001	202.5
1010	225.0
1011	247.5
1100	270.0
1101	292.5
1110	315.0
1111	337.5

デューティサイクル選択設定は表10-6で説明されています。

表10-6 ダイナミック・デューティサイクル選択の設定

DDUTY[3:0]	スタティックモードでの等価 DUTY(1/16期間)	コメント
0000	0	非サポート
0001	1	非サポート
0010	2	
0011	3	
0100	4	
0101	5	
0110	6	
0111	7	
1000	8	
1001	9	
1010	10	
1011	11	
1100	12	
1101	13	
1110	14	
1111	15	非サポート

オプションの外付けコンデンサ

EXHPLLDとEPLLDは共にオプションの外付けコンデンサを用いることで、オンチップ・ループフィルタの周波数特性を変えることができます。外付けコンデンサが使用されているとき、位相検出器入力(Fpd)における周波数は2MHz程度まで低くでき、従ってPLLの下側動作範囲を広げることが可能になります。外付けコンデンサは上側動作範囲に効果はありません。IPexpress™ は、外付けコンデンサが必要かどうか決定するために位相検出器周波数をチェックします。

外付けコンデンサの有無に対するPLLパラメータの許容範囲はLatticeECP2/Mファミリ・データシートで説明されています。

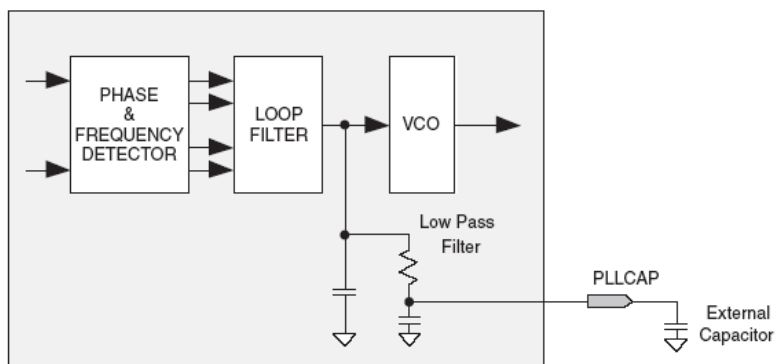
オプション外付けコンデンサの推奨仕様

- 値； 5.6 nF, +/- 20%
- タイプ； セラミックのチップコンデンサ、NPO誘電体
- パッケージ； 1206、又はそれ以下

各デバイスには2本の外付けコンデンサ用ピンがあり、左側PLLのために1本、および右側PLLのための1本です。これらのピンは決まった位置にあります。これらは専用ピンで、ユーザI/Oとしては使用できません。

外付けコンデンサ・ピンがデバイスの一方の側のPLLによって用いられるとき、いかなる他のPLLも同じ側でそのピンを用いることはできません。これは、1デバイスあたり最大2個のPLL(左側で1つ、右側で1つ)で外付けコンデンサを取り付けることができることを意味します。

図10-13 外付けコンデンサの使い方

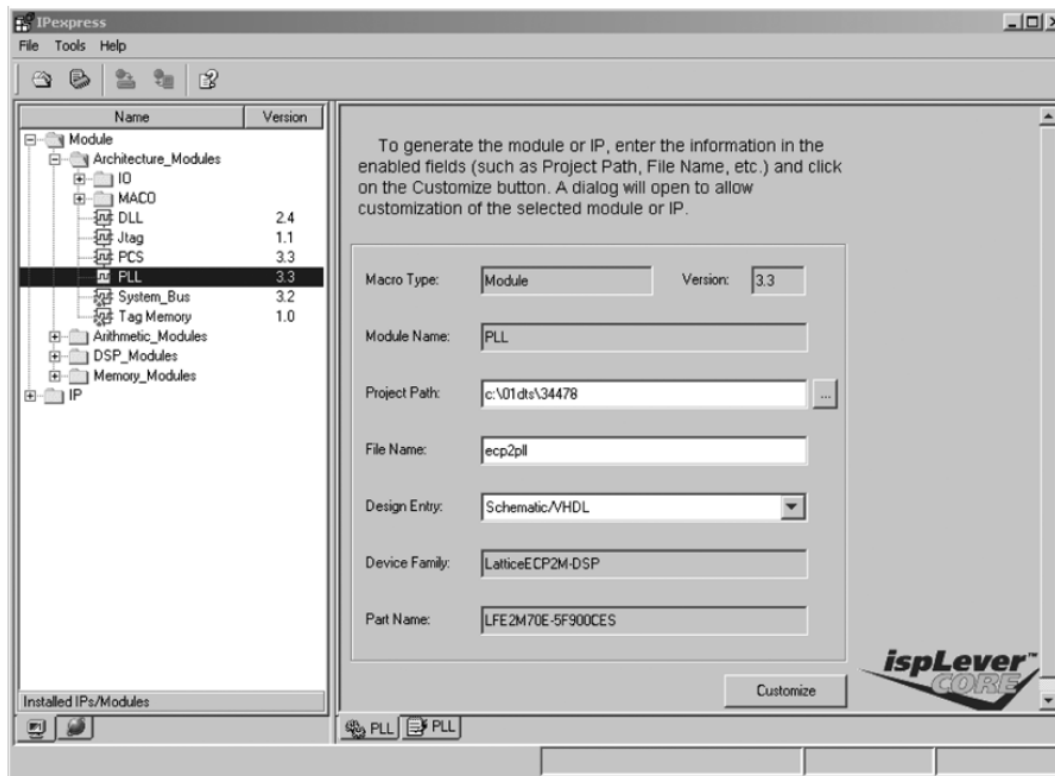


IPexpressでのPLLの設定方法

PLLを生成してコンフィギュレーションする為にIPexpressが用いられます。設計者は、PLL用にパラメータを選択するためにグラフィカル・ユーザインターフェースを用います。その結果は、シミュレーションと合成フローで用いられるHDLが生成されます。

図10-14は、PLLが選択された場合のメインウィンドウを示します。このウィンドウで必要な唯一の入力はモジュール名です。他のエントリーはプロジェクト設定を元にセットされます。必要に応じてこれらのエントリーを変えても構いません。モジュール名を入れた後にCustomizeをクリックすると、図10-15で示されるようなConfiguration Tabウィンドウが開きます。

図10-14 IPexpressメインウィンドー



コンフィグレーション・タブ

Configuration Tabは、ユーザがアクセスできる全アトリビュートをデフォルト値に設定してリストアップします。完了した後、Generateをクリックしてソースコードと制約ファイルを生成します。ユーザは、パラメータをロードするために .lpcファイルを用いることもできます。

モード

PLLのConfiguration Tabで構成する2つのモードがあり、それらは周波数モードと分周器モードです。

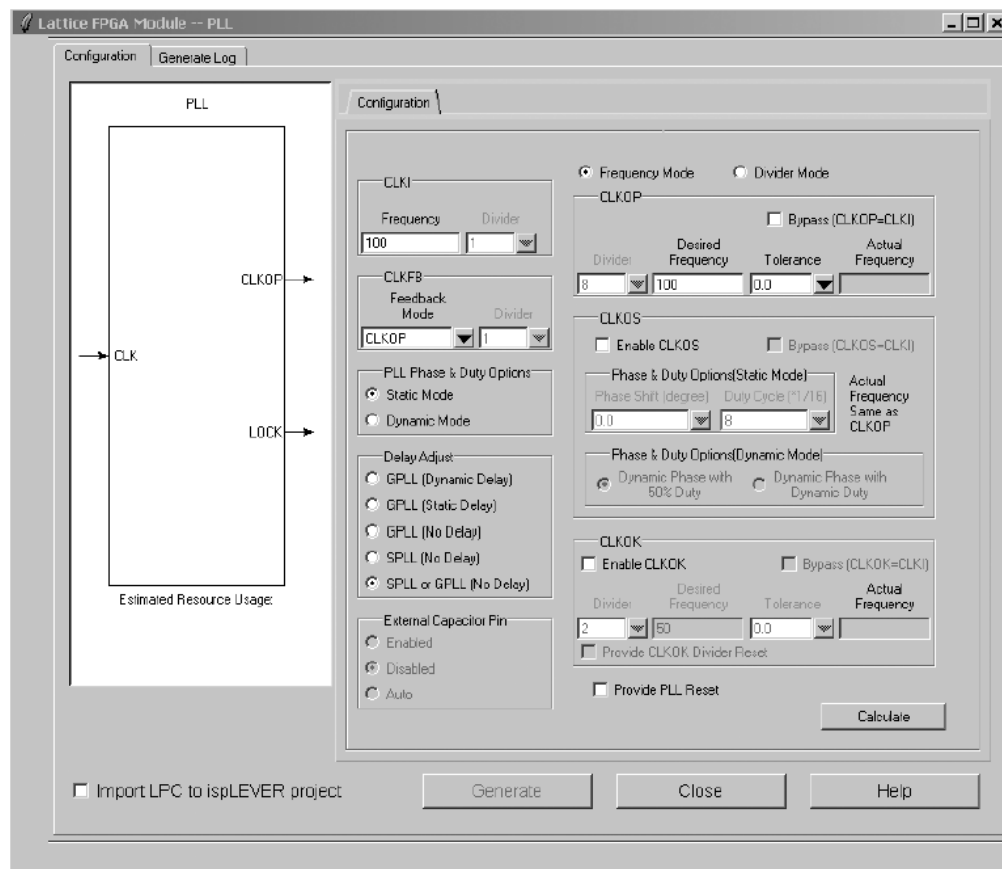
周波数モード (Frequency Mode)

このモードでは、ユーザは入出力クロック周波数を入力し、ソフトウェアが分周器の設定値を計算します。ユーザが入力した出力周波数が実現できない場合、最も近い周波数が “Actual” テキスト・ボックスに表示されます。入出力周波を入力した後に、**Calculate**ボタンをクリックすると、周波数が表示されます。

分周器モード (Divider Mode)

このモードでは、ユーザは入力周波数と分周値を設定します。ユーザがPLL動作に詳しいことを想定しています。最適なPLL性能を実現するために、ユーザは、 f_{VCO} を最大にするCLKOP分周値を選ばなければなりません。入力周波数と分周器値を設定した後に**Calculate**ボタンをクリックすると、周波数が表示されます。図10-15にコンフィグレーション・タブを示します。

図10-15 LatticeECP2/M PLL コンフィグレーション・タブ



注；GUI中の**External Capacitor Pin**部ではテキストが灰色で表示され、PLL設定に従った外付けループ・コンデンサの要件を示します。これはユーザーに対して外付けループ・コンデンサが必要であることの注意を喚起する意図です。**Auto**設定は、PAR（配置配線）ステップでPLLがSPLLかGPLLのどちらかに配置された後、ソフトウェア(ispLEVER)が外部コンデンサが必要かどうかを決定することを意味します。各セクションがイネーブルされると、他のグレイアウトされている部分が入力可能になります。

表10-7はIPexpress GUIの全ユーザーパラメータについて記述しています。

表10-7 コンフィギュレーションGUIのユーザパラメータ

パラメータ		記 述	値の範囲	デフォルト
Frequency Mode		所望の入・出力周波数	ON / OFF	ON
Divider Mode		所望の入力周波数と分周値設定	ON / OFF	OFF
CLKI	Frequency	入力周波数。外付けコンデンサなし	25M (33M ¹) Hz ~ 420MHz	100MHz
		入力周波数。外付けコンデンサあり	2MHz ~ 420MHz ²	--
	Divider	分周値。外付けコンデンサなし	1 ~ 16 (12 ¹)	1
		分周値。外付けコンデンサあり	1 ~ 64	--
CLKFB	Feedback Mode	フィードバック・モード	Internal, CLKOP, User Clock	CLKOP
	Divider	分周値。外付けコンデンサなし	1 ~ 16 (12 ¹)	1
		分周値。外付けコンデンサあり	1 ~ 10	--
CLKOP	Bypass	PLLのバイパス ; CLKOP=CLKI	ON / OFF	OFF
	Desired Frequency	所望出力周波数。外付けコンデンサなし	25M (33M ¹) Hz ~ 420MHz	100MHz
		所望出力周波数。外付けコンデンサあり	5MHz ~ 50MHz ²	--
	Divider	CLKOP分周器の設定値(Divider Mode)	2, 4, 8, 16, 32, 48, 64, 80, 96, 112, 128	8
	Tolerance	許容できるCLKOP誤差	0.0, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0	0.0
	Actual Frequency	実際の周波数 (リードオンリ)	--	--
CLKOS	Enable	CLKOS出力をイネーブル	ON / OFF	OFF
	Bypass	PLLのバイパス ; CLKOS=CLKI	ON / OFF	OFF
	Phase – Static	CLKOSスタティック位相シフト	0,22.5,45, ...,337.5	--
	Duty – Static	CLKOSスタティック・デューティ比選択	2 ~ 14	8
	Dynamic Phase with 50% Duty	ダイナミック位相と50%デューティ比	ON / OFF	ON
	Dynamic Phase with Dymanic Duty	ダイナミック位相とダイナミック・デューティ比	ON / OFF	ON
CLKOK	Enable	CLKOK出力をイネーブル	ON / OFF	OFF
	Bypass	PLLのバイパス ; CLKOK=CLKI	ON / OFF	OFF
	Desired Frequency	所望出力周波数。外付けコンデンサなし	0.195MHz ~ 210MHz	50MHz
		所望出力周波数。外付けコンデンサあり	0.016MHz ~ 25MHz ³	--
	Divider	CLKOK分周器の設定値(Divider Mode)	2 ~ 128 (偶数のみ)	2
	Tolerance	許容できるCLKOK誤差	0.0, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0	0.0
	Actual Frequency	実際の周波数 (リードオンリ)	--	--
PLL Phase & Duty Option		ダイナミック/スタティックモードの選択	Dynamic / Static Mode	Static
Delay Adjust		ダイナミック遅延/スタティック遅延/遅延なし の選択	Dynamic / Static / No Delay	No Delay ⁴
Provide PLL Reset		PLLリセットポートを提供	ON / OFF	OFF
Provide CLKOK Divider Reset		CLKOK分周リセットを提供	ON / OFF	OFF
Import LPC to ispLEVER project		.lpcファイルのプロジェクトへのインポート	ON / OFF	OFF

1. 値はSPLLに適用。他の全ての値はGPLLとSPLLの両方に適用。
2. 位相比較器の入力周波数範囲は 2MHz ～ 50MHz
3. $f_{IN} < 5\text{MHz}$ では $f_{OUT_max} = 10 * f_{IN}$
4. IPexpress では “Delay Adjust” セクションの設定に従って、ユーザがGPLLかSPLLを選択するか、或いはソフトウェアが選択するかを指定できる

周波数の計算

表10-8は位相検出器入力周波数範囲について示しています。ユーザは、位相検出器周波数が規定範囲の中におさまるように、CLKI分周器とCLKFB分周器の値を選択しなければなりません。

“M=CLKI分周器の値” とします。

“N=CLKFB分周器の値” とします。

“V=CLKOP分周器の値” とします。

基本式は以下の通りです。

$$\text{CLKOP周波数} = \text{CLKI周波数} * N / M$$

$$f_{VCO} (\text{VCO周波数}) = \text{CLKOP周波数} * V$$

$$f_{PFD} (\text{PFD周波数}) = \text{CLKI周波数} / M = \text{CLKFB周波数} (= \text{CLKOP周波数}) / N$$

例: CLKI周波数が外付けコンデンサなしで25MHzである場合、CLKI分周器の値は1です。

PLL位相検出器の周波数範囲

表10-8 位相検出器周波数 (f_{PFD}) の範囲

PLLタイプ	外付けコンデンサ	周波数範囲
GPLL	外付けコンデンサなし	25MHz ～ 420MHz
	外付けコンデンサあり	2MHz ～ 50MHz
SPLL	外付けコンデンサなし	33MHz ～ 420MHz
	外付けコンデンサあり	2MHz ～ 50MHz ¹

1. $f_{IN} < 5\text{MHz}$ では $f_{OUT_max} = 10 * f_{IN}$ です

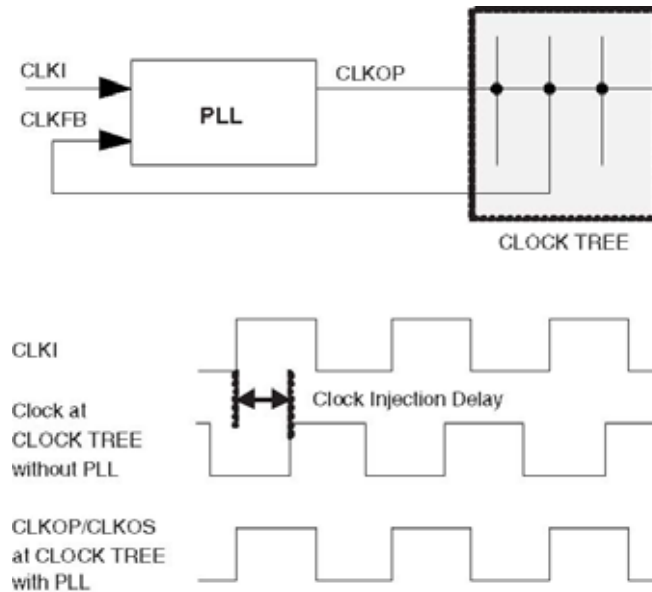
PLL動作モード

PLLには論理設計で多くの用途があります。最もよく用いられるのは、クロック注入（遅延の）除去とクロック位相調整です。これら2つの動作モードについて以下で説明します。

PLLクロック注入除去

このモードで、PLLはクロック注入遅延を減少させるために用いられます。クロック注入遅延は、デバイスの入力ピンからフリップフロップなどの目的とするエレメントまでの遅延です。PLLの位相検出器はCLKFBをCLKIにアライン（整合）させます。CLKFB信号がクロックツリー(CLKOP)から来る場合、結果としてPLLの遅延とクロックツリーの遅延を取り除きます。図10-16はブロック図と波形について例示します。

図10-16 クロック注入遅延除去アプリケーション

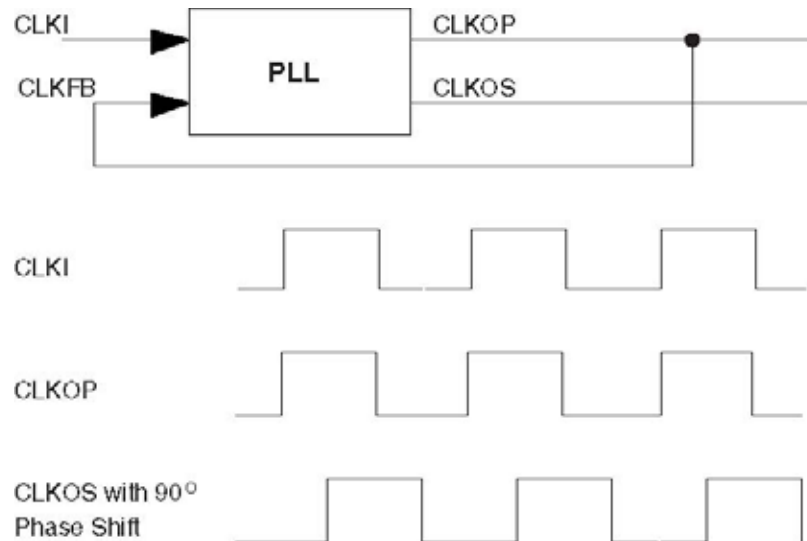


PLLクロック位相調整

このモードで、PLLは22.5度ステップで固定の位相関係を生成するために用いられます。固定位相の生成は、クロックとデータとの特定の関係が必要なクロック転送インターフェイスで役立ちます。

固定の位相関係はCLKIとCLKOSの間、またはCLKOPとCLKOSの間で用いることができます。

図10-17 CLKOPからのCLKOSの位相調整



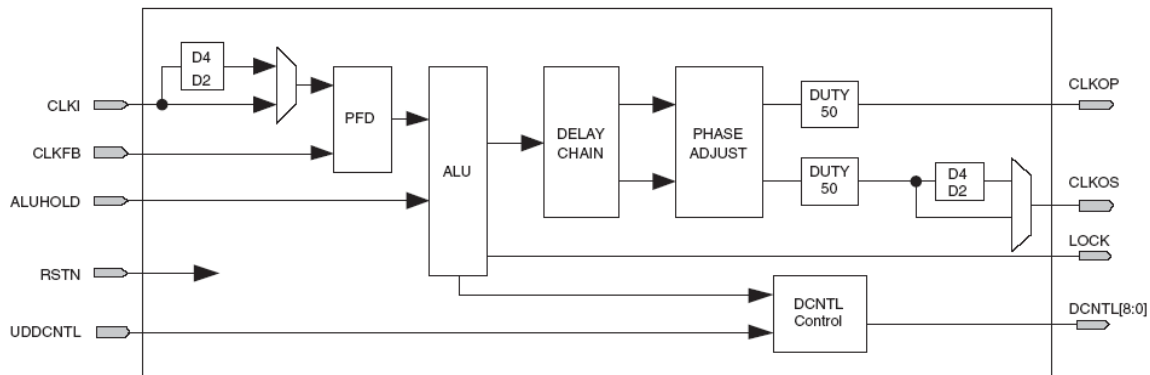
sysCLOCK DLL

LatticeECP2/M DLLはクロック注入遅延除去や、遅延マッチ、時間リファレンス遅延(90度位相遅延)、および出力位相調整などの機能を提供します。DLLは、特定の位相関係を作成するために、CLKI入力信号に遅延を加えるることによるクロック操作を実行します。DLL出力には2つのタイプがあります。

1番目はPLL CLKOPとCLKOSと同様のクロック信号です。他方のタイプの出力は遅延制御ベクタ

(DCNTL[8:0])です。遅延制御ベクタは、DLL内の遅延セルにマッチし、I/Oロジックに配置されたスレーブ遅延線（Slave Delay Line；DLLDEL）要素に接続されます。この遅延ベクタで、DLLは入力信号をダイナミックに特定の量を遅らせられます。図10-18はLatticeECP2/M DLLブロック図を示します。

図10-18 LatticeECP2/M DLL ブロック図



クロック注入遅延除去と出力位相調整は共にDLLのクロック出力だけを用います。時間リファレンス遅延モードと遅延マッチ・モードは、遅延制御ベクタ出力を使用します。本ドキュメントでは後ろでこれらの機能の特定の例について議論します。

DLLの概要

LatticeECP2/M DLLはIPexpressによって生成され、コンフィギュレーションされます。以下はDLLのポート名と記述のリストです。DLLの実装に用いられる3つのライブラリ要素があります。即ちCIDDLA (Clock Injection Delay)、およびTRDDL (Time Reference Delay)です。IPexpressは、ユーザの選択に基づくカスタマイズされたDLLモジュールを作成するため、これらのライブラリ要素の1つをラッピングします。

DLLの入力と出力

CLKI入力

CLKI信号はDLLの基準クロックです。どのようなタイプのFPGA配線やピンも、CLKIの入力ソースとして使用できます。このDLL毎のCLKI入力は最小の遅延と最良の性能を提供します

CLKFB入力

ユーザクロック信号をフィードバックとして用いるか、或いはユーザが、クロック遅延マッチ・モードを選ぶ場合にだけ、CLKFB入力を利用できます。内部フィードバックかCLKOS / CLKOPがフィードバックに用いられる場合、その接続はモジュール内部で行われます。クロック注入遅延除去モードでは、DLLは入力クロックを遅らせることによって、入力クロックの位相をフィードバック・クロック位相に合わせます。

クロック注入遅延マッチ・モードでは、DLLはCLKIとCLKFB信号間の差を計算します。そして、この遅延値はDCNTLベクタとして出力となります。DLL CLKFB入力としては、本ドキュメントの後ろで議論するDLLあたり1本の推奨ピンがあります。このピンは最小の遅延と最良の性能を提供します。

CLKOP出力

DLLの出力はCLKIレートに基づいています。CLKOP出力はプライマリクロックとエッジクロック配線をドライブすることができます。

CLKOS出力

CLKIレートに基づPLL出力で、分周と同時に、或いは位相シフトができます。CLKOS出力はプライマリクロックとエッジクロック配線をドライブすることができます

DCNTL[8:0] 出力

このDLL出力は、信号を特定の分だけ遅らせるために用いられます。DCNTL[8:0]ベクタはスレーブ遅延線エレメントに接続します。そして、DLLは単独のDLLから複数の入力遅延を制御することができます。

UDDCNTL入力

この入力はDCNTL[8:0]のアップデートをイネーブルするか、またはディセーブルするのに用いられます。DLLブロック内の同期化部で信号を捕捉することを確実にするため、アップデートが必要なときに、少なくとも2クロック周期と等しい時間Highにドライブされる必要があります。信号がHighにドライブされ、その状態に保持されていれば、DCNTL[8:0]出力は継続的にアップデートされます。

ALUHOLD入力

このHighアクティブの入力は、DLLがCLKI信号に遅延を加えたり減じたりするのを止めます。DCNTL[8:0]、CLKOP、およびCLKOS出力は、まだ有効ですが、現在の遅延設定から変化しません。

ロック出力

Highアクティブのロックインジケータ出力です。CLKIとCLKFB信号が同位相にあるとき、LOCK出力はHighです。CLKI入力が止まると、LOCK出力はアサートされたままになります。クロックが止められるので、LOCK出力をネゲートするためのクロックがないためです。これは、入力クロックが止まるとVCOがフリーランするPLLの動作と異なることに注意してください。

RSTN

Lowアクティブ入力でDLLをリセットします。また、GSRNは随意にDLLをリセットすることができます。DLLにリセットが必要な場合、そのリセット信号はFPGAロジックのリセットと同じにしないことを推奨します。通常のロジックは、リセット条件の間、クロックが動作していることを必要とします。データパス・リセットがDLLをリセットするように設計すると、ロジックのクロックソースも止まるため、これはロジックに問題を引き起こすかもしれません。

DLLアトリビュート

LatticeECP2/M DLLはソース制約や、IPexpress、またはプリファレンス・ファイルによってコンフィグレーションを可能にする幾つかのアトリビュートを利用します。以下のセクションはこれらのアトリビュートとそれらの用い方を詳しく述べます。

2分周もしくは4分周のCLKOS出力時のDLL Lock

通常、DLLは周波数で1倍も分周もできない“x1倍”のリソースです。しかしLatticeECP2/M DLLは“2分周”或いは“4分周”のCLKOS出力が得られます。オプションで二つの2分周/4分周ブロックがCLKOSと共にCLKI入力に置かれ、これによりDLLフィードバックパスに分周されたCLKOSの使用を可能にします。このため、DLLは2分周か4分周クロックにクロック注入除去を実行でき、I/Oバッファ動作のDDRX2モードとDDRX4モードで有用です。

このオプションのクロック分周器がCLKOS出力パスだけで用いられるとき、DLLは異なる周波数でかつ時間が合わせられたクロックを2系統出力できます。分周器が2分周か4分周に設定されるとき、“ダミー遅延”がCLKOP出力パスに挿入され、クロックをCLKOS分周器のQ遅延に一致させます。

オプションとしてのクロックの微細位相シフト

CLKOS出力パスにあるオプションの微細(fine)位相シフトは、メインの遅延チェーン内にある他の4ブロックと一致する遅延ブロックから作られています。この位相シフトブロックはCLKOS出力の通常位相に対して45度、22.5度、或いは11.25度の位相シフトが可能です。

GSR

もしGSRがイネーブルされていると、これによりPLLとDLLをリセットすることができます。GSRキーワードをENABLEDかDISABLEDに設定できます。このオプションはIPexpress GUIにあります。以下に本制約の使用に関する例を示します。

```
ASIC "dll/dll_0_0" TYPE "CIDDLLA" GSR=DISABLED;
```

DLLロック時間の制御

CLKIとCLKFB位相が一致すると、DLLがロックします。シミュレーション環境で、ロック時間は100 μ s(デフォルト)に固定されています。HDLパラメータ、もしくはバックアノテーション・シミュレーション用の制約(プリファレンス)を通してこの値を変えることができます。DLLはLOCK_DELAYというパラメータを含んでおり、ロック出力がHighになるまでの合計時間を、 μ sec単位の整数値で入力できます。以下はフロントエンド・シミュレーションでこの値をどう設定するかについての例です。

Verilog:

```
defparam mydll.mypll_0_0.LOCK_DELAY=500;
mydll dll_inst, (.CLKI(clkin), .CLKOP(clk1), .CLKOS(clk2))
```

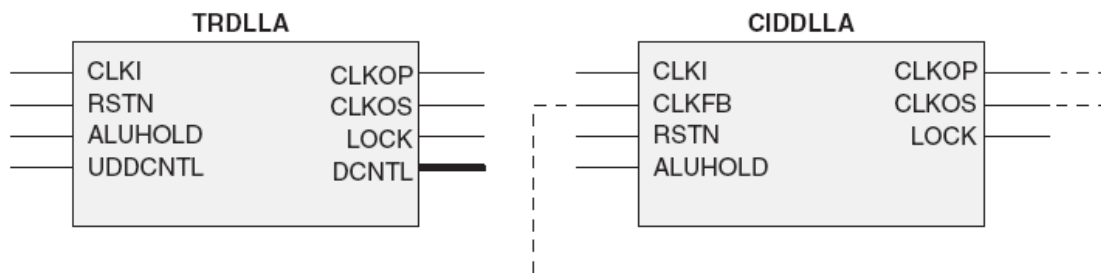
VHDL:

サポートされません。バックアノテーション・シミュレーションのためには、LOCK_DELAYはプリファレンス(制約)ファイルの中で設定される必要があります。以下はPLLの例です。

```
ASIC "pll/pll_0_0" TYPE "EHXPLLA" LOCK_DELAY=200;
```

DLLライブラリ・シンボル

図10-19 DLL ライブラリ・シンボル



DLLライブラリ要素の定義

ラティス・ライブラリは設計者がDLLを活用することができるライブラリ要素を含んでいます。これらは、"DLLアトリビュート" セクションで定義される属性を用います。

2つの動作モードが、以下にリストアップされているプリミティブ(又はライブラリ要素)として提示されます。

表10-9 DLL ライブラリ要素

ライブラリ名	動作モード	記 述
TRDLLA	時間参照遅延DLL	本モードは4位相のクロック（0 / 90 / 180 / 270）と共に、これらの位相を生成するために用いられる制御設定を生成します。
CIDDLLA	クロック注入除去DLL （4遅延セルモード）	本モードはクロックツリー遅延を取り除き、外部フィードバック・クロックを基準クロックに揃えます。それには、4番目の遅延ブロックから来る一本の出力があります。

DLL ライブラリの I/O

表10-10 DLL ライブラリ要素の I/O 記述

信号	I/O	記 述
CLKI	I	専用クロック入力ピン、他の I/O または論理ブロックからのクロック入力ピ
CLKFB	I	専用フィードバック入力ピン、内部フィードバック、他の I/O、または論理ブロックからのクロック・フィードバック入力ピン。この信号はユーザは選択できません
RSTN	I	Low アクティブの同期リセット。専用ピンか内部ノードから
ALUHOLD	I	"1" は ALU をフリーズします。TRDLLA、および CIDDLLA 用
UDDCNTL	I	CIB からのアクティブ High 同期イネーブルで、PIC 遅延のデジタル制御をアップデートするための信号。少なくとも2クロック周期の間、High にドライブされる必要があります
DCNTL[8:0]	O	デジタル遅延制御コード
CLKOP	O	全てのモード用のプライマリクロック出力
CLKOS	O	セカンダリクロック出力で、微細(fine)位相シフト、及び/または、2分周か4分周されています
LOCK	O	アクティブ High の位相ロックインジケータ。基準クロックとフィードバック・クロックが同位相にあることを意味します

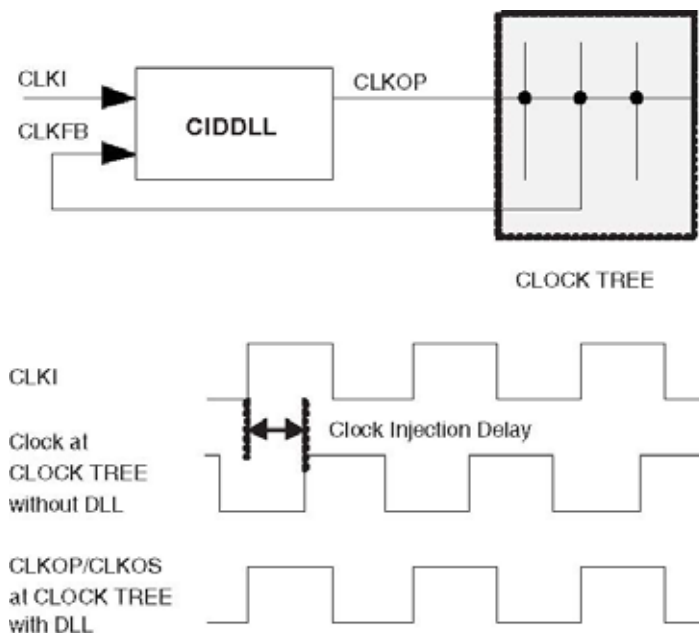
注: 周波数の仕様値についてはデバイス・データシートを参照して下さい。

DLL 動作モード

クロック注入除去モード (CIDDLLA)

クロック注入遅延を減少させるために DLL を用いることができます。クロック注入遅延は、デバイスの入力ピンからフリップフロップなどの目的とする要素に達するまでの遅延です。DLL は、CLKI を CLKFB に合わせるために、CLKI 入力に遅延を加えます。CLKFB 信号がクロックツリー (CLKOP、CLKOS) から来る場合、クロック経路全体から DLL とクロックツリーの遅延を取り除きます。図10-20は回路例と波形を示します。

図10-20 DLLによるクロック注入遅延除去

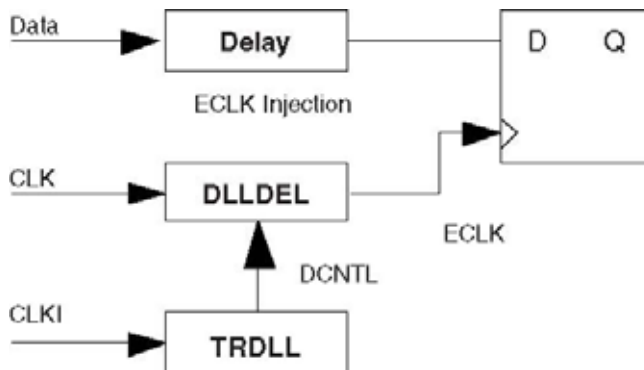


また、クロック注入除去モードはDCNTLポートを提供できます。このモードでは、他の入力信号も同じ量遅らせることができるように、CLKI信号に付加された遅延値がDCNTLポートに出力されます。これは、同じ回路で複数のクロックが用いられる場合、必要となるDLLの数を最小にするために非常に役立ちます。DCNTLを用いるとき、DLL遅延はDCNTLベクタの範囲に制限されます。従ってIPexpressはCLKIを300MHzから700MHzに制限するでしょう。

時間参照遅延モード(TRDLLA ; 90度の位相遅延)

DLLの時間参照遅延(TRDLLA)モードは、DCNTLベクタとして生成する90°の遅延量を計算するために用いられます。このモードは、DDRインターフェイスの場合のように、クロックを90°遅延させる場合に役立ちます。図10-21はこのモードの回路例を示します。

図10-21 時間参照遅延の回路例



このモードで、CLKIはクロック入力を受け付けます。DLLはDCNTLベクタを生成し、CLKI信号の一周期に対しての90°分、入力信号を遅らせます。そしてこのDCNTLベクタをスレーブ遅延線(DLLDELA)に接続し、CLKIの90°分、信号を遅らせられます。

IPexpressでのDLLの生成

IPexpressはDLLを生成して構成するのに用いられます。IPexpress GUIでユーザはDLLのパラメータを選択します。結果はシミュレーションと論理合成で用いられるHDLモデルです。

コンフィグレーション・タブ

- **Usage Mode** (使用モード) ~ DLLのモード (時間参照遅延; Time Reference Delay TRDLLA、またはクロック注入遅延除去; Clock Injection Delay Removal CIDDLLA) を選択します。この選択でGUIのさらなるオプションをイネーブルするか、またはディセーブルするでしょう。
- **CLKI Frequency** (CLKI周波数) ~ MHz単位のCLKI入力レート。
- **CLKOS Divider** (CLKOS分周器) ~ CLKOS出力の分周器を、分周なし、2分周、または4分周に設定します。
- **CLKOS Phase Shift** (CLKOS位相シフト) ~ CLKOP出力に対するCLKOSの位相オフセットを設定します。選択された位相シフト量に従って、CLKOPはCLKOSより位相が進みます。位相の増分は11.25°刻みです。プルダウンリストの数値は小数点に簡略化されています。
- **CLKFB Feedback Mode** (CLKFBフィードバック・モード) ~ DLLのフィードバック・モードをCLKOP、CLKOS、またはUser Clockのどれかに設定します。CLKOP / CLKOSを選択すると、特定の出力クロックのクロックツリー注入遅延を取り除きます。User Clockを選択すると、DLLにはCLKFBポートが提供されます。
- **CLKFB Frequency** (CLKFB周波数) ~ これはUser Clockフィードバック・モードのみで用いられます。
- **Provide RSTN Port** (RSTNポートの提供) ~ RSTNポートにより、ユーザ信号によってDLLをリセットできます。
- **Enable GSR to Reset DLL** (DLLリセット用GSRのイネーブル) ~ 選択されると、DLLはGSRによってリセットされます。ユーザ信号は不要です。RSTNポートも用いることができます。
- **Provide DCNTL Port** (DCNTLポートの提供) ~ クロック注入遅延モードで、CLKIポートに付加された遅延をDCNTLポート上に得ることが可能です。DCNTLベクタをDELAYエレメントに接続することで、他のクロック入力を同じ量遅らせるために、これを用いることができます。

PLL / DLLの従属接続

PLLとDLLを接続する場合いくつかの方法が可能です。3つの従属接続方法があり得ます。

- PLLからPLL
- PLLからDLL
- DLLからDLL

DLLをPLLに接続することはできません。

遅延設定を変えるとき、DLLは突然の変化を出力クロックに発生させます。PLLは、これをPLLが正しくロックされるのを妨げる急激な位相変化であるとみなします。

IPexpress出力

デザインでの使用に重要なIPexpressの出力が2つあります。1番目はファイル <module_name>.[v | vhd] です。これは論理合成とシミュレーションの両方で用いられる、ツールが生成するユーザ命名のモジュールです。第二のファイルはテンプレートで <module_name>_tmpl.[v | vhd] です。このファイルはモジュールの

インスタンス例を含んでいます。このファイルは、ユーザがインスタンス例をコピー/ペーストして使用するよう意図されており、論理合成フローやシミュレーションで直接用いるものではありません。

PLL / DLL用に、IPexpressは選択されたデータレートに特定したアトリビュートをHDLモジュールに設定します。これらのアトリビュートを容易に変えることもできますが、PLL / DLLの性能を維持するためには、GUIを再操作することによってのみそれらを変更するべきです。デザインフローのマッププロセスの後、PLL / DLLから生成されるクロックに制約を与えるために、FREQUENCY制約が自動的にプリファレンス・ファイルに含まれます。

クロック分周器(CLKDIV)

クロック分周器は高速クロックを2分周、4分周、および8分周します。そして、分周出力をプライマリクロックやセカンダリクロック入力として用いることができます。クロック分周器は、低速FPGAクロックをシフトレジスタ(x2、x4、x8)や、DDR/SPI4 I/Oロジック・インターフェイスに供給するのに用いられます。2つのクロック分周器が1つつつデバイスの左右両側にあります。

CLKDIVライブラリ要素の定義

ユーザはこのセクションで定義されるようなソースコードでCLKDIVをインスタンスすることができます。図10-22、表10-11、および表10-12はCLKDIVBの定義について説明します。

図10-22 CLKDIV ライブラリ・シンボル

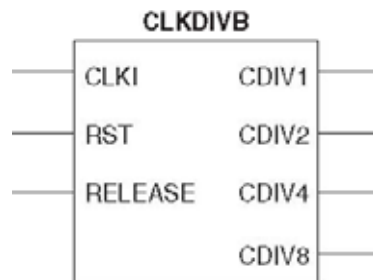


表10-11 CLKDIVB ポートの定義

名 前	記 述
CLKI	クロック入力
RST ¹	リセット入力。非同期にすべての出力を強制的にLowにします
RELEASE ¹	入力クロックに同期して出力をリリースします
CDIV1	1分周出力
CDIV2	2分周出力
CDIV4	4分周出力
CDIV8	8分周出力

1. 注: 未使用のRSTはグラウンドに接続し、未使用のRELEASEはVCCに接続しなければなりません。

表10-12 CLKDIVB アトリビュートの定義

名 前	記 述	値	デフォルト
GSR	GSRイネーブル	ENABLED / DISABLED	DISABLED

VHDLソースコードにおけるCLKDIV宣言

```

COMPONENT CLKDIVB
-- synthesis translate_off

```

```

        GENERIC (
            GSR : in String);
    -- synthesis translate_on
    PORT (
        CLKI,RST, RELEASE:          IN std_logic;
        CDIV1, CDIV2, CDIV4, CDIV8:OUT std_logic);
    END COMPONENT;

    attribute GSR : string;
    attribute GSR of CLKDIVinst0 : label is "DISABLED";

begin
CLKDIVinst0:CLKDIVB
-- synthesis translate_off
    GENERIC MAP(
        GSR => "disabled"
    );
-- synthesis translate_on
    PORT MAP(
        CLKI  => CLKIsig,
        RST   => RSTsig,
        RELEASE => RELEASEsig,
        CDIV1 => CDIV1sig,
        CDIV2 => CDIV2sig,
        CDIV4 => CDIV4sig,
        CDIV8 => CDIV8sig
    );
end

```

VerilogソースコードにおけるCLKDIVの使い方

```

module clkdiv_top(RST,CLKI,RELEASE,CDIV1,CDIV2,CDIV4,CDIV8);
    input CLKI,RST,RELEASE;
    output CDIV1,CDIV2,CDIV4,CDIV8;

    CLKDIVB CLKDIBinst0 (.RST(RST),.CLKI(CLKI),.RELEASE(RELEASE),
        .CDIV1(CDIV1),.CDIV2(CDIV2),.CDIV4(CDIV4),.CDIV8(CDIV8));

    defparam CLKDIBinst0.GSR = "DISABLED";
endmodule

```

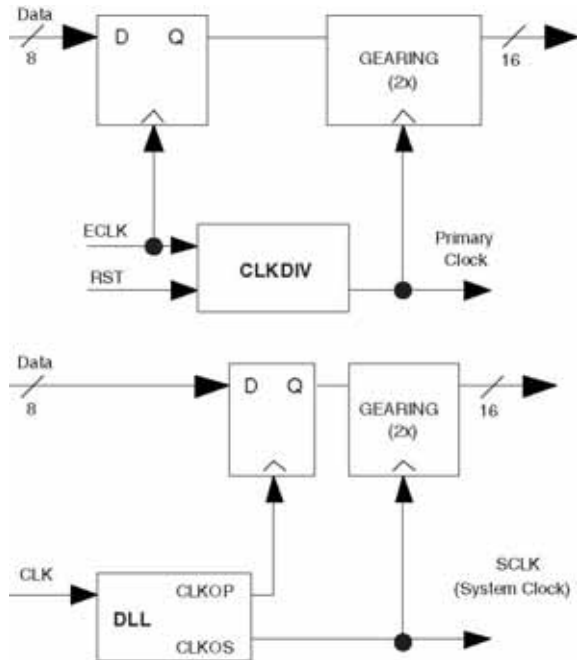
CLKDIVの回路例

クロック分周器(CLKDIV)は、クロックを2分周か4分周することができ、プライマリクロック・ネットワークをドライブします。クロック分周器は、低速FPGAクロックをI/Oシフトレジスタ(x2、x4)とDDR(x2、x4) I/Oロジック・インターフェイスに提供するのに役立ちます。8分周が遅い速度/低電力動作に提供されています。

I/Oロジックで同期転送を保証するためには、CLKDIVクロック入力のエッジクロックから来ることと、出力がプライマリクロックをドライブすることが必要です。この場合、それらの位相は揃っています。これは特に図10-23で示されるように、Mux / DeMuxギアリングがデータバス全体を同期させるために使用されているとき、I/Oロジックを同時にリセットするために役立ちます。エッジクロック配線の低スキュー特性を用いて、Mux / DeMuxギアリングを同期させる為に、データバスの全ビットにリセットを与えることができます。

第二の回路は、x2とx4アプリケーションのためにDLLをCLKDIVに取り替えることができることを示しています。

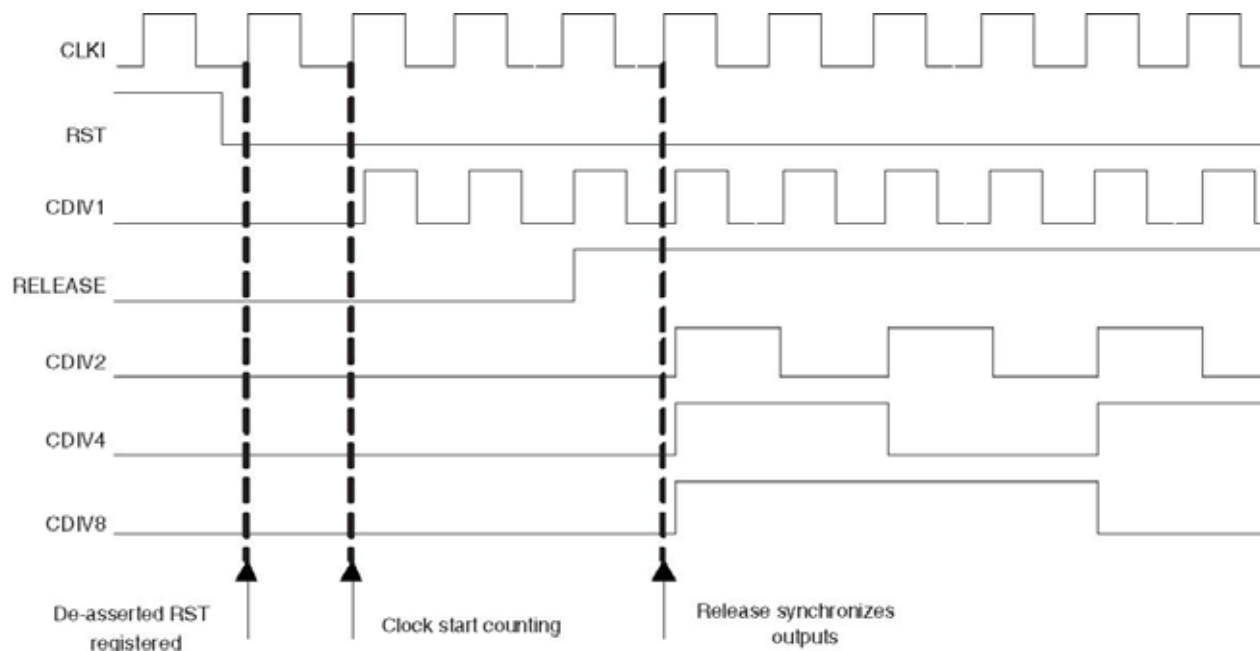
図10-23 CLKDIVアプリケーションの例



リリースの振舞い

ポートRELEASEは、RSTがネグートされた後に全出力を同期させるために用いられます。図10-24はRELEASEの振舞いについて図示します。

図10-24 CLKDIV RELEASEの振舞い



DLLDEL(スレーブ遅延線Slave Delay Line)

スレーブ遅延線は、DDR/SPI4アプリケーションに必要な遅延を生成するように設計されています。遅延コントロール入力(DCNTL[8:0])は汎用DLL出力から加えられます。ライブラリ要素の定義は図10-25と表10-13

で説明されています。

図10-25 DLLDELA ライブラリ・シンボル

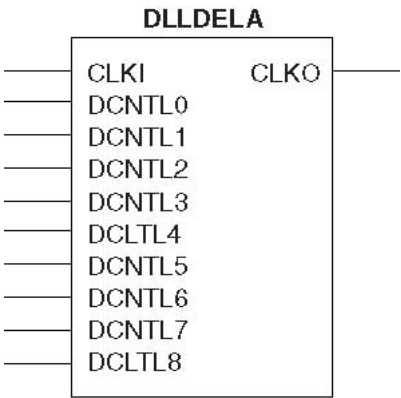


表10-13 DLLDELA I/O

名 前	I/O	記 述
CLKI	I	クロック入力
DCNTL[8:0]	I	遅延制御ビット
CLKO	O	クロック出力

VHDLソースコードにおけるDLLDELAの宣言

```

COMPONENT DLLDELA
  PORT (
    CLKI :IN std_logic;
    DCNTL0 :IN std_logic;
    DCNTL1 :IN std_logic;
    DCNTL2 :IN std_logic;
    DCNTL3 :IN std_logic;
    DCNTL4 :IN std_logic;
    DCNTL5 :IN std_logic;
    DCNTL6 :IN std_logic;
    DCNTL7 :IN std_logic;
    DCNTL8 :IN std_logic;
    CLKO :OUT std_logic
  );
END COMPONENT;

begin
  DLLDELAinst0: DLLDELA1
    PORT MAP (
      CLKI => clkisig,
      DCNTL0 => dcntl0sig,
      DCNTL1 => dcntl1sig,
      DCNTL2 => dcntl2sig,
      DCNTL3 => dcntl3sig,
      DCNTL4 => dcntl4sig,
      DCNTL5 => dcntl5sig,
      DCNTL6 => dcntl6sig,
      DCNTL7 => dcntl7sig,
      DCNTL8 => dcntl8sig,
      CLKO => clkosig
    );
end

```

VerilogソースコードにおけるDLLDELAの使い方

注；DLL0 (TRDLLA) はIPexpressでサブモジュールとして生成する必要があります。

```
module dldel_top (rst,d,clkkin,clkkin2,clkout,aluhold,uddcntl,q);

input rst,d,clkkin,clkkin2,aluhold,uddcntl;
output clkout,q;

wire [8:0]DCntl_int;
reg qint;

    DLL0 dllinst0 (.clk(clkkin2), .aluhold(aluhold), .uddcntl(uddcntl), .clkop(),
        .clkos(), .dcntl(DCntl_int),.lock());
    DLLDELA delinst0 (.CLKI(clkkin),.DCNTL0(DCntl_int[0]),.DCNTL1(DCntl_int[1]),
        .DCNTL2(DCntl_int[2]), .DCNTL3(DCntl_int[3]), .DCNTL4(DCntl_int[4]),
        .DCNTL5(DCntl_int[5]), .DCNTL6(DCntl_int[6]), .DCNTL7(DCntl_int[7]),
        .DCNTL8(DCntl_int[8]), .CLKO(clk90)); //synthesis syn_black_box
    assign clkout = clk90;
    assign q = qint;

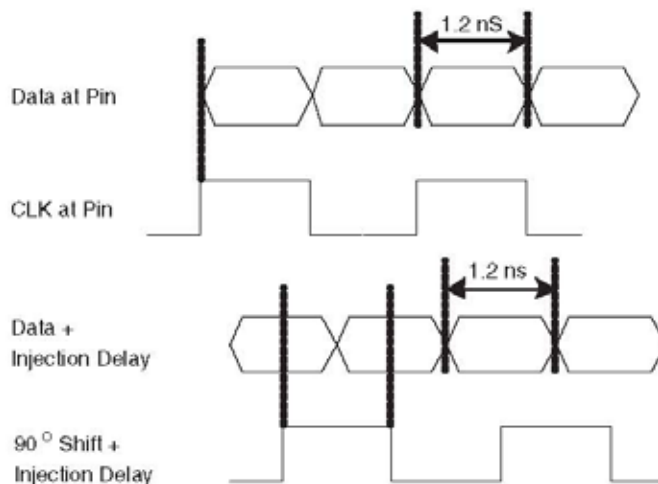
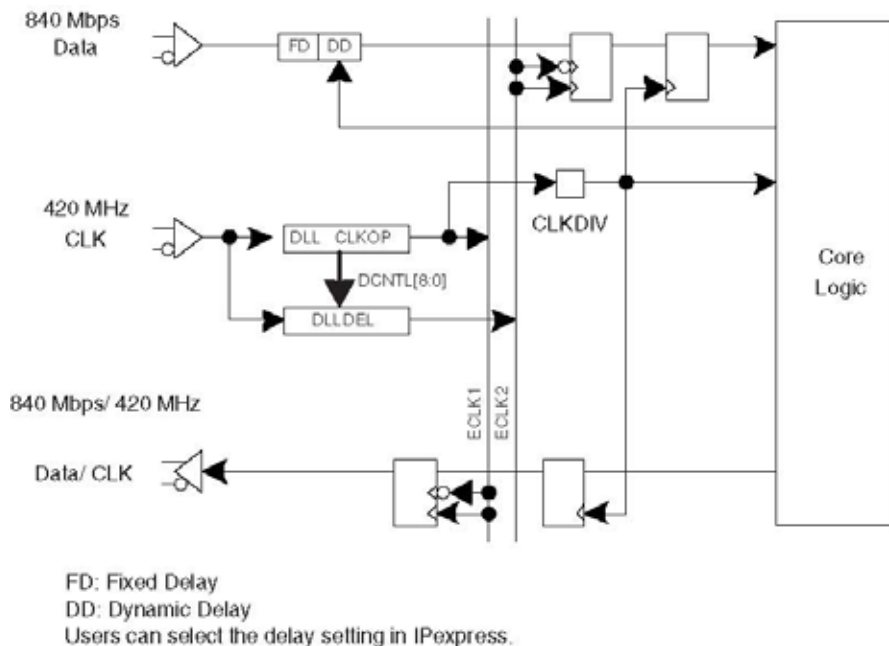
    always@(posedge clk90 or negedge rst)
        if (~rst)
            qint =1'b0;
        else
            qint = d;

endmodule
```

DLLDELAアプリケーションの例

図10-26はDLLDELアプリケーションの例を示します。タイミング・ダイヤグラムで示されるように、DLLDELはデータウィンドウの中央に両エッジを置くためにクロックを90°シフトさせます。

図10-26 SPI4.2とDDRレジスタ・インターフェイスのアプリケーション



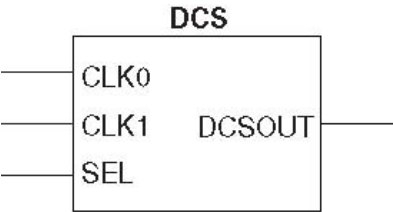
DQSDLLとDQSDEL

LatticeECP2/Mデバイス・ファミリにはDLL、スレーブ遅延線、DQSDLL、およびDQSDELの別の組み合わせがあります。この組は設計と機能においてはDLLやDLLDELと同様ですが、使い方はDDR実装に制限されます。これ以上の情報に関してはテクニカルノートTN1102 LatticeECP2/M sysIOユーザーズガイドを参照してください。

ダイナミック・クロック選択 (DCS)

DCSは洗練されたマルチプレクサ機能を取り入れたグローバル・クロックバッファです。2系統の独立した入力クロックソースから、いつイネーブル信号がトグルしたかに関わらず、出力クロックに細いパルスが出るのを避けます。DCSは各クオドラントに2つあります。DCS出力はそのあと供給線を通してプライマリ・クロックツリーに達します。図10-27はDCSのブロック図を示します。

図10-27 DCSのプリミティブ・シンボル



DCSライブラリ要素の定義

表10-14はDCSのI/Oポートを定義しています。全部で8モードあり、表10-15でその構成を示します。

表10-14 DCS I/O定義

I/O	名称	説 明
入力	SEL	入力クロック選択
	CLK0	クロック入力0
	CLK1	クロック入力1
出力	DCSOUT	クロック出力

表10-15 DCS動作モード

アトリビュート 名	説 明	出力		値
		SEL=0	SEL=1	
DCS MODE	立ち上がりエッジトリガ、ラッチ状態はHigh	CLK0	CLK1	POS
	立ち上がりエッジトリガ、ラッチ状態はLow	CLK0	CLK1	NEG
	SELはHighアクティブ、ディセーブル出力はLow	0	CLK1	HIGH_LOW
	SELはHighアクティブ、ディセーブル出力はHigh	1	CLK1	HIGH_HIGH
	SELはLowアクティブ、ディセーブル出力はLow	CLK0	0	LOW_LOW
	SELはLowアクティブ、ディセーブル出力はHigh	CLK0	1	LOW_HIGH
	CLK0用バッファ	CLK0	CLK0	CLK0
	CLK1用バッファ	CLK1	CLK1	CLK1

DCS タイミング図

各モードで独自の動作をします。クロック出力タイミングは入力クロックとSEL信号のエッジで決定されます。図10-28で各モードのタイミングを示します。

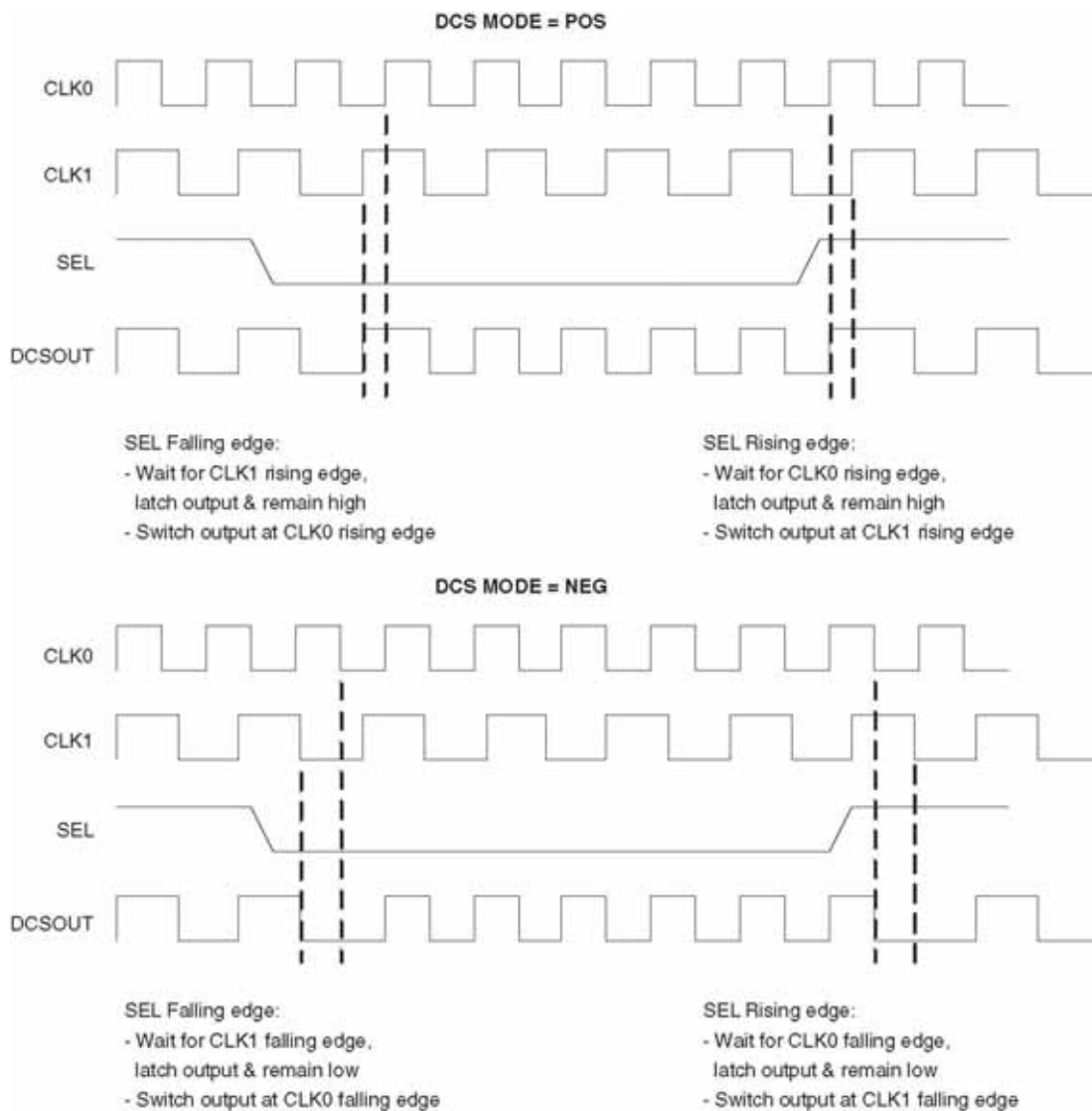
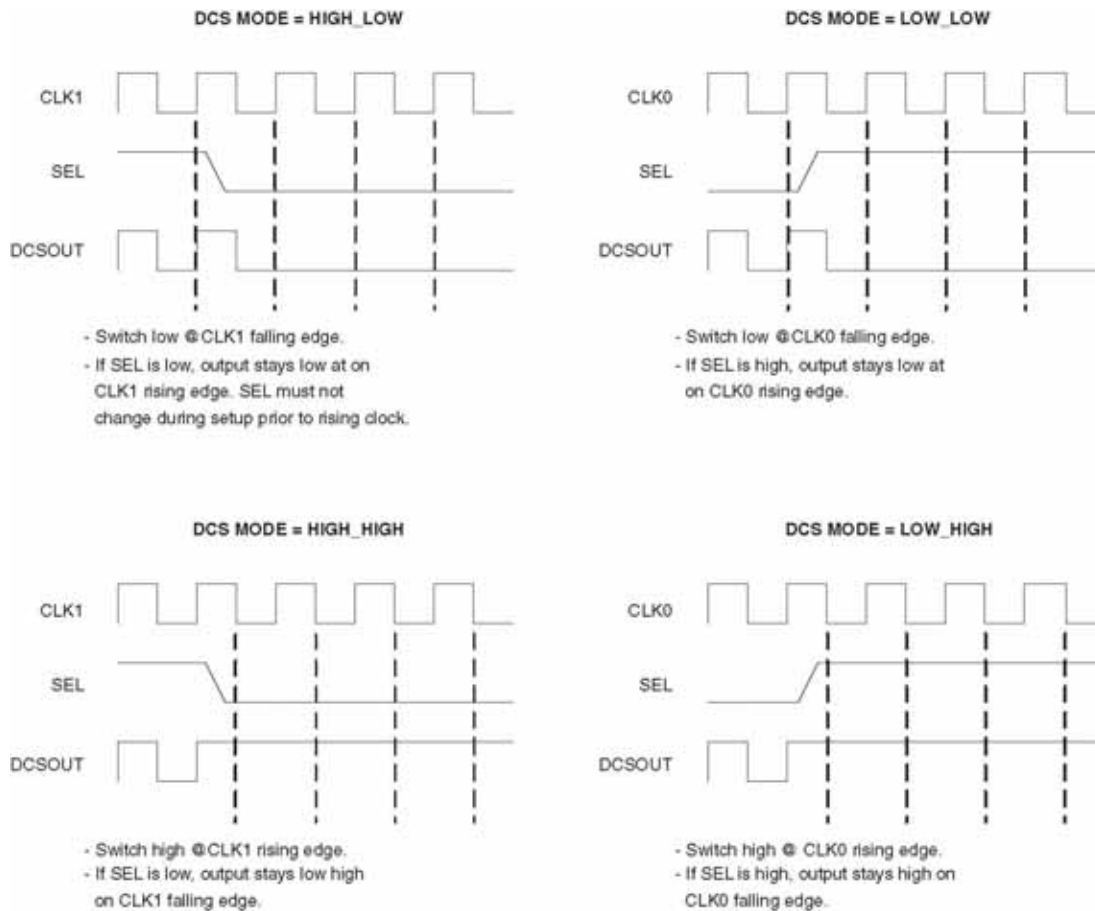
図10-28 DCS MODE ごとのタイミング図

図10-28 DCS MODEごとのタイミング図 (つづき)



VHDLでのDCS使用例

```

COMPONENT DCS
-- synthesis translate_off
  GENERIC (
    DCSMODE : string := "POS"
  );
-- synthesis translate_on

  PORT (
    CLK0 :IN std_logic;
    CLK1 :IN std_logic;
    SEL:IN std_logic;
    DCSOUT :OUT std_logic);
END COMPONENT;

attribute DCSMODE : string;
attribute DCSMODE of DCSinst0 : label is "POS";

begin
DCSinst0: DCS
-- synthesis translate_off
  GENERIC MAP (
    DCSMODE => "POS"
  )
-- synthesis translate_on

```

```

PORT MAP (
  SEL => clkssel,
  CLK0 => dcsclk0,
  CLK1 => sysclk1,
  DCSOUT => dcsclk
);
end

```

VerilogでのDCS使用例

```

module dcs(clk0,clk1,sel,dcsout);
  input clk0, clk1, sel;
  output dcsout;

  DCS DCSInst0 (.SEL(sel),.CLK0(clk0),.CLK1(clk1),.DCSOUT(dcsout));
  defparam DCSInst0.DCSMODE = "CLK0";

endmodule

```

発振器(OSCD)

LatticeECP2/Mデバイスには専用の発振器があり、その出力をユーザは利用できます。発振器周波数は2.5M～130MHzの範囲でプログラマブルです。また、クロックツリーへの入力クロックとして発振器の出力を配線することができます。必要であれば、より低い周波数とするために、発振器出力を内部ロジック(ユーザロジック)で分周することができます。発振器が使用されない場合は、不動作になります。この発振器出力は精度の良いクロックではありません。それは正確なクロックを必要としない、付加的なクロックとしての使用を意図されています。

表10-16 OSCDポート定義

I/O	名 前	記 述
Output	OSC	発振器のクロック出力

表10-17 OSCDアトリビュート定義

ユーザ・アトリビュート	アトリビュート名	値 (MHz)	デフォルト
Nominal Frequency	NOM_FREQ	2.5, 4.3, 5.4, 6.9, 8.1, 9.2, 10.0, 13.0, 15.0, 20.0, 26.0, 30.0, 34.0, 41.0, 45.0, 55.0, 60.0, 130.0	2.5

仕様についてはデータシートを参照してください。

OSCライブラリ・シンボル(OSCD)

図10-29 OSCシンボル



VHDLでのOSCの使い方

```

COMPONENT OSCD
-- synthesis translate_off

```

```

    GENERIC(NOM_FREQ: string);
-- synthesis translate_on
    PORT (CFGCLK:OUT std_logic);
    END COMPONENT;

attribute NOM_FREQ : string;
attribute NOM_FREQ of OSCins0 : label is "2.5";

begin

OSCInst0: OSCD
-- synthesis translate_off
    GENERIC MAP (NOM_FREQ => "2.5")
-- synthesis translate_on
    PORT MAP ( CFGCLK=> osc_int);
end

```

VerilogでのOSCの使い方

```

module OSC_TOP(OSC_CLK);
    output OSC_CLK;

    OSCD OSCinst0 (.CFGCLK(OSC_CLK));

    defparam OSCinst0.NOM_FREQ = "2.5";

endmodule

```

クロック入力の共有

図10-30と図10-31に示されるように、パッドからの基準クロックをPLLとDLLで共有することができます。1つのクロックソースだけが複数のPLL/DLLに利用できるとき、この機能は役に立ちます。

図10-30 入力クロックの共有 (LatticeECP2)

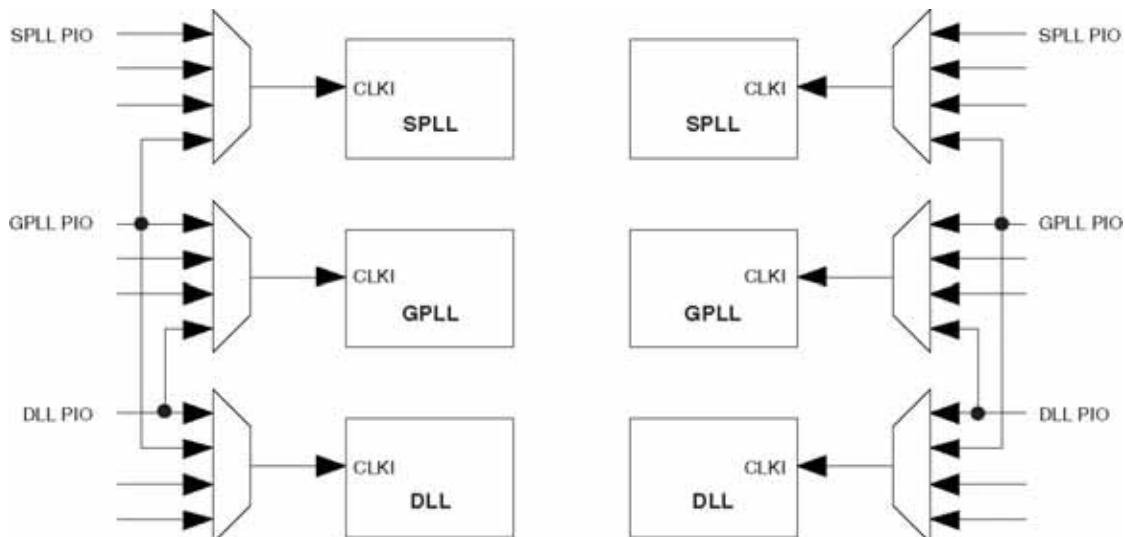
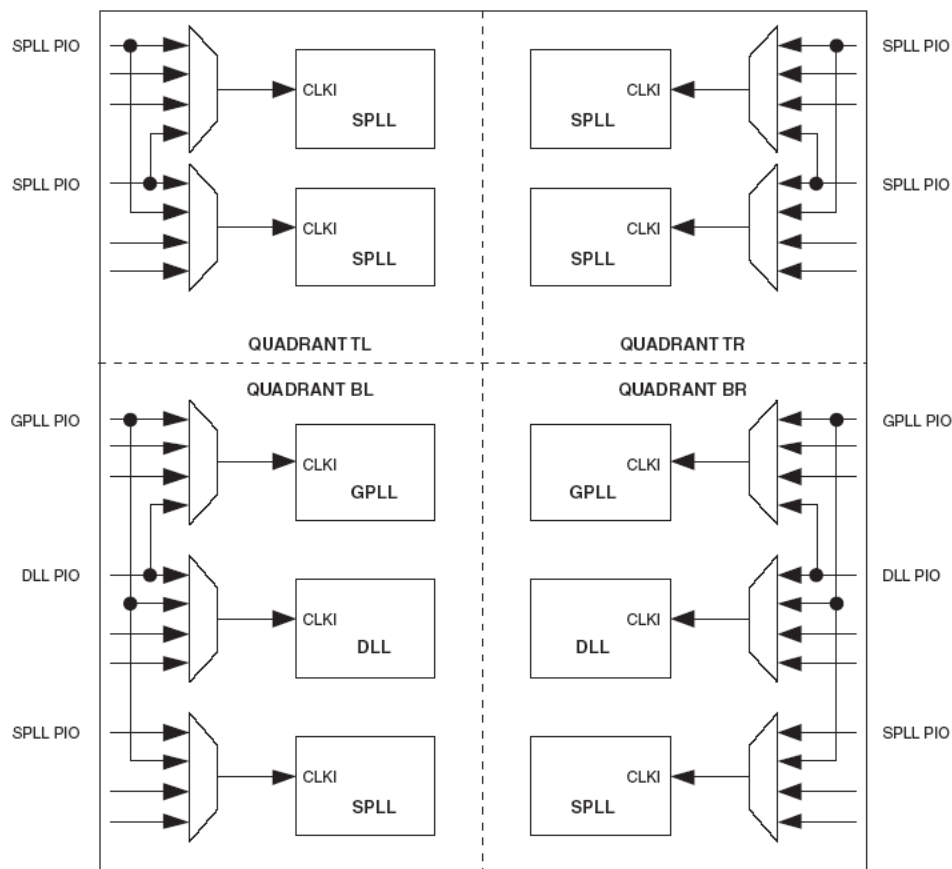


図10-31 入力クロックの共有(SPLLが6つあるLatticeECP2M)



クロック・プリファレンス（制約）の設定

クロック・プリファレンスで、設計者は必要な性能にクロックを実装することができます。ispLEVER中のプリマップ・プリファレンスエディタ、又はプリファレンス・ファイルでプリファレンスを設定することができます。頻繁に使用されるプリファレンスは付録Cで説明されています。追加情報はispLEVERのオンラインヘルプ・システムを参照してください。

供給電源

各PLLにはそれ自身の供給電源があります。ECP2-6, ECP2-12, ECP2-20の各デバイスではPLL電源はVCCと共通になっています。従ってこれらにはVCCピンのみがあります。

これ以外のECP2と全ECP2MデバイスにはPLL専用の供給電源ピンV_{CCPLL}があります。V_{CC}とV_{CCPLL}は通常同じ1.2Vであるので、同一回路基板では同じ電源から供給されることを推奨します。その結果リークを最小にできます。さらに、それぞれの電源はそれらの間のノイズ結合を最小にするために、フィルタ技法を用いることで、ボード上の1.2V主電源から適切に分離されるべきです。

DLLはFPGAコア供給電源から電源を得ます。

テクニカル・サポート支援

ホットライン: 1-800-LATTICE (North America)

+1-503-268-8001 (Outside North America)

e-mail: techsupport@latticesemi.com

インターネット: <http://www.latticesemi.com>

変更履歴（日本語版）

Rev.#	日付	変更箇所
1.5J	Aug.2008-	既発行
		英語版 1.6 (Aug.2008)に対応し、1.5Jを更新し、全面的に文章・用語を校正。以下はその差分
		1, Secondary clock (p.10-3); "quadrant" --> "region" 2, 図10-5 を更新 3, 図10-8 と図10-9; tSU --> tRSTREC 4, 表10-3; DPAMODE の記述追加 5, LatticeECP2/M PLL Modules (p.10-10)と図10-10; 節と図を新規追加 6, 図10-11. LatticeECP2/M PLL Library Symbols (p.10-11); 図を更新 7, Optional External Capacitor (p.10-13); 1MHz --> 2MHz 8, 図10-14/-15; 図を更新、図10-15下の注記を更新 9, 表10-7; 1, CLKOK2 行削除、2, 脚注 4 更新 10, 表10-8; 脚注 1. 追加 11, UDDCNTL Input (p.10-20); DCNTL --> DCNTL[8:0] 12, Duty Cycle Selection (p.10-20); 節を削除 13, DLL Lock Time Control (p.10-20); "This option is also available in the IPexpress GUI."削除 14, DLL Library Definitions (p.10-21); "three modes" --> "two modes" 15, CIMDLLA (p.10-22); 節と図を削除
1.6J0	Jan.5, 2008	<< 共通（複数） >> 16, CIMDLLA を削除 ・ DLL Overview (p.10-19) ・ 表10-10 内の ALUHOLD 記述 (p.10-21) ・ Configuration Tab - Usage Mode (p.10-23) ・ Appendix C. Clock Preferences (p.10-42) - ASIC "my_dll" TYPE "CIMDLLA" --> "TRDLLA" 17, 用語の変更; "Primitive (element)" --> "Library (element)" ・ 図10-19, 表10-9/-10, DLL各節のタイトルなど (p.10-21) ・ CLKDIV (p.10-24) ・ DCS (pp.10-29:30) ・ OSC Library Symbol (OSCD) (p.10-34) 18, VHDL 記述例の最後に "end" を追加修正 ・ CLKDIV Declaration in VHDL Source Code (pp.10-23:24) ・ DLLDELA Declaration in VHDL Source Code (pp.10-27:28) ・ DCS Usage with VHDL - Example (p.10-33) ・ OSC Usage with VHDL - Example (p.10-34)

付録A プライマリクロック・ソースと分配

図10-32 LatticeECP2プライマリクロック・ソースと分配

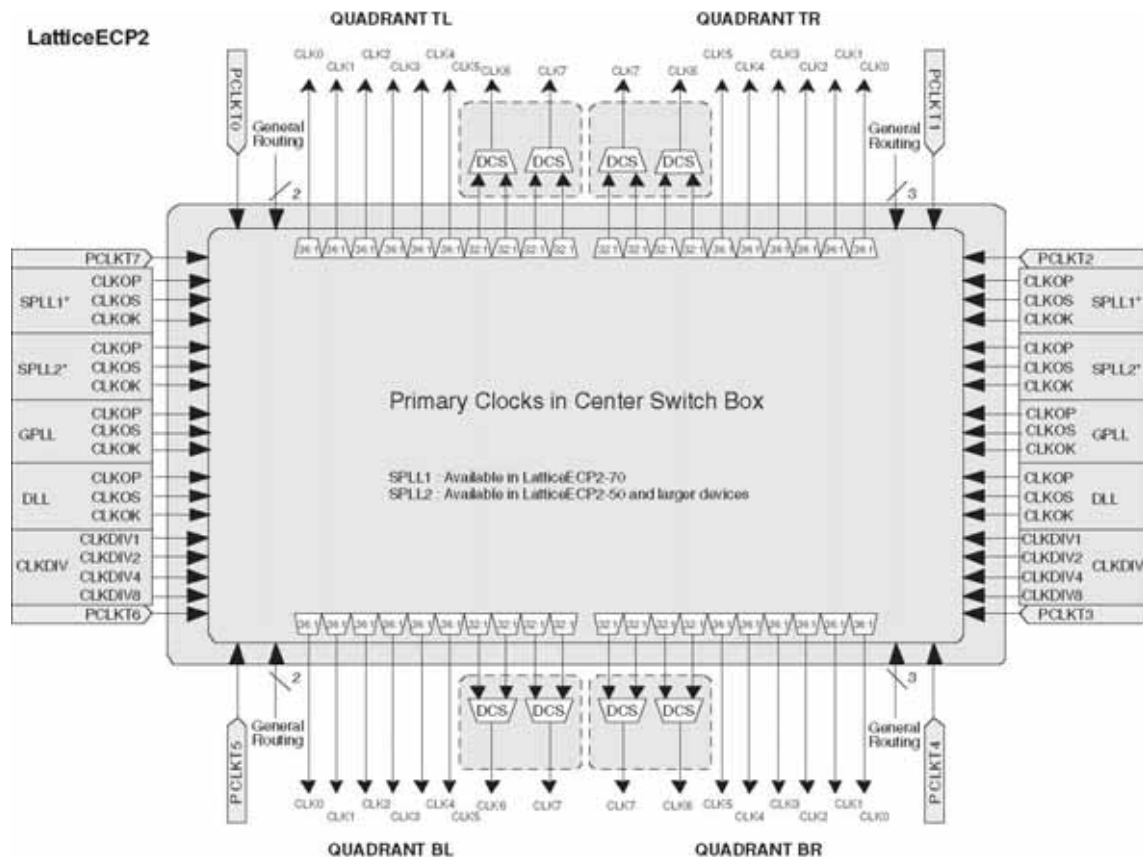


図10-33 LatticeECP2プライマリクロック・マルチプレクサ

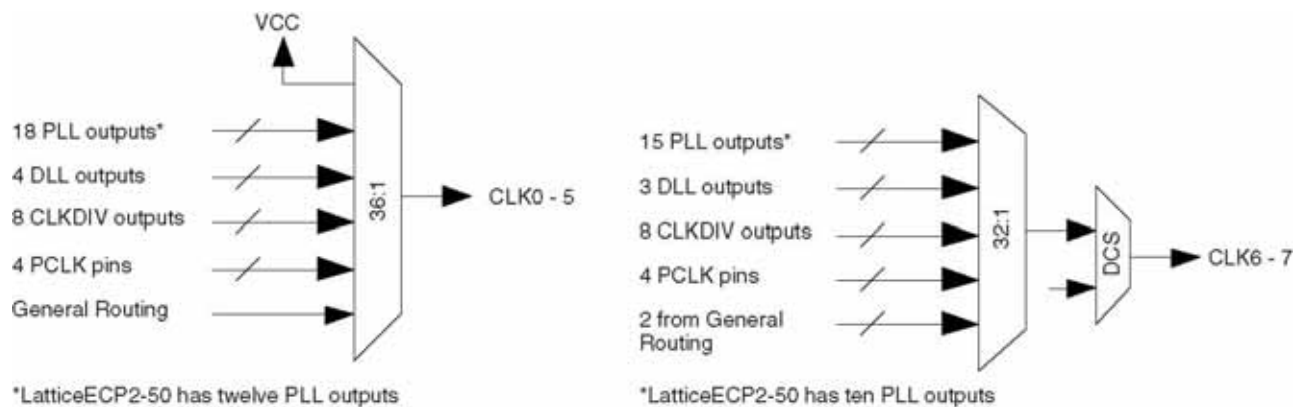


図10-34 LatticeECP2M プライマリクロック・ソースと分配

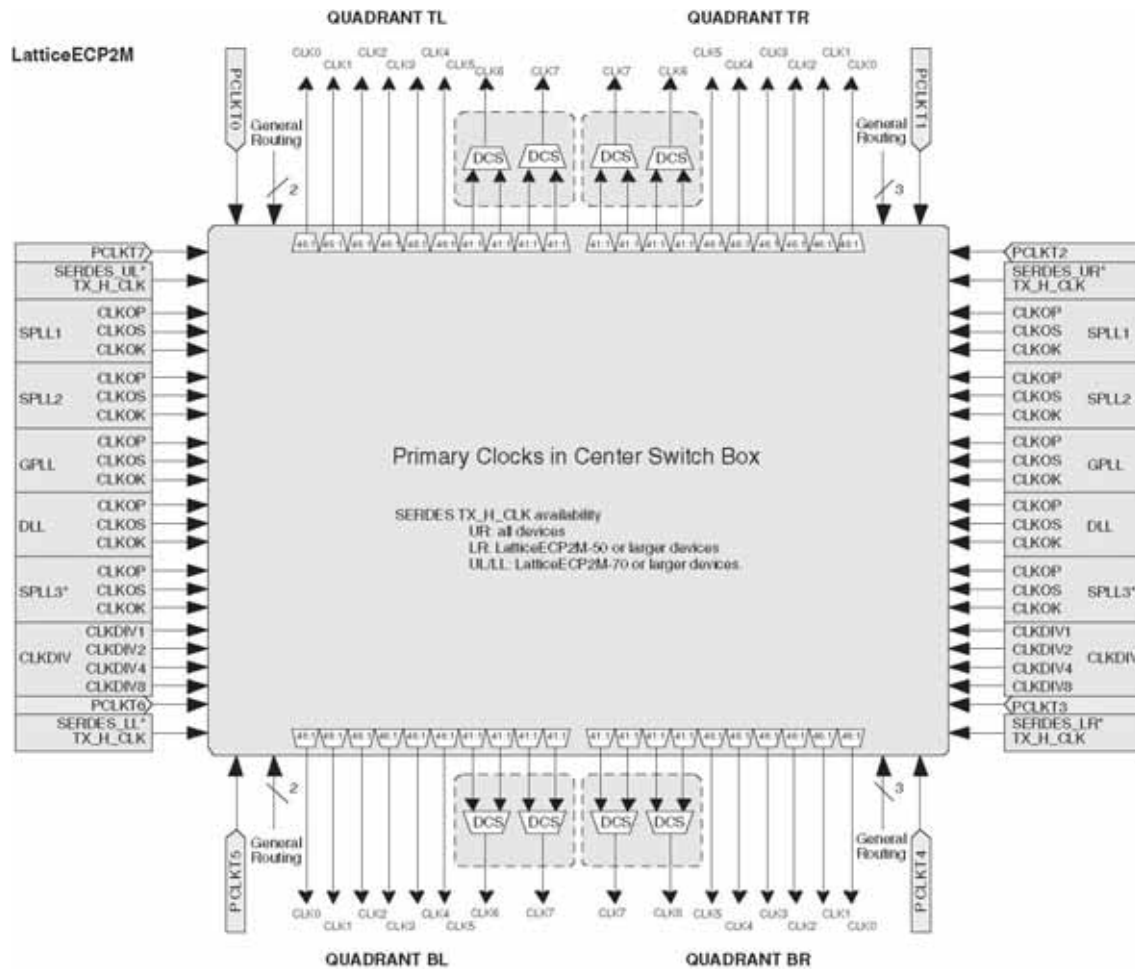
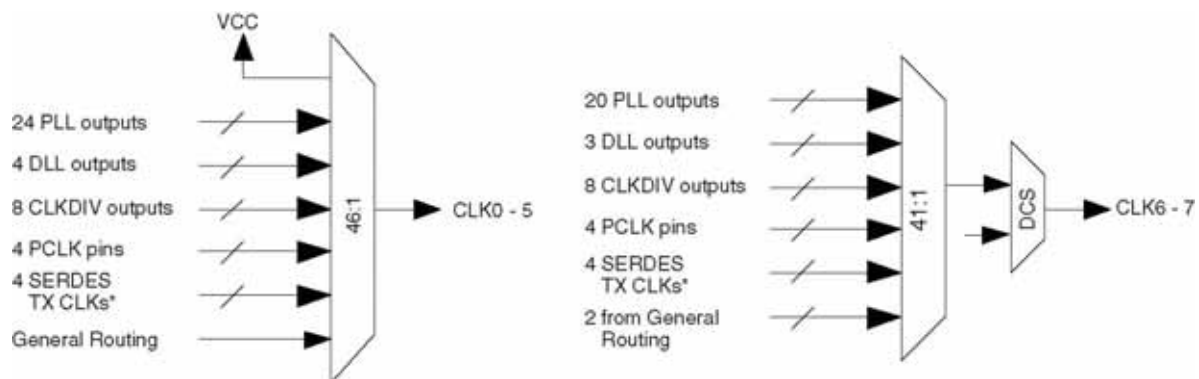


図10-35 LatticeECP2M プライマリクロック・マルチプレクサ

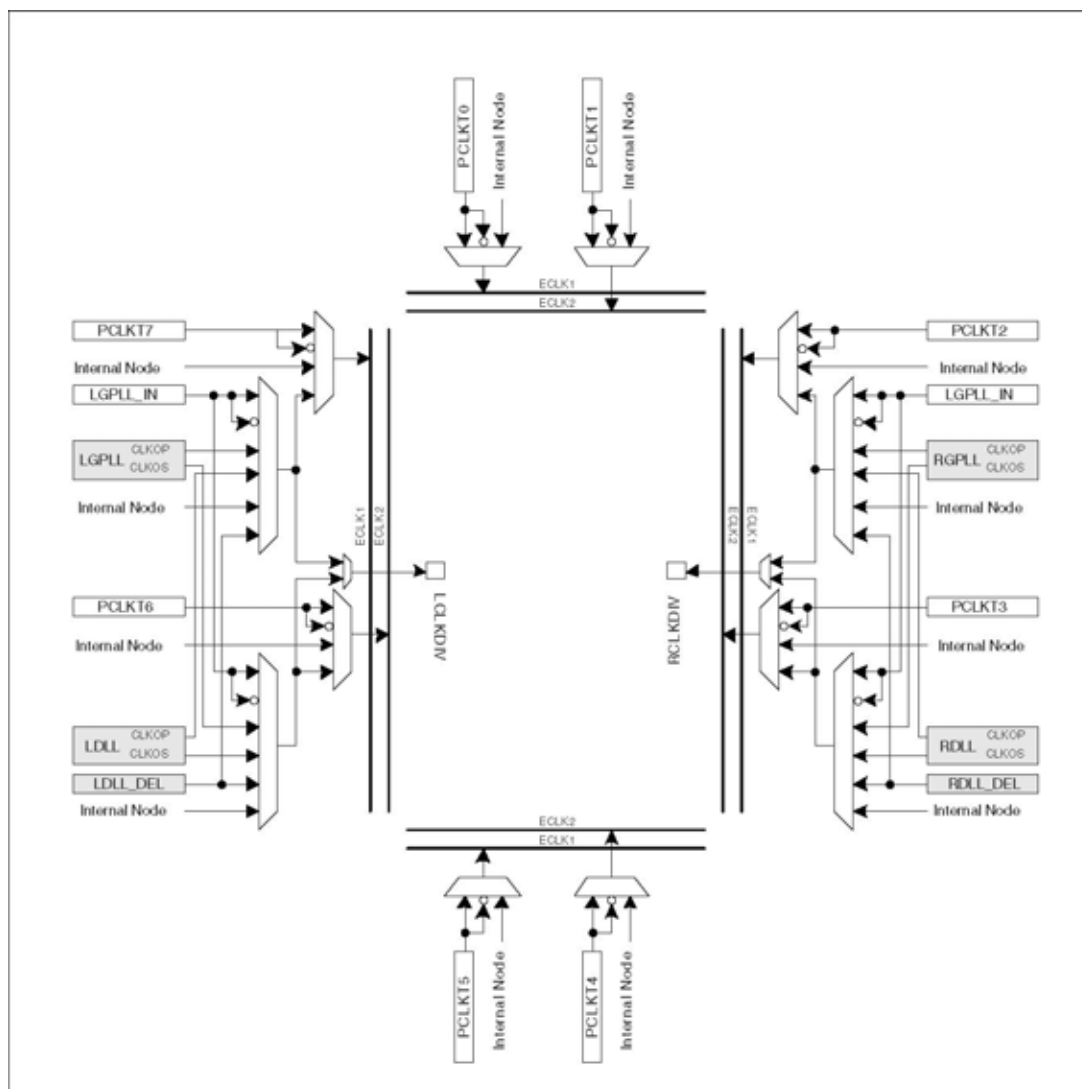


*LatticeECP2M-50 has two SERDES TX CLK outputs

付録B PLL、DLL、CLKDIV、ECLKの位置と接続性

図10-36はPLL、DLLs、CLKDIV、およびECLKの位置、サイト名、および接続性を示します。

図10-36 PLL、DLL、CLKDIV、ECLKの位置、および接続性



付録C クロック制約

幾つかの主な制約を以下に紹介します。より詳細の情報とその他の制約についてはソフトウェアのヘルプを参照してください。

ASIC

次の制約はCIMDLLA CLKOPに90°の位相を指定します。

```
ASIC "my_dll" TYPE "TRDLLA" CLKOP_PHASE=90;
```

FREQUENCY

次の物理制約はclk1というネットに100 MHzを指定します。

```
FREQUENCY NET "clk1" 100 MHz;
```

次の制約は各クロックドメインにホールドマージンを指定します。

```
FREQUENCY NET "RX_CLKA_CMOS_c" 100.000 MHz HOLD_MARGIN 1 ns;
```

MAXSKEW

次の制約はNetBというネットに5 nsの最大スキューを指定します。

```
MAXSKEW NET "NetB" 5 NS;
```

MULTICYCLE

次の制約はCOMPAからCOMPB (NET1)のパスに対して50 nsの周期を指定します。

```
MULTICYCLE "PATH1" START COMP "COMPA" END COMP "COMPB" NET "NET1" 50 NS ;
```

PERIOD

次の制約はClk1というポートに30 nsのクロック周期を指定します。

```
PERIOD PORT "Clk1" 30 NS;
```

PROHIBIT

本制約はbf_clkというクロックをプライマリクロックに割り当てることを禁止します。

```
PROHIBIT PRIMARY NET "bf_clk";
```

USE PRIMARY

指定されたネットを配線するのにプライマリクロック・リソースを用います。

```
USE PRIMARY NET clk_fast;  
USE PRIMARY DCS NET "bf_clk";  
USE PRIMARY PURE NET "bf_clk" QUADRANT_TL;
```

USE SECONDARY

指定されたネットを配線するのにセカンダリクロック・リソースを用います。

USE SECONDARY NET "clk_lessfast" QUADRANT_TL;

USE EDGE

指定されたネットを配線するのにエッジクロック・リソースを用います。

USE EDGE NET "clk_fast";

CLOCK_TO_OUT

本制約はクロックに対する出力の許容最大遅延値を指定します。

以下の2つの制約はCLKPORTとCLKNETというキーワードを用いた例で、それぞれに対するタイミング解析 (TRACE) レポートも示します。

CLKNET制約はPLLの前のパスについて解析を行いません。従ってPLLタイミング補償値を得ることができません。

CLOCK_TO_OUT PORT "RxAddr_0" 6.000000 ns CLKNET "pll_rxclk";

これにより、以下のクロックパス解析結果を得ます。

Physical Path Details:

Clock path pll_inst/pll_utp_0_0 to PFU_33:

Name	Fanout	Delay (ns)	Site	Resource
ROUTE	49	2.892	ULPPLL.MCLK to R3C14.CLK0	pll_rxclk

2.892 (0.0% logic, 100.0% route), 0 logic levels.

CLKPORT制約はクロックポート・リソースまでの解析を完了し、PLLタイミング補償値を得ます。

CLOCK_TO_OUT PORT "RxAddr_0" 6.000000 ns CLKPORT "RxClk";

これにより、以下のクロックパス解析結果を得ます。

Clock path RxClk to PFU_33:

Name Fanout Delay (ns) Site Resource

IN_DEL	---	1.431	D5.PAD to D5.INCK	RxClk
ROUTE	1	0.843	D5.INCK to ULPPLL.CLKIN	RxClk_c
MCLK_DEL	---	3.605	ULPPLL.CLKIN to ULPPLL.MCLK	pll_inst/pll_utp_0_0
ROUTE	49	2.892	ULPPLL.MCLK to R3C14.CLK0	pll_rxclk

8.771 (57.4% logic, 42.6% route), 2 logic levels.

INPUT_SETUP

次の制約はクロックに対する入力のセットアップ値要件を指定します。

```
INPUT_SETUP PORT "datain" 2.000000 ns HOLD 1.000000 ns CLKPORT "clk" PLL_PHASE_BACK ;
```

PLL_PHASE_BACK

次の制約は直前のクロックエッジに対するタイミング解析をしたい場合INPUT_SETUPと共に用います。この制約はPLL出力を位相調整なしに使用する際に有用です。これは負の位相調整機能はないため、PLL_PHASE_BACKがあたかもその機能として作用するためです。

例えば、CLKOSに -90° のシフトが求められる場合、ユーザは位相を 270° にし、PLL_PHASE_BACKとINPUT_SETUPを共に用います。

プリマップ・プリファレンスエディタにおけるPLL_PHASE_BACKの使い方:

PLL_PHASE_BACKアトリビュートを設定するのにプリマップ・プリファレンスエディタを用いることができます。

1. デザインプランナ(Pre-Map)を開きます。
2. デザインプランナ・コントロール・ウィンドウの**View**の下で、**Spreadsheet View**を選択します。
3. Spreadsheet Viewウィンドウで、**Input_setup/Clock_to_out**を選択してマウスでクリックします。
4. INPUT_SETUP/CLOCK_TO_OUTプリファレンス・ウィンドウを図10-37に示します。

図10-37 INPUT_SETUP/CLOCK_TO_OUTプリファレンス・ウィンドウ

INPUT_SETUP is a logical preference. It specifies a setup time requirement for input ports relative to a clock net.

Type:

- ☒ INPUT_SETUP
- ☐ CLOCK_TO_OUT

Second Type:

- ☒ All Ports
- ☐ Individual Ports
- ☐ Group

Available Input Groups:

Available Clock Ports/Nets:

- clk
- clk_62p5
- clk_125
- clk_c
- pll/CLKOP

Clock Out Ports:

Filter: Go

Time: ns

Hold Time: ns

☐ Input Delay

☒ PLL Phase Back

Existing Preference:

Type	Object Ty...	Object Na...	Clock	Delay	Time	Phase Ba...	In/Out Del...	Clock Out
INPUT_SET...	PORT	datain	CLKPORT clk	2.000000 ns	1.000000 ns	ON	Not Input D...	
INPUT_SET...	ALLPORTS		CLKPORT clk	2.000000 ns	1.000000 ns	ON	Not Input D...	

Add Delete Modify Undo/Modify OK Cancel Help

