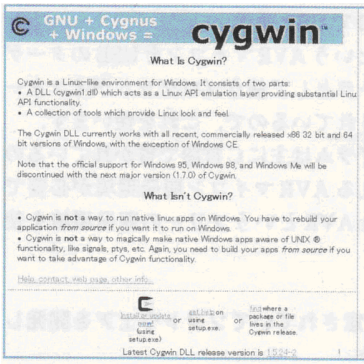

フリーのソフトウェアと汎用マイコンで作る
超低コスト USB I/Oアダプタの製作

武藤 佳恭
Yoshiyasu Takefuji

汎用ワンチップ・マイコンATtiny2313(アトメル)を使ってUSB接続のI/Oアダプタ(以下，USB I/Oアダプタ)を製作し，フリーのソフトウェアCygwin上で開発したCプログラムで直接制御することに成功しましたので報告します(**図1**)．

パソコン側は，Cygwinというフリーのソフトウェアを経由して，C言語のプログラムを動かします．

マイコン側では，やはり同様にC言語で開発したUSB通信のプログラム(ファームウェア)を動かします．WinAVRというフリーのソフトウェアを使ってプログラムを開発しました．

▶簡単なファン・コントローラを作ってみる

例題として簡単なファン・コントローラを作ってみました(**写真1**)．部品点数が少ないので，はんだ付けのいらないブレッドボード上で試してみました．

USB I/Oアダプタに繋いだセンサからの情報をパソコン上のソフトウェア(**図2**)で判断し，設定温度を越えたらAC100 Vの電源を制御してファンをON/OFF

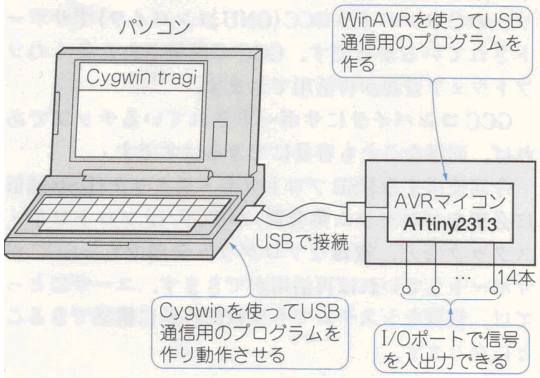

図1　フリーのツールでUSB接続の機器が作れる
C言語やパソコンについての知識があれば低コストで実現可能

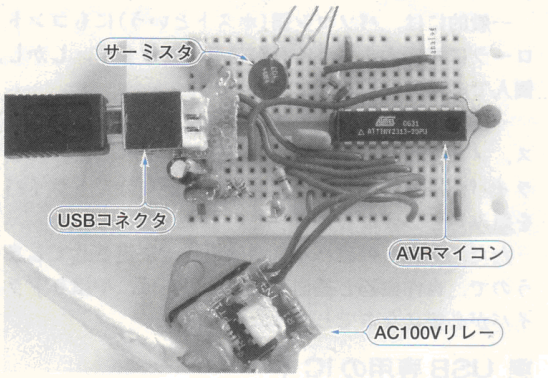

写真1　製作したUSB I/Oアダプタで AC100 Vリレーを ON/OFF制御
パソコンのアプリケーション・ソフトウェアへ温度データを取り込み，同じソフトウェアからAC100 V電源ON/OFF制御するリレーを操作可能

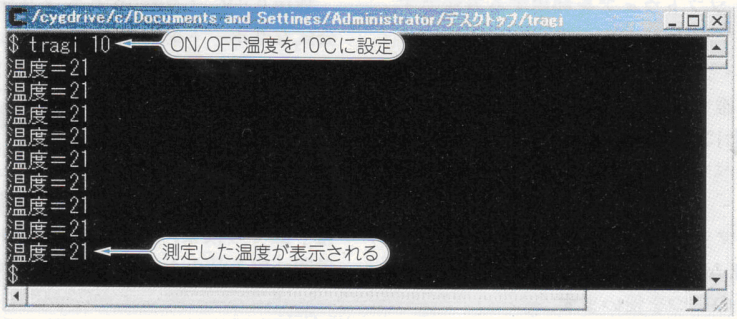

図2　USB I/Oアダプタを制御するアプリケーション・ソフトウェアの画面
ON/OFFの設定温度(10℃)を入力すると温度の計測結果が表示される

します．

ソフトウェアを改良すれば，携帯電話に電子メールで温度情報を送信することもできます．

低コストでUSB機器を作る

昔，パソコンと電子機器をつなぐときはRS-232Cを使うのが一般的でした．しかし，最近のノート・パソコンではRS-232Cのインターフェースが付いていないので，ノート・パソコンに自作の電子機器を接続することは難しくなってきています．

現在，パソコンの標準的なインターフェースはUSBです．携帯電話などの小形機器でもUSBインターフェースが使われるようになってきています．

USB接続の機器を作りにくい理由

機器側に通信用のICが必要

USBを使うには，接続する周辺機器(ターゲットという)のほうにも，USB通信のコントローラICが必要になります．

パソコン側のソフトウェアが開発しにくい

一般的には，パソコン側(ホストという)にもコントローラに対応するドライバが必要になります．しかし，個人での開発は困難です．

一部のよく使うデバイス，例えばキーボードやマウス，メモリ・カード・リーダなどは，OSに標準のドライバが組み込まれていて，ユーザが自分でドライバを組み込む必要はなくなっています．

ただ，それらのデバイスでは用途が限定されてしまうので，自作機器と通信したいと思えば，やはりドライバが必要になるでしょう．

USB専用のICや特定用途のドライバを使わない

専用ICや高価なツールを購入せず，USB機器を作る方法があります．

オープンソース・ソフトウェアを利用する方法です．

オープンソース・ソフトウェアとは

「ソフトウェアは誰でも自由に使用されるべきものである」という理念に基づき，プログラムの中身(ソース・コード)もすべて公開してあるソフトウェアのことです．

今回は，AVRマイコン開発用のWinAVRと，UNIX用のプログラムをWindows上で動作させることができるCygwinの二つを使います．

マイコンのI/OでUSB通信

AVR-USB[1]というAVRマイコンをUSBのターゲット・コントローラとして使うプログラムがインターネット上で公開されているので，これを使います．

ただし，プログラムは主にC言語で書かれているので，C言語が使えるAVRマイコン開発環境が必要です．そこで，WinAVRというフリーの開発環境を使います．

ドライバが用意されていてソフトウェアを開発しやすい環境を使う

Windowsのソフトウェアやドライバをきちんと作ろうとすると面倒です．

UNIXという，パソコンとは使えるプログラムが違うコンピュータがあります．本来はそのUNIX上で動作するプログラムをWindows上で動かすためのソフトウェア(DLLなど)がCygwinです．

ソフトウェアをC言語で記述するなら，Cygwinを使っていることをそれほど意識する必要はありません．

世界中のソフトウェア資産を活用できる

今回アトメルのAVRマイコンを使った理由は，フリーのCコンパイラGCC(GNUコンパイラ)でサポートされているからです．GCCで構築された多くのソフトウェア資源が再活用できます．

GCCコンパイラにサポートされているチップであれば，同様なことも容易にできるはずです．

今回使用するUSBプロトコル・スタック(USB通信に必要なプログラム集合群)やTCP/IPプロトコル・スタックなど，複雑なプログラム資源でも，GCCがサポートしていれば再活用ができます．ユーザにとっては，複雑なシステムでも比較的容易に構築できることになります．

ある程度の自助努力は必要

マイコンのソフトウェアとパソコンのソフトウェア，両方を開発する必要があります．

WinAVR，Cygwinともフリーのツールですから，公式なサポート窓口はありません．トラブルの解決は，基本的には自分で行う必要があります．

パソコンと外部機器とで信号をやりとりしてみる

プログラムや回路の大部分は，慶應義塾大学環境情報学部での授業，春学期「電子おもちゃ設計論」，秋学期「組み込みシステム」の教材の一部です．

USBデバイスはパソコンの要求通りに動作する

パソコンとUSBデバイスは主従関係をもちます．

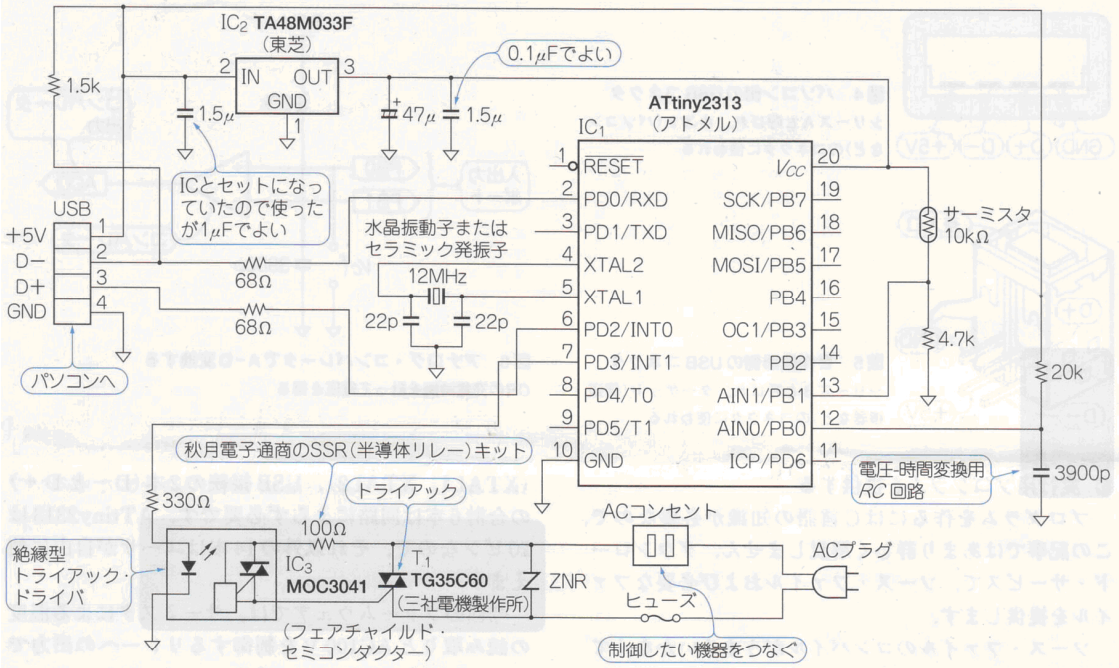

図3　実験したファン・コントローラの回路
USBの通信をAVRマイコンで行っている

一般的には，パソコン側からUSBデバイスへ命令を出します．パソコンとUSBデバイス間の命令の流れは，次のようになります．

① パソコンからUSBデバイスへセンサ情報を要求する．
② USBデバイスは要求されたセンサ情報をパソコンへ転送する．
③ パソコン側でセンサ情報を判断し再びUSBデバイスへ制御命令を送る．同時に，必要があれば，パソコンから直接電子メールでセンサ情報を携帯電話へ送ることもできる．
④ USBデバイスは，パソコンから受け取った命令を実行し，制御を実行する．

USBの仕様として，USBデバイス側からパソコン側に通信を要求することはできません．

1.5 Mbpsのロー・スピードに対応

今回のUSBデバイスは，USB1.1またはUSB2.0のロー・スピードに対応しています．

USB通信を受信するICは，1.5 Mbpsで流れてくるNRZIコードに変換されたデータをデコードし続ける必要があります．

USBホストが自分を呼び出したかどうか判断できないと，正常動作が不可能になってしまうからです．

NRZIのデコードは，1タイミング前のデータと現在のデータとで排他的論理和を取れば可能です．

AVRマイコンはRISCなのでほぼ1クロックに1命令を実行できます．クロックを12 MHzとすれば1秒間に12×10^6回の命令実行が可能ですが，1.5 Mbpsのデータをデコードするのにはビット当たり$12 \div 1.5 = 8$で8命令しか使えません．

これはかなり難易度の高いプログラムです．AVR-USBではうまく工夫して[2]ATtinyのような高速とは言えないマイコンでもきちんと動作させています．

パソコン以外に書き込み器が必要

パソコン以外に必要なハードウェアは，AVRマイコン用のプログラム・ライタです．

私はUSBaspというライタをブレッドボード上で自作して使っていますが，市販のAVRライタを購入してもよいでしょう．購入する目安は，

(1) avrdudeでサポートされているか？
(2) Fusesやlock bitsの設定も可能か？
(3) AVRチップの多くをサポートしているか？

などです．

マイコンのプログラムはファームウェアという概念のものになる

USBホストになるパソコン側のソフトウェアをアプリケーション・ソフトウェア，USBターゲットになるデバイス側の制御のために機器に組み込まれるソフトウェアをファームウェアと呼んでいます．

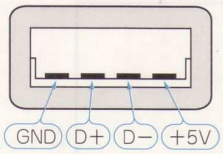

図4　パソコン側のUSBコネクタ
シリーズAと呼ばれ，ホスト(パソコンなど)のコネクタに使われる

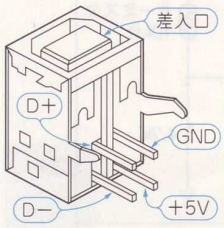

図5　自作機器側のUSBコネクタ
シリーズBと呼ばれ，ターゲット(周辺機器など)のコネクタに使われる

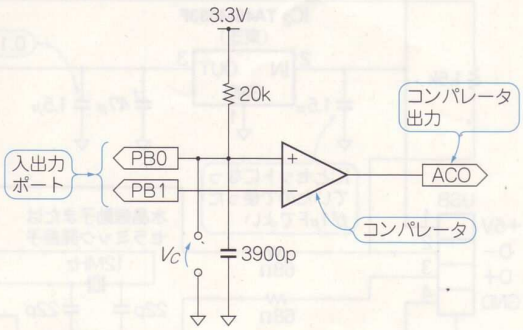

図6　アナログ・コンパレータでA-D変換する
*CR*の充電時間を計って電圧を得る

● 実行用プログラムを提供する

プログラムを作るにはC言語の知識が必要なので，この記事ではあまり詳しく解説しません．ダウンロード・サービスで，ソース・ファイルおよび必要なファイルを提供します．

ソース・ファイルのコンパイルがうまくいかなくても大丈夫なように，Cygwin上で動く実行ファイルtragi.exe，AVRマイコンに書き込むプログラムのhexファイルmain.hex，デバイス・ドライバなども一緒にしておきます．

ハードウェアの作り方

● 回路図はシンプル

回路を**図3**に示します．通信機能はマイコンのソフトウェアで実現しているので，ハードウェアはたいへんシンプルです．

● USBのコネクタについて

パソコンのUSBインターフェースのコネクタを**図4**に示します．シリーズAと呼ばれるコネクタです．

それに対して，これから作るUSBデバイス側は，シリーズBと呼ばれるコネクタ(**図5**)を使います．

先にも書いたように，USBははっきりと主従の関係があるシステムですから，ホスト(主)側は通常シリーズAのコネクタを，デバイス(従)側は通常シリーズBのコネクタをもちます．

よって，今回ATtiny2313のデバイス側のソケットも**図5**のようなシリーズBのコネクタになります．

● マイコンのI/Oピンは14本使える

USB通信に必要な配線は，2本のデータ線(D－とD＋)だけです．D－とD＋の線をマイコンのPD0とPD2にそれぞれ接続します．

電源の2本(＋5VとGND)，水晶発振子の2本(XTAL1，XTAL2)，USB接続の2本(D－とD＋)の合計6本は回路によらず必要です．ATtiny2313は20ピンなので，それ以外の14本はユーザが自由に使えます．

今回のファームウェアでは，サーミスタによる温度の読み取りとAC100Vを制御するリレーへの出力で合計3本の端子を使っています．

● 3端子レギュレータで3.3Vを生成しマイコンを供給

USB通信で飛び交うデータの電気信号は直接マイコンでも受信できるTTLレベルに近い電圧ですが，電圧変動があると簡単に誤動作します．

安定したUSB間の信号のやりとりを実現するために，USBの＋5Vの電圧を3端子レギュレータで3.3Vに変換し，安定な3.3Vの電源電圧でAVRマイコンを動かします．

● サーミスタで温度を検出

温度センサとして，手元にあったサーミスタを使いました．抵抗と直列にして，中点の電圧を測ることで温度変化を取り出せます．今回は特に補正などを考慮していないので，簡易的なものと考えてください．

▶簡易的なA-D変換を使う

今回使ったATtiny2313には，アナログの電圧をデータとして取り込むA-D変換の機能がありません．

PB0とPB1のアナログ・コンパレータを使って，*CR*の充電時間と比較することで，簡易的にA-D変換機能を実現しています．

読み取りたい電圧をPB1(ピン番号13)に，PB0(ピン番号12)に*RC*回路を接続します(**図6**)．

ATtiny2313内のアナログ・コンパレータを使って測定したいアナログ値と比較し，ACOが0から1に変化するまでの時間を測定します(**図7**)．ACOの値は，内部レジスタACSRを使って簡単に読み取れます(ACSR&0x20)．

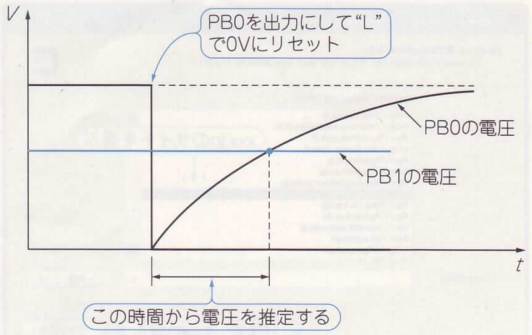

図7　CRの充電電圧がアナログ入力電圧を越えるまでの時間を見る
コンパレータがあればこのような方法でアナログ電圧を読み取れる

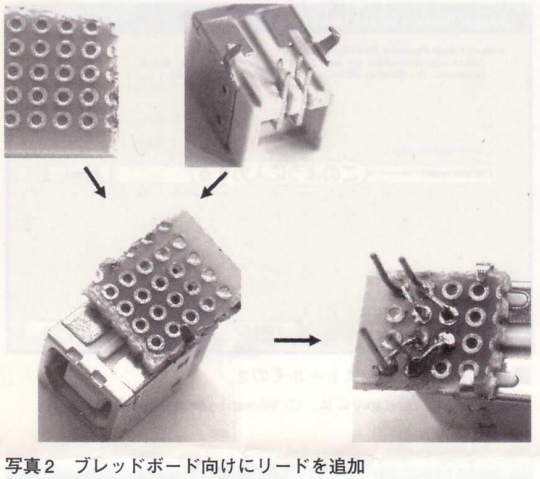

写真2　ブレッドボード向けにリードを追加

● USBまわりに3本の抵抗が必要

USBインターフェースのインピーダンス・マッチングやデバイス認識のため，3本の抵抗を使います．

1.5 kΩの抵抗は，USBのD－信号の5 Vプルアップ抵抗になります．

D－とD＋の線は，それぞれ68 Ωの直列抵抗を入れて，マイコンの端子に接続します．

● ブレッドボードでの注意点

▶USBコネクタにブレッドボード用リードを追加

ブレッドボードに挿せるようにUSBコネクタを**写真2**のようにはんだ付けしました．

スルー・ホール基板の余り(4×4の穴)をニッパなどで切って，抵抗などの部品の切りくずを足に使っています．

▶ブレッドボードの配線には単線を使う

ブレッドボード配線用には0.65 mmの単線を使います．撚り線ではうまくありません．

▶USBコネクタは接着したほうがよさそう

回路が問題なく動くようになったら，エポキシ系の接着剤でUSBコネクタを固めましょう．

ソフトウェア開発用ツールのインストール

Windowsパソコンに，WinAVR(AVRマイコン上のファームウェア開発用)とCygwin(パソコン上のソフトウェア開発用)をインストールします．

Cygwinは，正確には細かなソフトウェアの集まりで，自分の使うパッケージを選んでインストールする必要があります．

今回はC言語でUSB接続のアプリケーション・ソフトウェアを開発します．パッケージとしては，Cコンパイラgcc-coreやUSBのライブラリlibusb-win32などが必要になります．

● Cygwinのインストール

① CygwinのWebサイト(3)からsetup.exeをダウンロードします．

② setup.exeをダブル・クリックすると，インストーラが起動します．［次へ］をクリックすると，Choose A Download Sourceと聞かれるので，［Install from Internet］を選びます．

③ 次の画面のRoot Directoryには，C:¥cygwinと入力します(**図8**)．

④ Local Package Directoryには，C:¥downloadedと入力します(**図9**)．

⑤ Select Your Internet Connectionと聞いてくるので［Direct Connection］を選択します．

⑥ 一瞬接続画面になってWebサイトからサーバのデータをダウンロードしたあと，Choose A Download Siteと聞いてきます．xxx.jpの日本サイトを選択します(**図10**)．

⑧ Select Packagesの画面になります．［view］ボタ

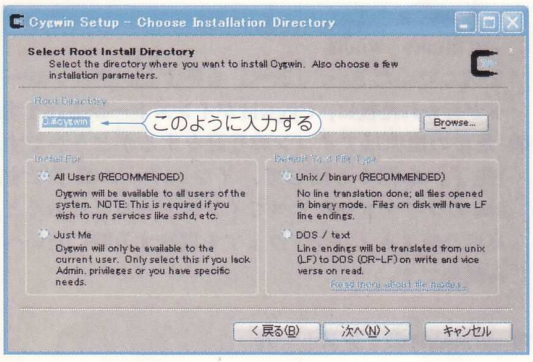

図8　Cygwinのインストールその1
Root DirectoryにはC:¥cygwinと入力する

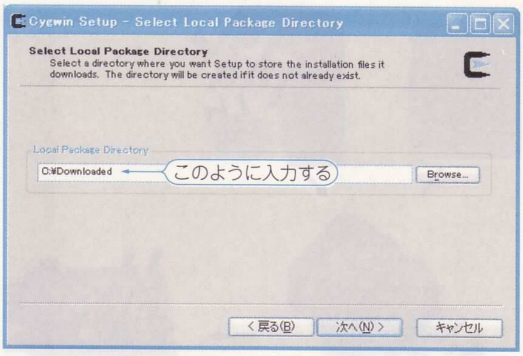

図9　Cygwinのインストールその2
Local Package Directoryには，C:¥downloadedと入力する

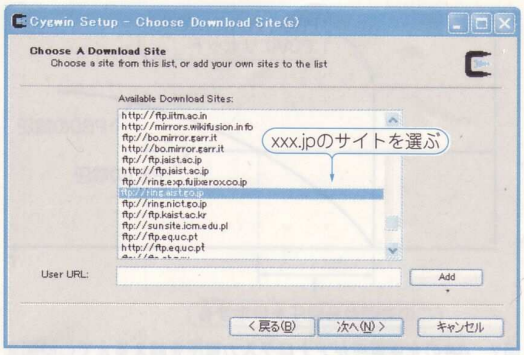

図10　Cygwinのインストールその3
通常は国内のサーバを選ぶほうが高速

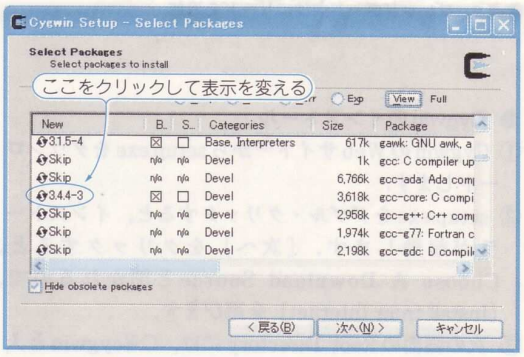

図11　Cygwinのインストールその4
必要なパッケージをインストールするよう設定

図12　WinAVRのインストーラ起動画面
Japanese(日本語)を選択する

ンを何度かクリックして，ボタン右の表示がFullになるよう設定します．

⑨ パッケージがアルファベット順で並んでいるので，必要なパッケージを選びます．パッケージの名前がある行の左端，Skipの部分をクリックすると，バージョン番号に変わります(**図11**)．表示がSkipでなくなっているパッケージがインストールされます．

- gcc - core
- gcc - g + +
- libusb - win32

を選んでください．[次へ]を押すと，ダウンロードが行われた後に，Cygwinがインストールされます．

⑩ 最後の画面はアイコンの登録をするかどうかです．[完了]を押すと閉じて，インストール終了です．libsのライブラリをすべてインストールしておくと，高度なホスト・ソフトウェアを開発する時にたいへん便利です．

● WinAVRのインストール

WinAVRのダウンロード・ページ(4)からインストーラをダウンロードします．

ダウンロードしたファイルをダブル・クリックすると，インストーラが起動します(**図12**)．

最初に日本語を選択すれば，インストール画面は日本語が表示されます．手順はボタンを押していくだけなので，省略します．

マイコン用プログラムのコンパイルと書き込み

マイコンのファームウェアを**リスト1**に示します．

データ転送はusbPoll関数の中で呼び出されているusbFunctionSetup関数を通して行います．

● WinAVRでファームウェアをコンパイル

WinAVRには，エディタとしてProgrammer's Notepadというソフトウェアが用意されています．プログラムの編集とコンパイル作業ができます．

Windowsのスタート・メニューから，[すべてのプログラム] - [WinAVR] - [ProgrammersNotepad]で起動できます．

起動した画面のメニュー・リストで[File] - [Open]と選び(**図13**)，Makefileとmain.cを読み込みます(**図14**)．

メニューの[Tools]から[WinAVR] Make Cleanを実行させ，[WinAVR] Make Allを実行します．

リスト1　AVRマイコンのファームウェア

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include <avr/wdt.h>
#include "usbdrv.h"                    ← USB通信用関数のヘッダ・ファイル

void delay(unsigned int p)
{
    unsigned char i;
    unsigned char j;    //one loop is 3.8us with 12MHz
    for(i=0;i<p;i++) for(j=0;j<10;j++);
}

uchar usbFunctionSetup(uchar data[1])  ← USB通信をしたときの動作をこの関数の中に記述する
{
    static uchar count, replybuf[1]
    usbMsgPtr = replybuf;
    if(data[1] == 0)                   ← 1バイト目の受信データ(変数modeの値)が0なら
    {
        count=0;
        PORTB=0;      // Vc=0          ← CR回路をリセット
        delay(9);
        DDRB=0xfc;    //PB0 and PB1 are set as inputs.
        while ((ACSR&0x20) == 0) {count++;}   ← 時間をカウント
        replybuf[0] = count;           ← 返信用バッファに時間カウント(湿度データ)を入れる
        DDRB=0xfd;
        return 1;
    }
    else if(data[1]==1){PORTD = data[2];}
    return 0;
}                                      ← modeが0でないならポートDの値を2バイト目の受信データ data [2](変数iの値)にする

int main(void)
{
    PORTD = 0;
    PORTB = 0;          /* no pullups on USB and ISP pins */
    DDRD = 0xFA;        /* all outputs except PD2 = INT0 and
                           PD0*/
    DDRB = 0xFD;        /* all output except PB1*/
    ACSR= 0x00;         //analog comparator enabled
    usbInit();          ← USBの初期化
    sei();
    for(;;){usbPoll();}
    return 0;
}                       ← USB通信をする関数．この中でusbFunctionSetup関数が呼ばれる
```

インストールがうまくいっていると，動作が無事に終了し(**図15**)，プログラムをマイコンに書き込むときのファイルmain.hexが生成されます．

● ファームウェアの書き込み

コンパイルしたmain.hexファイルは，プログラムライタを経由してマイコンへ書き込みます．

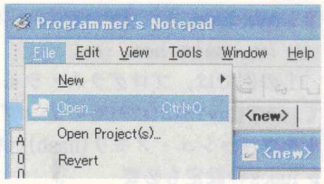

図13　WinAVRのエディタProgrammer's Notepad
[すべてのプログラム]-[WinAVR]-[ProgrammersNotepad]で起動

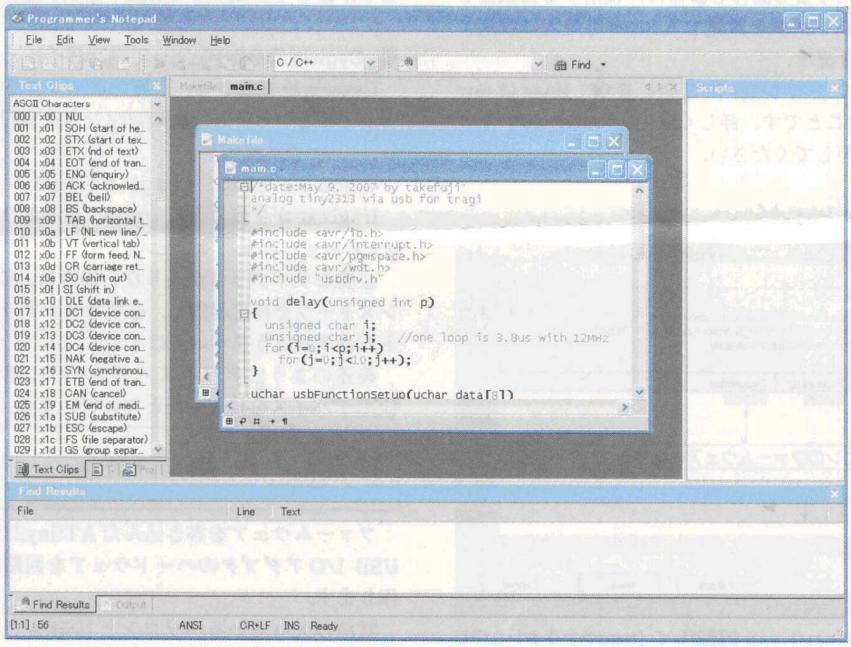

図14　Makefileとmain.cの二つのファイルを開く
[File]-[Open]で出てくるダイアログでファイルを指定する

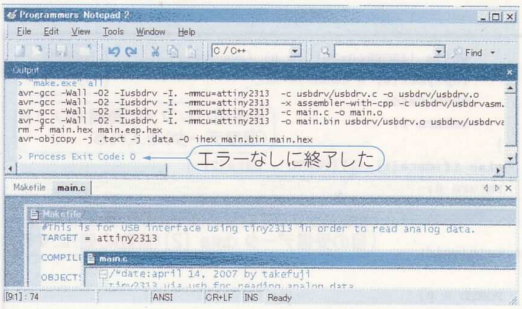

図15　コンパイルに成功したときの表示

Process Exit Code: 0と表示される

書き込みには，やはりオープンソースのavrdudeというプログラムを使います．これはWinAVRに含まれています．

avrdudeを使ってmain.hexをマイコンのATtiny2313に書き込むには，次のようなコマンドをCygwin上で実行します．

```
avrdude -c usbasp -p t2313 -U flash
:w :main.hex :a
```

上の例では，プログラム・ライタ(USBasp)を用いて，ATtiny2313(t2313)のチップへmain.hexファイルをフラッシュ・メモリ(flash)へ書き込んでいます．

▶ fusesの設定も必要

これでプログラムそのものは書き終わりましたが，これではまだうまく動作しません．使用する発振器などの動作の基本情報を設定するfusesを適切に設定する必要があります．

次のコマンドでfuses設定をします．lfuseはlow byte fuseのことです．詳しくは，ATtiny2313のマニュアルを参照してください．

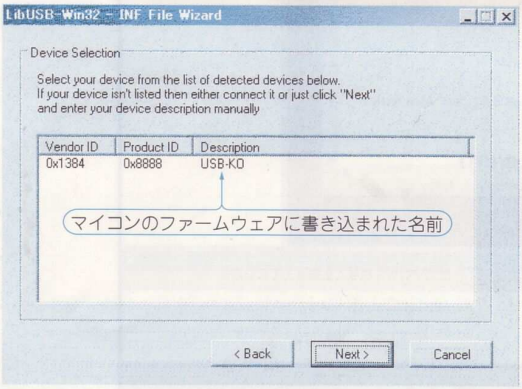

図16　inf-wizard.exeを起動して［Next＞］ボタンを押した画面

inf-wizard.exeはC:¥cygwin¥lib¥libusb¥にある

```
avrdude -c usbasp -p t2313 -U lfuse
:w :0xef :m
```

fusesの設定は，ICごとに違うので注意が必要です．

今回の例では，クロックは外部に12 MHzのセラミック発振子を使用するので，lfuse = 0xefとなります．

このような外部クロック供給の設定をすると，外部からクロックを供給しない限りfusesの書き換えはできなくなります．

fuses設定を使うと，外部Reset入力のピンもI/Oに使える設定にできますが，ごく一部のプログラマでしか書き換えできなくなるので，外部Reset入力はなるべくディセーブルしないほうがよいでしょう．

● 製品にはベンダIDとプロダクトIDの取得が必要

USBデバイスを販売するには，すべての機器で重複しないようにベンダIDおよびプロダクトIDの付与が必要です．

実験に使う程度であれば，今回のファームウェアをそのまま使ってもかまいませんが，製品として販売するならば正式なIDの取得が必要です．

パソコンで動かすアプリケーション・ソフトウェアの作成

● パソコンのアプリケーション・ソフトウェア

パソコン側のソフトウェアを**リスト2**に示します．

libusbの基本関数を使ってUSB通信を実現しています．詳しくはlibusbの解説サイト(5)を参照ください．

● Cygwinの GCC を使ってコンパイル

Cygwinを起動させ，現在のディレクトリにtragi.cファイルをコピーし，次のコマンドを実行してください．

```
gcc -o tragi tragi.c -lusb
```

tragi.exeが生成できていれば，libusbライブラリやgccの設定に問題ありません．

必要なライブラリが足りない場合は，Cygwinのインストーラを再度起動し，必要なライブラリをインストールすればうまくいくはずです．

後述のようにデバイス・ドライバに関する作業は，USBデバイスを接続したときに行います．

動作テスト

ファームウェアを書き込んだATtiny2313を使って，USB I/Oアダプタのハードウェアを回路図どおりに作ります．

電源の入っているパソコンとUSB I/Oアダプタを USBケーブルでつなぐと，パソコンはUSBデバイスを認識し，新しいデバイス・ドライバを要求してきま

リスト2　パソコンで動かすアプリケーション・ソフトウェア(GCCでコンパイルする)

```
#include <usb.h>
#include <stdio.h>
#include <string.h>
#define VENDOR_ID   0x1384                  /*VID must be changed. */
#define PRODUCT_ID  0x8888                  /*PID must be changed. */
static int usbOpenDevice(usb_dev_handle **device, int idvendor, int idproduct)
{
        struct usb_bus *bus;
        struct usb_device *dev;
        usb_dev_handle *udh=NULL;
        int retp, retm,errors;
        char string[256];
        usb_init();
        usb_find_busses();
        usb_find_devices();
        for (bus = usb_busses; bus; bus = bus->next)
          for (dev = bus->devices; dev; dev = dev->next)
                {   udh=usb_open(dev);
                    retp = usb_get_string_simple(udh, dev->descriptor.iProduct, string, sizeof(string));
                    retm=usb_get_string_simple(udh, dev->descriptor.iManufacturer, string, sizeof(string));
                    if (retp > 0 && retm > 0)
                    if (idvendor==dev->descriptor.idVendor && idproduct==dev->descriptor.idProduct)
                                                                { *device=udh;return errors=0;}
                }
        usb_close(udh);return errors=1;
}

int main(int argc, char **argv)
{       usb_dev_handle *d=NULL;
        unsigned char buffer[8];
        int i=0,j,mode=0,ret;
        if(argc<2){printf("you need temperature data\n");return 0;}
        j=2*atoi(argv[1]);
        usb_init();
        for(;;)
        {       mode=0;i=0;
                ret=usbOpenDevice(&d,VENDOR_ID,PRODUCT_ID);
                if(ret!=0){ printf("usbOpenDevice failed %d \n",ret); return 0;}
                ret=usb_control_msg(d, USB_TYPE_VENDOR | USB_RECIP_DEVICE | USB_ENDPOINT_IN,
                        mode, i, 0, (char *)buffer, sizeof(buffer), 5000);
                printf("温度=%d \n", buffer[0]/2);
                if(buffer[0]>=j)
                {       mode=1;i=0x08;
                        ret=usbOpenDevice(&d,VENDOR_ID,PRODUCT_ID);
                if(ret!=0){ printf("usbOpenDevice failed\n"); return 0;}
                ret=usb_control_msg(d, USB_TYPE_VENDOR | USB_RECIP_DEVICE | USB_ENDPOINT_IN, mode, i, 0,
                (char *) buffer, sizeof(buffer), 5000);
                }
                else
                {
                        mode=1;i=0x00;
                        ret=usbOpenDevice(&d,VENDOR_ID,PRODUCT_ID);
                        ret=usb_control_msg(d, USB_TYPE_VENDOR | USB_RECIP_DEVICE | USB_ENDPOINT_IN,
                        mode, i, 0, (char *)buffer, sizeof(buffer), 5000);
                }
        }
        return 0;
}
```


す．デバイス・ドライバそのものは，Cygwinのlibusbパッケージに含まれていますが，デバイスの情報ファイル(infファイル)が必要です．

● デバイス・ドライバ用infファイルの作成

infファイルの作成にはCygwinのパッケージlibusbの中に含まれているプログラムを使います．筆者のインストール環境では，libusbは次のディレクトリになります．

C:¥cygwin¥lib¥libusb

① libusb内のinf-wizard.exeをダブル・クリックして実行します．起動画面で［Next>］を押すと，**図16**のような画面が現れるはずです．もし他にもUSBデバイスを接続していると，そのデバイスも表示されます．

② 今回製作したデバイスであるUSB-KOを選んで

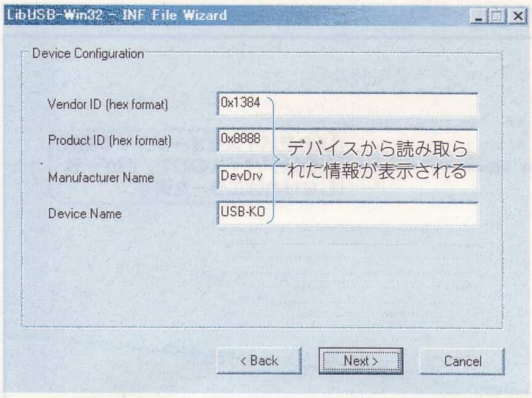

図17　USB－KOのIDなどが表示される
USB－KOを選んで［Next＞］ボタンを押すとこの表示になる

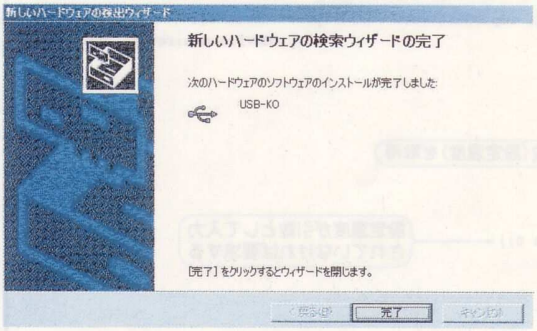

図19　デバイス・ドライバが正常に組み込まれたときの画面

［Next>］ボタンをクリックします．

③ **図17**のようなIDの表示画面に変わります．さらに［Next>］ボタンをクリックすると，ファイルの保存ダイアログ・ボックスが現れるので，ファイル名をusb－ko.infに変更して保存してください．

④ デバイス・ドライバを要求してきたときは，作成したusb－ko.infを読み込ませればドライバが認識されUSBデバイスが動きます．

● デバイス・ドライバのインストール

USBデバイスを認識したとき表示される画面（**図18**）で，［一覧または特定の場所からインストールする］を選択し，［次へ］ボタンをクリックします．

フォルダ指定を要求されるので，先ほど生成したinfファイルを保存したディレクトリを参照します．**図19**のような画面が出れば，インストールは成功です．

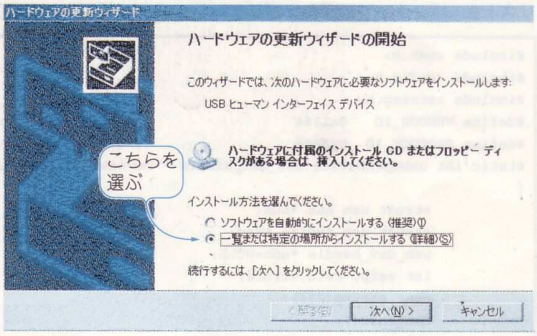

図18　デバイス・ドライバの要求画面

● 動作の確認

インストールが完了していれば，すでに使える状態になっています．Cygwinをダブル・クリックして起動し，次のコマンドを実行してください．

```
tragi 10
```

これは，tragi.exeを10（設定温度10℃）というオプションで起動するというコマンドです．**図2**のように温度を表示し，AC100 Vの電源をONにするはずです．

表示温度よりも1度高く，例えばtragi 22とすると，AC100 Vの電源はOFFになります．この状態でサーミスタに息を吹きかけると，電源が再びONになります．

発展した応用として，ダウンロード・ファイルには，センサ温度をファイルにログ保存し，特定の設定された温度で電子メールを出せるプログラムも含めています．参考にしてみてください．

■プログラムの入手方法
筆者のご厚意により，この記事の関連プログラムを小誌ホームページに登録する予定です．（編集部）
http://www.cqpub.co.jp/toragi/

◆参考文献◆

(1) USB－AVR
http://www.obdev.at/products/avrusb/index.html
(2) Implementing USB 1.1 in Firmware
http://www.obdev.at/developers/articles/00003.html
(3) Cygwin Information and Installation
http://www.cygwin.com/
(4) SourceForge.net : WinAVR
http://sourceforge.net/projects/winavr/
(5) LibUsb－Win32
http://libusb－win32.sourceforge.net/