AN87216

GPIF™ II マスター インターフェースの設計

作成者: Sai Krishna Vakkantula

関連プロジェクト: あり

関連製品ファミリ: CYUSB3014

最新 FX3 SDK: こちらをクリック

関連アプリケーションノート: AN65974

その他のサンプル コードが必要な場合は、以下を参照してください。

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

AN87216 は、外部の同期スレーブ FIFO と通信するために、EZ-USB® FX3™マスター インターフェースを設計する方法を示します。この設計では、グラフィカルなステート マシン エントリによってインターフェースを開発するために FX3 GPIF™ II Designer ツールを使用します。この設計をテストするために、GPIF II インターフェースを介して 2 つの FX3 開発キット間で接続します。1 つは(本ノートの主題である)マスターとして、残りの 1 つはテスト スレーブとして動作します。マスターとスレーブの FX3 キットのファームウェア ソースコードおよび GPIF II ステートマシンはこのアプリケーション ノートに添付されています。

目次

1 はじめに 1
2 詳細情報 2
2.1 EZ-USB FX3 ソフトウェア開発キット 3
2.2 GPIF™ II Designer 3
3 GPIF II 入門 3
4 同期スレーブ FIFO インターフェース 5
4.1 同期スレーブ FIFO の アクセスシーケンス と インターフェース の タイミング 6
4.2 同期スレーブ FIFO の読み出しシーケンス 7
4.3 同期スレーブ FIFO の書き込みシーケンス 8
4.4 FX3 DMA のアーキテクチャの基本 9
4.5 スレーブ FX3 ファームウェアにおける DMA チャネル構成 10
4.6 フラグの設定 11
5 GPIF II Designer ツール 11
5.1 インターフェースの定義 12
5.2 ステートマシンの実装 13
5.3 GPIF II のアクション 15
5.4 GPIF のイベント 20
6 GPIF II マスターステートマシンの実装 20
6.1 FX3 GPIF II マスターの完成したステートマシン 21
6.2 FX3 マスターから FX3 スレーブへの書き込み ... 22
6.3 FX3 マスターが FX3 スレーブから読み出し 22
7 FX3 マスター ファームウェアの実装 23
8 ハードウェアの接続 24
9 デモを実行する手順 25
10 関連プロジェクト ファイル 29
11 まとめ 29
12 関連アプリケーションノート 29
付録: FX3 Development Kit (CYUSB3KIT-001)を使ったハードウェアの設定 30
改訂履歴 31
ワールドワイドな販売と設計サポート 32

1 はじめに

サイプレスの FX3 は、高い集積度と柔軟な機能を提供し、開発者があらゆるシステムに USB 3.0 機能を追加できるようにする USB 3.0 周辺機器コントローラです。

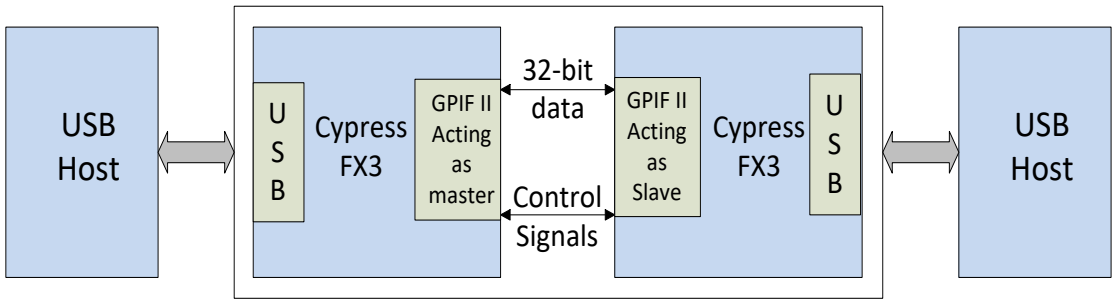
FX3 は、GPIF II と呼ばれる設定可能な、併用、汎用プログラマブル インターフェースを備えています。これはサイプレスの第二世代の汎用プログラマブル インターフェースです。GPIF II は外部のプロセッサ、ASIC、または FPGA に接続可能です。GPIF II は、非同期 SRAM、非同期および同期アドレス データ多重化インターフェースなど、人気の高い多くのインターフェースに固定しない接続を提供します。

GPIF II がよく使用される実装には、同期スレーブ FIFO インターフェースがあります。このインターフェースは、FX3 に接続した外部デバイスが FX3 FIFO にアクセスしてデータを読み出すまたは書き込むアプリケーションに使用されます。レジスタへのダイレクト アクセスは、スレーブ FIFO インターフェースを介しては実行されません。

このアプリケーション ノートは同期 FIFO マスター インターフェースを中心に説明します。マスターは転送を開始し、アドレス バスが存在する場合はそれを駆動し、常にスレーブにクロックを提供します。この設計に使用されるスレーブ デバイスは、GPIF II ユニットがスレーブ FIFO として動作するようにプログラムされる別の FX3 デバイスです。この設計をテストするために、各 FX3 開発キット(DVK)の GPIF II インターフェースを使用して、それら 2 つのキットを接続します。その内 1 つの FX3 DVK は FIFO スレーブ ユニットとして動作するようにプログラムされます。FIFO マスターの実装について詮索する前に、このスレーブ FIFO インターフェースの動作を本ノートで詳しく説明します。スレーブ FIFO インターフェースの詳細については、AN65974、EZ-USB® FX3™スレーブ FIFO インターフェースの設計を参照してください。

この例では、1 台の PC を使用してテスト データを 2 個の USB 3.0 ポートをとおして転送しますが、2 番目の PC が使用され 2 台の PC 間でデータを移動できます(図 1 を参照)。

図 1. ホスト間の相互接続ケーブル



次の節では、GPIF II、スレーブ FIFO インターフェースおよび GPIF II Designer ツールを使用するステート マシン実装の概要について説明します。

2 詳細情報

サイプレスは www.cypress.com で豊富なデータを提供して、デザインに適したデバイスを選択し、デバイスをデザインに素早く効率的に統合するのを支援します。包括的なリソースリストについては、ナレッジベースの記事 KBA87889, FX3/FX3S を使用した設計方法 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 製品選択:FX3, FX3S, CX3, HX3, Benicia
- 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幅広いトピックをカバーする多数の USB アプリケーションノートを提供しています。FX3 を使い始めるための推奨アプリケーションノートは次のとおりです。
  - AN75705 – EZ-USB FX3 入門
  - AN76405 – EZ-USB FX3 ブートオプションについて
  - 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
  - AN65974 – EZ-USB FX3 スレーブ FIFO インターフェースを使った設計
  - 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
  - AN86947 – EZ-USB FX3 で USB3.0 のスループットを最適化
  - AN84868 – EZ-USB FX3 を使用した USB 経由での FPGA コンフィギュレーション

- AN68829 – EZ-USB FX3 のスレーブ FIFO インターフェース: 5 ビットアドレスモード
- AN73609 – EZ-USB FX2LP/ FX3 のバルク - ループ例の Linux での開発
- AN77960 – EZ-USB FX3 ハイ - スピード USB ホストコントローラの導入
- AN76348 – EZ-USB FX2LP と EZ-USB FX3 アプリケーションの実装の違い
- AN89661 – EZ-USB FX3S を使用した USB RAID 1 ディスク設計

- サンプルコード
  - USB ハイ-スピード
  - USB フル-スピード
  - USB スーパースピード
- テクニカル リファレンス マニュアル (TRM)
  - EZ-USB FX3 テクニカルレファレンスマニュアル
- 開発キット
  - CYUSB3KIT-003, EZ-USB FX3 スーパースピード エクスプローラ キット
  - CYUSB3KIT-001, EZ-USB FX3 開発キット
- モデル: IBIS

2.1 EZ-USB FX3 ソフトウェア開発キット

サイプレスは 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、SuperSpeed USB を組込みアプリケーションに容易に統合します。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促進するツール、ドライバ、およびアプリケーションの例が付属しています。

2.2 GPIF™ II Designer

GPIF II Designer は、デザイナーが EZ-USB FX3 USB 3.0 デバイスコントローラの GPIF II インターフェースを設定できるグラフィカルソフトウェアです。

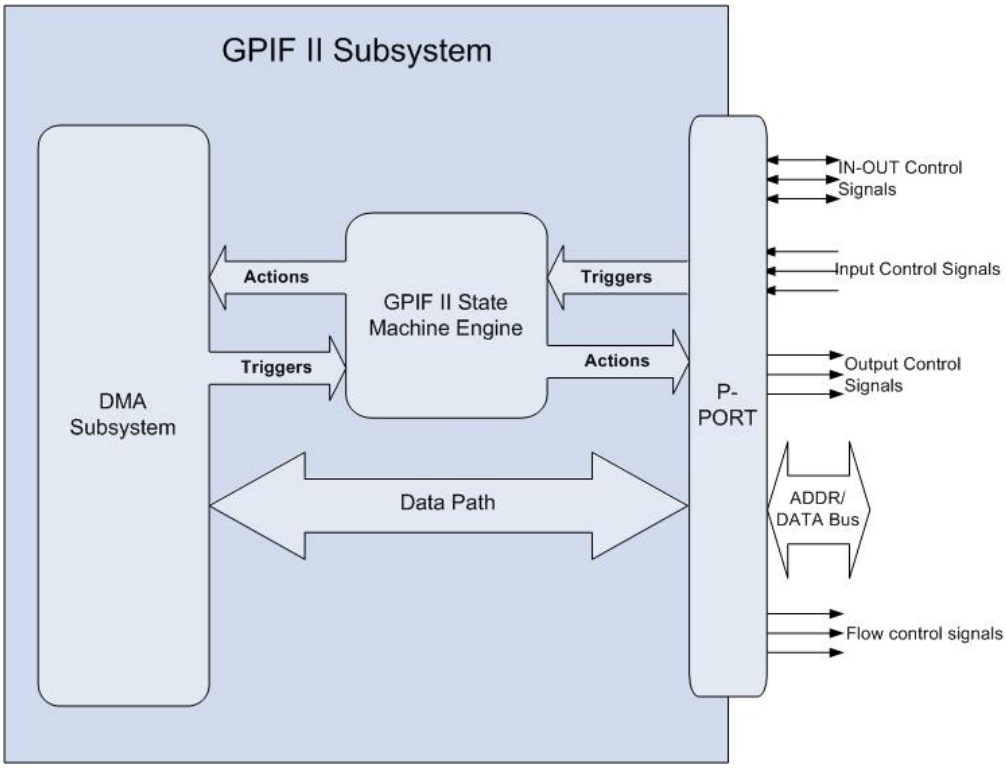
このツールを使用すると、サイプレスが提供する 5 つのインターフェースのいずれかを選択したり、独自の GPIF II インターフェースを作成できます。サイプレスは、非同期および同期スレーブ FIFO、非同期および同期 SRAM などの業界標準のインターフェースを提供しています。すでにシステム内にこれらの事前定義されたインターフェースの 1 つを持っているデザイナーは、単に選択肢のインターフェースを選択するだけで、バス幅（x8,16、x32）エンディアン、クロック設定などの標準パラメータのセットから選択し、インターフェースをコンパイルします。このツールは、カスタマイズされたインターフェースを必要とするユーザー向けに、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備えています。ユーザーはまず、ピン構成と標準パラメータを選択できます。第 2 に、構成可能なアクションを使用して仮想ステートマシンを設計できます。最後に、ユーザーは出力タイミングを見て、予想されるタイミングと一致することを確認できます。この 3 段階のプロセスが完了すると、インターフェースをコンパイルして FX3 と統合できます。

3 GPIF II 入門

GPIF II は、業界標準または独自のインターフェースのフレキシブルな実装を可能にする、プログラム可能なステート マシンです。GPIF II は外部のデバイスに対してマスターまたはスレーブのいずれとしても機能できます。

FX3 の P ポートとしても知られている GPIF II ポートは、最大 32 本の双方向データ ライン付きのパラレル インターフェースを提供します。データ ラインはアドレス ラインに分割されるか時分割されます。13 本の制御ラインが IN または OUT として設定可能です。プログラマブルなステート マシン エンジンは GPIF II ポートの動作と制御信号を制御します。ステート マシンの動作は、FX3 への入力として設定された制御信号を使用して外部のプロセッサによって制御されます。

図 2. FX3 の GPIF II サブシステム



GPIF ハードウェアは、バッファの準備状況を示すか、ユーザーがプログラムした閾値（フロー制御用に外部プロセッサによって使用可能）に基づいた一連の DMA 状態フラグの生成もサポートしています。転送中にデータ損失を起こさないようにこれらのフラグを使用してください。

カウンタおよびコンパレータ ブロックは GPIF ハードウェアに対応しており、ステート マシンによって使用されます。コンパレータは、アドレス、データまたは制御信号の現時点の状態が特定のパターンにマッチするかどうかを確認し、ステート マシンによりトリガとして使用できるマッチ信号を生成します。カウンタはステート マシンのアクションを介してリセットまたは更新され、さらにトリガとして使用できる限度マッチ信号も生成します。

GPIF II Designer ツールは、ユーザーが入力した設計に対応するコンフィギュレーション データを生成します。FX3 ファームウェアで実行している API は、これを入力として使用し、GPIF II ハードウェアを初期化します。FX3 の SDK には、GPIF インターフェースを設定、制御および監視するための一式の API が含まれています。

GPIF ハードウェアは、ステート マシンのキャンバスで選択したユーザー指定のアクションをとおして割込みを FX3 のオンボード ARM コア、および外部のプロセッサにトリガできます。GPIF II ステートマシンはまた、ファームウェアによって生成された入力信号を使用して、GPIF インターフェースの動作を制御できます。

GPIF II の特長は、以下のとおりです。

- マスターまたはスレーブとして動作
- ファームウェアでプログラム可能な 256 のステートを提供
- 8 ビット、16 ビット、32 ビットのパラレル データ バスをサポート
- 最大 100MHz まで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は、14 本の設定可能な制御ピンをサポート。すべての制御ピンは、入力／出力または双方向ピンのどちらにも利用できます。

- 16/8 ビット データ バスを使用する場合は、16 本の設定可能な制御ピンをサポート。すべての制御ピンは、入力／出力または双方向ピンのどちらにも利用できます。

GPIF II の状態遷移は、入力信号に基づいて発生し、制御出力信号は GPIF II の状態によって駆動されます。ステート マシンの動作は、必要なインターフェース仕様を満すように設計されたディスクリプタによって定義されます。GPIF II ディスクリプタは、基本的に一連のプログラム可能なレジスタの値です。FX3 レジスタ空間における 8K バイトは、GPIF II 波形メモリ専用であり、ここに GPIF II ディスクリプタが格納されます。 GPIF-II Designer ツールの詳細については、「GPIF-II Designer ユーザガイド」を参照してください。

GPIF II での一般的な実装は同期スレーブ FIFO インターフェースであり、以下の節で説明されます。続く節では、FX3 の GPIF II Designer ツールを使用して、同期スレーブ FIFO に対応するマスター インターフェースを設計する方法について説明します。

4 同期スレーブ FIFO インターフェース

このアプリケーション ノートの後節を最大限に活用するには、同期 FIFO インターフェースについての基礎知識があることを前提にしています。GPIF II マスター ステート マシンを理解するにあたって、同期スレーブ FIFO インターフェースの必要な詳細がここに説明されています。

本節には、同期スレーブ FIFO インターフェースの相互接続図、および信号のピン マッピングを示します。スレーブ FIFO インターフェースで読み出しおよび書き込み処理を実行するためのタイミング図は同期スレーブ FIFO の アクセスシーケンス と インターフェース の タイミング節に示します。スレーブ FX3 ファームウェアにおける P ポートのソケット、フラグおよび DMA チャネル構成はスレーブ FX3 ファームウェアにおける DMA チャネル構成節に示します。

同期スレーブ FIFO インターフェースは、外部プロセッサやデバイスが FX3 の内蔵 FIFO バッファに対して順次データ読み出し／書き込みアクセスを実行する必要があるアプリケーションに最適です。レジスタへのアクセスは、スレーブ FIFO インターフェースを介しては実行されません。普通は、高い処理能力の要件に対応するために、同期スレーブ FIFO インターフェースは USB アプリケーションのインターフェースとして選択されています。

図 3 には、同期スレーブ FIFO インターフェースの図を示します。

図 3. 同期スレーブ FIFO インターフェース

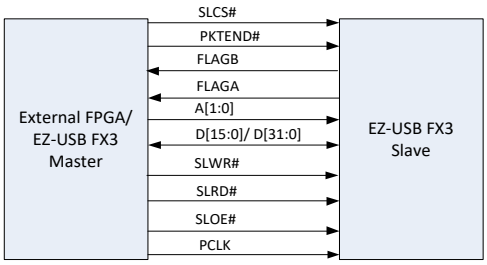


表 1. 同期スレーブ FIFO インターフェースの信号

信号名	信号の説明
SLCS#	スレーブ チップ セレクト、アクティブ ロー。マスターは、スレーブ インターフェースにアクセスするためにこの信号をアサートします。
SLWR#	スレーブ ライト ストローブ、アクティブ ロー。マスターは、(マスターからスレーブへの)書き込み処理を実行するために、この信号をアサートします。
SLRD#	スレーブ リード ストローブ、アクティブ ロー。マスターは、(スレーブからマスターへの)読み出し処理を実行するために、この信号をアサートします。
SLOE#	スレーブ出力イネーブル、アクティブ ロー。マスターは、スレーブがそのデータ バスを駆動するように、この信号をアサートします。デアサートされると、スレーブ データ バスはフロートです。

信号名	信号の説明
FLAGA／FLAGB	FX3 フラグの出力。これらは、様々なスレーブ状態を示すために、FX3 内でプログラムされます。このアプリケーションでは、FLAGA はスレーブ側でデータの可用性を、FLAGB はスレーブ側でフリー バッファの可用性を示すために設定されます。
A[1:0]	アドレス バス。この FX3 GPIF II の例では、スレーブ FX3 上にスレッドを選択するために 2 本のアドレス線を必要とします。
D[31:0]	これは、スレーブ FIFO インターフェースの 16 ビットまたは 32 ビット データ バスです。
PKTEND#	パケット終了、真。マスターは、この信号をアサートして、FX3 スレーブに長さゼロまたはショート パケットを USB 経由で送信するように命令します。
PCLK	マスターはこのクロックをスレーブに提供します。すべてのデータとタイミング処理はこのクロックを基準とします。

4.1 同期スレーブ FIFO の アクセスシーケンス と インターフェース の タイミング

本節では、同期スレーブ FIFO インターフェースのアクセス シーケンスとタイミングについて説明します。

このアプリケーション ノートに添付されている設計では、マスターFX3 は、スレーブ FX3 の内部 FIFO バッファへのバーストデータのアクセスを実行します。マスターFX3 は 2 ビット アドレスを ADDR ライン上で駆動し、読み出しか書き込みストローブをアサートします。スレーブ FX3 は USB 転送を実行し、空（FIFO 読み出しの場合）またはー杯（FIFO 書き込みの場合）の状態を示すために FIFO FLAG 信号をアサートします。

図 4 には、外部 FPGA／プロセッサから見たスレーブ FIFO インターフェースの論理図を示します。表 2 は FX3 GPIF-II マスタータイミング パラメータを示します。

表 2. FX3 GPIF-II マスター タイミング パラメータ

パラメータ	説明	Min	Max	Unit
Freq	クロック周波数	100	–	MHz
tDS_addr	入力 アドレス セットアップ 時間	3	–	ns
tDH_addr	入力 データ ホールド 時間	1.5	–	ns
tDO_addr	クロックからアドレス出力までの遅延	–	8	ns
tDOH_addr	出力アドレス ホールド 時間	2	–	ns
tS	入力コントロール セットアップ 時間	3	–	ns
tH	入力コントロール ホールド 時間	1.5	–	ns
tDS	入力データ セットアップ 時間	3	–	ns
tDH	入力データ ホールド 時間	1.5	–	ns
tDO	クロックからデータ出力までの遅延	–	8	ns
tDOH	出力データ ホールド 時間	2	–	ns
tCTLO	クロックからコントロール出力までの遅延	–	8	ns
tCOH	出力コントロール ホールド 時間	2	–	ns
tHZ	クロックから High-Z まで	–	8	ns

図 4. スレーブ FIFO インターフェースの論理図

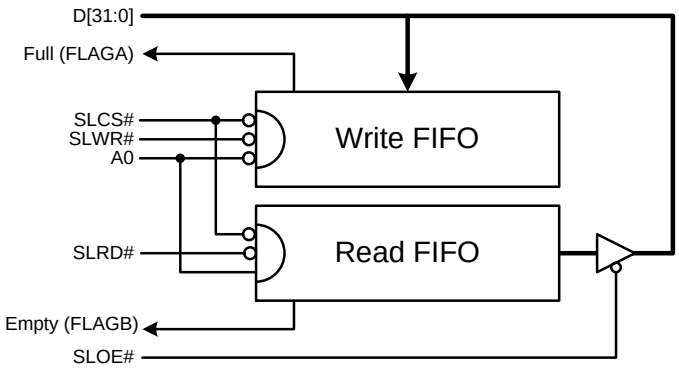
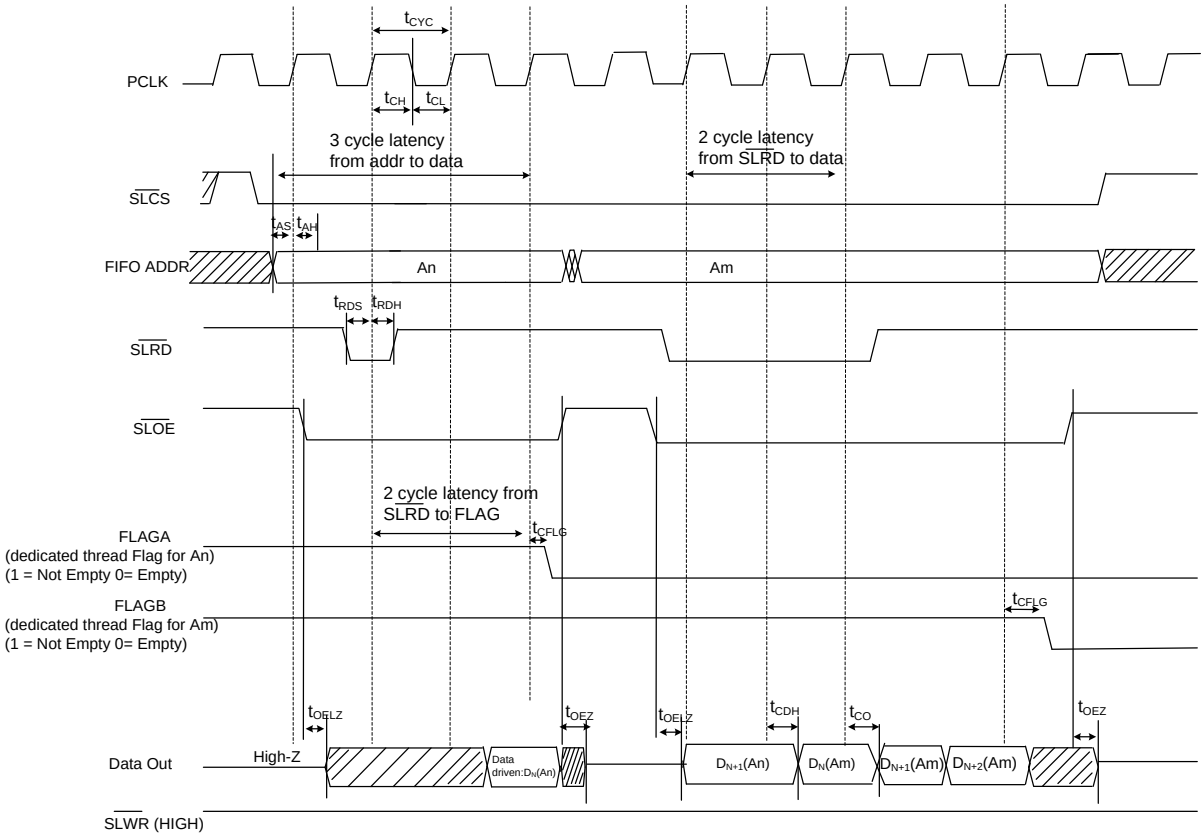


図 5. 同期スレーブ FIFO の読み込みシーケンス



4.2 同期スレーブ FIFO の読み出しシーケンス

図 5 には、マスターがプログラム済みの FX3 DVK であるスレーブから 1 個の 32 ビット ワードを読み出す方法を示します。この図は、FIFO アドレス「Am」からのバースト読み出しの次に FIFO アドレス「An」からの単一ワード読み出しを示します。

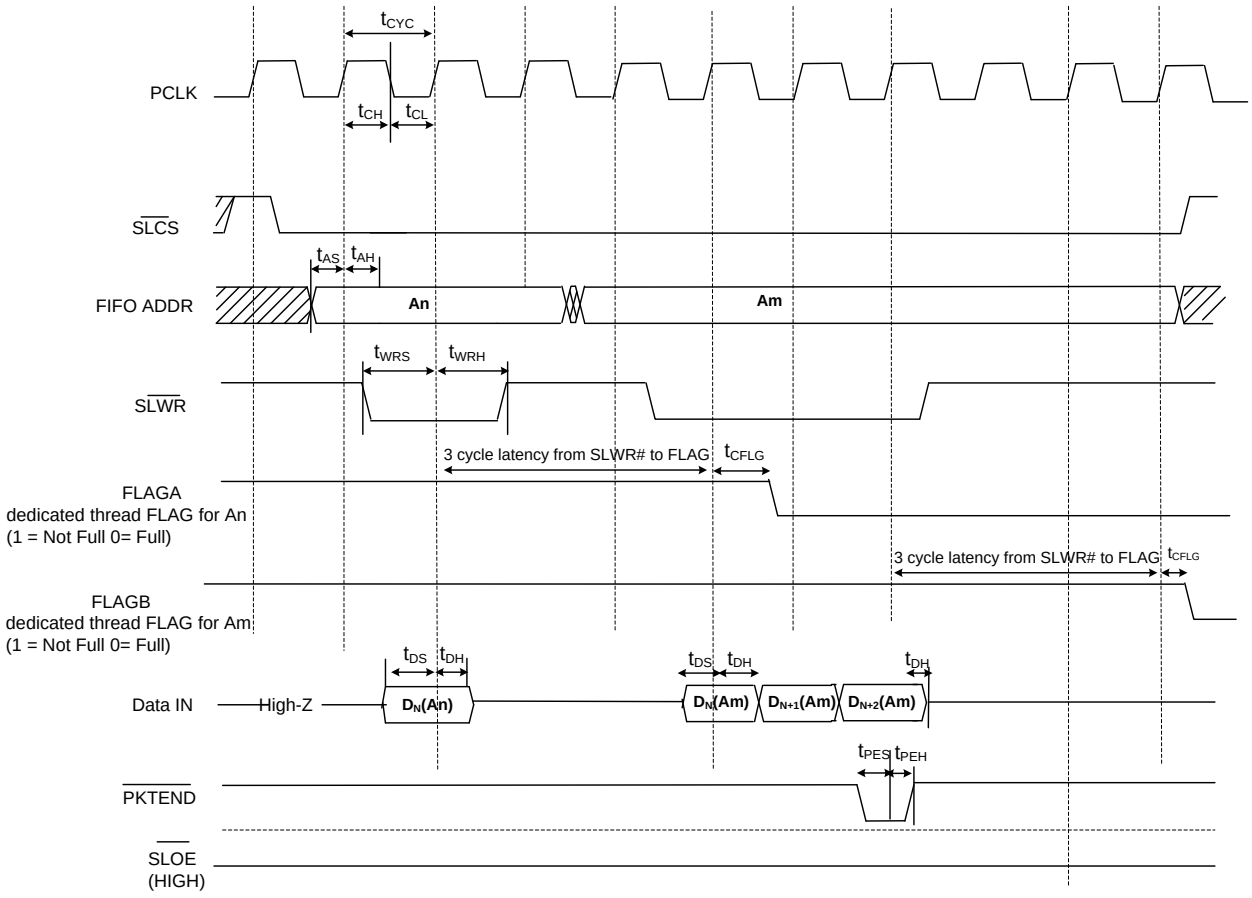
32 ビット ワード読み出しの転送は以下の手順に従います。

1. マスターは FIFO アドレス「An」を駆動して、SLCS#をアサートします。マスターは、アドレスが SLCS#をサンプリングする PCLK 立ち上がりエッジ の設定時間 tAS 要件を満たすことを確認しなければなりません。
2. スレーブがデータ バスを駆動し始めるために、マスターは SLOE#をアサートします。
3. マスターは SLRD#をアサートして、1 PCLK サイクルの間はローを保持します。これは、データ バスにアドレス指定された FIFO からのデータ伝播を開始します。

FIFO ポインタは、SLRD#のアサート中、PCLK の立ち上がりエッジで更新されます。これは、データ バスに新たにアドレス指定された FIFO からのデータ伝播を開始します。後の 2 つのクロック(PCLK の立ち上がりエッジから数えて)、新しいデータ値はクロック エッジ後に tCO を表示します。N は、FIFO から読み出される最初のデータ値です。データ バスを駆動するために、SLOE#をアサートする必要もあります。

読み出しバースト転送は図 5 の下半分に示されます。バースト モードでは、読み出し期間中は SLRD#と SLOE#がアサートされたままです。SLOE#が最初にアサートされると、以前にアドレス指定された FIFO からのデータを持つデータ バスが駆動されます。PCLK の後続の各立ち上がりエッジで、SLRD#がアサートされている間に、FIFO ポインタがインクリメントされ、次のデータ値がデータ バスに配置されます。図 5 の例では、取得した FIFO ワードは FIFO での最後のワードであり、FLAGB 信号によって「空」として表示されます。

図 6. 同期スレーブ FIFO の書き込みシーケンス



4.3 同期スレーブ FIFO の書き込みシーケンス

図 6 には、FIFO アドレス「Am」へのバースト書き込みの次に FIFO アドレス「An」へのシングル ワード書き込みを示します。

32 ビット ワード書き込みの転送は以下の手順に従います。

4. マスターは FIFO アドレス「An」を駆動してから、SLCS#をアサートします。マスターは、アドレスが SLCS#をサンプリングする PCLK 立ち上りエッジ の設定時間 tAS 要件を満たすことを確認しなければなりません。
5. マスターはそのデータをデータ バス上に駆動します。
6. マスターは、早くも SLCS#アサーション後の次のクロックで SLWR#をアサートします。
7. SLWR#がアサートされている間、マスターはデータを FIFO に書き込み、PCLK の立ち上がりエッジで FIFO ポインタがインクリメントされます。
8. FIFO フラグは、3 クロック+クロックの立ち上がりエッジからの tCFLG 遅延後に更新されます。

同じ一連のイベントはバースト書き込みを示します。

バースト モードでは、バースト書き込み期間中はマスターが SLWR#と SLCS#をアサートされたままにします。バースト書き込みモードでは、マスターが SLWR#をアサートした後、データ バス上の値が PCLK の各立ち上がりエッジで FIFO に書き込まれます。SLWR#がアサートされている限り、FIFO ポインタは、PCLK の各立ち上がりエッジで更新されます。

ショート パケット:マスターは、PKTEND#信号を使用して USB ホストにショート パケットを転送します。マスターFX3 は、最後のデータ ワードとそれに対応する SLWR#パルスと共に PKTEND#をアサートします。そうでない場合、即ち、SLWR#パルスなしで PKTEND#をアサートすると ZLP(長さゼロのパケット)になります。マスターは、FIFOADDR ラインを PKTEND#のアサート中は一定に保持しなければなりません。SLWR#と PKTEND#のアサート時に、スレーブ FX3 の GPIF II ステート マシンは、パケットをショート パケットとして解釈し、USB インターフェースに転送します。プロトコルがショート パケットを転送する必要がない場合、PKTEND#信号はハイ レベルに固定されることがあります。

読み出し方向では、ショート パケットの送信元が USB であることを示す特定の信号がないことに注意してください。空の FLAG は、全データが読み出された時点を判断するために、マスターFX3 によって監視される必要があります。

4.4 FX3 DMA のアーキテクチャの基本

FX3 デバイスには、GPIF II インターフェースを内部システム メモリおよび他のシリアル周辺に接続するための DMA 内部組織があります。GPIF II インターフェースを介して内部メモリ バッファからか、または内部メモリ バッファへのデータ転送が実行されます。FX3 で動作しているファームウェア アプリケーションは、このデータ経路を(DMA 組織を使用して)USB ホストまたはシリアル周辺などの適切なソースあるいはシンクに接続する責任を負います。

4.4.1 ソケット

USB 3.0 デバイスでの各ポートはデータ フローの終了に対応するソケットをサポートしており、独立にアドレス指定されます。FX3 P ポート(プロセッサ ポート)または GPIF II ポートは最大 32 個のソケットをサポートしています。つまり、 独立した 32 データ経路がこのインターフェースを介して設定されます。

ファームウェア アプリケーションは、ソケットを適切なデータ ソースまたはシンクに接続するために、使用されているすべてのソケットに対応するメモリ バッファを割り当てる責任があります。この設定を実行する機能については、 FX3 SDK API ガイドの DMA チャネル API を参照してください。

4.4.2 スレッド

FX3 デバイスの GPIF ポートは、データのプロバイダとコンシューマとの接続のために 32 個のアドレス指定可能なソケットを利用可能にしますが、その内は 4 本のデータ ハイウェイ、いわゆる「スレッド」のみが同時にデータを転送できます。つまり、アプリケーションがこれらのスレッドに結び付けられている最大 4 個のソケットを選択し、追加の待ち時間なしでそれらの間で切り替えられます。

一般的な FX3 アプリケーションでは、各終了時点でソケットを関連づけることによって 1 つ以上のスレッドを開始します。その内、1 つのソケットはデータを提供し、残りの 1 つはデータを使います。転送されたデータの各ワードに使用するスレッドは、入力アドレスを提供すること、または GPIF ステート マシンで IN_DATA あるいは DR_DATA アクション設定(GPIF II のアクション節を参照) のターゲット スレッドを指定することによって直接指定されます。

注:このアプリケーションに必要とされる技術ではないですが、アプリケーションの実行中に、ソケットとスレッドの関連割り当てをダイナミックに変更することもできます。スレッドにバインドされたアクティブなソケットを変更するのに追加の待ち時間が必要です。この切り替えは、ソケット アドレスを指定することで自動的に行われる(アドレス バスが 3～5 ビット幅の場合)、またはFX3 デバイス内のファームウェアの介入で行われます。

GPIF ハードウェアはスレッドごとに一連の DMA 状態フラグを提供します。スレッド固有の DMA フラグは常にそのスレッドでのアクティブなソケットの状態を反映します。スレッドでアクティブなソケットを切り替える場合、フラグを使用してデータ転送を制御する前に、そのフラグが新しいソケットの状態を反映するのを確実にするために十分な時間を割り当ててください。

スレッド マッピングへのソケットは完全に柔軟ではないことに注意してください。各ソケット N は番号付けられたスレッドのみにバインドされます(N MOD 4)。例えば、ソケット 7 はスレッド 3 のみを使用します(7:4 の余り)。ソケット 11 もスレッド 3 を使用します。つまり、ソケット 0、4、8 などを異なるスレッドにバインドしてそれらを同時に使用できません。これは制限を示すわけではありません。ソケットがスレッドよりも多い(32 対 4)ので、多くの選択肢があります。GPIF データ転送のためにソケットを選択する時はこの定数を考慮してください。

4.5 スレーブ FX3 ファームウェアにおける DMA チャネル構成

ファームウェアは、要求されたプロデューサとコンシューマ ソケットで DMA チャネルを設定します。

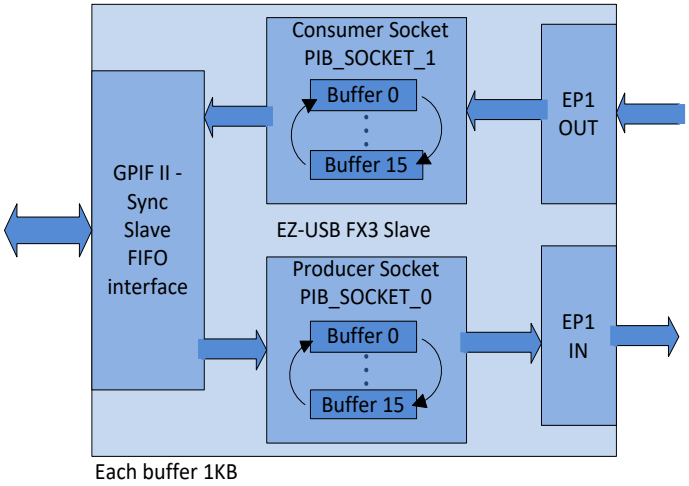
データをスレーブ FIFO インターフェースから USB インターフェースへ転送する場合、P ポートまたは GPIF II ポートはプロデューサとなり、USB はコンシューマとなります。その逆の場合も同様です。

したがって、データがスレーブ FIFO インターフェースを介して双方向で転送される場合、2 本の DMA チャネルを設定します。ファームウェアにおける CyU3PDmaChannelCreate API を使用してください(API の詳細については、FX3 API ガイドを参照)。1 本のチャネルの P ポートをプロデューサ、残りのチャネルの P ポートをコンシューマとして使用します。

P ポート プロデューサ ソケットは、外部デバイスがスレーブ FIFO インターフェースを介して書き込むソケットです。P ポート コンシューマ ソケットは、外部デバイスがスレーブ FIFO インターフェースを介して読み出すソケットです。

このアプリケーション ノートに添付されている例では、P ポート側でソケット 0 はプロデューサ ソケットとして、ソケット 1 はコンシューマ ソケットとして設定されます。(図 7 を参照)。

図 7. 同期スレーブ FIFO ファームウェアで使用される PIB ソケット



DMA チャネル内の P ポート ソケット番号は、A1:A0 でアドレス指定されるソケット番号となることに注意してください。

チャネルの設定中に、複数のバッファを特定の DMA チャネルに割り当てられます。フラグがバッファごとに一杯/空の状態を示すことに注意してください。(バッファの最大サイズは 64 kB-16 です。)

例えば、1024 バイトの 2 つのバッファが DMA チャネルに割り当てられた場合、フル FLAG は、1024 バイトが最初のバッファに書き込まれた時に一杯の状態を示します。それは、DMA チャネルが 2 番目のバッファに切り替わるまで、一杯の状態を示したままです。DMA チャネルが次のバッファに切り替わるのに要する時間は、普通は数マイクロ秒ですが、予測できません。外部マスターは、この切り替えが完了して次のバッファがデータ アクセス用に使用可能になる時点を判断するために、FLAG を監視する必要があります。

次の節は、異なるスレッドの状態を示すために FLAG をどのように設定するかについて説明します。

4.6 フラグの設定

フラグは、空、一杯、部分的に空、または部分的に一杯の信号として設定されます。これらは、GPIF II ステート マシンによって制御されず、FIFO を FX3 RAM から出力させる EZ-USB FX3 の内蔵 DMA ハードウェア エンジンによって制御されます。フラグは、特定のスレッドまたは現時点でアドレス指定されているスレッドに動的に関連付けられています。どの場合でも、そのスレッドにマッピングされたソケットの状態を示します。

フラグは、ソケットの方向(ソケットの初期化時に設定された)に基づいて、空または一杯の状態を示します。そのため、フラグは、データがソケットから読み出されている時に空の状態を示し、データがソケットに書き込まれている時に一杯の状態を示します。

使用可能な FLAG のタイプは以下のとおりです。

- 専用のスレッド フラグ(空/一杯、または部分的に空/一杯)
- 現時点のスレッド フラグ(空/一杯、または部分的に空/一杯)

4.6.1 専用のスレッドフラグ

フラグは、特定のスレッド 0~3 の状態を示すように設定できます。この場合、そのフラグはそのスレッド専用となり、どのスレッドがアドレス バス上でアドレス指定されているかに関わらず、その特定のスレッドにマッピングされたソケットの状態のみを示します。

この場合、外部プロセッサ/デバイスは、どのフラグがどのスレッド専用であるかを追跡し、他のスレッドがアドレス指定されるたびに正しいフラグを監視する必要があります。

例えば、FLAGA がスレッド 1 専用、FLAGB がスレッド 0 専用である場合は、外部プロセッサがスレッド 1 へのアクセスを実行する際に、 FLAGA を監視する必要があります。外部プロセッサがスレッド 0 にアクセスする際、FLAGB を監視する必要があります。

フラグは、アクセスされるすべてのスレッド専用になれます。アプリケーションが 4 つのスレッドにアクセスする必要がある場合は、4 つの対応するフラグがあります。

このアプリケーション例では、FLAGA はスレッド 1 専用、FLAGB はスレッド 0 専用です。したがって、マスターFX3 がスレーブ FIFO インターフェースで書き込み処理を実行する場合、FLAGB で HIGH を待機する必要があります。FLAGB での HIGH は、このデータ経路に割り当てられた DMA バッファが FULL ではないことを示します。FDMA バッファが FULL になると、FLAGB が LOW になります。同様に、スレーブ FX3 デバイスで読み出し処理を実行する場合、マスターFX3 は FLAGA で HIGH を待機する必要があります。FLAGA での HIGH は、このデータ経路に割り当てられた DMA バッファが EMPTY ではないことを示します。DMA バッファにデータがない場合は、FLAGA は LOW のままです。

スレーブ FIFO インターフェース実装の詳細については、AN65974、EZ-USB® FX3™スレーブ FIFO インターフェースでの設計を参照してください。

5 GPIF II Designer ツール

以上、スレーブ FIFO インターフェースの要件を定義しました。次は GPIF II Designer を使用して GPIF II の設計を実装する方法を学びましょう。GPIF II Designer は、ステート マシン エントリおよびメニュを使用して FX3 の GPIF II インターフェースを設定するためのグラフィカル ツールです。

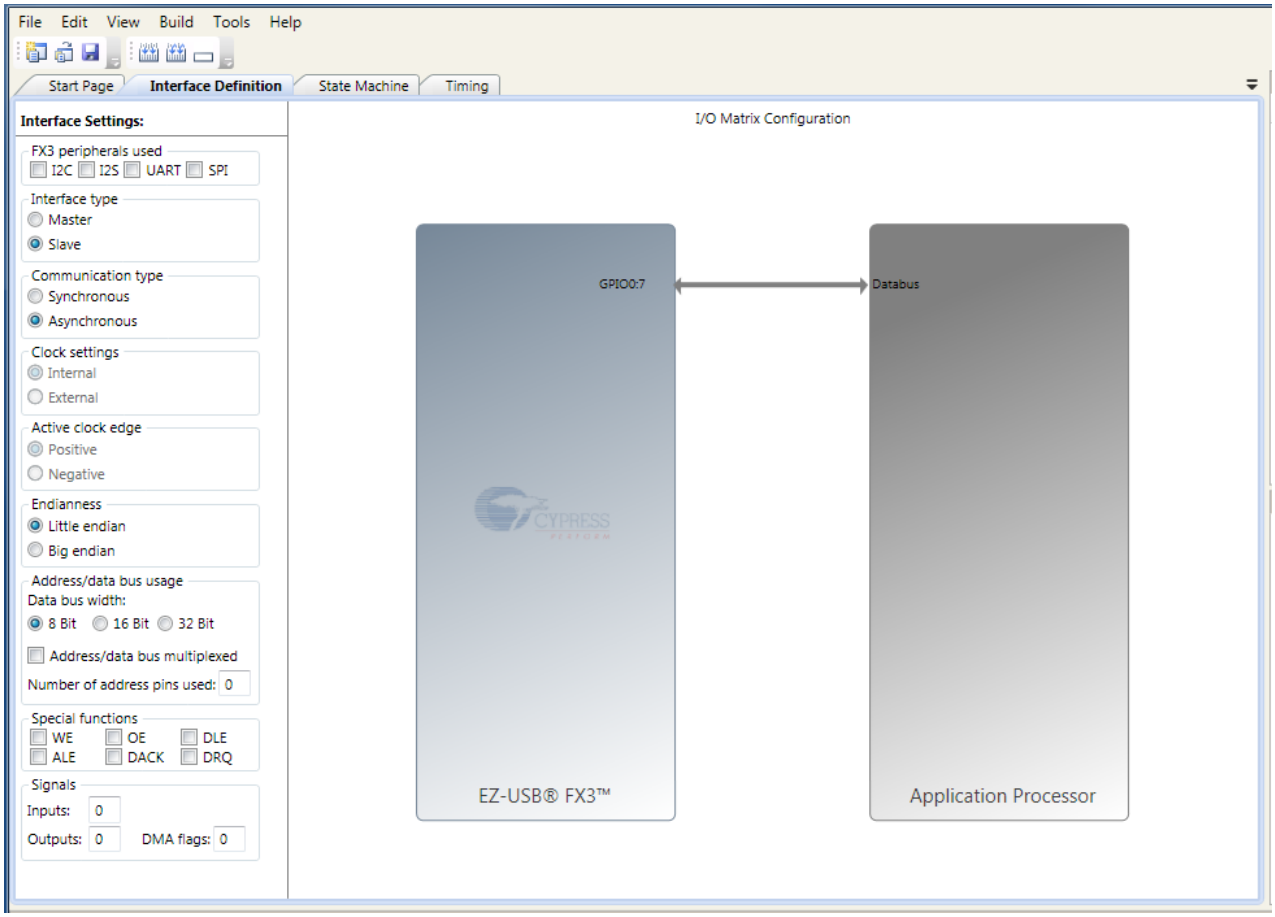
GPIF II Designer を使用することにより、サイプレスの作成済みインターフェースのライブラリから選択するか、または最初から GPIF II インターフェースを作成できます。 サイプレスは、非同期と同期スレーブ FIFO、非同期と同期 ADMUX(アドレス/データ多重化 RAM)および非同期 SRAM などの業界標準インターフェースを供給します。あらかじめ定義されたこれらのインターフェースのいずれかを既にシステム内に備えている設計者は、バス幅(x8、x16、x32)、エンディアンおよびクロック設定など一連の標準記号からインターフェースを選択して、それをコンパイルできます。

このツールでカスタム インターフェースを設計するために、最初は「インターフェース定義」タブで使用可能なピン設定および標準記号を選ばなければなりません。そして、「ステート マシン」タブを選択し、設定可能な動作を使用してステート マシンを設計します。それを完了すると、GPIF II Designer は FX3 プロジェクトにおける処理に適するようにインターフェースを C ヘッダファイルにコンパイルします。

GPIF II は FX3 SDK インストールの一部です。スタンドアロンのインストール ファイルは、サイプレスのウェブページの次の場所からダウンロードできます。

図 8 には、新期プロジェクトの GPIF II Designer スクリーンの一部分を示します。

図 8. GPIF II Designer ツール



5.1 インターフェースの定義

まず、「インターフェース設定」タブ（図 8）でエントリに記入することによって GPIF II 外界インターフェースを設定します。オプションを選択または選択解除すると、中央パネルの回路図はインターフェースの変更を反映します。FX3 信号が FX3 ブロックでラベル付けられているため、FX3 のピン マッピングを調べる手間を省けます。

1. どの FX3 シリアル周辺がアプリケーション全体に使用されていますか？ボックスを選択することにより、GPIF II Designer は、「I2C」が選択された場合に、SCL および SDA などのピンが他の FX3 ペリフェラルに割り当てられることを防げます。
2. FX3 はインターフェースのマスターまたはスレーブとして動作しますか？いずれかが実装されている場合、マスターは転送を開始し、アドレス バスを駆動します。マスターまたはスレーブを選択すると、アクションのリスト（右側のパネル）は、有効な選択肢を示すために自動的に更新します。
3. このインターフェースはクロックを使用しますか？使用する場合は、「Synchronous」を選択してください。非同期インターフェースの例としては、アドレスとデータ バス、読み出しと書き込みストローブはあるがクロックがないスタティック RAM を取りあげます。例えば、読み出し動作は、アドレスをアサートして、チップ イネーブルと読み出しストローブをアサートすることによって発生します。データは、読み出しストローブ後にクロックに関係しない伝播時間で出力します。
4. FX3 GPIF II はインターフェースのクロックを駆動することを望みますか？望む場合は内部であり、望まない場合は外部です。質問 3 で「同期」を選択しないと、このセクションは淡色表示になります。

5. あなたは GPIF II がいつインターフェースの信号をサンプリングしようとしますか？ポジティブ エッジですか、それともネガティブ エッジですか？質問 3 で「Synchronous」を選択しないと、このセクションは淡色表示になります。
6. あなたのデバイスのエンディアンは？リトル エンディアンですか、ビッグ エンディアンですか？「エンディアン」とは、上位（ビッグ）または下位（リトル）バイトのマルチ バイト整数のバイト オーダーを示すものです。
7. FX3 に接続している外部デバイスのデータ バス幅は 8 ビット、16 ビットまたは 32 ビットです。
8. 外部のデバイスはアドレス ラインを必要としますか？ソケットなど FX3 のリソースを選択する場合に、アドレス ラインが必要です。アドレスとデータ バスは多重化されていますか？
9. 「特別な機能」はユニークなハードウェアピンと構成を示します。
10. **WE**:WE の特別な機能は GPIO_18 にのみ接続されます。この接続では、GPIO_18 ラインをアサートすると、入力パスのデータ バスの（FX3 デバイスへの）方向を設定します。

    OE:この機能は GPIO_19 でのみ使用可能です。この機能では、GPIO_19 をアサートすると、出力パスのデータ バスの（FX3 からの）方向を直接変更できます。

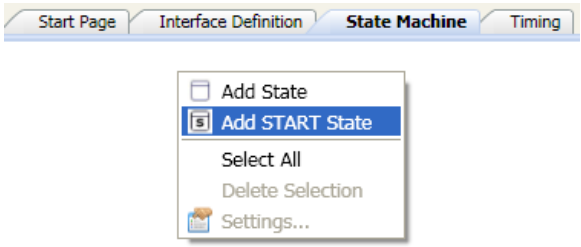
    DLE:データ ラッチ イネーブル（DLE）機能は、GPIO_18 ラインのデアサートを使用して、データ ラインを数ナノ秒追加ラッチします。FX3 デバイスが GPIO_18 のデアサート エッジで入力パスのデータ ラインを読み出し中に、この機能を使用します。
11. FX3 によっていくつの信号が監視されますか？いくつが制御されますか？また、アプリケーションでいくつの DMA フラグが必要ですか？入力または出力を追加すると、ステート マシンの設計者はそれらをいかなる状態でテスト（入力）またはアサート（出力）されるかを選択肢として自動的に追加します。

5.2 ステートマシンの実装

インターフェースの定義が完了した後、次のステップはステート マシンを実装することです。参照のため、以下に示される画像をご覧ください。

すべてのステート マシンには START 状態があります。START 状態を追加するために、右クリックして「START 状態を追加」を選択してください。START 状態には着信遷移がありません。START 状態に由来する遷移の方程式は「LOGIC_ONE」に固定されています。

図 9. START 状態を追加



状態を追加するために、右クリックして「状態を追加」を選択します。図 10 はこの手順を示します。

図 10. 状態を追加

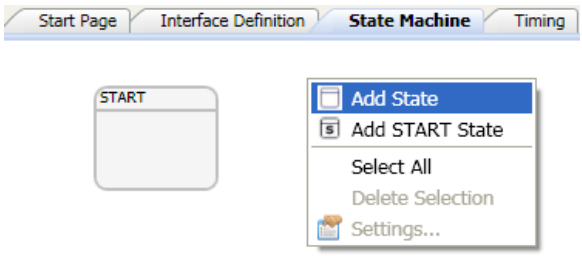
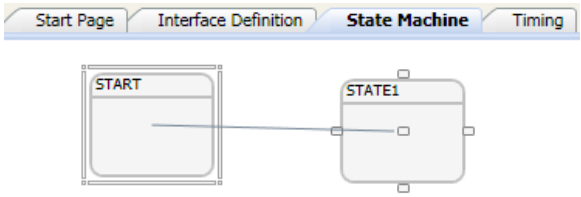


図 11 に示すように、2 つの状態を接続できます。図 11 は START から STATE1 への遷移を示します。

図 11. 異なる状態の接続

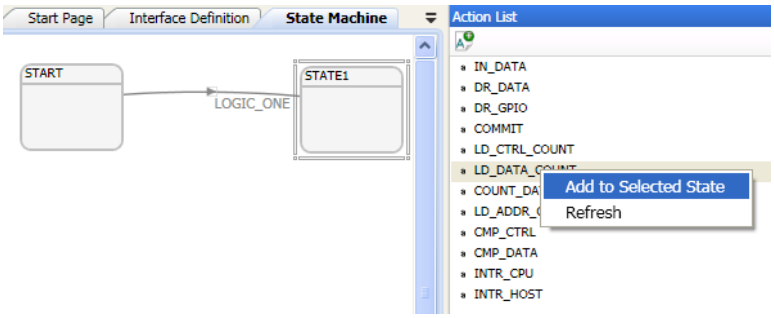


アプリケーションの要件に基づいて、状態にアクションを追加します。マスター スレーブ FIFO インターフェースを実装するのに必要なアクションのリストは次の節で説明します。

図 12 には、状態にアクションを追加する方法を示します。状態ボックスを選択した後、「アクションのリスト」パネル内の選択肢のいずれかを右クリックして「選択した状態に追加」を選択してください。または、選択した状態ボックスにアクションをドラッグできます。各アクションのエントリの上にマウス ポインタを移動させると、そのアクションを説明するヒントが表示されます。状態からアクションを削除するために、状態ボックス内の名前テキストに右クリックして「アクション削除」を選択してください。

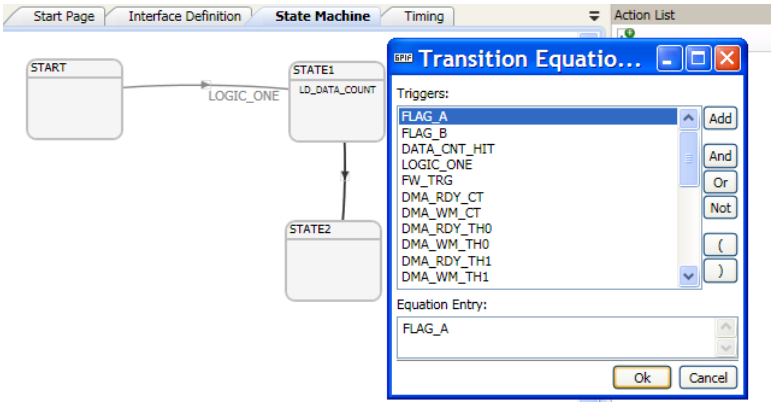
1 つ以上のアクションが STATE ボックスに割り当てられることに注意してください。同じ状態に競合のアクションをインクルードしようとすると、GPIF II Designer はエラー メッセージを発行します。例として、同じ状態でデータ カウンタをロードおよびインクリメントしようとする場合です。

図 12. 状態にアクションを追加



遷移ラインにダブルクリックした後、その状態の使用可能なイベントのリストを取得します。図 13 に示したように、遷移の方程式は追加されます。

図 13. 遷移の方程式を追加



5.3 GPIF II のアクション

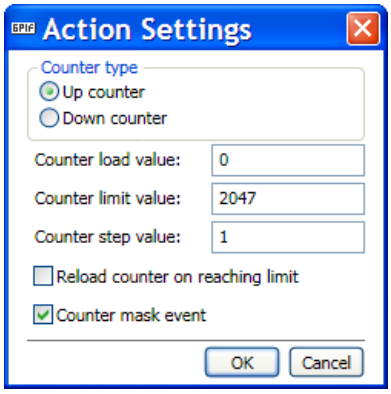
GPIF II ステート マシンの各状態は 1 つ以上の GPIF II アクションを実行するようにプログラムされています。ある状態で実行されているアクションは、別の状態に遷移するまで、一回またはクロック エッジごとに発生するようにプログラムされます。マスター ステート マシンの実装で使用される GPIF II のアクションはここに説明します。GPIF II のアクションの説明については、GPIF II Designer におけるヘルプに移動し、トピックにクリックする、または F1 キーを押してください。新たに開いた GPIF II Designer ウィンドウの内容タブを選択し、GPIF II ステート マシンのプログラミングセクションの GPIF II アクションサブセクションをブラウズします。GPIF II Designer ウェブページから同じ詳細を見られます。

5.3.1 LD_DATA_COUNT (ロード データ カウンタ)

この処理は特定の状態に追加されるとデータ カウンタを設定します。設定された LD_DATA_COUNT 処理は別の値で繰り返せません。

FX3 ファームウェア API において CyU3PgpifInitDataCounter()関数を使用することによって、等価のアクションを行えます。FX3 API ガイドは関数パラメータの情報を提供します。CyU3PGpifInitDataCounter()関数でデータ カウンタを設定しているにも関わらず、このアクションは GPIF II ステートマシンの状態に追加されることに注意してください。そうでない場合は、DATA_CNT_HIT イベントは遷移方程式のリストで使用できなくなります。この API を使用することによって、実行中にデータ カウンタを変更できます。

図 14. LD_DATA_COUNT アクションの設定



以下のパラメータはこのアクションに関連付けられています。

- Counter Type(カウンタの種類)(アップ／ダウン):この場合は、アップを選択します。
- Counter load value(カウンタのロード値):このアクションが実行されると、初期のカウントはロードされます。この場合は、0 でロードします。
- Counter limit value(カウンタの制限値):イベントが生成されるカウント値です。32 ビットのデータ バスを使用して 8K バイトのデータを読み出すために 2047 でロードします。この値は簡単な式で計算できます。

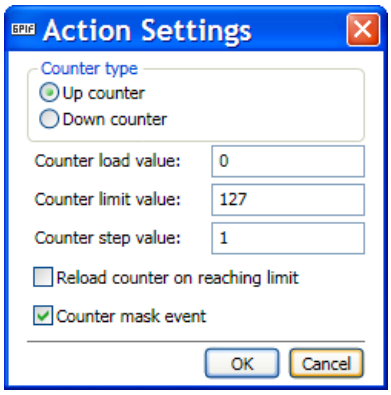
  カウンタの制限値 = (読み出すバイト数/ バイト単位のデータ バス幅)-1*。

  カウンタのロード値が0の場合にのみ、*1は減算されます。
- Reload counter on reaching limit(限界値到達時のカウンタ リロード):このパラメータのボックスをチェックすることでカウントが制限値に達すると、カウントのロード値がリロードします。そうでなければ、ワンタイムのカウンタとして動作します。
- Counter mask event(カウンタ マスク イベント):このボックスがチェックされない場合、カウンタが限界値に達するとファームウェア イベントを生成します。
- Counter step value(カウンタのステップ値):COUNT_DATA アクションを使用するたびに、カウンタのステップ値は加算／減算されます。DATA_CNT_HIT イベントを有効にするために、この値は 0 よりも大きくなければなりません。

5.3.2 LD_ADDR_COUNT (ロード アドレス カウンタ)

このアクションは LD_DATA_CNT と同様です。

図 15. LD_ADDR_COUNT アクションの設定



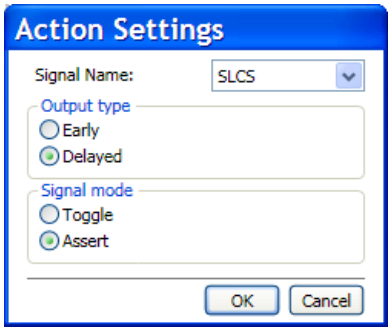
以下のパラメータはこのアクションに関連付けられています。

- Counter Type（カウンタの種類）（アップ／ダウン）：カウンタは昇順または降順でカウントするように設定されます。
- Counter load value（カウンタのロード値）：このアクションを実行すると、初期のカウントがロードします。
- Counter limit value（カウンタの制限値）：イベントが生成されるカウント値です。32 ビットのデータ バスを介して 512KB のデータを読み出すために 127 でロードします。
- Reload counter on reaching limit（限界値到達時のカウンタ リロード）：このパラメータのボックスをチェックすることでカウントが制限値に達すると、カウントのロード値がリロードします。
- Counter mask event（カウンタ マスク イベント）：このボックスがチェックされない場合、カウンタが限界に達するとファームウェア イベントは生成されます。
- Counter step value（カウンタのステップ値）：COUNT_ADDR アクションを使用するたびに、カウンタのステップ値は加算または減算されます。

5.3.3 DR_GPIO(GPIO を駆動)

DR_GPIO アクションは GPIO ピンを HIGH、LOW に駆動またはトグルします。アクションは、同期処理の場合は 2 または 3 サイクル、 非同期処理の場合は 25ns または 30ns と指定される「アサーション遅延時間」後に実行します。このアクションによって駆動された GPIO は、次の状態への遷移中にデアサートされます。つまり、ステートマシンのすべての状態で特定の GPIO を HIGH か LOW にアサートしようとする場合は、全状態にこのアクションを追加する必要があります。すべての状態にアクションを追加しない場合は、最初の状態にのみ追加してトグル モードを選択してください。

図 16. DR_GPIO アクション設定



以下のパラメータはこのアクションに関連付けられています。

- Signal Name（信号名）：信号を示すユーザー定義の英数字の文字列をここに記入します。この名称はステート マシンのキャンバスに表示します。
- Output Type（出力の種類）：このパラメータは、駆動されている出力信号の遅延を制御します。Early パラメータ ボックスがチェックされている場合、遅延時間は、非同期モードでは 25ns であり、同期モードでは 2 クロック サイクルです。Delayed パラメータ ボックスがチェックされている場合、遅延時間は、非同期モードでは 30ns であり、同期モードでは 3 クロック サイクルです。
- Signal Mode（信号モード）：トグル モードでは、含有状態を終了すると、信号値はトグルされます。アサート モードでは、信号の値は、出力用に指定されたアサート極性で駆動されます。アサート極性を選択するために、インターフェース定義のタブに切り替え、アプリケーション プロセッサの信号名をダブルクリックします（例えば、「OUTPUT0」）。これにより、ピンの初期値およびアクティブ HIGH またはアクティブ LOW 極性を設定するためのダイアログを示します。例えば、「アサート」のアクションが実行している場合、アクティブ LOW を選択するとピンがローに駆動されます。

注：

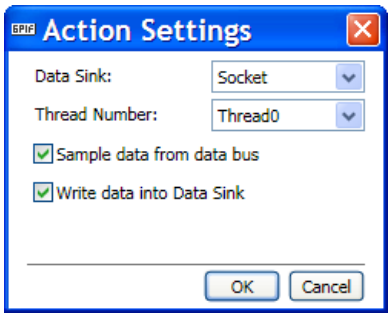
1. 遅延出力および信号モードの設定は各信号に対してはグローバルであり、アクションが他の状態で使用されるたびに変更することはできません。ツールは、各出力信号のための最終設定に対応するコンフィギュレーション ファイルを生成します。
2. 状態設定のダイアログ ボックスの「次の遷移までアクションを繰り返す」は DR_GPIO アクションの動作に影響を与えません。DR_GPIO アクションは各クロック サイクルごとに繰り返されます（クロックはインターフェース クロックまたは FX3 の内部クロックです）。「トグル」を選択すると、GPIO は含有状態の間すべてのクロックでトグルします。

5.3.4 IN_DATA (入力データ)

IN_DATA アクションはデータ バスからデータをサンプリングして指定された転送先に移動させます。転送先は DMA チャネルあるいはファームウェア アプリケーションです。データ シンクをレジスタとして選択する場合、CyU3PgpifReadDataWords()API はファームウェアにデータを取得するために使用されます。この API の詳細については、FX3 API のガイドを参照してください。4 つの転送先のスレッドから直接 1 つを選択するオプションは、転送先はアドレス指定モードがステート マシンによって選択されたスレッド（アドレス線の数 =0）に設定された DMA チャネルである場合にのみ使用可能です。

データ バスからのデータのみをラッチし、選択した転送先に格納しない、または以前にラッチされたデータをその転送先に格納することが可能です。これらのオプションは、バス上のデータがデータ可用性を示すストローブ信号をリードするという特定のプロトコルを満たすために、使用可能とされます。

図 17. IN_DATA アクションの設定



以下のパラメータはこのアクションに関連付けられています。

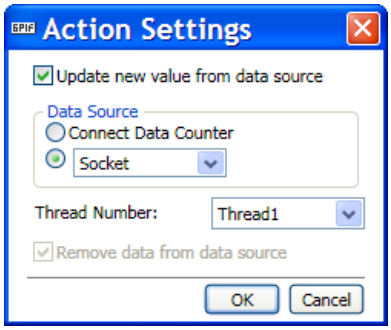
- *Data Sink*（データ シンク）– レジスタ／ソケット／PP レジスタ
- *Thread Number*（スレッド番号）– データ シンクが関連付けられているスレッド番号スレッド 0～3 を選択可能。（この機能は、スレッド番号を選ぶためのアドレス線がない場合、またはマスター モードが有効な場合に利用できます。）
- *Sample data from data bus*（データ バスからデータをサンプリング）– このオプションはデータ バスからデータをサンプリングしますが、データを指定されたデータ シンクに書き込みません。
- *Write data into Data Sink*（データ シンクにデータを書き込み）– ユーザーがサンプリングされたデータを指定のデータ シンクに転送しようとする時に、このオプションはチェックされます。

こられ 2 つの選択肢により、データをサンプリングしてその転送先に書き込むことが可能です。

5.3.5 DR_DATA (データを駆動)

DR_DATA アクションはデータを指定された転送元からデータ バス上にデータを駆動します。転送元は DMA チャネルまたはファームウェア アプリケーションです。CyU3PgpifWriteDataWords()API は、データ ソースがレジスタとして選択された場合、ファームウェアからデータを書き込むために使用されます。この API の詳細については、FX3 API のガイドを参照してください。転送元をスレッドとして選択するオプションは、転送元はアドレス指定モードがステート マシンによって選択されたスレッド（アドレス線の数 =0）に設定された DMA チャネルである場合にのみ使用可能です。

図 18. DR_DATA アクションの設定



以下のパラメータはこのアクションに関連付けられています。

- Update new value from data source（データ ソースから新しい値を更新）– このオプションは、データ ソースにあるデータ ワードでデータ バスを更新し、指定されたソースからのワードを削除します。
- Data Source（データ ソース） – このオプションは、ユーザーがデータ カウンタまたはレジスタ／ソケット／ PP レジスタを選択するのに役立ちます。
- Thread Number（スレッド番号）– データ ソースが関連付けられているスレッド番号。スレッド 0～3 を選択可能。（この選択は、スレッド番号を選ぶためのアドレス線がない場合にのみ利用できます。）つまり、インターフェース定義のタブにおける「使用されたアドレス ピン数」のフィールドが 0 にセットされています。
- Remove data from data source（データ ソースからデータを削除）：このオプションが無効の場合、ソース データが破棄されていないために別の状態で使用できます。

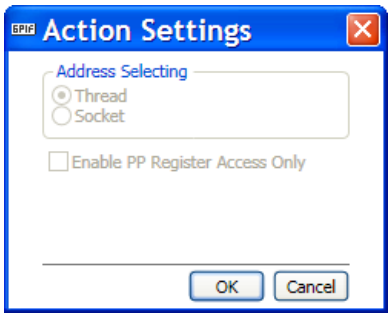
注：状態設定のダイアログ ボックスでは、「データ ソースから新しい値を更新」のフラグは「次の遷移までアクションを繰り返し」のフラグをオーバーライドします。つまり、「データ ソースから新しい値を更新」のチェックボックスが選択されている場合、「次の遷移までアクションを繰り返し」のフラグをチェックしないで、GPIF II がその状態になるとデータはすべてのクロック サイクル内に更新します。

5.3.6 IN_ADDR (入力アドレス)

IN_ADDR アクションにより、GPIF ハードウェアがアドレス バスから値をサンプリングし、それを使用して DMA スレッドまたはソケットを選択します。0 アドレス ビットがインターフェース定義のタブに指定されている場合、アドレス選択の選択肢が淡色表示になります。

アドレスのバス幅が 2 ビット以下の場合、アドレスは 4 つの DMA スレッドの内 1 つのみを選択できます。アドレスは 3～5 ビット幅の場合、DMA スレッドまたは特定のソケットのいずれかを選択可能です。上のダイアログに示されている「アドレス選択」のパラメータはこの選択のために使用されます。

図 19. IN_ADDR アクションの設定



以下のパラメータはこのアクションに関連付けられています。

- Address Selecting（アドレス選択）– スレッドまたはソケット

インターフェース定義の値を変更すると、アクションの選択肢は自動的に即座に変更されます。例えば、「使用のアドレス ピン数」を 0 に変更すると、IN_ADDR の選択肢はアクションのリストから消えます。

5.3.7 COUNT_DATA, COUNT_ADDR

LD_DATA_COUNT アクションを介して、設定されたステップ値でデータ カウンタを更新します。この更新によりカウントが指定された制限に達すると、DATA_CNT_HIT のトリガは真となります。COUNT_ADDR はアドレス カウンタと同じアクションを行います。

このアクションに関連付けられているパラメータがありません。

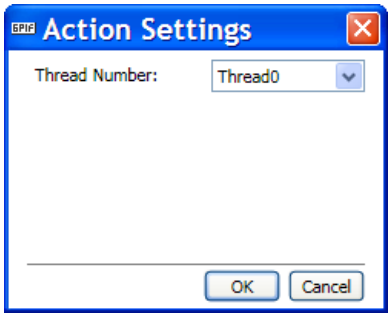
5.3.8 INTR_CPU

オンチップ CPU を割り込み、ファームウェア アプリケーションによって処理される CYU3P_GPIF_EVT_SM_INTERRUPT イベントを生成します。INTR_CPU アクションの状態を選択する場合、INTR_PENDING イベントは遷移方程式のリストで使用可能になります。INTR_PENDING 信号がクリアされる前に、アクションを実行する必要がある場合、CyU3PGpifRegisterSMIntrCallback 関数を使用してください。このコールバックは、割込みが CPU によってクリアされる前に呼び出されます。

5.3.9 COMMIT

選択した Ingress の DMA チャネルでデータ パケット／バッファを転送します（DMA チャネルは FX3 デバイスにデータを読み込むためのものです）。バッファはパイプの反対側に転送されます。このアクションは一般的にステートマシンを用いてバッファ／パケットを終了させるのに使用されます。

図 20. COMMIT アクションの設定



以下のパラメータはこのアクションに関連付けられています。

- *Thread Number（スレッド番号）* – スレッド 0～3（アドレス ビットの数が 0 に設定、またはマスター モードが有効の場合にのみ使用可能です）。

5.4 GPIF のイベント

GPIF II ステート マシンの遷移を引き起こすトリガは、トリガ変数を使用して形成されたブール演算式です。次の表は、マスター ステート マシンの実装で使用されている GPIF II アクションの結果として生成されたイベントを取り込みます。

表 3. GPIF II イベントの説明

GPIF II のイベント	イベントを引き起こすアクション	説明
入力信号名	なし;これは内部イベント	入力として設定された GPIO に関連する名称は遷移方程式でトリガとして使用されます。
LOGIC_ONE	なし	無条件に次の状態に移動します。
DATA_CNT_HIT	COUNT_DATA	アクション設定のカウンタ制限値に達すると、このトリガは真となります。COUNT_DATA アクションの結果としてこのトリガは生成されます。
ADDR_CNT_HIT	COUNT_ADDR	アクション設定のカウンタ制限値に達すると、このトリガは真となります。COUNT_ADDR アクションの結果としてこのトリガは生成されます。
DMA_RDY_CT、DMA_RDY_TH0、DMA_RDY_TH1	IN_DATA DR_DATA	DMAがデータを送信または受信する準備ができていると、このトリガは真となります。

6 GPIF II マスターステートマシンの実装

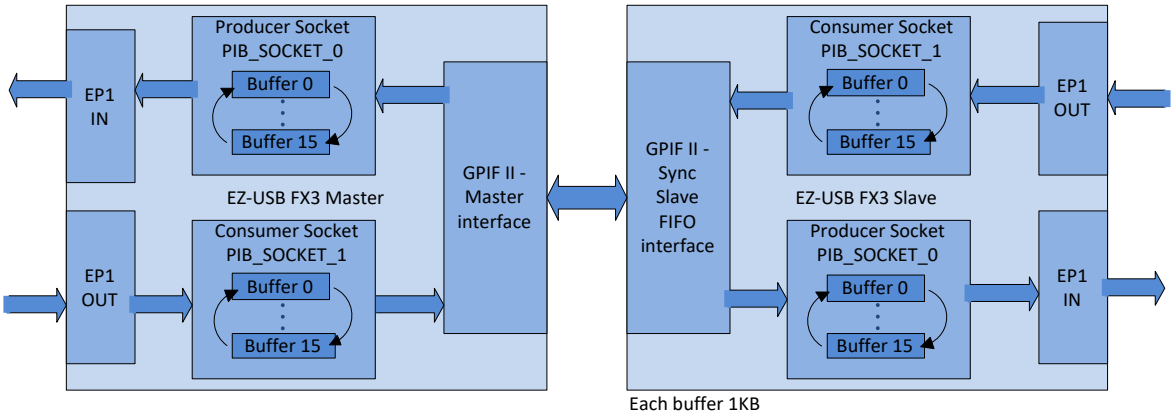
この節では、上記説明したアクションとイベントを使用して、マスター ステート マシンを実装し、第 2 の FX3 の DVK によって実装された同期スレーブ FIFO からデータを送信または検索します。

マスター ステートマシンは、スレーブ FX3 デバイスからのフラグに基づいて、読み出しまたは書き込み処理を決定します。

スレーブ側では、FLAGA はスレッド 1(Thread_1_ DMA_Ready)の状態を、FLAGB はスレッド 0(Thread_0_ DMA_Ready)の状態を示すために設定されます。したがって、FLAGA はスレーブ側でデータの可用性を、FLAGB はスレーブ側でフリー バッファの可用性を示します。

2 つの DMA チャネルは、双方向のデータ転送を行うために、マスター ファームウェアで作成されます。データを USB の側から FX3 の GPIF II に転送するために、PIB ソケット 1 はコンシューマ ソケットとして、USB ソケットはプロデューサとして設定されます。データを GPIF II の側から USB に転送するために、PIB ソケット 0 はプロデューサ ソケットとして、USB ソケットはコンシューマとして設定されます(図 21 を参照)。

図 21. FX3 スレーブとマスターの PIB ソケット



マスターFX3 は、ソケット／レジスタ／アドレス カウンタを利用して、アドレスをアドレス バスをとおして駆動できます。このアプリケーションでは、アドレスはスレーブ FX3 でスレッドを選択する必要があります。実装をより簡単にするために、2 本のアドレス線（A1 と A0）を GPIO として設定します。これらの GPIO を駆動し、スレーブ FX3 のスレッドをアドレス指定します。

クロックはマスターFX3 からスレーブ FX3 に駆動され、データのバス幅は 32 ビットです。スレーブ FIFO インターフェースに必要な制御信号はマスターFX3 の GPIF II から駆動されます。

6.1 FX3 GPIF II マスターの完成したステートマシン

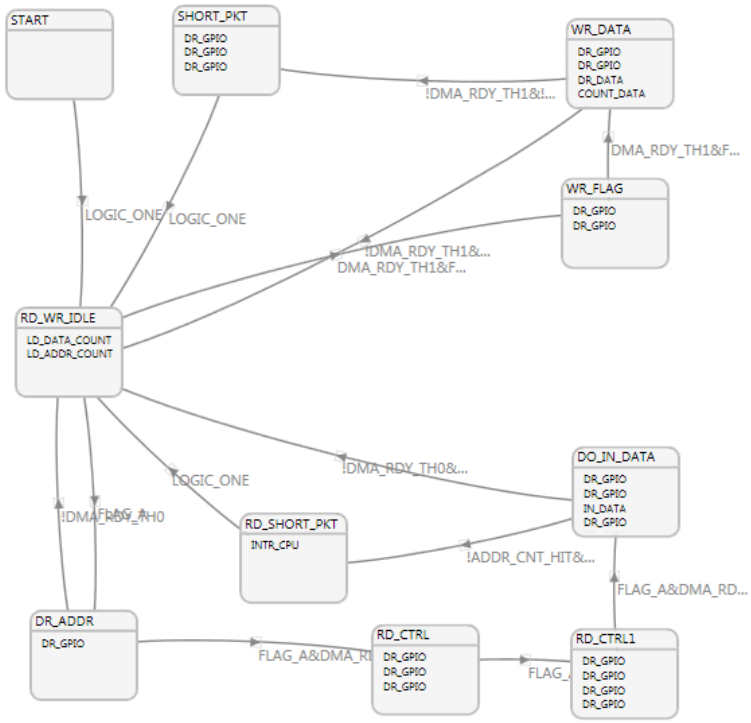
GPIF II マスターのステート マシンは図 22 に示されます。マスターFX3 およびスレーブ FX3 の間でデータの転送がない場合、マスター ステート マシンは RD_WR_IDLE 状態を維持します。データとアドレス カウンタはこの状態でロードされます。添付されているサンプル プロジェクトでは、データ カウンタとアドレスカウンタは 511 でロードされ（パケット サイズの倍数、USB 2.0の場合は 512、USB 3.0 の場合は 1024）、バッファからまたはバッファに書き込まれたデータをカウントします。

遷移方程式が図 22、図 23 および図 24 からクリアされない場合は、添付されている GPIF プロジェクトを参照してください。

この状態（RD_WR_IDLE）は、ステート マシンの読み出しまたは書き込み部分に分岐できるように、FLAGA または（DMA_RDY_TH1&FLAG_B）イベントが真となるかどうか連続して確認します。スレーブ FX3 のコンシューマ ソケット 1 の DMA バッファに幾つかのデータがある場合は、FLAGA は HIGH になります。したがって、FLAGA はスレーブに有効なデータがあり、マスターが読み出し処理を開始する必要があることを示します。読み出し処理に必要な制御信号を駆動する前に、マスターはフリー バッファの有無を確認しなければなりません。DMA_RDY_TH0 は、バッファの状態を決定するために使用されます。

書き込み処理を行うために、マスターFX3 でコンシューマ ソケット 1 の DMA バッファに有効なデータがあるはずです。この条件について調べるために、DMA_RDY_TH1 を使用してください。さらに、スレーブ側でフリー バッファがあるはずです。この条件は FLAGB で検証されます。

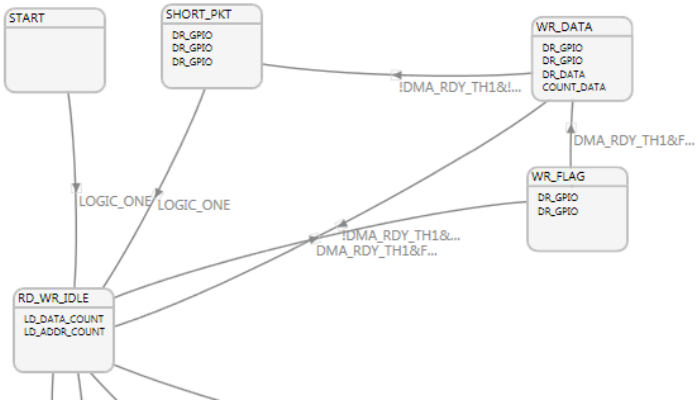
図 22. GPIF II マスター ステート マシン



6.2 FX3 マスターから FX3 スレーブへの書き込み

ステート マシンの FX3 マスターがスレーブ FX3 にデータを書き込む部分は図 23 に示されます。スレーブ FIFO にデータを書き込むためにマスターFX3 から駆動される必要がある信号については、図 6 をご覧ください。

図 23. マスター ステート マシンの書き込み部分



DMA バッファのマスター側で幾つかのデータがあり、スレーブ側でフリー バッファがある場合は、マスター ステート マシンは RD_WR_IDLE 状態から WR_FLAG 状態に移動します。DMA_RDY_TH1 は、マスター側でデータの可用性を、FLAGB はスレーブ側でフリー バッファの可用性を示します。この状態では、マスターは制御信号 SLCS#と SLWR#を WR_DATA 状態に移動する前に駆動します。この状態では、マスターはデータおよび制御信号 SLCS#と SLWR#を駆動します。

チャネルを設定する場合、複数のバッファを特定の DMA チャネルに割り当てられます（ファームウェアで CyU3PDmaChannelCreate API を使用する場合、API の詳細については FX3 の API ガイドを参照）。FLAG がバッファごとに一杯／空の状態を示すことに注意してください。

すべてのバッファは、最大サイズは 64KB-16 です。例えば、512 バイトの 2 つのバッファが DMA チャネルに割り当てられた場合、一杯 FLAG は、512 バイトが最初のバッファに書き込まれた時に一杯の状態を示します。FLAG は、DMA チャネルが 2 番目のバッファに切り替わるまで、一杯の状態を示したままです。DMA チャネルが次のバッファに切り替わるのに要する時間は、普通は数マイクロ秒ですが、予測できません。

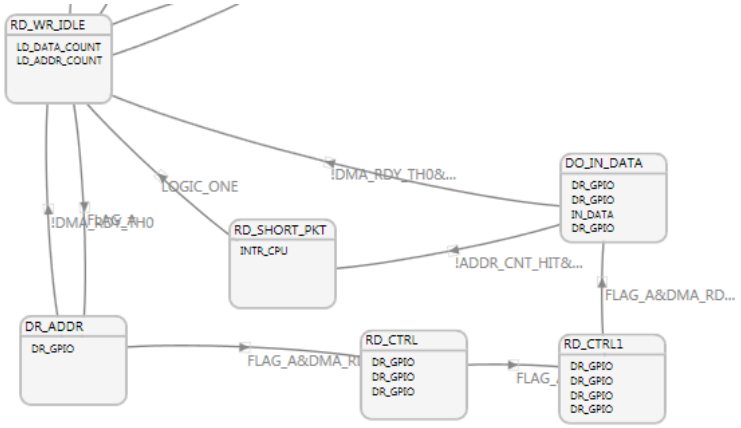
マスター ステート マシンは、RD_WR_IDLE 状態に移動して、このバッファ切り替えの遅延時間中にアドレス カウンタをリロードします。ADDR_CNT_HIT イベントは、このバッファ切り替えの条件を調べるために使用されます。DMA_RDY_TH1 のフラグを更新するのに 1 クロック サイクルの内部遅延時間が必要であるため、このフラグはこの目的に使用できません。次のバッファが使用可能になると、ステート マシンは WR_DATA 状態に戻ります。

DMA バッファにデータがなくなり、ADDR_CNT_HIT イベントが生成されない場合、ショート パケットは識別されます。PKTEND#および SLWR#信号はショート パケットと共に駆動されます。

6.3 FX3 マスターが FX3 スレーブから読み出し

ステート マシンの FX3 マスターがスレーブ FX3 からデータを読み出す部分は 図 24 に示されます。スレーブ FIFO にデータを読み出すためにマスターFX3 から駆動される必要がある信号については、図 5 をご覧ください。

図 24. マスター ステート マシンの読み出し部分



DMA バッファのスレーブ側で幾つかのデータがある場合、マスター ステート マシンは RD_WR_IDLE 状態から DR_ADDR 状態に移動します。この状態では、マスターはアドレス線を駆動します。スレーブ FX3 のスレード 1 をアドレス指定するために、A0 は HIGH に駆動されます。その後、フリー バッファの有無を確認し、DMA バッファの可用性に基づいて RD_CTRL 状態に移動します。この状態では、マスターはアドレスおよび制御信号 SLCS#と SLOE#を駆動します。そして、次の状態に移動し、既に駆動された制御信号に加えて、SLRD#信号を駆動します。その次は、DO_IN_DATA 状態に移動し、必要な制御信号を駆動することによってデータを継続的に読み出します。

ステート マシンの書き込み部分で説明したように、バッファ切り替えの遅延時間があります。この遅延時間中、RD_WR_IDLE 状態に移動して、アドレス カウンタをリロードし、次のバッファがレディになると、DO_IN_DATA 状態に戻ります。

スレーブ FX3 の DMA バッファにデータがなくなり、ADDR_CNT_HIT イベントが生成されない場合、ショート パケットが識別されます。その後、ステート マシンは RD_SHORT_PKT 状態に移動し、INT_CPU アクションを読み出します。これは、DMA チャネルをラップアップしている FX3 ファームウェアに割込みを生成します。添付されているファームウェアで CyFxApplnGPIFEventCB()を探してください。

注: INT_CPU アクションの代わりに COMMIT アクションを使用することもできます。ただし、COMMIT アクションに加えて、IN_DATA アクションを使用する必要があります。そうでない場合は、長さゼロのパケット(ZLP)が発生します。そのため、INT_CPU アクションを COMMIT および IN_DATA アクションで置き換える場合、LD_DATA_COUNTER アクションの設定で 1 カウントを減らすように注意してください。

注: レジスタを介して読み出しおよび書き込み処理を実行する際は、OUT_REG_VALID と IN_REG_VALID イベントを使用してください。(IN_DATA または DR_DATA アクションで「データ ソース」を「レジスタ」として選択します)。

周辺機器の特定のアドレス位置にデータを書き込もうとすると、そのアドレスおよびデータをファームウェアの出力レジスタ(FX3 デバイスからデータを送信するためのレジスタ)に書き込みます。これにより、ステート マシンでこれらのイベントを監視し、有効なデータがあるかどうかを確認できます。

しかし、DMA バッファを用いて多くのデータを転送する場合、DMA レディ フラグを確認する必要があります。DMA レディ フラグは、バッファに有効なデータがあることを通知します。

7 FX3 マスター ファームウェアの実装

GPIF-II ステートマシンとは別に、FX3 ファームウェアは、DMA チャネルの作成、データ転送の開始、データの値とアドレスカウンタの設定を行います。 データを転送するために、USB ソケットと GPIF ソケットの間に 2 つの AUTO DMA チャネルが作成されます。 各 DMA チャネルは、USB 3.0 ポートに接続されている場合、それぞれ 2048 バイトの 16 個のバッファで構成されています(USB 2.0 ポートに接続されている場合は 512 バイト)。CyFxApplnSetPibDllParameters()API は、マスターモードで動作しているときに正しいクロック位相を設定するために使用されます。

また、データカウントレジスタは、USB 3.0 ポートに接続されている場合は 511(2047 バイト)に、USB 2.0 ポートに接続されている場合は 127(512 バイト)に初期化されます。

GPIF-II ステートマシンから INTR_CPU アクションをとおして割込みが発生すると、CyFxAppInGPIFEventCB()API の USB に転送するために、現在のバッファをラップアップしてコミットします。

8 ハードウェアの接続

GPIF II マスターの設計は、同期スレーブ インターフェースとしてプログラムされた第 2 の FX3 DVK 基板を使用してテストされます。2 つの FX3 チップ間のインターフェース接続は図 25 に示されます。2 台の FX3 SuperSpeed Explorer キット(CYUSB3KIT-003)を使用した FX3 のバックツーバック設定を図 26 に示します。2 つの SuperSpeed Explorer ボードが接続されているため、それらの GPIF インターフェースも接続されています。インターポージング コネクタ(Samtec 部品番号 SSW-120-03-G-D)には 2 つの機能があります。USB 3.0 コネクタが干渉しないように物理的に 2 つのボードを離して置いてください。インターポージング コネクタの一部のピンは、電圧ピンと低速周辺デバイスピンが 2 つのボード間で短絡されないように、インターポージング コネクタから切り離す必要があります。図 27 は、FX3 ボードにプラグを差し込む前に、インターポージング コネクタ上のどのピンを切り離すべきかを示しています。

注: CYUSB3KIT-001 を使用している場合は、ハードウェアのセットアップ情報については付録: FX3 Development Kit (CYUSB3KIT-001)を参照してください。

図 25. ハードウェア接続

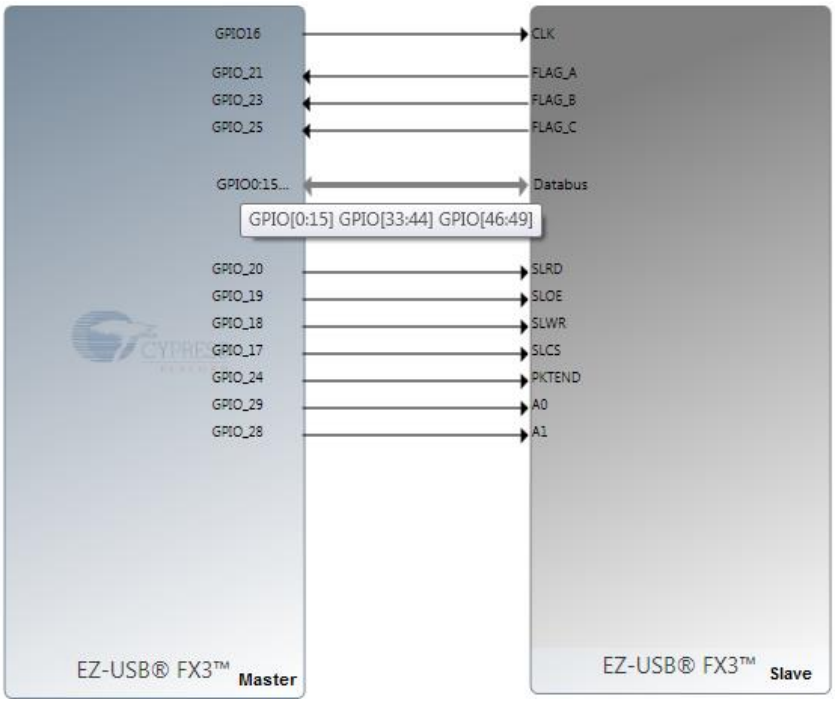


図 26. CYUSB3KIT-003 を使った FX3 バックツーバック設定

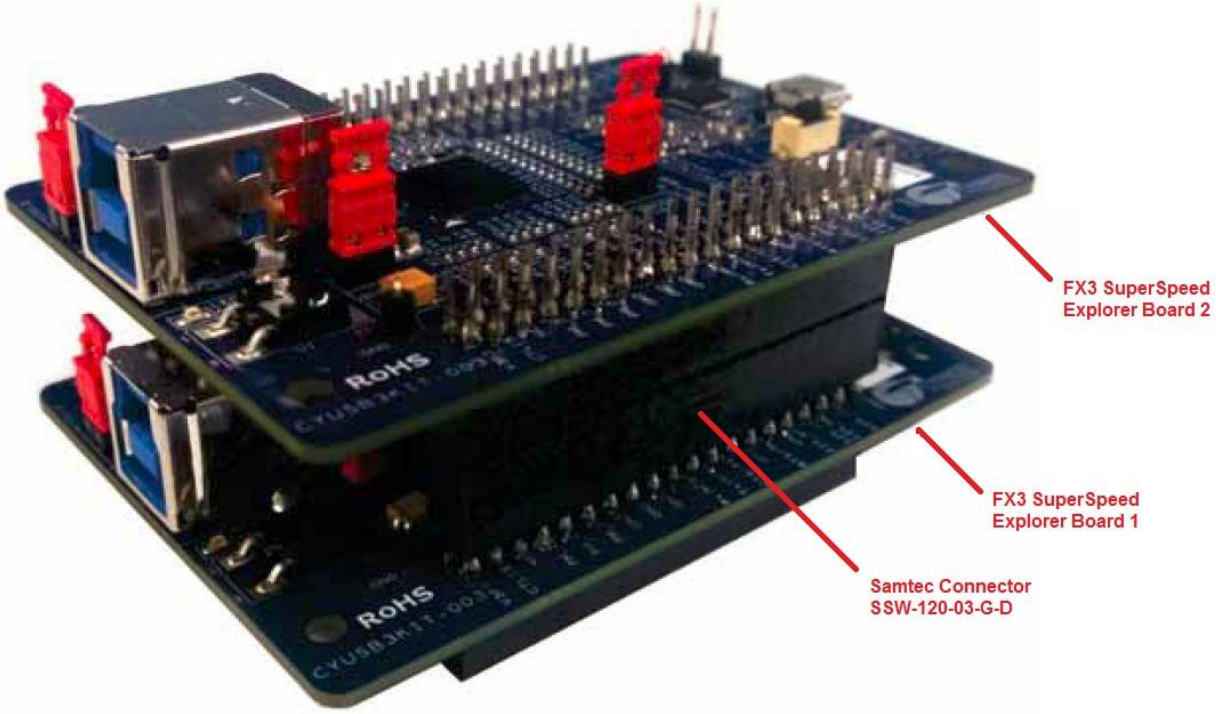
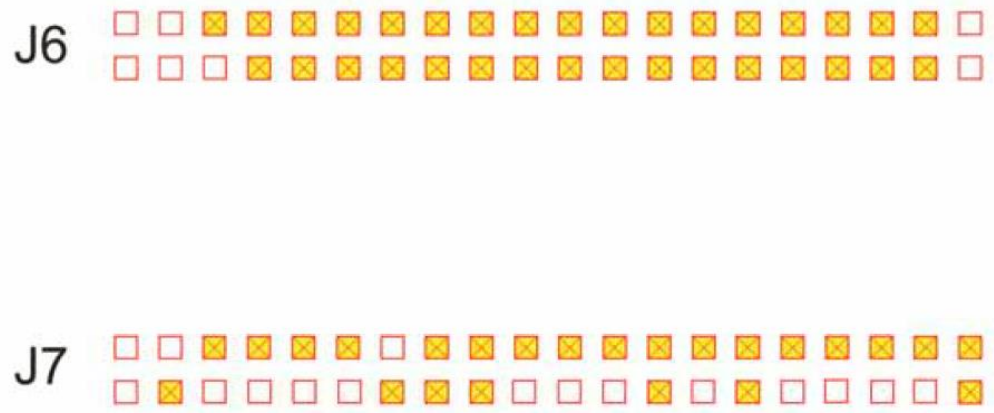


図 27. 白いピンはインターポージング コネクタの切り離し部



9 デモを実行する手順

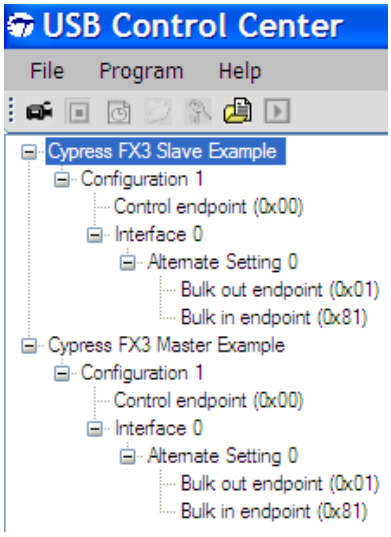
スレーブとマスターFX3 のファームウェア ソースコードおよび GPIF II ステートマシンはこのアプリケーション ノートに添付されています。この節では、本アプリケーション ノートに添付されているファイルを使用してデモを実行する手順について説明します。

1. 相互接続用の基板を用いて 2 つの FX3 DVK を接続します。ハードウェア接続の節で図 26 を参照してください。
2. USB 3.0 ケーブルを使用して 2 つの FX3 DVK を PC に接続します。そして、これらの両方のデバイスをサイプレスのブートローダ デバイスとして並べています。
3. USB コントロール センターを利用して、ファームウェア イメージを 2 つの FX3 デバイスの RAM にダウンロードします。

注: マスター デバイス用のファームウェア イメージ（添付ファイル内のビン フォルダの Automaster.img）をダウンロードする前に、スレーブ デバイス用に作成済みのファームウェア イメージ（添付ファイル内のビン フォルダの Autoslave.img）をダウンロードします。

4. 図 28 に示すように、スレーブおよびマスター デバイスが列挙された様子を見られます。スレーブ FX3 デバイスは 0x4B4の VID および 0x00F2 の PID を使用します。マスターFX3 は 0x4B4 の VID および 0x00F4 の PID を使用します。

図 28. マスターおよびスレーブ FX3 デバイス



5. 各デバイスには、バルク OUT エンドポイントおよびバルク IN エンドポイントがあります。スレーブ FX3 のバルク OUT エンドポイントにデータを転送して、マスターFX3 のバルク IN エンドポイントから同じデータを読み戻します。Transfer File-OUT（ファイルを転送 - OUT）ボタンにクリックし、添付ファイル内のビン フォルダにある 8192_count.hex を開きます。この手順によって、図 29 に示すとおりに、8KB のデータをスレーブ FX3 のバルク OUT エンドポイントに転送します。図 30 には、マスターFX3 のバルク IN エンドポイントから読み出されたデータを示します。

図 29. スレーブ FX3 のバルク OUT エンドポイントにデータを転送

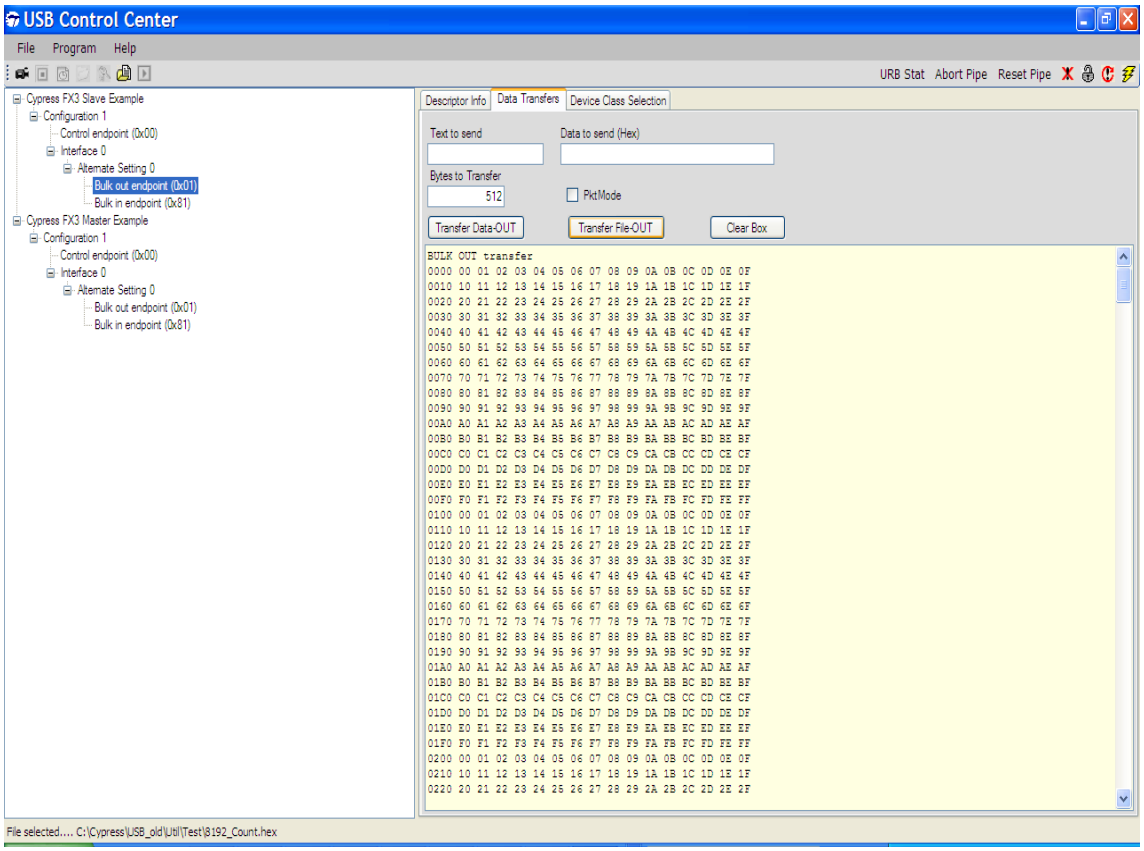
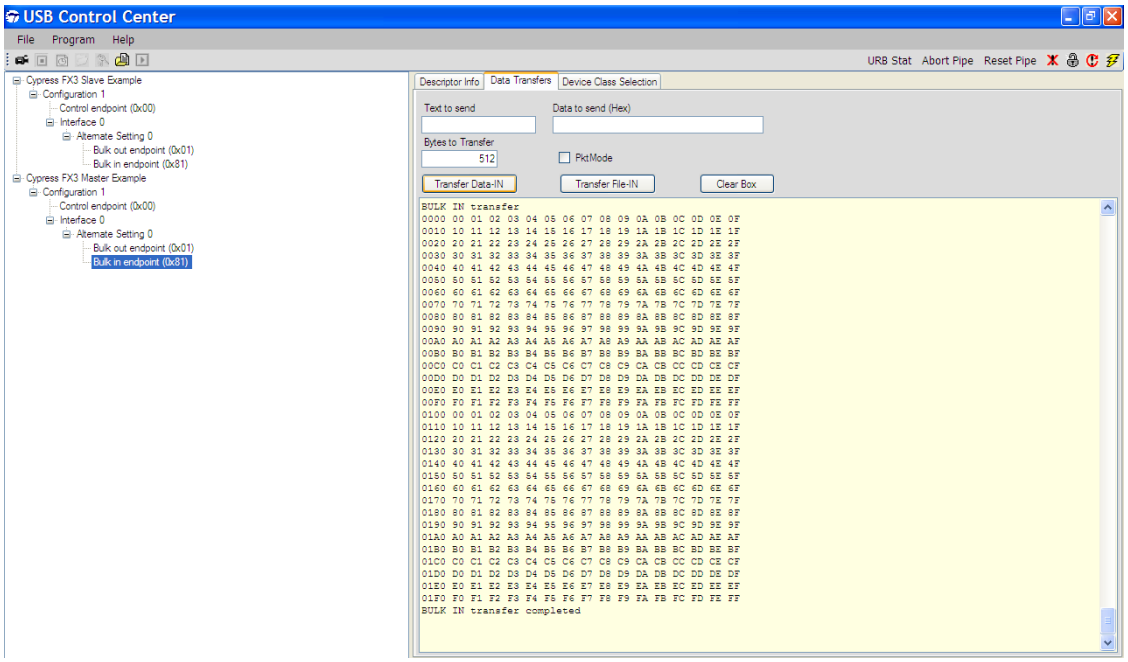


図 30. マスターFX3 のバルク IN エンドポイントからデータを読み出し



6. また、幾つかのショート パケットを一方から他方に転送できます。図 31 には、マスターFX3 のバルク OUT エンドポイントへの 18 バイトの転送を示します。図 32 には、スレーブ FX3 のバルク IN エンドポイントから読み出されたデータを示します。32 ビットのデータ バスにより、0 の余分の 2 バイトを観察できます。

図 31. マスターFX3 のバルク OUT エンドポイントへのショートパケット転送

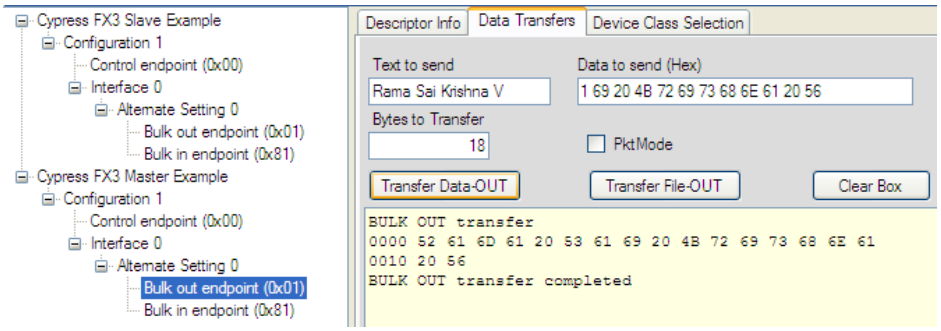
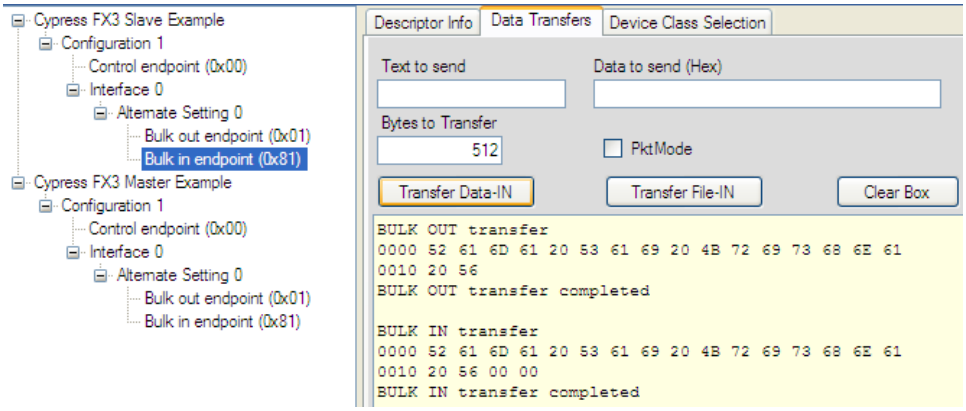


図 32. スレーブ FX3 のバルク IN エンドポイントからショートパケットを読み出し



7. 図 33 には、マスターFX3 のバルク OUT エンドポイントへの 8 バイトの転送を示します。図 34 には、スレーブ FX3 のバルク IN エンドポイントから読み出されたデータを示します。

図 33. マスターFX3 のバルク OUT エンドポイントにショートパケットを転送

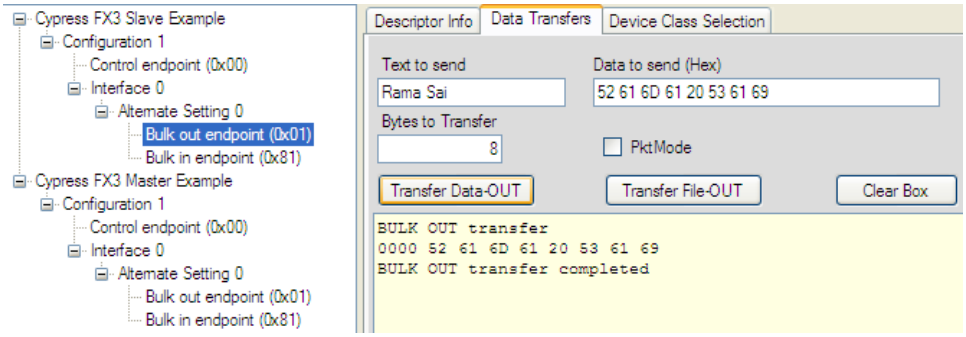
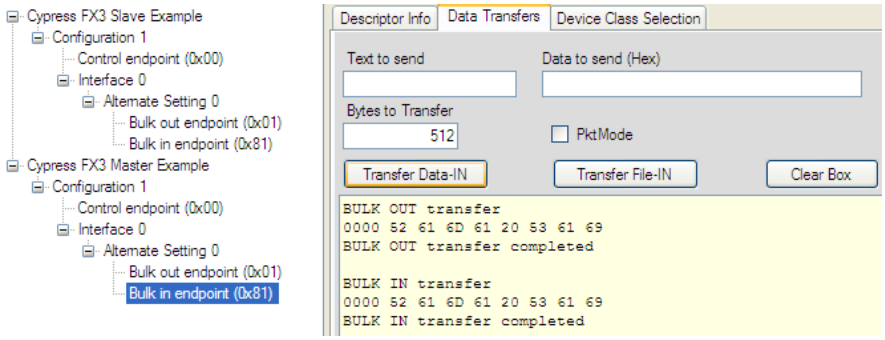


図 34. スレーブ FX3 のバルク IN エンドポイントからショートパケットを読み出し



10 関連プロジェクト ファイル

表 4 は、このアプリケーションノートに添付されているファイルについての説明です。

表 4. 本アプリケーションノートの添付ファイルの説明

ファイル/フォルダ	説明
FX3 ファームウェア	このフォルダには以下のフォルダが含まれています。 *AutoSlave* – スレーブ FX3 用の FX3 ファームウェア ソースファイル *AutoMaster* – マスターFX3 用の FX3 ファームウェア ソースファイル
GPIF II のプロジェクト	このフォルダには以下のフォルダが含まれています。 *master_read_write_sync.cydsn* – マスターFX3 用の GPIF II プロジェクト *sync_slave_fifo_2bit_editable_latestGPIF.cydsn* – スレーブ FX3 用の GPIF II プロジェクト
ビン	このフォルダには以下のファイルが含まれています。 *AutoSlave.img* – スレーブ FX3 用の FX3 ファームウェア イメージのファイル *AutoMaster.img* – マスターFX3 用の FX3 ファームウェア イメージのファイルこのアプリケーションのクイック デモを見るために、これらのイメージ ファイルを FX3 DVK にダウンロードします。 *8192_count.hex* – 8192 バイトのインクリメンタル データのファイルこれは FX3 の OUT エンドポイントで 8K バイトのファイル転送をするために使用されます。

11 まとめ

このアプリケーション ノートでは、FX3 の GPIF II Designer ツールを使用して、同期スレーブ FIFO インターフェースに対応するマスター インターフェースを実装するために必要な詳細情報について説明しました。アプリケーションの要件に応じて、この設計を GPIF II マスターのステート マシンを開発するための開始点として使用できます。

12 関連アプリケーションノート

AN65974 – EZ-USB® FX3 スレーブ FIFO インターフェースを使った設計

著者について

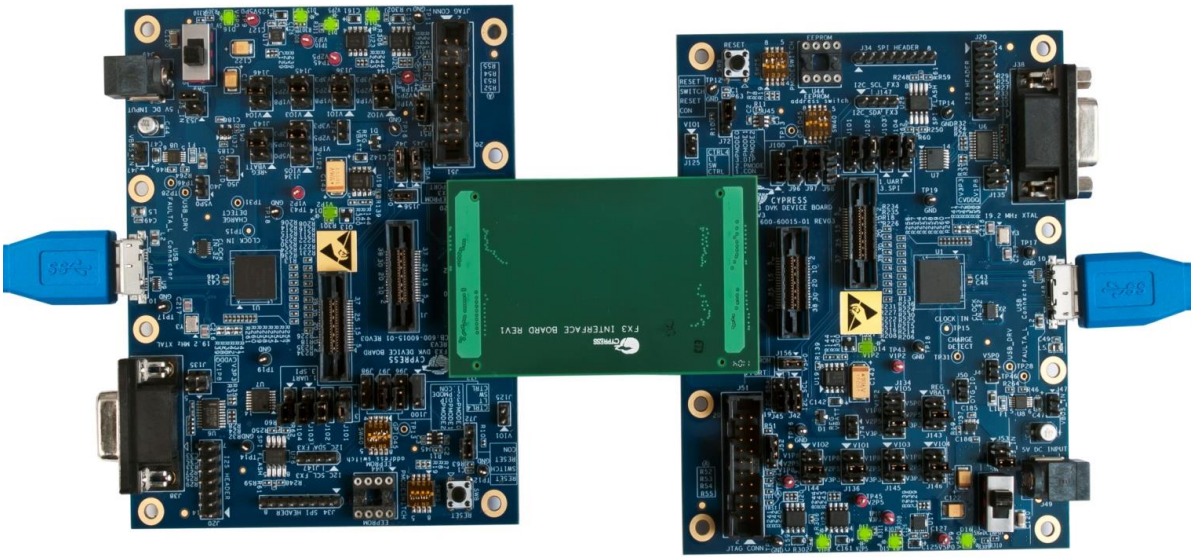
氏名: Rama Sai Krishna

役職: 上級アプリケーションエンジニア

付録: FX3 Development Kit (CYUSB3KIT-001)を使ったハードウェアの設定

図 35 は、バックツーバックで接続された 2 つの FX3 開発キット（CYUSB3KIT-001）を使用したハードウェア設定を示しています。

図 35. Rev3 DVKs を使用した FX3 バックツーバック設定



注: 両方の FX3 DVK のポスト 1 と 2 の間にジャンパ J100 が接続されていることを確認してください。これは CTRL [4]（GPIO [21]）を FLAGA として設定します。2 つの FX3 DVK 間の相互接続ボードの回路図を入手するには、サイプレスにお問い合わせください。

改訂履歴

文書名: AN87216 – GPIF™ II マスター インターフェースの設計

文書番号: 001-92083

版	ECN	発行日	変更内容
**	4339867	2014-04-10	これは英語版 001-87216 Rev *A を翻訳した日本語版 Rev. **です。
*A	5710937	2017-04-25	これは英語版 001-87216 Rev *C を翻訳した日本語版 Rev. *A です。
*B	6823338	2020-03-04	これは英語版 001-87216 Rev *D を翻訳した日本語版 Rev. *B です。
*C	6981032	2020-10-12	これは英語版 001-87216 Rev *E を翻訳した日本語版 Rev. *C です。

ワールドワイドな販売と設計サポート

サイプレスは、事業所、ソリューション センター、メーカー代理店および販売代理店の世界的なネットワークを保持しています。お客様の最寄りのオフィスについては、サイプレスのロケーション ページをご覧ください。

製品

Arm® Cortex® Microcontrollers	cypress.com/arm
車載用	cypress.com/automotive
クロック&バッファ	cypress.com/clocks
インターフェース	cypress.com/interface
IoT (モノのインターネット)	cypress.com/iot
メモリ	cypress.com/memory
マイクロコントローラ	cypress.com/mcu
PSoC	cypress.com/psoc
電源用 IC	cypress.com/pmic
タッチセンシング	cypress.com/touch
USB コントローラー	cypress.com/usb
ワイヤレス	cypress.com/wireless

PSoC®ソリューション

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

サイプレス開発者コミュニティ

コミュニティ | サンプルコード | Projects | ビデオ | ブログ | トレーニング | Components

テクニカル サポート

cypress.com/support

本書で言及するその他すべての商標または登録商標は、それぞれの所有者に帰属します。

Cypress Semiconductor
An Infineon Technologies Company
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2013-2020. 本書面は, Cypress Semiconductor Corporation 及び Spansion LLC を含むその子会社 (以下「Cypress」という。) に帰属する財産である。本書面 (本書面に含まれ又は言及されているあらゆるソフトウェア若しくはファームウェア (以下「本ソフトウェア」という。) を含む) は, アメリカ合衆国及び世界のその他の国における知的財産法令及び条約に基づき Cypress が所有する。Cypress はこれらの法令及び条約に基づく全ての権利を留保し, 本段落で特に記載されているものを除き, その特許権, 著作権, 商標権又はその他の知的財産権のライセンスを一切許諾しない。本ソフトウェアにライセンス契約書が伴っておらず, かつ Cypress との間で別途本ソフトウェアの使用方法を定める書面による合意がない場合, Cypress は, (1) 本ソフトウェアの著作権に基づき, (a) ソースコード形式で提供されている本ソフトウェアについて, Cypress ハードウェア製品と共に用いるためにのみ, かつ組織内部でのみ, 本ソフトウェアの修正及び複製を行うこと, 並びに (b) Cypress のハードウェア製品ユニットに用いるためにのみ, (直接又は再販売者及び販売代理店を介して間接のいずれかで) 本ソフトウェアをバイナリーコード形式で外部エンドユーザーに配布すること, 並びに (2) 本ソフトウェア (Cypress により提供され, 修正がなされていないもの) が抵触する Cypress の特許権のクレームに基づき, Cypress ハードウェア製品と共に用いるためにのみ, 本ソフトウェアの作成, 利用, 配布及び輸入を行うことについての非独占的で譲渡不能な一身専属的ライセンス (サブライセンスの権利を除く) を付与する。本ソフトウェアのその他の使用, 複製, 修正, 変換又はコンパイルを禁止する。

適用される法律により許される範囲内で, Cypress は, 本書面又はいかなる本ソフトウェア若しくはこれに伴うハードウェアに関しても, 明示又は黙示をとわず, いかなる保証 (商品性及び特定の目的への適合性の黙示の保証を含むがこれらに限られない) も行わない。いかなるコンピューティングデバイスも絶対に安全ということはない。従って, Cypress のハードウェアまたはソフトウェア製品に講じられたセキュリティ対策にもかかわらず, Cypress は, Cypress 製品への権限のないアクセスまたは使用といったセキュリティ違反から生じる一切の責任を負わない。加えて, 本書面に記載された製品には, エラッタと呼ばれる設計上の欠陥またはエラーが含まれている可能性があり, 公表された仕様とは異なる動作をする場合がある。適用される法律により許される範囲内で, Cypress は, 別途通知することなく, 本書面を変更する権利を留保する。Cypress は, 本書面に記載のある, いかなる製品若しくは回路の適用又は使用から生じる一切の責任を負わない。本書面で提供されたあらゆる情報 (あらゆるサンプルデザイン情報又はプログラムコードを含む) は, 参照目的のためのみに提供されたものである。この情報で構成するあらゆるアプリケーション及びその結果としてのあらゆる製品の機能性及び安全性を適切に設計, プログラム, かつテストすることは, 本書面のユーザーの責任において行われるものとする。Cypress 製品は, 兵器, 兵器システム, 原子力施設, 生命維持装置若しくは生命維持システム, 蘇生用の設備及び外科的移植を含むその他の医療機器若しくは医療システム, 汚染管理若しくは有害物質管理の運用のために設計され若しくは意図されたシステムの重要な構成部分としての使用, 又は装置若しくはシステムの不具合が人身傷害, 死亡若しくは物的損害を生じさせるようなその他の使用 (以下「本目的外使用」という。) のためには設計, 意図又は承認されていない。重要な構成部分とは, それの不具合が装置若しくはシステムの不具合を生じさせるか又はその安全性若しくは実効性に影響すると合理的に予想できるような装置若しくはシステムのあらゆる構成部分をいう。Cypress 製品のあらゆる本目的外使用から生じ, 若しくは本目的外使用に関連するいかなる請求, 損害又はその他の責任についても, Cypress はその全部又は一部をとわず一切の責任を負わず, かつ Cypress はそれら一切から本書により免除される。Cypress は Cypress 製品の本目的外使用から生じ又は本目的外使用に関連するあらゆる請求, 費用, 損害及びその他の責任 (人身傷害又は死亡に基づく請求を含む) から免責補償される。

Cypress, Cypress のロゴ, Spansion, Spansion のロゴ及びこれらの組み合わせ, WICED, PSoC, CapSense, EZ-USB, F-RAM, 及び Traveo は, 米国及びその他の国における Cypress の商標又は登録商標である。Cypress のより完全な商標のリストは, cypress.com を参照すること。その他の名称及びブランドは, それぞれの権利者の財産として権利主張がなされている可能性がある。