Travaux Pratiques d'Interfaçage Logiciel
deuxième séance

ENSIBS 2^{eme} année, Spécialités Mécatronique et Génie Industriel

1 Objectifs

Au cours de ce deuxième TP, vous allez continuer la manipulation d'une carte à base de microcontrôleur pour le pilotage de périphériques.

L'objectif est maintenant de maîtriser la communication série. Vous devrez donc, à l'issue de la séance savoir :

- configurer la liaison série
- envoyer un caractère
- envoyer une chaîne de caractères
- recevoir des caractères et des chaînes de caractères
- gérer la liaison série avec des interruptions

Couplée aux LEDs, la liaison série est un élément incontournable dans la mise au point de logiciels embarqués.

2 Description du matériel

Pour cette deuxième séance, nous allons utiliser le port série identifié *PORT A* sur la carte (celui qui se trouve à côté de l'alimentation. Au niveau du microcontrôleur, il s'agit du périphérique *UART2*.

2.1 Le périphérique UART2

L'UART est décrite à la section 19 du *manuel de référence* de la série *dsPIC30F* disponible sur le site de Microchip.

Elle se configure à l'aide de trois registres :

- *UxMODE* : qui permet de contrôler le mode de fonctionnement
- *UxSTA* : registre de contôle et de status, qui fournit quelques bits de configuration supplémentaires par rapport à *UxMODE* mais aussi des registres informant de l'état courant de la communication.
- *BRG* : le *Baud Rate Generator*, permet de configurer la fréquence de transmission. Attention, cette fréquence dépend de la fréquence du composant.

Une fois configurée, on peut utiliser l'UART pour envoyer et recevoir des données à l'aide des registres *UxRXREG* et *UxTXREG*.

L'UART peut aussi être utilisée en mode interruption.

3 Réalisations

3.1 Configuration de l'UART et envoi d'un caractère

La première étape du TP consiste à déterminer la configuration à appliquer au niveau du périphérique afin d'envoyer un caractère vers le PC.

La configuration du port série que vous allez tester est la suivante :

- 9600 bits/sec
- 8 bits de données
- 1 bit de stop
- pas de parité
- aucun contrôle de flux

Afin d'être certain de la fréquence de fonctionnement du dsPIC, on peut spécifier explicitement dans le programme la configuration du composant tel que présenté au listing 1. Ici on configure le dsPIC pour fonctionner sur une horloge externe avec un coefficient multiplicateur de 4. Comme le dsPIC nécessite quatre *périodes d'horloge* pour réaliser un *cycle instruction*, la fréquence d'un cycle sera égale à la fréquence de l'horloge externe soit 7,37MHz.

Il faut aussi s'assurer que le *watchdog* est bien désactivé.

Listing 1 – Configuration de l'horloge dsPIC

```
1 #include <p30f6014.h>
2
3 _FOSC(CSW_FSCM_ON & XT_PLL4); /* horloge externe x4 */
4 _FWDT(WDT_OFF);               /* desactivation du watchdog */
```

Le programme de test qu'il vous est demandé d'écrire doit envoyer deux caractères différents de manière répétée sur le port série (en direction du PC).

Pour pouvoir vérifier que le caractère est bien reçu par le PC, il vous faudra utiliser un logiciel de communication par le port série. Sous MS-Windows, le logiciel par défaut est l'*Hyperterminal* que vous trouverez dans *Programmes -> Accessoires -> Communications*. Il vous suffit de créer une nouvelle connexion utilisant le port *COM1* avec les mêmes paramètres que vous aurez choisi pour la configuration de la liaison série sur le microcontrôleur.

3.2 Ecriture d'une routine d'envoi d'un caractère

Lorsque l'on souhaite afficher une chaîne de caractères sur le terminal, on ne fait *jamais* directement appel au périphérique. On utilise une fonction chargée de réaliser le travail.

Avant d'envoyer une chaîne complète, il faut d'abord savoir envoyer un caractère seul. C'est ce que nous proposons de faire à l'aide de la fonction `envoyer_caractère`, qui prends pour argument un caractère et ne renvoie rien (généralement on renvoie un code d'erreur permettant au programme appelant de savoir si l'envoi s'est bien déroulé). Il vous est demandé de compléter cette fonction proposée au listing 2. Cette fonction sera appelée avant le `main`.

Listing 2 – Squelette de la fonction envoyer_caractère

```
1 void envoyer_caractere (char c) {
2
3  /* Placez votre code ici */
4
5 }
```

Modifiez ensuite votre programme pour utiliser la fonction.

3.3 Routine d'envoi d'une chaîne de caractères

Une chaîne de caractère en C est un tableau terminant par le caractère `'\0'`.

Proposez une fonction C permettant d'envoyer une chaîne de caractères et utilisant la routine d'envoi d'un caractère précédemment écrite.

3.4 Utilisation d'une interruption pour l'envoi d'une chaîne de caractères

Lors de l'envoi d'une chaîne de caractères, le processeur passe une grande partie de son temps à attendre que le tampon contenant les caractères soit vide. On peut illustrer ce constat par le bout de programme suivant :

Listing 3 – Clignotage de LED et envoi de données sur le port série

```
1  while (1) {
2    ecrire_chaine("Toute␣la␣chaine␣doit␣avoir␣ete␣"
3        "transmise␣avant␣que␣le␣microcontroleur␣"
4        "passe␣a␣l'instruction␣suivante␣ce␣qui␣peut␣"
5        "parfois␣prendre␣du␣temps.\r\n");
6
7    for (i = 0; i < 10; i++) {
8      for (j = 0; j < 0xFF; j++)
9        for (k = 0; k < 0x2FF; k++); /* Une attente */
10
11     LEDS ^= LED1;
12   }
13 }
```

L'utilisation d'une interruption permet d'éviter cette attente. Le principe est de stocker la chaîne de caractère en mémoire et de configurer l'interruption série pour s'activer lorsque le tampon d'émission d'un caractère est vide. L'interruption sera alors chargée de placer le caractère suivant dans le tampon.

Ecrivez une nouvelle version de `ecrire_chaine` qui fonctionne en utilisant une interruption.

3.5 Réception d'un caractère

Afin de vérifier que vous êtes capables de recevoir un caractère sur votre carte, écrivez une routine renvoyant chaque caractère reçu par le port série.