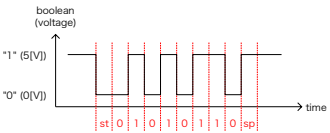
PSoCでUARTを使う

UARTは、マイコンでシリアル通信を実現するために用いるモジュールです。シリアル通信とは、1本の信号線でデータ通信をおこなう伝送方式のことです。シリアル通信を用いれば、マイコンとマイコン間や、マイコンとPC間でのデータのやり取りができるようになります。また、マイコン内部の変数の値などを外部に出力して、モニタすることができるようになるので、デバッグにも有用です。

1[Byte]=8[bit]のデータを転送する際の波形は、下図のようになります。



これは10進数の"106"を送信したときの波形です。UARTの波形には、いくつかの決まりがあります。

(1)何も信号が無いときは"1"です。

(2)スタートビットは、データ通信開始の信号です。信号を"0"にします。図では"st"のところです。

(3)ストップビットは、データ通信終了の信号です。信号を"1"にします。図では"sp"のところです。

(4)データは1[Byte]単位で通信します。スタートビットとストップビットの間に、データを2進数で表したときの8[bit]を挟みます。ただし、LSBが最初で、MSBが最後です。図では、10進数の"106"を送信しています。10進数の"106"は2進数で表すと"01101010"になります。LSBとMSBが逆になるので、信号の波形は"01010110"になります。

(5)スタートビット、ストップビット、およびデータのビットは同じ幅です。

UARTを実装するには、送信を担当するTXモジュールと、受信を担当するRXモジュールを用います。実は送受信を両方できるUARTモジュールというものも用意されているのですが、使い方はほぼ一緒です。今回はRXモジュールとTXモジュールを別々に解説します。

UARTを使う上で注意すべきは、送信側の通信速度の設定と、受信側の通信速度の設定を合わせることです。通信速度の設定が送信側と受信側で異なると、通信不良や文字化けなどの原因になります。

通信速度には2種類あります。ビットレートとは1秒間に送信できるビット数です。単位は[bps]=[Hz]です。また、ボーレートとは1秒間に送信できるバイト数です。（つまり、ビットレートの8倍がボーレートに等しくなります。）以下、通信速度はビットレートのことを指すこととします。

そして、TX・RXモジュールには、通信速度の8倍のクロックを与えます。（つまりはボーレートの周波数です。）

では、通信速度1[kbps]のときのプロジェクトを作成していきます。

PSoCでTXを使う

TXはシリアル信号の送信をおこなうモジュールです。

まず新規プロジェクトを作成します。

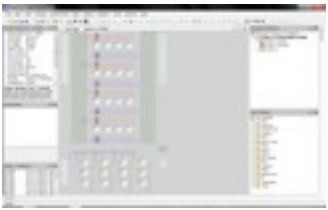


図.1

右下からTX8を探し、配置します。

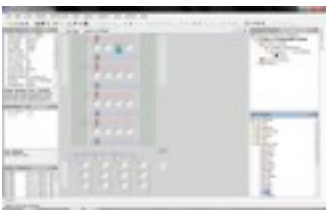


図.2

クロックを作成します。通信速度が1[kbps]なので、その8倍の8[kHz]が必要です。システムクロックを分周して、VC3を8[kHz]としました。

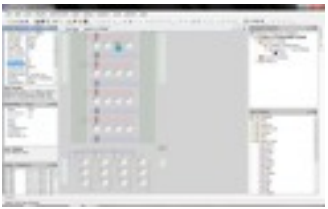


図.3

TX8モジュールのプロパティを設定します。"Clock"は8[kHz]のVC3です。"Output"の項目は後で設定することにして、残りの項目を埋めます。（あまり本質的な設定ではないので説明は省略します。図.4のスクリーンショットを真似てください。）

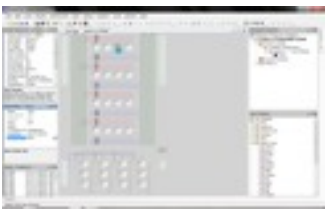


図.4

ピンの設定をします。ピンの設定画面を開き、UARTの信号を出力したいピンの"Select"を"GlobalOut..."、"Drive"を"Strong"にします。

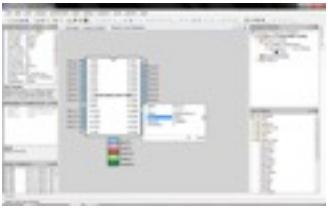


図.5

内部配線をします。TX8からピンへ信号が伝わるように配線を延ばします。

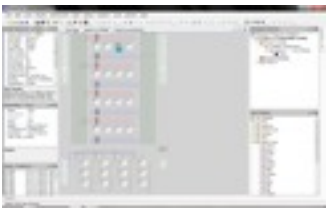


図.6

最後にプログラムを記述します。

図.7

```
TX8_1_Start(TX8_1_PARITY_NONE);
```

TX8モジュールを起動します。

```
TX8_1_SendData(0b01101010);
```

データの送信をおこないます。

```
while(!(TX8_1_bReadTxStatus() &
TX8_1_BUFFER_EMPTY));
```

データの送信が完了するまで待つ、という記述になります。

これでGenerateし、マイコンに書き込みます。うまくいけば、所定のピンからシリアル通信の信号が出ます。オシロスコープなどで確認すると良いです。

図.8

通信速度が1[kbps]なので、1[bit]あたり1[ms]の周期になります。

PSoCでRXを使う

RXはシリアル信号の受信をおこなうモジュールです。TXとやることは似ています。

まず新規プロジェクトを作成します。

図.9

右下からRX8を探し、配置します。

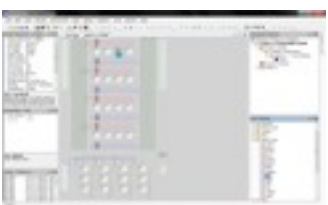


図.10

クロックを作成します。TXに合わせ、VC3を8[kHz]にします。

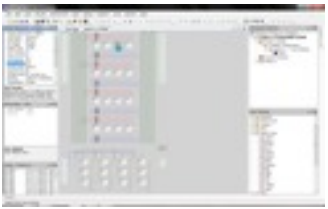


図.11

RX8モジュールのプロパティを設定します。"Clock"は8[kHz]のVC3です。"Input"の項目は後で設定することにして、残りの項目を埋めます。（あまり本質的な設定ではないので説明は省略します。図.12のスクリーンショットを真似てください。）

図.12

ピンの設定をします。ピンの設定画面を開き、UARTの信号を出力したいピンの"Select"を"GlobalIn..."、"Drive"を"HighZ"にします。

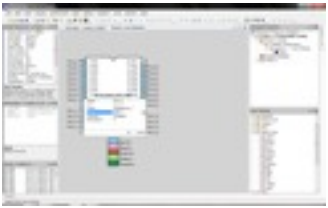


図.13

内部配線をします。ピンからRX8へ信号が伝わるように配線を延ばします。

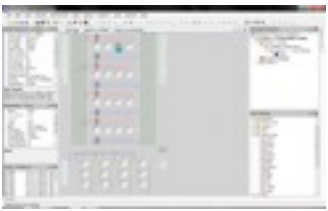


図.14

デバッグ用にLEDモジュールを置いておきます。うまく受信できたら点灯、受信できなかったら消灯させます。

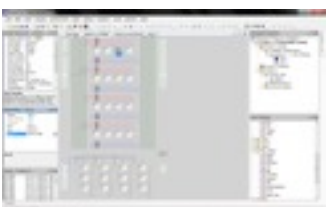


図.15

最後にプログラムを記述します。

図.16

```
RX8_1_Start(RX8_1_PARITY_NONE);
```

RX8モジュールを起動します。

```
while(!(RX8_1_bReadRxStatus() &
RX8_1_COMPLETE));
```

データの受信が完了するまで待つ、という記述になります。

```
ReveivedData = RX8_1_bReadRxData();
```

受信したデータを変数に代入します。

これでGenerateし、マイコンに書き込みます。

送信用のマイコンと、受信用のマイコンを用意して、それぞれの送信ピンと受信ピンをつなぎ、両方に電源を入れます。うまく通信ができていれば、LEDが点灯するはずです。

おわり