サイプレスはインフィニオン テクノロジーズになりました

この表紙に続く文書には「サイプレス」と表記されていますが、これは同社が最初にこの製品を開発したからです。新規および既存のお客様いずれに対しても、引き続きインフィニオンがラインアップの一部として当該製品をご提供いたします。

文書の内容の継続性

下記製品がインフィニオンの製品ラインアップの一部として提供されたとしても、それを理由としてこの文書に変更が加わることはありません。今後も適宜改訂は行いますが、変更があった場合は文書の履歴ページでお知らせします。

注文時の部品番号の継続性

インフィニオンは既存の部品番号を引き続きサポートします。ご注文の際は、データシート記載の注文部品番号をこれまで通りご利用下さい。

AN44168

外部マイクロコントローラーを使用した PSoC® 1 デバイス プログラミング (HSSP)

関連プロジェクト: あり

関連製品ファミリ: CY8C21x23、CY8C21x34、CY21x45、CY8C22x45、CY8C23x33、CY8C24x23、CY8C24x94、CY8C27x43、CY8C28xxx、CY8C29x66、CY8CTST1xx、CY8CTMG1xx、CY8CTMA120

ソフトウェア バージョン: PSoC Designer™ 5.4 SP1

関連文書: 完全なリストについては、ここをクリックしてください。

AN44168 はモジュール化された C コードを使用して外部マイクロコントローラーで PSoC® 1 のデバイスをプログラムする方法について説明しています。ホスト ソース シリアル プログラミング (HSSP) と呼ばれるこのプロセスでは、ホスト マイクロコントローラーがインシステム シリアル プログラミング (ISSP) インターフェースを介して PSoC 1 をプログラムします。C コードは、最小限の変更で任意のマイクロコントローラにも移植できるように書かれていますので、PSoC4 用の HSSP アプリケーション開発期間を短縮します。このコードは PSoC 1 ISSP プログラミング仕様書で説明されているプログラミング手順に基づいてビルドされます。

目次

1 はじめに 1
2 HSSP ファームウェアのアーキテクチャ 2
2.1 ISSP プロトコルの物理層 2
2.2 ISSP プロトコルのパケット層 2
2.3 HSSP プログラミング ステップ 3
3 PSoC 1 HSSP プログラミングのハードウェア接続 4
4 HSSP アプリケーションのホスト プログラマへの移植 4
4.1 移植すべきファイル 4
4.2 移植時に必要なコード変更 5
4.3 特性の選択 5
4.4 低レベル ドライバーの変更 6
4.5 RAM バッファへのデータ書き込み 7
5 フラッシュ ブロックのシーケンスまたは数量の変更 7
6 内蔵テスト ポイントによるフラッシュ書き込み時間の検証 7
7 UART デバッグ処理のメッセージ 9
8 制約 9
9 要約 10
10 関連文書: 10
A 付録 A: ポート ビット操作関数 11
改訂履歴 12
ワールドワイドな販売と設計サポート 13
製品 13
PSoC®ソリューション 13
サイプレス開発者コミュニティ 13
テクニカル サポート 13

1 はじめに

サイプレスのプログラマブル システムオンチップ (PSoC) は簡単に使用でき、柔軟性があり、費用効果の高い、再プログラム可能なアナログとデジタル リソースの組み合わせから成ります。これらの機能は、基板搭載のホスト プロセッサによって PSoC を連続的にプログラムするなど、創造的な設計の機会を数多く提供します。この方法は、現場でファームウェアをインストールまたは更新したり、別の機能のために PSoC を再プログラミングするために使われます。

HSSP ソース コードは、システム設計者に各自のシリアル プログラミング ソフトウェアを開発するスタート ポイントを提供するために、サイプレスによって作成されました。設計者は各自のホスト プログラマーに適合させるために、コードに最小限の修正を加える必要があります。ソース コードはさまざまな PSoC デバイスを考慮し、高抽象度を提供します。本アプリケーション ノートの考慮対象となるデバイスは「関連製品ファミリ」に記載されています。ISSP の詳細情報については、www.cypress.com/?rID=2907 と www.cypress.com/?rID=2908 で PSoC 1 ISSP プログラミング仕様をご参照ください。

CY8C20xx6、CY8CTMG2xx、および CY8CTST2xx デバイス用の HSSP アプリケーションについてはアプリケーション ノート AN59389 をご参照ください。AN59389 は CY8C20045、CY8C20055、CY8C20065、CY8C20xx6A、CY8C20xx7 などの ISSP プログラミング仕様書で説明したプログラミング手順を実現します。

ブートローダと HSSP の相違点:

組込みアプリケーションでは、ブートローダは標準的な通信インターフェースを通してシステム ファームウェアを更新します。ブートローダのサポートに加えて、PSoC 1 は HSSP アプリケーションを使用したインシステム プログラミングが可能です。HSSP とブートロードとの主な違いは本節で説明します。お客様の要件に最も適切な方法をご選択ください。

- デバイスの部分的なファームウェア更新と完全なプログラミング: HSSP は部分的なファームウェア アップグレードをサポートせず、PSoC 1 のフラッシュ メモリの完全なプログラミングのみをサポートします。一方、ブートローダではフラッシュ メモリの特定の部分 (ブートローダブルな領域) のみを変更できます。フラッシュ メモリの一部 (ブートローダ領域) はブートロード用に予約されており、この部分は更新できません。
- 外部ホストと PSoC 1 間の通信インターフェース: ブートローダは USB、I^2C、UART、 SPI 等の多くの標準通信インターフェースの 1 つを使用して PSoC 1 ファームウェアをアップグレードします。PSoC 1 の HSSP は常に ISSP プロトコルを使用してプログラミングします。

2 HSSPファームウェアのアーキテクチャ

PSoC 1 用の HSSP は、モジュール化された C コードを使用することで複数の層で実装されています。これらの階層は以下の通りです。

1. ISSP プロトコルの物理層
2. ISSP プロトコルのパケット層
3. HSSP プログラミング ステップ層

各階層間の制御フローについては、図 1 をご参照ください。

2.1 ISSPプロトコルの物理層

ISSP プロトコルの物理層を構成するファイルは以下に記載されています。

ソース ファイル	説明
ISSP_PHYSICALLAYER (.C)	このファイルはプログラミング ピン (SDA と SCL) を操作し、かつターゲット デバイスのリセットと電源ピンを制御するマクロと関数を含む。ビット バンギングを使用して、これらのピンで信号を生成。このファイルでの関数はパケット層でのより高いレベルの関数に呼び出される

このファイルでは、ファームウェアは PSoC 1 CY8C29466 がホストのマイクロコントローラーであるとして書かれています。他のホスト マイクロコントローラーの場合、すべての関数とマクロを適切に修正する必要があります。

2.2 ISSPプロトコルのパケット層

ISSP プロトコルのパケット層を構成するファイルは以下に記載されています。

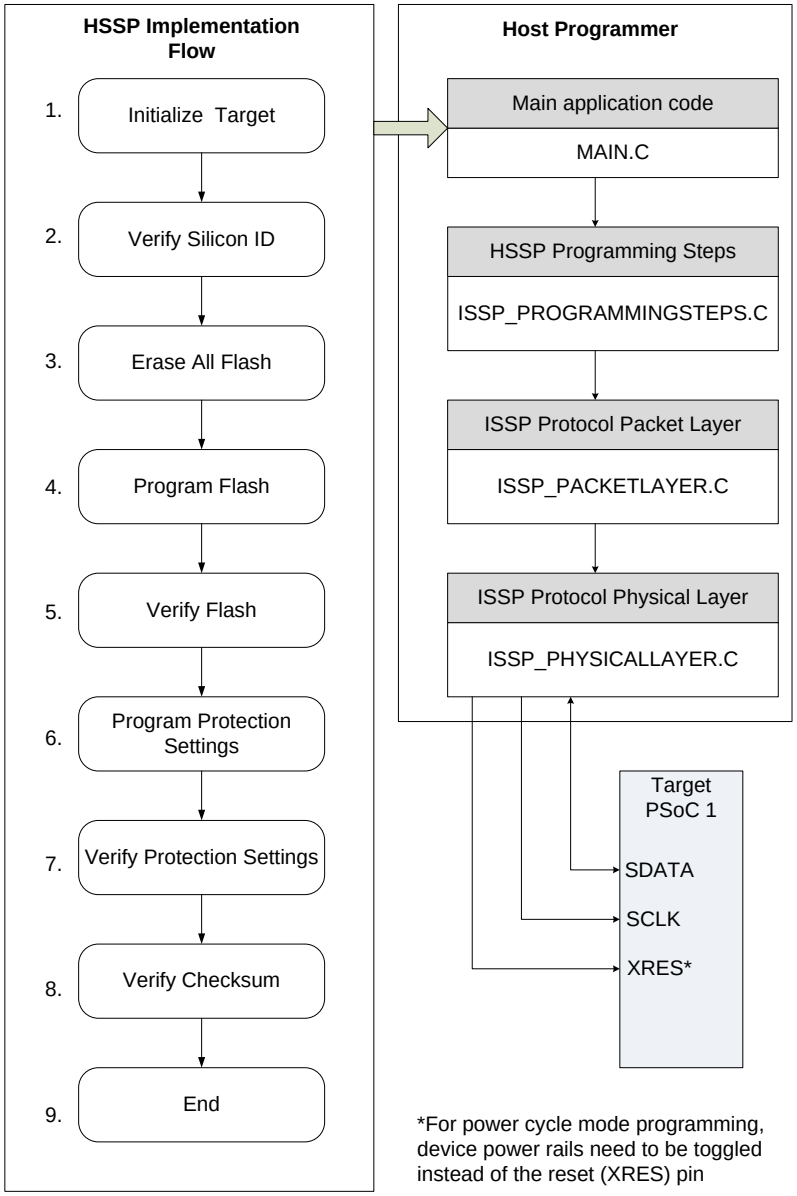
ソース ファイル	説明
ISSP_PACKETLAYER (.C)	このファイルでは、ISSP_PhysicalLayer で定義された関数を使用して、ISSP ベクタを送信。ISSP ベクタは命令セットを表すビットのシーケンスにすぎない。これらの命令はホスト プロセッサにより送信され、ターゲット デバイスで実行される。異なるプログラミング段階用に様々なベクタが定義される。ISSP ベクタの詳細については、PSoC 1 ISSP プログラミング仕様書を参照。このファイルの関数はデータ読み出しとターゲット デバイスへのベクタ パッケージの書き込みを行う。この層内で定義された関数は ISSP_PROGRAMMINGSTEPS.C ファイル内の関数により直接呼び出される。

2.3 HSSPプログラミング ステップ

HSSP プログラミング ステップを構成するファイルは以下に記載されています。

ソース ファイル	説明
ISSP_PROGRAMMINGSTEPS (.C)	このファイルは HSSP アプリケーションのトップ レベルの関数を含んでいる。各ステップの詳細は PSoC 1 ISSP プログラミング仕様書に記載されている

図 1. PSoC 1 用 HSSP ファームウェア アーキテクチャ



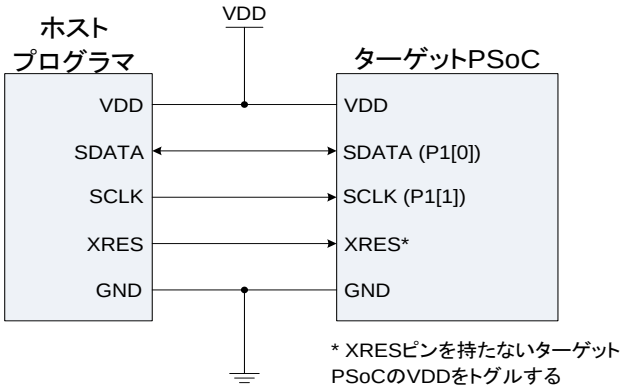
3 PSoC 1 HSSPプログラミングのハードウェア接続

図 2 に、ホスト プログラマとターゲット PSoC1 デバイスの間に必要となる基本的なハードウェア接続を示しています。PSoC デバイスはリセット モードとパワー サイクル モードという 2 つの異なるモードでプログラミングできます。推奨のプログラミング モードであるリセット モードは PSoC デバイスには 1 本の XRES ピンがある場合に使用されます。XRES ピンがないデバイスはパワー サイクル モードでプログラミングする必要があります。この場合、HSSP マイクロコントローラーはデバイスの電源をオン／オフに切り替えます。各プログラミング モードにおいて、ホストは 3 本の I/O ピンを必要とします。リセット モードではシリアル データ (SDATA)、シリアル クロック (SCLK)、外部リセット (XRES) ピンで、パワー サイクル モードでは SDATA、SCLK、PSoC パワー (PWR) 制御です。これらのピンはファームウェアで操作されます。

ホスト プロセッサの SDATA ピンは双方向性でなければなりません。ホストは、このピンの特性を変更できなければなりません (書き込み時はストロング駆動で、読み出し時は High-Z 状態です)。

プログラミング モード、プログラミング インターフェース接続、プログラミング仕様の詳細情報については、www.cypress.com/?rID=2907 と www.cypress.com/?rID=2908 で PSoC 1 ISSP プログラミング仕様をご参照ください。

図 2. 基本的なホスト／ターゲットの接続



4 HSSPアプリケーションのホスト プログラマへの移植

このアプリケーション ノートで提供されているプロジェクトは PSoC 1 CY8C29466 をホスト プログラマとして使用します。HSSP アプリケーションでは、ホスト プログラマは任意の別のマイクロコントローラーにすることができます。本節は、ターゲット デバイスをプログラムするために使用される特定のホストに、この HSSP アプリケーション コードを移植するのに必要な変更を説明しています。

4.1 移植すべきファイル

表 1 にシステムのホスト デバイスに移植すべきファイルを示しています。

表 1. 移植するファイル

移植するヘッダー ファイル	移植するソース ファイル
ISSP_DEFS.H	ISSP_PHYSICALLAYER.C
ISSP_DELAYS.H	ISSP_PACKETLAYER.C
ISSP_DIRECTIVES.H	ISSP_PROGRAMMINGSTEPS.C
ISSP_ERRORS.H	
ISSP_REVISION.H	
ISSP_VECTORS.H	

4.2 移植時に必要なコード変更

表 2 に添付 HSSP アプリケーション コードを PSoC 1 以外のホスト プログラマに移植する際に必要なソース ファイル更新を示しています。

表 2. コード変更

ファイル	説明	必要な変更
ISSP_DEFS.H	このファイルはすべての PSoC 1 デバイス用のバンク数、バンクごとのブロックおよびバンク情報ごとのセキュリティ バイトを含んでいる	変更は不要
ISSP_DELAYS.H	このファイルは待機およびポーリング イベント、XRES アサーション、およびターゲット デバイスのパワーサイクルのタイムー設定を含んでいる	ホスト プロセッサの実行速度に基づいて、遅延カウントを調整
ISSP_DIRECTIVES.H	このファイルはターゲット PSoC 1 デバイスの選択、デバッグ処理コード (UART、テスト ポイント) の有効化／無効化、およびターゲット電圧とプログラミング モード (リセットまたはパワー サイクル) の選択をするオプションを提供	特性の選択の節を参照
ISSP_ERRORS.H	このファイルはプログラミング プロセス中の異なるエラーの#define ステートメントを含んでいる	変更は不要
ISSP_EXTERN.H	このファイルは関数および変数プロトタイプを含んでいる	変更は不要
ISSP_REVISION.H	このファイルはリビジョン制御のために使用される	変更は不要
ISSP_VECTORS.H	このファイルはすべての ISSP 命令のベクタを含んでいる	変更は不要
ISSP_PHYSICALLAYER.C	このファイルはターゲット デバイスと通信する低レベル ドライバーを含んでいる	低レベル ドライバーの変更
ISSP_PACKETLAYER.C	このファイルは様々なプログラミング段階のベクタ パッケージを作成する関数を含んでいる	プログラム データの取得にコード更新が必要。RAM バッファへのデータ書き込みの節を参照
ISSP_PROGRAMMINGSTEPS.C	このファイルはプログラミング プロセスの各段階の個々の関数を呼び出すコードを含んでいる	変更は不要

4.3 特性の選択

設計者はプロパティ、ラベルおよび説明の 3 つの特性を設定しなければなりません。そのためには ISSP_DIRECTIVES.H ファイル内のいくつかの#define ステートメントをコメントまたはコメント削除します。これらの#define ステートメントは「User Attention Required (ユーザーは要注意)」と明確に示されているため、簡単に見つけることができます。また、各ラベルのページ検索を行うこともできます。以下は各特性とそのラベルの説明です。

プロパティ: UART デバッグ
ラベル: USE_TX
説明: この#define ステートメントをコメント アウトして、UART を介してメッセージを送信するコードを無効にします。非コメント化すると、UART 関数の実行を有効にします。

プロパティ: テスト ポイント
ラベル: USE_TP
説明: この#define ステートメントをコメント アウトして、テスト ピンをビットバングするコードの実行を無効にします。このピンはデバッグ処理に使用されます。これは最終設計に不要です。

プロパティ: ターゲットの供給電圧
ラベル: TARGET_SUPPLY_VOLTAGE
説明: ターゲットの動作電圧が 3.3V の場合にはこの #define ステートメントをコメントアウトし、5V の場合には非コメント化します。

プロパティ: プログラミング モード
ラベル: PROGRAMMING_MODE
説明: パワー サイクル モードを使用する場合はこの#define ステートメントをコメント アウトします。#define を非コメント化すると、ターゲットがリセット モードでプログラミングされるようになります。

プロパティ: 対象の PSoC デバイス
ラベル: TARGET_PSOC
説明: このセクションでターゲット PSoC を選択します。いかなる場合でも 1 台のデバイスしか有効にはできず、その他のデバイスはコメント アウトされます。リストにデバイスが記載されていない場合には、サイプレスのテクニカル サポートまでお問い合わせください。

4.4 低レベル ドライバーの変更

設計者はターゲットとなる PSoC のプログラミングに関連するピンを操作するための、ホスト固有のコードを提供しなければなりません。これらの API は ISSP_PHYSICALLAYER.C に「Processor Specific (プロセッサ固有)」と「User Attention Required (ユーザーは要注意)」と示されています。

- **ポート ビット マスク:** 使用されているホスト プロセッサに応じて調整する必要があるポート ビット マスクが 4 つあります。4 つのビットを設定する必要がありますが、3 つのみがプログラミングで使用され、どの 3 つかはプログラミング方法の選択によります: リセット モードではその 3 つは SDATA、SCLK、XRES になり、パワー サイクル モードでは SDATA、SCLK、PWR になります。
- **Delay(n) 関数:** この関数は「while」ループの繰り返しが最低 2µs となるように調整します。基本的には、ループ時間に上限はありません。しかし、ループ時間に要する時間が長いほど、ターゲットのプログラミングする時間も長くなります。例えば、ホスト マイクロコントローラーも PSoC の場合、各反復ごとは 2µs に加えて 7µs のオーバーヘッドが発生します。そのため、関数は 2n+7µs (n は関数に渡されるパラメーター) の遅延を生み出します。ホスト プロセッサの遅延時間を調整するには、ISSP_DELAYS.H の#define ステートメントを変更します。
- **ポートビット操作関数:** これらの関数は PSoC のプログラミングに必要な信号を生成するようにホスト ピンを操作します。それらはピンを HIGH か LOW に駆動したり、High-Z 状態にしたりします。以下は関数の一覧です。ほとんどの関数の機能は見ればすぐ分かりますが、すべてコード内で説明されています。

```
fSDATACheck()
SCLKHigh()
SCLKLow()
SetSCLKStrong()
SetSDATAHigh()
SetSDATALow()
SetSDATAHiZ()
SetSDATAStrong()
SetXRESStrong()
AssertXRES()
DeassertXRES()
SetSCLKHiZ()
SetTargetVDDStrong()
ApplyTargetVDD()
RemoveTargetVDD()
```

- **UART 関数:** これらの関数はデバッグ処理に使用されます。プログラミング プロセスの進行を知らせるために、メッセージが UART を介して送信されます。ホスト プロセッサに基づいて以下の関数を変更する必要があります。TX8 ユーザー モジュールが UART トランスミッタのためにデザインに配置されます。

```
InitTx()
SendDebugMessage()
PrintReceivedSiliconID()
DisplaySecurityValues()
```

- **テスト ポイント関数:** これらの関数はデバッグ処理に使用されます。 ホスト プロセッサに基づいて次の関数を変更します。

```
InitTP()
SetTPHigh()
SetTPLow()
ToggleTP()
```

4.5 RAMバッファへのデータ書き込み

HSSP コードは 64 バイト バッファからデータを読み出し、連続的に PSoC のフラッシュ ブロックをプログラミングします。このプロセスは最下位のブロック アドレスから開始します。最初のブロックがプログラミングされた後、その他のフラッシュ ブロックをプログラミングする際にも同じバッファが使用されます。

設計者はデータ ソース (USB、RS-232、SD カード等) に応じて、このバッファを埋めるコードを提供しなければなりません。使用されている特定のホスト プロセッサ用に書き込まれる関数が 2 つあります — `LoadProgramData()` と `LoadSecurityData()`。これらの関数は `ISSP_PACKETLAYER.C` にあり、「Processor Specific (プロセッサ固有)」と「User Attention Required」(ユーザーは要注意)」と示されています。 現在、これらの関数は増加する値をバッファにロードします。

5 フラッシュ ブロックのシーケンスまたは数量の変更

場合によっては、フラッシュの特定のエリアをプログラミングする必要があります。例は、特性評価、較正、またはファームウェア フィールドの更新用に設けられた領域です。これらの機能は通常、EEPROM ユーザー モジュールを使用して実装されます。しかし、場合によってはそれらを PSoC に直接プログラミングすることで、上限に近付いた時にコード空間を節約します。

対象のブロックの開始アドレスと、ブロックをプログラムする順番を変更することができます。各プログラミング シーケンスにはブロック アドレスが含まれているため、それによって問題が生じることはありません。しかし、以下の点に注意してください:

- フラッシュ バンク番号はブロックの書き込みごとに一度しか設定されません。他のファミリはバンクが 1 つしかないため、これは CY8C29x66、CY8C28xxx、CY8C24x94、CY8CTST120、CY8CTMG120 および CY8CTMA120 製品ファミリにしか適用されません。バンクの詳細については、PSoC テクニカル リファレンス マニュアル (TRM)をご参照ください。FLS_PR1 レジスタは、プログラミングの呼び出しがどのフラッシュ ブロックに影響を与えるかを決定します。
- プログラミングのループが変更されたら、検証のループにも同じ変更が行わなければなりません。行なわれないと、検証は失敗します。
- その間、コードはチェックサムを累算します。コードは、その時点までのフラッシュ全体に対してチェックサムを検査します。フラッシュの 1 つのセクションのみをプログラムする場合、必要に応じて変数 `iChecksumData` を設定する必要があります。

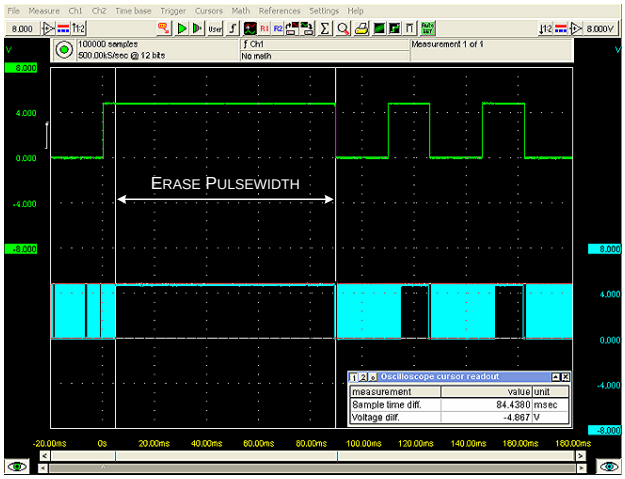
6 内蔵テスト ポイントによるフラッシュ書き込み時間の検証

ホスト ソースプログラミングを成功させる上で最も重要な要素の 1 つは、適切な消去および書き込みパルス幅を用いることです。プロセスをサポートするために、プログラムにはいくつかの戦略的に配置されたテスト ポイント (TP) の呼び出しが行われています。デバッグ モードを有効にするには、`ISSP_DIRECTIVES.H` 内の `USE_TP #define` のコメントを解除してください。先述のピン操作関数に類似している、デバッグ モードに関連するいくつかの関数があります。システム設計者はピンを HIGH か LOW に駆動するか、ピンをトグルするためにホストの特定コードを提供しなければなりません。

正しくデバッグするためには TP と SDATA ラインを監視する必要があります。消去およびプログラミング パルスの両方とも測定されなければなりません。最も良い測定方法は、2 チャンネルのオシロスコープを使用し、それを TP チャンネルの立ち上がりエッジからシングルシーケンス モードでトリガーをかけることです。

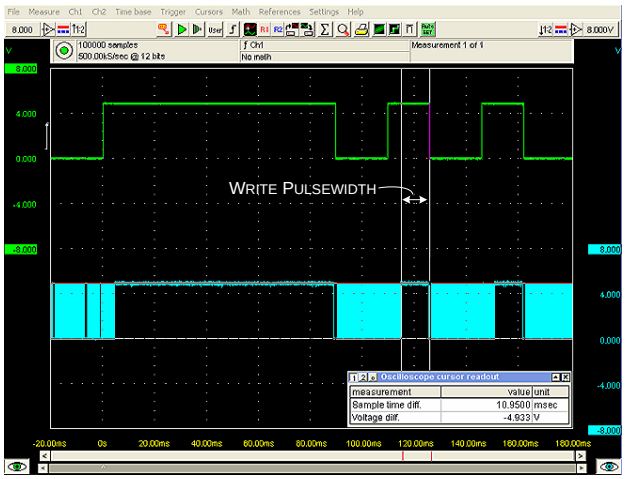
消去パルス幅は、図 3 に示すようにデータ バーストの終わりから TP の立ち下がりエッジまで測定します。TP の立ち上がりエッジはデータ バーストの終わりと整列しないことに注意してください。これは、TP ピンが HIGH に駆動された瞬間から、ホストがデータを送信し始めるまでのオーバーヘッドによって生じる遅延のためです。

図 3. 消去パルス幅の測定



プログラミング パルス幅も同様にデータ バーストの終わりから TP の立ち下がりエッジまで測定されます。図 4 にこの測定結果を示しています。消去パルス幅と同様に、TP 信号の立ち上がりエッジはデータ バーストの終わりと合致しません。

図 4. 書き込みパルス幅の測定



さまざまなPSoCデバイスの理想的な消去および書き込みパルス幅は、それぞれのデバイスのデータシートにおける「AC プログラミング仕様」表をご覧ください。測定値は理想値の–3～+15%内でなければなりません。この要件を満たさないと、プログラミングを適切に行うことはできず、規定のフラッシュ データ保持期間よりも保持期間が短くなったり[1]、期待よりも消去と書き込みサイクルが少なくなる[2] などの望ましくない悪影響となります。

[1] デバイス データシートの「DC プログラミング仕様」節に $Flash_{DR}$ のシンボルで記載されます。

[2] デバイス データシートの「DC プログラミング仕様」節に $Flash_{ENPB}$ と $Flash_{ENT}$ シンボルで記載されます。

7 UARTデバッグ処理のメッセージ

UART が有効な現時点コードはデバッグ メッセージを送信します。図 5 には、プログラミング プロセス中に送信されるステータス メッセージの例を示しています。

図 5. ステータス メッセージ



8 制約

main.c の冒頭部分のコメントは、システム設計者が考慮する必要がある、有用で重要な情報を含んでいます。HSSP コードにはそれらのコメントで説明されている、いくつかの制約があります。以下はその概要です。

- シリアル プログラミングは 5℃～50℃の温度範囲内でのみ行います。
- HSSP プログラムは 3.0V 未満の電圧に対応していません。
- プログラミング手順は 1 つの電圧範囲のみで実施されます。デバイスが 5.0V で初期化されている場合には、手順全体を 5.0V 範囲で 5.0V ベクターで完了する必要があります。

- CY8C20x34 および旧 PSoC バージョンは現在サポートされていません。このアプリケーション ノートでサポートされていないデバイス一覧については、添付のプロジェクトの *main.c* のヘッダーセクションをご覧ください。
- SCLK の周波数の上限はデバイス データシートの「AC プログラミング仕様」節に F_{SCLK} の記号で示されています。
- サイプレスでは CY8C20x36/46A/66A/96A/ CY8CTMG2xx/ CY8CTST2xx 製品の ISSP バスラインを他の PsoC デバイスと共有することを推奨しません。しかし、CY8C20x36/46A/66A/96A/ CY8CTMG2xx/ CY8CTST2xx 製品の ISSP バスを他の PSoC デバイスと共有する場合には、CY8C20x36/46A/66A/96A/ CY8CTMG2xx/ CY8CTST2xx 製品がリセット状態時に「AC52」を見ることがないようにしてください。詳細情報については、「CY8C20X36A/46A/66A/96A: ISSP バスの共有問題」という知識ベースの記事参照してください。

9 要約

本アプリケーション ノートは、設計者が、独自のシリアル プログラミング ソフトウェアを開発できるように、HSSP コード例を提供しています。また、PSoC 1 ホストプログラミング ソリューションを実装するためのスタート ポイントを提供し、ポータブルな C ソース コードにより、ソリューションの開発時間を節約します。

10 関連文書:

- PSoC® 1 プログラミング仕様 - CY8C21x23、CY8C21x34、CY8C23x33、CY8C24x23A、CY8C27x43、CY8CTMG110、CY8CTST110
- PSoC® 1 ISSP プログラミング仕様 - CY8C21x45、CY8C22x45、CY8C24x94、CY8C28xxx、CY8C29x66、CY8CTST120、CY8CTMA120、CY8CTMG120、CY7C64215
- ISSP プログラミング仕様書
- AN59389 - CY8C20xx6A 用ホストソース シリアル プログラミング

A 付録A: ポート ビット操作関数

表 3. ポート ビット操作関数

関数名	説明
SetSCLKStrong()	SCLK ピンを出力 (ストロング駆動モード) にセット
SetSCLKHiZ()	SCLK ピンを Hi-Z にする
SetSDATAHigh()	SDATA ピンを HIGH にセット
SetSDATALow()	SDATA ピンを LOW にセット
SetSDATAStrong()	SDATA ピンを出力 (ストロング駆動モード) にセット
SetSDATAHiZ()	SDATA ピンを (ターゲットによって駆動されるように) Hi-Z にする
AssertXRES()	XRES ピンを HIGH にセット
DeassertXRES()	XRES ピンを LOW にセット
SetXRESStrong()	XRES ピンを出力 (ストロング駆動モード) にセット
ApplyTargetVDD()	ターゲット PSoC に電源を供給
RemoveTargetVDD()	ターゲット PSoC の電源を切断
SetTargetVDDStrong()	PWR ピンを出力 (ストロング駆動モード) にセット

改訂履歴

文書名: AN44168 - 外部マイクロコントローラーを使用した PSoC® 1 デバイス プログラミング (HSSP)

文書番号: 001-82518

版	ECN	改版者	発行日	変更内容
**	3732377	HZEN	09/03/2012	これは英語版 001-44168 Rev. *F を翻訳し た日本語版 001-82518 Rev. **です。
*A	4669798	MIOU	03/26/2015	これは英語版 001-44168 Rev. *G を翻訳し た日本語版 001-82518 Rev. *A です。
*B	5002847	HZEN	11/06/2015	これは英語版 001-44168 Rev. *H を翻訳した日本語版 001-82518 Rev. *B 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

車載用	cypress.com/go/automotive
クロック & バッファ	cypress.com/go/clocks
インターフェース	cypress.com/go/interface
照明 & 電源管理	cypress.com/go/powerpsoc
メモリ	cypress.com/go/memory
PSoC	cypress.com/go/psoc
タッチ センシング	cypress.com/go/touch
USB コントローラー	cypress.com/go/usb
ワイヤレス／RF	cypress.com/go/wireless

PSoC®ソリューション

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

サイプレス開発者コミュニティ

コミュニティ | フォーラム | ブログ | ビデオ | トレーニング

テクニカル サポート

cypress.com/go/support

PSoC はサイプレス セミコンダクタ社の登録商標であり、PSoC Creator は同社の商標です。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
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2008-2015. 本文書に記載される情報は、予告なく変更される場合があります。Cypress Semiconductor Corporation (サイプレス セミコンダクタ社) は、サイプレス製品に組み込まれた回路以外のいかなる回路を使用することに対して一切の責任を負いません。サイプレス セミコンダクタ社は、特許またはその他の権利に基づくライセンスを譲渡することも、または含意することもありません。サイプレス製品は、サイプレスとの書面による合意に基づくものでない限り、医療、生命維持、救命、重要な管理、または安全の用途のために使用することを保証するものではなく、また使用することを意図したものでもありません。さらにサイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことが合理的に予想される生命維持システムの重要なコンポーネントとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。
このソースコード (ソフトウェアおよび／またはファームウェア) はサイプレス セミコンダクタ社 (以下「サイプレス」) が所有し、全世界の特許権保護 (米国およびその他の国)、米国の著作権法ならびに国際協定の条項により保護され、かつそれらに従います。サイプレスが本書面によりライセンシーに付与するライセンスは、個人的、非独占的かつ譲渡不能のライセンスであり、適用される契約で指定されたサイプレスの集積回路と併用されるライセンシーの製品のみをサポートするカスタム ソフトウェアおよび／またはカスタム ファームウェアを作成する目的に限って、サイプレスのソース コードの派生著作物をコピー、使用、変更そして作成するためのライセンス、ならびにサイプレスのソース コードおよび派生著作物をコンパイルするためのライセンスです。上記で指定された場合を除き、サイプレスの書面による明示的な許可なくして本ソース コードを複製、変更、変換、コンパイル、または表示することはすべて禁止します。
免責条項: サイプレスは、明示的または黙示的を問わず、本資料に関するいかなる種類の保証も行いません。これには、商品性または特定目的への適合性の黙示的な保証が含まれますが、これに限定されません。サイプレスは、本文書に記載される資料に対して今後予告なく変更を加える権利を留保します。サイプレスは、本文書に記載されるいかなる製品または回路を適用または使用したことによって生ずるいかなる責任も負いません。サイプレスは、誤作動や故障によって使用者に重大な傷害をもたらすことが合理的に予想される生命維持システムの重要なコンポーネントとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。
ソフトウェアの使用は、適用されるサイプレス ソフトウェア ライセンス契約によって制限され、かつ制約される場合があります。