

AVR1622 : XMEGA用TWIブートローダ

要点

- Atmel® AVR® XMEGA®ブートローダ
 - ・ TWIインターフェース使用
- 自己プログラミング用C-コード
- TWIGEN(AVR1624) Windows®ユーティリティで動くように設計
- フラッシュメモリの読み書き

1. 序説

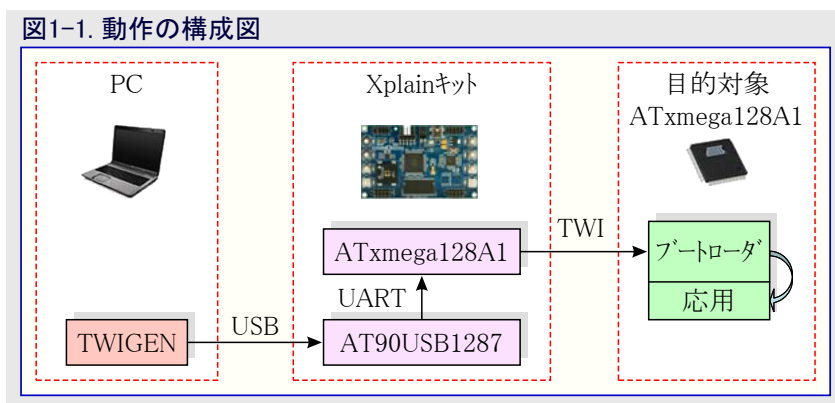
多くの電子機器設計が急速に進化するため、既に出荷または販売した製品の更新を可能にする必要が増えています。ブートローダを支援するマイクロコントローラは外部書き込み器の必要なしに、応用フラッシュ領域の更新を容易にし、現場で応用が更新されなければならない状況で大いに役立ちます。ブートローダはSPI, UART, TWI, イサネットなどのような様々なインターフェースを使うことができます。

この応用記述は応用領域を更新するのにXMEGAシステムデバイスのブートローダの使用法と、どうXMEGAを自己プログラミングに構成設定できるかを記述します。実演に使われる目的対象デバイスはAtmel STK®600上のAtmel ATxmega128A1です。それはUSBと(Atmel AVR1624応用記述からの)TWIGENを通してPCにインターフェースされるAtmel AVR Xplainキットに、TWIインターフェース経由で通信します。XplainキットはUSB-TWIブリッジとして働き、プログラミングに関して目的対象デバイスへTWIパケットを送ります。これは外部書き込み器の必要なしでのフラッシュプログラミングを許します。例のコードはSTK600目的対象基板上のATxmega128A1を使います。

XMEGAのプログラムメモリは、応用プログラム領域とブートプログラム領域の2つの領域に分けられます。応用フラッシュ領域を更新するためのブートローダプログラムはブートプログラム領域に置かれます。ブートローダは応用プログラムをダウンロード(取得/書き込み)するのにどのインターフェースも使うことができ、ここではTWIインターフェースを使います。一旦プログラミングされると、フラッシュメモリのブートと応用の両領域に異なる保護レベルを個別に適用することができます。AVRは広範囲な等級のメモリ保護を使用者に許す、独自の柔軟性を提供します。

自己プログラミングについての全般的な情報に関しては「AVR109:自己プログラミング」応用記述を参照してください。

図1-1. 動作の構成図



2. 準備と走行

本章はハードウェアの構成設定によって準備と走行に関する基本段階を簡単に片付けます。適切な情報と共に必要な構成と必要条件が記述されます。

2.1. ハードウェア構成設定

この項はTWIインターフェースを通すブートローダを用いるプログラミングに関するハードウェア構成設定のために従うべき手続きを説明します。

この応用記述で与えられるブートローダプログラムはPC上のユーザーインターフェースとしてTWIGENを使います。ブートローダプログラムはフラッシュ領域の読み込み/更新のための読み込みと書き込みのルーチンを実装します。



8ビット Atmel
マイクロコントローラ

応用記述

本書は一般の方々の便宜のため有志により作成されたもので、Atmel社とは無関係であることを御承知ください。しおりのはじめにでの内容にご注意ください。

Rev. 8437A-09/11, 8437AJ2-06/21

TWIがPCと直接的に通信するのに使うことができないので、橋渡し(ブリッジ)機能のため、間にAtmel AVR Xplainキットが使われます。Xplainキット内のAtmel AT90USB1287マイクロ コントローラはUSB-UART(CDC)応用でプログラミングされ、Xplainキット内のAtmel ATxmega128A1はUSART-TWIブリッジ応用でプログラミングされています。TWIGENソース コードと共に橋渡し機能はAtmelのAVR1624応用記述で入手可能です。

Xplainキット上のATxmega128A1のPD0とPD1(ポートD上のTWIインターフェースのSDAとSCL)をAtmel STK600上のATxmega128A1のPC0とPC1(ポートC上のTWIインターフェースのSDAとSCL)に接続してください。XplainキットからのGNDをAtmel STK600上のGNDに接続してください。XplainキットのATxmega128A1上の応用で内部プルアップが許可されるため、ここに外部プルアップは全く必要とされません。

STK600上で、スイッチが押されつつある時にプログラミング動作への移行をデバイスに許すためにPB0をSW0に接続してください。

正しい配線とソケットのカードでデバイスが装着されてポートピンがスイッチに接続されるために、Atmel AVR Studio®ヘルプで利用可能なSTK600使用者の手引きを参照してください。

2.2. ブリッジとしてのXplainキット設定

この項はSTK600内のATxmega128A1目的対象デバイスとPC間のUSB-UARTブリッジとしてXplainキットを働かせる手続きを説明します。'Xplain_USB.a90', 'at90usbxxx_cdc.inf'と'XplainSerialToI2CBootLoader Bridge.hex'はAtmelのAVR1624で入手可能です。

1. Atmel AVR JTAGICE mk IIをXplainキット上のJTAG USBに接続してください。
2. JTAGICE mk IIとXplainキットに通電してください。
3. AVR Studioを開始してください。
4. プログラミング ダイアログを開いてJTAGインターフェースを通してAtmel AT90USB1287へ接続してください(JTAGICE mk IIとXplainキットの両方が給電されているのを確実にしてください)。
5. Program(書き込み)タブを選択してください。'Flash(フラッシュ メモリ)'下で入力HEXファイルに関してXplain_USB.a90を含むフォルダを指示して同じファイルを書き込んで(プログラミングして)ください。
6. プログラミング ダイアログを閉じてください。
7. ソフトウェアのインストールを指示するポップアップが起きます。'一覧または指定位置からインストール(上級者)'を選び、AVR1624で提供された'at90usbxxx_cdc.inf'ファイルを指示してください。より多くの詳細についてはAVR1624を参照してください。
8. 次にJTAGICE mk IIをXplainキット上のJTAG&PDI XMEGAコネクタに接続してください。
9. プログラミング ダイアログを開いてJTAGインターフェースを通してAtmel ATxmega128A1を接続してください(JTAGICE mk IIとXplainキットの両方が給電されているのを確実にしてください)。
10. Programタブを選択してください。'Flash'下で入力HEXファイルに関してXplainSerialToI2CBootloaderBridge.hexを含むフォルダを指示して同じものを書き込んでください。

2.3. STK600上のATxmega128A1のプログラミング

この項はブートローダでAtmel STK600上のATxmega128A1とそれのヒューズをプログラミングする方法を説明します。デバッグなしでHEXファイルのプログラミングを望むなら、以下の手順に従ってください。

1. JTAGICE mk IIをSTK600上のJTAGヘッダに接続してください。
2. JTAGICE mk IIとSTK600を電源ONにしてください。
3. Atmel AVR Studioを開始してください。
4. プログラミング ダイアログを開いてJTAGインターフェースを通してATxmega128A1へ接続してください(JTAGICE mk IIとSTK600の両方が給電されているのを確実にしてください)。
5. Program(書き込み)タブを選択してください。'Flash(フラッシュ メモリ)'下で入力HEXファイルに関してTWI_BL.a90を含むフォルダを指示して同じファイルを書き込んでください。
6. Fuses(ヒューズ)タブを選択してください。BOOTRSTヒューズをチェックして書き込んでください。
7. プログラミング ダイアログを閉じてください。

故に、毎回デバイスは通電され、ポート領域に入って特定条件(ここではSTK600上でSW0スイッチが押される)を調べます。特定条件に出会うと自己プログラミング動作に移行してTWIインターフェース上のデータでデバイスのプログラミングを始めます。さもなければ応用領域に飛んで応用コードの実行を始めます。

2.4. デバッグ作業の開始

この項はデバッグ作業開始のために従うべき段階的な手続きを簡単に説明します。デバッグ時にデバイスがプログラミングされるため、再びフラッシュをプログラミングする必要はありません。これとてAVR用IAREWB(IAR Embedded Workbench®) 5.51が使われました。

1. IAREWBを開始してください。
2. Open(開く)⇒Workspace(作業空間)任意選択を選択してTWI_BL.ewwを含むフォルダを指示してください。
3. プロジェクトを構築して異常がないことを確認してください。
4. AVR Studioを開始してください。
5. TWI_BL.dbgファイルを用いて新しいプロジェクトを作成してください。

- デバイスとしてATxmega128A1を、基盤(Platform)としてJTAGICE mk IIを選択してください。
- AVR Studioでプログラミング ダイアログを開いてJTAGまたはPDIのインターフェースを通してATxmega128A1に接続してください(JTAGICE mk IIとSTK600の両方が給電されているのを確実にしてください)。
- Fuses(ヒューズ)タブを選択してください。BOOTRSTヒューズを設定してヒューズを書き込んでください。
- AVR Studioでデバッグ作業を始めてSTK600でSW0押下を保っている間に応用を走らせてください。
- プログラミング動作に於いて、プログラムはTWIGENを通してPCから受信した指令を順に渡すAtmel AVR Xplainキットから指令を受信します。

3. ブートローダとの通信

本章はハードウェア構成を通してブートローダと通信するための基本的な段階を説明します。必要な構成設定と必要条件が適切な情報と共に記述されます。

3.1. チップ消去

この項は'チップ消去'指令の実行方法を説明します。

ブートローダ領域内でデバイスを待たせるための段階の保証は「Atmel STK600上で外部RESETとSW0の両方を押し、先にRESETをその後SW0を開放する」で行われます。

ダブルクリックしてx128A1_chip_erase.batと呼ばれるバッチプログラムを走らせてください。これはデバイスとしてAtmel ATxmega128A1、従装置アドレスとして\$55、COMポート番号として14の設定を持ちます。

このバッチファイルはフラッシュを消去するために以下の指令を動かします。

```
twigen -e -a 3 0x00 0x00 0x00 -s 0x55 -p 14
```

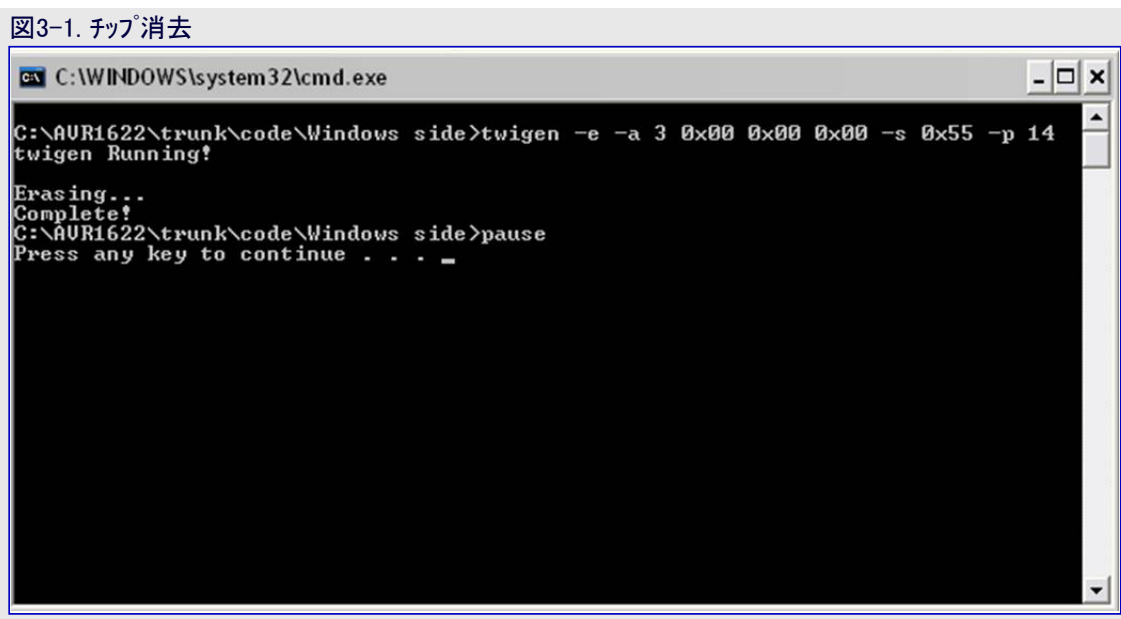


図3-1.で示されるコマンド行ウィンドウが現れるでしょう。バッチプログラムは以下の操作を実行します。

- 与えられたCOMポート番号を走査します。
- プログラミング動作へ移行します。
- チップ消去指令を発行します。
- プログラミング動作を抜け出します。

3.2. フラッシュへのファイル書き込み

この項はフラッシュにファイルを書き込む方法を説明します。

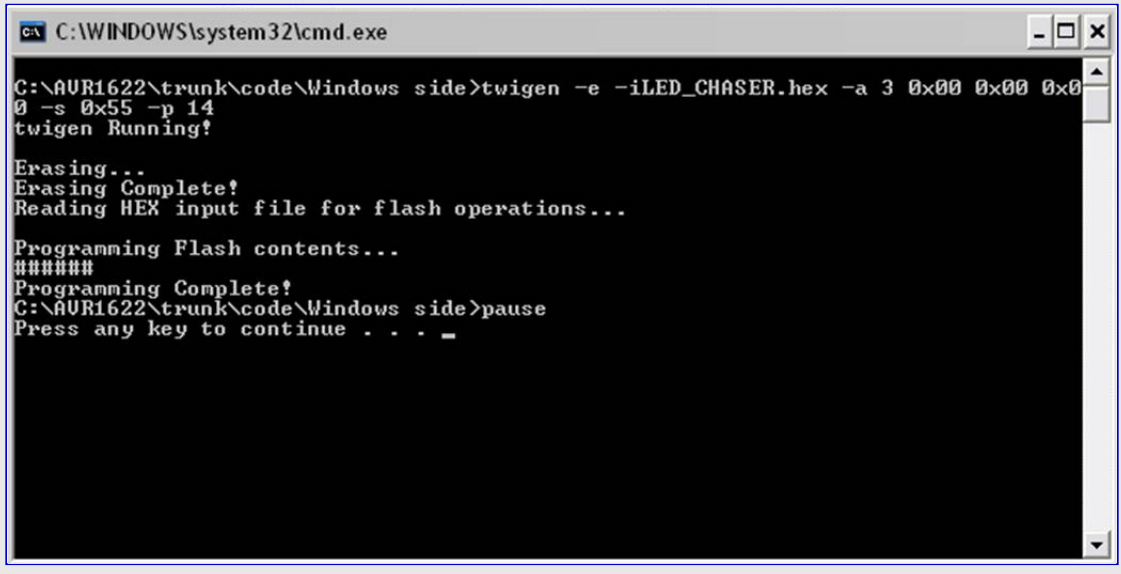
ブートローダ領域内でデバイスを待たせるための段階の保証は「Atmel STK600上で外部RESETとSW0の両方を押し、先にRESETをその後SW0を開放する」で行われます。

ダブルクリックしてx128A1_program_hex.batと呼ばれるバッチプログラムを走らせてください。これはデバイスとしてAtmel ATxmega128A1、書き込み開始アドレスとして\$000000、従装置アドレスとして\$55、COMポート番号として14の設定を持ちます。

このバッチファイルはフラッシュにHEXファイルを書き込むために以下の指令を動かします。

```
twigen -e -iLED_CHASER.hex -a 3 0x00 0x00 0x00 -s 0x55 -p 14
```

図3-2. フラッシュへのダウンロード(書き込み)



```

C:\WINDOWS\system32\cmd.exe
C:\AUR1622\trunk\code\Windows side>twigen -e -iLED_CHASER.hex -a 3 0x00 0x00 0x00
0 -s 0x55 -p 14
twigen Running!

Erasing...
Erasing Complete!
Reading HEX input file for flash operations...

Programming Flash contents...
#####
Programming Complete!
C:\AUR1622\trunk\code\Windows side>pause
Press any key to continue . . . _
  
```

図3-2.で示されるコマンド行ウィンドウが現れるでしょう。パッチプログラムは以下の操作を実行します。

1. 与えられたCOMポート番号を走査します。
2. 直前項で説明されたようにチップ消去を実行します。
3. プログラミング動作へ移行します。
4. 与えられたHEXファイルを通して解析します。
5. HEXファイル内のデータをフラッシュに書き込みます。
6. プログラミング動作を抜け出します。

3.3. 書き込まれた応用の走行

この応用記述で提供されるファームウェアはプログラミング動作抜け出し後にソフトウェアリセットを行います。

LED_CHASER.hexファイルはATxmega128A1でLED追跡を進行します。STK600上のポートDをLEDポートに接続してその出力を観察してください。

4. 異なるTWIの選択

本章は必要な場合に異なるTWIを選ぶ方法を説明します。この応用記述での試供ポートローダコードはTWICを使います。異なるTWIの使用が必要な場合は以下の位置で変更を行ってください。

1. flash_write_example.cで割り込み処理ルーチン内の割り込みベクタ名をTWIC_TWIS_vectから必要とされるTWIx_TWIS_vectに変更してください。
2. twi_slave_driver.cでTwi_SlaveInitializeDriver関数へ渡す時にTWI名をTWICから必要とされるTWIxに変更してください。

5. 異なるデバイスへの変更

本章はこの応用記述でのポートローダ例をAtmel ATxmega128A1以外の違うデバイスに変更する方法を説明します。

1. IARWWBを開始してください。
2. Open(開く)⇒Workspace(作業空間)任意選択を選択してTWI.BL.ewwを含むフォルダを指示してください。
3. Project(プロジェクト)⇒Options(任意選択)任意選択を選択してください。
4. 'General Options(全般任意選択)'に於いて、'Target(目的対象)'タブ下で'processor configuration(プロセッサ構成設定)'を--cpu=xm128a1,ATxmega128A1から望むデバイスに変更してください。
5. 'C/C++ compiler(C/C++コンパイラ)'に於いて'Preprocessor(プロセッサ)'タブ下で'Defined symbols(定義されたシンボル)'を_ATxmega128A1_から望むデバイスに変更してください。
6. link_bootloader.xclファイルに於いてデバイスに従ってデータシートを参照して詳細を変更してください。代わりにそれらの詳細に関してIAR™から対応するデバイスに対するリンクファイルを参照することもできます。IARのリンクファイルは\$TOOLKIT_DIR¥src¥template¥で利用可能です。
7. デバイスに応じてsp_driver.s90とsp_driver.hのファイル内のFLASH_PAGE_SIZEを変更してください。

注: Atmel AVR1624でのWindowsユーティリティ(TWIGEN)も、プロジェクト内で対応するページ容量で変更されて、再コンパイルされるべきです。

6. 推奨読み物

自己プログラミングとTWIインターフェースについての全体的な知識を得るため、以下の応用記述を読むことが推奨されます。

- [AVR109:自己プログラミング](#) – この応用記述はSPM命令を持つデバイスがどう自己プログラミングに構成設定されるかを説明します。例えばそれがAtmel tinyAVR®とAtmel megaAVR®のデバイス用に与えられるとしても、それは自己プログラミングについての一般的な情報を与えます。
- [AVR1316:XMEGA自己プログラミング](#) – この応用記述はAtmel AVR XMEGA自己プログラミングの基本的な機能を記述します。
- [AVR1308:XMEGA TWIの使い方](#) – この応用記述はXMEGAでTWIを構成設定して使う方法を記述します。
- [AVR1624:UART-TWIブリッジとしてのATxmega128A1 Xplainキットの使用](#) – この応用記述はUART-TWIブリッジとしてAtmel Xplainキットを作るための構成設定方法を記述します。

7. 目次

要点	1
1. 序説	1
2. 準備と走行	1
2.1. ハードウェア構成設定	1
2.2. ブリッジとしてのXplainキット設定	2
2.3. STK600上のATxmega128A1のプログラミング	2
2.4. デバッグ作業の開始	2
3. ブートローダとの通信	3
3.1. チップ消去	3
3.2. フラッシュへのファイル書き込み	3
3.3. 書き込まれた応用の走行	4
4. 異なるTWIの選択	4
5. 異なるデバイスへの変更	4
6. 推奨読み物	5
7. 目次	5



Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131
USA
TEL (+1)(408) 441-0311
FAX (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited

Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
TEL (+852) 2245-6100
FAX (+852) 2722-1369

Atmel Munich GmbH

Business Campus
Parking 4
D-85748 Garching b. Munich
GERMANY
TEL (+49) 89-31970-0
FAX (+49) 89-3194621

Atmel Japan

141-0032 東京都品川区
大崎1-6-4
新大崎勸業ビル 16F
アトメル ジャパン合同会社
TEL (+81)(3)-6417-0300
FAX (+81)(3)-6417-0370

© 2011 Atmel Corporation. 不許複製

Atmel[®]、Atmelロゴとそれらの組み合わせ、それとAVR[®]、AVR Studio[®]、megaAVR[®]、STK[®]、tinyAVR[®]、XMEGA[®]とその他はAtmel Corporationの登録商標または商標またはその付属物です。他の用語と製品名は一般的に他の商標です。

お断り: 本資料内の情報はAtmel製品と関連して提供されています。本資料またはAtmel製品の販売と関連して承諾される何れの知的所有権も禁反言あるいはその逆によって明示的または暗示的に承諾されるものではありません。Atmelのウェブサイトに表示する販売の条件とAtmelの定義での詳しい説明を除いて、商品性、特定目的に関する適合性、または適法性の暗黙保証に制限せず、Atmelはそれらを含むその製品に関連する暗示的、明示的または法令による如何なる保証も否認し、何ら責任がないと認識します。たとえAtmelがそのような損害賠償の可能性を進言されたとしても、本資料を使用できない、または使用以外で発生する(情報の損失、事業中断、または利益と損失に関する制限なしの損害賠償を含み)直接、間接、必然、偶然、特別、または付随して起こる如何なる損害賠償に対しても決してAtmelに責任がないでしょう。Atmelは本資料の内容の正確さまたは完全性に関して断言または保証を行わず、予告なしでいつでも製品内容と仕様の変更を行う権利を保留します。Atmelはここに含まれた情報を更新することに対してどんな公約も行いません。特に別の方法で提供されなければ、Atmel製品は車載応用に対して適当ではなく、使用されるべきではありません。Atmel製品は延命または生命維持を意図した応用での部品としての使用に対して意図、認定、または保証されません。

© HERO 2021.

本応用記述はAtmelのAVR1622応用記述(doc8437.pdf Rev.8437A-09/11)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意識されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には()内に英語表記や略称などを残す形で表記しています。

青字の部分はリンクとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。