XAPP174 (v1.1) 2000 年 1 月 24 日

Spartan-II FPGA におけるディレイ ロック ループの使用

概要

Spartan™-II ファミリには完全にデジタル化された専用のオンチップ DLL (ディレイ ロック ループ) 回路が 4 つ含まれており、ゼロ伝搬遅延、デバイスに分散している出力クロック信号間の低クロック スキュー、高度なクロック ドメイン制御を可能にします。これらの専用 DLL を使用すると、システム レベルのデザインを改良および簡素化する回路をインプリメントできます。

はじめに

デバイス間で高い帯域のデータ レートをサポートするには、高度なクロック管理技術が必要となります。クロック遅延とクロック スキューは、デバイスのパフォーマンスに大きく影響します。Spartan-II FPGA の DLL (ディレイ ロック ループ) 回路は、ゼロ伝搬遅延と、デバイスに分散している出力クロック信号間の低クロック スキューを提供します。各 Spartan-II デバイスには 4 つの完全にデジタル化された専用オンチップ DLL が含まれているので、外部クロックと内部クロックを非常に正確に同期させることができます。

各 DLL は、デバイス内で最大 2 つのグローバル クロック配線ネットワークを駆動できます。グローバル クロック分散ネットワークは、負荷の差異によるクロック スキューを最小限にします。DLL は DLL 出力クロックのサンプルを監視することによって配線ネットワークの遅延を補正するので、外部入力ポートからデバイス内部のクロック負荷への遅延を事実上なくすことができます。

ユーザー ソース クロックに関してゼロ遅延を提供することに加え、DLL は複数位相のソース クロックを提供できます。また、クロックを 2 倍にすることや、ユーザー ソース クロックを最大 16 分周することもできます。

クロックの増倍が可能になると、デザインの選択肢が広がります。たとえば、50MHz のソース クロックを DLL で 2 倍にすると、100MHz で動作する FPGA デザインを駆動できます。この手法を利用すると、ボード上のクロック パスに 100MHz という高速な信号が分散しなくなるので、ボード デザインを簡素化できます。また、タイム ドメインを多重化すると、各クロック サイクルで 1 つの回路を 2 回使用して、同じ回路を 2 つ使用する場合よりもボード エリアを小さくすることもできます。

DLL は、クロック ミラーとしても機能します。DLL の出力をチップ外に出してから戻すと、複数のデバイス間でボード レベルのクロックに関するスキューを短縮できます。

デバイスが起動する前にシステム クロックが確立することを保証するため、DLL はロックを確立するまでデバイスのコンフィギュレーション プロセスの終了を遅らせることができます。

DLL を利用してオンチップのクロック遅延を削減すると、高いファンアウトの高速クロックに関してシステム レベルのデザインを著しく簡素化し、性能を改善できます。

基礎知識

クロックの遅延を削減する回路には、PLL (フェーズ ロック ループ) と DLL の 2 種類があります。クロック分散遅延を削減するという基本的な機能に加えて、DLL と PLL には周波数合成 (クロックの増倍と分周) やクロック調節 (デューティ サイクルの補正と位相シフト) などの追加機能もあります。また、複数のクロック出力間でスキューを短縮することもできるので、複数のクロック ドメインを利用できます。

DLL (ディレイ ロック ループ)

図 1 に示すように、最も単純化された DLL は、可変の遅延線と制御ロジックから構成されます。遅延線は、入力クロック CLKIN を遅延させたクロックを生成します。クロックは、クロック分散ネットワークによってすべての内部レジスタとクロック フィードバック CLKFB ピンに送られます。制御ロジックは、入力クロックとフィードバック クロックをサンプリングして、遅延線を調整します。

遅延線は、電圧制御遅延または複数の独立した遅延素子を使用して作成できます。パフォーマンスを最高にするため、ザイリンクスの DLL では独立したデジタル遅延線を使用しています。

DLL が動作すると、入力クロックとフィードバック クロックの位相差が 360° になり立ち上がりエッジが一致するまで、2 つのクロックの間に遅延が挿入されます。入力クロック ラインのエッジとフィードバック クロックのエッジが一致すると、DLL はロックします。DLL がロックされると、回路を測定しない限り、2 つのクロックの違いは識別できません。このように DLL の出力クロックがクロック分散ネットワークの遅延を補正するので、ソース クロックと負荷の間で遅延が削減されます。

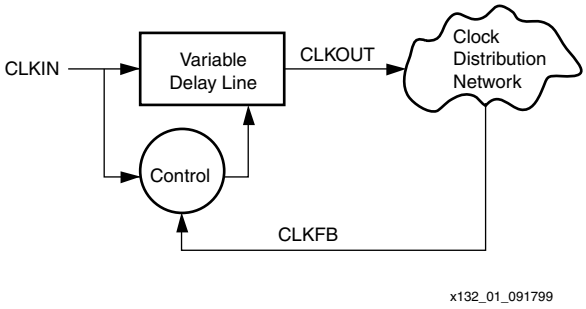

図 1: DLL のブロック図

PLL (ディレイ ロック ループ)

PLL の基本的な機能は DLL と同じですが、タスクの実現には異なるアーキテクチャを使用します。図 2 に示すように、PLL と DLL の基本的な違いは、PLL では入力クロック CLKIN に似たクロック信号を生成する電圧制御オシレータを、遅延線の代わりに使用するということです。位相検出回路とフィルタから構成される制御ロジックが、オシレータの周波数と位相を調整してクロック分散遅延を補正します。

PLL の制御ロジックは、入力クロックとフィードバック クロック CLKFB を比較し、入力クロックの立ち上がりエッジとフィードバック クロックの立ち上がりエッジが一致するまでオシレータのクロックを調整します。そして、PLL がロックします。

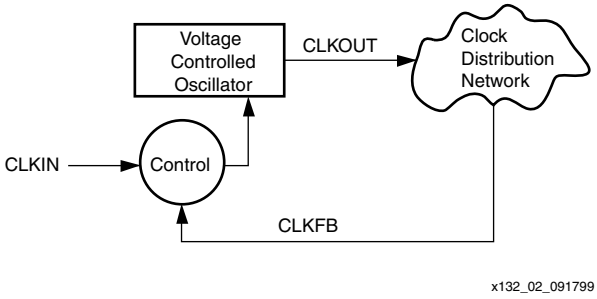

図 2: PLL のブロック図

インプリメンテーション

DLL と PLL は、アナログ回路またはデジタル回路を使用してインプリメントできます。これらには、それぞれの利点があります。慎重にデザインされたアナログによるインプリメンテーションでは、タイミング分解性能が優れた DLL または PLL を作成できます。また、シリコン エリアを少なくすることもできます。

これに対して、デジタルによるインプリメンテーションでは、ノイズ感度、低電力消費、優れたジッター パフォーマンスという利点が得られます。また、クロックを停止してパワー マネージメントを促進することもできます。アナログによるインプリメンテーションでは、追加電源と電源のきめ細かい制御が必要となり、新しいプロセス技術への移行で問題が生じる可能性があります。

DLL と PLL の比較

特定のアプリケーションについて PLL と DLL のどちらかを選択するには、アーキテクチャの違いを理解する必要があります。PLL で使用するオシレータには不安定な性質があり、位相エラーが蓄積します。このため、クロック分散ネットワークの遅延を補正する場合は、PLL のパフォーマンスが低下します。逆に、DLL は無条件に安定しているので、位相エラーは蓄積しません。

このため、遅延の補正とクロックの調節には、アーキテクチャとして DLL を選択します。

DLL のライブラリ シンボル

図 3 に、簡素化したザイリンクスの DLL マクロのライブラリ シンボル、BUFGDLL を示します。このマクロを使用すると、デバイス内における伝搬遅延がゼロのシステム クロックがすばやく効率的に得られます。図 4 と図 5 に、2 種類の DLL のライブラリ プリミティブを示します。これらのシンボルを使用すると、より複雑なアプリケーションをインプリメントする場合に DLL の完全な機能を使用できます。

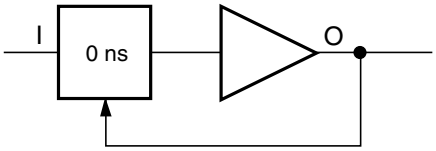

図 3: 簡素化した DLL マクロのシンボル BUFGDLL

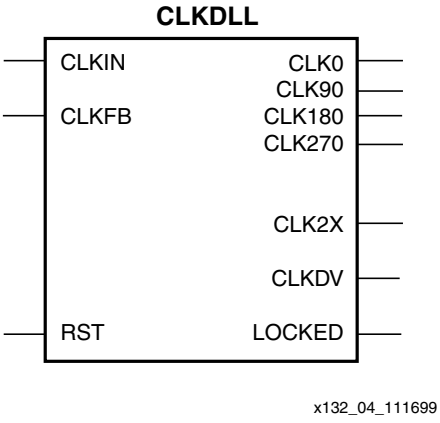

図 4: 標準的な DLL シンボル CLKDLL

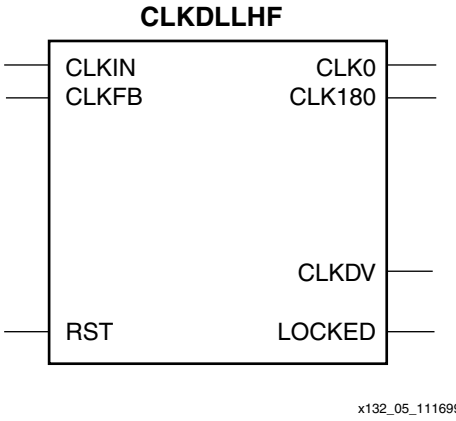

図 5: 高周波 DLL シンボル CLKDLLHF

BUFGDLL マクロのピン

BUFGDLL マクロを使用すると、外部入力による高いファンアウトのオンチップ クロックについてゼロ伝搬遅延を簡単に実現できます。このマクロでは、IBUFG、CLKDLL、BUFG の各プリミティブを使用して、図 6 に示すような最も基本的な DLL アプリケーションをインプリメントしています。

IBUFG は、DLL の専用クロック入力ピンです。このピンは各 DLL ごとに 1 本ずつあり、全部で 4 本あります。これらは入力専用で、可能な入力電圧規格のいずれかを使用できますが、バンキング ルールに従う必要があります (対応するバンク番号はデータシートに記載されています)。IBUFG には、対応する DLL への専用配線があります。このため、DLL を使用する場合の遅延が最小になり精度が向上します。ただし、DLL の入力は、他のデバイス ピンによっても駆動できます。

BUFG はグローバル クロック バッファで、デバイス全体で高速な低スキュー信号を駆動します。IBUFG、CLKDLL、BUFG は、すべて同じエッジ (上または下) のものを使用する必要があります。

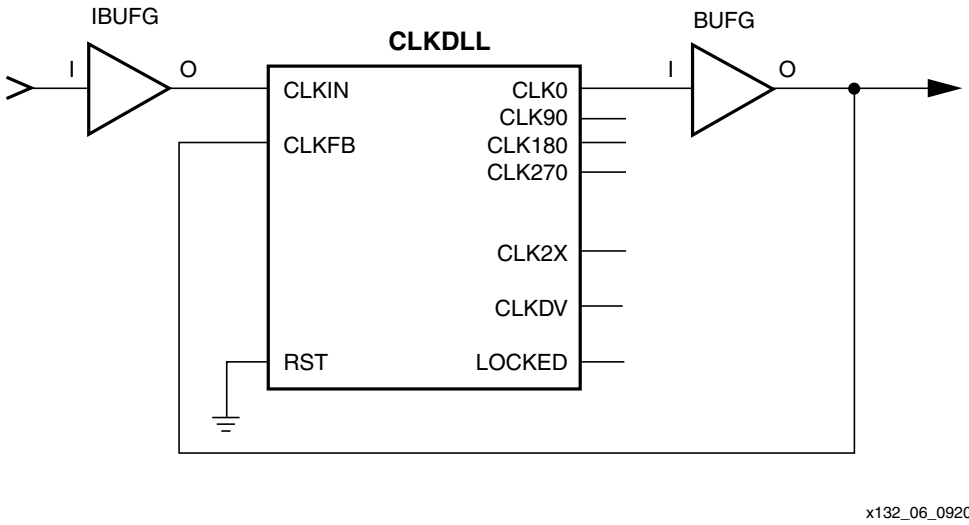

図 6: BUFGDLL の回路図

このシンボルを使用する場合、DLL のクロック増倍および分周機能や高度なクロック ドメイン制御は使用できません。また、DLL の RST ピンと LOCKED ピンも使用できません。これらの機能を使用するには、次のセクションで説明する DLL プリミティブを使用する必要があります。

ソース クロック入力 - I

I ピンは、ユーザー ソース クロック (DLL が動作するためのクロック信号) を BUFGDLL に提供します。BUFGDLLの場合、ソース クロックの周波数はデータシートに記載されている低周波範囲に収める必要があります。BUFGDLL では、外部信号によるソース クロックが必要です。したがって、BUFGDLL の I ピンを駆動する信号は、外部入力ポートのみ供給できます。

クロック出力 - O

クロック出力ピン O からは、遅延を補正したソース クロックが出力されます。この信号はデバイスの専用グローバル クロック配線リソースを利用しており、グローバル クロック バッファ シンボル BUFG によって供給されます。

デューティ サイクル補正プロパティが有効な場合、出力クロックのデューティ サイクルは 50/50 です。

CLKDLL プリミティブのピン

CLKDLL ライブラリ プリミティブは、DLL でより複雑なアプリケーションをインプリメントする場合に必要となるすべての機能を使用できるようにします。

ソース クロック 入力 - CLKIN

CLKIN ピンは、ユーザー ソース クロック (DLL が動作するためのクロック信号) を DLL に提供します。CLKIN の周波数は、データシートに記載されている範囲内に収める必要があります。このクロック信号は、別の CLKDLL が駆動するグローバル クロック バッファ (BUFG) またはデバイス内の同じエッジ (上または下) にあるグローバル クロック入力バッファ (IBUFG) のいずれかによって供給する必要があります。

フィードバック クロック入力 - CLKFB

DLL は、遅延を補正した出力を提供するために、基準となる信号、つまり、フィードバック信号を必要とします。DLL の CLK0 または CLK2X 出力をフィードバック クロック入力 (CLKFB) ピンに接続するだけで、必要なフィードバックが DLL に与えられます。このクロック信号は、グローバル クロック バッファ (BUFG) またはデバイス内の同じエッジ (上または下) にあるグローバル クロック入力バッファ (IBUFG) のいずれかによって供給する必要があります。

IBUFG から CLKFB ピンにクロックを供給する場合は、次のような特別な規則が適用されます。

1. IBUFG の I ピンを駆動する信号は、外部入力ポートから供給する必要があります。
2. CLK0 出力と CLK2X 出力の両方がオフチップのデバイスを駆動している場合は、CLK2X 出力をフィードバックする必要があります。
3. この信号は OBUF を直接駆動する必要があります。他のものを駆動してはいけません。

これらの規則は、CLKFB ピンのソースとなる DLL クロック出力をソフトウェアが決定できるようにするためのものです。

リセット入力 - RST

リセット ピン RST がアクティブになると、4 ソース クロック サイクル以内に LOCKED 信号がアクティブでなくなります。RST ピンはアクティブ High です。ダイナミック ピンまたはグランドに接続する必要があります。DLL の遅延タップがゼロにリセットされると、DLL のクロック出力ピンでグリッチが発生する可能性があります。また、RST ピンをアクティブにすると、クロック出力ピンのデューティ サイクルにも重大な影響があります。さらに、DLL の出力クロックと他のクロックの間でスキューが短縮されなくなります。これらの理由から、デバイスを再コンフィギュレーションする場合と入力周波数を変更する場合を除くと、リセットピンを使用することはほとんどありません。

2 倍クロック出力 - CLK2X

出力ピン CLK2X は、周波数が 2 倍のクロックを提供します。デューティ サイクルは、自動的に 50/50 に補正されます。CLKDLL がロックするまで、CLK2X 出力からはデューティ サイクルが 25/75 で入力クロックと同じ周波数のクロックが出力されます。これは、DLL がソース クロックに関して正しいエッジでロックできるようにするためです。このピンは、CLKDLLHF プリミティブでは使用できません。

クロック分周出力 - CLKDV

クロック分周出力ピン CLKDV は、ソース クロックより低い周波数のクロックを提供します。CLKDV を制御するには、CLKDV_DIVIDE プロパティを使用して、ソース クロックを何分周するか指定します。指定できる値は、1.5、2、2.5、3、4、5、8、16 です。

デューティ サイクルが自動的に補正されるので、CLKDV からは常にデューティ サイクルが 50/50 のクロックが出力されます。

1 倍クロック出力 - CLK[0|90|180|270]

1 倍クロック出力ピン CLK0 は、ソース クロック信号 (CLKIN) の遅延を補正したクロックを提供します。CLKDLL プリミティブには、CLK0 信号の位相をシフトさせたクロックが 3 種類あります。CLKDLLHF の場合は、位相が 180° シフトしたものしかありません。位相のシフトと対応する周期のシフトの関係については、表 1 を参照してください。

表 1: 位相シフト出力クロックと周期のシフトの関係

位相 (°)	周期がシフトする割合
0	0%
90	25%
180	50%
270	75%

図 7 のタイミング図に、DLL のクロック出力の特性を示します。

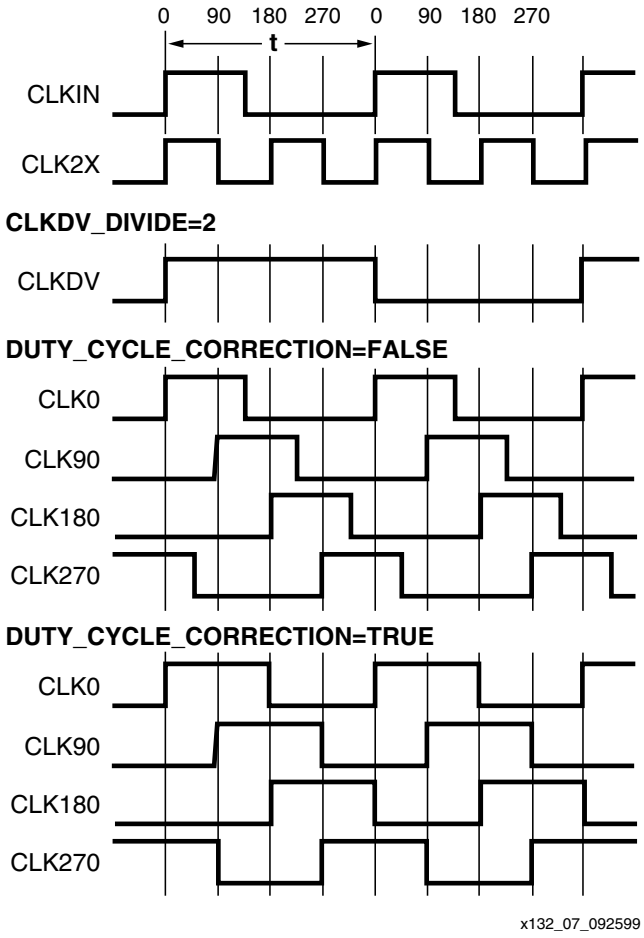

図 7: DLL の出力特性

DLL から出力されるすべての 1 倍クロックは、デフォルトで 50/50 になるようにデューティ サイクルが補正されます。この機能を制御するには、DUTY_CYCLE_CORRECTION プロパティを使用します (デフォルトは TRUE)。DLL のデューティ サイクル補正を無効にするには、DLL シンボルに対して DUTY_CYCLE_CORRECTION=FALSE というプロパティを指定します。デューティ サイクルが補正されない場合、出力クロックのデューティ サイクルはソース クロックと同じになります。

DLL のクロック出力は、OBUF と BUFG を駆動できます。デスティネーション クロック ピンに直接配線することもできます。DLL のクロック出力は、同じエッジ (上または下) にある BUFG のみ駆動できます。

ロック出力 - LOCKED

DLL がロックを確立するには、数千サイクルのクロックをサンプリングすることが必要な場合があります。DLL がロックを確立すると、LOCKED 信号がアクティブになります。ロックに必要な時間の予測値は、データシートの DLL のタイミング パラメータに関するセクションに記載されています。

デバイスが起動する前にシステム クロックが確立することを保証するため、DLL はロックを確立するまでデバイスのコンフィギュレーション プロセスの終了を遅らせることができます。この機能を有効にするには、STARTUP_WAIT プロパティを使用します。

LOCKED 信号がアクティブになるまで DLL の出力クロックは無効で、グリッチやスパイクなどが発生する場合があります。特に CLK2X は、デューティ サイクルが 25/75 の 1 倍クロックとして出力されます。

DLL のプロパティ

Spartan-II ファミリの DLL 機能 (クロック分周やデューティ サイクル補正など) を設定するには、プロパティを使用します。

デューティ サイクル補正プロパティ

1 倍クロック出力の CLK0、CLK90、CLK180、CLK270 は、デフォルトでデューティ サイクルが補正され、50/50 になります。この機能を制御するには、DUTY_CYCLE_CORRECTION プロパティを使用します (デフォルトは TRUE)。1 倍クロック出力について DLL のデューティ サイクル補正を無効にするには、DLL シンボルに対して DUTY_CYCLE_CORRECTION=FALSE というプロパティを指定します。デューティ サイクルを無効にすると、出力クロックのデューティ サイクルはソース クロックと同じになります。

クロック分周プロパティ

CLKDV_DIVIDE プロパティでは、CLKDV ピンの信号を CLK0 ピンに対して何分周するか指定します。このプロパティで指定できる値は、1.5、2、2.5、3、4、5、8、16 です (デフォルトは 2)。

スタートアップ遅延プロパティ

STARTUP_WAIT プロパティの値は TRUE または FALSE です (デフォルトは FALSE)。TRUE を指定すると、DLL がロックされるまで DONE 信号が High になりません。

Spartan-II ファミリの DLL ロケーション制約

DLL は、デバイスの四隅に 1 つずつ配置されます。DLL のロケーションを制御するには、0、1、2、または 3 という識別番号がある DLL シンボルに対してロケーション制約 LOC を指定します。4 つの DLL の方向と対応するクロック リソースを、図 8 に示します。

LOC プロパティでは、「LOC=DLL2」という形式を使用します。

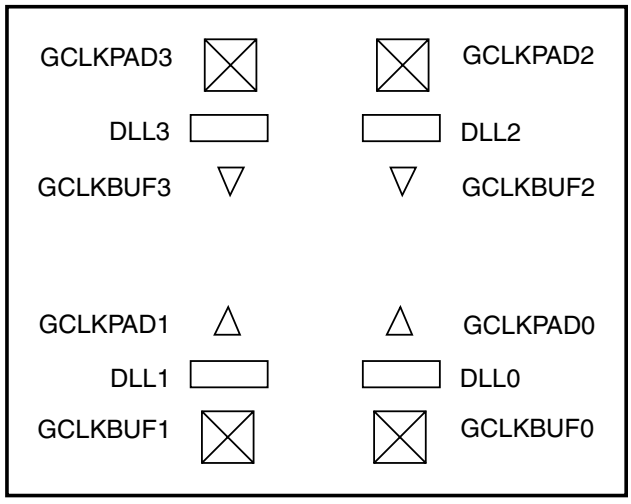

図 8: Spartan-II ファミリの DLL の方向

デザイン時の考慮事項

次の事項を確認して、陥りやすいミスに注意し、ザイリンクス デバイスのデザインを成功させてください。

入力クロック

DLL の出力クロック信号は入力クロック信号を遅延させたものなので、入力クロックの不安定性が出力波形に反映されます。このため、DLL の入力クロックの品質は、DLL が生成する出力クロックの品質に直接影響を与えます。DLL の入力クロックに関する必要条件は、データシートに記載されています。

ほとんどのシステムでは、クリスタル オシレータを使用してシステム クロックを生成します。DLL は、市販されている任意のクリスタル オシレータと共に使用できます。たとえば、ほとんどのクリスタル オシレータは、周波数誤差が 100PPM、つまり、クロック周期の変化が 0.01% の出力波形を生成します。DLL は、周波数ドリフトが 1ns 以内の入力波形については安定して動作しますが、市販されているすべてのクリスタル オシレータをサポートするには、さらに精度が必要です。サイクル間のジッターは、低周波の場合は 300ps 以下、高周波の場合は 150ps 以下に保つ必要があります。

入力クロックの変更

入力クロックの周期を最大のドリフト量以上に変更する場合は、手動で CLKDLL をリセットする必要があります。DLL のリセットに失敗すると、ロック信号と出力クロックの信頼性が失われます。

入力クロックは、DLL にほとんど影響を与えずに停止できます。デバイスの冷却を最低限にするため、クロックを停止する期間を 100 μs 未満にします。クロックは、位相が Low のときに停止する必要があります。クロックを復元するときは、High の周期の先頭から始めます。このとき、LOCKED は High になり、クロックを復元するまで High の状態が続きます。

クロックを停止すると、遅延線に残っているクロックが出力されるまで、1 ～ 4 クロックが出力されます。クロックを再開すると、遅延線の影響で、初めの 1 ～ 4 クロックは出力されません。多くの場合、これは 2 ～ 3 クロックになります。

同様の方法で、入力クロックの位相シフトを行うこともできます。位相シフトは、シフトを開始してから 1 ～ 4 クロック遅れて出力に伝搬されますが、CLKDLL の制御に影響はありません。

出力クロック

DLL のピンに関するセクションで説明したように、出力ピンの接続にはいくつかの制限があります。DLL のクロック出力は、OBUF とグローバル クロック バッファ BUFG を駆動できます。デスティネーション クロック ピンに直接配線することもできます。DLL のクロック出力で駆動できる BUFG は、同じエッジ (上または下) にある 2 つの BUFG だけです。

DLL の出力クロック信号は、LOCKED 信号がアクティブになるまで使用しないでください。LOCKED 信号がアクティブになるまで DLL の出力クロックは無効で、グリッチやスパイクなどが発生する場合があります。

実用的なアプリケーション例

DLL は、有意義で実用的な各種のアプリケーションで使用できます。以下の例では、一般的なアプリケーションを示します。Verilog と VHDL のサンプル ファイルは、ftp://ftp.xilinx.com/pub/applications/xapp/xapp174.zip で入手できます。

標準的な使用法

図 9 に示す回路は、CLKDLL の RST ピンと LOCKED ピンに対するアクセスを提供するようにインプリメントされている点で、BUFGDLL マクロと共通しています。

この回路の VHDL と Verilog によるインプリメンテーションが、`xapp174.zip` の dll_standard ファイルに含まれています。

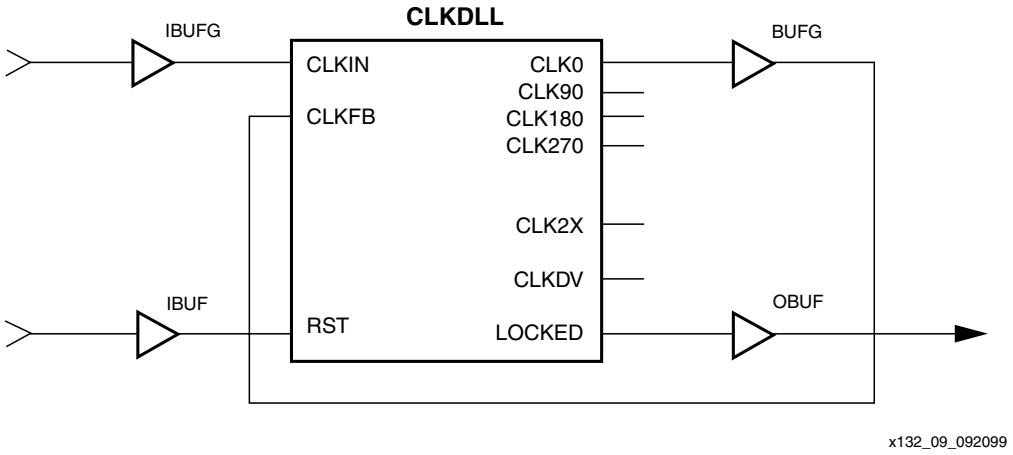

図 9: 標準的な DLL のインプリメンテーション

Spartan-II 以外の複数デバイスのボード レベルのスキュー短縮

図 10 に示す回路を使用すると、同じボード上にある Spartan-II チップと他の Spartan-II 以外のチップの間のシステム クロックのスキューを短縮できます。一般に、このアプリケーションは、SRAM デバイスや DRAM デバイスなどの標準的な製品と共に Spartan-II デバイスを使用する場合に使用されます。ボード レベルの配線をデザインするときは、ソースに戻るネットの遅延が関連する他のチップに対する遅延と等しくなるようにします。

DLL の出力クロック信号は、LOCKED 信号がアクティブになるまで使用しないでください。LOCKED 信号がアクティブになるまで DLL の出力クロックは無効で、グリッチやスパイクなどが発生する場合があります。

この回路の VHDL と Verilog によるインプリメンテーションが、`xapp174.zip` の `dll_mirror_1` ファイルに含まれています。

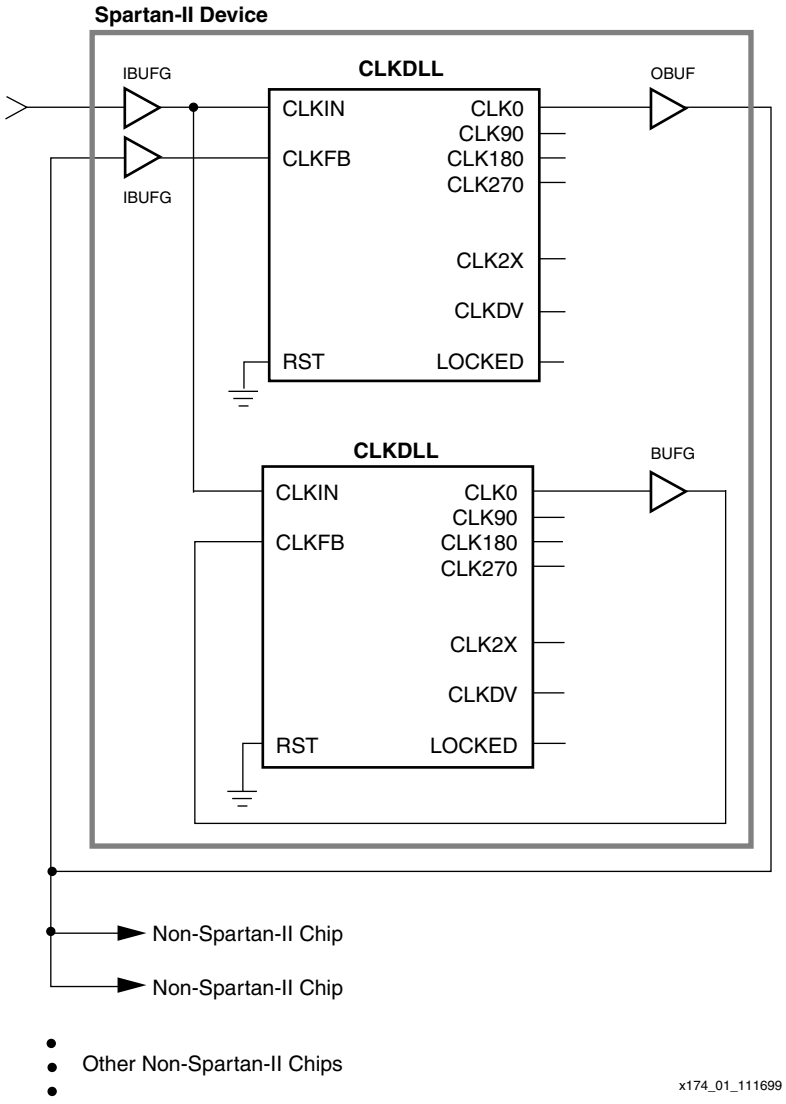

図 10: Spartan-II デバイスと複数の Spartan-II 以外のデバイス間で DLL を使用するボード レベルのクロック スキューの短縮

複数の Spartan-II デバイスのボード レベルのスキュー短縮

図 11 に示す回路を使用すると、同じボード上にある複数の Spartan-II チップの間のシステム クロックのスキューを短縮できます。ボード レベルの配線をデザインするときは、ソースに戻るネットの遅延が関連する他のチップに対する遅延と等しくなるようにします。

DLL の出力クロック信号は、LOCKED 信号がアクティブになるまで使用しないでください。LOCKED 信号がアクティブになるまで DLL の出力クロックは無効で、グリッチやスパイクなどが発生する場合があります。

この回路の VHDL と Verilog によるインプリメンテーションが、`xapp174.zip` の `dll_mirror_2` ファイルに含まれています。

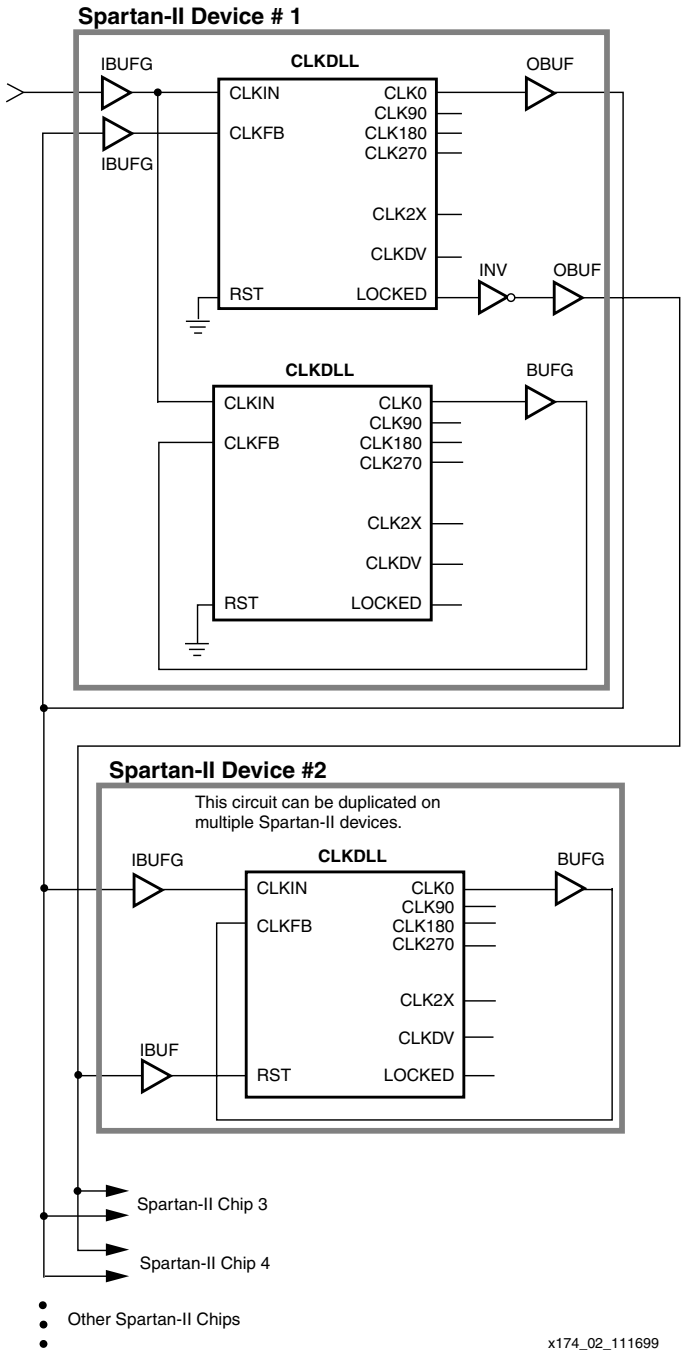

図 11: 複数の Spartan-II デバイス間で DLL を使用するボード レベルのクロック スキューの短縮

クロックとその 2 倍クロックのスキュー短縮

図 12 に示す回路は、2 倍クロック増倍器をインプリメントしています。また、CLK0 出力を使用して、同一チップ内にあるレジスタ間のスキューをゼロにしています。同様の接続を使用して、クロック分周回路もインプリメントできます。

単一の DLL でアクセスできる BUFG は 2 つまでなので、この例の DLL に出力クロック信号を追加する場合は、高速バックボーン配線を使用する必要があります。

この回路の VHDL と Verilog によるインプリメンテーションが、`xapp174.zip` の `dll_2x` ファイルに含まれています。

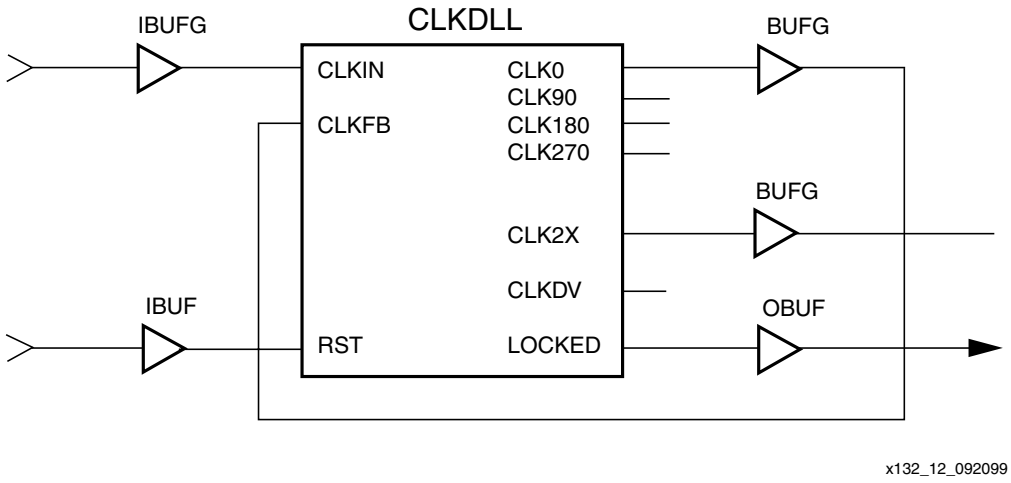

図 12: DLL によるクロックとその 2 倍クロックのスキューの短縮

4 倍クロックの生成

図 13 に示すように、2 倍クロック増倍器をインプリメントする 2 つの DLL 回路を直列に接続すると、同一デバイス内のレジスタ間でスキューがゼロの 4 倍クロックをインプリメントできます。

別のクロック出力が必要な場合は、2 つの DLL のロケーションをデバイスの反対側のエッジ (上と下) に制約した場合のみ、BUFG を駆動できます。

この回路を使用する場合は、初めのチップ リセットの後で SRL16 セルを使用して 2 番目の DLL をリセットする必要があります。このようにしないと、入力が 1 倍 (25/75) から 2 倍 (50/50) に変化したときに 2 番目の DLL が周波数の変化を認識しない場合があります。

この回路の VHDL と Verilog によるインプリメンテーションが、`xapp174.zip` の `dll_4x` ファイルに含まれています。

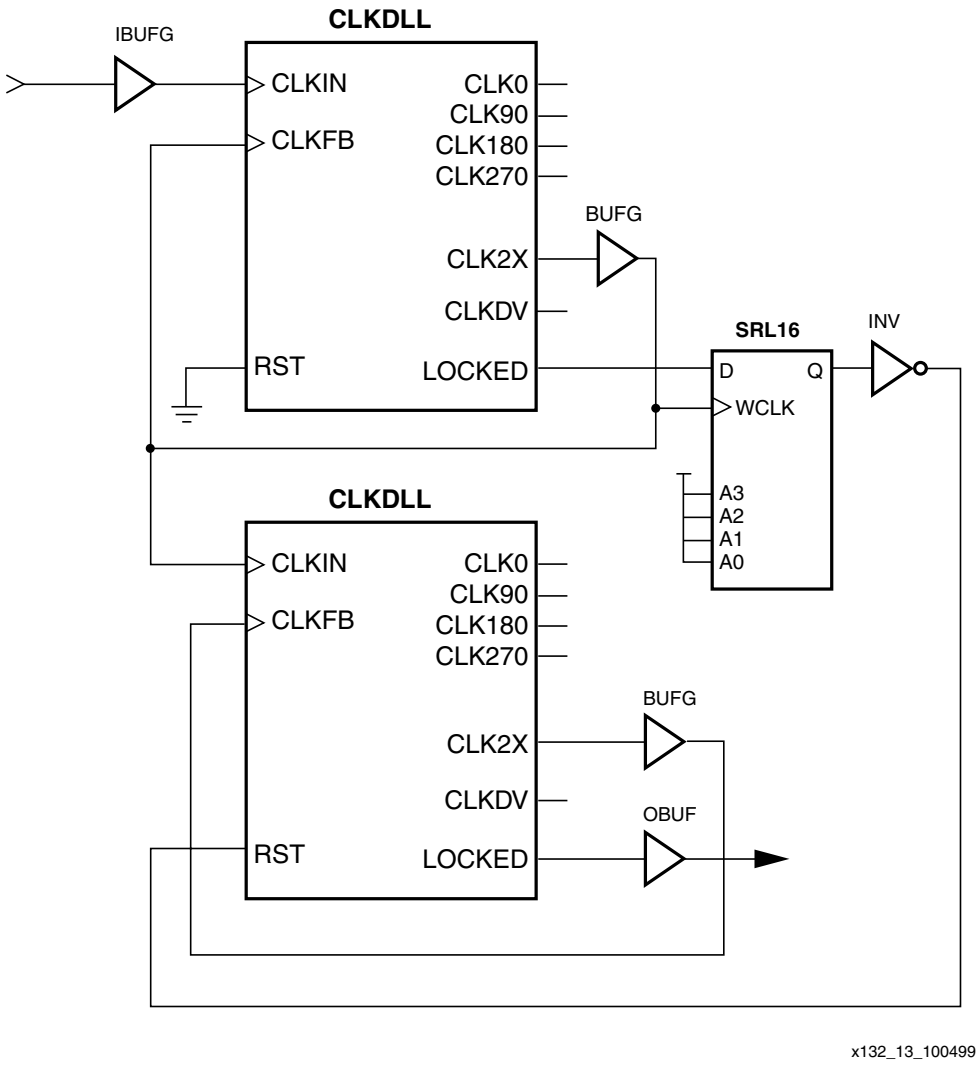

図 13: DLL による 4 倍クロックの生成

改訂履歴

次の表に、このドキュメントの改訂履歴を示します。

日付	バージョン	改訂内容
11.23.99	1.0	初期リリース
01.24.00	1.1	4 倍クロックの生成を追加