XMC4000 用の ASC ブートストラップローダー

XMC4000

About this document

Scope and purpose

このアプリケーションノートでは、ASC BSL を使用して、XMC4000 マイクロコントローラファミリのプログラムをフラッシュにダウンロードする方法について説明します。サンプルコードは、XMC4000 デバイスでフラッシュ消去、プログラミング、フラッシュリードバックを実行する方法を示すためにアプリケーションノートとともに提供されます。該当する製品は XMC4000 マイクロコントローラファミリです。サンプルコードは、XMC4200/XMC4400/XMC4500/XMC4800 アプリケーションボードでテストされています。

Intended audience

このアプリケーションノートは、XMC4000 ファミリのフラッシュプログラミングの開発者、および ASCBSL を使用してプログラムをフラッシュでダウンロードすることを希望するお客様を対象とします。

Table of contents

About this document **1**
Table of contents **1**
1 はじめに **3**
1.1 ツールチェーン 3
1.1.1 フラッシュプログラムの例 3
2 ASC ブートストラップのロード **5**
2.1 フラッシュローダー 7
2.2 DAVE4 プロジェクト設定 7
2.3 Keil プロジェクトの設定 7
3 フラッシュメモリの構成 **8**
3.1 XMC4800 8
3.2 XMC4500 9
3.3 XMC4400 9
3.4 XMC4200 10
4 通信プロトコル **11**
4.1 Mode 0：プログラムフラッシュページ 11
4.2 Mode 1：フラッシュからユーザープログラムの実行 12
4.3 Mode 2：iCache からユーザープログラムの実行 13
4.4 Mode 3：フラッシュセクタの消去 13
4.5 Mode 4：フラッシュ保護ステータスの読み出し 13
4.6 Mode 5：フラッシュの保護または保護解除 14
4.7 ホストへの応答コード 14
5 ホスト PC プログラムの例 **16**
6 デモンストレータの使用 **19**
6.1 ハードウェアの準備 19

6.2 デモンストレータのファイル構成 19
6.3 デモンストレータの実行 20
7 参考資料 23
改訂履歴 24

1 はじめに

XMC4000 マイクロコントローラー ファミリには、フラッシュプログラミングに使用できるブートストラップローディング (BSL) メカニズムが組み込まれています。このメカニズムについては、XMC4000 ユーザーマニュアルの BootROM の章で詳しく説明されています。しかし、XMC4000 ファミリの製品には、フラッシュの書き込み、読み出し、消去、検証などのフラッシュプログラミングを実行するための、ハードコードされたブートストラップローダールーチンは BootROM には実装されていません。したがって、フラッシュルーチンを提供するフラッシュローダープログラムは、ユーザーが実装する必要があります。

XMC4000 ファミリは、非同期シリアルインターフェース (ASC) BSL とコントローラーエリアネットワーク (CAN) BSL の両方をサポートします。このアプリケーションノートでは、ASC インターフェースを使用したブートストラップロードについて説明します。

ターゲットデバイスは、ASC インターフェースを介して PC に接続されます。このアプリケーションノートで示されるフラッシュローダーシステムは、次の 2 つの部分で構成されます。

- フラッシュローダープログラム
  - フラッシュローダープログラムは、組込みのブートストラップロードメカニズムを使用してターゲットデバイスに送信されます。プログラムが送信されて実行されると、フラッシュローダープログラムは、PC 上で実行されている HOST プログラムからコマンドを受信するための通信プロトコルを確立し、ターゲットデバイスのフラッシュプログラミングを制御します。
- HOST PC プログラム
  - PC 上で実行されている HOST プログラムは、フラッシュローダーによって定義された通信プロトコルを使用します。フラッシュプログラミングコマンドとプログラムするコードバイトを送信します。HOST プログラムはアプリケーション固有であるため、このアプリケーションノートの HOST プログラムは単なる例です。

1.1 ツールチェーン

ASC 用のフラッシュローダープログラムは、次のツールチェーンを使用して開発されます。

- DAV4 開発プラットフォーム v4.4.2
- Keil ツールチェーン v.5.1

この例で提供されているこれら 2 つのツールチェーンのプロジェクトファイルは互いに独立しており、ユーザーは 2 つのいずれかを使用することを選択できます。

1.1.1 フラッシュプログラムの例

フラッシュプログラムの例である、P3.9 によって制御される LED を切り替えるプロジェクト LED_Blinky は、2 つのツールチェーンすべてに提供されています。Blinky.hex ファイルはフラッシュメモリにダウンロードできます。XMCLoad HOST PC プログラムは、Microsoft Visual C++ 2010 で開発されています。サンプルのソースコードは、次のフォルダにあります。

- .\DAV4\XMC4x00\ASCLoader には、GCC コンパイラを使用して開発された ASCBSL ローダーが含まれます。
- .\Keil\XMC4x00\ASCLoader には、Keil コンパイラを使用して開発された ASCBSL ローダーが含まれます。
- .\DAV4\XMC4x00\LED_Blinky には、GCC コンパイラを使用して開発されたフラッシュサンプルプログラムが含まれます。

- .\Keil\XMC4x00\LED_Blinky には、Keil コンパイラを使用して開発されたフラッシュサンプルプログラムが含まれます。
- .\XMCLoad\は、フラッシュプログラミングのプロセス全体を示すサンプルの HOST PC プログラムを保持します。プロジェクトファイルは、Microsoft Visual C++2010 でコンパイルできます。

第 6 章では、デモンストレータを使用して独自のプログラムをフラッシュにダウンロードして実行する方法について詳しく説明します。

2 ASC ブートストラップのロード

PC とターゲットデバイス間の通信は、ASC インターフェースを介して確立されます。Figure 1 は、このアプリケーションのハードウェア設定を示しています。ターゲットデバイス側では、USIC0 (U0C0) のチャネル 0 が ASC として使用されます。ポート P1.4 と P1.5 は、それぞれ RxD と TxD として使用されます。

- ピン P1.4 でピン RxD を受信 (USIC0_DX0B)
- ピン P1.5 でピン TxD を送信 (USIC0_DOUT0)

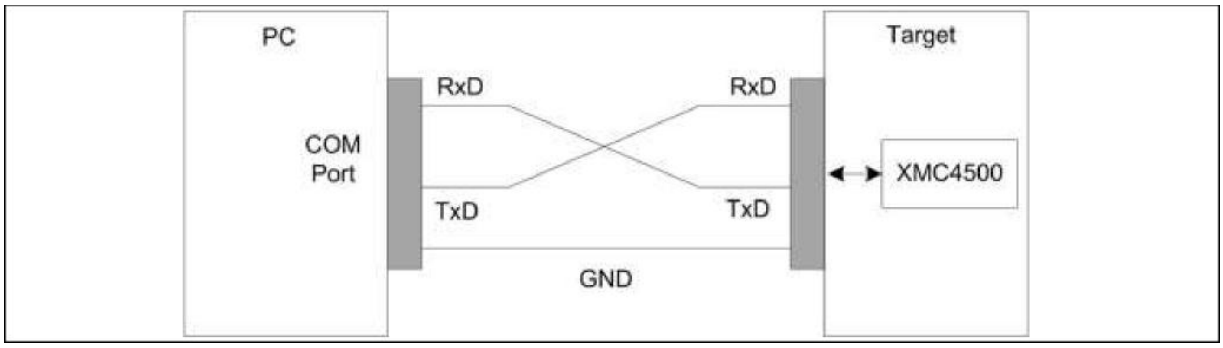


Figure 1 XMC4000 ブートストラップロード用の PC とターゲットシステム間の接続

このプログラムを実行するための最初のステップは、ターゲットデバイスを ASC BSL モードにすることです。

ブートピン TMS = 0 および TCK = 0 の場合、デバイスのリセット時に ASC ブートストラップローダーモードに入ります。これらは、ターゲットボードの DIP スイッチによって設定されます。

XMC4000 のコンフィグレーションピン TCK および TMS は、通常、XMC4000 ボードの DIP スイッチに接続されます。

TMS がスイッチピン 1 に接続され、TCK がスイッチピン 2 に接続されると仮定した場合、DIP スイッチの設定を Figure 2 に示します。

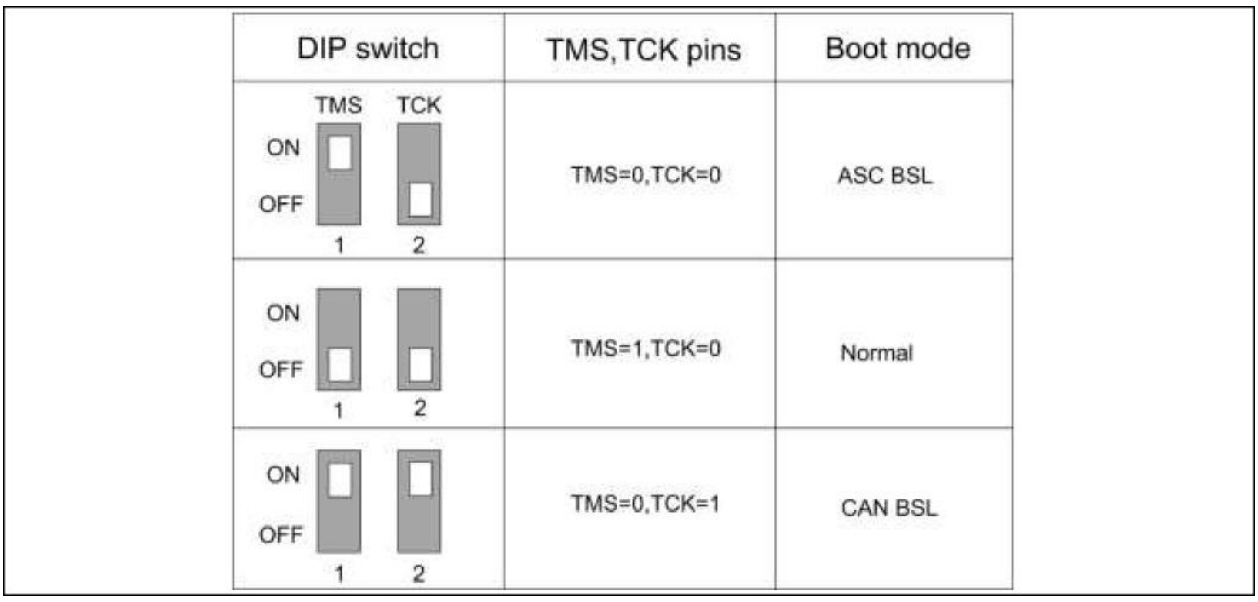


Figure 2 ブートモードの DIP スイッチ設定

ASC ブートストラップのロード

ブートストラップローダーの手順を Figure 3 に示します。

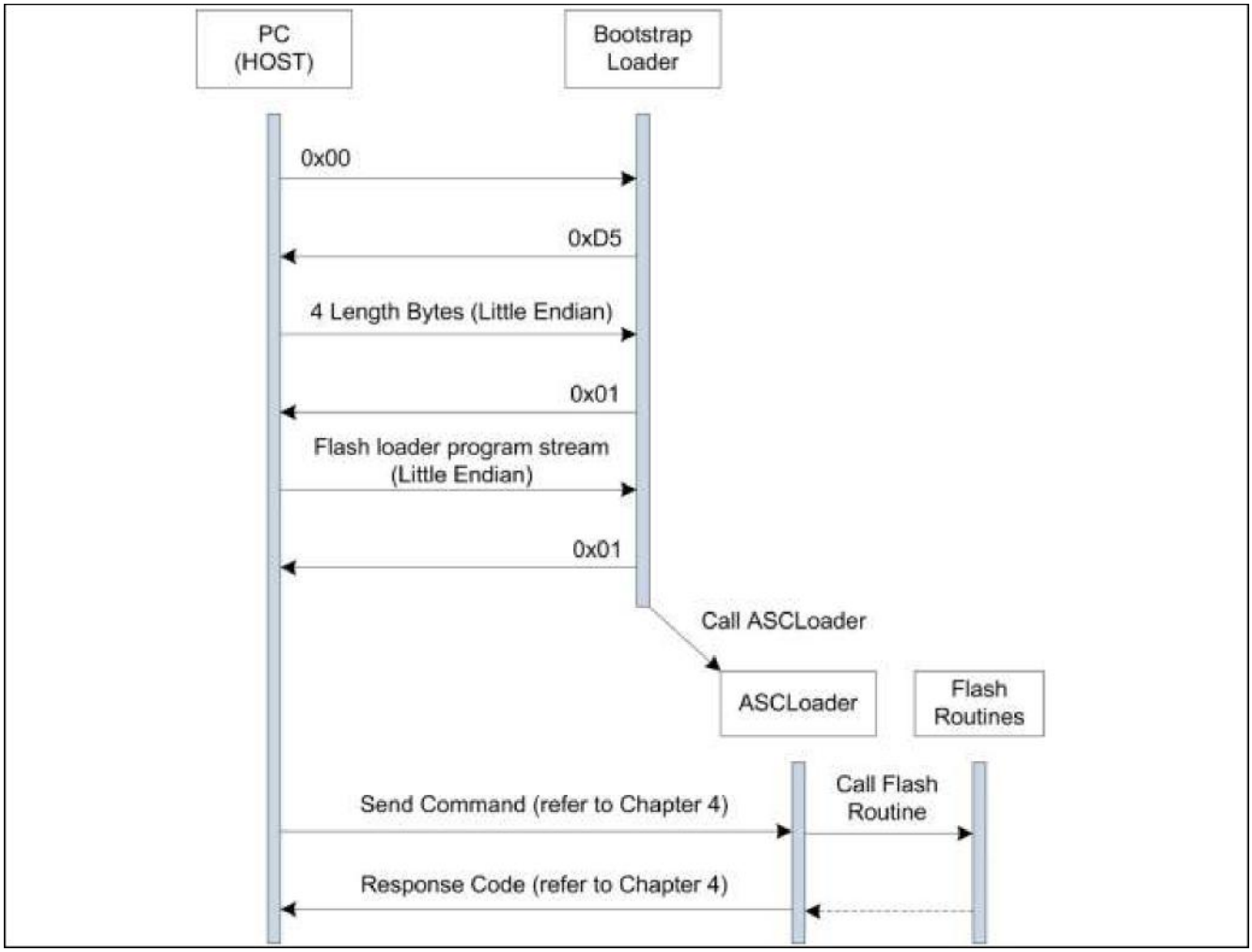


Figure 3　フラッシュプログラミング用の ASC ブートストラップローダー手順

HOST は、デバイスがボーレートを検出できるようにゼロバイトを送信することから始めます。XMC4000 デバイスは、最大 115200 ビット/秒のボーレートをサポートします。ASC インターフェースは、8 データビットと 1 ストップビット用に初期化されます。

ボーレートがデバイスによって検出された後、ブートストラップローダーは確認応答バイト D5H をホストに送り返します。次に、ホストからのフラッシュロードプログラムの長さを表す 4 バイトを待機します。最下位バイトが最初に受信されます。アプリケーションの長さが BSL で許容できると判断された場合、OK (0x01H) バイトがホストに送信され、ホストはフラッシュローダーのバイトストリームを送信します。バイトストリームが受信されると、BSL は最後の OK バイトを送信してプロトコルを終了し、フラッシュローダープログラムに制御を移します。

アプリケーションの長さにエラーがある場合 (つまり、アプリケーションの長さがデバイスの PSRAM サイズよりも大きい場合)、N_OK バイト (0x02H) が HOST に返送され、BSL は正しいバイト長の待機を再開します。

ファイル ASCLoader.hex には、フラッシュロードプログラムが含まれます。ASCLoader は PSRAM にダウンロードされて実行された後、最初に PC とターゲットデバイス間の通信を確立し、次にフラッシュ操作を実行します。

2.1 フラッシュローダー

フラッシュローダーはフラッシュルーチンを実装し、PC とターゲットデバイス間の通信を確立します。ASCLoader の主要部分 (main.c) は、次の機能を提供するフラッシュルーチンを実装します。

- フラッシュセクタの消去
- プログラムフラッシュページ
- プログラムされたフラッシュページの検証
- フラッシュ保護の設定
- フラッシュ保護の削除
- フラッシュ保護ステータスの読み出し
- フラッシュと iCache の両方からコードの実行

フラッシュを消去してプログラムするには、セクタとページアドレスを指定する必要があります。無効なアドレス (フラッシュの境界内にないアドレス) は、アドレスエラーになります。XMC4000 のメモリ構成については、フラッシュメモリ構成の章で説明します。

フラッシュユーザーコードは、フラッシュベースアドレス 0xC0000000 および iCache ベースアドレス 0x08000000 から実行できます。

2.2 DAVE4 プロジェクト設定

フラッシュローダーDAV4 プロジェクトは、.\DAV4\ASCLoader フォルダにあります。プロジェクトは、次の手順で DAV4IDE にインポートできます。

- DAV4 IDE を開く
- インフィニオン DAVE プロジェクトをインポート
- ルートディレクトリを.\DAV4\ASCLoader として選択
- インポートを終了

2.3 Keil プロジェクトの設定

フラッシュローダー用の Keil プロジェクトは、.\Keil\ASCLoader フォルダにあります。Keil コンパイラのバージョンは v5.1 です。プロジェクトは、次のように Keil µVersion にインポートできます。

- Project に移動->open project
- フォルダ.\Keil\Flash_Loader に移動 ->プロジェクトファイル“ASCLoader.uvproj”を選択 ->open

フラッシュローダーは PSARM から実行する必要があるため、メモリは次のように定義する必要があります。

- ターゲットに移動
- IROM1 は 0x10000000 (XMC4500)または 0x1FFFC000 (XMC4400/4200)で開始、サイズは 0x10000 (XMC4500)または 0x4000 (XMC4400/4200)、Startup->yes
- IRAM1 は 0x20000000 で開始、サイズは 0x10000

デフォルトでは、Keil コンパイラは ELF 形式でファイル拡張子が.axf のオブジェクトファイルを生成します。ただし、フラッシュローダーには HEX 形式のファイルが必要です。HEX ファイルの出力を取得するには、次の場所に移動します。

- Option を開く->Output->Create HEX file

3 フラッシュメモリの構成

XMC4x00 ファミリの組込みフラッシュモジュールには、コードまたは定数データ用に最大 1.0 MB のフラッシュメモリ (プログラムフラッシュと呼ばれる) が含まれます。PMU には、キャッシュ可能またはキャッシュ不可能なアドレス空間を介してアクセス可能な 1 つの PFLASH バンクが含まれます。

PFlash メモリは、そのセクタアーキテクチャとページ構造によって特長付けられます。セクタは、さまざまなサイズのフラッシュメモリパーティションです。各セクタのオフセットアドレスは、Table 1 に示されるバンクのベースアドレスを基準にしています。派生デバイス (XMC4000 データシートを参照) は、フラッシュメモリが少なくなる可能性があります。PFLASH バンクは、番号の大きい物理セクタを遮断することで縮小します。

Table 1 フラッシュメモリマップ

範囲の説明	サイズ	開始アドレス
PMU0 プログラムフラッシュバンク キャッシュされない空間	2M バイト (XMC4800) 1M バイト (XMC4500) 512K バイト (XMC4400) 256K バイト (XMC4200)	0xC000000H
PMU0 プログラムフラッシュバンク キャッシュされた空間 (同じ物理メモリに対して異なるアドレス空間、キャッシュされないアドレス空間にマップ)	2M バイト (XMC4800) 1M バイト (XMC4500) 512K バイト (XMC4400) 256K バイト (XMC4200)	0x8000000H

- フラッシュ消去はセクタごとです。
- セクタはページに細分されます。
- フラッシュメモリのプログラミングはページ単位です。
- PFlash ページには 256 バイトが含まれます。

次の表に、XMC4x00 ファミリの製品の論理セクタ構造を示します。

3.1 XMC4800

XMC4800 では、フラッシュモジュール PMU0 に 2MB の Pflash メモリが含まれます。Table 2 に、XMC4800 のフラッシュ論理セクタ構造を示します。

Table 2 XMC4500 の PFLASH のセクタ構造

セクタ	アドレス範囲	サイズ
0	0xC000000-0xC003FFF	16 KB
1	0xC004000-0xC007FFF	16 KB
2	0xC008000-0xC00BFFF	16 KB
3	0xC00C000-0xC00FFFF	16 KB
4	0xC010000-0xC013FFF	16 KB
5	0xC014000-0xC017FFF	16 KB
6	0xC018000-0xC01BFFF	16 KB
7	0xC01C000-0xC01FFFF	16 KB
8	0xC020000-0xC03FFFF	128 KB
9	0xC040000-0xC07FFFF	256 KB

セクタ	アドレス範囲	サイズ
10	0xC080000-0xC0BFFFF	256 KB
11	0xC0C0000-0xC0FFFFF	256 KB
12	0xC100000-0xC13FFFF	256 KB
13	0xC140000-0xC17FFFF	256 KB
14	0xC180000-0xC1BFFFF	256 KB
15	0xC1C0000-0xC1FFFFF	256 KB

3.2 XMC4500

XMC4500 では、フラッシュモジュール PMU0 に 1MB の Pflash メモリが含まれます。Table 3 に、XMC4500 のフラッシュ論理セクタ構造を示します。

Table 3 XMC4500 の PFLASH のセクタ構造

セクタ	アドレス範囲	サイズ
0	0xC000000-0xC003FFF	16 KB
1	0xC004000-0xC007FFF	16 KB
2	0xC008000-0xC00BFFF	16 KB
3	0xC00C000-0xC00FFFF	16 KB
4	0xC010000-0xC013FFF	16 KB
5	0xC014000-0xC017FFF	16 KB
6	0xC018000-0xC01BFFF	16 KB
7	0xC01C000-0xC01FFFF	16 KB
8	0xC020000-0xC03FFFF	128 KB
9	0xC040000-0xC07FFFF	256 KB
10	0xC080000-0xC0BFFFF	256 KB
11	0xC0C0000-0xC0FFFFF	256 KB

3.3 XMC4400

XMC4400 では、フラッシュモジュール PMU0 に 512KB の Pflash メモリが含まれます。Table 4 に、XMC4400 のフラッシュ論理セクタ構造を示します。

Table 4 XMC4400 の PFLASH のセクタ構造

セクタ	アドレス範囲	サイズ
0	0xC000000-0xC003FFF	16 KB
1	0xC004000-0xC007FFF	16 KB
2	0xC008000-0xC00BFFF	16 KB
3	0xC00C000-0xC00FFFF	16 KB
4	0xC010000-0xC013FFF	16 KB
5	0xC014000-0xC017FFF	16 KB
6	0xC018000-0xC01BFFF	16 KB
7	0xC01C000-0xC01FFFF	16 KB

セクタ	アドレス範囲	サイズ
8	0xC020000-0xC03FFFF	128 KB
9	0xC040000-0xC07FFFF	256 KB

3.4 XMC4200

XMC4200 では、フラッシュモジュール PMU0 に 256KB の Pflash メモリが含まれます。Table 5 に、XMC4200 のフラッシュ論理セクタ構造を示します。

Table 5 XMC4200 の PFLASH のセクタ構造

セクタ	アドレス範囲	サイズ
0	0xC000000-0xC003FFF	16 KB
1	0xC004000-0xC007FFF	16 KB
2	0xC008000-0xC00BFFF	16 KB
3	0xC00C000-0xC00FFFF	16 KB
4	0xC010000-0xC013FFF	16 KB
5	0xC014000-0xC017FFF	16 KB
6	0xC018000-0xC01BFFF	16 KB
7	0xC01C000-0xC01FFFF	16 KB
8	0xC020000-0xC03FFFF	128 KB

4 通信プロトコル

フラッシュローダープログラム「ASCLoader」は、ホスト PC からコマンドを受信するための通信構造を確立します。

HOST は、転送ブロックを介してコマンドを送信します。次の 3 種類のブロックが定義されます。

ヘッダーブロック

バイト 0	バイト 1	バイト 2 ... 14	バイト 15
ブロックタイプ (0x00)	Mode	Mode 固有のコンテンツ	チェックサム

ヘッダーブロックの長さは 16 バイトです。

データブロック

バイト 0	バイト 1	バイト 2 ... 257	バイト 258 ... 262	バイト 263
ブロックタイプ (0x01)	検証オプション	256 データバイト	未使用	チェックサム

データブロックの長さは 264 バイトです。

EOT ブロック

バイト 0	バイト 1 ... 14	バイト 15
ブロックタイプ (0x02)	未使用	チェックサム

EOT ブロックの長さは 16 バイトです。

HOST に必要なアクションは、ヘッダーブロックの Mode バイトに示されます。

フラッシュローダープログラムは、有効なヘッダーブロックの受信を待機し、対応するアクションを実行します。ブロックが正しく受信されたかどうかは、チェックサムによって判断されます。チェックサムは、ブロックタイプバイトとチェックサムバイト自体を除くすべてのブロックバイトの XOR 合計として計算されます。

ASC BSL モードでは、すべてのブロックバイトが UART インターフェースを介して一度に送信されます。さまざまなモードで、ASCLoader によって実行されるフラッシュルーチンが指定されます。Mode とそれに対応する通信プロトコルについては、この章の次のセクションで説明します。

4.1 Mode 0：プログラムフラッシュページ

ヘッダーブロック

バイト 0	バイト 1	バイト 2 ... 5	バイト 6 ... 14	バイト 15
ブロックタイプ (0x00)	Mode (0x00)	ページ アドレス	未使用	チェックサム

- ページアドレス (32 ビット)
  - プログラムするフラッシュページのアドレスです。このアドレスは 256 バイト配列で、有効な範囲でなければなりません (第 3 章参照)。そうでない場合は、アドレスエラーが発生します。バイト 2 は最上位バイト、バイト 5 は最下位バイトを示します。

ヘッダーブロックの受信後、デバイスは確認応答として **0x55** を送信するか、無効なブロックのエラーコードを送信します。ローダーは、以下に示す形式で後続のデータブロックを受信するのを待機するループに入ります。

ループは、EOT ブロックをターゲットデバイスに送信することで終了します。

データブロック

バイト 0	バイト 1	バイト 2 ... 257	バイト 258 ... 262	バイト 263
ブロックタイプ (0x01)	検証オプション	256 データバイト	未使用	チェックサム

- 検証オプション
  - このバイトを 0x01 に設定して、プログラムされたページバイトの検証を要求します。
  - 0x00 に設定すると、検証は実行されません。
- コードバイト
  - ページコンテンツ。

各受信データブロックの後、デバイスは確認応答として **0x55** を PC に送信するか、エラーコードを送信します。

EOT ブロック

バイト 0	バイト 1 ... 14	バイト 15
ブロックタイプ (0x02)	未使用	チェックサム

EOT ブロックを受信するたびに、デバイスは確認応答として **0x55** を PC に送信するか、エラーコードを送信します。

4.2 Mode 1：フラッシュからユーザープログラムの実行

ヘッダーブロック

バイト 0	バイト 1	バイト 2 ... 14	バイト 15
ブロックタイプ (0x00)	Mode (0x01)	未使用	チェックサム

このコマンドにより、フラッシュベースアドレス **0xC000000** にジャンプします。デバイスは、確認応答として **0x55** を送信した後、BSL モードを終了します。

4.3 Mode 2：iCache からユーザープログラムの実行

ヘッダーブロック

バイト 0	バイト 1	バイト 2 ... 14	バイト 15
ブロックタイプ (0x00)	Mode (0x02)	未使用	チェックサム

このコマンドにより、iCache ベースアドレス **0x8000000** にジャンプします。デバイスは、確認応答として **0x55** を送信した後、BSL モードを終了します。

4.4 Mode 3：フラッシュセクタの消去

ヘッダーブロック

バイト 0	バイト 1	バイト 2 ... 5	バイト 6 ... 9	バイト 10 ... 14	バイト 15
ブロックタイプ (0x00)	Mode (0x03)	セクタアドレス	セクタサイズ	未使用	チェックサム

- セクタアドレス (32 ビット)
  - –消去するフラッシュセクタのアドレス。アドレスは有効なセクタアドレスである必要があります (第 3 章を参照)。そうしないと、アドレスエラーが発生します。
  - –バイト 2 は最上位アドレスバイトを示します
  - –バイト 5 は、最下位アドレスバイトを示します。
- セクタサイズ (32 ビット)
  - –消去するフラッシュセクタのサイズ。サイズは有効なセクタサイズである必要があります (第 3 章を参照)。
  - –バイト 6 は最上位アドレスバイトを示します
  - –バイト 9 は、最下位アドレスバイトを示します。

デバイスは、確認応答として 0x55 を PC に送信するか、エラーコードを送信します。

4.5 Mode 4：フラッシュ保護ステータスの読み出し

ヘッダーブロック

バイト 0	バイト 1	バイト 2 ... 14	バイト 15
ブロックタイプ (0x00)	Mode (0x04)	未使用	チェックサム

- コマンドにはフラッシュ保護ステータスが必要です。デバイスは、**0x55** を「フラッシュ保護なし」として送信した後、または **0xF8** を「フラッシュ保護」として送信した後、BSL モードを終了します。

4.6 Mode 5：フラッシュの保護または保護解除

ヘッダーブロック

バイト 0	バイト 1	バイト 2...5	バイト 6...9	バイト 10	バイト 11...12	バイト 13...14	バイト 15
ブロックタイプ (0x00)	Mode (0x05)	ユーザーパスワード 1	ユーザーパスワード 2	フラッシュメモリモジュール	保護構成	未使用	チェックサム

ユーザーパスワード 1 (32 ビット)：最初のユーザーパスワード。バイト 2 は最上位バイトを示し、バイト 5 は最下位バイトを示します。

ユーザーパスワード 2 (32 ビット)：2 番目のユーザーパスワード。バイト 6 は最上位バイトを示し、バイト 9 は最下位バイトを示します。

フラッシュモード：予約済み

保護構成 (16 ビット)：保護するフラッシュセクタの選択。保護構成ワードの構造は次のとおりです。

ProtectioConfig ビットスキーム

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Sn = 0：セクタ n は保護されません。

Sn = 1：セクタ n が保護されます。

Note: セクタ n が存在しない場合は、ビット Sn を 0 に設定する必要があります。フラッシュのセクタ化の詳細については、第 3 章を参照してください。

フラッシュが保護されていない場合、このヘッダーブロックを送信した後に保護されます。同じパスワードでフラッシュ保護されたデバイスに送信された同じブロックは、フラッシュの保護を解除します。フラッシュ保護されたデバイスに送信されたすべての消去またはプログラムコマンドは、保護エラーを引き起こします。

Attention: フラッシュ保護／保護解除コマンドを送信した後、コマンドを有効にするためにデバイスをリセットする必要があります。

4.7 ホストへの応答コード

フラッシュローダープログラムは、応答コードを送信することにより、ブロックが正常に受信されたかどうか、および要求されたフラッシュルーチンが正常に実行されたかどうかをホストに通知します。

Table 6 応答コード

応答コード	内容
0x55	確認、エラーなし
0xFF	無効なブロックタイプ
0xFE	無効なモード

応答コード	内容
0xFD	チェックサムエラー
0xFC	無効なアドレス
0xFB	フラッシュ消去中にエラーが発生
0xFA	フラッシュプログラミング中にエラーが発生
0xF9	検証エラー
0xF8	保護エラー

5 ホスト PC プログラムの例

C++で開発された XMC4000_Bootloader HOST プログラムは、第 4 章で説明されている通信構造を使用します。

ファイル XMCload_API.cpp には、ASCLoader と直接通信するための API が含まれます。API には、次の関数が含まれます。

Table 7 API 関数

API 関数	説明
init_uart	PCCOM インターフェースを初期化
init_ASC_BSL	ASC BSL を初期化
send_loader	ASCLoader を送信
bl_send_header	ASC インターフェースを介してヘッダーブロックを送信
bl_send_data	ASC インターフェースを介してデータブロックを送信
bl_send_EOT	ASC インターフェースを介して EOT ブロックを送信
bl_erase_flash	PFlash セクタを消去
bl_download_pflash	PFlash にコードをダウンロード
make_flash_image	HEX ファイルから Flash イメージを作成

メインプログラム (XMCLoad.cpp) は、ASC を初期化し、ASCLoader をターゲットデバイスに送信します。

ユーザーは、ダウンロードする HEX ファイルを指定する必要があります。HEX ファイルの例(Blinky.hex)が提供されています。ユーザーコードは最初に Flash にダウンロードされ、次にユーザーは Flash と iCache の両方からダウンロードしたコードを実行できます。

- Figure 4 に示すように、フラッシュ消去手順は関数 bl_erase_flash()に実装されます。
- Figure 5 に示すように、Flash プログラミング手順は bl_download_pflash()に実装されます。

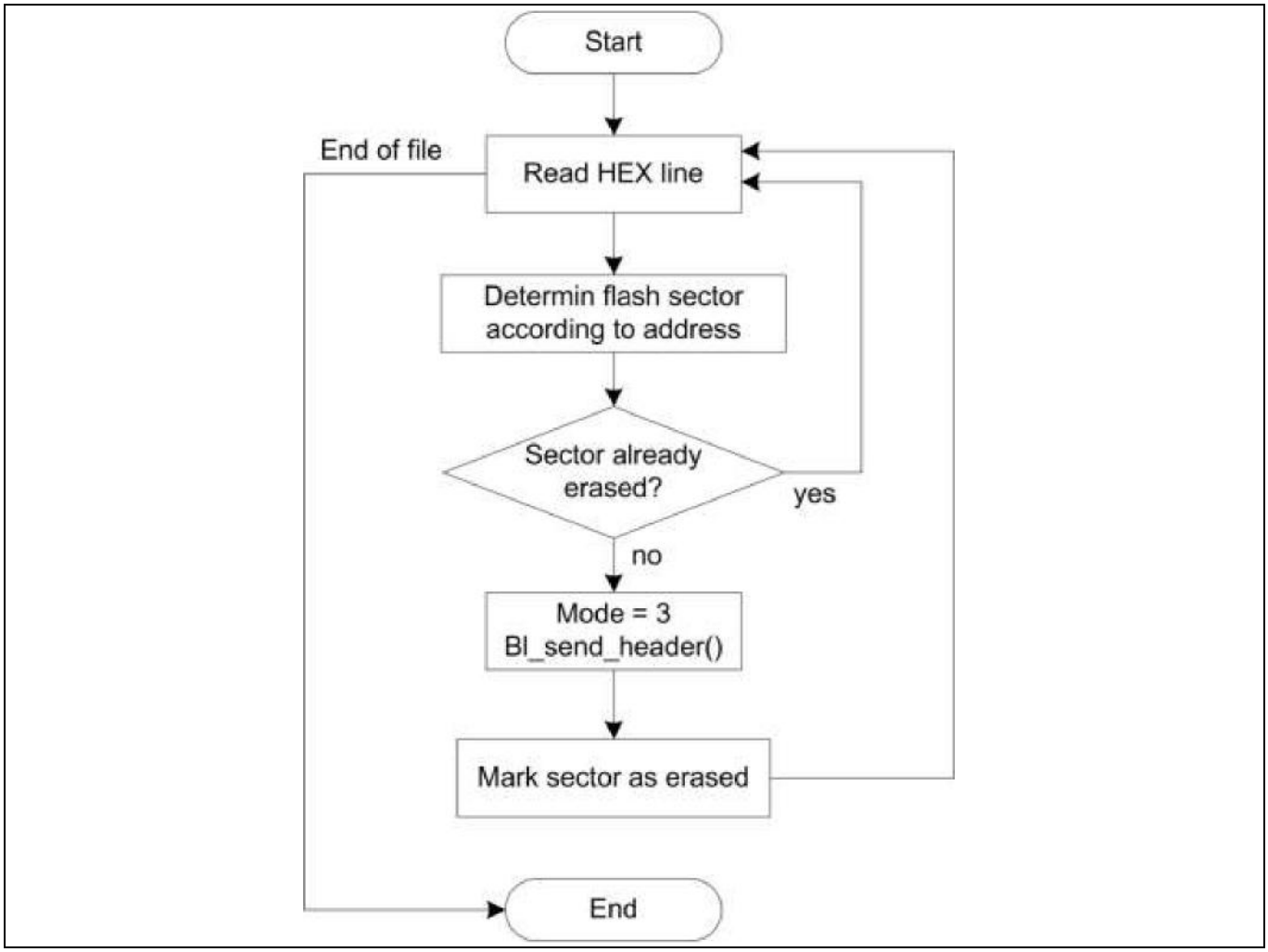


Figure 4　bl_erase_flash()に実装されたフラッシュ消去手順

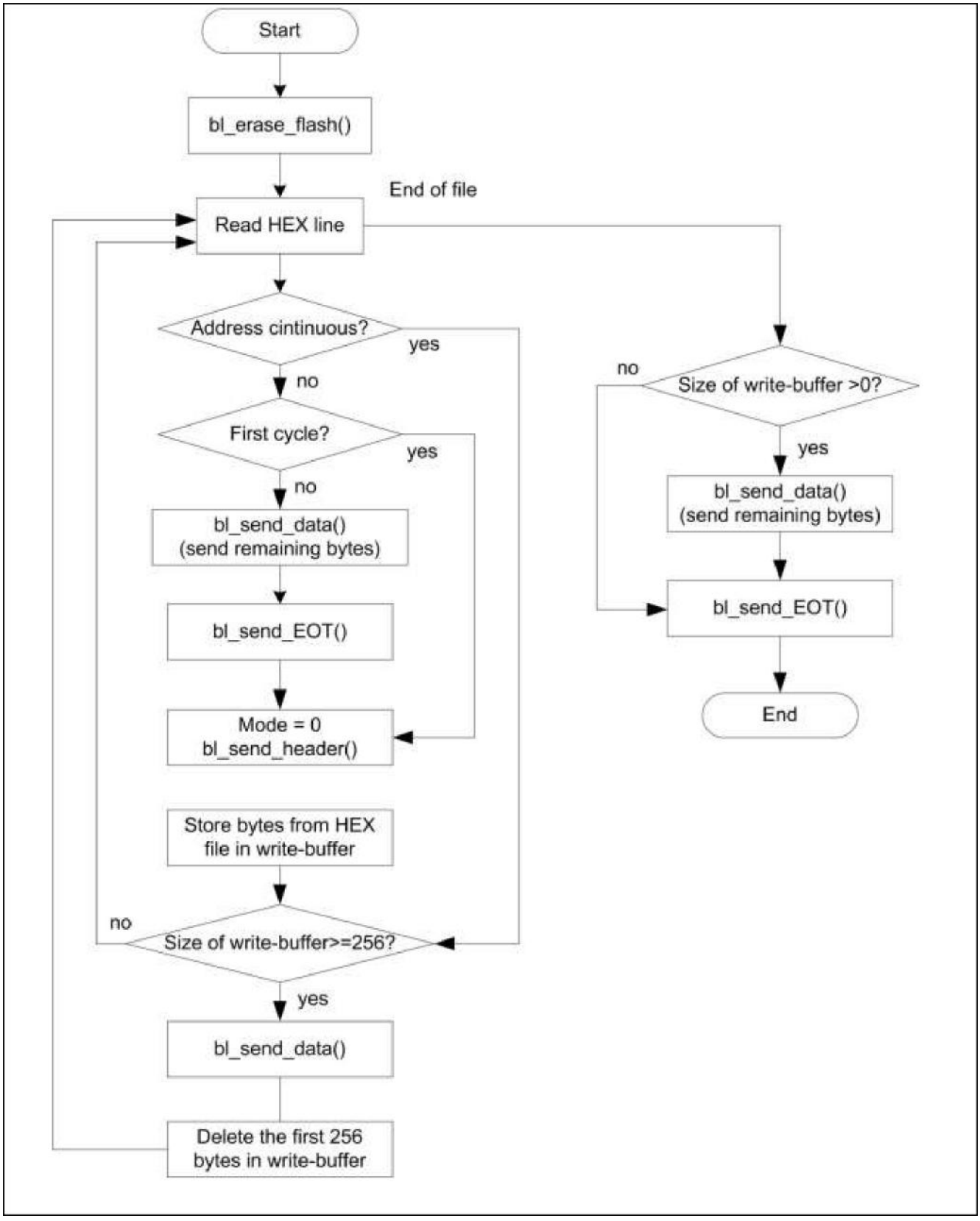


Figure 5　bl_download_pflash()に実装されたフラッシュプログラミング手順

6 デモンストレータの使用

サンプルプログラムは、InfineonXMC4x00 アプリケーションボードでテストされています。ユーザーはサンプルプログラムを使用して、ユーザーコード (hex ファイル形式) を Flash にダウンロードできます。ここでは、その方法について説明します。

6.1 ハードウェアの準備

XMC4x00 アプリケーションボードの ASC 出力ピンには 3.3v しかありませんが、PCASC 出力には通常 5v があります。ASC インターフェースを介して PC と XMC4000 ボード間の通信をセットアップするには、電圧アダプタ (Infineon Xspy-Adapter など) で電圧差を調整する必要があります。

XMC4x00 デバイスを搭載した Infineon アプリケーションボードをテストで使用する場合は、次のハードウェアセットアップが必要です。

- ASC ブートストラップロードモードでは、ボード上の DIP スイッチジャンプをジャンプ 1 (ON) およびジャンプ 2 (OFF) として設定してください。
- アプリケーションボード上の VCOM インターフェースを PCASC インターフェースに接続してください。

6.2 デモンストレータのファイル構成

次の図は、サンプルプログラムのファイル構造を示します。



Figure 6 サンプルプログラムのファイル構成

- このアプリケーションノートは、フォルダ .\App に含まれます。
- フォルダ .\DAV4、.\IAR4 および .\Keil は、異なるコンパイラを使用して生成されたプロジェクトです。
- フォルダ \XMC4500、\XMC4400 および \XMC4200 は、対応する BSL フラッシュローダープログラムが保存されているデバイスフォルダです。
- ASCLoader プロジェクトには、ASC ブートストラップローダープログラムが含まれます
- LED_Blinky プロジェクトは、LED 点滅のサンプルプロジェクトです。
- .\XMCLoad には、ホスト PC 用の Microsoft Visual C++ 2010 プロジェクトが含まれます。
- ASCLoader.hex ファイルとサンプル LEDBlinkyhex ファイルは .\XMCLoad\Release\XMC4x00 と .\XMCLoad\XMCLoad\XMC4x00 にそれぞれ保存されます。

Attention: XMC4400 ボード上の VCOM はピン P1.4 および P1.5 に接続されていないため、インフィニオン XMC4400 アプリケーションボード上の VCOM は ASCLoader で使用できません。

6.3 デモンストレータの実行

デモンストレータを起動する前に、Flash にダウンロードする必要のある hex ファイルを使用するデバイスに応じて、.\XMCLoad\Debug\XMC4x00 および .\XMCLoad\XMCLoad\XMC4x00 フォルダにコピーする必要があります。たとえば、XMC4500 デバイスを使用する場合、hex ファイルは次のようにコピーする必要があります。

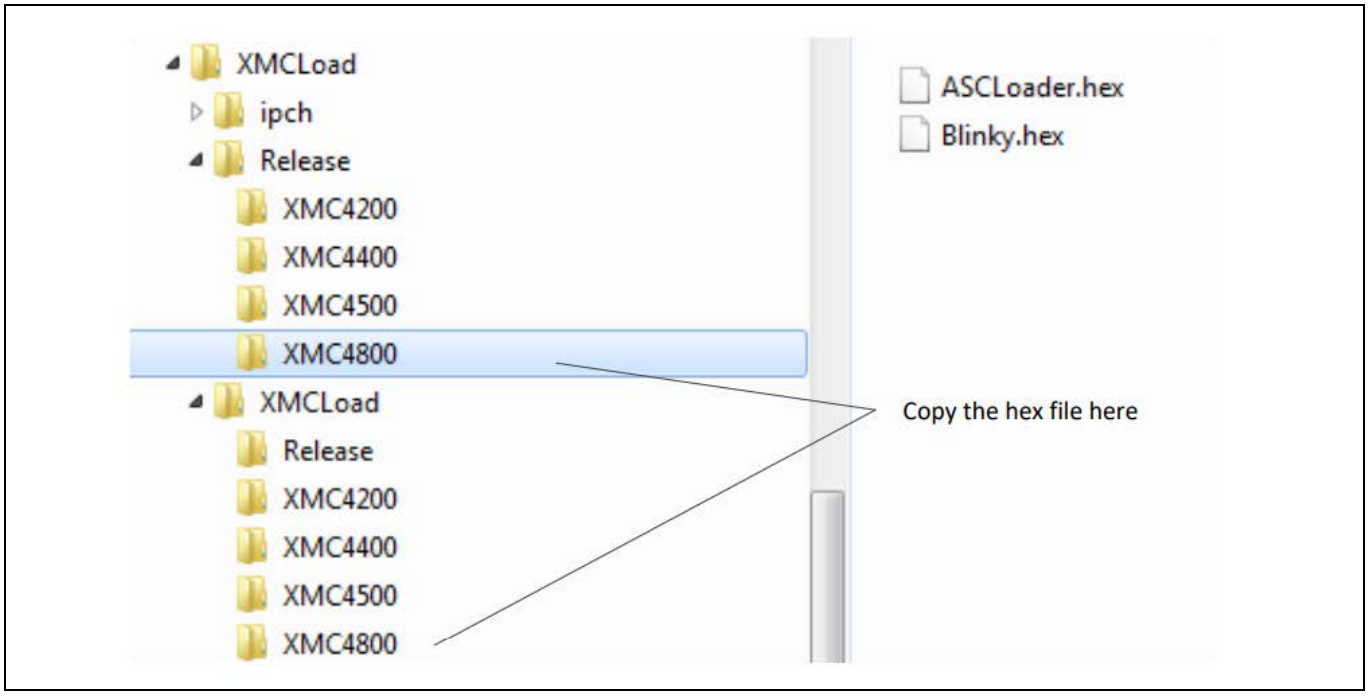


Figure 7 フラッシュするオブジェクトの hex ファイルの場所

デモンストレータを開始するには 2 つの方法があります。

1. .\XMCLoad\Release の下にあるファイル XMCLoad.exe をダブルクリックしてください。

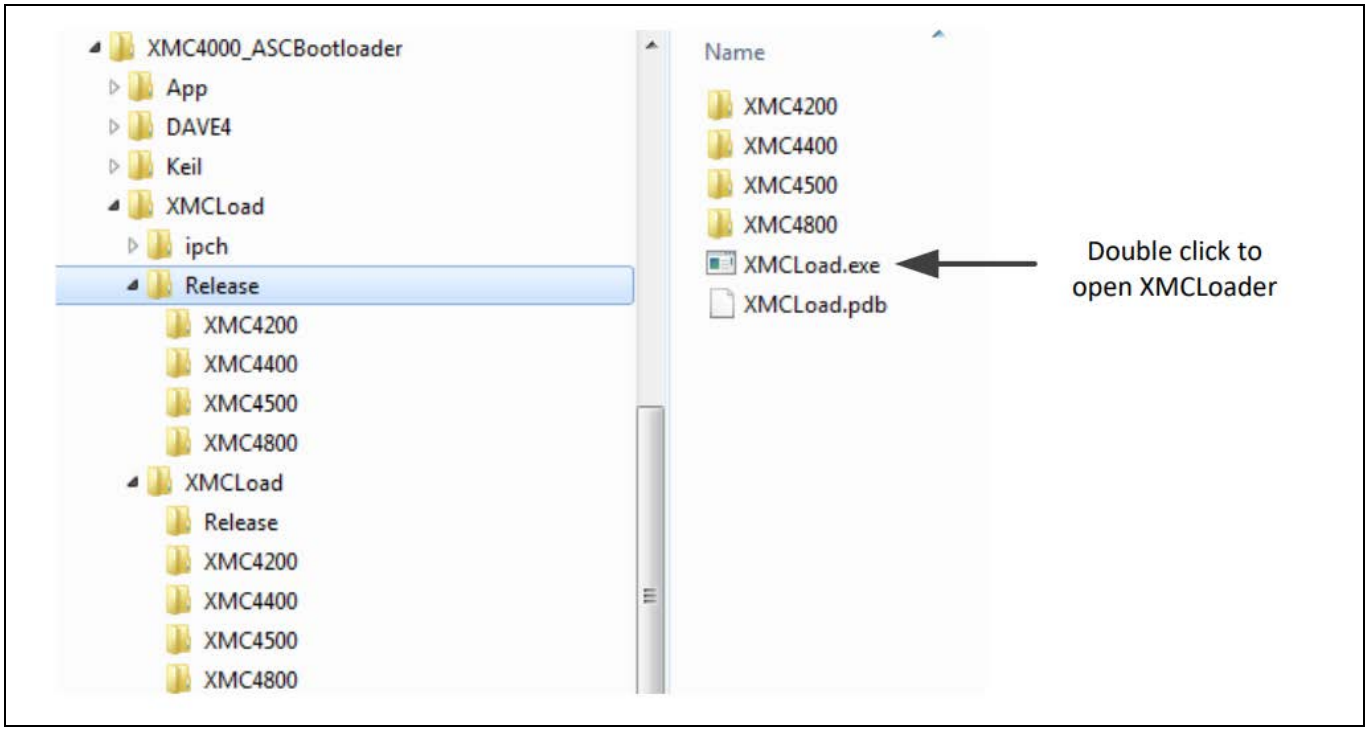


Figure 8　デモンストレータの直接開始

2. フォルダ .\ XMCLoad 内のファイル XMCLoad.sln ファイルをダブルクリックして、Microsoft Visual C++ プロジェクトを開きます。この AppNote のプロジェクトは、Microsoft Visual C++ 2010 を使用して開発されています。

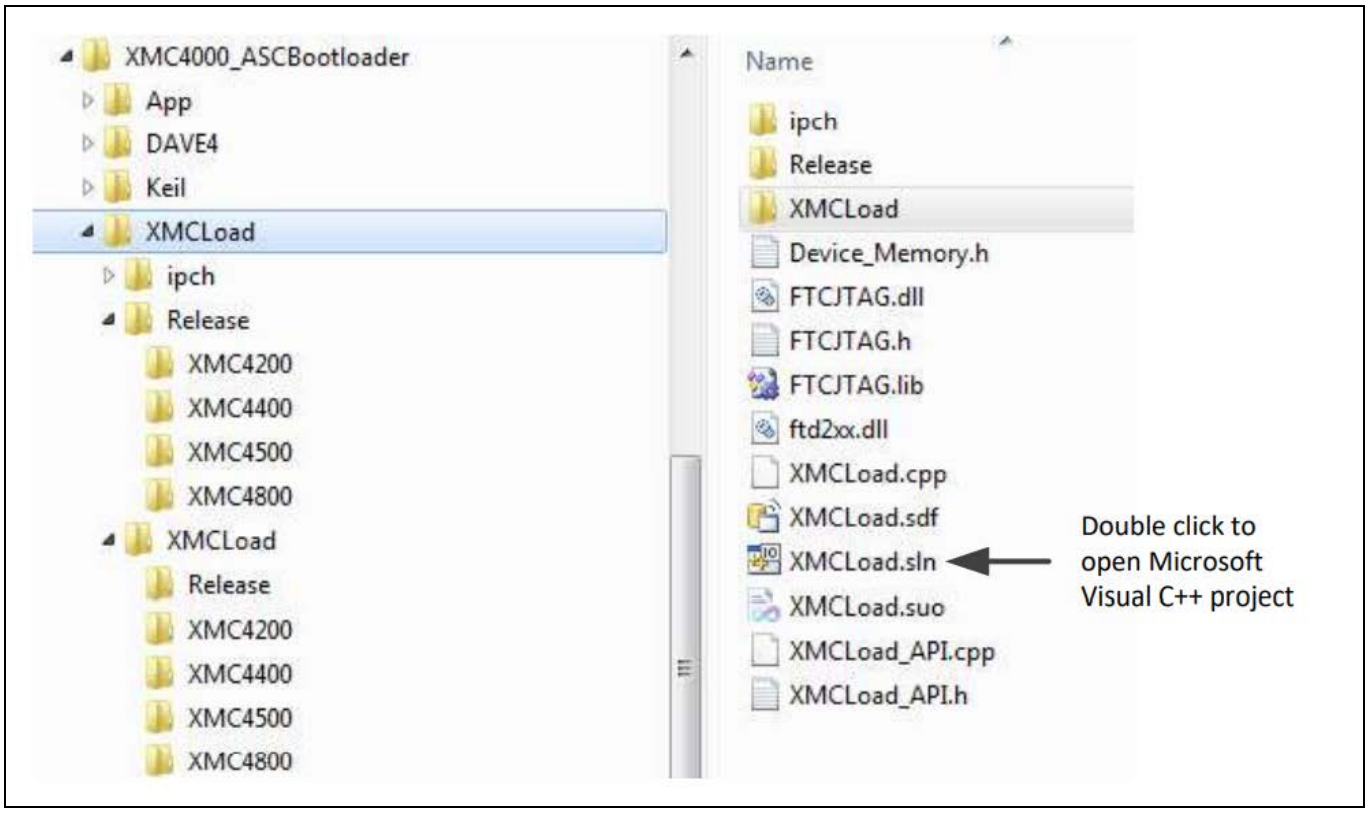


Figure 9　Microsoft Visual C++プロジェクトを使用したデモンストレータの開始

デモンストレータの使用

Microsoft Visual プロジェクトワークベンチでは、プロジェクトは「F5」キーから開始できます。デモンストレータを起動すると、次のウィンドウが表示されます。

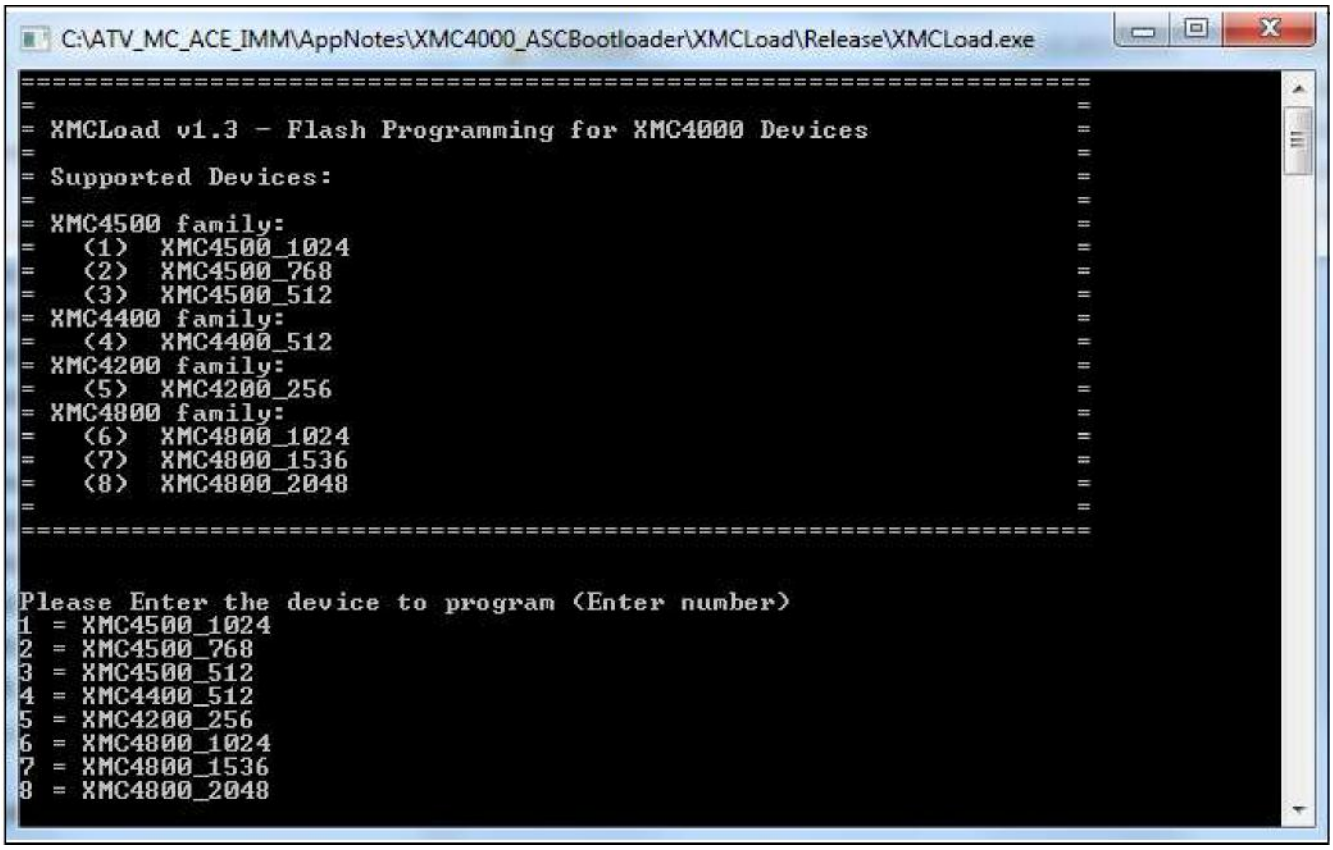


Figure 10 Microsoft Visual プロジェクトを使用した開始

ウィンドウの指示に従って、フラッシュプログラミングを終了するか、フラッシュ保護を設定するか、フラッシュ保護を取り外します。

Note: フラッシュにプログラムされる hex ファイル名は、ファイル拡張子を付けて完全に指定する必要があります。例：Blinky.hex。それ以外の場合、プログラムはファイル名を認識しません。フラッシュローダープログラムは、hex ファイル形式のみを受け入れます。さらに、ACLoader.hex は 4096 バイト未満であるため、4 バイトのアプリケーションの長さは 4096 で指定する必要があります。

hex ファイルが Flash にプログラムされた後、プログラムは Flash と iCache の両方から実行できます。

7 参考資料

Table 8 参考資料

文書	説明	場所
XMC4000 ユーザーズマニュアル	XMC4000 デバイスのユーザーズマニュアル	**http://www.infineon.com/XMC4000**

改訂履歴

Document version	Date of release	Description of changes
**	2021-12-15	本版は英語版 AP32235 ASC Bootstrap Loader for XMC4000 V1.4, 2019-10 について、CYPRESS DEVELOPER COMMUNITY の参画者によって日本語に翻訳されたドキュメントです。

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

Edition 2021-12-15
Published by
Infineon Technologies AG
81726 Munich, Germany

© 2021 Infineon Technologies AG.
All Rights Reserved.

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
AP32235

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。