EZ-USB™ FX3 入門

About this document

Scope and purpose

AN75705 は、サイプレス EZ-USB™ FX3 USB 3.0 デバイス コントローラーに詳しくなる手助けになります。本アプリケーションノートは、FX3 の主な機能と応用を強調しながら、FX3 の開発の手助けになる様々な設計リソースへのアクセスを提供しています。また、Eclipse 統合開発環境 (IDE) を用いたサイプレスの無料のツールを使用した FX3 ファームウェア開発を開始できるように各ステップの説明もしています。最後に、FX3 ソフトウェアとハードウェア開発キットの概要も説明しています。

関連製品ファミリ

EZ-USB™ FX3

ソフトウェア バージョン

SDK 1.3.4

関連アプリケーション ノート

AN70707

さらにサンプル コードをお求めでしょうか？以下のとおり対応いたします。

USB SuperSpeed のサンプル コードの総合リストは **http://www.cypress.com/101781** にアクセスしてください。

Table of contents

About this document **1**
Table of contents **1**
1 はじめに **4**
2 関連リソース **5**
2.1 EZ-USB™ FX3 ソフトウェア開発キット 6
2.2 GPIF II Designer 6
3 FX3 とは？ **7**
4 FX3 の特長 **8**
4.1 USB インターフェース 8
4.2 GPIF II 8
4.3 CPU 8
4.4 JTAG インターフェース 9
4.5 UART インターフェース 9
4.6 I²C インターフェース 9
4.7 I²S インターフェース 9
4.8 SPI インターフェース 9
4.9 ブート オプション 9

Table of contents

4.10 クロッキング 9
4.11 電圧ドメイン 10
5 FX3 を使用したアプリケーション開発 11
5.1 FX3 に接続した FPGA／ASIC 11
5.2 FX3 経由の FPGA のコンフィギュレーション 11
5.3 FX3 に接続した画像センサー 12
5.4 FX3 ハードウェアの設計 12
5.5 FX3 への FX2LP 設計のアップグレード 12
6 設計リソース 13
7 FX3 の用語 14
8 FX3 使用による初めての USB 3.0 転送 21
8.1.1 USBBulkLoopAuto ファームウェアのビルドと実行 26
8.1.2 FX3 RAM への USBBulkLoopAuto ファームウェアのロード 28
8.1.3 BULK 転送の実行 29
9 開発ツール 31
9.1 SuperSpeed Explorer Kit の紹介 31
9.2 FX3 SDK の紹介 32
9.2.1 ファームウェア スタックと API 33
9.2.2 フレームワーク API 33
9.3 FX3 サンプル ファームウェア 34
9.3.1 USB BULK データ ループバックの例 34
9.3.2 USB ISOCHRONOUS データ ループバックの例 34
9.3.3 スレーブ FIFO アプリケーションの例 34
9.3.4 シリアル インターフェースの例 34
9.3.5 USB BULK／ISOCHRONOUS データ ソース シンクの例 34
9.3.6 フラッシュ プログラマの例 34
9.3.7 大容量ストレージ クラスの例 34
9.3.8 USB オーディオ クラスの例 35
9.3.9 2 段起動の例 35
9.3.10 USB ホストおよび OTG の例 35
9.4 FX3 ファームウェア開発ツール 35
9.4.1 Eclipse IDE 35
9.4.2 GNU ツール チェーン 35
9.4.3 GPIF II Designer 35
9.4.4 内蔵デバッガ 35
9.5 Windows ソフトウェアの概要 36
9.5.1 Windows USB デバイス ドライバー 36
9.5.2 機能 36
9.6 アプリケーション インターフェース 37
9.6.1 CyAPI.lib 37
9.6.2 CyUSB.dll 37
9.7 Windows ソフトウェアの例 37
9.7.1 BULKLoop の例 37
9.8 Streamer の例 37
9.9 Linux 用 FX3 SDK およびソフトウェア 39
9.9.1 Linux 用 EZ-USB™ FX3 SDK 39
9.9.2 CyUSB Suite for Linux 39
9.10 有用なデバッグ ツール 40
9.10.1 USB 3.0 プロトコル アナライザ 40
9.10.2 ロジック アナライザ 40

Table of contents

10 付録 A. USB 3.0 の概要 41
10.1 電気的インターフェース 41
10.2 ケーブルおよびコネクタ 42
10.3 USB 3.0 と USB 2.0 の比較 43
11 付録 B. Windows 上の FX3 DVK ドライバー インストール 45
12 付録 C. FX3 DVK の紹介 47
12.1 FX3 DVK 用の JTAG デバッガ 48
改訂履歴 49

1 はじめに

サイプレスの EZ-USB™ FX3 (これ以下、FX3 という) は、USB 3.0 ペリフェラル コントローラーで統合された柔軟な機能を提供します。FX3 により、開発者は USB 3.0 の機能をシステムに追加することができます。

AN75705 は、FX3 の主な用途、応用および特長を強調しており、FX3 を理解する手助けになります。また、サイプレスの包括的な設計リソース リストも提供しています。本アプリケーションノートは、SuperSpeed Explorer Kit (CYUSB3KIT-003) による USB 転送を設計する手順について説明します。

付録 A は USB 3.0 の概要を説明します。**付録 B** では FX3 Windows ドライバーが未インストールの場合、そのインストール方法を説明します。FX3 DVK (CYUSB3KIT-001) の情報は [12] を参照してください。

2 関連リソース

サイプレスは **www.cypress.com** に大量のデータを掲載しており、ユーザーがデザインに適切な PSoC デバイスを選択し、デバイスをデザインに迅速で効果的に統合する手助けをしています。トレーニングビデオについては、**EZ-USB™ FX3 Architecture Overview** を参照してください (本アプリケーションノートと同じウェブページで利用可能)。

- トレーニング ビデオ: **EZ-USB™ FX3 Architecture Overview** (本アプリケーションノートと同じウェブページで利用可能)
- 知識ベース: **EZ-USB™ FX3 Knowledge Base Articles**
- EZ-USB™ FX3 の基本を理解して設計開発時間を短縮させるには、書籍 John Hyde「**SuperSpeed Device Design by Example**」を参照してください。
- 概要: **USB ポートフォリオ**、**USB ロードマップ**
- USB 3.0 製品セレクター: **FX3**、**FX3S**、**CX3**、**HX3**
- アプリケーション ノート: サイプレスは基本レベルから高度なレベルまでの様々なトピックに触れる大量の USB アプリケーション ノートを提供しています。以下は FX3 入門用の推奨アプリケーション ノートです。
  - **AN75705 – Getting Started with EZ-USB™ FX3**
  - **AN76405 – EZ-USB™ FX3 Boot Options**
  - **AN70707 – EZ-USB™ FX3/FX3S Hardware Design Guidelines and Schematic Checklist**
  - **AN65974 – Designing with the EZ-USB™ FX3 Slave FIFO Interface**
  - **AN75779 – How to Implement an Image Sensor Interface with EZ-USB™ FX3 in a USB Video Class (UVC) Framework**
  - **AN86947 – Optimizing USB 3.0 Throughput with EZ-USB™ FX3**
  - **AN84868 – Configuring an FPGA over USB Using Cypress EZ-USB™ FX3**
  - **AN68829 – Slave FIFO Interface for EZ-USB™ FX3: 5-Bit Address Mode**
  - **AN73609 – EZ-USB™ FX2LP/ FX3 Developing Bulk-Loop Example on Linux**
  - **AN77960 – Introduction to EZ-USB™ FX3 High-Speed USB Host Controller**
  - **AN76348 – Differences in Implementation of EZ-USB™ FX2LP and EZ-USB FX3 Applications**
  - **AN89661 – USB RAID 1 Disk Design Using EZ-USB™ FX3S**
  - **AN90369 – How to Interface a MIPI CSI-2 Image Sensor With EZ-USB™ CX3**
- サンプル コード:
  - **USB Hi-Speed**
  - **USB Full-Speed**
  - **USB SuperSpeed**
- テクニカル リファレンス マニュアル (TRM):
  - **EZ-USB™ FX3 テクニカル リファレンス マニュアル**
- 開発キット:
  - **CYUSB3KIT-003**、**EZ-USB™ FX3 SuperSpeed Explorer Kit**
- モデル: **IBIS**

関連リソース

2.1 EZ-USB™ FX3 ソフトウェア開発キット

サイプレスは SuperSpeed USB をさまざまな組込みアプリケーションに容易に統合するため、FX3 用の完全なソフトウェアとファームウェア スタックを提供しています。**ソフトウェア開発キット** (SDK) は、アプリケーション開発を加速させるツール、ドライバーやアプリケーション例を含みます。

Table 1 FX3 設計リソース

設計	リソース	リソースへのアクセス
FX3 ファームウェア	GCC コンパイラを備えた Eclipse IDE のインストール	**EZ-USB™ FX3 SDK** のインストールに含まれる
	様々な機能を実行するための API	
	ファームウェア例	
	SDK の使用に関する資料	**FX3 Programmer's Manual**
	SDK で提供されるすべての API に関する資料	**FX3 SDK API Guide**
ホストソフトウェア	USB 3.0 ドライバー – *cyusb3.sys*	**EZ-USB™ FX3 SDK** のインストールに含まれる
	ホスト アプリケーション例 – コントロール センターおよびストリーマー アプリケーション	
	サイプレス USBSuite アプリケーションの開発 – クイック スタート ガイド	**Cypress USBSuite Application Development - Quick Start Guide**
GPIF II インターフェース設計	GPIF II Designer ツールにより、ステート マシンを使用したカスタムの GPIF II インターフェースを設計することができる。ツールは、FX3 ファームウェアに統合される必要なコードを生成する	**EZ-USB™ FX3 SDK** のインストールに含まれる
	スレーブ FIFO、SRAM スレーブ、ADMux スレーブを含んだ一般的な GPIF II 実装例	GPIF II Designer ツール - **EZ-USB™ FX3 SDK** のインストールに含まれる
	ツールの使用方法などが記載されている GPIF II の資料	GPIF II 設計ガイド - GPIF II Designer ツールに含まれる (**EZ-USB™ FX3 SDK** のインストール)
ファームウェア デバッグ	JTAG デバッガのセットアップと使用	**FX3 Programmer's Manual** の 12 章

2.2 GPIF II Designer

GPIF II Designer は、設計者が EZ-USB™ FX3 USB 3.0 デバイス コントローラーの GPIF II インターフェースを設定できるようにするグラフィカル ソフトウェアです。

このツールでは、サイプレスが提供する 5 つのインターフェースのうちの 1 つを選ぶか、独自の GPIF II インターフェースを一から開発することができます。サイプレスは非同期と同期のスレーブ FIFO、非同期と同期の SRAM および非同期 SRAM などの業界標準インターフェースを提供します。既にあらかじめ定義されたこれらのインターフェースのいずれかをシステム内で使用している場合、設計者はバス幅（x8、x16、x32）、エンディアンおよびクロック設定など一連の標準パラメーターでインターフェースを選択して、コンパイルすることができます。このツールはインターフェースをカスタマイズしたいユーザーに対して、合理化された 3 段階の GPIF インターフェース開発プロセスを提供します。最初に、ユーザーはピン構成と標準パラメーターを選択します。次に、設定可能な操作で仮想ステート マシンを設計します。最後に、出力タイミングが期待通りになっているかを確かめます。この 3 段階のプロセスが終わると、このインターフェースをコンパイルし、FX3 と統合します。

3 FX3 とは？

FX3 は、Arm® Arm9 プロセッサを統合した USB 3.0 ペリフェラル コントローラーです。パラレルとシリアルのインターフェースはユーザー システムの他のデバイスと高速な接続ができます。

システム内の FX3 デバイスの主な機能は、カメラやスキャナなどのペリフェラルと USB ホストとの間で高帯域幅のデータ転送をすることです。強力な内蔵 Arm9 プロセッサにより、FX3 はデータ ストリームにアクセスしてデータを効果的に処理することができます。FX3 でのデータ処理が不要なシステムでは、Arm9 ファームウェアは USB とデータ消費／供給デバイスの 2 つのインターフェース間でデータ転送の初期化と管理をするだけです。

FX3 には、I^2C、SPI、UART、I2S シリアル インターフェースに加えて、第二世代汎用プログラマブル インターフェース (GPIF II) と知られている高度にフレキシブルで、プログラム可能なインターフェースがあります。GPIF II はプログラムできるため、FX3 を FPGA、画像センサ、ADC、アプリケーション プロセッサなどの各種デバイスに接続することができます。これにより、FX3 は広範囲で高性能な USB 3.0 アプリケーションに良好な選択肢となります。

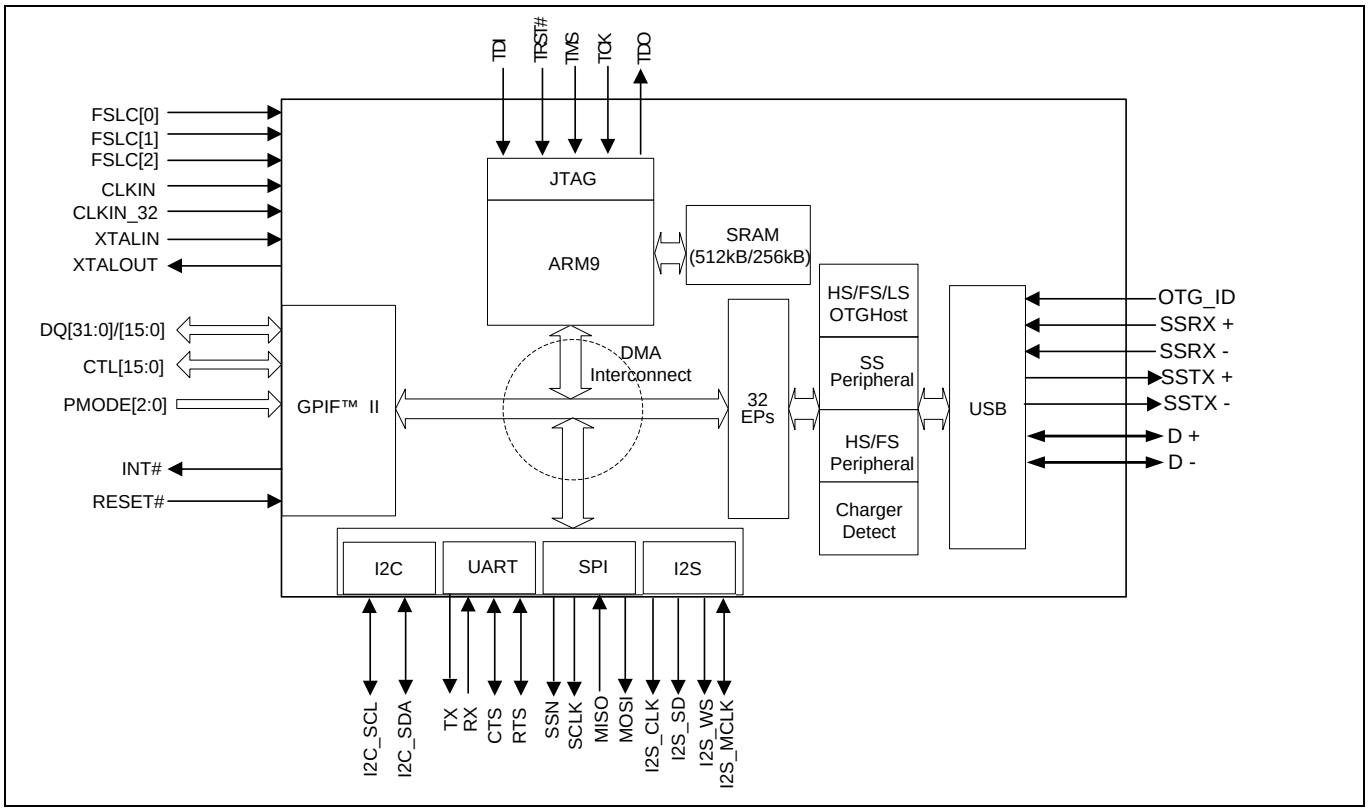


Figure 1 FX3 ブロック図

4 FX3 の特長

本節では、FX3 の主な特長を簡単に説明します。

4.1 USB インターフェース

FX3 USB インターフェースの特長は以下のとおりです。

- USB 3.0 仕様、Rev 1.0 に準拠した USB SuperSpeed と Hi-Speed ペリフェラル機能に対応。この仕様に基づいて設計された FX3 などのデバイスは、USB 2.0 との後方互換性がある
- OTG Rev 2.0 に準拠。FX3 は Hi-Speed、Full-Speed、Low-Speed OTG デュアルロール デバイス機能をサポートする。ペリフェラルとしては、FX3 は SuperSpeed、Hi-Speed、Full-Speed で動作できる。ホストとしては、Hi-Speed、Full-Speed、Low-Speed で動作できる
- CEA-936A 仕様に基づき、USB D+／D–ラインでの Carkit パススルーUART 機能に対応
- 最大 16 の入力および 16 の出力エンドポイントに対応

4.2 GPIF II

高性能 GPIF II (プロセッサ インターフェース ブロック (PIB) の一部) は、FX2LP の GPIF とスレーブ FIFO インターフェースに似ていますが、より高度な機能を持ちます。GPIF II はプログラマブル ステート マシンであり、Arm9 から独立して独自の高速クロックにて動作する柔軟性があるインターフェースを可能にします。GPIF II は、業界標準または独自インターフェースでマスタまたはスレーブとして機能します。パラレルとシリアルのインターフェースの両方を GPIF II で実装できます。

GPIF II の主な特長は以下のとおりです。

- マスタまたはスレーブとして動作
- 256 のプログラマブル ステートを提供
- 8 ビット、16 ビット、24 ビット、32 ビットのパラレル データ バスに対応
- 最大 100MHz のインターフェース周波数に対応
- 32 ビット データ バスを使用する場合、14 本の設定可能な I/O ピン (制御信号として動作) に対応。制御ピンは、入力、出力または双方向にできる
- 16／8 データ バスを使用する場合、16 本の制御 I/O ピンに対応。制御ピンは、入力、出力または双方向にできる。

サイプレスの GPIF II Designer ツールでは、GPIF II ステート マシンの迅速な開発を可能にし、共通のインターフェース サンプルを含む。**EZ-USB™ FX3 SDK** をインストールすると、GPIF II Designer ツールが使用できる

GPIF II の一般的な実装は、多くの FPGA インターフェースで使用される同期スレーブ FIFO インターフェースです。同期スレーブ FIFO インターフェースの詳細は「**AN65974 - Designing with the EZ-USB™ FX3 Slave FIFO Interface**」を参照してください。

4.3 CPU

FX3 は、32 ビット 200MHz ARM926EJ-S コア CPU を内蔵しています。コアは 16KB の命令密接合メモリ (TCM) および 8KB のデータ TCM に直接アクセスすることができます。ARM926EJ-S コアはファームウェア デバッグ用に JTAG インターフェースを備えます。

FX3 は、(選択された製品番号に応じて) 512KB または 256KB の SRAM を組込み、コードを起動する 4 つの方式 (USB、GPIF II、I^2C、SPI) をサポートします。

FX3 の特長

FX3 はさまざまなペリフェラル (USB、GPIF II、I2S、SPI、UART など) 間で効果的で柔軟性がある DMA 接続を可能にします。FX3 ファームウェアがペリフェラル間のデータ アクセスを設定した後、DMA ファブリックは Arm9 コアから独立して転送を管理します。**EZ-USB™ FX3 SDK** のインストール後に、FX3 サンプル ファームウェアが利用可能です。

4.4 JTAG インターフェース

FX3 の JTAG インターフェースは、JTAG デバッガへ接続するために標準の 5 ピン インターフェースを備えており、CPU コアのオンチップ デバッグ回路を介してファームウェアをデバッグすることができます。ARM926EJ-S コア用の業界標準デバッグ ツールは、FX3 アプリケーション開発に使用できます。

4.5 UART インターフェース

FX3 の UART インターフェースは、全二重通信に対応しており、TX、RX、CTS および RTS 信号があります。UART は FX3 ファームウェアで選択可能な 300bps～4608Kbps のボーレートを発生させることができます。

4.6 I^2C インターフェース

FX3 の I^2C インターフェースは、I^2C マスタとして動作して、I^2C スレーブ デバイスとの通信を可能にします。例えば、FX3 は I^2C インターフェースに接続している EEPROM から起動することができます。FX3 の I^2C マスタ コントローラーは、マルチマスタ機能に対応しており、I^2C クロック ストレッチを可能にします。I^2C コントローラーは、100kHz、400kHz、1MHz のバス周波数をサポートしています。

4.7 I^2S インターフェース

FX3 には、外部オーディオ コーデックをサポートする I2S トランスミッタおよび他の I2S レシーバがあります。I2S インターフェースが対応するサンプリング周波数は 32kHz、44.1kHz、48kHz です。

4.8 SPI インターフェース

FX3 は最大 33MHz の動作周波数で SPI マスタ インターフェースをサポートしています。SPI コントローラーは、4 ビット～32 ビットのトランザクション サイズで 4 つの SPI 通信モードに対応しています。

4.9 ブート オプション

FX3 は以下のソースからブート イメージをロードすることができます。

USB、I^2C、SPI、GPIF II (ブートローダでサポートされた同期 ADMux、非同期 ADMux、または非同期 SRAM インターフェースを経由)。

設計に適切なブート方式の選択は「**AN76405 - EZ-USB™ FX3 Boot Options**」で詳しく説明しています。

4.10 クロッキング

FX3 は水晶または外部クロックを接続することができます。サポートされる水晶周波数は 19.2MHz であり、外部クロックの周波数は 19.2、26、38.4 および 52MHz です。

FX3 へのクロック入力は、**EZ-USB™ FX3 Datasheet** で指定されている位相ノイズとジッタの要件を満たす必要があります。

4.11 電圧ドメイン

FX3 には、それぞれの機能ブロック (GPIF II、UART／SPI／I2S、I^2C、JTAG、USB、クロックおよびコア) 用の独立した電圧供給ドメインがあります。

FX3 への電圧入力は、**EZ-USB™ FX3 Datasheet** で指定されている要件を満たす必要があります。

FX3 ハードウェアを設計する際は「**AN70707 - EZ-USB™ FX3/FX3S Hardware Design Guidelines and Schematic Checklist**」を参照してください。

5 FX3 を使用したアプリケーション開発

本節では、FX3 のシステムで主な応用を強調します。

完全な FX3 設計は、ソフトウェア、ファームウェアおよびハードウェアの開発を含みます。サイプレスはこれらの側面に設計リソースを提供しています。FX3 設計リソースは **Table 1** に一覧表示されています。

5.1 FX3 に接続した FPGA／ASIC

いくつかのアプリケーションでは、FPGA または ASIC が高速パラレル GPIF II インターフェースを介して FX3 に接続されます。データのソースまたはシンクである別のデバイスが FPGA または ASIC に接続されます。例として、データ収集デバイス、プリンター、スキャナー、画像デバイスなどが挙げられます。

このようなアプリケーションでは、FX3 は USB ホストとの間の高速データ パイプとして動作します。これらのアプリケーションでは、通常、FX3 と FPGA／ASIC 間のインターフェースは同期スレーブ FIFO インターフェースです (**Figure 2**)。

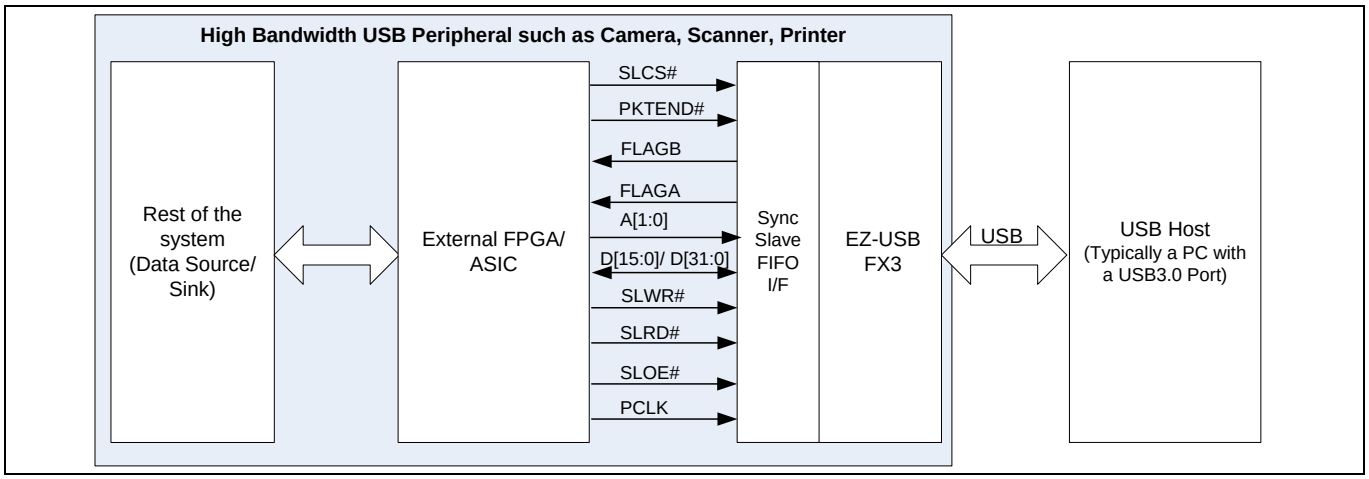


Figure 2 FX3 に接続した FPGA／ASIC

スレーブ FIFO インターフェースの詳しい説明、および FPGA が FX3 に接続している完全な設計例は「**AN65974 - Designing with the EZ-USB™ FX3 Slave FIFO Interface**」を参照してください。

GPIF II が一般的に実装されるもう 1 つのインターフェースは、同期アドレスデータ多重化インターフェースです。

5.2 FX3 経由の FPGA のコンフィギュレーション

FX3 が FPGA に接続しているアプリケーションでは、FPGA ビット ファイルを FX3 を介してロードすることが推奨されています。この場合、FPGA コンフィギュレーション ファイルは USB ホストから FX3 を介して FPGA に転送されます。これにより、FPGA ブート メモリのコストを節約でき、PC からの FPGA 再コンフィギュレーション (更新を含む) を可能にします。実装例は「**AN84868 - Configuring an FPGA over USB Using Cypress EZ-USB™ FX3**」を参照してください。

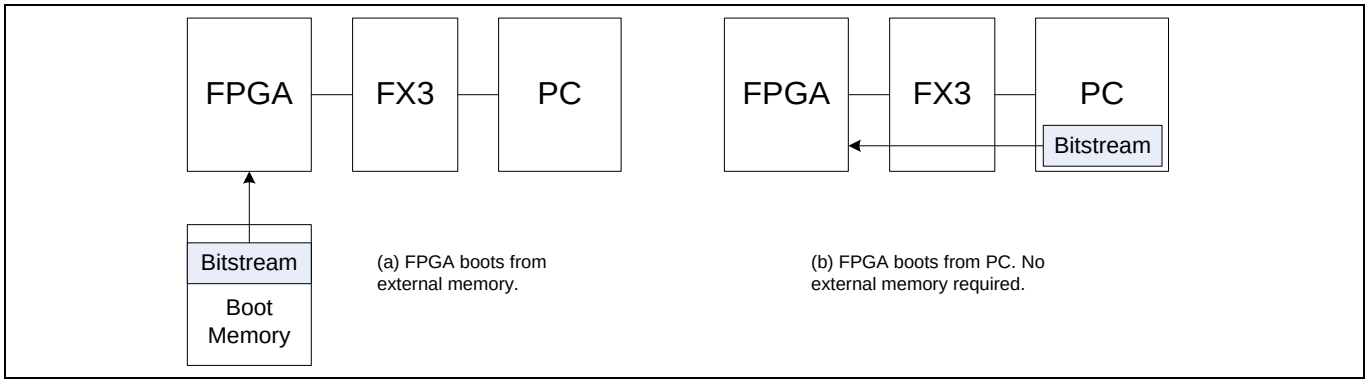


Figure 3　FX3 経由の FPGA のコンフィギュレーション

5.3　FX3 に接続した画像センサー

画像アプリケーションでは、画像センサーはパラレル GPIF II インターフェースを介して FX3 に直接接続され、ビデオはセンサーから FX3 を介して USB ホストにストリーミングされます。

システムは、適切なファームウェアで FX3 をプログラムすることにより、USB ビデオ クラスと互換性を保ちます。このようなアプリケーションの一例は「**AN75779 - How to Implement an Image Sensor Interface with EZ-USB™ FX3 in a USB Video Class (UVC) Framework**」を参照してください。

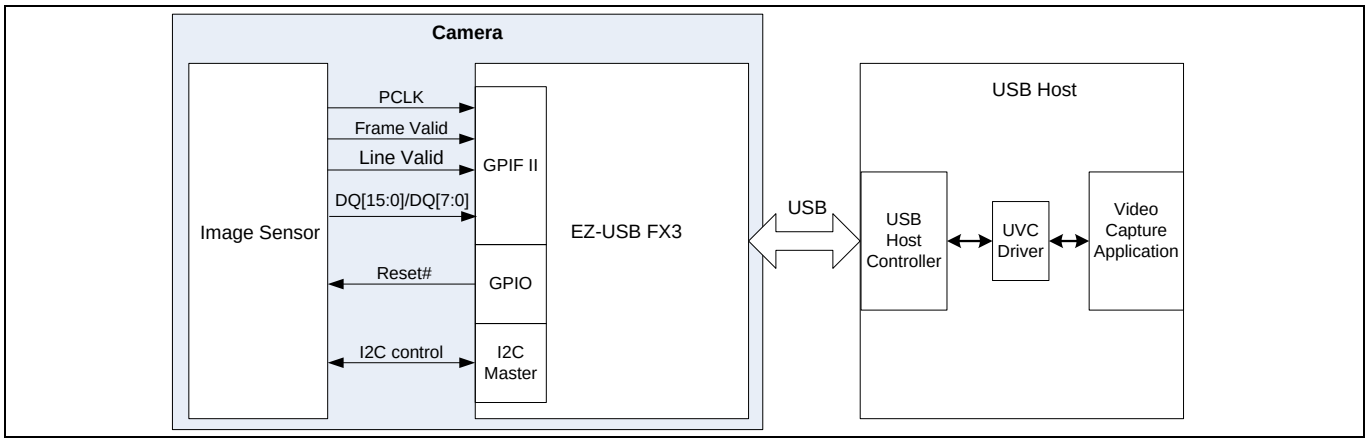


Figure 4　FX3 に接続した画像センサー

5.4　FX3 ハードウェアの設計

USB 3.0 の速度と電力のため、PC 設計とレイアウトに十分な注意を払ってください。FX3 ハードウェアを設計する際は「**AN70707 - EZ-USB™ FX3/FX3S Hardware Design Guidelines and Schematic Checklist**」を参照してください。回路図設計例は「**FX3 DVK Schematic**」を参照してください。

5.5　FX3 への FX2LP 設計のアップグレード

既存の FX2LP 設計を FX3 使用のためにアップグレードする場合は「**AN76348 - Differences in Implementation of EZ-USB™ FX2LP and EZ-USB™ FX3 Applications**」を参照してください。

6 設計リソース

Table 1 に、FX3 を使用した開発の手助けになるサイプレスのさまざまなリソースを示します。USB SuperSpeed のサンプル コードの完全な一覧については、ここをクリックしてください。

7 FX3 の用語

FX3 の入力と出力データ転送を理解するには、以下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
- GPIF スレッド

ソケットとは、ペリフェラル ハードウェア ブロックと FX3 RAM の接続点です。USB、GPIF、UART、SPI などの FX3 上の各ペリフェラル ハードウェア ブロックには、それぞれに対応する一定数のソケットがあります。各ペリフェラル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くなります。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んでいます。

DMA ディスクリプタとは、FX3 RAM に割り当てられたレジスター式です。これは DMA バッファのアドレスとサイズ情報および次の DMA ディスクリプタへのポインタを格納しています。これらのポインタは DMA ディスクリプタ チェーンを作ります。

DMA バッファとは、RAM の一部であり、FX3 デバイスを介して転送されるデータの中間記憶領域として使用されます。DMA バッファは FX3 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 ディスクリプタに格納されています。

GPIF スレッドとは、GPIF II ブロック内の専用データ経路であり、外部データ ピンをソケットに接続します。

ソケットは、イベントにより互いに直接通知するか、または割込みで FX3 CPU に通知します。ファームウェアは信号方式を設定します。GPIF II ブロックから USB ブロックまでのデータ ストリームを例に取ってみます。GPIF ソケットはデータで DMA バッファが一杯になったことを 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 チャネルと呼ばれています。自動 DMA チャネルの実装は通常、FX3 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ています。

一方、GPIF ソケットは FX3 CPU に割込みを送信して GPIF ソケットが DMA バッファを満たしたことを通知することもできます。FX3 CPU はこの情報を USB ソケットに伝達します。USB ソケットは、FX3 CPU に割込みを送信して USB ソケットが DMA バッファを空にしたことを通知することができます。そして、FX3 CPU はこの情報を GPIF ソケットに伝達します。これは、手動 DMA チャネル実装と呼ばれています。この実装は通常、FX3 CPU がデータ ストリームでデータを追加したり、除去したり、修正する必要がある場合に使用されています。**UVC アプリケーション ノート (AN75779)** 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 ソケットと呼ばれています。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 ソケットと呼ばれています。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 バッファが一杯か空になった後、ある DMA ディスクリプタから別の DMA ディスクリプタに切り替えるために限られた時間 (数マイクロ秒) かかります。切り替え中にソケットはデータを転送できません。この待ち時間は、フロー制御を備えていないインターフェースにとって問題となります。一例としては、画像センサー インターフェースを挙げられます。

この課題は GPIF II ブロック内で、複数の GPIF スレッドを使用して対処できます。GPIF II ブロックは 4 つの GPIF スレッドを実装しています。一度に 1 つの GPIF だけがデータを転送できます。GPIF II ステート マシンはデータを転送するためにアクティブな GPIF スレッドを選択する必要があります。

GPIF スレッド選択メカニズムはマルチプレクサに似ています。GPIF II ステート マシンは、内部制御信号または外部入力 (**スレーブ FIFO 2 ビットのアプリケーションノート**ではアドレス ライン A1、A0) を使用し、アクティブな GPIF スレッドを選択します。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。GPIF スレッドの切り替えに待ち時間はありません。

ソケットと GPIF スレッドのデフォルトのマッピングは、**Figure 5** に示されています。ソケット 0 は GPIF スレッド 0 に、ソケット 1 は GPIF スレッド 1 に、ソケット 2 は GPIF スレッド 2 に、ソケット 3 は GPIF スレッド 3 に対応します。

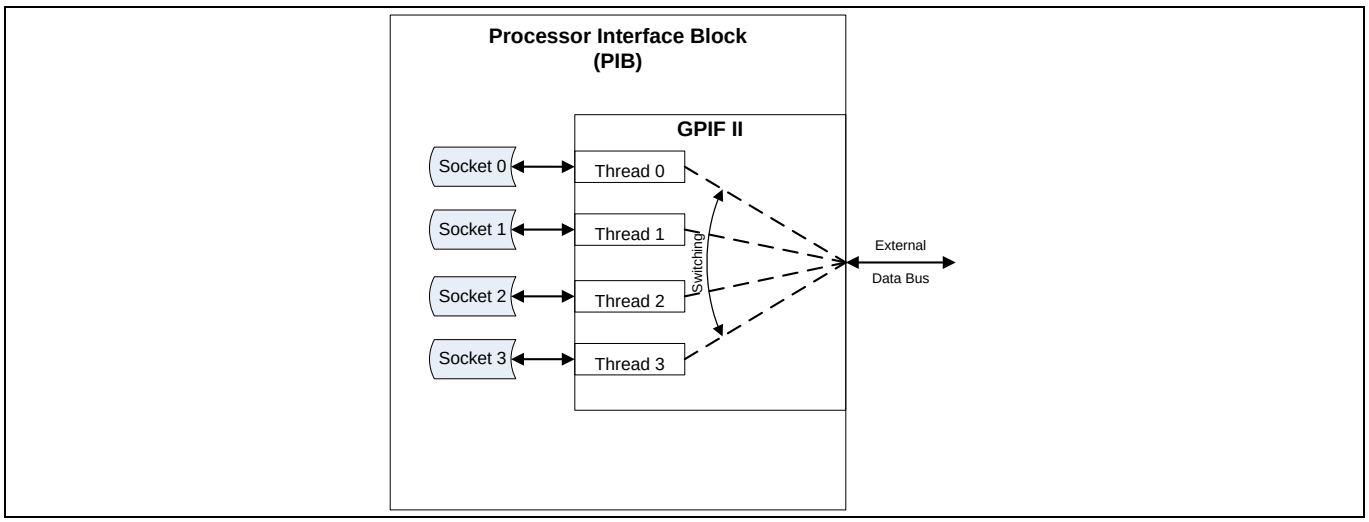


Figure 5　デフォルトの GPIF II ソケットとスレッドのマッピング

以下の 4 つの図に、DMA 転送を理解するためのソケットの概念を示します。

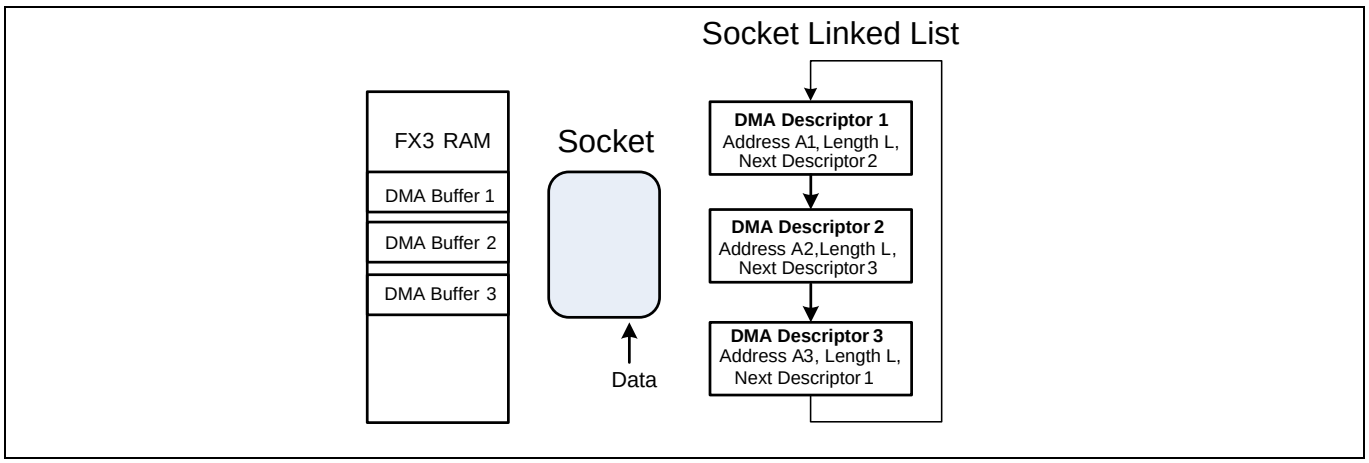


Figure 6　DMA ディスクリプタ リストに従ってデータを送信するソケット

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 メモリにあるデータ構造の一式です。それぞれのディスクリプタは、DMA バッファのアドレスと長さおよび次の DMA ディスクリプタへのポインタを指定します。ソケットが動作するとき、DMA ディスクリプタを個別に読み出し、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 バイトが転送されると、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、任意の数の DMA バッファをメモリ内の任意の場所に作成することができ、自動的に一緒に連結することができるため、ソケットを汎用的に使用することができます。例えば、**Figure 7** 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。

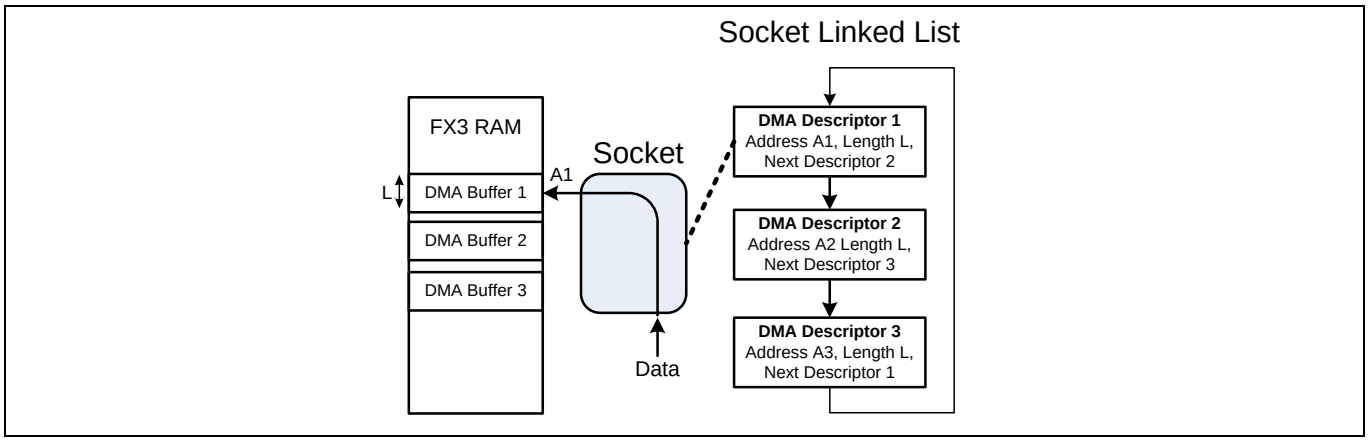


Figure 7　DMA ディスクリプタ 1 で動作するソケット

Figure 7 では、ソケットは DMA ディスクリプタ 1 をロードします。DMA ディスクリプタ 1 は、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。そのとき、ソケットは DMA ディスクリプタ 2 を読み出します。その後、アドレス A2 と長さ L の設定で続きます (**Figure 8** を参照してください)。

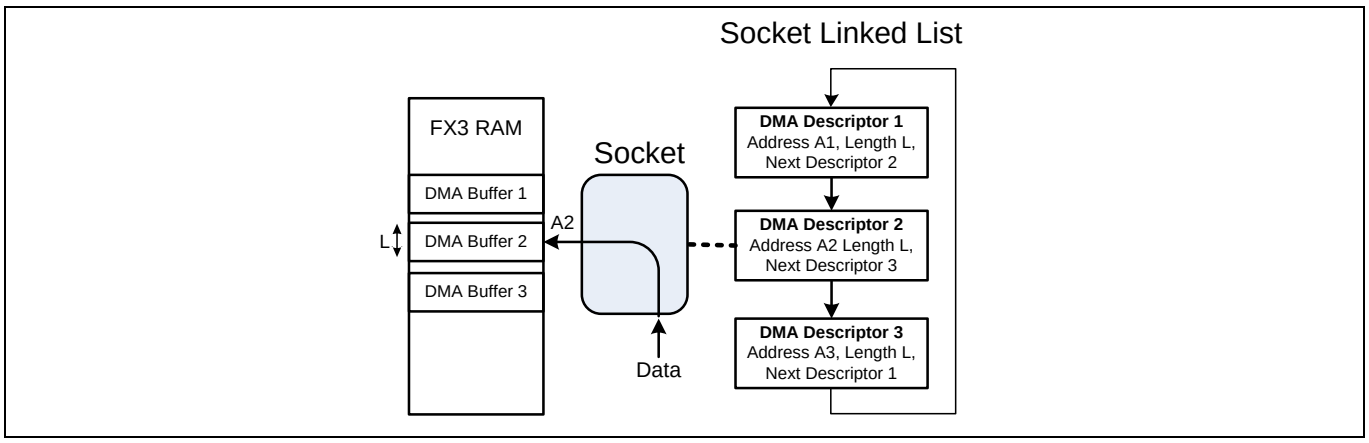


Figure 8　DMA ディスクリプタ 2 で動作するソケット

Figure 9 では、ソケットは 3 番目の DMA ディスクリプタを読み出し、A3 から始まるデータを転送します。L バイトを転送した後、シーケンスは DMA ディスクリプタ 1 で繰り返します。

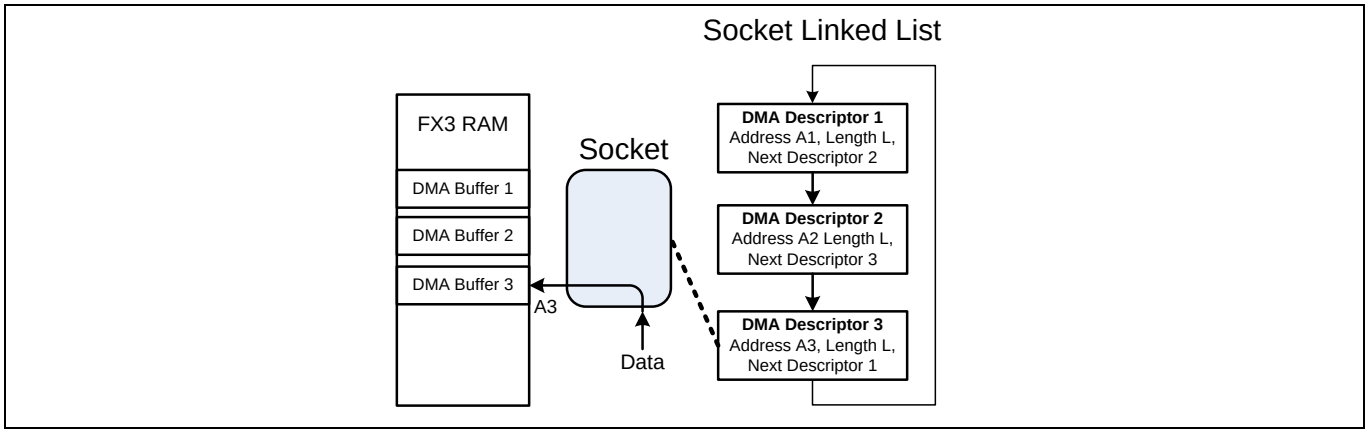


Figure 9　DMA ディスクリプタ 3 で動作するソケット

Figure 10 に、詳細な DMA データ転送を示します。循環ループで連結した長さ L の 3 つの DMA バッファを使用するアプリケーションを例に取ってみます。FX3 のメモリ アドレスは左側です。青い矢印は、メモリからソケット リンク リスト ディスクリプタをロードしていることを示します。赤い矢印は、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 バッファに移動されるときのソケット 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 から始まる 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 L のデータを転送した後、ステップ 3 に進みます。

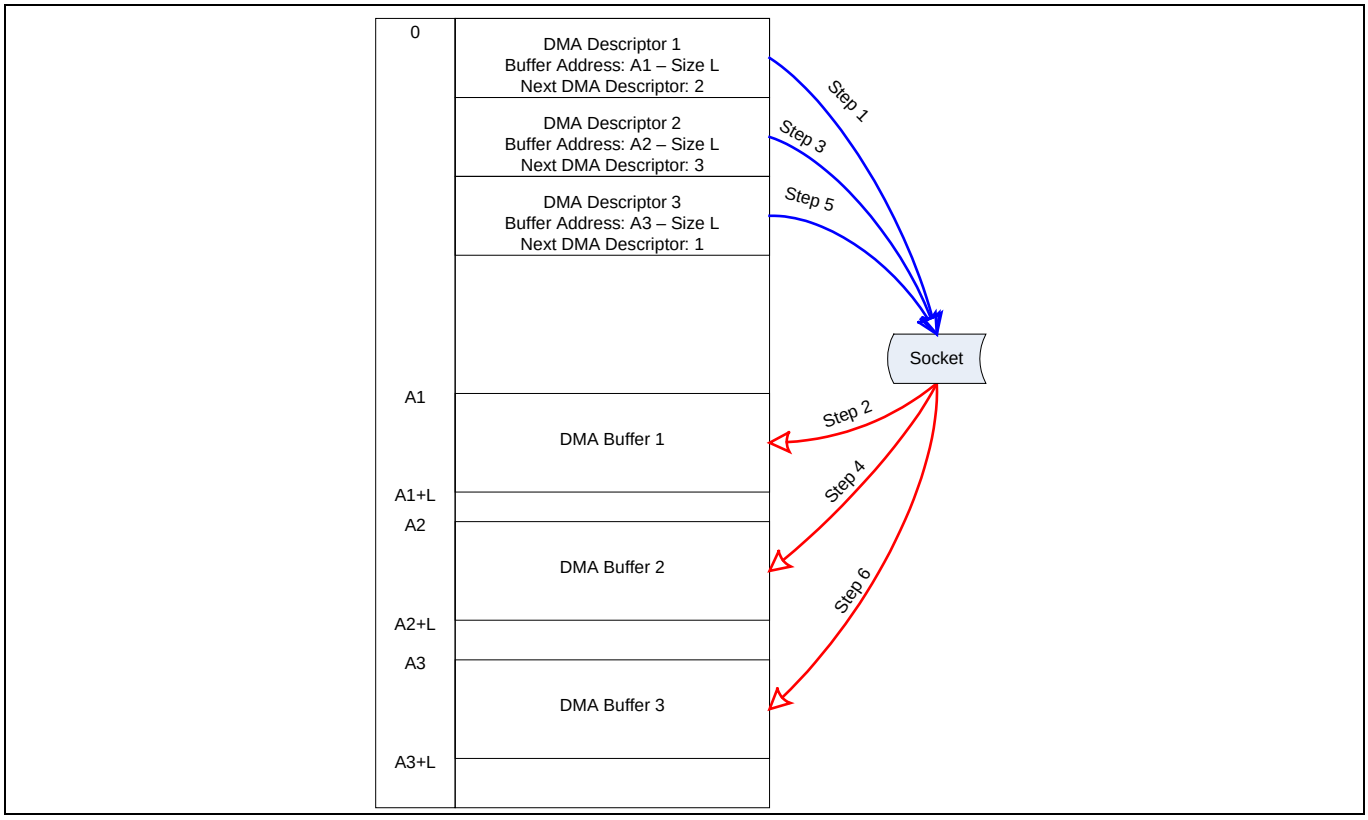


Figure 10　DMA 転送の例

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: A2 から始まる 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 L のデータを転送した後、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: A3 から始まる 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 L のデータを転送した後、ステップ 1 に進みます。

この簡単な方式は、ソケットがメモリから次の DMA ディスクリプタを読み出すときに (通常 1 マイクロ秒)、外部のペリフェラルがデータを送信すると、データ損失を引き起こします。より良い解決策は、ソケットが 1 クロック サイクル以内に待ち時間なしに切り替わるという利点を利用することです。従って、外部のペリフェラルがフロー制御メカニズムを備えていない場合、2 つのソケットを使用することは良い方法です。デュアル ソケットを使用するデータ転送は、実行ステップ番号付きで **Figure 11** 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、それぞれ赤い矢印と緑の矢印 (個々のソケットのデータ パス) で区別されています。各ステップの「a」と「b」の部分は同時に起こります。このハードウェアの並列動作により、DMA ディスクリプタの読み出しデッドタイムがなくなり、GPIF II はデータを連続して内部メモリに流すことができます。

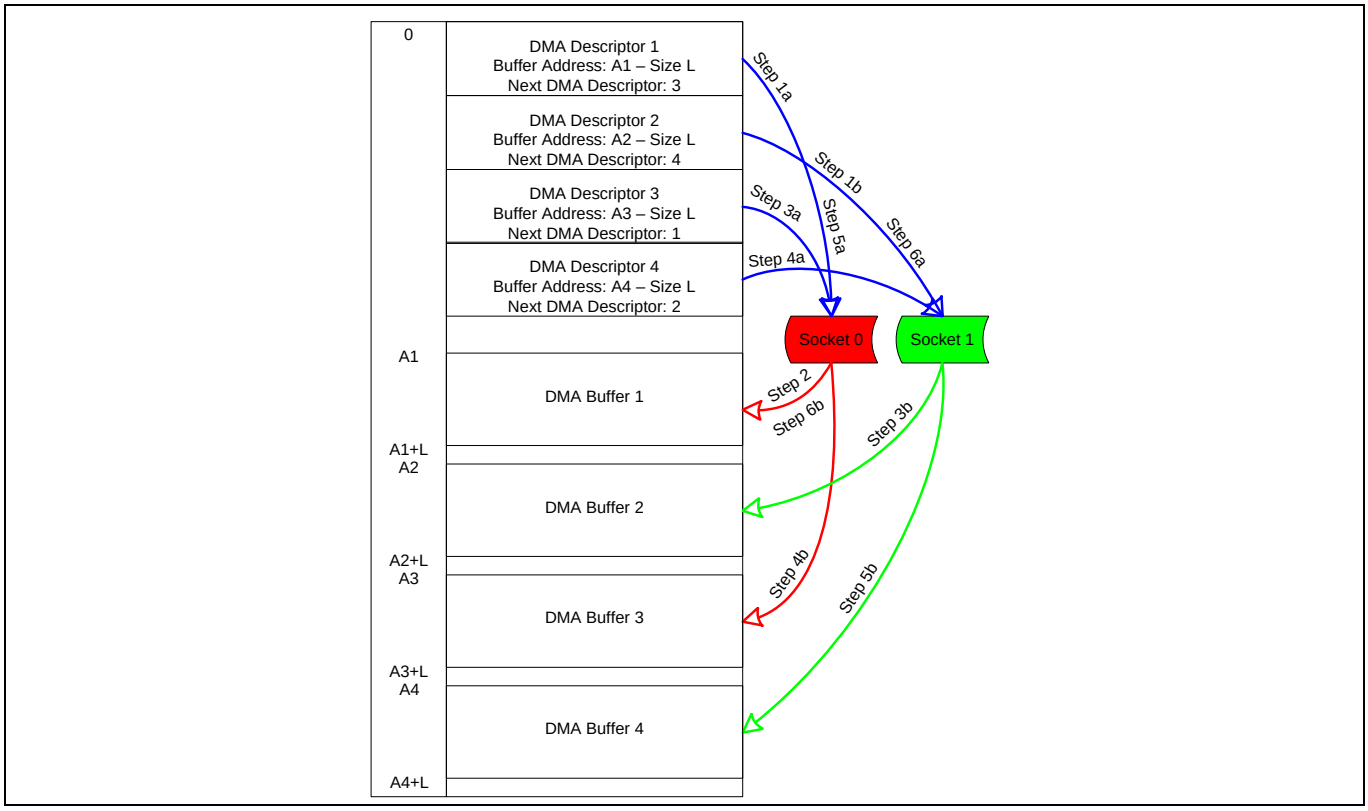


Figure 11　デュアル ソケットにより継ぎ目のない転送を実現する

ステップ 1: ソケット 0 およびソケット 1 の初期化時に、それぞれの 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: データが使用可能になるとすぐに、ソケット 0 はデータを DMA バッファ 1 に転送します。転送長は L です。この転送が終わった後、ステップ 3 に進みます。

ステップ 3: GPIF II は GPIF スレッドを切り替えて、従って、データ転送用のソケットも切り替わります。ソケット 1 はデータを DMA バッファ 2 に転送開始し、それと同時にソケット 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 のデータ量を転送し終えるまでに、ソケット 0 はデータを DMA バッファ 3 に転送する準備が終わります。

ステップ 4: GPIF II はこの時点で元の GPIF スレッドに戻ります。ソケット 0 はこの時点で長さ L のデータを DMA バッファ 3 に転送します。それと同時に、ソケット 1 は DMA ディスクリプタ 4 をロードし、データを DMA バッファ 4 に転送する準備を終えます。ソケット 0 が長さ L のデータの転送を終了した後、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 に転送します。それと同時に、ソケット 0 は DMA ディスクリプタ 1 をロードし、DMA バッファ 1 にデータを転送する準備を終えます。ステップ 5a は、ソケット 1 が初期化されることではなく、データを同時に転送することを除いて、ステップ 1a と同じであることに注意してください。

ステップ 6: GPIF II はソケットを再び切り替え、ソケット 0 は長さ L のデータを DMA バッファ 1 に転送開始します。ここまで、DMA バッファが UIB コンシューマ ソケットによって使い果たされ、空になっていると想定しています。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、データを DMA バッファ 2 に転送する準備を終えます。この時点で、実行パスのステップ 3 に進みます。

FX3 の用語

コンシューマ側 (USB) が GPIF II から次のデータ チャンクを受信するために間に合って DMA バッファを空にして解放した場合にのみ、GPIF II ソケットはデータを転送することができます。コンシューマの速度が十分に速くない場合、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。

8 FX3 使用による初めての USB 3.0 転送

本節では、USB ホストから FX3 デバイスに BULK OUT と BULK IN の転送を実行する簡単なサンプル ファームウェアを作成し、実行します。

この演習を始める前に、以下を実施してください。

1. **EZ-USB™ FX3 SuperSpeed Explorer Kit (CYUSB3KIT-003)** を入手します。この演習では、FX3 ファームウェアは USB ホストからダウンロードされるため、SuperSpeed Explorer Kit 基板を USB ブートに設定する必要があります。USB ブートを選択するために、すべてのジャンパを **Figure 12** に示すように閉じます。これらのジャンパ設定は、赤い長方形または円形で示されています。キット基板の詳細は**キット ユーザー ガイド**を参照してください。SuperSpeed Explorer Kit 基板は USB ケーブルでのみ電源供給されます (バス給電)。

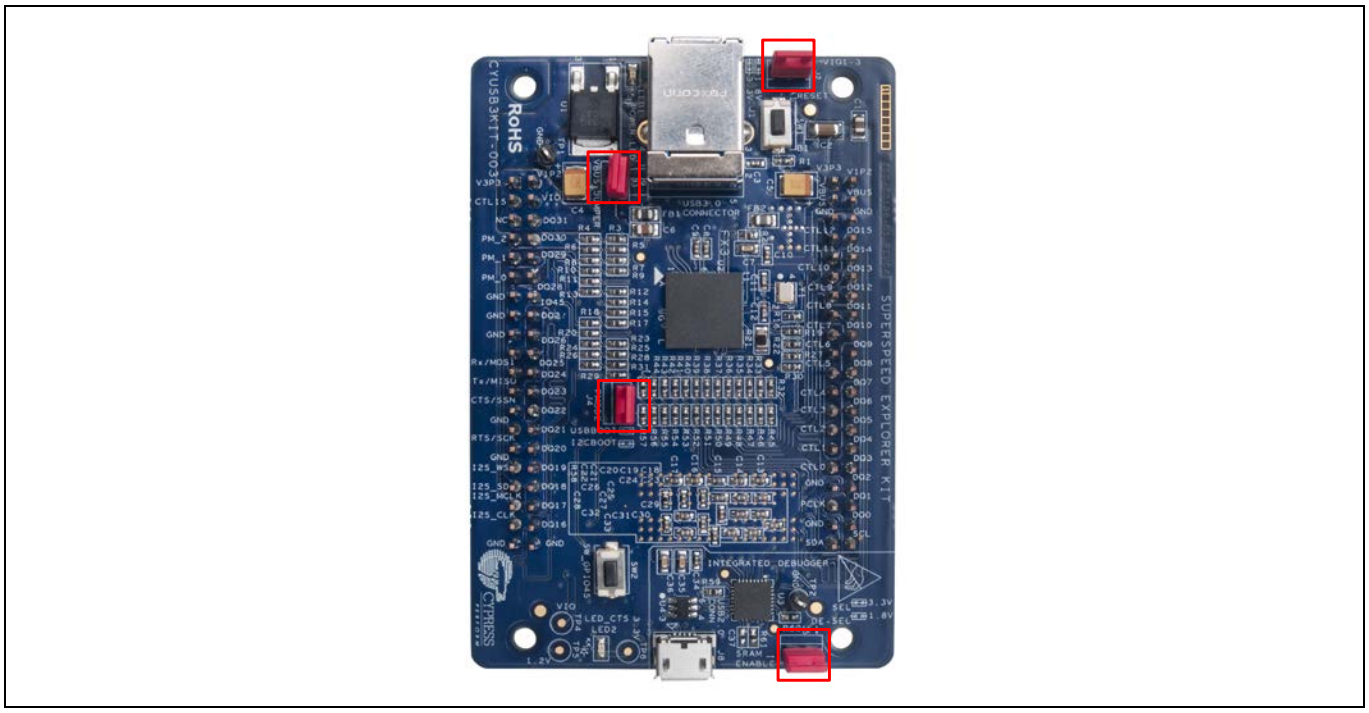


Figure 12 USB ブート用の SuperSpeed Explorer Kit 基板設定

2. **EZ-USB™ FX3 SDK** をインストールします。**Installation Type** にデフォルトの **Typical** (標準) 設定を選択します。FX3 SDK のインストール後、サンプル ファームウェアを Eclipse IDE ワークスペースにインポートします (Eclipse IDE のインストールは SDK に含まれます)。IDE にすべてのサンプル ファームウェアをインポートする手順は、以下のとおりです。

   a) Start メニューに移動し、Eclipse IDE を開きます。**All Programs** > **Cypress** > **Eclipse** > **EZ USB Suite** を選択します。デフォルトのワークスペース フォルダを使用します。ただし、後でご自身で場所を設定することができます。

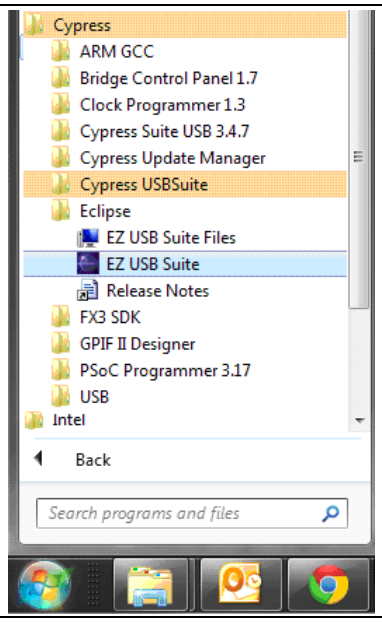


Figure 13　Start メニューから Eclipse IDE を開く

b) 初めて Eclipse IDE を開く際、**Figure 14** に示すように、すべての Eclipse プロジェクトを格納する Workspace フォルダを作成するようプロンプトが表示されます。Workspace を作成する場所を覚えておいてください。コンパイルするすべてのファームウェアは Workspace フォルダに配置されます。

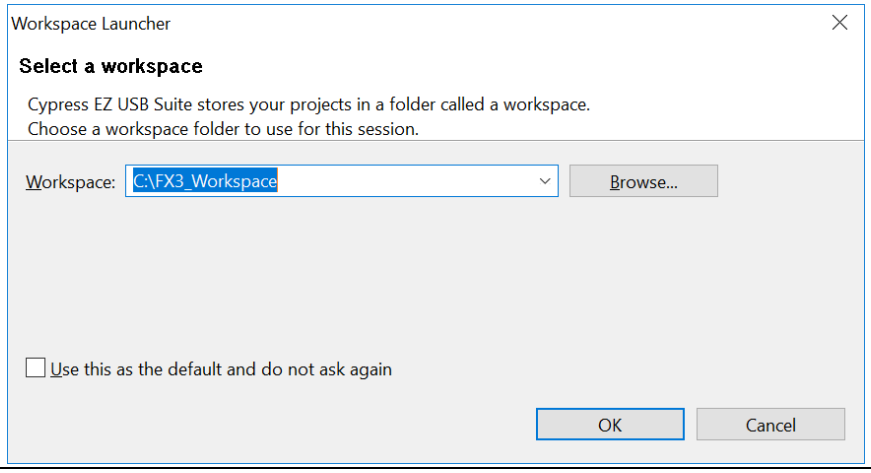


Figure 14　Workspace フォルダの作成

c) これは、「Project Explorer」タブにプロジェクトがない空の Eclipse IDE を開きます。次に、サイプレス サンプル プロジェクトを「Project Explorer」にインポートします。

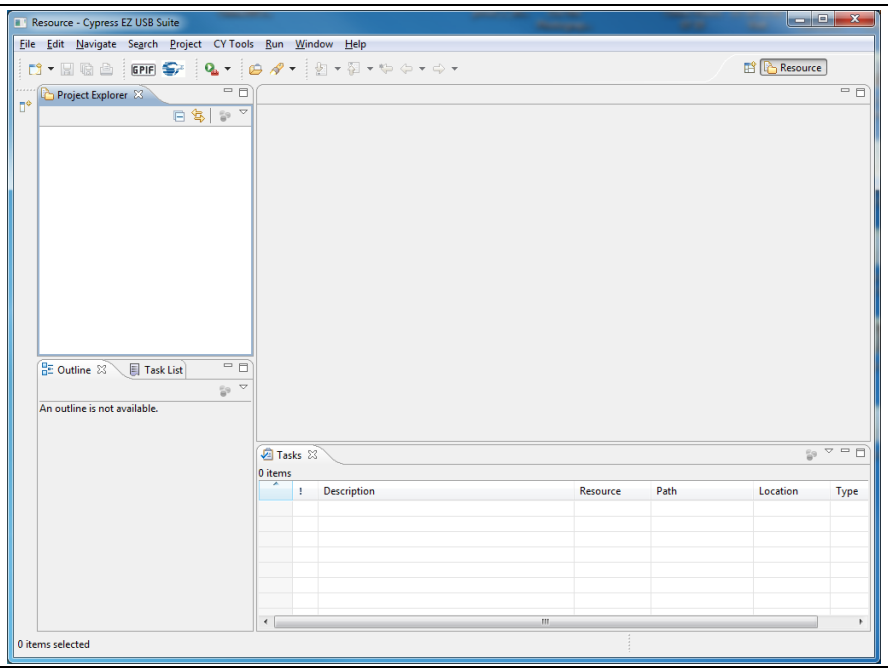


Figure 15　プロジェクト インポート前の Eclipse IDE

d) **File** > **Import** を選択します。

e) **General** > **Existing Projects into Workspace** を選択し、**Next** をクリックします。

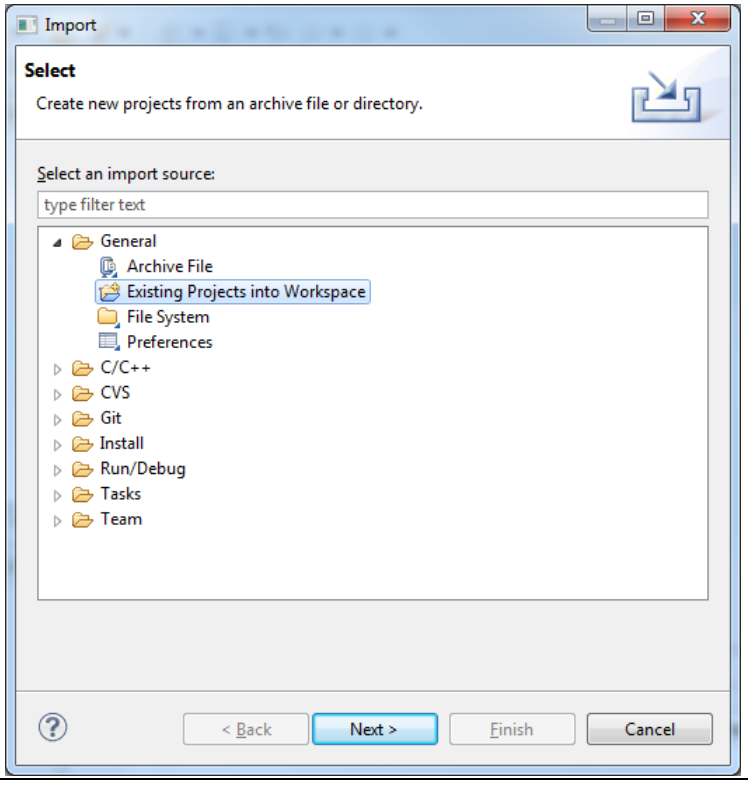


Figure 16　「Existing Projects into Workspace」を選択する

f) SDK インストールでファームウェア ディレクトリをブラウズします。**Browse...**ボタンをクリックして、サイプレス SDK**firmware** フォルダに移動します。標準的な Windows インストールの場合、このフォルダは C:\Program Files (Program Files (x86 on 64-bit machines)\Cypress\EZ-USB FX3 SDK\firmware に配置されています。

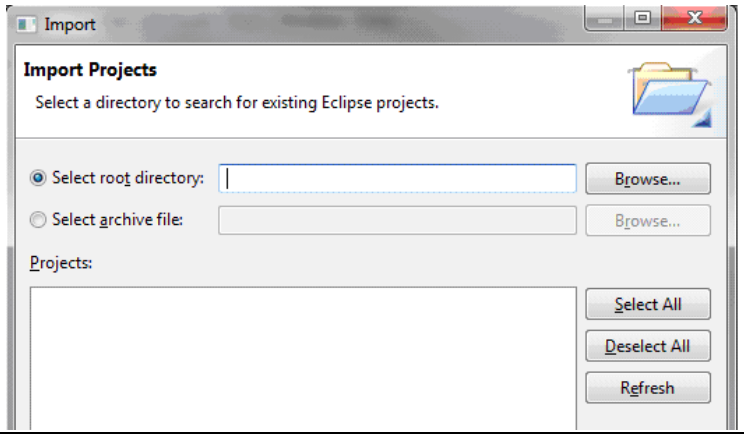


Figure 17　ルート ディレクトリを選択する

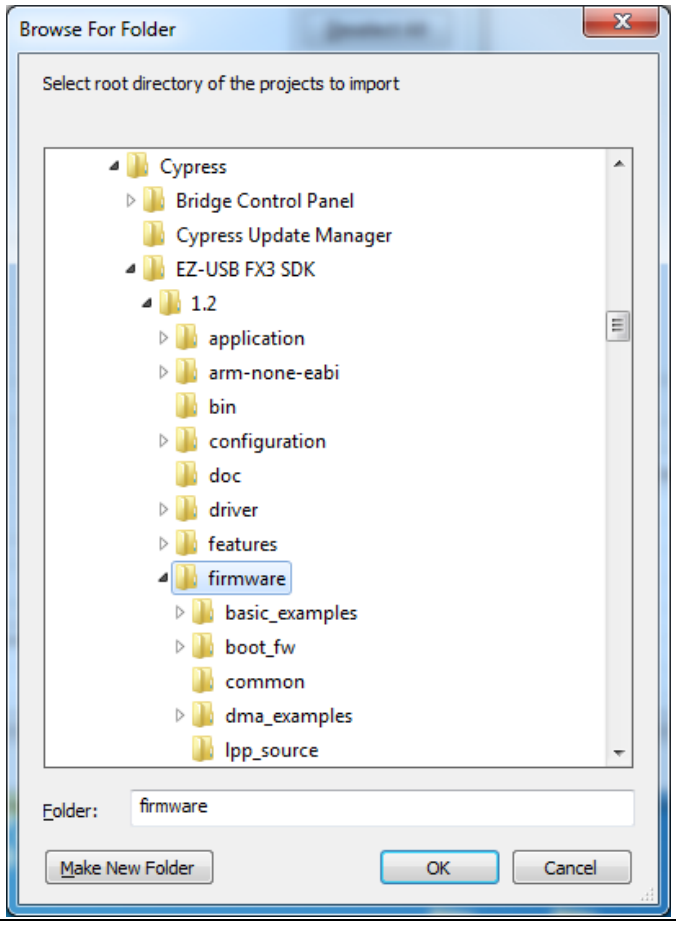


Figure 18　SDK インストールで「firmware」ディレクトリをブラウズする

g) すべてのサイプレス サンプル プロジェクトをチェックマークで選択します。あるいは、**Select All** ボタンをクリックします。また、**Copy projects into workspace** にチェックマークを付けて、**Finish** をクリックします。

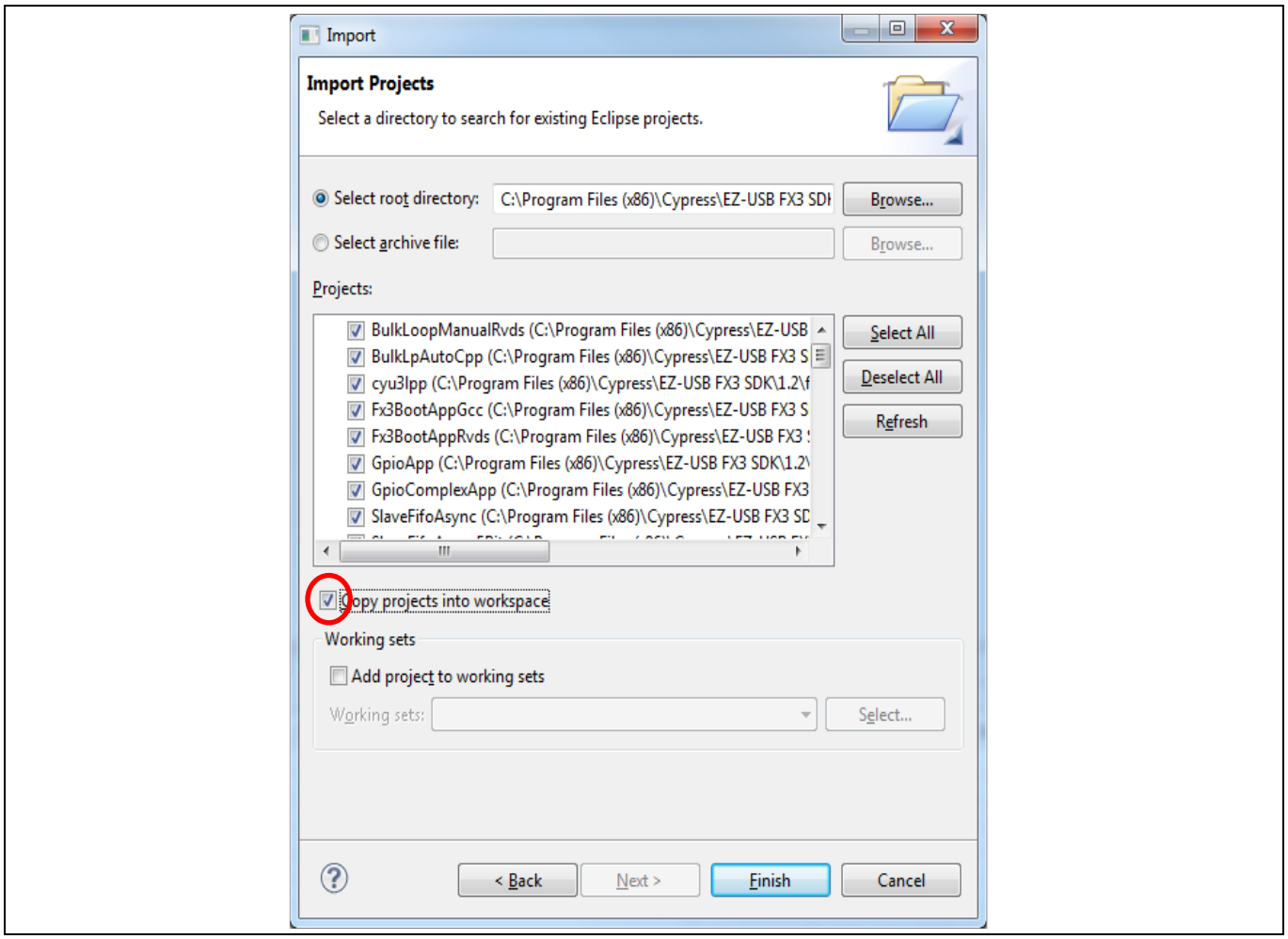


Figure 19　ファームウェア ディレクトリですべてのプロジェクトを選択する

h) すべてのサンプル ファームウェアはワークスペースにインポートされた後、Eclipse の「Project Explorer」に表示されます。すべてのプロジェクトはインポート時に自動的にビルドされたため、インポート操作は全部をロードしてビルドするのに数分かかります。ロード後に、それらをそのまま実行するか、または変更して再ビルドすることができます。

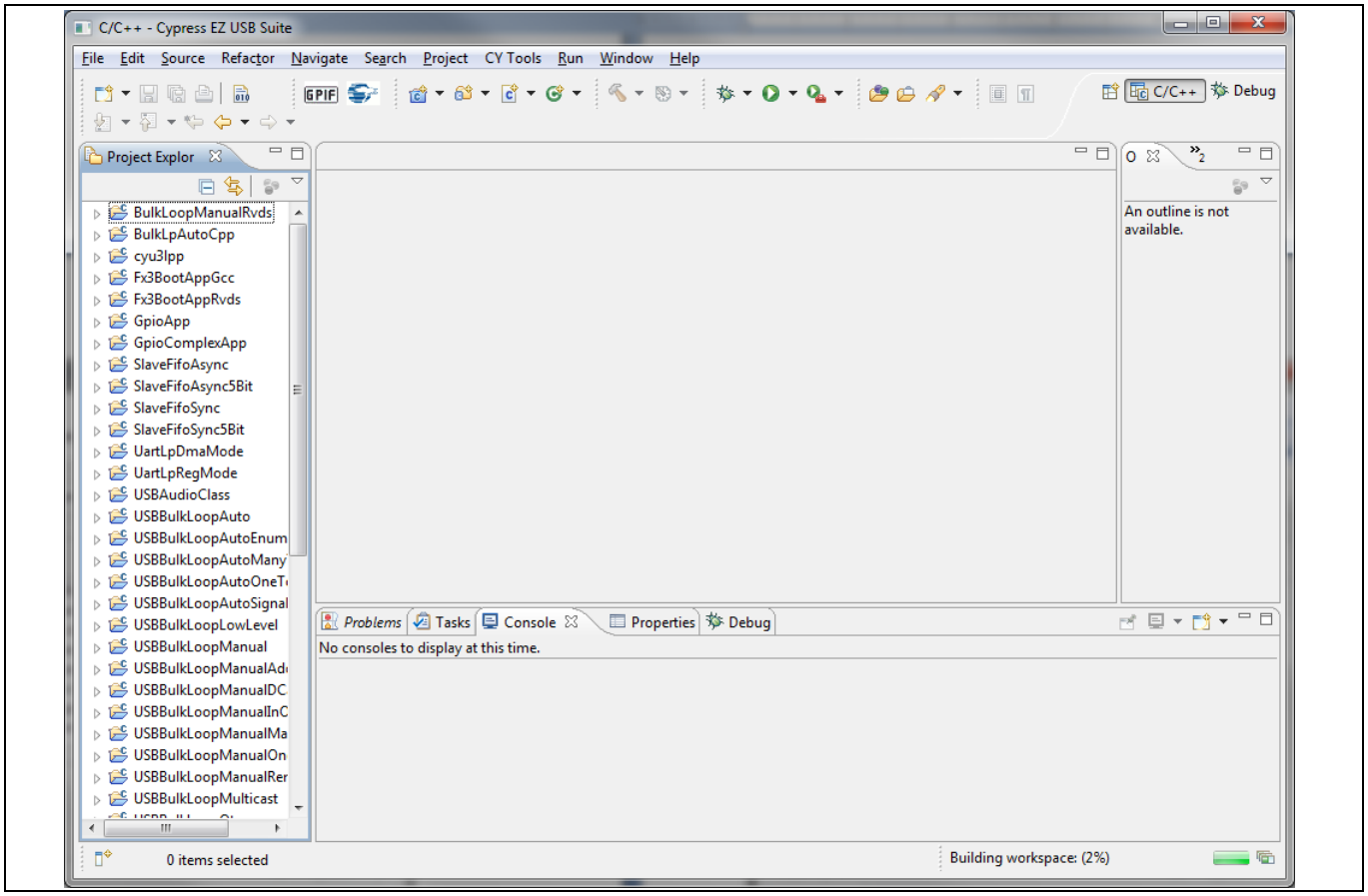


Figure 20　すべてのプロジェクトがワークスペースにインポートされる

8.1.1　USBBulkLoopAuto ファームウェアのビルドと実行

本例の次のステップとして、基本的なサンプル ファームウェアをビルドして実行します。

FX3 SDK に含まれた USBBulkLoopAuto サンプル ファームウェアは、デフォルトでは BULK エンドポイント 1 を IN に BULK エンドポイント 1 を OUT に設定します。USB ホストが OUT エンドポイントに送信したデータは、FX3 ファームウェアによって IN エンドポイントにループバックされます。USB ホストは、IN エンドポイントからそのデータを受信することができます。

1. Eclipse IDE で USBBulkLoopAuto ファームウェアを開くために、その名前をダブルクリックするか、またはその名前の左側にある小さい拡張矢印をクリックします。このファームウェアをデフォルト設定でビルドするか、または以下のステップのとおりにエンドポイント番号を変更してファームウェアを修正することができます。このファームウェアを何も変更せずに実行する場合、**8.1.2** 節に進んでください。ファームウェアに若干の変更を加えてから実行する場合、ステップ 2 に進みます。Eclipse ツール チェーンを詳しく理解するには、変更のステップは推奨されています。
2. ファイル リストから **cyfxbulkloopauto.h** をクリックして、それをエディタ ウィンドウに表示します。エンドポイント番号およびそれらに対応するソケットは、*cyfxbulkloopauto.h* ファイルの終わりの近くにある識別子 CY_FX_EP_PRODUCER、CY_FX_EP_CONSUMER、CY_FX_EP_PRODUCER_SOCKET、および CY_FX_EP_CONSUMER_SOCKET を使用して、*.h* ファイル内で定義されます。

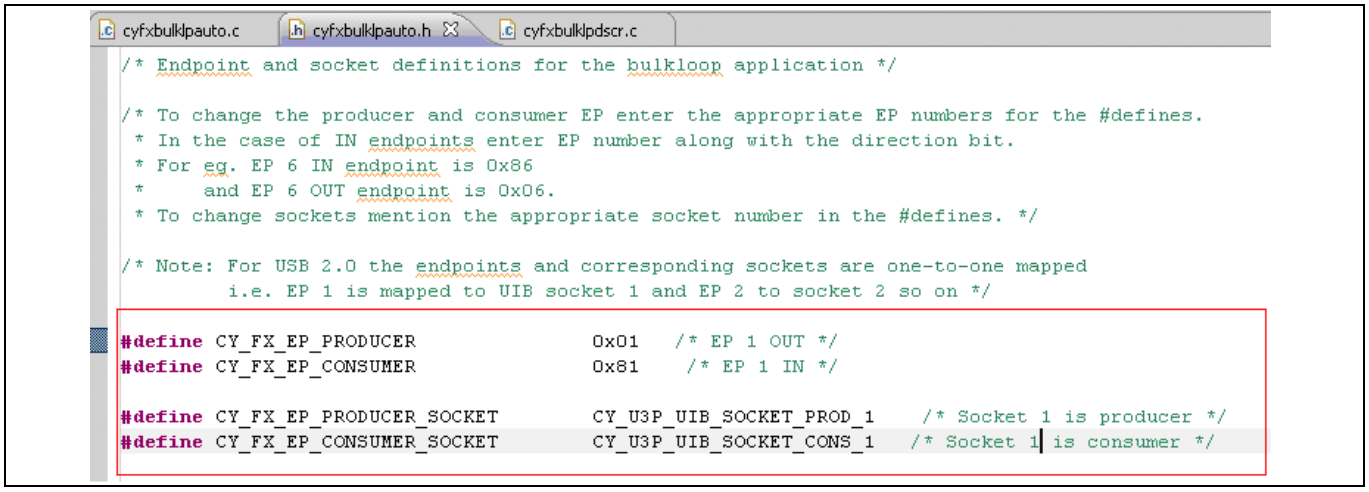


Figure 21　cyfxbulklpauto.h のエンドポイントとソケットの定義

3. **Figure 22** に示すように、上記でハイライトされた識別子を使用して、エンドポイント番号および対応するソケットは EP1 から EP2 に変更されます。この時点で、修正されたファームウェアでは、EP2 IN は BULK IN となり、EP2 OUT は BULK OUT となります。

Note: *USB では、エンドポイントは方向を示すビット 7 (1=入力、0=出力) とビット 6:0 を占める 7 ビット アドレスから成るバイトで識別されます。USB 転送の場合は、FX3「ソケット」はエンドポイントに相当します。USB ドメイン内のソケットは、データ転送のために、他のペリフェラル ドメイン (例えば、GPIF II、または CPU でも) 内のソケットに接続します。*

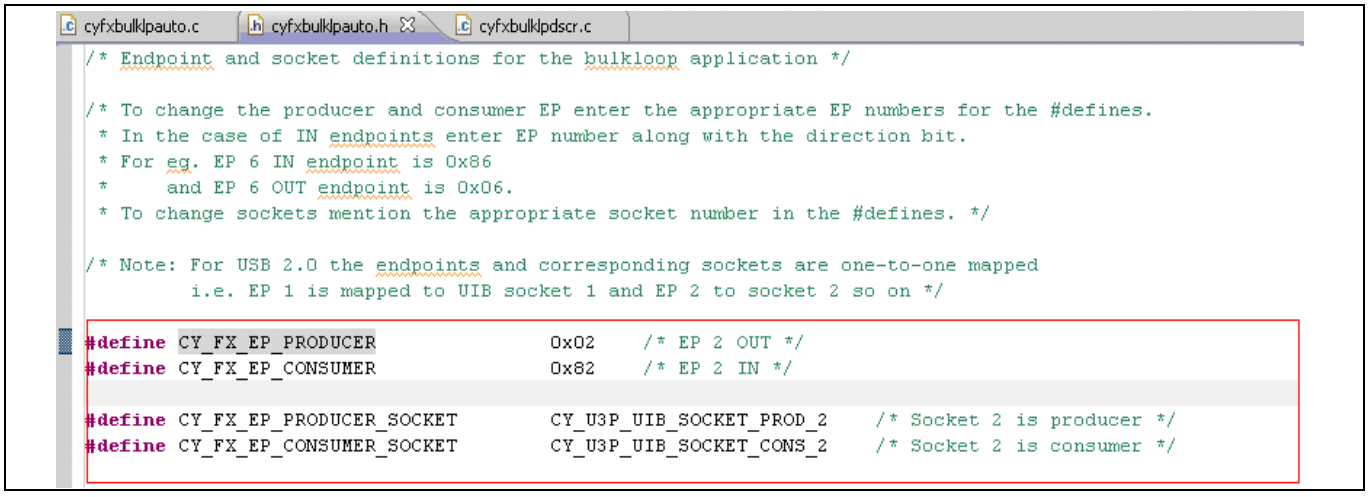


Figure 22　エンドポイント番号および対応するソケットが EP1 から EP2 に修正される

4. 編集された *cyfxbulklpauto.h* ファイルを手動で保存してください (**File** > **Save**)。Eclipse ビルド プロセスは、新たに編集されたファイルを自動的に保存しません。USBBulkLoopAuto プロジェクトを Project Explorer ペインで拡張してから、**Project** > **Build Project** を選択して、プロジェクトをビルドします。
プロジェクトのビルド前に **Build Configurations** を選択することもできます。Project Explorer ペインで、プロジェクト名を右クリックして、**Build Configurations** > **Set Active** > **Debug/Release** を選択します (**Figure 23** を参照してください)。

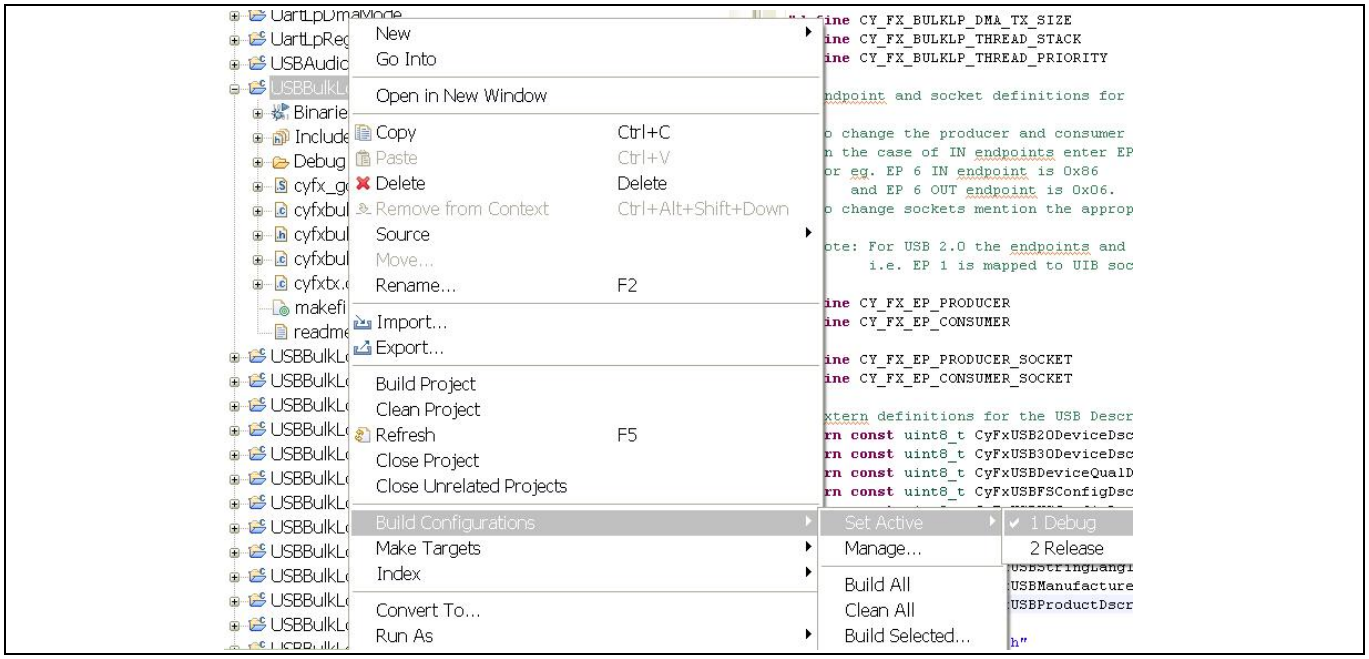


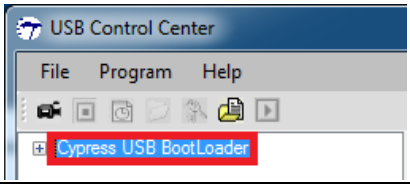
Figure 23 Eclipse IDE で「Build Configurations」を選択する

Note: *リリース モードのファームウェア イメージ サイズは、デバッグ モードのイメージに比べて小さいです。デバッグ モード イメージは、ユーザーが JTAG を使用してデバッグするための追加のデバッグ シンボルを含んでいます。*

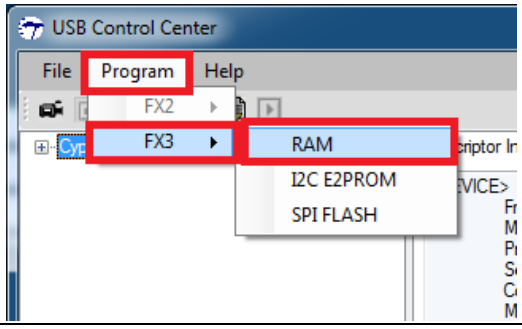
8.1.2 FX3 RAM への USBBulkLoopAuto ファームウェアのロード

サイプレス コントロール センター（Cypress Control Center）というアプリケーションは、コードを FX3 にロードします。コントロール センター アプリケーションは FX3 SDK に含まれます。

1. Start メニューからサイプレス コントロール センター アプリケーションを開きます。**Start** > **All Programs** > **Cypress** > **Cypress USBSuite** > **Control Center** を選択します。
2. キットに同梱されている USB 3.0 ケーブルを使用し、SuperSpeed または Hi-Speed USB ポートのいずれか経由でキット基板を開発 PC に接続します。キットの設定およびドライバー ソフトウェアが既にインストールされている場合、USB Control Center は「Cypress USB Bootloader」を接続済みのデバイスとして示します。しかし、「no drive found」というメッセージが表示された場合、**付録 B** を参照し、Windows ドライバーをインストールしてください。



3. FX3 RAM にファームウェアをロードするためには、**Program** > **FX3** > **RAM** を選択します。



4. FX3 にロードするファームウェア イメージ ファイルに移動します。このファイルは、Eclipse がビルド プロセスの最終ステップで生成した *USBBulkLoopAuto.img* ファイルです。デフォルト ワークスペースの場合、このファイルは C:\Users\antioch\Work\USBBulkLoopAuto\Debug に配置されています。ファイルを選択して **Open** をクリックするか、またはファイル名をダブルクリックします。
USB Control Center が新しい FX3 コードをダウンロードした後、Windows は「Cypress USB BulkLoopExample」という新しい USB デバイスを認識します。これは、FX3 の再列挙です。FX3 は初めは Windows でブートローダとして認識されますが、新しいコードをロードした後、USB から電気的に切断して、ダウンロードしたコード (BULK ループバ) で定義された新しいデバイスとして再接続します。コントロール センター メニューには、FX3「Cypress USB Bootloader」が非表示になり、「Cypress USB BulkloopExample」という新しい USB ペリフェラルが表示されることに注意してください。FX3 は、0x04B4 と 0x00F0 の新しいベンダーID／製品 ID の組合せで再列挙しました。

Note: *新しいプロジェクトをロードする準備ができたら、RESET スイッチを使用して FX3 をリセットし、ブートローダ モードに戻ります。*

8.1.3 BULK 転送の実行

1. 「Cypress USB Bulkloop Example」エントリ下のすべてのサブフィールドを拡張します。8.1.1 で説明したようにファームウェアを修正してビルドした場合、BULK EP2 OUT と BULK EP 2 IN が「Interface 0」の「Alternate Setting 0」の下に表示されます (**Figure 24**)。

Note: *ファームウェアを修正しなかった場合、デフォルトの「BULK out endpoint (0x01) 」と「BULK in endpoint (0x81) 」がコントロール センターに表示されます。*

2. BULK OUT 転送を実行するために、**Data Transfers** タブをクリックし、左側のペインで **BULK out endpoint (0x02)** を選択します。**Bytes to Transfer** フィールドに転送バイト数を入力し、**Data to send (Hex)** フィールドに転送データを入力します。コントロール センターは自動的に 16 進数の 2 桁ごとにスペースを挿入します。**Transfer Data-OUT** をクリックします。**Figure 24** に、BULK OUT エンドポイント (0x02) への 5 バイトの転送を示します。

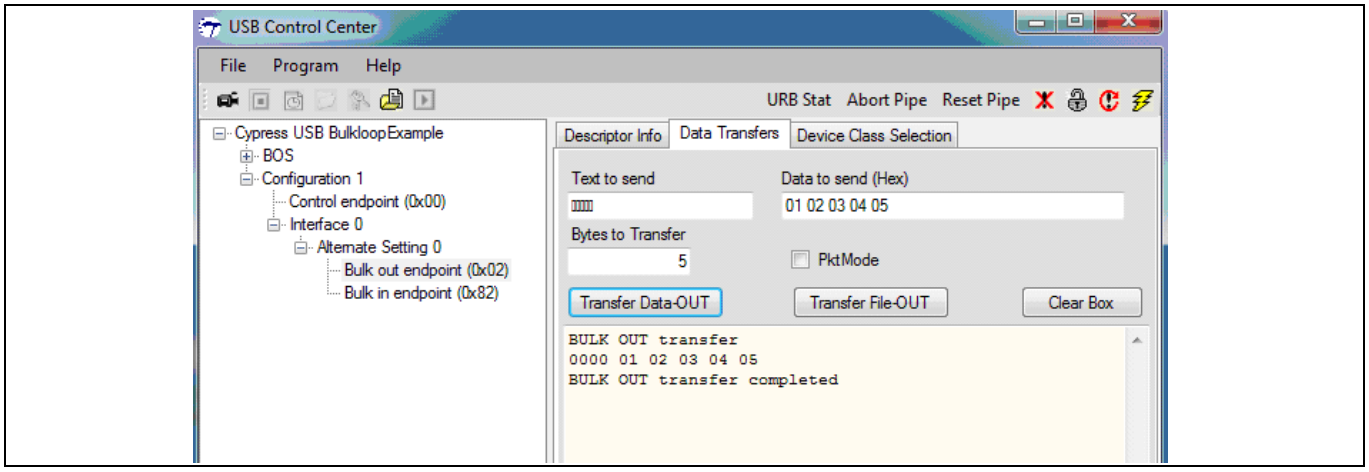


Figure 24　BULK OUT 転送

3. BULK ループ プログラムは、対応して番号付けられた IN エンドポイントを使って OUT エンドポイントで受信したデータを返します。BULK IN 転送を実行するには、左のペインに表示される **BULK in endpoint (0x82)** を選択します。転送するバイト数を **Bytes to Transfer** フィールドに入力します。**Transfer Data-IN** をクリックします。**Figure 25** に、IN エンドポイント (0x82) に同じ 5 バイトが返されたことを示します。

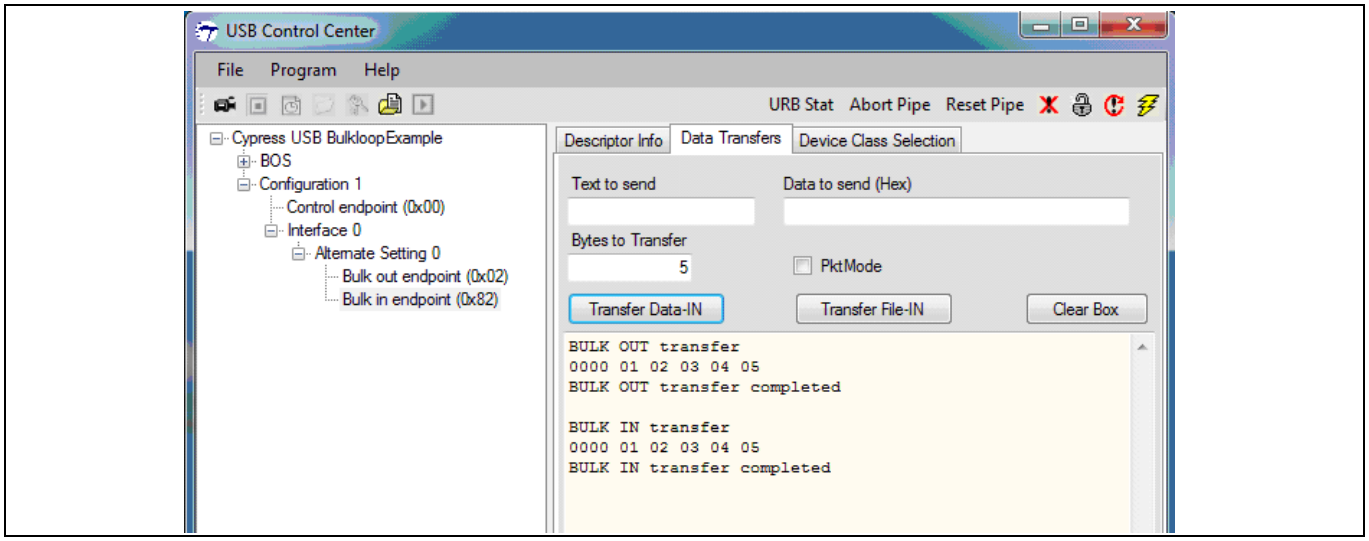


Figure 25　BULK IN 転送

9 開発ツール

これまで、FX3 デバイスの基本操作を行いました。ここでは SuperSpeed Explorer Kit (CYUSB3KIT-003) と FX3 SDK について簡単に紹介します。

9.1 SuperSpeed Explorer Kit の紹介

サイプレスの SuperSpeed Explorer Kit は、開発を開始するために必要なハードウェアを提供します。この PCB は FX3 に必要なクロックおよび電圧をすべて提供し、また I/O 電圧を設定可能にします。キット基板には、外部デバイスと接続するための高速なコネクタを備えています。SuperSpeed Explorer Kit は、USB ブートと I^2C ブート オプションに対応しています。8M ビットの I^2C EEPROM が、ファームウェアを格納し、I^2C EEPROM からのブートをテストするためにキット基板に用意されています。SDK に同梱されるサイプレスのサンプル プロジェクトは、DVK 基板を使ってこれらのデバイスをプログラムするファームウェアを含みます。SuperSpeed Explorer Kit をまだお持ちでない場合、ここで見つけることができます。

FX3 DVK 基板の外部にある、ハードウェアの 2 つの重要な部品は、USB 3.0 ホスト (通常は PC) と GPIF II インターフェースに接続する外部デバイス (画像センサー、FPGA など) です。SuperSpeed Explorer Kit の使用方法の詳細はキット ユーザー ガイドを参照してください。**Figure 26** は、重要な領域が示された SuperSpeed Explorer Kit 基板の画像です。

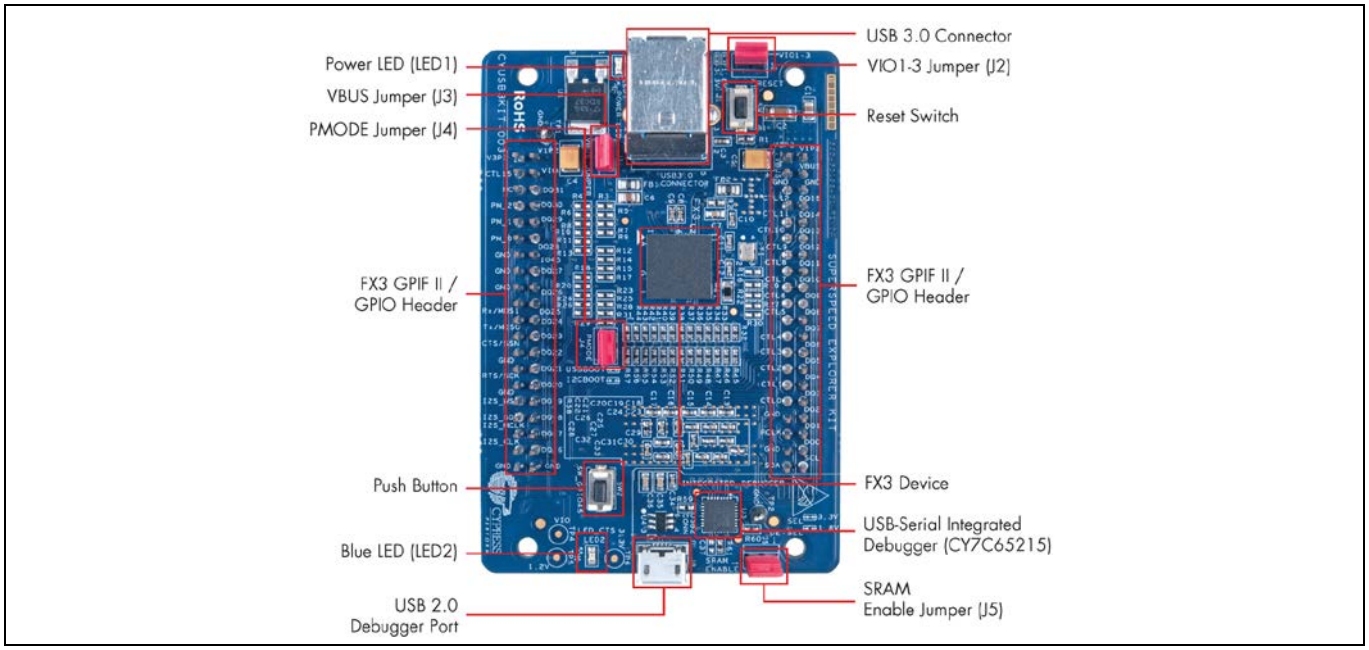


Figure 26 FX3 DVK 基板

開発ツール

Table 2 に、DVK 基板の主なジャンパとそれらのデフォルト設定を示します。

Table 2　SuperSpeed Explorer Kit 基板のジャンパ

ジャンパ	ジャンパの状態	機能
J2	短絡	電源ドメイン VIO1、VIO2、VOI3 の電圧を 3.3V に設定し、外部 3.3V インターフェースと基板搭載 SRAM をサポート
J3	短絡	開発基板は USB 3.0 VBUS から電源供給。通常動作のためにこの設定のままにしておく
J4	開放	有効なファームウェア イメージが EEPROM にある場合、開発基板の I^2C EEPROM からブート。それ以外の場合、USB ブートに戻る。 注: デフォルトでは、LED 点滅ファームウェアの USBBulkSourceSink が EEPROM に格納されている
J5	開放	外部 SRAM を選択解除

9.2　FX3 SDK の紹介

サイプレスは、FX3 が組込みシステム環境に USB アプリケーションを統合するための完全なソフトウェアとファームウェア スタックを提供します。ソフトウェア開発キット (SDK) には、アプリケーション開発を加速させるサンプル アプリケーションが含まれています。SDK はサイプレス ウェブサイトからダウンロードできます。Figure 27 に、SDK がインストールされたときに作成されるディレクトリ構造を示します。

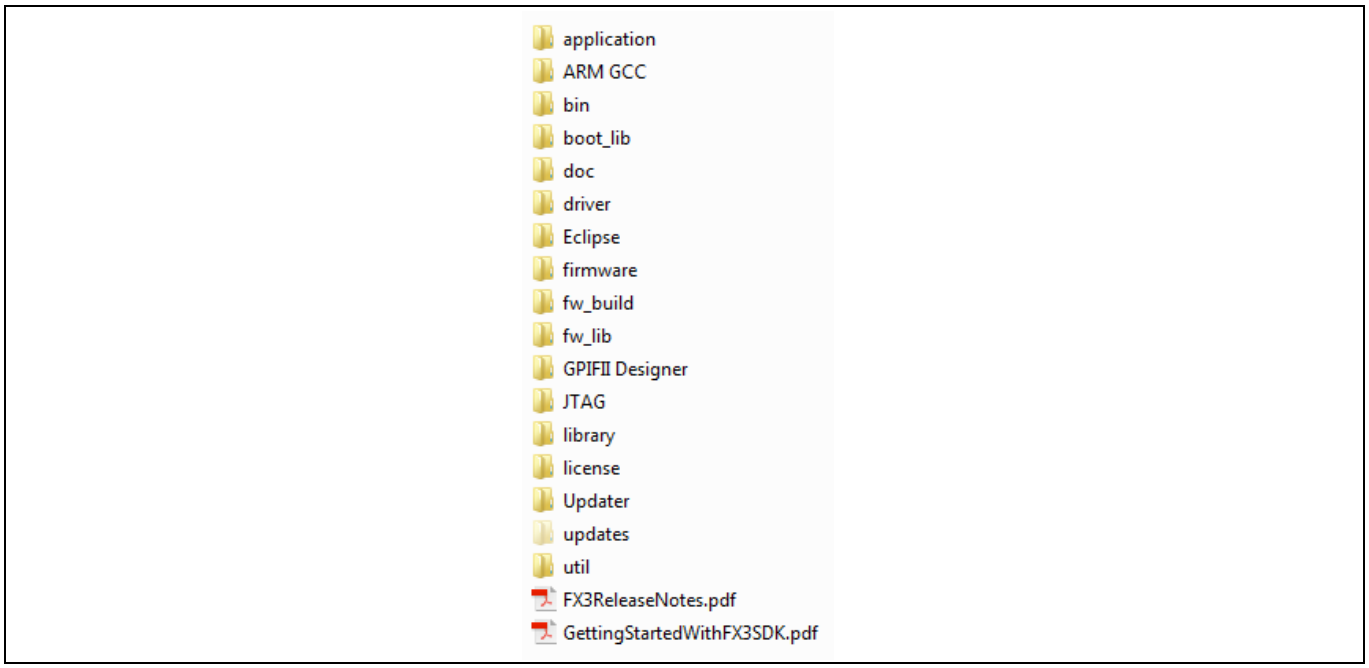


Figure 27　ディレクトリ構造

9.2.1 ファームウェア スタックと API

サイプレスは、複雑な設計においてもファームウェアの開発を容易にするための強力な API ライブラリを提供します。以下は FX3 SDK のいくつかの利点です。

- RTOS
  - 同梱の ThreadX Real Time OS (RTOS) はファームウェア開発手順を簡易化します。RTOS により複数のスレッドが容易に作成でき、ファームウェア フローを簡素化します。
- モジュラ アプローチ
  - API ベース アプローチにより、開発者が FX3 のレジスタとビットの知識を必要とせずにファームウェア ロジックとフローに集中できます。このアプローチは使いやすく、またデバッグ、修正、サポートが容易です。

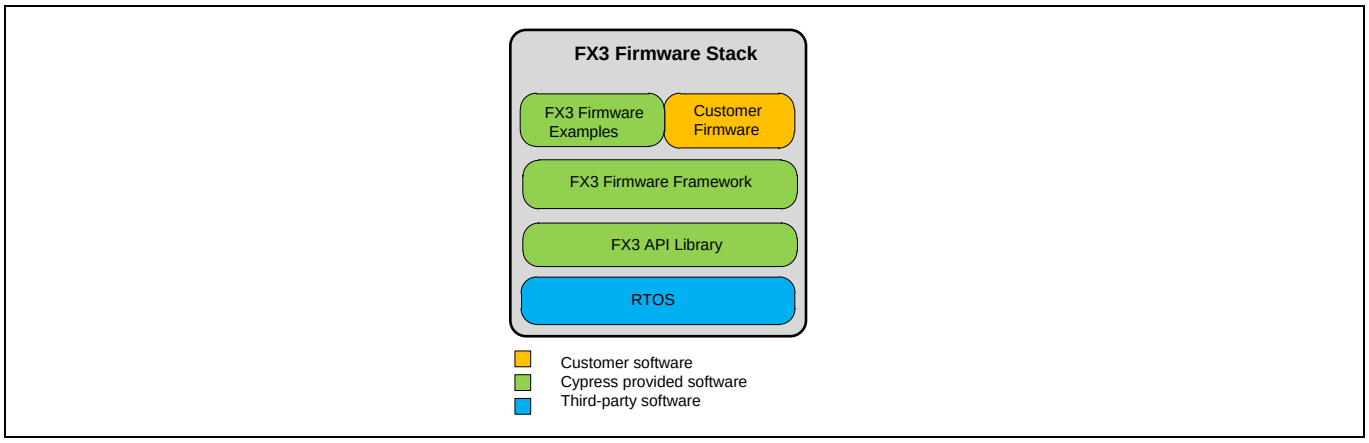


Figure 28 ファームウェア SDK の構造

API では以下のことが可能です。

- FX3 デバイスの各ブロック (GPIF II、USB、シリアル インターフェース) のプログラミング
- DMA エンジンのプログラミングと、これらブロック間のデータ フローの設定
- アプリケーションの要求に応じた ThreadX OS の呼び出し
- デバッグ機能
- USB ホスト動作モード
- 電源管理機能

9.2.2 フレームワーク API

ファームウェア (またはアプリケーション) フレームワークには、起動および初期化コードが含まれています。また、USB、GPIF II およびシリアル インターフェース ブロック用の個別のドライバーも含まれています。フレームワークは以下の機能を実行します。

- プログラムのエントリ ポイントの定義
- スタック セットアップの実行
- カーネル初期化の実行
- アプリケーション スレッド起動コード用のプレースホルダの提供

9.3 FX3 サンプル ファームウェア

SDK には、上記の Eclipse インストールに見られるように多くのサンプル ファームウェアが含まれています。本節では、より一般的なサンプル ファームウェア プロジェクトの内一部を詳細に説明します。現在利用可能なサンプル ファームウェアの完全なリストは、SDK のリリース ノートで参照できます。サンプル ファームウェアは USB 2.0 と USB 3.0 の両方の速度に対応しています。

Note: *サンプル ファームウェアは、個別の Eclipse プロジェクトの形式で提供されています。これにより、すべてのサンプル ファームウェアを Eclipse Project Explorer で一目で見ることができます。*

9.3.1 USB BULK データ ループバックの例

これらの例では、USB BULK エンドポイント間の簡単なループバック メカニズムを示します。USB ホスト PC からのすべての標準セットアップ要求は、FX3 のアプリケーション例により処理されます。例は、DMA AUTO または MANUAL チャネルを使用してループバックを行います。

DMA マルチチャネルの例は、3 つのエンドポイントをループバックに使用します。

9.3.2 USB ISOCHRONOUS データ ループバックの例

これらの例は、USB ISOCHRONOUS エンドポイント間のループバック メカニズムを示します。使用されるエンドポイントが BULK ではなく ISOCHRONOUS であることを除いて、これらの例は BULK ループバックの例と同様です。

9.3.3 スレーブ FIFO アプリケーションの例

スレーブ FIFO アプリケーションの例は、USB ホストと外部 FIFO コントローラー間のデータ転送を示します。この例は、USB ホストと外部マスター間にある 2 本の単方向データ パイプで構成されています。GPIF II インターフェースは、16 ビットまたは 32 ビット バスを使用して同期または非同期スレーブ FIFO 転送用に設定可能です。

9.3.4 シリアル インターフェースの例

これらの例では、GPIO、I^2C、SPI および UART へのデータ アクセスを示します。

9.3.5 USB BULK／ISOCHRONOUS データ ソース シンクの例

これらの例は、2 つの USB BULK／ISOCHRONOUS エンドポイントとのデータ ソースとデータ シンクのメカニズムを示します。

9.3.6 フラッシュ プログラマの例

この例では、USB からの I^2C EEPROM や SPI フラッシュ デバイスのプログラミングを示しています。読み書き動作は、あらかじめ定義されたベンダー コマンドを使用して行われます。ユーティリティは、これらのデバイスにブート イメージをフラッシュするために使用できます。

9.3.7 大容量ストレージ クラスの例

この例は、FX3 デバイス RAM の小さい領域をストレージ デバイスとして使用して USB 大容量ストレージ クラス (バルク オンリー転送) のデバイスの実装を示します。例は、大容量ストレージ コマンドが FX3 ファームウェアでどのように解析され、処理されるかを示します。

9.3.8 USB オーディオ クラスの例

この例では、SPI フラッシュ メモリに保存された PCM オーディオ データを USB ホストにストリーミングする、USB オーディオ クラス準拠のマイク デバイスを作成します。例は USB 2.0 の速度にのみ対応しています。

9.3.9 2 段起動の例

2 段起動を実行するための簡単な API 一式が独立したライブラリとして提供されます。この例は、これら API の使用方法を示します。また、Real View ツール チェーンで使用できるコンフィギュレーション ファイルも提供されます。

9.3.10 USB ホストおよび OTG の例

これらの例は、FX3 USB ポートのホスト動作モードと OTG 動作モードをデモします。

9.4 FX3 ファームウェア開発ツール

9.4.1 Eclipse IDE

FX3 SDK の一部として、C／C++開発者向け Eclipse IDE が提供されます。この IDE には、ベースとなる Eclipse プラットフォームおよび CPP 機能が含まれます。開発に必要なプラグインが IDE に含まれます。

- GNU Arm C／C++開発サポート
- Zylin Embedded CDT
- これは FX3 ファームウェアのデバッグを GNU デバッガで可能にする Eclipse IDE 用の汎用プラグインです。
- Java(TM) Platform、Standard Edition Runtime Environment Version 7 (JRE)

9.4.2 GNU ツール チェーン

FX3 SDK の一部として提供されている GNU ツール チェーンには、以下が含まれます。

- GCC コンパイラ (gcc)
- GNU リンカー (ld)
- GNU アセンブラ (as)
- GNU デバッガ (gdb)

これらの実行ファイルは Eclipse IDE によって呼び出されます。

9.4.3 GPIF II Designer

GPIF II Interface Design Tool は、FX3 のユーザーが FX3 SDK の一部として入手する Windows アプリケーションです。ツールは、対象のデバイスに必要なインターフェースを指定するグラフィカル ユーザー インターフェースを提供します。ユーザーは、ステート マシンのエントリを使用してインターフェースを設計し、そしてツールは Eclipse プロジェクトに格納される C ヘッダ ファイルにインターフェースを変換します。

9.4.4 内蔵デバッガ

SuperSpeed Explorer Kit は **CY7C65215 USB-Serial IC** を内蔵デバッガとして使用します。CY7C65215 は 2 つのコンフィギュレーション可能なシリアル通信ブロック (SCB) を備えた Full Speed USB シリアル ブリ

ッジ コントローラーです。SuperSpeed Explorer Kit は、CY7C65215 の 1 つ目のシリアル チャネルを UART として使用し、2 つ目のシリアル チャネルを JTAG インターフェースとして使用します。

SuperSpeed Explorer Kit は内蔵デバッガにより提供される JTAG インターフェースのみをサポートします。内蔵デバッガは OpenJTAG プロトコルで動作し、ホスト上の OpenOCD デーモン ツールを使用してデバッグ セッションを実行します。

内蔵デバッガの詳細は **SuperSpeed Explorer Kit User Guide** を参照してください。

9.5 Windows ソフトウェアの概要

サイプレスは、Windows 上で FX3 USB アプリケーションを開発するためにデバイス ドライバーとインターフェース API ライブラリを提供します。

9.5.1 Windows USB デバイス ドライバー

cyusb3.sys は 32 ビット Windows XP、32／64 ビット Windows Vista、32／64 ビット Windows 7 および 32／64 ビット Windows 8 用の USB デバイス ドライバーです。これは USB 2.0 および USB 3.0 仕様に準拠するデバイスと通信することができます。ドライバーは汎用であり、基本的な USB コマンドを理解します。しかし、このデバイスは USB デバイス クラスのプロトコルを実装しません。例えば、ドライバーは Windows ファイル システムに USB 大容量ストレージ デバイスを直接接続しません。アプリケーション ファームウェアはこのロジックを実装する必要があります。

この汎用的なサイプレス ドライバーは、0x04B4 の VID、および 0x00F0、0x00F1、0x00F3 の PID を対象に署名されています。独自の VID／PID ペアを使用される場合、Microsoft WHQL によって署名されたドライバーを入手する必要があります。開発フェーズでは、コンピューター上のドライバー署名の強制を無効にすることで、署名が入っていないドライバーを使用できます。ドライバーは、カスタム USB アプリケーションからベンダー固有のデバイスへ通信するための理想的手段です。またはドライバーは、実験または診断アプリケーション用の任意の USB デバイスへ、低レベルの USB リクエストを送信するために使用することもできます。デバイスと通信するためにドライバーを使用するには、Windows はデバイスをドライバーにマッチさせる必要があります。クラス ライブラリである *CyAPI.lib* および *Cyusb.dll* は、ドライバーに高レベルのプログラミング インターフェースを提供します。

9.5.2 機能

- Windows Driver Foundation (WDF) 準拠
- 任意の USB 2.0 準拠デバイス互換
- サイプレス USB 3.0 準拠デバイス互換
- 基本的な USB 3.0 機能のサポート
- Windows プラグ アンド プレイおよび電源管理をサポート
- USB 遠隔ウェイクアップをサポート
- CONTROL、BULK、INTERRUPT、ISOCHRONOUS エンドポイントをサポート
- 同時に接続された複数の USB デバイスをサポート
- ドライバーを再構築せずにカスタマイズ可能なドライバーGUID をサポート
- フレームごとに複数のパケットの高帯域幅のデータ転送をサポート

サイプレス製のドライバーを使用する必要はありません。FX3 は、標準 USB クラス デバイスを実装するようにプログラムすることが可能です。それらのデバイスを実装する際には、USB クラス ドライバーを使用してください。例えば FX3 が USB ビデオ クラス (UVC) を実装している場合は、サイプレス汎用ドライバーを必要とせず、OS が提供する UVC ドライバーを使用します。特定の標準ドライバーが既に OS によって提供されているため、Windows、Linux および Mac サポートが容易になります。

9.6 アプリケーション インターフェース

9.6.1 CyAPI.lib

CyAPI.lib は簡単かつ強力な C++プログラミング インターフェースを USB デバイスに提供します。C++クラス ライブラリは *cyusb3.sys* デバイス ドライバーに高レベルのプログラミング インターフェースを提供します。本ライブラリは、このドライバーが対応する USB デバイスとだけ通信できます。

CyAPI.lib の詳細はプログラマのリファレンス C++ライブラリを参照してください。これは SDK の一部である USB スイート ヘルプ文書に記載されています。

9.6.2 CyUSB.dll

CyUSB.dll はマネージド Microsoft .NET クラス ライブラリであり、USB デバイスへの高レベル プログラミング インターフェースを提供します。アプリケーションは、低レベル Win32 API 呼び出しを介して USB デバイス ドライバーと直接通信するのではなく、ライブラリ メソッドとプロパティを介して USB デバイスにアクセスできます。*CyUSB.dll* がマネージド.NET ライブラリであるため、そのクラスやメソッドは、Visual Basic.NET、C#、Visual J#、マネージド C++ などのいかなる Microsoft Visual Stuido.NET マネージド言語からもアクセスすることができます。ライブラリを使用するには、*CyUSB.dll* への参照を、対象プロジェクトの **References** フォルダに追加します。そして、CyUSB 名前空間にアクセスするソース ファイルには、適切な構文で名前空間を追加するための行を含める必要があります。

CyUSB.dll の詳細はプログラマのリファレンス C#ライブラリを参照してください。これは SDK の一部である USB スイート ヘルプ文書に記載されています。

CyAPI.lib と *CyUSB.dll* の使用方法詳細は **Cypress USBSuite Application Development - Quick Start Guide** を参照してください。

9.7 Windows ソフトウェアの例

9.7.1 BULKLoop の例

アプリケーション BulkLoop は、BULK エンドポイント経由のデータ転送のループバックをテストするために使用されます。

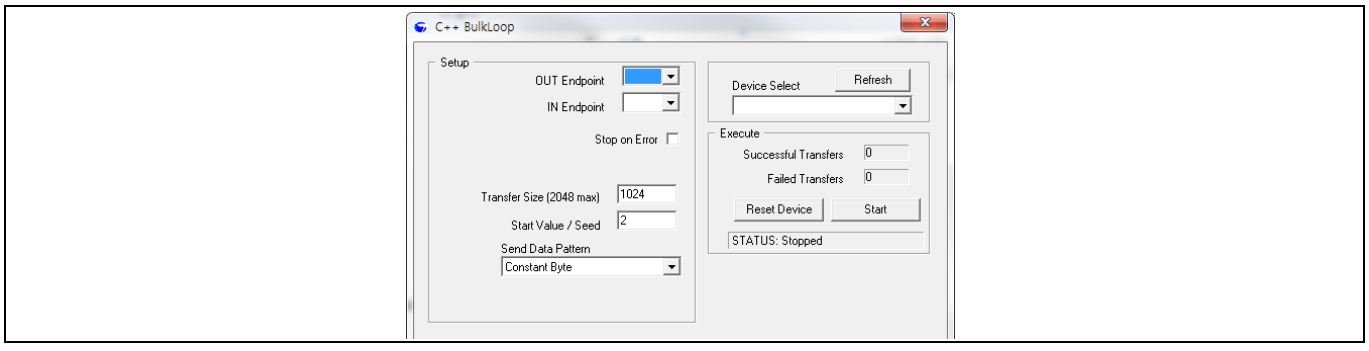


Figure 29 サイプレス C++ BULKLoop アプリケーション

9.8 Streamer の例

アプリケーション Streamer は、BULK／ISOCHRONOUS エンドポイントのデータ転送をテストするために使用されます。

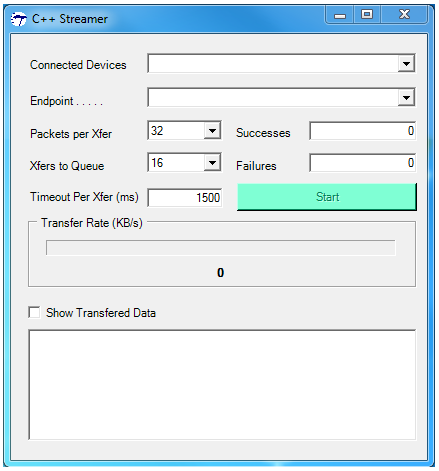


Figure 30　サイプレス C++ Streamer アプリケーション

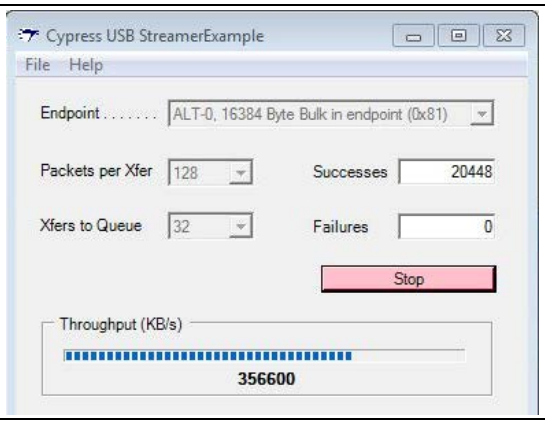


Figure 31　サイプレス C# Streamer アプリケーション (スレーブ FIFO アプリケーションの通信量)

FX3 SDK をインストールすると、BULKLoop と Streamer アプリケーションの C++と C＃の実装は両方とも、「applications」ディレクトリで使用可能になります。

9.9 Linux 用 FX3 SDK およびソフトウェア

9.9.1 Linux 用 EZ-USB™ FX3 SDK

FX3 SDK は、Eclipse IDE を使用したファームウェア開発と、Linux プラットフォーム上で J-Link JTAG デバッガ プローブを使用したデバッグをサポートします。

Linux 用の EZ-USB™ FX3 SDK は以下を含みます。

- FX3 ファームウェア ライブラリとサンプル コード
- Sourcery Arm GNU ツールチェーン
- 32 ビット Linux OS インストール用 Eclipse IDE
- 64 ビット Linux OS インストール用 Eclipse IDE
- CyUSB Suite for Linux ソフトウェア

インストール手順では、これらアーカイブを抽出し、環境変数を設定します。インストール ステップの詳細は **FX3 SDK for Linux** に同梱されている *FX3_SDK_Linux_Support.pdf* を参照してください。

9.9.2 CyUSB Suite for Linux

CyUSB Suite for Linux ソフトウェアにより、FX3 デバイスにファームウェア イメージをダウンロードし、デバイス上の様々なインターフェースをテストすることができます。**FX3 SDK for Linux** インストール フォルダ: fx3_sdk_<version>_linux\cyusb_linux_1.0.4\docs にある *cyusb_linux_user_guide.pdf* を参照してください。この文書では、ソフトウェアのインストール、FX3 へのファームウェア ダウンロード、ベンダー拡張機能のテスト、BULK OUT／IN 転送、ISOCHRONOUS OUT／IN 転送の方法について説明します。

「CyUSB Suite for Linux - Programmers Reference Manual」 (同じフォルダにある *cyusb_linux_programmers_guide.pdf*) は、Linux 用の cyusb ライブラリと、ユーザーがプログラムしたアプリケーションをライブラリでビルドおよび統合する方法を説明します。このソフトウェアでは、以下のことができます。

- 接続されたデバイスのデバイス、コンフィギュレーション、インターフェース、代替インターフェースおよびエンドポイント記述子を表示する
- 通信用の特定インターフェースと代替インターフェースを選択する
- デバイスを FX3 デバイスにプログラムし (ファームウェアをダウンロードし)、またファームウェアを RAM、I^2C ベースの EEPROM または SPI ベースのフラッシュにもダウンロードする
- ユーザー独自のコマンド (ベンダ拡張機能) を実行する特定のファームウェアをダウンロードした後、コマンドをテストする
- BULK OUT エンドポイントを、定数またはランダムまたはインクリメント データ パターンの送信によりテストする。また BULK IN エンドポイントを、OUT データの送信後にデータをループバックすることによりテストする
- ISOCHRONOUS OUT／IN エンドポイントをテストし、データ転送レートを測定する

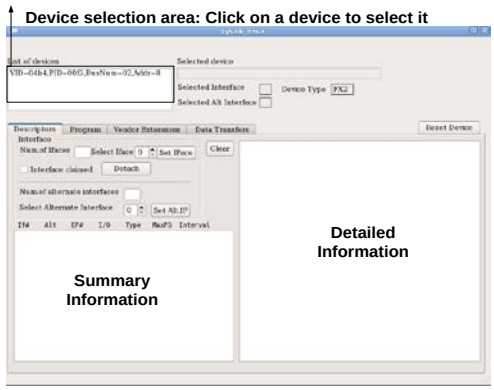


Figure 32 CyUSB Suite for Linux

9.10 有用なデバッグ ツール

9.10.1 USB 3.0 プロトコル アナライザ

USB 3.0 プロトコル アナライザは有用なデバッグ ツールです。FX3 とホスト間の USB 上のトラフィックを分析します。その後、各アナライザに付属されたソフトウェア ツールがデータを USB 転送パケットにデコードします。このデータを分析することにより、問題を容易に特定し、性能を最大にすることができます。今日の市場には入手可能な USB 3.0 アナライザが複数あります。サイプレスは特定のアナライザを推奨しませんが、次のようなオプションがあります。

- スタンドアロン USB 3.0 プロトコル アナライザ
  - **Ellisys USB Explorer 280**
  - **LeCory USB Voyager M3i**
  - **Beagle USB 5000 SuperSpeed プロトコル アナライザ**
- PC ソフトウェア USB 3.0 プロトコル アナライザ
  - **SourceQuest SourceUSB**
  - **SysNucleus USBTrace**

9.10.2 ロジック アナライザ

ロジック アナライザは、デジタル信号の簡単な分析を可能にします。これらは FX3 と他のペリフェラル間の様々な信号を観察するために使用することができます。以下の 2 種類のロジック アナライザが市場で販売されています。

- スタンドアロン型ロジック アナライザ
  - **Agilent 16800 シリーズ ポータブル ロジック アナライザ**
- PC ベース型ロジック アナライザ
  - **USBee ロジック アナライザ**
  - **ZeroPlus ロジック アナライザ**

考慮すべき要素として、アナライザの信号周波数範囲が分析される信号より高くなければなりません。

10 付録 A. USB 3.0 の概要

USB 3.0 は 5Gbps のデータ転送速度と消費電力の削減を可能にし、USB 2.0 と下位互換です。USB-IF により発行された USB 仕様はここで参照できます。**Figure 33** に USB 3.0 アーキテクチャを示します。

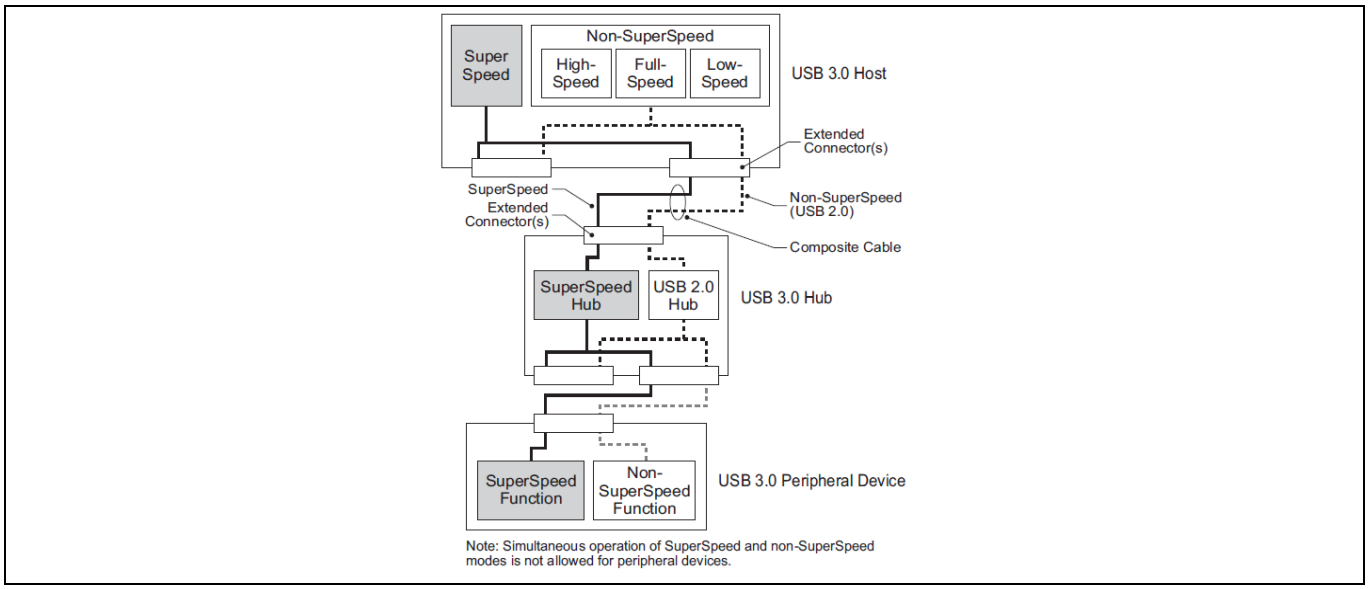


Figure 33 USB 3.0 デュアル バス アーキテクチャ

出典: **http://usb.org**

USB 3.0 の物理インターフェースは USB 2.0 接続に加え、SuperSpeed 転送用の 2 つの差動ペアとグラウンドで構成されています。これにより、USB 2.0 への下位互換を実現しています。

10.1 電気的インターフェース

USB 3.0 のピン配置は USB 2.0 とは異なります。USB 2.0 に必要な VBUS、D-、D+、GND ピンに加え、USB 3.0 にはさらに 5 つのピン、すなわち 2 つの差動ペアおよび 1 つのグランド (GND_DRAIN) を備えています。2 つの差動ペアは、SuperSpeed データ転送用で、デュアル シンプレックスの SuperSpeed 信号方式をサポートしています。GND_DRAIN ピンは、ドレイン線終端、信号品質の管理および EMI 性能に用いられます。**Table 3** は、9 個のピンの説明です。

Table 3 USB 3.0 ピンの説明

ピン名	説明
VBUS	電源
D-	USB 2.0 差動ペア
D+	
GND	電源リターン用グランド
SSRX-	SuperSpeed 受信差動ペア
SSRX+	
SSTX-	SuperSpeed 送信差動ペア
SSTX-	
GND_DRAIN	信号リターン用グランド

10.2 ケーブルおよびコネクタ

USB 3.0 はデータ転送用に 4 本の追加データ ライン (SSRX+、SSRX-、SSTX+、SSTX-) およびドレイン線終端、信号品質管理、EMI 性能用に 1 本の追加グランド ラインを備えています。**Figure 34** に USB 3.0 ケーブルのアーキテクチャを示しています。**Table 4** でこれらラインを説明しています。

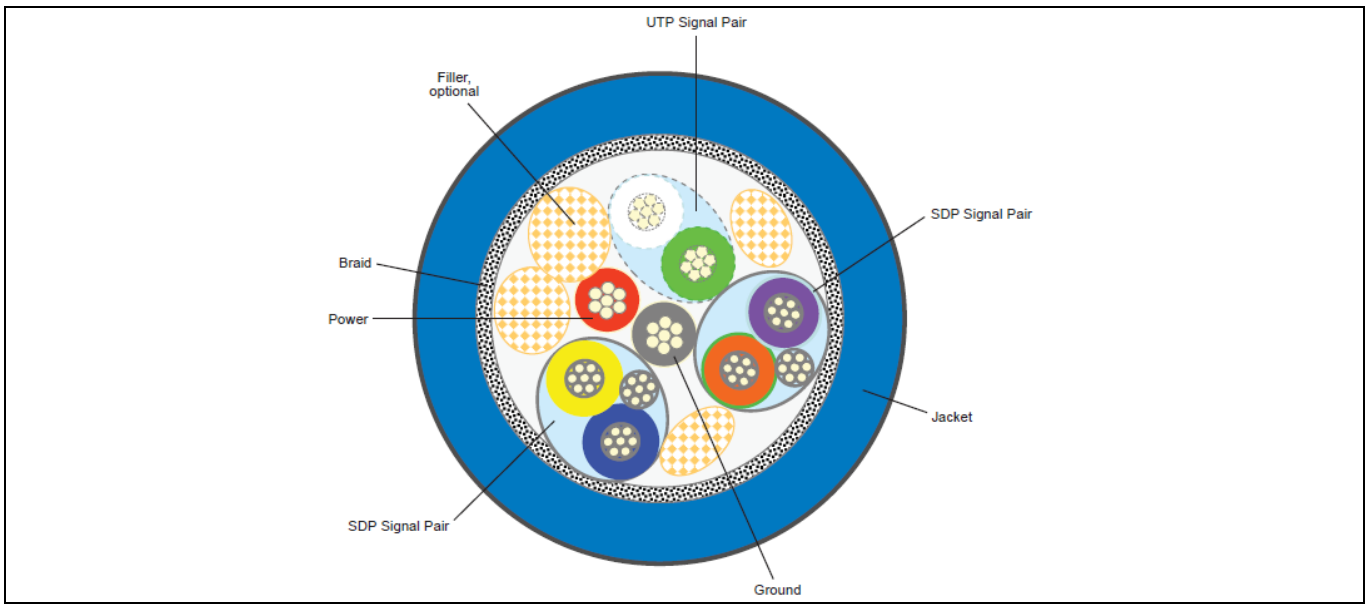


Figure 34 USB 3.0 ケーブル アーキテクチャ

出典: **http://usb.org**

Table 4 USB 3.0 ケーブルの説明

名称	説明	色
PWR	VBUS	赤色
UTP_D-	USB 2.0 D-	白色
UTP_D+	USB 2.0 D+	緑色
グランド	電源ドレイン用グランド	黒色
SDP1-	シールド差動ペア 1	青色
SDP1+		黄色
SDP1_Drain	SDP1 用ドレイン線	
SDP2-	シールド差動ペア 2	紫色
SDP2+		オレンジ色
SDP2_Drain	SDP2 用ドレイン線	

USB 3.0 仕様は次のコネクタも定義します。

- **USB 3.0 標準-A プラグおよびレセプタクル**

USB 3.0 の標準-A コネクタは、SuperSpeed 規格でホスト コネクタとして定義されています。これは USB 2.0 の標準-A コネクタのデザインをベースとしていますが、追加の SuperSpeed 信号を備えています。USB 3.0 標準-A レセプタクルは、USB 3.0 標準-A プラグまたは USB 2.0 標準-A プラグのどちらも差し込み可能です。USB 3.0 対応の標準-A コネクタには、識別を容易にするために、個別の色を使用します。**Figure 34** では、カラー コーディングの推奨を示しています。

- **USB 3.0 標準-B プラグおよびレセプタクル**

USB 3.0 標準-B コネクタは、外部ハードディスク ドライブやプリンターなどの大きな固定式装置のために定義されます。USB 3.0 標準-B レセプタクルは、USB 3.0 標準-B プラグまたは USB 2.0 標準-B プラグのどちらも差し込み可能です。USB 2.0 標準-B レセプタクルに USB 3.0 標準-B プラグを差し込むことはできません。

- **USB 3.0 電源-B プラグおよびレセプタクル**

USB 3.0 電源-B コネクタは、USB 3.0 デバイスが外部電源なしで USB アダプタに電源を供給するためのものです。フォーム ファクタでは USB 3.0 標準-B コネクタと同じですが、電源用 (DPWR) およびグランド用 (DGND) に 1 本ずつ、計 2 本のピンをさらに備えています。

- **USB 3.0 Micro-B プラグおよびレセプタクル**

USB 3.0 Micro-B コネクタは、小型ハンドヘルド デバイスのために定義されています。

- **USB 3.0 Micro-AB および USB 3.0 Micro-A コネクタ**

USB 3.0 Micro-AB レセプタクルは、キーイングが異なる以外は、USB 3.0 Micro-B レセプタクルに類似しています。USB 3.0 Micro-A プラグ、USB 3.0 Micro-B プラグ、USB 2.0 Micro-A プラグ、USB 2.0 Micro-B プラグの差し込みが可能です。USB 3.0 Micro-AB レセプタクルは、ホストまたはデバイスのいずれかとして機能できる OTG 製品のみで使用できます。USB 3.0 Micro-AB レセプタクルの他のあらゆる使用は禁止されています。

USB 3.0 Micro-A プラグは、キーイングおよび ID ピン接続が異なる以外は USB 3.0 Micro-B プラグに類似しています。USB 3.0 Micro-A プラグ、USB 3.0 Micro-AB レセプタクル、USB 3.0 Micro-B レセプタクルおよびプラグは、USB 3.0 Micro コネクタ ファミリに属します。これらインターフェースはキーイングのみ異なります。USB 2.0 Micro-A プラグと同様、USB 3.0 Micro-A プラグは OTG アプリケーションのみを対象にして定義されています。

10.3 USB 3.0 と USB 2.0 の比較

USB 3.0 は、USB 2.0 および 3.0 をサポートするデュアル バス アーキテクチャを備えています。次の表では、USB 3.0 と USB 2.0 との主な相違点を示します。

Table 5 USB 3.0 と USB 2.0 間の比較表

機能	USB 2.0	USB 3.0
データ転送レート	480M ビット/秒 (High Speed) 12M ビット/秒 (Full Speed) 1.5M ビット/秒 (Low Speed)	5.0G ビット/秒 (SuperSpeed) 480M ビット/秒 (High Speed) 12M ビット/秒 (Full Speed) 1.5M ビット/秒 (Low Speed)
データ インターフェース	半二重 二線差動信号方式	デュアル シンプレックス 四線差動信号方式
ケーブル信号カウント	4 つの信号: - USB 2.0 データ用の 2 つ (D、D-) - VBUS と GND 用の 2 つ	9 つの信号: - SuperSpeed データ用の 4 つ - USB 2.0 データ用の 2 つ (D、D-) - VBUS と GND 用の 3 つ

付録 A. USB 3.0 の概要

機能	USB 2.0	USB 3.0
バス トランザクション プロトコル	ホスト主導型 ポーリング トラフィック フロー パケットがすべてのダウンストリーム デバイスへ送信される データ ストリームの多重化なし	ホスト主導型 非同期通知 パケットが対象のデバイスへのみ送信される BULK 転送で複数のデータ ストリームが可能
電源管理	2 モード - アクティブ - サスペンド	4 モード - アクティブ (U0) - アイドル、高速 (U1) - アイドル、低速 (U2) - サスペンド、低速 (U3)
バス電源	低電力デバイス: 100mA 高電力デバイス: 500mA	低電力デバイス: 150mA 高電力デバイス: 900mA
ポート状態	ポート ハードウェアが接続を検出。 ポートを有効な状態に移行させるために、システム ソフトウェアがポート コマンドを使用	ポート ハードウェアが接続を検出し、ポートを SuperSpeed データ通信可能な動作状態に移行させる
最大ケーブルの長さ	5 メートル	電気的仕様に基づく 実際には、26 AWG 銅の 3 メートル
データ転送タイプ	4 データ転送タイプ: CONTROL、BULK、INTERRUPT、ISOCHRONOUS	SuperSpeed が可能な USB 2.0 タイプ。 BULK はストリーム可能

11 付録 B. Windows 上の FX3 DVK ドライバー インストール

Windows コンピューターに FX3 DVK キットをインストールしたことがなく、コンピューターに DVK を初めて接続するときに次のメッセージが表示されます。

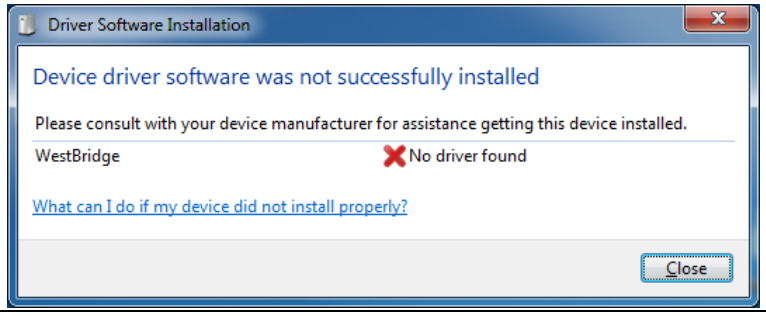


Figure 35 ドライバーがインストールされていない場合の Windows メッセージ

メッセージ ボックスを閉じ、Windows の Device Manager に移動します。これを行うには、Windows の **Start** ボタンをクリックし、右側の列の **Computer** を右クリックし、システム情報を表示するために **Properties** を選択します。その後、左側の列の上部にある **Device Manager** をクリックします。

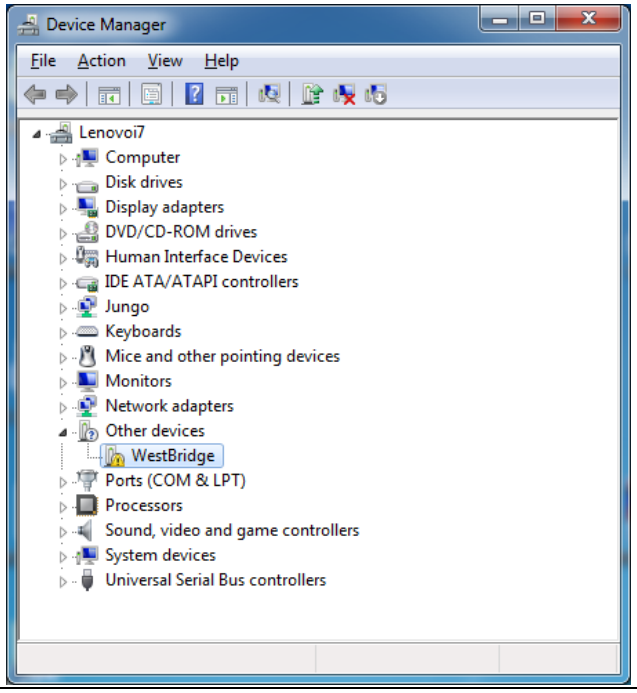


Figure 36 Windows の Device Manager

WestBridge を右クリックして **Browse my computer for driver software** を選択します。64 ビットの Windows 7 マシンでは、ドライバーが C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\driver\bin\win7\x64 にあります。ユーザーの SDK のバージョン番号が 1.2 以降でも可能です。

Device Manager ウィンドウは WestBridge エントリを削除し、SDK 基板を Cypress USB Bootloader (トップ エントリ) として識別します。

付録 B. Windows 上の FX3 DVK ドライバー インストール

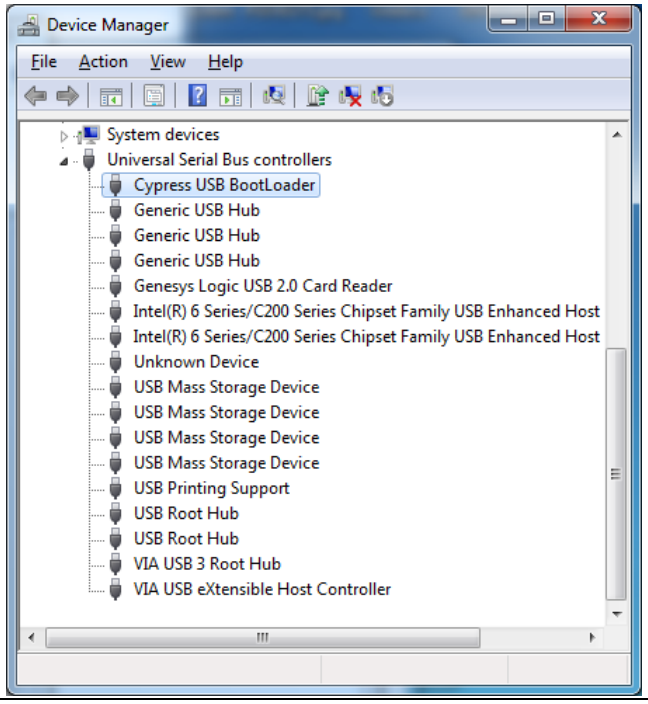


Figure 37　ドライバー インストール後の Device Manager のサイプレス USB ブートローダ

12 付録 C. FX3 DVK の紹介

サイプレスの FX3 DVK は、開発を開始するために必要なハードウェアを提供します。この PCB は FX3 に必要なクロックおよび電圧をすべて提供し、また I/O 電圧を設定可能にします。DVK は、外部デバイスと接続するための高速なコネクタを備えています。さらに、DVK は様々なブート モードを実装します。I^2C EEPROM ソケットが、I^2C EEPROM からのブートのプログラミングとテストに使用できます。基板搭載 SPI フラッシュ チップが、SPI バス経由のプログラミングとブートを可能にします。SDK に同梱されるサイプレスのサンプル プロジェクトは、DVK 基板を使ってこれらのデバイスをプログラムするファームウェアを含みます。DVK をまだお持ちでない場合、**こちら**から入手できます。

FX3 DVK 基板の外部にある、2 つの重要なハードウェア部品は、USB 3.0 ホスト (通常は PC) と GPIF II インターフェースに接続する外部デバイス (画像センサー、FPGA など) です。FX3 DVK 使用方法の詳細は DVK の**ユーザー ガイド**を参照してください。**Figure 38** は、重要な領域が示された FX3 DVK 基板の画像です。

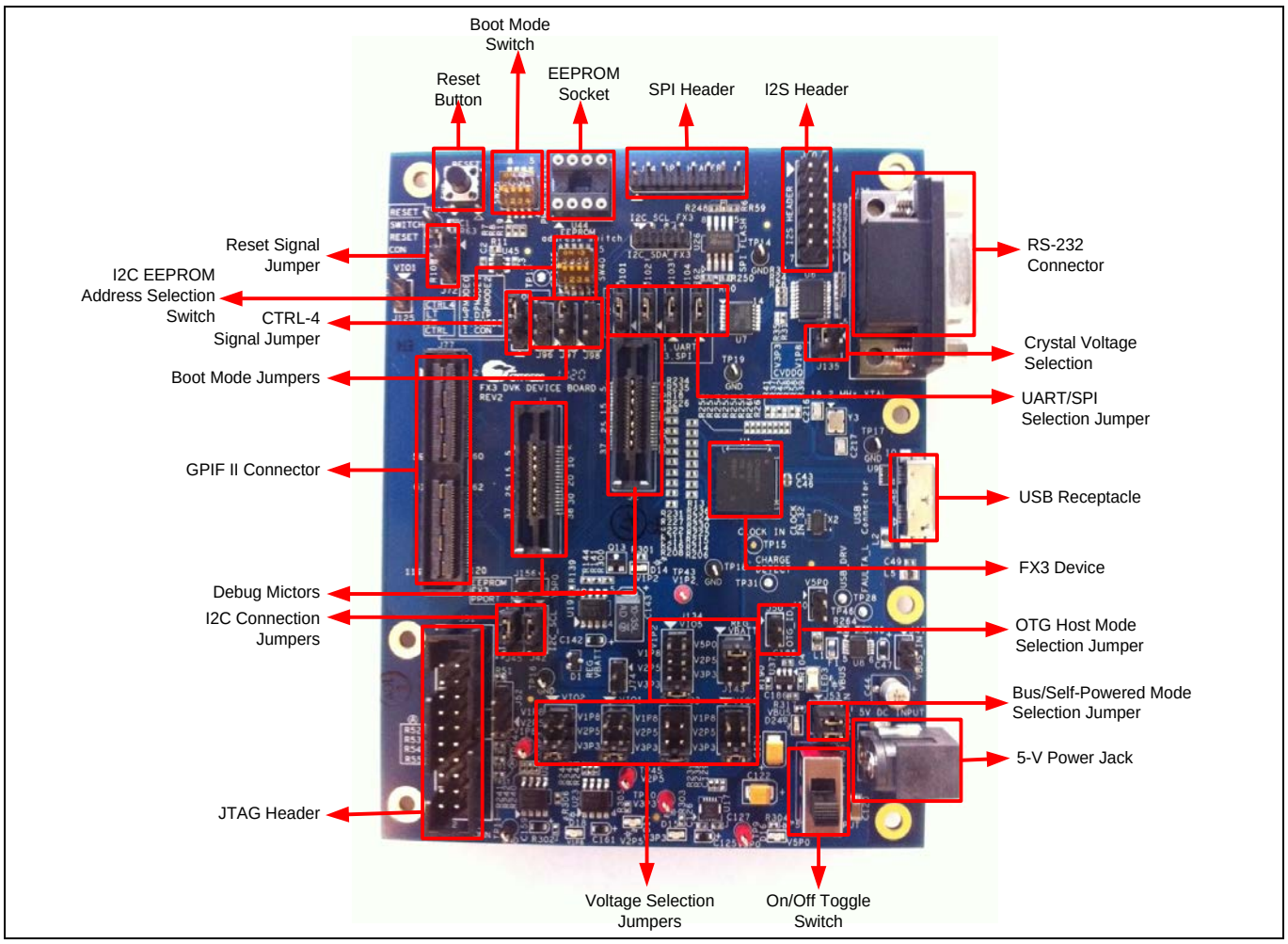


Figure 38 FX3 DVK 基板

付録 C. FX3 DVK の紹介

Table 6 に、DVK 基板の主なジャンパとそれらのデフォルト設定を示します。

Table 6　　FX3 DVK 基板のジャンパ

ジャンパ／スイッチ	ジャンパで短絡したピン (デフォルト設定)	機能 (デフォルト設定)
J101	1 と 2	GPIO_46=UART_RTS
J102	1 と 2	GPIO_47=UART_CTS
J103	1 と 2	GPIO_48=UART_TX
J104	1 と 2	GPIO_49=UART_RX
J136	3 と 4	VIO1 (3.3V)
J144	3 と 4	VIO2 (3.3V)
J145	3 と 4	VIO3 (3.3V)
J146	3 と 4	VIO4 (3.3V)
J134	4 と 5	VIO5 (3.3V)
J135	2 と 3	CVDDQ (3.3V)
J143	1 と 6	VBATT (2.5V)
J96 および SW25	2 と 3	• SW25 を使用する PMODE0 ピンの状態 (オン／オフ) の選択 • SW25.1=オフ
J97 および SW25	2 と 3	• SW25 を使用する PMODE0 ピンの状態 (オン／オフ) の選択 • SW25.2=オフ
J98	1 と 2	PMODE2 ピン開放
J72	1 と 2	リセット
J42	未実装	GPIO_58=I2C_SCL
J45	未実装	GPIO_59=I2C_SDA
J100	1 と 2	GPIO_21=CTL4

12.1　　FX3 DVK 用の JTAG デバッガ

Segger J-Link プローブは FX3 SDK に推奨されている JTAG プローブです。このプローブは Segger J-Link Arm GDB サーバーと共にデバッグに使用されます。Eclipse IDE はファームウェアをデバッグするために、J-link GDB サーバーに接続されます。Eclipse を GDB サーバーで動作させるために、J-link 用のデバッグ コンフィギュレーションを作る必要があります。詳細は「**FX3 Programmer's Manual**」の第 12 章を参照してください。

デバッグするために、Olimex Arm-USB-OCD プローブなど他の JTAG プローブを使用することができます。この場合、OpenOCD ツールは GDB 接続のために使用できます。詳細は「**FX3 Programmer's Manual**」の第 12 章を参照してください。

改訂履歴

Document version	Date of release	Description of changes
**	2012-12-14	これは英語版 001-75705 Rev. *A を翻訳した日本語版 001-85370 Rev. **です。
*A	2014-04-08	これは英語版 001-75705 Rev. *C を翻訳した日本語版 001-85370 Rev. *A です。
*B	2015-08-06	これは英語版 001-75705 Rev. *E を翻訳した日本語版 001-85370 Rev. *B です。
*C	2017-04-19	更新されたロゴと著作権。
*D	2018-10-03	これは英語版 001-75705 Rev. *I を翻訳した日本語版 001-85370 Rev. *D です。
*E	2021-07-20	テンプレートの変更を実施。 これは英語版 001-75705 Rev. *J を翻訳した日本語版 Rev. *E です。

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

Edition 2021-07-20
Published by
Infineon Technologies AG
81726 Munich, Germany

© 2021 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?
Go to www.cypress.com/support

Document reference
001-85370 Rev. *E

重要事項

本文書に記載された情報は、いかなる場合も、条件または特性の保証とみなされるものではありません（「品質の保証」）。本文に記された一切の事例、手引き、もしくは一般的価値、および／または本製品の用途に関する一切の情報に関し、インフィニオンテクノロジーズ（以下、「インフィニオン」）はここに、第三者の知的所有権の不侵害の保証を含むがこれに限らず、あらゆる種類の一切の保証および責任を否定いたします。

さらに、本文書に記載された一切の情報は、お客様の用途におけるお客様の製品およびインフィニオン製品の一切の使用に関し、本文書に記載された義務ならびに一切の関連する法的要件、規範、および基準をお客様が遵守することを条件としています。

本文書に含まれるデータは、技術的訓練を受けた従業員のみを対象としています。本製品の対象用途への適合性、およびこれら用途に関連して本文書に記載された製品情報の完全性についての評価は、お客様の技術部門の責任にて実施してください。

本製品、技術、納品条件、および価格についての詳しい情報は、インフィニオンの最寄りの営業所までお問い合わせください（**www.infineon.com**）。

警告事項

技術的要件に伴い、製品には危険物質が含まれる可能性があります。当該種別の詳細については、インフィニオンの最寄りの営業所までお問い合わせください。

インフィニオンの正式代表者が署名した書面を通じ、インフィニオンによる明示の承認が存在する場合を除き、インフィニオンの製品は、当該製品の障害またはその使用に関する一切の結果が、合理的に人的傷害を招く恐れのある一切の用途に使用することはできないこと予めご了承ください。