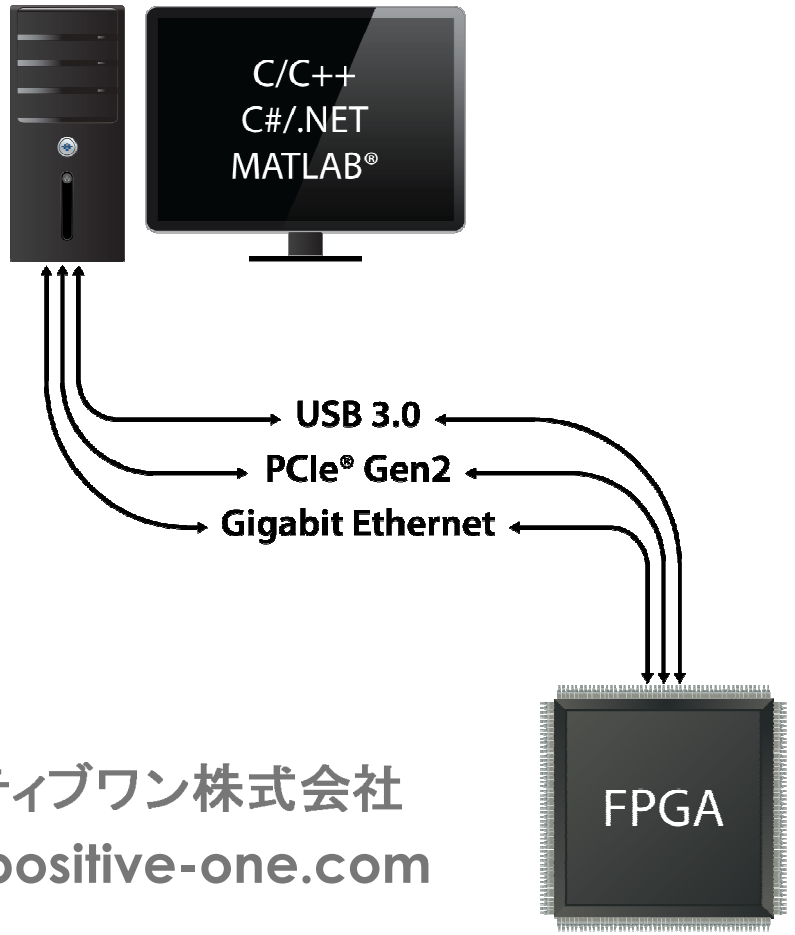
FPGA Manager

Streaming, made simple.

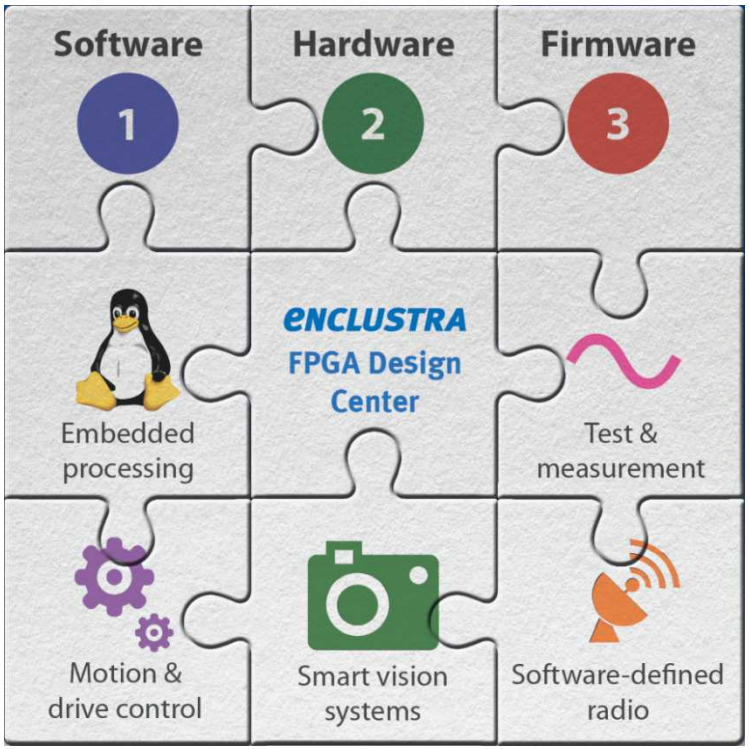


ポジティブワン株式会社
www.positive-one.com

Enclustra情報

FPGAデザインセンター
- ハードウェア（高速、アナログ、RF）
- HDLファームウェア（VHDL、Verilog）
- 組み込みソフトウェア（FPGAプロセッサ用）

FPGA ソリューションセンター
- IPコア
- FPGAモジュール



XILINX
ALLIANCE PROGRAM
CERTIFIED MEMBER – BASE

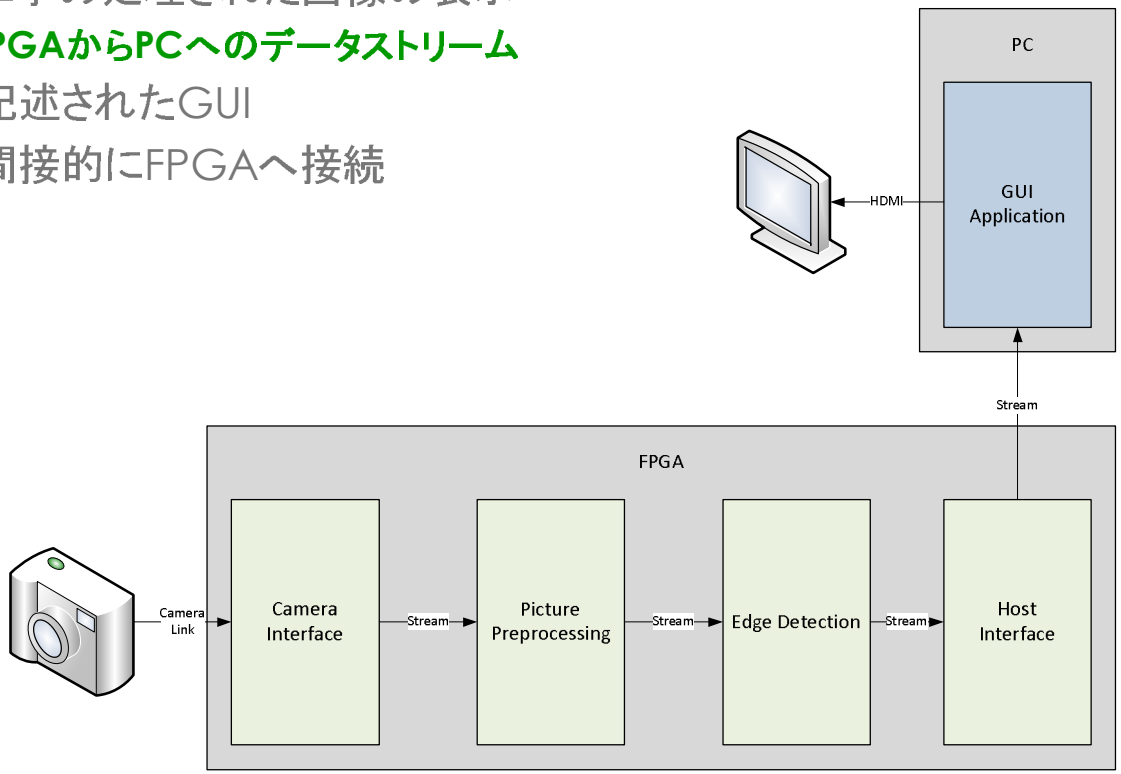
Altera Design Services Network™

LEADER
Lattice Exclusive Alliance of Design Engineering Resources

Microsemi

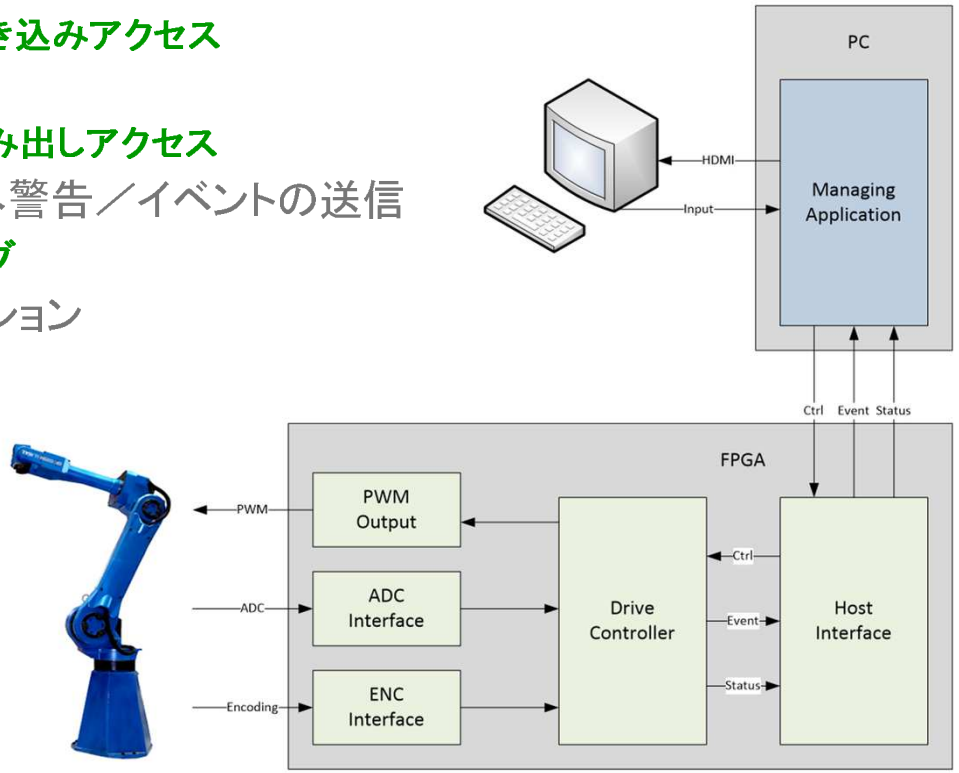
FPGAを高級言語へリンクする理由

- ユースケース1：エッジ検出を伴うカメラ
  - 高解像度と極めて高いリフレッシュレートのため、FPGA内でデータ処理
  - PC上に予め処理された画像の表示
    - **FPGAからPCへのデータストリーム**
  - C#で記述されたGUI
  - PCは間接的にFPGAへ接続



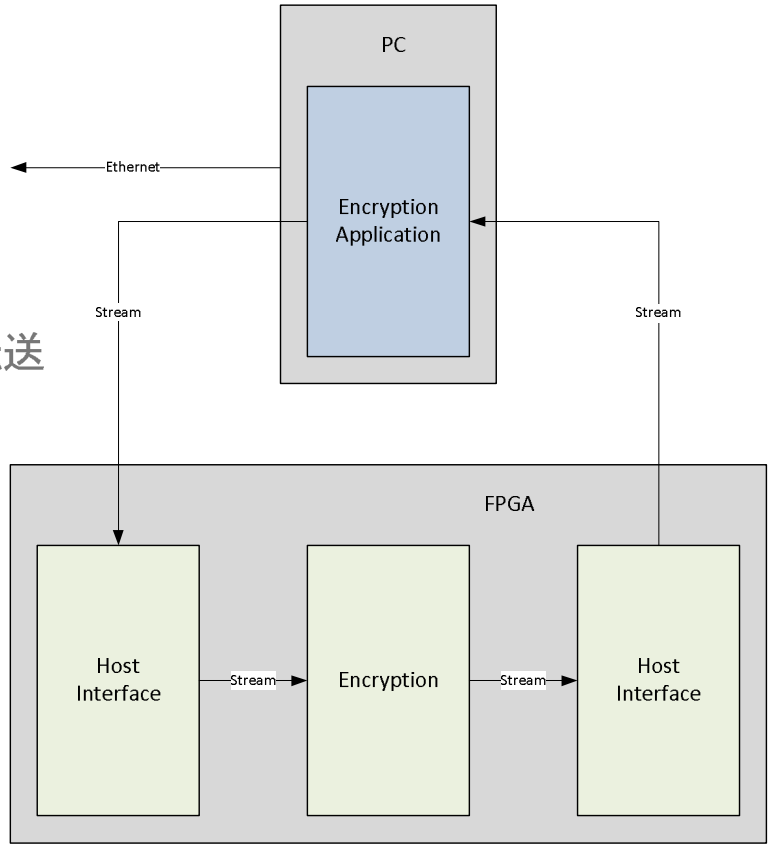
FPGAを高級言語へリンクする理由

- ユースケース2：駆動コントローラの設定及びステータス監視
  - 高い制御ループレートのため、FPGA内に駆動コントローラ
  - PCからの動きベクトルの設定
    - **PCからFPGAへレジスタ書き込みアクセス**
    - PCからステータス監視
    - **PCからFPGAへレジスタ読み出しアクセス**
  - 駆動コントローラからユーザへ警告／イベントの送信
    - **FPGAからPCへシグナリング**
  - C++で記述されたアプリケーション
  - PCは間接的にFPGAへ接続



FPGAを高級言語へリンクする理由

- ユースケース3：ハードウェア加速のソフトウェア
  - リアルタイムでネットワーク・トラフィックの暗号化
  - FPGAによるネットワーク・トラフィックの暗号化
    - **PCからFPGAへデータストリーム**
  - PCによる暗号化データの送信
    - **FPGAからPCへデータストリーム**
  - Cで記述された小さなアプリケーション、非同期転送
  - PCとFPGAは同一システム



FPGAを高級言語へリンクする理由

- 3種の相互作用
  - FPGAとPCの間のデータストリーム
    - 通常、データ作成（FPGAからPCへ）やデータ処理（PCからFPGAへ）のために使用
    - **データストリーム**
  - PCによるFPGA上のレジスタアクセス
    - 通常、設定データの書き込みやステータスのポーリングのために使用
    - **メモリマップド**
  - FPGAからPCへイベントシグナリング
    - 通常、割り込み（完了、エラー等）
    - **シグナリング**

FPGAを高級言語へリンクする時の要件(1)

- 相互作用の3種全てに対応するものとする
  - データストリーム
    - フレームストリーム、バイトストリーム
    - データストリームにメタデータを付加（例、タイムスタンプ）
  - メモリマップド
    - 読み出し、書き込み、読み出し／修正／書き込み
    - 単一アクセス、バースト
  - シグナリング
    - 割り込み
    - エッジ、レベル
    - プロパティ

FPGAを高級言語へリンクする時の要件(2)

- 多様性
  - 多様な異なるリンクや帯域幅
    - PCIe Gen1/2/…、USB 2.0/3.0/…、ETH100/1000Mbps/…
  - 多様なFPGAベンダ、CPUアーキテクチャ、OSとFPGAボード
    - Altera、Xilinx、.. とX86、ARM、... とWindows、Linux ....
  - 多様なプログラミング言語
    - C#/.NET、C/C++、Matlab、Java等
- 統一性
  - 全てのリンク、FPGAやボードに対して同じAPI
  - 異なるオペレーティングシステムに対して同じAPI
  - 異なるプログラミング言語に対して「同じ」API（機能上同等、可能な場合は構文上も同等）
  - 全てのリンク、FPGAやボードに対して、FPGA内のユーザロジックには同じインタフェース
  - FPGA内のユーザロジックには標準インタフェースの使用

FPGAを高級言語へリンクする時の要件(3)

- 多重化
  - 同一物理リンク上に複数の接続チャネル
    - 例、同じUSB3.0リンク上に3データストリーム・チャネル、2メモリマップド・チャネル、2シグナリング・チャネル
  - 異なるチャネルへ複数アプリケーションの同時接続
- パフォーマンス
  - 帯域幅(MB/s)
  - レイテンシ(s)
  - CPU負荷(%)
  - フロー制御

FPGAを高級言語へリンクする時の要件(4)

- ブロッキングと非ブロッキング
  - 同期及び非同期転送に対応

FPGAへリンクすることは、ユーザにとって**シンプル**であるべき！！！

ユーザは自身の最終目標に集中できなくてはならない。FPGAへリンクすることは、主にその目的への手段に過ぎない。

FPGAを高級言語へリンクする時の課題(1)

- 多様性 vs. 統一性
  - 全ての観点において統一されたインタフェース（リンク、FPGAベンダ、CPUアーキテクチャ、オペレーティングシステムやプログラミング言語）
- パフォーマンス vs. リソース
  - 最大のパフォーマンスを実現するためには、通常はより多くリソースが必要
- シンプル性 vs. フレキシブル性
  - APIがどれほどシンプルになるか、そしてそれでも最大のフレキシブル性を提供可能か？
- 様々なリンクには追加のチップ／ドライバ／ライブラリが必要
  - PCIe => MGTs とPCIeハードIP
  - USB => USB PHY、FIFOコントローラ（例、Cypress FX3やFTDI 2232H）、FIFOコントローラ・ドライバとアクセス・ライブラリ
  - イーサネット => ETH PHY、ホストMACドライバとOSのソケットインタフェース

FPGAを高級言語へリンクする時の課題(2)

- 帯域幅／スループット vs. レイテンシ
  - 最大帯域幅には大きなデータパケットが最適で、低レイテンシには小さなパケットが最適で、最大スループットにはできる限り多くのオペレーションが1つのパケットに含まれていなくてはならない。
- モジュール性／保守性 vs. 最適化
  - 保守性とモジュール性を損なう前にあなたはどれだけ最適化できますか？

主な課題は、**これら全てのパラダイムを1つの役割の下に持ってくること！**

これらのパラダイムをどう扱う？ (1)

- 多様性 vs. 統一性
  - ソフトウェア・ハードウェア開発者にとってリンクはブラックボックスであるものとする。
    - ホストとFPGA上のインタフェース抽出
  - リンクに特有なプロトコル／フレームの中にカプセル化可能な、リンクに依存しないストリーミング・プロトコルを使用する
    - 多重化、エラー検出、スロットリング等を可能にする
  - 既にリンクに依存していないストリーミング・プロトコルの中にカプセル化されるべき、リンクに依存しないメモリマップド・プロトコルを使用する
    - メモリマップド・アクセスを可能にする

これらのパラダイムをどう扱う？ (2)

- 多様性 vs. 統一性
  - ソフトウェア・ライブラリ内の、OSアブストラクション・レイヤを伴うオペレーティングシステムを抽出する
  - 1つのプログラミング言語のソフトウェア・ライブラリを作成し、異なる言語のアクセスAPIしかない（例、C++のライブラリ、C、C++、Java、Matlab …のためのAPI）
  - ファームウェアのコア部内にはFPGAベンダの特有な構文を使用しない
  - ファームウェア内のリンク抽出モジュールを使用してリンクを抽出する
    - ここでFPGAベンダに依存する構文も使用されることがある（例、PCIeハードIP）
  - ファームウェア内で標準インタフェースを使用する（AXIとAXIS）

これらのパラダイムをどう扱う？ (3)

- パフォーマンス vs. リソース
  - パフォーマンスとリソースに影響する全てのパラメタをユーザがアクセス可能にする
    - 要求されたパフォーマンスを達成するための最小必要リソースを示す特定のアプリケーションにソリューションを調整することを可能にする
- シンプル性 vs.フレキシブル性
  - シンプルなAPIを作成する
  - API機能性の動作を変更するためのパラメタを提供する
  - 「ほとんど」のアプリケーションに適合するデフォルトパラメタのセットを作成する
- シンプル性 vs. 複雑性
  - 複雑なものをユーザから遠ざける
  - あり得る全てのエラーケースからのリカバーを試みない

これらのパラダイムをどう扱う？(4)

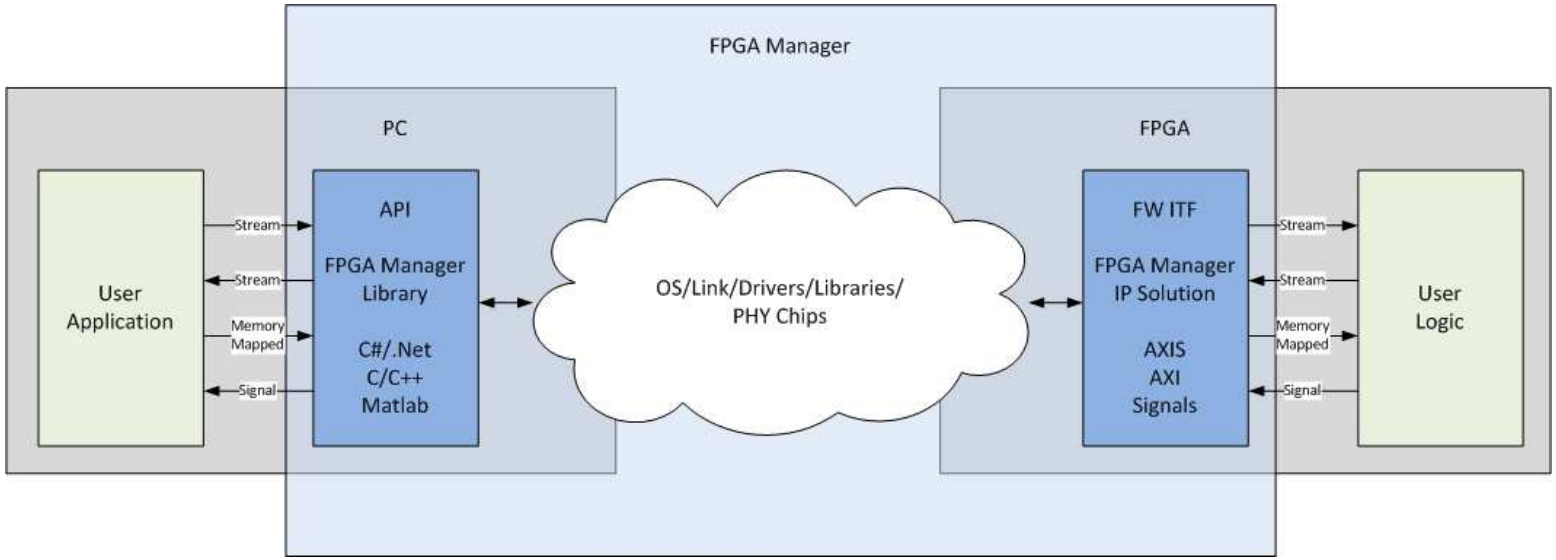
- 帯域幅 vs. レイテンシ
  - 帯域幅、レイテンシとスループットに影響する全てのパラメタをユーザがアクセス可能にする
    - アプリケーションによって求められる最小レイテンシと最大スループットを伴って最大帯域幅を示す特定のアプリケーションにソリューションを調整することを可能にする
- モジュール性／メンテナンス性 vs. 最適化
  - パーツが最適化されたバージョンと置き換えられるように、そしてそれでもコアパーツを維持できるように、明確なインタフェースを伴うモジュラ・アーキテクチャを作成する
  - 特定のアプリケーション用に最適化するためのパラメタを提供する

なぜ標準ソリューションが完全に合理的なのか

- これら全ての課題ーそして私達の行いたい唯一の事はFPGAと通信すること
- あなたは「典型的に」リファレンス設計から始め、ストレス増加という追加ボーナスも付いて、結局あなたのプロジェクト予算（お金、時間）の50%を使う（浪費？）ことになる
- 作業の繰り返し：各リンク、OSやFPGAに対して再び
- FPGAとの通信は目的への手段に過ぎないことが多い

FPGAマネージャの概要
FPGAマネージャとは？

- ソフトウェアーファームウェアの共通ソリューション
  - **FPGAマネージャのファームウェアIPソリューション**
    - FPGA内のユーザロジックへアクセス
  - **FPGAマネージャのソフトウェア・ライブラリ**
    - ユーザ・アプリケーションのためのAPI



まとめ

- FPGAの高級言語への接続は**代表的なアプリケーション**
- FPGAの接続は**目的への手段に過ぎない**ことが多い
- 接続は**シンプル**であるものとする
- 接続は**フレキシブル**であるものとする
- **マイグレーション**は**容易**であるものとする
- 多くの場合、**標準的なソリューションの使用は自ら開発するより安い**

- FPGAからホストへの通信の課題
  - 多様性
  - データストリーミングとメモリマップド・アクセス
  - 多重化
  - パフォーマンス
  - エラー処理
  - デバイス列挙
  - 設計セキュリティ
  - 互換性
  - FPGAの具体的な課題
  - PCIe、USBとイーサネットの具体的な課題

FPGAからホストへの通信の課題

- 多様性
  - 多様なインタフェース、速度、世代（PCIe Gen1/2/3/4, USB2/3/4, 100Mbps/1Gbps/10Gbps ETH）
  - 多様なFPGAベンダ及びFPGAボード(Altera、Xilinx等／評価ボード、カスタムボード等）
  - 多様なオペレーティング・システム及びプログラミング言語（Linux、Windows等／C++、.NET、Matlab等）
- データストリーミング
  - ストリームのフレーム化（フレームストリーム対バイトストリーム、メタデータ関連付け（例、タイムスタンプ等））
  - フレームの断片化と再構築（フレーム対パケット、フレーム対バッファ）
  - フレームサイズはバッファサイズより大きくても可能（カットスルー転送、フレームポイズニング）
- メモリマップド・アクセス
  - 異なるアクセスタイプ（単一ワード、バースト、同一アドレスのバースト、アトミックな読み出し／修正／書き込み等）
  - 先読み不可のターゲットからの読み出し（例、FIFO）とパケットロス・シナリオ下での実行順序付け
- 多重化
  - 同一物理インタフェース上の複数チャネル（例、ステレオビジョンや4チャネル・オシロスコープ等）
  - 同一デバイスへの複数の同時接続（例、デーモンの監視、ユーザGUI等）

FPGAからホストへの通信の課題

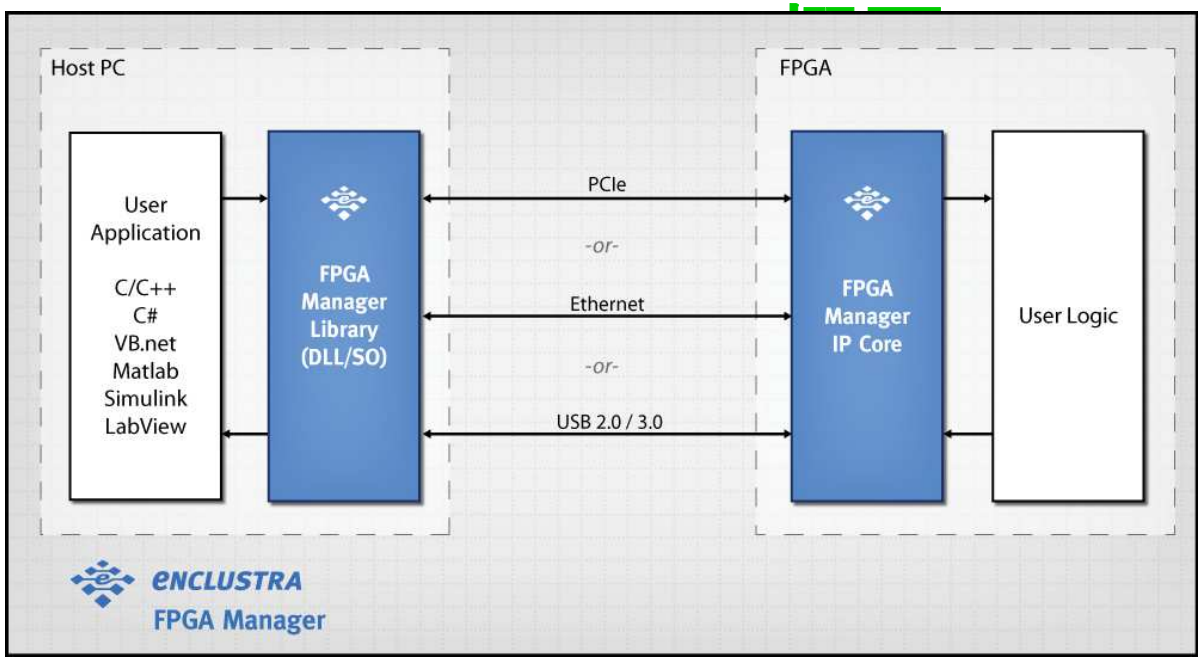
- パフォーマンス
  - 帯域幅(MBytes/sec)対レイテンシ(sec)
    - 高帯域幅は大きいフレームサイズを求め、一方で低レイテンシは小さいフレームサイズを求める
    - 高帯域幅はホストにおける最小限のデータコピー数も求める
  - オペレーションのスループット(MOps/sec)
    - ホスト内とFPGA内での動作パイプライン化、フレームごとのマルチオペレーション
  - CPU使用率
    - ポーリングの代わりにイベント駆動型
  - スケジューリング
    - あるストリームの伝送による他のストリームへのインパクトを最小化
- エラー処理
  - Non-fatalエラー処理（バッファオーバーフロー、チェックサムエラー、パケットロス、タイムアウト等）
  - Fatalエラー処理（不正パケットフォーマット、ネットワークプロトコルエラー、例外、スタックオーバーフロー等）
- デバイス列挙
  - 全リンク（PCIe、USB、イーサネット等）上の「あなたの」デバイスを発見し、デバイス情報を入手

FPGAからホストへの通信の課題

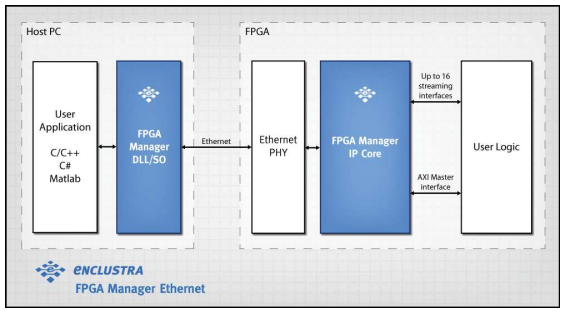
- FPGAコンフィギュレーション
  - FPGAファームウェアのダウンロードやフェイルセーフFPGAファームウェアのアップデート
- 設計セキュリティ
  - FPGAビットストリームのコピープロテクション、暗号化データフロー、ファームウェア／ソフトウェア認証、ライセンス許諾等
- 互換性
  - ソフトウェアバージョン／ネットワークプロトコルの後方・前方互換性を受け入れるためのFPGAファームウェア
- FPGAインタフェース
  - 複数のバスプロトコル（AXI4,、Avalon等）
  - 複数のデータ幅とストリームフォーマット（バイト、フレーム、ブロック）
  - 複数のクロックドメイン
- FPGAデータバッファリング
  - データ／パケットレートのスロットリング（例、ホストやリンクがデータソースほど速くない場合）
  - パケットロスシナリオにおける再送信能力
- PCIe、USBやイーサネットの具体的な課題

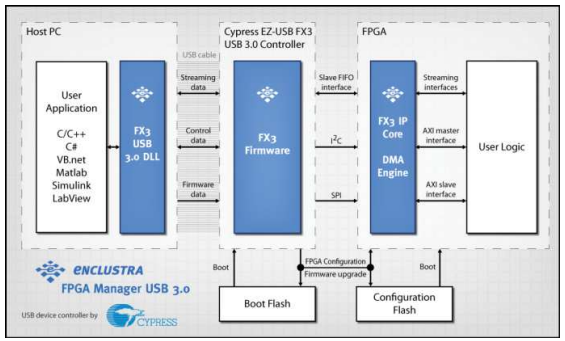
- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

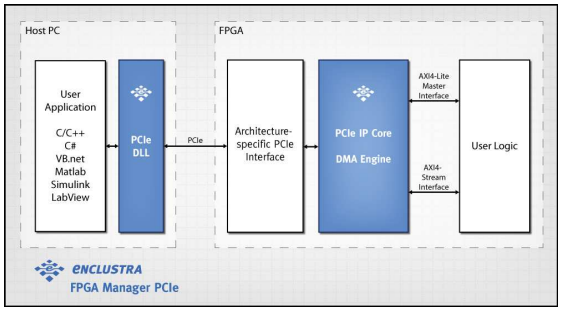
FPGAマネージャ



- 全リンクタイプに共通な一つのホストSW API
- 全リンクタイプに共通な一つのFPGA IPコア・ユーザインタフェース
  - 方面毎に最大16ストリーミングポート
  - メモリマップド・オプション（1つのアップストリーム・ポートと1つのダウンストリーム・ポートを消費する）







FPGAマネージャ凡例

- システム
  - 方向毎に最大16の独立した同等のストリーム
  - メモリマップド・コンバータへのストリーム（例、 AXI4-MMへのAXI4-ストリーム）
  - ユーザは誤ったデータや順序の誤ったデータを決して受け取らない
- フレーム
  - データは様々なサイズのフレームのシーケンスで構成される。
  - パケットを特定のストリーム、フレームやポジションへ対応付けるために、各パケットのヘッダにおいて、フレーム番号とフレームバイト・オフセットが使われる。
  - ホスト上で受信したフレームはペンディング受信バッファよりも短かったり長かったりするかもしれない。多くのメモリ容量を消費することなく、フレームサイズにおいて大きなダイナミクスに対応するために、フレームは複数の受信バッファに及ぶことがある。
- エラー処理
  - データロスは許容される。（PCIe以外）
  - コールバックや例外を用いて、ユーザに全てのエラー状態を通知する。
  - すべてのオペレーション上で（オプションの）タイムアウトによって活性を確保する。

FPGAマネージャ凡例

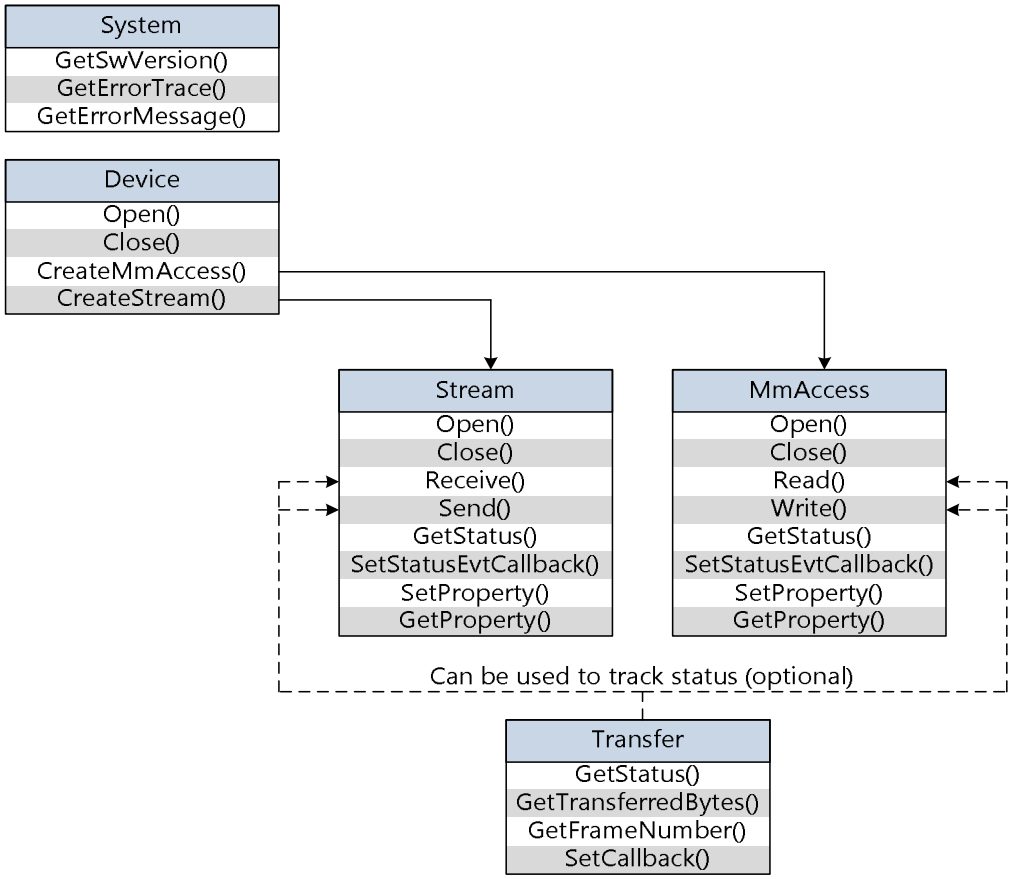
- FPGA IPコア
  - ユーザ側のAXIインタフェースを、ユーザが選択可能なクロック周波数で動作させるために、IPコアは非同期FIFOを含む。
  - ユーザはコンパイル時に、各ストリームのアップストリームとダウンストリームのバッファサイズを特定することができる。
- ホストソフトウェアAPI
  - マルチスレッドのAPIアクセス（例、データストリーム毎に1ユーザスレッド）
  - 各ストリーム上の送受信オペレーションをいくつでも保留できる。
  - 送信が完了すると、コールバックや例外を用いてユーザに通知する。

FPGAマネージャ 設計目標

- ソリューションは容易に習得でき、追加設定なしで動作するものとする。
- FPGAマネージャを異なるリンクタイプ、FPGAベンダ、プログラミング言語やオペレーティングシステムで使用する際、ユーザが最小限の違いしか見ないものとする。
- IPコンポーネントはFPGAベンダのツールと上手く統合するものとする。（ドラッグとドロップ）
- 機能一式を必要としないアプリケーションに対するライセンスのコストを低く保つために、ライセンス許諾はできる限りフレキシブルであるものとする。
- 最大限のパフォーマンスが得られるものとする。
- ストリームは、別のストリームを機能停止させないものとする。
- メモリマップド・アクセスは再送信に対応し、実行順序を重視するものとする。
  - このゴールが達成されると、ユーザのアプリケーション・デザインは実質的に簡略化される
- 全てのデータ転送は（ワードレベルではなく）バイトレベルで指定されるものとする。
  - 各転送において、いかなる数値のバイトでも読み書き可能であるものとする。送信元アドレスと送信先アドレスには、アライメント制約がないものとする。
  - ホストソフトウェアは、FPGA上のバス幅を知る必要がないものとする。

- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

FPGAマネージャ
ホストーマスタAPIクラス図

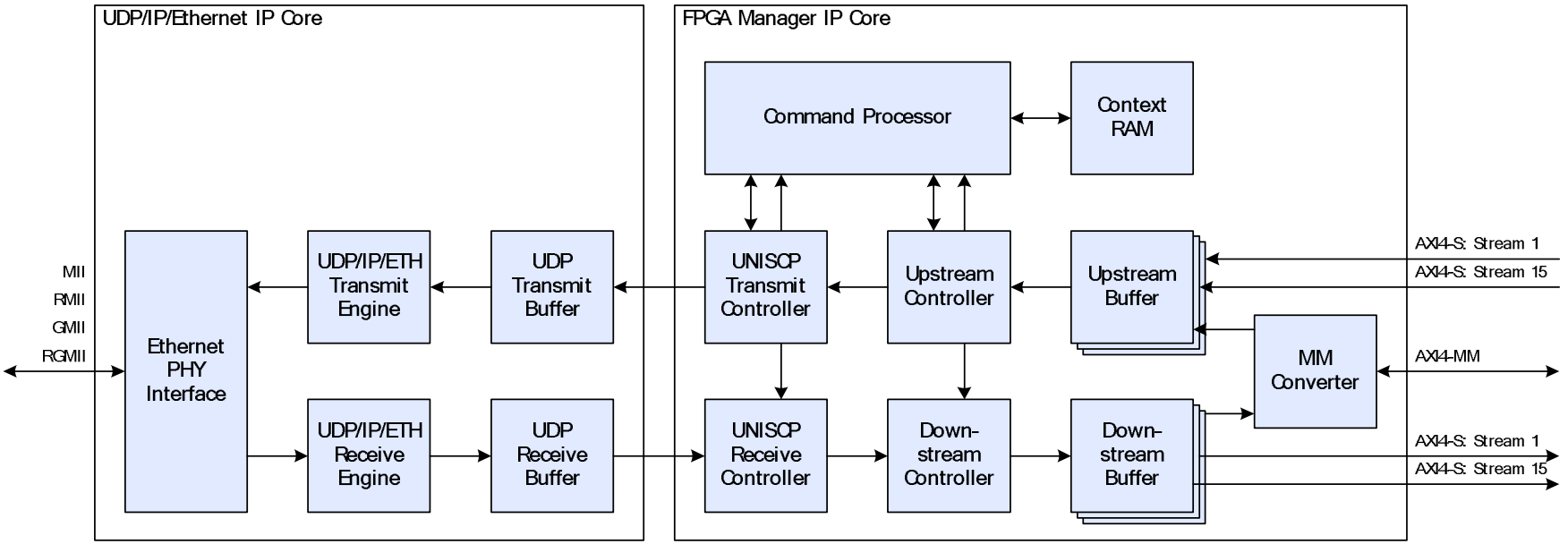


FPGAマネージャ
ホストーC#ストリーミング例

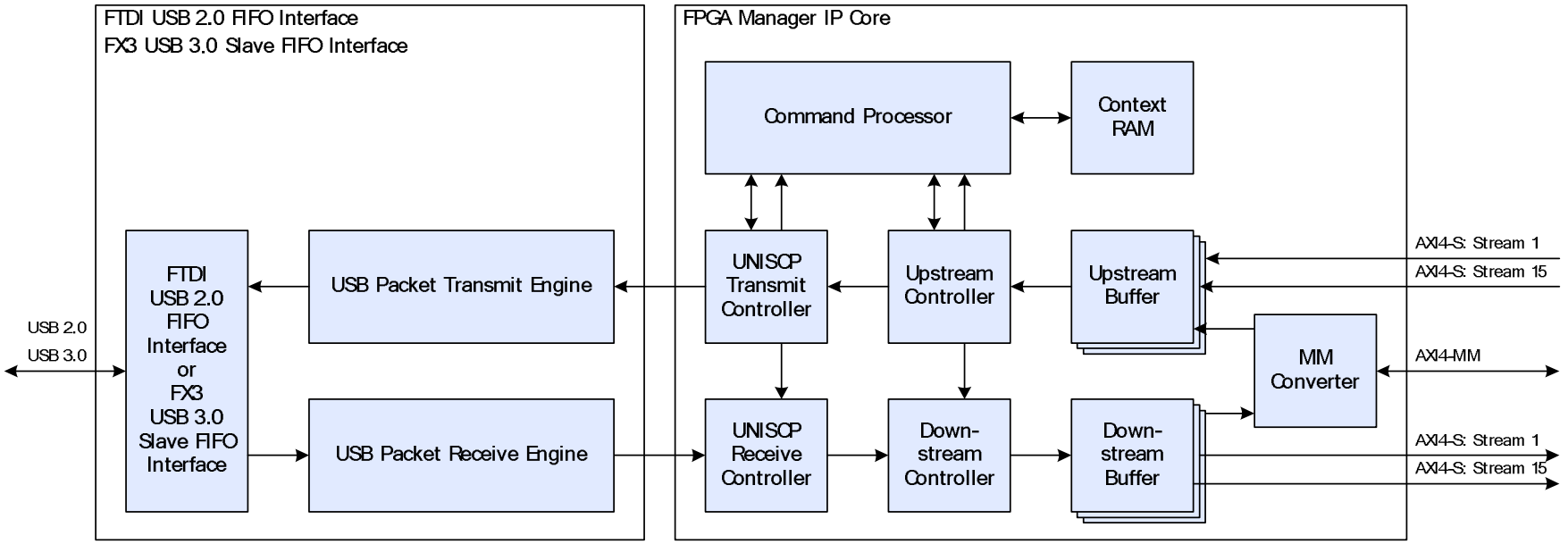
```
void main()
{
    //Create send and receive buffers
    byte[] receiveArray = new byte[4];
    byte[] sendArray = new byte[4] { 1, 2, 3, 4 };
    //Open a device with one stream
    CDevice myDevice = new CDevice("udp://192.168.33.12", 1);
    //Create Stream 0, Frame based, Upstream Enabled, Downstream Enabled
    CStream myStream = myDevice.CreateStream(0, true, true, true);
    //Open
    myDevice.Open();
    myStream.Open();
    //Blocking send
    myStream.Send(ref sendArray, null);
    //Blocking receive
    myStream.Receive(ref receiveArray, null);
    //Close (non-forcing)
    myStream.Close(false);
    myDevice.Close(false);
}
```

- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

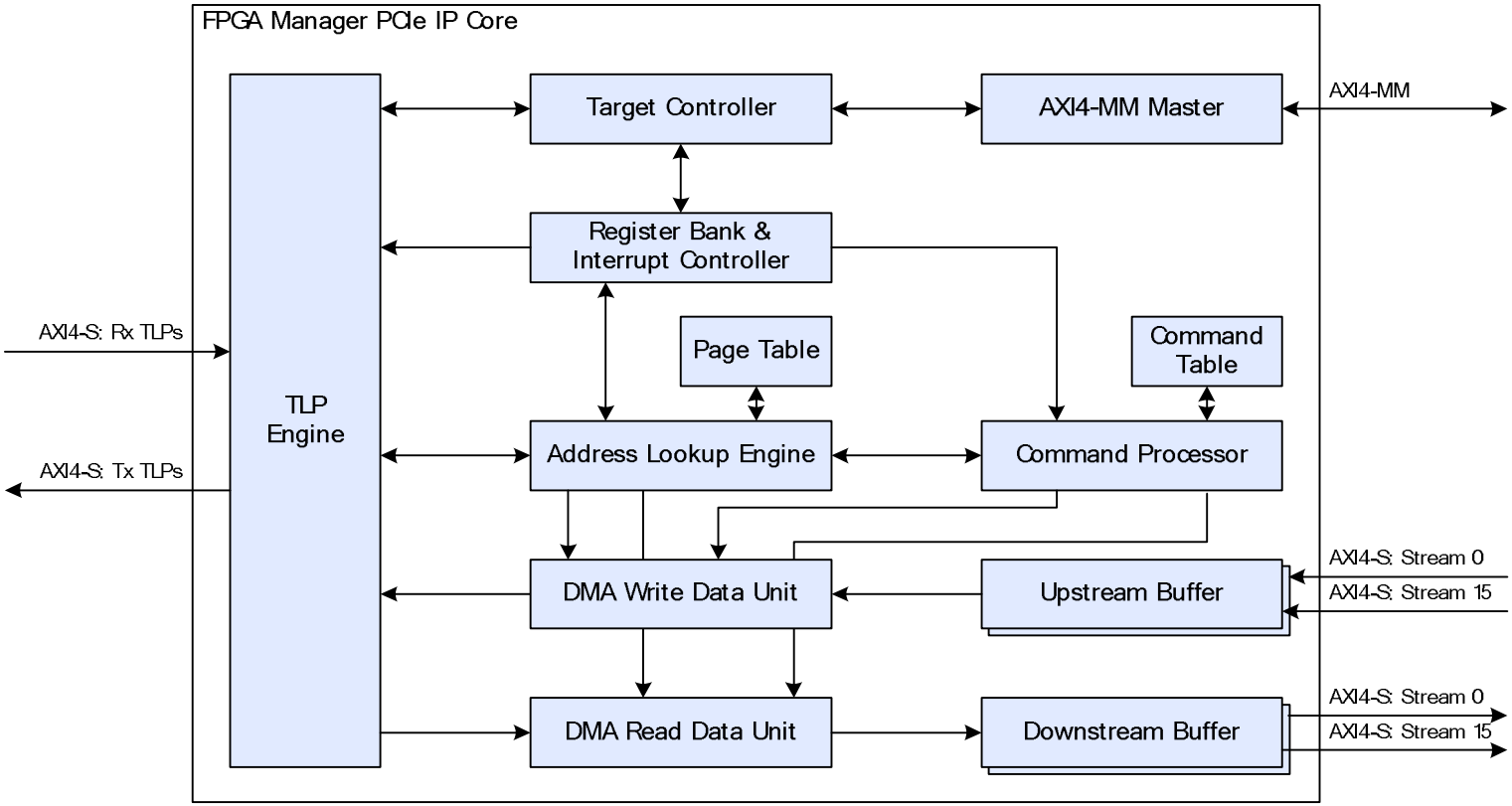
FPGAマネージャ
FPGA – イーサネットブロック図



FPGAマネージャ
FPGA – USBブロック図

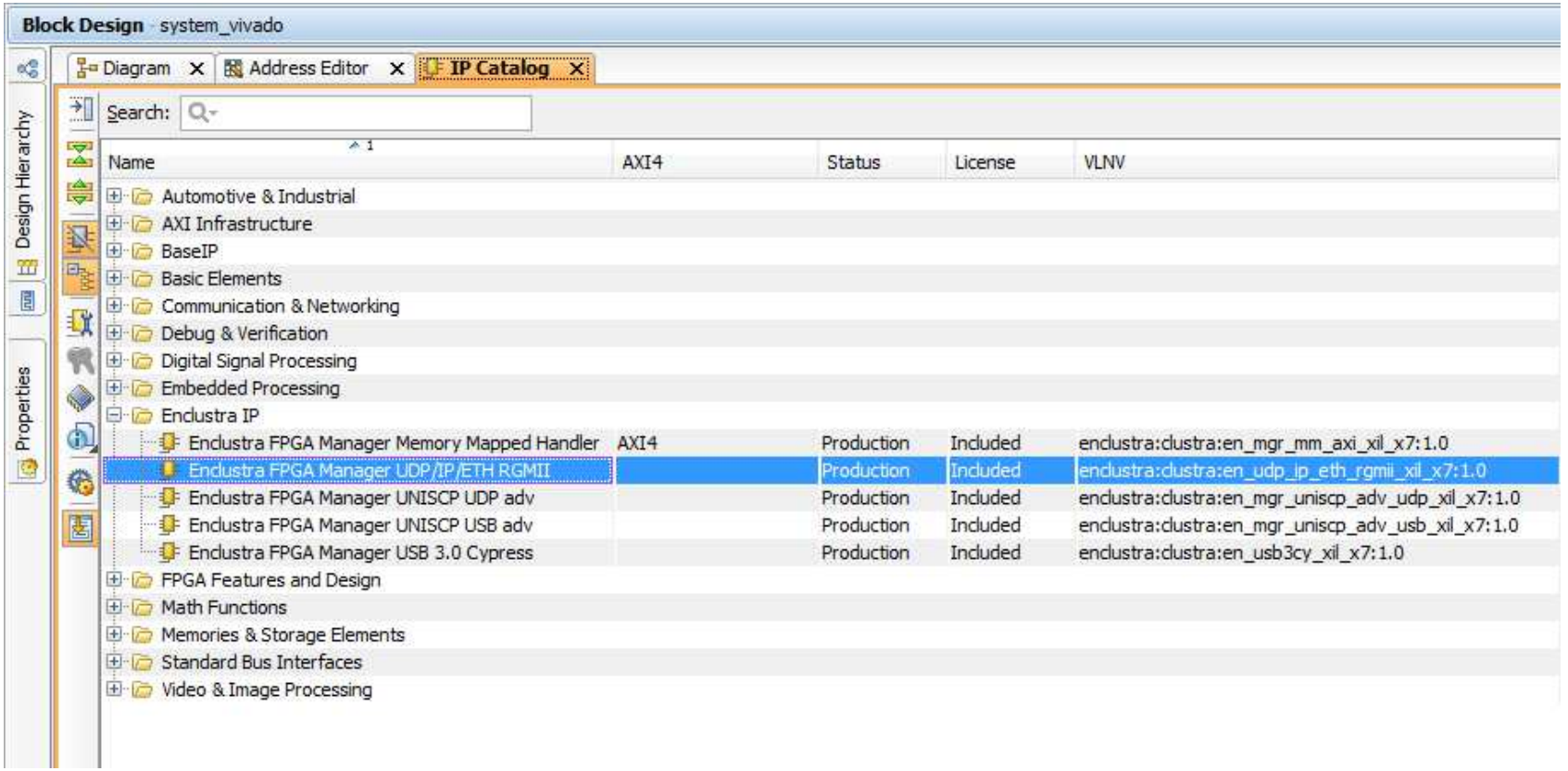


FPGAマネージャ
FPGA – PCIeブロック図

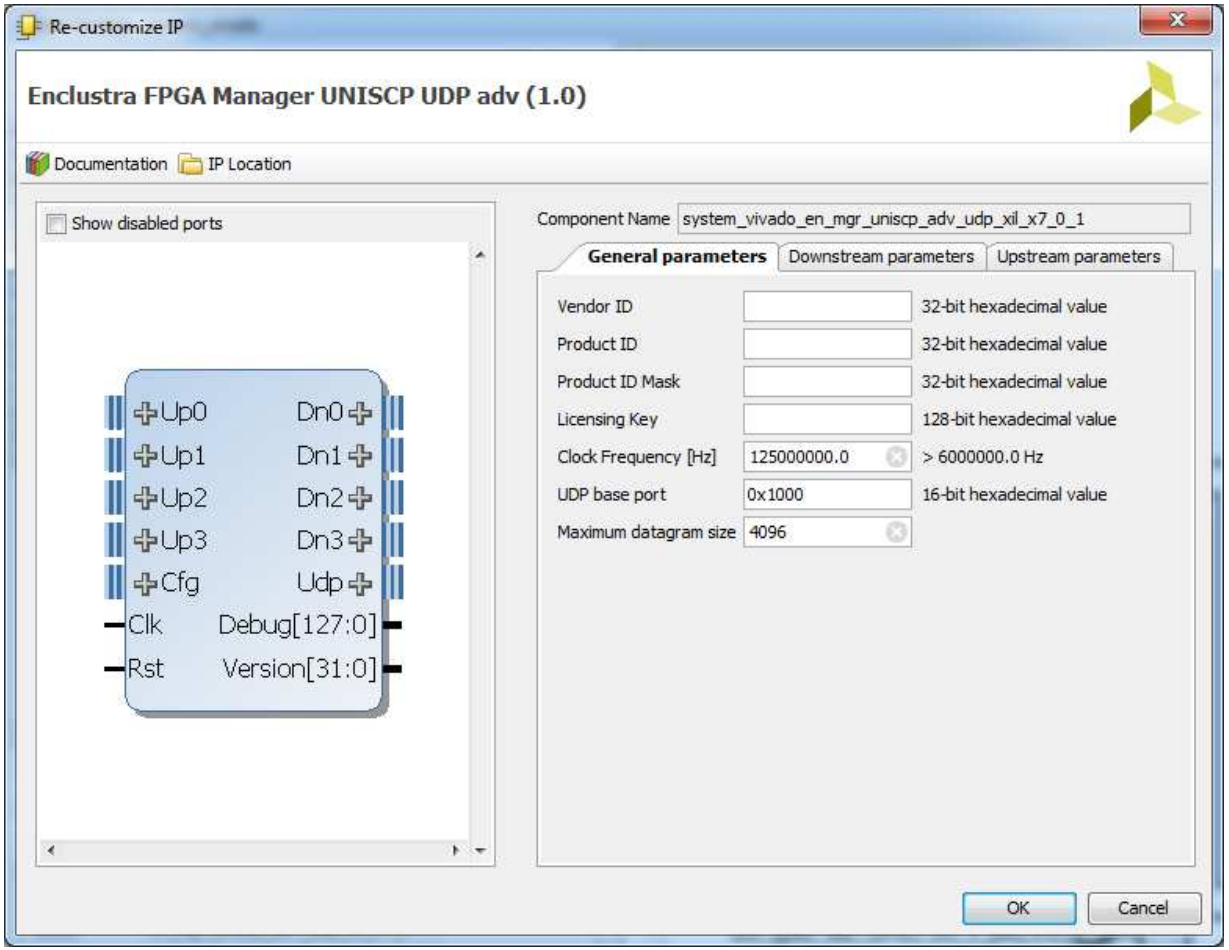


- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

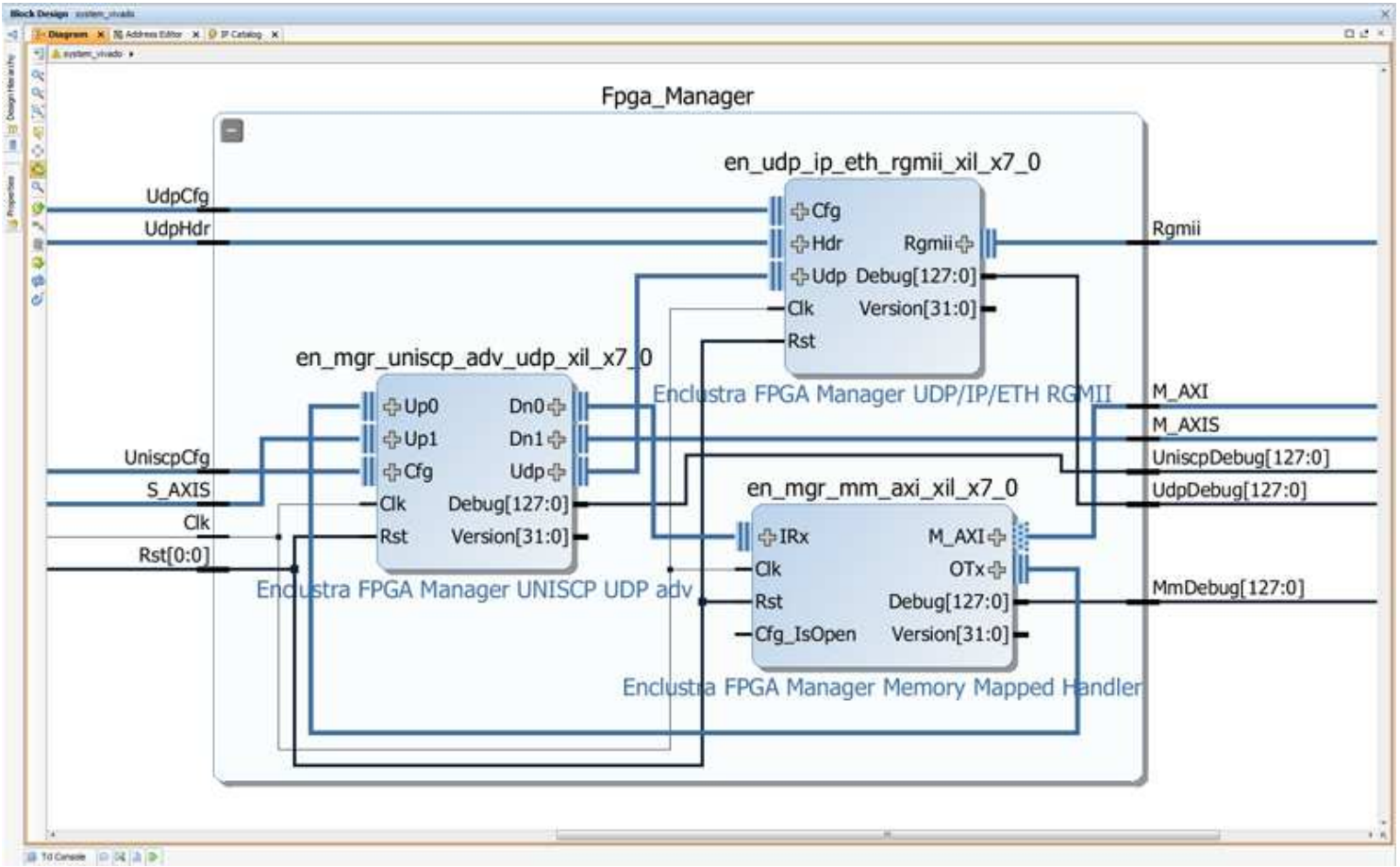
FPGAマネージャ
Xilinx Vivado IP Integratorへの組み込み



FPGAマネージャ
Xilinx Vivado IP Integratorへの組み込み



FPGAマネージャ
Xilinx Vivado IP Integratorへの組み込み



- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

FPGAマネージャ ライセンス許諾

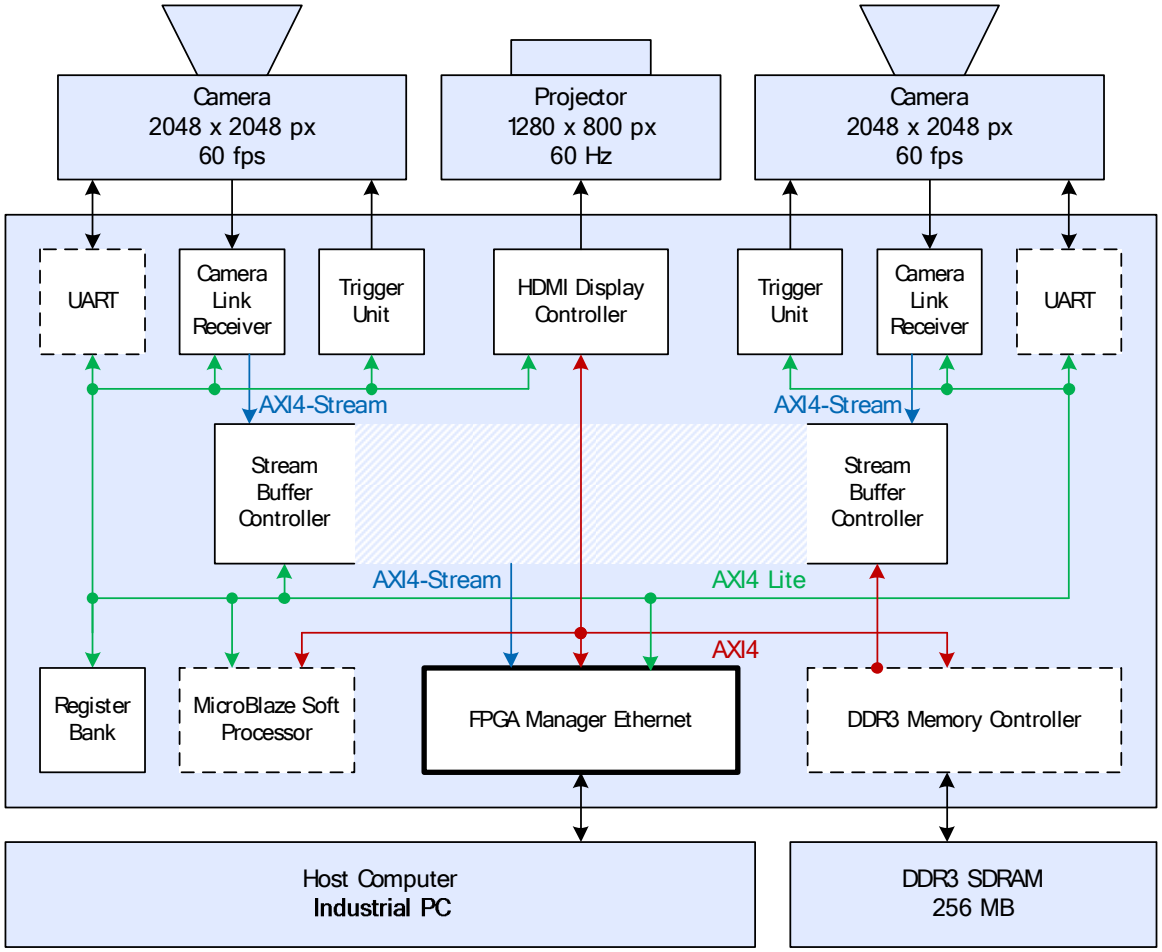
Type	Ordering Code	Description
Base	EN-MGR	FPGA Manager 2 data streams, 1 memory-mapped interface C/C++/C# language support
Link types	EN-MGR-OPT-USB2FT	USB 2.0 FTDI support
	EN-MGR-OPT-USB3CY	USB 3.0 Cypress support
	EN-MGR-OPT-ETH1G	Fast & Gigabit Ethernet support
	EN-MGR-OPT-PCIE64	PCIe 64-Bit support
Targets	EN-MGR-OPT-XIL	Xilinx FPGA support
	EN-MGR-OPT-ALT	Altera FPGA support
Operating Systems	EN-MGR-OPT-WIN	Windows operating system support
	EN-MGR-OPT-LIN	Linux operating system support
Options	EN-MGR-OPT-ADV	Advanced feature pack 16 total streams, multiple memory-mapped interfaces, multiple stream widths
	EN-MGR-OPT-ML	MATLAB language support

- フレキシブルなライセンス許諾とコスト効率のためのモジュール機能セット
- 業界標準SignOnceライセンス契約
  - ソースコードやバイナリ／暗号化のライセンスタイプ
  - プロジェクトやサイトのライセンスタイプ

SIGNONCE IP LICENSE

- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

FPGAマネージャ ユースケース – フォレンジック・アプリケーション用 3Dスキャナ



- スキャン手順
  - 干渉縞パターンの画像をオブジェクトに映し、両方のカメラを同時に作動させて両方の画像をDDR3 SDRAM に保存する(さらに多くのパターンで繰り返す)
  - ホストコンピュータ(オフライン)へ取得画像を転送
- FPGAマネージャ・イーサネット
  - メモリマップド・コンフィグレーションレジスタ・アクセスのための1つのAXI4-Lightマスタポート
  - DDR3 SDRAMへのメモリマップド・アクセスのための1つのAXI4マスタポート
  - 各カメラのために1つのAXI4-Streamのアップストリーム・ポート

- FPGAマネージャ
  - 概要
  - 凡例
  - 設計目標
  - ホスト–マスタAPIクラス図
  - ホスト–C#ストリーミング例
  - FPGA – イーサネットブロック図
  - FPGA – USBブロック図
  - FPGA – PCIeブロック図
  - Xilinx Vivado IP Integratorへの組み込み
  - ライセンス許諾
  - ユースケース – フォレンジック・アプリケーション用3Dスキャナ
  - 将来の計画
  - 評価ハードウェアと詳細情報
- 質問

FPGAマネージャ　将来の計画

- SoCエディション
  - FPGAマネージャAPI を経由するARMサブシステムとプログラマブルロジックの間の通信
  - 同一のAXI4バスポート上の最大16の独立したストリームとスキャッタ／ギャザーDMAの導入
- Androidエディション
  - Socデバイス内部のプログラマブルロジックとの通信用のAndroid JAVAレイヤのためのFPGAマネージャAPIの提供
- FPGAマネージャのスレーブAPI
  - FPGAをソフトウェアの実装と置き換え、信頼できるTCP/IP接続を経由して通信
    - ユースケース1：FPGAモデリング（ソフトウェアにおけるゴールデンモデル、ユーザソフトウェアとFPGAを並行して実装）
    - ユースケース2：ループ内のFPGA（TCP/IP経由でホストPCとSoC ARMサブシステムの間で通信）
- 追加のリンクタイプ（10 Gbpsイーサネット、PCIe Gen3 x8、USB 4.0、Thunderbolt等）
- 追加のプログラミング言語と環境
- 追加のオペレーティングシステム
- あなた自身のアイディア・・・

FPGAマネージャ
評価ハードウェアと詳細情報

- 評価ハードウェア
  - Enclustra Mercury KX1 PE1キット：Xilinx Kintex-7 FPGAとUSB 3.0、PCIe Gen2 x4とギガビット・イーサネット
  - Enclustra Mercury CA1スターターキット：Altera Cyclone IV FPGAとギガビット・イーサネット
  - Enclustra Mars ZX3 PM3キット：Xilinx Zynq-7020 SoCとUSB 3.0とギガビット・イーサネット
  - Enclustra Mars EB1 AX3キット：Xilinx Artix-7 FPGAとギガビット・イーサネット
  - Altera評価ボード
  - Xilinx評価ボード
  - 第三者ボード（例、Digilent Zedboard ）
  - あなた自身のカスタムボード（必ず私達のFPGAマネージャ・ハードウェア・デザインガイドに従ってください）

ポジティブワン株式会社
ポジティブワン・システムズ株式会社

〒150-0043 東京都渋谷区道玄坂1-12-1渋谷マークシティー・ウェスト22階
TEL 03-3256-3933　FAX 03-4360-5301
〒651-0087 兵庫県神戸市中央区御幸通8-1-6神戸国際会館 22F
TEL　078-570-5715　　FAX 078-570-5601
www.positive-one.com
Poc_sales@positive-one.com