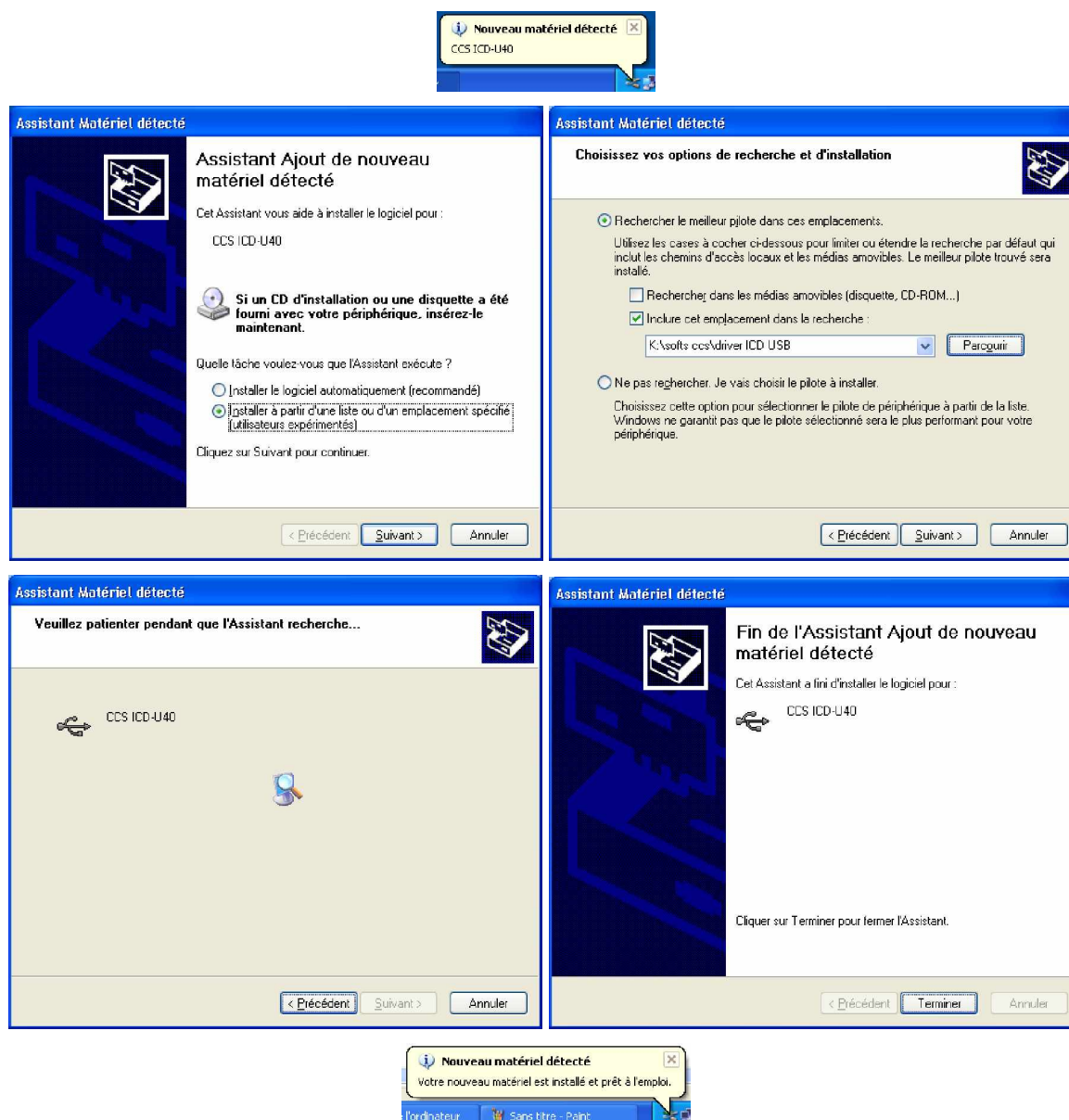


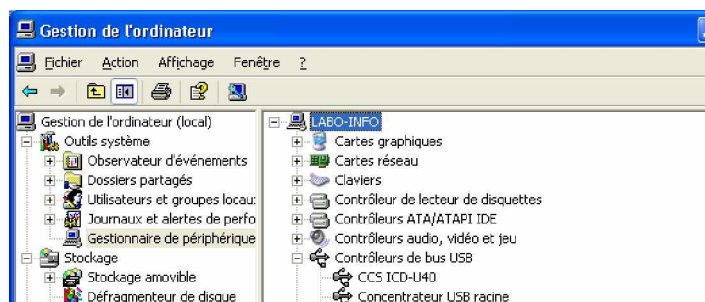
1/ Mise en place de l'environnement ICD :

On dispose d'un PC (ici sous Windows XP) et des drivers pour l'ICD en USB.

A la connexion de l'ICD l'assistant de détection demande le chemin des drivers et il faut l'orienter vers le répertoire correspondant (driver ICD USB) :



On peut vérifier la présence du driver dans Windows :



Voilà Windows sait maintenant accéder à l'ICD.

On peut maintenant installer le gestionnaire individuel de l'ICD qui permet de tester les connexions mais aussi de programmer un PIC avec un fichier code machine (*.hex) puis le lancer.

Développement en C avec CCS et ICD.

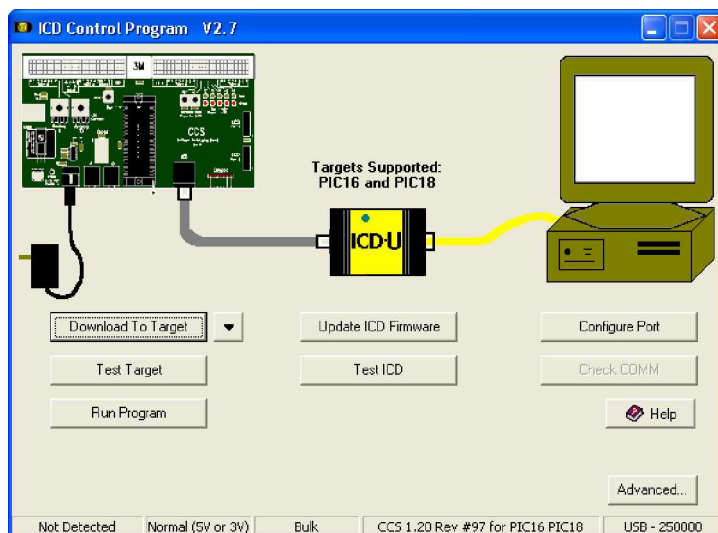


On lance le programme : icdu.exe et une fois l'installation terminée on obtient l'icône du bureau :

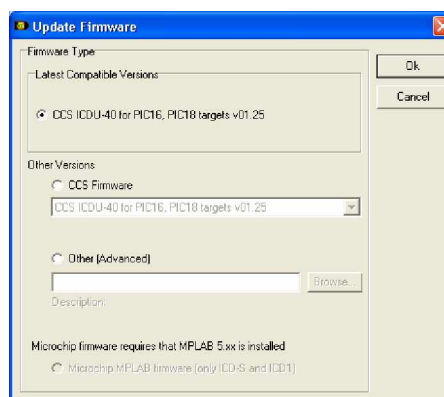


On peut lancer le programme :

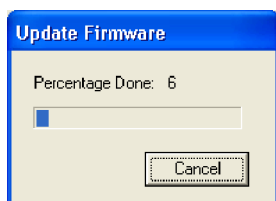
Cette fenêtre indique que logiciel contenu dans l'ICD est une version plus ancienne que celle du gestionnaire Windows, on accepte la mise à jour.



La version du gestionnaire est ici V2.7 alors que le firmware est CCS 1.20

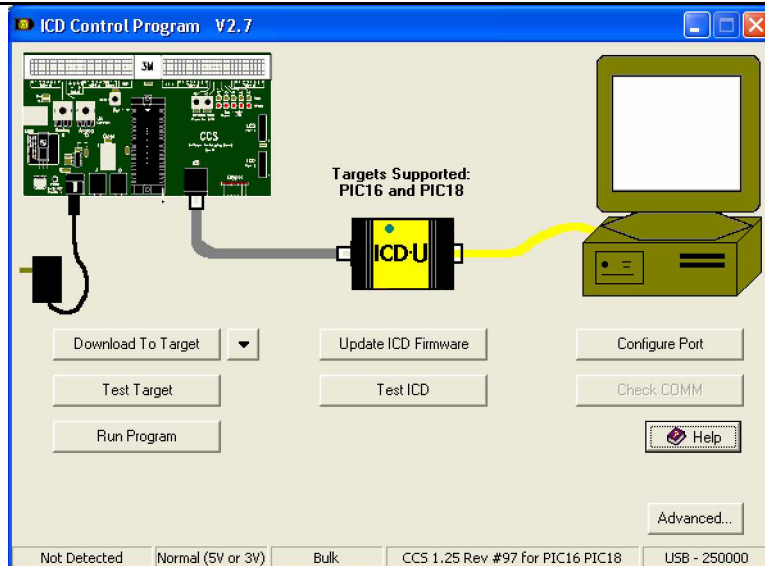


Le bouton Update ICD Firmware permet la mise à jour :

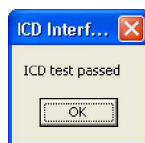


On accepte :

Développement en C avec CCS et ICD.



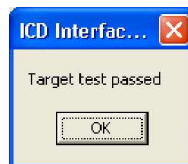
On obtient :



On peut alors tester l'ICD avec Test ICD :

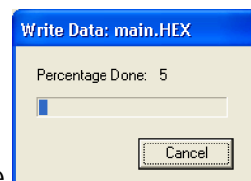


La cible (target) n'étant pas connecté, on obtient :

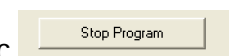


Si on connecte une carte cible, on obtient :

Nous pouvons alors utiliser les boutons :



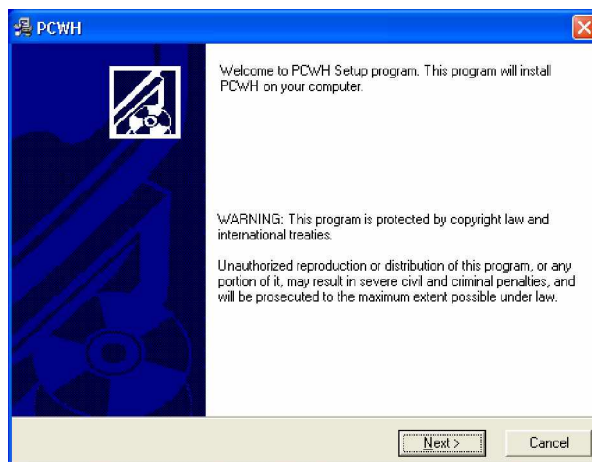
- Download to target pour programmer le PIC de la cible
- Run program pour lancer un programme. On pourra arrêter le programme avec



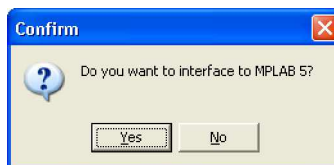
où

2/ Mise place de l'environnement de développement.

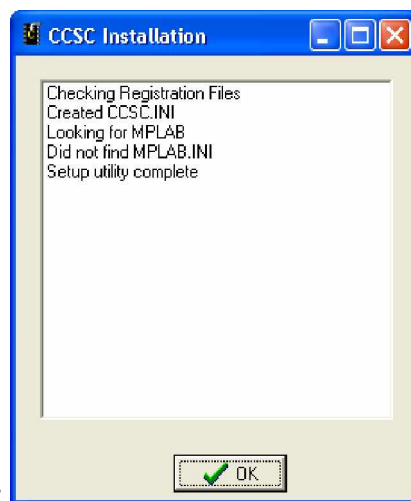
Il faut installer le compilateur C en lançant le programme.



On accepte le répertoire  qui contient déjà le gestionnaire de l'ICD.



Si on utilise pas MPLAB de Microchip on répond No :



L'installation est terminée :

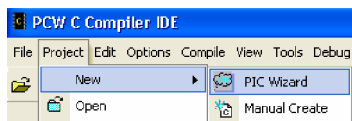


L'icône permet de lancer l'environnement de développement et le menu Help>About permet de voir la version des outils :



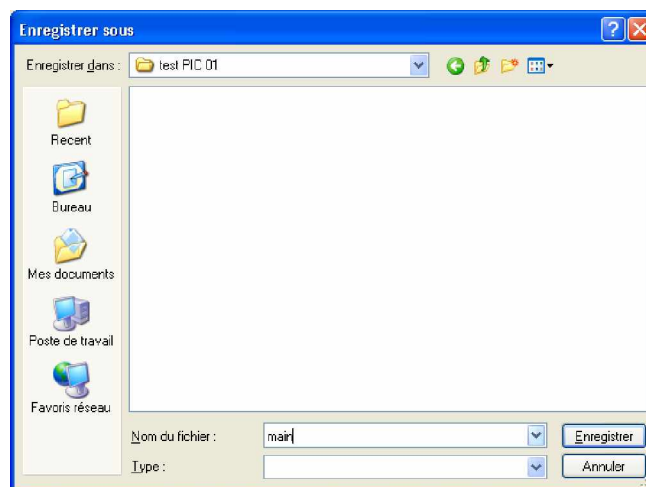
Développement en C avec CCS et ICD.

Nous voilà prêt pour un premier projet :

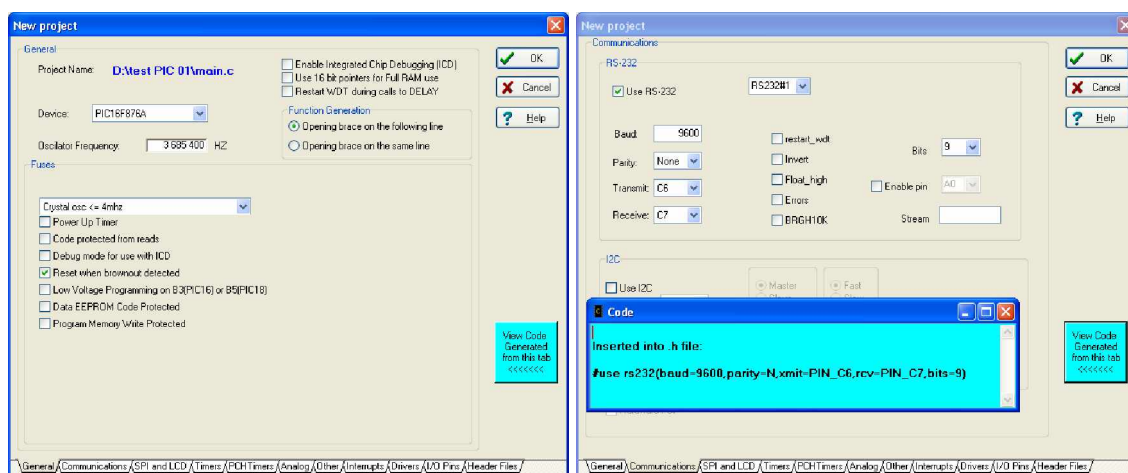


On choisit l'assistant PIC Wizard :

On choisit une localisation pour le projet (répertoire) puis un nom (main).



On choisit les caractéristiques (processeur, fréquence ...)



Le fait de cocher ou non des options va inclure des options pour la configuration du PIC lors de sa programmation.

Une fois accepté ces options on arrive au projet qui comporte 3 fichiers :



On distingue :

- main.pjt qui décrit le projet
- main.c le fichier source principal
- main.h appelé dans main.c et qui contient les options de programmation du PIC choisi.

On remarque que main.h fait appel au fichier de définition du PIC choisi. On peut réorganiser le projet et inclure les lignes de main.h dans main.c. On peut également modifier les lignes obtenues depuis l'assistant pour en ajouter, en supprimer ou les modifier.

On a :



Développement en C avec CCS et ICD.

```
main.c | main.h
#include "D:\test pic 01\main.h"

void main()
{
    port_b_pullups(TRUE);
    setup_adc_ports(M0_ANALOGS);
    setup_adc(ADC_OFF);
    setup_spi(FALSE);
    setup_counters(RTCC_INTERNAL,RTCC_DIV_2);
    setup_timer_1(T1_DISABLED);
    setup_timer_2(T2_DISABLED,0,1);
}
```

```
main.c | main.h
#include <16F876.h>
#define ICD=TRUE
#define adc=8
#define delay(clock=20000000)
#define fuses RC, NOPROTECT, BROWNOUT
#define rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=9)
```

On peut remanier notre projet en supprimant main.h après avoir modifié main.c comme suit :

```
main.c
#include <16F876.h>
#define ICD=TRUE
#define adc=8
#define delay(clock=20000000)
#define fuses RC, NOPROTECT, BROWNOUT, NOWDT
#define rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=9)

void main()
{
}
```

Attention, ne pas oublier l'option NOWDT qui dévalide le watchdog sans quoi le programme utilisateur redémarre automatiquement (si on ne réarme pas le chien de garde).

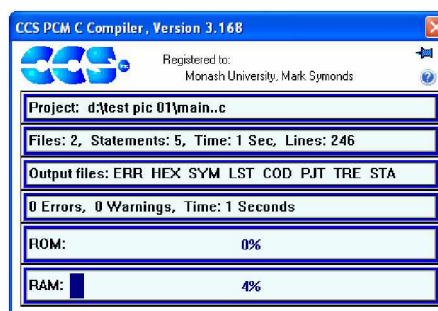
Allons y pour un petit programme.


```
#include <16F876.h>
#define ICD=TRUE
#define fuses RC, NOPROTECT, BROWNOUT, NOWDT

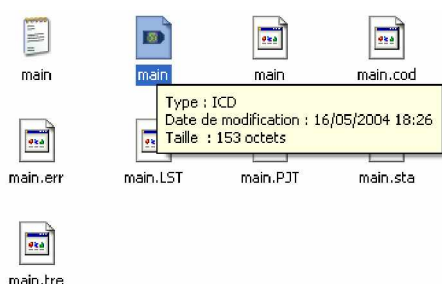
int a,b;

void main()
{
    a=0;
    b=1;

    while (1)
    {
        a=a+b;
    }
}
```

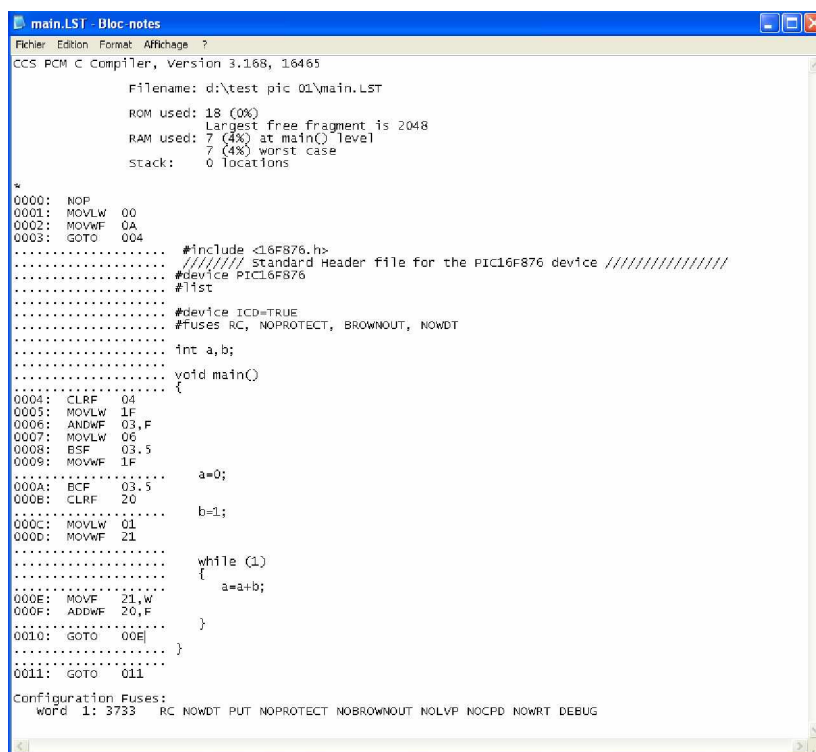


L'action sur  lance la compilation ci-dessus et on obtient les fichiers suivants :



Le fichier associé à l'ICD est le fichier code machine en mode .hex.


Le fichier .lst contient le listing que l'on peut observer pour voir comment opère le compilateur.

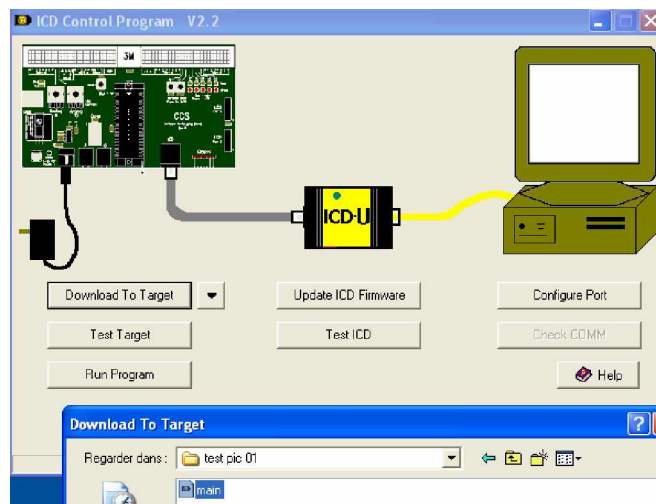


3/ Implantation du programme dans la cible.

Pour implanter ce programme sur une cible, on peut utiliser directement la cible connecté à l'ICD. On a deux cas de figure :

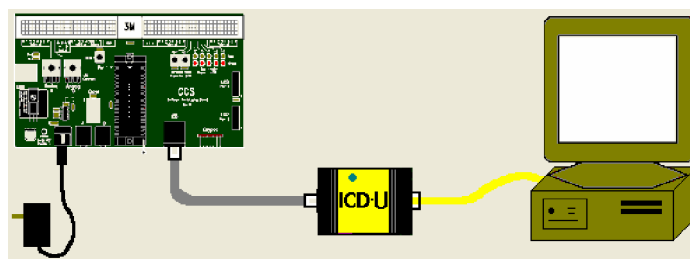
a/ Programmation directe

Le programme fonctionne et on désire programmer la cible ; on utilise directement l'ICD  pour programmer la cible et lancer le programme. Si le programme fonctionne comme il faut, une fois l'ICD déconnecté, c'est terminé.



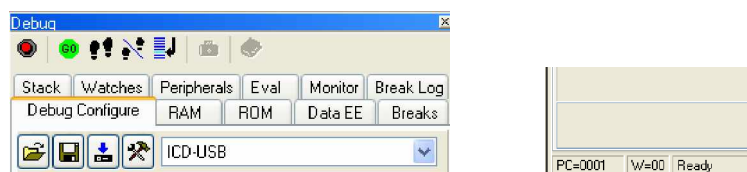
b/ Test et implantation.

Si l'on désire débbugger le programme, on utilise, après avoir connecté la cible et l'ICD, directement



dans CCS

Si la connexion s'effectue correctement, le programme est téléchargé dans la mémoire du PIC, et le débbugueur donne la main en affichant ready.








Si la connexion ne marche pas, il faut vérifier, dans les options du PIC, par exemple, l'horloge qui doit correspondre à celle de la cible (XT pour quartz inférieur à 4 MHz , HS ou RC).

```
main.c
#include <16F876.h>
#define ICD=TRUE
#fuses XT, NOPROTECT, BROWNOUT, NOWDT
```

On peut utiliser les boutons de commande pour commander le test d'un programme :

Développement en C avec CCS et ICD.

-  permet de faire un reset du programme.
-  permet de lancer un programme en permanence ou jusqu'à un point d'arrêt.
-  permet de faire du pas à pas en rentrant dans les fonctions.
-  permet de faire du pas à pas sans rentrer dans les fonctions.
-  permet de lancer un programme jusqu'à la ligne où l'on a placé le point d'insertion dans le fichier source.

Pour placer un point d'insertion, on positionne le curseur à l'endroit voulu dans le fichier source, puis dans la fenêtre break du débogueur, on clique sur le bouton plus. Le point rouge indique alors le point d'arrêt. Un seul point d'arrêt est autorisé et le bouton moins permet de supprimer un point d'arrêt (après l'avoir sélectionné).

