

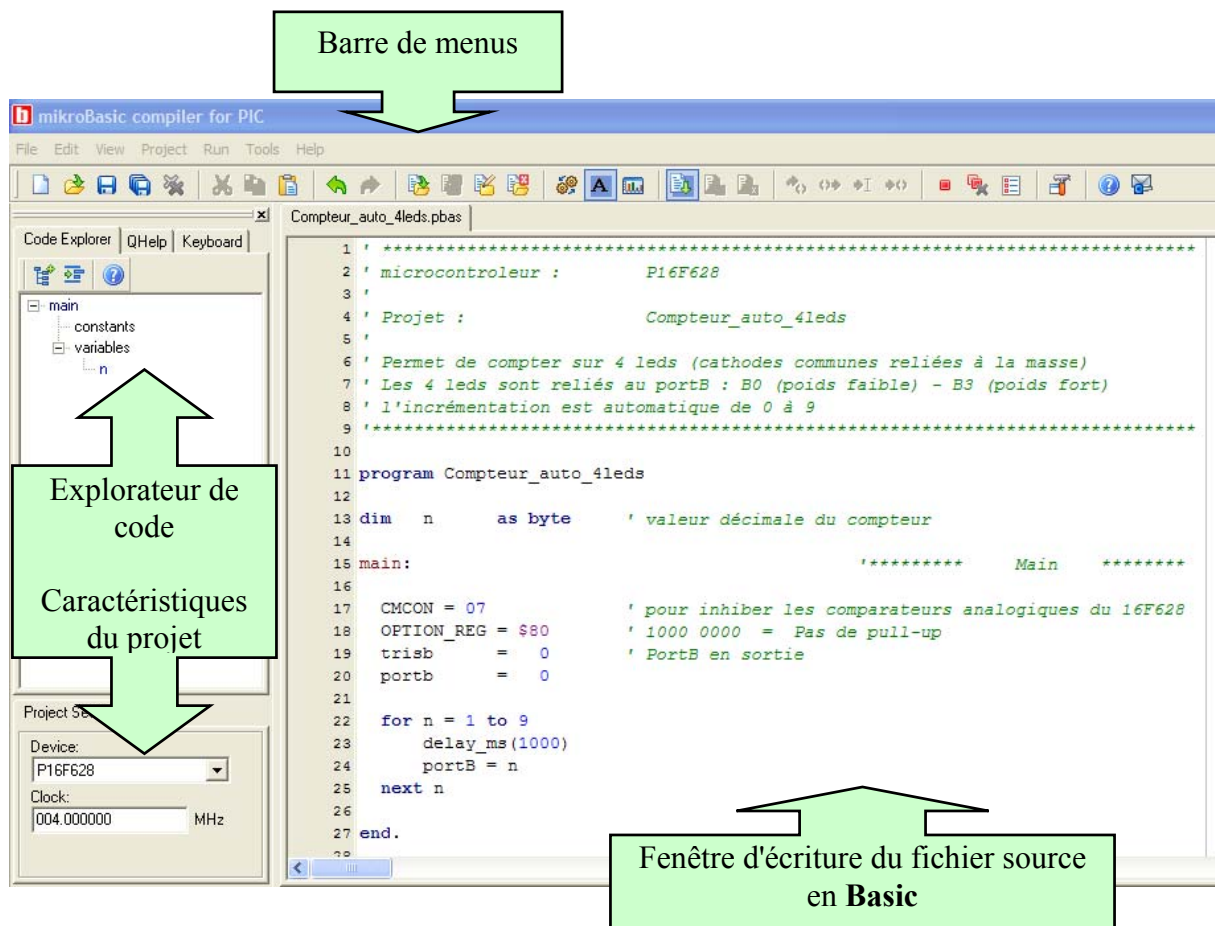


1 Développement de l'application

Le logiciel MikroBasic possède un environnement de développement intégré (IDE).

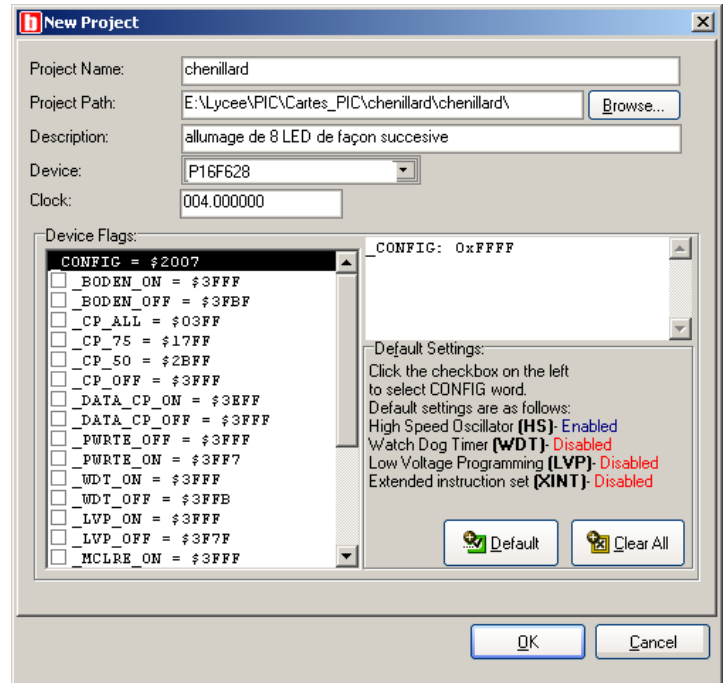
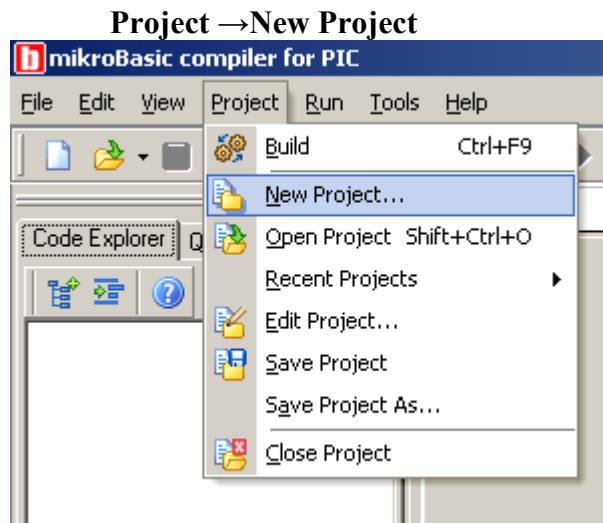
Il est constitué entre autres d'un éditeur et d'un compilateur et de toutes commandes nécessaires à la compilation (création du fichier assembleur) et à l'assemblage (création du fichier objet en hexadécimal).

Toute erreur de syntaxe est signalée par le compilateur et stoppe la compilation. Une aide contextuelle est disponible (voir paragraphe ci-après). Une fois les erreurs supprimées, le logiciel assemble et crée le fichier objet qui sera transféré ensuite dans le PIC **lire la notice « Utilisation du programmeur »**.



2 Création d'un projet (fichier.pbp)

. Il faut absolument un projet pour un programme.



Renseigner les paramètres du projet :

- « project Name » nom du projet
- « project path » emplacement
- « description » commentaires descriptifs
- « device » nom du circuit PIC
- « clock » fréquence de l'oscillateur
- « device flag » éventuellement son type

3 Rédaction du programme source en Basic dans la fenêtre principale.

3.1 Rappels :

Les opérateurs arithmétiques élémentaires :

- + : addition
- - : soustraction
- * multiplication
- / : division
- div : exécute la division et restitue la partie entière
- mod : exécute la division et restitue le reste de la division

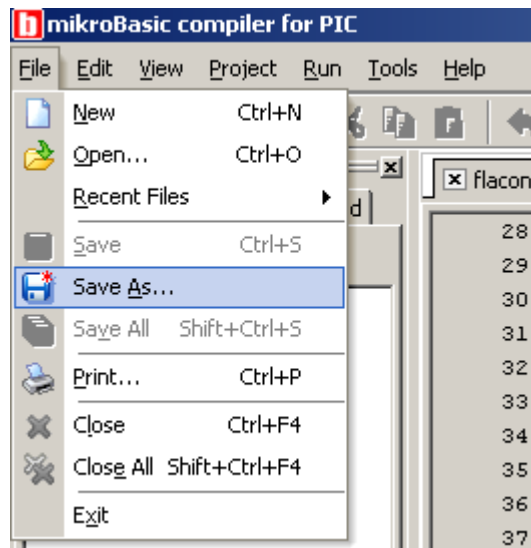
Les opérateurs logiques élémentaires :

- & : (AND)
- ! : (OR)
- ^ : (XOR)
- not : inverseur
- >> x : où x indique le nombre de décalages à droite successifs dans un mot binaire
- << x : où x indique le nombre de décalages à gauche successifs dans un mot binaire

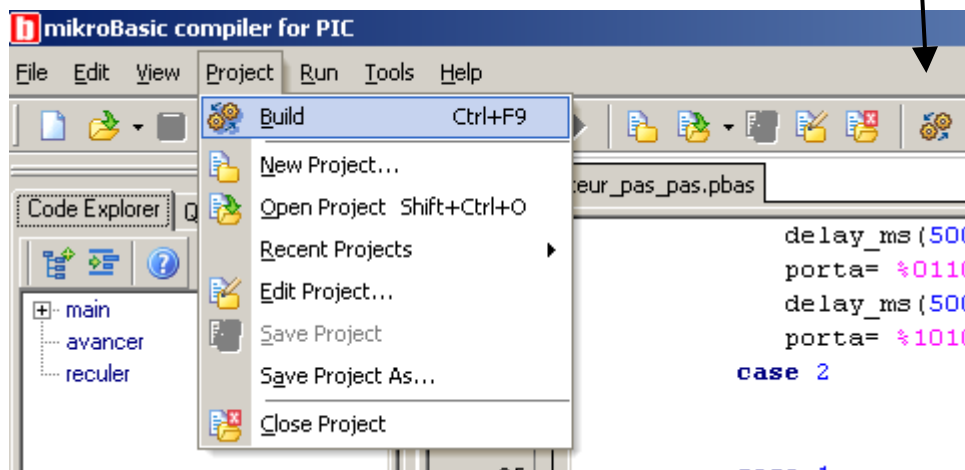
Les principales instructions :

Syntaxe	Commentaires	Exemple
program ... end.	Program précise à la première ligne le nom du fichier en basic. La fin du programme est repérée par end avec un point	program essai end.
dim as byte (integer,...)	Permet de déclarer les variables utilisées dans le programme en précisant leur type : <ul style="list-style-type: none"> • byte : octet (8 bits) • integer : 16 bits • word : nom alphanumérique 	dim i, j, k as byte dim counter as word dim tab as longint[100]
const	Déclare une donnée constante de type numérique ou caractère	const MIN = 1000 const SWITCH = "n" const vals as byte[12] = (31,12,17)
symbol	Déclaration d'alias	symbol t1s = delay_ms(1000) symbol led = PortB.3
sub ... end sub	Déclaration des sous programmes (procédures ou fonctions) pour une meilleure structure du programme. S'écrivent avant le programme principal	sub procedure calcul n = a * (b +3) end sub
main :	Étiquette de début de programme principal, toujours suivi de deux points	main :
if ... then ... (else) ... end if	Structure de contrôle pour réaliser un test à l'aide d'une expression booléenne. Exécute un traitement si condition vraie (ou éventuellement un autre si faux)	if plus = 1 then i = i+1 end if
while wend	Pour répéter un traitement tant qu'une condition est vraie. Rem : s'utilise aussi pour créer une boucle sans fin	while i < 4 i = i+1 wend while true ... wend
Select case case 0 case 3 case else end select	Suivant que la variable vaut 0 ou 3 ou autre → faire	Select case j case 0 portB=%00001111 case 3 portB=%01111111 case else portB=0 end select
for ... to ... next	Permet de réaliser une itération à l'aide d'une variable	for i = 0 to 4 portB = i next i
delay_us (n) delay_ms (m)	Fonctions prêtes à l'emploi pour réaliser une temporisation de n microsecondes ou m millisecondes	delay_ms (500)
goto	Renvoi incondtionnel à une ligne de programme définie par une étiquette. L'étiquette est indiquée par son nom suivi de deux points (:) A éviter autant que possible.	goto main

3.2 Enregistrement du programme source File → Save (fichier.pbas)



3.3 Compilation (puis assemblage) par le menu Project → Build ou par l'icône ici

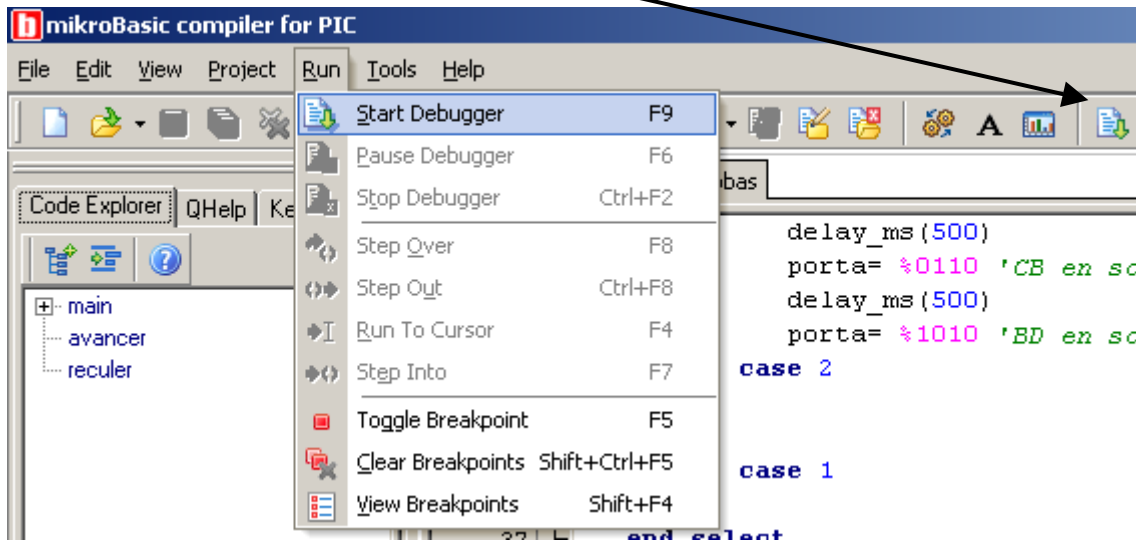


3.4 Correction des éventuelles erreurs signalées par le compilateur et recompilation

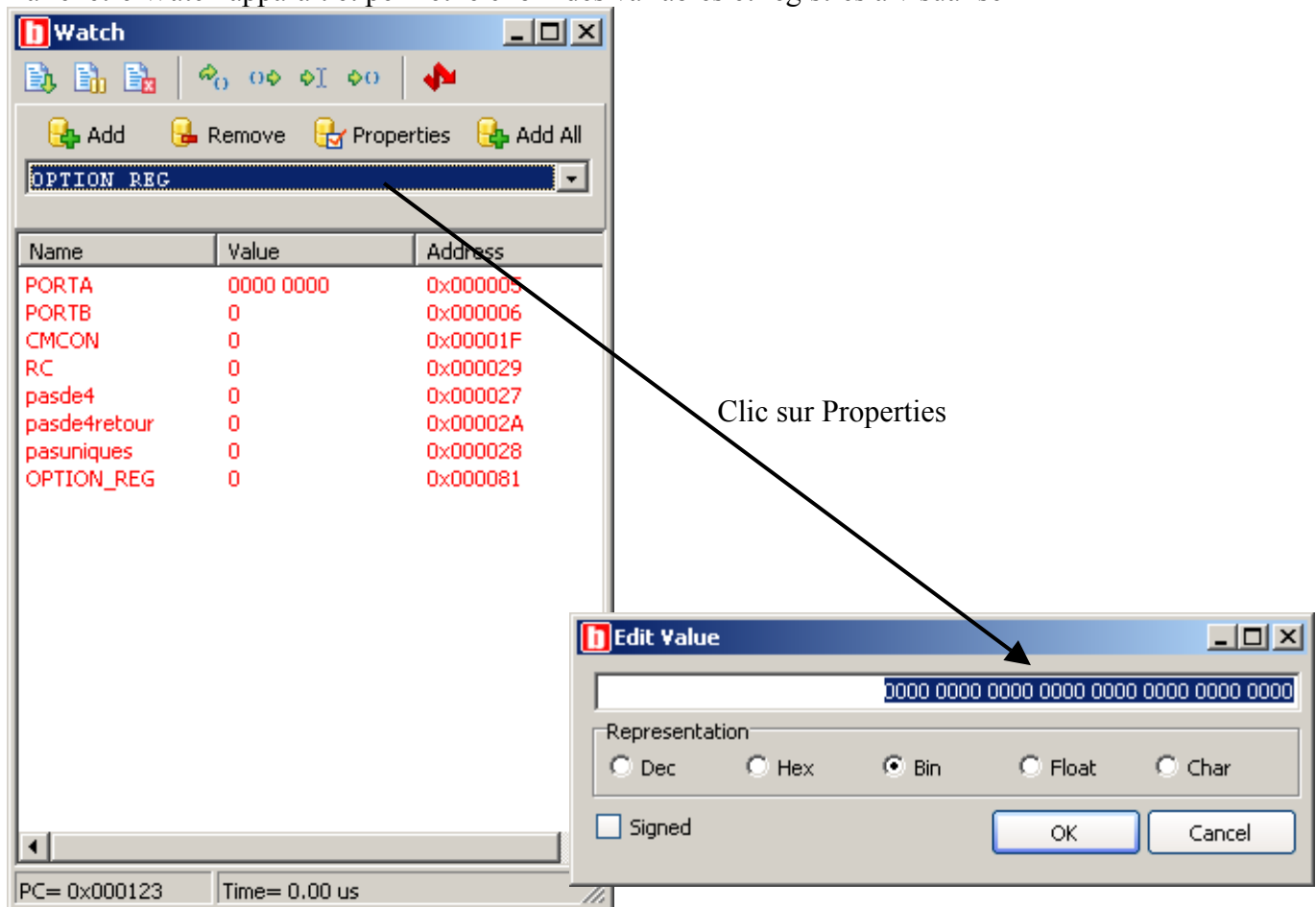
Après exécution de ces étapes, nous disposons du fichier **.hex** à implanter dans la mémoire du microcontrôleur PIC. Avant de programmer le circuit, il est conseillé de contrôler le bon fonctionnement du programme par une simulation pas à pas et une visualisation des états des différentes variables et registres internes

3.5 Debugger

Menu : Run → Start Debugger ou l'icône ou F9



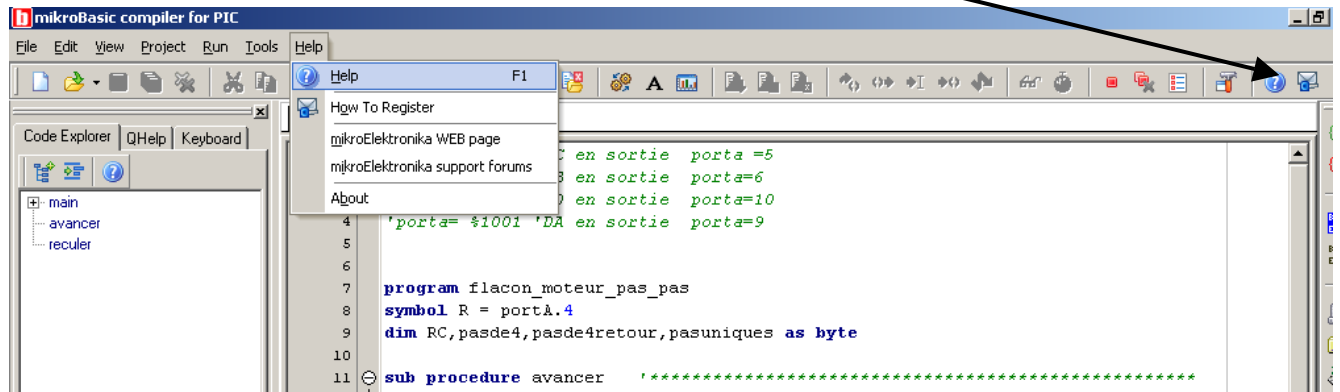
La fenêtre Watch apparaît et permet le choix des variables et registres à visualiser



4 Utilisation de l'aide

MikroBasic intègre également une application d'aide à la rédaction du programme source :

Menu : Help→Help ou la touche de fonction F1 ou l'icône



La fenêtre d'aide apparaît

