



サイト内検索:



AND検索



OR検索

検索

管理メニュー: [トップ](#) [新規](#) [編集](#) [リロード](#) [添付](#) [凍結](#) [差分](#) [最終更新](#) [バックアップ](#) [MENU編集](#) [一覧](#)

AVR/HIDaspX

[Prev](#)[AVR](#)[Next](#)

Counter: 16296, today: 13, yesterday: 77

- ===== はじめに =====
 - HIDaspXとは(2008年9月)
- [HIDaspXリンク集 --- \[関連ページへの移動にご利用ください\]](#)
- [HIDaspX\(USB接続のAVRライタ\)の紹介](#)
 - [HIDaspXの回路図と製作例](#)
- [最新のHIDaspX用アーカイブ\(移動しました\)](#)
 - [開発に協力していただける方へ\(開発環境の補足\)](#)
 - [関連プロジェクト\(ATmega88/168用\)](#)
- [ファームウェア\(ATtiny2313\)のFUSE設定](#)
- [hidspX, hidmonなどのコマンドについて](#)
- [動作確認済みのAVRリストと動作速度の目安](#)
- [HIDaspXをライタとして利用する](#)
 - [hidspXのオプションについて](#)
- [AVRライタなしでHIDaspX用のファームを書き込む方法](#)
- [USB-HUBの利用で高速化](#)
- [HIDaspとの出会い\(コラム\)](#)

===== はじめに ===== ↑

HIDaspXも2008年9月に公開以降、多くの情報が蓄積し、高度な話題に驚く方が多いようです。そのため、このページではHIDaspXに関する基本的な記述に留めました。

- 「HIDaspXの概要」 [kumanさんのページ](#)
- HIDaspX用の制御ソフト [hidspXのTips集](#) [AVR/hidspX_tips](#)
- HIDaspXのQ&A [AVR/HID_QA00](#)

を参照してください。

↑

HIDaspXとは(2008年9月) ↑

(2008-09-02 (火) 09:14:33)

HIDaspXは、「ATtiny2313マイコン1個で構成するUSB接続のAVRマイコン用プログラマ」です。HIDクラスを採用したことでWindowsやLinux, Mac OSのいずれでもドライバのインストールが不要で、USB-I/Oとしても利用可能になっています。

こうしたライタを製作するに至った経緯を説明します。

どんな組み込みマイコンの開発を行う場合でも、特殊なドライバや開発用ソフトウェアのインストールが必要です。しかし、これが許可されない環境(特に教育現場)も少なからずあることに気がきました。特に教育現場では、「ドライバやソフトウェアのインストールは禁止」という環境は珍しくありません。

一方、現在ではWinAVRなどの開発環境はCD-ROMやUSBメモリに格納しても利用可能ですから、ドライバ不要のAVRライタがあれば、インストール不要のAVRマイコン開発環境を実現できます。

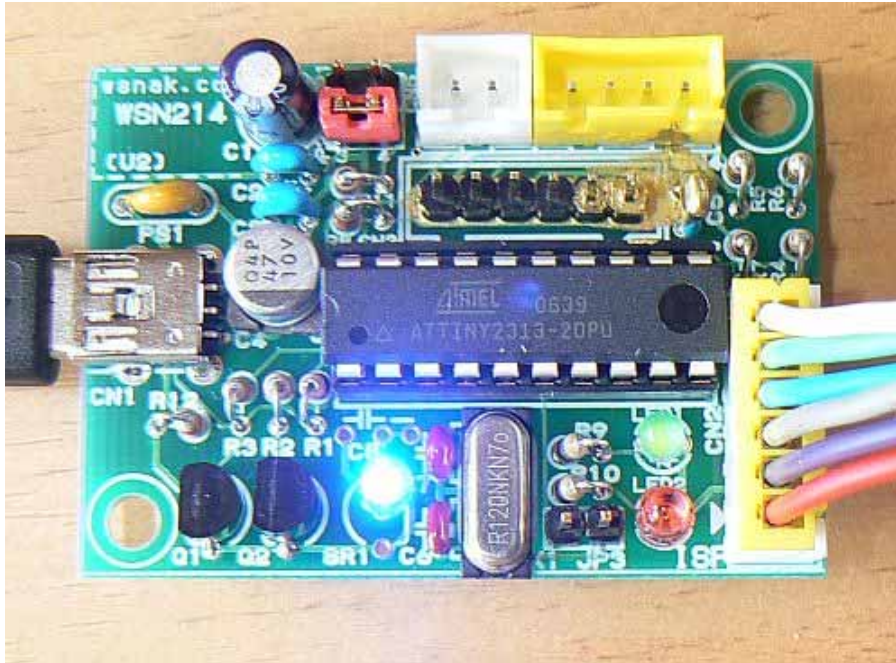
ドライバのインストールが出来ない環境でもマイコン関連の学習を可能にする為、HIDaspやHIDspHなどのライタやAVRUSBのプロジェクトを参考に、[irukaさん](#)の協力を得て、HIDaspをベースに独自に安定化と高速化などの改良したのが**HIDaspX**で、ハードウェアとファームウェア一式を意味します。

HIDaspXをAVRライターとして利用するには、PC用プログラムが必要になります。
これには私が保守しているavrspXを元に、HIDaspXをサポートを追加した**hidspX**コマンドを開発しました。なお、HIDaspX対応のコードには、瓶詰堂さんの作成されたコード(hidaspc.cの改造版)を含んでいます。

2008年10月以降、**hidmon**が利用可能です。hidmonコマンドにてHIDaspXの各種I/Oを操作可能で、USB接続のIOとして利用できます。また、ATtiny2313のI/Oや各種のレジスタを制御できるため、マイコン学習教材としても有用です。

以下が、私が製作したHIDaspXです。+3.3Vの安定化回路を実装し、この「ライター」だけで、3.3V(乾電池2本分)の回路実験が可能です。COMポートが付いていないノートパソコンによるマイコン開発用に小型ケースに組み込んでみました。同様の目的にはUSBaspも使えますが、USBaspではドライバのインストールが必要です。HIDaspXはドライバのインストールが不要という大きな特徴があります。

○HIDaspX専用プリント基板の例([WSN214](#))
私のデザインした回路を忠実に再現しています。



○USB-IO用ブレッドボード対応モジュール([WSN216](#))
ブレッドボードへの実装を前提に小型化しましたが、意外にも製作は容易です。
(クリスタルはAVRマイコンの真下に実装しており、写真からは見えません)

■ HIDaspXの特徴

- どのUSBコネクタに繋いでも動作する(ドライバのインストール不要)
 - COMポートの無いIPCでも利用可能(今はこれが一般的です)
 - Windows 2000/XP/Vistaで動作(Windows95/98は未サポート)
 - MacOSやLinuxでもWindowsと同様に利用可能
- 使い勝手に優れた **hidspX(avrspXの別名)**コマンドが利用できる
 - インストール不要(CD-ROMやUSBメモリからの実行も可能)
 - ターゲットのAVRマイコンを自動認識(新・旧のAVRマイコンをサポート)
 - 旧タイプの一部を除き、CHIP名を指定する必要なし
 - BASCOM-AVR, mikroC, AVR studioなどの開発環境にも組み込み可能
 - hidspXのショートカットに、HEXファイルをD&DでもOK
- ISPコネクタを抜き差しすることなく開発可能(ターゲット実行時の影響は最小限)
- 電源の投入順序も気にする必要なし(ライターRESET時、自動再接続)
- 製作コストは最小限(安価(秋月電子では100円!)なATtiny2313のみで実現)

- 材料代は500円程度？(専用プリント基板が入手可能)
- 小型(USBメモリの形状で製作も可)
- 十分な速度で動作(USBaspとほぼ同じ)
 - HUBを利用すると更に動作速度が向上(USBaspでは速度は低下する)
- USB-IOとして利用できる(hidmonを利用)
 - コマンドによる対話方式で内蔵レジスタを操作可能
 - スクリプトによる一括実行も可能
 - DLLを使って、各種のプログラム言語(C言語, Visual BASICなど)から操作可能
 - **rubyを使っている制御が可能になりました**(2009年1月2日)
 - 詳細は[AVR/HIDmon03](#)をご覧ください

HIDaspとHIDAspxは同一のハードウェアですが、ソフトウェアの互換性はありませんので、混同しないようにしてください。なお、HIDAspxは、**エイチ・アイ・ディー アスペックス**とお読みください。

NEWS 2009/2/20


2009年1月末から、WS NAKさんより、HIDAspx用のプリント基板がリリースされました。これは、優れた特徴を有するHIDAspxを、手軽で確実に製作できるように、私(senshu)とWSNAKさんと検討を重ね、設計を行ったものです。HIDAspxの製作を考えている方は、ぜひこれを利用して製作してください。

HIDAspx用プリント基板(HIDAspx以外にも利用できます) <http://wsnak.com/kit/214/index.htm>

USB-IO用ブレッドボード対応モジュール <http://wsnak.com/kit/216/index.htm>

↑

HIDAspxリンク集 ---【関連ページへの移動にご利用ください】[↑]

- HIDAspxに関する情報 [タイトル一覧](#) [AVR/HIDAspx00](#)
- HIDAspx [更新履歴](#) [AVR/HIDAspx_news](#)
- HIDAspxに関するQ&A [質問はこちら](#) [AVR/HID_QA00](#)
- HIDAspxの製作レポート [利用者の声](#) [AVR/HID_reports](#)
- HIDAspxによる教材開発 [AVR/HID_kyouzai](#)
- AVRライタに関するFAQ集 [AVR/writer_FAQ](#)
- 掲示板にも関連情報があります [AVR/news_contents_all](#)
- 「**V-USB**」(このソフトがHIDAspxを支えています)
 - <http://www.obdev.at/products/vusb/index.html>
- 2008-10-29時点のこのページの内容  [HIDAspx-1029.pdf](#)
- HIDAspxをWindows以外のOSで利用する [AVR/HIDAspx_other_OS](#)
- HIDAspxをUSB-IOとして利用する例
 - HIDmonについて [AVR/HIDmon00](#)
- HIDAspxのハードウェアをusb-RS232のブリッジとして利用 [AVR/usbRS232](#)

ページの簡素化を図りました。以前のページの内容は↑のPDFをご覧ください。

関連するページへのリンク

AVR開発ツールのリンク(外部)

- [ドライバをインストールする例\(マイコン講座/環境の構築\)](#)
 - avrspの使い方は、hidspcにもそのまま利用可能
- ポータブルWinAVR <http://www.chip45.com/index.pl?page=PortableWinAVR&lang=en>
- 瓶詰堂さん(HIDaspの開発元) http://www.binzume.net/library/avr_hidasp.html
 - このページにもリンクしていただきました
- irukaさんのサイト(瓶詰堂さんもお薦め)
 - <http://hp.vector.co.jp/authors/VA000177/html/A3C8A3C9A3C4A3E1A3F3A3F0.html>
- kumanさんの回路図を含む、実践レポート(一読をお勧めします)

- <http://www.geocities.jp/kuman2600/n6programmer.html#13> (10/12追記あり)
- hidsp-1012b.zipの感想が書かれています
- [HIDasp kuman流の使い方](#) (10/19追加)
- 「工研Wiki」...電通大の学生さんによるHIDaspの解説
 - <http://delegate.uec.ac.jp:8081/club/koken/wiki/?AVR%2FHIDasp>
- AVRライターで有名なTADさんのページにもHIDaspが登場
 - <http://homepage2.nifty.com/denshiken/AVW021.html>
- 「Fight with life」(RAINさんのBlog)..HIDasp(x)を4台も作成されています
 -  <http://amenotiyukizora.blog76.fc2.com/>
- 岩永さんによるHIDaspの解説 → [HIDasp AVR用USBライターの究極 安価・高速・簡単](#)
- HIDaspの使い方(マウス操作での利用法)
 - [AVR/HIDasp/使い方](#)
- 中学生向けの学習教材にHIDaspを適用した例
 - <http://www.ne.jp/asahi/ja/asd/gijutu/HIDapio/>

↑

↑

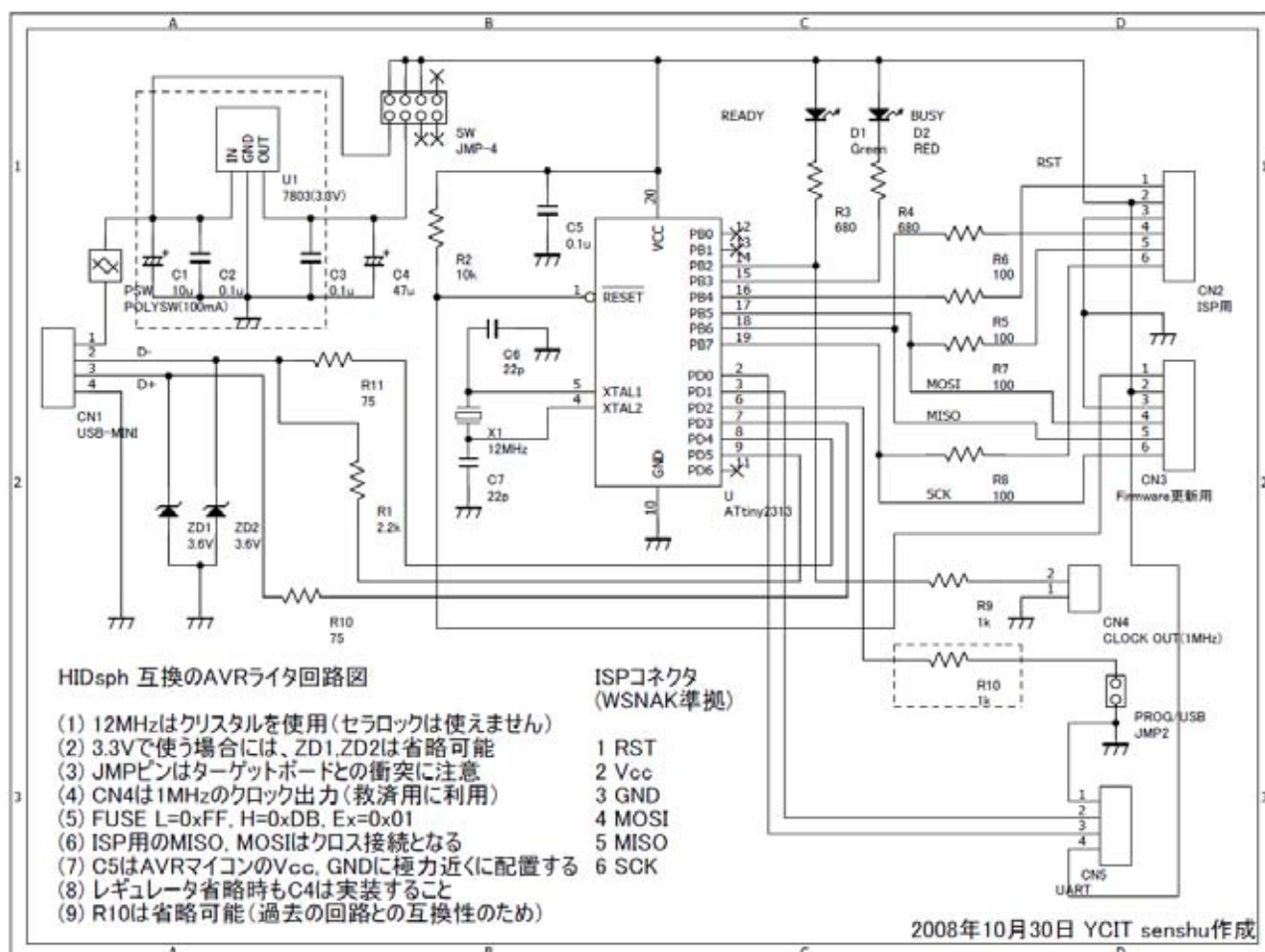
HIDasp(USB接続のAVRライター)の紹介[↑]

HIDaspの回路図と製作例[↑]

■ HIDaspの回路例(HIDspと同じ回路です。ISPコネクタ配置はWSNAKボードに準拠)

クリックで拡大可能(後半で紹介しているHIDspと同じものです)

三端子レギュレータの入出力に接続している47 μ Fのコンデンサは、USB規格としては大きすぎるようです。発振止めにはこの程度が必要だと思いますが、規格内に収めるなら、10 μ F程度に留める必要があります。(メーカーさんではどうやっているのでしょうか)



なお、UARTコネクタはusbRS232での利用を想定しています。AVRライターとして使うだけなら、このコネクタを実装する必要はありません。

水晶発振子は必須です

ある方から、HIDaspではセラロックで動作するのにHIDaspはセラミック振動子では動作しないとの報告がありました。HIDaspでは高速化を実現する為にパケット長を32バイトに拡大しており、発振周波数の誤差に敏感です。
回路図にも明記していますが、セラミック振動子(商品名の例:セラロック)では動作しないと考えてください。

↑

最新のHIDasp用アーカイブ(移動しました) [†]

こちらのページに移動しました。 [AVR/HIDasp_news02](#)

↑

開発に協力していただける方へ(開発環境の補足) [†]

上記のFirmwareは、WinAVR-20060421をインストールした後に、AVR-Wikiから入手できる「機能追加版 2006-04-22」を上書きした環境で作成しています。

<http://avrwiki.jpn.ph/wiki.cgi?page=WinAVR%A5%D0%A5%B0%BE%F0%CA%F3>

ATtiny2313用のファームウェアを作成したい方は、このavr-gccを利用してください。
これ以外のコンパイラでは、ATtiny2313用のFLASHサイズ2048バイトに収めることは困難です。

2006/04/22	19:14	369,664	avr-ar.exe
2006/04/22	19:14	514,560	avr-as.exe
2006/04/22	21:23	91,136	avr-c++.exe
2006/04/22	19:14	399,360	avr-c++filt.exe
2006/04/22	21:23	90,624	avr-cpp.exe
2006/04/22	21:23	91,136	avr-g++.exe
2006/04/22	21:23	89,088	avr-gcc.exe
2006/04/22	21:23	26,112	avr-gcov.exe


バージョンの確認は、CMD窓にて、以下の方法で確認できます。

```
>avr-gcc -v
Reading specs from C:/WinAVR/lib/gcc/avr/3.4.6/specs
Configured with: ./configure --target=avr --prefix=/c/WinAVR
--enable-languages=c,c++ --with-dwarf2 --enable-win32-registry=WinAVR
--disable-nls --enable-c-mbchar
Thread model: single
gcc version 3.4.6
```

--disable-nls --enable-c-mbcharの表示があるのを確認ください。実行ファイル群の日付を確認するを忘れずに行ってください。

```
>which avr-gcc
```

でコンパイルに利用するコンパイラの実体(フルパス名)を確認できます。

- 開発者用のWinAVRアーカイブ(15MB)  [WinAVR-0604.7z](#)
 - ファームウェアをコンパイルしたい方だけ、ダウンロードしてください
 - 展開したファイル群をWinAVR20060421のディレクトリに上書きします
 - いつでも戻せるように、事前のバックアップを忘れずに行ってください

パソコン用の実行ファイルは、MinGWとBorlabd C++ でコンパイルしています。


また、今まで公開してきたアーカイブは、irukaさんのサイトから入手可能です。
(大量のファイルを保管していただき、感謝いたします。)

<http://psp.dip.jp/web/upload/filelist.cgi>

ここにあるアーカイブに不具合を発見した場合には、旧版もお試ください。

↑

関連プロジェクト(ATmega88/168用) ↑

- bootloader + hidmon88
 - bootmon(ファーム・実行ファイル, ソースコード) ...  [bootmon-1127.zip](#)
 - RC発振器でbootloadHIDの動作が可能(RC発振モードは評価目的に公開)
 - クリスタル発振子(12/20MHz)で動作確認済み

↑

ファームウェア(ATtiny2313)のFUSE設定 ↑

クリスタル発振器の場合(1028版以降の設定)

```
Low: 11111111 (0xFF)
    |||++++- CKSEL[3:0] システムクロック選択
    ||+--- SUT[1:0] 起動時間
    |+- CKOUT (0:PD2にシステムクロックを出力)
    +- CKDIV8 クロック分周初期値 (1:1/1, 0:1/8)

High:11-11011 (0xDB)
    |||||+--- RSTDISBL (RESETピン 1:有効, 0:無効(PA2))
    |||++++- BODLEVEL[2:0] (111:0ff, 110:1.8, 101:2.7, 100:4.3)
    |||+--- WDTON (WDT 0:常時ON, 1:通常)
    ||+--- SPIEN (1:ISP禁止, 0:ISP許可) Parallel時のみ
    |+- EESAVE (消去でEEPROMを 1:消去, 0:保持)
    +- DWEN (On-Chipデバッグ 1:無効, 0:有効)

Ext: -----1 (0xFF)
    +- SPEN (SPM命令 1:無効, 0:有効)
```

1028版以降では、PD2端子は12MHzのクロック出力ではなく、AVR programmer/USB-IO の識別用に利用することにした。

回路図とおりに製作している場合には、1kΩの抵抗があるので、従来のファームを使っている場合でも悪影響はありません。**この変更は、一つのハードウェアをAVR programmer/USB-IOとして共用する場合に重要な仕様変更です。**オープンでProgrammer、GNDに接続時、USB-IOとして利用します。このモード変更は、HIDAspxモジュールがRESET時に一度だけ判断します。動作後にPINを変更してもモードは変わりません。hidspkで利用する時には、Programmer モード、hidmonから利用する時には、USB-IOモードに設定してください。特に、Programmerモードでは、PB2には1MHzのクロック信号が出力されます。この状態では、PB2をUSB-IOとしては使えませんので、ご注意ください。

HIDAspxをAVR programmerとしてのみ使う場合には、従来(以下の設定)のままでも利用可能です。

PD2からの外部クロック出力を有効にする場合(従来の設定)

```
Low: 10111111 (0xBF)
    |||++++- CKSEL[3:0] システムクロック選択
    ||+--- SUT[1:0] 起動時間
    |+- CKOUT (0:PD2にシステムクロックを出力)
    +- CKDIV8 クロック分周初期値 (1:1/1, 0:1/8)
```

↑

hidspk, hidmonなどのコマンドについて ↑

hidspk.exe, hidmon.exeなどのコマンドは、GUIツールから呼び出すことも可能ですが、

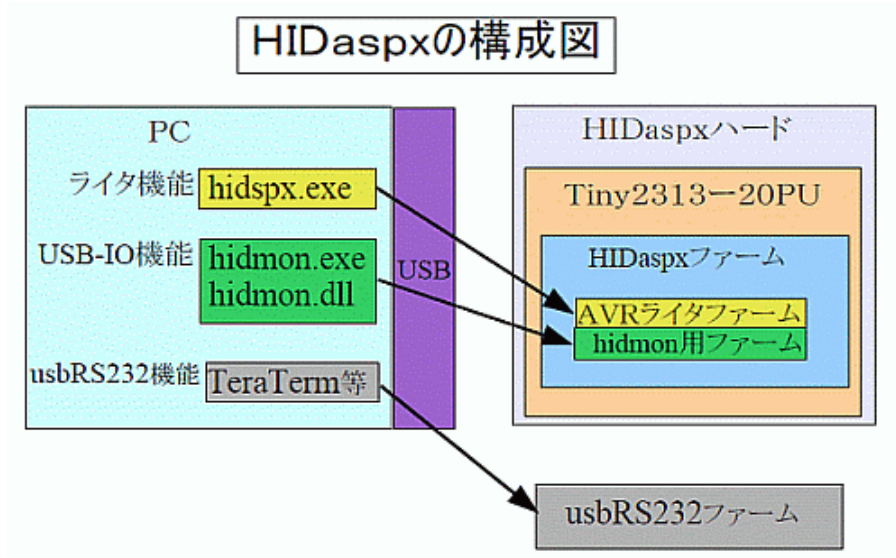
基本的にはコマンドプロンプト上で利用するツールです。

一度はコマンドプロンプト(CMD窓)でご利用いただき、CUIの便利さを堪能ください。

現在(2009年)は、hidspkからAVRマイコンのデータシートやヒューズ設定などを確認できるWebページを自動で開く機能なども利用できます。

HIDAspxに関連する用語について、特徴や相互の関係を整理してみます。

参考「HIDAspxのシステム構成図(aduinさんによるもの)」



(1) HIDAspx 解説ページ [AVR/HIDAspx](#)

1. HIDクラスを採用しATtiny2313一個で実現した「AVRライタ兼USB-IO」
(hidaspを参考にスタートしましたが、今は互換性はありません)
2. シンプルでローコストに製作できるハードウェア
3. HIDクラスの採用により、ドライバのインストールは不要
4. USB接続で高速に動作する(特にUSB-HUB経由の利用時)

(2) hidspkコマンド

1. HIDAspxをAVRライタとして使う場合に使用するプログラム
2. SETUPコマンドで、簡単に導入できる
3. AVRマイコンを自動認識し、旧タイプのAVRマイコンもサポート
4. BASCOM-AVRやAVRstudioとの統合利用も可能

(3) hidmonコマンド [AVR/HIDmon00](#)

1. HIDAspxをUSB-IOとして利用する時に使用する
2. HIDAspxのみでAVRマイコンの学習が可能
3. hidmon.dllを使えば、一般的な言語(C言語やVB, VBAなど)から制御可能

(4) bootmon [irukaさんによる解説](#)

1. シンプルでローコストに製作できるハードウェア
2. ATmega88/168用(12/16/20MHzのクリスタル発振子、あるいは内蔵RC発振器で動作)
3. ブートローダ&簡易AVRマイコン用モニタです(現時点ではAVRライタ機能はありません)
 - i. bootloadHID互換のbootloader機能
 - ii. AVRマイコンモニタ(簡易デバッガ)

(5) bootloadHIDコマンド

1. bootmon用の書き込み用ツール
2. 任意の開発ツールで作成したHEXファイルを書き込み、実行できる

(6)hidmon88コマンド

1. AVRマイコン内のRAMやI/Oの読み書き、EEPROMやFLASHの読み出しが可能

2. 対話的(一括実行も可能)な操作が可能
3. シンボリック(レジスタ名は名前で指定)に操作可能

bootmonの名前が示すとおり、現時点ではAVRライタ機能はありません。
HIDaspXの完成度が高いので、AVRライタとしてはHIDaspXを使ってください。
HIDaspXが複数台あっても、HIDaspXは多機能で汎用性があり無駄にはなりません。
シリアル情報を設定すれば、一台のPCに複数台のHIDaspXを接続して利用できます。

このように、AVRマイコンを使って学習を考えている方には魅力的な仕様です。
どちらも、irukaさんと共同で開発を行ない、多くの方からの助言を元に、現在に至っています。
(irukaさんの作成されたbootmonに対し、シリアル情報のサポート、RC発振対応やmega168対応、各周波数のHEXファイルの一括生成やsetup機能を追加しました)

なお、bootmonではRC発振モードを利用できますが、hidmon88用のコードが入りません。12や20MHzのクリスタルでの利用をお勧めします。



動作確認済みのAVRリストと動作速度の目安 [†]

MCU名	Device Signature	FLASH	page	EEPROM
AT90S2313	1E-91-01	2048		128
ATtiny2313	1E-91-0A	2048	32 x 64	128
ATtiny26L	1E-91-09	2048	32 x 64	128
ATtiny45	1E-93-06	4096	64 x 64	256
ATtiny85	1E-93-0B	8192	64 x 128	512
AT90S4433	1E-92-03	4096		256
AT90S8515	1E-93-01	8192		512
ATmega8515	1E-93-06	8192	64 x 128	512
ATmega48	1E-92-05	4096	64 x 64	256
ATmega8	1E-93-07	8192	64 x 128	512
ATmega88	1E-93-0A	8192	64 x 128	512
ATmega168	1E-94-06	16384	128 x 128	512
ATmega64	1E-96-02	65536	256 x 256	2048
ATmega644P	1E-96-0A	65536	256 x 256	2048
ATmega128	1E-97-02	131072	256 x 512	4096

これらのAVRマイコンは、8MHz以上のクロックで動作していれば -d1でRead/Write可能です。
その時の読み取りは 4kB/秒程度です。書き込み速度はベリファイが必要なため、この1/2程度になります。この時間は、hidmon benchテストで15kB/秒の場合です。

HIDaspXの利用者から、「ATmega8, ATmega64, ATtiny45でも使える」との報告がありました。

1014a版を使い、ATtiny2313で2kBの全領域をWrite/Verify時間は、
8MHzの場合 (-d1) で約1秒、
1MHzの場合 (-d4) で約3秒です。

```
>timeit hidspx -d0 2kB.hex
Detected device is ATtiny2313.
Erase Flash memory.
Write   Flash: 2048/2048 B
Verify  Flash: 2048/2048 B
Passed.
```


Elapsed Time: 0:00:00.953

↑

HIDaspXをライターとして利用する[†]

HIDaspXをAVRマイコン用のライターとして利用するには、**hidspX**コマンドを使います。
USB IOとして操作する場合には、**hidmon**コマンドを使ってください。

↑

hidspXのオプションについて[†]

1. HIDaspXライターとターゲットボードを6Pケーブルで接続します。
 - i. 電源はターゲットボードから供給が基本です。
- コマンドプロンプトを起動し、CDコマンドでHEXファイルのあるディレクトリに移動します。
 - [そのフォルダ内で直接コンソールを開く方法](#)を利用すると便利です。

コマンド	オプション	機能	備考
hidspX	無し	オプション書式とサポートライター類を表示	(-?で詳細表示)
hidspX	--show-options	コマンド行を表示後に実行	iniファイルの設定内容を含む
hidspX	--atmel-avr	Atmel社のAVRのページを開く	Web browser (要インターネット接続)
hidspX	--avr-devices	AVR マイコンのDEVICE一覧	Web browser (要インターネット接続)
hidspX	-pu?	USBのサーチ結果を表示	DLLバージョンも同時に表示
hidspX	-pcN	com writerを指定	Nは接続COMポート番号
hidspX	-pbN	com spi Bridgeを指定	Nは接続COMポート番号
hidspX	-phN	HIDaspXを指定	シリアル番号の指定も可能
hidspX	-?	サポートデバイス一覧を表示	
hidspX	-r	Device Read (デバイス仕様を表示)	
hidspX	-rf	Read Fuse	
hidspX	-rF	Read Fuse (command line example)	
hidspX	-ri	Read Fuse Information	Web browser (要インターネット接続)
hidspX	-rd	Read Document	Web browser (要インターネット接続)
hidspX	XXX.hex	Flash Write (複数ファイルを指定可)	
hidspX	XXX.eep	EEPROM Write (Flashと同時指定が可能)	
hidspX	-v- XXX.hex	Verify無しのWrite (複数ファイルを指定可)	
hidspX	-v XXX.hex	Verify (複数ファイルを指定可)	
hidspX	-rp	Read Program	> 出力ファイル名でファイルに出力可能
hidspX	-fL<hex>	Fuse Lowの設定	-ri で詳細を確認可能
hidspX	-fH<hex>	Fuse Highの設定	-ri で詳細を確認可能
hidspX	-fX<hex>	Fuse eXtendの設定	-ri で詳細を確認可能
hidspX	-e	Erase (ロックビットの解除)	
hidspX	-l<hex>	メモリロックビット設定	
hidspX	-qDEVICE	デバイス確認後に実行	
hidspX	-tDEVICE	デバイス特定	
hidspX	-dNN	Delay設定	
hidspX	-w1	Wait	処理完了後、キー入力を待つ

hidspX -wN

N秒Wait

処理完了後、N(2以上)秒間、動作を停止

<hex>は、0x1b のように書くことができます。00011101のように2進数で書くこともできます。
詳細は、hidspXに同梱のavrx-tool.txtをご覧ください。

Tips集は、[こちら](#)にまとめました。

↑

AVRライタなしでHIDaspX用のファームを書き込む方法 [†]

HIDaspXはAVRマイコンで制御している為、AVRライタを持っていない方がHIDaspXを作成する場合、ファームウェアを書き込みをどうするかが大きな問題になります。

これはHIDaspXに限らずUSBasp等でも同様であり、「鶏と卵問題」とも呼ばれています。

hidspX(avrspX)では、この問題の解決策として、RC232Cコネクタに簡単な回路を接続して書き込む方法を提案しています。(説明不足のためか、これを使って書き込んだという報告はありません)

タイトルの「AVRライタなしで」はやや誇張した表現ですが、簡易に製作できるAVRライタを製作し、それで書き込もうというアイデアです。このアイデアの原点は、以下のTADさんのページです。この考えを元に、kkkさんが、avrspXにこのライタ機能を実装し、hidspXはそれを継承しています。

私は「本物のRC232Cコネクタが利用できるPC」で実行する必要があると考えていましたが、TADさんのページには[秋月電子のUSB変換ケーブルでもOK](#)と書かれています。

<http://homepage2.nifty.com/denshiken/AVW009.html>

HIDaspXの部品の多くを流用できるので、追加コストはD-SUBのコネクタ、若干のRC、スイッチ、電池など、電子工作マニアなら手持ちの部品が大半です。

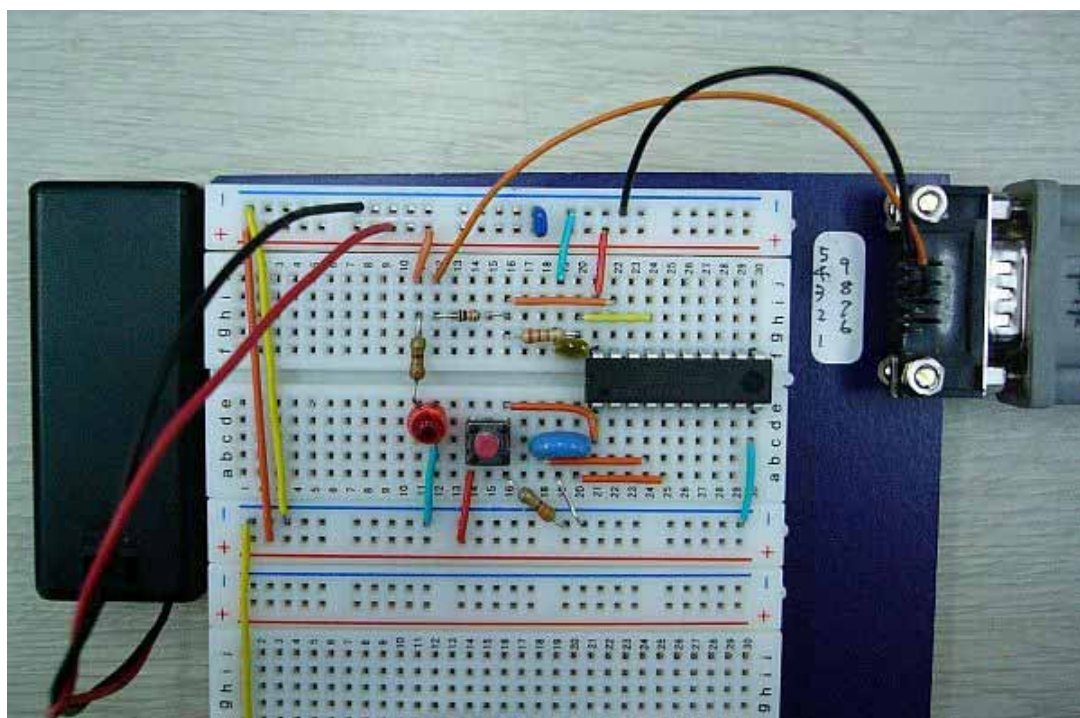
これほど簡単に製作可能なら、これを使ってみようと思う方がいるかもしれませんが、RS232Cのポートが必要で、書き込みが遅く、書き込んだ内容を検証することもできません。最初の1個を書くための道具と考えてください。

ブレッドボードで試作した「鶏と卵問題を解決する書き込み器」
...AVRライタが手元にたくさんあるので、気合が入っていません。😓
このライタでは正確な12MHzは必要なく、4～10MHz程度のセラミック振動子でも問題なく利用できます。また、購入直後のTiny2313なら、出荷時内蔵オシレータに設定されておりセラロック不要です。しかし、HIDaspX用のFUSE設定を行うと、発振器無しでは動作しませんが、FUSE設定が不明のATTiny2313を使う場合には発振器を実装してください。

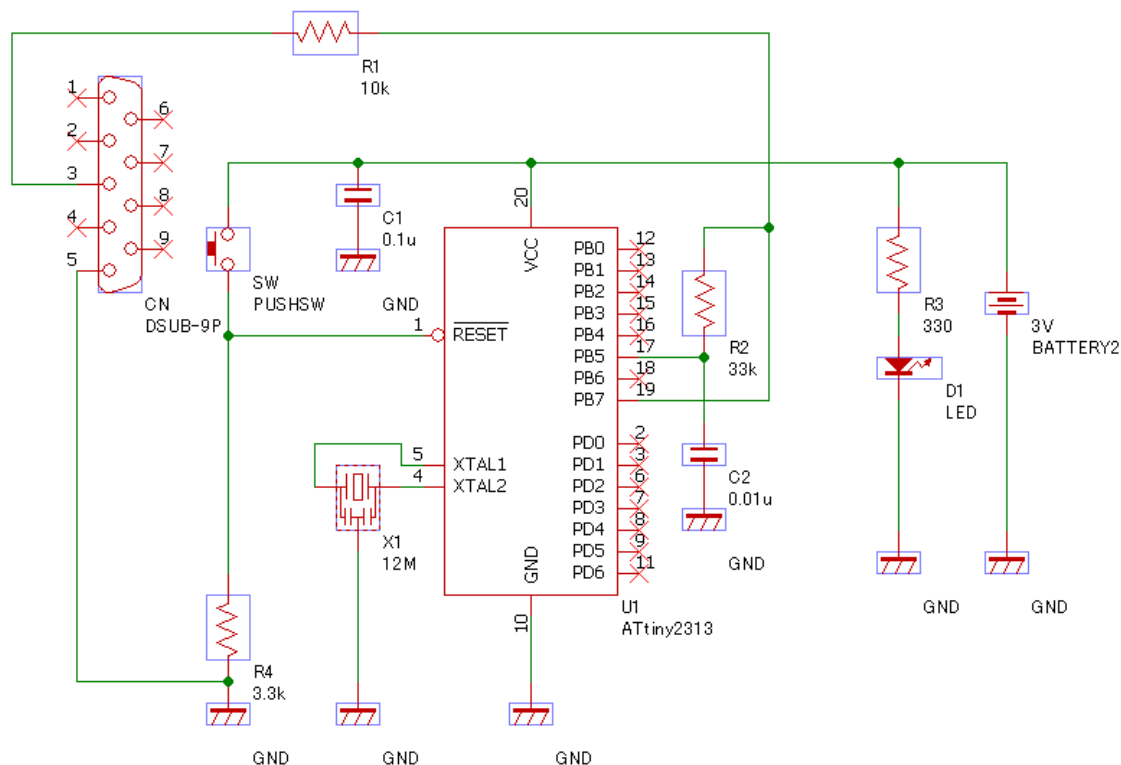
また、通電確認のためのLEDと負荷抵抗の省略や、スイッチ操作に自信があれば、ジャンパー線を手配線で代用することも可能でしょう。

自分なりに工夫した回路で、HIDaspXのファームを書き込んでください。

(ただし、HIDaspXにはクリスタル発振子が必須です)



「鶏と卵問題を解決する書き込み器」の回路図



AVRライター無しでHIDaspx用のAVRマイコン書き込む回路
(RS232Cポートがあれば書き込めます)

2009年2月19日作成

書き込み用のBATファイル

==== egg-write.bat ====

```
@echo off
echo RS232-RC方式の書き込みを開始します。
echo RESET 解除SWを押してから、Enterキーを押下してください
pause
```

```
echo 書き込み終了まで、約40秒です
hidspcx -pf1 -ttiny2313 -fL0xFF -fH0xDB -fX0xFF main-12.hex
echo 書き込み完了しました
```

書き込みの実行例

```
RS232-RC方式の書き込みを開始します。
RESET 解除SWを押してから、Enterキーを押下してください
続行するには何かキーを押してください . . .
書き込み終了まで、約40秒です
Detected device is ATtiny2313.
Erase Flash memory.
Flash memory...
Writing [#####] 2010, 0.02s
Passed.
Fuse Low byte was programmed.
Fuse High byte was programmed.
Fuse Extend byte was programmed.
Progress : 0....1....2....3....4....5....6....7....8....9.... done.
Total read/write size = 2010 B /36.53 s (0.05 kB/s)
書き込み完了しました
```

この書き込みを数回行ない、全て正常にHIDAspx用の制御チップとして機能しました。
ただし、上記のライタは書き込みのみでベリファイを行わないので、HIDAspxが完成後、別のチップにHIDAspxで書き込みを行ない、差し替えることをお勧めします。

kumanさんのページも大変参考になります。

<http://www.geocities.jp/kuman2600/n6programmer.html#10>

↑

USB-HUBの利用で高速化[↑]

HIDAspxの転送速度を向上させるには、OHCIのUSB I/Fを使うほかに、UHCIでもUSB 2.0規格のHUBを介することで、OHCI以上の速度が得られ、格段に使用感は向上します。

HUB経由の利用を初めて行う方は、hidmonのbenchコマンドで転送速度を検証してください。
このテストで異常がなければ、HIDAspxをHUB経由で利用することが可能です。

関連情報 [hidmonの紹介](#)

```
>hidmon
^^^^^^
UHCIのポートに接続
AVR> bench
hid write start
hid write end 38000 10953 s 3469 byte/s
AVR> q
Bye.

UHCIのポートからHUB経由で接続
AVR> bench
hid write start
hid write end 38000 2250 s 16888 byte/s
AVR> q
Bye.
```

このように4～5倍程度の違いが生じます。17kB/sの転送速度は、RS232C(115.2kbps)の速度を上回るものです。

USB 2.0規格のHUB経由で利用すれば、UHCI規格のPCでも転送速度が上がるため、HIDAspx全体の処理速度が向上し、結果としてUSBAspxに迫る性能が得られます。

↑

HIDaspとの出会い(コラム) [†]

先の問題を改善する為、ネット上で公開されている[瓶詰堂さん](#)開発の「HIDを使ったAVRライター(HIDasp)」を試作してみました。

HIDaspは、ATtiny2313のみで構成され、安価に製作でき、ドライバのインストールが不要です。非常に魅力的な仕様を持っていますが、製作して使ってみると、

- Windows 2000ではエラーになり使えない(Windows Vistaでは動作しない)
- かなりの頻度でISP移行時にエラー(AVRマイコンを認識できない)になる
- HIDaspの電源ON直後の通信でエラーになることが多い
- 類似の構成のUSBaspに比べ、処理に時間がかかる
- ISP用の信号がHi-Zになっておらず、回路構成によって不具合が生じる

などの問題を確認しました。改善したいと細かな修正を試みましたが、なかなかうまく行きません。

追記

この件を瓶詰堂さんに報告したところ、HIDasp(≠HIDaspx)の修正版を公開していただきました。以下の問題点が改善されました(私は動作を確認しておりません)。

- Windows 2000で使えない問題
- 電源ON時にリセット状態

siteDev extends PukiWiki 1.4.4 Copyright © 2001-2004 PukiWiki Developers Team. License is GPL.
Based on "PukiWiki" 1.3 by [yu-ji](#) customized by [php spot](#).

