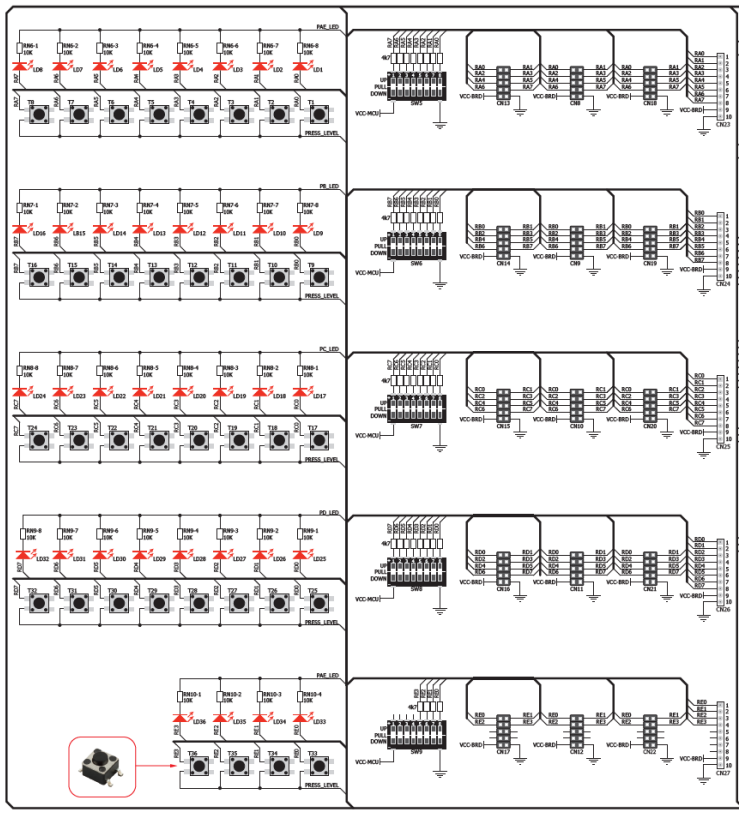
Université Mohammed Premier
Ecole Nationale des sciences appliquées
- Oujda –

Génie des Systèmes Electronique
Informatique et Réseaux
GSEIR

TP 2 – Informatique embarquée

GESTION DES PORTS I/O NUMERIQUE

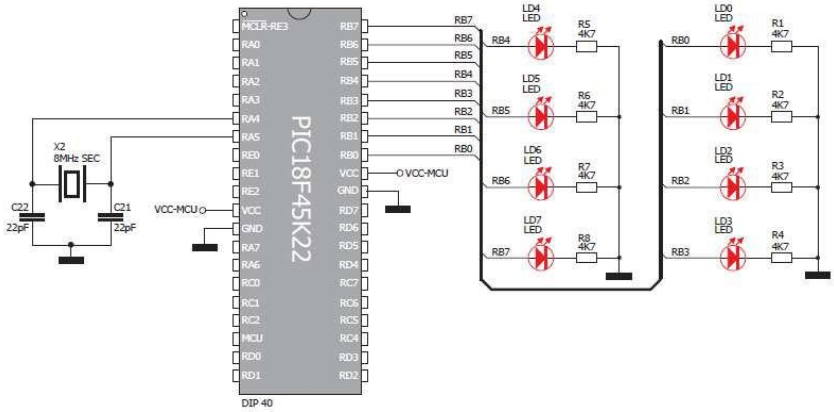


Encadré par :

Pr. EMHARRAF Mohamed

1. Objectifs

Le test des différents PORTS, créer un projet, le compiler et l'exécuter sur la platine. Le premier exemple consister à clignoter toutes les LEDs du PORTB. Le montage est le suivant :



2. Création de projet

Un fichier "projet" contient les informations suivantes :

- le nom du projet et éventuellement une description,
- le *µC* utilisé,
- la fréquence d'horloge,
- la liste des fichiers sources,
- Des fichiers binaires etc.

A présent, nous allons créer un nouveau projet, écrire du code, le compiler et le tester. Le but est de faire clignoter les LEDs du port B de notre *µC*.

1- Double cliquer sur l'icône du compilateur mikroC PRO for PIC® du menu Start ou sur le raccourci du bureau de votre ordinateur pour le faire démarrer. L'environnement de travail intégré du compilateur s'affiche sur l'écran.

2- Sélectionnez l'option "New Project" dans le menu Project ou cliquez directement sur l'icône "New Project" dans la barre d'outils "Project". Il ne reste plus qu'à se laisser guider, appuyer sur le bouton "Next"

Step1 : Configuration du projet (Project Settings)

La première des choses à faire est de spécifier des informations générales sur le projet, son nom, son emplacement, le type de *µC* et la fréquence de l'horloge (8 MHz).

Modifier le nom du projet (LedPortB) et le chemin du projet (cliquer sur Browse puis, sous le bureau, créer un répertoire TP2/LedPortB). Le µC renseigné par défaut étant le PIC 18F45K22, ne pas modifier la case Device Name. Idem pour l'horloge de 8 MHz.

Cliquer sur Next.

Step2 – Ajout de fichiers (Add files)

Si vous devez intégrer des fichiers déjà existants dans votre projet, vous pouvez le faire à ce stade. Ce n'est pas encore le cas ici, contentez-vous de cliquer sur Next.

Step3 – Inclure les librairies (Include Libraries)

Cette étape vous permet d'inclure ou non toutes les librairies dans votre projet. Le fait d'inclure toutes les librairies ne sera pas pénalisant au niveau de mémoire. En effet, seules les librairies explicitement appelées par le programme seront activées. Au total, ce sont 500 fonctions qui peuvent être appelées dans le code. Elles peuvent être visualisées dans le "Code Assistant", [CTRL+space].

Inclure toutes les librairies et cliquer sur Next.

Step4 – Fin

La dernière fenêtre permet de configurer la source d'horloge et la PLL ainsi que quelques bits de configuration. Nous utiliserons ici la configuration par défaut (oscillateur HS et PLL inactive), ne pas cocher la case et cliquer sur Finish.

Votre nouveau projet vient d'être créé. Il inclue un fichier source appelé "LedPortB.c" qui contient la fonction principale void main(). Vous remarquerez également que votre projet est configuré avec les paramètres que vous avez renseignés précédemment.

3- Exemple de code

Nous allons maintenant pouvoir écrire notre premier programme en C. La première chose à faire est d'initialiser le port B pour qu'il fonctionne en sortie numérique. :

```
// set PORTB to be digital output
TRISB = 0;
```

Le registre LATB est utilisé pour les sorties numériques (plutôt que le registre PORTB). Nous l'initialisons avec des 0 sur chacune des broches :

```
// Turn OFF LEDs on PORTB
LATB = 0;
```

Finalement, dans une boucle while(), nous allons basculer périodiquement la valeur du port B après une temporisation de 1000 ms. Cette temporisation évite que le clignotement soit trop rapide.

```
while(1) {
  // Toggle LEDs on PORTB
  LATB = ~LATB;

  // Delay 1000 ms
  Delay_ms(1000);
}
```

Tapez ce code dans la fenêtre principale. Enregistrer.

Remarque :

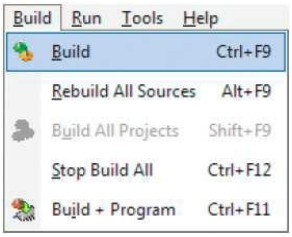
La fonction delay_ms est une fonction intégrée au compilateur. Attention, elle fonctionne selon l'horloge système (attention à la boucle PLL…).

Si vous devez afficher une fenêtre particulière, suivez le chemin View et activer la ligne correspondante. Les fenêtres peuvent être insérées directement dans l'espace de travail principal ou réduites sur les côtés (épingles).

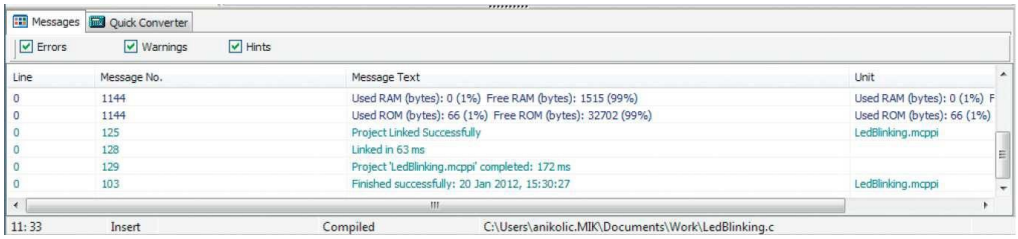
4- Compilation

A présent, nous allons compiler le projet afin de créer le fichier .hex qui sera chargé dans le µC. La compilation inclue ici la compilation à proprement parler (génération d'un code machine par fichier), l'édition des liens ou linking (lien entre fichiers et bibliothèques) et l'optimisation, tâches qui seront faite de manière automatique.

Pour compiler le projet, cliquer soit sur l'icône de la barre des tâches  ou dans le menu "Build", cliquer sur Build [CTRL+F9].



La fenêtre "message", si elle est activée, contient des détails sur le résultat de la compilation. Le compilateur créé automatiquement les fichiers de sortie, dont le fichier LedPortB.hex.

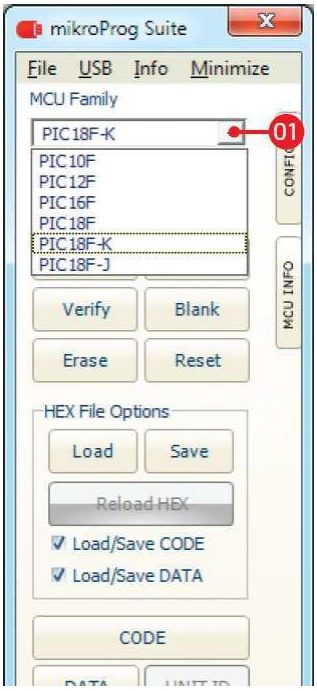


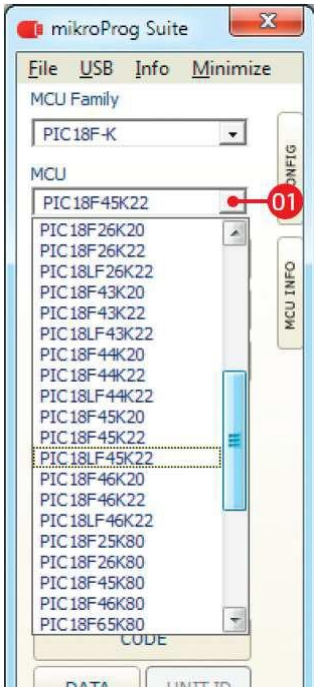
5- Programmation de la puce (µC)

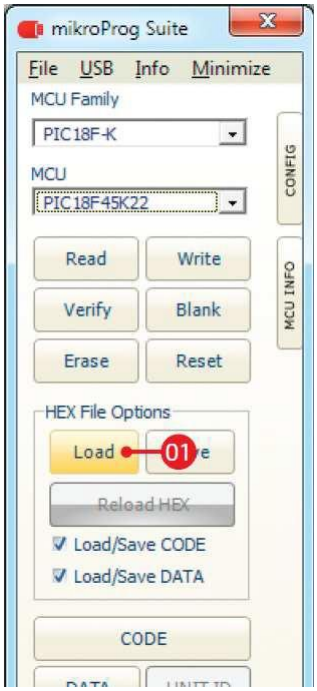
Le programmateur mikroProg embarqué sur la platine de test fonctionne avec un programme spécial appelé mikroProg Suite for PIC. Pour le lancer, cliquer sur le raccourci situé sur le bureau de votre machine

Connecter maintenant le programmateur avec le câble USB fournit, positionner le switch SW3 sur PORTB et allumer la platine de test (alimentation via le câble USB). L'icône représenté en bas à droite de la fenêtre doit passer au rouge.

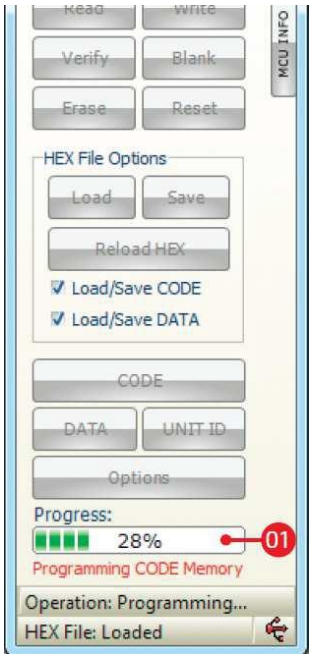
Afin de charger le fichier .hex, vous devez dans l'ordre sélectionner la famille du µC (PIC18F-K), ensuite le modèle (PIC18F45K22), sélectionner le fichier hex. (Load puis LedPortB.c).







Cliquer sur le bouton Write, le chargement est terminé lorsque Operation passe à None.



Vérifier que le programme fonctionne correctement en observant le clignotement des Leds.

6- Gestion des PORTS ABCDE par exécution pas a pas

1. Revenir sous MikroC Pro for PIC et, s'il était encore ouvert, fermer le projet précédent (File > close).

2. Créer un nouveau projet que vous appellerez **LedPortABCDE**.

3. Retapez le code .c donné en annexe (fichier LedPortABCDE.c).

4. Compiler le projet et charger le fichier LedPortABCDE.hex dans le *µC*.

5. Sur la platine valider les LEDS des ports de A à E grâce au switch **SW3**.

Questions

a) Vérifier que toutes les Leds s'allument tour à tour. Selon vous, pourquoi les Leds RA6 et RA7 ne s'allument-elles pas?

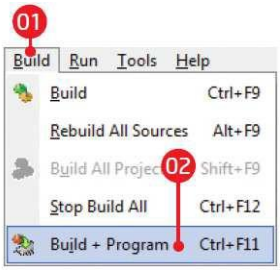
b) Expliquez le fonctionnement des lignes : LATA |= 1 << counter et LATA &= ~(1 << counter)

6. Il vous faut maintenant autoriser le débogueur mikroICD.

Revenir sous MikroC Pro for PIC.

Suivre le lien "Project Settings – Build /Debugger Type", sélectionner l'option "ICD Debug" pour autoriser la création d'un fichier.hex de débogage puis l'option « mikroICD » afin d'autoriser l'utilisation du débogueur.

A. A présent, il nous faut recompiler et reprogrammer la puce. Pour cela, cliquer sur Build > Build + Program [CTRL+ F11] ou directement sur l'icône correspondante de la barre d'outils.



Le compilateur va alors directement compiler le projet et démarrer le logiciel mikroProg Suite for Pic.

B. Pour démarrer mikroICD, cliquer sur le menu Run puis sur Start Debugger [F9].

C. Pour afficher la fenêtre des variables à surveiller, cliquer sur View > Debug Windows

› Watch Window [Shift+F5] ou directement sur l'icône de la barre d'outils.

Pour afficher la fenêtre de la RAM (Data) et visualiser son contenu, cliquer sur View › Debug Windows › RAM Window. Les valeurs affichées dans les cases mémoires sont exprimées en hexadécimal. Vous pouvez les modifier à tout moment, il suffit pour cela de rentrer une nouvelle valeur et d'appuyer sur la touche Enter de votre clavier.

De même, le contenu des mémoires Programme (Flash) et EEPROM peuvent être visualisées :

Flash : View › Debug Windows › CODE Window

EEPROM : View › Debug Windows › EEPROM Window

D. Lancer l'exécution instruction par instruction par la commande et vérifier le résultat de l'exécution de chaque ligne de code sur l'état des Portes en sorties

7- Exercices :

1. Incrémenter la valeur de PORTB (retard de 1s) qui doit être configure en sortie.

2. Afficher la somme des valeurs de PORTC et PORTB sur le PORTD.

3. Surveille l'état de l'entrée RA1 :

- Si RA1 = 0 => PORTB = 00001111
- Si RA1 = 1 => PORTB = 11110000

4. Jeux de lumière selon l'état de l'entrée, et tableau ci-contre :

- Propose un bus pour l'entrée et un autre pour la sortie.
- Donner le programme qui permet de réaliser le fonctionnement demandé, avec un retard de 500 ms entre chaque clignotement.

L'état de l'entrée	La sortie
PORT= 0b00000000	11000011 00111100
PORT= 0b00000001	10011001 01100110
PORT= 0b00000010	10101010 01010101
PORT= 0b00000011	11111111 00000000

5. Unité arithmétique et logique sur 8bits. L'ALU comporte 2 entrées de données C et B et une entrée de choix d'Operations E sur 3 bits, plus la sortie PORTD.

Valeur de E	Opération	Valeur de E	Opération
000	C AND B	100	C+ B
001	C OR B	101	C – B
010	Rotation gauche 2 bits (PORTB)	110	C / B (division euclidienne)
011	Décalage Droit 1bit (PORTB)	111	C * B

//Annexe : Code LedPortABCDE.c

```
char counter;

void main() {
  TRISA = 0x00; LATA = 0x00;
  TRISB = 0x00; LATB = 0x00;
  TRISC = 0x00; LATC = 0x00;
  TRISD = 0x00; LATD = 0x00;
  TRISE = 0x00; LATE = 0x00;

  while (1) {
    for (counter=0; counter<8; counter++) {
      LATA |= 1 << counter;
      LATB |= 1 << counter;
      LATC |= 1 << counter;
      LATD |= 1 << counter;
      LATE |= 1 << counter;
      Delay_ms(200);}

    for (counter=0; counter<8; counter++) {
      LATA &= ~(1 << counter);
      LATB &= ~(1 << counter);
      LATC &= ~(1 << counter);
      LATD &= ~(1 << counter);
      LATE &= ~(1 << counter);
      Delay_ms(200);}
  }
}
```