

この章では、Stratix® V コア・ファブリックのロジック・アレイ・ブロック (LAB) の機能について説明します。LAB は、ロジック・ファンクション、演算ファンクション、およびレジスタ・ファンクションを実装するようにコンフィギュレーションできるアダプティブ・ロジック・モジュール (ALM) から構成されます。LAB および ALM は、Stratix V デバイスの基本的なビルディング・ブロックです。ALM は効率的なロジック使用を可能にする最新機能を提供し、完全な下位互換性を備えています。

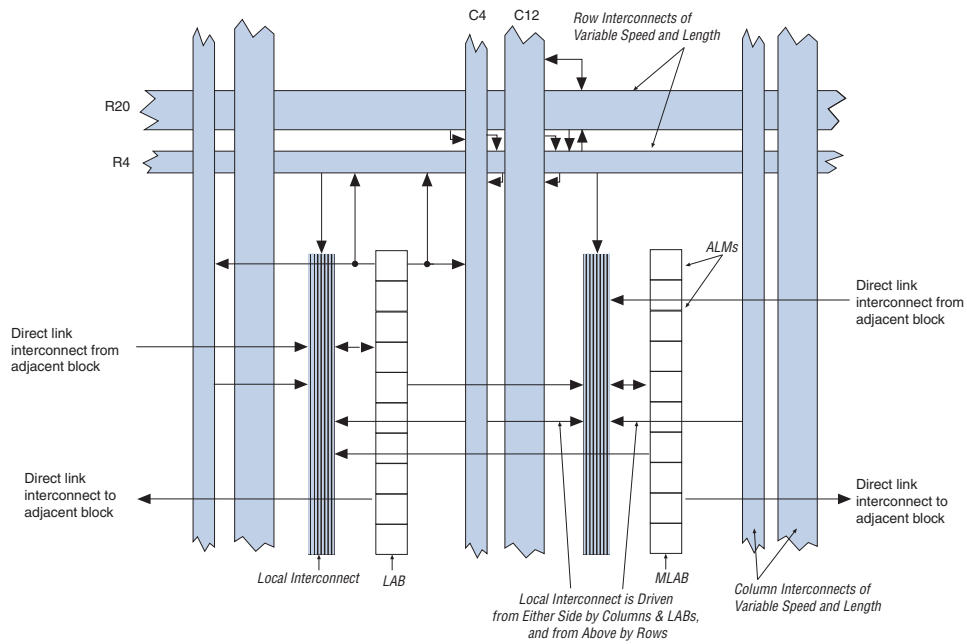
この章は、以下の項で構成されています。

- 1-1 ページの「ロジック・アレイ・ブロック」
- 1-5 ページの「アダプティブ・ロジック・モジュール」

ロジック・アレイ・ブロック

各 LAB は、10 個の ALM、多種のキャリー・チェーン、共有演算チェーン、LAB Stratix V コントロール信号、ローカル・インタコネクタ、およびレジスタ・チェーン接続ラインで構成されています。ローカル・インタコネクタは、同一 LAB 内で ALM 間の信号を転送します。ダイレクト・リンク・インタコネクタにより、LAB は左または右に隣接するローカル・インタコネクタをドライブできます。レジスタ・チェーン接続は、ALM レジスタの出力を LAB 内の隣接する ALM レジスタに転送します。Quartus® II コンパイラは LAB または隣接 LAB 内に関連ロジックを生成し、ローカル接続、共有演算チェーン接続、およびレジスタ・チェーン接続の使用を可能にして、性能と面積効率を高めます。図 1-1 に、Stratix V の LAB 構造および LAB インタコネクタを示します。

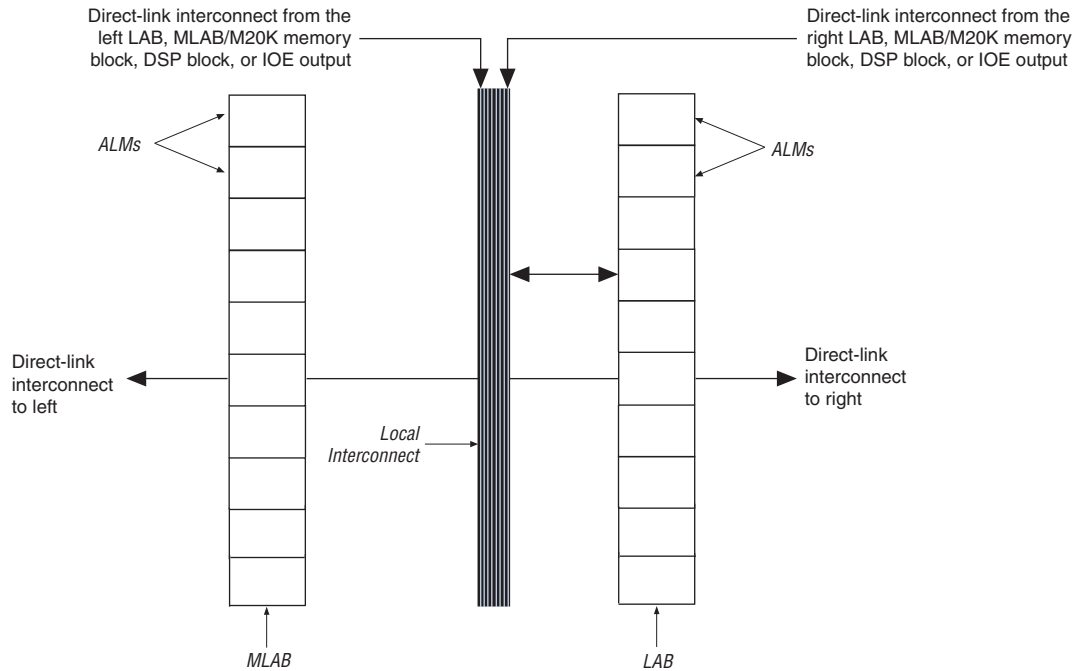
図 1-1. Stratix V デバイスの LAB 構造



メモリ LAB (MLAB) は Stratix V LAB の微分です。図 1-2 に示すように、MLAB はルック・アップ・テーブル (LUT) ベースの SRAM 機能を LAB に追加します。MLAB は、最大 640 ビットのシンプル・デュアル・ポート SRAM (スタティック・ランダム・アクセス・メモリ) をサポートします。MLAB の各 ALM は、 64×1 または 32×2 ブロックとして設定可能で、それぞれ 64×10 または 32×20 のシンプル・デュアル・ポート SRAM ブロックというコンフィギュレーションにさせます。すべての Stratix V ファミリでは、MLAB および LAB ブロックは常にペアで共存します。MLAB は LAB のスーパーセットで、LAB の機能をすべて備えています。

図 1-3 に、ダイレクト・リンク接続を示します。

図 1-3. ダイレクト・リンク接続



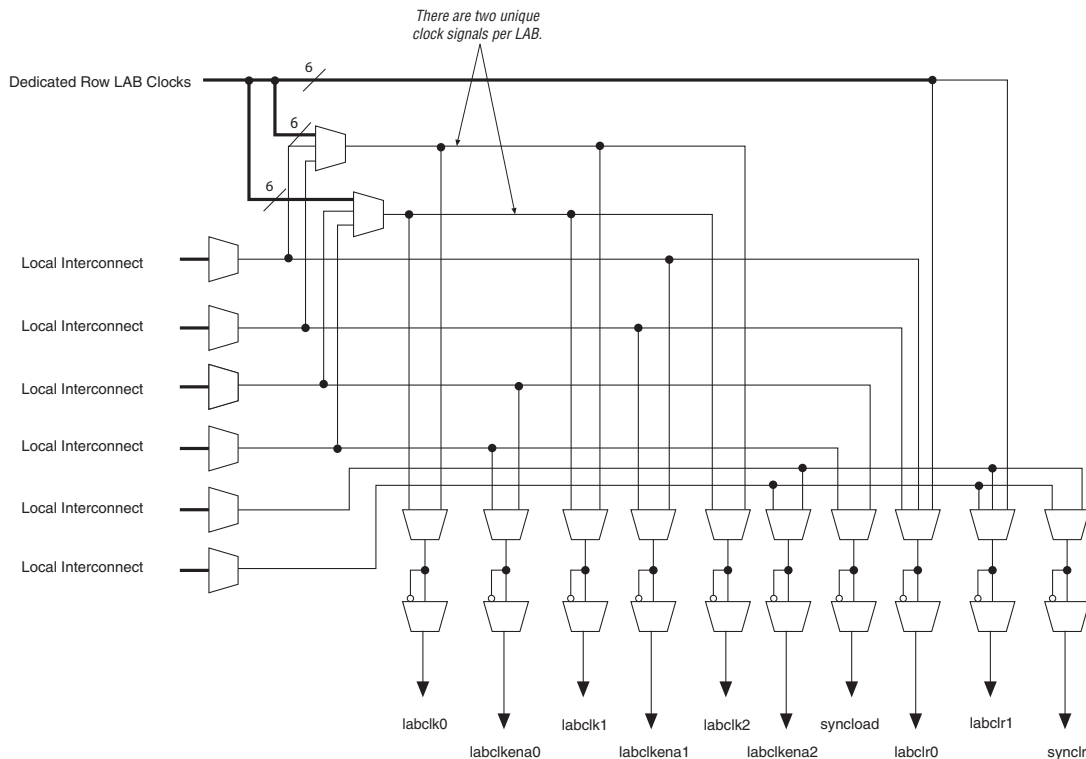
LAB コントロール信号

各 LAB には、ALM にコントロール信号をドライブするための専用ロジックが内蔵されています。一度に最大 10 本のコントロール信号に対して、このコントロール信号には、3 本のクロック、3 本のクロック・イネーブル、2 本の非同期クリア、同期クリア、および同期ロードが含まれます。一般に同期ロード信号および同期クリア信号は、カウンタを実装する際に使用されますが、他のファンクションでも使用できます。

図 1-4 に示すように、各 LAB には、2 本の固有のクロック・ソースおよび 3 本のクロック・イネーブル信号があります。LAB コントロール・ブロックは、2 本のクロック・ソースと 3 本のクロック・イネーブル信号を使用して、最大 3 本のクロックを生成することができます。各 LAB のクロック信号とクロック・イネーブル信号はリンクされています。例えば、labclk1 信号を使用する特定の LAB の ALM は、labclkena1 信号も使用します。クロックの立ち上がり立ち下がりの双方のエッジが LAB 内で使用される場合、LAB ワイドのクロック信号を 2 本とも使用されます。クロック・イネーブル信号がディアサートされると、対応する LAB ワイドのクロック信号はオフになります。

LAB ロー・クロック [5..0] および LAB ローカル・インタコネクトは、LAB ワイド・コントロール信号を生成します。MultiTrack インタコネクトに固有の低スキューにより、データの他にクロックとコントロール信号を分配することができます。MultiTrack インタコネクトは、デザイン・ブロック間およびデザイン・ブロック内の接続に使用される長さや速度が異なる最適性能の連続配線ラインで構成されます。

図 1-4. LAB ワイド・コントロール信号



アダプティブ・ロジック・モジュール

Stratix V アーキテクチャのロジックの基本的なビルディング・ブロックは ALM です。ALM は効率的なロジック利用を可能にする最新機能を提供します。各 ALM には、2 つの組み合わせアダプティブ LUT (ALUT) および 2 個のレジスタ間で分割できる多様な LUT ベースのリソースが含まれています。2 個の組み合わせ ALUT への最大 8 本の入力により、1 個の ALM で 2 つのファンクションの様々な組み合わせを実装できます。この適応性により、ALM は 4 入力 LUT アーキテクチャとの完全な下位互換性を提供します。1 個の ALM で、最大 6 本の入力を持つ任意のファンクションおよび特定の 7 入力ファンクションを実装することも可能です。

アダプティブ LUT ベースのリソースに加えて、各 ALM には 2 個のプログラマブル・レジスタ、2 個の専用の全加算器、1 本のキャリー・チェーン、1 本の共有演算チェーン、および 1 本のレジスタ・チェーンも含まれています。これらの専用リソースにより、ALM は様々な演算ファンクションやシフト・レジスタを効率的に実装することができます。各 ALM は、ローカル、ロウ、カラム、キャリー・チェーン、共有演算チェーン、レジスタ・チェーン、およびダイレクト・リンクを含むあらゆるタイプのインタコネクトをドライブします。図 1-5 に、Stratix V ALM の上位レベルのブロック図を示します。

図 1-5. Stratix V ALM の上位レベルのブロック図

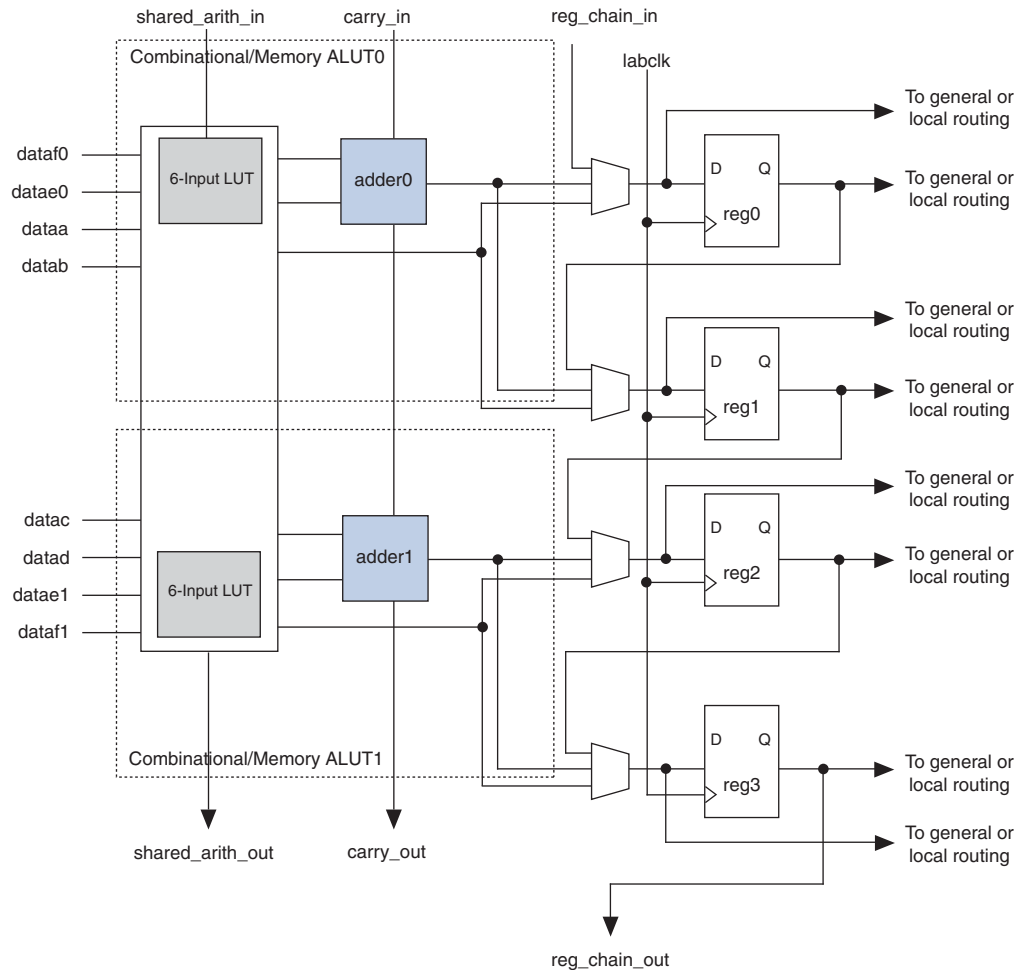
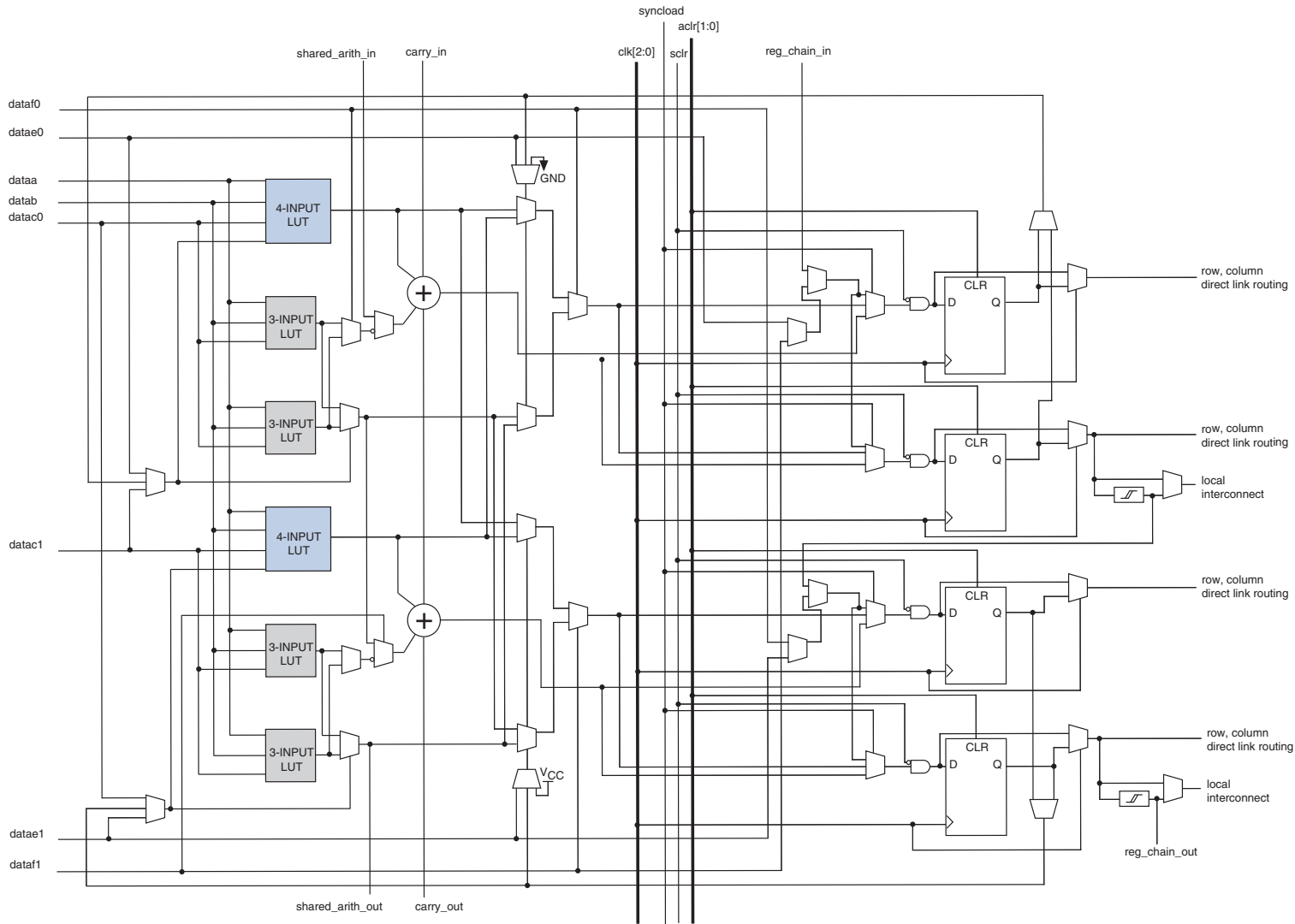


図 1-6 に ALM 内のすべての接続の詳細図を示します。

図 1-6. Stratix V デバイスの ALM 接続の詳細



1 個の ALM には 2 個のプログラマブル・レジスタが含まれています。各レジスタには、データ、クロック、クロック・イネーブル、同期/非同期クリア、および同期ロード/クリアの各入力があります。グローバル信号、汎用 I/O ピン、または任意の内部ロジックでレジスタのクロック・コントロール信号とクリア・コントロール信号をドライブすることができます。汎用 I/O ピンまたは内部ロジックのいずれかが、クロック・イネーブルをドライブできます。組み合わせファンクションを実現するときには、レジスタがバイパスされ、LUT の出力が ALM の出力を直接ドライブします。

各 ALM には、ローカル、ロウ、およびカラム配線リソースをドライブする 2 セットの出力があります。LUT、加算器、またはレジスタ出力は、これらの出力をドライブできます (図 1-6 を参照)。出力ドライバの各セットについて、2 本の ALM 出力がカラム、ロウ、またはダイレクト・リンク配線接続をドライブできます。これらの ALM 出力の 1 本はローカル・インタコネクタ・リソースもドライブできます。これにより、レジスタがある出力をドライブしている状態で、LUT が別の出力をドライブすることが可能になります。

この機能はレジスタ・パッキングと呼ばれ、デバイスの稼働率を向上させます。これはレジスタと組み合わせロジックを全く別の機能として使用できるからです。別の特殊パッキング・モードでは、レジスタ出力を同一 ALM の LUT にフィードバックさせて、レジスタに独自のファンアウト LUT をパッキングすることができます。これにより、フィッティング機能を向上させる別のメカニズムが実現します。また、ALM はラッチされた出力およびラッチされていない出力の両方の LUT 出力もドライブ・アウト可能です。

ALM 動作モード

Stratix V ALM は、次のいずれかのモードで動作することができます。

- 1-9 ページの「ノーマル・モード」
- 1-10 ページの「拡張 LUT モード」
- 1-11 ページの「演算モード」
- 1-13 ページの「共有演算モード」

各モードでは、ALM のリソースがそれぞれ異なる形で使用されます。各モードで、LAB ローカル・インタコネクタからの 8 本のデータ入力、前の ALM または LAB からの carry-in、前の ALM または LAB からの共有演算チェーン接続、およびレジスタ・チェーン接続の 11 本の ALM への入力が異なるデスティネーションに転送され、目的のロジック・ファンクションを実装します。LAB ワイドの信号として供給可能なものは、レジスタへのクロック、非同期クリア、同期クリア、同期ロード、およびクロック・イネーブル・コントロールの各信号です。このような LAB ワイドの信号は、すべての ALM モードで使用できます。

LAB ワイド・コントロール信号について詳しくは、1-4 ページの「LAB コントロール信号」を参照してください。

Quartus II ソフトウェアおよびサポートされるサードパーティの合成ツールは、LPM (Library of Parameterized Modules) などのパラメータ化されたファンクションと併用することによって、カウンタ、加算器、減算器、および演算ファンクションなどの一般的なファンクションに対して適切なモードを自動的に選択します。

ノーマル・モード

ノーマル・モードは、汎用のロジック・アプリケーションや組み合わせファンクションに適しています。このモードでは、LAB ローカル・インタコネクタからの最大 8 本のデータ入力組み合わせロジックの入力になります。ノーマル・モードでは、1 個の Stratix V ALM で 2 つのファンクション、または 1 個の ALM で最大 6 本の入力を持つ 1 つのファンクションを実装できます。ALM は、完全に独立したファンクションの特定の組み合わせおよび共通の入力を持つファンクションの様々な組み合わせをサポートできます。

図 1-7 は、ノーマル・モードでの LUT の組み合わせを示しています。

図 1-7. ノーマル・モードの ALM (注 1)

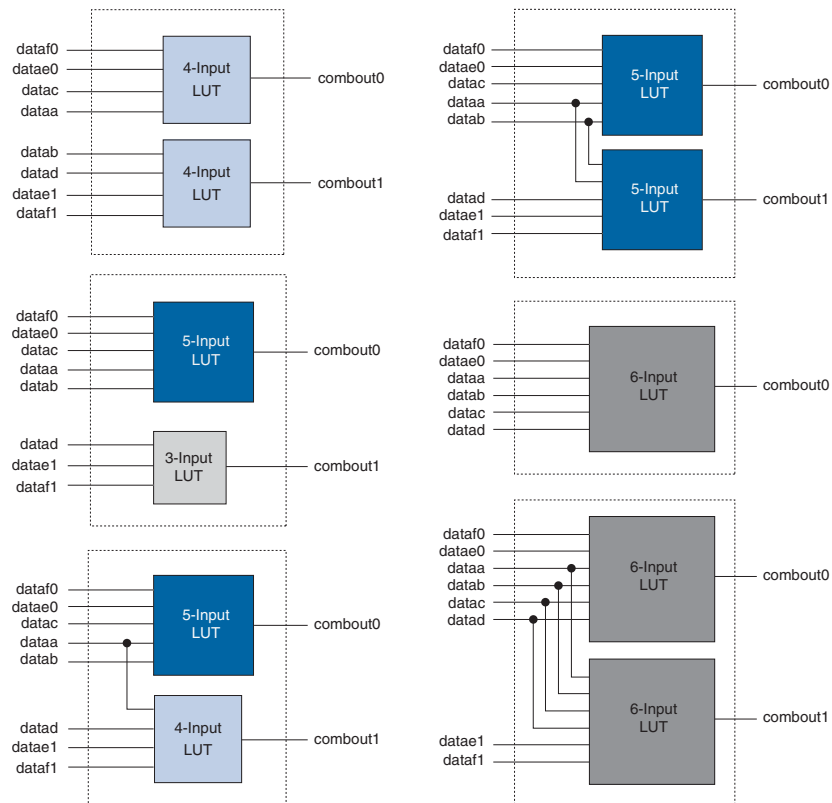


図 1-7 の注 :

- (1) ここに示したものより入力数が少ないファンクションの組み合わせもサポートされています。例えば、以下の入力数を持つファンクションの組み合わせがサポートされます。それは 4 と 3、3 と 3、3 と 2、5 と 2 です。

ノーマル・モードでは、4 入力 LUT アーキテクチャとの完全な下位互換性が提供されます。

2 つの 5 入力ファンクションを 1 個の ALM 内にパッキングするには、これらのファンクションに少なくとも 2 本の共通入力を持たせる必要があります。共通入力は、dataa および datab です。4 入力ファンクションと 5 入力ファンクションの組み合わせには、1 本の共通入力 (dataa または datab のいずれか) が必要です。

1 個の ALM に 2 つの 6 入力ファンクションを実装する場合は、4 本の入力を共有させる必要があります。また組み合わせファンクションは同じでなければなりません。ALM の使用頻度の低いデバイスでは、Quartus II ソフトウェアを使用して 1 個の ALM 内に配置可能なファンクションを別の ALM に実装して、最高性能を達成することができます。デバイスの使用率が高くなり始めると、Quartus II ソフトウェアは自動的に Stratix V ALM を最大限に活用します。Quartus II のコンパイラは、共通入力を使用するファンクションまたは完全に独立したファンクションを自動的にサーチし、それらを 1 つの ALM に配置してデバイス・リソースを効率的に使用します。さらに、位置の割り当てを設定することにより、リソース使用量を手動でコントロール可能です。

dataa、datab、datac、datad、および datae0 と dataf0 または datae1 と dataf1 の入力を利用して、任意の 6 入力ファンクションを実装できます。datae0 と dataf0 を使用する場合、出力は register0 にドライブされるか、または register0 がバイパスされる、または出力が register0 にドライブされて register0 がバイパスされる、そして、そのデータが出力ドライバのトップ・セットを使用してインタコネクต์に出力されます (図 1-8 を参照)。datae1 と dataf1 を使用する場合、出力は register1 にドライブされるか、または register1 をバイパスし出力ドライバのボトム・セットを使用してインタコネクต์にドライブされます。Quartus II のコンパイラは、LUT への入力を自動的に選択します。ノーマル・モードの ALM は、レジスタ・パッキングの機能をサポートします。

図 1-8. ノーマル・モードの入力ファンクション (注 1)

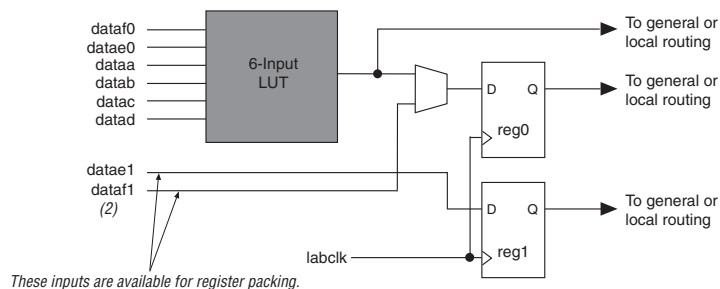


図 1-8 の注:

- (1) datae1 と dataf1 が 6 入力ファンクションの入力として使用される場合、datae0 と dataf0 はレジスタ・パッキングに使用できます。
- (2) 6 入力ファンクションがラッチされない場合に限り、dataf1 入力はレジスタ・パッキングに使用できます。

拡張 LUT モード

拡張 LUT モードを使用して、特定の 7 入力ファンクションのセットを実装します。このセットは、4 入力を共有する任意の 2 つの 5 入力ファンクションから信号が供給される 2 対 1 マルチプレクサでなければなりません。図 1-9 に、拡張 LUT モードを使用してサポートされる 7 入力ファンクションのテンプレートを示します。このモードでは、7 入力ファンクションがラッチされない場合は、未使用の 8 番目の入力をレジスタ・パッキングに使用できます。

図 1-9 に示すテンプレートに適合するファンクションは、デザインで自然に生じます。これらのファンクションは多くの場合、デザインに Verilog HDL または VHDL コードの「if-else」文として現れます。

図 1-9. 拡張 LUT モードでサポートされる 7 入力ファンクションのテンプレート

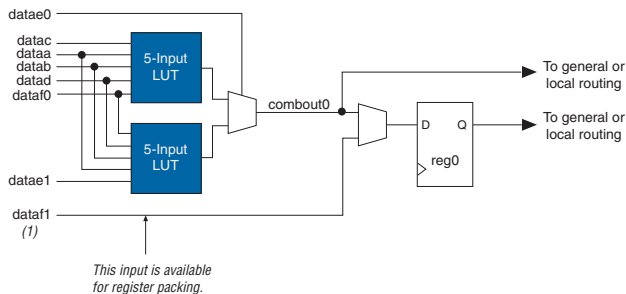


図 1-9 の注 :

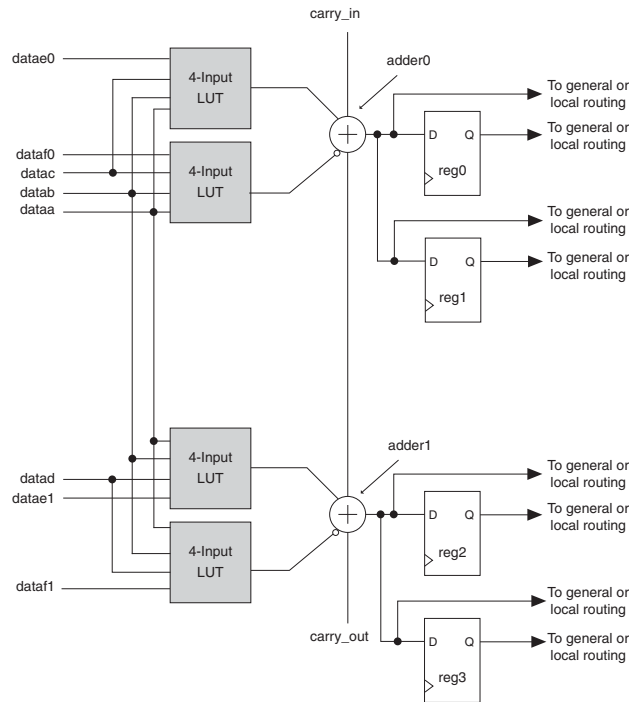
- (1) 7 入力ファンクションがラッチされない場合は、未使用の 8 番目の入力をレジスタ・パッキングに使用できます。第 2 のレジスタ reg1 は使用できません。

演算モード

演算モードは、加算器、カウンタ、乗算累積器、幅広いパリティ・ファンクション、およびコンパレータの構成に最適です。演算モードの ALM は、2 個の専用全加算器と共に 2 個の 4 入力 LUT を 2 組使用します。専用加算器によって、LUT は加算器前ロジックを実行できるため、各加算器は 2 つの 4 入力ファンクションの出力を加算することができます。

4 個の LUT は、dataa および datab 入力を共有します。図 1-10 に示すように、キャリー・イン信号は adder0 に供給され、adder0 からのキャリー・アウト信号は adder1 の carry-in に供給されます。adder1 からのキャリー・アウト信号は、LAB 内の次の ALM の adder0 にドライブされます。また、演算モードの ALM では、ラッチされた加算器出力とラッチされていない加算器出力のいずれか一方、または両方をドライブ・アウトできます。

図 1-10. 演算モードの ALM



演算モードで動作している間、ALM は組み合わせロジックの出力と加算器のキャリ出力の同時使用をサポートできます。この動作では加算器の出力は無視されます。このように加算器と組み合わせロジックの出力を併用すると、この機能を使用可能なファンクションのリソースが最大 50% 節約されます。

演算モードではクロック・イネーブル、カウンタ・イネーブル、同期アップ/ダウン・コントロール、加算/減算コントロール、同期クリアおよび同期ロードの各信号も提供されています。クロック・イネーブル、カウンタ・イネーブル、同期アップ/ダウン・コントロール、および加算/減算コントロール各信号は、LAB ローカル・インタコネクタからのデータ入力により生成されます。これらのコントロール信号は、ALM 内の 4 個の LUT 間で共有される入力の候補として適当です。同期クリアと同期ロードのオプション信号は、LAB ワイドの信号であるため、LAB 内のすべてのレジスタに影響を与えます。これらの信号は、レジスタごとに個別にディセーブルまたはイネーブルできます。Quartus II ソフトウェアは、カウンタに使用されていないレジスタを自動的に他の LAB に配置します。

キャリー・チェーン

演算モードまたは共有演算モードにおいて、キャリー・チェーンは、専用加算器間でのキャリー・ファンクションを高速化します。Stratix V デバイスの 2 ビット・キャリー選択機能は、ALM 内でキャリー・チェーンの伝播遅延を半減します。キャリー・チェーンは、LAB 内の最初の ALM または 5 番目の ALM のどちらからも開始できます。最後のキャリー・アウト信号は ALM に接続され、そこでローカル、ロウ、カラムのいずれかのインタコネクタに供給されます。

Quartus II Compiler は、デザイン処理中にキャリー・チェーン・ロジックを自動的に作成しますが、ユーザーがデザインの入力時に手動で作成することもできます。LPM ファンクションなどのパラメータ化されたファンクションは、キャリー・チェーンの利点を自動的に活用して、適切な機能を実現します。

Quartus II Compiler は、複数の LAB を自動的にリンクさせることにより、20 個（演算モードまたは共有演算モードでは 10 個）を超える ALM で構成される長いキャリー・チェーンを作成します。フィッティング機能を強化するため、長いキャリー・チェーンは垂直に並べ、MLAB/M20K メモリおよび DSP ブロックへの水平方向の接続を高速化することができます。キャリー・チェーンはカラム全体に延長できます。

高ファンイン演算ファンクションが実装されたときにデバイス内の 1 つの小さな領域で配線が密集するのを防ぐために、LAB は次の LAB に接続する前に LAB の上半分または下半分のいずれかのみを使用するキャリー・チェーンをサポートできます。これにより、LAB 内の ALM の別の半分をノーマル・モードでより幅の狭いファンイン・ファンクションの実装に使用できます。最初の LAB 内の上位 5 個の ALM を使用するキャリー・チェーンは、カラム内で次の LAB 内の ALM の上半分に取り込みます。最初の LAB 内の下位 5 個の ALM を使用するキャリー・チェーンは、カラム内で次の LAB 内の ALM の下半分に取り込みます。LAB カラムは 1 つおきに上半分がバイパス可能で、他の LAB カラムは下半分がバイパス可能です。

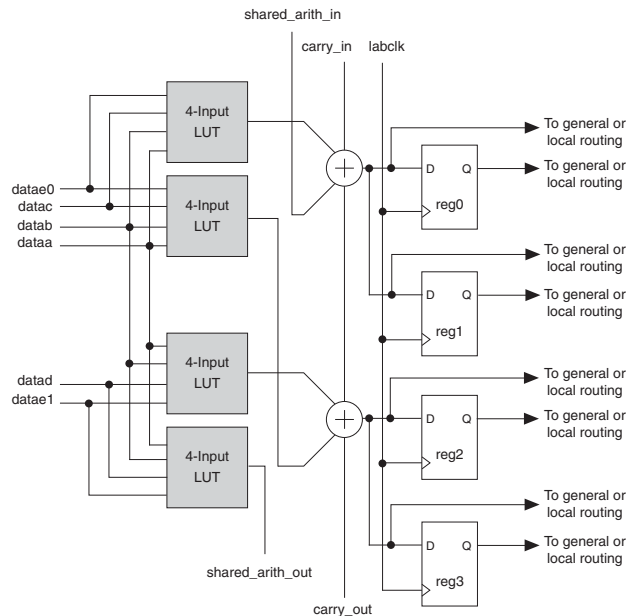
キャリー・チェーン・インタコネクタについて詳しくは、1-16 ページの「ALM インタコネクタ」を参照してください。

共有演算モード

共有演算モードでは、ALM で 3 入力加算を実装できます。このモードでは、ALM は 4 個の 4 入力 LUT で構成されます。各 LUT は、3 本の入力の和または 3 本の入力のキャリーのいずれかを計算します。キャリー計算の出力は、共有演算チェーンと呼ぶ専用の接続を使用して、次の加算器（同じ ALM の adder1 または LAB 内の次の ALM の adder0）に供給されます。この共有演算チェーンは、加算器ツリーの実装に必要なサメーション・ステージの数を減らすことによって、加算器ツリーの性能を大幅に向上させることができます。

図 1-11 に、共有演算モードの ALM を示します。

図 1-11. 共有演算モードの ALM



加算器ツリーは様々なアプリケーションで使用されています。例えば、ロジック・ベースの乗算器での部分積の合計をツリー構造で実装することができます。別の例は、スペクトラム拡散テクノロジーを使用して送信されたデータを回復またはデスペクトルするため、大きな加算器ツリーを使用して一定時間内のフィルタされたデータ・サンプルの和をとることができる相関器ファンクションです。

共有演算チェーン

拡張演算モードで使用可能な共有演算チェーンは ALM による 3 入力加算の実装を可能にします。これにより、大きな加算器ツリーまたは相関器ファンクションを実装するのに必要なリソースが大幅に削減されます。

共有演算チェーンは LAB 内の最初の ALM または 6 番目の ALM のいずれでも開始できます。Quartus II Compiler は、複数の LAB を自動的にリンクさせることにより、20 個以上（演算モードまたは共有演算モードでは 10 個の ALM）で構成される長い共有演算チェーンを作成します。フィッティング機能を強化するため、長い共有演算チェーンは垂直に並べ、MLAB/M20K メモリおよび DSP ブロックへの水平方向の接続を高速化することができます。共有演算チェーンはカラム全体に延長できます。

キャリー・チェーンと同様に、LAB カラムは 1 つおきに共有演算チェーンも上半分または下半分をバイパス可能です。この機能により、共有演算チェーンを LAB 内の ALM の半分でカスケード接続し、別の半分を幅の狭いファンイン・ファンクションに使用できます。他の LAB カラムの下半分がバイパス可能ですが、一つおきの LAB カラムの上半分がバイパス可能です。

共有演算チェーン・インタコネクトについて詳しくは、1-16 ページの「ALM インタコネクト」を参照してください。

レジスタ・チェーン

一般配線出力に加えて、LAB 内の ALM にはレジスタ・チェーン出力があります。レジスタ・チェーン配線により、同一 LAB 内のレジスタをカスケード接続できます。レジスタ・チェーン・インタコネクタにより、LAB は LUT を 1 つの組み合わせファンクションに使用しつつ、レジスタを別のシフト・レジスタの実装に使用することができます。これらのリソースは ALM 間の接続を高速化し、同時にローカル・インタコネクタ・リソースの節約を図ります (図 1-12 を参照)。Quartus II Compiler は自動的にこれらのリソースを活用して、稼働率とパフォーマンスの向上を図ります。

図 1-12. LAB 内のレジスタ・チェーン (注 1)

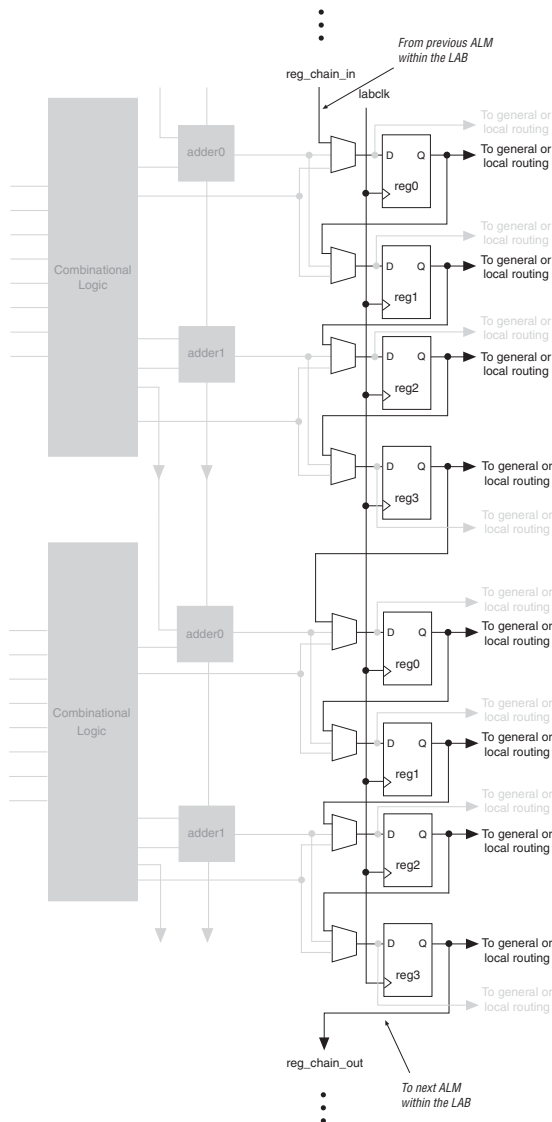


図 1-12 の注 :

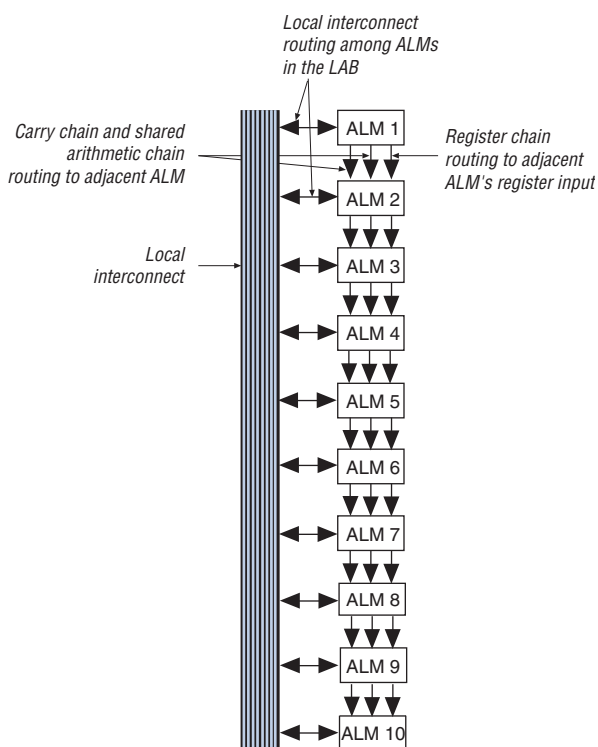
- (1) 組み合わせロジックまたはアダー・ロジックを使用して、独立したラッチされないファンクションを実装できます。

レジスタ・チェーン・インタコネクタについて詳しくは、1-16 ページの「ALM インタコネクタ」を参照してください。

ALM インタコネク

ALM の間には、レジスタ・カスケード、キャリア・チェーン、および共有演算チェーンの3つの専用パスがあります。Stratix V デバイスは LAB 内部のインタコネク構造を拡張し、共有演算チェーンおよびキャリア・チェーンを配線して効率的な演算ファンクションを実現します。レジスタ・チェーン接続により、1つの ALM のレジスタ出力を LAB 内の次の ALM のレジスタ入力に直接接続し、高速シフト・レジスタを実現できます。これらの ALM 間の接続はローカル・インタコネクをバイパスします。Quartus II Compiler は自動的にこれらのリソースを活用して、稼働率とパフォーマンスの向上を図ります。図 1-13 に、共有演算チェーン、キャリア・チェーン、およびレジスタ・チェーンのインタコネクを示します。

図 1-13. 共有演算チェーン、キャリア・チェーン、およびレジスタ・チェーンのインタコネク



クリアおよびプリセット・ロジック・コントロール


レジスタのクリア信号を実現するロジックは、LAB ワイド信号によって制御されません。ALM は非同期クリア機能を直接サポートします。Quartus II ソフトウェアの **NOT-gate push-back logic** オプションを使用して、レジスタ・プリセットを実現できます。各 LAB は最大2本のクリアをサポートします。

Stratix V デバイスは、デバイス内のすべてのレジスタをリセットするデバイス・ワイドのリセット・ピン (DEV_CLRn) を備えています。このピンは、Quartus II ソフトウェアでコンパイルを行う前に設定されたオプションによって制御されます。このデバイス・ワイドのリセット信号は、他のすべてのコントロール信号よりも優先されます。

LAB 消費電力管理手法

以下の手法を使用して、LAB 内のスタティックおよびダイナミック消費電力を管理します。

- AC 電力を節約するために、Quartus II は ALM 加算器が使用されていないときは、すべての加算器入力を Low にします。
- Stratix V LAB は、高性能モードまたは低消費電力モードで動作します。Quartus II ソフトウェアは、速度とリークのトレードオフを最適化するために、デザインに基づいて LAB に適切なモードを自動的に選択します。
- クロックは、高いスイッチング動作と長いパスのために、ダイナミック消費電力の大きな部分に関係します。クロック信号を LAB 内のレジスタに分配する LAB クロックは、クロックの全消費電力の多くの部分に関係します。各 LAB のクロック信号とクロック・イネーブル信号はリンクされています。例えば、labclk1 信号を使用する特定の LAB 内の組み合わせ ALUT またはレジスタは、labclk1 信号も使用します。クロック・ツリー全体をディセーブルしないで LAB ワイド・クロックの電力消費をディセーブルするため、LAB ワイド・クロック・イネーブルを使用して LAB ワイド・クロックをゲートします。Quartus II ソフトウェアは、レジスタ・レベルのクロック・イネーブル信号を自動的に LAB レベルに昇格させます。共通クロックおよびクロック・イネーブルを共有する LAB 内のすべてのレジスタは、共有ゲート・クロックで制御されます。これらのクロック・イネーブルを利用するには、HDL コード内でレジスタに対するクロック・イネーブル構造を使用します。

 LAB 内のスタティック消費電力およびダイナミック消費電力の実装について詳しくは、「Quartus II ハンドブック Volume 2」の「*Power Optimization*」の章を参照してください。

改訂履歴

表 1-1 に、本資料の改訂履歴を示します。

表 1-1. 改訂履歴

日付	バージョン	変更内容
2010 年 12 月	1.1	Quartus II ソフトウェア 10.1 に対して、この章の内容に変更はありません。
2010 年 7 月	1.0	初版

