Introduction aux dsPIC

But de la manipulation

L'objectif de cette manipulation est d'introduire un microcontrôleur d'une famille couramment utilisée : les PIC de Microchip. Quelques systèmes simples basés sur des entrées-sorties seront réalisés.

En parallèle, ce premier labo vous aidera à vous familiariser avec le langage C.

Par la suite, nous verrons comment réaliser un programme régi par le temps.

Prérequis

Avant d'entrer au laboratoire, il est conseillé de lire les sections 1 à 4 du document *Programmation d'une carte à microcontrôleur*

Objectifs

A la fin de ce laboratoire vous devez être capable :

- De réaliser un programme simple pour microcontrôleur.
- D'expliquer la notion d'entrées-sorties ainsi que la sortance.
- D'utiliser une base de temps dans votre programme par le biais d'un Timer.

Introduction

Durant ces travaux pratiques, vous serez amenés à utiliser une carte à microcontrôleur : l'*Explorer16*. En plus de la programmer, vous devrez comprendre le fonctionnement de ses différents périphériques et comment l'interfacer avec le monde extérieur.

Ce premier laboratoire a pour but de vous faire prendre en main la carte et d'utiliser quelques périphériques de base. Vous apprendrez à interagir de manière simple avec le microcontrôleur de la carte.

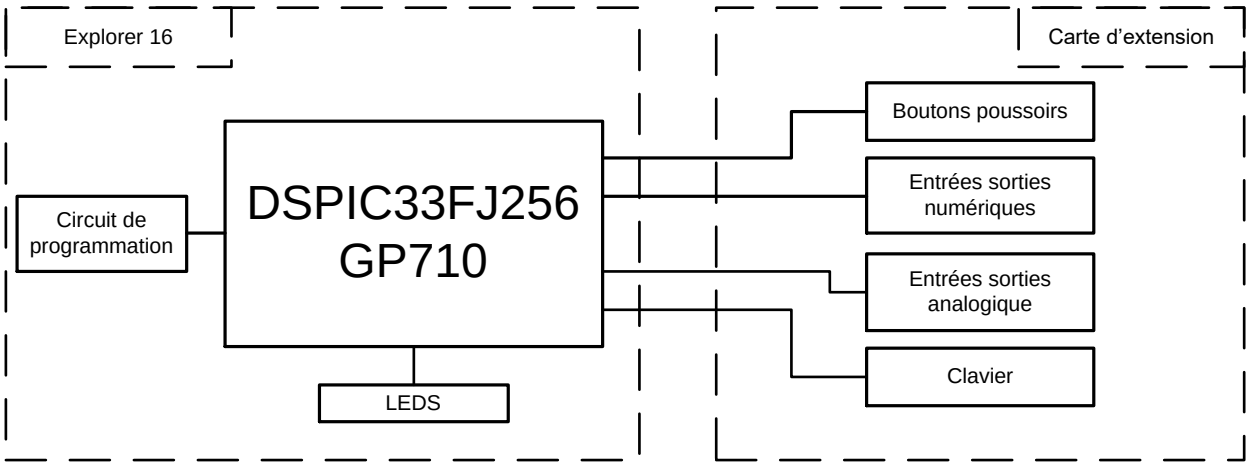

Figure 1 : schéma-bloc de l'Explorer16 et de sa carte d'extension

Le fonctionnement de la carte ainsi que du dsPIC est expliqué dans le document *Programmation d'une carte à microcontrôleur*.

Pour les deux premiers labos, il vous est conseillé d'utiliser en particulier la section *Première prise en main.*

Utilisation des entrées-sorties

Votre premier programme utilisera les GPIO du dsPIC.

Le cahier des charges est simple : on désire que la LED branchée à la borne RA0 s'allume lorsque l'on appuie sur un bouton au choix, et s'éteigne lorsqu'on le relâche.

Pour cela :

- Choisissez un bouton-poussoir de la carte d'extension et repérez à quelle I/O du dsPIC il est connecté.
- Créez un nouveau projet basé sur le projet *EmptyProj* (voir *Utilisation de MPLAB X*).
- Dans la fonction *main()* :
    - Configurez les I/O utilisées en entrée ou en sortie.
    - Modifiez la boucle principale pour respecter le cahier des charges.
    - Compilez et chargez le code sur la carte à µC, et vérifiez son fonctionnement.

Ajout d'une LED ultrabrillante

Pour améliorer notre système, on décide de remplacer la LED de l'Explorer16 par une LED ultrabrillante connectée à une des I/O de la carte d'extension.

Sachant que cette LED consomme 20mA et a une tension de seuil de 3,2V, expliquez :

- Pourquoi une connexion directe entre le dsPIC et la LED n'est pas une bonne solution ? Faites le lien avec la notion de sortance statique.
- Comment l'utilisation d'un circuit du type *buffer* résout le problème. Tracez un schéma-bloc de la solution.

Nous allons utiliser le *buffer* 74ACT244 dont la notice est fournie en annexe.

- Réalisez un montage permettant d'allumer la LED.
    - N'oubliez pas de dimensionner la résistance de limitation du courant dans la diode.
    - Alimentez le buffer en 0V/5V
- Modifiez votre programme pour qu'une pression sur le bouton allume désormais la LED externe.

Utilisation d'un Timer

Ajoutons maintenant une deuxième fonctionnalité à notre système : faire clignoter en permanence la LED de RA1 à 1Hz.

Pour cela, nous utiliserons le *Timer2*. Dans un premier temps, pour apprendre à l'utiliser, vous allez faire commuter la sortie *RC1* à 10kHz :

- Quelle valeur doit contenir *PR2* pour que le *Timer2* déborde à la bonne fréquence ?
- Quelle est la période maximale (pour rappel, Fcy = 40MHz) ?
- Dans votre code :
    - Configurez le *Timer2* et *RC1*.
    - Utilisez le *Polling* pour modifiez RC1 à chaque débordement du *Timer2*.
- Vérifiez le bon fonctionnement de votre programme à l'oscilloscope.

Maintenant que vous savez utiliser un Timer,

- Modifiez le programme pour faire clignoter la LED de RA1 à 1Hz.