応用記述

AVR911 : AVR公開ソース書き込み器

Atmel 8ビット マイクロ コントローラ

要点

- 公開ソースC++コード
- 機能単位設計
- Atmel® AVR Studio®のXMLファイルからデバイス情報読み込み
- Atmel AVR®109でのブートローダを支援
- Atmel AVR910での実装書き込み(ISP)を支援
- AVR Studioのコマンド行ツールと等価なコマンド行
- 他の書き込み器形式へ拡張可能
- 他の通信チャネル、例えばUSBへ拡張可能

序説

Atmel AVR公開ソース書き込み器(AVROSP:AVR Open Source Programmer)はAVR Studioに含まれるAVRprogツールと等価なAVR書き込み器応用です。それはAVR Studio内の他のコマンド行ツールと同じ構文を使うコマンド行ツールです。

公開ソース コードとそれの機能単位設計は他の基盤への応用の移転と、他の書き込み器形式と通信チャネルに対する支援追加を容易にします。現在、AVROSPは標準的なPCシリアル ポートを通してAtmelのAVR109とAVR910の応用記述で記述される書き込み器を支援します。この応用記述はより多くの支援を追加する方法を記述します。

AVROSPはIntel® HEXファイルを読み書きし、必要な装置パラメータを得るのに既存のAVR Studioインストールを使うことができます。これはAVROSPがAVR Studioによって支援される全てのデバイスを自動的に支援することを意味します。AVR Studioインストールが最新を保つことを除き、更なるAVRデバイスに対して更新は全く必要とされません。

Rev. 2568B-10/2012, 2568BJ2-05/2021

目次

1. 背景と理屈

使用者の視点から、Atmel AVRﾃﾞﾊﾞｲｽのﾌﾟﾛｸﾞﾗﾐﾝｸﾞは基本的に次の段階から成ります。書き込み器をﾃﾞﾊﾞｲｽに配線し、書かれるべき2進ﾌｧｲﾙを用意して、最後に使う特定の書き込み器用の応用を開始します。これらの段階は基本的に書き込み器形式と無関係に同じです。AVROS P応用はこの手続きを1つの応用内への標準化を試み、それによって各書き込み器形式に関する異なる構文と使い方を持つ各種応用の必要を無くします。ﾌﾞｰﾄﾛｰﾀﾞ、実装書き込み器、または第三者の書き込み器のどれを使うにしても、手続きは基本的に同じです。AVROSPはﾌﾟﾛｸﾞﾗﾐﾝｸﾞ操作に対して一貫したｲﾝﾀｰﾌｪｰｽを与えます。

AtmelのAVR109とAVR910の応用記述はﾌﾞｰﾄﾛｰﾀﾞと実装書き込み(ISP)を記述します。それらはﾁｯﾌﾟ上のUARTを通すﾌﾞｰﾄﾛｰﾀﾞとAT90S1200に基づく書き込み器を通すISPの両方を同じ操作で支援します。ﾌﾞｰﾄﾛｰﾀﾞは特定ﾃﾞﾊﾞｲｽに対してｺﾝﾊﾟｲﾙされ、当然にそれのﾌﾟﾛｸﾞﾗﾐﾝｸﾞを支援します。ISPはAtmel ATmega163以降のﾃﾞﾊﾞｲｽを直接支援するように更新されませんが、実装書き込みを支援する全てのﾃﾞﾊﾞｲｽのﾌﾟﾛｸﾞﾗﾐﾝｸﾞに使うことができる"万能命令"を持ちます。AVROSPは両方の書き込み器を支援します。それは書き込み器の識票を読んで書き込み器との通信にどの命令を使うかを決めます。従って書き込み器が"書き込み器ID読み込み"命令に正しく応用するなら、使用者は書き込み器形式を指定する必要がありません。規約のより多くの情報についてはAVR109とAVR910を参照してください。

ﾌﾟﾛｸﾞﾗﾐﾝｸﾞのために必要なﾃﾞﾊﾞｲｽ情報(ﾒﾓﾘ量、施錠とﾋｭｰｽﾞのﾋﾞｯﾄなど)の最小の組はAtmel AVR Studioｲﾝｽﾄｰﾙでの XML形式ﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙで入手可能です。AVROSPはそれらのﾌｧｲﾙから必要な情報を読みます。AVR Studioがｲﾝｽﾄｰﾙされていない場合、例えば、AVROSPが製品書き込みで使われる、または他の基盤(例えば、Linux®)に移転された場合、ﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙを未だ使用することができます。応用は最初に現在のﾃﾞｨﾚｸﾄﾘ、AVROSPﾎｰﾑ ﾃﾞｨﾚｸﾄﾘ、PATH環境変数で指定されたﾃﾞｨﾚｸﾄﾘ、そして最後にAVR Studioｲﾝｽﾄｰﾙ ﾃﾞｨﾚｸﾄﾘを検索します。従ってﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙはPATH内ﾃﾞｨﾚｸﾄﾘに複写することができ、AVR Studioのｲﾝｽﾄｰﾙは全く必要ありません。

元のﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙは非常に大きく、それらを解析するのにいくらかの時間がかかります。従ってAVROSPは重要なﾊﾟﾗﾒｰﾀだけを含む小さなXMLﾌｧｲﾙを作成し、AVROSPのﾎｰﾑ ﾃﾞｨﾚｸﾄﾘにそれを格納します。例えば、AVR Studioの更新によって、元のﾌｧｲﾙに更新が行われた場合、ｷｬｯｼｭされたXMLﾌｧｲﾙはそれらを再生成するよう、AVROSPに告げるために削除されるべきです。

現在、ﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙは各ﾃﾞﾊﾞｲｽのﾌﾟﾛｸﾞﾗﾐﾝｸﾞ算法の仕様のどんな情報も含みません。従ってAVROSPのISP単位部は最近の殆どのAVRﾃﾞﾊﾞｲｽで使われる算法を実装します。このため、僅かに異なる算法を使ういくつかのﾃﾞﾊﾞｲｽの実装書き込みは支援されません。これは次のAtmelﾃﾞﾊﾞｲｽに適用します。ATtiny26とATtiny2313。

AVROSPのﾌﾞｰﾄﾛｰﾀﾞ単位部はﾌﾞｰﾄﾛｰﾀﾞ能力を持つ全てのﾃﾞﾊﾞｲｽを支援します。

注: Atmel ATtiny28LはISPやﾌﾞｰﾄﾛｰﾀﾞ ﾌﾟﾛｸﾞﾗﾐﾝｸﾞを支援せず、AVROSPによって支援されません。使用者は他の書き込み器、例えば、高電圧直列書き込み器を支援するようにｺｰﾄﾞを独自化することができます。

(訳補) 上記ﾃﾞﾊﾞｲｽ名は原書改訂時点の量産ﾃﾞﾊﾞｲｽだけを対象とし、打ち切り製品などは含まれません。

2. 即時開始の情報

本章は応用ｺｰﾄﾞを変更または独自化する必要が全くない場合に、いきなり素早く動かすのに必要な段階を記述します。

avrosp.exe実行ﾌｧｲﾙはAVROSPを使うのに必要とされる唯一のﾌｧｲﾙです。それはこの応用記述と共に来るavr911.zipﾌｧｲﾙ内に含まれます。ZIPﾌｧｲﾙは完全なｿｰｽ ｺｰﾄﾞとAVR Studioｲﾝｽﾄｰﾙからのﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙの複製も含みます。

新しいﾃﾞｨﾚｸﾄﾘへ実行可能ﾌｧｲﾙを複写してPATH環境変数へそのﾃﾞｨﾚｸﾄﾘ名を追加してください。Atmel AVR Studioのｲﾝｽﾄｰﾙを望まない場合、(元ﾌｧｲﾙの上書きからｷｬｯｼｭされたﾌｧｲﾙを保護するために)副ﾃﾞｨﾚｸﾄﾘへXMLﾌｧｲﾙを複写し、PATHへそれも追加してください。今や全ては使用準備が整っているべきです。

注: 通信ﾎﾟｰﾄ設定(ﾎﾞｰﾚｰﾄ、ﾊﾟﾘﾃｨ制御など)はAVROSPを使う前に手動で設定されなければなりません。例えば、115200bps、ﾊﾟﾘﾃｨ制御なし、8ﾋﾞｯﾄ ﾃﾞｰﾀを持つCOM1を通して通信するﾌﾞｰﾄﾛｰﾀﾞとでAVROSPを使うには、以下のDOS命令を動かしてください。

```
mode com1 baud=115200 parity=n data=8
```

3. ｺﾏﾝﾄﾞ行構文

全てのﾊﾟﾗﾒｰﾀは-、1つ以上の文字、そして多数の任意選択値で始まらなければなりません。-、文字または任意選択値の間に空白は有ることができません。ﾊﾟﾗﾒｰﾀの順番は重要ではありません。ﾊﾟﾗﾒｰﾀがかち合う場合、例えば、通信に対してCOM1とCOM2の両方を選択する場合、常に最後のﾊﾟﾗﾒｰﾀが勘定されます。支援されるｺﾏﾝﾄﾞ行ﾊﾟﾗﾒｰﾀは表3-1.で一覧にされます。

表3-1. ｺﾏﾝﾄﾞ行ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ	説明
-d<名前>	ﾃﾞﾊﾞｲｽ名。ﾃﾞﾊﾞｲｽをﾌﾟﾛｸﾞﾗﾐﾝｸﾞする時に加えられなければなりません。
-if<入力ﾌｧｲﾙ>	ﾌﾗｯｼｭ入力ﾌｧｲﾙ名。ﾌﾗｯｼｭ ﾒﾓﾘの書き込みと検証に必要。ﾌｧｲﾙ形式はIntel拡張HEXです。
-ie<入力ﾌｧｲﾙ>	EEPROM入力ﾌｧｲﾙ名。EEPROMの書き込みと検証に必要。ﾌｧｲﾙ形式はIntel拡張HEXです。
-of<出力ﾌｧｲﾙ>	ﾌﾗｯｼｭ出力ﾌｧｲﾙ名。ﾌﾗｯｼｭ ﾒﾓﾘの読み出しに必要。ﾌｧｲﾙ形式はIntel拡張HEXです。
-oe<出力ﾌｧｲﾙ>	EEPROM出力ﾌｧｲﾙ名。EEPROMの読み出しに必要。ﾌｧｲﾙ形式はIntel拡張HEXです。
-s	識票ﾊﾞｲﾄ読み込み。
-O<ｱﾄﾞﾚｽ>	ﾃﾞﾊﾞｲｽから発振校正ﾊﾞｲﾄ読み込み。ｱﾄﾞﾚｽは任意選択。
-O#<値>	使用者定義発振校正値。-O<ｱﾄﾞﾚｽ>でﾃﾞﾊﾞｲｽから読む代わりに独自校正値を提供するのにこれを使ってください。
-Sf<ｱﾄﾞﾚｽ>	発振校正ﾊﾞｲﾄをﾌﾗｯｼｭ ﾒﾓﾘに書きます。ｱﾄﾞﾚｽはﾊﾞｲﾄ ｱﾄﾞﾚｽです。
-Se<ｱﾄﾞﾚｽ>	発振校正ﾊﾞｲﾄをEEPROMに書きます。ｱﾄﾞﾚｽはﾊﾞｲﾄ ｱﾄﾞﾚｽです。
-e	ﾃﾞﾊﾞｲｽ消去。ﾃﾞﾊﾞｲｽは他のどんなﾌﾟﾛｸﾞﾗﾐﾝｸﾞ処理にも先立って消去されます。
-p<t>	ﾃﾞﾊﾞｲｽ書き込み。tをﾌﾗｯｼｭに対してf、EEPROMに対してe、両方に対してbを設定してください。対応する入力ﾌｧｲﾙが必要です。
-r<t>	ﾃﾞﾊﾞｲｽ読み出し。tをﾌﾗｯｼｭに対してf、EEPROMに対してe、両方に対してbを設定してください。対応する出力ﾌｧｲﾙが必要です。
-v<t>	ﾃﾞﾊﾞｲｽ検証。tをﾌﾗｯｼｭに対してf、EEPROMに対してe、両方に対してbを設定。-p<t>と共にまたはこれだけで使うことができます。対応する入力ﾌｧｲﾙが必要です。
-l<値>	施錠ﾊﾞｲﾄ設定。値は8ﾋﾞｯﾄの16進値です。
-L<値>	施錠ﾊﾞｲﾄ検証。値は対照して検証するための8ﾋﾞｯﾄの16進値です。
-y	施錠ﾊﾞｲﾄ読み返し。
-f<値>	ﾋｭｰｽﾞ ﾊﾞｲﾄ設定。値は上位と下位のﾋｭｰｽﾞ ﾊﾞｲﾄ用の設定を記述する16ﾋﾞｯﾄの16進値です。
-E<値>	拡張ﾋｭｰｽﾞ ﾊﾞｲﾄ設定。値は拡張ﾋｭｰｽﾞ ﾊﾞｲﾄ設定を記述する8ﾋﾞｯﾄの16進値です。
-F<値>	ﾋｭｰｽﾞ ﾊﾞｲﾄ検証。値は対照して検証するための16ﾋﾞｯﾄの16進値です。
-G<値>	拡張ﾋｭｰｽﾞ ﾊﾞｲﾄ検証。値は対照して検証するための8ﾋﾞｯﾄの16進値です。
-q	ﾋｭｰｽﾞ ﾊﾞｲﾄ読み返し。
-x<値>	値(00～FF)で未指定位置を満たす。既定は入力ﾌｧｲﾙで指定されない位置を書かないことです。
-af<開始>,<終了>	ﾌﾗｯｼｭ ｱﾄﾞﾚｽ範囲。操作のｱﾄﾞﾚｽ範囲を指定します。既定はﾌﾗｯｼｭ全体です。16進でのﾊﾞｲﾄ ｱﾄﾞﾚｽです。
-ae<開始>,<終了>	EEPROMｱﾄﾞﾚｽ範囲。操作のｱﾄﾞﾚｽ範囲を指定します。既定はEEPROM全体です。16進でのﾊﾞｲﾄ ｱﾄﾞﾚｽです。
-c<ﾎﾟｰﾄ>	COM1～COM8の通信ﾎﾟｰﾄ選択。このﾊﾟﾗﾒｰﾀが省略された場合、ﾌﾟﾛｸﾞﾗﾑは書き込み器に関してCOMﾎﾟｰﾄを走査します。
-b<t>	取り付けた書き込み器の改訂番号取得。tをﾊｰﾄﾞｳｪｱに対してh、ｿﾌﾄｳｪｱに対してsを設定。
-g	静寂動作。画面に全く出力しません。
-z	進捗指示部なし。例えば、記録目的のためにﾌｧｲﾙにﾊﾟｲﾌﾟ出力する場合に指示部に使われる文字を避けるためにこの任意選択を使ってください。
-Y<ｱﾄﾞﾚｽ>	ATtiny4/5/9/10/20/40ﾃﾞﾊﾞｲｽの内部RC発振器校正に使用。AVR057応用記述を参照してください。
-h	ﾍﾙﾌﾟ情報(他の全ての設定を無効化)。
-?	-hと同じ。

いくつかの例は以下のとおりです。

```
avrosp -dATmega128 -pf -vf -ifprogram.hex -e
```

上の例はAtmel ATmega128ﾃﾞﾊﾞｲｽに取り付けるため、最初にﾒﾓﾘ内容全体を消去してその後にprogram.hexに含まれるﾃﾞｰﾀを書いて検証します。

```
avrosp -dATmega32 -re -oedump.hex -ae0,ff -cCOM2
```

上の例はAtmel ATmega32のEEROMの最初の256ﾊﾞｲﾄをdump.hexﾌｧｲﾙに読みます。COM2だけが使われます。

```
avrosp -dATmega64 -O#a0 -Se0 -lc0
```

上の例は独自の発振校正値の$A0をEEPROMｱﾄﾞﾚｽの$0内に書いてその後に施錠ﾋﾞｯﾄに$C0を書くことによってAtmel ATmega64のﾒﾓﾘを保護します。

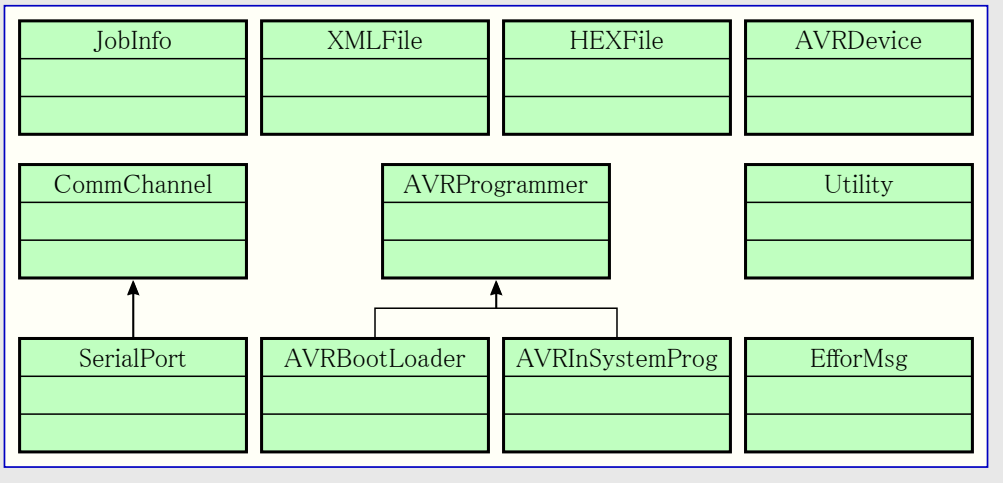
注: ﾋｭｰｽﾞ ﾋﾞｯﾄのﾌﾟﾛｸﾞﾗﾐﾝｸﾞ時、ﾋﾞｯﾄ模様はﾃﾞﾊﾞｲｽに対して有効なﾋｭｰｽﾞ設定かを検査されません。これが操作不能にし得るため、無効なﾋｭｰｽﾞ設定を書かないように注意が払われるべきです。高電圧ﾌﾟﾛｸﾞﾗﾐﾝｸﾞだけがこのような状況から回復するための方法になり得ます。

4. 実装

本章は読者がｵﾌﾞｼﾞｪｸﾄ志向ﾌﾟﾛｸﾞﾗﾐﾝｸﾞの概念と特にC++ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語の或る程度の知識を持つと仮定します。

ｿｰｽ ｺｰﾄﾞは使用者が応用を変更や強化して彼らがしたいようにそれを再分配することができることを意味する、常に自由です。ﾌﾘｰｿﾌﾄｳｪｱのより多くの情報は以下のURL:http://www.gnu.org/philosophy/free-sw.htmlで入手可能です。

図4-1. AVROSPｸﾗｽ構成図



最上部ﾚﾍﾞﾙの作業の殆どはJobInfoｸﾗｽでｶﾌﾟｾﾙ化されます。それはXMLとHEXのﾌｧｲﾙの読み書きとﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙからﾃﾞﾊﾞｲｽ情報を抽出するために、XMLFile、HEXFile、それとAVRDeviceのｸﾗｽのｵﾌﾞｼﾞｪｸﾄを使います。

応用全体を通してUtilityとErrorMsgの2つのﾍﾙﾊﾟｰ ｸﾗｽが使われます。

書き込み器と通信するJobInfoの部分は使う通信ﾁｬﾈﾙの種類が何かを知る必要はありません。それは命令行を復号して必要とされる派生ｸﾗｽ、例えば、SerialPortｸﾗｽのｲﾝｽﾀﾝｽを作成します。ｺｰﾄﾞの残りは単に標準化されたCommChannel親ｸﾗｽを通して動きます。現在、PCのCOMﾎﾟｰﾄ用のｸﾗｽだけが実装されますが、例えば、USBやTCP/IPの通信を使うためにCommChannel基底ｸﾗｽから特殊化したｸﾗｽを引き出し、命令行解析部にこのﾁｬﾈﾙ形式に対する検査を追加することができます。

同じ方法が書き込み器形式に使われます。書き込み器上で操作するｺｰﾄﾞはどの形式の書き込み器が取り付けられているかを知る必要がありません。JobInfoｸﾗｽは書き込み器ID文字列を取得して特定の書き込み器に対して適切なｵﾌﾞｼﾞｪｸﾄを作成します。ｺｰﾄﾞの残りは標準化されたAVRProgrammerｲﾝﾀｰﾌｪｰｽを通して操作します。現在、Atmel AVR109応用記述で記述されるﾌﾞｰﾄﾛｰﾀﾞとAtmel AVR910応用記述で記述される実装書き込み器用のｸﾗｽだけが実装されます。けれども、AVRProgrammer基底ｸﾗｽからあなた自身の特殊化した書き込み器を引き出して、JobInfoのID文字列復号部分でそれに対する検査を追加することができます。

この設計は応用を非常に柔軟にします。他の通信ﾁｬﾈﾙと書き込み器形式での将来の拡張は容易な作業です。

4.1. ｸﾗｽ説明

公開ｲﾝﾀｰﾌｪｰｽ ﾒｿｯﾄﾞだけがここで記述されます。読者は様々なｸﾗｽの内部動作の詳細な情報について注釈されたｺｰﾄﾞを参照すべきです。各ｸﾗｽはｸﾗｽの目的に対する簡単な紹介で記述され、その後にそれの公開ﾒｿｯﾄﾞの各々が戻り形式、目的ﾊﾟﾗﾒｰﾀで記述されます。

AVRDevice

このｸﾗｽはﾒﾓﾘの大きさと識票ﾊﾞｲﾄのような、現在ﾌﾟﾛｸﾞﾗﾐﾝｸﾞされたﾃﾞﾊﾞｲｽに関連したﾊﾟﾗﾒｰﾀ用のｺﾝﾃﾅを提供します。このｸﾗｽはAtmel AVR Studioｲﾝｽﾄｰﾙからﾃﾞﾊﾞｲｽ情報を取得するための機能も含みます。

AVRDevice

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これは、それから情報を取得するためのﾃﾞﾊﾞｲｽの名前である、1つの文字列ﾊﾟﾗﾒｰﾀを取ります。このﾊﾟﾗﾒｰﾀは自動的に取得されません。ｸﾗｽはAVR Studioに関する情報を読むためのﾒｿｯﾄﾞを提供しますが、派生ｸﾗｽはこの情報を得る他の方法を実装することができます。

~AVRDevice

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これは現在何の機能も持たず、将来の拡張に対する単なる代替物です。

readParametersFromAVRStudio

これは必要なXMLﾌｧｲﾙに関して、現在のﾃﾞｨﾚｸﾄﾘ、応用のﾎｰﾑ ﾃﾞｨﾚｸﾄﾘ、PATH環境変数でのﾃﾞｨﾚｸﾄﾘ、それと利用可能ならばAVR Studioｲﾝｽﾄｰﾙを検索します。AVR Studioｲﾝｽﾄｰﾙのﾊﾟｽを得るために、Windows®のﾚｼﾞｽﾄﾘ ﾃﾞｰﾀﾍﾞｰｽは“HKEY_LOCAL_MACHINE¥Software¥Atmel¥AVRTools¥AVRToolsPath”と名付けられた鍵に関して問われます。ﾒｿｯﾄﾞはその後にXMLﾌｧｲﾙを解析して必要な情報を取得します。ﾌｧｲﾙが見つからない、またはﾌｧｲﾙが異常を含む場合、例外が投じられます。ﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀとして検索ﾊﾟｽの一覧を取り、値を返しません。

getFlashSize

これはﾌﾗｯｼｭ ﾒﾓﾘ容量ﾊﾟﾗﾒｰﾀのためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾊﾟﾗﾒｰﾀを取らず、ﾌﾗｯｼｭ ﾊﾞｲﾄ数を示すlong値を返します。

getEEPROMSize

これはEEPROM容量ﾊﾟﾗﾒｰﾀのためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾊﾟﾗﾒｰﾀを取らず、EEPROMﾊﾞｲﾄ数を示すlong値を返します。

getPageSize

これはﾌﾗｯｼｭ ﾍﾟｰｼﾞ容量ﾊﾟﾗﾒｰﾀのためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾌﾗｯｼｭ ﾍﾟｰｼﾞを持つAVRﾃﾞﾊﾞｲｽに対してだけ有効です。他のﾃﾞﾊﾞｲｽは-1の値を返します。ﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、各ﾌﾗｯｼｭ ﾍﾟｰｼﾞ内のﾊﾞｲﾄ数を示すlong値を返します。

getFuseStatus

これはﾃﾞﾊﾞｲｽがﾋｭｰｽﾞ ﾋﾞｯﾄを持つかを調べるためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾊﾟﾗﾒｰﾀを取らず、XMLﾌｧｲﾙ内に"FUSE"項目があればtrueを、さもなければfalseを返します。

getXFuseStatus

これはﾃﾞﾊﾞｲｽが拡張ﾋｭｰｽﾞ ﾋﾞｯﾄを持つかを調べるためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾊﾟﾗﾒｰﾀを取らず、ﾃﾞﾊﾞｲｽが拡張ﾋｭｰｽﾞを持つならばtrueを、さもなければfalseを返します。

getSignature

これはﾃﾞﾊﾞｲｽに関する識票ﾊﾞｲﾄを取得するためのｱｸｾｽ ﾒｿｯﾄﾞです。これが実際のﾃﾞﾊﾞｲｽから読んだ識票ﾊﾞｲﾄではなく、XMLﾌｧｲﾙから読んだ識票であることに注意してください。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀとして3つのlongﾎﾟｲﾝﾀを取り、値を返しません。識票ﾊﾞｲﾄはﾊﾟﾗﾒｰﾀによって指示された変数に複写されます。どれかのﾎﾟｲﾝﾀがnullﾎﾟｲﾝﾀの場合に例外が投じられます。

AVRProgrammer

このｸﾗｽは指定AVR書き込み器、例えば、ﾌﾞｰﾄﾛｰﾀﾞまたは実装書き込み器用のｲﾝｽﾀﾝｽを実装するための枠組みを提供する抽象ｸﾗｽです。殆ど全てのﾒｿｯﾄﾞは仮想且つ空で、派生ｸﾗｽによって多重定義されなければなりません。書き込み器は標準化されたﾊﾞｲﾄ志向の通信ﾁｬﾈﾙを通して動き、そしてそれはこのｸﾗｽのｲﾝｽﾀﾝｽを作成する時に提供されなければなりません。

AVRProgrammer

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これはCommChannelｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである、1つのﾊﾟﾗﾒｰﾀを取ります。これは後で実際の書き込み器との通信に使われるﾁｬﾈﾙです。通信ﾁｬﾈﾙにnullﾎﾟｲﾝﾀが提供される場合に例外が投じられます。

~AVRProgrammer

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これは現在何の機能も持たず、将来の拡張に対する単なる代替物です。

readProgrammerID

これは接続された書き込み器のID文字列を読むための静的ﾒｿｯﾄﾞです。このﾒｿｯﾄﾞはCommChannelｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである、このﾒｿｯﾄﾞの唯一のﾊﾟﾗﾒｰﾀで供給された通信ﾁｬﾈﾙを通して書き込み器へ'S'を送ります。どれが作成する派生書き込み器ｸﾗｽかを決めるのにこのﾒｿｯﾄﾞを使ってください。このﾒｿｯﾄﾞは書き込み器の7文字のID文字列を返します。通信ﾁｬﾈﾙにnullﾎﾟｲﾝﾀが提供される、または通信中に異常が起きた場合に例外が投じられます。

setPagesize

これは書き込み器のﾌﾗｯｼｭ ﾍﾟｰｼﾞ容量情報のためのｱｸｾｽ ﾒｿｯﾄﾞです。これはどのﾌﾗｯｼｭ操作にも先立って使われなければなりません。このﾒｿｯﾄﾞはﾊﾞｲﾄでのﾌﾗｯｼｭ ﾍﾟｰｼﾞ容量である、1つのlongﾊﾟﾗﾒｰﾀを取り、値を返しません。

enterProgorammingMode

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これはﾌﾟﾛｸﾞﾗﾐﾝｸﾞ動作形態に入ることを取り付けた書き込み器に告げるべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、この操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起きた場合に例外が投じられるべきです。

leaveProgorammingMode

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これはﾌﾟﾛｸﾞﾗﾐﾝｸﾞ動作形態を抜け出すことを取り付けた書き込み器に告げるべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、この操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起きた場合に例外が投じられるべきです。

chipErase

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽの内容を消去することを取り付けた書き込み器に告げるべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、この操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起きた場合に例外が投じられるべきです。

RC_Calibrate

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これはAtmel ATtiny4/5/9/10/20/40ﾃﾞﾊﾞｲｽの内部RC発振器を校正することを取り付けた書き込み器に告げるべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、書き込み器によって送られた校正値、さもなければ0を返します。Atmel AVR057応用記述を参照してください。

readOSCCAL

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽから発振校正ﾊﾞｲﾄの1つを読むべきです。このﾒｿｯﾄﾞはどのOSCCAL値を読むかを指示するlongと値が複写されるべき変数へのlongﾎﾟｲﾝﾀの2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

readSignature

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽから識票ﾊﾞｲﾄを読むべきです。このﾒｿｯﾄﾞは値が複写されるべき3つの変数へのlongﾎﾟｲﾝﾀの3つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

checkSignature

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽと対照して供給された識票ﾊﾞｲﾄを検査すべきです。このﾒｿｯﾄﾞは検査されるべき識票を含む3つlongﾊﾟﾗﾒｰﾀである、3つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、または供給された識票が取り付けたﾃﾞﾊﾞｲｽの識票と一致しない場合に例外が投じられるべきです。

writeFlashByte

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽのﾌﾗｯｼｭ ﾒﾓﾘにﾊﾞｲﾄを書くべきです。このﾒｿｯﾄﾞはﾌﾗｯｼｭのﾊﾞｲﾄ ｱﾄﾞﾚｽを示すlongと書くためのﾊﾞｲﾄ値を含むlongの2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる場合に例外が投じられるべきです。

writeEEPROMByte

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽのEEPROMにﾊﾞｲﾄを書くべきです。このﾒｿｯﾄﾞはEEPROMのﾊﾞｲﾄ ｱﾄﾞﾚｽを示すlongと書くためのﾊﾞｲﾄ値を含むlongの2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる場合に例外が投じられるべきです。

writeFlash

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは供給されたHEXﾌｧｲﾙ ｵﾌﾞｼﾞｪｸﾄの内容を取り付けたﾃﾞﾊﾞｲｽのﾌﾗｯｼｭ ﾒﾓﾘに書くべきです。このﾒｿｯﾄﾞは必要とされるHEXFileｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

readFlash

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽのﾌﾗｯｼｭ ﾒﾓﾘから、供給されたHEXﾌｧｲﾙ ｵﾌﾞｼﾞｪｸﾄ内にﾃﾞｰﾀを読むべきです。このﾒｿｯﾄﾞは必要とされるHEXFileｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

writeEEPROM

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは供給されたHEXﾌｧｲﾙ ｵﾌﾞｼﾞｪｸﾄの内容を取り付けたﾃﾞﾊﾞｲｽのEEPROMに書くべきです。このﾒｿｯﾄﾞは必要とされるHEXFileｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

readEEPROM

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽのEEPROMから、供給されたHEXﾌｧｲﾙ ｵﾌﾞｼﾞｪｸﾄ内にﾃﾞｰﾀを読むべきです。このﾒｿｯﾄﾞは必要とされるHEXFileｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

writeLockBits

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽの施錠ﾋﾞｯﾄを供給された値に設定すべきです。このﾒｿｯﾄﾞは望む施錠ﾋﾞｯﾄを含むlongの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる場合に例外が投じられるべきです。

readLockBits

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽの施錠ﾋﾞｯﾄを読むべきです。このﾒｿｯﾄﾞは施錠ﾋﾞｯﾄが複写されるべきlongへのﾎﾟｲﾝﾀの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

writeFuseBits

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽのﾋｭｰｽﾞ ﾋﾞｯﾄ(下位と上位の両ﾊﾞｲﾄ)を供給された値に設定すべきです。望むﾋｭｰｽﾞ値を含むlongの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる場合に例外が投じられるべきです。

readFuseBits

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽのﾋｭｰｽﾞ ﾋﾞｯﾄ(下位と上位の両方のﾊﾞｲﾄ)を読むべきです。このﾒｿｯﾄﾞは拡張ﾋｭｰｽﾞ ﾋﾞｯﾄが複写されるべきlongへのﾎﾟｲﾝﾀの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

writeExtendedFuseBits

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽの拡張ﾋｭｰｽﾞ ﾋﾞｯﾄを供給された値に設定すべきです。望む拡張ﾋｭｰｽﾞ値を含むlongの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる場合に例外が投じられるべきです。

readExtendedFuseBits

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは取り付けたﾃﾞﾊﾞｲｽの拡張ﾋｭｰｽﾞ ﾋﾞｯﾄを読むべきです。このﾒｿｯﾄﾞはﾋｭｰｽﾞ ﾋﾞｯﾄが複写されるべきlongへのﾎﾟｲﾝﾀの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

programmerSoftwareVersion

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは書き込み器のｿﾌﾄｳｪｱ版番号を読むべきです。このﾒｿｯﾄﾞは主と副の版番号が複写されるべき変数への2つのﾎﾟｲﾝﾀである、2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

programmerHardwareVersion

このﾒｿｯﾄﾞは仮想でこの抽象基底ｸﾗｽで実装されません。これは書き込み器のﾊｰﾄﾞｳｪｱ版番号を読むべきです。このﾒｿｯﾄﾞは主と副の版番号が複写されるべき変数への2つのﾎﾟｲﾝﾀである、2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはこの操作が書き込み器によって支援される場合にtrueを、さもなければfalseを返します。何か通信異常が起こる、またはnullﾎﾟｲﾝﾀが提供される場合に例外が投じられるべきです。

AVRBootloader

これはAVRProgrammer基底ｸﾗｽから派生したｸﾗｽです。これはAtmel AVR109応用記述で記述されるﾌﾞｰﾄﾛｰﾀﾞ応用に対する標準ｲﾝﾀｰﾌｪｰｽを提供します。これはそれの基底ｸﾗｽから全ての仮想ﾒｿｯﾄﾞを無効にし、等価ﾊﾟﾗﾒｰﾀを持つｺﾝｽﾄﾗｸﾀを提供します。

AVRInSystemProg

これはAVRProgrammer基底ｸﾗｽから派生したｸﾗｽです。これはAtmel AVR910応用記述で記述される実装書き込み器応用に対する標準ｲﾝﾀｰﾌｪｰｽを提供します。これはそれの基底ｸﾗｽから全ての仮想ﾒｿｯﾄﾞを無効にし、等価ﾊﾟﾗﾒｰﾀを持つｺﾝｽﾄﾗｸﾀを提供します。

CommChannel

これはﾊﾞｲﾄ志向の通信ﾁｬﾈﾙのための枠組みを提供する抽象ｸﾗｽです。全てのﾒｿｯﾄﾞが仮想且つ空で派生ｸﾗｽによって多重定義されなければなりません。

~CommChannel

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これは現在何の機能も持たず、将来の拡張に対する単なる代替物です。

openChannel

このﾒｿｯﾄﾞは通信ﾁｬﾈﾙを開くために必要な操作を実行すべきです。これは常にどの通信にも先立って呼ばれるべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、値を返しません。何か異常が起きた場合に例外が投じられるべきです。

closeChannel

このﾒｿｯﾄﾞは通信ﾁｬﾈﾙを閉じるために必要な操作を実行すべきです。これは常に全ての通信が終了された時に呼ばれるべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、値を返しません。何か異常が起きた場合に例外が投じられるべきです。

sendByte

このﾒｿｯﾄﾞは通信ﾁｬﾈﾙにﾊﾞｲﾄ送るべきです。このﾒｿｯﾄﾞは送られるべきﾊﾞｲﾄを含むlongの1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞは値を返しません。何か通信異常が起こる、またはﾁｬﾈﾙが開かれていない場合に例外が投じられるべきです。

getByte

このﾒｿｯﾄﾞは待機して通信ﾁｬﾈﾙからﾊﾞｲﾄを受け取るべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、受信したﾊﾞｲﾄを含むlongを返します。何か通信異常が起こる、またはﾁｬﾈﾙが開かれていない場合に例外が投じられるべきです。

flushTX

このﾒｿｯﾄﾞは送信緩衝部を破棄するべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、値を返しません。何か通信異常が起こる、またはﾁｬﾈﾙが開かれていない場合に例外が投じられるべきです。

flushRX

このﾒｿｯﾄﾞは受信緩衝部を破棄するべきです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、値を返しません。何か通信異常が起こる、またはﾁｬﾈﾙが開かれていない場合に例外が投じられるべきです。

sendMultiple

このﾒｿｯﾄﾞは通信ﾁｬﾈﾙにﾊﾞｲﾄの配列を送るべきです。このﾒｿｯﾄﾞは配列内の最初のunsigned charへのﾎﾟｲﾝﾀとﾊﾞｲﾄでの配列の大きさを示すlongの2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞは値を返しません。何か通信異常が起こる、またはﾁｬﾈﾙが開かれていない場合に例外が投じられるべきです。

SerialPort

これはCommChannel基底ｸﾗｽから派生したｸﾗｽです。これは標準PC COMﾎﾟｰﾄに対して標準通信ﾁｬﾈﾙ ｲﾝﾀｰﾌｪｰｽを提供します。これはそれの基底ｸﾗｽからの全ての仮想ﾒｿｯﾄﾞを無効にし、それ自身の特有のｺﾝｽﾄﾗｸﾀを持ちます。

SerialPort

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これはﾎﾟｰﾄを初期化しますが、通信ﾁｬﾈﾙを開きません。このｺﾝｽﾄﾗｸﾀはCOMﾎﾟｰﾄ番号(1～8)を含むlongと秒での通信時間超過制限を含むlongの2つのﾊﾟﾗﾒｰﾀを取ります。例外が投じられるのは無効なﾎﾟｰﾄ番号または時間超過値が提供される場合です。

HEXFile

これはIntel拡張HEXﾌｧｲﾙの読み書きに関する基本的な機能を提供するｸﾗｽです。これは使われるべきﾒﾓﾘ範囲を定義するためのﾒｿｯﾄﾞも持ちます。これはAVRﾒﾓﾘの部分だけを読み書きするのに有用です。

HEXFlile

これはこのｸﾗｽのｺﾝｽﾄﾗｸﾀです。これは必要とされる最大ﾃﾞｰﾀ緩衝部容量を示すlongと緩衝部を初期化する時に使われるべき既定ﾊﾞｲﾄ値を含むlongの2つのﾊﾟﾗﾒｰﾀを取ります。充分なﾒﾓﾘが利用可能でない場合に例外が投じられます。

~HEXFlile

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これは前に割り当てられた全てのﾒﾓﾘを割り当て解除します。

readFile

このﾒｿｯﾄﾞはHEXﾌｧｲﾙからﾃﾞｰﾀを読みます。このﾒｿｯﾄﾞはHEXﾌｧｲﾙ名の1つのﾊﾟﾗﾒｰﾀを取り、値を返しません。何かﾌｧｲﾙ ｱｸｾｽ異常が起こる、またはﾌｧｲﾙ形式が無効の場合に例外が投じられます。

writeFile

このﾒｿｯﾄﾞはHEXﾌｧｲﾙにﾃﾞｰﾀを書きます。このﾒｿｯﾄﾞは必要とされるHEXFileｵﾌﾞｼﾞｪｸﾄへのﾎﾟｲﾝﾀである1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはHEXﾌｧｲﾙ名の1つのﾊﾟﾗﾒｰﾀを取り、値を返しません。何かﾌｧｲﾙ ｱｸｾｽ異常が起こる場合に例外が投じられます。

setUsedRange

このﾒｿｯﾄﾞはﾒﾓﾘ範囲指示子を上書きします。これは読み書き操作に対して範囲を制限するのに使うことができます。このﾒｿｯﾄﾞは各々、新しい開始と終了の境界を含む2つのlongである、2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞは値を返しません。与えられた範囲が無効の場合に例外が投じられます。

clearAll

このﾒｿｯﾄﾞはﾃﾞｰﾀ緩衝部全体を望むﾊﾞｲﾄ値に設定します。このﾒｿｯﾄﾞは望むﾊﾞｲﾄ値を含むlongの1つのﾊﾟﾗﾒｰﾀを取り、値を返しません。

getRangeStart

これは現在の範囲の開始ｱﾄﾞﾚｽのためのｱｸｾｽ ﾒｿｯﾄﾞです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、開始ｱﾄﾞﾚｽを返します。

getRangeEnd

これは現在の範囲の終了ｱﾄﾞﾚｽのためのｱｸｾｽ ﾒｿｯﾄﾞです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、終了ｱﾄﾞﾚｽを返します。

getData

これは緩衝部内のﾃﾞｰﾀのためのｱｸｾｽ ﾒｿｯﾄﾞです。このﾒｿｯﾄﾞはﾊﾞｲﾄ ｱﾄﾞﾚｽを含むlongの1つのﾊﾟﾗﾒｰﾀを取り、ﾊﾞｲﾄ値を含むlongを返します。ｱﾄﾞﾚｽが正当な範囲外の場合に例外が投じられます。

setData

これは緩衝部にﾃﾞｰﾀを設定するためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾊﾞｲﾄ ｱﾄﾞﾚｽを含むlongとﾊﾞｲﾄ値を含むlongの2つのﾊﾟﾗﾒｰﾀを取ります。ｱﾄﾞﾚｽが正当な範囲外の場合に例外が投じられます。

getSize

これは緩衝部容量を取得するためのｱｸｾｽ ﾒｿｯﾄﾞです。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、ﾊﾞｲﾄでの緩衝部容量を含むlongを返します。

XMLFile

このｸﾗｽはAtmel AVR Studioと共にやって来るAVRﾃﾞﾊﾞｲｽ記述ﾌｧｲﾙを読むための簡単なXML解析部を提供します。これは標準的なXML解析に関して他のﾌﾟﾛｼﾞｪｸﾄでも使うことができます。このｸﾗｽは例えそのようなﾀｸﾞに出会った場合で異常が全く生成されなくても、ﾀｸﾞの内側の属性を支援しないことに注意してください。属性は単純に無視されます。このｸﾗｽはXMLﾌｧｲﾙ全体を読んで含まれた情報からﾒﾓﾘ内在木構造を構築します。

XMLFlile

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これは解析されるべきXMLﾌｧｲﾙ名の1つの文字列ﾊﾟﾗﾒｰﾀを取ります。このｺﾝｽﾄﾗｸﾀは直ちにﾌｧｲﾙを読み、故に例外が投じられない場合、ｺﾝｽﾄﾗｸﾀ終了時にXML木構造が構築されて準備が整います。

~XMLFlile

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これはﾒﾓﾘ内在XML木構造用に割り当てられた全てのﾒﾓﾘを割り当て解除します。

exists

このﾒｿｯﾄﾞは与えられたﾊﾟｽで節(ﾉｰﾄﾞ)が存在するかを調べます。このﾒｿｯﾄﾞは節名を含む完全なﾊﾟｽの1つの文字列ﾊﾟﾗﾒｰﾀを取り、節のstring値を返します。節が見つからない場合に例外が投じられます。

print

このﾒｿｯﾄﾞは完全なXML木構造の内容を短い形式で出力します。このﾒｿｯﾄﾞは元々ﾃﾞﾊﾞｯｸﾞ目的用に実装されました。これはﾊﾟﾗﾒｰﾀを取らず、値を返しません。

JobInfo

これは命令行ﾊﾟﾗﾒｰﾀから抽出した全ての情報を保持するｸﾗｽです。このｸﾗｽは必要な操作を実行するための機能も含みます。

JobInfo

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これは全ての情報を既定値に初期化します。

parseCommandline

このﾒｿｯﾄﾞは命令行ﾊﾟﾗﾒｰﾀを解析します。これはmain()関数から良く知っているint argcとchar *argv[]の2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを返しません。何か無効なﾊﾟﾗﾒｰﾀに出会う場合に例外が投じられます。

doJob

このﾒｿｯﾄﾞは命令行ﾊﾟﾗﾒｰﾀから抽出された全ての作業を完全に満たすのに必要な全ての作業を実行します。これは必要とされる通信ﾁｬﾈﾙと書き込み器ｵﾌﾞｼﾞｪｸﾄの作成も処理します。このﾒｿｯﾄﾞはﾊﾟﾗﾒｰﾀを取らず、値を返しません。何か異常が起きる場合に例外が投じられます。

Utility

このｸﾗｽは度々使われる関数に対するｺﾝﾃﾅと名前空間として扱います。それはｿｰｽ ﾌｧｲﾙでｲﾝｽﾀﾝｽを作成され、ﾍｯﾀﾞ ﾌｧｲﾙで提供されるUtilｵﾌﾞｼﾞｪｸﾄへの外部参照です。これは特に記録と進捗のﾒｯｾｰｼﾞ、それと静寂操作許可に使われます。

Utilty

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これはﾊﾟﾗﾒｰﾀを取りません。このｺﾝｽﾄﾗｸﾗは記録と進捗の両ﾒｯｾｰｼﾞを許可するために内部の記録と進捗の状態を初期化します。

~Utility

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これは現在何の機能も持たず、将来の拡張に対する単なる代替物です。

muteLog

このﾒｿｯﾄﾞは画面での表示から更なる全ての記録ﾒｯｾｰｼﾞを防ぎます。これはﾊﾟﾗﾒｰﾀを取らず、値を返しません。

muteProgress

このﾒｿｯﾄﾞは画面での表示から更なる全ての進捗ﾒｯｾｰｼﾞを防ぎます。これはﾊﾟﾗﾒｰﾀを取らず、値を返しません。

log

このﾒｿｯﾄﾞは閉鎖(ﾐｭｰﾄ)されていなければ、画面に記録形式のﾒｯｾｰｼﾞを出力します。このﾒｿｯﾄﾞはﾒｯｾｰｼﾞ文字列の1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞは値を返しません。

progress

このﾒｿｯﾄﾞは閉鎖(ﾐｭｰﾄ)されていなければ、画面に進捗形式のﾒｯｾｰｼﾞを出力します。このﾒｿｯﾄﾞはﾒｯｾｰｼﾞ文字列の1つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞは値を返しません。

convertHex

このﾒｿｯﾄﾞは16進文字列を数値に変換します。このﾒｿｯﾄﾞは文字列の1つのﾊﾟﾗﾒｰﾀを取り、変換された値を含むlongを返します。何か変換異常が起こる場合に例外が投じられます。

convertLong

このﾒｿｯﾄﾞは指定された基数を用いて数値を文字列に変換します。このﾒｿｯﾄﾞは変換されるべき数値を含むlongと使われるべき望む基数を含むlongの2つのﾊﾟﾗﾒｰﾀを取ります。

getRegistryValue

このﾒｿｯﾄﾞはWindowsのﾚｼﾞｽﾄﾘ ﾃﾞｰﾀﾍﾞｰｽから値を取得します。このﾒｿｯﾄﾞはﾚｼﾞｽﾄﾘ鍵のﾊﾟｽを含む文字列と鍵名を含む文字列の2つのﾊﾟﾗﾒｰﾀを取ります。このﾒｿｯﾄﾞは取得した値を含む文字列を返します。ﾃﾞｰﾀﾍﾞｰｽ操作中に何か異常が起きる場合に例外が投じられます。

ErrorMsg

このｸﾗｽは例外として投じられるべき異常ﾒｯｾｰｼﾞに対するｺﾝﾃﾅとして扱います。

ErrorMsg

これはこのｸﾗｽに対するｺﾝｽﾄﾗｸﾀです。これは異常ﾒｯｾｰｼﾞ文字列の1つのﾊﾟﾗﾒｰﾀを取ります。

~ErrorMsg

これはこのｸﾗｽに対するﾃﾞｽﾄﾗｸﾀです。これは現在何の機能も持たず、将来の拡張に対する単なる代替物です。

What

これは異常ﾒｯｾｰｼﾞ文字列のためのｱｸｾｽ ﾒｿｯﾄﾞです。これはﾊﾟﾗﾒｰﾀを取らず、異常ﾒｯｾｰｼﾞの複製を返します。

5. 改訂履歴

文書改訂	日付	注釈
2568A	2004年7月	初版文書公開
2568B	2012年10月	新雛形といくつかの些細な修正。改訂Aで言及していた打ち切り製品を削除。

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
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 G.K.
141-0032 東京都品川区
大崎1-6-4
新大崎勧業ﾋﾞﾙ 16F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ合同会社
TEL (+81)(3)-6417-0300
FAX (+81)(3)-6417-0370

本応用記述はAtmelのAVR911応用記述(doc2568.pdf Rev.2568B-10/2012)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訳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。