Recipe for Raspberry Pi by RATOC

Triple R

Raspberry Pi I2C 絶縁型 シリアルボード

RPi-GP60

サンプルプログラムについて

RATOC Systems, Inc.

準備

本プログラムを動作させる前に、下記の準備をおこなってください。
すでに準備が終わっていれば、この章は読みとばしてかまいません。

- OS（RASPBIAN）のインストールを実行
- GPIO40pin の I2C 設定
- Raspberry Pi に RPi-GP60（本製品）を接続
- シリアルドライバの設定

上記設定がすべて終われば、次にサンプルプログラムを実行する準備をおこないます。

1）Python サンプルプログラムを実行するディレクトリを作成する。

mkdir コマンドを使って **RPi-GP60** という名前のディレクトリを作成します。(ディレクトリ名や作成場所は任意です)

```
$ mkdir RPi-GP60  ↵
```

ls コマンドを実行して **RPi-GP60** ディレクトリが作成されていること確認します。

```
$ ls  ↵
```

cd コマンドを実行して **RPi-GP60** ディレクトリに移動します。

```
$ cd RPi-GP60  ↵
```

Python サンプルファイルについて

サンプルファイル名称

sampleGp60.py

RPi-GP60 を使用してシリアルの送受信を行う Python サンプルプログラムです。
サンプルプログラムでは下記の処理を行っています。

1.RPi-GP60 の初期設定 init_GP60()

GPIO の初期設定を行います。

※ハードウェアに依存する設定ですので変更しないでください。

- ○ GPIO を GPIO 番号で指定するように設定
- ○ 絶縁回路用電源を ON に設定
  電源 ON 後、安定するまで待ちます。

2.シリアル設定情報変更 input_param(serial)

pyserial モジュールのシリアル設定パラメータを入力値へ設定変更します。
以下のパラメータを指定可能です。変更しない場合は enter のみを入力してください。

- ○ ボーレート [bps]
- ○ バイト長 [bit]
- ○ パリティ [なし , 奇数 , 偶数 , スペース , マーク]
- ○ ストップビット長 [bit]
- ○ 受信タイムアウト時間 [sec]
- ○ フロー制御 XON/XOFF 設定 [False/True]
- ○ フロー制御 RTC/CTS 設定 [False/True]
- ○ フロー制御 DSR/DTR 設定 [False/True]

3.メニュー表示

次のメニューを表示します。

1: 送受信テスト (RS232/RS422 全二重) 2:1: 送受信テスト (RS485 半二重) 3: 設定 0: 終了 >

2.送受信テスト

メニューで 1 または 2 が入力された場合は、シリアル送受信テストを行います。
送信ポート番号と受信ポート番号 (0,1) を入力します。enter のみでメニューに戻ります。

送信ポート番号 (0, 1) enter: 戻る >　　受信ポート番号 (0, 1) enter: 戻る >

メニューで 2 が入力されていれば RS485 の全二重モード動作 (差動ドライバの自動イネーブル制御) を有効にします。(次ページにつづく

前ページのつづき）

送信データ入力 enter: 戻る >

で、送信する文字列を入力します。送信後に受信文字列を表示します。

受信は設定されたタイムアウトが発生するまで継続されます。

タイムアウト後に、送信データ入力に戻ります。

送信データ入力時に enter のみでメニューに戻ります。

5.設定

メニューで 3 が入力された場合は、シリアル設定を変更します。

ポート 0 とポート 1 についてシリアル設定情報変更 input_param() を行います。

6.終了

メニューで 0 が入力された場合は、プログラムを終了します。

Python サンプルファイルの使い方

サンプルファイル名の前に、`python3` をつけて実行します。

※ 実際のシリアル送受信テストの例は次ページに記載

ex）シリアル送受信テスト

ボーレート 115200bps でポート 0 からポート 1 へ折り返し送受信テストをする場合

```
$ python3 sampleGp60.py
RPi-GP60 検査プログラム
1: 送受信テスト (RS232/RS422 全二重 ) 2:1: 送受信テスト (RS485 半二重 ) 3: 設定 4: 出荷検査 0: 終了 > 3
[Port0 現在の設定 ]
Serial<id=0x76a49b90, open=False>(port='/dev/ttySC0', baudrate=9600, bytesize=8, parity='N', stopbits=1,
timeout=0.5, xonxoff=False, rtscts=False, dsrdtr=False)
設定値を入力 enter: 変更なし >
 baudrate(300,1200,2400,4800,9600,14400,19200,38400,57600,115200,230400,460800,921600) = 115200
 bytesize(5, 6, 7, 8) =
 parity('N','E','O','S','M') =
 stopbits(1, 1.5, 2) =
 timeout =
 xonxoff(False, True) =
 rtccts(False, True) =
 dsrdtr(False, True) =
[Port1 現在の設定 ]
Serial<id=0x76a49bb0, open=False>(port='/dev/ttySC1', baudrate=9600, bytesize=8, parity='N', stopbits=1,
timeout=0.5, xonxoff=False, rtscts=False, dsrdtr=False)
設定値を入力 enter: 変更なし >
 baudrate(300,1200,2400,4800,9600,14400,19200,38400,57600,115200,230400,460800,921600) = 115200
 bytesize(5, 6, 7, 8) =
 parity('N','E','O','S','M') =
 stopbits(1, 1.5, 2) =
 timeout =
 xonxoff(False, True) =
 rtccts(False, True) =
 dsrdtr(False, True) =
1: 送受信テスト (RS232/RS422 全二重 ) 2:1: 送受信テスト (RS485 半二重 ) 3: 設定 4: 出荷検査 0: 終了 > 1
送信ポート番号 (0, 1) enter: 戻る > 0
受信ポート番号 (0, 1) enter: 戻る > 1
送信データ入力 enter: 戻る > ABCD
受信データ：ABCD

送信データ入力 enter: 戻る > EFGH
受信データ：EFGH

送信データ入力 enter: 戻る > UUUU
受信データ：UUUU

送信データ入力 enter: 戻る >
1: 送受信テスト (RS232/RS422 全二重 ) 2:1: 送受信テスト (RS485 半二重 ) 3: 設定 4: 出荷検査 0: 終了 > 0
```