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。





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。





最後にプログラムを記述します。

Project_nx - PSoC Designer 5.1		
Ele Edit View Broject Interconnect Build Debug Program Tools Window Help		
1월달달 [오태인]백천[씨관]2월 [바리아아, Indon > H티아이라, IV 에운송[고임]고무무명 [삼신원인 [4	038	
Start Page (project or [Cho]* (Project rouge Project) (main_e*)		+ 4 5 3
1 //		
2 // C main line		
3 //		
24 4		
5 #include <m8c.h> // part specific constants and macros</m8c.h>		
6 #include "PSoCAPI.h" // PSoC API definitions for all User Modules		
7		
8		
9 void main(void)		
10 (
11 // M8C_EnableGInt ; // Uncomment this line to enable Global Interrupts		
12 // Insert your main routine code nere.		
13 BITE ReceivedData)		
10 BVG 1 PERMETRY 1 DIBTRY NOND).		
16 wble(1)/		
17 while(!(RX8 1 bReadByStatus() & RX8 1 BX COMPLETE));		
18 ReceivedData = RX8 1 bReadRxData();		
19 if (ReceivedData == 0b01101010) (
20 LED_1_On();		
21 - }		
22b else(
23 LED_1_Off();		
24 - }		
25 - }		
26 2		
27		
		,
Output		
laded LED User Module.	Ln 27	COI 1

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