

Utilisation de MPLAB-X avec le DSPIC33FJ16MC304

Les indications ci-dessous vous aiderons à créer un projet, le déboguer avec le simulateur, le charger et le déboguer dans le DSPIC avec l'ICD3.

Il vous faudra cependant découvrir par vous-même les autres fonctionnalités du logiciel MPLAB.

Travail préalable :

- Créer un répertoire personnel à votre nom pour y stocker vos projets.
- Créer dans ce répertoire des répertoires nommés TP1, TP2,...TP7
- Copier les fichier "*TP1.s*" dans le répertoire **TP1**.

Lancer MPLAB-X

- Créer un nouveau projet (File- New Project)
- Microchip Embedded – Standalone Project
- Family : **16bits DSCs(dsPIC33)**
- Device : **dspic33FJ16MC304**
- Hardware Tools : Choisir **Simulator** pour la simulation sur PC

ICD3 sera utilisé par la suite pour programmer ou déboguer le DSPIC.

- Compiler Toolchain : **XC16**
- Projet Name : **TP1**
- Projet Location : Choisir le répertoire **TP1** créé précédemment
- Cochez la case : **Set as main projet**

La fenêtre **Projects** doit s'afficher (Windows- Projects permet de la faire réapparaître)

- Ajouter le fichier **TP1.s** à **Sources Files** (Cliquez avec le bouton droit)
- Ajouter **p33FJ16MC304.gld** à **Linker Files** sinon un fichier .gld par défaut sera utilisé. Ce fichier se trouve dans le répertoire d'installation de XC16 (Microchip\xc16\v1.20\support\dsPIC33F\gld\)

Vous pouvez maintenant Simuler et ainsi vérifier le fonctionnement du programme:

- Debug – Debug Main Project
 - Afficher l'état des registres et de la RAM avec : *Window – PIC Memory Views et Debug - New Watch*
 - Le pas à pas (**Step Into**) permet d'exécuter le programme instruction par instruction
 - Placer des points d'arrêt (**Breakpoint**) et utiliser **Continue** pour les portions de codes trop longues à simuler (boucles)
 - On peut aussi demander d'exécuter jusqu'au curseur de la souris (très utile)

Pour charger le programme dans le DSP

- Remplacer **Simulator** par **ICD3** dans les **propriétés** du projet (bouton droit sur le projet)
- Run- Run Main Project
- ou Debug - Debug Main Project pour déboguer dans le DSPIC -

TP1

Échantillonnage d'une entrée analogique

Il s'agit de se familiariser avec l'utilisation de quelques périphériques intégrés au DSPIC : Ports d'entrées /sorties, Timer, ADC...

Le générateur de fonction est relié à l'entrée analogique 0 (AN0) sans filtre, le signal d'entrée doit être compris entre 0 et 3.3V (une résistance et une diode zéner 3.3V protègent l'entrée...).

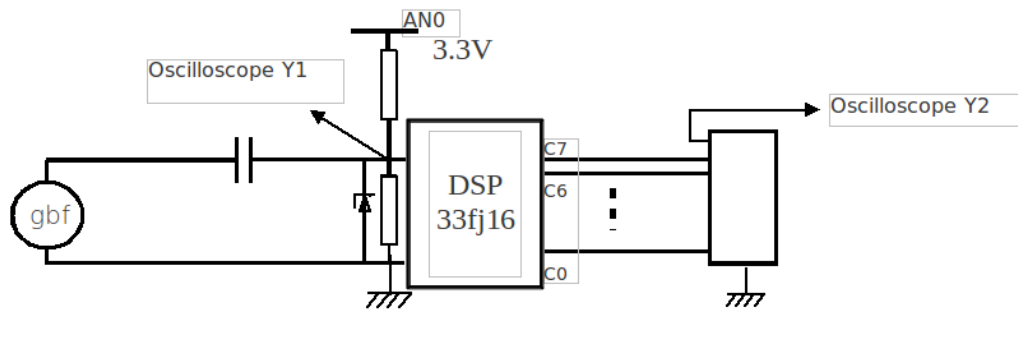


Schéma du dispositif expérimental

Le programme fourni, lit à intervalle de temps régulier l'entrée analogique AN0 et envoie les 8bits de poids forts de la valeur lue sur un port parallèle associé à un convertisseur numérique analogique ici un simple réseau R/2R.

Simulation en mode pas à pas

1) Etudier le programme assembleur "Tp1.s" (Utiliser fichiers .pdf spécifiques pour la description des registres 70183d.pdf pour l'ADC,, 70205d.pdf pour le timer, 70293d.pdf pour les entrées sorties tout ou rien 70157F.pdf pour les instructions)

2) Lancer la simulation en mode pas à pas en affichant les registres systèmes et périphériques concernés, vérifier le résultat de chacune des instructions.

3) Utiliser le simulateur pour déterminer la période d'échantillonnage T_e (intervalle de temps entre deux conversions successives ($f_{cy} = 3,6864\text{MHz}$ pilote le Timer3))

Transfert et tests sur la cible (DSPIC)

4) Mesurer T_e à l'aide de l'oscilloscope et comparer au résultat de la question 3)

5) La boucle d'attente qui utilise le Timer1 permet de régler la période d'échantillonnage. Modifier sa valeur pour échantillonner à la fréquence $F_e = 10\text{ kHz}$.

6) Faire varier la fréquence du signal sinusoïdal d'entrée et mettre en évidence le phénomène de repliement du spectre, mesurer la fréquence de Shannon à l'oscilloscope et comparer à F_e (synchroniser de préférence l'oscilloscope sur le signal de sortie)

7) Modifier le programme pour avoir $F_e = 40\text{kHz}$, vérifier à l'oscilloscope ...

8) Supprimer la boucle d'attente et mesurer à l'oscilloscope la période d'échantillonnage comparer à la vitesse du convertisseur ADC... et conclure...

9) Créer un nouveau projet TP1_C avec le programme source ci-dessous écrit en langage C

- Le simuler (Simulator)
- Le charger dans la cible (ICD3)

10) Refaire la question 8 en C

11) Faire les modifications pour obtenir $F_{cy} = 40\text{MHz}$ en utilisant l'oscillateur interne FRC et la PLL (voir documentation 70186E.pdf)
Mesurer la nouvelle valeur de T_e à l'oscilloscope

;Programme source en langage assembleur : Tp1.s

```

;-----
;Le programme lit en boucle à intervalles de temps réguliers l'entrée analogique AN0
;et envoie les 8bits de poids forts du résultat de la conversion sur le port C
;Le Timer 3 est utilisé pour fixer l'intervalle de temps entre 2 conversions
;Le programme scrute en permanence le timer 3 (on n'utilise pas les interruptions dans ce TP)
;-----

;**** Désactivation du watchdog****
.section __FWDT.sec,code
.global __FWDT
__FWDT: .pword 0xFF7F
;*****

;**** Le programme ****
.global __reset ; Utilisé par le fichier 33fj16mc304.gld pour implanter le programme
.section .text ; indique au linker d'utiliser la mémoire programme pour la suite
;Adresses Instructions ; Commentaires
__reset:
    mov #0xF00,W1 ; initialisation du pointeur de pile en haut de la RAM_Y
    rcall __init_timer3 ; appel de sous-programmes
    rcall __init_adc
    rcall __init_PORTC
boucle1:
    clr TMR3 ; remise à 0 du Timer 3
    mov #0x200,W1
boucle2:
    mov TMR3,W0 ; On reste dans la boucle 2
    sub W1,W0,W0 ; tant que le Timer 3
    bra GT,boucle2 ; n'a pas atteint 0x200
    mov ADC1BUF0,W0 ;lecture de l'entrée analogique AN0
    lsr W0,#8,W0 ;décalage à droite de huit bits
    mov W0,PORTC ;envoi sur le port D
    bra boucle1 ;on recommence à la ligne "boucle1"
;*****
;fin du_programme principal
;*****
; Les sous-programmes
;*****
__init_PORTC:
    clr TRISC
    return
__init_timer3:
    mov #0x8000,W0
    mov W0,T3CON
    mov #0xFFFF,W0
    mov W0,PR3
    return
__init_adc:
    mov #0xFFFE,W0
    mov W0,AD1PCFGL
    mov #0x0000,W0
    mov W0,AD1CON2
    mov #0x010A,W0
    mov W0,AD1CON3
    mov #0x0000,W0
    mov W0,AD1CHS0
    mov #0x82E6,W0
    mov W0,AD1CON1
    return
.end ; fin du fichier

```

;Programme source en langage C : Tp1.C

```

/*;-----
.Le programme lit en boucle à intervalles de temps réguliers l'entrée analogique AN0
et envoie les 8bits de poids forts du résultat de la conversion sur le port D
Le Timer 3 est utilisé pour fixer l'intervalle de temps entre 2 conversions
Le programme scrute en permanence le timer 3 (on n'utilise pas les interruptions )
-----*/

#include "p33FJ16MC304.h"
#pragma config FWDTEN = OFF          // Watchdog Timer Enable (Watchdog timer enabled/disabled by user software)

//Liste des fonctions *****
void init_portC(void);
void init_timer3(void);
void init_adc(void);
//*****

//Le programme principal
int main(void)
{
    unsigned int a;
    init_timer3();                //appel de sous-programmes
    init_adc();
    init_portC();
    while(1)                      //Boucle infinie
    {
        TMR3=0;
        while(TMR3<0x200);        //le programme reste sur cette ligne tant que TMR3 est inférieur à 0x200
        a = ADC1BUF0;              //lecture AN0
        a=a>>8;                    //décalage à droite de huit bits
        PORTC =a;                  //envoi en sortie
    }
}

//fin du_programme principal
//*****

//*****
//Définition des fonctions
//*****

void init_portC(void)
{
    TRISC=0x0000;
}

void init_timer3(void)
{
    T3CON=0x8000;
    PR3=0xFFFF;
}

void init_adc(void)
{
    AD1PCFGL=0xFFFE;
    AD1CON2=0x0000;
    AD1CON3=0x010A;
    AD1CHS0=0x0000;
    AD1CON1=0x82E6;
}

```