MIPI CSI-2 イメージセンサーを EZ-USB™ CX3 にインターフェースする方法

本書について

適用範囲と目的

本アプリケーションノートでは、MIPI CSI-2 イメージセンサーを EZ-USB™ CX3 に接続するための実装の詳細について説明します。

対象者

本書は、USB 3.2 Gen 1 アプリケーションを使用するすべての方を対象とします。具体的には、EZ-USB™ CX3 と接続された MIPI CSI-2 イメージセンサーを搭載したカメラが、非圧縮データを PC にストリーミングするケースです。

キーポイント

- EZ-USB™ CX3 による MIPI CSI-2 センサーの統合
- USB ビデオクラス (UVC) の実装
- ファームウェア, ハードウェア, およびツールのサポート

本製品ファミリーについて

製品ファミリー

EZ-USB™は、産業オートメーション, 医療用画像処理, マシンビジョン, および民生機器を含む、幅広い組込みシステムに USB 3.2 Gen 1 (5 Gbps) 接続機能を追加するために設計された、高性能 USB 周辺機器コントローラーファミリーです。

対象アプリケーション

- ヘルスケア
- 産業オートメーション
- マシンビジョン
- ロボティクス
- Industry 4.0 向け産業用ロボットシステムソリューション
- 存在検知

目次

本書について 1
本製品ファミリーについて 1
目次 2
1 はじめに 4
2 イメージ センサーを CX3 へインターフェースする 7
2.1 MIPI CSI-2 インターフェース 7
2.2 GPIF II ブロック 8
2.2.1 DMA の基本 9
2.2.2 2 個のソケットが使用される理由 9
2.2.3 GPIF II ステートマシン 10
3 DMA システムのセットアップ 12
3.1 DMA バッファ 18
4 USB ビデオクラス (UVC) 20
4.1 エニュメレーションデータ 20
4.2 動作コード 20
4.3 UVC の要件 21
4.3.1 UVC の USB ディスクリプタ 21
4.3.1.1 ビデオ制御 (VC) インターフェース 22
4.3.1.2 ビデオストリーミング (VS) インターフェース 22
4.3.2 UVC 固有の要求 23
4.3.2.1 制御要求: 輝度, PTZ, 色相, 彩度, その他 28
4.3.2.2 ストリーミング要求: プローブとコミット制御 28
4.3.2.3 ビデオデータフォーマット: YUY2 28
4.3.2.4 UVC ビデオデータヘッダ 29
5 CX3 MIPI 設定ツール 31
5.1 ツールの起動 31
5.2 CX3 Configuration tool へのセンサー詳細の追加 33
5.2.1 パラレルデータ幅の選択に基づく、入力データ形式から出力ビデオ形式へのマッピング 36
5.3 CX3 MIPI レシーバーの設定 39
6 CX3 ファームウェア 42
6.1 アプリケーションスレッド 44
6.1.1 USB サスペンドイベント 44
6.2 DMA チャネルリセットイベント 44
6.3 UVC 要求の処理 45
6.4 初期化 45
6.5 エニュメレーション 45
6.6 MIPI CSI-2 コントローラーの設定 45
6.7 イメージセンサーの設定 46
6.8 ビデオストリーミングの開始 46
6.9 フレーム設定の選択と切り替え 46
6.10 DMA バッファの設定 47
6.11 ビデオストリーミング中の DMA バッファの処理 47
6.12 フレーム終端でのストリーミングの再開 48
6.13 ビデオストリーミングの終了 48
7 ハードウェアのセットアップ 49

目次

7.1 CX3 RDK を使うテスト 49
7.1.1 キットの調達 49
7.2 独自基板の設計 49
8 UVC ベースのホストアプリケーション 50
9 トラブルシューティング 51
10 まとめ 53
11 関連リソース 54
改訂履歴 55
免責事項 56

1 はじめに

USB 3.2 Gen1 が提供した広帯域幅は、周辺装置を USB に接続する IC に対する高度な要求をします。一般的な例としては、非圧縮データを PC にストリーミングするカメラがあります。USB 3.2 Gen1 では USB 帯域幅が増加したため、PC に接続するカメラの解像度も高くなりました。広帯域幅のカメラインターフェース要件に対応するために携帯機器プロセッサ インターフェース (MIPI) と呼ばれる協会がカメラシリアルインターフェース 2 (CSI-2) という仕様を策定しました。

本アプリケーションノートは、このために特別に設計された EZ-USB™ FX3 の応用品種であるインフィニオン EZ-USB™ CX3 を使用する MIPI CSI-2－USB 3.2 Gen 1 コンバータの実装について詳しく説明します。

インフィニオン EZ-USB™ 製品ファミリーの初心者である場合、詳細については「AN75705 – EZ-USB™ FX3 入門」アプリケーションノートを参照してください。

CX3 は、MIPI CSI-2 インターフェースに対応したイメージセンサーに対し、USB Video Class (UVC) を介した SuperSpeed USB 接続機能を追加できます。UVC に準拠することで、OS 標準のドライバを使用してカメラを動作させることができ、e-CAMView, Webcamoid, Media Player Classic, VLC Media Player といったホストアプリケーションとの互換性が確保されます。

CX3 は FX3 からの派生品であり、FX3 との相違点は次のとおりです。

- MIPI CSI-2 レシーバーインターフェースを備えた MIPI CSI-2 コントローラーが追加されます。
- USB 2.0 OTG と充電器検出機能は取り除かれます。
- 2 つのクロックリファレンス (コア用に CLKIN、MIPI CSI-2 コントローラー用に REFCLK) が必要です。
- CLKIN については 19.2 MHz の発振器のみがサポートされます。

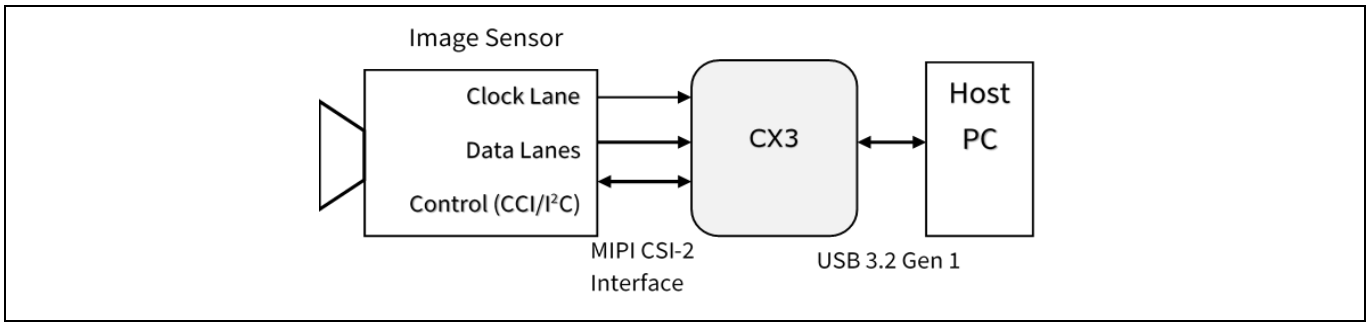

Figure 1　カメラアプリケーション

CX3 は FX3 の特殊バージョンであるため、すべての FX3 開発ツールとほとんどの FX3 リファレンスが CX3 に適用されます。この互換性の例としては、FX3 を使用するカメラ設計例 (パラレルインターフェース) について記載した AN75779 があります。双方のチップのデータ転送アーキテクチャが同一であるため、CX3 の内部動作についての参考資料の多くは、FX3 のアプリケーションノートから持ってきています。

本アプリケーションノートでは、利用可能なインフィニオンのファームウェアプロジェクトの理解に役立つように詳しく説明します。お客様の設計がこのアプリケーションノートと同じイメージセンサーを使用している場合は、コード変更はほとんど、あるいはまったく必要ありません。異なるイメージセンサーの場合は、若干の変更が必要なコードモジュールを注記で示します。

Figure 1 にカメラアプリケーションを示します。右側は SuperSpeed USB 3.2 Gen 1 ポートを搭載した PC です。左側は次のことに対応できる MIPI CSI-2 インターフェースを備えたイメージセンサーです。

- 1～4 本の MIPI CSI-2 データ レーン

1 はじめに

- RAW8/10/12/14, YUV422 (CCIR/ITU 8/10 ビット), RGB888/666/565, M-JPEG のような圧縮されたフォーマットとユーザー定義の 8 ビット, 16 ビット, および 24 ビットの画像フォーマット

例えば、OV5640 センサーは VGA 60 fps, HD (720p) 60 fps, フル HD (1080p) 30 fps, 5 M-Pixel 15 fps をサポートします。以下のセクションでは、cycx3_uvc_ov5640 サンプルプロジェクト (FX3/CX3 SDK インストールで利用可能) を、VGA, HD, およびフル HD 解像度で YUV 形式をストリーミングするように設計する方法について詳しく説明します。

Figure 2 はサブブロックに番号を付けたブロックダイヤグラムです。各サブブロックで実行されるタスクを説明します。

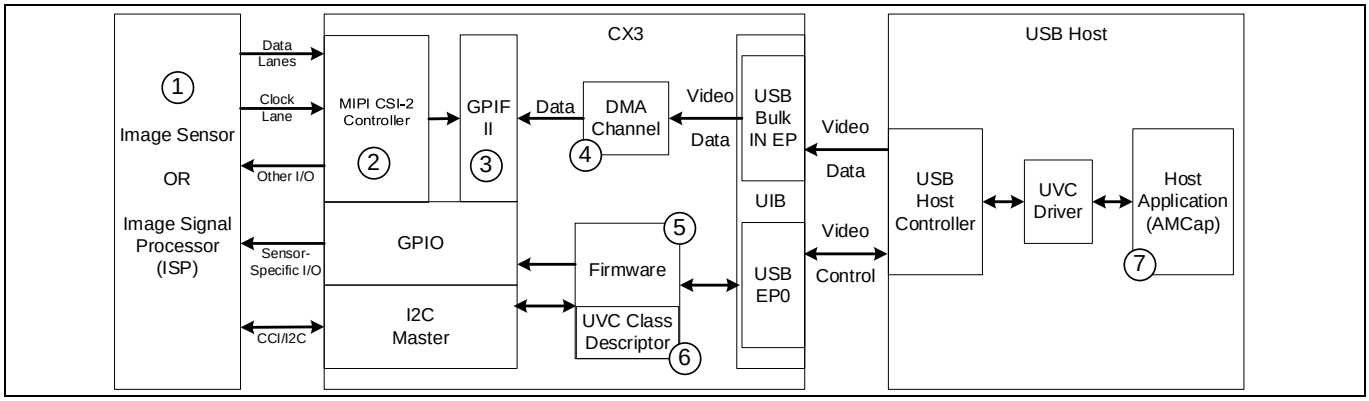

Figure 2　システムブロックダイヤグラム

1. MIPI CSI-2 に基づいたイメージセンサーを CX3 に接続し、カメラ制御インターフェース (CCI) バスを使用してイメージセンサーを設定します。
2. イメージセンサーから画像データを読み出し、パケットを取り出し、GPIF II ブロックに送信するよう、CX3 の MIPI CSI-2 コントローラーを設定します。MIPI CSI-2 インターフェースを参照してください。
3. 画像データフォーマットに従って GPIF II ブロックを設定します。GPIF II ブロックを参照してください。
4. 画像データを GPIF II ブロックから USB インターフェース ブロック (UIB) に転送する DMA チャネルを構築します。このアプリケーションでは、UVC 仕様に準拠するためにヘッダデータをイメージセンサーのビデオデータに追加する必要があります。このために、DMA は CPU が DMA バッファに必要なヘッダを追加できるように構成されています。このチャネルは、ビデオをイメージセンサーから PC にストリーミングするために最大帯域幅を使用できるように設計する必要があります。DMA システムのセットアップを参照してください。
5. CX3 のファームウェアは、CX3 のハードウェアブロックを初期化し (初期化を参照)、イメージセンサー (イメージセンサーの設定を参照) および MIPI CSI-2 コントローラー (MIPI CSI-2 コントローラーの設定を参照) を設定し、UVC カメラとしてデバイスをエニュメレートし (エニュメレーションを参照)、UVC 固有の要求を処理し (UVC 要求の処理を参照)、CCI (I2C) インターフェースを介してイメージセンサーにビデオ制御設定 (輝度など) を変換し、ビデオデータストリームに UVC ヘッダを追加し (ビデオストリーミング中の DMA バッファの処理を参照)、USB にヘッダ付きのビデオデータを転送します。
6. 適切な USB ディスクリプタを提供し、ホストが周辺装置を UVC に準拠したデバイスとして認識できるようにします。USB ビデオクラス (UVC) を参照してください。
7. UVC 制御インターフェースを介してイメージセンサーを設定し、UVC ストリーミングインターフェースを介してビデオデータを受信するために、UVC ドライバにアクセスするホストアプリケーション (e-CAMView, Media Player Classic, または VLC Media Player など) を使用します。UVC ベースのホストアプリケーションを参照してください。

1 はじめに

カメラが USB 2.0 ポートに差し込まれたら、CX3 のファームウェアは CCI バスを使用して低い USB 帯域幅に対応できるよう、フレーム レートとフレーム サイズを (利用可能な場合) 低減する設定を選びます。ホストはビデオ制御インターフェースを任意に使用して、輝度, コントラスト, 色相, 飽和, 露光, オートフォーカス, および PTZ (パン, チルト, ズーム) の調整をカメラに送信できます。

2 イメージ センサーを CX3 へインターフェースする

MIPI CSI-2 イメージ センサーを CX3 に接続し、そこからデータを読み出すには、CSI-2 インターフェースと CX3 の DMA 機能を理解する必要があります。

2.1 MIPI CSI-2 インターフェース

カメラアプリケーションが精密になるため、高解像度のイメージセンサーに対して高い要求が寄せられます。この要求は、拡大が困難で、多くの相互接続を必要とする一般的なパラレルイメージセンサーインターフェースの限界に迫っています。そこで、MIPI アライアンス (携帯機器産業プロセッサインターフェース協会) は、カメラシリアルインターフェース 2 (CSI-2) 規格を作成し、広範囲のイメージングソリューションをサポートする標準的, 強固, 低消費電力, 高速シリアル インターフェースを策定しました。

MIPI CSI-2 インターフェースは、データ信号とクロック信号を持つ単方向の差動シリアルインターフェースです。最大 4 本のデータレーンがあり、それぞれ最大 1 Gbps でデータを転送します。

イメージセンサーを設定するために使用される制御インターフェースは、I²C 規格に準拠し、カメラ制御インターフェース (CCI) と呼ばれています。

CX3 の MIPI CSI-2 コントローラーは、イメージセンサーが一般的に必要とするこれら 2 種類のインターフェースと複数の追加信号を提供します。

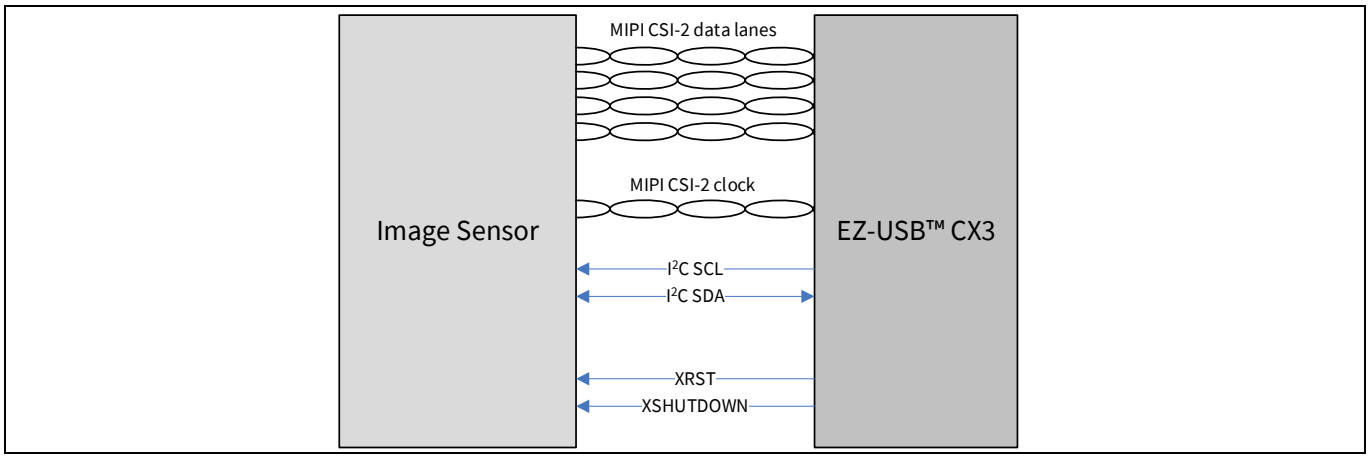

Figure 3 CX3 とイメージセンサーとのインターフェース

このインターフェースにある 3 種類の追加信号を次に示します。

XRST: イメージセンサーはコンフィギュレーション前に CX3 からのリセット信号を常に必要とします。このリセットは CX3 の XRST ピンを使用して行います。

XSHUTDOWN: ホストがイメージセンサーからビデオストリームを要求していない場合は、センサーはシャットダウンまたはスタンバイモードに入り消費電力を低減できます。これは、CX3 の XSHUTDOWN ピンを使用して制御できるセンサー内のシャットダウン端子を使用して行われます。

*Note: FX3 SDK の **CyU3PMipicsiSetSensorControl** API を使用して、センサーへの XRES および XSHUTDOWN 信号を駆動できます。*

MCLK: 外部クロック発振器により供給される、イメージセンサーに必要なマスター (またはリファレンス) クロックです。CX3 が供給する MCLK は、イメージセンサーのテスト用です。この信号は最終製品で

の使用には適していません。量産時には、センサーへのクロック入力として外部クロックジェネレーターを使用してください。

CSI-2, CCI, および 3 つの追加信号のピンマッピングについては、データシート「EZ-USB™ CX3 MIPI CSI-2 to USB 5 Gbps Camera Controller」を参照してください。また完全な例については、CX3 RDK 回路図を参照してください。

MIPI CSI-2 コントローラーは、1〜4 つのデータレーン, 様々なデータ形式 (RAW, YUV, または MJPEG など), および様々なカメラの解像度に対応するように設定できます。詳細については、CX3 ファームウェアを参照してください。このアプリケーションでは、4 つのデータレーンに YUV 画像データを送信するように設定されたセンサーを使用します。

設定された後、MIPI CSI-2 コントローラーは、イメージセンサーからシリアル画像データを受け取り、パケットを取り出し、さらにパラレルインターフェースを介して送信されるパラレルデータにそれを変換します。このインターフェースは次の信号を使用します。

- FV: フレーム有効 (フレームの開始と終了を示します)
- LV: ライン有効 (ラインの開始と終了を示します)
- PCLK: ピクセルクロック
- データ: 画像データ用の 8 ビット, 16 ビット, および 24 ビットのデータ バス

この信号のタイミングは AN75779 の「イメージセンサーインターフェース」セクションで説明するパラレル インターフェースと同じです。

CX3 で使用できる最大 GPIF II データ バス幅が 24 ビットで、サポートされる最大 PCLK が 100 MHz であるため、CX3 にサポートされる最大スループットは 2.4 Gbps です。

CX3 MIPI 設定ユーティリティにおける CSI クロック設定は DDR モード (クロックの立ち上りと立ち下りの両エッジでデータをサンプリングするモード) となります。以下の表は、各 MIPI レーン設定における最大サポート CSI クロックと、各設定でのスループットを示します。

MIPI データレーン数	最大許容 CSI クロック周波数	MIPI レーンあたりの最大データレート	対応する最大ビットレート
1	500 MHz	1 Gbps	1 Gbps
2	500 MHz	1 Gbps	2 Gbps
3	400 MHz	800 Mbps	2.4 Gbps
4	300 MHz	600 Mbps	2.4 Gbps

Note: ブランキング時間をゼロと仮定し、CSI クロックを用いてCX3 の最大サポートビットレート (CSI クロック× 2 × MIPI レーン数) を算出してください。

2.2 GPIF II ブロック

MIPI CSI-2 コントローラーからのパラレル出力インターフェースは、プロセッサインターフェースブロック (PIB) の一部である GPIF II ブロックに接続されます。

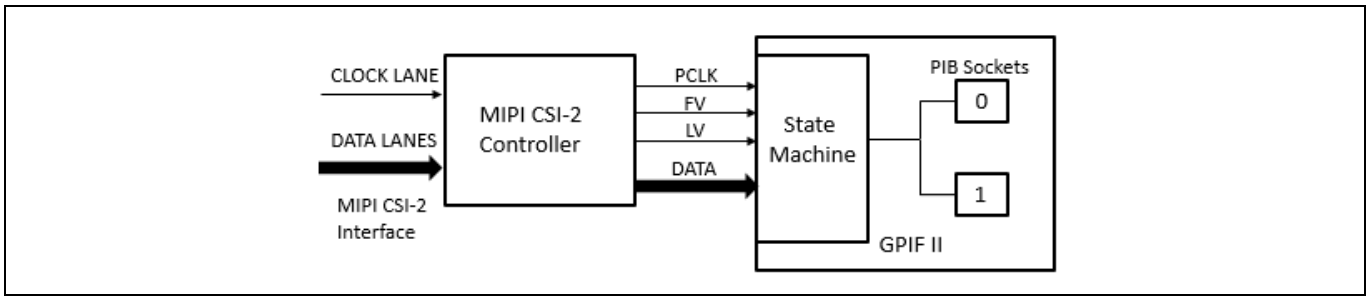

Figure 4　GPIF II ブロック

GPIF II ブロックは、ステートマシンを使用して、このインターフェースから 2 つの DMA ソケットへデータを読み込みます。データ転送の仕組みを理解するには、CX3 の DMA 機能についての知識が必要です。

2.2.1　DMA の基本

CX3 の DMA (ダイレクト メモリ アクセス) は、任意の 2 ブロック間で高速で中断しないデータ転送を可能にします。これらのデータ転送を理解するには次の用語を理解することが重要です。

- ソケット
- DMA ディスクリプタ
- DMA バッファ

ソケットは周辺装置のハードウェアブロックと CX3 RAM との接点です。USB, GPIF, UART, および SPI などの CX3 上の各周辺装置のハードウェアブロックには、それぞれに関連付けられた一定数のソケットがあります。各周辺装置を流れる個別のデータフロー数は、それに対応するソケット数に等しいです。ソケットの実装は、アクティブな DMA ディスクリプタを指すレジスター式とソケットに対応するイネーブルまたはフラグ割込みを含みます。

DMA ディスクリプタは CX3 RAM に割り当てられたレジスター式です。DMA バッファのアドレスとサイズの情報および次の DMA ディスクリプタへのポインタを格納します。これらのポインタは DMA ディスクリプタチェーンを形成します。

DMA バッファは RAM の一部であり、CX3 デバイスを介して転送されるデータの中間記憶装置として使用されます。DMA バッファは CX3 ファームウェアによって RAM から割り当てられ、それらのアドレスは DMA ディスクリプタの一部として格納されます。

2.2.2　2 個のソケットが使用される理由

2 個のソケットが使用される理由を理解するにあたり、ソケットの詳細を理解する必要があります。

ソケットはイベントにより互いに直接通知するか、または、割込みにより CX3 CPU に通知します。この信号方式はファームウェアにより設定されます。例えば、GPIF II ブロックから USB ブロックまでのデータストリームを考えます。GPIF ソケットは DMA バッファをデータで充填したことを USB ソケットに通知でき、USB ソケットは DMA バッファが空になったことを GPIF ソケットに通知できます。この実装は自動 DMA チャネルと呼ばれます。自動 DMA チャネルの実装は通常、CX3 CPU がデータストリームでデータの修正が必要ない場合に使用されます。

一方、GPIF ソケットは CX3 CPU に割込みを送信し、GPIF ソケットが DMA バッファを充填したことも通知できます。CX3 CPU はこの情報を USB ソケットに送ります。USB ソケットは CX3 CPU に割込みを送信し、USB ソケットが DMA バッファを空にしたことを通知できます。そして CX3 CPU はこの情報を GPIF ソケットに送り返します。この実装は*手動 DMA チャネル*と呼ばれます。

手動 DMA チャネルの実装は通常、CX3 CPU がデータ ストリームのデータを追加、除去あるいは修正する必要がある場合に使用されます。本アプリケーションノートのファームウェア例では、ファームウェアが UVC ビデオデータヘッダを追加する必要があるため、手動 DMA チャネルの実装を使用します。

DMA バッファにデータを書き込むソケットは、プロデューサソケットと呼ばれます。DMA バッファからデータを読み出すソケットは、コンシューマソケットと呼ばれます。ソケットはデータ管理のために DMA ディスクリプタに格納された DMA バッファアドレス, DMA バッファサイズ, および DMA ディスクリプタチェーンを使用します。

ソケットは、DMA バッファを満杯か空にした後、DMA ディスクリプタから他の DMA ディスクリプタに切り替えるのに限られた時間 (最大で数マイクロ秒) を要します。切り替えの間、ソケットはデータを転送できません。GPIF II ブロックでは 2 個の GPIF スレッドを使用して、このレイテンシを除去できます。

GPIF スレッドはソケットにデータピンを接続する GPIF II ブロックの専用データパスです。一度に 1 個の GPIF スレッドだけがデータを転送できます。

GPIF スレッド選択のメカニズムはマルチプレクサに似ています。アクティブな GPIF スレッドを切り替えるとデータ転送用のアクティブなソケットが切り替わります。結果として、データ転送用の DMA バッファが変更されます。この切り替えは 1 クロックサイクルのレイテンシがあるため、本質的には瞬間的に行われます。GPIF II ステートマシンはこの切り替えを DMA バッファの境界で行うため、新しい DMA ディスクリプタへの GPIF ソケット切り替えのレイテンシをマスクできます。これにより DMA バッファが満杯になった時、GPIF II ブロックはセンサーからデータを損失せずに取り込めます。

Figure 5 に、データフローに沿った本アプリケーションで使用するソケット, DMA ディスクリプタ, および DMA バッファ接続を示します。2 個の GPIF スレッドは、代替 DMA バッファを充填するために使用されます。これら GPIF スレッドは、個別の GPIF ソケット (プロデューサ ソケットとして動作) と DMA ディスクリプタチェーン (ディスクリプタチェーン 1 とディスクリプタチェーン 2) を使用します。USB ソケット (コンシューマソケットとして動作) は、正しい順序でデータを読み出すために 3 番目の DMA ディスクリプタチェーン (ディスクリプタチェーン 3) を使用します。ソケット, ソケット切り替え, および関連する遅延についての詳細は DMA システムのセットアップを参照してください。

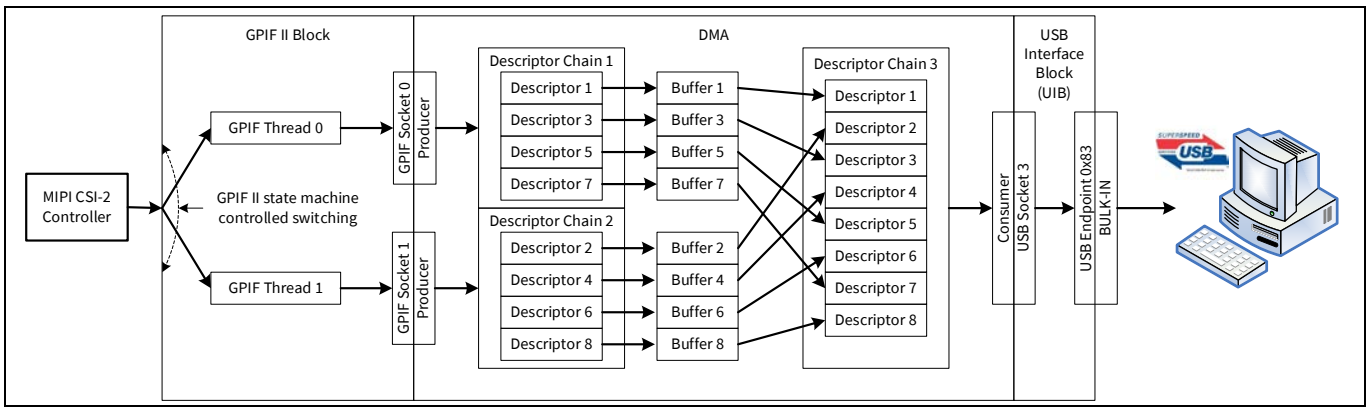

Figure 5 CX3 データの転送アーキテクチャ

2.2.3 GPIF II ステートマシン

GPIF II ブロックは内部でステートマシンを使用して、MIPI CSI-2 コントローラーのパラレル出力インターフェースからビデオデータを読み出します。

前セクション (2 個のソケットが使用される理由を参照) で説明したように、ステートマシンは DMA ディスクリプタ (および実際には、DMA バッファ) 間で切り替える時にデータの損失を防ぐためにビデオデータを 2 個のソケットにロードします。

これはソケットに読み出したデータ量を継続して追跡するためにカウンターを使用し、カウンターが限界値に達すると、他のソケットに切り替わることで行われます。カウンターの限界値は DMA バッファのサイズに設定されます。

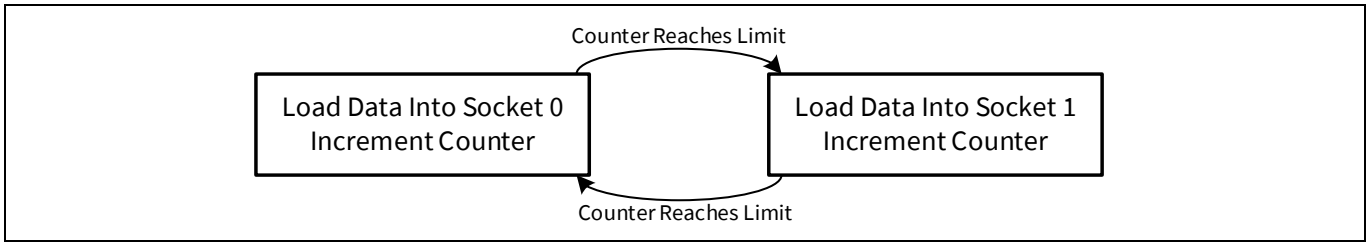

Figure 6　2 個のソケットへのデータ転送

カウンターは、クロックサイクルごとに 1 増加します。したがって、インターフェースのデータバス幅によってカウンター限界値が変わります。例えばデータバス幅が 16 ビット、DMA バッファサイズが 16KB (即ち、16,384) の場合、サイクルごとに 2 バイトの読み出しが行われるため、プログラムされる限界値は (16384/2) – 1 = 8191 になるはずです。

一般的に DMA バッファカウント限界値は次のとおりです。

$$count = \left(\frac{producer_buffer_size(L)}{data_bus_width\ (in\ bytes)} \right) - 1$$

バス幅について:

GPIF II データバス幅は、イメージセンサーのデータフォーマットに応じて選択される必要があります。このアプリケーションノートでは、センサーが 16 ビットの YUV422 データを出力するように設定されているため、データバスが 16 ビットに設定されます。各イメージフォーマットのバス幅の詳細については、FX3 SDK API ガイドの「CyU3PMipicsiDataFormat_t」セクションを参照してください。

フレームが送信された後、ステートマシンはフレーム転送の正常終了を示すために CPU を割り込んで、その結果ヘッダを追加して他のフレームの終了タスクを実行できます。

次のセクションでデータをストリーミングする DMA チャネルおよび UVC をサポートするファームウェアの詳細を説明します。

画像のラインサイズに関する注意:

解像度の幅 (またはバイト単位のラインサイズ) は、4 バイトの倍数である必要があります。この条件が満たされない場合、CX3 の CSI-2 MIPI コントローラーブロックは、ラインデータに追加のパディングバイトを付加します。

3 DMA システムのセットアップ

GPIF II ブロックは最大 100 MHz で動作し、最大 24 ビットのデータ (300 Mbps) を持ちます。内部の DMA バッファにデータを転送するためには、GPIF II は、DMA プロデューサソケットに接続された 2 個の GPIF スレッドを使用します (2 個のソケットが使用される理由セクションで説明)。ソケットと GPIF スレッドのデフォルトのマッピング (Figure 7) はこのアプリケーションに使用されます。ここでは、ソケット 0 が GPIF スレッド 0 に、ソケット 1 が GPIF スレッド 1 に接続されます。この場合、利用可能なスレッドの 4 つの中で、2 つだけが使用されることに注意してください。GPIF のスレッドの切り替えは、前セクションで説明した GPIF II ステートマシンで実行されます。

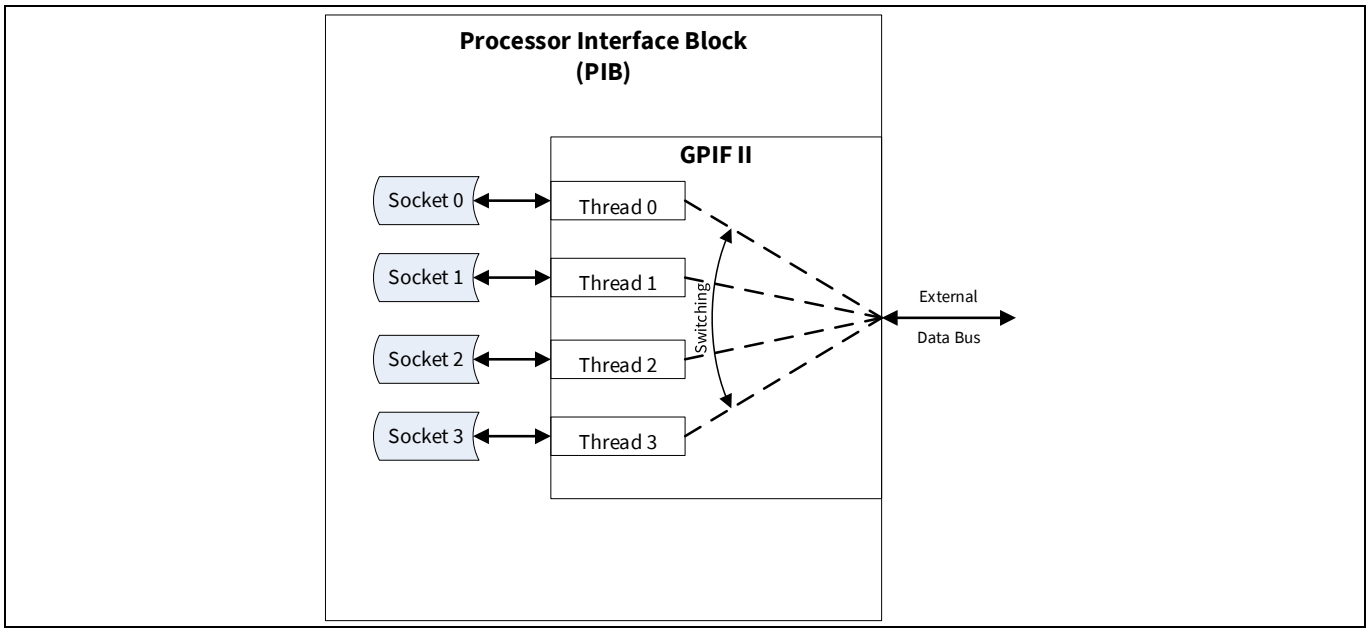

Figure 7 GPIF II ソケットとスレッドのデフォルトマッピング

DMA 転送を理解するために、DMA の基本セクションで説明されたソケットの概念をさらに次の 4 つの図に示します。Figure 8 に、ソケットの 2 つの主な属性であるリンクリストとデータルータを示します。

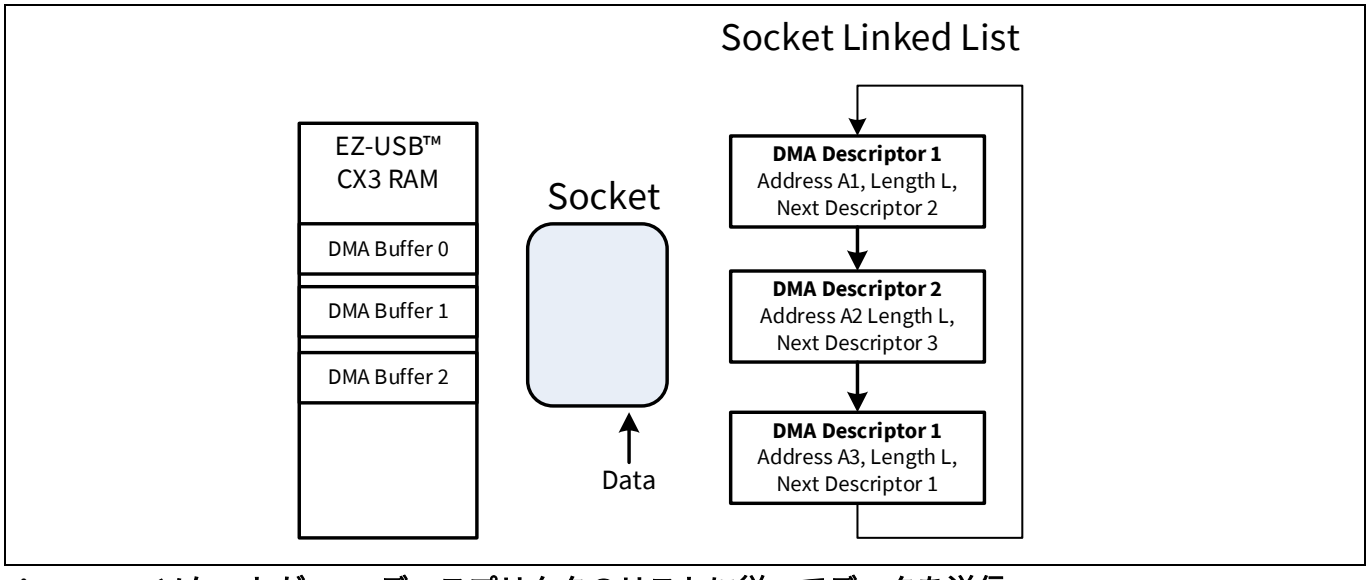

Figure 8 ソケットが DMA ディスプリクタのリストに従ってデータを送信

3 DMA システムのセットアップ

ソケットリンクリストは DMA ディスクリプタと呼ばれるメインメモリ内の一連のデータ構造です。それぞれのディスクリプタは、DMA バッファのアドレス (An) と長さ (L) および次の DMA ディスクリプタへのポインタを指定します。ソケットは動作する時、DMA ディスクリプタを 1 つずつ読み出し、ディスクリプタのアドレスと長さで指定された DMA バッファにデータを転送します。L バイトが転送された時、ソケットは次のディスクリプタを読み出して、バイトを別の DMA バッファに引き続き転送します。

この構造により、DMA バッファをメモリ内の場所を問わず何個でも作って自動的に一緒に連結できるため、ソケットを多目的に使用できます。例えば Figure 9 のソケットは、繰り返しループで DMA ディスクリプタを読み出します。

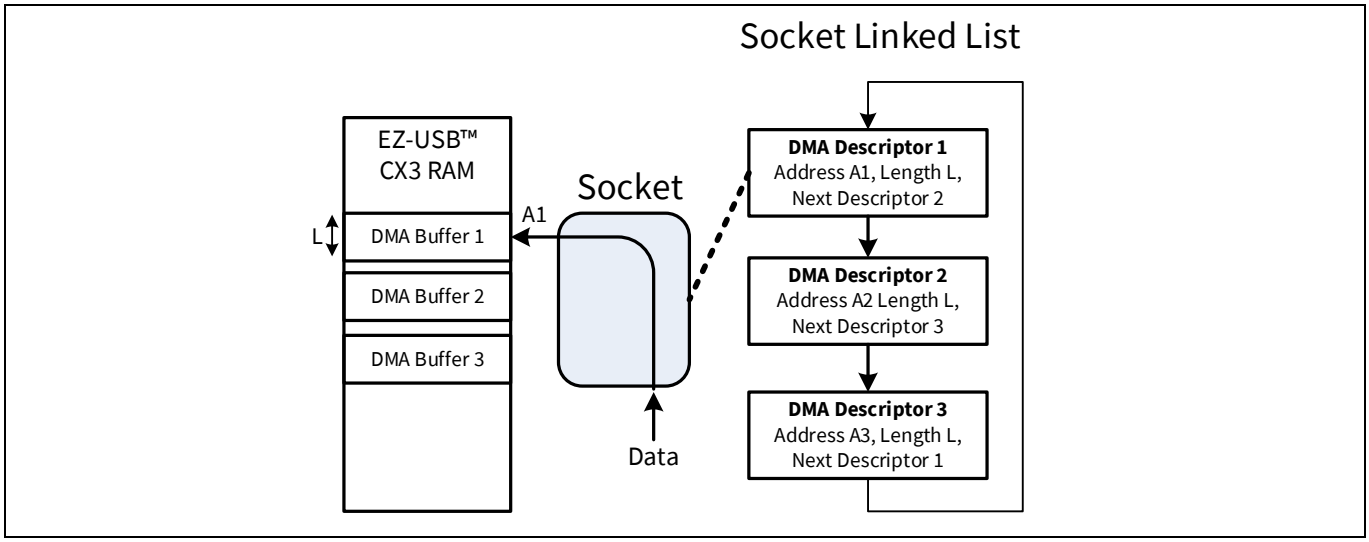

Figure 9　DMA ディスプリクタ 1 とともに動作するソケット

Figure 9 では、ソケットは DMA ディスクリプタ 1 をロードしました。DMA ディスクリプタはソケットが L バイトを転送するまで、A1 から始まるバイトを転送するように指示します。それに続いて、ソケットは DMA ディスクリプタ 2 を読み出し、L バイトを転送するまで、A2 から始まるバイトを転送します (Figure 10)。

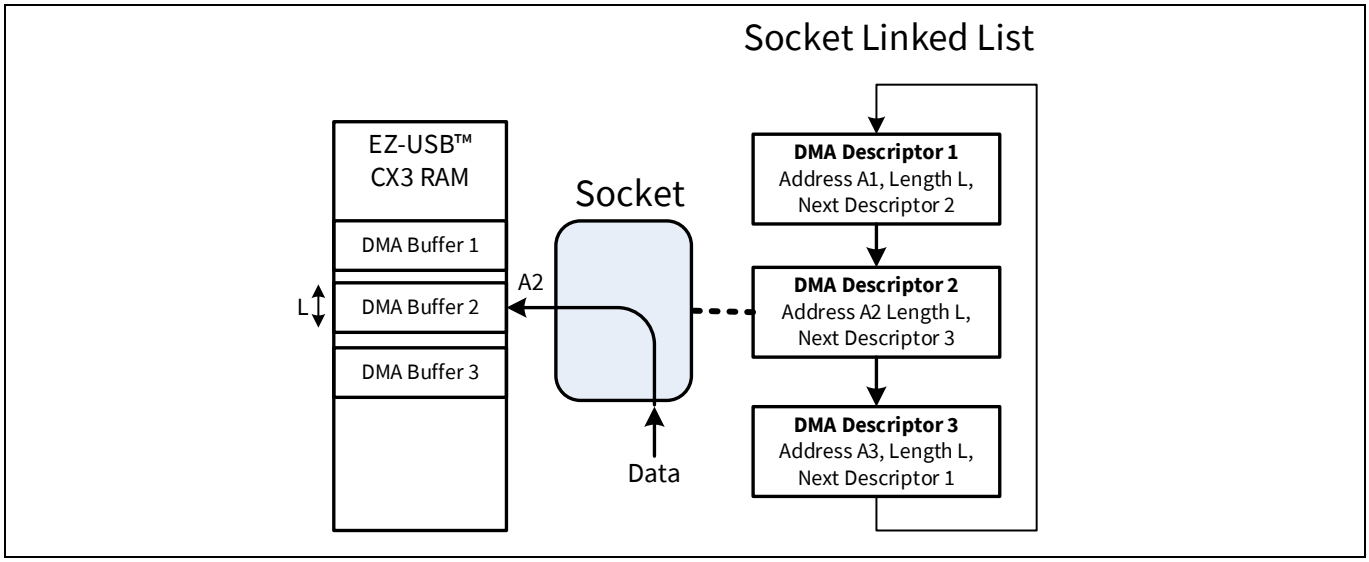

Figure 10　DMA ディスプリクタ 2 とともに動作するソケット

3 DMA システムのセットアップ

Figure 11 ではソケットは 3 番目の DMA ディスクリプタを読み出して、A3 から始まってデータを転送します。L バイトを転送した後、DMA ディスクリプタ 1 に戻って動作を繰り返します。

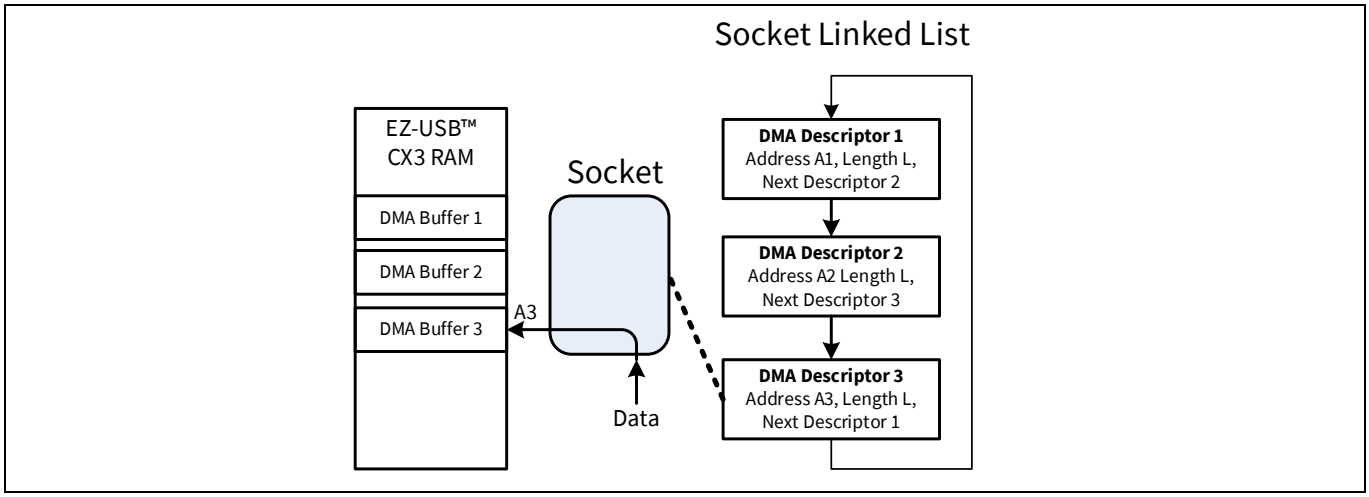

Figure 11　DMA ディスプリクタ 3 とともに動作するソケット

Figure 12 には、詳細な DMA データ転送を示します。この例は、円形ループで連結した長さ L の 3 つの DMA バッファを使用しています。CX3 メモリアドレスは左側にあります。青色の矢印はソケットがソケットリンクリストのディスクリプタをメモリからロードしていることを示します。赤色の矢印は、結果となるデータパスを示します。以下の手順は、データが内部 DMA バッファに転送される時のソケットのシーケンスを示します。

ステップ 1: DMA ディスクリプタ 1 をメモリからソケットにロードします。DMA バッファ位置 (A1), DMA バッファサイズ (L), および次のディスクリプタ (DMA ディスクリプタ 2) の情報を取得します。ステップ 2 に進みます。

ステップ 2: A1 から始まって DMA バッファ位置にデータを転送します。DMA バッファサイズである L のデータ量を転送した後、ステップ 3 に進みます。

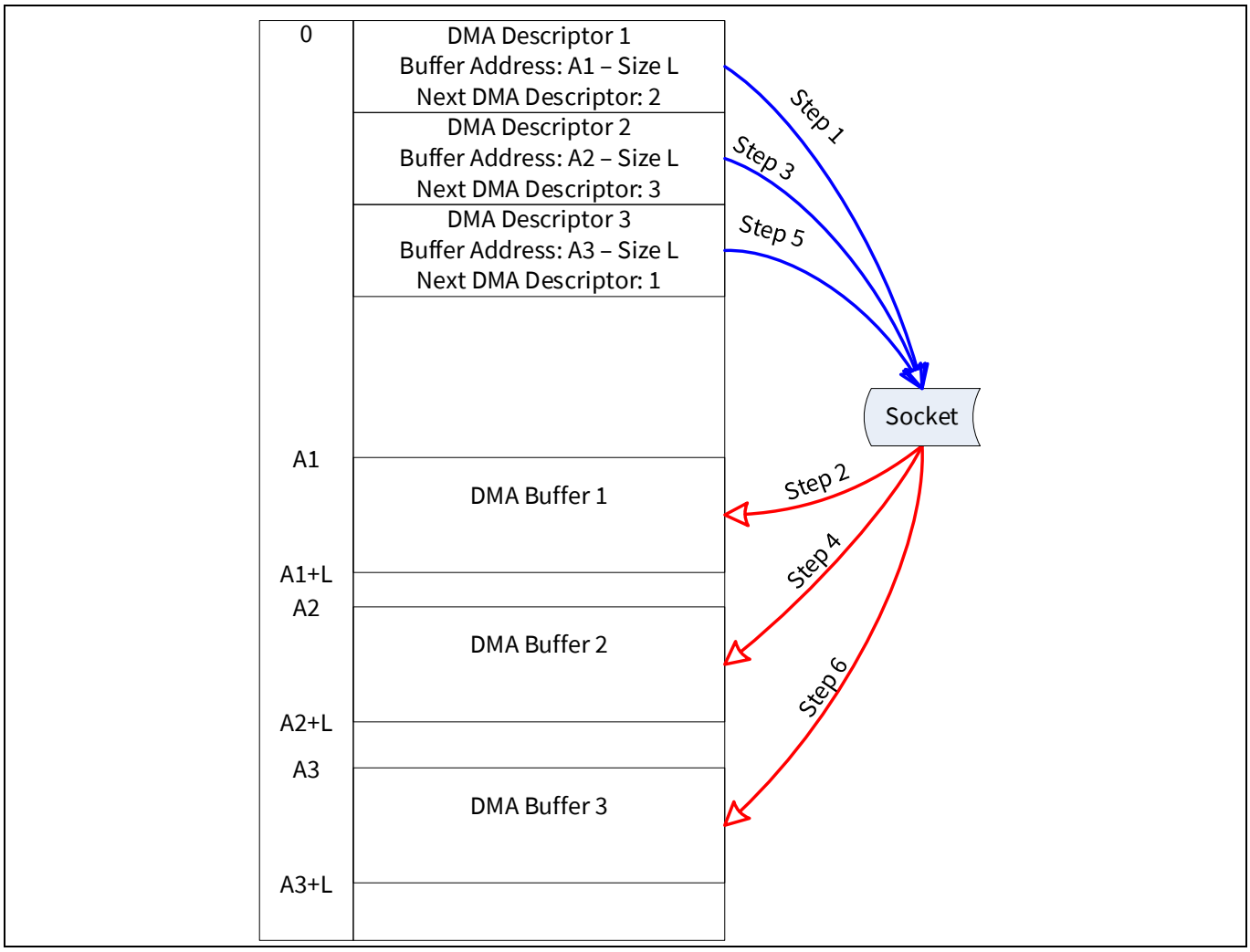

Figure 12　DMA 転送例

ステップ 3: 現時点の DMA ディスクリプタ 1 が指している DMA ディスクリプタ 2 をロードします。DMA バッファ位置 (A2)、DMA バッファサイズ (L) および次のディスクリプタ (DMA ディスクリプタ 3) の情報を取得します。ステップ 4 に進みます。

ステップ 4: A2 から始まって DMA バッファ位置にデータを転送します。DMA バッファサイズである L のデータ量を転送した後、ステップ 5 に進みます。

ステップ 5: 現時点の DMA ディスクリプタ 2 が指している DMA ディスクリプタ 3 をロードします。DMA バッファ位置 (A3)、DMA バッファサイズ (L) および次のディスクリプタ (DMA ディスクリプタ 1) の情報を取得します。ステップ 6 に進みます。

ステップ 6: A3 から始まって DMA バッファ位置にデータを転送します。DMA バッファサイズである L のデータ量を転送した後、ステップ 1 に戻ります。

この単純な方式は、カメラアプリケーションの問題があります。ソケットはメモリから次の DMA ディスクリプタを読み出すのに通常 1 µs かかります。この転送では、データ転送の途中に一時停止が発生すればビデオデータは失われます。この損失を防ぐために、DMA バッファ サイズはビデオラインの長さの倍数に設定できます。これにより、DMA バッファ切り替えの一時停止時間はビデオラインの非アクティブ時間 (即ち、センサーの垂直帰線消去時間) と一致します。しかし、この方法は、例えばビデオの解像度が変更された場合、柔軟性を持ちません。

3 DMA システムのセットアップ

DMA バッファサイズをラインサイズに全く等しく設定することは、BULK 転送にとって USB 3.2 Gen 1 の最大バースト速度が良い効果を出さないため、良い解決策ではありません。USB 3.2 Gen 1 は BULK エンドポイント上で最大 16 バースト (1 バーストが 1024 バイト) を可能にします。そのため、DMA バッファサイズが 16 KB に設定されます。

より良い解決策は、ソケットが 1 クロックサイクル以内にレイテンシなしに切り替わるという利点を利用することです。したがって、4 個のインターリーブ DMA バッファにデータを格納するために、2 つのソケットを使用することは意味があります。

Note: データの損失を防止するために、ソケットごとに少なくとも 2 つのバッファが必要となります (すなわち、ソケットが 1 つのバッファに書き込んでいる間にホストが他のバッファから読み出せるという 2 つのバッファです)。

Note: Figure 5 に示すもの (ソケットごとに 4 バッファ) と次の図に示すもの (ソケットごとに 2 バッファ) が異なります。それは説明を簡略化するためです。しかし、実際のファームウェアはソケットごとに 4 つのバッファを設定します。

デュアルソケットを使用したデータ転送は、実行ステップを番号付けした Figure 13 で説明します。ソケット 0 とソケット 1 の DMA バッファへのアクセスは、それぞれ赤色と緑色の矢印 (個別のソケットのデータ経路) で区別されます。各ステップの「a」と「b」部分は同時に実行されます。このハードウェアの並行動作により、DMA ディスクリプタの回復デッド タイムが無くなり、GPIF II がデータを内部メモリに連続してストリーミングできます。これらのステップは、Figure 12 に示す「ステップ」ラインに対応します。

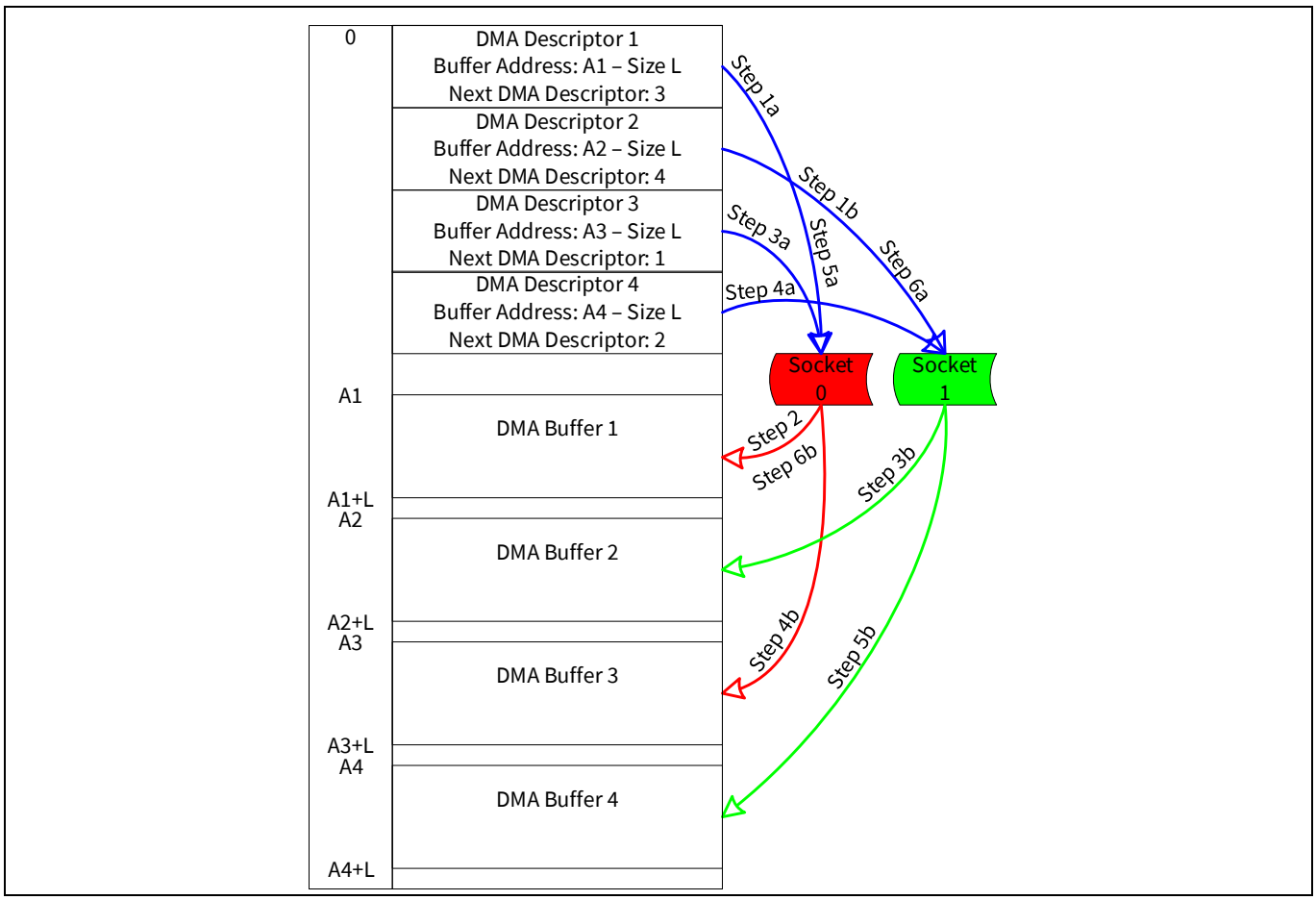

Figure 13 デュアルソケットを使用したシームレスな転送

3 DMA システムのセットアップ

ステップ 1: ソケットの初期化時に、ソケット 0 とソケット 1 は、それぞれ DMA ディスクリプタ 1 と DMA ディスクリプタ 2 をロードします。

ステップ 2: 転送データの準備ができると、ソケット 0 はすぐにデータを DMA バッファ 1 に転送します。転送の長さは L です。この転送が完了すると、ステップ 3 に進みます。

ステップ 3: GPIF II の固定機能ステート マシンは GPIF スレッドを切り替えるため、データ転送用のソケットも切り替わります。ソケット 1 が DMA バッファ 2 へのデータ転送を開始すると同時に、ソケット 0 は DMA ディスクリプタ 3 をロードします。ソケット 1 が L のデータ量の転送を終了するまでに、ソケット 0 は DMA バッファ 3 へのデータ転送の準備ができます。

ステップ 4: ここで、GPIF II は元の GPIF スレッドに切り替わります。ソケット 0 は長さ L のデータを DMA バッファ 3 に転送します。同時に、ソケット 1 は DMA ディスクリプタ 4 をロードして、DMA バッファ 4 へのデータ転送の準備ができます。ソケット 0 が長さ L のデータの転送を終了したら、ステップ 5 に進みます。

ステップ 5: GPIF II は、ソケット 1 のデータを DMA バッファ 4 に送信します。同時にソケット 0 は DMA ディスクリプタ 1 をロードして、DMA バッファ 1 にデータを転送する準備をします。ソケット 1 を初期化せずにデータを同時に転送する点を除き、ステップ 5a はステップ 1a と同様であることに注意してください。

ステップ 6: GPIF II は再びソケットを切り替えて、ソケット 0 は DMA バッファ 1 への長さ L のデータの転送を開始します。DMA バッファは、ここまでは UIB コンシューマソケットにより使い果たされたから、空になっていることが想定されます。それと同時に、ソケット 1 は DMA ディスクリプタ 2 をロードして、DMA バッファ 2 へのデータ転送の準備ができます。ここで、サイクルは実行経路のステップ 3 に進みます。

コンシューマ側 (USB) が GPIF II から次のビデオデータのチャンクを受信するのに間に合うように DMA バッファを空にして解放した場合にのみ、GPIF II ソケットはビデオ データを転送できます。コンシューマの速度が十分に速くなかったら、ソケットの DMA バッファへの書き込みが無視されるためソケットはデータを失います。その結果、バイトカウンターは実際の転送との同期を失います。このような同期喪失は次のフレームに伝播することがあります。したがって、フレームが長時間転送されない場合は、クリーンアップメカニズムが必要となります。このメカニズムをアプリケーションスレッドで説明します。

フレーム転送は、次の 4 つの可能なシナリオのいずれかで終了できます。

- ソケット 0 が一杯の DMA バッファを転送した
- ソケット 1 が一杯の DMA バッファを転送した
- ソケット 0 が部分的な DMA バッファを転送した
- ソケット 1 が部分的な DMA バッファを転送した

最後の 2 つシナリオでは、CPU は USB コンシューマに部分的な DMA バッファを転送する必要があります。

このアプリケーション ノートでは、SDK_INSTALL_PATH\firmware\cx3_examples\cycx3_uvc_ov5640 フォルダにあるプロジェクトを使用します。ここで SDK_INSTALL_PATH は FX3 SDK インストールのロケーションです。32 ビットシステムの初期設定インストール パスは、C:\Program Files\Cypress\EZ-USB FX3 SDK\1.3 です。64 ビットシステムの場合、このパスは C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3 です。

DMA チャネルは、cycx3_uvc.c ファイル内の「CyCx3Applnit」と呼ばれるファイルの関数を使って初期化されます。DMA チャネルコンフィギュレーションの詳細は、同じ関数内にある「dmaCfg」構造体でカスタマイズされます。DMA チャネル タイプは MANUAL_MANY_TO_ONE に設定されます。

また、USB 3.2 Gen 1 ホストへデータをストリーミングする USB エンドポイントは、BULK エンドポイントを介した 16 パケット (パケットサイズが 1024 バイト) のバースト転送を可能にするように設定されます。これは、エンドポイントが「CX3_EP_BULK_VIDEO」に固定して設定された「CyU3PSetEpConfig」関数内に渡された「endPointConfig」構造体を使用して設定されます。

3.1 DMA バッファ

ここでは CX3 DMA バッファの作成方法とこのアプリケーション内での使用方法の概要を説明します。DMA チャネルに不可欠な部分は DMA の基本で説明します。

本アプリケーションでは、GPIF II ユニットが生産者 (プロデューサ) であり、USB ユニットが消費者 (コンシューマ) です。このアプリケーションでは、データ損失を避けるために、GPIF II ブロック内で GPIF スレッドの切り替え機能を使用します。

プロデューサソケットは、DMA ディスクリプタをロードすると、対応する DMA バッファをチェックして書き込み動作の準備ができているかを確認します。プロデューサソケットは、DMA バッファが空であることを検出した場合、CX3 RAM にデータを書き込むためにアクティブ状態に移行します。プロデューサソケットは、書き込み動作用に DMA バッファをロックします。

プロデューサソケットは DMA バッファへの書き込みを終了すると、コンシューマソケットが DMA バッファにアクセスできるようにロックを解除します。このアクションは「バッファ ラップアップ」または簡単に「ラップアップ」と呼ばれます。その後、DMA ユニットは DMA バッファを CX3 RAM に転送するように指示されます。プロデューサソケットは、DMA バッファを生成したといわれます。プロデューサが積極的に DMA バッファを満たさない時にのみ、DMA バッファをラップアップする必要があります。

DMA バッファが完全に満たされると、フレーム中に繰り返すため、ラップアップ動作は自動的に行われます。プロデューサソケットは、DMA バッファのロックを解除して CX3 RAM に DMA バッファを転送し、空の DMA バッファへ切り替え、ビデオデータストリームの書き込みを継続します。

コンシューマ ソケットが DMA ディスクリプタをロードすると、読み出し動作の準備ができたか確認するために対応する DMA バッファをチェックします。コンシューマ ソケットは、DMA バッファが転送されたことを検出した場合、CX3 RAM からデータを読み出すためにアクティブ状態に移行します。コンシューマ ソケットは、読み出し動作用に DMA バッファをロックします。

- コンシューマ ソケットは DMA バッファからすべてのデータを読み出した後、プロデューサソケットが DMA バッファにアクセスできるようにロックを解除します。コンシューマソケットは、DMA バッファを消費したといわれます。

- 同じ DMA ディスクリプタがプロデューサソケットとコンシューマソケットによって同時に使用されている場合、DMA バッファの満杯/空の状態は、DMA ディスクリプタとソケット間のイベントを介してプロデューサソケットとコンシューマソケット間で自動的にやり取りされます。

- このアプリケーションでは、CPU が 12 バイトの UVC ヘッダを追加する必要があるため、プロデューサソケットとコンシューマソケットは異なる DMA ディスクリプタセットをロードする必要があります。プロデューサソケットによりロードされた DMA ディスクリプタは、コンシューマソケットによりロードされた DMA ディスクリプタが指定する対応 DMA バッファから 12 バイトのオフセット補正をした DMA バッファを指します。

- プロデューサソケットとコンシューマソケットに対応する異なる DMA ディスクリプタに起因して、CPU はプロデューサソケットとコンシューマソケット間の DMA バッファ状態の情報伝達を管理する必要があります。そのため、DMA チャネルの実装が「手動 DMA」チャネルと呼ばれます。

3 DMA システムのセットアップ

- DMA バッファが GPIF II ブロックにより生成された後、CPU は割込みを介して通知されます。CPU はその後、ヘッダ情報を追加し、コンシューマ (USB インターフェースブロック) に DMA バッファを転送します。
- GPIF II 側では、DMA バッファ内のビデオ データは自動的にラップアップされ、フレーム内の最後の DMA バッファ以外、CX3 RAM にすべてのバッファ内のデータを転送します。
- フレームの終わりで最終の DMA バッファは、完全に満たされない場合があります。この場合 CPU が介入して手動で DMA バッファをラップアップし、CX3 RAM にそれを転送します。これは、「強制的なラップアップ」と呼ばれます。

Note: DMA バッファサイズ (バイト単位) を UVC フレームサイズ (バイト単位) の倍数ではない値に設定すると、フレームの最後のデータは DMA バッファを完全には埋めない状態になります。この「部分的に埋まった」DMA バッファを利用することで、画像データのストリーミング中にフレーム終了 (EOF) 条件を検出し、適切な UVC ヘッダを付加することが可能になります。

Note: 結果として得られる部分 DMA バッファのサイズは 4 の倍数である必要があります。これは、`CyU3PDmaSocketSetWrapUp` API が 4 の倍数単位でしかデータをコミットできないためです。

4 USB ビデオクラス (UVC)

ここでは、ビデオデータを PC へ転送するための、ユニバーサルシリアルバス (USB) 仕様の上にあるプロトコルである USB ビデオクラスについて説明します。

UVC クラスに準拠するには、2 個の CX3 コードモジュールが必要です。

- エニュメレーション データ
- 動作コード

4.1 エニュメレーションデータ

SDK インストールの cycx3_uvc_ov5640 プロジェクトは、cycx3_uvcdscr.c と名付けられた、UVC エニュメレーションデータを持つファイル (UVC の USB ディスクリプタを参照) を含みます。UVC ディスクリプタの形式を定義する USB 仕様は usb.org から入手できます。ここではディスクリプタについて概説します。UVC デバイスには 4 つの論理要素があります。各要素は 1 つのディスクリプタにより定義されます。

- カメラ入力端子 (IT)
- 出力端子 (OT)
- プロセッシングユニット (PU)
- エクステンションユニット (EU)

これら要素はディスクリプタ内で Figure 14 のように接続します。要素間の接続はディスクリプタの端子番号を関連付けることで行われます。例えば、入力 (カメラ) 端子ディスクリプタはその ID を 1 と宣言し、プロセッシングユニットディスクリプタはその入力接続の ID が 1 と指定します。したがって、プロセッシングユニットディスクリプタは論理的には入力端子に接続します。出力端子ディスクリプタは、どの USB エンドポイントを使用するかを示します。この場合は BULK-IN エンドポイント 3 です。

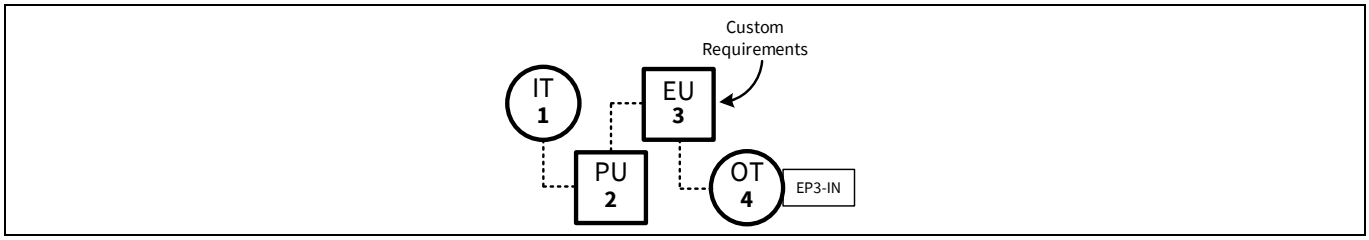

Figure 14 カメラのアーキテクチャの UVC 図

ディスクリプタは、ビデオの特性 (幅, 高さ, フレームレート, フレームサイズ, ビット深度など) および制御特性 (輝度, 露光, 増幅率, コントラスト, パン/チルト/ズーム即ち PTZ) も含みます。

4.2 動作コード

ホストがカメラをエニュメレートした後、UVC ドライバは一連の要求をカメラに送信して動作の特性を判断します。これは、「機能要求段階」と呼ばれます。この段階は、ホストアプリケーションがビデオのストリームを開始するストリーム段階に先行します。

例えば、UVC デバイスはその USB ディスクリプタのいずれかで輝度制御をサポートしていることを示します。機能要求段階中に、UVC ドライバはデバイスに問い合わせて、関連する輝度パラメーターを得ます。

ホストアプリケーションが輝度値を変更するよう要求すると、UVC ドライバは SET 制御要求 (SET_CUR) を発行して輝度を変更します。これは USB コントロールエンドポイント (EP0) により実行されます。

同様に、ホストアプリケーションがサポートされたビデオフォーマット, フレームレート, フレームサイズをストリーミングに選択すると、UVC ドライバはストリーミング要求を発行します。要求は、PROBE および COMMIT という 2 種類あります。COMMIT 要求は変更を実行するために使用されますが、PROBE 要求は、UVC デバイスがストリームモードへの変更を受け入れる準備ができたかを判断するために使用されます。ストリーミングモードは、イメージフォーマット, フレーム サイズ, およびフレームレートの組合せです。

4.3 UVC の要件

ここではどのように UVC 要件が満たされるかについてサンプルプロジェクトで説明します。UVC はデバイスに次の動作を要求します。

- UVC 固有の USB ディスクリプタでエニュメレートする
- USB ディスクリプタで報告された UVC 制御とストリーム機能の SET/GET UVC 固有の要求を処理する
- UVC 準拠の色フォーマットでビデオ データをストリーミングする
- 各イメージ ペイロードに UVC 準拠のヘッダを追加する

これらの要件の詳細は、UVC 仕様に記載されています。

4.3.1 UVC の USB ディスクリプタ

cycx3_uvcdscr.c ファイルは USB ディスクリプタの表を含みます。配列「CyCx3USBHSConfigDscr」 (Hi-Speed) および「CyCx3USBSSConfigDscr」 (SuperSpeed) は UVC 固有のディスクリプタを含みます。これらのディスクリプタは以下のツリー構造のサブディスクリプタを含みます。

- コンフィギュレーションディスクリプタ
  - インターフェースアソシエーションディスクリプタ
  - ビデオ制御 (VC) インターフェースディスクリプタ
    - VC インターフェースヘッダディスクリプタ
      - 入力 (カメラ) 端子ディスクリプタ
      - プロセッシングユニット ディスクリプタ
      - エクステンションユニットディスクリプタ
      - 出力端子ディスクリプタ
    - VC ステータス割込みエンドポイントディスクリプタ
  - ビデオストリーミング (VS) インターフェースディスクリプタ
    - VS インターフェース入力ヘッダディスクリプタ
      - VS フォーマットディスクリプタ
    - VS フレームディスクリプタ
  - BULK-IN ビデオエンドポイントディスクリプタ

コンフィギュレーションディスクリプタは、そのサブディスクリプタで USB デバイスの機能を定義する標準 USB ディスクリプタです。インターフェースアソシエーションディスクリプタは、デバイスが標準 USB クラスに準拠していることをホストに示します。ここで、このディスクリプタはビデオ制御 (VC) インターフェースとビデオストリーミング (VS) インターフェースを備えた UVC 準拠のデバイスを報告します。2 つの個別のインターフェースを備えた UVC デバイスは USB 複合デバイスとなります。

4.3.1.1 ビデオ制御 (VC) インターフェース

VC インターフェースディスクリプタおよびそのサブディスクリプタは、制御インターフェースに関連するすべての機能を報告します。例としては、輝度, コントラスト, 色相, 露光, PTZ 制御などがあります。

VC インターフェースヘッダディスクリプタは、この VC インターフェースが属している VS インターフェースを指す UVC 固有のインターフェースディスクリプタです。

入力 (カメラ) 端子ディスクリプタ, プロセッシングユニットディスクリプタ, エクステンションユニットディスクリプタ, および出力端子ディスクリプタは、それぞれの端子またはユニットがサポートしている機能を記述するビットフィールドを含みます。

カメラ端子は、ビデオストリームを送信するデバイスの露光や PTZ などの機械的な機能 (または相当するデジタル機能) を制御します。

プロセッシングユニットは、それを介してストリーミングされるビデオの輝度, コントラスト, および色相などのイメージ特性を制御します。

エクステンションユニットは、標準の USB ベンダー要求のようにベンダー固有の機能を追加することを可能にします。この設計ではエクステンションユニットは空ですが、ディスクリプタはカスタム機能用のプレースホルダとして含まれます。エクステンションユニットを利用する場合、認識するよう変更しなければ標準ホストアプリケーションはその機能を認識しません。

出力端子はそれらユニット (IT, PU, EU) とホスト間のインターフェースを記述します。VC ステータス割込みエンドポイントディスクリプタは、割込みエンドポイント用の標準 USB ディスクリプタです。このエンドポイントは、UVC 固有のステータス情報を通信するために使用できます。このエンドポイントの機能は本アプリケーションノートの範囲外です。

UVC 仕様ではクラス固有の制御要求の実装を容易に構成できるよう、これらの機能を分割します。ただし、これらの機能の実装はアプリケーションによって異なります。サポートされた制御機能は、それぞれの端子またはユニットディスクリプタの「bmControls」ビットフィールド (cycx3_uvcdscr.c) で、対応する機能のビットを「1」にセットすることで報告されます。UVC デバイスドライバは、エニュメレーション時に制御の情報をポーリングします。情報のポーリングは EP0 要求により実行されます。そのようなすべての要求 (ビデオ ストリーミング要求を含む) は、*cycx3_uvc.c* ファイル内の CyCx3AppUSBSetupCB 関数により処理されます。

4.3.1.2 ビデオストリーミング (VS) インターフェース

VS インターフェースディスクリプタおよびそのサブディスクリプタは、様々なフレームフォーマット (非圧縮, MPEG, H.264 など)、フレーム解像度 (幅, 高さ, ビット深度) およびフレームレートを報告します。報告された値に基づいてホストアプリケーションは、サポートされたフレームフォーマット, フレーム解像度, およびフレームレートの組合せを選択してストリーミングモードを切り替えます。

VS インターフェース入力ヘッダディスクリプタは、それに続く VS フォーマットディスクリプタの数を指定します。

VS フォーマットディスクリプタは、イメージのアスペクト比および非圧縮や圧縮などの色フォーマットを含みます。

VS フレームディスクリプタは、イメージ解像度およびそれのサポートされたすべてのフレームレートを含みます。カメラが異なる解像度をサポートする場合、複数の VS フレームディスクリプタが VS フォーマットディスクリプタに続きます。

BULK-IN ビデオエンドポイントディスクリプタは、ビデオをストリーミングするのに使用するバルクエンドポイントの情報を含む標準 USB エンドポイントディスクリプタです。

この例では、2 組の解像度とフレームレートをサポートするために、2 つのフレームディスクリプタを使用します。そのイメージ特性は、以下の 3 つの表に示されるように 3 つのディスクリプタに含まれます (関連のあるバイトオフセットだけが示されます)。

Table 1　VS フォーマットディスクリプタの値

VS フォーマットディスクリプタのバイトオフセット	特性	SuperSpeed の値	Hi Speed の値
23〜24	幅対高さ比	16:9	4:3

Table 2　第 1 の VS フレームディスクリプタの値

VS フォーマットディスクリプタのバイトオフセット	特性	SuperSpeed の値	Hi Speed の値
5〜8	幅対高さ比	0x780, 0x438 (1920 × 1080)	0x140, 0xF0 (320 × 240)
17〜20	バイト単位での最大イメージサイズ	0x3F4800 (1920 × 1080 × 2)	0x25800 (320 × 240 × 2)
21〜24, および 26〜29	100 ns 単位でのフレーム間隔	0x51615 (30 fps)	0x1B207 (90 fps)

Table 3　第 2 の VS フレームディスクリプタの値

VS フォーマットディスクリプタのバイトオフセット	特性	SuperSpeed の値	Hi Speed の値
5〜8	幅対高さ比	0x500, 0x2D0 (1280 × 720)	0x280 ,0x1E0 (640 × 480)
17〜20	バイト単位での最大イメージサイズ	0x1C2000 (1280 × 720 × 2)	0x96000 (640 × 480× 2)
21〜24, および 26〜29	100 ns 単位でのフレーム間隔	0x28B0A (60 fps)	0x28B0A (60 fps)

複数バイトの値が LSB ファースト (リトルエンディアン) で格納され送信される点に注意してください。したがって、例えば最初の SuperSpeed フレームレートを 30fps とすると、フレーム間隔は次のように計算されます。

$$FrameInterval = \frac{1}{30\ fps \times 100ns} = 333333 = 0x51615$$

これはディスクリプタで 0x15, 0x16, 0x05, および 0x00 として保存されます。この設計は、上記の表の項目を変更することで、異なるイメージ解像度に対応できるように適用できます。

4.3.2　UVC 固有の要求

UVC 仕様では、USB 制御エンドポイント EP0 を使用して制御とストリーミング要求を UVC デバイスに通信します。これらの要求は、ビデオ関連の制御特性を調べ、変更するのに使用されます。UVC 仕様では、これらのビデオ関連の制御が機能として定義されています。これらの機能により、イメージ特性を変更する、またはビデオをストリーミングできます。

4 USB ビデオクラス (UVC)

機能は、ビデオ制御特性 (輝度, コントラスト, 色相など) またはビデオストリーミングモードの特性 (色フォーマット, フレームサイズ, フレーム レートなど) です。機能は USB コンフィギュレーションディスクリプタの UVC 固有のセクションを介して報告されます。各機能は属性を持っています。機能の属性は以下のとおりです。

- 最小値
- 最大値
- 最小値と最大値間の値の数
- 初期設定値
- 現行値

SET と GET は、2 タイプの UVC 固有の要求です。SET は属性の現行値を変更するのに使用され、GET は属性を読み出すのに使用されます。

UVC 固有の要求のリストは以下のとおりです。

- SET_CUR は唯一の SET 要求タイプです。
- GET_CUR は現在の値を読み出します。
- GET_MIN はサポートされた最小値を読み出します。
- GET_MAX はサポートされた最大値を読み出します。
- GET_RES は、解像度 (最小値と最大値間のサポートされた値を示すステップ値) を読み出します。
- GET_DEF は初期設定値を読み出します。
- GET_LEN はバイト単位で特性のサイズを読み出します。
- GET_INFO は状態/特定の機能のサポートを問い合わせます。

UVC 仕様はこれらの要求が与えられた機能ごとに必須または任意であるかを規定しています。例えば、SET_CUR 要求が特定の機能において任意である場合、その存在は GET_INFO 要求により判断されます。カメラがある機能に対する特定の要求をサポートしていない場合、これを示すために、要求がホストからカメラに発行される時、制御エンドポイントをストールします。

これらの要求には、対象となる機能を検証するバイトフィールドがあります。これらバイトフィールドは、UVC の USB ディスクリプタで説明された UVC 固有のディスクリプタと同じ構造を持つ階層を持っています。第 1 レベルはインターフェースを識別します (ビデオ制御またはビデオ ストリーミング)。

第 1 レベルがインターフェースをビデオ制御と識別した場合、第 2 レベルは端子かユニットを識別し、第 3 レベルはその端子かユニットの機能を識別します。例えば、対象となる機能が輝度制御である場合を下記に示します。

- 第 1 レベル=ビデオ制御
- 第 2 レベル=プロセッシング ユニット
- 第 3 レベル=輝度制御

第 1 レベルがインターフェースをビデオストリーミングと識別した場合、第 2 レベルは PROBE か COMMIT となります。第 3 レベルはありません。

UVC デバイスにストリーミングを開始させる、またはストリーミングモードを変更させるために、まずホストはデバイスが新しいストリーミングモードをサポートするかを判断します。このために、ホストは第 2 レベルを PROBE にセットして一連の SET と GET 要求を送信します。デバイスはストリーミングモードへの変更を受け入れるかまたは拒否します。デバイスが変更要求を受け入れた場合、ホストはこれを確認するために、第 2 レベルを COMMIT にセットして SET_CUR 要求を送信します。

4 USB ビデオクラス (UVC)

以下の 3 つのフローチャートは、ホストがどのように UVC デバイスとやり取りするかを示します。

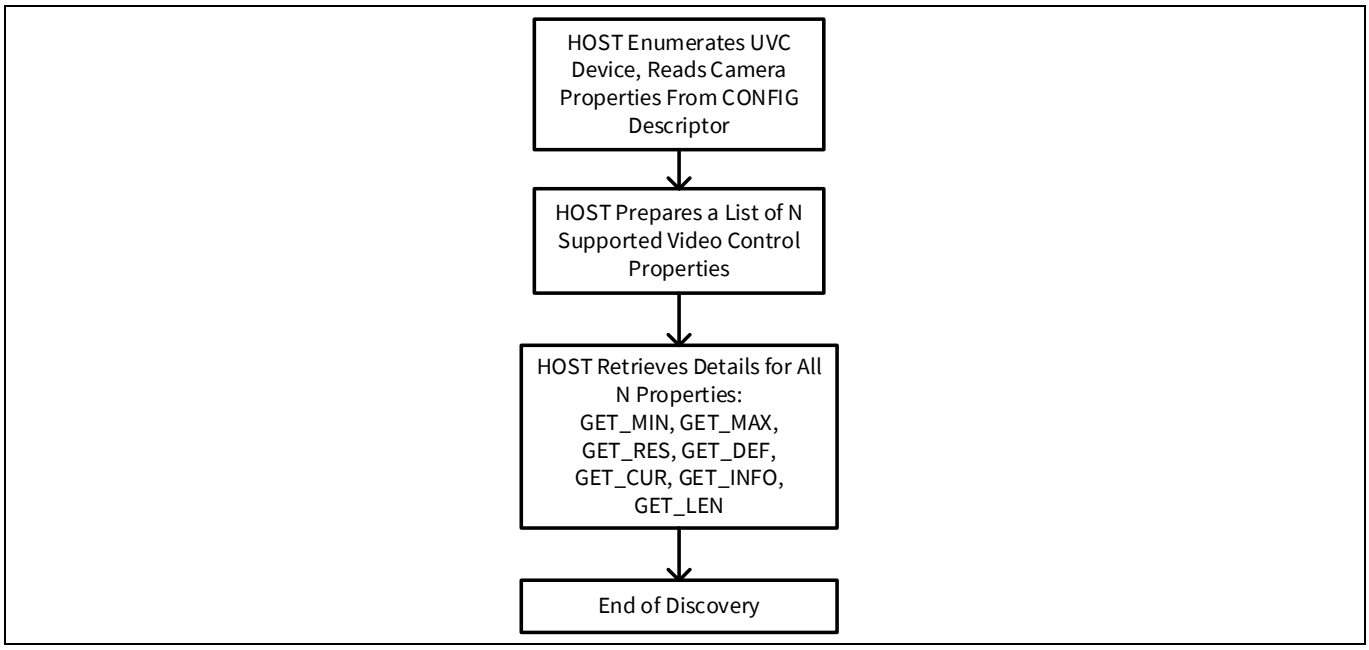

Figure 15　UVC エニュメレーションと検出フロー

UVC デバイスが USB に差し込まれると、ホストはそれをエニュメレートし、カメラがサポートする特性の詳細情報を調べます (Figure 15)。

操作中にカメラマンはホストアプリケーションによって表示されたダイアログから、輝度などカメラの特性を変更できます。Figure 16 にこのやり取りを示します。

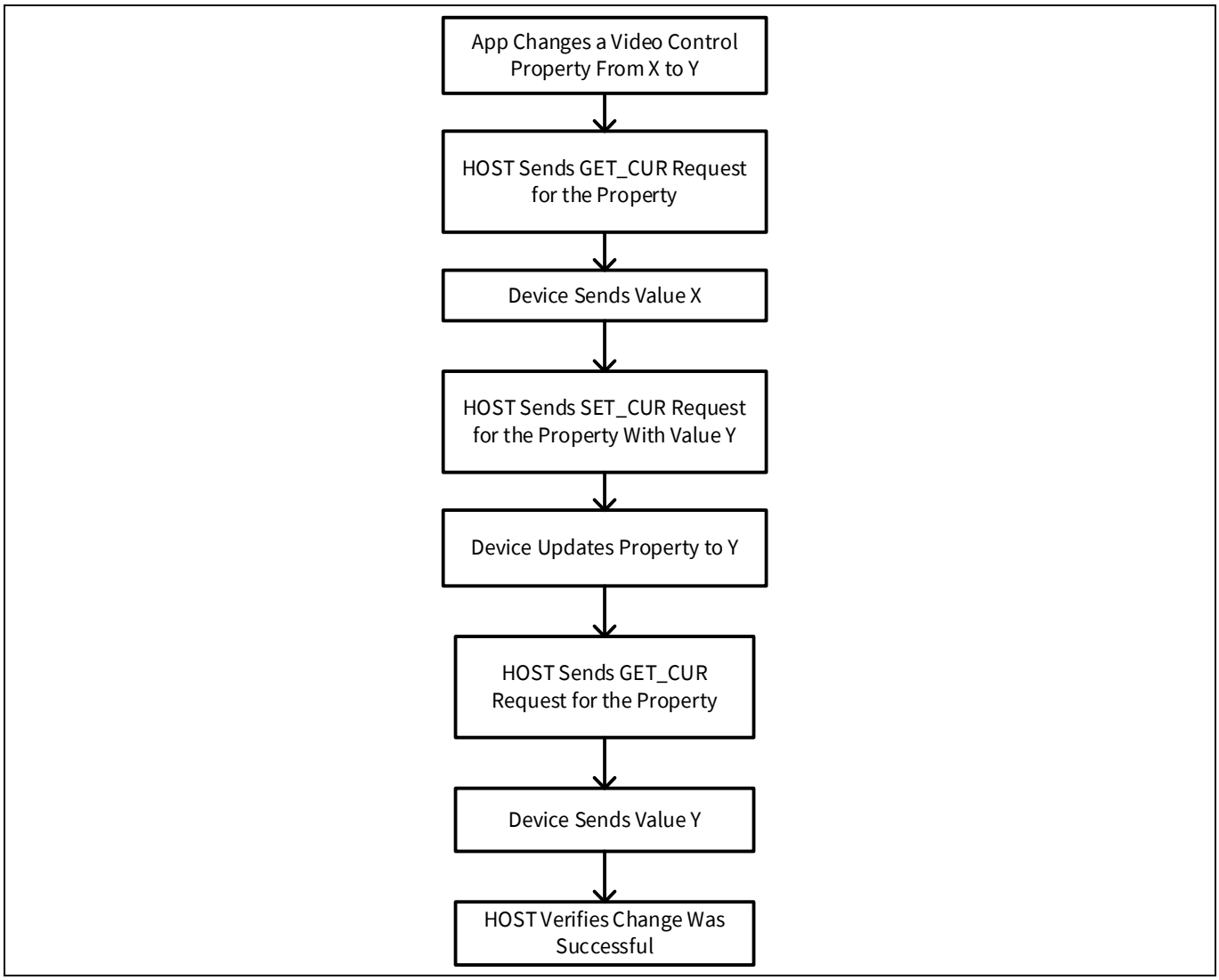

Figure 16　ホストアプリケーションがカメラ設定を変更

ストリーミングを開始する前に、ホストアプリケーションは、一連の PROBE 要求を発行して可能なストリーミングモードを調べます。デフォルトストリーミングモードが決定された後、ホストは希望する解像度とフレームレートのフレームおよびフォーマット ID を持つ COMMIT 要求を発行します。このプロセスを Figure 17 に示します。COMMIT 要求と共にホストによって送信される構造を Table 4 に示します。

この時点で、UVC ドライバは UVC デバイスからビデオをストリーミングする用意ができます。

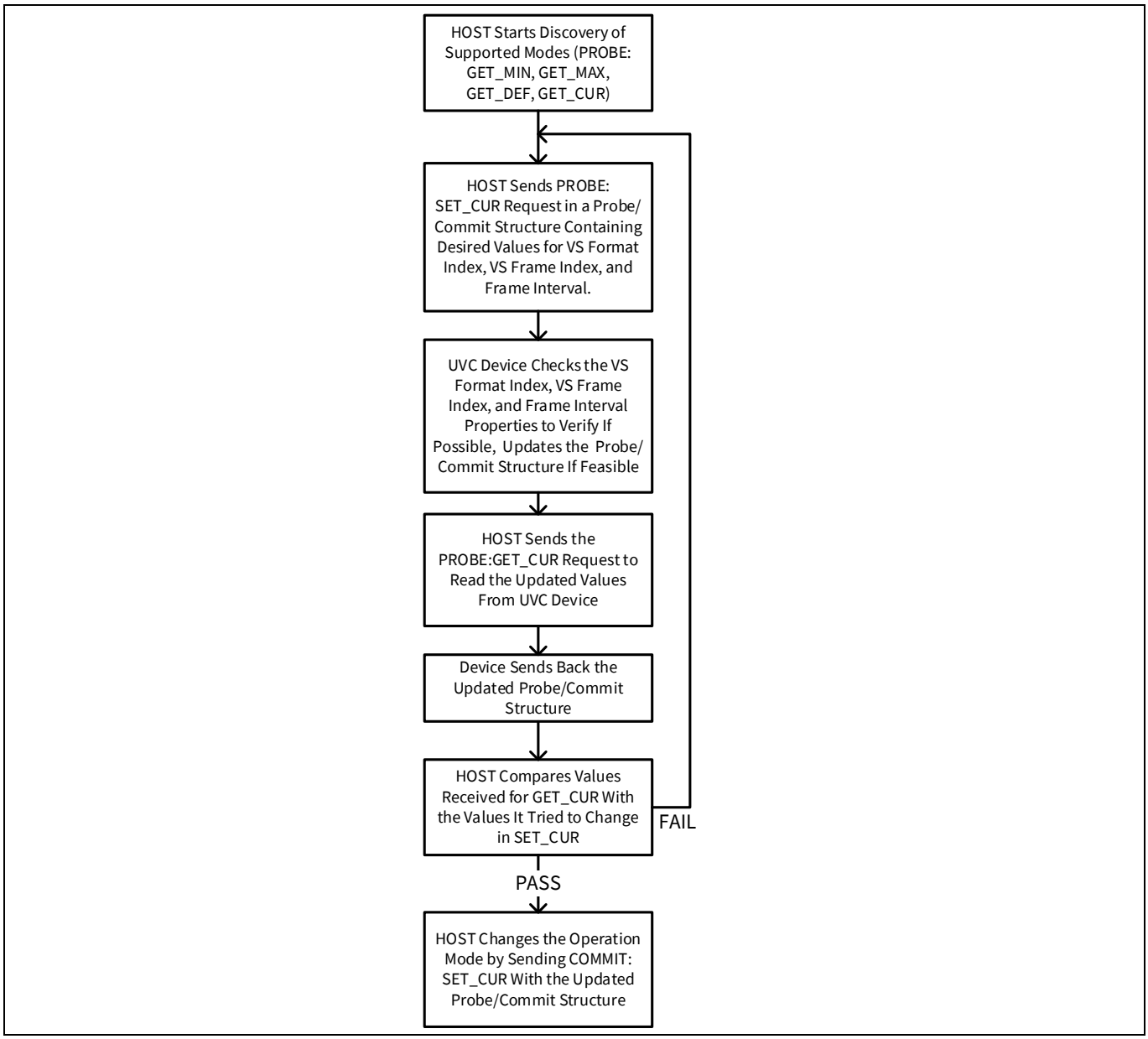

Figure 17　ホストカメラのストリーミング前のダイアログ

Table 4　1080p@30fps ストリームを開始する前にホストによって送信されるコミット構造の値

プローブ/コミット構造のバイトオフセット	特性	値
2	フォーマットインデックス	1
3	フレームインデックス	1
4〜7	100 ns 単位でのフレーム間隔	0x51615 (30 fps)
18〜21	バイト単位での最大イメージサイズ	0x3F4800 (1920 × 1080 × 2)

4.3.2.1 制御要求: 輝度, PTZ, 色相, 彩度, その他

イメージ センサーは、輝度, 色相, 彩度などの画像の機能の制御をサポートでき、外付けハードウェアが PTZ (パン, チルト, ズーム) のサポートのために加えられることがあります。

CX3 ファームウェアでこれをサポートするには、ビデオ制御ディスクリプタが特定の制御のサポートを示すよう修正する必要があります。例えば、輝度の制御をサポートするには、プロセッシングユニット ディスクリプタの bmControls フィールドのビット 0 をセットする必要があります。詳細については、エニュメレーションデータで言及される UVC 仕様を参照してください。

これでホストは指定された要求を特定の端子に送信できるようになります。ホストが送信できる様々な要求のタイプについて UVC 固有の要求で説明します。これらの要求は、CyCx3AppUSBSetupCB 関数により処理されます。

要求を受信すると、ファームウェアはイメージセンサーまたは関連するハードウェアに要求を送信できます。実装はアプリケーションによって異なります。

4.3.2.2 ストリーミング要求: プローブとコミット制御

CyCx3AppUSBSetupCB 関数はストリーミング関連の要求を処理します。UVC ドライバが UVC デバイスからビデオをストリーミングする必要がある場合、第 1 段階は折衝です。この段階では、ドライバは GET_MIN, GET_MAX, GET_RES, GET_DEF などの PROBE 要求を送信します。これに応じて CX3 ファームウェアは PROBE 構造体を返します。この構造体は、ビデオフォーマット, ビデオフレーム, フレームレート, 最大フレームサイズ, ペイロード サイズ (UVC ドライバが 1 回の転送で取り出せるバイト数) の USB ディスクリプタを含みます。

この例では、GET_CUR 要求が処理され、それに続いて現時点のアクティブな接続を確認し、適切な PROBE 構造体を送信します。

GET_CUR の処理に続き、SET_CUR 要求が処理されます。これらの要求はストリーミング段階の最初の時点で送信されます。

COMMIT 制御用の SET_CUR 要求は、要求が完了した後にホストがビデオのストリーミングを開始することを示します。ホストが送信したフレーム ディスクリプタ インデックスの種類 (ホストに送信されたデータ構造の詳細については、Table 4 を参照してください) に応じ、イメージ センサーは、要求された解像度とフレームレートでのビデオ データを送信するよう設定され、続いて、適切な MIPI CSI-2 インターフェースパラメーターがセットされます。これでビデオストリーミングが開始します。

4.3.2.3 ビデオデータフォーマット: YUY2

UVC 仕様では、ビデオ データ用に色フォーマットの 1 サブセットだけがサポートされます。このため、UVC 仕様に準拠した色フォーマットでイメージをストリーミングするイメージセンサーを選択する必要があります。本アプリケーションノートでは、ほとんど (すべてではない) のイメージセンサーによってサポートされた、YUY2 という非圧縮の色フォーマットについて説明します。YUY2 色フォーマットは YUV 色フォーマットをダウンサンプルした 4:2:2 バージョンです。輝度値 Y はピクセルごとにサンプリングされますが、色度 U と V は偶数ピクセルごとにのみサンプリングされます。これは「マクロ ピクセル」を形成します。各マクロ ピクセルは、合計 4 バイトを使う 2 イメージピクセルを表します。他のすべてのバイトは Y 値であり、U と V 値は偶数ピクセルのみを表します。

Y0, U0, Y1, V0 (最初の 2 ピクセル)

Y2, U2, Y3, V2 (次の 2 ピクセル)

Y4, U4, Y5, V4 (次の 2 ピクセル)

色フォーマットの詳細情報についてはウィキペディアを参照してください。

Note: UVC 仕様においてRGB フォーマットはサポートされません。しかし、マイクロソフトは Windows 向けにRGB888 やRGB565 を含む様々な種類の他のフォーマットをサポートするためにUVC 仕様の拡張版を提供しています。それらはビデオストリーミング形式のディスクリプタに適切なGUID を含めるよう指定されます。詳細については、Microsoft のWeb ページのメディア タイプを参照してください。

Note: モノクロ画像がUVC の仕様の一部としてサポートされませんが、8 ビットRAW データをY 値に使用し、すべてのU とV 値を0x80 にセットすることで、8 ビットのモノクロ画像をYUY2 フォーマットで表示できます。これはイメージ センサー(ISP) によって処理されます。

4.3.2.4 UVC ビデオデータヘッダ

UVC クラスは非圧縮ビデオペイロード用に 12 バイトヘッダを必要とします。ヘッダは転送されるイメージデータの特性を記述します。例えば、イメージセンサーコントローラー (CX3) がフレームごとにトグルする「新フレーム」ビットを含みます。

CX3 コードはヘッダ内でエラービットをセットして、現行フレームのストリーミングに問題が発生したことも示せます。この UVC データヘッダはすべての USB 転送に必要です。詳細情報については UVC 仕様を参照してください。

Table 5 に UVC ビデオデータヘッダのフォーマットを示します。

Table 5 UVC ビデオデータヘッダフォーマット

バイトオフセット	フィールド名	説明
0	HLF	HLF (ヘッダ長フィールド) は、ヘッダの長さをバイト単位で示す
1	BFH	BFH (ビットフィールド ヘッダ) は、イメージデータのタイプ, ビデオストリームの状態, 他のフィールドの有無を示す
2〜5	PTS	PTS (プレゼンテーションタイムスタンプ) は、ソース クロック時間をネイティブなデバイスクロック単位で示す
6〜11	SCR	SCR (ソースクロックリファレンス) は、システム時刻および USB のフレームの開始 (SOF) トークンカウンターを示す

HLF の値は常に 12 です。PTS と SCR フィールドは任意です。このファームウェアの例ではこれらフィールドは 0 になります。

ビットフィールド ヘッダ (BFH) はフレームの最後で変化する値を保持します。Table 6 に UVC ビデオデータヘッダの一部である BFH のフォーマットを示します。

4 USB ビデオクラス (UVC)

Table 6　ビットフィールドヘッダ (BFH) のフォーマット

バイトオフセット	フィールド名	説明
0	FID	FID (フレーム識別子) ビットは各イメージフレーム開始境界でトグルし、イメージフレームの残りの期間で変わらない
1	EOF	EOF (フレームの終了) ビットはビデオの終了を示し、イメージフレームに属す最後の USB 転送でのみセット
2	PTS	PTS (プレゼンテーションタイム スタンプ) ビットは UVC ビデオデータヘッダ内の PTS フィールドの有無を示す (1=存在している)
3	SCR	SCR (ソースクロックリファレンス) ビットは UVC ビデオデータヘッダ内の SCR フィールドの有無を示す (1=存在している)
4	RES	予約済み、0 にクリア
5	STI	STI (スティルイメージ) ビットはビデオサンプルが静止画像であるかを示す
6	ERR	ERR (エラー) ビットはデバイスストリーミングのエラーを示す
7	EOH	EOH (ヘッダの終了) ビットは、セットされた場合、BFH フィールドの終了を示す

Figure 18 に、このアプリケーションでヘッダがどのようにビデオデータに追加されるかを示します。12 バイト ヘッダは各 USB バルク転送に追加されます。ここでは、各 USB 転送は合計 16 バルクパケットを持ちます。USB 3.2 Gen1 バルクパケットサイズは 1024 バイトです。

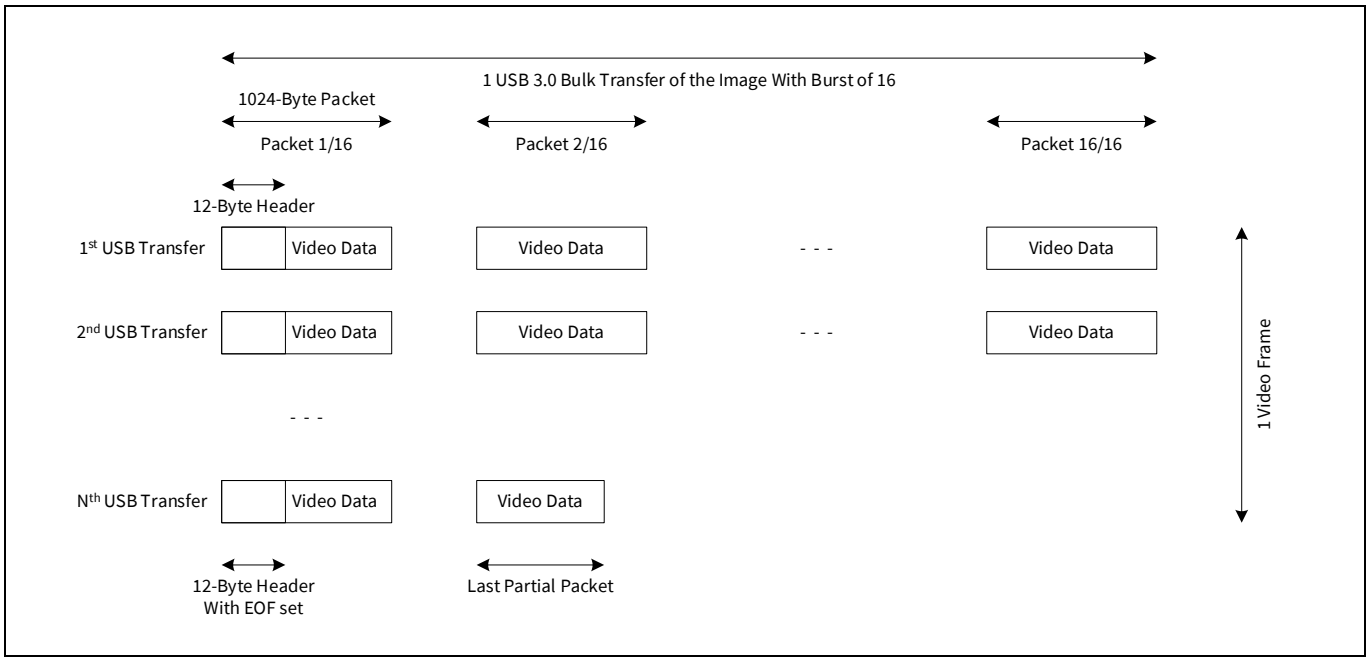

Figure 18　UVC ビデオデータ転送

5 CX3 MIPI 設定ツール

インフィニオンは、CX3 プロジェクトを迅速に生成するための CX3 Configuration Tool を提供しています。このツールは、FX3 SDK の一部として EZ-USB™ Suite IDE に同梱されています。本ツールは、入力されたイメージセンサーのパラメーターに基づいて、CX3 の MIPI CSI-2 コントローラーの設定を行うために使用されます。また、ユーザーの入力内容に応じて、UVC クラスディスクリプタ, UVC リクエストの処理, およびビデオストリーミング用 DMA 設定を含む、完全な CX3 ファームウェアプロジェクトを生成することも可能です。

ここでは、CX3 MIPI Configuration tool version 2.0 (FX3 SDK version 1.3.5 に同梱) の詳細について説明します。

5.1 ツールの起動

EZ-USB™ Suite を開き、**Create New CX3 Configuration** ボタンをクリックします (Figure 19 を参照)。

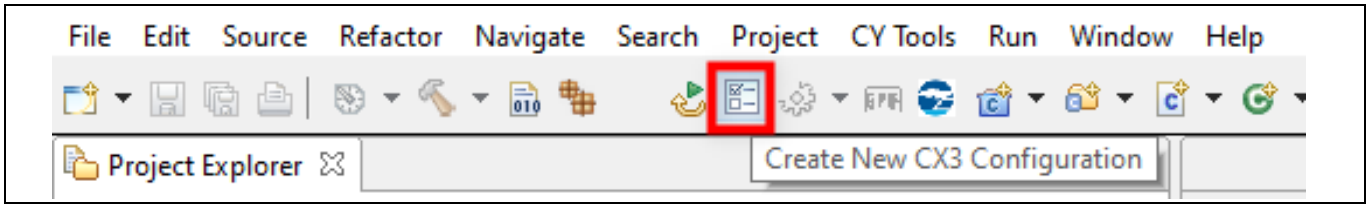

Figure 19 CX3 Configuration Tool のショートカット

CX3 Configuration Tool v2.0 には、新しい CX3 設定を作成するための以下のオプションがあります (Figure 20 を参照)。

Table 7 CX3 Configuration Tool v2.0

オプション	説明
Create New MIPI Receiver Configuration Project	CX3 プロジェクトを作成するためには、このオプションを選択します。
Create a Configuration with Basic Settings	基本設定で CX3 プロジェクトを作成するためには、このオプションを選択します。
Select a Pre Defined Configuration	サポートされているセンサー用の CX3 プロジェクト作成を効率的に開始するためには、このオプションを選択します。
Select an User Defined Configuration	ツール内の **Save as user defined configuration** オプションを使用すると、センサー設定を作成し、CX3 MIPI 設定ツールに追加できます。
Add Only MIPI Receiver Configuration File	すでに CX3 プロジェクトがあり、CX3 MIPI レシーバー設定ファイルのみを生成する必要がある場合は、このオプションを使用します。

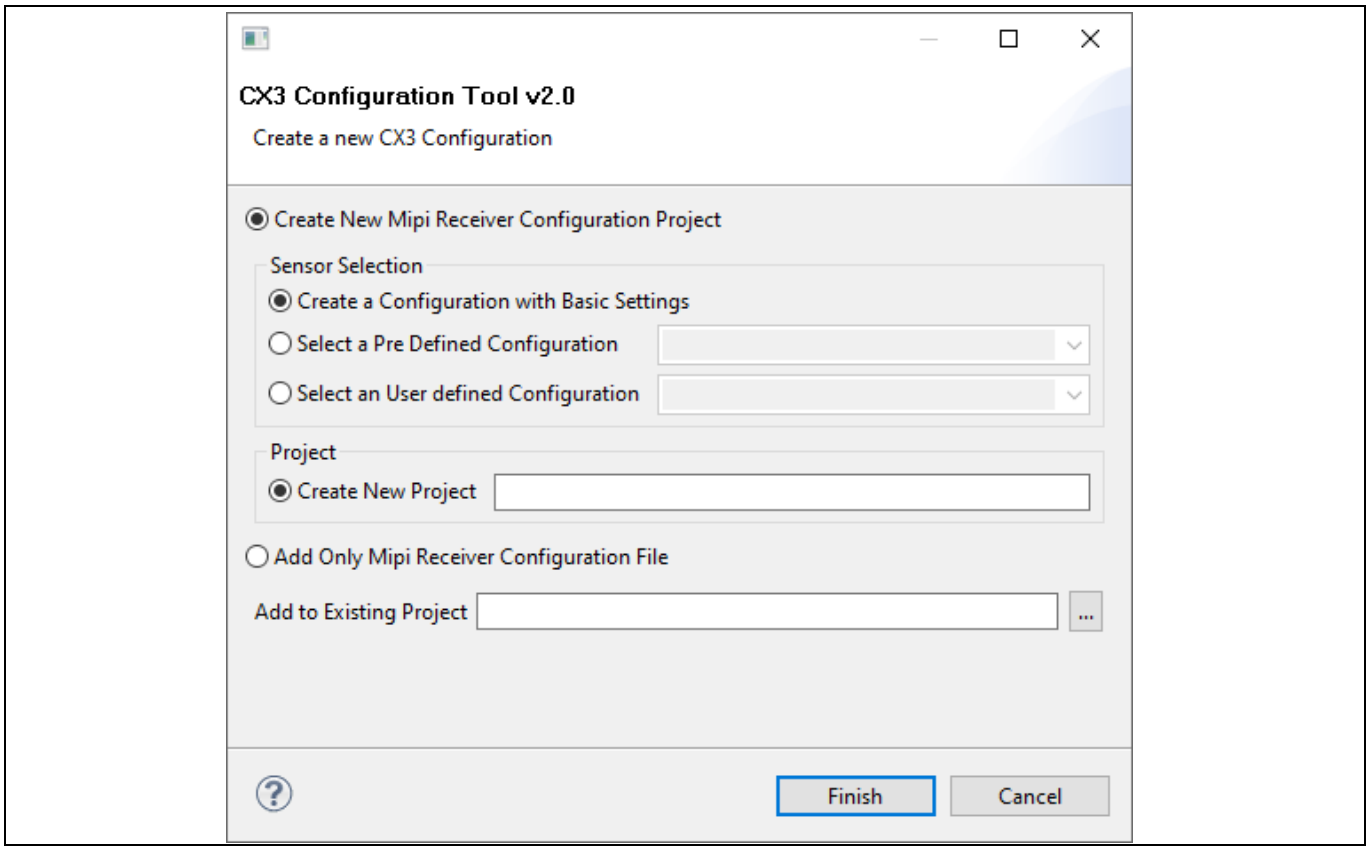

Figure 20　CX3 Configuration Tool のスタートページ

5.2 CX3 Configuration tool へのセンサー詳細の追加

CX3 Configuration tool の **Image Sensor Configuration** タブでは、CX3 の MIPI レシーバーに必要なセンサー関連のすべての情報を設定します。Figure 21 に、このタブの各要素を示します。

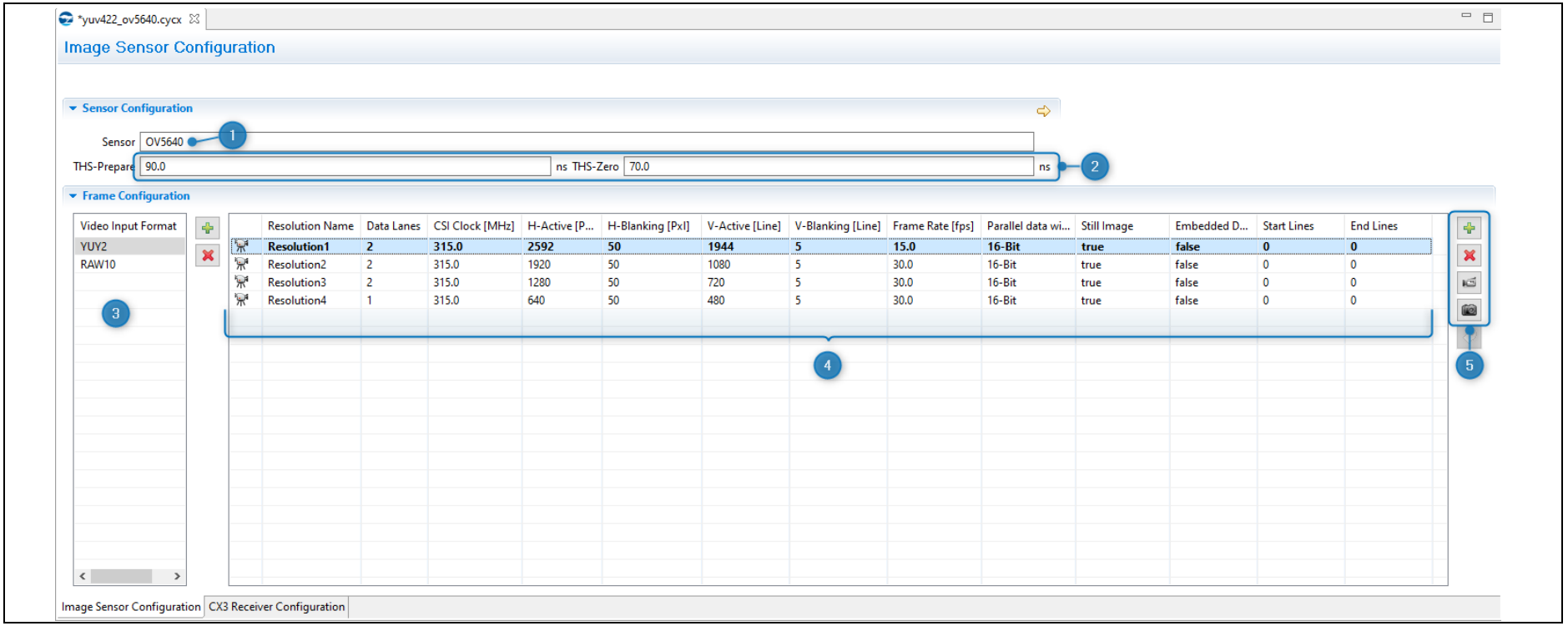

Figure 21 CX3 Configuration Tool のセンサー設定ページ

5 CX3 MIPI 設定ツール

Image Sensor Configuration タブには、以下の要素が含まれます。

1. **Sensor name**: センサー名を入力します。この文字列は、生成されるファームウェア内のセンサー名の代わりにツールで使用されます。
2. **THS-Prepare および THS-Zero**: これらのフィールドに、センサーの THS-Prepare および THS-Zero の値をナノ秒 (ns) 単位で入力します [1]。ツールはこれらの値を使用して、CX3 の MIPI CSI-2 レシーバーの THS-Settle または PHY Time Delay パラメーターを設定します。
3. **Video Input formats**: CX3 の設定に使用する入力ビデオ形式をリストから選択します。

CX3 ツールは、以下のビデオ形式をサポートします。

Table 8　ビデオ形式

入力ビデオ形式	詳細
YUY2	ピクセルあたり 16 ビット。センサーからのピクセルは P [15:0]として配置。Y1, U1, Y2, および V1 の順に配置
UYVY	ピクセルあたり 16 ビット。センサーからのピクセルは P [15:0]として配置。U1, Y1, V1, および Y2 の順に配置
YVYU	ピクセルあたり 16 ビット。センサーからのピクセルは P [15:0]として配置。Y1, V1, Y2, および U1 の順に配置
YUV422_10	ピクセルあたり 10 ビット。センサーからのピクセルは P [9:0]として配置
RAW8	ピクセルあたり 8 ビット
RAW10	ピクセルあたり 10 ビット
RAW12	ピクセルあたり 12 ビット
RAW14	ピクセルあたり 14 ビット
RGB565_MODE0	ピクセルあたり 24 ビット。 センサーからのピクセルは、P {2'b0, R[4:0] , 3'b0 , G[5:0] , 2'b0 , B[4:0] , 1'b0} として配置。
RGB565_MODE1	ピクセルあたり 24 ビット。 センサーからのピクセルは、P {3'b0, R[4:0] , 2'b0 , G[5:0] , 3'b0 , B[4:0] } として配置。
RGB565	ピクセルあたり 16 ビット。 センサーからのピクセルは、P {R[4:0] , G[5:0] , B[4:0] } として配置。
RGB666_MODE_0	ピクセルあたり 24 ビット。 センサーからのピクセルは、P {2'b0, R[5:0] , 2'b0 , G[5:0] , 2'b0 , B[5:0] } として配置。
RGB666_MODE_1	ピクセルあたり 24 ビット。 センサーからのピクセルは、P {6'b0 , R[5:0] , G[5:0] , B[5:0] } として配置。
RGB24	ピクセルあたり 24 ビット。 センサーからのピクセルは、P {R[7:0] , G[7:0] , B[7:0] } として配置。
MJPEG	圧縮ビデオ形式
YUY2_SWAP	ピクセル順でバイトスワップを行った YUY2。U1, Y1, V1, および Y2 の順に配置
UYVY_SWAP	ピクセル順でバイトスワップを行った UYVY。U1, Y1, V1, および Y2 の順に配置

4. Table 9 にセンサーパラメーターを記載します。

[1] この情報については、センサーのデータシートを参照するか、センサーベンダーにお問い合わせください。THS-Prepare および THS-Zero は、MIPI D-PHY 仕様で定義されている標準パラメーターです。

5 CX3 MIPI 設定ツール

Table 9　センサーパラメーター

パラメーター	詳細
Data Lanes	解像度に対応する MIPI データレーン数
CSI Clock (MHz)	MIPI クロック周波数 例えば、レーンあたりのデータレートが 600 Mbps の場合、必要な CSI クロックは 300 MHz です。これは、データ転送時に MIPI クロックが DDR モードで動作するとみなされるためです。
H-Active	水平幅の解像度
H-Blanking	1 ラインあたりのブランキングピクセル数
V-Active	垂直高さの解像度
V-Blanking	1 フレームあたりのブランキングライン数
Frame Rate	1 秒あたりのフレーム数 (fps)
Parallel data width	パラレルバス幅。CX3 は 8 ビット, 16 ビット, および 24 ビットのパラレルバスをサポートします。また、ピクセルデータのパッキングやゼロパディングにも対応しています。詳細については、パラレルデータ幅の選択に基づく、入力データ形式から出力ビデオ形式へのマッピングを参照してください。
Still Image	静止画キャプチャ機能を有効にする
Embedded data	センサーからの埋め込みデータ用領域を確保する
Start lines	フレーム先頭の埋め込みデータライン数
End lines	フレーム末尾の埋め込みデータライン数

Note:　イメージセンサーの設定は、イメージセンサーのベンダーから入手してください。

Note:　解像度の幅またはラインサイズ (バイト単位) は、4 バイトの倍数である必要があります。この条件が満たされない場合、CX3 の CSI-2 MIPI コントローラーブロックは、ラインデータに追加のパディングバイトを付加します。

5. ボタン

Table 10　ボタン

ボタン	詳細
	ビデオ形式またはビデオ解像度を追加する
	ビデオ形式またはビデオ解像度を削除する
	現在の解像度を最適なビデオ解像度として選択する (UVC フォーマットディスクリプタにおいて最適な解像度として設定されます)
	現在の解像度を最適な静止画解像度として選択する (UVC 静止画キャプチャディスクリプタにおいて最適な解像度として設定されます)

5.2.1 パラレルデータ幅の選択に基づく、入力データ形式から出力ビデオ形式へのマッピング

CX3 の CSI-2 コントローラーは、パラレル出力において、複数のピクセルをパッキングするか、あるいは各ピクセルにゼロパディングを施すかを選択する機能を提供します。ピクセルのパッキングは、CX3 の 24 ビットパラレルバス幅を最大限に活用するのに役立ちます。一方、パディングは、ピクセルごとのバス幅に合わせるためにピクセルデータにゼロを付加するものです。

例:

入力ビデオ形式が RAW8 での入力ビデオ解像度 1920 × 1080 (Hactive × Vactive) を考えます。各ピクセルのサイズは 8 ビットです。その結果、入力フレームサイズは Hactive × Vactive × 8 ビットとなります。

RAW8 は、あらゆる標準的な UVC アプリケーションで画像を表示できるように、CX3 ツールによって YUY2 出力形式 (ピクセルあたり 16 ビット) にマッピングされます。

この形式において、CX3 ツールで選択可能なパラレルバス幅のオプションは以下のとおりです。

選択した項目	出力
8-bit	CX3 は 1 クロックあたり 8 ビットを出力します。 出力ピクセルサイズは 16 ビット (YUY2) であるため、センサーからの入力フレームサイズと CX3 からの出力フレームサイズを整合させるには、Hactive の値を Hactive × 0.5 に調整する必要があります。その結果、解像度は (1920/2) × 1080 × 16 となります。
16-bit pack	CX3 は 1 クロックあたり各 8 ビットのピクセルを 2 つ出力します。 出力ピクセルサイズは 16 ビット (YUY2) であるため、センサーからの入力フレームサイズと CX3 からの出力フレームサイズを整合させるには、Hactive の値を Hactive × 0.5 に調整する必要があります。その結果、解像度は (1920/2) × 1080 × 16 となります。
16-bit zero pad	CX3 は 1 クロックあたり 16 ビットのピクセルを 1 つ出力します (8 ビットのピクセルデータと 8 ビットのゼロデータ)。 この場合、入出力のピクセルサイズがいずれも 16 ビット (YUY2) であるため、解像度の調整は行われません。その結果、解像度は 1920 × 1080 × 16 となります。
24-bit pack	CX3 は 1 クロックあたり各 8 ビットのピクセルを 3 つ出力します。 出力されるピクセルサイズは 16 ビット (YUY2) であるため、センサーからの入力フレームサイズと CX3 からの出力フレームサイズを整合させるには、Hactive の値を Hactive × 0.5 に調整する必要があります。その結果、解像度は (1920 / 2) × 1080 × 16 となります。
24-bit zero pad	CX3 は 1 クロックあたり 24 ビットのピクセルを 1 つ出力します (8 ビットのピクセルデータと 16 ビットのゼロデータ)。 出力ピクセルサイズは 16 ビット (YUY2) であるため、入力と出力のフレームサイズを合わせるには、Hactive を Hactive × (24/16) に調整する必要があります。 その結果、解像度は (1920 × 1.5) × 1080 × 16 となります。

各種入力ビデオ形式とパラレルデータ幅設定の対応関係を、Table 11 に示します。

Note: この対応関係に関して、ユーザーが何らかの操作を行う必要はありません。CX3 Configuration Tool は、入力ビデオ形式およびパラレルバス幅の設定に基づき、出力ビデオ解像度を自動的に調整します。

Table 11　各種入力ビデオ形式とパラレルデータ幅設定との対応関係

入力ビデオ形式	ピクセルあたりの入力ビット数	パラレルデータ幅	結果としてのピクセルあたりのビット数	入力フレームサイズ	出力ビデオ形式 (UVC ディスクリプタでのビデオ形式)	ピクセルあたりの出力ビット数	出力フレームサイズ	入力および出力のフレームサイズを合わせるための調整	例: 1920 x 1080 入力時の出力解像度
RAW8	8	8-bit	8	Hactive × Vactive × 8	YUY2	16	Hactive × Vactive × 16	Hactive × 0.5	960 × 1080
		16-bit pack	8	Hactive × Vactive × 8				Hactive × 0.5	960 × 1080
		16-bit zero pad	16	Hactive × Vactive × 16				Hactive	1920 × 1080
		24-bit pack	8	Hactive × Vactive × 8				Hactive × 0.5	960 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive ×1.5	2880 × 1080
RAW10	10	16-bit pack	10	Hactive × Vactive × 10				Hactive × 0.625	1200 × 1080
		16-bit zero pad	16	Hactive × Vactive × 16				Hactive	1920 × 1080
		24-bit pack	10	Hactive × Vactive × 10				Hactive × 0.625	1200 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive × 1.5	2880 × 1080
RAW12	12	16-bit pack	12	Hactive × Vactive × 12				Hactive ×0.75	1440 × 1080
		16-bit zero pad	16	Hactive × Vactive × 16				Hactive	1920 × 1080
		24-bit pack	12	Hactive × Vactive × 12				Hactive ×0.75	1440 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive ×1.5	2880 × 1080
RAW14	14	16-bit pack	14	Hactive × Vactive × 14				Hactive × 0.875	1680 × 1080
		16-bit zero pad	16	Hactive × Vactive × 16				Hactive	1920 × 1080
		24-bit pack	14	Hactive × Vactive × 14				Hactive × 0.875	1680 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive ×1.5	2880 × 1080
RGB 565 mode 0	24	24-bit	24	Hactive × Vactive × 24	RGB565	16	Hactive × Vactive × 16	Hactive ×1.5	2880 × 1080
RGB 565 mode 1		24-bit	24	Hactive × Vactive × 24				Hactive ×1.5	2880 × 1080
RGB 565	16	16-bit	16	Hactive × Vactive × 16				Hactive	1920 × 1080
RGB 666 mode 0	24	24-bit	24	Hactive × Vactive × 24	RGB888	24	Hactive × Vactive × 24	Hactive	1920 × 1080
RGB 666 mode 1		24-bit	24	Hactive × Vactive × 24				Hactive	1920 × 1080
RGB 24		24-bit	24	Hactive × Vactive × 24				Hactive	1920 × 1080

入力ビデオ形式	ピクセルあたりの入力ビット数	パラレルデータ幅	結果としてのピクセルあたりのビット数	入力フレームサイズ	出力ビデオ形式 (UVC ディスクリプタでのビデオ形式)	ピクセルあたりの出力ビット数	出力フレームサイズ	入力および出力のフレームサイズを合わせるための調整	例: 1920 x 1080 入力時の出力解像度
UYVY	16	16-bit	16	Hactive × Vactive × 16	UYVY	16	Hactive × Vactive × 16	Hactive	1920 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive × 1.5	2880 × 1080
		24-bit pack	16	Hactive × Vactive × 16				Hactive	1920 × 1080
YUY2	16	16-bit	16	Hactive × Vactive × 16	YUY2			Hactive	1920 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive × 1.5	2880 × 1080
		24-bit pack	16	Hactive × Vactive × 16				Hactive	1920 × 1080
YVYU	16	16-bit	16	Hactive × Vactive × 16	YVYU			Hactive	1920 × 1080
		24-bit pack	16	Hactive × Vactive × 16				Hactive	1920 × 1080
		24-bit zero pad	24	Hactive × Vactive × 24				Hactive × 1.5	2880 × 1080
YUV422_10	10	16-bit pad	16	Hactive × Vactive × 16	YUY2			Hactive	1920 × 1080
MJPEG	16	16-bit	16	Hactive × Vactive × 16	MJPEG			Hactive	1920 × 1080
		24-bit	16	Hactive × Vactive × 16				Hactive	1920 × 1080

Note: このマッピングを変更する場合、あるいは YUY2 として報告する代わりに特定のフォーマット (例: RAW8) を直接パススルーさせる場合は、cycx3_uvcdscr.c ファイルを編集し、カスタム UVC ホストアプリケーションを使用して受信したピクセルを表示できます。

5.3 CX3 MIPI レシーバーの設定

CX3 設定ツールを使用して、完全な CX3 プロジェクト、または (既存のプロジェクトに追加可能な) MIPI レシーバー設定のみを生成します。

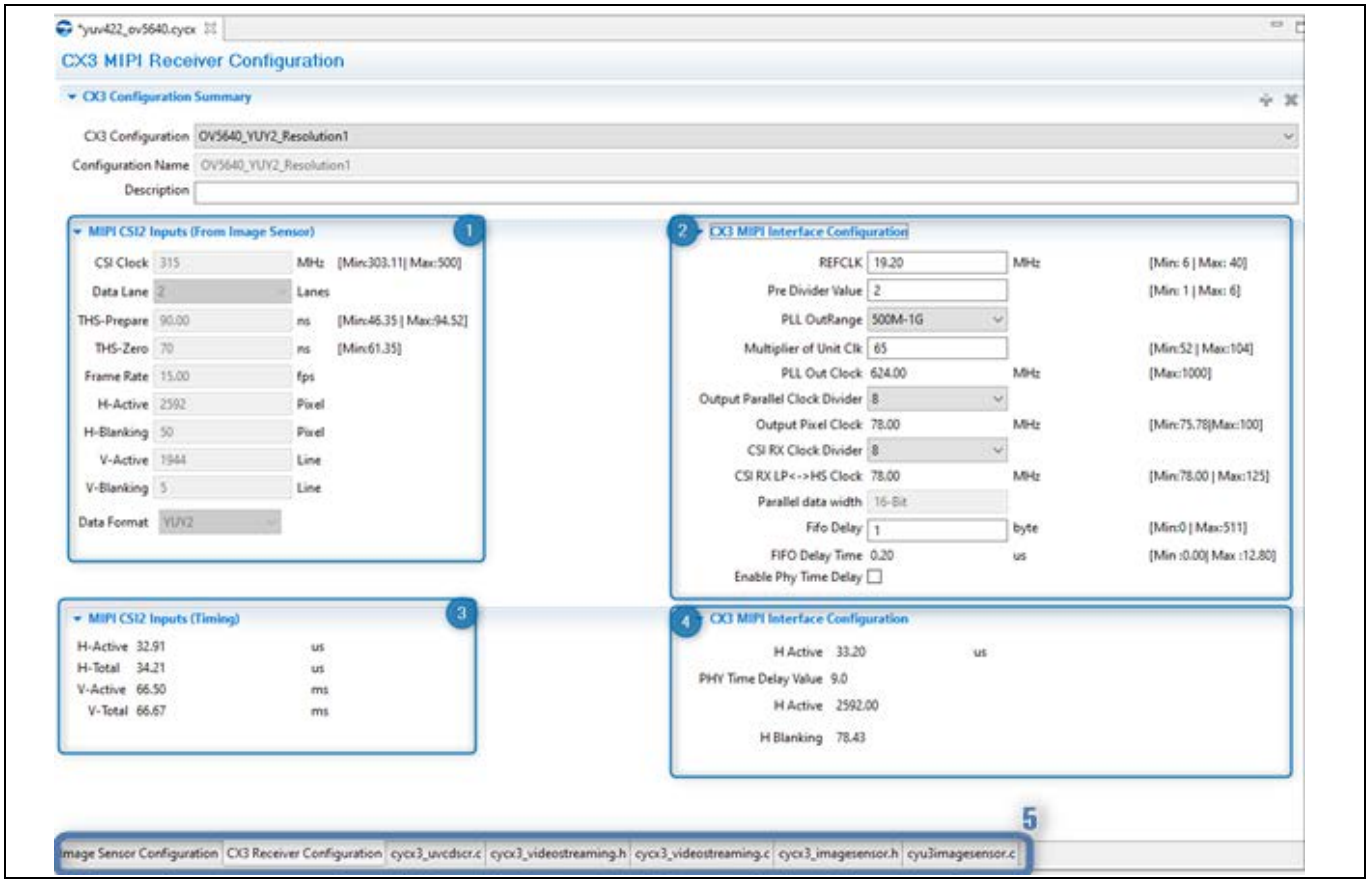

Figure 22 CX3 MIPI レシーバー設定ツール

MIPI Receiver Configuration タブ (Figure 22) 内のフィールドは以下のとおりです。

1. **MIPI CSI2 Inputs (From Image Sensor)**: MIPI CSI2 Input タブのパラメーター (CSI クロック, データレーン, THS-Prepare, THS-Zero, フレームレート, H-Active, H-Blanking, V-Active, V-Blanking, データフォーマットなど) は、Image Sensor Configuration タブから自動的に入力されます。これらは CSI-2 レシーバーインターフェースの入力パラメーターです。
   例えば、Figure 22 は、CSI クロック 315 MHz, H-Blanking 50 ピクセル, V-Blanking 5 ラインで YUY2 (2592 x 1944) をストリーミングする CSI-2 センサインターフェースを示します。MIPI コントローラーの設定パラメーターの妥当性確認には、これらの入力に基づく計算が使用されるため、これらの値は実際の CSI-2 インターフェースのパラメーターと一致している必要があります。
   このセクションのすべてのパラメーター (データフォーマットを含む) は、計算のみに使用されます。このセクションで指定されるデータフォーマットは、ストリーミングされる形式の 1 ピクセルあたりのビット数を特定するために使用されます。

2. **CX3 MIPI Interface Configuration**: CX3 MIPI Interface Configuration タブのパラメーター (REFCLK, Pre Divider Value, PLL OutRange, Multiplier of Unit Clk, PLL Out Clock, Output Parallel Clock Divider, Output Pixel Clock, CSI RX Clock Divider, CSI RX LP <-> HS Clock, Parallel data width, Fifo Delay, FIFO Delay Time, Phy Time Delay チェックボックスなど) を使用して、MIPI CSI-2 コントローラー内で使用される 3 つのクロック (PLL クロック, CSI RX LP <-> HS クロック, およびパラレル出力ピクセルクロック) を設定します。
   これら 3 つのクロックの詳細については、以下を参照してください。
   - CX3 TRM の MIPI CSI-2 Block Clocks セクション
   - Analysis of CX3 clocking parameters – KBA226758

   Output Pixel Clock (ツールで指定された範囲内である必要があります), Data format (パラレル出力インターフェースのデータバス幅を決定します), および FIFO Delay は、CX3 内の MIPI コントローラーが正常に動作するかどうかを決定する重要なパラメーターです。
   サポートされているデータフォーマットと、各フォーマットでサポートされるパラレル出力データバス幅については、CX3 TRM の CX3 MIPI CSI-2 Stream Formats セクションを参照してください。
   FIFO Delay パラメーターを設定すると、各ラインの開始時にパラレル出力に遅延が追加されます。FIFO 遅延は、設定された出力パラレルピクセルクロックが、ツールが推奨する「最小」出力パラレルピクセル PCLK 値よりも高い場合に必要となります。Figure 23 に FIFO 遅延の概念を示します。ただし、FIFO 遅延の最小値と最大値は、MIPI の H-active 時間と出力パラレルの H-active 時間の単純な引き算によって計算されるわけではないことに注意が必要です。MIPI コントローラー内にはラインバッファが存在し、FIFO 遅延時間を計算する際にはこれを考慮に入れる必要があります。この調整はツールによって自動的に行われます。

Note: 現在の MIPI ラインデータは、次の MIPI ラインデータが到着する前に、パラレルインターフェース経由ですべて送信し終える必要があります。この規則が守られない場合、ビデオフレームに垂直方向の分断が生じることがあります。したがって、FIFO 遅延の設定は慎重に行ってください。

このツールは、THS-Prepare および THS-Zero の値フィールドを使用して、CSI-2 MIPI コントローラーの PHY Time Delay パラメーターを算出します。CX3 MIPI Interface Configuration タブの「Enable Phy Time Delay」チェックボックスを選択することで、API 呼び出しの有効/無効を切り替えることができます。

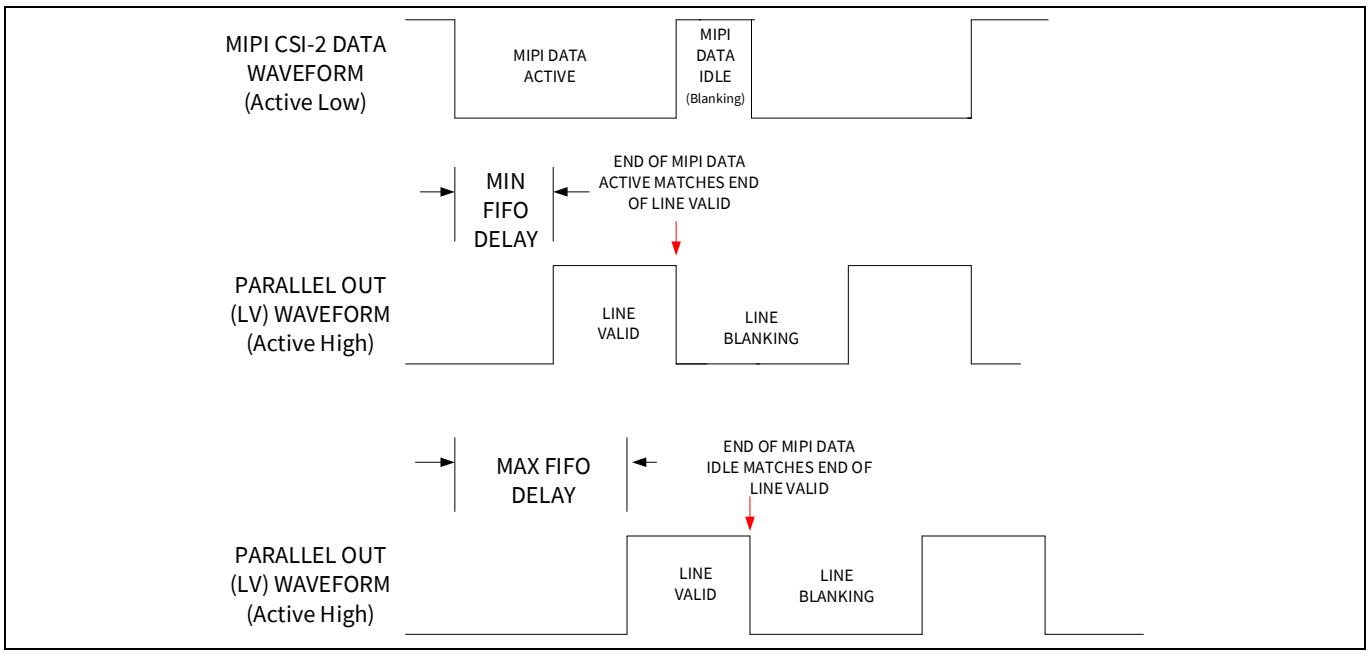

Figure 23 FIFO 遅延の概念

3. **MIPI CSI2 Inputs (Timings)**: CSI-2 入力パラメーターに基づいて算出されたタイミング (**H-Active**, **H-Total**, **V-Active**, および **V-Total** など) です。
4. **CX3 MIPI Interface Configuration**: MIPI CSI-2 インターフェース設定に基づいて算出されたタイミング (**H-Active**, **PHY Time Delay Value**, **H-Active**,および **H-Blanking** など) です。
5. **Save Modified files**: これらのファイルには、GUI 上で行った変更に基づくソースコードの修正内容が反映されています。各ファイルのタブをクリックし、(Ctrl+S) を使用して保存してください。

Note: *変更内容をプロジェクトの元のファイルに確実に反映させるためには、この手順が必須です。*

別のセンサーを使用して別の形式でストリーミングを行うように設定するには、FX3 SDK で新規プロジェクトを作成し、センサー設定ページ (Figure 22 参照) で入力ビデオ形式を選択してください。RGB 形式のストリーミングも、ピクセルあたりのバイト数によっては YUV 形式と同様の手順で行うことができます。RGB 形式のストリーミングに関する詳細については、FX3 SDK 内の他のサンプルプロジェクトを参照してください。異なる解像度での MJPEG 形式のストリーミングや、CX3 デバイスと ISP (画像信号プロセッサ) の接続については、THine Solutions 社の CX3 ベースの THEIA-CAM のソースコードを参照してください。なお、CX3 ベースの THEIA-CAM のファームウェアソースを入手するには、THine ISP 13MP リファレンスデザインキット (RDK) を購入する必要があります。

OV5640 センサーを使用した MJPEG ファームウェアのサンプルが必要な場合は、OmniVision 社と NDA (秘密保持契約) を締結し、テクニカルサポートの問い合わせ (ケース) を作成してください。

6 CX3 ファームウェア

本アプリケーションノートは FX3/CX3 SDK に添付されるサンプルプロジェクト cycx3_uvc_ov5640 を使用します。ファームウェアは、まず CX3 の CPU を起動してその I/O を設定します。その後、ThreadX RTOS を開始するために CyU3PKernelEntry 関数を呼び出します。RTOS は自ら初期化し、いくつかの内部スレッドを作成した後、CyFxApplication Define を呼び出してユーザーにアプリケーション固有のスレッドを作成させます。

この例では 2 つのアプリケーションスレッドが作成されます。uvcAppThread と uvcMipiErrorThread です。RTOS は、これらのアプリケーションスレッドを実行するためにリソースを割り当て、2 つのアプリケーションスレッド関数 (CyCx3UvcAppThread_Entry と CyCx3UvcMipiErrorThread) のそれぞれの実行をスケジュールします。Figure 24 に基本プログラム構造を示します。

本アプリケーションノートで記述するサンプル ファームウェアには、OV5640 イメージセンサーを使用した基本的な画像ストリーミング機能だけが実装されます。別のセンサーを使用する場合、または OV5640 よりも多くの機能を使用する場合は、それに対応するよう cycx3_uvc.c ファイルを修正する必要があります。Table 12 は各モジュールに実装するコードモジュールと関数をまとめたものです。

Table 12 サンプルプロジェクトのファイル

ファイル	説明
cyfxtx.c	変更は不要。このファイルをそのままプロジェクトに使用 これは、RTOS と CX3 API ライブラリがメモリマッピングに使用する定数および CX3 API ライブラリがメモリ管理に使用する関数を含む
cycx3_uvc.c	UVC アプリケーションの主なソースファイル。輝度や PTZ などの制御をサポートするためにコードを修正する際、および異なるビデオストリーミングモードへのサポートを追加するために修正する際には変更が必要 以下の関数を含む: • *main：CX3 デバイスを初期化し、キャッシュをセットアップし、CX3 I/O を設定し、RTOS カーネルを起動* • *CyFxApplicationDefine: RTOS が実行する 2 つのアプリケーション スレッドを定義* • *CyCx3AppThread_Entry: この関数は最初のアプリケーション スレッドにより実行される。CX3 の内部ブロック用の初期化関数を呼び出し、デバイスをエニュメレートして、デバイスの一時停止とフレームの完了タスクを処理* • *CyCx3AppMipiErrorThread: この関数は 2 番目のアプリケーション スレッドにより実行される。MIPI CSI-2 コントローラー内のエラーを示すイベントを待った後、CyU3PMipicsiGetErrors でそれを読み出す* • *CyCx3AppDebugInit: デバッグ メッセージのプリント用の CX3 UART ブロックを初期化* • *CyCx3AppInit: CX3 の GPIO ブロック, プロセッサブロック, または PIB (GPIF II は PIB の一部), I2C/CCI インターフェース, MIPI CSI-2 Rx インターフェース, およびセンサー (SuperSpeed モードでは 1080p 30fps に設定) を初期化。また、エニュメレートのためには USB ブロックを、USB 転送のためにエンドポイントコンフィギュレーションメモリを初期化するほか、2 つの GPIF II ソケットから USB ソケットにデータを転送するためには DMA チャネル コンフィギュレーションを作成* • *CyCx3AppGpifCB: GPIF II ステート マシンから生成される CPU 割込みを処理*

6 CX3 ファームウェア

ファイル	説明
	• *CyCx3AppDmaCallback: CX3 からホストへの出力ビデオ データを継続的にトラック。イメージ データにヘッダを追加するほか、最初のアプリケーション スレッドに通知してフレームの完了を示せる* • *CyCx3AppUSBSetupCB: ホストが送信するすべての制御要求を処理し、UVC 固有の要求がホストから受け取られたことを示すイベントを設定し、ストリーミングが停止する時点を検出。ビデオストリーミングの開始と停止に必要となる、UVC クラス固有の PROBE と COMMIT 制御要求も処理* • *CyCx3AppUSBEventCB: 一時停止, ケーブル切断, リセット, 再開などの USB イベントを処理* • *CyCx3AppErrorHandler: エラー処理関数。必要に応じてエラー処理を実施するために使用されるプレースホルダ関数* • *CyCx3AppAddHeader: 有効なストリーミング中に UVC ヘッダをビデオデータに追加*
cycx3_uvcdscr.c	UVC アプリケーションの USB エニュメレーションディスクリプタを含む。フレームレート, 画像解像度, ビット深度, またはサポートされているビデオ制御を変更する場合は、このファイルも変更が必要。UVC 仕様に必要な詳細情報が含まれている
cycx3_uvc.h	フレームカウントと MIPI エラースレッドのデバッグプリントをオン/オフにすることでアプリケーション動作を修正するスイッチを含む。 cycx3_uvc.c と cycx3_uvcdscr.c ファイルに共通に使用される定数を含む
cyfx_gcc_startup.s	CX3 CPU の起動コードを含むアセンブリソースファイル。スタックと割込みベクタをセットアップする関数を含む。 変更は不要。
cyu3imagesensor.c	イメージセンサーの設定ルーチン, 構造体, I2C レジスタアドレス, および値を格納。
cyu3imagesensor.h	イメージセンサー関連機能の宣言を含む
cycx3_videostreaming.c	MIPI 設定用のイメージセンサーのビデオ解像度およびデータ構造に関するルーチンを含む
cycx3_videostreaming.h	MIPI 関連のマクロ, UVC マクロ, およびビデオ解像度の宣言を含む
cycx3gpifwaveform.h	GPIF 設定用のデータ構造を含む

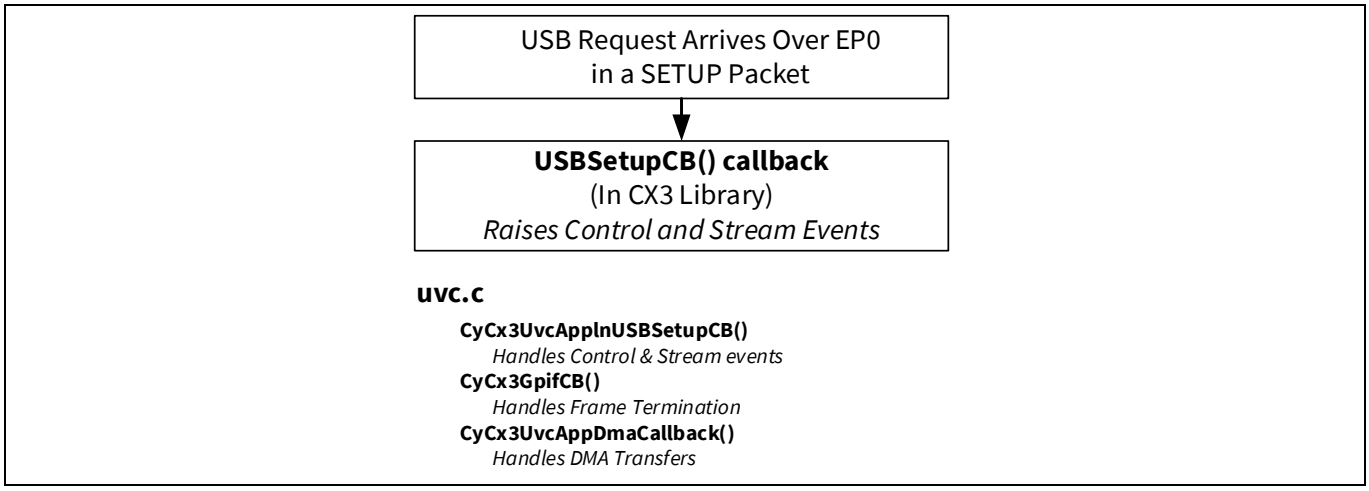

Figure 24　高レベルカメラプロジェクトの構造

6.1　アプリケーションスレッド

2 つのアプリケーションスレッドが並列機能を有効にします。

uvcAppThread (アプリケーションスレッド) は CX3 ペリフェラルの初期化を行い、2 つの内部イベントを処理します: *USB サスペンドと DMA チャネルのリセット*。

uvcMipiErrorThread は MIPI CSI-2 のエラーを監視し、エラーカウントを返します。

6.1.1　USB サスペンドイベント

CX3 がホストにより一時停止状態にされた時、USB サスペンドイベント (CX3_USB_SUSP_EVENT_FLAG) がトリガーされ下記を実行します。

1. MIPI CSI-2 コントローラーのクロックを無効にし、インターフェースを低消費電力モードにします。
2. イメージセンサーをスリープ状態にします。
3. USB バスの動作を復帰ソースにして CX3 コアを低消費電力サスペンドモードに移行します。

USB バスで動作が検出された場合、デバイスは復帰します。復帰後、イメージセンサーは電源投入され、MIPI CSI-2 コントローラーは再びアクティブになります。

6.2　DMA チャネルリセットイベント

フレーム転送に時間が掛かりすぎて完了できない場合、古いデータが消されビデオストリームが再起動されます。500 ms タイマー (ウォッチドッグとして使われる) が切れるとセットされた CX3_DMA_RESET_EVENT フラグを使ってこれが行われます。

このタイマーは、フルフレームがホストに転送される度にリセットされます。また、1 フレームが DMA バッファに長時間滞留した時にのみ切れます。ホストへの DMA バッファの転送に失敗したら、ビデオストリームも再起動されます (DMA チャネルがシーケンスから外れてしまう結果となる可能性があります)。

CX3_ERROR_THREAD_ENABLE プリプロセッサマクロが有効になったら、CSI-2 エラーを監視するために他のスレッド (uvcMipiErrorThread) が作られます。

このスレッドのエントリ関数 (CyCx3UvcMipiErrorThread) は MIPI CSI-2 コントローラーからエラー件数を取得するように周期的に CyU3PMipicsiGetErrors を呼び出します。空白スクリーンが見えたら、エラーカ

ウンターはデバッグ処理のため使用されます (MIPI CSI-2 エラーカウンターで修正する方法の詳細な情報は、トラブルシューティング を参照してください)。

6.3 UVC 要求の処理

CX3 ファームウェアはコントロールエンドポイント (EP0) を介して UVC 固有の制御要求 (UVC 固有の要求 に記述された SET_CUR, GET_CUR, GET_MIN, GET_MAX) を処理します。クラス固有の制御要求は CyCx3AppUSBSetupCB (CB=コールバック) 関数で処理されます。CX3 がこれらいずれかの制御要求を受け取る度に、この関数が受け取ったセットアップデータをデコードして処理します。

ビデオストリーミングインターフェースに直接送信したプローブ制御要求 (ストリーミング要求: プローブとコミット制御を参照) の場合、ホストに該当するプローブ制御構造体を送信します。コミット制御要求が生成されたら、ビデオストリーミングが起動されます。

この例ではビデオコントロール機能に対応してないため、ビデオコントロールインターフェースに送信されたすべての要求 (Get Status 要求以外) (制御要求: 輝度, PTZ, 色相, 彩度, その他を参照) が機能停止します (USB ホストが STALL ハンドシェークを受け取ります)。サンプルが輝度, 露出, PTZ, もしくはいずれか他の制御に対応するなら、ここで処理する必要があります。

6.4 初期化

CyCx3UvcAppThread_Entry 関数は、UART デバッグ処理機能を初期化するために CyCx3AppDebugInit を呼び出し、残りの必要なブロック, DMA チャネル, および USB エンドポイントを初期化するために CyCx3AppInit を呼び出します。

6.5 エニュメレーション

CyCx3AppInit 関数では、CyU3PUsbSetDesc 関数への呼び出しは CX3 を UVC デバイスとしてエニュメレートします。UVC ディスクリプタは cycx3_uvcdscr.c ファイルで定義されます。これらのディスクリプタは、非圧縮 YUY2 フォーマットを使用して、ピクセルあたり 16 ビットを送信する、30 fps の 1920x1080 ピクセルと 60fps の 1280x720 ピクセルの 2 つの解像度のイメージセンサー用に定義されます。これらの設定を変更する必要がある場合は UVC の USB ディスクリプタを参照してください。

6.6 MIPI CSI-2 コントローラーの設定

CyCx3AppInit 関数は最初に MIPI CSI-2 コントローラーにインターフェースする I^2C, GPIO, および PIB ブロックを初期化します。

DMA チャネルが作成された後、GPIF II ステートマシンをロードするために、データバス幅とバッファサイズをパラメーターとして取り入れる CyU3PMipicsiGpifLoad を呼び出します。プロデューサソケットから見て、パラメーターとして渡されるバッファサイズは DMA バッファサイズとなります。つまりロードされる値は以下となります。

dma_buffer_size – (dma_header_size + dma_footer_size)

選定されたバス幅は、CSI-2 を介して送信されているイメージデータのタイプと一致するべきです。各々の対応するイメージデータのタイプに応じて必要とされるバス幅は SDK API ガイド で定義されます。

それで、ステートマシンが起動され、USB ホストによってイメージストリーミングが要求されるまで休止状態を続けます。

ステートマシンはロードされ起動された後、MIPI CSI-2 コントローラーは初期化されます。これは、XRES ピンをアサートしてイメージセンサーをリセット状態にする CyU3PMipicsiInit を呼び出すことによ

って行われます。センサーをリセット状態から復帰させるには、CyU3PMipicsiSetSensorControl 関数を使用して XRES ピンをディアサートします。

MIPI CSI-2 コントローラーが (CyU3PMipicsiSetIntfParams 関数により) 設定されます。この関数はコンフィギュレーションパラメーターを含む CyU3PMipicsiCfg_t 型の構造体を取り込みます。この構造体は、CX3 MIPI 設定ツール によって生成されます。

6.7 イメージセンサーの設定

MIPI CSI-2 インターフェースが設定された後、イメージセンサーを適切に設定する必要があります。これは CyCx3AppInit 関数の終わりの近くで行われます。インフィニオンは OV5640 イメージ センサーを初期化、設定する関数を持つあらかじめ編集されたライブラリを備えます。

他のイメージセンサーの場合、イメージセンサーを設定するのに使える *cyu3imagesensor.c* ファイルがいくつかのヘルパー関数を持ちます。

イメージセンサーは CX3 の I²C マスターブロックを使って設定されます。*cyu3imagesensor.c* ファイル内の SensorWrite2B, SensorWrite, SensorRead2B, および SensorRead 関数は、I²C を介してイメージセンサーのコンフィギュレーションを読み書きするために使用できます。

データをイメージセンサーへ書き込むために関数 SensorWrite2B と SensorWrite は標準 API CyU3PI2cTransmitBytes を呼び出します。データをイメージセンサーから呼び出すために関数 SensorRead2B と SensorRead は標準 API CyU3PI2cReceiveBytes を呼び出します。これらの API の詳細は、FX3 SDK API ガイドを参照してください。

6.8 ビデオストリーミングの開始

VLC Player, e-CAMView, Media Player Classic, または VirtualDub などの USB ホストアプリケーションは、USB インターフェースと USB alternate setting (代替設定) の組合せを、ビデオストリーミングするもの (通常 Interface 0 Alternate setting 1 と呼ばれる) に設定し、PROBE/COMMIT 制御要求を送信することを目的として UVC ドライバの最上部に配置されます。これはビデオデータのストリーミングをまもなく開始するようホストによる指示です。ストリーミングイベントにおいて、USB ホストアプリケーションは CX3 に画像データの要求を開始します。CX3 はその後イメージ センサーから USB 3.2 Gen1 ホストへ画像データの送信を開始するものとします。

CX3 がストリーミングイベントを受信する場合、USB イベントのコールバック (CyCx3cAppUSBSetupCB) は、受信したフレーム インデックスに従ったイメージ センサーと MIPI CSI-2 インターフェースを設定します。コンフィギュレーションパラメーターはホストが要求した解像度とフレームレートに依存します。これはフレーム設定の選択と切り替えで説明します。

GPIF II ステート マシンはホストがイメージ データを要求するまで動作する必要がないため、一時停止します。ストリーミングイベントが届くと、CyCx3UvcStart 関数が呼び出されます。これは、ステートマシン (CyU3PGpifSMControl を通じて) を再開し、MIPI CSI-2 インターフェース (CyU3PMipicsiWakeup を通じて) を復帰させ、イメージ センサーに電源を入れます。

SDK API ガイドは前後のセクションで使用される MIPI CSI-2 と GPIF II に関連した関数の詳細な情報を含みます。

6.9 フレーム設定の選択と切り替え

カメラは複数の解像度とフレームレートに対応できます。対応される解像度/フレームレート設定は、ビデオストリーミング (VS) インターフェースで説明されるようにそのビデオストリーミングフレームディスクリプタに格納されます。

解像度かフレーム レートの設定が USB ホストアプリケーションで選ばれると、USB ホストは SET_CUR コミット制御要求をビデオストリーミングインターフェースへ送信します。受信した構造体の 4 バイト目は、選択されるフレームディスクリプタを示します。

提供されたサンプルプロジェクトでは、デバイスが SuperSpeed で動作する場合、インデックス 0 は 720p@60 fps のビデオストリーミングに対応し、インデックス 1 は 1080p@30 fps のビデオストリーミングに対応します。USB ホストは 1080p のビデオを流すために、フレームインデックスが「1」に設定された SET_CUR 要求を送信します。

CyCx3UvcAppUSBSetupCB 関数はこのインデックスを読み出して、受信した内容に応じてイメージセンサーと MIPI CSI-2 コントローラーを設定して、要求された設定のビデオを流します。

6.10 DMA バッファの設定

UVC の仕様では、各 USB 転送 (このアプリケーションでは、各 16 KB の DMA バッファ) に、12 バイトのヘッダを追加する必要があります。しかし、CX3 アーキテクチャでは、DMA ディスクリプタに関連付けられている DMA バッファが 16 バイトの倍数の大きさを備えている必要があります。

DMA バッファ内の 12 バイトを CX3 CPU が書き込むものとして予約すると、DMA バッファの境界が 16 バイトの倍数にならない場合があります。このため、DMA のバッファ サイズは、16,384 から 12 を引くのではなく 16 を引いた値にする必要があります。CX3 ファームウェアが追加する 12 バイトのヘッダを含まない DMA のバッファサイズは、16,384-16=16,368 バイトです。DMA バッファは、先頭に 12 バイトヘッダ、次に 16,368 ビデオバイト、そして DMA バッファの最後に 4 つの未使用バイトの順で構成されます。こうして 16 x 1024 バイトのパケット (16,384 バイト) である USB BULK の最大バーストサイズを利用できる DMA バッファを生成します。

6.11 ビデオストリーミング中の DMA バッファの処理

CyCx3AppInit 関数は、プロデューサとコンシューマのイベント用のコールバック通知付きの手動 DMA チャネルを作成します。

コンシューマの通知はホストが読み出したデータ量を追跡するために使用されます。ホストがフレーム全体を読み出した後で、追跡変数がリセットされ、GPIF II ステートマシンが再起動されます。プロデューサイベント通知が 12 バイトのヘッダを付加し、ホストにデータを転送するために使用されます。

DMA コールバックでは、ファームウェアは CyU3PDmaMultiChannelGetBuffer を介して DMA バッファが生成されたかを確認します。GPIF II プロデューサの DMA バッファが転送されるか、または強制的に CX3 CPU によってラップアップされる場合、DMA バッファは CX3 CPU に使用可能になります。アクティブフレーム期間中に、イメージセンサーは、データをストリームし、GPIF II は、一杯の DMA バッファを生成します。この時点では、CX3 CPU は USB に 16,380 バイトのデータを転送します。

フレームの最後で、通常、最後の DMA バッファが部分的に満たされます。この場合、プロデューサイベントをトリガーするためにファームウェアは強制的にプロデューサ側の DMA バッファをラップアップする必要があり、その後適切なバイト数を DMA バッファから USB に転送します。CyU3PDmaMultiChannelSetWrapUp の呼び出しを使う DMA バッファ (GPIF II で生成された) の強制的ラップアップは GPIF II コールバック関数 (CyCx3GpifCB) の中で実行されます。1 フレームが終了する時に発生する CPU の割込みを GPIF II が設定すると、このコールバック関数がトリガーされます。

UVC ヘッダは、フレーム識別子およびフレーム終了マーカーに関する情報を格納します。フレーム送信の終わりで、ファームウェアは 2 番目の UVC ヘッダバイト (CyCx3UvcAddHeader を参照) のビット 1 をセットし、ビット 0 をトグルします。さらに「hitFV」変数がセットされ、イメージセンサーから取り込まれるフレームが終了したことを示します。

glDmaDone 変数は DMA バッファを追跡して、すべてのデータが CX3 FIFO から取り出されたことを確保します。

6.12 フレーム終端でのストリーミングの再開

フレームの最後で、GPIF II ステートマシンは前述した一連のイベントを開始する CPU 割込みを生成します。フレームの最後の DMA バッファは USB ホストにより読み出されると、glDmaDone 変数がゼロになります。

その後、CX3 ファームウェアは CyU3PGpifControlSWInput API を使用してファームウェアトリガーアクションを実行し、ステートマシンを再起動して、次に入力されるフレームを処理します。

CX3 ファームウェアの旧バージョンでは、CyU3PGpifSMSwitch 関数を使用して、GPIF II ステートマシンをいずれかの START ステートから再起動していました。フレームの最後のバッファがソケット 0 によって読み出された場合は、GPIF II ステートマシンを ALPHA_CX3_START_SCK1 から再起動してソケット 1 への次のフレームの読み出しを開始し、その逆の場合も同様の動作を行います。

また UVC 仕様はヘッダ内のフレーム ID ビットを必要としフレームごとに切り替えられます。これは最後のバッファが USB ホストに転送される直前に、CyCx3UvcAddHeader 関数の中で行われます。

6.13 ビデオストリーミングの終了

イメージストリーミングを終了する方法は 3 つあります。

- カメラをホストから切り離す
- USB ホストプログラムを終了する
- USB ホストがリセットを発行するか CX3 への要求を一時停止する

FIFO に残留データがある時にもビデオストリーミングを終了できるため、適正なクリーンアップが必要となります。

ビデオストリーミングを終了する時、ファームウェアはストリーミング関連の変数をリセットし、DMA チャネルをリセットし、そして GPIF II ステート マシンを一時停止します。カメラと MIPI CSI-2 インターフェースの電源も切断します。これらは CyCx3AppStop 関数の中で行います。

USB ホストのアプリケーションが終了する場合、ファームウェアは Windows プラットフォーム上で機能のクリア要求を出したり、Mac プラットフォーム上で 代替設定=0 のインターフェースの設定要求を出します。この要求を受信するとストリーミングが停止します。この要求は、ターゲットが CY_U3P_USB_TARGET_ENDPT で、かつ要求が CY_U3P_USB_SC_CLEAR_FEATURE の場合、CyCx3AppUSBSetupCB 関数で処理されます。

ビデオストリーミングを終了した後、イメージセンサーは電源切断され、CX3 コアは CyU3PSysEnterSuspendMode を使って一時停止されます。USB バス上で動作が発生すると、CX3 コアが復帰しイメージセンサーに電源を投入してビデオストリーミングを再開します。

CCI (I^2C) コマンドか CX3 の XSHUTDOWN 端子のいずれかを使うことによって、イメージセンサーは電源を切断できます。このピンは最初のパラメーターとして、CY_U3P_CSI_IO_XSHUTDOWN を持った CyU3PMipicsiSetSensorControl 関数を使って制御できます。

7 ハードウェアのセットアップ

7.1 CX3 RDK を使うテスト

本プロジェクトは、OmniVision OV5640 イメージセンサーを内蔵する CX3 リファレンスのデザインキット (RDK) を備えた構成でテストされます。キットに関する詳細はインフィニオン ウェブサイトにある **Kits** タブから入手できます。

7.1.1 キットの調達

1. e-Con Systems 社から RDK を購入してください。
2. インフィニオンは OV5640 センサーの基本的な操作を実行できるバイナリライブラリを CX3 SDK に同梱して提供します。追加機能のサポートが必要な場合は、OmniVision 社と NDA を締結した上で、テクニカルサポートまでお問い合わせください。NDA の確認後、インフィニオンより OmniVision 固有のソースファイルを提供します。
3. SuperSpeed 性能を評価するために、USB 3.2 Gen1 ホスト対応のコンピュータを使用します。

CX3 RDK にはクイックスタートガイド, ハードウェアユーザーマニュアル, およびファームウェアビルドマニュアルが付属しており、お客様が RDK の使用を開始する手助けをします。

7.2 独自基板の設計

独自に基板を設計する際に、基板の動作を確実にするためにいくつかのガイドラインに従う必要があります。FX3/FX3S のハードウェア設計に関する推奨事項 (CX3 にも適用可能) については、FX3/FX3S hardware design guidelines アプリケーションノートを参照してください。詳細については、CX3 hardware FAQs KBA も併せて参照してください。

MIPI CSI-2 信号については、以下に示す配線ガイドラインに従う必要があります。

- 基板上の配線長は、100 mm を超えてはいけません。
- 差動信号配線の伝送線路インピーダンス (Z0) は、100 Ω +/- 10%にしてください。
- P-N が組となる配線長の差は 0.5 mm 未満にしてください。
- MIPI CSI レーン信号の差動配線長の差は 1.5 mm 未満にしてください。
- P-N が組となる信号の配線間隔は線幅の 2 倍にします。

8 UVC ベースのホストアプリケーション

様々なホストアプリケーションにより、UVC デバイスからビデオを表示して取り込めます。Windows OS では、Windows 標準の Camera アプリや Media Player Classic がよく利用されます。その他、Windows 向けのアプリケーションとして VirtualDub (オープンソース) や e-CAMView も挙げられます。

Linux システムでは、ビデオをストリーミングするために V4L2 ドライバと Webcamoid を使用できます。

Mac プラットフォームでは、Webcamoid, Photo Booth, Debut Video Capture といったソフトウェアを使用し、UVC デバイスと連携してビデオをストリーミングできます。

9 トラブルシューティング

デバイスまたはファームウェアで問題に直面した場合、最初のステップはより多くのデータを取得し問題の原因を絞り込むことです。これをさらに実現するには、UART と JTAG デバッグを有効にできます。

cycx3_uvc.h ファイルの「CX3_DEBUG_ENABLED」スイッチを有効にして、様々なカウンター, エラーメッセージ, および色々なデバッグデータを印刷できるようにします。UART ケーブルまたは USB-UART 間ブリッジを介して基板 (または CX3 RDK) 上の UART ポートを PC に接続します。Hyperterminal, Tera Term, または PC 上の COM ポートにアクセスする別のユーティリティを開きます。転送を始める前に UART コンフィギュレーションを 115,200 ボー, パリティなし, 1 ストップ ビット, フロー制御なし, 8 ビットデータに設定します。この設定はデバッグプリントを取り込むのに十分です。

また JTAG デバッガは各ステップでファームウェアをデバッグするために使用できます。JTAG 経由のデバッグ処理に関する詳細については、プログラマーマニュアル (**Start > All Programs > Cypress > EZ-USB FX3 SDK**) のセクション 12.2.2.3「Executing and Debugging」を参照してください。

CX3 のファームウェア, ハードウェア, アプリケーションソフトウェアについての FAQ は KBA91297, KBA91295, および KBA91298 で文書化されました。

いくつかの FAQ については、以下の KBA に記載されています。

- KBA233853 - EZ-USB™ CX3 troubleshooting guide
- KBA91297 - CX3 Firmware: Frequently Asked Questions
- KBA91295 - CX3 Hardware: Frequently Asked Questions
- KBA91298 - CX3 Application Software / USB Driver: Frequently Asked Questions

いくつかの一般的な問題やこれらの原因、対策を一覧表示します。

問題: PC 上でデバイスがエニュメレート (認識) されません。

- **原因 1**: ファームウェア内の API でエラーが発生した可能性があります。これによりエラーハンドラが呼び出され、ファームウェアが停止したり、エニュメレートに失敗したりすることがあります。
  解決策 1: JTAG デバッガまたは UART/RS232 ケーブルを使用し、すべての API 呼び出しの戻り値のステータスを確認して、失敗したものがないか調べてください。その後、API ガイドを参照して、呼び出しが失敗した原因を確認してください。
- **原因 2**: ボード上の USB 信号品質が不十分です。
  解決策 2: 設計ガイドラインのアプリケーションノートを読み、適切なボード設計を確認してください。詳細については、独自基板の設計セクションを参照してください。
- **原因 3**: 適切なドライバがインストールされていません。
  解決策 3: Device Manager でデバイスをアンインストールし、再インストールを試してください。また、VID/PID のペアを変更することで、強制的に新しいデバイスとしてインストールさせることも可能です。
- **原因 4**: USB ディスクリプタの値が正しくない可能性があります。
  解決策 4: エラーの一般的な原因として、長さ (length) フィールドの設定ミスが挙げられます。コンフィギュレーションディスクリプタの `wTotalLength` フィールドが、正しい全体の長さを表していることを確認してください。また、クラス固有のビデオコントロール (Video Control) ディスクリプタやビデオストリーミング (Video Streaming) ディスクリプタについても、同様のフィールドを確認してください。

問題: デバイスのエニュメレート (認識) は行われますが、USB ホストアプリケーション (e-CAMView など) の画面が真っ黒になります。

- **原因 1**: `CyCx3UvcAppThread_Entry`関数内の`glDMATxCount`変数を調べ、その値が増加しているか確認してください。増加していない場合、CX3 とイメージセンサー間のインターフェースに問題がある可能性があります。
  解決策 1: イメージセンサーが CX3 に正しく接続されているか確認してください。次に、センサーの初期化, MIPI CSI-2 コントローラーの設定, および GPIF II のバス幅選択が正しく行われているか確認してください。
- **原因 2**: `glDMATxCount`の値が増加している場合は、送信されている画像データを確認する必要があります。まず、1 フレームあたりに送信されているデータの総量を調べます。これを確認するには、ヘッダ内の「フレーム終了 (EOF) ビット」がセットされているデータパケットの長さを特定します (フレーム終了転送の場合、ヘッダの 2 バイト目は`0x8E`または`0x8F`になります)。それ以外のデータパケットについては、転送されるデータ長は「DMA バッファサイズ」から「フッターサイズ」を引いた値になります。
  1 フレームで転送される画像データの総量 (UVC ヘッダを除く) は、「幅 × 高さ × 2」になるはずです。
  解決策 2: 画像データの総量が規定値より少ない（または多い）場合、イメージセンサーが規定量より少ない (または多い) データを送信している可能性があります。イメージセンサーおよび MIPI CSI-2 コントローラーの設定を確認してください。

問題: Hi-Speed 接続では動画をストリーミングできるが、SuperSpeed 接続ではできない。

- **原因**: SSTX/SSRX ラインの信号品質が不十分であることが考えられます。SuperSpeed の配線が AN70707 に記載されているガイドラインに従っているか確認してください。また、認証済みのケーブルのみを使用するようにしてください。
  さらに、CX3_ERROR_THREAD_ENABLE スイッチを有効にすると MIPI CSI-2 エラーのログ記録が行われ、CSI-2 インターフェースでエラーが発生したかどうかを確認できます。エラーの種類に関する詳細については、SDK API ガイドを参照してください。
  それでも問題が解決しない場合は、Infineon Developer Community で類似の事例を確認してください。

10 まとめ

本アプリケーションノートでは、MIPI CSI-2 インターフェース付きの USB ビデオクラスに準拠したイメージセンサーが、どのような方法でインフィニオンの EZ-USB™ CX3 を使用して実装できるかを説明しています。特に、以下のことを示します。

- ホストアプリケーションおよびドライバが UVC デバイスとやり取りする方法
- UVC デバイスが UVC 固有の要求を管理する方法
- 一般的なイメージセンサーからのデータを受信するために MIPI CSI-2 インターフェースを設定する方法
- ホストアプリケーションでビデオのストリームを表示し、カメラのプロパティを変更する方法
- オープンソースのホストアプリケーションプロジェクトを含む様々なプラットフォーム上で使用可能なホストアプリケーションを検索する方法
- 必要に応じて、CX3 ファームウェアの問題を解決しデバッグする方法

11 関連リソース

インフィニオンは、設計に適した USB デバイスの選定や、そのデバイスを迅速かつ効果的に設計へ組み込むために役立つ豊富なデータを、www.infineon.com にて提供しています。リソースの包括的なリストについては、CX3 のウェブページを参照してください。

概要: USB Portfolio

- USB 3.0 製品セレクター: FX3, FX3S, CX3, HX3, SX3
- アプリケーションノート: インフィニオンは、基礎から応用まで幅広いトピックを網羅した多数の USB アプリケーションノートを提供しています。CX3 を使い始めるにあたり推奨されるアプリケーションノートは以下のとおりです。
  - AN75705 - EZ-USB™ FX3 入門
  - AN75779 – USB ビデオクラス (UVC) フレームワーク内で EZ-USB™ FX3 を使用してイメージセンサーインターフェースを実装する方法
  - AN76405 - EZ-USB™ FX3 ブートオプション
  - AN70707 - EZ-USB™ FX3/FX3S ハードウェア設計ガイドラインおよび回路図チェックリスト
  - AN86947 - EZ-USB™ FX3 による USB 3.0 スループットの最適化
  - AN231295 - EZ-USB™ SX3 入門
- サンプルコード:
  - USB Super-Speed
- テクニカルリファレンスマニュアル (TRM):
  - EZ-USB™ CX3 テクニカルリファレンスマニュアル
- Knowledge Base Articles (KBA):
  - Analysis of CX3 Video Timing Parameters - KBA226779
  - Analysis of CX3 Clocking Parameters - KBA226758
  - CX3 Firmware: Frequently Asked Questions - KBA91297
  - CX3 Hardware: Frequently Asked Questions - KBA91295
  - CX3 Application Software / USB Driver: Frequently Asked Questions - KBA91298
  - EZ-USB™ CX3: Interfacing with the onsemi AP1302 ISP and AR1335 image sensor - KBA237926
  - EZ-USB™ CX3: Interfacing with the onsemi AR0234CS sensor - KBA236855
  - EZ-USB™ CX3 based USB3 Vision solution - KBA236151
  - Knowledge Base - Cypress Semiconductor Cage Code - KBA89258
  - Designing Type-C products based on EZ-USB™ FX3 and CX3 - KBA218460
  - Using Cypress CX3 Based USB 3 Camera Kits with Raspberry Pi 4 - KBA229301
- 開発キット:
  - EZ-USB™ CX3 THEIA-CAM - 13MP PDAF UVC Camera Solution
  - Denebola - USB 3.0 UVC Reference Design Kit (RDK)
- モデル:
  - CX3 Device OrCad Schematic Symbol
  - CYUSB306x - IBIS

改訂履歴

版数	発行日	変更内容
**	2014-05-22	これは英語版 001-90369 Rev. **を翻訳した日本語版 001-92480 Rev. **です。
*A	2025-10-06	これは英語版 001-90369 Rev. *C を翻訳した日本語版 001-92480 Rev. *A です。
*B	2017-06-28	最新のテンプレートおよびロゴに変更しました。
*C	2020-06-17	これは英語版 001-90369 Rev. *D を翻訳した日本語版 001-92480 Rev. *C です。
*D	2026-10-16	これは英語版 001-90369 Rev. *G を翻訳した日本語版 001-92480 Rev. *D です。

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2026-10-16

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2026 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

001-92480 Rev. *D

重要: 注意事項

本文書の全ての日本語情報は、英文で提供されている情報の参考訳です。英文版と和文版に齟齬がある場合は常に英文版の内容が優先します。

製品 (サンプルも含みます。また、ハードウェア、ソフトウェアまたはその両方で構成される製品も含みます。以下同じです。) の販売、提供または引渡しは、インフィニオンテクノロジーズAGおよびその関連会社 (以下まとめて「インフィニオン」といいます)、お客様とインフィニオンによって締結された基本契約その他の書面により合意された条件、そのようなの条件がない場合はインフィニオンの適用ある販売条件に従います。お客様の一般的な取引条件またはインフィニオンの販売条件と乖離した条件は、インフィニオンが書面で明示的に同意した場合に限り、インフィニオンを拘束するものとします。

疑義を避けるため、インフィニオンは、第三者の権利を侵害していないことの保証、および特定の使用/目的に対する適合性または商品性の保証などの黙示的保証を負いません。

インフィニオンは、サンプル、アプリケーション、もしくはお客様による製品の具体的な使 用方法、または本書面に記載された例や標準値に関するいかなる情報についても責任いません。

本書面に記載されているデータは、技術的資格を有する熟練したお客様のご担当者をのみを対象としています。お客様は、意図された用途およびお客様の特定用途に対する製品の適合性を評価し、意図された用途および顧客の特定用途において本書面に含まれるすべての関連技術データを検証する責任を負います。お客様は、意図されたアプリケーションの機能性および安全性を適切に設計、プログラミング、テストし、その使用に関連する法令を遵守してください。

インフィニオンの明示的な承認がない限り、製品は、製品の故障またはその使用による結果が人身傷害につながると合理的に予想されるいかなる用途にも使用しないでください。ただし、上記は、インフィニオンが明示的に設計し、販売した使用分野でお客様が製品を使用することを妨げるものではありません。

インフィニオンは、適用される法律、例えばドイツ著作権法 (UrhG) 第44b条に従って、そのコンテンツを商業的なテキストおよびデータのマイニング (TDM) のために使用する権利を明示的に留保します。

製品にセキュリティ機能が含まれている場合:

いかなるコンピューティング・デバイスも絶対的に安全であることはあり得ないため、製品にセキュリティ対策が施されているにもかかわらず、インフィニオンは、製品に侵入、データの盗難もしくは損失、またはその他の侵害 (「セキュリティ侵害」といいます) がないことを保証せず、インフィニオンは、セキュリティ侵害に起因するいかなる責任も負わないものとします。

本書面がソフトウェアを含んでいるまたはソフトウェアに言及している場合：

ソフトウェアは、米国、ドイツ、および世界各国の知的財産権法および条約に基づきインフィニオンが所有しています。インフィニオンはすべての権利を留保し、無断複写、複製および転写を禁じます。お客様はソフトウェアに同梱されたソフトウェア使用許諾契約に従ってのみソフトウェアを使用できます。

ソフトウェア使用許諾契約が適用されない場合、インフィニオンはここに、ソフトウェアの知的所有権に基づき、(a) ソースコード形式で提供されたソフトウェアについて、インフィニオンのハードウェア製品と共に使用する目的のためにのみソフトウェアを修正および複製するための、そして(b)インフィニオンのハードウェア製品上での使用目的に限り、ソフトウェアをバイナリコード形式でエンドユーザーに外部配布するための、個人的、非排他的、譲渡不能なライセンス (サブライセンス不可) をお客様に付与します。それ以外のソフトウェアの使用、複製、変更、翻訳またはコンパイルは禁止されています。製品、技術、納品条件、および価格に関する詳細については、最寄りのインフィニオンオフィスに連絡するか、https://www.infineon.comにアクセスしてください。