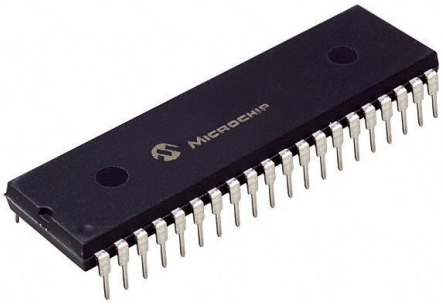
UNIVERSITE JOSEPH FOURIER
SCIENCES. TECHNOLOGIE. MEDECINE. GRENOBLE - ALPES

IUT1

Travaux pratiques de micro-contrôleur



INSTITUT UNIVERSITAIRE DE TECHNOLOGIE 1

Génie Électrique et Informatique Industrielle

T. Airault
J-L. Amalberti
F. Argeles
J-C. Castro
O. Deleage
B. Despinasse
S. Dumont
C. Florescu
J-E.Gomez-Balderas
P. Goubier
J-P. Guiramand

2013 – 2014

Planning des TPs

S1	TP1 : Utilisation des ports d'entrée-sortie
	TP2 : Mise en œuvre des Timer
	TP3 : Signaux PWM
S2	TP4 : Interruptions 1ère partie
	TP5: Le CAN
	TP6 : Liaison série RS232
	TP7 : Interruptions 2ème partie
	TP8 : TP Bilan

La carte de développement utilisée est PICDEM2 +

Vous trouverez la documentation sur cette carte à l'adresse http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en010072 (Premier site sur google en tapant « PICDEM2 PLUS »)

Attention cette carte a été fabriquée en trois versions. Une rouge, une verte et une noire. Les fonctionnalités de ces cartes sont différentes essentiellement pour la commande de l'afficheur.

Le microcontrôleur utilisé est le PIC16F887.

Nous utiliserons l'environnement de travail MPLAB.

Le compilateur utilisé est CC5X : Vous trouverez une version gratuite ainsi qu'une documentation en anglais à l'adresse suivante : http://www.bknd.com/cc5x/download.shtml

L'outil de programmation et/ou de débogage est l'ICD3 ou ICD2. Il vous suffit de regarder ce qui est marqué sur le debugger contenu dans votre kit.

TP1 : Utilisation des ports d'entrée-sortie (1ère partie)

Note : Pour tous les programmes, il vous sera demandé :

- Faire l'analyse structurée.
- Écrire le programme en C.
- Utiliser des sous-programmes (SP_INIT(), SP_TEMPO() SP_SENS(),etc…) sans oublier les commentaires.
- Vérifier vous-même et faire valider le fonctionnement par votre enseignant.

Préparation (à faire AVANT la séance de TP)

A) Dessiner le schéma de principe d'une carte avec un microcontrôleur PIC, le quartz, le circuit de reset et permettant de commander 4 LEDs connectées sur RB0 à RB3 (un niveau haut en sortie du microcontrôleur doit allumer les leds).

B) *(Préparation relative au **TP1A**)* Indiquer quel est l'état des registres TRISB et PORTB pour que les LEDs connectées sur le port B présentent le code suivant : RB0 = éteinte; RB1 = allumée; RB2 = éteinte; RB3 = allumée.

C) Aux pages 15 et 16 du manuel d'utilisation du compilateur CC5X vous trouverez les déclarations des variables : Quelles sont les valeurs min et max que peuvent prendre les principales variables utilisées en TP ? (**uns8 , uns16, char , int , bit**)

D) *(Préparation relative au **TP1B**)* Écrire un programme permettant de faire clignoter une LED (RB0) avec une temporisation d'environ **0,5 seconde** (0,5 seconde allumée et 0,5 seconde éteinte, les autres LEDs seront éteintes) . Le temps de 0,5s sera fait à l'aide d'une fonction appelée « tempo_environ_500ms() » avec une structure de type « for (….) ». On ne passera pas de paramètre à cette fonction (vous trouvez la syntaxe exacte de cette structure pages 32 et 33 du manuel de CC5X).

E) *(Préparation relative au **TP1D**)* Écrire un programme pour faire allumer et éteindre successivement et indéfiniment les 4 LEDs du kit (PB0 à PB3) pendant 0,2s environ à chaque pas.

RB3	RB2	RB1	RB0
Éteinte	Éteinte	Éteinte	**allumée**
Éteinte	Éteinte	**allumée**	Éteinte
Éteinte	**allumée**	Éteinte	Éteinte
Allumée	Éteinte	Éteinte	Éteinte
Éteinte	Éteinte	Éteinte	**allumée**
Éteinte	Éteinte	**allumée**	Éteinte
...	...	...	...

TP1A : Découverte du kit de développement

Objectifs : Être capable d'utiliser l'outil de Microchip MPLAB et le compilateur CC5X.
Être capable de mettre en œuvre les E/S TOR du microcontrôleur PIC.

Travail demandé :

1. Créer un nouveau répertoire (par exemple « TP_Info_microcontroleur ») dans votre répertoire Z : Suivre les différentes étapes de la création d'un nouveau projet comme expliqué dans le « guide de mise en route » que vous trouverez sur l'espace pédagogique (TICE) de O. DELEAGE.
   Choisir le débugger ICD2 ou ICD3 en fonction du matériel disponible sur votre kit.

2. Écrire un programme permettant d'avoir l'état des LED suivant :
   RB0 = éteinte; RB1= allumée ; RB2 = éteinte ; RB3 = allumée.

TP1B : Clignoteur temporisé

Objectif : Être capable de mettre en œuvre les structures de base du C.

- Déclaration de variables uns8, uns16, char ..
- Utilisation de sous-programmes.
- Utilisation de la boucle For (….).

Travail demandé :

Faire clignoter une LED (RB0) pendant environ **0,5 seconde** (les autres LEDs seront éteintes).
Vous devez écrire un programme principal et une fonction de temporisation (sans passage de paramètres) que vous nommerez « tempo_environ_500ms » en utilisant l'instruction for (….).

Travail complémentaire :

1) Mesurer le temps exact d'exécution de votre fonction avec la fonction STOP /WATCH.
   Indiquer sur le compte rendu votre procédure d'essai, pour pouvoir la renouveler sans aide dans les futurs TPs.

TP1C : Clignoteur avec temporisation réglable

Objectif : Mise en œuvre du passage de paramètres dans une fonction.

Travail demandé :

Faire clignoter une LED (RB0) dont le temps à l'état haut et à l'état bas sont de **X seconde**s (les autres LEDs seront éteintes).
On passera à la fonction « Temporisation_variable » le paramètre X.

TP1D : Chenillard simple

Objectif : Vérifier les acquis des deux premiers exercices.

Travail demandé :

Programmer les sorties TOR du port B pour faire clignoter successivement et indéfiniment les 4 LEDs du kit (RB0 à RB3) pendant 0,2s environ à chaque pas.

RB3	RB2	RB1	RB0
Éteinte	Éteinte	Éteinte	**allumée**
Éteinte	Éteinte	**allumée**	Éteinte
Éteinte	**allumée**	Éteinte	Éteinte
allumée	Éteinte	Éteinte	Éteinte
Éteinte	Éteinte	Éteinte	**allumée**
Etc ...	...	...	...

La fonction « Temporisation_variable_base_100ms » recevra un paramètre qui sera la durée de celle-ci en **dixièmes de secondes**.

TP1E : Chenillard avec fonction

Objectif : Mise en œuvre d'une fonction avec paramètre d'entrée**.**

Travail demandé :

Reprendre le programme ci-dessus mais en utilisant cette fois-ci une fonction qui prend en entrée la numéro de la LED à allumer et gère l'allumage et l'extinction des 4 LED.
Prototype : void allume_LED(uns8 led)

Utiliser cette fonction judicieusement dans votre programme principal.

TP1F : Chenillard avec fonction plus évoluée

Objectif : Mise en œuvre d'une fonction avec deux paramètres d'entrées.

Travail demandé :

Reprendre le programme ci-dessus en rajoutant une fonction qui prend deux paramètres d'entrée : le numéro de la LED à allumer, ainsi que le temps d'allumage de celle-ci en dixièmes de secondes.
Prototype : void allume_LED_temporise(uns8 led, uns8 duree)
Cette fonction devra bien sûr faire appel aux deux fonctions précédentes déjà écrites …

Utiliser cette nouvelle fonction judicieusement dans votre programme principal.

TP1 : Utilisation des ports d'entrée-sortie (2ème partie)

Préparation (à faire AVANT la séance de TP)

A) *(Préparation relative au **TP1G**)* Clignoteur temporisé avec bouton Marche/Arrêt : Le bouton câblé sur RA4 sert de bouton Marche/Arrêt. Lorsque le bouton est sur ON , la LED (RB0) doit clignoter à la fréquence d'environ 1Hz et lorsqu'il est sur OFF , la LED doit rester éteinte.

1) Proposer une analyse structurée du programme, avec une structure de type if( ) {} else {}
2) Écrire le programme en C.

B) (*Préparation relative au **TP1H**)* Clignoteur temporisé avec bouton Marche/Arrêt et choix de la temporisation.
Le bouton câblé sur RB0 sert de bouton marche/arrêt :
- RB0 enfoncé : le système sera à l'arrêt.
- RB0 relâché : le système sera commandé par RA4

Le bouton câblé sur RA4 permet de sélectionner la fréquence de clignotement :
- RA4 enfoncé : la fréquence de clignotement sera de 1Hz.
- RA4 relâché : la fréquence de clignotement sera de 5Hz.

La LED à faire clignoter est la LED placée sur **RB1.**

- Proposer une analyse structurée de votre programme avec une structure de type if( ) {} else {}.

C) *(Préparation relative au **TP1I**)* CHENILLARD avec M/A en bout de ligne.
Le bouton câblé sur RA4 sert de bouton marche/arrêt.
- RA4 enfoncé : le système sera à l'arrêt.
- RA4 relâché : le chenillard marche normalement.

(le chenillard démarre de la première LED).
Pensez à utiliser certaines fonctions que vous avez déjà réalisées.
Lorsque le bouton est enfoncé, la ligne du chenillard entamée se poursuivra jusqu'au bout de la ligne (le chenillard s'arrête sur le dernier code). Le chenillard reste bloqué tant que le bouton est enfoncé.
On utilisera les 4 LEDs connectées sur RB0 à RB3 (un niveau haut en sortie du microcontrôleur doit allumer les LEDs).

- Proposer une analyse structurée de votre programme avec une structure de type if( ) {} else {}.

D) *(Préparation relative au **TP1J**)* CHENILLARD avec M/A sur la LED :
Reprendre le chenillard précédent avec cette fois-ci un marche/arrêt sur chaque LED. Dès que le bouton RA4 sera sur OFF, le chenillard devra s'arrêter sur la LED et bien sûr redémarrer de cette LED lorsque le bouton passe à ON.

- Proposer une analyse structurée de votre programme avec une structure de type if( ) {} else {}.

TP1G : Clignoteur temporisé avec bouton Marche/Arrêt

Objectifs : Être capable de mettre en œuvre les ports en entrée et en sortie.
Être capable de mettre en œuvre la structure IF THEN ELSE en C.

Travail demandé : Tester votre programme fait en préparation.

TP1H : Clignoteur temporisé avec bouton Marche/Arrêt et choix de la temporisation

Travail demandé : Écrire puis tester le programme C à partir de votre analyse structurée faite en préparation.

Attention ! : page 49 de la documentation → paragraphe 3.4.1, page 50 → register 3-4 et page 52 → figure 3-9. Ces informations sont à prendre en compte sérieusement avant d'arriver à la conclusion que ça ne marche pas ...

TP1I: CHENILLARD avec M/A en bout de ligne

Travail demandé : Écrire puis tester le programme C à partir de votre analyse structurée faite en préparation.

TP1J: CHENILLARD avec M/A sur la LED

Objectif : Élaboration d'un programme demandant une analyse structurée plus fine.

Travail demandé : Écrire puis tester le programme C à partir de votre analyse structurée faite en préparation.

TP1K: CHENILLARD avec gestion du sens

Objectif : Élaboration d'un programme demandant une analyse structurée plus fine.

Travail demandé :

Réaliser le chenillard du TP1F en y ajoutant la possibilité de changer de sens sur la LED. Dès que RA4 changera de position, le chenillard devra changer de sens.
Proposer une analyse structurée de votre programme avec une structure de type if( ) {} else {}, puis écrire et tester le programme.

TP1L: CHENILLARD avec gestion du sens et M/A sur la LED

Objectif : Élaboration d'un programme demandant une analyse structurée plus fine.

Travail demandé :

Reprendre le chenillard précédent avec cette fois-ci un marche/arrêt (bouton RB0) et un changement de sens (bouton RA4) sur chaque LED.
Ici nous utiliserons un chenillard sur 3 LEDs (RB0 étant pris par le bouton poussoir).
Proposer une analyse structurée de votre programme avec une structure de type if( ) {} else {}, puis écrire et tester le programme.