

# TP0 PIC32MX LED et A/D

# **OBJECTIFS**

Cette première manipulation (**réalisation individuelle**) a pour but la découverte de la réalisation d'un projet avec le MHC (MPLAB Harmony Configurator). Les actions prévues sont les suivantes :

- Déclenchement d'un traitement cyclique dans l'application toutes les 100 ms.
- Lecture des 2 potentiomètres avec l'AD et affichage de la valeur brute.
- Réalisation d'un chenillard avec les 8 leds du kit.

## PRINCIPE DE LA REALISATION

La réalisation se fait en 2 étapes :

- 1) Mise en place du projet avec le MHC.
- 2) Ajout et adaptations nécessaires pour obtenir le comportement souhaité.

### INSTALLATION ET CONFIGURATION DE L'ENVIRONNEMENT

Il est nécessaire que MPLABX, Harmony et XC32 soient installés et configurés conformément au chapitre 2 du cours TP.

Veuillez créer ou utiliser le répertoire suivant :

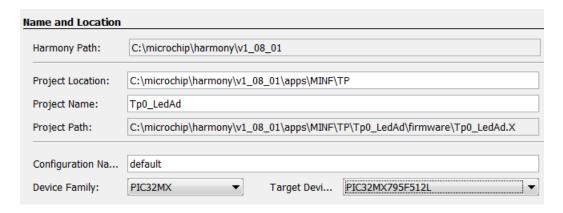
C:\microchip\harmony\v<n>\apps\MINF\TP

## MISE EN PLACE DU PROJET AVEC LE MHC

A faire sur la base de la démo au beamer et du chapitre 2 (surtout §2.3) du cours TP.

Nom du projet : TP0\_LedAd.

A placer sous : C:\microchip\harmony\v<n>\apps\MINF\TP





Actions au niveau MPLABX, lorsque le projet est créé :

Avec MPLAB Harmony Configurator, Onglet Options

- 1. Sélection du BSP spécifique pour le kit ES (développer BSP Configuration), importation de la configuration par défaut.
- 2. Harmony Framework Configuration, Drivers, Drivers. Créez une instance statique d'un driver de timer avec interruption. Utilisation du timer1 avec un cycle de 100 ms. Choisir un niveau de priorité de 3.
  - Attention: Le timer1 n'a que les prescalers 1, 8, 64 ou 256.
- 3. Génération du code

#### Puis dans MPLAB X

- 4. Propriétés du projet : Hardware Tool ICD3. Optimization 0 et additional warnings
- 5. Build

#### ADAPTATION DU PROJET

L'essentiel de l'adaptation est réalisé au niveau du fichier app.c, mais il y a d'autres modifications.

#### MODIFICATION DU FICHIER APP.H

Il faut ajouter APP STATE WAIT=1, dans le type énuméré APP\_STATES.

Il faut ajouter S\_ADCResults AdcRes; dans le typedef struct APP\_DATA (nécessaire d'inclure le .h correspondant).

```
Il faut ajouter le prototype de la fonction APP_UpdateState :
void APP UpdateState ( APP STATES NewState );
```

#### MODIFICATION DU FICHIER APP.C

Il faut ajouter l'implémentation de la fonction APP\_UpdateState.

Au niveau de la fonction APP\_Tasks, dans le *switch* (*appData.state*), on introduira les traitements suivants :

## Au niveau du case APP\_STATE\_INIT :

- Initialisation du LCD et activation rétro-éclairage (voir fichier Mc32DriverLcd.h dans bsp).
- Affichage sur 1ère et 2ème lignes (avec la fonction printf\_lcd):

```
Tp0 Led+AD 202x-2y
Nom
```

- Initialisation de l'AD (voir fichiers Mc32DriverAdc.h ou Mc32DriverAdcAlt.h dans bsp).
- Allumer toutes les leds
- Lancer le timer1 en appelant la fonction DRV\_TMR0\_Start()
- Etat suivant = APP\_STATE\_WAIT



Aucune action dans le case APP\_STATE\_WAIT, mais il faut l'ajouter dans le switch.

Au niveau du case APP\_STATE\_SERVICE\_TASKS:

- Lecture des 2 pots
- Affichage sur 3ème ligne Ch0 xxxx Ch1 yyyy
- Eteindre toutes les leds (seulement la 1ère fois que l'état est **SERVICE\_TASKS**)
- Gestion du chenillard en allumant 1 seule led à chaque cycle (à disposition : les fonctions LED définies dans bsp\_config.h).
- Etat suivant = APP STATE WAIT

A compléter avec les include nécessaires ainsi que l'ajout de variables.

Remarque : Il y a 3 moyens de gérer les ports connectés aux leds : soit écriture directe, soit utilisation des fonctions BSP, ou encore utilisation des fonctions de PLIB\_PORTS.

## MODIFICATION DU FICHIER SYSTEM\_INTERRUPT.C

Il est demandé d'ajouter dans la routine de réponse à l'interruption du timer1 (cycle 100 ms) un mécanisme qui établit **APP\_STATE\_SERVICE\_TASKS** après 3 secondes (30 cycles) et qui par la suite établit **APP\_STATE\_SERVICE\_TASKS** à chaque cycle en utilisant la fonction **APP\_UpdateState**.

# TRAVAIL ET ETABLISSEMENT DU MINI-RAPPORT

Il n'est pas demandé de rapport complet. Les éléments suivants sont demandés au minimum :

- Démonstration du fonctionnement.
- Une copie d'oscilloscope montrant le timing du chenillard avec 4 des 8 leds.
- Listing de app.c.
- Listing du contenu de la routine de réponse à l'interruption du timer1.

Votre travail sera évalué sur la base de :

- Qualité, facilité de réutilisation/modification et taux d'aboutissement du code.
- Fonctionnement et taux d'aboutissement.

# **DUREE DE LA MANIPULATION**

A réaliser en 1 séance.