XAPP1296 (v1.0) 2017 年 6 月 23 日

UltraScale+ FPGA の ICAP を使用するマルチブートおよびフォールバック

著者: Guruprasad Kempahonnaiah

概要

このアプリケーション ノートでは、UltraScale+™ FPGA の重要な機能の 1 つであるマルチブートについて説明します。UltraScale+ FPGA のマルチブート機能を使用すると、FPGA アプリケーションの制御により複数の FPGA ビットストリームを読み込むことができます。FPGA アプリケーションがマルチブート動作を開始すると、FPGA は別のコンフィギュレーション ビットストリームでリコンフィギュレーションを実行します。マルチブート動作が開始されると、FPGA は通常どおりのコンフィギュレーション プロセスを再開します。この資料では、ICAP を使用したマルチブート機能を実装する手順、フォールバックを開始するさまざまな方法、およびブート ステータス (BOOTSTS) レジスタを使用してマルチブートまたはフォールバック動作をデバッグおよび検証する方法について説明します。このアプリケーション ノートには、SPI モードで ICAP を使用した UltraScale+ FPGA のマルチブート機能を評価するためのリファレンス デザインが付属します。

このアプリケーション ノートのリファレンス デザイン ファイルは、ザイリンクスのウェブサイトからダウンロードできます。デザイン ファイルの詳細は、「リファレンス デザイン」を参照してください。

はじめに

UltraScale+ FPGA のマルチブートおよびフォールバックは、フィールドでのシステム アップデートをサポートする機能です。UltraScale™ アーキテクチャは、SPI x1、x2、および x4 でのマルチブートをサポートしており、FPGA は複数のビットストリームが格納されている SPI フラッシュ デバイスからビットストリームを読み込むことができます。フィールドでビットストリーム イメージをアップグレードできるため、設計者には非常に大きなメリットとなります。FPGA のマルチブート機能を使用すると、リアルタイムにイメージを切り替えることができます。マルチブート コンフィギュレーション プロセス中にエラーが検出されると、FPGA はフォールバック機能を開始して、検証済みのデザイン (ゴールデン イメージ) をデバイスに読み込むことができます。

このアプリケーション ノートでは、SPI (x1/x2/x4) コンフィギュレーション インターフェイスを使用する UltraScale+ FPGA のマルチブートおよびフォールバック機能について説明します。ここでは、ザイリンクスの KCU116 開発ボードに実装された Micron 社製 MT25QU01 シリアル NOR フラッシュ メモリ デバイスを SPI x4 コンフィギュレーション モードで使用します。SPI x4 コンフィギュレーション インターフェイスの詳細は、『UltraScale アーキテクチャ コンフィギュレーション ユーザー ガイド』(UG570) [参照 1] を参照してください。

マルチブートおよびフォールバックの基本

FPGA アプリケーションがマルチブート動作を開始すると、FPGA は別のコンフィギュレーション ビットストリームでリコンフィギュレーションを実行します。マルチブート動作が開始されると、FPGA は通常どおりのコンフィギュレーション プロセスを再開し、専用のマルチブート ロジック、ウォーム ブート開始アドレス (WBSTAR) レジスタ、および BOOTSTS レジスタ以外のコンフィギュレーション メモリを消去します。その後、SPI フラッシュ デバイスからの新しいビットストリームでリコンフィギュレーションを実行します。

フォールバックが開始される条件

コンフィギュレーション中に次のエラーが生じるとフォールバックが開始されます。

- IDCODE エラー
- 巡回冗長検査 (CRC) エラー
- ウォッチドッグ タイマーのタイムアウト エラー

フォールバックは、ビットストリーム オプションの BITSTREAM.CONFIG.CONFIGFALLBACK で有効にできます。フォールバック リコンフィギュレーション中は、ウォッチドッグ タイマーが無効になります。フォールバック リコンフィギュレーションでエラーが発生すると、コンフィギュレーションが停止し、INIT_B と DONE の両方が Low に維持されます。

ゴールデン イメージ

FPGA に電源が投入されると、アドレス `0x0` からゴールデン イメージが読み込まれます (図 1)。電源投入時には、まずゴールデン イメージが読み込まれます。マルチブート開始イベントが認識されると、FPGA は上位アドレス空間からマルチブート イメージを読み込みます。マルチブート イメージは複数用意しておくことが可能で、任意のデザインからほかのイメージの読み込みを開始できます。マルチブート イメージのブート中にエラーが発生し、コンフィギュレーションを完了できなかった場合は、フォールバック回路がアドレス `0x0` からゴールデン イメージの読み込みを開始します。

マルチブート イメージ

マルチブート イメージは上位アドレス空間から読み込まれます。このイメージのコンフィギュレーションが正常に終了しなかった場合は、フォールバックが自動的に開始され、アドレス `0x0` に格納されているゴールデン イメージが読み込まれます。フォールバック機能は、マルチブート イメージの読み込みが正常終了しなかった場合にシステムを回復し、ゴールデン イメージの読み込みを可能にします。

マルチブート リファレンス デザイン

このセクションでは、マルチブート リファレンス デザインの予想される動作、およびリファレンス デザインをコンパイルして KCU116 評価ボードで検証する方法について説明します。このリファレンス デザインは、まずゴールデン イメージで初期システムをセットアップした後、マルチブート機能のデモを開始します。

ゴールデン イメージによる初期システムのセットアップ

FPGA に電源が投入されると、アドレス 0 からゴールデン イメージが読み込まれます。次に、このゴールで イメージのデザインによってマルチブート イメージの読み込みが開始されます。このような方式は、システム チェックを実行してからランタイム イメージを読み込む場合に適しています。つまり、システムのチェック/診断機能をゴールデン イメージに含めておき、実際のシステムとしての動作はマルチブート イメージに含めておきます。電源投入時に読み込まれるゴールデン イメージにより、上位アドレス空間からのブートが開始されます。マルチブート イメージは複数用意しておくことが可能で、任意のデザインからほかのイメージの読み込みを開始できます。上位アドレス空間からマルチブート イメージを読み込み中にエラーが発生した場合は、フォールバック回路がアドレス `0x0` からゴールデン イメージの読み込みを開始します。図 1 に、ゴールデン イメージの初期セットアップ フローを示します。

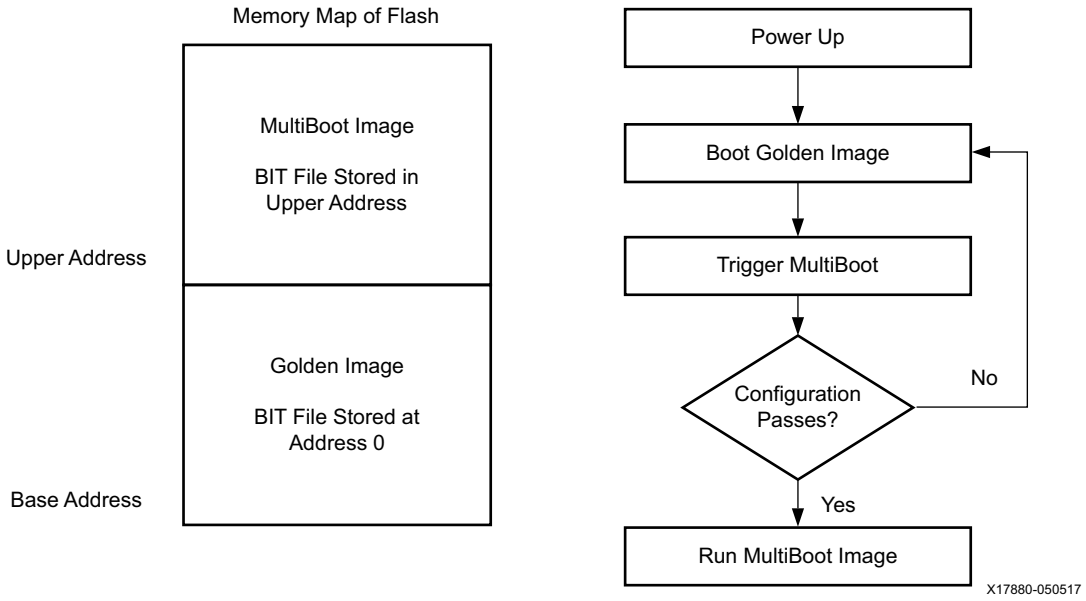


図 1: ゴールデン イメージの初期フロー図

予想されるイメージの動作

電源を投入すると、ゴールデン イメージでコンフィギュレーションが実行され、デザインの動作として GPIO LED [0:7] が順番に点灯します。読み込まれたイメージは UART 経由で確認することもできます。

ゴールデン イメージで動作中に DIP SW13[1] をトグル (0 → 1 → 0) すると、IPROG が発行されて `0x01000000` のマルチブート イメージへジャンプします。このリファレンス デザインは ICAP を使用してマルチブート イメージへジャンプします。IPROG が発行されると、UART にメッセージが表示されます。

マルチブート イメージのコンフィギュレーションに成功すると、デザインの動作としてすべての GPIO LED [0:7] が点滅します。読み込まれたイメージは UART 経由で確認することもできます。マルチブート イメージで動作中に DIP SW13[1] が1 になるとシステムは FPGA の IDCODE を読み出し、UART 経由で表示します。

ゴールデン イメージで動作中に DIP SW13[4] をトグル (0 → 1 → 0) すると、IPROG によるジャンプは ICAP 経由で `0x02000000` に発行されます。このアドレスには有効なビットストリーム (コンフィギュレーション イメージ) が存在しないため、ウォッチドッグ タイマーがタイムアウトしてゴールデン イメージへのフォールバックが開始されます。

CCLK を 51.0MHz に設定した場合、SPI x4 インターフェイス経由のコンフィギュレーション時間は 1 秒未満です。デフォルトの CCLK 周波数では、ウォッチドッグ タイマーがタイムアウトするのに約 15 秒かかります。

マルチブート リファレンス デザインのコンパイル

マルチブート リファレンス デザインは IP インテグレーター (IPI) を使用してインプリメントされています。ゴールデン イメージおよびマルチブート イメージのファイルをコンパイルして生成するには次の手順を実行します。

1. [スタート] → [すべてのプログラム] → [Xilinx Design Tools] → [Vivado 2017.1] → [Vivado] をクリックして、Vivado® ツールを起動します。
   a. [Open Project] をクリックします。KCU116 マルチブート リファレンス デザイン (`<Directory>\KCU116_MB.xpr`) を開きます (図 1)。

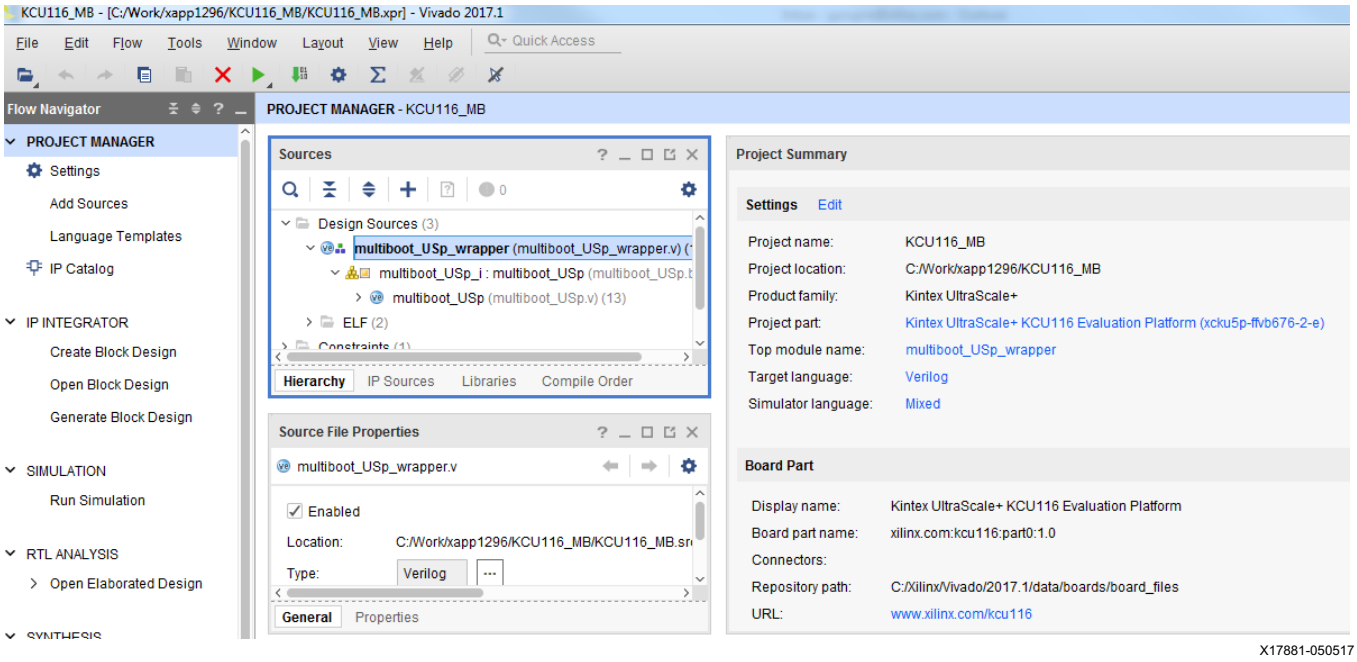


図 2: マルチブート リファレンス デザインのプロジェクト

   b. このリファレンス デザインは再コンパイルまたは SDK へのエクスポートが可能です。
      - 再コンパイルするには、[synth_1] を右クリックし、[Reset Runs] → [Generate Bitstream] をクリックします。
      - SDK にエクスポートするには、インプリメント済みデザインを開いて [File] → [Export] → [Export Hardware] をクリックします。

c. [File] → [Launch SDK] をクリックして SDK を起動します (図 2)。

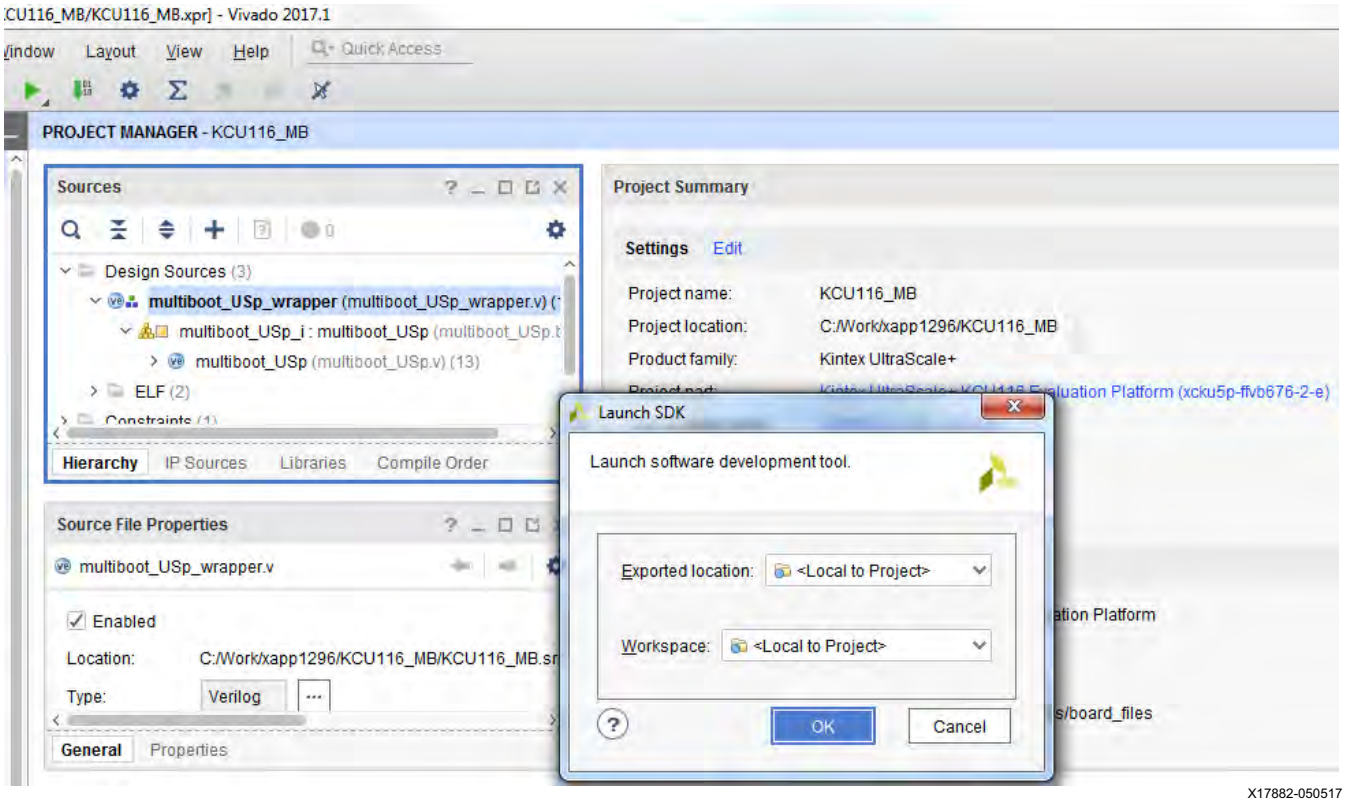


図 3: SDK の起動

2. SDK でのソフトウェアのコンパイル: SDK ではプロジェクトによって自動的に ELF ファイルがビルドされます。完了したら、SDK を終了して Vivado ツールに戻ります (図 4)。

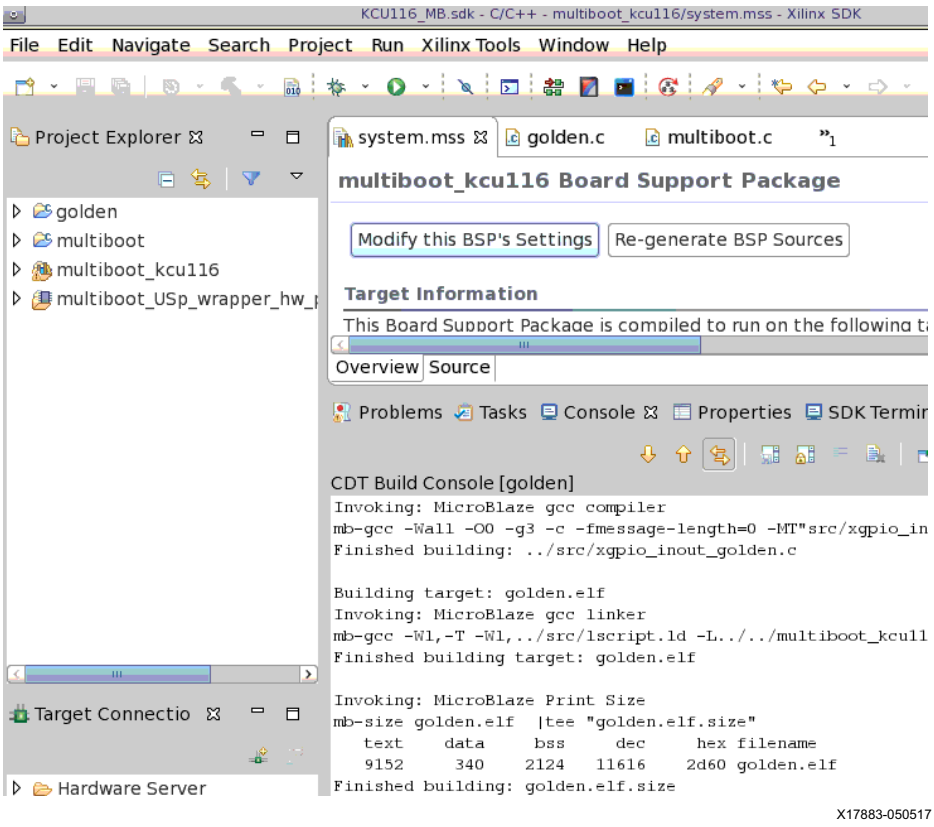
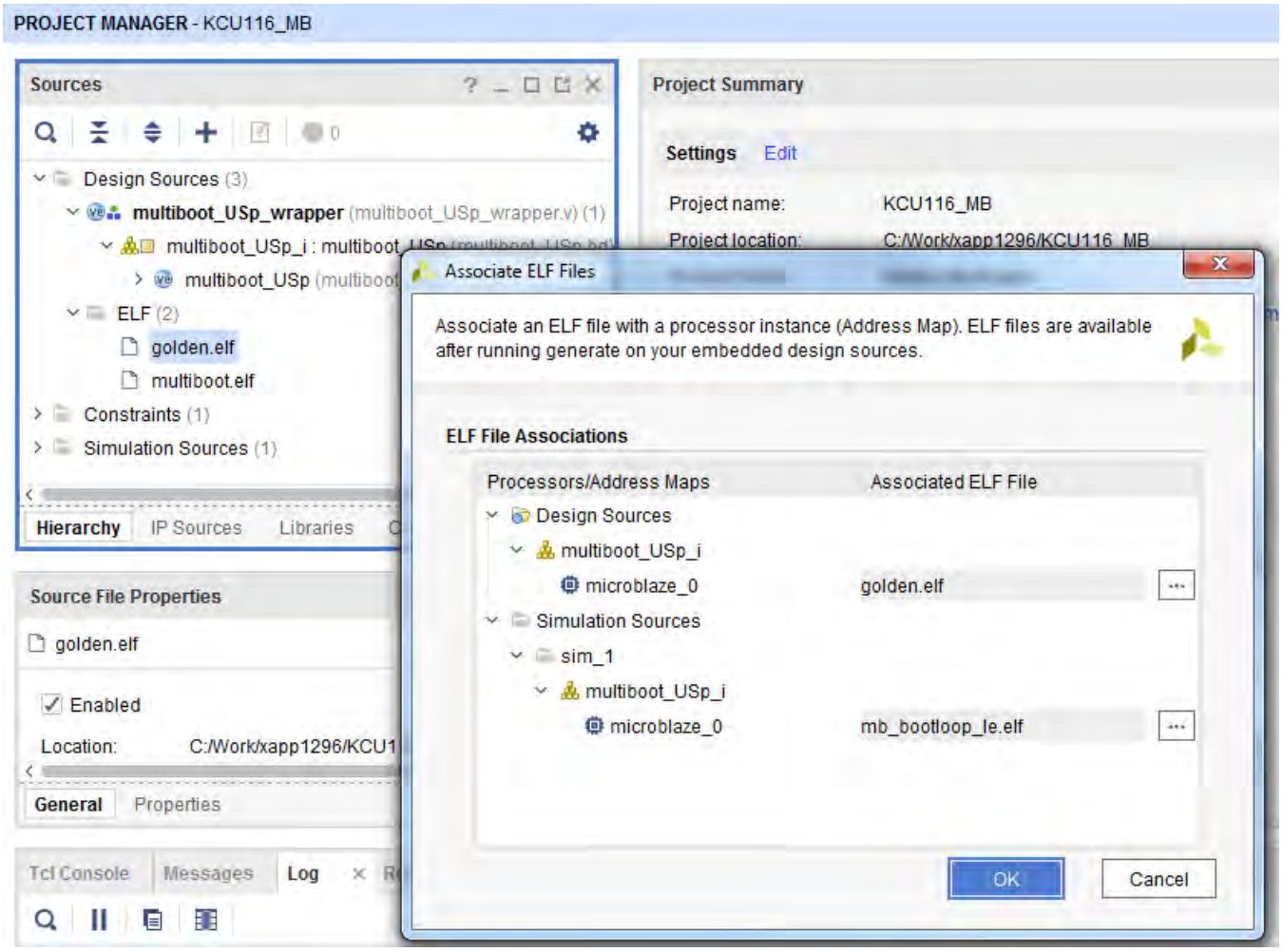


図 4: SDK で ELF ファイルをコンパイル

3. SDK で ELF ファイルをコンパイルしたら、[Associate ELF Files] コマンドを使用してデザインに関連付ける必要があります。SDK の ELF ファイルは次のディレクトリにあり、関連付けは Vivado プロジェクトで行います。

```
<Directory>\KCU116_MB.sdk\golden\Debug\golden.elf
<Directory>\KCU116_MB.sdk\multiboot\Debug\multiboot.elf
```

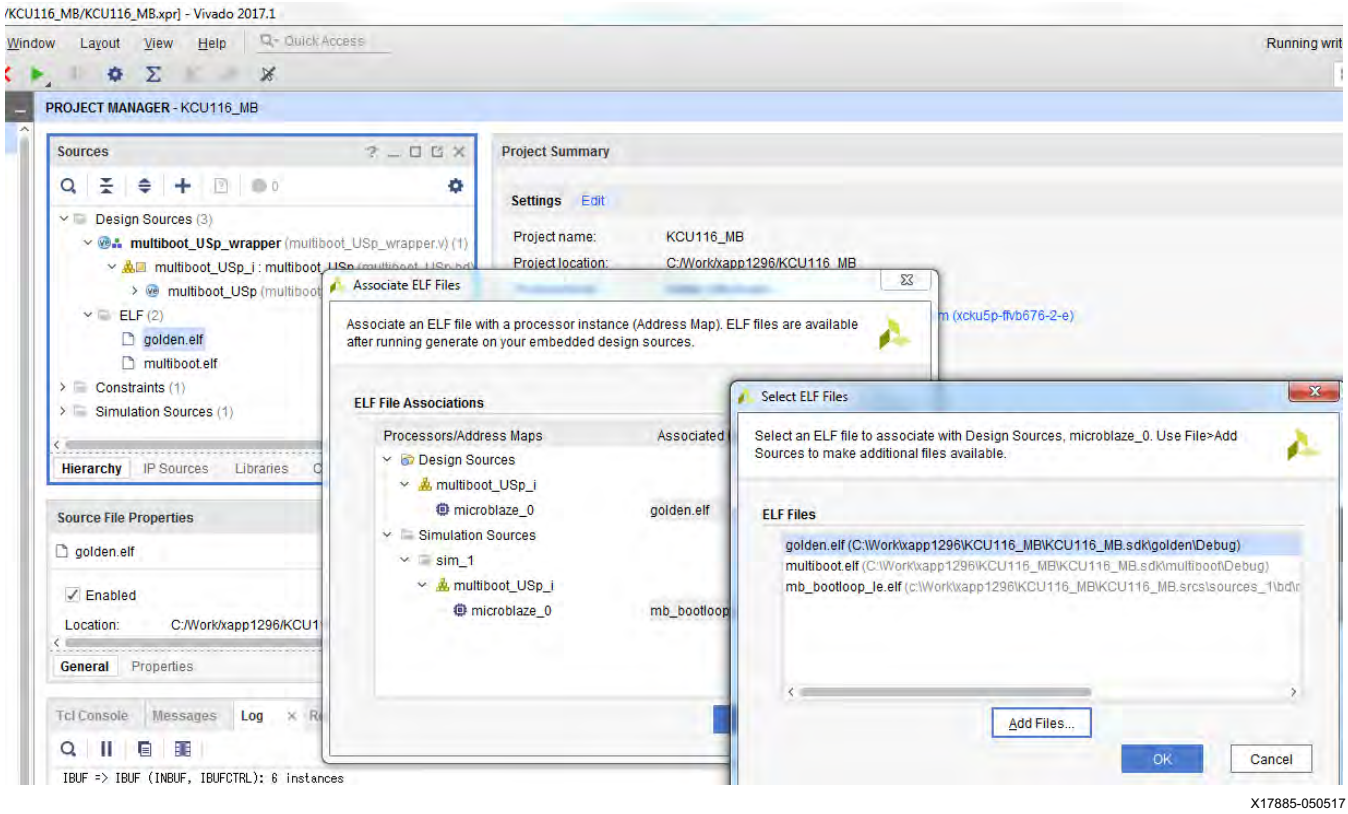
4. いずれかの ELF ファイルを右クリックして [Associate ELF Files] をクリックします (図 5)。



X17884-050517

図 5: Vivado ツールの [Associate ELF Files] ダイアログ ボックス

5. [Multiboot.elf] の右にあるボタンをクリックし、[golden.elf] を選択して [OK] を 2 回クリックします (図 6)。



X17885-050517

図 6: Vivado ツールで golden.elf ファイルを関連付け

6. Vivado ツールで write_bitstream を実行します。これで、ゴールデン イメージのビットストリームが生成されます。
7. 次の手順で上書きされないように、生成された BIT ファイルの名前を変更します。

8. `[Multiboot.elf]` の右にあるボタンをクリックし、[multiboot.elf] を選択して [OK] を 2 回クリックします (図 7)。

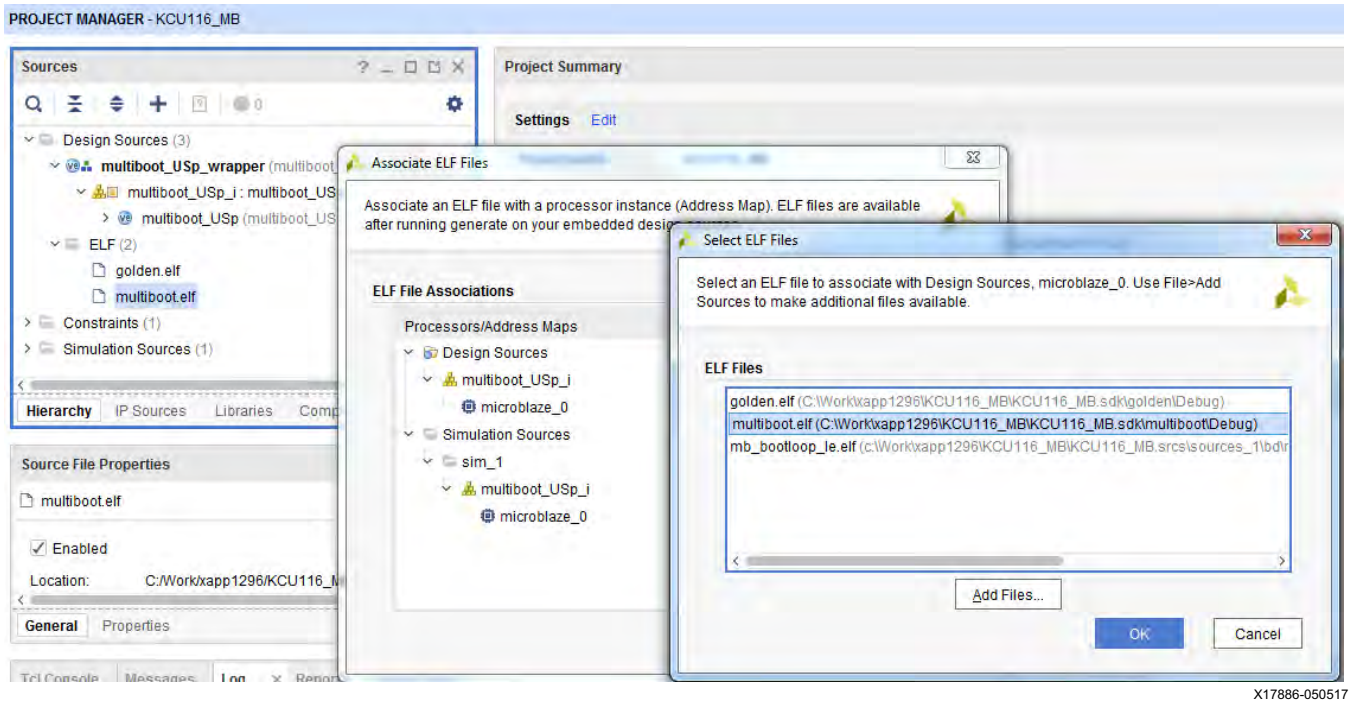


図 7: Vivado ツールで multiboot.elf ファイルを関連付け

9. Vivado ツールで write_bitstream を実行します。これで、マルチブート イメージのビットストリームが生成されます。
10. `Create_MCS.tcl` を使用して MCS ファイルを生成します。

SDK のファイル `golden.c` は DIP SW13 を継続的に監視し、DIP SW13 のステータスに基づいて IPROG の発行を制御します。IPROG が発行されると、ICAP には図 8 に示すデータ シーケンスがプログラムされます。この ICAP コマンドのシーケンスは、『UltraScale アーキテクチャ コンフィギュレーション ユーザー ガイド』(UG570) [参照 1] の表「ICAP を使用した IPROG のビットストリーム例」に記載されているものです。

```
// Reusing IDCODE array and code to send IPROG
// From UG570  Table 11-3
static u32 ReadId[HWICAP_EXAMPLE_BITSTREAM_LENGTH] =
{
    0xFFFFFFFF, /* Dummy Word */
    0xAA995566, /* Sync Word*/
    0x20000000, /* Type 1 NO OP */
    0x30020001, /* Write WBSTAR cmd */
    0x01000000, /* Warm boot start address (Load the desired address) */
    0x30008001, /* Write CMD */
    0x0000000F, /* Write IPROG */
    0x20000000, /* Type 1 NO OP  */
};
```

X19171-050517

図 8: IPROG 用の ICAP コマンド

図 9 に、SDK の golden.c のマルチブート アドレスを示します。

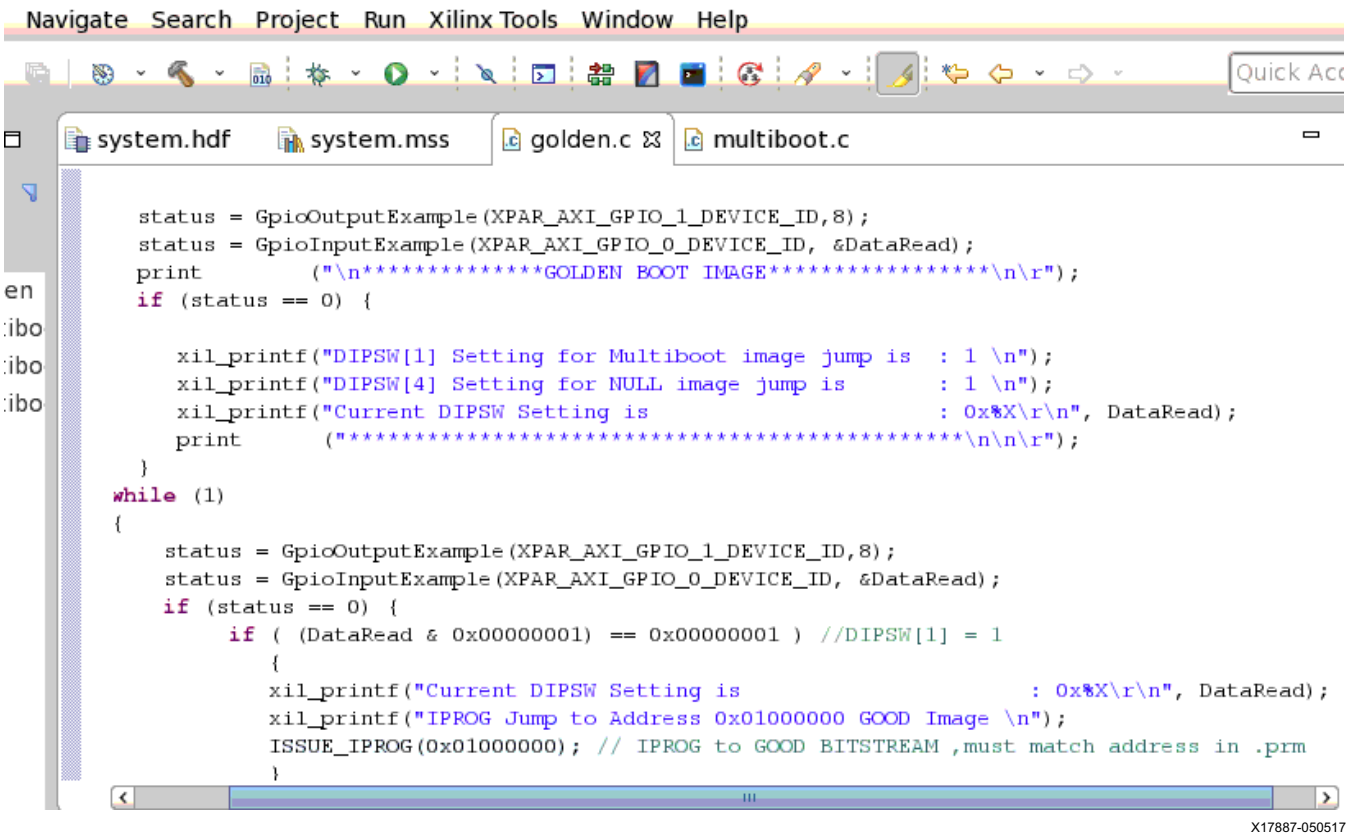


図 9: SDK の golden.c のマルチブート アドレス

DIP SW13[1] をトグル (0 → 1 → 0) すると、デザインは WBSTAR アドレスを 0x01000000 に設定して IPROG を発行します。

DIP SW13[4] をトグル (0 → 1 → 0) すると、デザインは WBSTAR アドレスを 0x02000000 に設定して IPROG を発行します。

マルチブート リファレンス デザインのビットストリーム設定

表 1 に、マルチブート リファレンス デザインで使用するビットストリーム設定を示します。これらの設定は、ゴールデン イメージとマルチブート イメージの両方のビットストリームに共通です。この表には、デフォルト値、設定可能な値、およびリファレンス デザインで使用されるオプションが記載されています。

表 1: ビットストリーム設定

設定	デフォルト値	設定可能な値	デザインの設定	説明
BITSTREAM.CONFIG.SPI_BUSWIDTH	None	None、1、2、4、8	4	SPI バスをクワッド (x4) モード SPI コンフィギュレーションに設定する。
BITSTREAM.CONFIG.CONFIGFALLBACK	Enable	Enable、Disable	Enable	コンフィギュレーションが正常に完了しなかった場合、デフォルトのビットストリームの読み込みを有効または無効にする。
BITSTREAM.CONFIG.SPI_32BIT_ADDR	No	No、Yes	Yes	SPI 32 ビット アドレス形式を有効にする。256Mb またはそれより大規模なストレージを持つ SPI デバイスの場合に必要。

表 1: ビットストリーム設定 (続き)

設定	デフォルト値	設定可能な値	デザインの設定	説明
BITSTREAM.CONFIG. CONFIGRATE	3	2.7、5.3、8.0、10.6、21.3、31.9、36.4、51.0、56.7、63.8、72.9、85.0、102.0、127.5、170.0	51.0	CCLK を 51.0MHz に設定する。
BITSTREAM.CONFIG. TIMER_CFG	-	-	0x01FFFFFF	コンフィギュレーション モードにおけるウォッチドッグ タイマーの値を設定する。
BITSTREAM.GENERAL. COMPRESS	False	True、False	True	複数のフレームをビットストリームに一度に書き込む機能を使用して、ビットストリーム (.bit) ファイルだけでなく、ビットストリームのサイズを小さくする。Compress オプションを使用しても、ビットストリーム サイズが小さくなるとは限らない。
BITSTREAM.CONFIG. SPI_FALL_EDGE	No	No、Yes	Yes	クロックの立ち下がりエッジを使用して SPI データを取り込むように FPGA を設定する。これによりタイミング マージンが改善され、コンフィギュレーションのクロック レートを向上できることがある。

ハードウェアでデザインを検証

このセクションでは、ハードウェアでマルチブート フォールバック リファレンス デザインを検証する方法を説明します。

ハードウェア要件

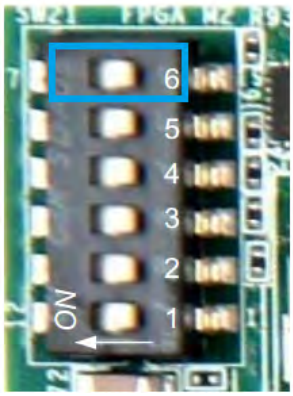
- KCU116 評価ボード
- KCU116 の Digilent USB-to-JTAG モジュールに接続するための USB A-to-micro-B ケーブルまたはザイリンクスのプラットフォーム ケーブル USB II
- KCU116 の USB UART インターフェイスに接続するための USB A-to-micro-B ケーブル

ソフトウェア要件

- Vivado Design Suite 2017.1 (SDK を含む)
- TeraTerm またはその他のターミナル ソフトウェア (UART 接続用)
  - [Baud rate]: 9600
  - [Parity]、[Flow control]: None
  - [Terminal setup] の [New-line] にある [Receive]: Auto

ボードのセットアップ

MODE ピンはマスター SPI に設定する必要があります。KCU116 ボードでは、M0 と M1 は固定値にハードワイヤードされています。M2 は SW21 を使用して 0 に設定します (図 10)。



X17888-050517

図 10: SW21.6 を 0 に設定

DIP SW13 の初期設定は 0000 とします (図 11)。

X17889-050517

図 11: DIP SW13[1:4] を 0000 に設定

フラッシュのプログラム

リファレンス デザインには生成済みの MCS ファイルが付属しており、これを使用して KCU116 ボード上の SPI フラッシュ MT25QU01 をプログラムできます。表 2 に、これらファイルの説明を示します。

表 2: MCS ファイルの説明

ファイル名	説明
`Golden_n_Multiboot.mcs`	ゴールデンおよびマルチブート イメージ。ゴールデン イメージとマルチブート イメージ両方のコンフィギュレーションが正常に行われることを確認できます。
`Golden_n_Multiboot_CRC_Err.mcs`	マルチブート イメージの CRC が破損しています。マルチブート イメージを読み込むと CRC エラーが発生し、フォールバックが開始します。
`Golden_n_Multiboot_ID_Err.mcs`	マルチブート イメージの FPGA IDCODE が破損しています。マルチブート イメージを読み込むと IDCODE エラーが発生し、フォールバックが開始します。

KCU116 ボードの SPI フラッシュをプログラムするには、Vivado Tcl コンソールで `Program_KCU116_SPI.tcl` を実行します。この Tcl ファイルは、プログラムする MCS ファイルに応じて編集が必要となることがあります。または、Vivado ツールのハードウェア マネージャーを使用して、表 2 に記載した MCS ファイルを SPIx4 MT25QU01 フラッシュにプログラムすることもできます。フラッシュ プログラミングの詳細は、『Vivado Design Suite ユーザー ガイド: プログラムおよびデバッグ』(UG908) [参照 2] を参照してください。

マルチブート動作の検証

フラッシュ デバイスにプログラムしたイメージ (`Golden_n_Multiboot.mcs`) で FPGA をブートするには、KCU116 ボードの SW5 を使用して PROGRAM_B をパルスします。FPGA が SPI フラッシュのゴールデン ビットストリームで正常にコンフィギュレーションされているかを検証します。検証方法は次のとおりです。

- ボード上の DONE ピン LED が点灯しているか。
- GPIO LED [0:7] が順番に点灯し、ゴールデン ビットストリームが正常に読み込まれたことを示しているか。
- UART ターミナルに図 12 のようなメッセージが表示されているか。

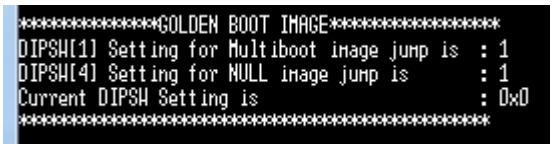


図 12: ゴールデン イメージの場合の UART の表示

- Vivado IDE で FPGA を右クリックして [Hardware Device Properties] をクリックし、デバイスをリフレッシュします。
- Vivado IDE の [Hardware Device Properties] ウィンドウで、[REGISTER] の下にある [BOOT_STATUS] と [CONFIG_STATUS] を展開表示します。BOOT_STATUS レジスタを見ると、通常のコンフィギュレーションに成功したことが確認できます。CONFIG_STATUS レジスタでは、DONE_PIN が High を示しています (図 13)。

REGISTER	
BOOT_STATUS	00000000000000000000000000000001
BIT00_0_STATUS_VALID	1
BIT01_0_FALLBACK	0
BIT02_0_INTERNAL_PROG	0
BIT03_0_WATCHDOG_TIMEOUT_ERROR	0
BIT04_0_ID_ERROR	0
BIT05_0_CRC_ERROR	0
BIT06_0_WRAP_ERROR	0
BIT07_0_SECURITY_ERROR	0
BIT08_1_STATUS_VALID	0
BIT09_1_FALLBACK	0
BIT10_1_INTERNAL_PROG	0
BIT11_1_WATCHDOG_TIMEOUT_ERROR	0
BIT12_1_ID_ERROR	0
BIT13_1_CRC_ERROR	0
BIT14_1_WRAP_ERROR	0
BIT15_1_SECURITY_ERROR	0
BIT16_RESERVED	0000000000000000
CONFIG_STATUS	00010000100100000111100111111100

X17891-050517

図 13: ゴールデン イメージの BOOT_STATUS

ゴールデン イメージが正常に読み込まれたことを確認したら、DIP SW13[1] をトグル (0 → 1 → 0) します。DONE LED が一瞬消灯し、マルチブート イメージが読み込まれます。IPROG が発行されると、UART に図 14 のようなメッセージが表示されます。

X17892-050517

図 14: IPROG が発行されたことを示す UART の表示

FPGA が SPI フラッシュのマルチブート ビットストリームで正常にコンフィギュレーションされているかを検証します。検証方法は次のとおりです。

- ボード上の DONE ピン LED が点灯しているか。
- GPIO LED [0:7] ですべての LED が点滅 (オン/オフ) し、マルチブート ビットストリームが正常に読み込まれたことを示しているか。
- UART ターミナルに図 15 のようなメッセージが表示されているか。

X17893-050517

図 15: マルチブート イメージの場合の UART の表示

- Vivado IDE で FPGA を右クリックして [Hardware Device Properties] をクリックし、デバイスをリフレッシュします。
- Vivado IDE の [Hardware Device Properties] ウィンドウで、[REGISTER] の下にある [BOOT_STATUS] と [CONFIG_STATUS] を展開表示します。BOOT_STATUS レジスタを見ると、マルチブート ビットストリームへのジャンプの要因となった IPROG (INTERNAL_PROG) フラグが High であることを確認できます。CONFIG_STATUS レジスタでは、DONE_PIN が High を示しています (図 16)。

REGISTER	
BOOT_STATUS	00000000000000000000000100000101
BIT00_0_STATUS_VALID	1
BIT01_0_FALLBACK	0
BIT02_0_INTERNAL_PROG	1
BIT03_0_WATCHDOG_TIMEOUT_ERROR	0
BIT04_0_ID_ERROR	0
BIT05_0_CRC_ERROR	0
BIT06_0_WRAP_ERROR	0
BIT07_0_SECURITY_ERROR	0
BIT08_1_STATUS_VALID	1
BIT09_1_FALLBACK	0
BIT10_1_INTERNAL_PROG	0
BIT11_1_WATCHDOG_TIMEOUT_ERROR	0
BIT12_1_ID_ERROR	0
BIT13_1_CRC_ERROR	0
BIT14_1_WRAP_ERROR	0
BIT15_1_SECURITY_ERROR	0
BIT16_RESERVED	0000000000000000
CONFIG_STATUS	00010000101100000111100111111100

X17894-050517

図 16: マルチブート イメージの BOOT_STATUS

マルチブート イメージが正常に読み込まれたことを確認したら、DIP SW13[1] を 1 にします。これによってデザインは FPGA IDCODE を読み出し、UART 経由で表示します (図 17)。

```
FPGA IDCODE is    :  4A62D93
```

X17895-050517

図 17: マルチブート イメージが FPGA IDCODE を読み出して UART に表示

フォールバックの例 - CRC エラー

フラッシュ デバイスにプログラムしたイメージ (`Golden_n_Multiboot_CRC_Err.mcs`) で FPGA をブートするには、KCU116 ボードの SW5 で PROGRAM_B をパルスします。ゴールデン イメージが正常に読み込まれたことを確認したら、DIP SW13[1] をトグル (0 → 1 → 0) します。DONE LED が一瞬消灯し、FPGA がマルチブート イメージの読み込みを試みます。このマルチブート イメージは破損しているため、FPGA はフォールバックしてゴールデン ビットストリームを読み込みます。

- Vivado IDE で FPGA を右クリックして [Hardware Device Properties] をクリックし、デバイスをリフレッシュします。
- Vivado IDE の [Hardware Device Properties] ウィンドウで、[REGISTER] の下にある [BOOT_STATUS] と [CONFIG_STATUS] を展開表示します。BOOT_STATUS レジスタを見ると、ジャンプの要因となった IPROG (INTERNAL_PROG)、CRC エラー、フォールバックのフラグが High であることが確認できます。CONFIG_STATUS レジスタでは、DONE_PIN が High を示しています (図 18)。

REGISTER	
BOOT_STATUS	00000000000000000010010100000011
BIT00_0_STATUS_VALID	1
BIT01_0_FALLBACK	1
BIT02_0_INTERNAL_PROG	0
BIT03_0_WATCHDOG_TIMEOUT_ERROR	0
BIT04_0_ID_ERROR	0
BIT05_0_CRC_ERROR	0
BIT06_0_WRAP_ERROR	0
BIT07_0_SECURITY_ERROR	0
BIT08_1_STATUS_VALID	1
BIT09_1_FALLBACK	0
BIT10_1_INTERNAL_PROG	1
BIT11_1_WATCHDOG_TIMEOUT_ERROR	0
BIT12_1_ID_ERROR	0
BIT13_1_CRC_ERROR	1
BIT14_1_WRAP_ERROR	0
BIT15_1_SECURITY_ERROR	0
BIT16_RESERVED	0000000000000000
CONFIG_STATUS	00010000101100000111100111111100

X17896-050517

図 18: CRC エラーの場合の BOOT_STATUS

フォールバックの例 - IDCODE エラー

フラッシュ デバイスにプログラムしたイメージ (`Golden_n_Multiboot_ID_Err.mcs`) で FPGA をブートするには、KCU116 ボードの SW5 で PROGRAM_B をパルスします。ゴールデン イメージが正常に読み込まれたことを確認したら、DIP SW13[1] をトグル (0 → 1 → 0) します。DONE LED が消灯し、FPGA がマルチブート イメージの読み込みを試みます。このマルチブート イメージの IDCODE は意図的に破損したものを使用しているため、FPGA はフォールバックしてゴールデン ビットストリームを読み込みます。

- Vivado IDE で FPGA を右クリックして [Hardware Device Properties] をクリックし、デバイスをリフレッシュします。
- Vivado IDE の [Hardware Device Properties] ウィンドウで、[REGISTER] の下にある [BOOT_STATUS] と [CONFIG_STATUS] を展開表示します。BOOT_STATUS レジスタを見ると、ジャンプの要因となった IPROG (INTERNAL_PROG)、ID エラー、フォールバックのフラグが High であることが確認できます。CONFIG_STATUS レジスタでは、DONE_PIN が High を示しています (図 19)。

REGISTER	
BOOT_STATUS	00000000000000000001010100000011
BIT00_0_STATUS_VALID	1
BIT01_0_FALLBACK	1
BIT02_0_INTERNAL_PROG	0
BIT03_0_WATCHDOG_TIMEOUT_ERROR	0
BIT04_0_ID_ERROR	0
BIT05_0_CRC_ERROR	0
BIT06_0_WRAP_ERROR	0
BIT07_0_SECURITY_ERROR	0
BIT08_1_STATUS_VALID	1
BIT09_1_FALLBACK	0
BIT10_1_INTERNAL_PROG	1
BIT11_1_WATCHDOG_TIMEOUT_ERROR	0
BIT12_1_ID_ERROR	1
BIT13_1_CRC_ERROR	0
BIT14_1_WRAP_ERROR	0
BIT15_1_SECURITY_ERROR	0
BIT16_RESERVED	0000000000000000
CONFIG_STATUS	00010000100100000111100111111100

X17897-050517

図 19: IDCODE エラーの場合の BOOT_STATUS

フォールバックの例 - ウォッチドッグ タイマー

フラッシュ デバイスにプログラムしたイメージ [1] で FPGA をブートするには、KCU116 ボードの SW5 で PROGRAM_B をパルスします。

ゴールデン イメージが正常に読み込まれたことを確認したら、DIP SW13[4] をトグル (0 → 1 → 0) します。DONE LED が消灯し、FPGA がマルチブート イメージの読み込みを試みます。ジャンプ先のアドレス `0x02000000` には有効なコンフィギュレーション イメージが存在しないため、FPGA ウォッチドッグ タイマーが約 15 秒後にタイムアウトし、フォールバックが開始します。FPGA はフォールバックしてゴールデン ビットストリームを読み込みます。

- Vivado IDE で FPGA を右クリックして [Hardware Device Properties] をクリックし、デバイスをリフレッシュします。
- Vivado IDE の [Hardware Device Properties] ウィンドウで、[REGISTER] の下にある [BOOT_STATUS] と [CONFIG_STATUS] を展開表示します。BOOT_STATUS レジスタを見ると、ジャンプの要因となった IPROG (INTERNAL_PROG)、ウォッチドッグ タイムアウト エラー、フォールバックのフラグが High であることが確認できます。CONFIG_STATUS レジスタでは、DONE_PIN が High を示しています (図 20)。

REGISTER	
BOOT_STATUS	00000000000000000000110100000011
BIT00_0_STATUS_VALID	1
BIT01_0_FALLBACK	1
BIT02_0_INTERNAL_PROG	0
BIT03_0_WATCHDOG_TIMEOUT_ERROR	0
BIT04_0_ID_ERROR	0
BIT05_0_CRC_ERROR	0
BIT06_0_WRAP_ERROR	0
BIT07_0_SECURITY_ERROR	0
BIT08_1_STATUS_VALID	1
BIT09_1_FALLBACK	0
BIT10_1_INTERNAL_PROG	1
BIT11_1_WATCHDOG_TIMEOUT_ERROR	1
BIT12_1_ID_ERROR	0
BIT13_1_CRC_ERROR	0
BIT14_1_WRAP_ERROR	0
BIT15_1_SECURITY_ERROR	0
BIT16_RESERVED	0000000000000000
CONFIG_STATUS	00010000101100000111100111111100

X17898-050517

図 20: ウォッチドッグ タイムアウト エラーの場合の BOOT_STATUS

1. イメージに使用する MCS ファイルは、付属のものならどれでもかまいません。

デバッグ

このセクションでは、SPI フラッシュを使用するマルチブートの一般的な問題をデバッグする際のチェックリストを示します。

デザイン

- ゴールデン イメージとマルチブート イメージの両方に対して、すべてのビットストリーム プロパティが適切に設定されているか (表 2 参照)。
- すべての SPI ビットストリーム プロパティが適切に設定されているか (表 2 参照)。
- フラッシュ プログラミング ファイルを生成するためのコマンドに適切なオプションとアドレス設定がすべて含まれているか。`Create_MCS.tcl` を参照してください。
- `Golden.c` で指定した IPROG ジャンプ先アドレスと、MCS ファイルを再生成する際に指定したアドレスが一致しているか。

ハードウェア

- MODE ピンがマスター SPI コンフィギュレーションに設定されているか。
- DIP SW13 が初期状態ですべて 0 に設定されているか。
- UART ボー レートが 9600 に設定されているか。
- デザインでフラッシュのプログラミングを実行する前に、フラッシュ デバイスが完全に消去されているか。消去されたかは、ブランク チェック オプションで確認できます。
- マルチブート リファレンス デザインのデバッグでは、BOOT_STATUS および CONFIG_STATUS レジスタを使用してエラーや動作の問題を確認します。レジスタを読み出す前に、デバイスのリフレッシュが完了していることを確認してください。
- SW16 がリファレンス デザインのリセットにマップされているか。このスイッチを押すとデザインがリセット状態になります。
- PROG_B SW5 スイッチが解放されているか。

まとめ

このアプリケーション ノートでは、フィールドでシステムをアップデートしたり異なるコンフィギュレーション イメージをリアルタイムで読み込んだりする場合に役立つ UltraScale+ FPGA のマルチブート機能を使用する方法について説明しました。SPI (x4) コンフィギュレーション インターフェイスでこの機能を実装する際の参考にしてください。マルチブート機能の動作を評価する際に使用できる KCU116 ボード用のリファレンス デザインが提供されています。

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 ファイルは、ザイリンクスのウェブサイトからダウンロードできます。表 3 に、リファレンス デザインの詳細を示します。

表 3: リファレンス デザインの詳細

パラメーター	説明
全般	
開発者	Guruprasad Kempahonnaiah
ターゲット デバイス	UltraScale+ FPGA
ソース コードの提供	あり
ソース コードの形式	Verilog、C
既存のザイリンクス アプリケーション ノート/リファレンス デザイン、またはサードパーティからデザインへのコード/IP の使用	N/A
シミュレーション	
論理シミュレーションの実施	N/A
タイミング シミュレーションの実施	N/A
論理シミュレーションおよびタイミング シミュレーションでのテストベンチの利用	N/A
テストベンチの形式	N/A
使用したシミュレータ/バージョン	N/A
SPICE/IBIS シミュレーションの実施	N/A
インプリメンテーション	
使用した合成ツール/バージョン	Vivado Design Suite 2017.1
使用したインプリメンテーション ツール/バージョン	Vivado Design Suite 2017.1 ザイリンクス SDK 2017.1
スタティック タイミング解析の実施	N/A
ハードウェア検証	
ハードウェア検証の実施	あり
使用したハードウェア プラットフォーム	KCU116 評価ボード

参考資料

注記: 日本語版のバージョンは、英語版より古い場合があります。

1. 『UltraScale アーキテクチャ コンフィギュレーション ユーザー ガイド』(UG570: 英語版、日本語版)
2. 『Vivado Design Suite ユーザー ガイド: プログラムおよびデバッグ』(UG908: 英語版、日本語版)
3. 『Vivado Design Suite Tcl コマンド リファレンス ガイド』(UG835: 英語版、日本語版)
4. 『KCU116 評価ボード ユーザー ガイド』(UG1239)
5. 『エンベデッド システム ツール リファレンス マニュアル』(UG1043: 英語版、日本語版)

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
2017 年 6 月 23 日	1.0	初版

法的通知

本通知に基づいて貴殿または貴社 (本通知の被通知者が個人の場合には「貴殿」、法人その他の団体の場合には「貴社」。以下同じ) に開示される情報 (以下「本情報」といいます) は、ザイリンクスの製品を選択および使用することのためにのみ提供されます。適用される法律が許容する最大限の範囲で、(1) 本情報は「現状有姿」、およびすべて受領者の責任で (with all faults) という状態で提供され、ザイリンクスは、本通知をもって、明示、黙示、法定を問わず (商品性、非侵害、特定目的適合性の保証を含みますがこれらに限られません)、すべての保証および条件を負わない (否認する) ものとします。また、(2) ザイリンクスは、本情報 (貴殿または貴社による本情報の使用を含む) に関係し、起因し、関連する、いかなる種類・性質の損失または損害についても、責任を負わない (契約上、不法行為上 (過失の場合を含む)、その他のいかなる責任の法理によるかを問わない) ものとし、当該損失または損害には、直接、間接、特別、付随的、結果的な損失または損害 (第三者が起こした行為の結果被った、データ、利益、業務上の信用の損失、その他あらゆる種類の損失や損害を含みます) が含まれるものとし、それは、たとえ当該損害や損失が合理的に予見可能であったり、ザイリンクスがそれらの可能性について助言を受けていた場合であったとしても同様です。ザイリンクスは、本情報に含まれるいかなる誤りも訂正する義務を負わず、本情報または製品仕様のアップデートを貴殿または貴社に知らせる義務も負いません。事前の書面による同意のない限り、貴殿または貴社は本情報を再生産、変更、頒布、または公に展示してはなりません。一定の製品は、ザイリンクスの限定的保証の諸条件に従うこととなるので、https://japan.xilinx.com/legal.htm#tos で見られるザイリンクスの販売条件を参照してください。IP コアは、ザイリンクスが貴殿または貴社に付与したライセンスに含まれる保証と補助的条件に従うことになります。ザイリンクスの製品は、フェイルセーフとして、または、フェイルセーフの動作を要求するアプリケーションに使用するために、設計されたり意図されたりしていません。そのような重大なアプリケーションにザイリンクスの製品を使用する場合のリスクと責任は、貴殿または貴社が単独で負うものです。https://japan.xilinx.com/legal.htm#tos で見られるザイリンクスの販売条件を参照してください。

自動車用のアプリケーションの免責条項

オートモーティブ製品 (製品番号に「XA」が含まれる) は、ISO 26262 自動車用機能安全規格に従った安全コンセプトまたは余剰性の機能 (「セーフティ設計」) がない限り、エアバッグの展開における使用または車両の制御に影響するアプリケーション (「セーフティ アプリケーション」) における使用は保証されていません。顧客は、製品を組み込むすべてのシステムについて、その使用前または提供前に安全を目的として十分なテストを行うものとします。セーフティ設計なしにセーフティ アプリケーションで製品を使用するリスクはすべて顧客が負い、製品の責任の制限を規定する適用法令および規則にのみ従うものとします。

この資料に関するフィードバックおよびリンクなどの問題につきましては、jpn_trans_feedback@xilinx.com まで、または各ページの右下にある [フィードバック送信] ボタンをクリックすると表示されるフォームからお知らせください。いただきましたご意見を参考に早急に対応させていただきます。なお、このメール アドレスへのお問い合わせは受け付けておりません。あらかじめご了承ください。