

はじめに

HardCopy™ デバイスは、対応する PLD と比較して、大幅なコスト削減、性能向上、および消費電力の低減を実現します。FPGA デバイスから対応する HardCopy デバイスに可能な限りスムーズに確実に移行するには、FPGA 実装の進行中に一連のデザイン・ルールに適合させるよう配慮する必要があります。標準的に認められている FPGA のコーディング方式に注意を払う時期が早いほど、容易に推奨ガイドラインに従うことができます。ここでは、設計者が回避する必要のある一般的な状況をいくつか説明し、そのような状況でのデザイン方法の代替手段を示します。

デザイン・アシスタント・ツール

Quartus® II ソフトウェアのバージョン 2.1 以降のデザイン・アシスタント・ツールでは、デザイン・プロセスの初期段階での潜在的なデザイン問題をチェックする簡単な方法を提供します。デザイン・アシスタントは、FPGA にコンパイルされた後にネットリスト内の個々の回路構造を検索するデザイン・ルール・チェック・ツールです。デザインに存在する違反ルールの要約と各違反例を詳細に提供します。ツールがチェックする一連のルールを指定することによって、一部のルール違反を許容できます。これは、重要でない特定のルールにデザインが違反することが分かっている場合に便利ことがあります。ただし、HardCopy デザインについては、すべてのデザイン・アシスタント・ツールをイネーブルにする必要があります。

デザイン・アシスタントでは、メッセージは以下の 4 つの重要度レベルによって分類されます。

表 11-1. デザイン・アシスタントのメッセージの重要度レベル (1 / 2)

重要度レベル	説明
重大	メッセージで説明されているルール違反は、デザインの信頼性に危機的な影響を及ぼします。アルテラは、これらの違反を綿密に調査しなければ、このデザインを正しく HardCopy デバイスに移行させることができません。
大	メッセージで説明されているルール違反は、デザインの信頼性に影響を及ぼします。アルテラでは、このデザインを HardCopy デバイスに移行させる前にこれらの違反を調査する必要があります。

表 11-1. デザイン・アシスタントのメッセージの重要度レベル (2 / 2)

重要度レベル	説明
中	メッセージで説明されているルール違反によって、実装が複雑になる可能性があります。この違反によって、デザインを HardCopy デバイスに移行させるのに必要なスケジュールや結果に影響を受けません。
情報のみ	メッセージにはデザイン・ルールに関する情報が含まれています。

目標は、デザイン・アシスタントが生成する情報メッセージ以外のメッセージの数をゼロにすることです。この目標が達成されない場合、移行を実施するアルテラの HardCopy デザイン・センタがデザイン・アシスタントの各メッセージを綿密に調査した後のみ、このデザインの HardCopy デザインへの実装を検討するようにしてください。アルテラは、これらのメッセージを調査した後、デザインが HardCopy の実装には適していないと判断することもあります。

非同期クロック・ドメイン

デザインには通常、それぞれがデザインのサブセクションをドライブするクロック・ソースをいくつか持たせることができます。1つのクロック・ソースによってドライブされる、このタイプのデザインのサブセクションは、クロック・ドメインと呼ばれます。全体のデザインには通常、数個のクロック・ソースが含まれており、これらのクロック・ソースそれぞれの周波数と位相を他とは別にすることができます。

図 11-1 のタイミング図を検討してみましょう。ここでは、非同期クロック・ドメインの性質を説明するために2つの自走クロックが使用されています。2つのクロック信号間に同期関係または固定された関係がない場合、これらは互いに非同期であると見なされます。良い例は、明白な高調波関係のない周波数で動作する2つのクロック信号です。

図 11-1. 明白な高調波関係のない2つのクロック信号 注 (1)、(2)

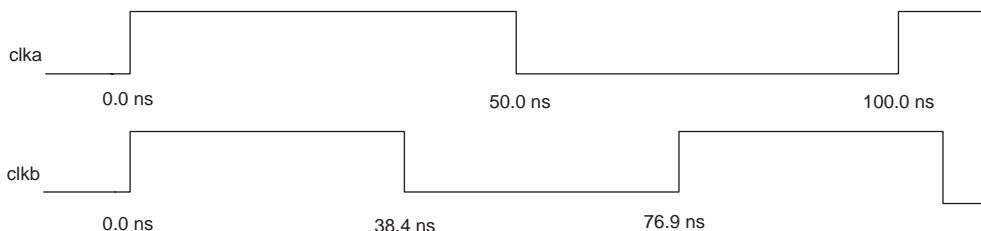


図 11-1 の注 :

- (1) $clk_a = 10 \text{ MHz}$, $clk_b = 13 \text{ MHz}$ 。
- (2) クロックのデューティ・サイクルはどちらも 50%。

信号 `clk_a` は、0.0 ns での立ち上がりエッジ、50 ns での立ち下がりエッジ、および 100 ns での次の立ち上がりエッジによって定義されます (1/10 MHz = 100 ns)。後続の `clk_a` の立ち上がりエッジが発生するのは、200 ns、300 ns、400 ns などです。

信号 `clk_b` は、0.0 ns での立ち上がりエッジ、38.45 ns での立ち下がりエッジ、76.9 ns での次の立ち上がりエッジによって定義されます。後続の `clk_b` の立ち上がりエッジが発生するのは、153.8 ns、230.7 ns、307.6 ns、384.5 ns などです。

`clk_a` と `clk_b` のエッジは、`clk_b` の 1,000 番目のクロック・エッジ (1000 × 76.9 = 7,690 ns) または `clk_a` の 7,690 番目のクロック・エッジ (7,690 × 100 = 7,6900 ns) までには一致しません。これら 2 つのクロックを 7,6900 ns ごとに同期させる意図があるとは考えにくいので、これら 2 つのクロック・ドメインは互いに非同期であると見なされます。

さらに微妙な非同期クロック・ドメインのケースは、2 つのクロック・ドメインが非常に明白な周波数および位相関係を持つ場合、特に一方が他方の倍数である場合に発生します。100 MHz および 50 MHz で動作するクロックを持つシステムを検討してみましょう。これらのクロックの一方のエッジは、常にクロックの他方のエッジから一定の時間間隔だけ離れています。この場合、クロック・ドメインが非同期であるか否かは、これら 2 つのクロック・ドメインの相互作用に関する設計者の本来の目的が何であるかに依存します。

同様に、同じ標準周波数で動作する 2 つのクロックは、両者の間に同期化させるメカニズムがなければ互いに非同期となることもあります。例えば、PC ボード上で 100 MHz で動作する 2 つの水晶オシレータは温度の変動によって周波数がいくらか変化しますが、この変化がオシレータごとに異なる場合があります。この結果、互いに位相が同じになったり外れたりしながら変動する 2 つの独立したクロック信号が生じます。

2 つの非同期クロック・ドメイン間でのデータ転送

2 つの非同期クロック・ドメインを互いに通信させなければならない場合、この動作を確実に実行させる方法について、若干の考慮が必要です。このセクションでは、以下のとおり、2 つの非同期クロック・ドメイン間でのデータの転送方法の例を 3 つ示します。

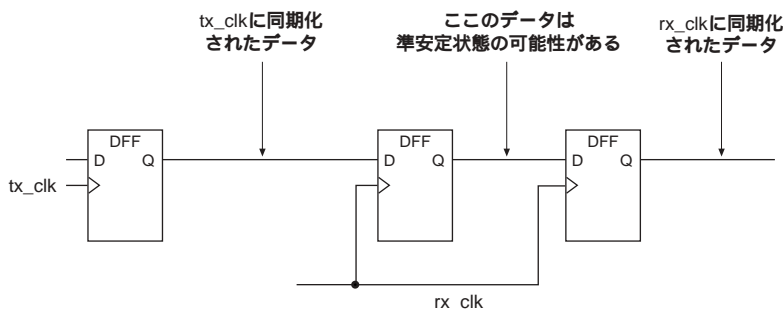
- ダブル・シンクロナイザの使用
- FIFO バッファの使用
- ハンドシェイク・プロトコルの使用

使用する方法は、個々のアプリケーション、クロックの境界を横切る非同期信号の数、およびクロス・ドメイン転送の実行に使用できるリソースに応じて選択されます。

ダブル・シンクロナイザの使用

図 11-2 の回路は、受信側のクロックによってクロック駆動される 2 ビット・シフト・レジスタ構造で構成されています。シフト・レジスタの第 2 ステージによって、第 1 レジスタから第 2 レジスタの出力に伝播するデータ出力に準安定状態（未知の状態）が発生する確率が低減されます。送信側クロック・ドメインからのデータは、レジスタから直接供給する必要があります。この手法は、クロック・ドメインにまたがってシングル・データ信号（つまり、データ・バスではない）を転送する必要がある場合にのみ推奨されます。これは、データ・バスの場合は、一部のビットが 1 つのクロック・サイクルでキャプチャされ、その他のビットが次のサイクルでキャプチャされる可能性があるからです。待ち時間は増加しますが、シンクロナイザ回路を 2 ステージ以上使用することもできます。ステージ数を増やす利点は、ステージを追加するたびに平均故障時間（MTBF）が長くなることです。

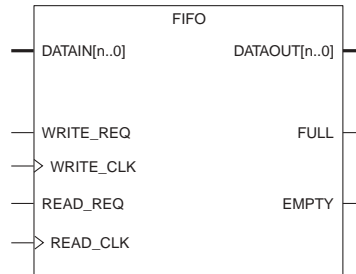
図 11-2. ダブル・シンクロナイザ回路



FIFO バッファの使用

図 11-3 の回路の利点は、アルテラの MegaWizard® Plug-In Manager を使用してきわめて容易に設計できることです。非同期クロック・ドメインにまたがってデータ・バス信号を転送しなければならない場合に最も役立ち、またこのデータを一時的に保存できる点も便利です。さらに、このタイプの回路がデザイン・アシスタントの警告を生成することはありません。

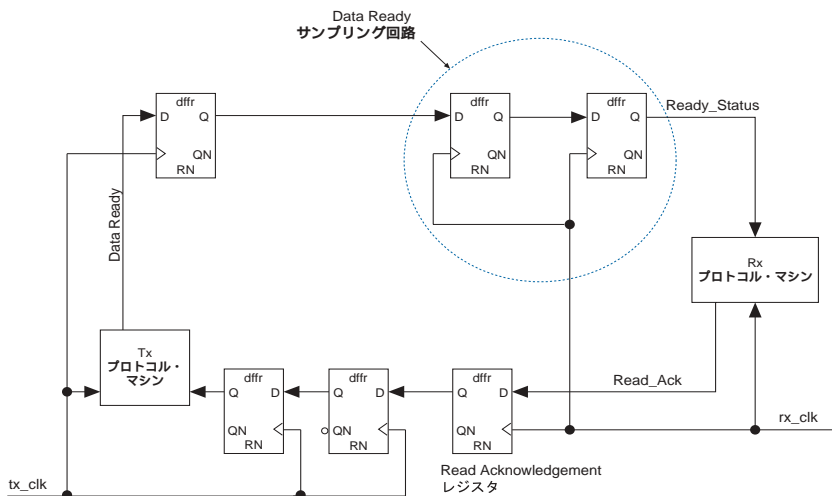
図 11-3. FIFO バッファ



ハンドシェイク・プロトコルの使用

この回路は、実装に使用されるロジック・セルの数量が少なく、非同期クロック・ドメインを横切るデータ・バスのすべてのビットが受信側クロック・ドメイン内で同じクロック・エッジによってレジスタに記憶されることを保証します。図 11-4 に示すこの回路は、FIFO バッファとして使用できるメモリがなく、クロック・ドメイン間で転送させるデータ・バスがデザインに多く含まれる場合に最適です。

図 11-4. ハンドシェイク・プロトコル回路

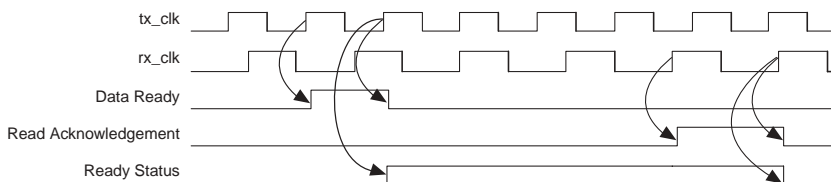


この回路の動作は、送信側クロック・ドメイン tx_clk 内で Data Ready 信号が high になると開始されます。この信号は data ready サンプリング・レジスタに送られ、Ready ステータス信号を high にします。Data

Ready 信号は、Rx ドメインで正しくサンプリングされるよう持続時間を十分長くする必要があります。これは、rx_clk 信号が tx_clk より遅い場合に重要です。

この時点では、受信側クロック・ドメイン rx_clk は送信側クロック・ドメイン tx_clk からのデータを読み込むことができます。この読み込み動作の終了後、受信側クロック・ドメイン rx_clk は同期 Read Acknowledge 信号を生成し、この信号は Read Acknowledge レジスタに記憶されます。レジスタに記憶された信号は、送信側ドメイン内の Read_Ack サンプリング回路でサンプリングされます。Read Acknowledge 信号は、Tx ドメイン内で正しくサンプリングされるよう持続時間を十分長くする必要があります。これは、送信側クロックが Rx クロックより速度が遅い場合に重要です。図 11-5 のタイミング図に示すとおり、このイベントの後、2 つの非同期ドメイン間のデータ転送が完了します。

図 11-5. 2 つの非同期クロック・ドメイン間のデータ転送



ゲート付き クロック

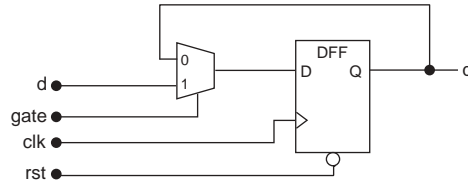
クロック・ゲーティングは、回路の一部を「ターン・オフ」するのに使用されることがあります。これはデバイスの全消費電力を低減させるのによく使用されます。これはゲート付きクロック信号は、ドライブするロジックがスイッチングして電力を消費するのを防止するためです。これは、クロック・ツリーのルートでゲーティングを行う場合に最も効果があります。クロックがリーフ・セルのレベルで（つまり、レジスタへの入力の直前に）ゲートされる場合、クロック・ネットワーク全体が依然としてトグルするため、電力はほとんど節約されません。このタイプの回路を使用するときの欠点は、一定のルールが守られていない場合、得られるゲート付きクロック信号に予期しないグリッチが生じる可能性があることです。

推奨されるクロック・ゲーティング回路

クロック信号をゲーティングする方法として推奨されるのは、図 11-6 に示すような純粋な同期回路を使用することです。この方法では、クロックは実際にはまったくゲートされません。正確に言えば、レジスタへのデータ信号がゲートされます。この回路は、クロック・イネーブル (CE)

ピン付きのレジスタとして表現されることがあります。この回路は、ゲート信号上のグリッチの影響を受けないため、レジスタまたは任意の複雑な組み合わせファンクションから直接生成できます。ゲートまたはクロック・イネーブル信号に対する制約は、ゲーティング・マルチプレクサの 'd' 入力に対するものとまったく同じです。これらの信号は両方とも、供給先のレジスタのセットアップ時間およびホールド時間に適合する必要があります。

図 11-6. 推奨されるクロック・ゲーティング回路



この回路を記述するのに必要なのは、数行の VHDL または Verilog HDL だけです。

以下は同期クロック・ゲーティング回路に対する VHDL コードの一部分です。

```
architecture rtl of vhdl_enable is
begin
  process (rst, clk)
  begin
    if (rst = '0') then
      q <= '0';
    elsif clk'event and clk = '1' then
      if (gate = '1') then
        q <= d;
      end if;
    end if;
  end process;
end rtl;
```

以下は同期クロック・ゲーティング回路に対する Verilog HDL コードの一部分です。

```
always @ (posedge clk or negedge rst)
begin
  if (!rst)
    q <= 1'b0;
  else if (gate)
    q <= d;
  else
    q <= q;
end
```

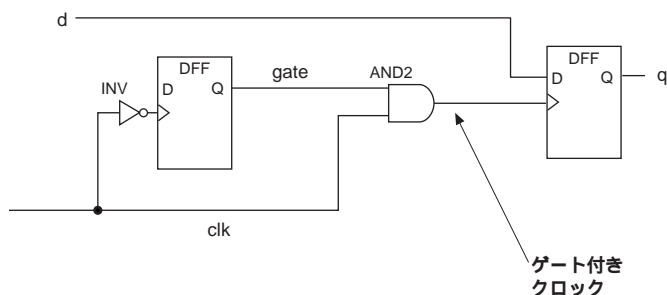
代替クロック・ゲーティング回路

デザインでクロック・ゲーティング回路が不可欠であれば、次の2つの回路のいずれかを使用することもでき、これらの回路に対してデザイン・アシスタントが違反を警告することはありません。

AND ゲートを使用するクロック・ゲーティング回路

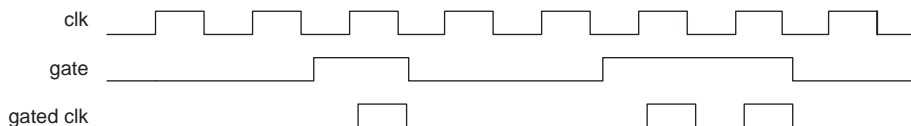
正エッジ・トリガ・レジスタに供給されるゲート付きクロック信号の場合、2入力 AND ゲートを使用します。AND ゲートへの一方の入力は、元のクロック信号です。AND ゲートへの他方の入力は、ゲーティング信号であり、同じ元のクロック信号の負エッジによってクロック駆動されるレジスタから直接ドライブする必要があります。このタイプの回路を [図 11-7](#) に示します。

図 11-7. AND ゲートを使用するクロック・ゲーティング回路



ゲート信号を生成するレジスタは、同じクロックの負エッジでトリガされるため、デザインで同じクロックの両方のエッジを使用することの影響を考慮する必要があります。[図 11-8](#)のタイミング図はこの回路の動作を示したものです。gate 信号はクロックの負エッジの後に発生し、レジスタから直接供給されます。この gate 信号と元の非反転クロックとの論理 AND によってクリーンなクロック信号が生成されます。

図 11-8. AND ゲートを使用するクロック・ゲーティング回路のタイミング図

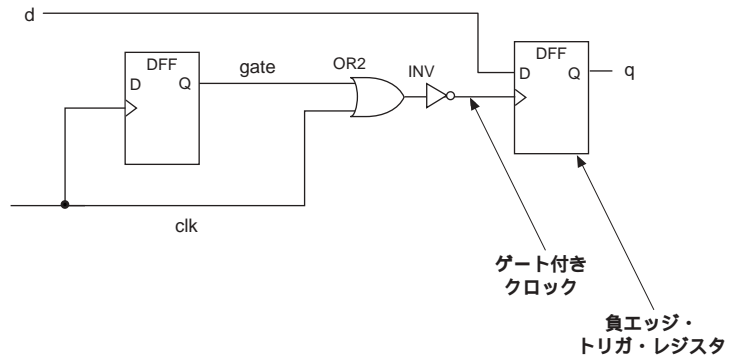


gate 信号を生成するレジスタと AND ゲートへの gate 入力との間の遅延がクロックの low 期間 (デューティ・サイクル 50% のクロックではクロック周期の半分) より長い場合、クロックのパルス幅は狭くなります。

OR ゲートを使用するクロック・ゲーティング回路

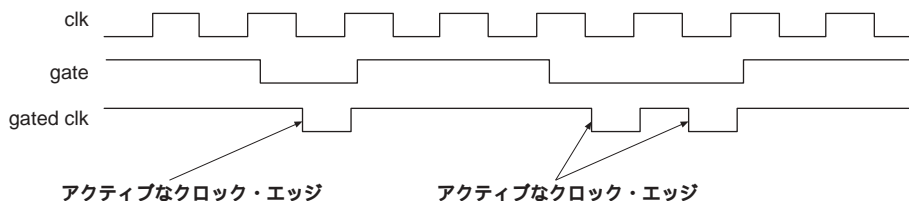
負エッジ・トリガ・レジスタに供給されるゲート付きクロック信号の場合、2 入力 OR ゲートを使用します。OR ゲートの一方の入力は元のクロック信号です。OR ゲートの他方の入力はゲーティング信号であり、同じ元のクロック信号の正エッジによってクロック駆動されるレジスタから直接ドライブする必要があります。この回路を図 11-9 に示します。

図 11-9. OR ゲートを使用するクロック・ゲーティング回路




ゲート信号を生成するレジスタは同じクロックの正エッジでトリガされるため、デザインの同じクロックの両方のエッジを使用することの影響を考慮する必要があります。図 11-10 のタイミング図はこの回路の動作を示したものです。gate 信号はクロックの正エッジの後に発生し、レジスタから直接供給されます。この gate 信号と元の非反転クロックとの論理 OR によってクリーンなクロック信号が生成されます。このクリーンなゲート付きクロック信号は、同じクロックの負エッジを使用するレジスタにのみ供給するようにします。

図 11-10. OR ゲートを使用するクロック・ゲーティング回路のタイミング図



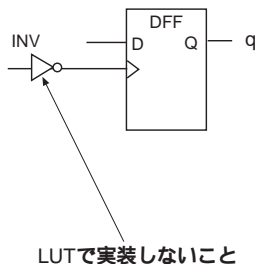
gate 信号を生成するレジスタと OR ゲートへの gate 入力との間の遅延がクロックの low 期間 (デューティ・サイクル 50% のクロックではクロック周期の半分) より長い場合、クロックのパルス幅は狭くなります。

 常に同期クロック・ゲーティング回路を使用するようにしてください。

反転クロック


図 11-11 に示すように、デザインでクロックの正エッジと負エッジの両方を使用する必要がある場合もあります。アルテラの FPGA では、各ロジック・エレメントがプログラマブルなクロック反転機能を備えており、これが反転クロックを生成する唯一の方法です。反転クロック信号を生成するインバータとして構成されたロジック・エレメント・ルック・アップ・テーブル (LUT) をインスタンス化しないでください。

図 11-11. インバータとして構成されたロジック・エレメント LUT



LUT を使用してクロックの反転を実行することは推奨しません。これは、デザインのタイミング・クロージャに対する重要な課題を提起するような、大きなクロック挿入遅延およびスキューを引き起こす可能性が

あるためです。また、必要以上のデバイス・リソースを消費することにもなります。このトピックの詳細については、11-13 ページの「クロック・エッジの混合」を参照してください。

 クロックを反転するのに使用されるロジック・エレメントをインスタンス化する回路図や RTL コードを作成しないでください。代わりに、反転クロックの実現については合成ツールに決定させます。

クロック・ピン以外のピンをドライブするクロック

一般的なガイドラインとして、クロック・ソースはレジスタのクロック・ピンをドライブするためだけに使用します。このルールの例外はありますが、例外をできるだけ無くすか、できればすべて除去する様あらゆる努力をするべきです。


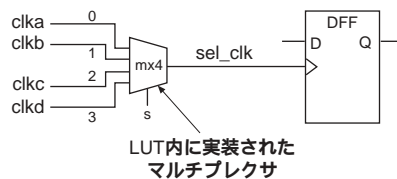
1 つの例外はゲート付きクロックであり、これについては前のセクションで説明しました。クロック信号がクロック・ピン以外のピンをドライブする別の例は、 11-12 に示すように、多数の異なるクロック・ソースから 1 つのクロックを選択するのにクロック・マルチプレクサ回路を使用する場合です。このタイプの回路は、HardCopy 実装と FPGA 実装の両方のスタティック・タイミング解析が複雑になるため、可能な限り避けてください。例えば、このクロックのタイミングを調べるには、マルチプレクサ出力ピンでクロックの割り当てを行う必要があります。マルチプレクサ出力ピンには特定の名前が付けられ、この名前は設計中に変更される場合があります。

図 11-12. LUT 内に実装されたマルチプレクサを示す回路

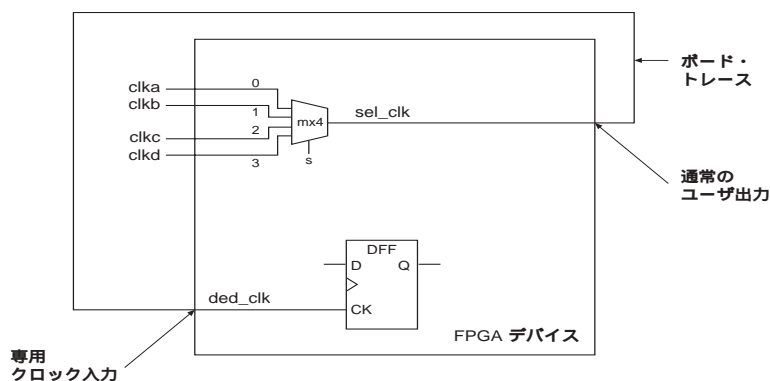


FPGA ではクロック・マルチプレキシング回路は、1 つまたは複数の LUT によって構築され、場合によってはマルチプレクサ出力クロックが専用クロック・リソースの 1 つを使用しなくなることが考えられます。その結果、この多重化クロックのスキューと挿入遅延が増大し、性能に悪影響を及ぼす可能性があります。

このタイプの機能がデザインに必要な場合は、マルチプレクサ出力が確実に FPGA 内のグローバル配線リソースの 1 つをドライブするようにしてください。例えば、この出力は APEX™ 20KE デバイスの FAST ライ

ン、あるいは Stratix デバイスのグローバル・クロックまたはリージョナル・クロックをドライブする必要があります。グローバル・クロック・リソースが確実にチップ全体でのクロック信号の分配に使用されるようにするための代替手段は、多重化クロック信号を主要出力ピンに配線することです。図 11-13 に、この回路を示します。デバイスの外側では、この出力ピンは、おそらくクロック挿入遅延を低減するために PLL を通して、同じデバイスの専用クロック入力の 1 つをドライブします。この手法はマルチプレキシング回路および外部ボード・トレースを通して、なおも大きな遅延が存在する点で完全ではありませんが、選択されたクロック信号用に専用クロック・リソースを使用しているため、クロックのスキューは結果的にごく小さなものになります。他の実装に対してこの回路が有利な点は、タイミング解析が非常に簡単になることであり、解析するのは主要入力ピンをソースとする 1 つのクロック・ドメインだけです。

図 11-13. 主要出力ピンへの多重化クロック信号の配線




クロック信号は専用クロック・リソースを使用

デザインのクロック信号はすべてターゲットの FPGA 内に存在するグローバル・クロック・ネットワークに割り当てるようにします。非専用クロック・ネットワークを使用するようにマップされたクロック信号は、デザイン性能に悪影響を与える可能性があります。これは、専用クロック・ネットワークよりも低速で、大きなスキューを持つ可能性がある標準 FPGA 配線リソースを使用してクロックを分配する必要があるためです。ターゲットの FPGA で供給されるよりも多くのクロックをデザインが必要とする場合、FPGA 内で専用クロック・リソースのみがクロック分配に使用されるようクロックの数を減らすことを検討してください。専用クロック・リソースの数を超える必要がある場合は、標準(非クロック

ク・ネットワーク)の配線リソースによって、最も小さいファン・アウトでクロックを実装する方法を選択してください。また、専用クロック・リソースをどのように割り当てるかを決定する際は、最も高速のクロック信号を優先するようにしてください。

Quartus II ソフトウェアでは、Global Signal ロジック・オプションを使用して、クロック信号がグローバル信号であることを指定できます。また、Auto Global Clock ロジック・オプションを使用して、フィタが自動的にクロック信号をグローバル信号として選択できるようにすることも可能です。

 常に FPGA の内蔵クロック・ネットワークを使用するようにしてください。

クロック・エッジの混合

デザインで1つのクロックの両方のエッジを使用することが可能であり、それが望ましいこともあります。希望する機能を実現するためにクロックの両方のエッジを使用しなければならない例は、DDR メモリ・インタフェースを使用する場合です。Stratix™ および HardCopy Stratix™ デバイスでは、このインタフェース・ロジックはデバイスの I/O セルに組み込まれており、このインタフェースに対して高精度のシミュレーションおよび特性評価を行うことにより堅牢さを保証しています。したがって、この専用回路はクロックの両方のエッジを使用するというルールに対する例外です。ただし、汎用ロジック・リソースを使用する一般的なデータ転送については、クロックの1つのエッジのみを使用しなければなりません。回路が1つのクロックの両方のエッジを使用しなければならない場合は、常にクロックのデューティ・サイクルをスタティック・タイミング解析ツールに正確に伝える必要があります。そうしないと、タイミング解析が不正確になる可能性があります。図 11-14 に2つのクロック波形を示します。一方はデューティ・サイクルが50%、他方はデューティ・サイクルが10%です。

図 11-14. デューティ・サイクル 50% および 10% のクロック波形

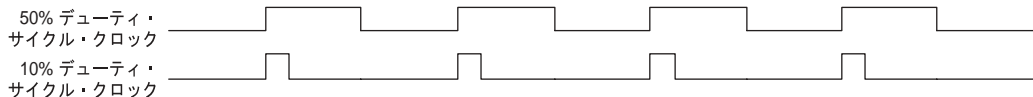


図 11-15 に、クロックの正エッジのみを使用する回路を示します。連続した正のクロック・エッジ間の間隔は常に同じ(つまり、クロック周期)です。この回路では、クロックのデューティ・サイクルは回路の性能に影響しません。

図 11-15. クロックの正エッジを使用する回路

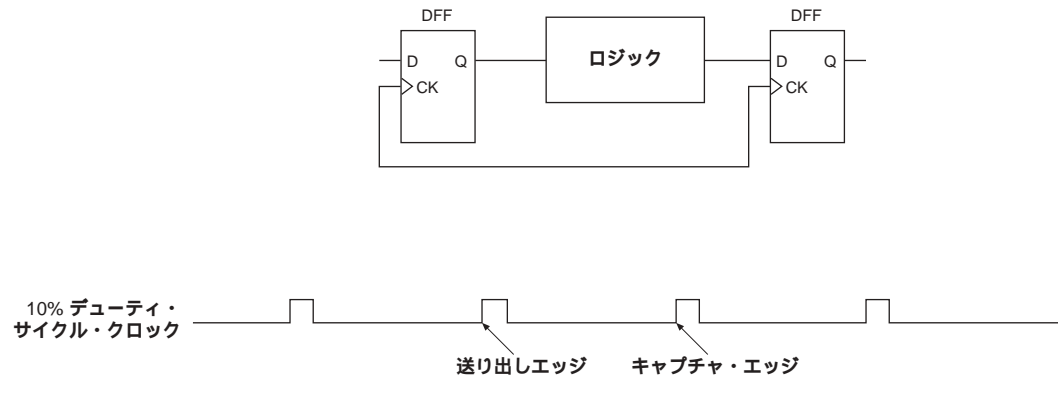
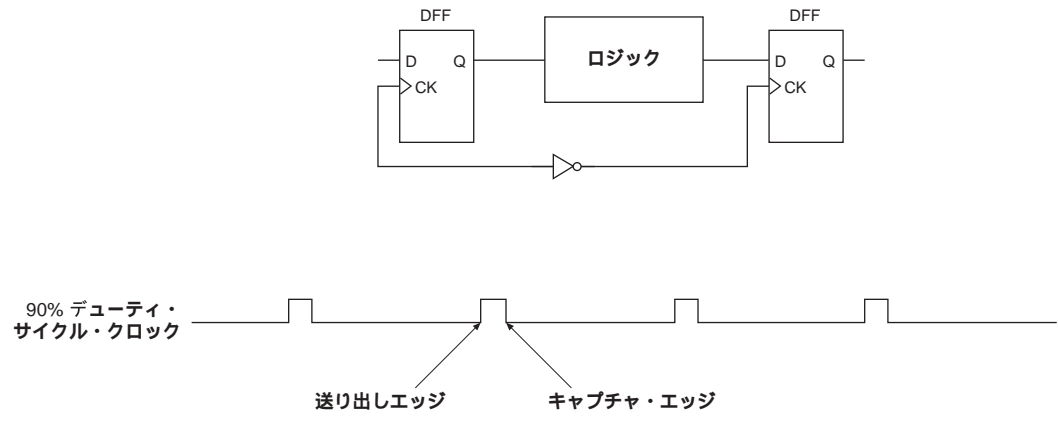



図 11-16 に、正のクロック・エッジを使用してデータを送り出し、負のクロック・エッジを使用してこのデータをキャプチャする回路を示します。このクロックのデューティ・サイクルは 10% なので、送り出しエッジとキャプチャ・エッジ間の時間が短く、キャプチャ・レジスタでセットアップ時間違反が発生しないように、合成ツールでロジックの塊を最適化しようとするのは大変です。

図 11-16. クロックの正エッジおよび負エッジを使用する回路



両方のクロック・エッジを使用する回路を設計すると、デザイン・アシスタントの警告メッセージ "Registers are Triggered by Different Edges of Same Clock. (レジスタが同じクロックの異なるエッジでトリガされます。)" を受け取る可能性があります。この警告メッセージは次の条件下では発生しません。

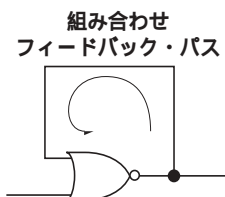
- 反対側のクロック・エッジがクロック・ゲーティング回路で使用されている場合
- ダブル・データ・レート (DDR) メモリ・インタフェース回路が使用されている場合

 デザインではクロックの一方のエッジのみを使用するようにしてください。

組み合わせ ループ

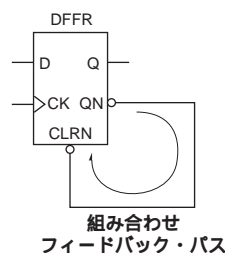
組み合わせセループが存在するのは (図 11-17)、ロジック・ゲート (1個または複数) の出力が最初にレジスタに送られることなく、同じゲートの入力にフィードバックする場合です。デザインに組み合わせセループが存在しないようにしてください。

図 11-17. 組み合わせセループを使用する回路



レジスタ出力ピンが同じレジスタのリセット・ピンをドライブする場合は、レジスタを使用して組み合わせセループを生成することも可能です (図 11-18)。

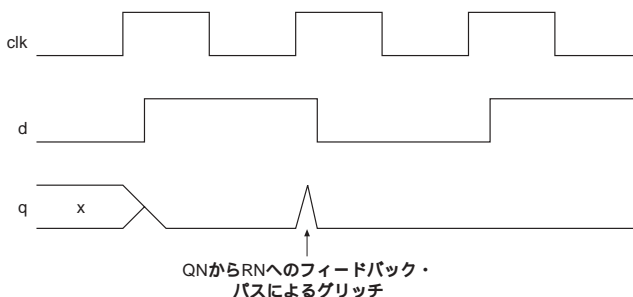
図 11-18. レジスタを使用した、組み合わせセループの生成



この回路のタイミング図を図 11-19 に示します。レジスタの D 入力でロジック 1 の値がクロック・インされると、ロジック 1 の値は立ち上がりクロック・エッジの後に Q 出力ピンに現れます。同じクロック・イベントによって QN 出力ピンが low になるため、レジスタは RN を通してリ

セットされます。その結果、レジスタ出力 Q は low になります。この回路は、QN から RN へのパスに十分な遅延がなければ動作しないことがあり、推奨しておりません。

図 11-19. 図 11-18 の回路のタイミング図




組み合わせフィードバック・ループは、意図的にまたは意図せずにデザインに導入されます。意図的なフィードバック・ループは一般に、おそらく最終段階での機能上またはタイミング上のバグを修正するために、インスタンス化されたラッチの形で導入されます。インスタンス化されたラッチは、ルック・アップ・テーブルから機能を生成する必要があり、FPGA ロジック・ファブリック内にはラッチの基本要素がないため、アルテラの FPGA 内の組み合わせフィードバック・ループの例であるといえます。意図的でない組み合わせフィードバック・ループは通常、RTL 内の IF-THEN または CASE 構成体が完全に指定されていないために存在します。デザイン・アシスタントは、これらの回路構造が存在するかどうかデザインをチェックします。発見された場合は、詳細に調べ、意図的でないラッチを除去するために RTL に修正を施すか、ラッチのインスタンス化が必要でなくなるように回路を再設計する必要があります。アルテラの FPGA では、多くのレジスタを使用できるため、ラッチを使用する必要はまったくありません。

これらの組み合わせループによって、デザインの安定性と信頼性に大きな問題が生じる可能性があります。なぜなら、組み合わせループの動作は多くの場合、ループのロジックの相対的な伝播遅延に依存するからです。この組み合わせループ回路構造は、個々の動作条件によって異なる動作をします。組み合わせループは本質的に非同期であり、EDA ツールは同期回路で最も良好に動作します。

レベルに敏感なラッチのようなストレージ・エレメント、またはエッジ・トリガ・レジスタには、特定のタイミング・チェックが伴います。例えば、エッジ・トリガ・レジスタのデータ入力に対するセットアップおよびホールド要求条件があります。同様に、ゲート信号によってラッチがトランスペアレントから非トランスペアレントに変化したときに、トランスペアレント・ラッチ内でデータを安定させるためのセットアップおよびホールド・タイミング要求条件もあります。ラッチが組み合わせゲー

トから生成されている場合、このようなタイミング・チェックは存在しないため、スタティック・タイミング解析ツールはこれらのラッチ回路に対して必要なチェックを実行できません。

 意図的な組み合わせループと意図的でない組み合わせループがあるかどうかデザインをチェックし、それらを除去してください。

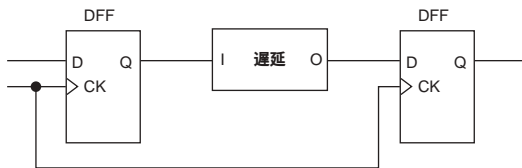
意図的な遅延

デザインに機能的な利点を提供しないセルを、意図的にインスタンス化する理由はありません。このセルには信号を遅延させる作用があるだけです。FPGA 内の専用クロックを使用する純粋な同期回路には、この遅延セルは必要ありません。図 11-20 にこの回路を示します。ASIC では、遅延セルを使用して、2 つのレジスタ間のクロック・スキューが、同じ 2 つのレジスタ間のデータ・パス遅延よりも大きくなることによって発生するホールド時間違反を修正します。FPGA の場合、このクロック・スキューおよび FPGA レジスタのクロックから Q までの時間は、遅延セルが不要となるよう注意深く設計されています。

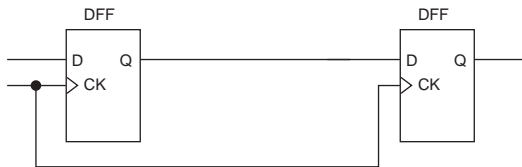
図 11-20 に同じ回路の 2 つのバージョンを示します。第 1 バージョンでは、第 1 レジスタの Q 出力から第 2 レジスタの D 入力へのデータ・パス内に（おそらく LUT を使用して実装された）遅延セルがあります。この回路の機能はシフト・レジスタです。遅延セルの機能は非反転バッファと同じです。この回路の第 2 バージョンもシフト・レジスタの機能を示しますが、データ・パス内に遅延セルがなく、どちらの回路も同様に動作します。

図 11-20. 意図的な遅延を有する回路の 2 つのバージョン

遅延付き回路

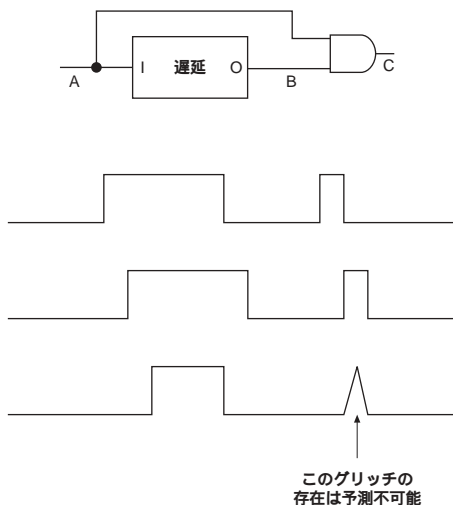


遅延なし回路




デザイン内に遅延チェーンが存在する場合、それらは非同期回路の兆候である可能性があります。このような例の1つを図 11-21 の回路に示します。この回路は AND ゲートの出力でパルスを生成するのに、AND ゲートの2つの入力間の遅延を利用します。パルスは A 入力ピンの波形の形状によって、生成される場合と生成されない場合があります。

図 11-21. 遅延チェーンを示す回路および対応するタイミング図



遅延チェーンを使用すると、さまざまなデザイン問題（デザインが動作条件に影響されやすくなる、デザインの信頼性が低下するなど）が生じる可能性があります。

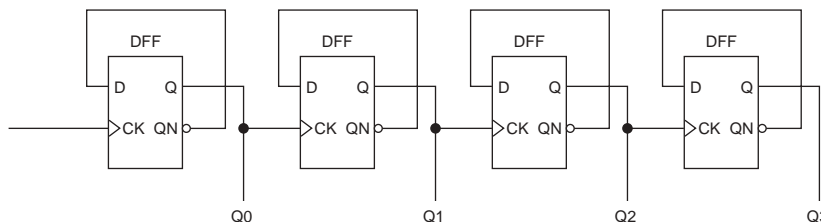
デザイン内の遅延チェーンの例のすべてが非同期回路によって生じるわけではないことに注意してください。デザイン・アシスタントのレポートがユーザの意に反して遅延チェーンの存在を示した場合、その遅延チェーンは事前に構築されたIPファンクションを使用した結果である可能性があります。これらの機能は通常パラメータ化でき、文字通り数千もの異なるパラメータ設定の組み合わせがあります。特定のパラメータ設定が使用されている場合、合成ツールは未使用のロジック・エレメントをすべて除去しない場合もありますが、結果として得られる回路は依然として同期しています。デザイン・アシスタントのすべての遅延チェーン警告メッセージを慎重に確認してください。

 遅延チェーンの使用に依存する回路を設計することは避け、常にデザイン・アシスタントのすべての遅延チェーン警告メッセージを慎重に確認するようにしてください。

リップル・カウンタ

デザインにはリップル・カウンタを含めないようにしてください。図 11-22 に示すリップル・カウンタは、最初のカウンタ・ステージの Q 出力が次のカウンタ・ステージのクロック入力をドライブする回路構造となっています。各カウンタ・ステージは同じレジスタの D 入力にフィードバックする、反転 Q 出力を持つレジスタで構成されています。

図 11-22. 標準的なリップル・カウンタ



このタイプの構造は可能な限り少数のロジックからカウンタを作るのに使用されます。ただし、アルテラの FPGA デバイス内の LE 構造によって、1 カウンタビットにつき 1 個の LE を使用してカウンタを構成できるため、リップル・カウンタ構造を使用しても実際にはロジックは節約されません。リップル・カウンタでは、カウンタの各ステージによって若干の位相遅延が生じ、この位相遅延は連続したカウンタのステージで次第に増大します。図 11-23 は、図 11-22 の位相遅延をタイミング図で示したものです。

図 11-23. 図 11-22 の回路の位相遅延を示すタイミング図

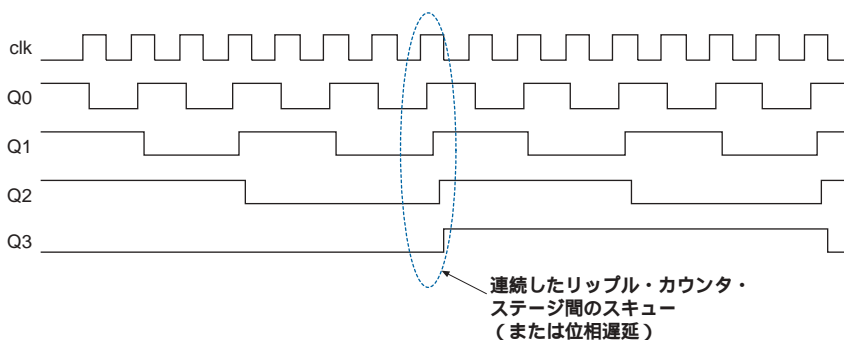
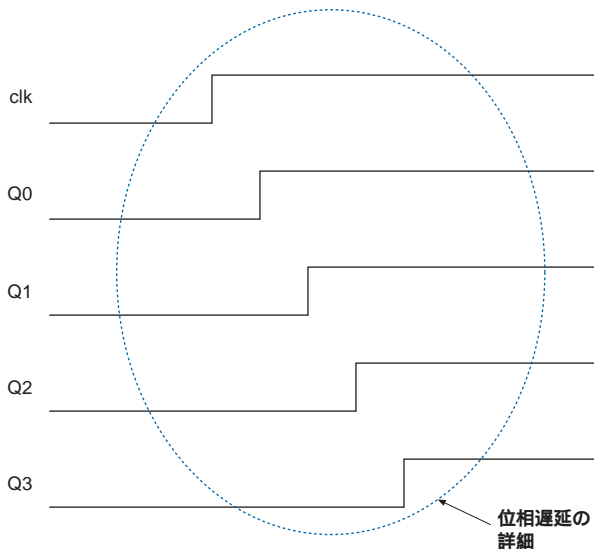



図 11-24 は、上図の位相遅延の詳細を示したものです。

図 11-24. 図 11-23 に示す位相遅延の詳細



リップル・カウンタの出力を他の回路のクロック信号として使用する場合は、この位相遅延が問題になります。リップル・カウンタの出力をクロック信号として使用する回路は、大きなスキューを持つ信号でクロック駆動されることになります。

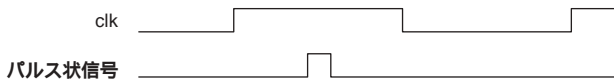
リップル・カウンタのステージごとに新しいクロック・ドメインを定義する必要があるため、スタティック・タイミング解析ツールでリップル・カウンタを解析するのは特に大変です。スタティック・タイミング解析ツールが処理しなければならないクロック・ドメインが多くなるほど、プロセスの複雑さと消費する時間が増えます。

 どのような状況下でもリップル・カウンタを使用するのは避けてください。

パルス・ジェネレータ

パルス・ジェネレータは、1つのクロック周期内で2回以上遷移する信号を生成する回路として定義されます。11-17 ページの「意図的な遅延」のセクションも参照してください。図 11-25 に、パルス・ジェネレータ波形の例を示します。

図 11-25. パルス・ジェネレータ波形の例



パルス・ジェネレータの作成

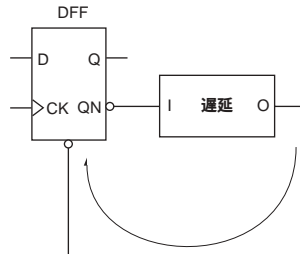
パルス・ジェネレータは2つの方法で作成できます。パルス・ジェネレータを作成する第1の方法は、2入力 AND、NAND、OR、または NOR ゲートを使用してグリッチの幅を広くすることです。この場合、2つのゲート入力のソースは同じですが、ゲート入力の1つに対するソースが遅延されます。このタイプの回路を図 11-26 に示します。

図 11-26. 2入力 AND を使用するパルス・ジェネレータ回路



パルス・ジェネレータを作成する第2の方法はレジスタを使用することであり、図 11-27 に示すように、レジスタ出力が遅延チェーンを通して専用の非同期リセット信号をドライブします。

図 11-27. 遅延チェーンを通してリセット信号をドライブするために、レジスタ出力を使用するパルス・ジェネレータ回路



これらのパルス・ジェネレータは本質的に非同期であり、デザイン・アシスタントによって容認できない回路構造として検出されます。パルス信号を生成しなければならない場合は、純粋に同期式の方法で行う必要があります。つまり、図 11-28 に示すとおり、パルスの持続時間をクロック周期と等しくするか、クロック周期の整数倍と等しくする方法です。

図 11-28. 同期式パルス・ジェネレータの例



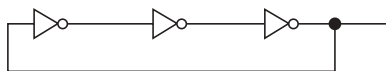
同期式パルス・ジェネレータの生成は、シンプルな Verilog または VHDL コードのセクションで達成できます。以下は、同期式パルス・ジェネレータ回路のための Verilog コードの一部です。

```
reg [2:0] count;
reg pulse;
always @ (posedge clk or negedge rst)
begin
    if (!rst)
        begin
            count[2:0] <= 3'b000;
            pulse <= 1'b0;
        end
    else
        begin
            count[2:0] <= count[2:0] + 1'b1;
            if (count == 3'b000)
                begin
                    pulse <= 1'b1;
                end
            else
                begin
                    pulse <= 1'b0;
                end
        end
    end
end
end
end
```

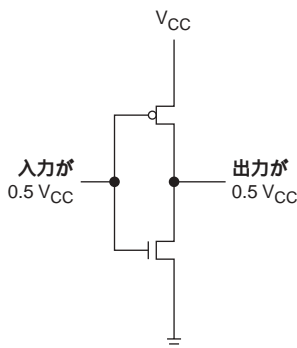
組み合わせオシレータ回路


図 11-29 に示す回路は、反転出力が同じゲートの入力の 1 つにフィードバックする組み合わせロジック・ゲートで構成されています。このフィードバック・パスによって出力の状態が変化するため発振します。

図 11-29. 組み合わせリング・オシレータ回路



この回路は、リング・オシレータとして知られる構造を持つ一連のカスケード接続インバータによって構築されることがあります。この回路が発振する周波数は、デバイスの温度、電圧、およびプロセス動作条件に依存し、デバイス内の他のどのクロック・ドメインに対しても完全に非同期です。さらに悪いことには、回路がまったく発振せず、図 11-30 に示すように、インバータの出力が供給電圧の半分の電圧で安定する場合があります。これにより、インバータ・チェーン内の PMOS および NMOS トランジスタの両方が、 V_{CC} から GND へのパスによって、同時に導通するため、インバータ機能はなく、スタティック電流を消費します。

図 11-30. $0.5 V_{CC}$ にバイアスされたインバータ

 すべての組み合わせフィードバック・オシレータ回路の実装を避けてください。

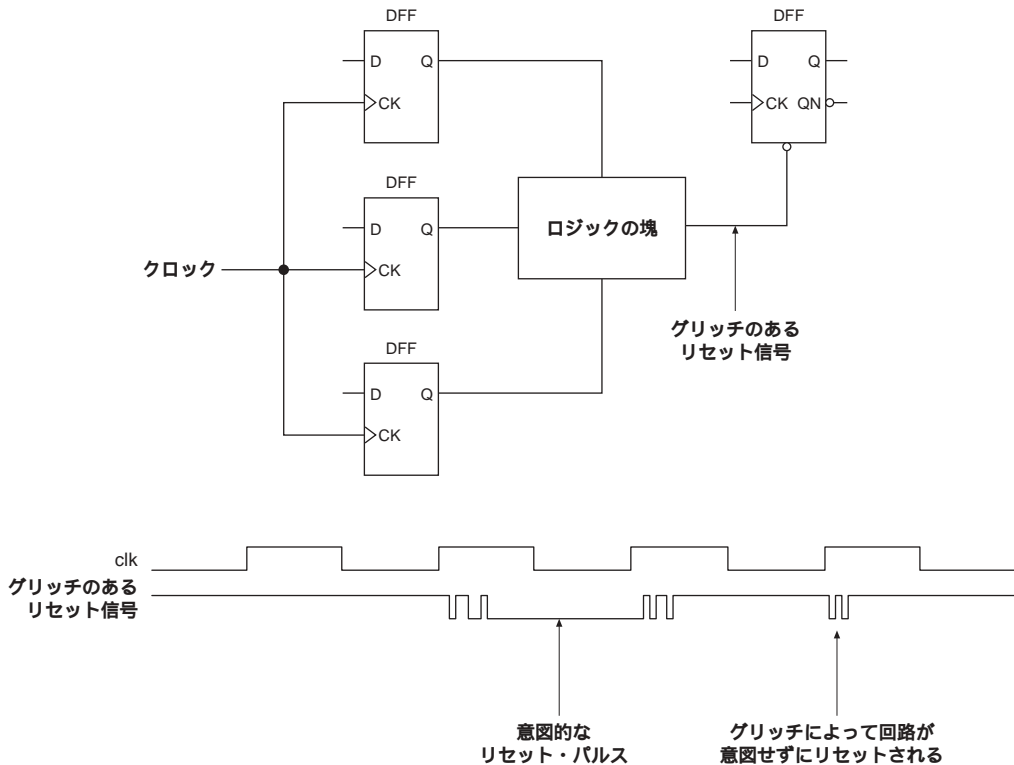
リセット回路

リセット信号は、同期的または非同期的にデザインのレジスタの状態に影響を与えるコントロール信号です。クロック信号に対する特別な配慮がリセット信号にも必要です。ここで使用する用語は「リセット」のみですが、ここに記載する情報は「セット」、「プリセット」、および「クリア」信号にも適用されます。リセット信号を使用するのは、回路を既知の初期条件の状態にする時だけにしてください。また、同じレジスタの set ピンと reset ピンの両方を使用することはできません。これらのピンをドライブする信号が同時にアクティブになると、レジスタのロジック状態が不確定になることがあります。

ゲート付きリセット

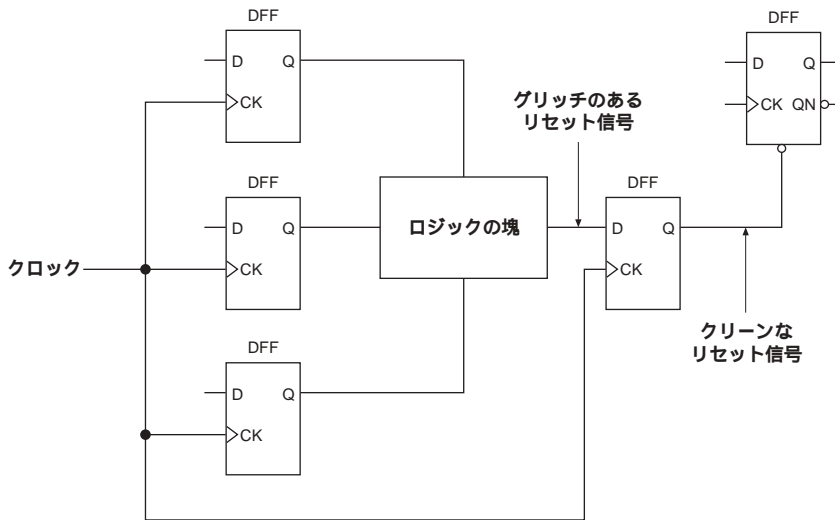
ゲート付きリセットは、組み合わせロジックがレジスタの非同期リセット・ピンに接続されると生成されます。ゲート付きリセット信号はグリッチを有する場合があります。宛先のレジスタを意図せずにリセットさせてしまう可能性があります。図 11-31 は、レジスタのリセット・ピンをドライブする信号にグリッチがあるため、意図しないリセットが発生するゲート付きリセット回路を示したものです。

図 11-31. ゲート付きリセット回路および関連するタイミング図



ゲート付きリセット回路を実装するより良いアプローチは、リセット・ゲーティング・ロジックの出力にレジスタを配置して、クロックに同期させることです。これにより、レジスタ出力は、デザインの残りの部分をドライブするグリッチのないリセット信号になります。ただし、得られるリセット信号は、追加されたクロック・サイクル分だけ遅延されます。

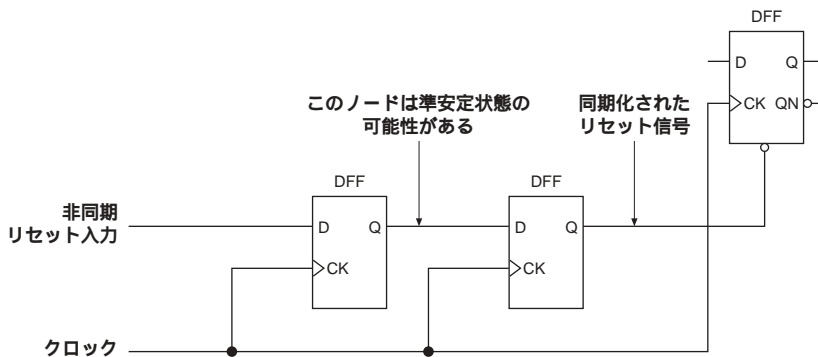
図 11-32. 図 11-31 のゲート付きリセット回路へのより良いアプローチ



非同期リセットの同期化

クロック信号がないときにデザインをリセット状態にする必要がある場合、これを実現する唯一の方法は非同期リセットの使用によるものです。ただし、図 11-33 に示すようなダブル・バッファ回路を使用して、非同期リセット信号から同期リセット信号を生成することもできます。

図 11-33. ダブル・バッファ回路



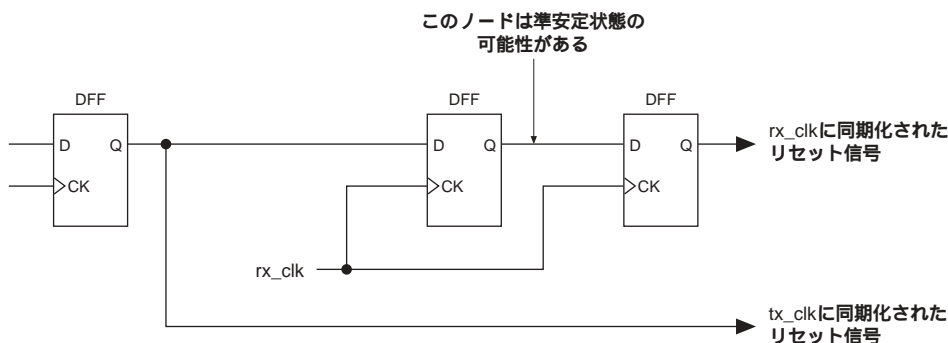
クロック・ドメイン間でのリセット信号の同期化

デザインでは、1つのクロック・ドメイン内で生成され、別の1つまたは複数の非同期クロック・ドメインで使用される、内部で生成されたリセット信号は同期化させる必要があります。同期化されていないリセット信号によって準安定状態の問題が生じる可能性があります。

ゲート付きリセットの同期化は、[図 11-34](#) に示すとおり、以下のガイドラインに従って行う必要があります。

- リセット信号を受信側の非同期クロック・ドメイン内の2つ以上のカスケード接続されたレジスタに同期させてください。
- カスケード接続されたレジスタは同じクロック・エッジでトリガされるようにしてください。
- 送信側のクロック・ドメインの出力と受信側の非同期クロック・ドメイン内のカスケード接続されたレジスタの間にロジックがないようにしてください。

図 11-34. 2つのクロック・ドメイン間で同期化されたリセット信号のための回路




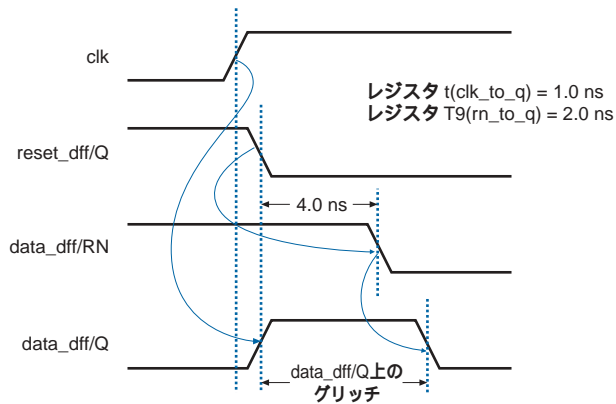
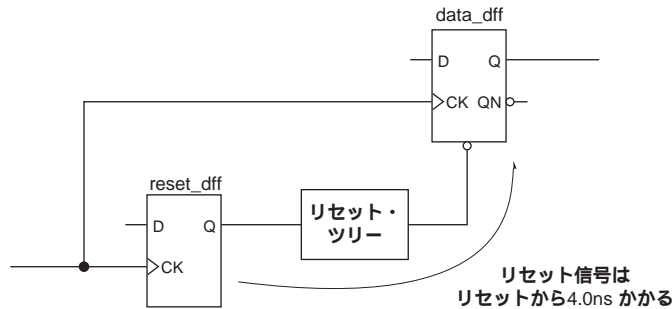

 [図 11-33](#) および [図 11-34](#) に示すリセット同期化回路のどちらも、リセットが印加されたとき、デザインのレジスタのQ出力が誤った信号を送信すると、瞬時に一部の主要出力ピンも誤った信号を送信してしまう可能性があります。[図 11-35](#) の回路および関連するタイミング図は、この現象を示したものです。

図 11-35. リセット同期化回路についての一般的な問題



純粋な同期リセット回路はこのような動作はしません。以下の Verilog RTL コードの一部分は、これを行う方法を示しています。

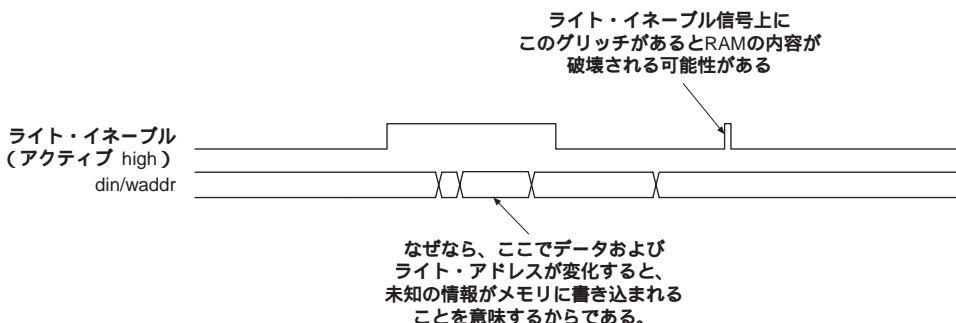
```
always @ (posedge clk)
begin
    if (!rst)
        q <= 1'b0;
    else
        q <= d;
end
```

 常に回路の初期化以外の目的でリセット信号を使用することを避け、リセット同期化回路が使用されている場合は、リセット信号のタイミングを把握するようにしてください。

非同期 RAM

アルテラの FPGA デバイスは、多数の異なるモードに構成可能な柔軟なエンベデッド・メモリ構造を備えています。可能なモードの 1 つに非同期 RAM があります。非同期 RAM 回路の定義は、[図 11-36](#) に示すとおり、RAM をドライブするライト・イネーブル信号によってデータが RAM に書き込まれ、クロックを必要としない回路です。これは、ライト・イネーブル信号にグリッチが存在する場合に、RAM が障害に影響されやすいことを意味します。また、RAM のデータおよびライト・アドレス・ポートはライト・パルスがアサートされる前に安定する必要があるため、ライト・パルスがデアサートされるまで安定していなければなりません。非同期モードでメモリ構造を使用する際のこれらの制限は、常に同期メモリが推奨されることを意味しています。同期メモリはデザイン性能も向上させます。

図 11-36. 非同期 RAM 構造使用時の潜在的な問題



常に同期 RAM 構造を使用し、できるだけレジスタ付き出力を持たせてください。

まとめ

ここで説明した問題のほとんどは、デザインが初期段階のうちに簡単に回避できます。これらの問題は HardCopy デバイスに適用するだけでなく、スタンダード・セルの ASIC、ゲート・アレイ、FPGA など、あらゆるデジタル・ロジック集積回路デザインに適用します。

上記のガイドラインの 1 つまたは複数に違反することが避けられない場合もありますが、その意味を理解することが非常に重要です。その場合、これらのルールを破る必要性を正当化できるようにし、できるだけ多くの文書によってそれを裏付ける必要があります。

ここで簡単に説明したガイドラインに従うことによって、最終的にデザインがより頑強なものになり、迅速に実装でき、デバッグしやすく、ターゲットのアーキテクチャに容易に適合できるようになるため、成功を収める可能性が高くなります。

