LOGIPIC

Logiciel de programmation graphique des microcontrôleurs PIC

Transmission de donnée par liaison série Révision1

La plupart des PIC sont dotés de la fonction UART *(Universal Asynchronous Receiver Transmitter)*, qui permet de transférer des trames de données par liaison série *(bit par bit),* pour communiquer avec un PC ou un autre microcontrôleur.

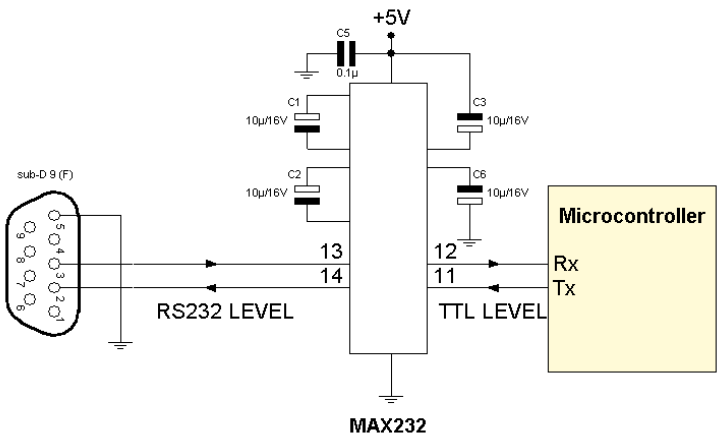
Il devient alors possible de piloter un système via une interface PC en envoyant des ordres (commande de rotation d'un moteur, affichage d'information sur un écran LCD…) et en recevant des informations (niveau de tension, température, position d'un codeur…).

Dans cet exercice, nous allons créer un circuit électronique composé de la PIC et d'un module de communication qui sera connecté à l'ordinateur. Un octet sera envoyé du PC depuis une application, celui-ci sera lu par la PIC, fera clignoter une led plus ou moins vite en fonction de sa valeur, et la pic renverra l'octet vers le PC. On aura donc une réception puis une émission par liaison série.

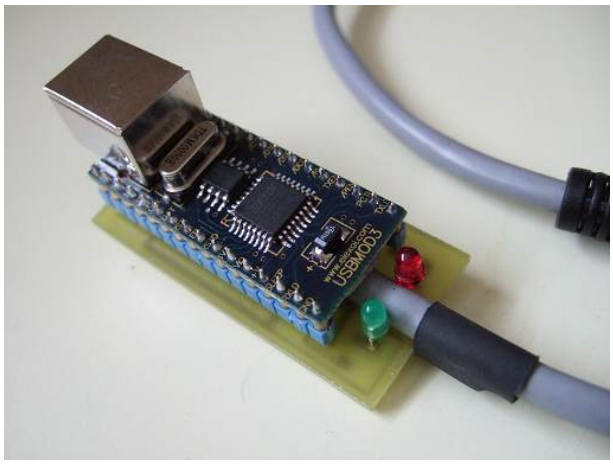
Liaison sériel

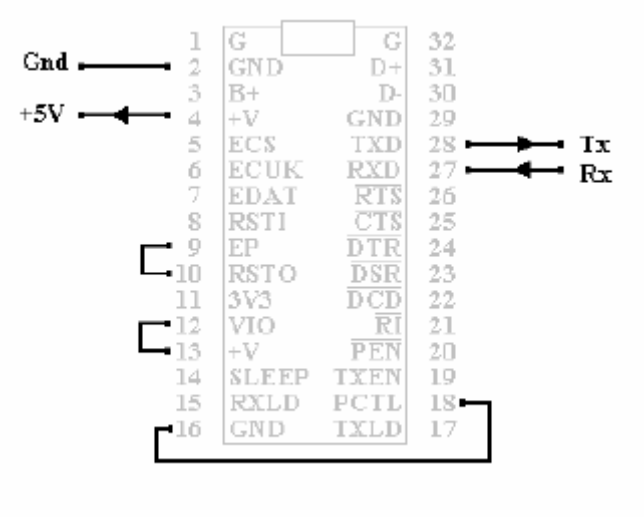
Pour communiquer avec l'ordinateur, nous avons deux possibilités :

- Soit l'utilisation du protocole RS232 via un driver du type max232 qui permet de convertir les données TTL de la PIC au format CMOS du PC. **Il est nécessaire que le PC soit doté d'une sortie série sub-DB9**.



- Soit la liaison USB en utilisant par exemple le Module USBMOD3 de chez RAVAR. Le module est détecté par le PC, mais un driver est nécessaire, il est disponible sur le site du fabriquant http://www.ftdichip.com/Drivers/VCP.htm . Le module est auto-alimenté par le bus USB et délivre les données RX et TX directement exploitable par la PIC.

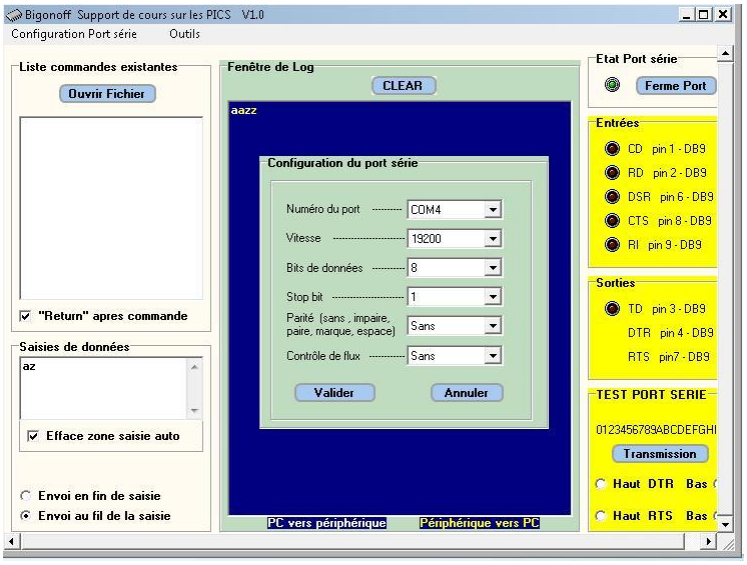




Le module USBMOD3 est composé de 2x16 pattes au format DIL32. Pour fonctionner correctement, il est nécessaire de réaliser quelques ponts. Vu du PC, le module est traité comme un port COM série. Il est possible d'ajouter deux LED pour visualiser le transfert via les pin 15 et 17 avec résistances de 330 Ohms reliées à +V.

Interface PC

Pour échanger des informations entre le PC et la PIC, on a besoin d'une application capable de géré la réception et l'émission de données du port choisi.
M. Bigonoff nous met à disposition l'application Bscp. Cet outil réalise de transfert d'information en continu, il est nécessaire de configurer le port COM et la vitesse19200 bauds (voir figure)



Fonctionnement des registres UART

Le standard RS-232 permet une communication sérielle, asynchrone et duplex entre deux équipements. Chaque trame est composé d'un bit de départ, de 7 à 8 bit de données, d'un bit de parité optionnel et d'un ou plusieurs bit d'arrêts. Le bit de départ à un niveau logique "0" tandis que le bit d'arrêt est de niveau logique "1". Le bit de donnée de poids faible est envoyé en premier suivi des autres.(wikipedia)

REGSITE TXSTA (pour l'envoi d'un octet)

b7 : non : non utilisé en mode asynchrone
b6 : TX9 : données codées sur 8 ou sur 9 bits. (1 = 9 bits, 0 = 8 bits)
b5 : TXEN : lance l'émission.
b4 : SYNC : travaille en mode synchrone (1) ou asynchrone (0).
b3 : non utilisé
b2 : BRGH : prédiviseur internes, mode grande vitesse (BRGH = 1) et un mode basse vitesse (BRGH = 0).
b1 : TRMT : indique quand TSR = 0, émission terminé.
b0 : TX9D : valeur du 9ème bit à envoyer.

REGSITE RCSTA (pour la réception d'un octet)

b7 : SPEN : mise en service de l'USART.
b6 : RX9 : réception sur 8 ou sur 9 bits (1 = 9 bits, 0 = 8 bits)
b5 : non utilisé
b4 : CREN : lance la réception continue.
b3 : ADDEN : permet de filtrer la réception.
b2 : FERR : indique une erreur de trame.
b1 : OERR : indique une erreur de type overflow.
b0 : RX9D : contient le 9ème bit de votre donnée reçue.

REGSITE SPBRG (calcul de la vitesse)

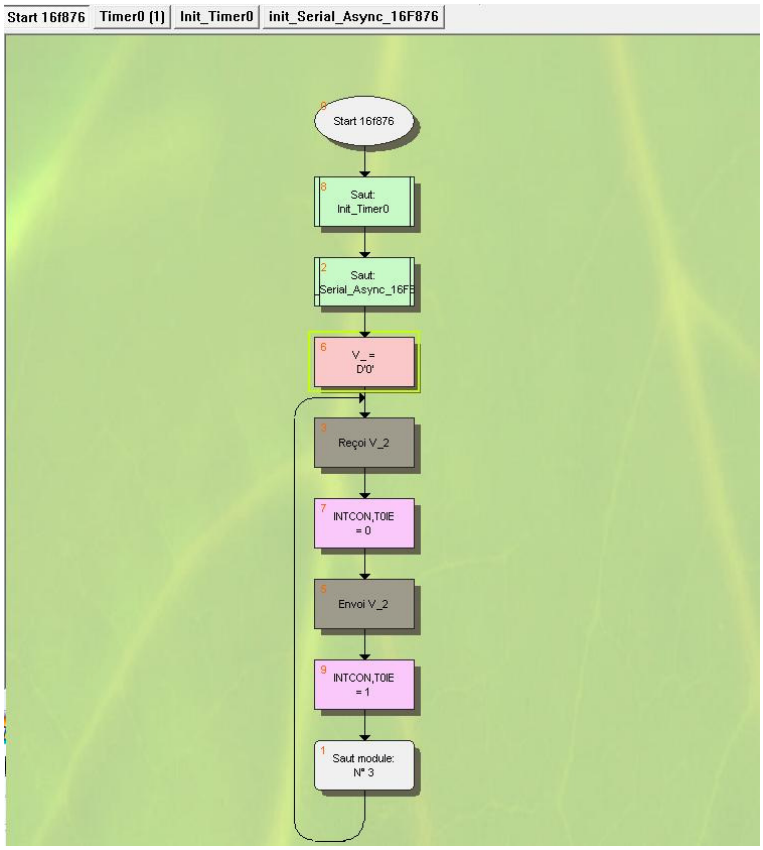
Si BRGH = 0 (basse vitesse) : SPBRG = (Fosc / (Débit * 64)) – 1
Si BRGH = 1 (haute vitesse) : SPBRG = (Fosc / (Débit * 16)) – 1

Fosc étant la fréquence d'oscillation du quartz et Débit la vitesse de transmission (ex : Fosc = 4MHz et 19200 Bauds)

REGISTRE TXREG : Le registre TXREG contient la valeur sous 8bits à envoyer

REGISTRE RCREG : Le registre RCREG contient la valeur sous 8bits reçue

Programmation de la PIC avec LogiPic



Le programme démarre avec l'appelle de deux routines Init_Timer et Init_Serial_Async. Ces routines initialisent le timer0 ainsi que la gestion de la communication série.

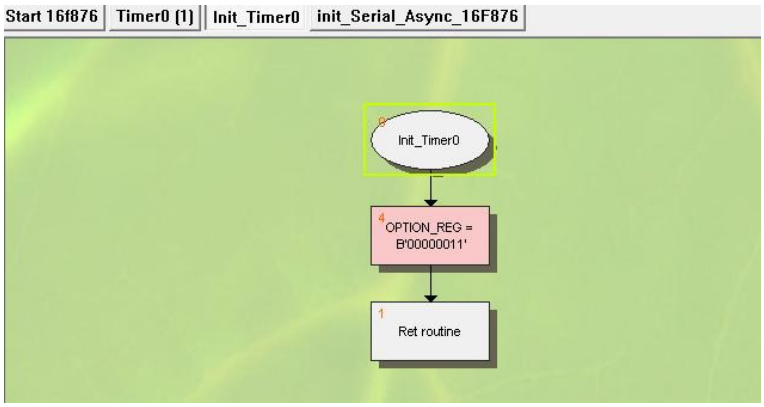
La variable V_ est positionnée à zéro, elle servira de compteur lors des interruptions timer0.

Le module suivant sert à réceptionner un octet sur le port série, celui-ci sera stocké dans V_2.
Le module de réception à la particularité de bloquer le programme jusqu'à réception d'un octet.

Ensuite on met hors service l'interruption du timer0 pour ne pas perturber l'émission de l'octet précédemment reçu.

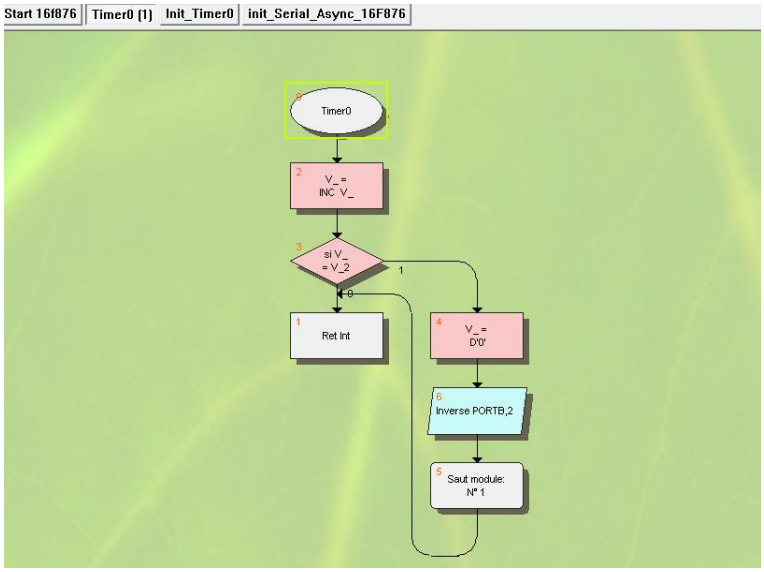
Le module de communication envoi la variable V_2

Et on remet en service l'interruption timer0, et on reboucle de programme.



La routine Init_Timer0 configure le registre OPTION_REG pour avoir un délai de 4,096 ms par interruption. (voir coach sur les timers)

PSA=0
PS2=0
PS1=1
PS0=1

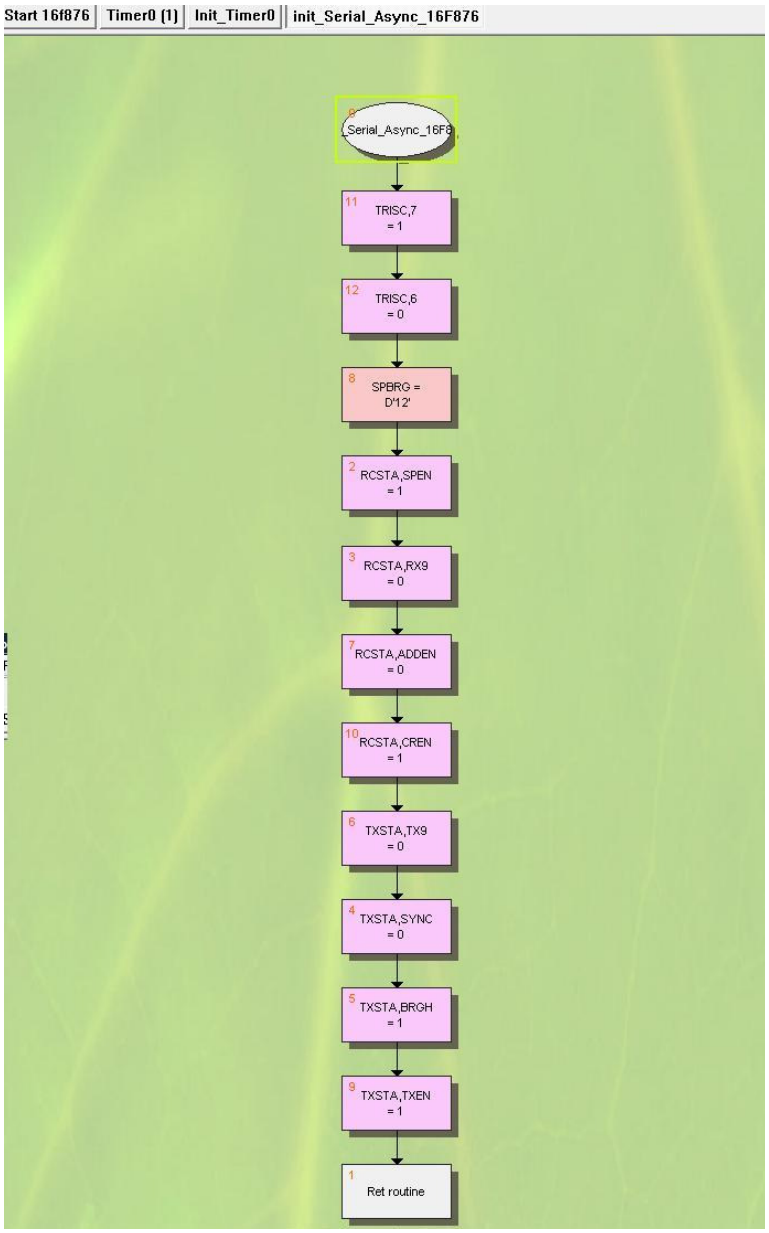


L'interruption Timer0 est donc appelée toute les 4,096 ms.

On souhaite faire varier la fréquence de clignotement de la led en fonction de la valeur de V_2, pour cela on incrémente la variable V_ et on réalise un test pour voir si elle est égale à V_2.

Si c'est le cas on replace V_ à zéro et on change l'état le la led reliée à la sortie 2 du port B en série avec une résistance de 200 ohms.

Sinon on réalise un retour interruption.



Pour pouvoir communiquer avec l'extérieur il nous faut faire quelques réglages :

On place la pin 7 du port C en sortie pour émettre et la pin 6 du port C en entrée pour recevoir.

Le registre SPBRG est initialisé à 12 pour avoir une vitesse de 19200 bauds avec un quartz à 4MHz

Pour BRGH = 1 (haute vitesse)
SPBRG = (Fosc / (Débit * 16)) – 1
SPBRG = (4000000 / (19200 * 16)) – 1
SPBRG = 12,02

RCSTA,SPEN=1 : mise en service de l'USART.
RCSTA,RX9=0 : réception sur 8 bits
RCSTA,ADDEN=0 : filtre inactif
RCSTA,CREN=1 : réception continue

TXSTA,TX9=0 : émission sur 8 bits
TXSTA,SYNC=0 mode asynchrone
TXSTA,BRGH=1 : haute vitesse
TXSTA,TXEN=1 : autorise l'émission

Et enfin le retour routine.

Vous pouvez à présent compiler le programme avec MPASM et le transférer dans la pic via un programmateur, ou par le bootloader tinyboot si le mini programme de boot est présent dans la pic.

Mise en pratique

A la mise sous tension, la led clignote,

lancez BSCP ou tinyboot en mode terminal, avec le port COM correspondant et la vitesse de 19200 bauds.

En envoyant un caractère ASCII comme ‘z’ qui correspond à la valeur (122) fera clignoter la led moins vite que si vous envoyez le caractère ‘0’ (48).

Bonne programmation