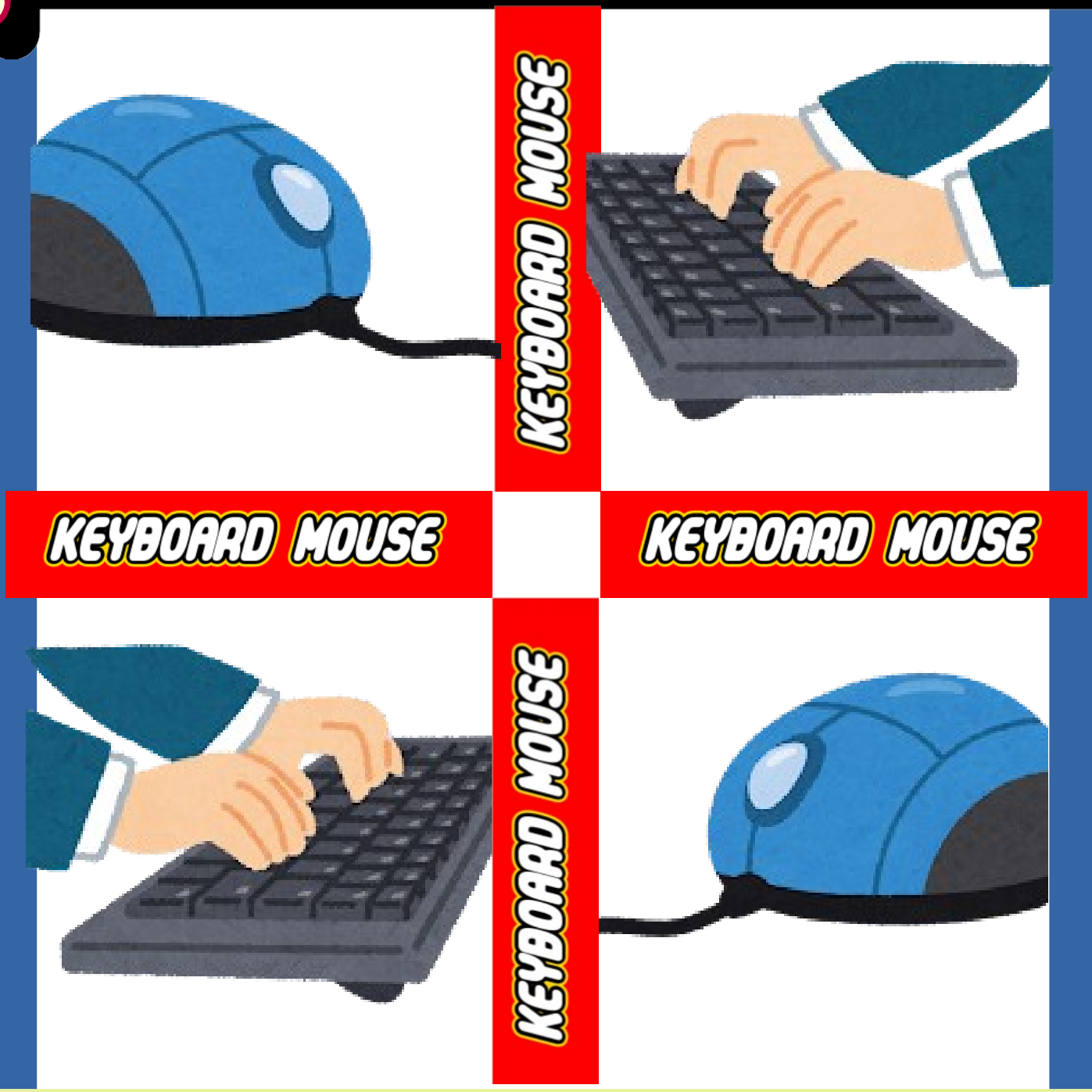
キーボード／マウス エミュレータ 解説書

～CH9329 活用シリーズ～



PRESENT BY みんなのラボ

Written by ICHI

－はじめに－

このたびは弊サークルのプロダクトを手に取って頂き、ありがとうございます。
商品には万全を期しておりますが、製作・使用する場合は御自身の責任において安全に充分注意して行ってください。
内容がもとで不利益・不具合が生じても一切の責任を負いません。あらかじめご了承ください。

本書は下記の環境で動作確認をしておりますが、全ての環境で動作を保証するものではありません。
動作しない場合はサンプルプログラムなどを解析して、御自分の環境に合わせた変更を実施願います。

Python 環境(USB 接続版)

Windows10(64bit)＋Python3.7(Pyserial を使用)

Android スマートフォン(Android9、OTG 対応)

Arduino 環境(シリアル接続版)

Arduino UNO R3(互換機)＋Arduino IDE(Windows10)

またサンプルプログラムや資料などは下記サポートサイトに保管しておりますので参照願います。

https://sites.google.com/site/ichiworkspace/

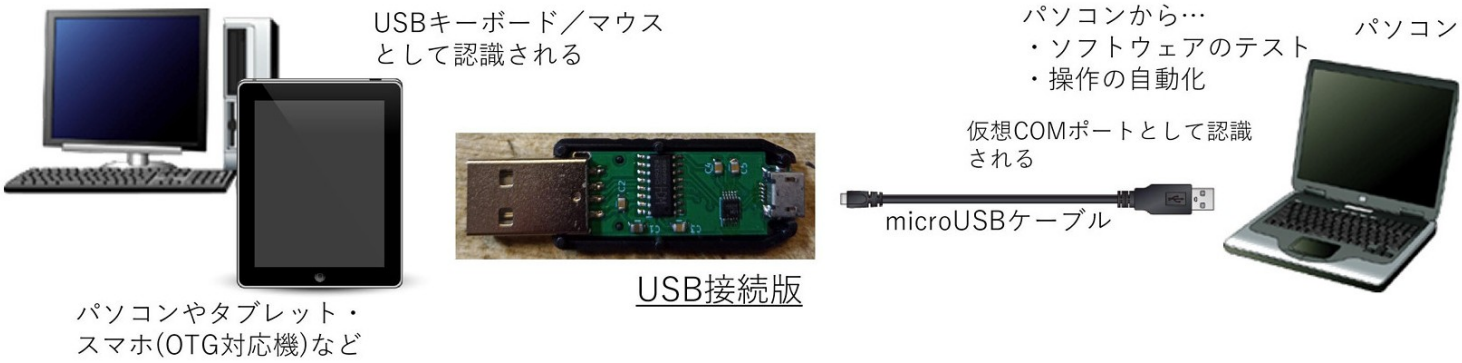
みんなのラボ http://minnanolab.net/
制作 ICHI (Twitter @atsuyuki1kawa)

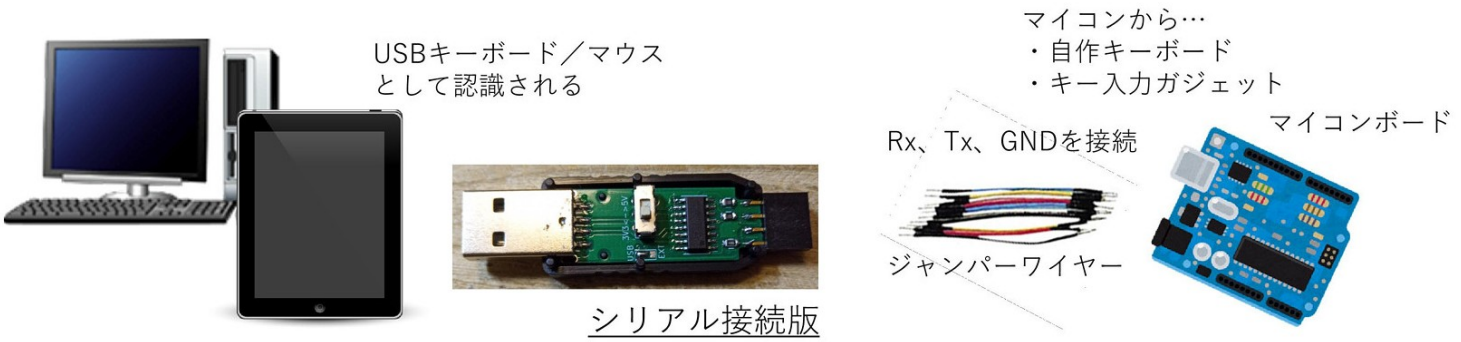
目次

1　ハードウェア解説

本プロダクトは CH9329 という、USB キーボード／マウスの機能をシリアル回線からコマンドを送信できるチップを利用したものです。接続されたパソコン側からは USB キーボード／マウスが接続されたように認識されており、任意のキー／マウス操作の情報を送信することが出来ます。

プロダクトは下記の 2 種類があります。用途に応じて使い分けて下さい。





1.1　USB 接続版

キーボード・マウス機能を模して(エミュレートして)外部からパソコンやタブレットなどを外部から操作する用途に向いています。パソコン操作の自動化やソフトウェアのテスト、マクロ入力の簡便化に向いています。

USB 接続版を使用する際は別途に microUSB ケーブル(通信用)を用意して下さい。本プロダクトをパソコンに接続すると自動的にドライバがインストールされます。(ホスト側、クライアント側ともに)

認識後は操作元のパソコンの仮想 COM ポートを使ってキー／マウスの情報を送信します。デバイスマネージャを利用して COM ポートの番号を控えておいて下さい。

操作対象パソコンの
USB ポートに接続

操作元のパソコンに接続
(microUSB ケーブル)

1.2 シリアル接続版

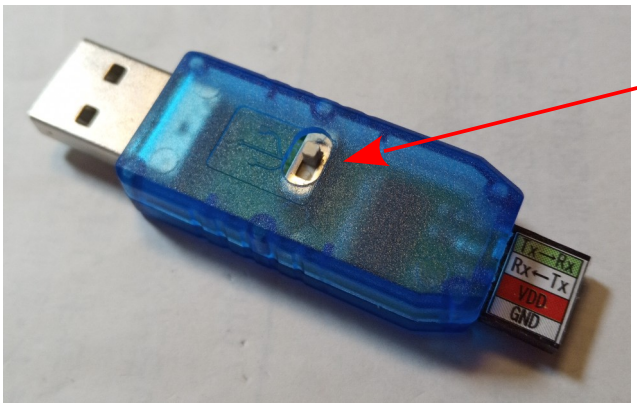
マイコンのシリアル通信機能(UART)を利用してキーボード／マウス機能を利用できます。これを用いることで USB 機能を持たないマイコンでもキーボード／マウス機能を簡単に実装できるようになります

ユーザーがお好きなマイコンを使って自作キーボードを作ったり、キーボード入力機能を追加する用途に向いています。自作キーボード／マウスだけでなく、簡易的なキーボード／マウス機能を持つガジェット製作に向いたプロダクトです。

また、USB 接続時の V-ID/P-ID(ベンダー ID／プロダクト ID)はメーカー(WCH 社)正規の物を利用するので実験用に適当な V-ID/P-ID をわざわざ用意する必要もありません。

シリアル接続版はジャンパーワイヤー(最低 3 本、別途用意)を使ってマイコンと接続します。USB 側からマイコンの電源を供給する場合は Vcc も接続して下さい。マイコン側のシリアル回線の電圧により 5V(USB 電圧に依存)と 3.3V を選択可能です。安全対策のため電流制限用のヒューズ(定格 500mA)を内蔵していますが完全に短絡を防ぐものではないので御注意下さい。

操作対象パソコンの
USB ポートに接続



電源電圧変更スイッチ
(3.3V ⇔ 5V)

操作元のマイコンに接続
(ジャンパーワイヤー)

ラベル	接続先
Tx → Rx	マイコンの Rx 端子へ接続
Rx ← Tx	マイコンの Tx 端子へ接続
Vcc	マイコンの電源端子へ接続 (注：USB 側から電源を供給する場合のみ)
GND	マイコンの GND 端子へ接続

2　通信プロトコル解説

パソコン側へキーボード／マウスの操作情報を送信する際はパケットと呼ばれる小さなブロックに収めて送信します。メーカーが作成した通信プロトコルの解説書(中国語)を翻訳したものをサポートサイトに保管していますので、こちらも合わせて参照してください)

パケットのフォーマット(書式)は基本的にフレームヘッダ(2 バイト)＋識別アドレス(1 バイト)＋コマンド(1 バイト)＋データ長(1 バイト)＋データ本体(0～64 バイト)＋チェックサムで構成されています。

フレームヘッダ	アドレス	コマンド	データ長	データ	累積和 チェックサム
HEAD	ADDR	CMD	LEN	DATA	SUM
2 バイト	1 バイト	1 バイト	1 バイト	N バイト (0～64)	1 バイト

先頭にあるフレームヘッダ(0x57 0xAB)はパケットを識別する記号となっており、アドレスは複数のチップをを使い分ける際に使用(チップが 1 個ならば 0 で固定)します。通常これらは固定と考えて下さい。

コマンドキーボード／マウスの操作情報に応じて使い分けます。本書で主に解説するのは以下のコマンドとなります。(詳しくは通信プロトコルの説明を参照して下さい)

コマンド	説明
0x02	USB キーボードの通常データを送信する
0x03	USB キーボードのマルチメディアデータを送信する
0x04	USB マウスの絶対座標のデータを送信する
0x05	USB マウスの相対座標のデータを送信する

コマンド毎に必要なデータが異なりますので、それに応じたデータ長(バイト数)を指定し、続けてデータ長の分だけデータを記載します。

最後のチェックサムはフレームヘッダからデータ本体までの全データを加算して下 2 ケタ(16 進数)を取り出したものです。これが誤っているとコマンドが実行されません。

以上で送信パケットの説明となります。基本的に全てのコマンドはこのようなパケットを作成して送信することで実行できます。

送信パケットを送信するとチップはコマンドの実行結果などを返信します。この返信パケット(7 バイト)はコマンド実行状況(エラー情報)が含まれて返ってきますが、本書のサンプルでは無視(読み取りバッファを強制的にクリアしている)しています。本書では割愛しますが、コマンドのエラーなどを確認したい場合は返信パケットを解析するプログラムを用意して下さい。

2.1　キー入力

通常のキー入力方法について説明します。

通常キーと同時に押す特殊()キー(Shift や Ctrl など)を指定でき、最大で通常キー 6 個、特殊キー 4 個(1 バイトで表現)を同時に押すことができます。

パケットの構成は以下の通りです。コマンドは 0x01 で、データ長は 8 バイトとなります。まず装飾するための特殊キーは 1 バイトの中のビットを立てる(=1)ことで指定します。ビットが 0 ならば押していないことになります。

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
右 Windows キー	右 Alt キー	右 Shift キー	右 Ctrl キー	左 Windows キー	左 Alt キー	左 Shift キー	左 Ctrl キー

特殊キーと通常キーの間には 1 バイト(0x00 固定)が入り、その後で通常キーに対応するコード番号(6 バイト分)を指定します。

キー入力のパケット作成に必要なデータは以下のような構成となっています。ここで注意が必要なことですが、キー入力コマンドはキーを「押しっぱなし」扱いになるので、キーを「押す」と「離す(何もキーを押していないデータを送信する)」でワンセットの操作となります。

具体的には以下のような流れとなります。

① Shift キーと「a」キーのコードを送信する(これで Shift と「a」キーは「押したまま」状態になる)

パケット「0x57 0xAB 0x00 0x02 0x08 0x02 0x00 0x04 0x00 0x00 0x00 0x00 0x00 0x12」

②次に何もキーが押されていないコード(キー情報が全て 0)を送信する(これで Shift と「a」キーが「離された」状態になる)

パケット「0x57 0xAB 0x00 0x02 0x08 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x0C」

③パソコン的にはキーボードで Shift を押しながら「a」キーを 1 回タイプした(その後で Shift を離した)ことになる

上記の流れを組み合わせることで「Ctrl と Alt と Delete を同時に押す」とか「Shift を押したままでスペースキーを押す(でも Shift は押したまま)」といったキー操作が可能になります。

2.2 メディアキー入力

次にメディアキー(機能キー)の入力方法について説明します。

例としては、音楽プレイヤーの操作(ミュートやボリューム操作など)や、ウェブブラウザの操作(戻るなど)の特殊機能を使用できます。

パケット作成に必要なデータは以下の通りです。コマンドは 0x03 となっており、データ長は 4 バイトです。データの中身は 4 バイト分のメディアキーの状態です。先ほどの特殊キー入力を指定するときのようにキーが該当するビットを立てる(=1)ことでメディアキーを押したことになります。詳しくは通信プロトコルのデータシートの付録を参照して下さい。

手順としては通常キーと同様に「キーを押す」「キーを離す」という一連の動作が必要です。

①メディアキー「Mute」を押す

パケット「0x57 0xAB 0x00 0x03 0x04 0x02 0x04 0x00 0x00 0x0F」

②何もメディアキーが押されていない

パケット「0x57 0xAB 0x00 0x03 0x04 0x02 0x00 0x00 0x00 0x0B」

2.3 マウス操作(絶対位置)

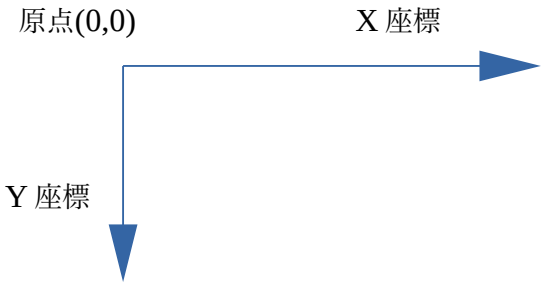
このチップの大きな特徴であるマウス機能について説明します。

マウスカーソルを画面の任意の位置(座標)へ移動します。移動する際に一緒にマウスボタンを押したままにすること(要するにドラッグ)も可能です。

コマンドは 0x04 で、データ長は 7 バイトです。データの先頭に 1 バイト(0x02 固定)があり、続いてマウスボタンの情報を指定します。キー入力を装飾する特殊キーのように該当するボタンのビットを 1 バイトの中のビットを立てる(=1)ことで指定します。ビットが 0 ならば押していないことになります。(BIT3～7 は 0 で固定)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
0	0	0	0	0	中ボタン	右ボタン	左ボタン

続いてマウスカーソルを移動させる情報を指定(3～6 バイト)します。ここで座標指定に計算が必要となりますので注意してください。まずは操作する画面の解像度(1920×1080 など)を確認してください。画面上の座標軸は左上が(0,0)となり、移動先のマウスカーソルの座標を(X,Y)とします。式中の 4096 は固定です。



$X_C = (4096 \times X) \div$ 解像度 X　⇒3 バイト目に X_C の下位バイト、4 バイト目に X_C の上位バイト

$Y_C = (4096 \times Y) \div$ 解像度 Y　⇒5 バイト目に Y_C の下位バイト、6 バイト目に Y_C の上位バイト

最後の 7 バイト目はマウススクロールの指定です。スクロール量は±127(単位はスクロール部のエンコーダカウント)の範囲です。0 ならばスクロールしません。マイナスで下方向に、プラスで上方向にスクロールします。0 ならばスクロールしません。マイナス(下方向)の場合は 0x100 ＋ スクロール量で補正して下さい。

なお、マウスカーソルの座標を調べる際は MPPUtility というアプリが便利です。詳細は「5 引用文献」を参照して下さい。

2.4 マウス操作(相対位置)

マウスカーソルを現在位置(座標)から相対的に移動(単位ピクセル)させることができます。移動量が(0,0)の場合はマウスカーソルは現在位置を維持します。

コマンドは 0x05 で、データ長は 5 バイトです。データの先頭に 1 バイト(0x01 固定)があり、続いてマウスボタンの情報を指定します。指定方法は前項のマウス絶対位置と同じです。

続いてマウスカーソルの移動量(単位はピクセル)は-128 ～ +127 の範囲で指定します。マイナスで X 軸は左方向、Y 軸は上方向、プラスで X 軸は右方向、Y 軸は下方向に移動します。マイナスの場合は 0x100 ＋ 移動量(ピクセル)で補正して下さい。3 バイト目に X 軸方向の移動量、4 バイト目に Y 軸方向の移動量を指定します。

最後の 5 バイト目は前項と同じくマウススクロールの指定です。0 ならばスクロールしません。

このコマンドでは移動量を(0,0)にすればマウスカーソルは移動しませんので、マウスボタンだけを押す操作(要するにマウスクリック)をする場合に向いています。その場合はキー入力と同様にマウスボタンを押す→離す操作を忘れずに組み合わせて下さい。

次章より各コマンドを使ったサンプルプログラムを紹介します。コマンドの詳細については通信プロトコルのデータシート(筆者が和訳したものがサポートサイトにあります)を参照願います。

3　サンプルプログラム

本書では通信プロトコルのデータシート(サポートサイトに和訳したものを置いてあります)をもとにサンプロプログラムを準備しました。

サンプルプログラムでは基本的な機能しか使用していませんので、皆さんの目的に合わせて適宜改造して使用して下さい。

3.1　USB 接続版(Python 環境)

本デバイスを Python 環境で使用する場合は pyserial ライブラリを使用します。事前に「pip install pyserial」でインストールして下さい。(Python のインストール方法は使用環境に合わせて下さい)

pyserial でデータを送信する際の注意は「バイト型データ列」として送信することです。具体的には下記のように pyserial.write メソッド(受信は pyserial.read メソッド)を利用します。

```
def sendpacket(self,data):
    self.ser.write(bytes(data))
    time.sleep(0.02)
    return self.ser.read(7)
```

サンプルプログラム内で使用している定数についてはソースコードをご覧下さい。

サンプルプログラムでは本デバイス専用のクラスを用いてオブジェクトを作成して各機能を利用します。オブジェクト作成時のパラメータとしては本デバイスを接続した COM ポート番号、通信速度(標準 9600bps、変更する場合はデバイス本体の設定を変更する必要があります)、画面の解像度(X,Y)を指定します。

```
drv=CH9329('COM4',9600,1920,1080)
```

以下、CH9329 クラスで定義したメソッドの使い方を紹介します。このまま使っても結構ですし、ソースコードを拡張して御自分の用途に改造して下さい。(詳細は CH9329.py を参照して下さい)

1. write メソッド

   1 文字キーをタイプする機能です。通常のキー以外(Enter キーなど)も入力できます。

   ```
   drv.write(“A”)
   drv.write(CH9329.ENTER)
   ```

2. print メソッド

   任意の文字列を入力する機能です。通常のキーのみ対応しています。

   ```
   drv.print(“Hello,world.”)
   ```

3. push メソッド

   直接キーコードを送信する機能です。例では Shift キーを押しながら「A」キーをタイプすることを表します。キーコードは「4 資料」のキーコードを指定して下さい。write、print メソッドは最終的には push メソッドを使用しています。

   ```
   drv.push(0x02,0x04)
   ```

4. media メソッド

   機能キーを入力します。例としては音量操作(ミュートやボリュームアップ)を実行しています。

```
drv.media(CH9329.MUTE)
drv.media(CH9329.VOLUMEP)
```

5. moveabs メソッド

   マウスカーソルを絶対座標へ移動する機能です。マウスカーソルを移動させる先の座標を指定して下さい。

```
drv.moveabs(100,200)
```

6. moverel メソッド

   マウスカーソルを現在座標から相対移動する機能です。マウスカーソルを移動させる量(単位ピクセル)指定して下さい。

```
drv.moverel(-30,0)
drv.moverel(0,50)
```

7. click メソッド

   マウスボタンを操作する機能です。押すボタン(右、左、中)を指定します。

```
drv.click(CH9329.L_BTN)
drv.click(CH9329.R_BTN)
drv.click(CH9329.M_BTN)
```

8. scroll メソッド

   マウスホイールを操作する機能です。ホイールを回転させる量(単位エンコーダカウント)を正または負で指定します。

```
drv.scroll(1)
drv.scroll(-10)
```

最後に、クラスを用いたプログラム例です。コメントを参考にして下さい。

```
drv=CH9329('COM4',9600,1920,1080)   #CH9329 オブジェクト作成
drv.write(“ABC”)                    #文字列「ABC」を入力
drv.write(CH9329.ENTER)             #Enter キーを入力
drv.moveabs(100,100)                #マウスカーソルを座標(100,100)へ移動
drv.click(CH9329.L_BTN)             #マウス左ボタンをクリック
drv.moverel(100,0)                  #マウスカーソルを右へ 100 ピクセル移動
drv.click(CH9329.R_BTN)             #マウス左ボタンをクリック
drv.push(0x05,0x4C)                 #Ctrl+Alt+Delete 同時押し
drv.close()                         #COM ポートを閉じる(無くても可)
```

3.2 シリアル接続版(Arduino 環境)

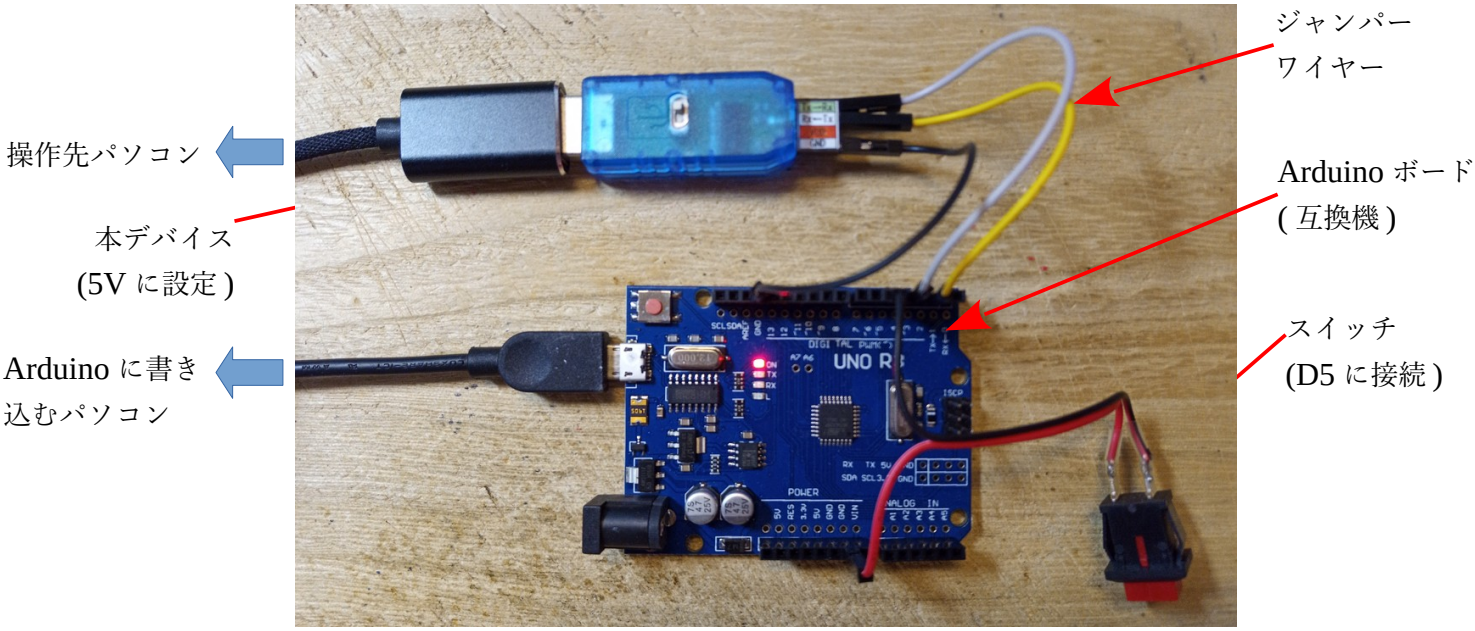
ここでは本デバイスを Arduino UNO R3(オリジナルでも互換機でも構いません、以下 Arduino と表記します)で使用する前提で説明します。なお、Arduino 自体の環境構築などについては割愛させて頂きますので、Arduino IDE から Arduino ボードにプログラムを書き込んで実行できる環境を準備して下さい。

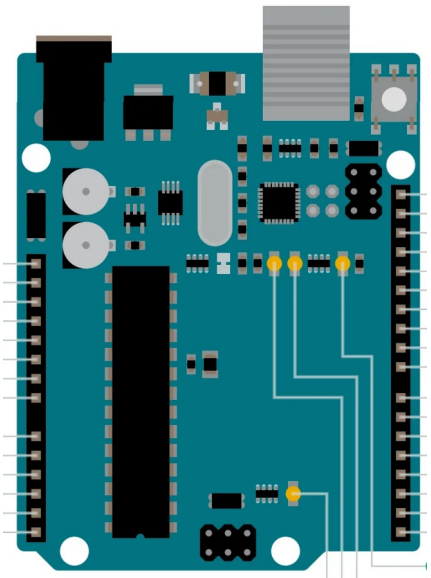
ここでは Arduino ボード からシリアル回線を利用するために SoftwareSerial ライブラリを使用します。Arduino では通常ピン 0,1 が Rx,Tx に割り当てられてプログラム転送などに使用していますが、SoftwareSerial ライブラリを使用することで任意のピンを Rx,Tx に割り当てることが可能です。(SoftwareSerial ライブラリ自体の説明はここでは割愛します)

マイコンから本デバイスへデータを送信する際は前述の Python 環境と同様に「バイト型データ列」として送信することです。具体的には softwareserial.write メソッド(受信は softwareserial.read メソッド)を利用します。SoftwareSerial オブジェクトの作成は下記のようにします。

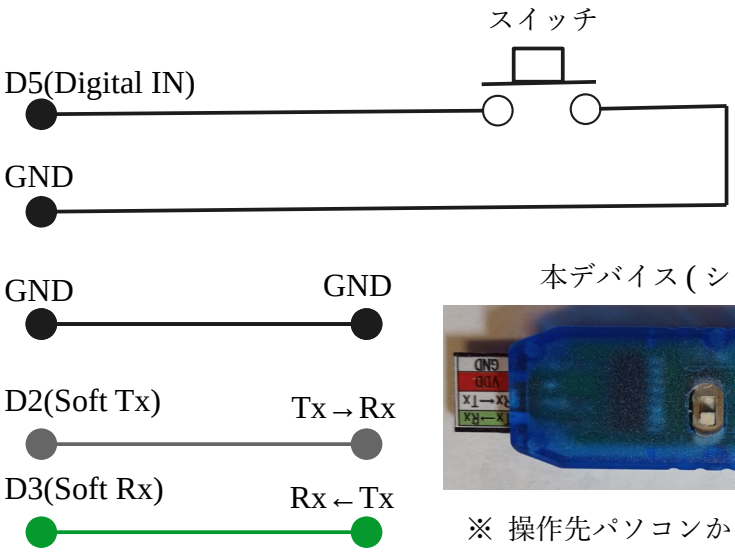
```
#include <SoftwareSerial.h>
const int rxPin=3;
const int txPin=2;
SoftwareSerial mySerial(rxPin, txPin); // Rx,Tx の順で指定
```

本書の実験では写真のように本デバイス(シリアル接続版)と Arduino ボードとスイッチ(押しボタン)を接続します。詳細な接続は以下のようにして下さい。





Arduino ボード



本デバイス (シリアル接続版)

※ 操作先パソコンから電源を供給する場合は Vcc も接続して下さい。

それでは Arduino ボードの D5 ピンに接続されたスイッチをキーに見立てて、キーを押すと Windows ログオン時に最初に押す Ctrl + Alt + Delete をワンキーで入力できるようにしてみましょう。なお、最低限のプログラムしかありませんのでサンプルプログラムを元に皆さんの用途に改造してみてください。

プログラムを Arduino ボードに書き込んで実行させるとスイッチが押されたかどうかの待ち状態になります。そしてスイッチを 1 回押すと Ctrl+Alt+Delete を同時押ししたのと同じ動作をするはずです。こういった便利機能をワンキーで代用したり、ショートカット機能の組み合わせなどをワンキーで実行したりすることも簡単にできます。

なお、プログラムを Arduino ボードに書き込み済みで、操作先パソコン側から電源を供給したい場合は Arduino ボードの USB ケーブルを抜いて本デバイスの Vcc(5V に設定しておく)を Arduino ボードの 5V ピンに接続して下さい

```
#include <SoftwareSerial.h>
const int rxPin=3;
const int txPin=2;
SoftwareSerial mySerial(rxPin, txPin); // RX, TX

void setup() {
  pinMode(5, INPUT_PULLUP); //スイッチを接続(プルアップ)
  mySerial.begin(9600); //SoftwareSerial開始(標準9600bps)
  delay(200);
}

void loop() {
  while(digitalRead(5)==1){} //スイッチが押されるまで待つ
  Press(0x05,0x4C); //CtrlとAltを押しながらDelete(0x4C)を押す
  delay(500);
}

void Press(byte ckey,byte ukey){
  byte sum=0x10C + ckey + ukey;
  byte keyPress[14]={0x57,0xAB,0x00,0x02,0x08,ckey,0x00,ukey,
    0x00,0x00,0x00,0x00,0x00,sum};
  byte keyRelease[14]={0x57,0xAB,0x00,0x02,0x08,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00,0x00,0x0C};

  mySerial.write(keyPress,14);
  while(mySerial.available()>0)
    {mySerial.read();}
  delay(5);
  mySerial.write(keyRelease,14);
  while(mySerial.available()>0)
    {mySerial.read();
}

void SendPacket(char *buf,char blen) {
  mySerial.write(buf,blen); //指定されたバイト数を送信
  while (mySerial.available()>0){ //readバッファをクリアするまで待つ
    mySerial.read();
  }
}
```

4　資料

日本語(109)キーボードのキーコード表 (コードは 16 進数表記)

キー	コード	キー	コード	キー	コード	キー	コード
半角／全角	35	Caps Lock	39	Alt(左)	E2	＋(ﾃﾝｷｰ)	56
！1	1E	A	04	Space	2C	Enter(右)	57
” 2	1F	S	16	Alt(右)	E6	Esc	58
＃3	20	D	07	Ctrl(右)	E4	F1	3A
＄4	21	F	09	Insert	49	F2	3B
％5	22	G	0A	Delete	4C	F3	3C
＆6	23	H	0B	←	50	F4	3D
’ 7	24	J	0D	Home	4A	F5	3E
（8	25	K	0E	End	4D	F6	3F
）9	26	L	0F	↑	52	F7	40
0	27	＋；	33	↓	51	F8	41
＝－	2D	＊：	34	Page Up	4B	F9	42
～＾	2E	｝］	31	Page Down	4E	F10	43
｜￥	89	Enter(左)	28	→	4F	F11	44
Backspace	2A	Shift(左)	E1	Num Lock	53	F12	45
Tab	2B	Z	1D	7(ﾃﾝｷｰ)	5F	Print Screen	46
Q	14	X	18	4(ﾃﾝｷｰ)	5C	Scroll Lock	47
W	1A	C	06	1(ﾃﾝｷｰ)	59	Pause	48
E	08	V	19	／(ﾃﾝｷｰ)	54	無変換	8B
R	15	B	05	8(ﾃﾝｷｰ)	60	変換	8A
T	17	N	11	5(ﾃﾝｷｰ)	5D	ｶﾀｶﾅひらがな	88
Y	1C	M	10	2(ﾃﾝｷｰ)	5A	Windows(左)	E3
U	18	＜,	36	0(ﾃﾝｷｰ)	62	WIndows(右)	E7
I	0C	＞.	37	9(ﾃﾝｷｰ)	55	Application	65
O	12	？／	38	6(ﾃﾝｷｰ)	61		
P	13	＿￥	87	3(ﾃﾝｷｰ)	5E		
‘ @	2F	Shift(右)	E5	.(ﾃﾝｷｰ)	5B		
｛［	30	Ctrl(左)	E0	－(ﾃﾝｷｰ)	63		

5　引用文献

- WCH 社資料(基本的に中国語、参考として筆者が和訳したものがサポートサイトにあります)

  製品紹介

  http://www.wch-ic.com/products/CH9329.html

  データシート

  http://www.wch-ic.com/downloads/CH9329DS1_PDF.html

  通信仕様書および関連ツールなど

  https://www.wch.cn/downloads/CH9329EVT_ZIP.html

- いらすとや(いつも何かとお世話になっています)

  https://www.irasutoya.com/

- おなかすいた Wiki!(こちらも色々とお世話になっています)

  https://wiki.onakasuita.org/pukiwiki/?HID%2F%E3%82%AD%E3%83%BC%E3%82%B3%E3%83%BC%E3%83%89

- MPPUtility(マウスカーソルの座標を表示してくれる使い勝手が良いツールです)

  https://www.vector.co.jp/soft/winnt/util/se487144.html

- 秋月電子通商(ここから CH340 のドライバがダウンロードできます)

  手動で CH340 のドライバをインストールする場合は下記を参照のこと。

  https://akizukidenshi.com/catalog/g/gI-13544/

6　あとがき

たまたま CH9329 という IC を発見して、これは面白そう！と試作品を作ってみたら意外と使いやすかったので驚きました。苦労したのはやはりデータシートやプロトコル解説書が中国語で書かれていたことですかね…けっこう中国語の勉強になりました。

キーボード／マウスというパソコンにとって基本的な入力デバイスを USB に関する知識が無くても気軽に使えるようになっているのは便利だと思います。みなさんのアイディア次第で面白い入力ガジェットを作ってみて下さい！

みんなのラボ PRESENTS

奥付

発行日：2022 年 8 月 13 日 初版発行

発行者：みんなのラボ (http://www.minnanolab.net/)

制作者：ICHI (Twitter @atsuyuki1kawa)

サポートサイト：https://sites.google.com/site/ichiworkspace/

印刷：コンビニコピー