MPLAB



Qu'est-ce que MPLAB

MPLAB est un outil de développement spécifique aux microcontrôleurs PIC, conçu et fourni gratuitement par Microchip.

Il permet d'écrire, mettre au point et optimiser les programmes, sous Windows.

En effet, en plus d'un éditeur et d'un assembleur, il inclut un ensemble d'outils permettant non seulement de fabriquer le code objet d'une application, mais aussi de simuler le programme, c'est à dire le voir dérouler à l'écran.

Il supporte plusieurs outils externes, parmi lesquels deux modèles de programmateurs (PICSTART PLUS et PROMATE 2) et deux émulateurs (PICMASTER et PICMASTER-CE).

MPLAB est facile à apprendre et à utiliser.

Il offre beaucoup de flexibilité aux développeurs, notamment grâce aux nombreuses fenêtres pouvant être ouvertes à tout moment lors d'une mise au point, permettant de voir le contenu d'un quelconque emplacement mémoire et d'un quelconque registre.

Comment obtenir MPLAB

MPLAB est téléchargeable à partir du site Internet de Microchip (www.microchip.com).

Si vous aviez besoin de le copier sur disquettes, prévoyez-en un bon nombre (il vous en faudra au moins huit).

Vous seriez en possession de fichiers compressés, que vous devriez décompresser au moyen d'un logiciel adapté (WinZip, par exemple).

Remarquez que MPLAB est un produit en constante évolution, et qu'en fonction de la version qui vous serait fournie, vous pourriez trouver un certain nombre de fichiers modifiés ou différents d'une version à l'autre.

La version que Microchip diffuse au moment où j'écris et dont je me sers dans mes exemples est la version 5.20

Caractéristiques requises par le PC pour l'installation de MPLAB

Pour pouvoir être installé, MPLAB nécessite la configuration minimale suivante :

- un PC compatible 486 ou mieux ;
- Microsoft Windows depuis 3.1 jusqu'à la version la plus récente ;
- écran VGA ;
- 8 Méga de mémoire (32 Méga recommandés) ;
- 20 Méga d'espace disponible sur le disque dur ;
- la souris.

Comment installer MPLAB

MPLAB s'installe automatiquement.

Pendant la procédure d'installation, qui démarre toute seule, plusieurs fenêtres vont s'ouvrir.

La première est celle qui vous souhaite la bienvenue et vous informe que vous vous préparez à installer la version 5.20 de MPLAB . Cliquez sur le bouton *Next*.



MPLAB v5.20 Installation X Select Components Choose which components to install by checking the boxes below. MPLAB IDE Files 3589 k MPASM/MPLINK/MPLIB Files 7559 k MPLAB-SIM Simulator Support Files 5911 k MPLAB-ICE Emulator Support Files 5111 k F PICMASTER Emulator Support Files 1198 k F PRO MATE Support Files 663 k PICSTART Plus Support Files 152 k MPLAB-ICD Debugger Support Files 245 k 🔽 Help Files 3010 k Disk Space Required: 20466 k 299897 k Disk Space Remaining: < <u>B</u>ack <u>N</u>ext > <u>C</u>ancel

Dans la fenêtre suivante, décochez les éléments que vous ne souhaitez pas installer.

A moins que vous ayez acheté un simulateur ou un émulateur (chose dont je doute fort !), vous ne devriez installer que les outils logiciels courants, et l'aide.

Décochez les autres.

Cliquez ensuite sur le bouton Next.



Dans la fenêtre *Select Language Components*, vous ne devriez maintenir actif que tout ce qui se rapporte à Windows.

Décochez tout ce qui se rapporte au DOS (Je n'aime pas les programmes tournant sous DOS. Vous non plus, j'imagine !).

La fenêtre suivante (*Select Destination Directory*) vous indique le chemin que l'installation va prendre dans votre PC.

| MPLAB v5.20 Installatio | n | × |
|---------------------------------------|---|---|
| | Select Destination 1 Please select the directory where the MP be installed. | D irectory LAB v5.20 files are to |
| *** | C:\Program Files\MPLAB | Browse |
| · · · · · · · · · · · · · · · · · · · | < <u>B</u> ack <u>N</u> ext > | <u>C</u> ancel |

Acceptez, en cliquant sur Next.

La fenêtre suivante vous demande si vous voulez créer des copies de sauvegardes des fichiers que l'installation pourrait être amenée à remplacer.

Acceptez la configuration par défaut (No) en cliquant sur Next.



MPLAB v5.20 Installation
Add to Start Menu?
Do you want to create shortcuts to access the installed files?
C Yes
No

< <u>B</u>ack

<u>N</u>ext >

<u>C</u>ancel

Dans la fenêtre suivante (*Add to Start Menu*?) acceptez la suggestion *Yes*.



Faites la même chose dans la fenêtre suivante :

Acceptez la configuration suggérée, en cliquant sur Next.

Puis acceptez encore ce qui vous est proposé dans la fenêtre qui suit :

| Select System Files | × |
|---|---|
| | Select System Files |
| | Would you like to install system DLL files to your \Windows\Sys directory? If you are running MPLAB installed on a common network, you may not be allowed to write files to this directory. If you do not install them in the \Windows\Sys directory, they will be put in the same directory as MPLAB. |
| , in the second s | C Install files to \Windows\Sys |
| | Install files to MPLAB install directory |
| | |
| | Kext > Cancel |

Vous arrivez enfin à la dernière fenêtre : celle qui vous dit que maintenant tout est prêt pour que l'installation démarre de vrai !

Cliquez sur Next : c'est parti.



Une fois l'installation terminée (*Installation Completed !*), vous pouvez ignorer le *Readme* proposé. Cliquez sur MPLAB.EXE pour créer un raccourci.

Vous pouvez démarrer.

L'environnement de développement de MPLAB est à vous !



Comment désinstaller MPLAB

Le fichier INSTALL.LOG enregistre les noms de tous les fichiers copiés au cours de la procédure d'installation.

Le jour que l'on souhaite désinstaller MPLAB, la procédure de désinstallation utilise INSTALL.LOG pour savoir quels sont les fichiers à supprimer.

Pour la désinstallation, il suffit de lancer UNWISE.EXE se trouvant dans le répertoire de MPLAB.

Comment démarrer avec MPLAB

Tout d'abord une constatation : l'écran de démarrage de MPLAB ressemble à la plupart des écrans Windows, en ce sens qu'il comporte - en haut - la barre des menus et la barre d'outils, et - en bas - une barre d'état informant de la façon dont le système est configuré.

En cliquant sur la toute première icône de la barre d'outils (icône en couleur représentant trois fiches en perspective : bleu, rouge, jaune):



vous pouvez déjà constater la présence d'autres barres d'outils (il y en a quatre) :



Comme l'on sait que dans tout logiciel bien conçu plus il y a d'outils disponibles et plus le travail est facile, la première évidence est que MPLAB est un logiciel sympathique.

A ajouter aussi que MPLAB est un logiciel sûr, et qu'il possède plusieurs points d'entrée.

MPLAB est un logiciel sûr : vous pouvez l'utiliser sans vous faire le moindre souci, car il ne vous laissera jamais modifier ou supprimer quoi que ce soit présentant le moindre danger pour l'intégrité des fichiers sans vous mettre en garde et vous demander de confirmer la commande;

MPLAB possède plusieurs points d'entrée : en ce sens que vous pouvez effectuer la même tâche en empruntant des chemins différents, sans obligatoirement obéir à un mode d'emploi préétabli. Il suffit, la plupart des fois, d'adapter et de croiser les grilles écran ouvertes sur le bureau.

Eu égard à cette souplesse, c'est-à-dire : étant donné qu'il est possible d'emprunter différents chemins pour accomplir la même tâche, la manière d'écrire un programme et de l'assembler que je vais vous indiquer, est seulement l'une des manières possibles. Au fur et à mesure que vous vous familiariserez avec MPLAB, vous en découvrirez d'autres.

Première étape : création du projet

Créer un projet

Pour pouvoir aboutir à un programme, MPLAB a besoin de naviguer entre plusieurs fichiers.

Pour cette raison, avant de vous autoriser à écrire un programme, MPLAB vous oblige à <u>créer un projet</u>.

Créer un projet revient à définir entre autres:

1) le nom que va porter le programme ;

2) les fichiers que MPLAB devra utiliser au cours du développement.

Un projet comporte un nom, avec extension .pjt

Convenons d'appeler notre projet *phase1* Le nom du projet sera donc : *phase1.pjt*

Comment créer un projet

Lancez MPLAB, puis allez dans :

Project ↓ New Project

| <u>P</u> roject | <u>E</u> dit | <u>D</u> ebug | Option |
|-----------------|--------------|---------------|--------|
| <u>N</u> ew | Projec | ot | |
| <u>O</u> pe | n Proje | ect Cti | l+F2 |
| <u>C</u> los | e Proje | ect | |
| <u>S</u> av | e Proje | :ct | |
| <u>E</u> dit | Projec | t Ct | 1+F3 |
| Mak | e Proja | ect | F10 |
| <u>B</u> uild | I All | Ctrl | F10 |
| Build | i Node | e Alt- | F10 |
| <u>I</u> nsta | all Lang | guage To | ol |

Une fenêtre s'ouvre (New Project) :

| New Project | | × |
|-----------------------------|-------------------|--------------|
| File <u>N</u> ame: | Directories: | OK |
| *.pit | c:\progra~1\mplab | Cancel |
| | 🚖 c:\ | A. |
| | 🔄 progra~1 | <u>H</u> elp |
| | mplab . | |
| | example | |
| | i cemplace | - |
| v | | <u> </u> |
| List Files of <u>Type</u> : | Dri <u>v</u> es: | |
| Project Files (*.pjt) | 🗇 c: win95b | - |

MPLAB vous rappelle que le nom du projet doit impérativement être suivi de l'extension **.pjt**

Dans le champ *File Name* entrez le nom que vous voulez donner à votre projet (nous avons convenu de l'appeler *phase1*) suivi de *.pjt* (*phase1.pjt*).

| New Project | | × |
|--|--|------------------------------|
| File <u>M</u> ame: phase1 .pjt | Directories: c:\progra~1\mplab c:\ progra~1 mplab example template | OK Cancel <u>H</u> elp |
| List Files of <u>Type</u> : Project Files (*.pjt) | Drives: | |

Puis cliquez sur OK.

Cette manœuvre ouvre une autre fenêtre (Edit Project):

| Edit Project | X |
|----------------------------------|-----------------|
| Project Target Filename | OK |
| phase1.hex | |
| Include Path | Cancel |
| | |
| Library Path | Неір |
| | |
| Linker Script Path | - |
| | |
| Development Mode: Editor Only16F | 84 Change |
| Language Tool Suite: Microchip | _ |
| Project Files | |
| phase1 [.hex] | Add Node |
| | Copy Node |
| | Delete Node |
| | Build Node |
| | |
| | Node Properties |

Dans cette fenêtre :

a) le *Target Filename* (le nom du fichier cible, c'est à dire celui auquel vous voulez aboutir) doit correspondre à celui que vous voulez créer, portant donc l'extension *.hex* (*phase1.hex*);

b) le *Development Mode* se rapporte bien au MPLAB-SIM Simulator PIC 16F84 et non pas à un autre mode de développement (sinon cliquez sur la touche *Change* et modifiez en conséquence; puis cliquez en répondant toujours OK);

b) le Language Tool Suite doit être celui de Microchip.

Après ces vérifications, cliquez sur la touche OK.

Dès lors, le projet est créé.

Il contient les paramètres qui lui sont nécessaires, et vous pourrez vous y référer par la suite.

Deuxième étape : écriture du programme source

Pour taper votre programme, allez dans :

File ↓ New



C'est à partir de maintenant que l'écriture à proprement parler des instructions du programme peut démarrer.

La fenêtre d'accueil est Untitled1 :



C'est celle de l'éditeur de texte de MPLAB.

Elle est prête à recevoir les lignes de votre programme.

Agrandissez-la à plein écran, et commencez la frappe.

Pour séparer les champs (labels, instructions, opérandes et commentaires) servez-vous de la touche de tabulation.

N'oubliez pas d'insérer des ; (point-virgule) avant les commentaires.

Lorsque toutes les lignes du programme ont été saisies, allez dans :



Une fenêtre s'ouvre (Save File As):

| Save File As | | × |
|---|---|--|
| File <u>N</u> ame: | Directories: c:\progra~1\mplab c:\ c:\ progra~1 mplab c: example template | OK Cancel <u>H</u> elp □ <t< th=""></t<> |
| List Files of <u>Type</u> : Source files (*.c;*.asm) | Dri <u>v</u> es: | <u> </u> |

Dans le champ *File Name* demandant le nom sous lequel vous voulez sauvegarder le fichier, tapez le nom du programme - qui doit être impérativement le même que celui du projet, suivi de l'extension **.asm** ainsi que MPLAB vous le rappelle (**phase1.asm**) - ou allez le chercher dans la liste déroulante, au moyen de l'ascenseur. Puis terminez en cliquant sur la touche *OK*.

| Save File As | | × |
|---|--|---|
| File <u>N</u> ame: phase1 asm idasm16.asm idasm17.asm mul8x8.asm readme.asm sample.asm sampobj.asm tutor.asm | Directories: c:\progra~1\mplab c:\ morectain progra~1 mplab morectain example morectain template | OK Cancel <u>H</u> elp <u>UNIX format</u> V Keep <u>b</u> ackup |
| List Files of <u>T</u> ype: Source files (*.c;*.asm) ▼ | Dri <u>v</u> es: C: win95b | R <u>é</u> seau ▼ |

Dès lors, ce fichier à extension **.asm** contient le « *programme source* », que l'on appelle plus couramment « *fichier source* », ou tout simplement « *source* ».

Ce fichier est sauvegardé.

Notez ou retenez bien son nom (*phase1.asm*), car vous aurez l'occasion d'y revenir.

Troisième étape: conversion du programme source en programme objet

Le but de cette opération est de convertir le *fichier source* (celui dont l'extension est *.asm*) en *fichier objet*, c'est à dire en un fichier ayant le même nom que celui du fichier source, mais avec extension *.hex*

Autrement dit : convertir *phase1.asm* en *phase1.hex*

Pour cela, allez dans :

Project ↓ Edit Project

| <u>P</u> roject | <u>E</u> dit | <u>D</u> ebug | PICSTART Plus |
|-----------------|--------------|---------------|---------------|
| <u>N</u> ew | Proje | ot | |
| <u>0</u> pe | n Proje | ect | Ctrl+F2 |
| <u>C</u> los | e Proje | ect | |
| <u>S</u> ave | e Proje | ect | |
| <u>E</u> dit | Projec | t | Ctrl+F3 |
| <u>M</u> ak | e Proje | ect | F10 |
| <u>B</u> uild | I All | | Ctrl+F10 |
| Build | l Node | ; | Alt+F10 |
| | | | |
| <u>I</u> nsta | all Lang | guage To | ol |

Une fenêtre s'ouvre :

| Edit Project | × |
|---|-----------------|
| Project | ОК |
| l arget Filename | |
| phase I. hex | Cancel |
| Include Path | |
| | |
| Library Path | |
| | |
| Linker Script Path | |
| | |
| Development Mode: MPLAB-SIM16 Language Tool Suite: Microchip | F84 Change |
| phase1 [.hex] | Add Node |
| | |
| | Copy Node |
| | Delete Node |
| | Build Node |
| | Node Properties |

Vous connaissez cette fenêtre pour l'avoir vue lors de l'ouverture du projet.

Assurez-vous d'abord que le nom du fichier figurant dans la zone Target Filename correspond à celui qui vous intéresse (*phase1.hex*). Le cas échéant, modifiez en cliquant sur la touche *Change*.

Cette fenêtre comporte déjà le nom du fichier cible (*phase1.hex*). Mais, à ce stade, MPLAB demande qu'un *nœud* soit créé entre le fichier cible et le fichier source.

En bas, à droite, une touche est disponible : Add Node.

Un clic sur cette touche, ouvre la fenêtre Add Node.

| Add Node | | ? × |
|---|---|-------------------------------|
| Nom de fichier : phase1.asm 12c508a.asm 12c508a2.asm demo877.asm idasm17.asm mul8x8.asm phase1.asm readme.asm | Dossiers : c:\progra~1\mplab c:\ c:\ mplab c: mplab c: template | OK Annuler <u>A</u> ide |
| <u>Types de fichiers :</u> Source files (*.c;*.asm) | Lecteurs : c: win95b | • |

Au moyen de l'ascenseur, cherchez-y le fichier de même nom à extension **.***asm* (*phase1.asm*) et cliquez sur ce fichier pour que son nom s'inscrive dans le champ *Nom de fichier*.

Puis cliquez sur OK.

| Edit Project | × |
|----------------------------------|-----------------|
| Project | or |
| Target Filename | |
| phase1.hex | |
| Include Path | Cancel |
| | |
| Library Path | Help |
| | |
| Linker Script Path | |
| | |
| | |
| Development Mode: Editor Only16F | 84 Change |
| Language Tool Suite: Microchip | ~ |
| - Project Files | |
| | |
| phase1 [.nex] phase1 [.asm] | Add Node |
| | Copy Node |
| | |
| | Delete Node |
| | Build Node |
| | |
| | Node Properties |
| | |

La fenêtre *Edit Project* s'ouvre une nouvelle fois :

Mais cette fois, dans la zone *Project Files*, figurent les noms de <u>deux</u> fichiers :

- celui à extension .hex
- et celui à extension .asm

Cliquez sur OK pour verrouiller le *nœud* entre MPLAB et MPASM (un autre utilitaire de l'environnement intégré) et lier ainsi ces deux fichiers.

| Puis allez dans : | | | | | |
|---|--|--|--|--|--|
| Project ↓ Build All | | | | | |
| <u>Project</u> <u>E</u> dit <u>D</u> ebug PIC <u>S</u> T/ | | | | | |
| New Project | | | | | |
| Open Project Ctrl+F2 | | | | | |
| Close Project | | | | | |
| Save Project | | | | | |
| <u>E</u> dit Project Ctrl+F3 | | | | | |
| Make Project F10 | | | | | |
| Build All Ctrl+F10 | | | | | |
| Build Node Alt+F10 | | | | | |
| Install Language Tool | | | | | |

Dès lors, la compilation démarre: les instructions du fichier *phase1.asm* écrites en langage assembleur (celui utilisant les mnémoniques) sont transformées en une suite de 0 et de 1 (seul langage compréhensible par le 16F84) pour constituer le fichier *phase1.hex*

Il arrive qu'une compilation soit bonne du premier coup.

A supposer que tel soit le cas, vous devriez voir s'ouvrir la fenêtre suivante :



Par le message : « *Build completed successfully* » (la compilation s'est achevée avec succès), cette fenêtre vous informe que l'assemblage de votre fichier à extension **.asm** (*phase1.asm*) a aboutit à un fichier à extension **.hex** (*phase1.hex*)

Félicitations ! Cela veut dire que dans votre programme source vous n'avez commis aucune erreur.

Vous pourriez - tout au plus - voir apparaître des « *messages* » ou des « *warnings* », c'est à dire des avertissements, des mises en garde.

Bien que les *messages* et les *warnings* n'empêchent pas le déroulement d'un programme, il est tout de même bon de leur jeter un coup d'œil et d'en comprendre les raisons.

Il suffit de demander à MPLAB de numéroter les lignes de votre programme (en cliquant sur l'icône dans l'une des barres des outils, en haut de l'écran)



Puis, de vous reporter à la ligne mentionnée par chaque *message* ou *warning* et de l'examiner attentivement. A cette ligne le compilateur assembleur de MPASM n'a pas trouvé une erreur à proprement parler, mais une sorte d'ambiguïté, une anomalie, et il vous prévient que cela pourrait conduire à une exécution défaillante de votre programme.

Quatrième étape: correction des erreurs

Si votre programme contient des erreurs, la compilation n'aboutit

Avec le message *Build failed* (construction ratée), vous verriez apparaître des messages mentionnant la ou les erreurs.

Dans ce cas, il faut impérativement revenir au fichier à extension *.asm* et corriger les erreurs.

Puis, relancer la compilation depuis le début par :

pas.

| <u>P</u> roject | <u>E</u> dit | <u>D</u> ebug | PICSTART Plus | | | |
|---------------------------------------|--------------|---------------|---------------|--|--|--|
| <u>N</u> ew | New Project | | | | | |
| <u>0</u> pe | n Proje | Ctrl+F2 | | | | |
| <u>C</u> lose Project | | | | | | |
| <u>S</u> ave Project | | | | | | |
| <u>E</u> dit Project | | | Ctrl+F3 | | | |
| <u>M</u> ake Project | | | F10 | | | |
| <u>B</u> uild | I A I | Ctrl+F10 | | | | |
| Build | Alt+F10 | | | | | |
| Install Language Tool | | | | | | |
| <u>1</u> c:\progra~1\mplab\phase1.pjt | | | | | | |

Vous revenez ainsi à la fenêtre Build Results.

Cela autant de fois que nécessaire, jusqu'à lire le message : « *Build completed successfully »*.

Cinquième étape: simulation du programme

Disons-le tout de suite : cette étape est facultative, car elle concerne la *simulation*.

La *simulation* est une sorte de vérification virtuelle permettant de voir à l'écran ce qui se passe lorsque les instructions sont exécutées par le microcontrôleur.

On peut la faire de différentes façons :

- en mode pas à pas
- en mode step over
- en mode exécution automatique
- avec ouverture de fenêtres.

Mode pas à pas

Pour simuler un programme en mode pas à pas, appelez à l'écran le fichier à extension **.asm**



 $\begin{array}{c} Debug\\ \downarrow\\ Run\\ \downarrow\\ Reset\end{array}$

(ou tapez le raccourci clavier F6) :

| <u>D</u> ebug | PRO <u>M</u> ATE | <u>O</u> ptions | <u>T</u> ools | \underline{W} | indow | <u>H</u> elp | |
|---|--|--------------------------------|----------------------------------|-----------------|--------------|----------------|---------------------|
| <u>R</u> un | | | | × | <u>B</u> un | | F9 |
| <u>E</u> xecu | ite | | | ⊁ | R <u>e</u> s | et | F6 |
| Simula | ator Stimulus | | | ⊁ | <u>H</u> alt | | F5 |
| Cente | r Debug <u>L</u> ocat | ion | | | Halt Anim | Trace nate | Shift+F5 Ctrl+F9 |
| Break | Settings | | F2 | | Step |) | F7 |
| Trace | Settings | | | | Step |) <u>O</u> ver | F8 |
| Trigge | er In/Out Settin | igs | | | Upd | ate All Re | egisters |
| Trigge | er Output Point | S | | | <u>C</u> hai | nge Progr | ram Counter |
| Clear. | All P <u>o</u> ints | | | | | | |
| Comp Code | le <u>x</u> Trigger Set Co <u>v</u> erage | tings | | | | | |
| <u>C</u> lear Syster <u>P</u> owe | Program Memo m Reset r-On-Reset | ory Ctrl+9 Ctrl+9 Ctrl+9 | Shift+F2 Shift+F3 Shift+F5 | | | | |

Cette manœuvre effectue un *Reset* logiciel du programme qu'on veut simuler.

En effet, le Program Counter est mis à zéro (adresse correspondant au vecteur Reset du 16F84).

La ligne de programme se trouvant à cette adresse passe en surbrillance, tandis que dans la barre d'état (située en bas de l'écran) PC prend la valeur 00.

Ensuite allez dans :

```
\begin{array}{c} Debug\\ \downarrow\\ Run\\ \downarrow\\ Step \end{array}
```



En tapant sur le raccourci clavier *F7* le PC (Program Counter) avance d'un pas à la fois, ainsi que vous pouvez le constater en lisant la valeur du Program Counter dans la barre d'état.

C'est l'exécution en mode pas à pas.

Vous pouvez aussi vous servir de la souris, en cliquant sur l'icône :



(Deux empreintes de pas: Pas à pas : F7)

Icône située dans la barre des outils, en haut de l'écran.

Devant une boucle, vous pourriez stopper la simulation soit en tapant sur le raccourci clavier *F5*, soit en cliquant sur l'icône du feu rouge :



(Feu rouge : *Halt* : F5)

Mode Step Over

Lorsque dans le programme il y a des boucles, l'exécution peut prendre beaucoup de temps.

Dans ce cas, il existe la possibilité de demander au simulateur d'ignorer les boucles.

C'est le mode Step Over.

Pour ce type d'exécution il faut choisir le chemin :

Debug ↓ Run ↓ Step Over

| Debug PRO MATE Op | tions | <u>T</u> ools | \underline{W} | (indow <u>H</u> elp | |
|----------------------------------|---------|----------------------|-----------------|---------------------|------------|
| <u>R</u> un | | | × | <u>R</u> un | F9 |
| <u>E</u> xecute | | | ۲ | R <u>e</u> set | F6 |
| Simulator Stimulus | | | ۲ | <u>H</u> alt | F5 |
| Center Debug Location | | | _ | Halt Trace | Shift+F5 |
| | | | | <u>A</u> nimate | Ctrl+F9 |
| <u>B</u> reak Settings | | F2 | | <u>S</u> tep | F7 |
| <u>T</u> race Settings | | | | Step <u>O</u> ver | F8 |
| Trigger In/Out Settings | | Update All Registers | | | |
| Trigger Output Points | | | | <u>Change</u> Progr | am Counter |
| Clear All P <u>o</u> ints | | | | | |
| Comple <u>x</u> Trigger Settings | | | | | |
| Code Co <u>v</u> erage | | | | | |
| <u>C</u> lear Program Memory | Ctrl+Sł | hift+F2 | | | |
| System Reset | Ctrl+Sł | hift+F3 | | | |
| <u>P</u> ower-On-Reset | Ctrl+Sł | hift+F5 | | | |

Pour faire avancer le programme dans ce mode, tapez sur le raccourci clavier

F8 ou cliquez de façon répétée sur l'icône montrant deux



empreintes de pas entrecoupées :

Le programme s'exécute alors selon un mode qui - *en apparence* - ressemble au mode pas à pas, mais qui *en réalité ignore toutes les boucles*. De ce fait, le temps de son exécution est considérablement plus court.

L'arrêt de la simulation peut se faire à tout moment, soit en cliquant sur l'icône du feu rouge, soit en tapant sur la touche F6.

L'on revient alors au début du programme.

Mode exécution automatique

Au lieu de faire du *pas à pas*, on peut faire exécuter le programme en entier, automatiquement, sans avoir besoin ni de cliquer sur une quelconque icône, ni de taper sur une quelconque touche de raccourci.

Il suffit d'aller dans : $Debug \rightarrow Run \rightarrow Animate$:

| Debug PRO MATE Op | tions <u>T</u> ools | <u>W</u> indow | <u>H</u> elp | |
|--|---|----------------|-----------------|----------|
| <u>B</u> un | | 🕨 <u>R</u> ur | า | F9 |
| <u>E</u> xecute | | ▶ R <u>e</u> : | set | F6 |
| Simulator Stimulus | | ▶ <u>H</u> al | t | F5 |
| Center Debug Location | | Hal | t Trace | Shift+F5 |
| Center Debug Location | | Anii | mate | Ctrl+F9 |
| <u>B</u> reak Settings | F2 | <u>S</u> te | p | F7 |
| <u>T</u> race Settings | | Ste | p <u>O</u> ver | F8 |
| Trigger In/Out Settings | | <u>U</u> pe | date All Regist | ters |
| Trigger Output Points | | <u>C</u> ha | ange Program | Counter |
| Clear All P <u>o</u> ints | | | | |
| Comple <u>x</u> Trigger Setting: Code Co <u>v</u> erage | s | | | |
| <u>C</u> lear Program Memory System Reset <u>P</u> ower-On-Reset | Ctrl+Shift+F2 Ctrl+Shift+F3 Ctrl+Shift+F5 | | | |

L'arrêt de la simulation peut se faire à tout moment, soit en cliquant sur l'icône du feu rouge



soit en tapant sur la touche *F6*. Dans les deux cas l'on revient au début du programme.

Ouverture de fenêtres

En complément à la simulation, vous pourriez avoir besoin de voir de plus près comment évolue une variable, une adresse, le contenu d'un compteur ou d'une quelconque zone mémoire.

Pour ce genre d'observation, MPLAB met à disposition un grand nombre de fenêtres (*Windows*) spécifiques.

Il suffit d'aller soit dans :

Window



Cette fenêtre permet d'ouvrir les trois fenêtres principales : celle relative à la pile (*Stack Window*), celle relative à la RAM (*File Register*

Window), ou celle relative aux registres à usage spécial (Special Function Register Window). Sinon on peut aller dans :

Window \downarrow Watch Windows \downarrow New Watch Window

| Window Help | |
|--|--|
| Program Memory Irace Memory EEPROM Memory Absolute Listing Map File | |
| <u>S</u> tack File Registers Special F <u>u</u> nction Registers S <u>h</u> ow Symbol List Ctrl+F8 Stopwatch | |
| Pr <u>o</u> ject | |
| Watch Windows | New Watch Window |
| <u>M</u> odify Tile <u>H</u> orizontal Tile <u>V</u> ertical | Load Watch Window Add to Active Watch Edit Active Watch Save Active Watch |
| <u>C</u> ascade Iconize All Arrange Icons | |

| dd Watch Symbol | × |
|-----------------------------|----------------|
| Symbol: | |
| | Add |
| eeadr eecon1 | <u>D</u> elete |
| eedata fsr | <u>C</u> lose |
| intcon option_reg ocl | Properties |
| pclath porta | <u>H</u> elp |

Cette manœuvre ouvre une fenêtre comportant tous les noms des registres et variables utilisé dans votre programme :

Il suffit de cliquer sur ce qui vous intéresse, pour aussitôt voir s'ouvrir la fenêtre correspondante, dans laquelle toute variable subissant une modification au cours de l'exécution du programme, change de couleur, attirant votre attention.

Il est même possible de suivre l'évolution de plusieurs variables et d'avoir sous vos yeux, constamment ouvertes, <u>plusieurs fenêtres</u>.

Pour cela, au moyen de l'ascenseur, sélectionnez la ligne correspondant à ce que vous voulez observer, et terminez - après avoir éventuellement sélectionné d'autres registres encore - en cliquant sur le bouton *Add*, puis sur le bouton *Close*.

Dès lors, plusieurs fenêtres se positionnent sur l'écran, rangée l'une derrière l'autre, comme les cartes dans la main d'un joueur, que vous pouvez rendre actives à souhait.

Faites ensuite avancer le programme en mode pas à pas (F7).