



## ETUDE GESTION DE LA CIRCULATION A UN CROISEMENT PAR FEUX TRICOLORES

### TP3

#### Objectif :

- Utilisation de sous programmes et de plusieurs fichiers C dans un projet
- Initiation à la programmation en C

#### Rôle du pré-processeur

Le pré-processeur ou pré-compilateur réalise des mises en forme et des aménagements du texte d'un fichier source, juste avant qu'il ne soit traité par le compilateur. Il existe un ensemble d'instructions spécifiques appelées **directives** pour indiquer les opérations à effectuer durant cette étape.

Les deux directives les plus courantes sont `#define` et `#include`.

`#define` correspond à une équivalence ex : `#define pi 3.14` ou une définition de macro

#### Rôle des fichiers d'inclusion

Les fichiers d'inclusion ou d'en tête **\*.h (header)** contiennent pour l'essentiel cinq types d'informations :

- Des définitions de nouveau type
- Des définitions de structure
- Des définitions de constantes
- Des déclarations de fonctions
- Des définitions de macro\_fonctions

En général ces fichiers contiennent des directives de compilation ou pré\_compilation conditionnelles. De ce fait ils ne sont pas toujours aisés à déchiffrer pour une personne qui débute en langage C. néanmoins il est indispensable d'en prendre petit à petit connaissance.

Il s'agit d'un fichier d'inclusion particulièrement important lorsqu'on travaille en C sur un micro-contrôleur : le fichier de définition des registres internes du micro-contrôleur `p18F4520.h` par exemple .

`p18f4520.h` possède les définitions des registres et des bits ce qui permet d'accéder directement aux registres du  $\mu$ contrôleur par leur nom (**ceux du data sheet**) et également de tester ou positionner individuellement les bits de ces registres de la façon suivante : **nom\_registre.nom\_bit**

Pour inclure un fichier contenant du code source (.c ou .h) dans un autre fichier il faut utiliser la directive

`#include` de la façon suivante :

`#include <Nomfichier>`

recherche du fichier dans :

- Les répertoires mentionnés à l'aide de l'option de compilation `/I`directory
- Les répertoires définis à l'aide de la variable d'environnement `INCLUDE`

`#include "Nomfichier"`

recherche du fichier dans :

Idem cas précédent +

Le répertoire courant

Il est également possible de préciser le chemin complet du fichier : `#include "c:\exo\monfichier.c"`

Un fichier source en C pour PIC18F4520 contiendra toujours la déclaration :

`#include <p18f4520.h>`

Ouvrir avec le Bloc-note le fichier `p18f4520.h` et visualiser ce fichier.

□ Le port A est un octet (unsigned char) défini dans un fichier externe (extern) dont la valeur peut être écrasée entre 2 appels (volatile).

□ La deuxième déclaration précise que `PORTAbits` est une union de structures **anonymes** de bits adressables. Du fait que chaque bit d'un registre de fonction peut avoir plusieurs affectations, il y peut y avoir plusieurs définitions de structures à l'intérieur de l'union pour un même registre. Dans le cas présent les bits du port A sont définis comme :

**1ère structure :**

port d'E/S parallèle (7 bits ; RA0 à RA6)

**2ème structure :**

port d'entrées analogiques (5 entrées AN0 à AN4) + entrée OSC2.

**3ème structure :**

Des entrées de tension de référence du CAN, entrée horloge externe du timer0 (T0CKI), entrée de sélection du port série synchrone (SS), sortie du timer0 (CLK0).

**4ème structure :**

entrée low voltage detect (LVDIN)

Le contenu du registre `ADCON1` déterminera l'affectation d'un bit (cf DS39564B page 182).

L'accès à un bit du portA se fait de la façon suivante :

**Nom\_union.nom\_bit**

**Exemple :**

`PORTAbits.RA0 = 1 ; // mise à l'état haut de RA0`



Q1 - Ouvrir le projet MPLAB : "Detection Appuis S3" qui se trouve dans l'Explorateur Windows.

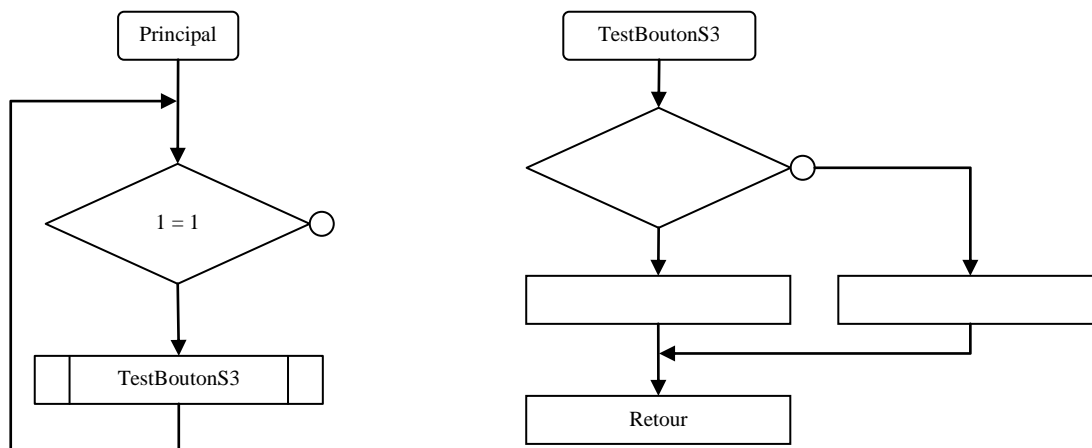
**Vérifier les "Build Options" :** Voir TP0 et TP1

Q2 - Compiler, télécharger et lancer le programme dans la cible PICDEM2 PLUS.

Q3 - Appuyer sur S3, que constatez-vous?

Q4 - Arrêter le programme.

Q5 - Compléter l'algorithme TestBoutonS3 :



**Penser à utiliser dans le fichier "Main.h" les définitions données aux bits pour modifier le programme**

Q6 - Modifier le programme pour allumer la led D4 sur RB2 lors de l'appui sur S3.

Q7 - Modifier le programme pour allumer les leds D3 sur RB1 et D4 sur RB2 lors de l'appui sur S3.

### **Remarques :**

- ☞ L'utilisation de plusieurs fichiers C est très utile lorsque les projets deviennent importants.
- ☞ De plus ils pourront être utilisés dans d'autres projets ce qui évitera de réécrire du code inutilement.
- ☞ Il faut aussi prendre soins de nommer les fichiers et les noms de variables le plus clairement possible. N'hésiter pas à mettre des noms un peu "long" comme dans l'exemple précédent : "TestBoutonS3".

