

HDMI 1.4/2.0 Transmitter Subsystem v2.0

製品ガイド

Vivado Design Suite

PG235 2017 年 4 月 5 日

この資料は表記のバージョンの英語版を翻訳したもので、内容に相違が生じる場合には原文を優先します。資料によっては英語版の更新に対応していないものがあります。日本語版は参考用としてご使用の上、最新情報につきましては、必ず最新英語版をご参照ください。

目次

IP の概要

第 1 章: 概要

アプリケーション	5
サポートされていない機能	5
ライセンスおよび注文情報	6

第 2 章: 製品仕様

規格	12
性能	12
リソース使用状況	12
ポートの説明	13
クロックとリセット	28

第 3 章: サブシステムを使用するデザイン

一般的なデザイン ガイドライン	29
インターレース ビデオ	34
クロッキング	36
リセット	39

第 4 章: デザイン フローの手順

サブシステムのカスタマイズおよび生成	40
サブシステムへの制約	46
シミュレーション	48
合成およびインプリメンテーション	48

第 5 章: サンプル デザイン

サンプル デザインを実行する	49
----------------------	----

付録 A: 検証、互換性、相互運用性

相互運用性	66
ハードウェア テスト	66
ビデオ解像度	67

付録 B: デバッグ

ザイリンクス ウェブサイト	71
デバッグ ツール	72
ハードウェア デバッグ	73
インターフェイスのデバッグ	73

付録 C: アプリケーション ソフトウェア開発

デバイスドライバ	75
----------------	----

付録 D: その他のリソースおよび法的通知

ザイリンクス リソース	88
参考資料	88
改訂履歴	89
お読みください: 重要な法的通知	90

はじめに

HDMI 1.4/2.0 Transmitter Subsystem は複数の HDMI® TX IP サブコアを一つにまとめた階層型 IP で、これらを一つの IP として出力します。HDMI 1.4/2.0 Transmitter Subsystem はそのままですぐに使用できるため、サブコアを一から組み立てることなく HDMI TX システムを構築して動作させることができます。

機能

- HDMI 2.0/1.4b 互換
- 1 つのクロック入力につき 2 または 4 シンボル/ピクセルをサポート
- 最大 4,096 x 2,160 @ 60fps の解像度をサポート
- 8、10、12、および 16 ビットのディープ カラーをサポート
- RGB、YUV 4:4:4、YUV 4:2:2、YUV 4:2:0 の色空間をサポート
- AXI4-Stream ビデオ入力ストリームとネイティブ ビデオ入力ストリームをサポート
- 最大 8 チャンルのオーディオをサポート
- InfoFrame
- データ ディスプレイ チャンネル (DDC)
- ホット プラグ検出
- 3D ビデオをサポート
- HDCP (High Bandwidth Digital Copy Protection) 1.4 をサポート (オプション)
- HDCP 2.2 をサポート (オプション)
- Video over AXIS 準拠の NTSC/PAL をサポート (オプション)
- Video over AXIS 準拠の YUV420 をサポート (オプション)
- HPD アクティブ極性をサポート (オプション)

この LogiCORE™ IP について	
サブシステムの概要	
サポートされるデバイスファミリ (1)	UltraScale+™ ファミリ (GTHE4) UltraScale™ アーキテクチャ (GTHE3) Zynq®-7000 All Programmable SoC 7 シリーズ (GTXE2、GTHE2) Artix®-7 (GTPE2)
サポートされるユーザーインターフェイス	AXI4-Lite、AXI4-Stream
リソース	Performance and Resource Utilization (ウェブ ページ)
サブシステムに含まれるもの	
デザイン ファイル	RTL
サンプル デザイン	Vivado IP インテグレーター
テストベンチ	なし
制約ファイル	XDC
シミュレーションモデル	なし
サポートされるソフトウェア ドライバー (2)	スタンドアロン
テスト済みデザイン フロー (3)	
デザイン入力	Vivado® Design Suite
シミュレーション	サポートされるシミュレータについては、 『Vivado Design Suite ユーザー ガイド: リリース ノート ガイド、インストール およびライセンス』 を参照してください。
合成	Vivado 合成
サポート	
ザイリンクス サポート ウェブ ページ で提供	

注記:

1. サポートされているデバイスの一覧は、Vivado IP カタログを参照してください。
2. スタンドアロンドライバーの詳細は、SDK ディレクトリ (<install_directory>/doc/usenglish/xilinx_drivers.htm) を参照してください。Linux OS およびドライバ サポートの情報は、[Xilinx Wiki ページ](#)を参照してください。
3. サポートされているツールのバージョンは、『[Vivado Design Suite ユーザー ガイド: リリース ノート ガイド、インストール およびライセンス](#)』を参照してください。

概要

HDMI 1.4/2.0 Transmitter Subsystem は、物理層 (PHY) と接続して HDMI® エンコード機能を実行するのに必要なロジックをすべて備えた高機能なソフト IP です。このサブシステムは階層型 IP で、HDMI TX に関連する複数のサブコアを 1 つの IP にまとめて出力します。これは入力ビデオおよびオーディオ ストリームを受け取り、HDMI ストリームに転送します。次に、このストリームをビデオ PHY 層へ転送します。

このサブシステムの性能と品質は、設計時に Vivado® 統合設計環境 (IDE) で完全にコンフィギュレーションできます。

アプリケーション

HDMI (High-Definition Multimedia Interface) はビデオとオーディオの転送に使用する一般的なインターフェイスとして、DVD、メディアプレーヤー、デジタルテレビ、デジタルビデオカメラ、モバイルタブレット、スマートフォンなどほとんどの民生ビデオ機器に採用されています。また、業務用カメラ、ビデオスイッチャー、コンバーター、モニター、およびビデオウォールや公共ディスプレイサインなど業務用途でも広く採用されています。

このサブシステムでテスト済みのビデオ解像度については、付録 A 「検証、互換性、相互運用性」を参照してください。

サポートされていない機能

このサブシステムでは、次の機能はサポートされていません。

- リップシンク
- CEC
- HEAC
- HDMI 2.0 デュアル表示
- HDMI 2.0 マルチストリームオーディオ

ライセンスおよび注文情報

ライセンス チェッカー

IP にライセンス キーが必要な場合、そのキーの認証が必要です。Vivado[®] デザイン ツールでは、設計フローにライセンスが必要な IP の使用をゲーティングする、ライセンス チェックポイントが複数あります。ライセンス チェックが正常に終了すると、IP の生成が継続されます。正常に終了しなければ、IP の生成はエラーとなり停止します。ライセンス チェックポイントが適用されるのは、次のツールです。

- Vivado 合成
- Vivado インプリメンテーション
- write_bitstream (Tcl コマンド)



重要: チェックポイントでは、IP のライセンス レベルは無視されます。有効なライセンスの有無のみを検証します。IP ライセンス レベルは確認しません。

ハードウェア評価ライセンスを使用している場合、コアはタイムアウト後に HDMI ストリームの送信を停止します。このタイムアウトはシステム CPU クロックに基づきます。たとえば、100MHz で動作するシステムでハードウェア評価ライセンスを使用した場合、通常動作を開始してから約 4 時間後に IP がタイムアウトします。

ライセンスの種類

このザイリンクス LogiCORE[™] IP モジュールは、[ザイリンクス コア ライセンス契約](#)の条件に基づいて提供されます。このモジュールは、Vivado[®] Design Suite に付属します。シミュレーションおよびハードウェアでサブシステムのすべての機能を利用するには、サブシステムのライセンスをご購入いただく必要があります。価格および提供状況については、[ザイリンクス販売代理店](#)にお問い合わせください。

詳細は、ザイリンクス サイトの [HDMI ウェブ ページ](#)を参照してください。

その他のザイリンクス LogiCORE IP モジュールに関する情報は、[ザイリンクス IP コア](#)のページから入手できます。その他のザイリンクス LogiCORE IP モジュールおよびツールの価格と提供状況については、[ザイリンクス販売代理店](#)にお問い合わせください。

製品仕様

この章では、サブシステムとそのパフォーマンスおよびリソース使用量の詳細について説明します。

HDMI 1.4/2.0 Transmitter Subsystem は階層型にパッケージされるため、Vivado® 統合設計環境 (IDE) のインターフェイスでパラメーターを設定すると、それに応じて必要なハードウェアがサブシステムによって生成されます。

図 2-1 に、HDMI 1.4/2.0 Transmitter Subsystem の概略ブロック図を示します。

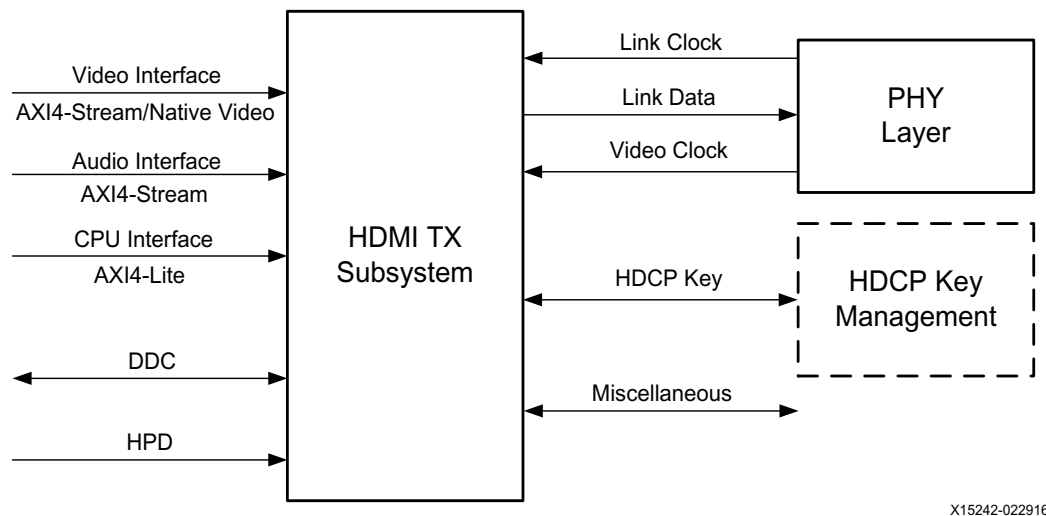


図 2-1: サブシステムのブロック図

HDMI TX Subsystem は HDMI TX コア の上位に構成されます。コンフィギュレーションパラメーターの設定に応じ、HDMI TX コアの周囲に各種のサポート モジュールが追加されます。HDMI TX コアはネイティブビデオインターフェイスをサポートするように設計されていますが、既存のビデオ処理 IP コアの多くは AXI4-Stream ベースです。このため、通常は AXI4-Stream ベースのビデオをサポートできるように Video Timing Controller、AXI4-Stream to Video Out Bridge などのサポート モジュールを追加して HDMI TX Subsystem を構成します。これにより、HDMI TX Subsystem をザイリンクスのほかのビデオ処理 IP コアとシームレスに連携させることができます。HDMI TX Subsystem は、オプション機能として HDCP 1.4 および HDCP 2.2 暗号化をサポートします。

図 2-2 に、Vivado IDE の [Video Interface] で [AXI4-Stream] を選択した場合の HDMI TX Subsystem の内部構造を示します。この図は、[Include HDCP 1.4 encryption]、[Include HDCP 2.2 encryption]、[Video over AXIS compliant NTSC/PAL Support]、[Video over AXIS compliant YUV420 Support] をすべてオンにした場合のもので、

HDMI 1.4/2.0 Transmitter Subsystem は、次の 2 つのビデオ インターフェイスをサポートしています。

- AXI4-Stream ビデオ インターフェイス
- ネイティブ ビデオ インターフェイス

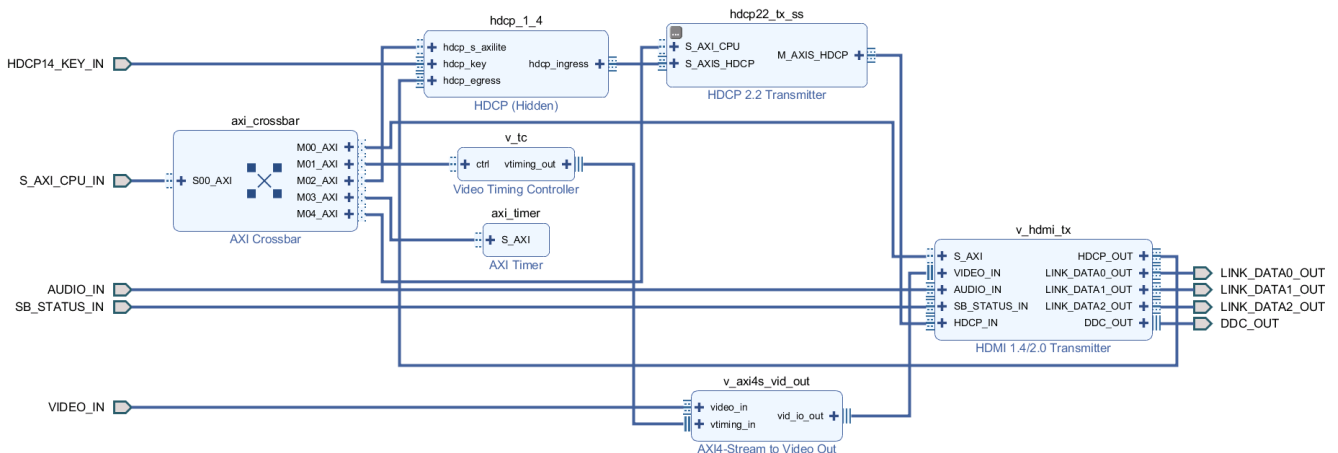


図 2-2: HDMI TX Subsystem の内部構造 (AXI4-Stream ビデオ インターフェイス モード)

HDMI TX Subsystem には、ネイティブ ビデオ インターフェイスをサポートするオプションもあります。Vivado IDE の [Video Interface] で [Native Video] を選択した場合、Video Timing Controller と AXI4-Stream to Video Out Bridge は HDMI TX Subsystem には含まれません。したがって、HDMI TX Subsystem は独自のビデオ デバイスからネイティブ ビデオを取り込んで HDMI 信号に変換できます。ネイティブ ビデオ モードでも、HDMI TX Subsystem はオプションで HDCP 1.4 および HDCP 2.2 暗号化をサポートできます。

図 2-3 に、Vivado IDE の [Video Interface] で [Native Video] を選択した場合の HDMI TX Subsystem の内部構造を示します。この図は、[Include HDCP 1.4 encryption] と [Include HDCP 2.2 encryption] の両方をオンにした場合のもので

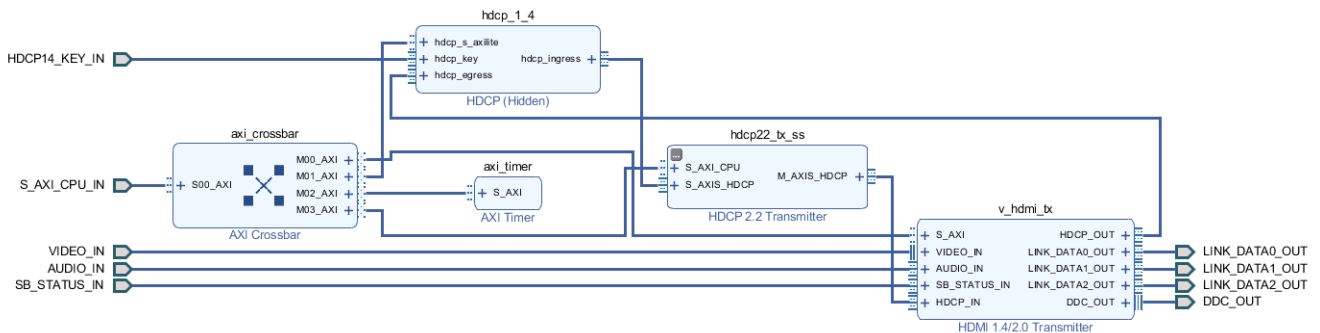


図 2-3: HDMI TX Subsystem の内部構造 (ネイティブ ビデオ インターフェイス モード)

ビデオ インターフェイスのデータ幅は、Vivado IDE のコンフィギュレーション パラメーター [Number of pixels per clock on Video Interface] と [Max bits per component] で設定します。

オーディオ インターフェイスは 32 ビット AXI4-Stream スレーブ バスで、複数チャネルの非圧縮オーディオ データをサブシステムに伝送します。

CPU インターフェイスは AXI4-Lite バス インターフェイスで、MicroBlaze™ または Zynq®-7000 SoC プロセッサに接続します。HDMI TX Subsystem は複数のサブモジュールで構成されており、ソフトウェア アクセスを必要とするサブモジュールはすべて AXI Crossbar で接続されます。したがって、MicroBlaze または Zynq-7000 SoC プロセッサは HDMI TX Subsystem 内の各サブモジュールに個別にアクセスして制御できます。



重要: これらのサブモジュールに対して直接レジスタ レベルでアクセスすることはできません。

HDMI TX Subsystem のデバイス ドライバーには API として抽象化したレイヤーがあり、これを利用して特定の機能をインプリメントします。この AXI4-Lite スレーブ インターフェイスは、シングルビートの読み出しおよび書き込みデータ転送をサポートします。バースト転送はサポートしません。

このサブシステムは、CPU インターフェイス経由でプロセッサコアが選択したビデオフォーマットに基づいてビデオストリームとオーディオストリームを HDMI ストリームに変換します。次に、このサブシステムは HDMI ストリームを物理層 (Video PHY Controller) に送信し、ここでデータを電気信号に変換してから HDMI ケーブルを介して HDMI シンクへ送信します。

以降のセクションでは、サブシステムがサポートする各機能について説明します。

オーディオ クロック再生成信号

トランスミッターのオーディオ パリフェラルには専用のオーディオ クロック再生成 (ACR) 入力インターフェイスがあります。

オーディオ クロックの再生成アーキテクチャは HDMI TX Subsystem の一部ではありません。ユーザーがアプリケーションにオーディオ クロックを供給する必要があります。このクロックは、オーディオ クロックの要件、オーディオ サンプル周波数およびジッターに応じて内部 PLL または外部クロック ソースを使用して供給します。HDMI TX Subsystem が DVI モードで使用されている場合は、ACR 入力は無視されます。設計者は、ACR 入力をオープンのままにするか、または一部の固定値に接続する (たとえば、acr_cts、acr_n、および acr_valid を 0 に接続する) かを選択できます。オーディオ パターン生成システムに含まれるサンプル ACR モジュールについては、[第 5 章「サンプル デザイン」](#) を参照してください。

データ ディスプレイ チャネル (DDC)

このサブシステムでは、エンド ユーザーの構築した HDMI ソース デバイスでターゲットの HDMI シンク デバイスとネゴシエーションを実行し、サポートされる機能と能力を決定できます。ソース デバイスとシンク デバイス間の通信は、DDC ライン (HDMI ケーブルに含まれる I2C バス) を用いて実装します。

ホット プラグ検出

このサブシステムは、HDMI ソース デバイスと HDMI シンク デバイス間の通信メカニズムの 1 つとしてホット プラグ検出 (HPD) 機能をサポートします。たとえば、HDMI ソース デバイスとシンク デバイスの間に HDMI ケーブルが挿入されると HPD 信号がアサートされ、これがトリガーになってサブシステムはシンク デバイスとの通信を開始します。

InfoFrame

すべての HDMI システムでは、補助ビデオ情報 (AVI) InfoFrame とオーディオ InfoFrame の 2 つの基本的な InfoFrame をサポートしています。どちらも HDMI TX Subsystem ドライバーで処理されます。HDMI TX Subsystem ドライバーには、3D ビデオなど特定の機能をサポートするためのベンダー固有 InfoFrame を構築および送信する機能があります。InfoFrame の詳細は、CEA-861-F を参照してください。

HDMI TX Subsystem ドライバーには、独自の InfoFrame を定義するための API が追加されています。ガイドラインとして、InfoFrame は 4 バイトのヘッダーと 32 バイトのデータ (ペイロード) で構成されます。InfoFrame API 関数呼び出しを送信する前にヘッダーとペイロードを構築しておく必要があります。また、HDMI シンクが InfoFrame を正しくデコードできるように、独自に CRC を計算して適切な位置に配置しておく必要があります。

次に、関数呼び出しの例を示します。

```
XV_HdmiTxSs_SendGenericAuxInfoframe (HdmiTxSsPtr, AuxPtr);
```

HdmiTxSsPtr は HDMI TX Subsystem へのポインターで、AuxPtr は InfoFrame のヘッダーとデータが格納されている配列へのポインターです。

図 2-4 は、HDMI データ アイランド パケットの一種である HDMI InfoFrame の構造を示しています。HDMI では、すべてのデータ アイランド パケットは 4 バイトのパケット ヘッダーと 32 バイトのパケット内容で構成されます。パケット ヘッダーには 24 ビット (3 バイト) のデータと 8 ビット (1 バイト) の BCH ECC パリティが含まれます。

Byte\Bit #	7	6	5	4	3	2	1	0
HB0	Packet Type							
HB1	packet-specific data							
HB2	packet-specific data							
ECC	ECC							

図 2-4: パケット ヘッダー

図 2-5 に示すパケット本体は、4つのサブパケットで構成されます。各サブパケットには 56 ビット (7 バイト) のデータと 8 ビット (1 バイト) の BCH ECC パリティが含まれます。

Subpacket #	Byte\Bit #	7	6	5	4	3	2	1	0
Subpacket0 (Checksum + 6 data bytes + ECC)	PB0	Checksum							
	PB1	Data Byte 1							
	PB2...PB5	Data Byte 2 - Data Byte 5							
	PB6	Data Byte 6							
	ECC1	ECC							
Subpacket1 (7 Data Bytes + ECC)	PB7	Data Byte 7							
	PB8...PB12	Data Byte 8 - Data Byte 12							
	PB13	Data Byte 13							
	ECC2	ECC							
Subpacket2 (7 Data Bytes + ECC)	PB14	Data Byte 14							
	PB15...PB19	Data Byte 15 - Data Byte 19							
	PB20	Data Byte 20							
	ECC3	ECC							
Subpacket3 (7 Data Bytes + ECC)	PB21	Data Byte 21							
	PB22...PB26	Data Byte 22 - Data Byte 26							
	PB27	Data Byte 27							
	ECC3	ECC							

図 2-5: パケット本体

注記:

1. ECC は HDMI 1.4/2.0 Transmitter Subsystem コア内で計算されます。したがって、ソフトウェア内で HDMI 仕様に従って HB0...HB2、および PB0、PB1...PB26、PB27 を構築する必要があります。
2. チェックサム値 (PB0) を計算する際は、ECC の値は無視されます。

InfoFrame の構造の詳細は、HDMI 1.4 仕様 [参照 10] のセクション 5.2.3.4 および 5.2.3.5 を参照してください。

HDCP

HDMI TX Subsystem に含まれるザイリンクス LogiCORE™ IP HDCP™ (High-bandwidth Digital Content Protection) コアのトランスミッターは、HDCP に対応した 2 つのデバイス間でオーディオ/ビジュアル コンテンツをセキュアに転送するために使用します。HDMI TX Subsystem には、HDCP 1.4 IP コアと HDCP 2.2 IP コアの両方のトランスミッターが含まれます。ただし、HDCP 2.2 は HDCP 1.4 プロトコルを置き換えるもので、下位互換性はないため、どちらのコンテンツ保護方式を使用するかを Vivado IDE で選択する必要があります。選択肢として次の 4 とおりがります。

- HDCP なし
- HDCP 1.4 のみ
- HDCP 2.2 のみ
- HDCP 1.4 および HDCP 2.2

目安として、HDCP 2.2 は UHD (Ultra-High Definition) 解像度のコンテンツを暗号化する場合に使用し、HDCP 1.4 はそれよりも低解像度のレガシ コンテンツを保護する場合に使用します。

図 2-6 に、HDCP 1.4 と HDCP 2.2 を両方有効にした場合の HDMI TX Subsystem の構成を示します。両方の HDCP プロトコルを有効にした場合、HDMI Subsystem は HDCP 1.4 と HDCP 2.2 を数珠つなぎにしたカスケード接続トポロジーとして自身をコンフィギュレーションします。HDMI TX コアの HDCP Egress インターフェイスからは暗号化されていない A/V データが送出され、これをアクティブな HDCP ブロックで暗号化した後、HDCP Ingress インターフェイス経由で HDMI TX コアに戻してからリンクに送信します。両方の HDCP プロトコルが同時にアクティブにならないように、HDMI TX Subsystem は適切な HDMI TX Subsystem API 関数を呼び出して片方の HDCP プロトコルをパッシブにします。

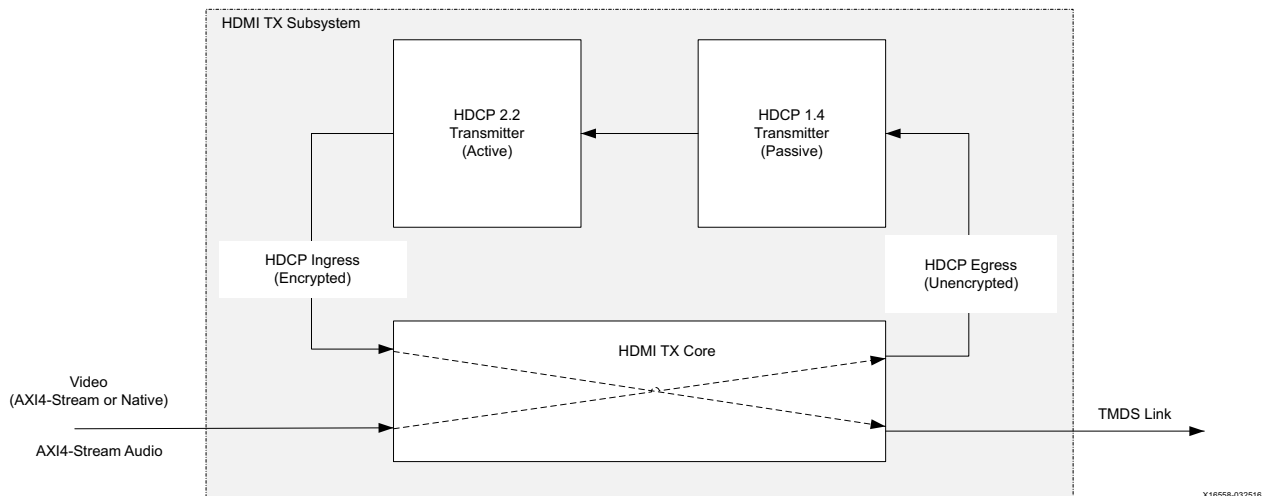


図 2-6: HDMI TX コアと HDCP 1.4 および HDCP 2.2 の組み合わせ

InfoFrame の詳細は、『HDCP v1.4 製品ガイド』(PG224) [参照 24] および『HDCP v2.2 製品ガイド』(PG249) [参照 23] を参照してください。

規格

HDMI 1.4/2.0 Transmitter Subsystem は、AXI4-Stream ビデオ プロトコルおよび AXI4-Lite インターコネクト規格に準拠しています。詳細は、『Vivado Design Suite AXI: リファレンス ガイド』(UG1037) [参照 1] を参照してください。HDMI 仕様 [参照 10] も参照してください。

HDMI TX Subsystem は HDMI 1.4b および HDMI 2.0 仕様 [参照 10] に準拠しています。

ザイリンクスの HDCP 1.4 コアは、『High-bandwidth Digital Content Protection system Revision 1.4』[参照 11] に準拠しています。

ザイリンクスの HDCP 2.2 は、Digital Content Protection (DCP) LLC が発行した HDCP 2.2 仕様『High-bandwidth Digital Content Protection, Mapping HDCP to HDMI, Revision 2.2』[参照 11] に準拠しています。

性能

性能およびリソース使用状況の詳細は、[Performance and Resource Utilization \(ウェブ ページ\)](#) をご覧ください。

最大周波数

DC 特性および AC スイッチ特性の詳細は、次の文書を参照してください。トランシーバーとコアを正しく動作させるには、これらデータシートに記載された周波数の範囲に従う必要があります。

- 『Kintex UltraScale FPGA データシート: DC 特性および AC スイッチ特性』(DS892) [参照 2]
- 『Virtex UltraScale FPGA データシート: DC 特性および AC スイッチ特性』(DS893) [参照 3]
- 『Kintex-7 FPGA データシート: DC 特性および AC スイッチ特性』(DS182) [参照 4]
- 『Virtex-7 T/XT FPGA データシート: DC 特性および AC スイッチ特性』(DS183) [参照 5]
- 『Artix-7 FPGA データシート: DC 特性および AC スイッチ特性』(DS181) [参照 6]
- 『Kintex UltraScale+ FPGA データシート: DC 特性および AC スイッチ特性』(DS922) [参照 7]
- 『Virtex UltraScale+ FPGA データシート: DC 特性および AC スイッチ特性』(DS923) [参照 8]
- 『Zynq UltraScale+ MPSoC データシート: DC 特性および AC スイッチ特性』(DS925) [参照 9]

リソース使用状況

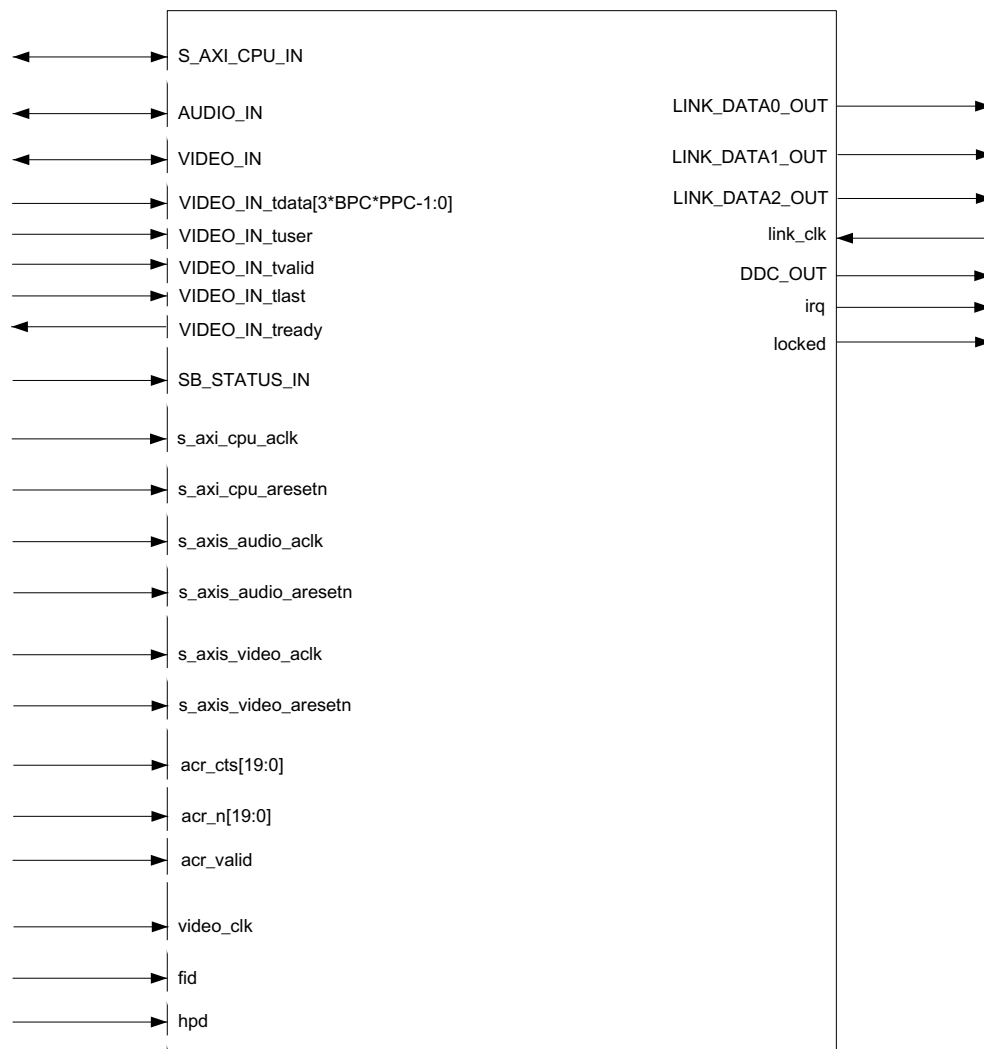
性能およびリソース使用状況の詳細は、[Performance and Resource Utilization \(ウェブ ページ\)](#) をご覧ください。

ポートの説明

図 2-7 ~ 図 2-10 に、AXI4-Stream ビデオ インターフェイスを選択した場合の HDMI 1.4/2.0 Transmitter Subsystem のポートを示します。AXI4-Stream ビデオ バス信号の詳細を示すために、VIDEO_IN ポートのみ展開して示しています。

このサブシステムには、デフォルトで次の 3 つのインターフェイスがあります。

- AXI4-Lite 制御インターフェイス (S_AXI_CPU_IN)
- ビデオ インターフェイス (VIDEO_OUT)
- オーディオ インターフェイス (AUDIO_OUT)



X15253-032516

図 2-7: HDMI TX Subsystem のピン配置 – AXI4-Stream ビデオ インターフェイス、HDCP なしの場合

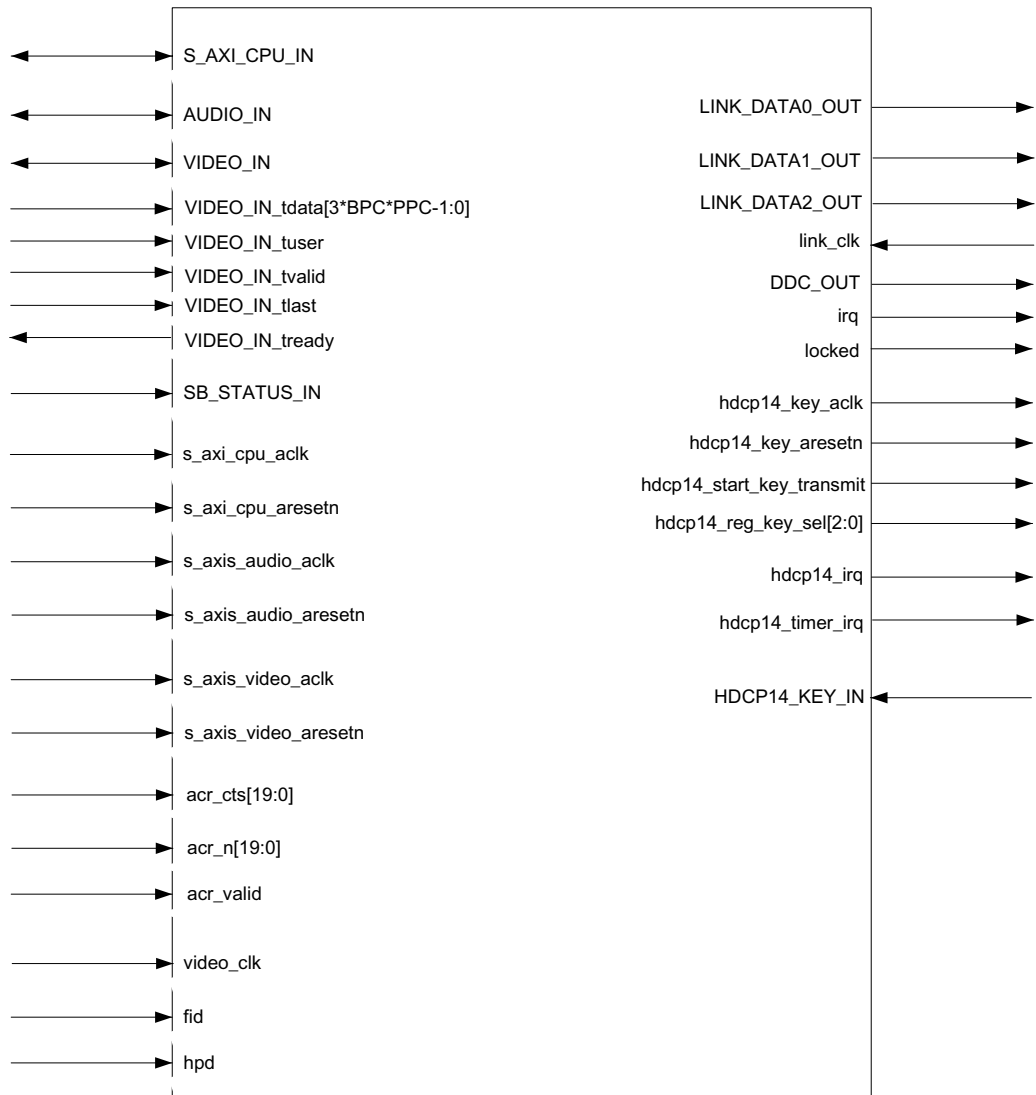
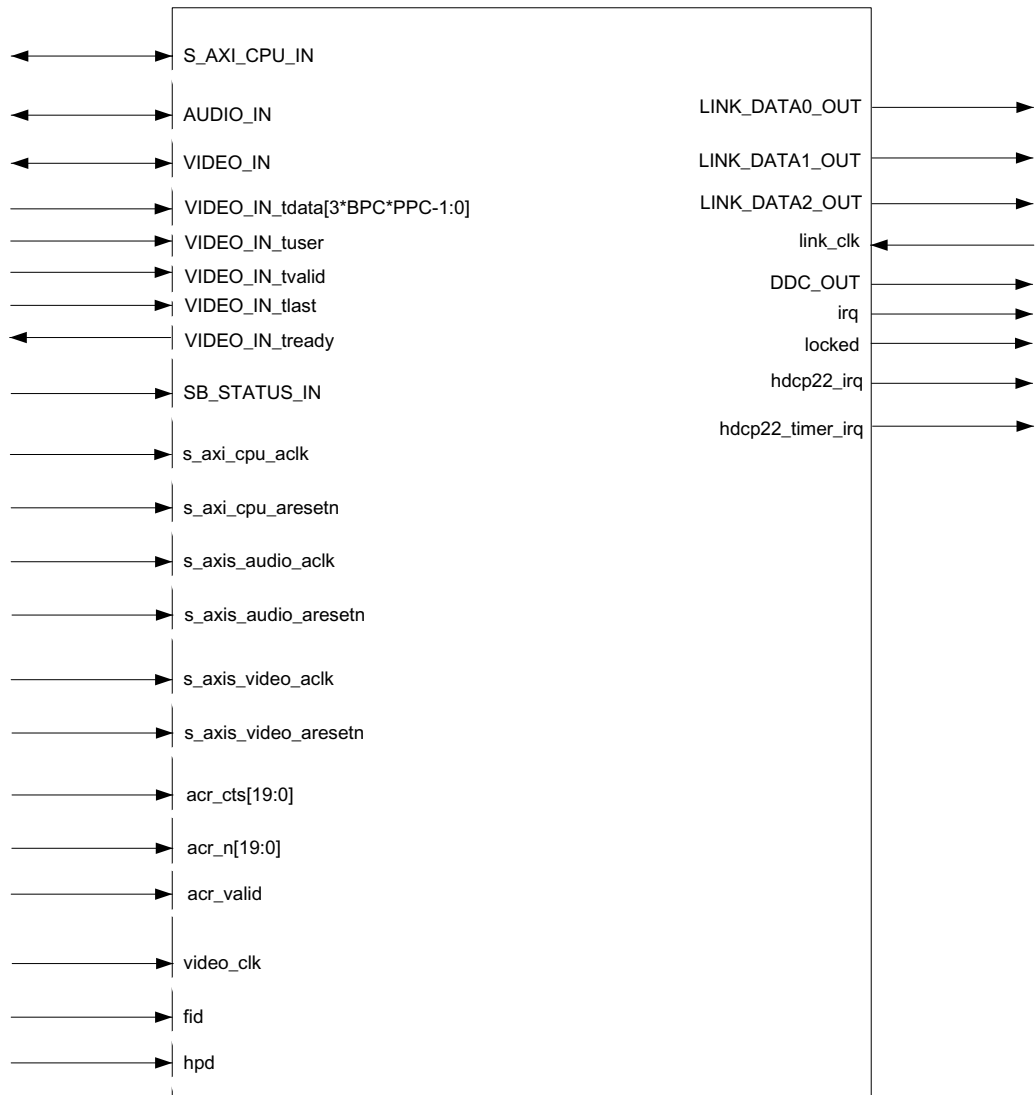


図 2-8: HDMI TX Subsystem のピン配置 – AXI4-Stream ビデオ インターフェイス、HDCP 1.4 のみの場合



X16560-032516

図 2-9: HDMI TX Subsystem のピン配置 – AXI4-Stream ビデオ インターフェイス、HDCP 2.2 のみの場合

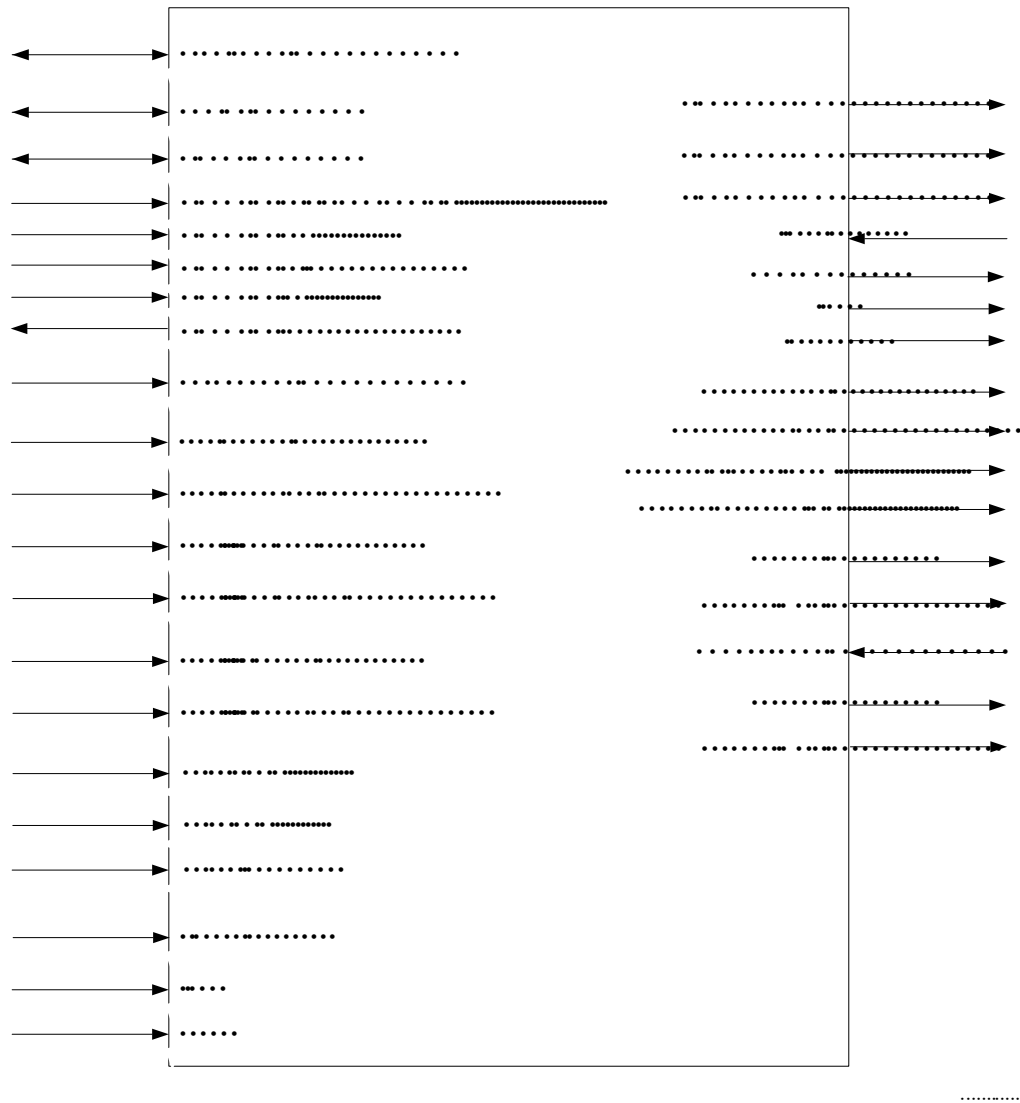
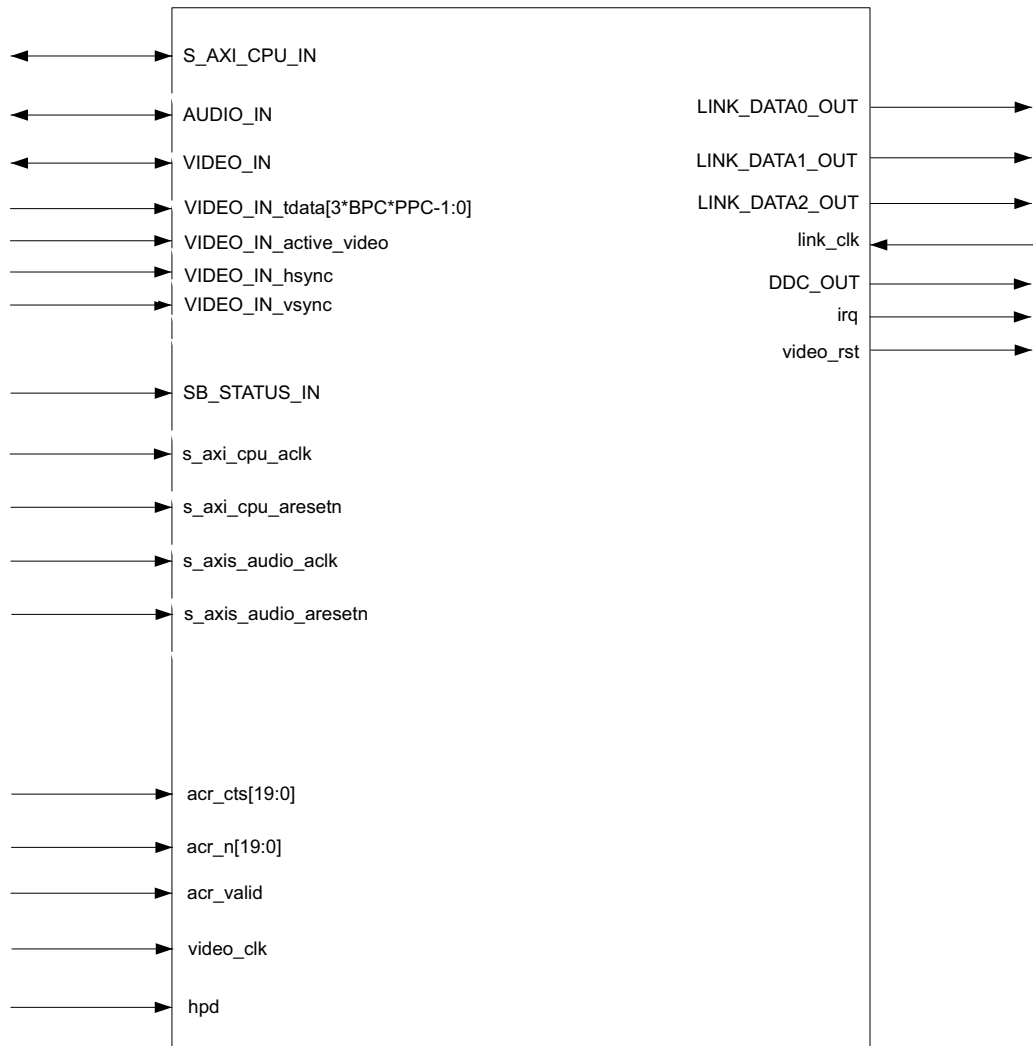


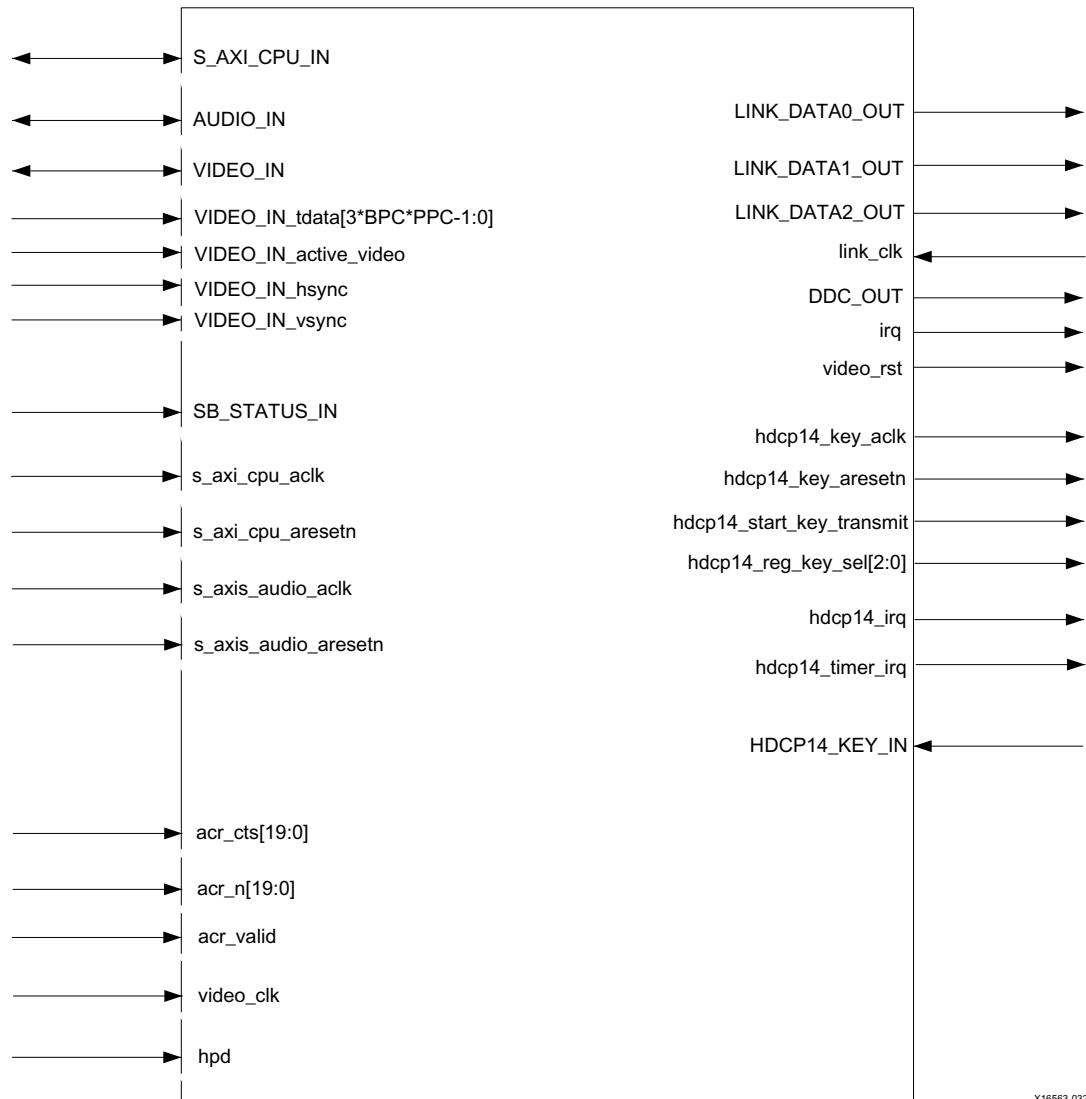
図 2-10: HDMI TX Subsystem のピン配置 – AXI4-Stream ビデオ インターフェイス、HDCP 1.4 および HDCP 2.2 の場合

図 2-11 ~ 図 2-14 に、ネイティブ ビデオ インターフェイスを選択した場合の HDMI 1.4/2.0 Transmitter Subsystem のポートを示します。ネイティブ ビデオ パス信号の詳細を示すために、VIDEO_IN ポートのみ展開して示しています。



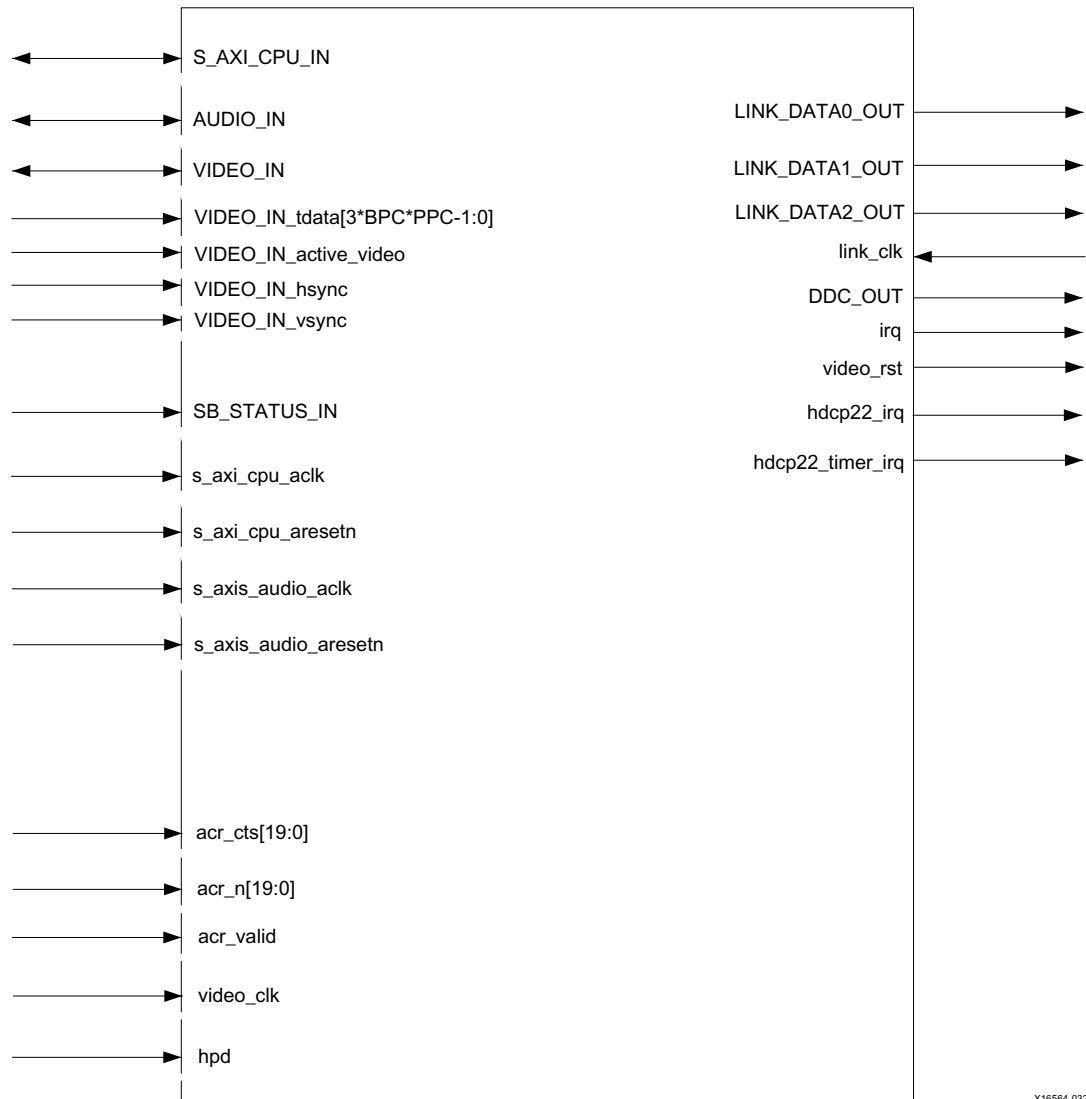
X16562-032516

図 2-11: HDMI TX Subsystem のピン配置 – ネイティブ ビデオ インターフェイス、HDCP なしの場合



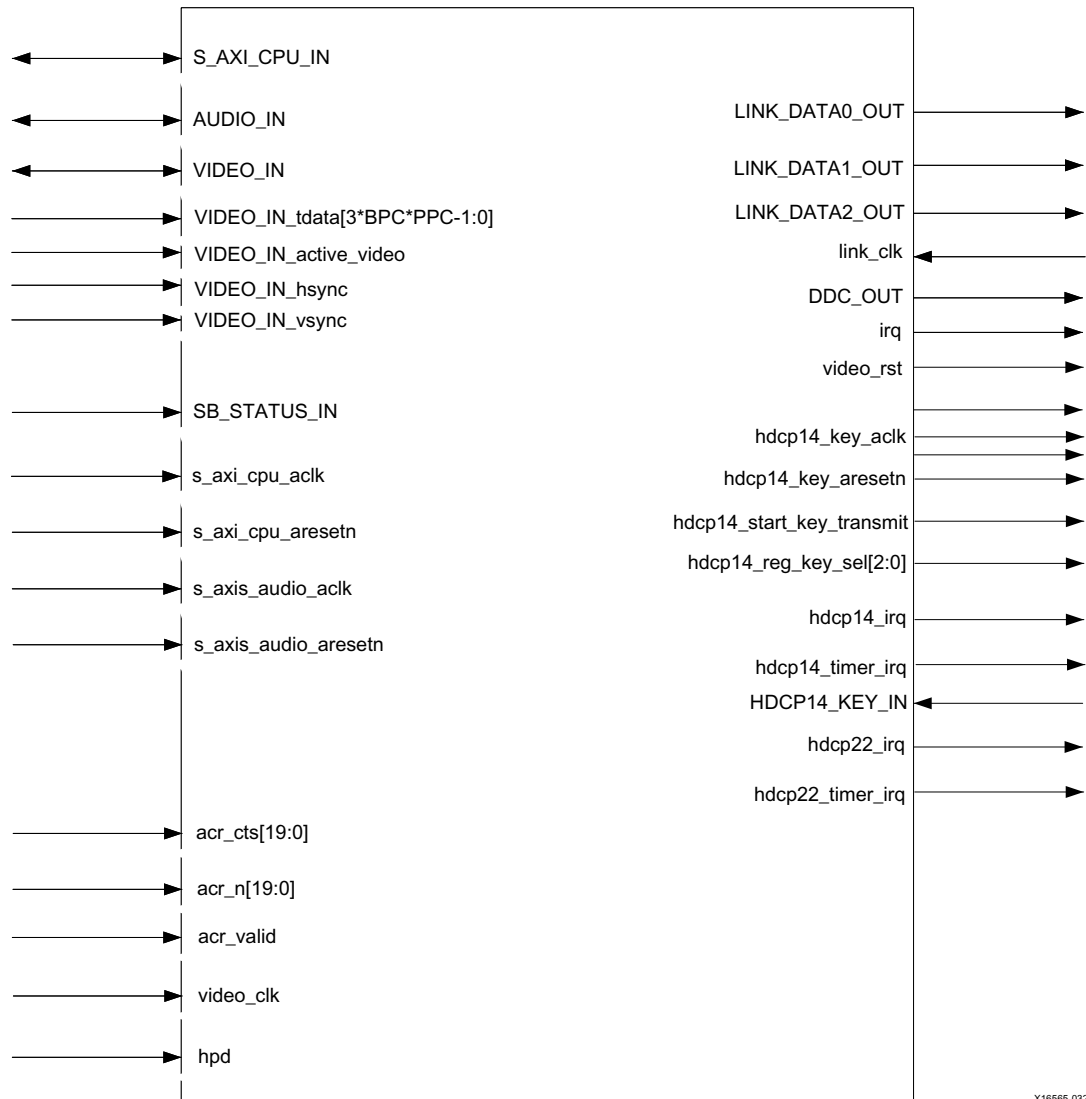
X16563-032516

図 2-12: HDMI TX Subsystem のピン配置 – ネイティブ ビデオ インターフェイス、HDCP 1.4 のみの場合



X16564-032516

図 2-13: HDMI TX Subsystem のピン配置 – ネイティブ ビデオ インターフェイス、HDCP 2.2 のみの場合



X16565-032516

図 2-14: HDMI TX Subsystem のピン配置 – ネイティブ ビデオ インターフェイス、HDCP 1.4 および HDCP 2.2 の場合

CPU インターフェイス

表 2-1 に、AXI4-Lite 制御インターフェイスの信号を示します。このインターフェイスは AXI4-Lite インターフェイスで、s_axi_cpu_aclk クロック レートで動作します。サブシステムの制御は、サブシステム ドライバー経由でのみサポートされます。



重要: これらのサブモジュールに対して直接レジスタ レベルでアクセスすることはできません。すべてのアクセスはドライバー API を介して実行されます。

表 2-1: CPU インターフェイスのポート

名称	方向	幅	説明
s_axi_cpu_aresetn	入力	1	リセット (アクティブ Low)
s_axi_cpu_aclk	入力	1	AXI4-Lite 制御インターフェイス用クロック
S_AXI_CPU_IN_awaddr	入力	18	書き込みアドレス
S_AXI_CPU_IN_awprot	入力	3	書き込みアドレス保護
S_AXI_CPU_IN_awvalid	入力	1	書き込みアドレスの Valid 信号
S_AXI_CPU_IN_awready	出力	1	書き込みアドレスの Ready 信号
S_AXI_CPU_IN_wdata	入力	32	書き込みデータ
S_AXI_CPU_IN_wstrb	入力	4	書き込みデータのストロブ
S_AXI_CPU_IN_wvalid	入力	1	書き込みデータの Valid 信号
S_AXI_CPU_IN_wready	出力	1	書き込みデータの Ready 信号
S_AXI_CPU_IN_bresp	出力	2	書き込み応答
S_AXI_CPU_IN_bvalid	出力	1	書き込み応答の Valid 信号
S_AXI_CPU_IN_bready	入力	1	書き込み応答の Ready 信号
S_AXI_CPU_IN_araddr	入力	18	読み出しアドレス
S_AXI_CPU_IN_arprot	入力	3	読み出しアドレス保護
S_AXI_CPU_IN_arvalid	入力	1	読み出しアドレスの Valid 信号
S_AXI_CPU_IN_aready	出力	1	読み出しアドレスの Ready 信号
S_AXI_CPU_IN_rdata	出力	32	読み出しデータ
S_AXI_CPU_IN_rresp	出力	2	読み出しデータの応答信号
S_AXI_CPU_IN_rvalid	出力	1	読み出しデータの Valid 信号
S_AXI_CPU_IN_rready	入力	1	読み出しデータの Ready 信号

ビデオ入カストリーム インターフェイス

HDMI 1.4/2.0 Transmitter Subsystem は次に示す 2 種類のビデオ入カストリーム インターフェイスをサポートしていますが、これらは最終的に HDMI 1.4/2.0 Transmitter Subsystem VIDEO_IN インターフェイスにマップされます。

- AXI4-Stream ビデオ インターフェイス
- ネイティブ ビデオ インターフェイス

表 2-2 に、AXI4-Stream ビデオ入カストリーミング インターフェイスの信号を示します。このインターフェイスは AXI4-Stream スレーブ インターフェイスで、s_axis_video_aclk クロック レートで動作します。データ幅は、Vivado IDE の [Max bits per component] (BPC) と [Number of pixels per clock on Video Interface] (PPC) で設定できます。

表 2-2: ビデオ入カストリーム インターフェイス

名称	方向	幅	説明
s_axis_video_aclk	入力	1	AXI4-Stream クロック
s_axis_video_aresetn	入力	1	リセット (アクティブ Low)
VIDEO_IN_tdata	入力	3*BPC*PPC	データ
VIDEO_IN_tlast	入力	1	ライン終了
VIDEO_IN_tready	出力	1	Ready 信号
VIDEO_IN_tuser	入力	1	フレーム開始
VIDEO_IN_tvalid	入力	1	Valid 信号

ネイティブ ビデオ入カインターフェイス

表 2-3 に、ネイティブ ビデオ入カインターフェイスの信号を示します。このインターフェイスは標準ビデオインターフェイスで、video_clk クロック レートで動作します。データ幅は、Vivado IDE の [Max bits per component] (BPC) と [Number of pixels per clock on Video Interface] (PPC) で設定できます。

表 2-3: ネイティブ ビデオ入カインターフェイス

名称	方向	幅	説明
video_clk	入力	1	ビデオ クロック
VIDEO_IN_active_video	入力	1	アクティブ ビデオ
VIDEO_IN_data	入力	3*BPC*PPC	データ
VIDEO_IN_hsync	入力	1	水平同期
VIDEO_IN_vsync	入力	1	垂直同期

注記:

1. ネイティブ ビデオ インターフェイスを選択した場合、s_axis_video_aclk と s_axis_video_aresetn は HDMI 1.4/2.0 Transmitter Subsystem のインターフェイス ポートから除外されます。
2. video_clk は Video PHY Controller によって生成されます。詳細は、『Video PHY Controller LogiCORE IP 製品ガイド』(PG230) [参照 22] を参照してください。
3. ネイティブ ビデオ インターフェイスを選択した場合、ハードウェア リセットはありません。

オーディオ入カストリーム インターフェイス

表 2-4 に、AXI4-Stream オーディオ入カストリーミング インターフェイスの信号を示します。オーディオ インターフェイスは 24 ビット オーディオ サンプルを IEC 60958 フォーマットで伝送します。最大 8 チャンネルがサポートされます。オーディオ インターフェイスは 32 ビット AXI4-Stream スレーブ インターフェイスで、s_axis_audio_aclk クロック レートで動作します。

表 2-4: オーディオ入カストリーム インターフェイス

名称	方向	幅	説明
s_axis_audio_aclk	入力	1	クロック (オーディオ ストリーミング クロックは、オーディオ サンプル周波数の 128 倍以上とする必要があります)
s_axis_audio_aresetn	入力	1	リセット (アクティブ Low)
AUDIO_IN_tdata	入力	32	データ [31] P (パリティ) [30] C (チャンネル ステータス) [29] U (ユーザー ビット) [28] V (Validity ビット) [27:4] オーディオ サンプル ワード [3:0] PR (プリアンプル コード) 4'b0001 サブフレーム 1/オーディオ ブロックの開始 4'b0010 サブフレーム 1 4'b0011 サブフレーム 2
AUDIO_IN_tid	入力	3	チャンネル ID
AUDIO_IN_tready	出力	1	Ready 信号
AUDIO_IN_tvalid	入力	1	Valid 信号

オーディオ クロック再生成インターフェイス

オーディオ クロック再生成 (ACR) インターフェイスにはサイクル タイム スタンプ (CTS) パラメーター ベクターとオーディオ クロック再生成値 (N) パラメーター ベクターがあります。どちらも 20 ビット幅です。CTS および N パラメーターが安定すると、Valid 信号が High に駆動されます。詳細は、HDMI 1.4 仕様 [\[参照 10\]](#) の第 7 章を参照してください。

Valid 信号の立ち上がりエッジで TX は ACR 入力インターフェイスから CTS および N パラメーターを読み出し、オーディオ クロック再生成パケットを送信します。

表 2-5 に、オーディオ クロック再生成 (ACR) インターフェイスの信号を示します。このインターフェイスは s_axis_audio_aclk のクロック レートで動作します。

表 2-5: オーディオ クロック再生成 (ACR) インターフェイス

名称	方向	幅	説明
acr_cts	入力	20	CTS
acr_n	入力	20	N
acr_valid	入力	1	Valid 信号

HDMI リンク出カインターフェイス

表 2-6 に、HDMI リンク出力インターフェイスの信号を示します。このインターフェイスは link_clk クロックレートで動作します。

表 2-6: HDMI リンク出カインターフェイス

名称	方向	幅	説明
link_clk	入力	1	リンク クロック
LINK_DATA0_OUT_tdata	出力	40	リンク データ 0
LINK_DATA0_OUT_tvalid	出力	1	リンク データ 0 の Valid 信号
LINK_DATA1_OUT_tdata	出力	40	リンク データ 1
LINK_DATA1_OUT_tvalid	出力	1	リンク データ 1 の Valid 信号
LINK_DATA2_OUT_tdata	出力	40	リンク データ 2
LINK_DATA2_OUT_tvalid	出力	1	リンク データ 2 の Valid 信号

データ ディスプレイ チャネル インターフェイス

表 2-7 に、データ ディスプレイ チャネル (DDC) インターフェイスの信号を示します。

表 2-7: データ ディスプレイ チャネル (DDC) インターフェイス

名称	方向	幅	説明
ddc_scl_i	入力	1	DDC シリアル クロック入力
ddc_scl_o	出力	1	DDC シリアル クロック出力
ddc_scl_t	出力	1	DDC シリアル クロック トライステート
ddc_sda_i	入力	1	DDC シリアル データ入力
ddc_sda_o	出力	1	DDC シリアル データ出力
ddc_sda_t	出力	1	DDC シリアル データ トライステート

HDCP 1.4 キー入カインターフェイス (AXI4-Stream スレーブ インターフェイス)

表 2-8 に、HDCP 1.4 キー インターフェイスの信号を示します。このインターフェイスは hdcpl4_key_aclk のクロックレートで動作します。

表 2-8: HDCP 1.4 キー入カインターフェイス

名称	方向	幅	説明
HDCP_KEY_IN_tdata	入力	64	HDCP 1.4 キー データ
HDCP_KEY_IN_tlast	入力	1	キー データ終了
HDCP_KEY_IN_tready	出力	1	Ready 信号
HDCP_KEY_IN_tuser	入力	8	キー データ開始
HDCP_KEY_IN_tvalid	入力	1	Valid 信号
hdcpl4_key_aclk	出力	1	AXI4-Stream クロック
hdcpl4_key_aresetn	出力	1	リセット (アクティブ Low)

表 2-8: HDCP 1.4 キー入力インターフェイス (続き)

名称	方向	幅	説明
hdcp14_start_key_transmit	出力	1	キー送信開始
hdcp14_reg_key_sel	出力	3	キー選択
hdcp14_irq	出力	1	HDCP 1.4 割り込み
hdcp14_timer_irq	出力	1	HDCP 1.4 タイマー割り込み

HDCP 1.4 のトランスミッターには、キーを AXI4-Stream インターフェイス経由で HDCP 1.4 コントローラーに送信するための HDCP キー管理モジュールが必要です。図 2-15 に、HDMI TX Subsystem をキー管理バス (AXI4-Stream) 経由で HDCP キー管理モジュールに接続する方法を例として示します。HDCP キー管理モジュールは HDMI TX Subsystem には含まれません。HDCP 1.4 デザインの詳細は、『HDCP v1.4 製品ガイド』(PG224) [参照 24] を参照してください。

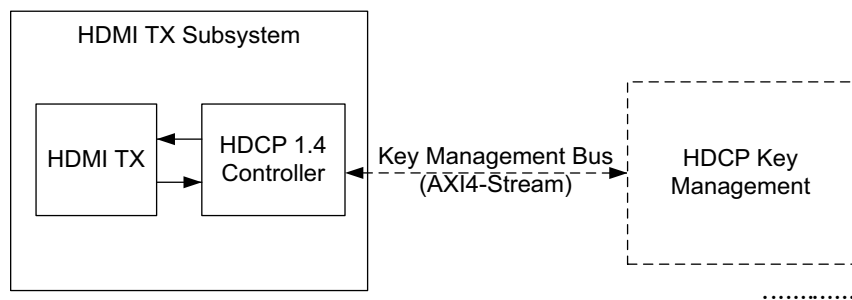


図 2-15: HDCP 1.4 キー管理バス (AXI4-Stream)

これに対し、HDCP 2.2 のキーの取り扱い方法はやや異なり、ソフトウェアアプリケーションのみで制御します。HDCP 2.2 ドライバーにロードするキーを安全に格納および取り出すためのインフラストラクチャは、ユーザーアプリケーションで用意する必要があります。ユーザーアプリケーションを使用してロードする必要のあるキーの詳細は、『HDCP v2.2 製品ガイド』(PG249) [参照 23] を参照してください。

HDCP 2.2 割り込み出力

表 2-9 に、HDCP 2.2 割り込み出力ポートの信号を示します。

表 2-9: HDCP 2.2 割り込み出力インターフェイス

名称	方向	幅	説明
hdcp22_irq	出力	1	HDCP 2.2 割り込み
hdcp22_timer_irq	出力	1	HDCP 2.2 タイマー割り込み

AXI4-Stream ビデオ インターフェイスのその他の信号

表 2-10 に、AXI4-Stream ビデオ インターフェイスを選択した場合のその他の信号を示します。

表 2-10: AXI4-Stream ビデオ インターフェイスのその他の信号

名称	方向	幅	説明
hpd	入力	1	XGUI オプションで [Hot Plug Detect Active] が [High] (デフォルト) の場合 0 - ホット プラグ検出がディアサートされている 1 - ホット プラグ検出がアサートされている XGUI オプションで [Hot Plug Detect Active] が [Low] の場合 ⁽¹⁾ 0 - ホット プラグ検出がアサートされている 1 - ホット プラグ検出がディアサートされている
locked	出力	1	サブシステムが入力ビデオ ストリームにロックしているかどうかを示すフラグ。 0 - ロックしていない 1 - ロックしている
irq	出力	1	CPU に対する割り込み要求。アクティブ High。
video_clk	入力	1	ネイティブ ビデオの基準クロック [Video Interface] で [AXI4-Stream] を選択した場合、HDMI TX Subsystem に追加される AXI4-Stream to Video Out Bridge モジュールによって AXI4-Stream ビデオがネイティブ ビデオに変換されます。HDMI TX コアはこの video_clk を使用してビデオ データを取り込みます。
SB_STATUS_IN_tdata	入力	2	サイド バンド ステータス入力信号 ビット 0: link_rdy ビット 1: video_rdy
SB_STATUS_IN_tvalid	入力	1	サイド バンド ステータス入力 Valid 信号
fid	入力	1	AXI4-Stream バスのフィールド ID。インターレス ビデオでのみ使用。 0 - 偶数フィールド 1 - 奇数フィールド このビットは AXI4-Stream バスの SOF と同時にサンプルされます。この信号を使用しない場合は、入力を Low に設定します。

1. hpd (ホット プラグ検出) 信号は HDMI シンクで駆動され、HDMI ケーブルが接続されるとアサートされます。これにより、HDMI ソースは HDMI シンクの存在がわかります。HDMI シンクは 5V 電源信号に単純に接続される場合もあります。したがって、PCB で分圧回路またはレベル シフターを使用する場合、hpd 信号の極性はアクティブ High のままです。これに対し、hpd 信号にインバーターを追加する場合は HDMI Transmitter Subsystem の GUI で [Hot Plug Detect Active] を [Low] に設定する必要があります。

ネイティブ ビデオ インターフェイスのその他の信号

表 2-11 に、ネイティブ ビデオ インターフェイスを選択した場合のその他の信号を示します。

表 2-11: ネイティブ ビデオ インターフェイスのその他の信号

名称	方向	幅	説明
hpd	入力	1	XGUI オプションで [Hot Plug Detect Active] が [High] (デフォルト) の場合 0 - ホット プラグ検出がディアサートされている 1 - ホット プラグ検出がアサートされている XGUI オプションで [Hot Plug Detect Active] が [Low] の場合 ⁽²⁾ 0 - ホット プラグ検出がアサートされている 1 - ホット プラグ検出がディアサートされている
irq	出力	1	CPU に対する割り込み要求。アクティブ High。
SB_STATUS_IN_tdata	入力	2	サイド バンド ステータス入力信号 ビット 0: link_rdy ビット 1: video_rdy
SB_STATUS_IN_tvalid	入力	1	サイド バンド ステータス入力 Valid 信号
video_rst	出力	1	video_clk ドメインのビデオ リセット信号。アクティブ High。

2. hpd (ホット プラグ検出) 信号は HDMI シンクで駆動され、HDMI ケーブルが接続されるとアサートされます。これにより、HDMI ソースは HDMI シンクの存在がわかります。ほとんどの場合、HDMI シンクは 5V 電源信号に単純に接続されます。したがって、PCB で分圧回路またはレベルシフターを使用する場合、hpd 信号の極性はアクティブ High のままです。これに対し、hpd 信号にインバーターを追加する場合は HDMI Transmitter Subsystem の GUI で [Hot Plug Detect Active] を [Low] に設定する必要があります。

クロックとリセット

表 2-12 に、クロックとリセットの概要を示します。詳細は、「クロッキング」および第 3 章の「リセット」を参照してください。

表 2-12: クロックとリセット

名称	方向	幅	説明
s_axi_cpu_aclk	入力	1	AXI4-Lite CPU 制御インターフェイス クロック。
s_axi_cpu_aresetn	入力	1	s_axi_cpu_aclk に関連するリセット (アクティブ Low)。s_axi_cpu_aresetn 信号により、データバスおよび AXI4-Lite レジスタを含むサブシステム全体がリセットされます。
s_axis_video_aclk	入力	1	AXI4-Stream ビデオ入力クロック。
s_axis_video_aresetn	入力	1	s_axis_video_aclk に関連するリセット (アクティブ Low)。ビデオ入力の AXI4-Stream データバスをリセットします。
s_axis_audio_aclk	入力	1	AXI4-Stream オーディオ入力クロック。オーディオストリーミング クロックは、オーディオ サンプル周波数の 128 倍以上とする必要があります。
s_axis_audio_aresetn	入力	1	s_axis_audio_aclk に関連するリセット (アクティブ Low)。オーディオ入力の AXI4-Stream データバスをリセットします。
link_clk	入力	1	HDMI リンク データ出力クロック。Video PHY Controller のリンク クロック出力に接続します。
video_clk	入力	1	ネイティブ ビデオ インターフェイス用クロック。

注記:

1. 関連するクロックが安定するまで、リセットをアサートする必要があります。

サブシステムを使用するデザイン

この章では、サブシステムを使用してデザインを完成させるためのガイドラインおよびその他の情報を紹介します。

一般的なデザイン ガイドライン

このサブシステムは、ほかのハードウェア コンポーネントに接続して完全な HDMI TX システムを構成します。通常、接続するハードウェア コンポーネントはデバイスごとに異なります。たとえば、Kintex[®]-7 デバイスと UltraScale[™] デバイスでは PLL アーキテクチャが異なります。したがって、システムを理解し、それに応じてサブシステムのパラメーターを調整する必要があります。サブシステム API をソフトウェア アプリケーションに統合する方法は、付録 C 「アプリケーション ソフトウェア 開発」で説明します。

オーディオ データ ストリーム

図 3-1 に、AXI4-Stream のオーディオ サイクルを示します。データは、Valid 信号 (TVLD) と Ready 信号 (TRDY) の両方がアサートされているときにキャプチャされます。HDMI 1.4/2.0 Transmitter Subsystem は、チャンネルが順番に並んでいることを前提に動作します。チャンネル データが順番に並んでいない場合、チャンネル データがほかのチャンネルのサンプル スロットにマップされることがあります。したがって、オーディオ ストリーム ソースからは隣接するチャンネルを順番に (CH0、CH1、など) 送信する必要があります。

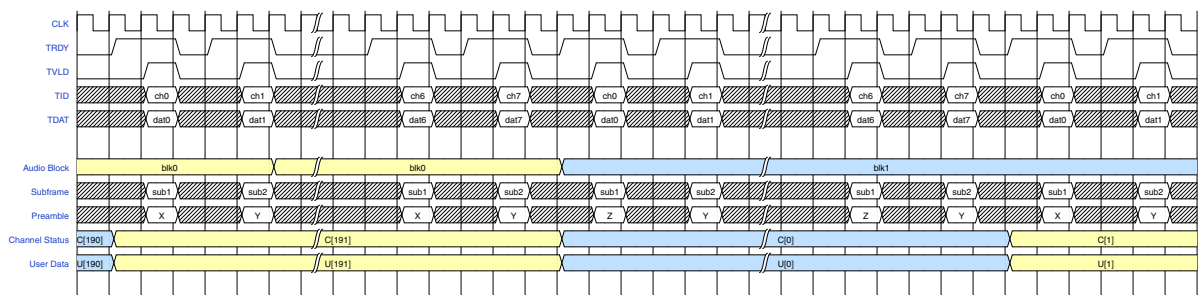


図 3-1: オーディオ サイクル

HDMI 1.4/2.0 Transmitter Subsystem では、オーディオ チャンネル数はソフトウェア ドライバーで設定します。実際のユース ケースに適した数のオーディオ チャンネルを有効にし、オーディオ チャンネル データを対応するチャンネル ID (TID) にマップします。たとえば、8 チャンネル オーディオを送信する場合、HDMI 1.4/2.0 Transmitter Subsystem ドライバーでオーディオ チャンネル数を 8 に設定します。次に、チャンネル数に応じたオーディオ データを準備し、図 3-1 に示すようにハードウェアで HDMI 1.4/2.0 Transmitter Subsystem に送信する必要があります。

ビデオ入カストリーム インターフェイス

AXI4-Stream ビデオ インターフェイスは RGB および YUV444 色空間で 2、4PPC (Pixels Per Clock) および 8、10、12、16BPC (Bits Per Component) をサポートしています。YUV422 色空間では、色深度は常に 12 ビット/ピクセルです。

[Max bits per component] を 16 に設定した場合、4PPC を選択した場合の AXI4-Stream ビデオ プロトコルに完全準拠した 4PPC のデータ フォーマットは図 3-2 のようになります。2PPC の場合は図 3-3 のようになります。

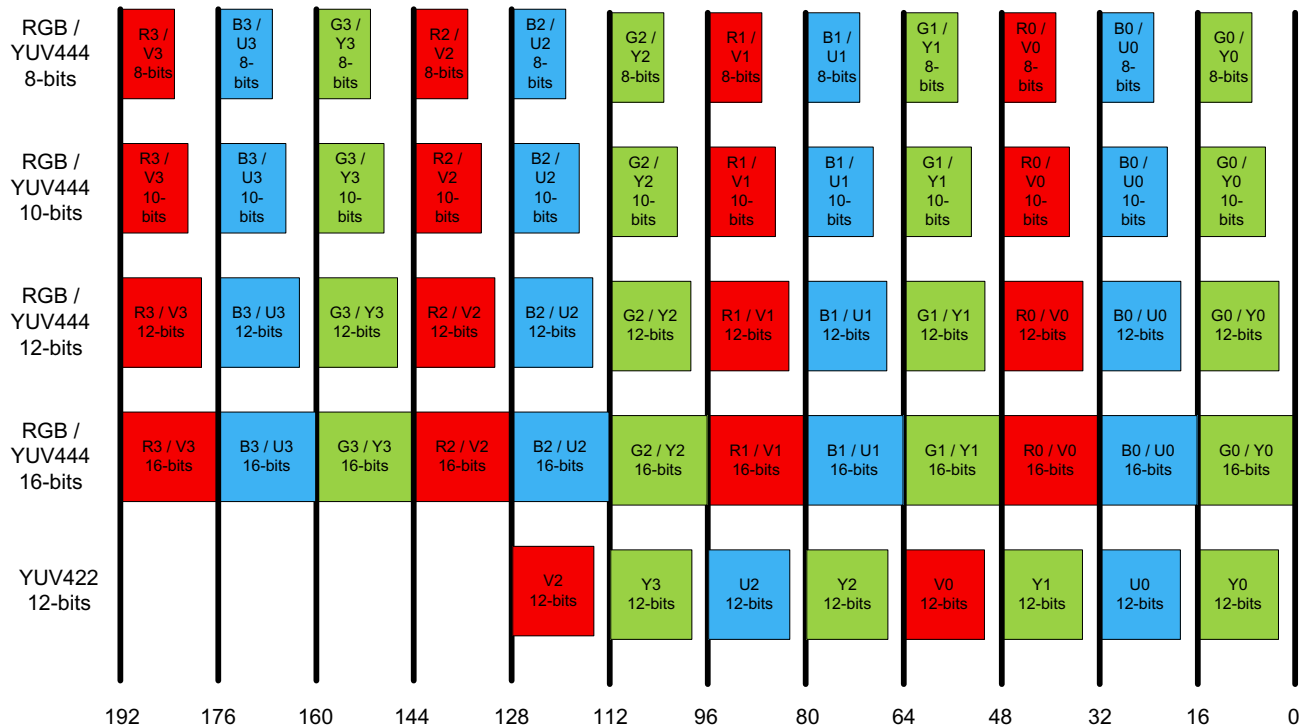


図 3-2: クワッド ピクセルのデータ フォーマット ([Max bits per component] = 16)

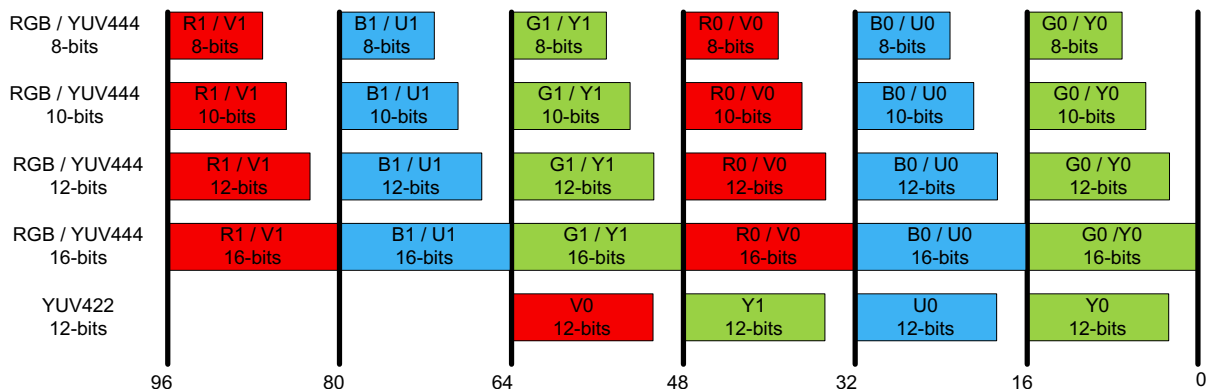


図 3-3: デュアル ピクセルのデータ フォーマット ([Max bits per component] = 16)

パラメーターの [Max bits per component] を 12 に設定した場合、実際の BPC が 12 より大きいビデオ フォーマットは、12 までで切り捨てられます (下位ビットが破棄される)。実際の BPC が Vivado IDE の [Max bits per component] で設定した値より小さい場合、すべてのビットが MSB 詰めで伝送され、下位ビットには 0 がパディングされます。この規則は [Max bits per component] のすべての設定に適用されます。

表 3-1: [Max bits per component] のサポート

[Max bits per component]	実際の BPC	ハードウェアによって伝送されるビット
16	8	[7:0]
	10	[9:0]
	12	[11:0]
	16	[15:0]
12	8	[7:0]
	10	[9:0]
	12	[11:0]
	16	[15:4]
10	8	[7:0]
	10	[9:0]
	12	[11:2]
	16	[15:6]
8	8	[7:0]
	10	[9:2]
	12	[11:4]
	16	[15:8]

たとえば、[Max bits per component] を 12 に設定した場合、AXI4-Stream ビデオ プロトコルに完全準拠した 4PPC のデータ フォーマットは図 3-4 のようになります。2PPC の場合は図 3-5 のようになります。

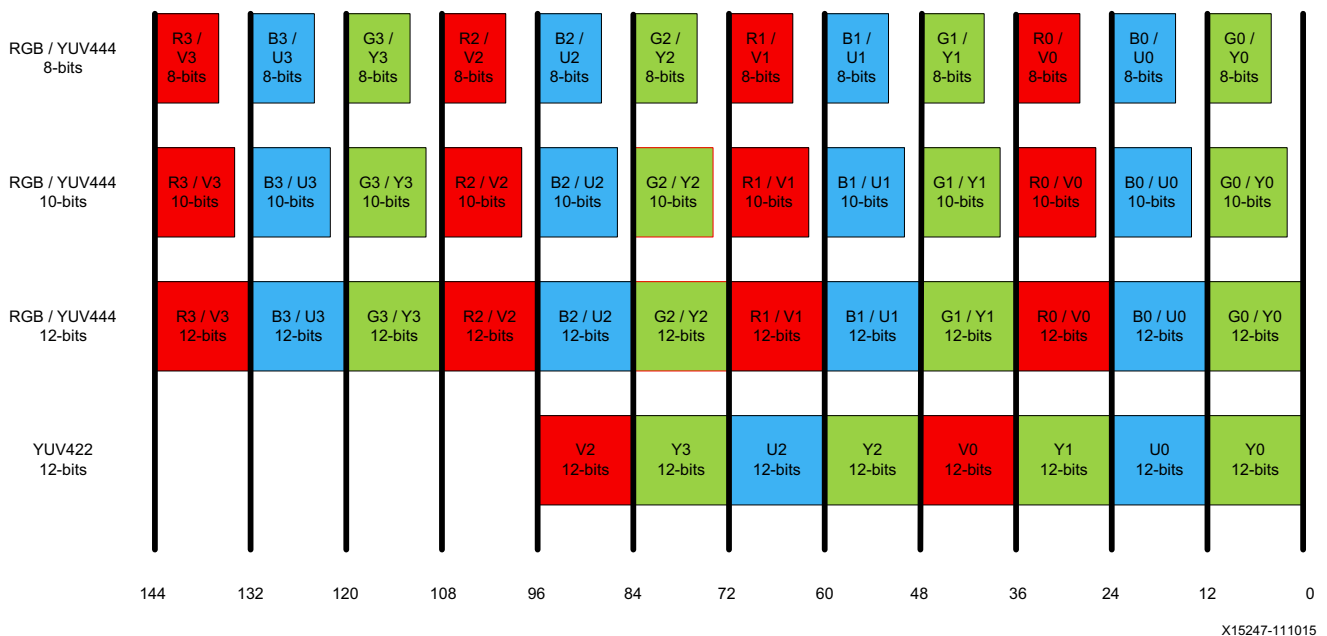
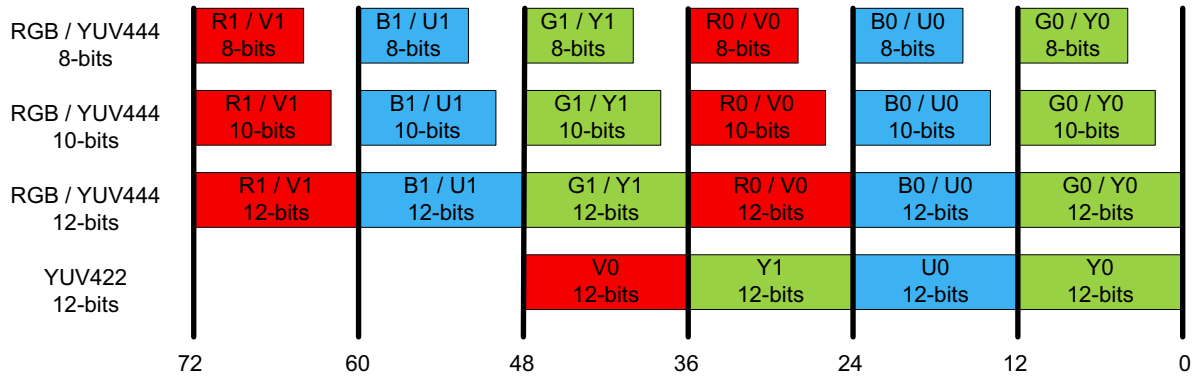


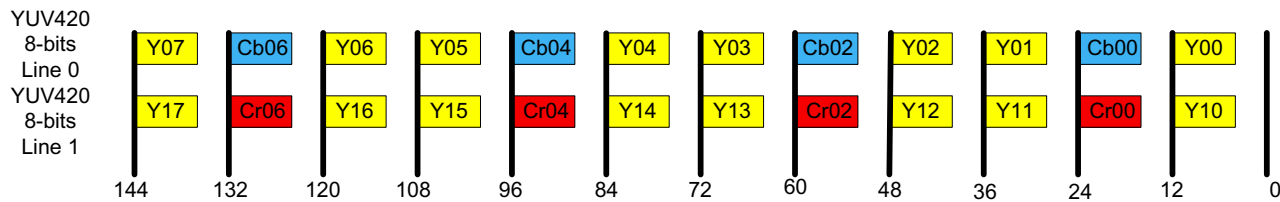
図 3-4: クワッド ピクセルのデータ フォーマット ([Max bits per component] = 12)



X15248-031016

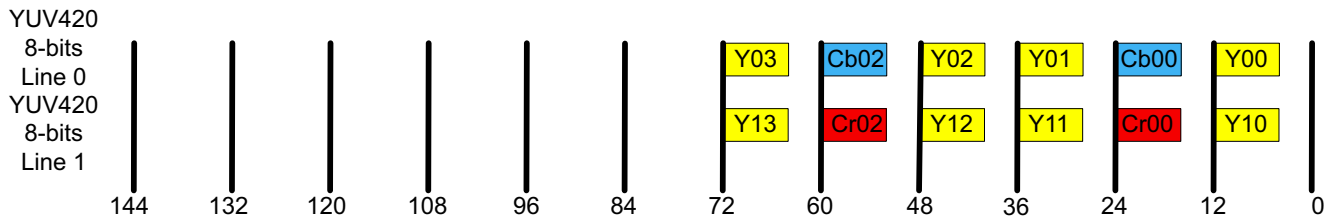
図 3-5: デュアルピクセルのデータフォーマット ([Max bits per component] = 12)

このビデオインターフェイスは YUV420 色空間でクワッドおよびデュアルピクセルも伝送できます。ただし、現在のデータフォーマットは AXI4-Stream ビデオプロトコルに準拠していません。図 3-6 と図 3-7 に、クワッドピクセルとデュアルピクセルのデータフォーマットを示します。



X15250-111015

図 3-6: YUV420 色空間、クワッドピクセルのデータフォーマット



X15251-111015

図 3-7: YUV420 色空間、デュアルピクセルのデータフォーマット

YUV 4:2:0 ディープカラー (10、12、または 16 ビット) の場合のデータ表現も図 3-6 および図 3-7 と同様です。1つのコンポーネントで伝送されるビット数が多い (10、12、および 16) 点のみ異なります。『AXI4-Stream Video IP およびシステムデザインガイド』(UG934) [参照 12] に準拠させるには、HDMI Transmitter Subsystem の GUI で YUV 4:2:0 を有効にします。

図 3-8 は、『AXI4-Stream Video IP およびシステム デザイン ガイド』[参照 12] に記載された YUV 4:2:0 AXI4-Stream ビデオ データ表現 (8 ビット ビデオの場合) です。

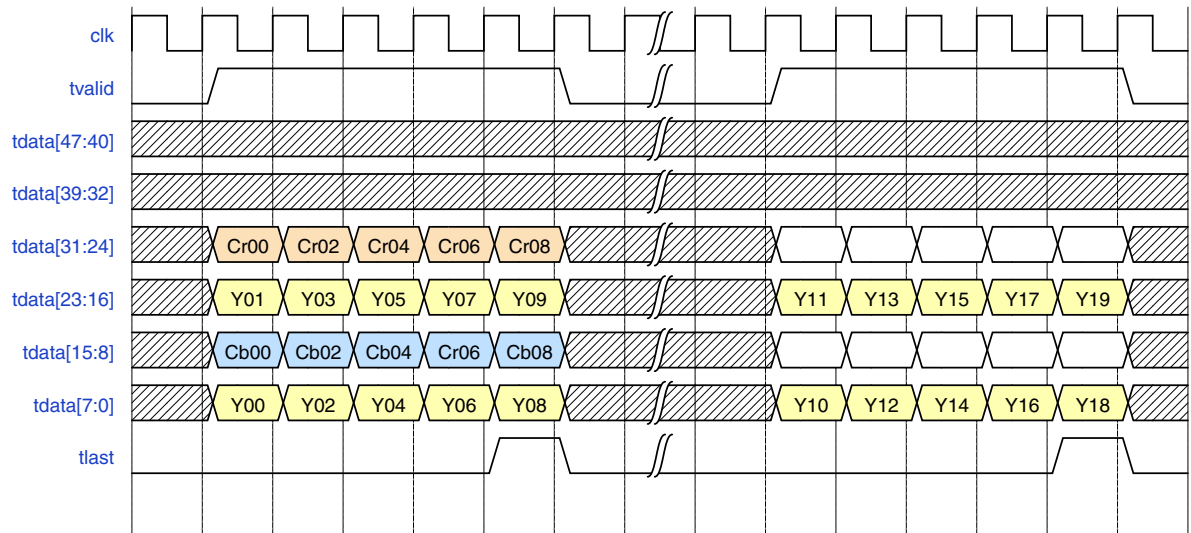


図 3-8: YUV 4:2:0 AXI4-Stream ビデオ データ (2PPC)

ただし、ネイティブ HDMI ビデオ インターフェイスでは、図 3-9 に示すようなビデオ データ表現が必要です。

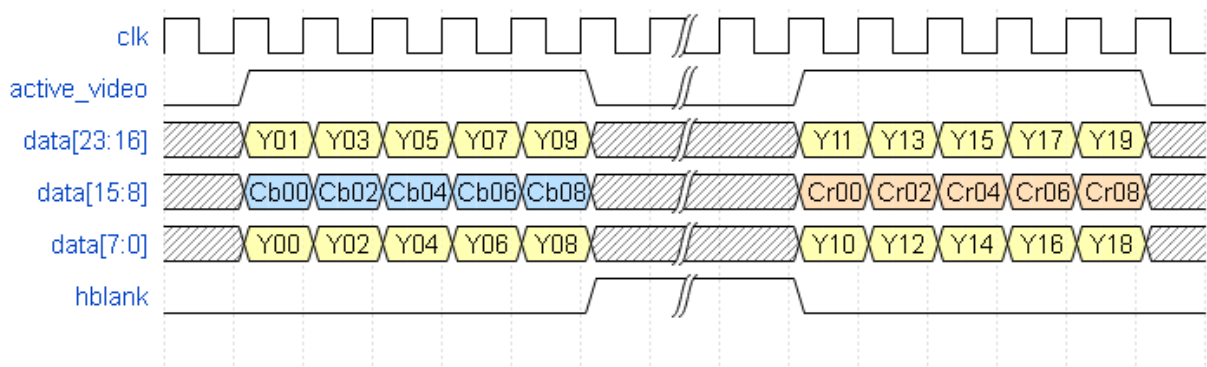


図 3-9: ネイティブ HDMI ビデオ インターフェイス

したがって、HDMI 1.4/2.0 Transmitter Subsystem にはネイティブ HDMI ビデオを AXI4-Stream ビデオに変換するマップ変更機能が追加されています。

このサブシステムは、パラメーターの [Maximum bits per component] と [Nnumber of pixels per clock on Video Interface] を使用してシステムを柔軟に構成できます。ビデオ クロックとリンク クロックがターゲット デバイスのサポート 範囲となるようにこれらのパラメーターを設定します。たとえば、2PPC を選択した場合、AXI4-Stream ビデオは 4PPC のデザインよりも高いクロック レートで動作する必要があります。この場合、システムがタイミング要件を満たすことはより難しくなります。したがって、高解像度ビデオ (4kp60 など) を送信するデザインには 4PPC データ マッピングを推奨します。

720p60 など、一部のビデオ解像度では水平タイミング パラメーター (1650) が 4 の倍数でないことがあります。この場合、2PPC データ マッピングを選択する必要があります。

AXI4-Stream ビデオ インターフェイスおよびビデオ データ フォーマットの詳細は、『AXI4-Stream Video IP およびシステム デザイン ガイド』(UG934) [参照 12] を参照してください。

インターレースビデオ

HDMI 1.4/2.0 Transmitter Subsystem は、AXI4-Stream ビデオ インターフェイスとネイティブ ビデオ インターフェイスの両方をサポートします。

- AXI4-Stream を選択した場合、AXI4-Stream to Video Out コアを使用して HDMI 1.4/2.0 TX Subsystem をサポートします。AXI4-Stream はアクティブ ビデオ データのみを伝搬するため、AXI4-Stream to Video Out コアは、AXI4-Stream スレーブ インターフェイスからの入力をネイティブ ビデオ ストリームに変換し、HDMI TX コアに供給します。
- ネイティブ インターフェイスを選択した場合、ネイティブ ビデオ ストリームを準備し、HDMI 1.4/2.0 Transmitter Subsystem の Video_In ポートに供給する必要があります。このポートは HDMI 1.4/2.0 Transmitter Subsystem 内で HDMI TX コアに直接接続されます。

HDMI 1.4/2.0 Transmitter Subsystem は、プログレッシブ ビデオとインターレース ビデオの両方をサポートするように設計されています。インターレース ビデオの方がプログレッシブ ビデオよりも簡単なので、このセクションではインターレース ビデオの処理方法に焦点を合わせます。

1920x1080@50Hz (I) を例として、詳細なタイミング情報を表 3-2 に示します。

表 3-2: タイミング データ

名称	タイミング フィールドのサブセット	値
HActive		1920
HBlank		720
	HFrontPorch	528
	HSyncWidth	44
	HBackPorch	148
HTotal		2640
VActive		540
F0VBlank		22
	F0PVFrontPorch	2
	F0PVSynWidth	5
	F0PVBackPorch	15
F0PVTotat		562
F1VBlank		23
	F1VFrontPorch	3
	F1VSynWidth	5
	F1VBackPorch	15
F1VTotat		563

インターレースビデオでは、各フレームが2つのフィールドで構成されます。1つのフィールドは奇数ラインを伝搬し、もう1つのフィールドは偶数ラインを伝搬します。両方のフィールドを組み合わせて、完全なフレームが得られます。したがって、次のようになります。

- 1つのフィールドあたりの垂直アクティブライン数 = 垂直アクティブライン数/2
- フレームレート = フィールドレート/2

この例では、次のようになります。

$$V_{Active} = 1080/2 = 540$$

$$\text{フィールドレート} = 50\text{Hz}$$

$$\text{フレームレート} = 50/2 = 25\text{Hz}$$

AXI4-Stream インターフェイスを使用するデザインでは、表 3-2 に示す値を使用したタイミングでビデオの2つのフィールドを生成します。タイミングの詳細は、CEA-861-F [参照 26] を参照してください。AXI4-Stream プロトコルに準拠したアクティブビデオデータのみが必要です。HDMI 1.4/2.0 Transmitter Subsystem 内の AXI4-Stream to Video Out コアは、AXI4-Stream ビデオをネイティブビデオに変換します。fid がフィールドビデオデータと揃えられて駆動されるようにします。詳細は、『AXI4-Stream to Video Out LogiCORE IP 製品ガイド』(PG044) [参照 25] を参照してください。

ネイティブインターフェイスを使用するデザインでは、表 3-2 に示す値を使用したタイミングでネイティブビデオの2つのフィールドを生成します。HSYNC および VSYNC が表 3-2 の値を使用して駆動されるようにします。1つのフレームが奇数本のラインで構成されることがあるため (たとえば 1080i50 では 1125 本)、2つのフィールドでラインの総数が一致しない場合もあります (フィールド 0 は 522 本、フィールド 1 は 523 本など)。

ピクセル反復によるインターレースビデオ

TMDS レートが 25MHz 未満 (たとえば 480i/NTSC では 13.5MHz) の一部のビデオフォーマットは、ピクセル反復方式で送信できます。

HDMI 1.4/2.0 Transmitter Subsystem 内では、次のようになります。

- NTSC/480i60 または PAL/576i50 で 2 のピクセル反復がサポートされる
- IP コアの生成時に GUI パラメーターで有効になる
- ピクセル反復は AXI4-Stream インターフェイスでのみ利用可能

HDMI 1.4/2.0 Transmitter Subsystem 内でピクセル反復が有効にされた後、通常と同じ方法でインターレースビデオを準備すると、HDMI 1.4/2.0 Transmitter Subsystem はデータを送信する前に各ピクセルを 2 回複製します。

ピクセル反復は NTSC/480i60 と PAL/576i50 でのみサポートされるため、Video Common ライブラリから次の適切な XVIDIC が選択されていることを確認します。

- XVIDIC_VM_1440x576_50_I => NTSC/480i60
- XVIDIC_VM_1440x480_60_I => PAL/576i50

次に例を示します。

480i60 ビデオは 720x480 @ 30Hz です。これは 720x240 @ 30Hz ビデオの2つのフィールドから作成されます。

ソフトウェア内で XVIDIC_VM_1440x480_60_I を選択する必要があります。次にハードウェアシステム内で、720x240 @ 30Hz ビデオ (AXI4-Stream ビデオ) の2つのフィールドを準備し、HDMI 1.4/2.0 Transmitter Subsystem に送信します。すると HDMI 1.4/2.0 Transmitter Subsystem は各ピクセルを 2 回繰り返します。ビデオが HDMI 1.4/2.0 Transmitter Subsystem によって送信される際には、1440x240 @ 30Hz ビデオの2つのフィールドとして送信されます。

クロッキング

S_AXI_CPU_IN、VIDEO_OUT、およびAUDIO_OUTはそれぞれ異なるクロックレートで動作可能です。HDMIリンクインターフェイスとネイティブビデオインターフェイスもそれぞれ異なるクロックレートで動作します。したがって、これらに対して5つのクロックインターフェイス(順にs_axi_cpu_aclk、s_axis_video_aclk、s_axis_audio_aclk、link_clk、およびvideo_clk)が用意されています。

オーディオストリーミングクロックは、オーディオサンプル周波数の128倍以上とする必要があります。オーディオクロックの再生成はHDMI TX Subsystemの一部ではないため、ユーザーがアプリケーションにオーディオクロックを供給する必要があります。このクロックは、内部PLLまたは外部クロックソースを使用して供給できます。



重要: AXI4-Lite CPU クロックは 10MHz で動作させる必要があります。

図 3-10 と表 3-3 に HDMI のクロック構造を示します。

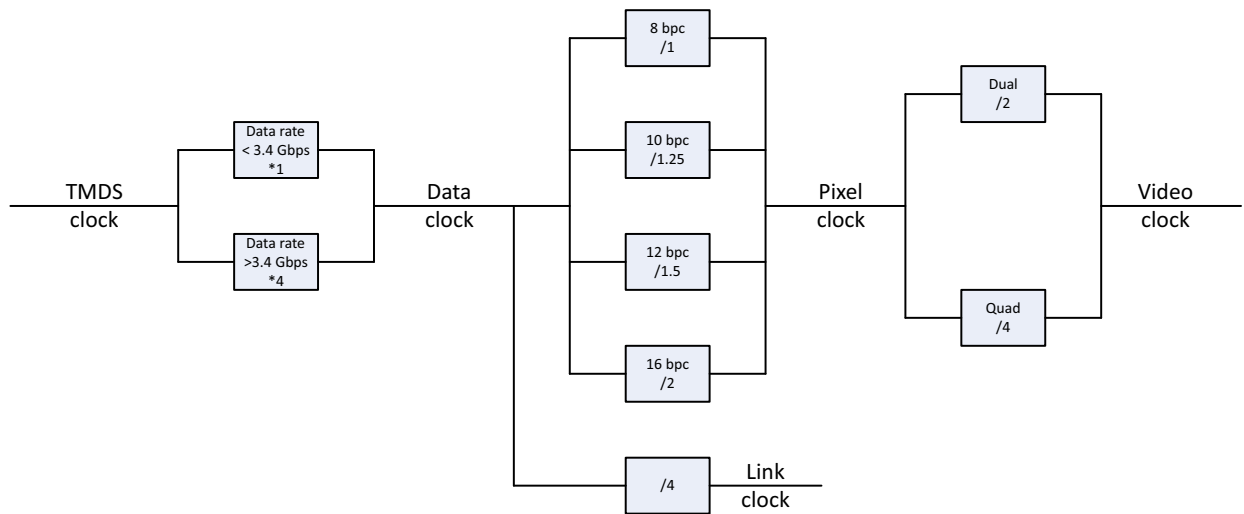


図 3-10: HDMI クロック構造

表 3-3: クロッキング

HDMI クロッキング			
クロック	機能	周波数/レート	例 ⁽¹⁾
TMDS クロック	HDMI インターフェイスに対するソース同期クロック (HDMI ケーブルで伝送される実際のクロック)。	= データ レートの 1/10 (データ レートが 3.4Gb/s 未満の場合) = データ レートの 1/40 (データ レートが 3.4Gb/s 超の場合)	データ レート = 2.97Gb/s の場合 TMDS クロック = $2.97/10 = 297\text{MHz}$ データ レート = 5.94Gb/s の場合 TMDS クロック = $5.94/40 = 148.5\text{MHz}$
データ クロック	実際のデータ レート クロックです。このクロックはシステムでは使用しません。クロックの関係を示すためだけに記載しています。	= TMDS クロック (データ レートが 3.4Gb/s 未満の場合) = TMDS クロック * 4 (データ レートが 3.4Gb/s 超の場合)	データ レート = 2.97Gb/s の場合 データ クロック = TMDS クロック * 1 = 297MHz データ レート = 5.94Gb/s の場合 データ クロック = TMDS クロック * 4 = 594MHz TMDS クロック = 148.5MHz
リンク クロック	HDMI 物理層モジュールとサブシステム間のデータ インターフェイスに使用するクロック	= データ クロックの 1/4	TMDS クロック = 297MHz データ クロック = 297MHz リンク クロック = $297\text{MHz}/4 = 74.25\text{MHz}$ データ クロック = 594MHz リンク クロック = $594\text{MHz}/4 = 148.5\text{MHz}$
ピクセル クロック	内部ピクセル クロックです。このクロックはシステムでは使用しません。クロックの関係を示すためだけに記載しています。	8bpc の場合: ピクセル クロック = データ クロック 10bpc の場合: ピクセル クロック = データ クロック/1.25 12bpc の場合: ピクセル クロック = データ クロック/1.5 16bpc の場合: ピクセル クロック = データ クロック/2	

表 3-3: クロッキング (続き)

HDMI クロッキング			
クロック	機能	周波数/レート	例 ⁽¹⁾
ビデオ クロック	ビデオ インターフェイスに 使用するクロック	デュアルピクセルの場合: ビデオクロック=ピクセル クロック/2 クワッドピクセルの場合: ビデオクロック=ピクセル クロック/4	2ピクセル幅インターフェイスの場合、 297MHz/2 = 148.5MHz 4ピクセル幅インターフェイスの場合、 297MHz/4 = 74.25MHz ターゲットデバイスにおけるPLLの正 しい選択方法の詳細は、『Video PHY Controller LogiCORE IP 製品ガイド』 (PG230) [参照 22] を参照してください。

注記:

- 「例」は参考にすぎず、すべての可能な解像度を記載しているわけではありません。各GTには固有のハードウェア要件と制限があります。したがって、GTの異なるデバイスでHDMI 1.4/2.0 Transmitter Subsystemを使用する場合は、クロック周波数を計算してターゲットデバイスでその周波数をサポートできるようにする必要があります。HDMI 1.4/2.0 Transmitter Subsystemとザイリンクス Video PHY Controller IP コアを組み合わせる使用する方法の詳細は、『Video PHY Controller LogiCORE IP 製品ガイド』(PG230) [参照 22] を参照してください。

例として1080p60、12BPC、および2PPCを使用し、すべてのクロックの算出方法を示します。

ビデオ解像度	水平 (合計)	水平 (アクティブ)	垂直 (合計)	垂直 (アクティブ)	フレーム レート (Hz)
1080p60	2200	1920	1125	1080	60

ピクセルクロックは、毎秒送信する必要があるピクセルの総数を示します。したがって、次のようになります。

$$\begin{aligned} \text{ピクセルクロック} &= \text{水平 (合計)} \times \text{垂直 (合計)} \times \text{フレームレート} \\ &= 2200 \times 1125 \times 60 \\ &= 148,500,000 \\ &= 148.5\text{MHz} \end{aligned}$$

$$\text{リンククロック} = (\text{データクロック})/4 = 222.75/4 = 55.6875\text{MHz}$$

$$\text{ビデオクロック} = (\text{ピクセルクロック})/\text{PPC} = 148.5/2 = 74.25\text{MHz}$$

$$\text{データクロック} = \text{ピクセルクロック} \times \text{BPC}/8 = 148.5 \times 12/8 = 222.75\text{MHz}$$

この例で結合法則を使用すると、

$$\text{データクロック} = 222.75\text{MHz} < 340\text{MHz}$$

したがって

$$\text{TMDSクロック} = \text{データクロック} = 222.75\text{MHz}$$

リセット

各 AXI インターフェイスにはそれぞれ専用のリセット信号があります。S_AXI_CPU_IN、VIDEO_OUT (AXI4-Stream ビデオ インターフェイス)、AUDIO_OUT に対して順に `s_axi_cpu_aresetn`、`s_axis_video_aresetn`、`s_axis_audio_aresetn` のリセット信号があります。これら3つのリセット信号はいずれもアクティブ Low です。リセット信号はサブシステム内の複数のサブブロックで使用されるため、すべてのクロックが安定するまでシステムをリセット状態に保持する必要があります。クロック生成ブロックからの `locked` 信号をリセット信号として使用できます。

注記: ネイティブ ビデオ インターフェイスを選択した場合、VIDEO_IN インターフェイスに対する専用のハードウェアリセットはありません。ただし、HDMI TX Subsystem から出力される `video_rst` 信号を使用してネイティブ ビデオ ソース生成モジュールをリセットできます。

デザイン フローの手順

この章では、サブシステムのカスタマイズと生成、制約、およびシミュレーション/合成/インプリメンテーションの手順について説明します。一般的な Vivado® デザイン フローおよび IP インテグレーターの詳細は、次の Vivado Design Suite ユーザー ガイドを参照してください。

- 『Vivado Design Suite ユーザー ガイド: IP インテグレーターを使用した IP サブシステムの設計』(UG994) [参照 13]
- 『Vivado Design Suite ユーザー ガイド: IP を使用した設計』(UG896) [参照 14]
- 『Vivado Design Suite ユーザー ガイド: 入門』(UG910) [参照 15]
- 『Vivado Design Suite ユーザー ガイド: ロジック シミュレーション』(UG900) [参照 16]

サブシステムのカスタマイズおよび生成

ここでは、ザイリンクス ツールを使用し、Vivado Design Suite でサブシステムをカスタマイズおよび生成する方法について説明します。

Vivado IP インテグレーターで HDMI 1.4/2.0 Transmitter Subsystem をブロック デザインに追加し、IP カタログを使用してカスタマイズできます。Vivado IP インテグレーターでサブシステムをカスタマイズおよび生成する方法は、『Vivado Design Suite ユーザー ガイド: IP インテグレーターを使用した IP サブシステムの設計』(UG994) [参照 13] を参照してください。IP インテグレーターは、デザインの検証または生成時に一部のコンフィギュレーション値を自動的に計算する場合があります。値が変わるかどうかを確認するには、この章のパラメーターの説明を参照してください。パラメーター値を確認するには、Tcl コンソールから `validate_bd_design` コマンドを実行してください。

サブシステムはユーザー デザインに合わせてカスタマイズできます。それには、IP サブシステムに関連する各種パラメーターの値を次の手順に従って指定します。

1. Flow Navigator で [Create Block Diagram] をクリックするか、[IP Integrator] の下にある [Open Block Design] をクリックします。
2. 図を右クリックし、[Add IP] をクリックします。

検索可能な IP カタログが開きます。IP インテグレーターのブロック デザイン キャンパスの左側にある [Add IP] ボタンをクリックしても IP を追加できます。

3. IP 名をクリックして Enter キーを押すか、IP 名をダブルクリックします。
4. 選択した IP ブロックをダブルクリックするか、右クリックして [Customize Block] コマンドをクリックします。

詳細は、『Vivado Design Suite ユーザー ガイド: IP を使用した設計』(UG896) [参照 14] および『Vivado Design Suite ユーザー ガイド: 入門』(UG910) [参照 15] を参照してください。

注記: この章の図には Vivado 統合設計環境 (IDE) のスクリーンショットが使用されていますが、現在のバージョンとはレイアウトが異なる場合があります。

[Toplevel] タブ

図 4-1 に [Toplevel] タブを示します。

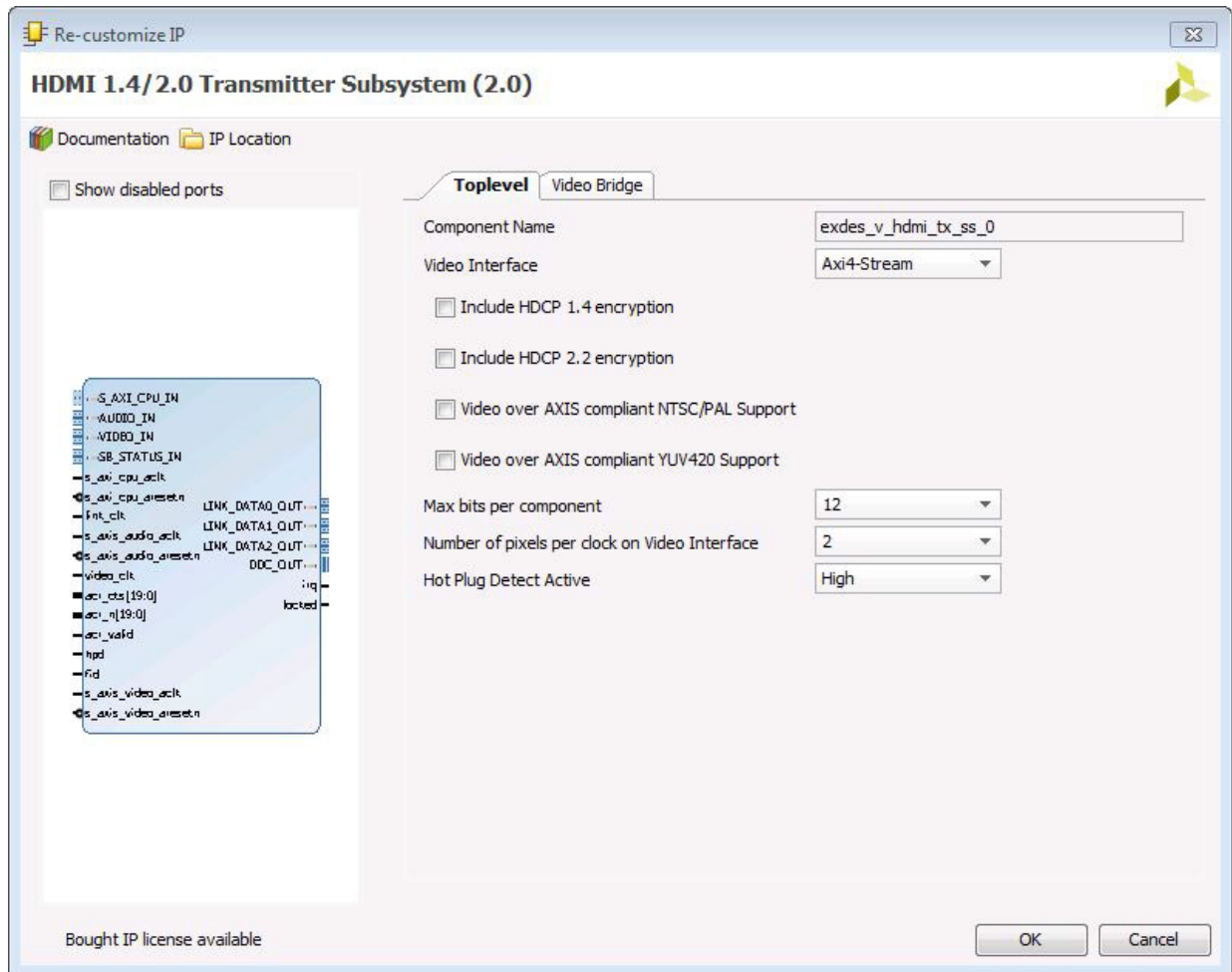


図 4-1: [Toplevel] タブ

[Toplevel] タブには次のパラメーターがあります。

[Component Name]: コンポーネント名は IP インテグレーターによって自動的に設定されます。

[Video Interface]: HDMI TX Subsystem のビデオ インターフェイスを選択します。[AXIS-Stream] または [Native Video] を選択できます。

[Include HDCP 1.4 encryption]: オンにすると HDCP 1.4 暗号化が有効になります。

[Include HDCP 2.2 encryption]: オンにすると HDCP 2.2 暗号化が有効になります。

[Max bits per component]: 1 つのコンポーネントあたりの最大ビット数 (BPC) を選択します。8、10、12、または 16 ビットを選択できます。このパラメーターは「許容可能な」最大 BPC を設定するものであり、実際の BPC はソフトウェア API で別の値に設定できます。ただし、実際の BPC として設定可能な値は [Max bits per component] の制約を受けます。たとえば、[Max bits per component] を 16 に設定した場合は、ソフトウェア API で実際の BPC は 8、10、12、16 のいずれの値にも設定できます。一方 [Max bits per component] を 8 に設定した場合、ソフトウェア API で設定できる実際の BPC は 8 のみです。

[Number of pixels per clock on Video Interface]: 1つのクロックあたりのピクセル数 (PPC) を選択します。2 または 4 ピクセルを選択できます。



重要: 1 クロックあたりのピクセル数 (PPC) は、IP 生成時にのみ選択することができ、デザインで変更することなく使用する必要があります。水平総解像度が 4 で割り切れない一部のビデオフォーマット (たとえば、720p60 の水平総ピクセルは 1650 であり、4 で割り切れない) はサポートされません。デザインでこのタイプのビデオフォーマットをサポートするには、Vivado ツールで [Number of pixels per clock on Video Interface] (PPC) を 2 に設定します。

[Video over AXIS compliant NTSC/PAL Support]: オンにすると、HDMI TX Subsystem は Video over AXIS 準拠の NTSC/PAL をサポートします。

- 現在のハードウェアは 2 のピクセル反復をサポートしています。
- 現在のソフトウェアは 480i60 および 576i50 の解像度をサポートしています。

[Video over AXIS compliant YUV420 Support]: オンにすると、HDMI TX Subsystem は Video over AXIS 準拠の YUV420 をサポートします。

[Hot Plug Detect Active]: HPD アクティブの極性を選択します。[High] または [Low] を選択できます。

[Video Bridge] タブ

図 4-2 に [Video Bridge] タブを示します。

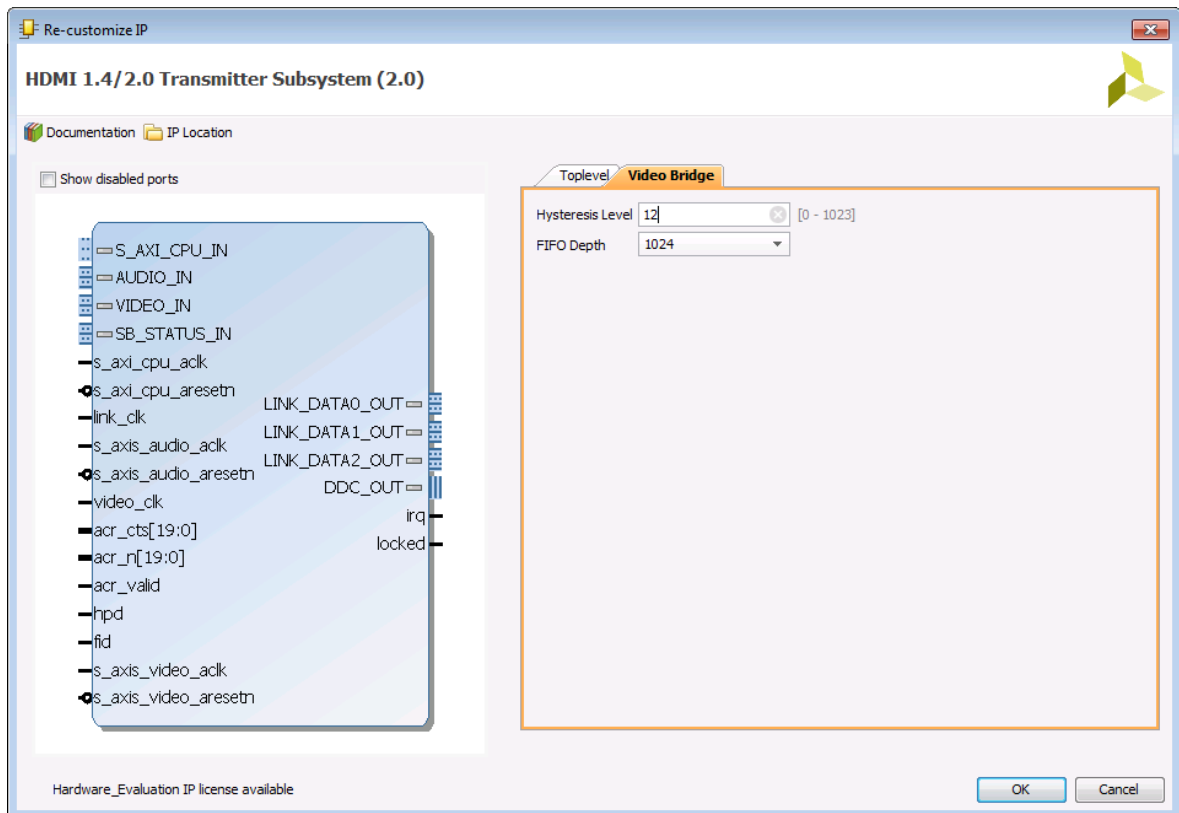


図 4-2: [Video Bridge] タブ

[Video Bridge] タブには次のパラメーターがあります。

[Hysteresis Level]: 0 ~ 1023 の範囲にある値を指定します。これにより、フレームバッファの「クッション」レベル (FIFO 動作が開始する最小フィルレベル) を定義します。通常は 12 ~ 20 の値を指定します。この値は、FIFO 深さから少なくとも 16 を引いた値、およびアクティブビデオライン数から少なくとも 16 を引いた値とする必要があります。

[FIFO Depth]: 入力 FIFO の段数を設定します。32、1024、2048、4096、または 8192 を指定できます。

ネイティブビデオ インターフェイスを選択した場合のオプション

図 4-3 に、ネイティブビデオ インターフェイスを選択した場合のオプション画面を示します。

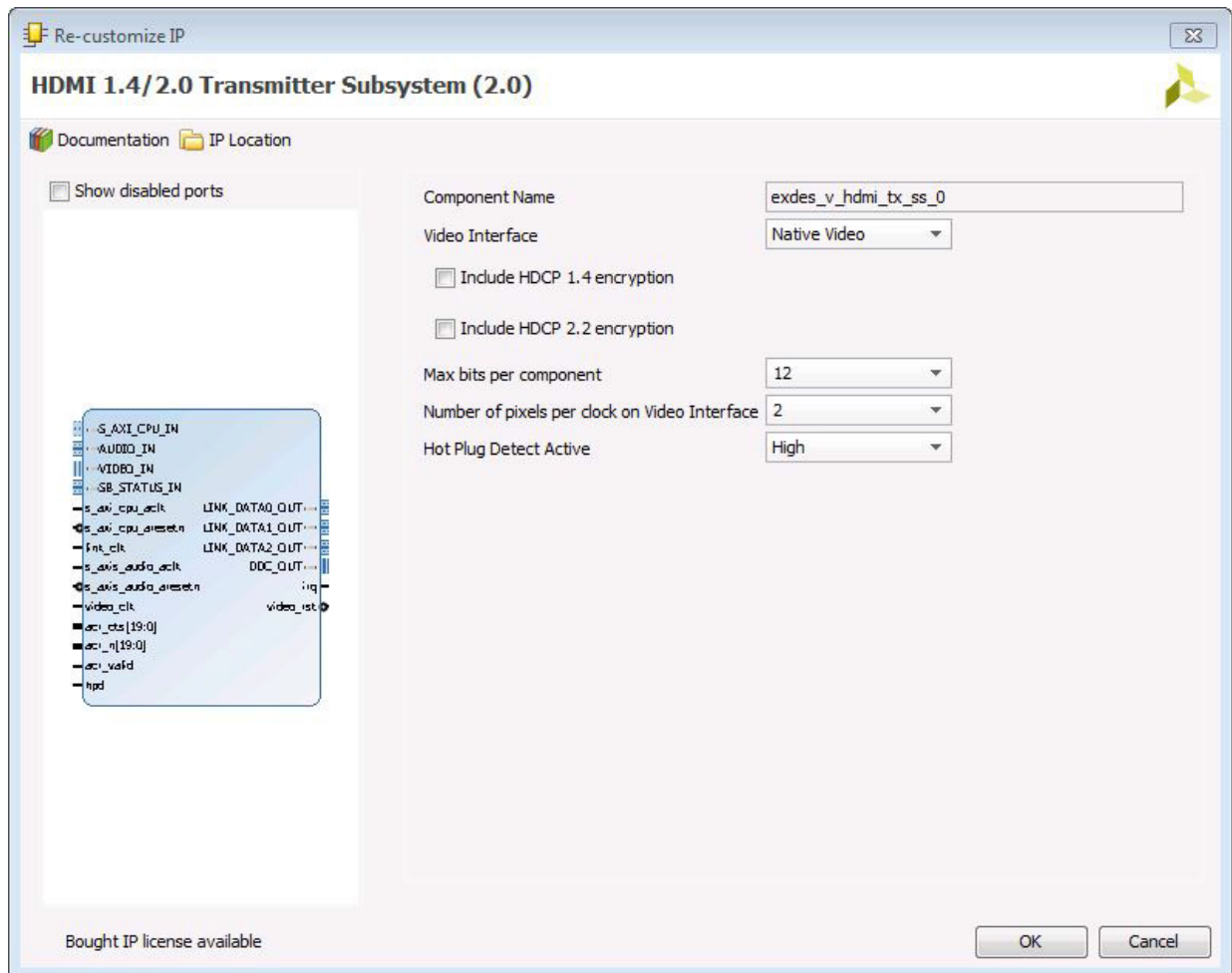


図 4-3: ネイティブビデオ インターフェイスを選択した場合のオプション

[Include HDCP 1.4 encryption]: オンにすると HDCP 1.4 暗号化が有効になります。

[Include HDCP 2.2 encryption]: オンにすると HDCP 2.2 暗号化が有効になります。

注記: HDCP 1.4 および 2.2 暗号化のオプションを設定できるのは HDCP ライセンスを取得している場合のみです。

図 4-4 に、有効な HDCP ライセンスがない場合の [Toplevel] タブを示します。

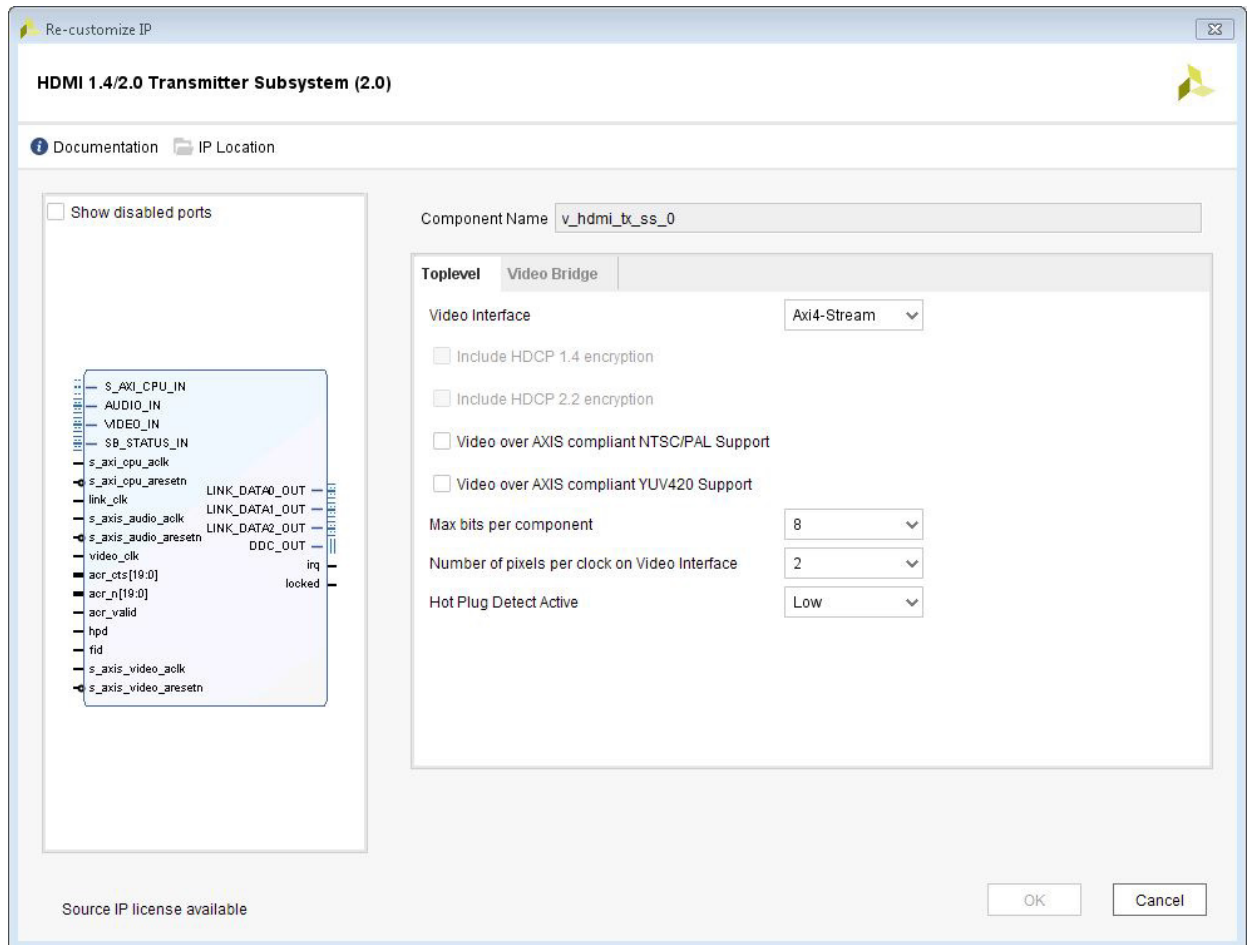


図 4-4: [Toplevel] タブ (HDCP ライセンスなし)

ユーザー パラメーター

表 4-1 に、Vivado IDE のフィールドとユーザー パラメーターの対応関係を示します。ユーザー パラメーターは Tcl コンソールで表示できます。

表 4-1: Vivado IDE のパラメーターとユーザー パラメーターの対応表

Vivado IDE のパラメーター / 値	ユーザー パラメーター / 値	デフォルト値
[Toplevel] タブ		
[Video Interface]	C_VID_INTERFACE	AXI4-Stream
AXI4-Stream	0	
Native Video	1	
[Include HDCP 1.4 encryption]	C_INCLUDE_HDCP_1_4	Exclude
Exclude (オフ)	FALSE	
Include (オン)	TRUE	

表 4-1: Vivado IDE のパラメーターとユーザー パラメーターの対応表 (続き)

Vivado IDE のパラメーター / 値	ユーザー パラメーター / 値	デフォルト値
[Include HDCP 2.2 encryption]	C_INCLUDE_HDCP_2_2	Exclude
Exclude (オフ)	FALSE	
Include (オン)	TRUE	
[Video over AXIS compliant NTSC/PAL Support]	C_INCLUDE_LOW_RESO_VID	Exclude
Exclude (オフ)	FALSE	
Include (オン)	TRUE	
[Video over AXIS compliant YUV420 Support]	C_INCLUDE_YUV420_SUP	Exclude
Exclude (オフ)	FALSE	
Include (オン)	TRUE	
[Max bits per component]	C_MAX_BITS_PER_COMPONENT	8
8	8	
10	10	
12	12	
16	16	
[Number of pixels per clock on Video Interface]	C_INPUT_PIXELS_PER_CLOCK	2
2	2	
4	4	
[Hot Plug Detect Active]	C_HPD_INVERT	High
High	High	
Low	Low	
[Video Bridge] タブ		
[Hysteresis Level]	C_HYSTERESIS_LEVEL	12
[FIFO Depth]	C_ADDR_WIDTH	1024
32	32	
1024	1024	
2048	2048	
4096	4096	
8192	8192	

出力の生成

詳細は、『Vivado Design Suite ユーザー ガイド: IP を使用した設計』(UG896) [参照 14] を参照してください。

サブシステムへの制約

ここでは、Vivado Design Suite でサブシステムに制約を指定する方法について説明します。

必須の制約

s_axi_cpu_aclk、s_axis_video_aclk、s_axis_audio_aclk、link_clk、およびvideo_clkに対するクロック周波数の制約があります。次に例を示します。

```
create_clock -name s_axi_cpu_aclk -period 10.0 [get_ports s_axi_cpu_aclk]
create_clock -name s_axis_audio_aclk -period 10.0 [get_ports s_axis_audio_aclk]
create_clock -name link_clk -period 13.468 [get_ports link_clk]
create_clock -name video_clk -period 6.734 [get_ports video_clk]
create_clock -name s_axis_video_aclk -period 5.0 [get_ports s_axis_video_aclk]
```

このサブシステムを Vivado[®] Design Suite フローで Video PHY Controller モジュールと組み合わせて使用する場合、link_clk と video_clk は Video PHY Controller から生成されます。したがって、クロック制約はこれらの生成済みクロックに対してではなく、Video PHY Controller の制約に対して設定されます。詳細は、『Video PHY Controller LogiCORE™ IP 製品ガイド』(PG230) [参照 22] の「クロッキング」を参照してください。

s_axi_cpu_aclk、s_axis_video_aclk、およびs_axis_audio_aclkの制約は、クロックウィザードを使用するなどしてシステムレベルで生成されます。

デバイス、パッケージ、スピード グレードの選択

デバイスの制約/依存性の詳細は、『Video PHY Controller LogiCORE IP 製品ガイド』(PG230) [参照 22] を参照してください。

表 4-2 に、HDMI 1.4/2.0 Transmitter Subsystem で使用するデバイスおよびスピード グレードの選択を示します。

表 4-2: デバイスおよびスピード グレードの選択

デバイス ファミリ	PPC	2				4			
	BPC	8	10	12	16	8	10	12	16
	スピード グレード								
Artix-7	-1	HDMI 1.4 ⁽¹⁾				HDMI 1.4 ⁽¹⁾			
	-2	HDMI 1.4 ⁽¹⁾				HDMI 1.4 ⁽¹⁾			
Kintex-7	-1	HDMI 1.4 ⁽²⁾				HDMI 1.4 ⁽¹⁾			
	-2	HDMI 2.0 ⁽¹⁾				HDMI 2.0 ⁽²⁾			
Kintex UltraScale	-1	HDMI 2.0 ⁽²⁾				HDMI 2.0 ⁽²⁾			
	-2								
Virtex-7	-1	HDMI 1.4 ⁽²⁾				HDMI 2.0 ⁽²⁾	HDMI 1.4 ⁽¹⁾		
	-2	HDMI 2.0 ⁽¹⁾				HDMI 2.0 ⁽²⁾			
Virtex UltraScale	-1	HDMI 2.0 ⁽²⁾				HDMI 2.0 ⁽²⁾			
	-2								

注記:

1. HDMI 1.4 のすべての解像度をサポートできます。
2. 4096 x 2160 @ 60fps までの HDMI 2.0 のすべての解像度をサポートします。

クロック周波数

AXI4-Lite CPU クロックは 10MHz で動作させる必要があります。詳細は、第3章の「クロッキング」を参照してください。

クロック管理

このセクションは、この IP サブシステムには適用されません。

クロック配置

このセクションは、この IP サブシステムには適用されません。

バンク設定

このセクションは、この IP サブシステムには適用されません。

トランシーバーの配置

このセクションは、この IP サブシステムには適用されません。

I/O 規格と配置

このセクションは、この IP サブシステムには適用されません。

シミュレーション

このサブシステムのシミュレーションはサポートされません。

合成およびインプリメンテーション

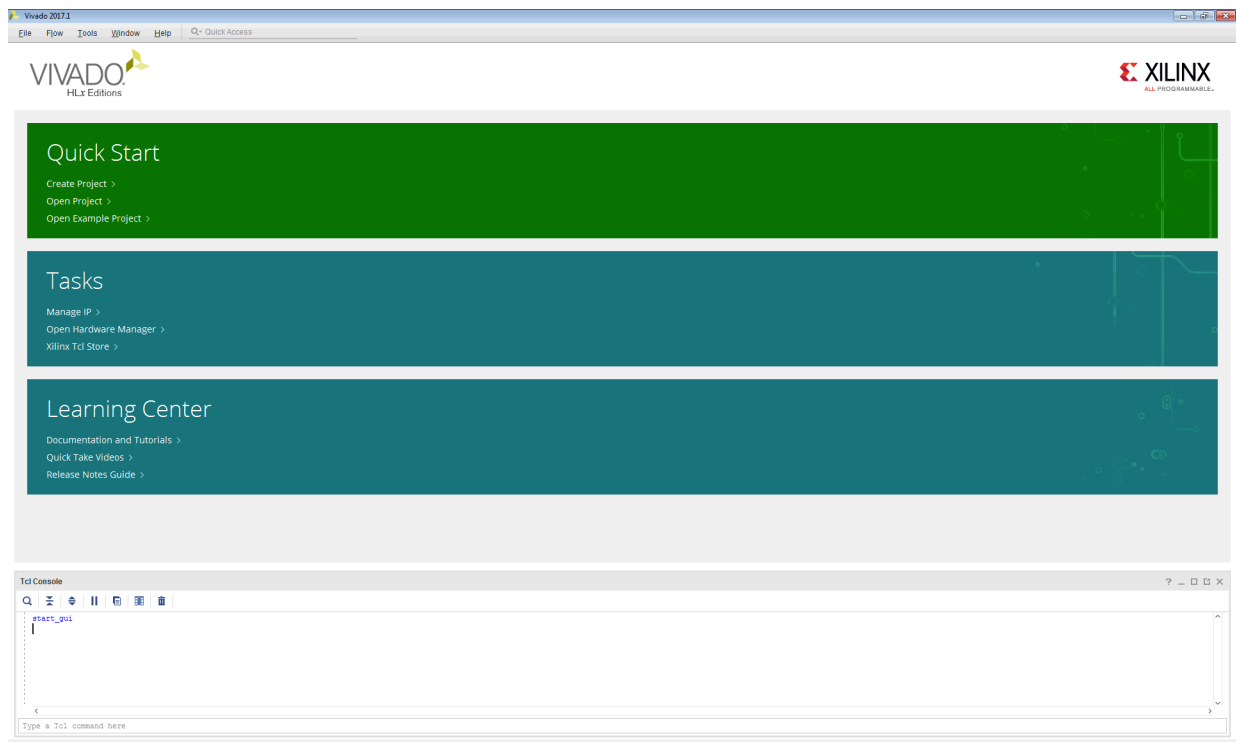
合成およびインプリメンテーションの詳細は、『Vivado Design Suite ユーザー ガイド: IP を使用した設計』(UG896) [参照 14] を参照してください。

サンプル デザイン

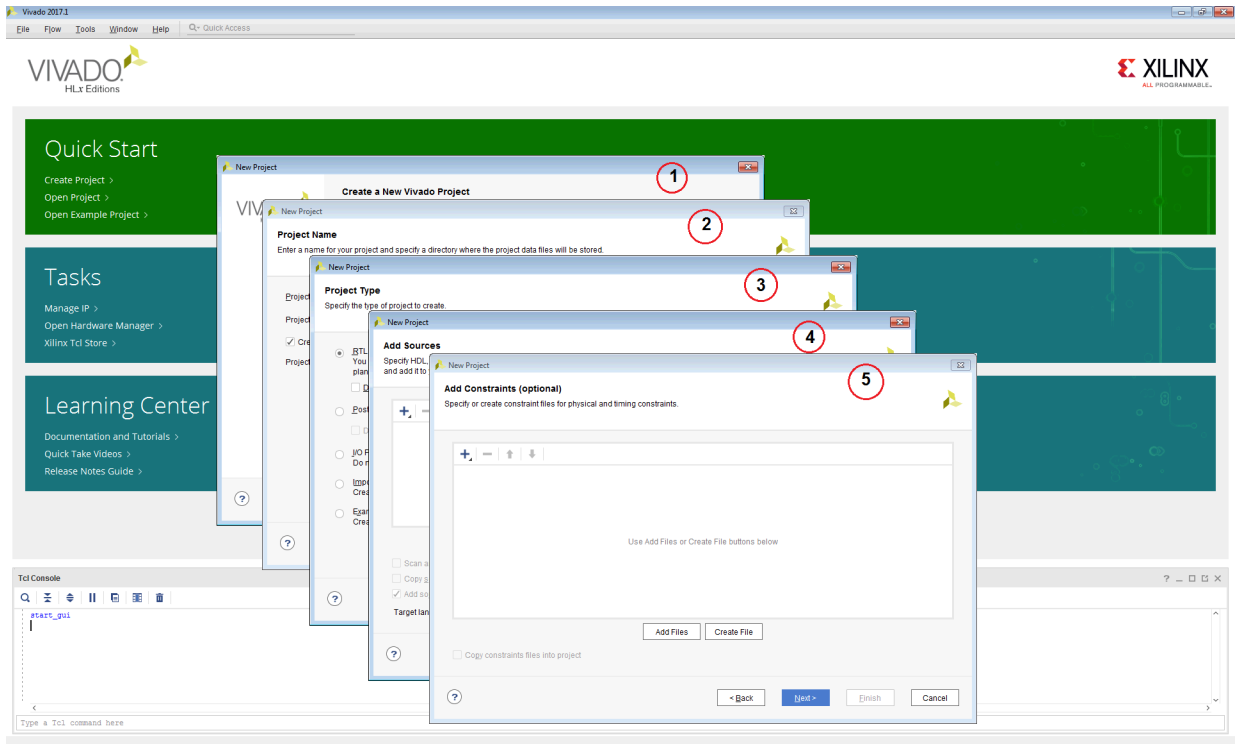
この章では、Vivado® フローを使用して HDMI 1.4/2.0 Transmitter Subsystem から HDMI サンプル デザインを生成する手順について説明します。

サンプル デザインを実行する

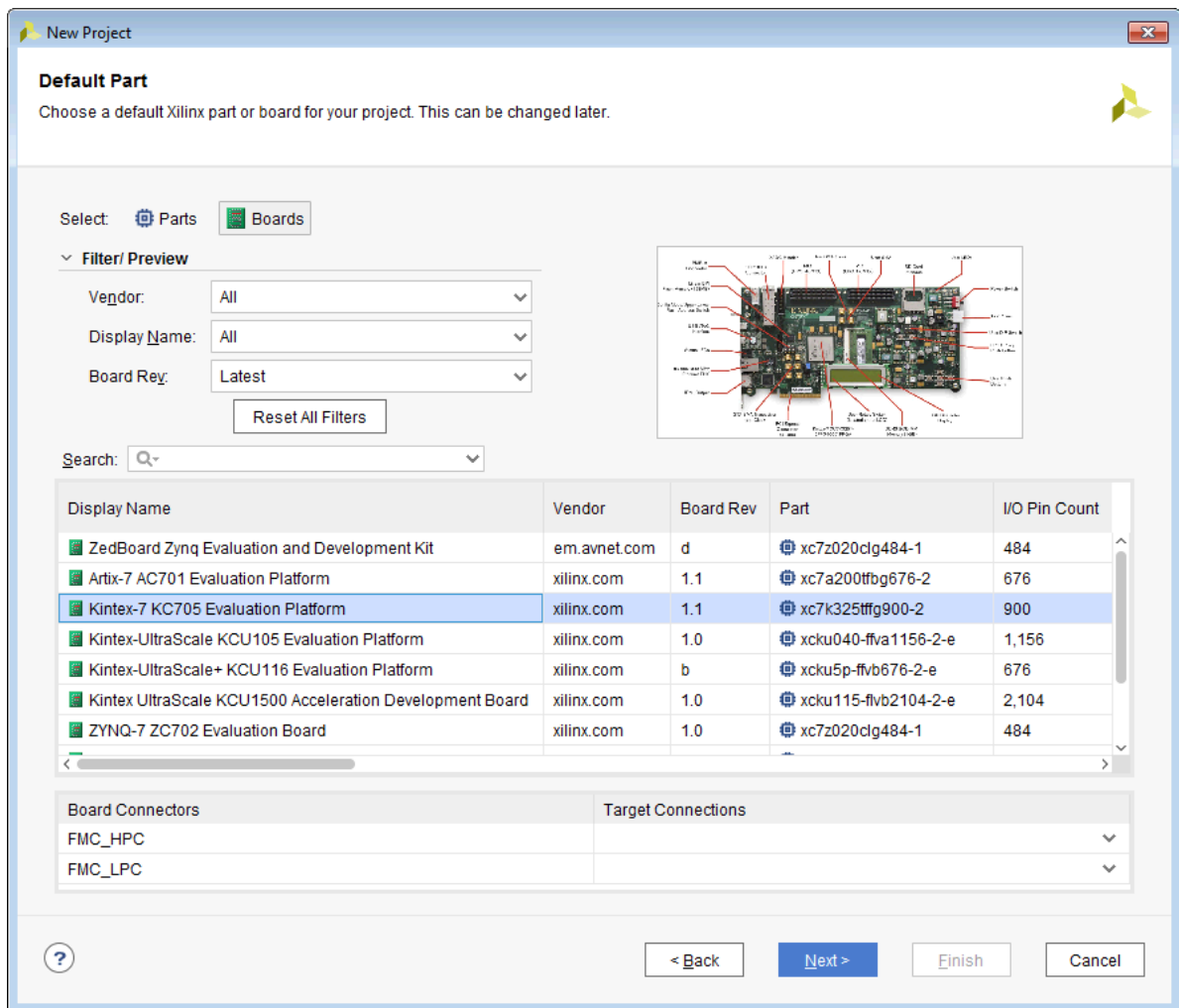
1. Vivado Design Suite を開き、新規プロジェクトを作成します。



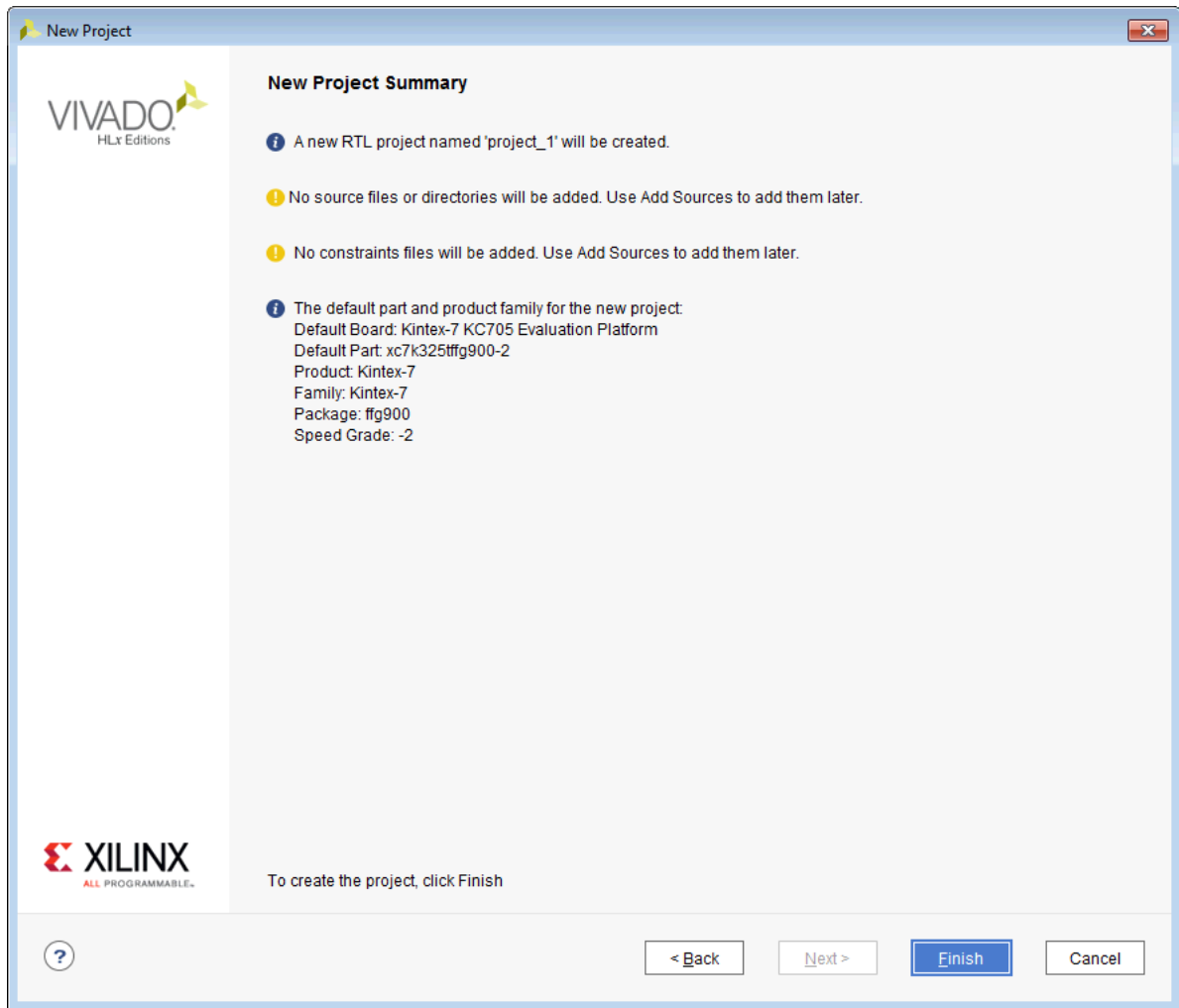
2. ダイアログ ボックスで [Next] を 5 回クリックします。



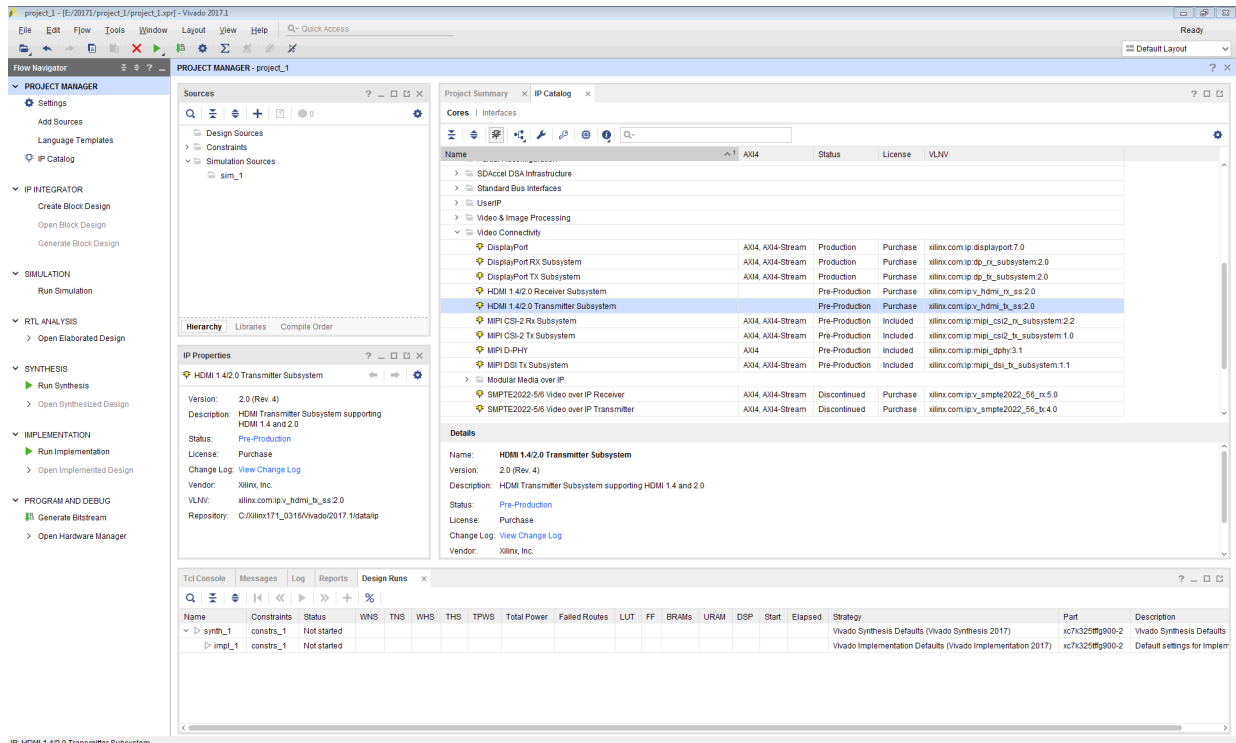
3. ボードを選択します (KC705、ZC706、KCU105 がサポートされる)。



4. [Finish] をクリックします。

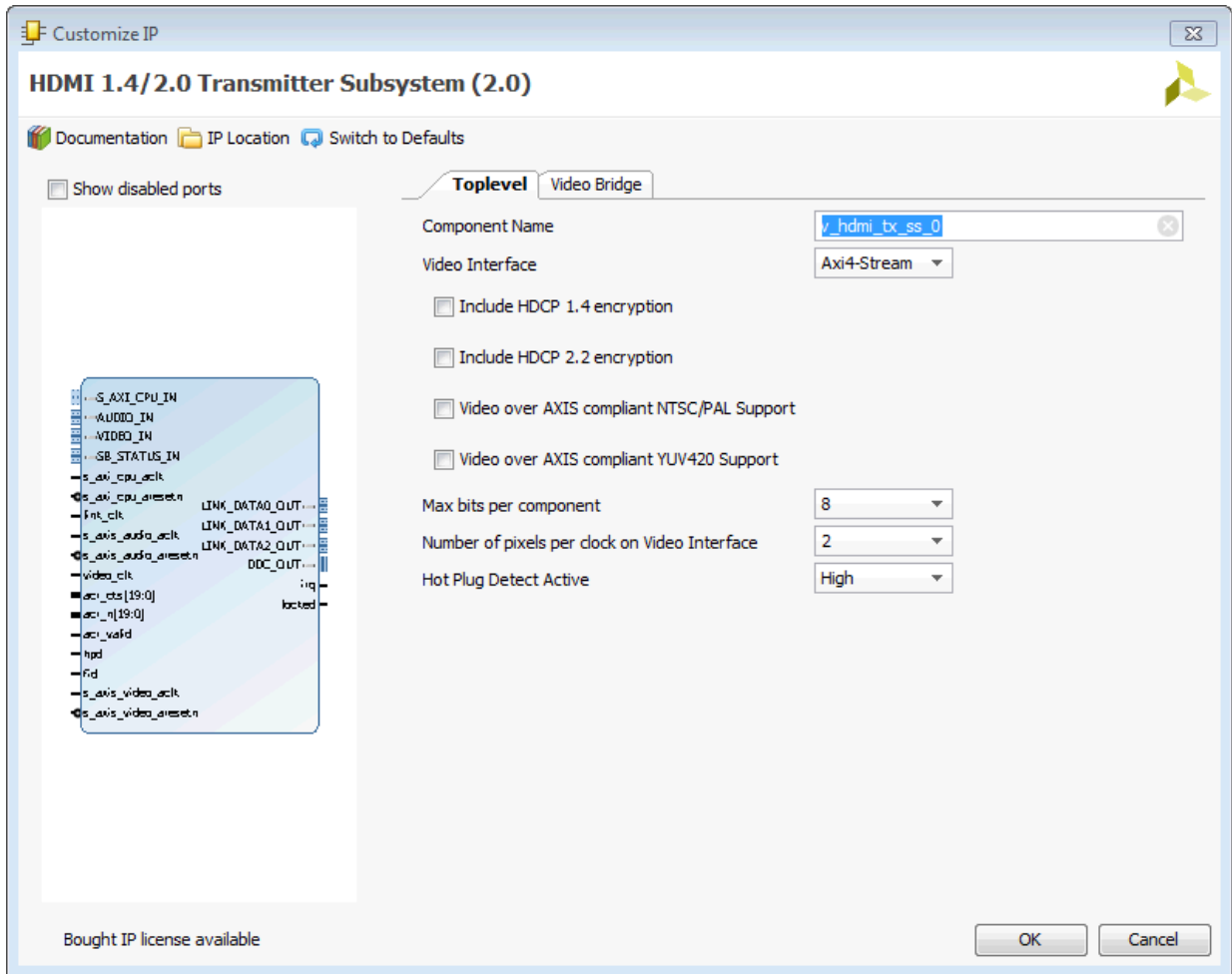


- [IP Catalog] をクリックし、[Video Connectivity] の下にある [HDMI 1.4/2.0 Transmitter Subsystem] をダブルクリックします。



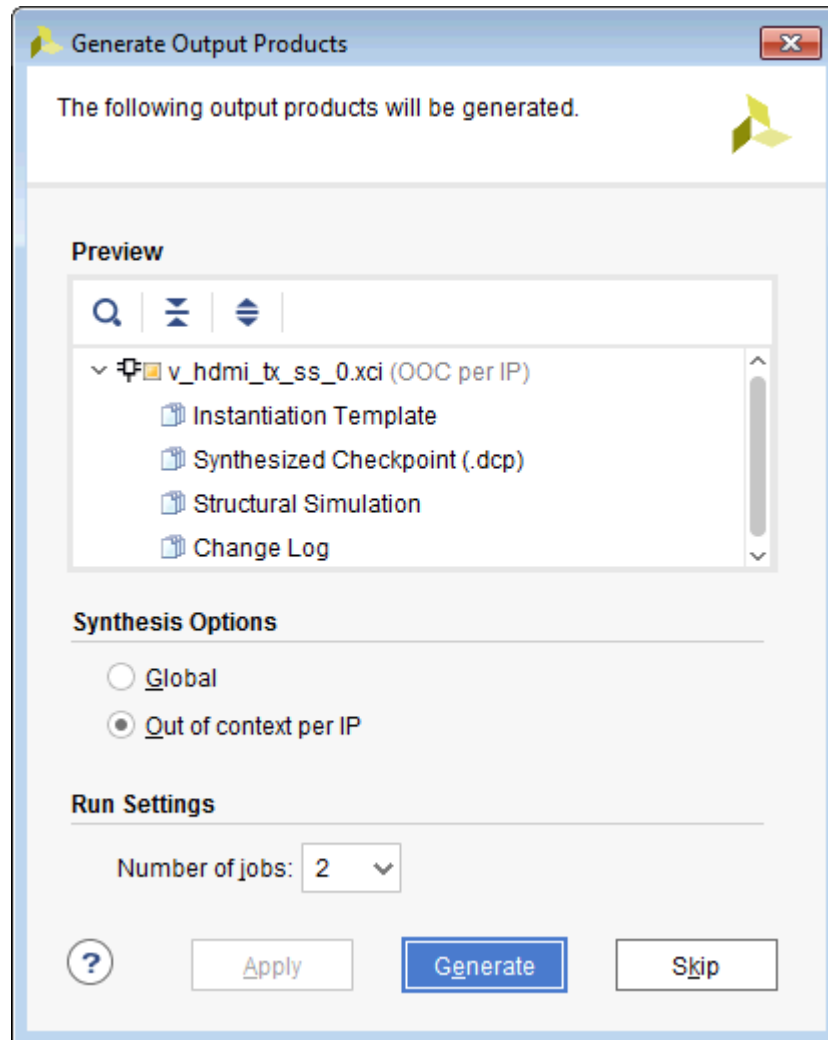
- このサンプル デザイン フローでは、ネイティブ ビデオ インターフェイスはサポートされません。
- IP コンポーネント名は変更可能です。この名前はサンプル デザインのプロジェクト名として使用されます。

6. HDMI 1.4/2.0 Transmitter Subsystem のコンフィギュレーションパラメーターを設定し、[OK] をクリックします。

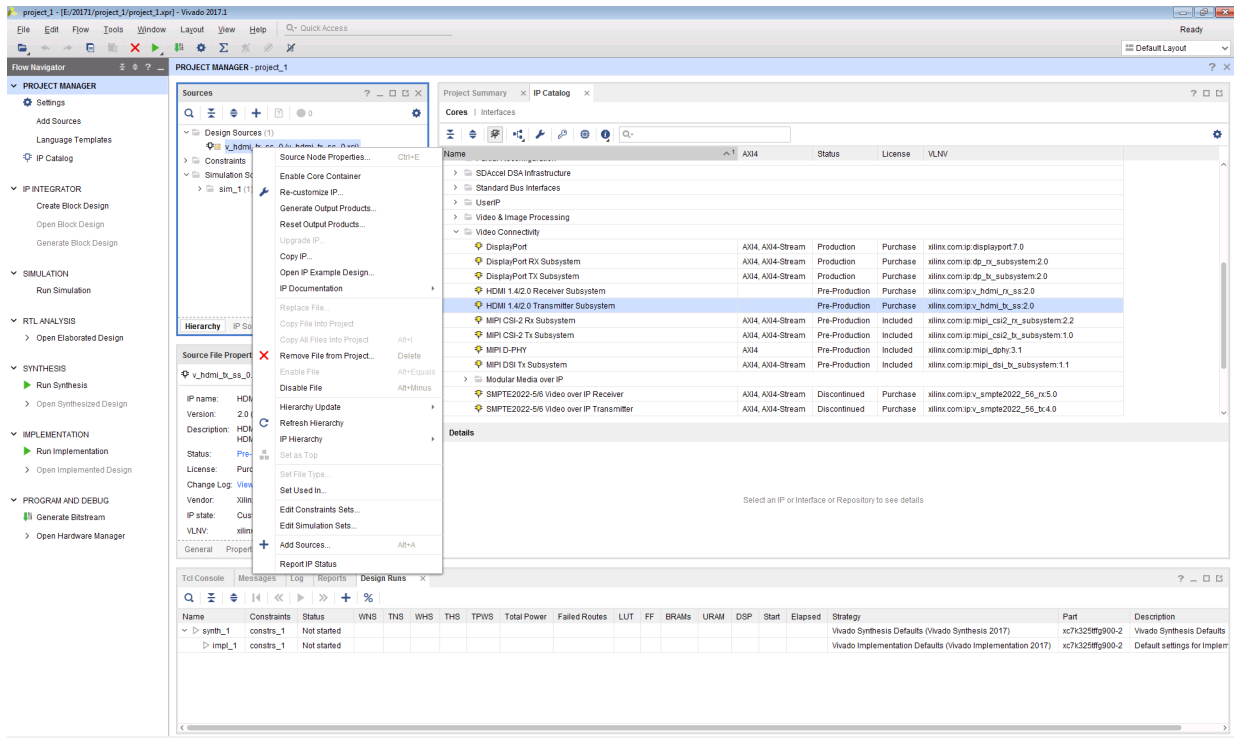


[Generate Output Products] ダイアログ ボックスが表示されます。

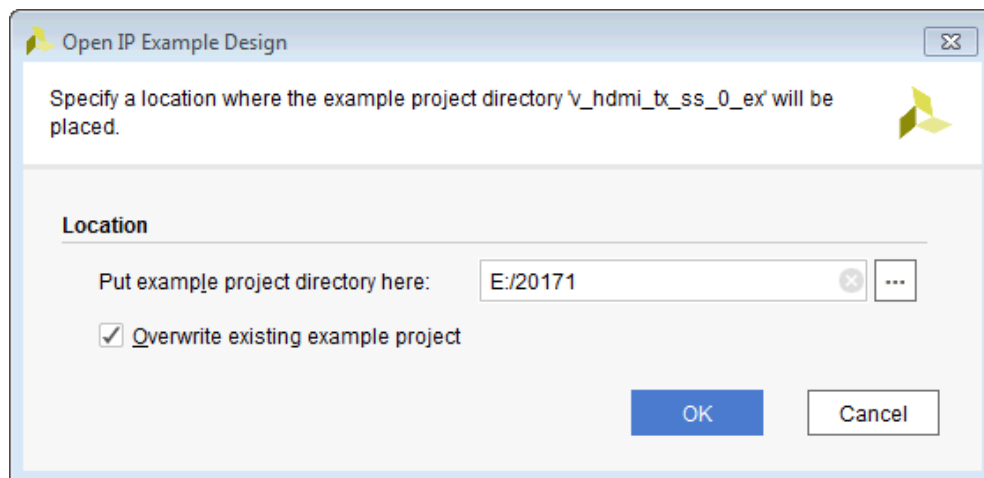
7. [Generate] をクリックします。
 - a. サンプル デザインのみを生成する場合は、[Skip] をクリックします。



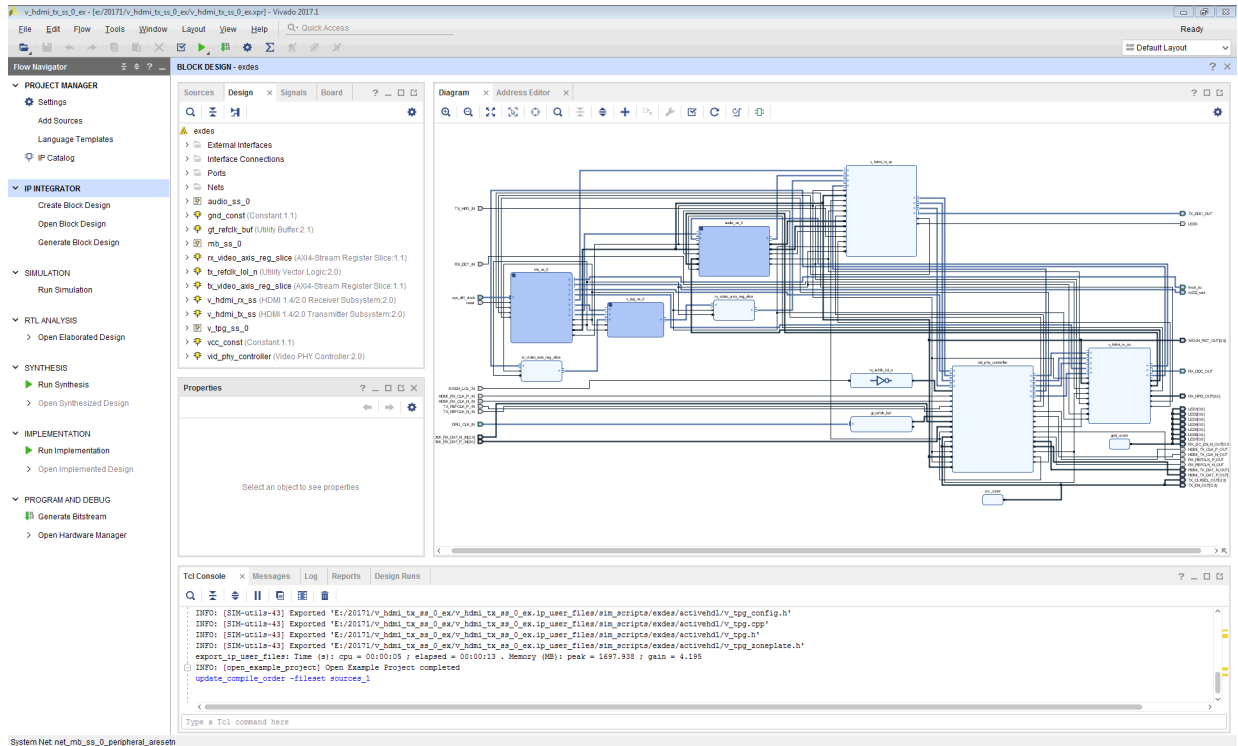
8. [Design Sources] の下にある [HDMI 1.4/2.0 Transmitter Subsystem] を右クリックし、[Open IP Example Design] をクリックします。



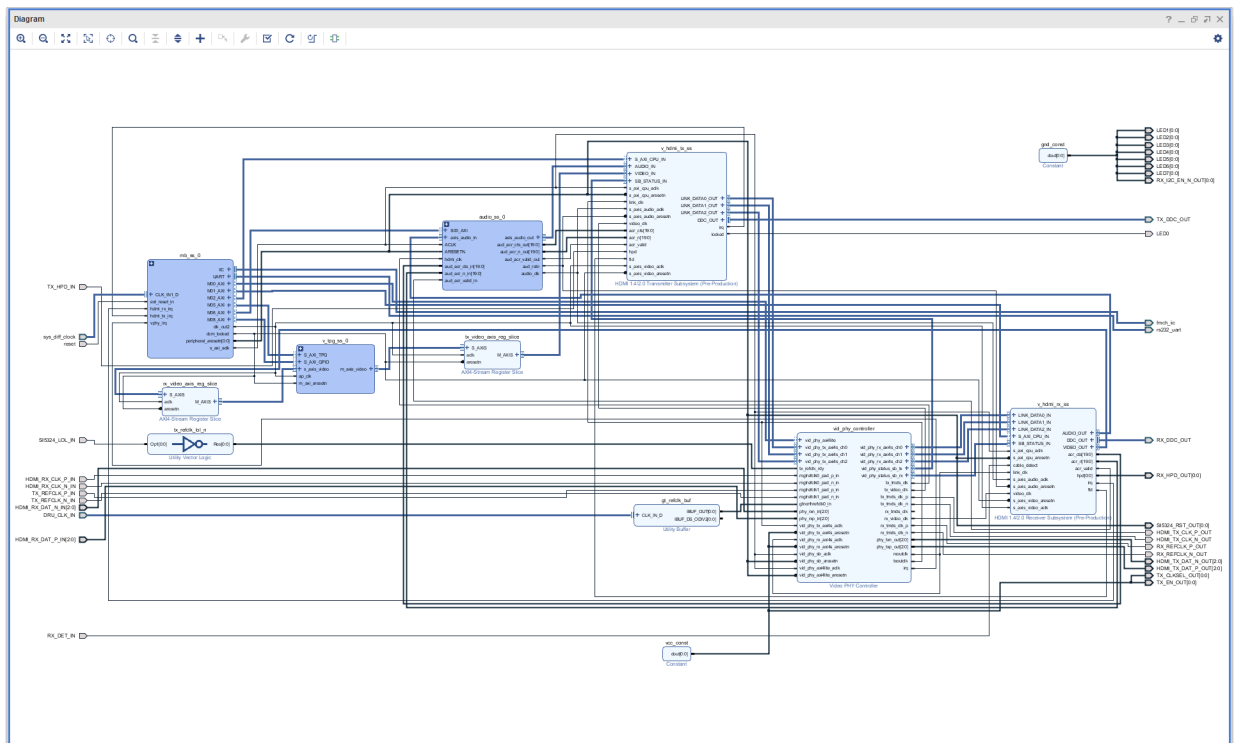
9. プロジェクトを配置する場所を選択して [OK] をクリックします。



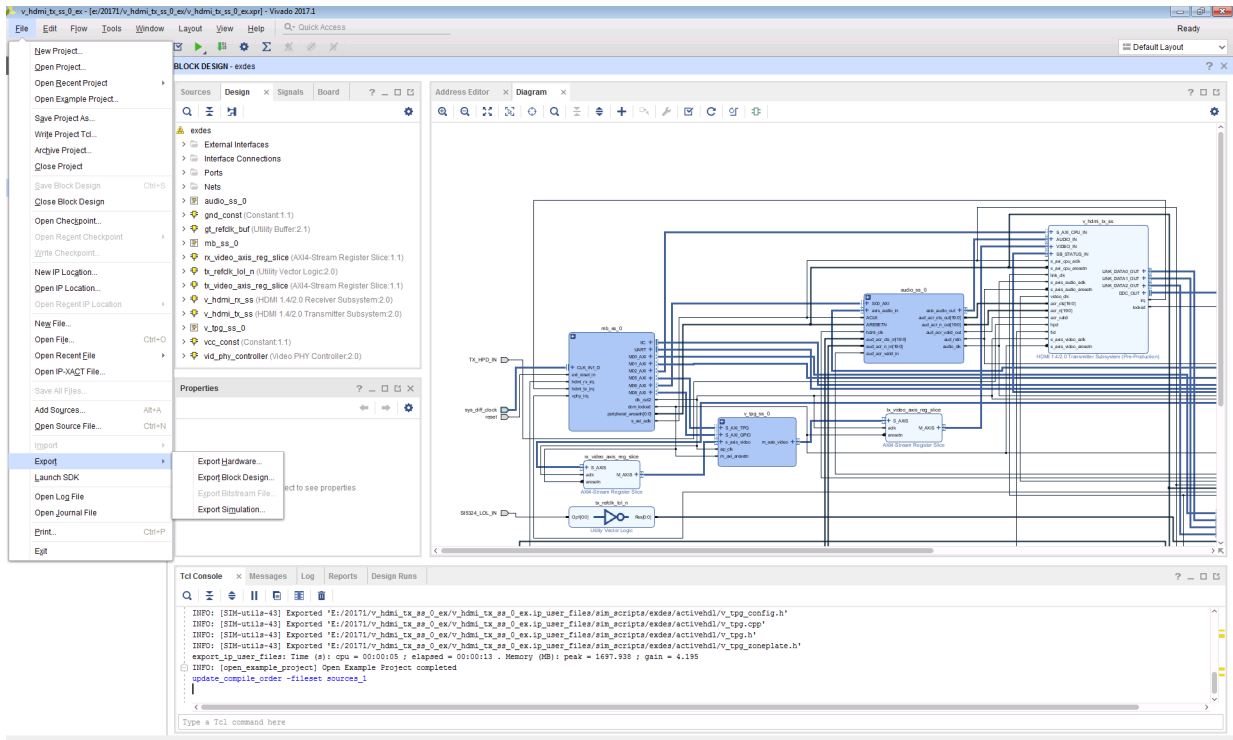
これで、IP インテグレーター デザインが生成されます。[Run Synthesis]、[Run Implementation]、または [Generate Bitstream] をクリックして実行できます。



KC705 を使用した場合のサンプル デザイン全体のシステム IP インテグレーターブロック図は次のとおりです。

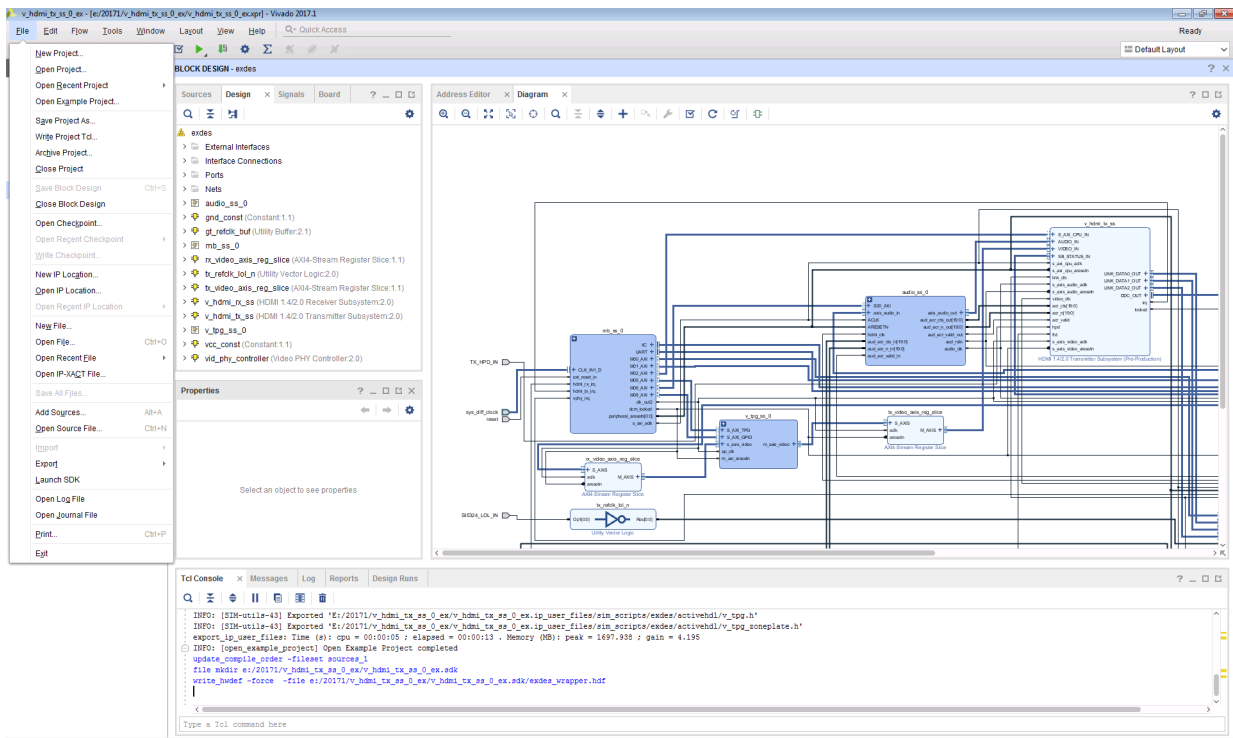


10. SDK サンプル デザイン フローを開始するために、ハードウェアをエクスポートします。

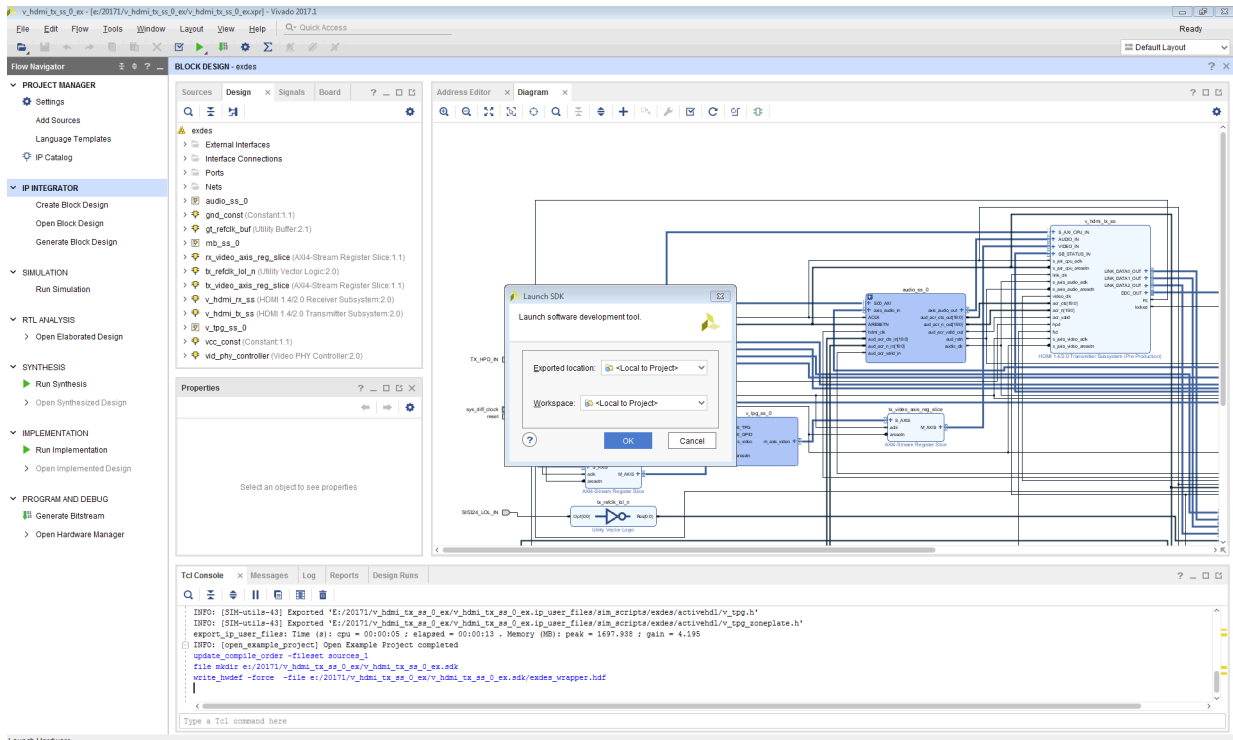


11. [OK] をクリックします。エクスポート先はサンプル デザインに対する <Local to Project> (デフォルト) のままとします。

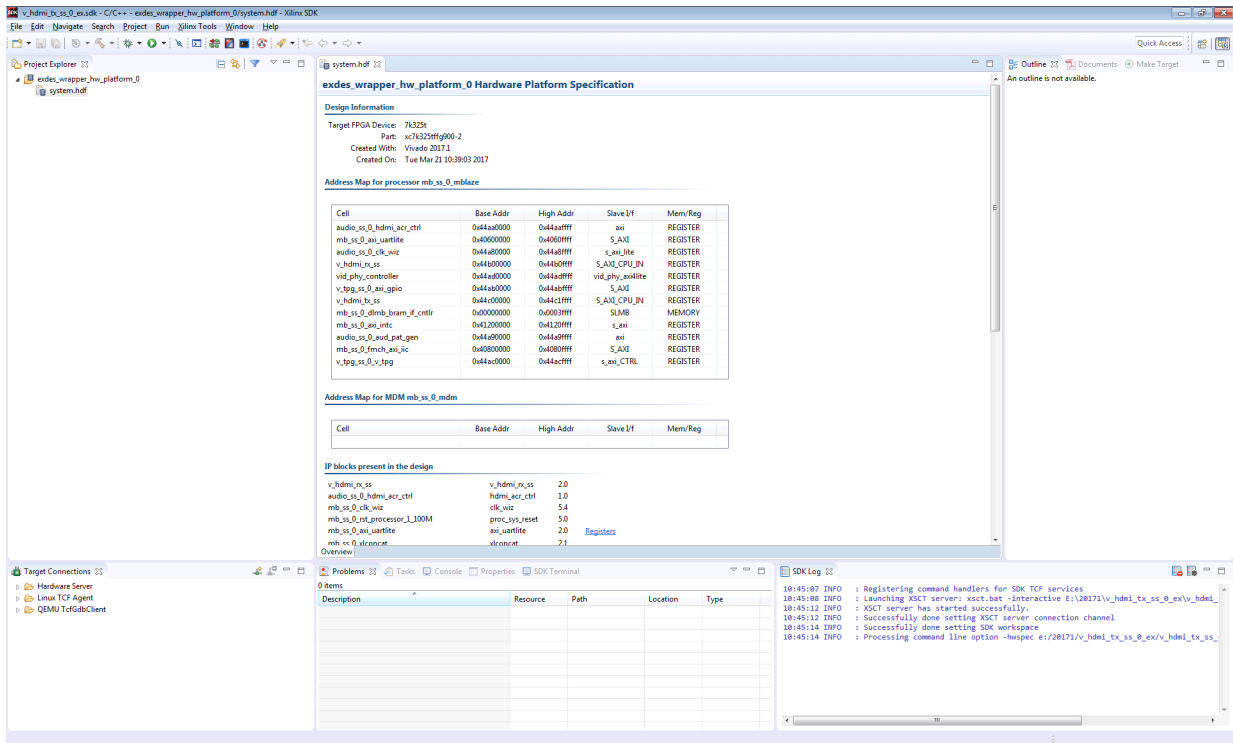
12. [Launch SDK] をクリックします。



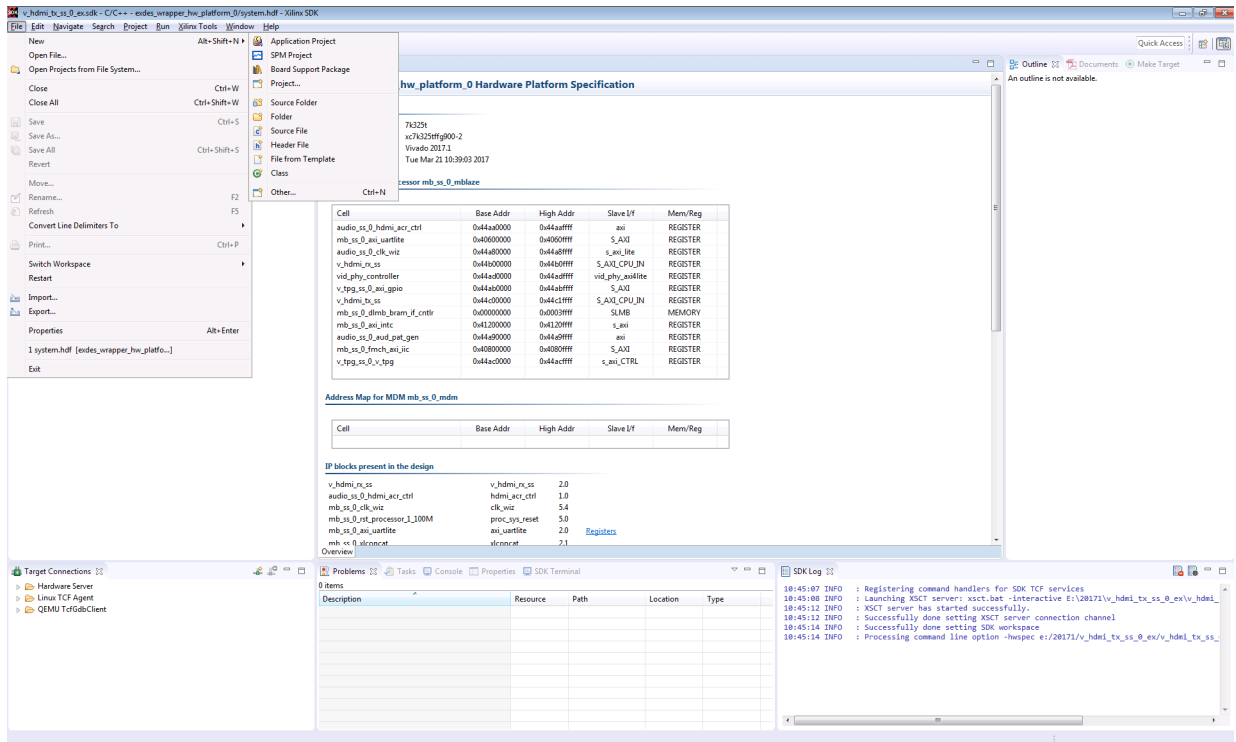
13. SDK のワークスペースの場所を選択します。デフォルトでは、<Local to Project> です。



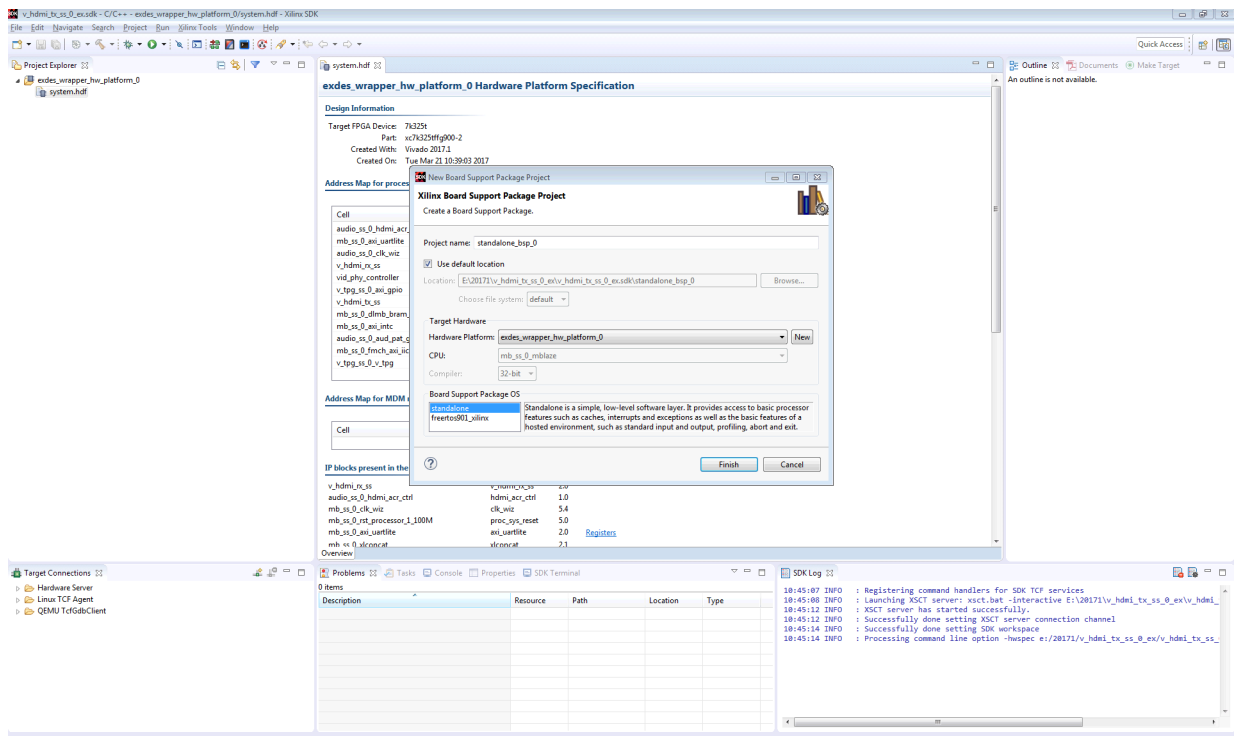
Vivado SDK が起動します。



14. ボード サポート パッケージを作成します。

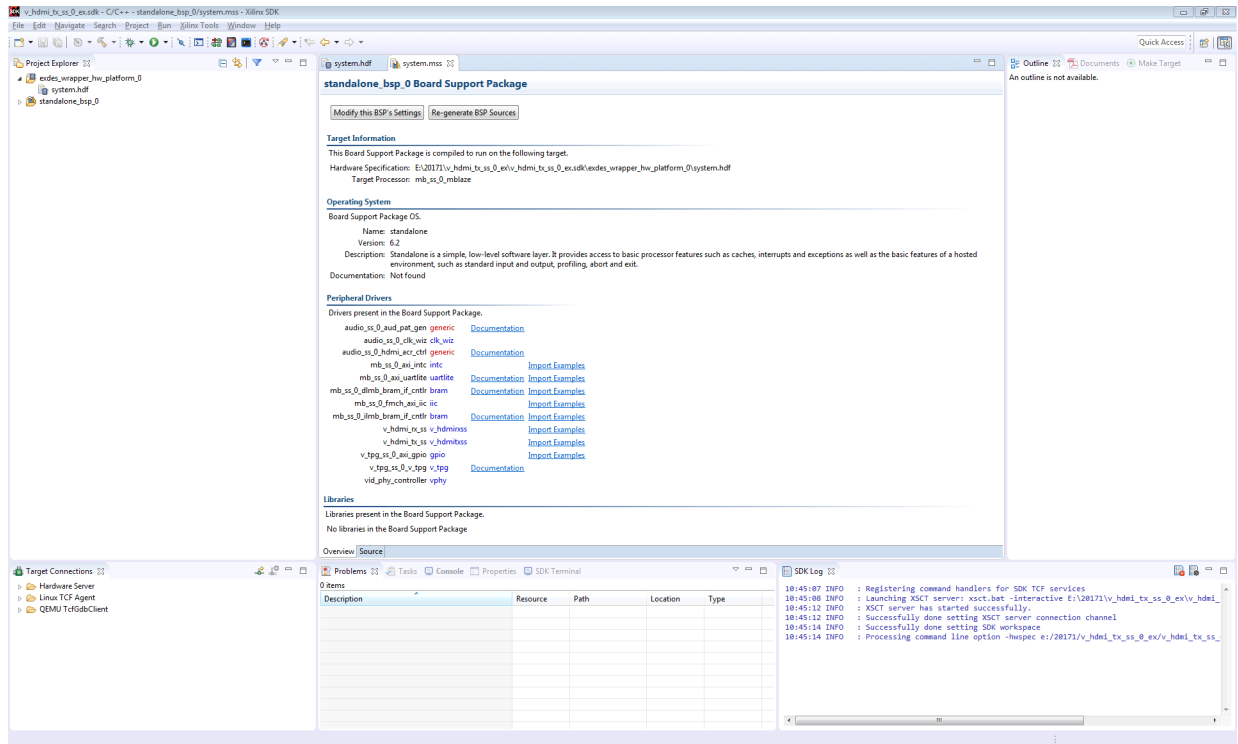


15. BSP プロジェクト名を入力し、[Finish] をクリックします。



16. [OK] をクリックします。

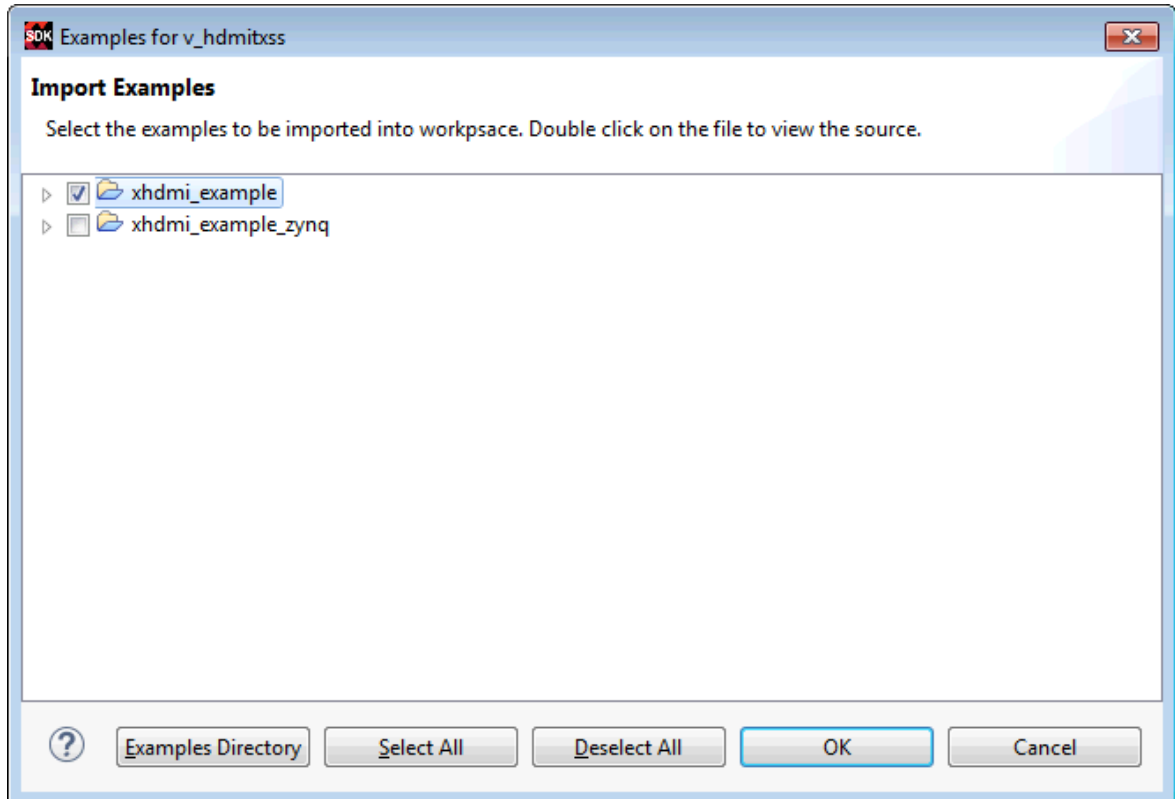
17. HDMI 1.4/2.0 Transmitter Subsystem を見つけて [Import Examples] をクリックします。



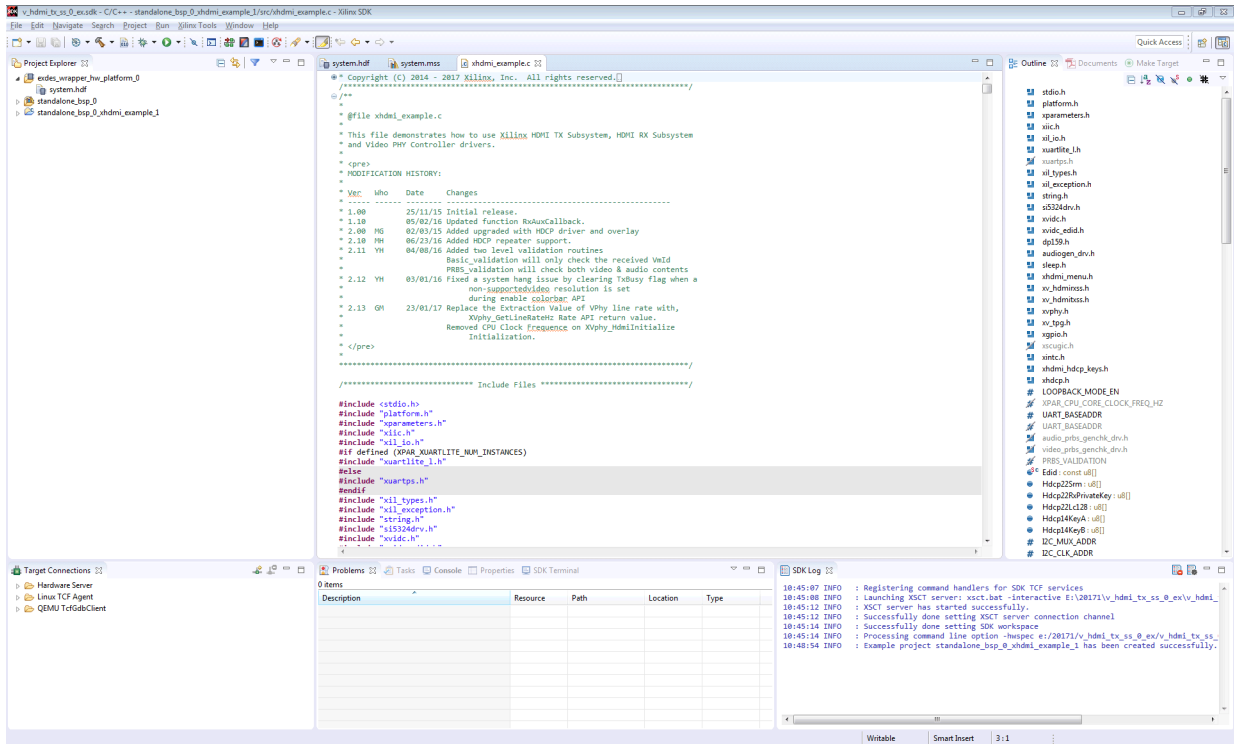
18. [xhdmi_example] をオンにします。

KC705 および KCU105 ボード (MicroBlaze™ ソフト プロセッサ コア ベース) 用に生成したプロジェクトの場合、[xhdmi_example] をオンにします。

ZC706 ボード (Zynq®-7000 SoC ARM プロセッサ ベース) 用に生成したプロジェクトの場合は、[xhdmi_example_zynq] をオンにします。



これでサンプル アプリケーションが正しくビルドされ、.elf が使用できるようになります。



リファレンス デザインを実行する (KC705)

サンプル デザインから生成したビットストリームとソフトウェア .elf を使用してシステムを実行するには、次の手順に従います。

1. [スタート] → [すべてのプログラム] → [Xilinx Design Tools] → [Vivado 2017.1] → [Vivado 2017.1 Tcl Shell] をクリックして Xilinx System Debugger を起動します。
2. ザイリンクス コマンド シェル ウィンドウで、サンプル デザインのプロジェクト ディレクトリに移動します。

```
Vivado% cd ./v_hdmi_tx_ss_0_ex
```

3. Xilinx System Debugger (xsdb) を起動します。

```
Vivado% xsdb
```

4. デバッグ ターゲットに接続します。

```
xsdb% connect
```

5. ビットストリームを FPGA にダウンロードします。

```
xsdb% fpga -file ./v_hdmi_tx_ss_0_ex.runs/impl_1/
exdes_wrapper.bit
```

6. ターゲット プロセッサを設定します。

```
xsdb% target -set 3
```

- ソフトウェア .elf を FPGA にダウンロードします。

```
xsdb% dow ./v_hdmi_tx_ss_0_ex.sdk/<name of bsp>_xhdmi_example_1/
Debug/<name of bsp>_xhdmi_example_1.elf
```

- ソフトウェアを実行します。

```
xsdb% stop
xsdb% rst
xsdb% con
```

- XSDB コマンド プロンプトを終了します。

```
xsdb% exit
```



重要: KCU105 ボードで TB-FMCH-HDMI4K サンプル デザインを使用する場合、サンプル デザイン フローで生成したビットストリームを FPGA にプログラムする前に FMC VADJ 1V8 電源レールを設定しておく必要があります。KCU105 ボードを使用する場合の VADJ 電源レールの設定手順は、「[KCU105 ボードの FMCH VADJ の調整](#)」を参照してください。KCU105 ボードの詳細は、『[KCU105 ボード ユーザー ガイド](#)』(UG917) [[参照 18](#)] を参照してください。

KCU105 ボードの FMCH VADJ の調整

KCU105 ボード システム コントローラーは、HDMI 2.0 FMC カード (TB-FMCH-HDMI4K) 用の VADJ 電源レールに電力を供給する必要があります。新しいボードのほとんどは事前にプログラムされており、正しく検出されるはずですが、(KCU105 ボードの電源スイッチの近くにある) DS19 LED が点灯しているときは、VADJ に電力が供給されています。

旧バージョンの KCU105 ボードを使用する場合や、ボードが適切にプログラムされていない場合は、ビットストリームのコンフィギュレーションの前に、HDMI 2.0 FMC カード用に VADJ 電源レールを手動で 1.8V に設定する必要があります。

UART ターミナルで VADJ 電源レールを設定するには、次の手順を実行します。

- KCU105 ボードの USB UART コネクタと Windows 搭載 PC を USB ケーブルで接続します。
- Windows デバイス マネージャーを使用して、Zynq-7000 AP SoC システム コントローラー用の UART に割り当てる仮想 COM ポートと、UltraScale FPGA 用の UART に割り当てる COM ポートを指定します。デバイス マネージャー画面の COM ポートの一覧で、CP210x に関連付けられる拡張版 COM ポートが、KCU105 ボード システム コントローラーに接続されるポートです。標準 COM ポートが、FPGA UART に接続されるポートです。
- ターミナル画面 (115200、8、N、1) を開き、その COM ポートを KCU105 ボード システム コントローラーと通信するポートに設定します。
- UART ターミナルに接続したら、KCU105 ボードの電源を切って入れ直し、UART ターミナルのシステム コントローラー メニューを更新します。システム コントローラー メニューで次のオプションをクリックします。
 - Adjust FPGA Mezzanine Card (FMC) Settings
- 次のメニューで、次のオプションをクリックします。
 - Set FMC VADJ to 1.8V

移行に関する注意

バージョン 2016.3 またはそれ以前のバージョンから移行する場合は、次の点に注意してください。

- HDMI 1.4/2.0 Transmitter Subsystem の GUI には [Hot Plug Detect Active] が追加されています。
ボード デザインに従い、このサンプル デザインでは [High] を選択してください。
- HDMI 1.4/2.0 Receiver Subsystem の GUI には [Hot Plug Detect Active] が追加されています。
ボード デザインに従い、このサンプル デザインでは [Low] を選択してください。
- HDMI 1.4/2.0 Receiver Subsystem の GUI には [Cable Detect Active] が追加されています。
ボード デザインに従い、このサンプル デザインでは [Low] を選択してください。
- サンプル デザインのアプリケーション ソフトウェアでは、HDCP 1.4/2.2 がデフォルトで有効になっています。
HDCP 1.4 または HDCP 2.2 を有効にするための UART オプションは廃止されています。
- サンプル デザインのアプリケーション ソフトウェアに自動切り換えが追加されています。

HDCP 1.4 または HDCP 2.2 を UART から選択する必要はありません。接続したソース/シンクの機能に応じた HDCP が選択されます。デバイスが HDCP 1.4 と HDCP 2.2 の両方をサポートしている場合、HDCP 2.2 が優先されます。

- HDCP リピーター機能が追加されています。
UART メニューで [h] を選択して有効/無効を切り換えることができます。
- システム ログが UART への直接プリントアウトからイベント ログに変更されています。
UART メニューで [z] を選択すると、イベント ログを表示できます。

検証、互換性、相互運用性

相互運用性

HDMI 1.4/2.0 Transmitter Subsystem の相互運用性テストは、次のハードウェア セットアップを使用して実施されています。

ハードウェア テスト

HDMI 1.4/2.0 Transmitter Subsystem の検証に使用したハードウェアは次のとおりです。

- Kintex[®]-7 FPGA 評価キット (KC705)
- Kintex[®] UltraScale™ FPGA 評価キット (KCU105)
- Inrevium Artix-7 FPGA ACDC A7 評価ボード
- Zynq[®]-7000 All Programmable SoC 評価ボード (ZC706)

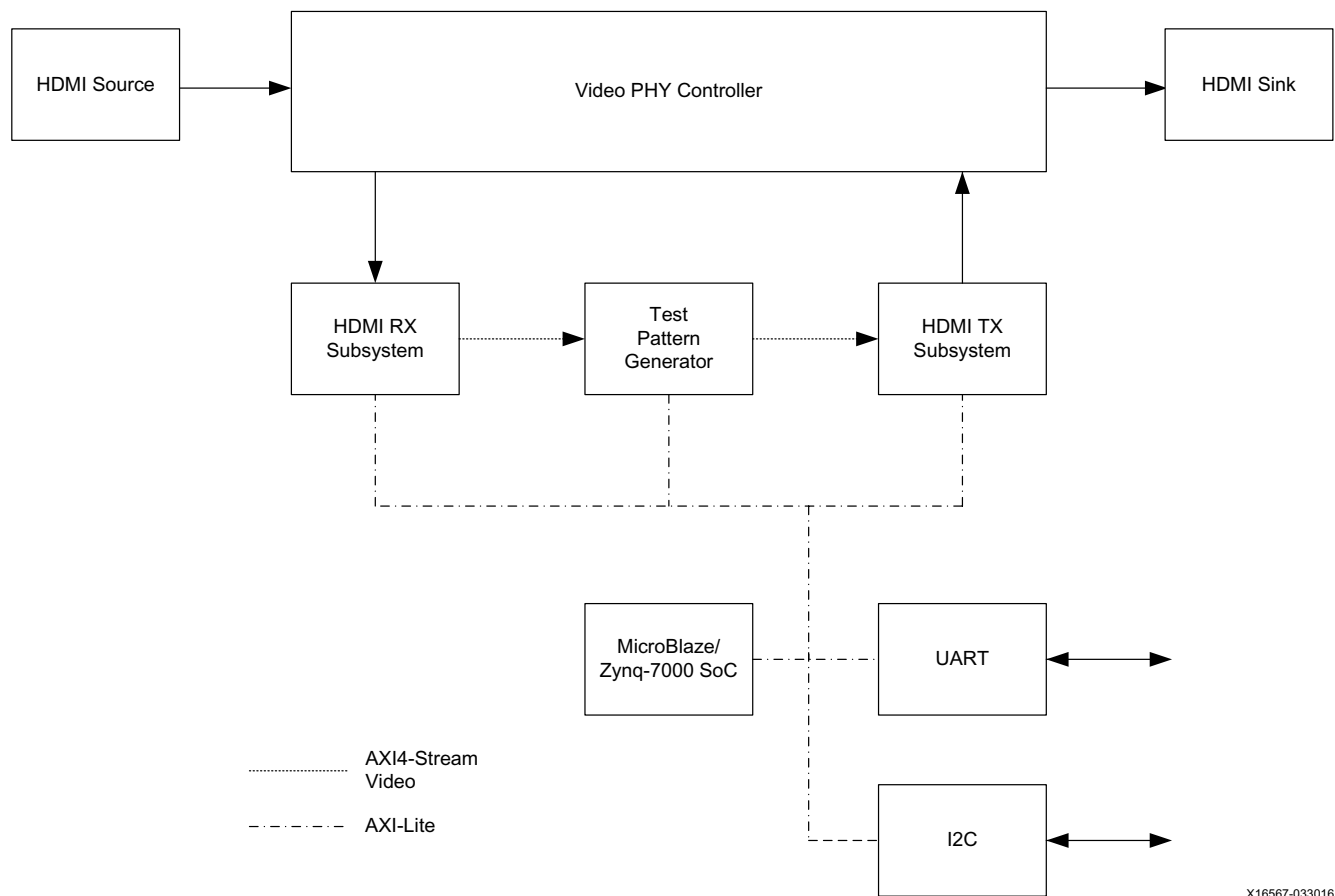
HDMI 1.4/2.0 Transmitter Subsystem のテストに使用したシンク デバイスは次のとおりです。

- Quantum Data 980B
- Quantum Data 780B
- Dell U2414Q
- Dell U2412M
- Dell U2713HM
- Acer S277HK
- Asus PQ321
- シャープ TV (LC-60LE740E)
- Philips TV (7800 シリーズ)
- Samsung UHDTV (UE40HU6900S)
- Murideo ビデオ アナライザー / SIX-A
- DELL P2415Q
- Philips 288P6LJEB
- LG 27mu67

ビデオ解像度

図 A-1 に、AXI4-Stream ビデオ インターフェイスを選択した場合のハードウェア構成を示します。HDMI ソースを Video PHY Controller に接続し、そこで HDMI ビデオをリンク データに変換した後、HDMI RX Subsystem に送信します。次に、HDMI RX Subsystem がリンク データを AXI4-Stream ビデオに変換し、Test Pattern Generator に送信します。Test Pattern Generator をパススルー モードに設定した場合、HDMI RX Subsystem からの AXI4-Stream ビデオはそのまま HDMI TX Subsystem に渡され、そこで再びリンク データに変換されてから Video PHY Controller に戻されます。次に、Video PHY Controller はリンク データを再び HDMI ビデオに変換し、HDMI シンクへ送信します。

Test Pattern Generator から AXI4-Stream ビデオ フォーマットのビデオ パターンを生成し、HDMI RX Subsystem から受信したビデオの代わりにこのビデオ パターンを使用して HDMI TX Subsystem のみをテストするようにもコンフィギュレーションできます。



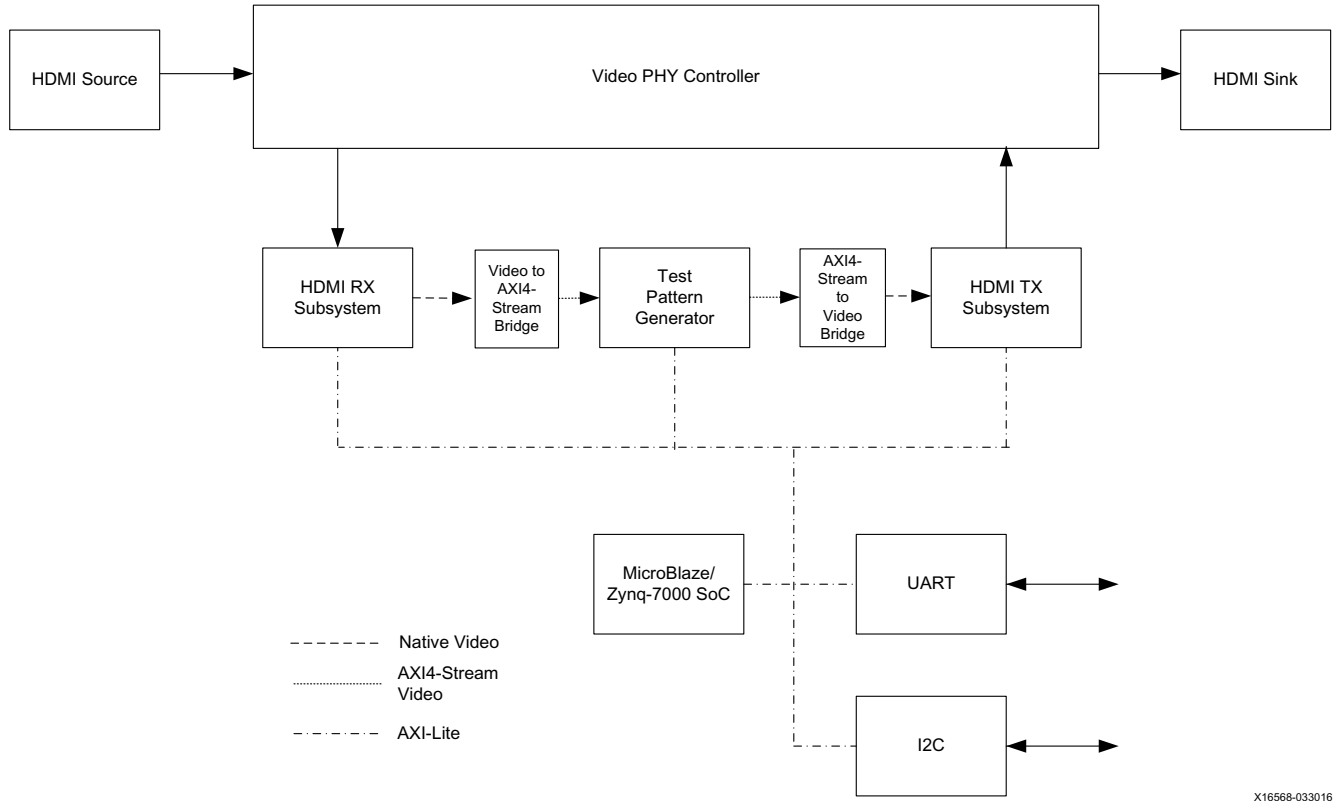
X16567-033016

図 A-1: AXI4-Stream ビデオ インターフェイスの場合のテスト構成

Video PHY Controller の設定と PLL 選択の詳細は、『Video PHY Controller LogiCORE IP 製品ガイド』(PG230) [参照 22] を参照してください。

図 A-2 に、ネイティブ ビデオ インターフェイスを選択した場合のハードウェア構成を示します。先ほどとの違いは、HDMI RX Subsystem と Test Pattern Generator の間、および Test Pattern Generator と HDMI TX Subsystem の間に Video Bridge モジュールが 1 つずつ追加されている点のみです。

これは、Test Pattern Generator から AXI4-Stream ビデオ フォーマットのビデオ パターンを生成し、HDMI RX Subsystem から受信したビデオの代わりにこのビデオ パターンを使用して HDMI TX Subsystem のみをテストするようにもコンフィギュレーションできるためです。



X16568-033016

図 A-2: ネイティブ ビデオ インターフェイスの場合のテスト構成

表 A-1、表 A-2、および表 A-3 に、このリリースでテスト済みのビデオ解像度をビデオフォーマットの種類別に示します。次の制限事項があります。

- AXI4-Stream ビデオ インターフェイスを選択して [Number of pixels per clock on Video Interface] (PPC) を 4 に設定した場合、水平総解像度が 4 の倍数のビデオフォーマットのみがサポートされます。たとえば、ビデオ解像度 720p60 の水平総ピクセル数は 1650 で、4 の倍数ではありません。したがって、PPC を 4 に設定したデザインではこの解像度はサポートされません。

表 A-1: RGB 4:4:4 および YCbCr 4:4:4 でテスト済みのビデオ解像度

解像度	水平		垂直		フレーム レート (Hz)
	合計	アクティブ	合計	アクティブ	
480i60	858	720	525	480	60
576i50	864	720	625	576	50
1080i50	2640	1920	1125	1080	50
1080i60	2200	1920	1125	1080	60
480p60	858	720	525	480	60
576p50	864	720	625	576	50
720p50	1980	1280	750	720	50
720p60	1650	1280	750	720	60
1080p24	2750	1920	1125	1080	24
1080p25	2640	1920	1125	1080	25

表 A-1: RGB 4:4:4 および YCbCr 4:4:4 でテスト済みのビデオ解像度 (続き)

解像度	水平		垂直		フレーム レート (Hz)
	合計	アクティブ	合計	アクティブ	
1080p30	2200	1920	1125	1080	30
1080p50	2640	1920	1125	1080	50
1080p60	2200	1920	1125	1080	60
1080p120	2200	1920	1125	1080	120
2160p24	5500	3840	2250	2160	24
2160p25	5280	3840	2250	2160	25
2160p30	4400	3840	2250	2160	30
2160p60	4400	3840	2250	2160	60
4096x2160p60	4400	4096	2250	2160	60
vgap60	800	640	525	480	60
svgap60	1056	800	628	600	60
xgap60	1344	1024	806	768	60
sxgap60	1688	1280	1066	1024	60
wxgap60	1440	1280	790	768	60
wxga+p60	1792	1366	798	768	60
uxgap60	2160	1600	1250	1200	60
wuxgap60	2592	1920	1245	1200	60
wsxgap60	2240	1680	1089	1050	60

注記:

1. VPHY の制限のため、一部の解像度はサポートされていません。詳細は、『Video PHY Controller LogiCORE IP 製品ガイド』(PG230) [参照 22] を参照してください。

表 A-2: YCbCr 4:2:2 (12BPC) でテスト済みのビデオ解像度

解像度	水平		垂直		フレーム レート (Hz)
	合計	アクティブ	合計	アクティブ	
1080i50	2640	1920	1125	1080	50
1080i60	2200	1920	1125	1080	60
480p60	858	720	525	480	60
576p50	864	720	625	576	50
720p50	1980	1280	750	720	50
720p60	1650	1280	750	720	60
1080p24	2750	1920	1125	1080	24
1080p25	2640	1920	1125	1080	25
1080p30	2200	1920	1125	1080	30
1080p50	2640	1920	1125	1080	50
1080p60	2200	1920	1125	1080	60

表 A-2: YCbCr 4:2:2 (12BPC) でテスト済みのビデオ解像度 (続き)

解像度	水平		垂直		フレーム レート (Hz)
	合計	アクティブ	合計	アクティブ	
2160p24	5500	3840	2250	2160	24
2160p25	5280	3840	2250	2160	25
2160p30	4400	3840	2250	2160	30
vgap60	800	640	525	480	60
svgap60	1056	800	628	600	60
wxgap60	1440	1280	790	768	60
wxga+p60	1792	1366	798	768	60
uxgap60	2160	1600	1250	1200	60
wuxgap60	2592	1920	1245	1200	60
wsxgap60	2240	1680	1089	1050	60

表 A-3: YCbCr 4:2:0 (8、10、12、16BPC) でテスト済みのビデオ解像度

解像度	水平		垂直		フレーム レート (Hz)
	合計	アクティブ	合計	アクティブ	
2160p60	4400	3840	2250	2160	60

デバッグ

この付録では、ザイリンクス サポート ウェブサイトより入手可能なリソースおよびデバッグ ツールについて説明します。



ヒント: IP の生成にエラーが発生し停止した場合、ライセンスに問題がある可能性があります。詳細は、[第 1 章の「ライセンスチェッカー」](#)を参照してください。

ザイリンクス ウェブサイト

HDMI 1.4/2.0 Transmitter Subsystem を使用した設計およびデバッグでヘルプが必要な場合は、[ザイリンクス サポート ウェブ ページ](#)から製品の資料、リリース ノート、アンサーなどを参照するか、テクニカル サポートでサービス リクエストを作成してください。

資料

この製品ガイドは HDMI 1.4/2.0 Transmitter Subsystem に関する主要資料です。このガイド、並びに設計プロセスで使用する各製品の関連資料はすべて、ザイリンクス サポート ウェブ ページ (<http://japan.xilinx.com/support>) または Xilinx Documentation Navigator から入手できます。

Xilinx Documentation Navigator は、[ダウンロード ページ](#)からダウンロードできます。このツールの詳細および機能は、インストール後にオンライン ヘルプを参照してください。

アンサー

アンサーには、よく発生する問題についてその解決方法、およびザイリンクス製品に関する既知の問題などの情報が記載されています。アンサーは、ユーザーが該当製品の最新情報にアクセスできるよう作成および管理されています。

このコアに関するアンサーの検索には、[ザイリンクス サポート ウェブ ページ](#)にある検索ボックスを使用します。より確かな検索結果を得るには、次のようなキーワードを使用してください。

- 製品名
- ツールで表示されるメッセージ
- 問題の概要

検索結果は、フィルター機能を使用してさらに絞り込むことができます。

HDMI 1.4/2.0 Transmitter Subsystem に関するマスター アンサー

AR: [65911](#)

テクニカル サポート

ザイリンクスは、製品資料の説明に従って使用されている LogiCORE™ IP 製品に対するテクニカル サポートを japan.xilinx.com/support で提供しています。次のいずれかに該当する場合、タイミング、機能、サポートは保証されません。

- 資料で定義されていないデバイスにソリューションをインプリメントした場合。
- 資料で定義されている許容範囲を超えてカスタマイズした場合。
- 「DO NOT MODIFY」とされているデザイン セクションに変更を加えた場合。

ザイリンクス テクニカル サポートへのお問い合わせに関しては、[ザイリンクス サポート ウェブ ページ](#)を参照してください。

デバッグ ツール

HDMI 1.4/2.0 Transmitter Subsystem デザインの問題を解決するには、いくつかのツールを利用できます。さまざまな状況をデバッグするのに有益なツールを理解しておくことが重要です。

Vivado Design Suite のデバッグ機能

Vivado® Design Suite のデバッグ機能は、Logic Analyzer および Virtual I/O コアをユーザー デザインに直接挿入します。デバッグ機能を使用すると、トリガー条件を設定して、アプリケーションおよび統合ブロックのポート信号をハードウェアに取り込むことができます。取り込まれた信号は、その後解析できます。Vivado IDE のこの機能は、ザイリンクス デバイスで実行されるデザインの論理デバッグおよび検証に使用されます。

Vivado ロジック解析は次の IP ロジック デバッグ コアと共に使用されます。

- ILA 2.0 (およびそれ以降のバージョン)
- VIO 2.0 (およびそれ以降のバージョン)

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 19] を参照してください。

リファレンス ボード

HDMI 1.4/2.0 Transmitter Subsystem はさまざまなザイリンクス開発ボードでサポートされています。これらのボードを使用してデザインのプロトタイプを作成し、サブシステムがシステムと通信できるようにします。

- 7 シリーズ FPGA 評価ボード
 - KC705
- UltraScale FPGA 評価ボード
 - KCU105
- Zynq-7000 All Programmable SoC 評価ボード
 - ZC706

ハードウェア デバッグ

ハードウェアの問題は、リンク立ち上げ時の問題から、テスト後に生じる問題までさまざまです。ここでは、一般的な問題のデバッグ手順を説明します。Vivado のデバッグ機能は、ハードウェア デバッグに有益なリソースです。次の各セクションに示す信号を Vivado のデバッグ機能でプローブすることで、個々の問題をデバッグできます。

一般的なチェック

- インプリメンテーション時に、すべてのタイミング制約およびその他の制約すべてが満たされていることを確認します。
- すべてのクロック ソースがアクティブでクリーンであることを確認してください。
- デザインで MMCM を使用している場合、locked ポートをモニターして、すべての MMCM がロックしていることを確認します。
- 出力が 0 になった場合は、ライセンスを確認してください。
 - ユーザー LED (KC705/KCU105/ZC706)
 - LED0 - HDMI TX Subsystem ロック (HDMI サンプル デザイン使用時)
 - デバッグ ポートを使用して、Video PHY Controller コアからリンク データを受信しているかどうかをチェックします。
 - 『Video PHY Controller LogiCORE IP 製品ガイド』(PG230) [参照 22] の付録「デバッグ」を参照し、クロッキングに問題がないことを確認します。

インターフェイスのデバッグ

AXI4-Lite インターフェイス

デフォルトがすべて 0 でないレジスタから読み出して、インターフェイスが機能していることを確認します。読み出しアドレスが有効になると出力 `s_axi_arready` がアサートされ、読み出しデータ/応答が有効になると `s_axi_rvalid` がアサートされます。インターフェイスが応答しない場合は、次を確認します。

- `s_axi_aclk` および `aclk` 入力が接続されており、トグルしていることを確認します。
- インターフェイスがリセット状態に保持されておらず、`s_axi_areset` がアクティブ Low のリセットであることを確認します。
- インターフェイスが有効になっており、`s_axi_aclken` がアクティブ High であることを確認します (使用されている場合)。
- メインのサブシステム クロックがトグルしており、イネーブル信号がアサートされていることを確認します。
- ILA に AXI4-Lite インターフェイスを追加し、`s_axi_rvalid` でトリガーしたときに取り込まれたデータを解析します。

AXI4-Stream インターフェイス

データが送信または受信されていない場合は、次を確認します。

- 受信の <interface_name>_tready が Low のままになる場合、サブシステムはデータを送信できません。AXI4 Stream のスレーブに問題がないかをチェックします。
- ac1k 入力が接続されており、トグルしていることを確認します。
- AXI4-Stream の波形に従っていることを確認します。
- サブシステムの設定を確認します。

AXI4-Stream オーディオ インターフェイス

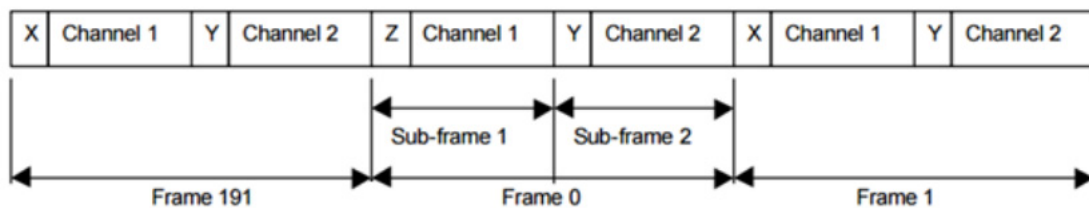
HDMI 1.4/2.0 Transmitter Subsystem でオーディオが正しく動作していることを確認するには、次に示す方法で AXI4-Stream を構成する必要があります。

HDMI 1.4/2.0 Transmitter Subsystem は最大 8 チャンネルのオーディオをサポートしています。オーディオ データは AXI4-Stream オーディオ インターフェイスで送信されます。これは、オーディオ サンプルに AES3 仕様で定義された サイドバンド信号を付加して送信できるように AXI4-Stream プロトコルをカスタマイズしたインターフェイスです。

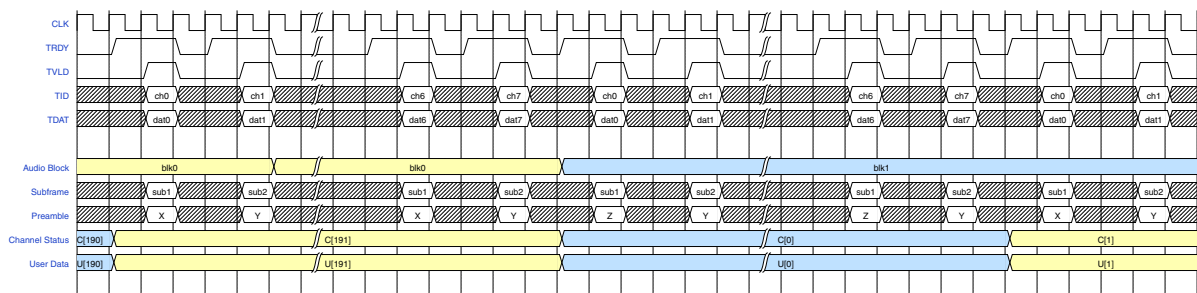
オーディオ サンプルのサブフレーム フォーマットは次のとおりです。

0	3	4	27	28	29	30	31		
Preamble	LSB		24-bit audio sample word		MSB	Validity bit	User data bit	Channel status bit	Parity bit

1 フレームは 2 つのサブフレームで構成されます。第 1 サブフレームは通常プリアンブル「X」で始まり、第 2 サブフレームは常にプリアンブル「Y」で始まります。ただし、192 フレームで 1 つの「オーディオブロック」を構成しており、「オーディオブロック」の第 1 サブフレームのみ、プリアンブルが「Z」で始まります (下図参照)。



チャンネル数が 2 を超える場合、2 チャンネルごとに 1 つの AES3 オーディオブロックと見なされます。たとえば、8 オーディオチャンネルを使用する場合、1 オーディオブロックは 192*8 オーディオ サンプルで構成されます。オーディオブロックの最初の 8 サンプルでは、オーディオ ch0、ch2、ch4、ch6 のプリアンブルが「Z」です。オーディオブロックの残りの部分では、オーディオ ch0、ch2、ch4、ch6 のプリアンブルは「X」です。オーディオブロック全体で、オーディオ ch1、ch3、ch5、ch7 のプリアンブルは常に「Y」です。8 チャンネル オーディオの場合を次に図示します。



アプリケーション ソフトウェア 開発

デバイス ドライバー

HDMI 1.4/2.0 Transmitter Subsystem ドライバー (以下、サブシステム ドライバー) はサブシステムに含まれるサポート モジュールを抽象化し、これらを制御するための API を提供します。API はユーザー アプリケーションへの統合が容易で、そのまますぐに使用できます。

サブシステム ドライバーはベアメタル ドライバーで、各サブコアが提供する機能セットを抽象化した形で可視化します。サブシステム ドライバーは、I/O ストリームの構成に基づいて実行時にプロセッシング エレメントからデータおよび制御フローを動的に管理します。また、内部でサブコア ドライバーを使用してサブコア IP ブロックをコンフィギュレーションします。

アーキテクチャ

サブシステム ドライバーには明確に定義された使いやすい API があります。このため、各サブコアを 1 つずつ理解してコンフィギュレーションするといった複雑な手順を踏まなくても、サブシステムは簡単にアプリケーションに統合できます。

サブシステム ドライバーの構成は次のとおりです。

- **サブシステム レイヤー**: エクスポートされたハードウェアに問い合わせでサブシステムのハードウェア構成を判定し、ビルド時にサブコア ドライバーを取り込みます。レジスタ レベルでハードウェアと通信するサブコア ドライバーを API 関数セットとして抽象化します。サブシステム ドライバーはこれらの API を使用して、プロセッシング エレメントからデータ フローを動的に管理します。
- **サブコア ドライバー**: サブシステムに含まれるすべてのサブコアにそれぞれドライバーがあり、コア ハードウェアとはこのドライバーが提供する API を介して接続します。

図 C-1 に、HDMI 1.4/2.0 Transmitter Subsystem のアーキテクチャを示します。

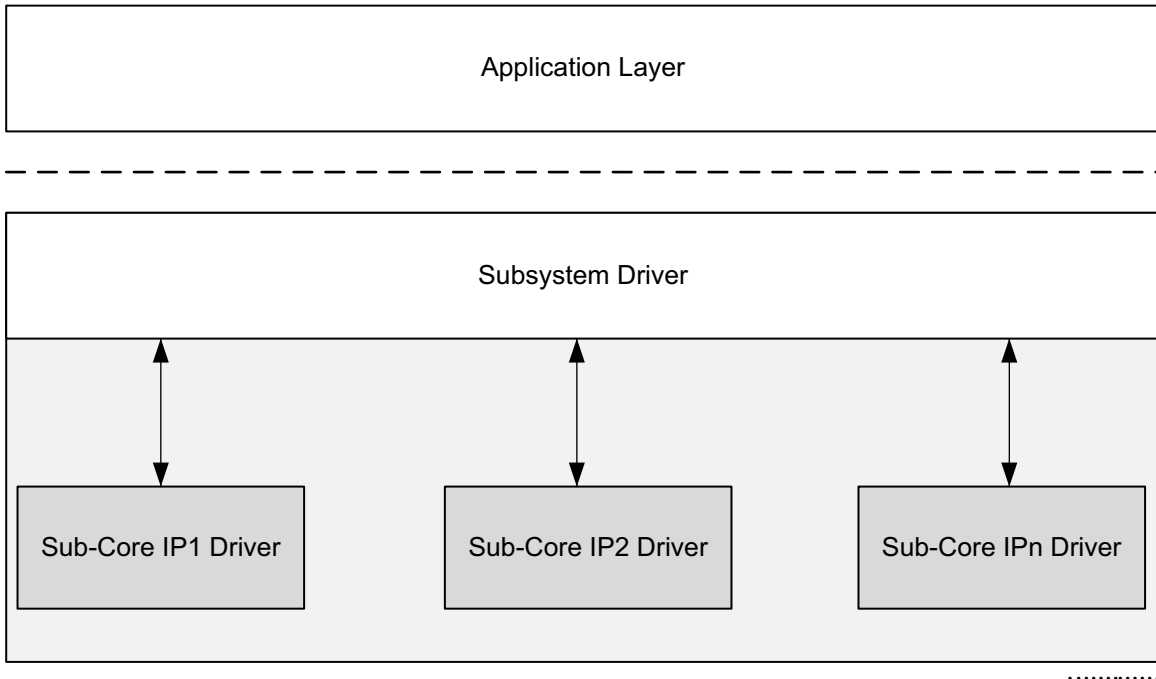
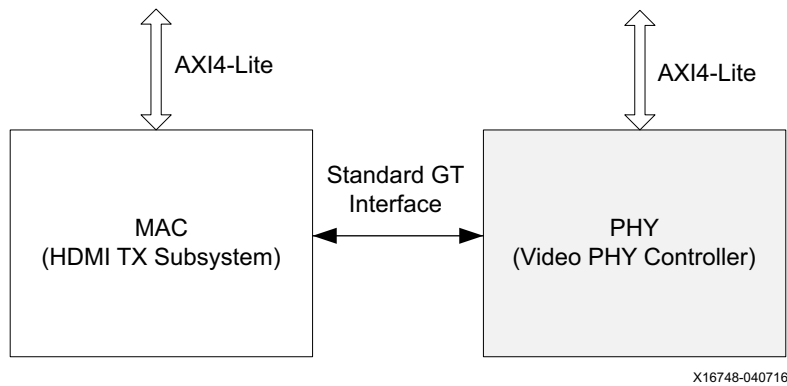


図 C-1: サブシステムドライバーのアーキテクチャ

HDMI 1.4/2.0 Transmitter Subsystem は MAC サブシステムで、Video PHY Controller (PHY) と組み合わせてビデオ コネクティビティ システムを構成します。HDMI 1.4/2.0 Transmitter Subsystem はザイリンクス Video PHY Controller と密に結合しています。Video PHY Controller はそれ自体独立しており、複数のプロトコルをサポートする柔軟なアーキテクチャを備えています。MAC と PHY はどちらも AXI4-Lite インターフェイスを利用して動的にプログラムできます。



X16748-040716

図 C-2: MAC と PHY のインターフェイス

使用法

HDMI 1.4/2.0 Transmitter Subsystem には、アプリケーション コードから使用可能な API 関数セットがあります。さらに、HDMI 1.4/2.0 Transmitter Subsystem でハードウェア割り込みが生成されると、サブシステムドライバが呼び出されてシステムが適切に設定されます。HDMI 1.4/2.0 Transmitter Subsystem には、ユーザー定義のコールバック関数をフックするためのコールバック構造があります。

ビデオ ストリームが開始していることを確認します。ビデオがロックされた後、有効な AUX データとオーディオ データを挿入できます。ただし、アプリケーションはどのビデオ フォーマットが送信され、どのオーディオ フォーマットが組み込まれるかを認識しているため、この情報を使用して、オーディオ ストリームの送信準備が完了する前に ACR の数値の計算と設定が可能です。

以降のセクションでは、HDMI に関連するモジュールのみを取り上げます。タイマー、UART、外部システム クロック ジェネレーターなどのシステム ペリフェラルはユーザー アプリケーションで管理する必要があります。

アプリケーションへの統合

図 C-3 に、HDMI 1.4/2.0 Transmitter Subsystem を実際のアプリケーションで使用方法をサンプル コードで示します。

```

1819     XV_HdmiTxSs HdmiTxSs;          /* HDMI TX SS structure */
1820     XV_HdmiTxSs_Config *XV_HdmiTxSs_ConfigPtr;
1821
1822     // Initialize HDMI TX Subsystem
1823     XV_HdmiTxSs_ConfigPtr =
1824         XV_HdmiTxSs_LookupConfig(XPAR_V_HDMI_TX_SS_0_V_HDMI_TX_DEVICE_ID);
1825
1826     if(XV_HdmiTxSs_ConfigPtr == NULL)
1827     {
1828         HdmiTxSs.IsReady = 0;
1829     }
1830
1831     //Initialize top level and all included sub-cores
1832     Status = XV_HdmiTxSs_CfgInitialize(&HdmiTxSs, XV_HdmiTxSs_ConfigPtr,
1833                                         XV_HdmiTxSs_ConfigPtr->BaseAddress);
1834     if(Status != XST_SUCCESS)
1835     {
1836         xil_printf("ERR:: HDMI TX Subsystem Initialization failed %d\r\n", Status);
1837     }
1838
1839     //Register HDMI TX SS Interrupt Handler with Interrupt Controller
1840     Status |= XIntc_Connect(&Intc,
1841                            XPAR_MICROBLAZE_SS_AXI_INTC_0_V_HDMI_TX_SS_0_IRQ_INTR,
1842                            (XInterruptHandler)XV_HdmiTxSS_HdmiTxIntrHandler,
1843                            (void *)&HdmiTxSs);
1844
1845     if (Status == XST_SUCCESS){
1846         XIntc_Enable(&Intc,
1847                     XPAR_MICROBLAZE_SS_AXI_INTC_0_V_HDMI_TX_SS_0_IRQ_INTR);
1848     }
    
```

図 C-3: アプリケーション サンプル コード

実際のアプリケーションに HDMI 1.4/2.0 Transmitter Subsystem ドライバーを統合して使用する場合は、次の手順に従います。

1. サブシステム オブジェクトを定義したサブシステム ヘッダー ファイル `xv_hdmitxss.h` をインクルードします。
2. アプリケーション コードでサブシステムのインスタンスを宣言し、空間を割り当てます。次に例を示します。

```
XV_HdmiTxSs HdmiTxSs;
```
3. サブシステム ドライバー インスタンスには、サブシステムのハードウェア構成を格納するメタデータ構造があります。アプリケーション コードでそのインスタンスを指し示すポインター変数を宣言します。

```
XV_HdmiTxSs_Config *XV_HdmiTxSs_ConfigPtr;
```
4. 各サブシステム インスタンスについて、手順 2 および 3 で宣言したデータ構造体をそれぞれのハードウェア構成に基づいて初期化する必要があります。ハードウェア構成は `xparameters.h` からメタデータ構造を通じて渡され、固有のデバイス ID で識別されます。

サブシステムを初期化するには、次の 2 つの API 関数を呼び出します。

```
XV_HdmiTxSs_Config* XV_HdmiTxSs_LookupConfig(u32 DeviceId);
int XV_HdmiTxSs_CfgInitialize(XV_HdmiTxSs *InstancePtr,
                              XV_HdmiTxSs_Config *CfgPtr,
                              u32 EffectiveAddr);
```

デバイス ID は `xparameters.h` に記述してあります。

```
XPAR_[HDMI TX Subsystem Instance Name in IPI]_DEVICE_ID
```

5. 各割り込みソースには、ISR との関連付けがサブシステムで定義されています。システム割り込みコントローラーに ISR を登録し、割り込みを有効にします。

```
int XIntc_Connect(XIntc          *InstancePtr,
                 u8              Id,
                 XInterruptHandler Handler,
                 void            *CallBackRef);
void XIntc_Enable(XIntc          *InstancePtr,
                 u8              Id);
```

ID は `xparameters.h` に記述してあります。

HDCP TX の概要

HDMI 1.4/2.0 Transmitter Subsystem ドライバーは HDCP 1.4 および HDCP 2.2 ドライバーの API を 1 つの共通 API に結合し、ユーザー レベル アプリケーションはこの共通 API を使用します。HDCP ドライバーの共通 API は、HDCP 1.4 のみ、HDCP 2.2 のみ、およびその両方の HDCP 構成を処理できます。両方のプロトコルを有効にした場合、共通 HDCP ドライバーは両方が同時にアクティブにならないようにします。

HDCP TX ドライバーの統合

このセクションでは、HDCP TX を初期化して実行するのに必要な手順について説明します。ドライバーを正しく動作させるには、おおよそここに示した順にアプリケーションで関数を呼び出す必要があります。HDCP 1.4 と HDCP 2.2 のどちらか一方のプロトコルのみを有効にした場合、使用しない関数呼び出しは不要です。

1. HDCP プロダクション キーを HDMI サブシステムに読み込みます。この関数は、読み込むキーごとに呼び出す必要があります。HDCP 1.4 と HDCP 2.2 を両方有効にした場合はすべてのキーを読み込みます。それ以外の場合は、必要なキーのみを読み込みます。HDCP のキー オクテット文字列を格納するバイト アレイはビッグ エンディアンのバイト順で定義される点に注意してください。
 - `XV_HdmiTxSs_HdcpSetKey`
 - `XV_HDMITXSS_KEY_HDCP14`
 - `XV_HDMITXSS_KEY_HDCP22_LC128` (128 ビット DCP ライセンス取得済み定数)

2. HDCP キーを読み込んだ後、HDMI 1.4/2.0 Transmitter Subsystem ドライバーを初期化します。サブシステムを初期化すると、HDCP 1.4/2.2 ドライバーが内部で開始します。
3. HDCP 割り込みハンドラーを割り込みコントローラーの割り込み ID に接続します。
 - XV_HdmiTxSS_HdcpIntrHandler
 - XV_HdmiTxSS_HdcpTimerIntrHandler
 - XV_HdmiTxSS_Hdcp22TimerIntrHandler
4. HDCP 認証済みユーザー コールバックを設定します。このコールバックは、HDCP 認証ステート マシンが認証済みステートに遷移すると実行されます。アプリケーション レベルでこの関数を使用しない場合は、コールバックを未定義のままにします。この関数を使用すると、認証に成功した後で暗号化を自動的に有効にできます。
 - XV_HdmiTxSs_SetCallback
 - XV_HDMITXSS_HANDLER_HDCP_AUTHENTICATE
5. ポーリング関数を実行して HDCP ステート マシンを実行します。この関数はどの HDCP プロトコルが有効かを確認し、アクティブなプロトコルのみを実行します。この関数呼び出しは、ユーザー アプリケーションのメインループに挿入して継続的に実行するようにします。HDCP TX ステート マシンはこのポーリング関数を使用して動作するため、特に認証の実行中などではこの関数に十分な CPU 実行時間が割り当てられるように注意する必要があります。
 - XV_HdmiTxSs_HdcpPoll
6. HDCP プロトコルを HDCP 1.4、HDCP 2.2、またはなしに設定します。どのプロトコルがアクティブかを HDMI サブシステムに通知し、プロトコルの競合を防ぐために、HDCP プロトコルを設定する必要があります。また、どのプロトコルがアクティブになっているかもチェックします。
 - XV_HdmiTxSs_HdcpSetProtocol
 - XV_HDMITXSS_HDCP_NONE
 - XV_HDMITXSS_HDCP_14
 - XV_HDMITXSS_HDCP_22
 - XV_HdmiTxSs_HdcpGetProtocol
7. アクティブな HDCP プロトコルを設定後、認証はいつでも開始できます。通常はホット プラグや解像度変更などの外部イベントに基づいて開始します。どのようなイベントで認証要求をトリガーするかは、個々のアプリケーションで決定します。
 - XV_HdmiTxSs_HdcpAuthRequest
8. 認証のステータスをチェックします。このチェックは、暗号化を有効にする前にも実行可能です。
 - XV_HdmiTxSs_HdcpIsAuthenticated
9. 認証に成功すると、アプリケーションで暗号化を有効にできます。暗号化を有効にするのは認証の成功後であればよく、アプリケーション側で決定できます。たとえば、標準コンテンツでは暗号化を無効にし、制限付きコンテンツのみ暗号化を有効にするようにアプリケーションで決めることもできます。
 - XV_HdmiTxSs_HdcpEnableEncryption
 - XV_HdmiTxSs_HdcpDisableEncryption
10. 暗号化ステータスをチェックします。これはその瞬間の暗号化ステータスであり、次のフレームでは変化することがあります。
 - XV_HdmiTxSs_HdcpIsEncrypted
11. 全体的な HDCP プロトコル ステータスとログ データをチェックします。報告されるログ情報の詳細レベルも設定できます。
 - XV_HdmiTxSs_HdcpInfo
 - XV_HdmiTxSs_SetInfoDetail

HDMI TX Subsystem で使用する Video PHY Controller ドライバーの統合

HDMI 1.4/2.0 Transmitter Subsystem は Video PHY Controller と密に結合して使用するため、Video PHY Controller をユーザー アプリケーションで使用方法をサンプル コードで示します。

```

2039     XVphy Vphy;                /* VPHY structure */
2040     XVphy_Config *XVphyCfgPtr;
2041     // Initialize Video PHY
2042     XVphyCfgPtr = XVphy_LookupConfig(XPAR_VID_PHY_CONTROLLER_0_DEVICE_ID);
2043     if (XVphyCfgPtr == NULL) {
2044         print("Video PHY device not found\n\r");
2045         return XST_FAILURE;
2046     }
2047
2048     /* Initialize HDMI VPHY */
2049     Status = XVphy_HdmiInitialize(&Vphy, 0,
2050                                   XVphyCfgPtr, XPAR_CPU_CORE_CLOCK_FREQ_HZ);
2051     if (Status != XST_SUCCESS) {
2052         print("HDMI VPHY initialization error\n\r");
2053         return XST_FAILURE;
2054     }
2055
2056     /* Register VPHY Interrupt Handler */
2057     Status = XIntc_Connect(&Intc,
2058                            XPAR_MICROBLAZE_SS_AXI_INTC_0_VID_PHY_CONTROLLER_0_IRQ_INTR,
2059                            (XInterruptHandler)XVphy_InterruptHandler,
2060                            (void *)&Vphy);
2061
2062     if (Status != XST_SUCCESS) {
2063         print("HDMI VPHY Interrupt Vec ID not found!\n\r");
2064         return XST_FAILURE;
2065     }
2066
2067     /* Enable VPHY Interrupt */
2068     XIntc_Enable(&Intc,
2069                  XPAR_MICROBLAZE_SS_AXI_INTC_0_VID_PHY_CONTROLLER_0_IRQ_INTR);
    
```

図 C-4: アプリケーション サンプル コード

アプリケーション コードに HDMI 1.4/2.0 Transmitter Subsystem 用の Video PHY Controller を統合して使用する場合は、次の手順に従います。

1. サブシステム オブジェクトを定義したサブシステム ヘッダー ファイル xvphy.h をインクルードします。
2. アプリケーション コードで Video PHY Controller のインスタンスを宣言し、空間を割り当てます。

例:

```
XVphy Vphy;
```

3. Video PHY Controller インスタンスには、コントローラーのハードウェア構成を格納するメタデータ構造があります。アプリケーション コードでそのインスタンスを指し示すポインター変数を宣言します。

```
XVphy_Config *XVphyCfgPtr;
```

4. Video PHY Controller インスタンスについて、上記のデータ構造体をそれぞれのハードウェア構成に基づいて初期化する必要があります。ハードウェア構成は `xparameters.h` からメタ構造を通じて渡され、固有のデバイス ID で識別されます。

Video PHY Controller を初期化するには、次の 2 つの API 関数を呼び出します。

```
XVphy_Config *XVphy_LookupConfig(u16 DeviceId);
u32 XVphy_HdmiInitialize(XVphy *InstancePtr,
    u8 QuadId,
    XVphy_Config *CfgPtr,
    u32 SystemFrequency);
```

デバイス ID は `xparameters.h` に記述してあります。

```
XPAR_[Video PHY Controller Instance Name in IPI]_DEVICE_ID
```

同様に、`SystemFrequency` (システム周波数) も `xparameters.h` に記述してあります。

注記:

- Video PHY Controller の初期化は、HDMI 1.4/2.0 Transmitter Subsystem の初期化が完了した後に実行することを推奨します。
- システム アプリケーションへの統合には、Video PHY Controller の割り込みを登録する手順も必要です。この手順は前のセクションで示したものと同じであるため、ここでは省略します。

割り込み

ここでは、HDMI 1.4/2.0 Transmitter Subsystem で生成されるすべての割り込みを示します。

1. **HPD** – HDMI ケーブルの 5.0V 信号を検出するペリフェラル I/O です。
 - a. 立ち上がりエッジ – ケーブル接続
 - b. 立ち下がりエッジ – ケーブル切断
2. **Link Ready** – Video PHY Controller をリコンフィギュレーションするたびに `link_clk` が再生成されます。`link_clk` ステータスの変化は、HDMI TX サブコアのレジスタ ビット (リンク ステータス ビット) に反映されます。`link_clk` の安定を検出すると、1 にセットされます。`link_clk` が不安定になると、0 にクリアされます。Link Ready はリンク ステータス ビットの変化を検出するための割り込みです。
 - a. 立ち上がりエッジ – リンク アップ
 - b. 立ち下がりエッジ – リンク ダウン
3. **Vertical Sync** – ビデオ インターフェイス バスで HDMI TX サブコアの `vsync` 入力信号が変化したことを知らせます。
 - a. 立ち上がりエッジ – Vertical Sync を検出した
4. HDCP 1.4 割り込み (ハードウェアで HDCP 1.4 を有効にした場合のみ利用可能)
5. HDCP 1.4 タイマー割り込み (ハードウェアで HDCP 1.4 を有効にした場合のみ利用可能)
6. HDCP 2.2 タイマー割り込み (ハードウェアで HDCP 2.2 を有効にした場合のみ利用可能)

表 C-1: 割り込みソースとアプリケーション コールバック関数のマップ

割り込み	コールバック
HPD	XV_HDMITXSS_HANDLER_CONNECT
Link Ready 注記: エッジトリガーです。	XV_HDMITXSS_HANDLER_STREAM_UP XV_HDMITXSS_HANDLER_STREAM_DOWN 注記: 2つのコールバックが同じ割り込みソースにマップされます。 Link Ready 立ち上がりエッジ: ストリーム アップ Link Ready 立ち下がりエッジ: ストリーム ダウン
Vertical Sync	XV_HDMITXSS_HANDLER_VS
HDCP 1.4 割り込み	
HDCP 1.4 タイマー割り込み	
HDCP 2.2 タイマー割り込み	
	XV_HDMITXSS_HANDLER_HDCP_AUTHENTICATE 注記: このコールバック関数はどの割り込みソースにも直接マップされません。 このコールバックは、HDCP 認証ステート マシンが認証済みステートに遷移すると実行されます。

アプリケーション コールバック関数

サブシステム ドライバーには、割り込みコンテキスト内で呼び出されるユーザー定義関数をアプリケーションから登録するためのメカニズムがあります。

アプリケーション コードで定義したコールバック関数は、次の定義済み API を使用して所定のハンドラーに登録する必要があります。

```
int XV_HdmiTxSs_SetCallback(XV_HdmiTxSs *InstancePtr,
                           u32 HandlerType,
                           void *CallbackFuncPtr,
                           void *CallbackRef);
```

利用可能なハンドラーは xv_hdmitxss.h に定義されています。

- [XV_HDMITXSS_HANDLER_CONNECT](#)
- [XV_HDMITXSS_HANDLER_VS](#)
- [XV_HDMITXSS_HANDLER_STREAM_UP](#)
- [XV_HDMITXSS_HANDLER_STREAM_DOWN](#)
- [XV_HDMITXSS_HANDLER_HDCP_AUTHENTICATE](#)

XV_HDMITXSS_HANDLER_CONNECT

この割り込みは、HDMI TX ケーブルの接続または切断イベント (HPD レベルの遷移) が発生するたびにトリガーされます。

このコールバック関数は、次を実行する必要があります。

1. イベントがケーブル接続かケーブル切断かをチェックします。

```
XV_HdmiTxSs *HdmiTxSsPtr = (XV_HdmiTxSs *)CallbackRef;
HdmiTxSsPtr->IsStreamConnected
1 - Connected
0 - Disconnected
```

2. ケーブル接続の場合は差動入力クロック バッファを有効にし、ケーブル切断の場合は無効にします。

```
void XVphy_IBufDsEnable (XVphy *InstancePtr,
u8 QuadId,
XVphy_DirectionType Dir,
u8 Enable);
```

3. 接続された HDMI シンクが HDMI 2.0 に対応しているか、およびケーブルが接続されているかを検出します。

```
int XV_HdmiTxSs_DetectHdmi20 (XV_HdmiTxSs *InstancePtr);
```

4. これで HDMI シンクの検出は完了です。シンクの EDID 情報を取得し、次の API を使用してローカル バッファ (256 バイト) に格納します。

```
int XV_HdmiTxSs_ReadEdid (XV_HdmiTxSs *InstancePtr,
u8 *Buffer);
```

XV_HDMITXSS_HANDLER_VS

この割り込みは、入力ビデオ ストリームの Vertical Sync が HDMI TX サブコアによって検出されるたびにトリガーされます。

コールバック関数を使用して InfoFrame を構築し、シンクへ送信できます。

```
void XV_HdmiTxSs_SendAuxInfoframe (XV_HdmiTxSs *InstancePtr,
void *Aux);
```

XV_HDMITXSS_HANDLER_STREAM_UP

この割り込みは、Video PHY Controller がリコンフィギュレーションされた後、出力クロックが安定して HDMI 1.4/2.0 Transmitter Subsystem がビデオ ストリームを送信できるようになるたびにトリガーされます。

このコールバック関数は、次を実行する必要があります。

1. システムで HDMI リタイマーまたはイコライザーを使用している場合、必要なライン レートに基づいてリタイマーを正しく設定します。

2. Video PHY Controller API を呼び出して TX TMDS クロックを有効にします。

```
void XVphy_Clkout10BufTdsEnable (XVphy *InstancePtr,
XVphy_DirectionType Dir,
u8 Enable);
```

3. HDMI 1.4/2.0 Transmitter Subsystem のサンプリング レートに Video PHY Controller の TX サンプリング レートを設定します。

```
void XV_HdmiTxSs_SetSamplingRate (XV_HdmiTxSs *InstancePtr,
u8 SamplingRate);
```

XV_HDMITXSS_HANDLER_STREAM_DOWN

この割り込みは、Video PHY Controller がリコンフィギュレーションされた後、出力クロックが安定せず HDMI 1.4/2.0 Transmitter Subsystem がビデオ ストリームを送信できなくなるたびにトリガーされます。

コールバック関数で Video PHY Controller API を呼び出して TX TMD5 クロックを無効にできます。

```
void XVphy_Clkout10BufTdsEnable(XVphy *InstancePtr,
                                XVphy_DirectionType Dir,
                                u8 Enable);
```

XV_HDMITXSS_HANDLER_HDCP_AUTHENTICATE

この割り込みは、ケーブルが接続された後、HDCP 1.4 または HDCP 2.2 が有効にされ、HDCP が認証済みステートに遷移するとトリガーされます。

このコールバック関数は、次を実行する必要があります。

1. HDCP 暗号化を有効にします。
2. 認証が正しく完了したことをシステムに通知します。

HDMI 1.4/2.0 Transmitter Subsystem に対する Video PHY Controller 割り込みハンドラー

Video PHY Controller ドライバーにはいくつかの割り込みハンドラーがあり、HDMI 1.4/2.0 Transmitter Subsystem の機能をサポートするユーザー定義のコールバック関数をフックできます。これらの割り込みハンドラーは xvphy.h に定義されています。

- XVPHY_HDMI_HANDLER_TXINIT
- XVPHY_HDMI_HANDLER_TXREADY

コールバック関数をアプリケーション コードで定義し、これらの割り込みハンドラーにフックする必要があります。

```
void XVphy_SetHdmiCallback(XVphy *InstancePtr,
                            XVphy_HdmiHandlerType HandlerType,
                            void *CallbackFunc,
                            void *CallbackRef);
```

XVPHY_HDMI_HANDLER_TXINIT

この割り込みは、Video PHY Controller が HDMI TX 基準クロックの変化を検出するたびにトリガーされます。

HDMI 1.4/2.0 Transmitter Subsystem に対する基準クロック変更プロセスをコールバック関数で初期化する必要があります。

```
void XV_HdmiTxSs_RefClockChangeInit(XV_HdmiTxSs *InstancePtr);
```

XVPHY_HDMI_HANDLER_TXREADY

この割り込みは、Video PHY Controller TX のリセット ロックが完了または Video PHY Controller TX のアライメントが完了するたびにトリガーされます。

コールバック関数は、アプリケーション ソフトウェアに対する Video PHY TX Ready 情報を更新できます。

サンプル デザインの作成方法は、第 5 章「サンプル デザイン」の手順に従ってください。このサンプル デザインには、実装されたすべてのプロシージャが含まれており、HDMI 1.4/2.0 Transmitter Subsystem を実際のシステムに統合する際のリファレンスとして使用できます。

ユース ケースの例

このセクションでは、代表的なユース ケースを取り上げ、動作中のシステムが特定のイベントに対してどのように応答するか、そしてユーザーが何を実行すべきかを示します。コールバック関数で実行される動作の詳細は、「[アプリケーション コールバック関数](#)」を参照してください。

ユース ケース 1: ケーブルの接続

ケーブル接続を示す HPD 割り込みが受信されます。

- 「[XV_HDMITXSS_HANDLER_CONNECT](#)」の割り込みタイプに登録されたコールバック関数が呼び出されます。

ユース ケース 2: ケーブルの切断

ケーブル切断を示す HPD 割り込みが受信されます。

- 「[XV_HDMITXSS_HANDLER_CONNECT](#)」の割り込みタイプに登録されたコールバック関数が呼び出されます。

ユース ケース 3: InfoFrame の送信

Vertical Sync (VS) 割り込みが受信されます。

- 「[XV_HDMITXSS_HANDLER_VS](#)」の割り込みタイプに登録されたコールバック関数が呼び出されます。

ユース ケース 4: ビデオ ストリームの送信

1. API を利用して HDMI 1.4/2.0 Transmitter Subsystem に対する Video PHY Controller の TDMS クロックを無効にします。

```
XVphy_Clkout10BufTdsEnable(XVphy *InstancePtr,
                            XVphy_DirectionType Dir,
                            u8 Enable);
```

例:

```
XVphy_Clkout10BufTdsEnable(VphyPtr,
                            XVPHY_DIR_TX,
                            (FALSE));
```

2. API を利用して HDMI 1.4/2.0 Transmitter Subsystem のストリーム パラメーターを設定します。

```
u32 XV_HdmiTxSs_SetStream(XV_HdmiTxSs *InstancePtr,
                           XVidC_VideoMode VideoMode,
                           XVidC_ColorFormat ColorFormat,
                           XVidC_ColorDepth Bpc,
                           XVidC_3DInfo *Info3D);
```

例:

```
TmdsClock = XV_HdmiTxSs_SetStream(HdmiTxSsPtr,
                                    VideoMode,
                                    ColorFormat,
                                    Bpc,
                                    NULL);
```

3. Video PHY Controller TX の基準クロックを設定します。

```
VphyPtr->HdmiTxRefClkHz = TmdsClock;
```

- Video PHY Controller の HDMI TX パラメーターを設定します。

```
u32 XVphy_SetHdmiTxParam(XVphy *InstancePtr,
                          u8 QuadId,
                          XVphy_ChannelId ChId,
                          XVidC_PixelsPerClock Ppc,
                          XVidC_ColorDepth Bpc,
                          XVidC_ColorFormat ColorFormat);
```

例:

```
Result = XVphy_SetHdmiTxParam(VphyPtr,
                              0,
                              XVPHY_CHANNEL_ID_CHA,
                              HdmiTxSsVidStreamPtr->PixPerClk,
                              HdmiTxSsVidStreamPtr->ColorDepth,
                              HdmiTxSsVidStreamPtr->ColorFormatId);
```

- Video PHY Controller に TMD5 基準クロックを供給するように外部クロック ジェネレーターをプログラムします。
- Video PHY Controller の HDMI TX Init 割り込みが受信されます。
 - 「XVPHY_HDMI_HANDLER_TXINIT」の割り込みタイプに登録されたコールバック関数が呼び出されます。
- Video PHY Controller の HDMI TX Ready 割り込みが受信されます。
 - 「XVPHY_HDMI_HANDLER_TXREADY」の割り込みタイプに登録されたコールバック関数が呼び出されます。
- HDMI TX Stream UP 割り込みが受信されます。
 - 「XV_HDMITXSS_HANDLER_STREAM_UP」の割り込みタイプに登録されたコールバック関数が呼び出されます。

ユースケース 5: マルチ チャネル オーディオをサポートする

定義: N = オーディオ チャネル数

- API でチャンネル数を設定してオーディオ InfoFrame を変更します。

```
void XV_HdmiTxSs_SendAuxInfoframe(XV_HdmiTxSs *InstancePtr, void *AuxPtr);

/* 2 Channel count. Audio coding type refer to stream */
InstancePtr->HdmiTxPtr->Aux.Data.Byte[1] = N - 1;
```

- 次の API を使用して HDMI TX SS オーディオ チャネル数を設定します。

```
void XV_HdmiTxSs_SetAudioChannels(XV_HdmiTxSs *InstancePtr, u8 AudioChannels);
```

例:

```
XV_HdmiTxSs_SetAudioChannels(&HdmiTxSs, N);
```

- サンプル デザインのアプリケーション ソフトウェアの場合、xhdmi_example.c コードの次の部分を書き換えます。

```
/* Enable 2-channel audio */
XhdmiAudGen_SetEnabChannels(&AudioGen, 2);
XhdmiAudGen_SetPattern(&AudioGen, 1, XAUD_PAT_PING);
XhdmiAudGen_SetPattern(&AudioGen, 2, XAUD_PAT_PING);
```

例: 8 チャネル オーディオをサポートする場合

```
/* Enable 8-channel audio */
XhdmiAudGen_SetEnabChannels(&AudioGen, 8);
XhdmiAudGen_SetPattern(&AudioGen, 1, XAUD_PAT_PING);
XhdmiAudGen_SetPattern(&AudioGen, 2, XAUD_PAT_PING);
XhdmiAudGen_SetPattern(&AudioGen, 3, XAUD_PAT_PING);
XhdmiAudGen_SetPattern(&AudioGen, 4, XAUD_PAT_PING);
```

```
XhdmiAudGen_SetPattern(&AudioGen, 5, XAUD_PAT_PING);  
XhdmiAudGen_SetPattern(&AudioGen, 6, XAUD_PAT_PING);  
XhdmiAudGen_SetPattern(&AudioGen, 7, XAUD_PAT_PING);  
XhdmiAudGen_SetPattern(&AudioGen, 8, XAUD_PAT_PING);
```

次に、オーディオチャンネルの割り当てを更新します。

詳細は、CED-861-D の Table 20 「Audio InfoFrame Data Byte 4」を参照してください。

次の API を使用します。

```
void XV_HdmiTxSs_SendAuxInfoframe(XV_HdmiTxSs *InstancePtr, void *AuxPtr);
```

CRC を計算する前にデータバイトの値を設定する必要があります。

例

```
/* Channel Allocation */  
InstancePtr->HdmiTxPtr->Aux.Data.Byte[4] = 0x13;
```

アプリケーションソフトウェアで独自の InfoFrame を構築することも、API `XV_HdmiTxSs_SendGenericAuxInfoframe` を使用して送信することもできます。

ユースケース 6: HDMI モードを有効にする

次の API を使用します。

```
XV_HdmiTxSs_SetHdmiMode(&HdmiTxSs);  
XV_HdmiTxSs_AudioMute(&HdmiTxSs, FALSE);
```

ユースケース 7: DVI モードを有効にする

次の API を使用します。

```
XV_HdmiTxSs_SetDviMode(&HdmiTxSs);  
XV_HdmiTxSs_AudioMute(&HdmiTxSs, TRUE);
```

その他のリソースおよび法的通知

ザイリンクス リソース

アンサー、資料、ダウンロード、フォーラムなどのサポート リソースは、[ザイリンクス サポート サイト](#)を参照してください。

参考資料

次の資料は、この製品ガイドの補足資料として役立ちます。

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. 『Vivado Design Suite: AXI リファレンス ガイド』(UG1037: [英語版](#)、[日本語版](#))
2. 『Kintex UltraScale FPGA データシート: DC 特性および AC スイッチ特性』(DS892: [英語版](#)、[日本語版](#))
3. 『Virtex UltraScale FPGA データシート: DC 特性および AC スイッチ特性』(DS893: [英語版](#)、[日本語版](#))
4. 『Kintex-7 FPGA データシート: DC 特性および AC スイッチ特性』(DS182: [英語版](#)、[日本語版](#))
5. 『Virtex-7 T/XT FPGA データシート: DC 特性および AC スイッチ特性』(DS183: [英語版](#)、[日本語版](#))
6. 『Artix-7 FPGA データシート: DC 特性および AC スイッチ特性』(DS181: [英語版](#)、[日本語版](#))
7. 『Kintex UltraScale+ FPGA データシート: DC 特性および AC スイッチ特性』(DS922: [英語版](#)、[日本語版](#))
8. 『Virtex UltraScale+ FPGA データシート: DC 特性および AC スイッチ特性』(DS923: [英語版](#)、[日本語版](#))
9. 『Zynq UltraScale+ MPSoC データシート: DC 特性および AC スイッチ特性』(DS925: [英語版](#)、[日本語版](#))
10. HDMI 仕様 (www.hdmi.org/manufacturer/specification.aspx)
11. HDCP 仕様 (www.digital-cp.com/hdcp-specifications)
12. 『AXI4-Stream Video IP およびシステム デザイン ガイド』([UG934](#))
13. 『Vivado Design Suite ユーザー ガイド: IP インテグレーターを使用した IP サブシステムの設計』(UG994: [英語版](#)、[日本語版](#))
14. 『Vivado Design Suite ユーザー ガイド: IP を使用した設計』(UG896: [英語版](#)、[日本語版](#))
15. 『Vivado Design Suite ユーザー ガイド: 入門』(UG910: [英語版](#)、[日本語版](#))
16. 『Vivado Design Suite ユーザー ガイド: ロジック シミュレーション』(UG900: [英語版](#)、[日本語版](#))
17. 『ISE から Vivado Design Suite への移行ガイド』(UG911: [英語版](#)、[日本語版](#))
18. 『KCU105 ボード ユーザー ガイド』([UG917](#))
19. 『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](#))
20. 『Vivado Design Suite ユーザー ガイド: インプリメンテーション』(UG904: [英語版](#)、[日本語版](#))
21. 『AXI Interconnect LogiCORE IP 製品ガイド』([PG059](#))

22. 『Video PHY Controller LogiCORE IP 製品ガイド』(PG230)
23. 『HDCP v2.2 製品ガイド』(PG249)
24. 『HDCP v1.4 製品ガイド』(PG224)
25. 『AXI4-Stream to Video Out LogiCORE IP 製品ガイド』(PG044)
26. ANSI/CTA 標準 (https://standards.cta.tech/kwspub/published_docs/ANSI-CTA-861-F-Preview.pdf)

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4月5日	2.0	<ul style="list-style-type: none"> • 1クロックあたり1ピクセル(1PPC)のサポートを削除。
2016年11月30日	2.0	<ul style="list-style-type: none"> • サンプル デザインの移行に関する注意を追加。
2016年10月05日	2.0	<ul style="list-style-type: none"> • サンプル デザイン フローを追加。 • HPD の XGUI オプションを追加。 • ソフトウェアのユース ケースを追加。 ザイリンクスの「自動車用のアプリケーションの免責条項」を更新。
2016年6月8日	2.0	<ul style="list-style-type: none"> • Video over AXI-Stream のサポート (オプション) を更新。
2016年4月6日	2.0	<ul style="list-style-type: none"> • 「IP の概要」に「機能」のセクションを追加。 • 「概要」の章の「サポートされていない機能」を更新。 • 「製品仕様」の章を更新。 • 「サブシステムを使用するデザイン」の章を更新。 • 「デザイン フローの手順」の章を更新。 • 「ハードウェア テスト」と「ビデオ解像度」のセクションを更新。 • 付録「アプリケーション ソフトウェア開発」を更新。
2015年11月18日	1.0	初版

お読みください: 重要な法的通知

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となります。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はリスクと責任は、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。

© Copyright 2015-2017 Xilinx, Inc. Xilinx, Xilinx のロゴ、Artix、ISE、Kintex、Spartan、Virtex、Vivado、Zynq、およびこの文書に含まれるその他の指定されたブランドは、米国およびその他の各国のザイリンクス社の商標です。すべてのその他の商標は、それぞれの所有者に帰属します。

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。フィードバックは日本語で入力可能です。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。