Talamidi.com تم تحميل هذا الملف من موقع

1. Introduction

Les étapes nécessaires permettant de voir un programme s'exécuter sur un µC 16F84 sont :

1. Ecrire un programme en langage assembleur dans un fichier texte et le sauvegarder avec l'extension **asm**.

2. Compiler ce programme avec l'assembleur **MPASM** fourni par MicroChip. Le résultat est un fichier avec l'extension **hex** contenant une suite d'instruction compréhensible par le µC.

3. Transférer le fichier **hex** dans la mémoire programme du µC (Flash ROM) à l'aide d'un programmateur adéquat. On peut utiliser les programmateurs de MicroChip ou tout autre programmateur acheté ou réalisé par soit même.

4. Mettre le µC dans son montage final, mettre sous tension et vérifier le fonctionnement.
MicroChip propose gratuitement l'outil de développement **MPLAB** qui regroupe l'éditeur de texte, le compilateur MPASM, un outil de simulation et le logiciel de programmation.

On utilisera MPLAB pour écrire, compiler et éventuellement simuler les programmes, ensuite on les transférera à l'aide du programmateur **JDM** piloté par le logiciel **ICPROG**, les deux sont disponibles gratuitement sur le Web.

2. Directives de MPASM

Les directives de l'assembleur sont des instructions qu'on ajoute dans le programme et qui seront interprétées par l'assembleur MPASM. Ce ne sont pas des instructions destinées au µC.

21. Directives les plus utilisées

☑ **LIST** : Permet de définir un certain nombre de paramètres comme le processeur utilisé (p), la base par défaut pour les nombres (r) ainsi que d'autres paramètres.
Exemple : LIST p=16F84, r=dec.

☑ **#INCLUDE** : Permet d'insérer un fichier source. Par exemple le fichier **p16F84.inc** contient la définition d'un certain nombre de constantes comme les noms des registres ainsi que les noms de certains bits du µC 16F84.
Exemple : #INCLUDE "p16f84.inc" ou #INCLUDE ⟨p16f84.inc⟩.

☑ **__CONFIG** : Permet de définir les 14 fusibles de configuration qui seront copiés dans le registre de configuration lors de l'implantation du programme dans le µC.
Exemple : __CONFIG B'11111111111001' ou __CONFIG H'3FF9'.
🔔 **Remarque** : Si le fichier p16f84.inc a été inséré, on peut utiliser les constantes prédéfinies :
__CONFIG _CP_OFF & _XT_OSC & _PWRTE_OFF & _WDT_OFF

☑ **EQU** : Permet de définir une constante.
Exemple : XX EQU 0x20.
Chaque fois que le compilateur rencontrera la valeur XX, il la remplacera par la constante 0x20. Ça peut être une constante s'il s'agit d'une instruction avec adressage immédiat, ou d'une adresse s'il s'agit d'une instruction avec adressage direct.

☑ **#DEFINE** : Définit un texte de substitution.
Exemple : #DEFINE monbit PORTA,1
Chaque fois que le compilateur rencontrera monbit, il le remplacera par PORTA,1.

☑ **ORG** : Définit la position dans la mémoire programme à partir de laquelle seront inscrites les instructions qui suivent cette directive ORG.
Exemple : ORG 0 ; adresse du début du programme principal.
ORG 4 ; adresse du début du sous-programme d'interruption.

Talamidi.com تم تحميل هذا الملف من موقع

☑ **CBLOCK …ENDC** : Définit un bloc de variables logé dans la zone mémoire RAM accessible par l'utilisateur.

Exemple :
```
CBLOCK 0x0C        ; zone mémoire RAM accessible par l'utilisateur débute à l'adresse 0x0C
Var1 : 1           ; zone de 1 octet
Var2 : 8           ; zone de 8 octets
ENDC
```

☑ **END** : Indique la fin du programme.

22. Format des nombres

L'assembleur reconnaît les nombres en décimal, hexadécimal, binaire ou octal. Pour préciser la base il faut utiliser les préfixes précisés dans le tableau de la figure 1. On peut à l'aide de la directive LIST définir un format par défaut. Si par exemple on place l'instruction LIST r=dec au début du programme, tous les nombres sans préfixe seront interprétés en décimal.

Figure 1

Base	Préfixe	Exemple : 36
Décimal	D'nnn'	D'36'
	.nnn	.36
Hexadécimal	H'nn'	H'24'
	0xnn	0x24
	nnh	24h
Binaire	B'...'	B'00100100'
Octal	O'nnn'	O'44'

23. Structure d'un programme écrit en assembleur

Un programme écrit en assembleur doit respecter une certaine syntaxe et un certain nombre de règles afin qu'il soit facile à lire et à débuguer :

1. Quelques lignes de commentaire précisant la fonction du programme.

2. Tout ce qui commence à la première colonne est considéré comme une étiquette (Label) permettant de faire des sauts et des appels aux sous-programmes.

3. Tout ce qui suit un point virgule est considéré comme un commentaire non interprété par le compilateur.

4. Un programme apparaît donc comme un texte écrit sur 3 colonnes :

☑ La colonne de gauche contient les étiquettes.
☑ La colonne du milieu contient les instructions.
☑ La colonne de droite contient les commentaires.

5. Si le programme utilise des interruptions, mettre à l'adresse 0, adresse du branchement du µC lors d'un Reset ou d'une mise sous tension, une instruction de branchement au début du programme principal.

Exemple :
```
ORG 0
GOTO  début
```

6. Ecrire le sous-programme d'interruption à l'adresse 4 ou mettre à cette adresse un branchement au même sous-programme.

Exemple :
```
ORG 4
Ecrire le sous-programme d'interruption ici.
RETFIE
```

🔔 **Remarque** : Si le programme ne gère pas les interruptions, on peut se passer des étapes 5 et 6.

Talamidi.com تم تحميل هذا الملف من موقع

7. Ecrire les sous-programmes (s'il y en a). Chaque procédure commence par une étiquette qui représente son nom, et se termine par l'instruction Return.

8. Ecrire le programme principal (commençant par l'étiquette début : si les étapes 4 et 5 sont présentes).

9. Terminer avec la directive END.

Exemple : La figure 2 montre un extrait du programme écrit en assembleur.

Figure 2

Etiquette	Assembleur	Commentaire
	LIST p=16F84	; indique le µC utilisé
	#INCLUDE <p16F84.inc>	; fichier contenant les noms des registres internes.
	__CONFIG H'3FF1'	; configuration des fusibles
		; **Initialisation**
	ORG 0	; début du programme à l'adresse 0
	BSF STATUS,5	; sélectionner la banque 1
	BCF OPTION_REG,7	; initialiser le registre option
	BCF TRISA,2	; configurer la broche 2 du port A en sortie
	BSF TRISB,2	; configurer la broche 2 du port B en entrée
	BCF STATUS,5	; repasser en banque 0
	BCF PORTA,2	; initialiser la broche RA_2 à 0