XAPP496 (v1.0) 2010 年 6 月 3 日

複数の Spartan-6 FPGA のメモリ コントローラー ブロックによるバス幅の広いメモリ インターフェイスの作成

著者 : Derek Curd

はじめに

メモリ コントローラー ブロック (MCB) は、Spartan-6 デバイスと DDR3、DDR2、DDR、LPDDR メモリ間の接続を大幅に簡略化する内臓された専用のマルチポート メモリ コントローラーです。Spartan®-6 デバイスには、4、8、または 16 ビット メモリへのコンポーネント インターフェイスを実装できる MCB が 2 ～ 4 個搭載されています。メモリ帯域幅または集積度の要件がより厳しいアプリケーションには、MCB 1 個で提供できる 16 ビットを超えるメモリ インターフェイスを使用します。このアプリケーション ノートでは、2 個またはそれ以上の MCB の動作を結合して 32 ビット以上の有効なメモリ インターフェイスを実装する方法を説明します。各 MCB は、シングル ポート コンフィギュレーション モードにする必要があります。このアプリケーション ノートおよびリファレンス デザインは、マルチポート コンフィギュレーション モードの MCB の結合には対応していません。関連するリファレンス デザインは、ハードウェア検証済みで、パフォーマンスおよびデバイス使用率の両観点から解析されています。

概要

MCB は、最も低いコストと消費電力を実現する汎用 SDRAM メモリ規格へのインターフェイスを提供し、大部分の Spartan-6 FPGA アプリケーションで求められるメモリ インターフェイス要件に対応します。MCB によって、これらメモリ デバイスとのやりとりにおける複雑さが緩和され、その他の FPGA ユーザー ロジックへのシングル データ レート (SDR) ユーザー インターフェイスとなります。図 1 に示すように、MCB は最大 6 つのポートを使用できるマルチポート メモリ コントローラーです。各ポートは、コマンド インターフェイスおよび読み出し/書き込みデータ インターフェイスで構成されています。

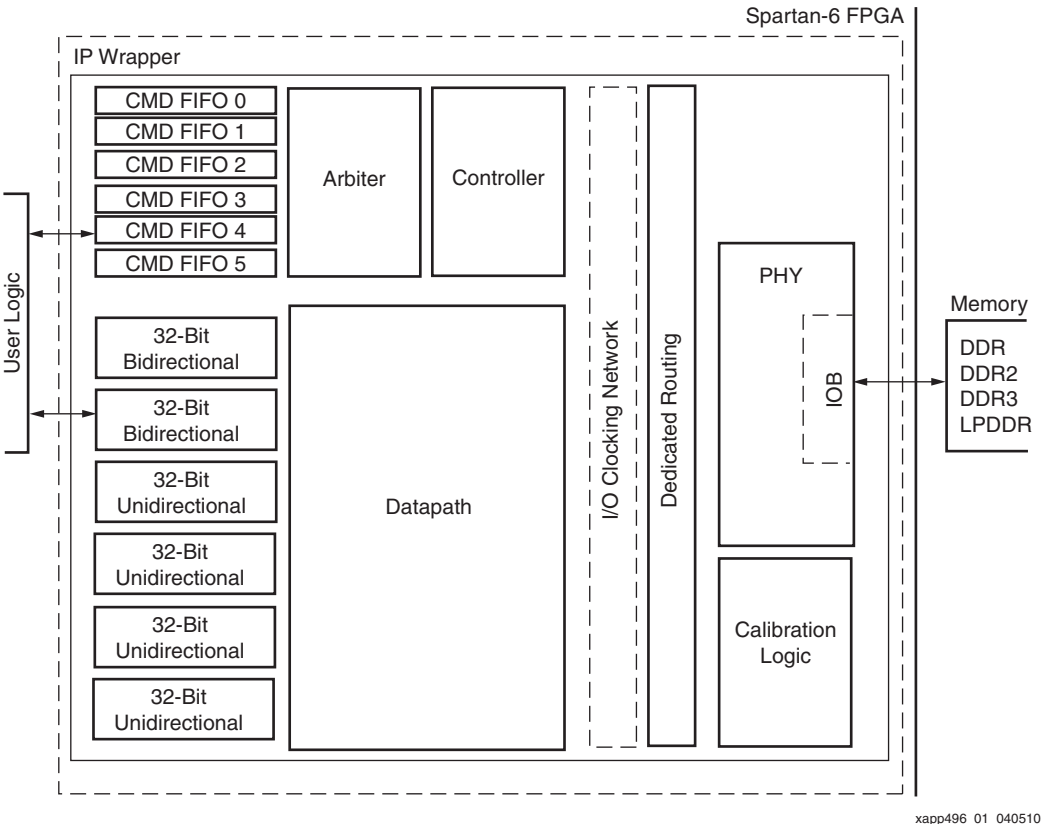

図 1 : MCB ブロック図

MCB が元々備える 2 つの 32 ビット双方向ポートおよび 4 つの 32ビット単一方向ポートを 5 とおりのポート コンフィギュレーションで組み合わせて、さまざまなビット幅のユーザー インターフェイスを作成できます。たとえば、4 つの 32 ビット双方向ポート、2 つの 64 ビット双方向ポート、あるいは 1 つの 128 ビット双方向ポートなどが可能です。マルチポート コンフィギュレーションを使用する場合、MCB 内部のアービタが外部メモリ デバイスへアクセスするポートを適宜決定します。

MCB に基づく外部メモリ インターフェイスの作成には、メモリ インターフェイス ジェネレーター (MIG) ツールを使用します。このツールは、CORE Generator™ アプリケーションから起動する GUI ウィザードで、MCB のコンフィギュレーションに必要な一連の手順を誘導します。MCB メモリ インターフェイスのシミュレーションやインプリメンテーションに必要な RTL コード、ユーザー制約ファイル (UCF)、およびスクリプト ファイルは自動的に生成されます。

MCB は最大 4Gb のメモリ集積度、最大 800Mb/s のデータ レート (DDR2、DDR3 SDRAM) に対応し、サポートする中で最も広い 16 ビットのシングル コンポーネント メモリ デバイスへ接続する際は最大 2.8Gb/s の帯域幅を実現します。つまり、MCB は大部分の Spartan-6 FPGA アプリケーションに十分なメモリ集積度および帯域幅を提供します。

システムによっては、1 つの MCB でサポートできないメモリ インターフェイス コンフィギュレーションが必要な場合があります。このアプリケーション ノートでは、2 つ (またはそれ以上) の MCB シングル コンポーネント インターフェイスそれぞれに MIG ツールで生成された RTL コードを使用する方法に加えて、これらのインターフェイスを結合して高集積度および広帯域幅のメモリに対応するバス幅の広いインターフェイスを作成する方法についても説明します。

デザインの概要

このアプリケーション ノートで説明する「リファレンス デザイン」は、シングル ポートモードの MCB を複数結合する場合にのみ対応するものです。つまり、それぞれの MCB 内でアービトレーションは行われません。各 MCB に別々のアービタが存在すると複雑さが増すため、マルチポート モードの MCB の結合は推奨しません。マルチポート インターフェイスが必要な場合は、この資料の手順に従ってバス幅の広い有効なシングル ポート インターフェイスを作成した後、アービタを備えるソフト マルチポート インターフェイスを FPGA ロジック内部に構築して先に作成したシングル ポート インターフェイスに接続することを推奨します。

図 2 に、2 つの MCB を結合してバス幅の広い有効なインターフェイスを作成する概念を示します。各 MCB はこのような構成でも、完全に独立したアドレス、コマンド、そしてデータ ピンを持ち、個別の外部メモリ デバイスに接続されるシングル コンポーネント インターフェイスをインプリメントします。ただし、FPGA 内部で、追加ロジックを使用して MCB のユーザー インターフェイスを組み合わせ、バス幅の広いユーザー インターフェイスを 1 つインプリメントします。各 MCB からの完全な読み出しおよび書き込みデータパスを結合して 2 倍のデータ バス幅を作成し、アドレスおよびコマンド信号を 2 つの MCB に分配してこれらが同期して動作するようにします。

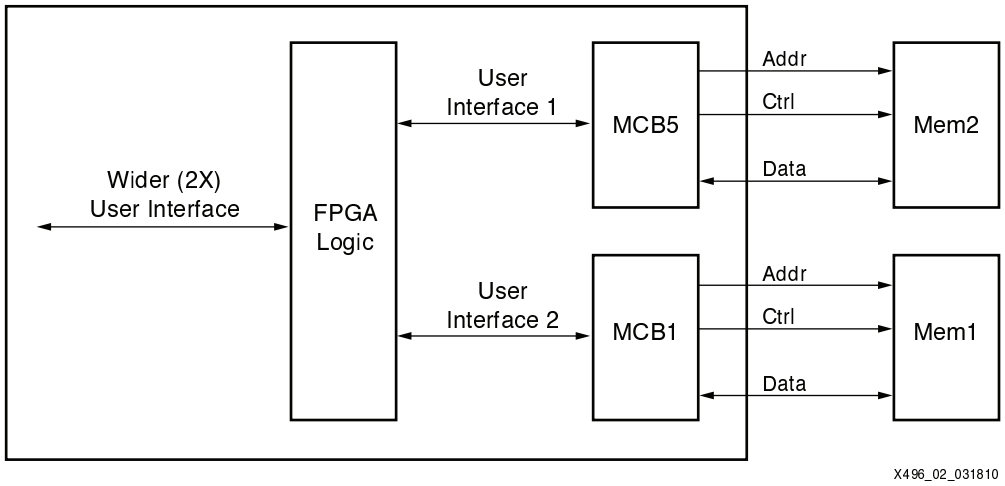

図 2 : 2 つの MCB を結合してバス幅の広い有効なインターフェイスを作成する

2 つの MCB はまったく同じようにコンフィギュレーションする必要があり、外部メモリ デバイスも同一のものを使用する必要があります。これにより、2 つのインターフェイスが可能な限り一致して動作します。ただし、2 つの外部インターフェイス間のタイミングが若干ずれるために、これらの動作で同期が失われることがあります。たとえば、2 つのメモリ デバイスがわずかに異なるタイミングでリフレッシュに遷移したために、一方の MCB がもう一方の MCB のデータ フローに対して割り込みを発生させる可能性があります。

このような準同期動作は、2 つの MCB のそれぞれのユーザー インターフェイス ポートから見て同期がとれている限り、バス幅の広いユーザー インターフェイスを作成する際に問題にはなりません。MCB は、特に非同期動作をサポートするユーザー インターフェイスで設計されたものです。つまり、ユーザー インターフェイス クロックと、MCB の物理インターフェイス (PHY) と外部メモリ インターフェイスを同期させる MCB システム クロックとの関係は完全に未定義でかまいません。ユーザー インターフェイスの FIFO が、必要なクロック ドメインの切り替えを管理します。詳細は、『Spartan-6 FPGA メモリ コントローラー ユーザー ガイド』[参照 1] の「クロック」セクションを参照してください。

MCB はタスクを順に実行するコントローラーであり、2 つの MCB は常に、共通のユーザー インターフェイスから送信される読み出しおよび書き込みコマンドを同じ順序で実行します。両 MCB の FIFO ステータス フラッグが共通するユーザー インターフェイスからの要求 (FIFO のオーバーフローやアンダーフロー条件など) を適切に制御するために監視され、使用される限り、これらの MCB はユーザー アプリケーションから見て完全に一致して動作します。以降のセクションでは、2 つの MCB の同期動作が維持されるよう共通のユーザー インターフェイスに追加する必要があるロジックについて説明します。

インプリメンテーションの説明

MCB のユーザー インターフェイスは、DDR SDRAM メモリ デバイスとの間で要求される複雑なタイミング関係やプロトコルを排除して、外部メモリ デバイスとやりとりする手段となるように設計されています。このユーザー インターフェイスでコマンド FIFO およびデータ FIFO を使用することで、ユーザー アプリケーションは推奨されているプロトコルを用いてメモリ トランザクション要求を発行できるようになります。

次に MCB 内部のコントローラー サブブロックが、これらの要求を MCB の PHY を介してメモリデバイスへ送信される DDR デバイス用の一連のコマンドに自動変換し、要求されたアクションを実行します。さらに、MCB はメモリ デバイスのリフレッシュを定期的かつ自動的に行います。ユーザー アプリケーションから見ると、外部 DDR メモリはバイト単位でアドレス指定可能な SDR メモリ空間のようなものです。ユーザー インターフェイスおよびプロトコルの詳細は、『Spartan-6 FPGA メモリ コントローラー ユーザー ガイド』[参照 1] を参照してください。

図 3 に、2 つの MCB のユーザー インターフェイスを統合する際にかかわる主要な信号を示します。コマンド ポート関連の信号 (cmd_byte_addr、cmd_bl、cmd_instr) は両ユーザー インターフェイスに並行して接続されるため、命令、アドレスおよびデータ バースト長に関する同じ情報が 2 つの MCB へ確実かつ同時に与えられます。各 MCB のユーザー インターフェイスの読み出しおよび書き込み用データパスは、2 倍のデータ幅を持つ 1 つのデータ バスとなるように連結されます。

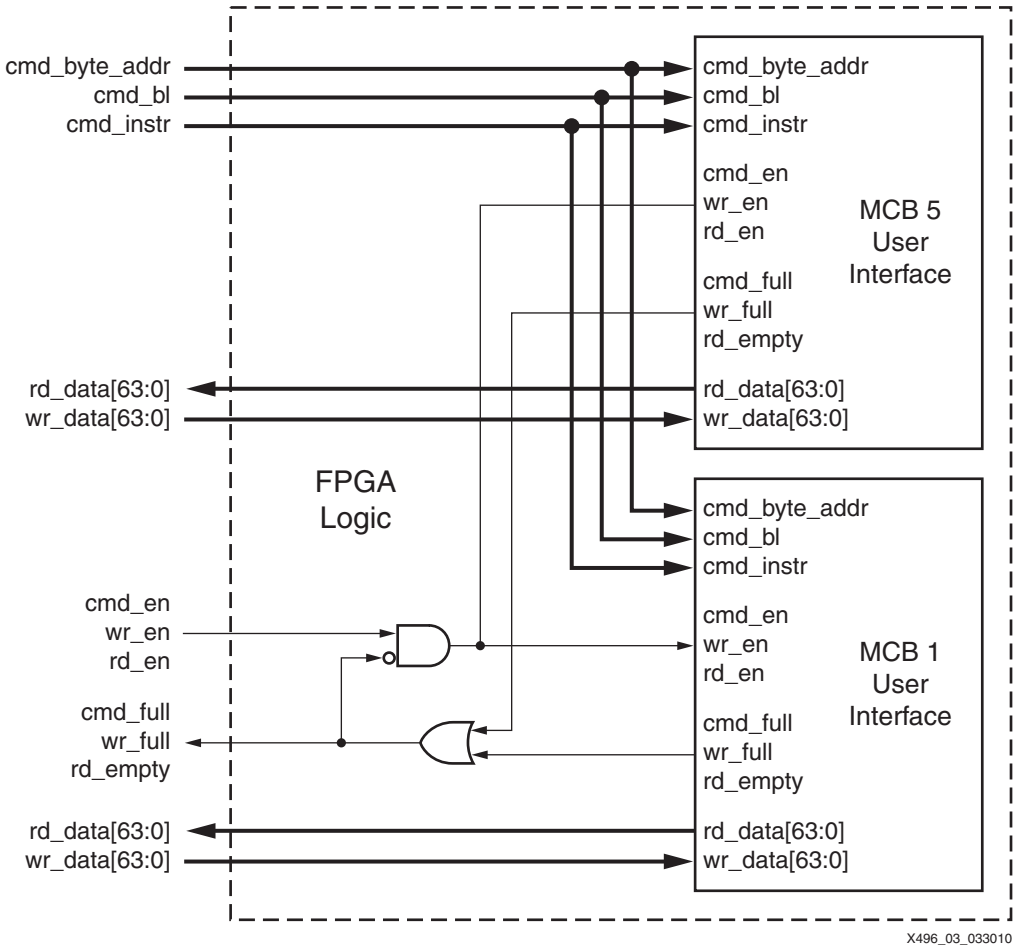

図 3 : 2 つの MCB を備えるユーザー インターフェイス

次に、FIFO ステータスおよび制御信号を結合するロジックを使用して、2 つのユーザー インターフェイスの動作を同期させる手順について説明します。2 つの MCB からの FIFO ステータス フラッグ (cmd_full、wr_full、rd_empty) は論理和が取られ、いずれかの MCB の FIFO が FULL または EMPTY 状態であることを示す 1 つのステータス フラッグとなります。統合されたステータス フラッグは、FIFO の各イネーブル入力 (cmd_en、wr_en、rd_en) をゲート制御するためにその後アクノリッジ (ACK) 信号として使用され、両ユーザー インターフェイスの同期を維持します。この結果ローカルで生じたイネーブル信号は、両 MCB のユーザー インターフェイスへ提供されます。

たとえば、2 つのコマンド パス FIFO のいずれか一方が FULL になると、アクノリッジ ロジックによって新しいコマンドはいずれの MCB にも送信されず、両インターフェイスの同期が保たれます。同様に、いずれか一方の読み出しデータ FIFO がもう一方よりも先に EMPTY となった場合 (DRAM のリフレッシュによって、割り込みがデータ フローに発生したことが原因と考えらる)、アクノリッジ ロジックによっていずれかの MCB からの追加データがクロックアウトしないように制御され、同期が保たれます。

アクノリッジ ロジックは FIFO のアクセスを直接制御することで 2 つのユーザー インターフェイスの同期を確実に維持します。しかし、ユーザー アプリケーションは、一方の MCB が FULL または EMPTY 状態を示した場合にメモリ トランザクションの要求が一時停止されるよう、統合された FIFO ステータ

ス フラッグを監視する必要があります。ローカル MCB のイネーブルがアクノリッジ ロジックによって非アクティブとなっているときに要求を継続して発行すると、トランザクションが損失し、関連データの仮定が無効になります。

『Spartan-6 FPGA メモリ コントローラー ユーザー ガイド』[参照 1] の「クロック」セクションでは、メモリ コンポーネントと MCB インターフェイスが 1 つずつの場合の一般的なインターフェイス向けの推奨クロッキング アーキテクチャについて説明しています。MIG ツールによって自動的に生成されたクロック インフラストラクチャは、推奨されるアーキテクチャに基づくものです。ただし、複数の MCB を同時に使用する場合はこの構成に一部修正を加える必要があります。

図 4 に、MCB を統合する場合のクロッキング アーキテクチャを 2 とおり示します。図 4 の (a) では、2 つの MCB がデバイスの同じ側に配置されています。MCB を 4 つ備えるデバイスで、各側に 2 つずつ配置されているときにのみこのようなアーキテクチャとなります。この場合、BUFPLL_MCB によって駆動される MCB システム クロックと 1 つの BUFG によって駆動されるユーザー インターフェイス クロックを、上部および下部の両 MCB へ配線できます。これは、2 つの MCB が共通のユーザー インターフェイスへ結合されているかどうかにかかわらず、デバイスの同じ側で 両 MCB を使用する場合の要件です。デバイスの各側に、1 つの BUFPLL_MCB 信号と 2 つの I/O クロック配線しかないため、別々のシステム クロックを同じ側にある MCB へ提供する方法はありません。

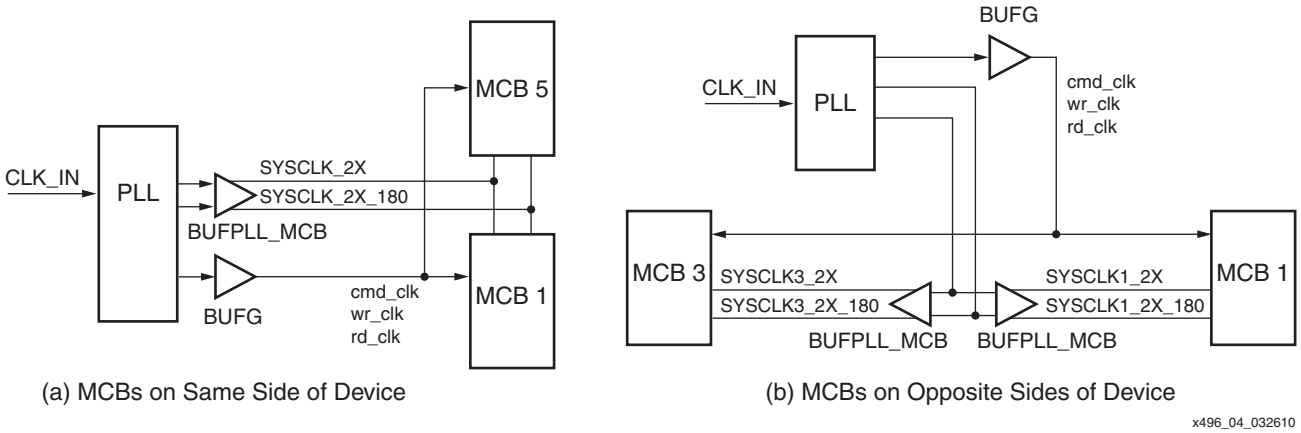

図 4 : 複数の MCB を統合する場合のクロッキング アーキテクチャ

図 4 の (b) では、MCB が 1 つずつデバイスの各側に配置されています。デバイスに 2 つの MCB しかない場合、または MCB を 4 つ備えるデバイスで MCB を組み合わせた場合にこのようなアーキテクチャとなります。この場合、反対側にある MCB を駆動するために PLL_MCB 信号を 1 つ追加する必要があります。各側に 1 つの BUFPLL_MCB とそれが駆動できる I/O クロック配線が 2 つあります。

リファレンス デザインのガイドライン

「リファレンス デザイン」のディレクトリ構造とファイルは、MCB が 1 つのデザインについて MIG が自動的に作成したものと非常に類似していますが、いくつかの例外があります。図 5 に、MIG ツールによって作成され、さらにこのリファレンス デザイン用に再生成された基本的なディレクトリ構造を示します。MIG が生成するディレクトリ構造およびファイルの詳細は、『Spartan-6 FPGA メモリ インターフェイス ソリューション ユーザー ガイド』[参照 2] を参照してください。

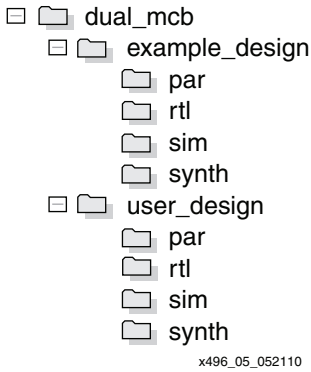

図 5 : MIG ツールによって生成された基本的なディレクトリ構造

MIG ツールは、example_design および user_design の 2 種類の MCB メモリ インターフェイス デザインを生成します。example_design には、完全に内蔵型の MCB ベースのメモリ インターフェイスをシミュレーションおよびインプリメンテーションするために必要な RTL ファイル、ユーザー制約ファイル、およびスクリプトがすべて含まれます。example_design には、デモまたはボードの立ち上げに使用できる、トラフィック ジェネレーターと呼ばれる合成可能なハードウェア テストベンチが含まれています。トラフィック ジェネレーターは、MCB ユーザー インターフェイスの動作を試行するさまざまな読み出し/書き込みスティミュラス パターンを作成できます。

user_design は、ハードウェア ベンチが含まれないという点を除いて基本的に example_design と同じです。このバージョンの MIG デザインは、システム デザイン全体へ直接インスタンシエーションできるようになっています。トラフィック ジェネレーター、example_design、および user_design の詳細は、『Spartan-6 FPGA メモリ インターフェイス ソリューション ユーザー ガイド』[参照 2] の第 1 章を参照してください。

このアプリケーション ノートの「リファレンス デザイン」は、デバイスの両側にある 2 つの MCB を統合します。各 MCB は、1 つの 1Gb DDR3 デバイスへの 16 ビット インターフェイスをインプリメントするようにコンフィギュレーションされています。各 MCB のユーザー インターフェイスは、64 ビットのシングル ポートとしてコンフィギュレーションされています。2 つの MCB を統合することで、リファレンス デザインは 32 ビット幅の DDR3 メモリ デバイスへの 128 ビット ユーザー インターフェイスを効率的にインプリメントします。サポートするシングル ポート コンフィギュレーションの MCB はいずれも同様に統合できます。

「リファレンス デザイン」を構築する最も直接的な方法は、MIG ツールを使用して 2 つの MCB インターフェイスをシングル パスで作成することです。具体的な手順は、『Spartan-6 FPGA メモリ インターフェイス ソリューション ユーザー ガイド』[参照 2] を参照してください。

MIG ツールの [Memory Selection] ページで、2 つの MCB に対して同じメモリ インターフェイス規格 (DDR3 SDRAM など) を選択してください。MIG ツールの GUI フローの各手順において、両 MCB に同じ設定 (選択するメモリ デバイスとそのスピード、ポートのコンフィギュレーション、終端方式など) を指定します。

必要な基本ファイルすべてが MIG の GUI ウィザードの 1 回の実行で生成される場合、後続のセクションで説明する修正をインプリメントすることで、2 つの独立した MCB インターフェイスを、ユーザー インターフェイスとメモリ インターフェイスのデータ幅が 2 倍の 1 つのデザインへ結合できます。example_design への修正手順が説明されていますが、user_design にも同様の手順を適用できます。

新しいラッパーの作成 (memc13_wrapper.v)

MIG ツールは、2 つの MCB ラッパー ファイル (memc1_wrapper.v および memc3_wrapper.v) を生成します。リファレンス デザイン ファイルをインプリメントするには、2 つの MCB ユーザー インターフェイスを 1 つのインターフェイスに結合する新しいラッパーが必要です。ここでは、新しいラッパー ファイル (memc13_wrapper.v) がこれら 2 つの基本ラッパー ファイルと最上位の example_top.v

デザイン ファイル間に作成されます。図 5 は、2 つの基本ラッパーではなく、example_top.v ファイルでインスタンシエートされた新しいラッパー層の挿入を示すものです。

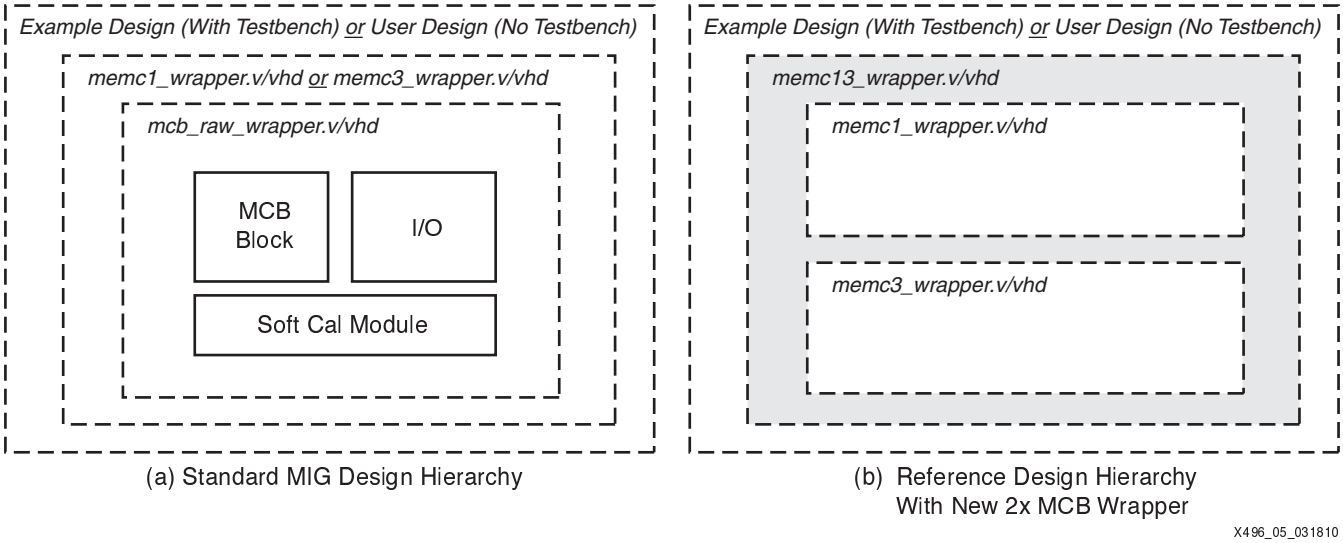

図 6： 新しいラッパー層の挿入

新しい memc13_wrapper.v ファイルは次の変更に対応し、これらのほとんどがリファレンス デザイン ファイルに直接コメントされています。

- 新しいラッパーは、2 つのユーザー インターフェイス ポートのコマンドおよびデータパスを統合するために、図 3 のコネクティビティおよびロジックをインプリメントします。コマンド信号は両 MCB へ送信され、統合されたデータパスは両 MCB からの個別データパスを連結したものになります。このラッパーはさらに、FIFO ステータス信号の論理和、および FIFO イネーブルをゲート制御するアクノリッジ ロジックをインプリメントします。
- 新しいラッパーは、2 つの基本ラッパーのインスタンシエーションに対して共通のパラメーターを渡します。
- 外部メモリ インターフェイス バスは、2 つの独立したコンポーネント インターフェイスとして維持されるため、分離しておきます。
- MCB システム クロック (sysclk1_2x、sysclk3_2x) およびクロック イネーブル (pll_ce1_0、pll_ce3_0) は、MCB がデバイスの両側にある場合、異なる BUFPLL_MCB ブロックによって駆動されるため、分離しておきます。デバイスの同じ側にある MCB を統合する場合は分離する必要はありません。

新しいクロック インフラストラクチャ ブロック (memc13_infrastructure.v)

MIG ツールによって、1 つしかない場合でも 2 つの クロック インフラストラクチャ ブロックが基本ファイル (memc1_infrastructure.v および memc3_infrastructure.v) に生成されます。一方を削除し、残った一方に memc13_infrastructure.v という名前を付けて次の修正を適用します。

- 既存の BUFPLL_MCB ブロックと同じ PLL 出力によって駆動される 2 つ目の BUFPLL_MCB ブロックを追加します。MCB がデバイスの同じ側にある場合は追加する必要はありません。
- 新しい BUFPLL_MCB ブロックの出力から、2 組目のシステム クロック (sysclk3_2x) およびクロック イネーブル (pll_ce3_0) を作成し、これらを新たに memc13_infrastructure.v と名付けたブロックから引き出します。

新しいテストベンチ ブロック (memc13_tb_top.v)

MIG ツールによって、1 つしかない場合でも 2 つのテストベンチ ブロックが基本ファイル (memc1_tp_top.v および memc3_top_tb.v) に生成されます。一方を削除し、残った一方に memc13_tb_top.v という名前を付けて次の修正を適用します。

- データ ポート サイズ (C_P0_DATA_PORT_SIZE) およびマスク サイズ (C_P0_MASK_SIZE) のグローバル パラメーターを新しいユーザー インターフェイス幅に対応するように 2 倍に設定します。
- error_status 出力の幅を、統合されたユーザー インターフェイスの新しい幅の分だけ増分させます (統合されたユーザー インターフェイスが 128 ビット幅の場合は、error_status バスに 128 ビット追加する)。
- ローカルの DWIDTH パラメーター (p0_DWIDTH) の値を、統合されたユーザー インターフェイス幅 (128 ビット) に設定します。

サンプル デザインの最上位ブロックへの変更 (example_top.v)

MIG ツールによって生成された基本ファイルには、最上位デザイン ブロック (example_top.v) に基本 MCB ラッパー、クロック インフラストラクチャ ブロック、およびテストベンチ ブロックのインスタンシエーションが 2 つ含まれます。次の変更により、前述の手順で修正されたブロックを用いる統合された 1 つの MCB インターフェイスがインプリメントされます。

- 2 つの外部メモリ インターフェイスへの入出力信号を分離しておきます。
- 両 MCB インターフェイスに共通するすべてのグローバルおよびローカル パラメーターに対してまとめたものを作成します (たとえば、個別の C1_MEMCLK_PERIOD と C3_MEMCLK_PERIOD パラメーターではなく、C13_MEMCLK_PERIOD)。この共通のパラメーター セットは、複数のサブブロックへ渡す必要があります。
- データ ポート サイズ (C13_P0_DATA_PORT_SIZE) およびマスク サイズ (C13_P0_MASK_SIZE) のグローバル パラメーターを新しいユーザー インターフェイス幅に対応するように 2 倍に設定します。
- 入力クロックおよびリセットをまとめたもの (c13_sys_clk_p、c13_sys_clk_n など) を作成し、各インターフェイスの個別のクロックと置き換えます。
- テストベンチのアドレス パラメーターを次のように変更します。
  - BEGIN_ADDRESS および END_ADDRESS パラメーターを 2 倍にします。詳細は、example_top.v ファイルのコード コメントを参照してください。
- システム クロック (c1_sysclk_2x) を除く内部信号 (c13_calib_done) とクロック イネーブル (c1_pll_ce_0) 用に共通の接続を作成します。
- エラーおよび calib_done 出力信号を c13_error および c13_calib_done 信号に直接割り当てます。
- 前述の手順で新たに作成した memc13_wrapper.v ラッパー、クロック インフラストラクチャ ブロック、テストベンチ ブロックをインスタンシエートし (それぞれについて 1 つのコピーのみ)、必要な信号に接続します。

シミュレーションおよびインプリメンテーションにおけるその他の変更

シミュレーションおよびデザイン インプリメンテーションに適したリファレンス デザインとなるように、変更を加えます。

- sim/functional サブディレクトリで、example_top.v ファイルに適用された信号の変更に対応できるように sim_tb_top.v ファイルを修正します (1 組のクロックおよびリセット信号の生成を含む)。
- par サブディレクトリで、example_top.v ファイルに適用された信号の変更に対応できるように example_top.ucf ファイルを修正します (1 組のクロックおよびリセット信号の指定を含む)。

- 次のように、トラフィック ジェネレーターを手動で変更します。

  example_design/rtl/traffic_gen ディレクトリで提供される mcb_traffic_gen.v モジュールを開きます。次のコードを検索します (317 行目から開始)。

```
MIG 3.4 Code:

     reg mcb_rd_empty;
     always @ (mcb_rd_empty_i, mcb_rd_empty_r)
     if ( FAMILY == "SPARTAN6")
        mcb_rd_empty = mcb_rd_empty_r;
    else
       mcb_rd_empty = mcb_rd_empty_i;

   reg mcb_wr_full;
   always @ (mcb_wr_full_i, mcb_wr_full_r1)
   if ( FAMILY == "SPARTAN6")
       mcb_wr_full = mcb_wr_full_r1;
   else
       mcb_wr_full = mcb_wr_full_i;
```

  次のコードに置き換えます。

```
Workaround Code:
    wire mcb_rd_empty;
    assign     mcb_rd_empty = mcb_rd_empty_i;

    wire mcb_wr_full;
    assign     mcb_wr_full = mcb_wr_full_i;
```

デバイスの使用 リソースと性能

表 1 に、関連するリファレンス デザインの使用リソースおよび性能をまとめています。

表 1 : デザインの使用リソースと性能の詳細

パラメーター	スピード グレード	仕様および説明
最大データ レート : 外部インターフェイス (スピード グレードごと)	-2	データシート DS162 : 『Spartan-6 データシート : DC 特性およびスイッチ特性』の「パフォーマンスの特性」セクションを参照してください。[参照 3]
	-3	
実現可能なデータ レート (公称値) : ユーザー インターフェイス (スピード グレードごと)	-2	87MHz(1)
	-3	100MHz(1)
ターゲット Spartan-6 FPGA		XC6SLX16
デバイス使用率 (テストベンチなし)		< 10 スライス
有効な外部インターフェイス バス幅		32 ビット
有効なユーザー インターフェイス バス幅		128 ビット
検証用のターゲット メモリ デバイス	シミュレーション	MT41J64M16LA_187E (Micron 社製 DDR3)
	ハードウェア	MT41J64M16LA_187E (Micron 社製 DDR3) EDE1116ACBG_8E_E (Elpida 社製 DDR2)

注記 :

1. ユーザー インターフェイスの最大データ レートは、デバイスおよびデザインの PAR の結果によって異なります。ここに示す数値は、このアプリケーション ノートのサンプル デザインで達成可能なパフォーマンスを反映したものです。

リファレンス デザイン

このアプリケーション ノートのリファレンス デザインは、次のサイトからダウンロードできます。

https://secure.xilinx.com/webreg/clickthrough.do?cid=147378

リファレンス デザインの詳細

表 2 に、リファレンス デザインで使用されるツール フローおよび検証手順の詳細を示します。

表 2 : リファレンス デザインの詳細

パラメーター	説明
一般	
開発元	ザイリンクス
ターゲット デバイス	Spartan-6 LX および LXT FPGA
ソース コードの提供	あり
ソース コードの形式	Verilog
リファレンス デザインに統合されているその他の IP	MIG によって生成される IP : MCB インターフェイス ラッパー
シミュレーション	
機能シミュレーションの実施	あり
タイミング シミュレーションの実施	なし
テストベンチの形式	Verilog
シミュレーター	Modelsim 6.5c
インプリメンテーション	
合成	XST (ISE® デザイン ツール v12.1)
インプリメンテーション	ISE デザイン ツール v12.1
スタティック タイミング解析の実施	あり
ハードウェア検証	
ハードウェア検証の実施	あり
検証に使用したハードウェア プラットフォーム	SP601 ボード

まとめ

MCB は、最も低いコストと消費電力を実現する汎用 SDRAM メモリ規格へのインターフェイスを提供し、大部分の Spartan-6 FPGA アプリケーションで求められるメモリ インターフェイス要件に対応します。しかし、メモリ帯域幅または集積度の要件がより厳しい一部のアプリケーションには、MCB 1 個で提供できる 16 ビットを超えるメモリ インターフェイスを使用します。このアプリケーション ノートでは、MCB 動作を結合して 32 ビット以上の有効なメモリ インターフェイスを実装する方法について説明してきました。それぞれの MCB が最大 800Mb/s のフル パフォーマンスで動作するため、ユーザー アプリケーションではこれらの専用内臓メモリ コントローラーを十分に活用して、よりデータ幅の広いインターフェイスを実装できます。

参考資料

1. UG388 :『Spartan-6 FPGA メモリ コントローラー ユーザー ガイド』
2. UG416 :『Spartan-6 FPGA メモリ インターフェイス ソリューション ユーザー ガイド』
3. DS162 :『Spartan-6 データシート : DC 特性およびスイッチ特性』

改訂履歴

次の表に、この文書の改訂履歴を示します。

日付	バージョン	内容
2010 年 6 月 3 日	1.0	初版リリース

Notice of Disclaimer