

## TPN°1:Prise en main l'environnement de programmation des Microcontrôleurs(3h)

### I: Objectifs:

L'objectif de ce TP est de se familiariser avec l'environnement de développement MPLAB et apprendre à créer un projet MPLAB, saisir un programme en assembleur et le compiler.

### II : Introduction

#### I. Introduction

Les PICs sont des microcontrôleurs à architecture RISC (Reduced Instructions Set Computer), ou encore composant à jeu d'instructions réduit. L'avantage est que plus on réduit le nombre d'instructions plus leur décodage sera rapide ce qui augmente la vitesse de fonctionnement du microcontrôleur.

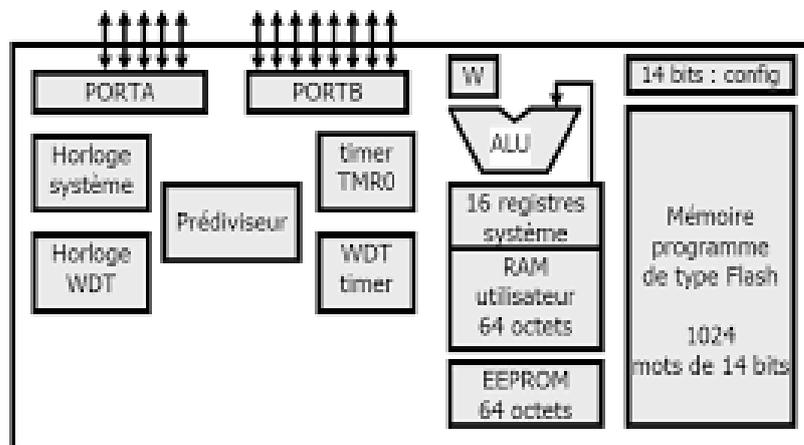
Les PICs sont subdivisés en 3 grandes familles:

- La famille Base-Line, qui utilise des mots d'instructions de 12 bits
- La famille Mid-Range, qui utilise des mots de 14 bits (et dont font partie les 16F8xx
- La famille High-End, qui utilise des mots de 16 bits (les PIC 18Fxxx.

#### II. PRESENTATION GENERALE DU PIC 16F84A

Le PIC 16F84 est un microcontrôleur 8 bits. Il dispose donc d'un bus de données de huit bits.

##### a) La structure générale du PIC 16F84A comporte 4 blocs



-**Mémoire de programme:** elle est de type FLASH d'une taille de 1024\*14 bits

# TP Système à microcontrôleurs

M1 Instrumentation BIOMEDICALE

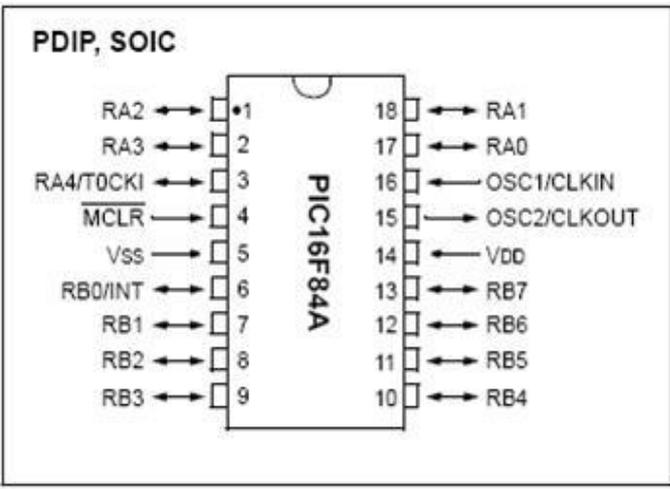
-Mémoire de données: RAM de 68 octets, EEPROM de 64 octets

-Processeur-

-Ressources auxiliaires (périphériques): ports d'entrées et de sorties.-temporisateur.-interruptions- chien de garde-mode sommeil.

b) Le PIC16F84 est un circuit intégré de 18 broches:

- VDD et VSS: Borne d'alimentation +5V et 0V. C'est ici que seront branchés le régulateur de tension et la masse.
- MCLR : "Reset" du circuit.
- OSC1 et OSC2 : Brôches recevant le montage horloge.
- RA0 à RA4 : Port d'E/S A.
- RB0 à RB7 : Port d'E/S B.



c) Les registres:

La mémoire de données RAM contient des registres spécifiques qui permettent de configurer le PIC. cette mémoire est séparée en deux pages ( page 0 et page 1 ) comme le montre la figure suivante

⊙	Banque 0	Banque 1	⊙
00	INDF	INDF	80
01	TMR0	OPTION	81
02	PCL	PCL	82
03	STATUS	STATUS	83
04	FSR	FSR	84
05	PORTA	TRISA	85
06	PORTB	TRISB	86
07			87
08	EEDATA	EECON1	88
09	EEADR	EECON2	89
0A	PCLATH	PCLATH	8A
0B	INTCON	INTCON	8B
0C			8C
	Mémoire utilisateur	Idem	
4F			CF

### 1. STATUS: Registre d'état

		RPO	TO	PD	Z	DC	C
--	--	-----	----	----	---	----	---

## TP Système à microcontrôleurs

M1 Instrumentation BIOMEDICALE

. **RP0** permet de sélectionner la page dans la mémoire RAM

Si  $RP0=0$  on accède à la page 0

Si  $RP0=1$  on accède à la page 1 .

Au reset, seul le bit **RP0** de sélection de page est fixé (  $RP0=0$  : page 0 )

. **TO/** (Time Out) : débordement du timer WDT

. **PD/** : (Power Down) caractérise l'activité du chien de garde WDT

. **C** : Ce bit flag passe à 1 lorsqu'il y a une retenue sur un octet

. **DC** : Ce bit flag passe à 1 lorsqu'il y a une retenue sur un quartet

. **Z** : Ce bit flag passe à 1 quand le résultat d'une opération est nul.

**Registres PORTA et PORTB** contiennent respectivement l'état des lignes du port A et du port B

**Registre TRISA** . Direction des données pour le port A : 0 pour sortir et 1 pour entrer

TRISA	X	X	X	1	0	1	1	0
PORTA	X	X	X	RA4 (entrée)	RA3 (sortie)	RA2 (entrée)	RA1 (entrée)	RA0 (sortie)

**Registre TRISB**. Direction des données pour le port B : 0 pour sortir et 1 pour entrer

TRISB	1	1	1	1	0	0	0	1
PORTB	RB7 (entrée)	RB6 (entrée)	RB5 (entrée)	RB4 (entrée)	RB3 (sortie)	RB2 (sortie)	RB1 (sortie)	RB0 (entrée)

### III: Présentation MPLAB

Microchip propose pour le développement de ses microcontrôleurs, un « Environnement de développement Intégré » ou IDE, sous la forme d'un logiciel du nom de MPLAB.

MPLAB est un environnement de développement qui contient tous les outils nécessaires à la mise au point d'une application avec un cœur de microcontrôleur PIC.

Il contient un:

- Editeur de texte interactif,
- Compilateur (assembleur et C)
- Simulateur.

## TP Système à microcontrôleurs

M1 Instrumentation BIOMEDICALE

### a) Création d'un nouveau projet avec « Project Wizard »

Lancer logicielle MPLAB IDE à partir de l'icône qui se trouve sur votre bureau.

Cliquer sur Project >> Project Wizard ...

Cliquer sur Suivant, puis sélectionner PIC16F84A dans le menu déroulant Device.

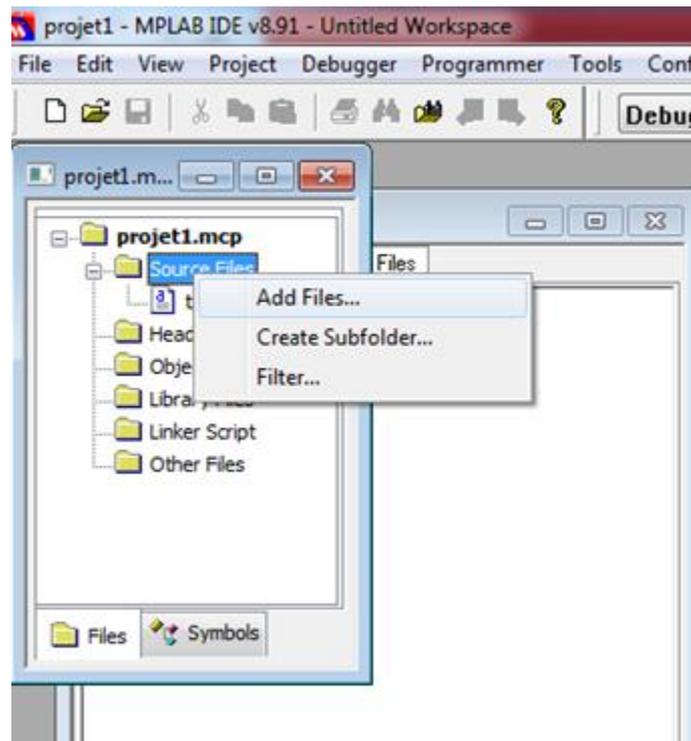
Cliquer sur Suivant > puis sélectionner « Microchip MPASM Toolsuite » et « MPASM Assembler » afin de pouvoir programmer en Assembleur.

Cliquer sur Suivant > puis dans le champ « Project Name » saisir le nom du projet (ex projet\_tp1)

Cliquer sur Suivant > puis sur Terminer.

### Édition du programme

Il faut ouvrir un fichier source (.asm) dans lequel on écrira le programme en langage assembleur :  
Bouton droit sur Source File, puis ajouter le fichier « TP1.asm »



Saisir le programme suivant et le sauvegarder, dans votre répertoire :

.....

**T1 EQU 0x30**

**MOVLW 0xFF**

**ADDLW B'01101010'**

# TP Système à microcontrôleurs

M1 Instrumentation BIOMEDICALE

**MOVWF T1**

**END**

////////////////////////////////////

Ajouter des commentaires au programme pour montrer que vous avez bien compris les différentes instructions. Une ligne de commentaire commence par « ; ».

## Compilation

Cliquer sur Project >> **Build All** ... ou sur l'icône de la barre de menu  ou encore **Ctrl+F10** pour compiler le projet complet.

Erreurs et warnings sont signalés dans la fenêtre Output ainsi que le résultat de la compilation **BUILD SUCCEEDED** ou **BUILD FAILED**.

NB: En double-cliquant sur une erreur ou un warning dans la fenêtre Output, vous serez amené directement à la ligne concernée dans votre programme. S'il n'y a pas des erreurs de compilation, on obtient le message « BUILD SUCCEEDED » et on aura la génération du fichier « TP1.hex ».

## Simulation

Choisissez l'outil de simulation MPLAB SIM comme "debugger".

Exécutez le programme pas à pas et observez l'évolution des registres PCL, Status (Onglet View -> Special Function Register)

Suivre les registres et les différentes variables (Onglet View -> Watch) : Ajouter, par le menu contextuel de la barre du nom des colonnes (clic droit) une colonne « Hex » afin d'obtenir les valeurs en hexadécimal du contenu des registres, que remarquez-vous ?

### b) Exercice :

Ajouter cette ligne dans le programme précède

```
BSF STATUS, RPO
```

Compiler le programme, que remarquez-vous ?

Ajouter les deux lignes suivantes :

```
List p=16F84A
```

```
#include <p16F84A.inc>
```

- Configurer le bit N° 3 de port A comme entrée et le bit N°5 de port B comme une sortie.
- Ecrire un programme en assembleur sous MPLAB qui permet d'allumer une LED.

## TPN°1: Commande d'une LED clignotant (3h)

### Objectifs du TP :

L'objectif de ce TP est d'introduire les étudiants à la programmation d'un microcontrôleur et apprendre à utiliser les interfaces parallèles en mode entrée (Lecture de l'état d'un bouton poussoir et en mode sortie (commande un LED)

### Manipulation

- Ecrire un programme en assembleur sous MPLAB qui permet d'allumer une LED (broche RB0) lors de l'appui sur le bouton poussoir (RA0).

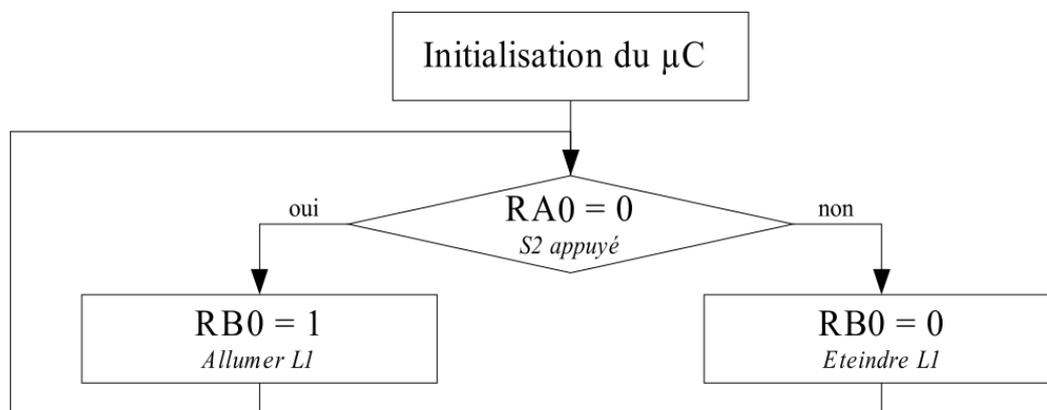
\_ Tâche 1 : lire l'état du bouton poussoir, broche RA0

- Tâche 2 : si le bouton est appuyé, on allume la diode électro-luminescente (LED) L1, broche RB0, sinon on l'éteint.

On notera que les boutons poussoirs sont branchés au microcontrôleur en logique négative. L'état « bouton

appuyé » correspond donc à un 0 logique. Les LED sont câblées en logique positive, elles s'allument pour un 1 logique.

Le fonctionnement demandé peut être reproduit par l'organigramme ci-dessous.



\_ Créer un nouveau projet dans MPLAB et le sauver dans le répertoire tp2/cligno puis saisir votre programme

## TP Système à microcontrôleurs

### M1 Instrumentation BIOMEDICALE



\_ Compiler votre programme et passer à la partie simulation.

\_ Tester le fonctionnement de votre programme dans le simulateur (Debugger) MPLAB SIM. Le simulateur fonctionne selon trois modes :

- le mode Step By Step qui permet de faire une exécution pas à pas du programme (vous êtes alors l'horloge
- du  $\mu\text{C}$ ).
- le mode Animate qui exécute automatiquement le programme mais à une vitesse réduite pour que l'on
- puisse suivre l'exécution.
- le mode Run où le programme est exécuté automatiquement à la vitesse du PC.

Nous nous intéressons aux modes Animate et Step By Step car ils nous permettent de voir qu'elle est l'influence des entrées sur les sorties.

\_ Pour visualiser les résultats de simulation, utiliser les deux outils intégrés par MPLAB SIM :

La fenêtre « Watch » (menu View >>Watch) qui affiche le contenu des différents registres du  $\mu\text{C}$  et des différentes variables du programme.

\_ La fenêtre « Logic Analyzer » permet l'affichage des éléments logiques (1 bit) en fonction du « Time Base ». Le choix des éléments à afficher se fait à l'aide du bouton « Channels ».

\_ Afficher :

- les registres TRISA et TRISB
- les registres PORTA, PORTB et W
- les bits RA0 et RB0