0 Présentation du TP :

Pré-requis : ⇒ Connaissance du langage C ANSI.
Durée estimée : ⇒ 1 heures
Objectifs : ⇒ Ecrire, simuler et débugger un programme C

Ce TP vous permettra d'appréhender les outils de débogage intégrés dans ALTIUM associés au **langage C**.
Cela vous permettra de simuler et mettre au point des programmes en **langage C**.
A l'issu de cette mise au point le code développé à pour finalité d'être descendu sur le processeur embarqué TSK3000A, ce qui nous renvoie au TP3.

Sommaire du TP:

1 Création du projet embarqué
2 Ecriture du programme C « Hello World ! »
3 Mise au point du programme en langage C
⇒ Simulation du programme avec un point d'arrêt
⇒ Visualisation d'une variable
⇒ Visualisation des registres SFR du processeur TSK3000A
⇒ Exécution du programme en mode pas à pas

Dans ce TP nous nous servirons d'un programme simple affichant la chaîne de caractères **« Hello World ! »**

Ceci est un petit clin d'œil à Messieurs Ritchie, Thomson et Kernigham :

⇒ L'origine du langage C : Il a été développé par Dennis **R**itchie et **K**en Thompson dans les années 70. Ce langage C des origines est nommé **K&R C**.
Brian Kernighan aida à populariser le langage à l'aide du livre **« The C Programming Language »** décrivant le langage enfin stabilisé en 1978.
Dans son livre Kernighan propose un programme d'exemple : Le programme Hello World.
Créer un programme affichant Hello World est depuis devenu l'exemple de référence pour présenter les bases d'un nouveau langage informatique.

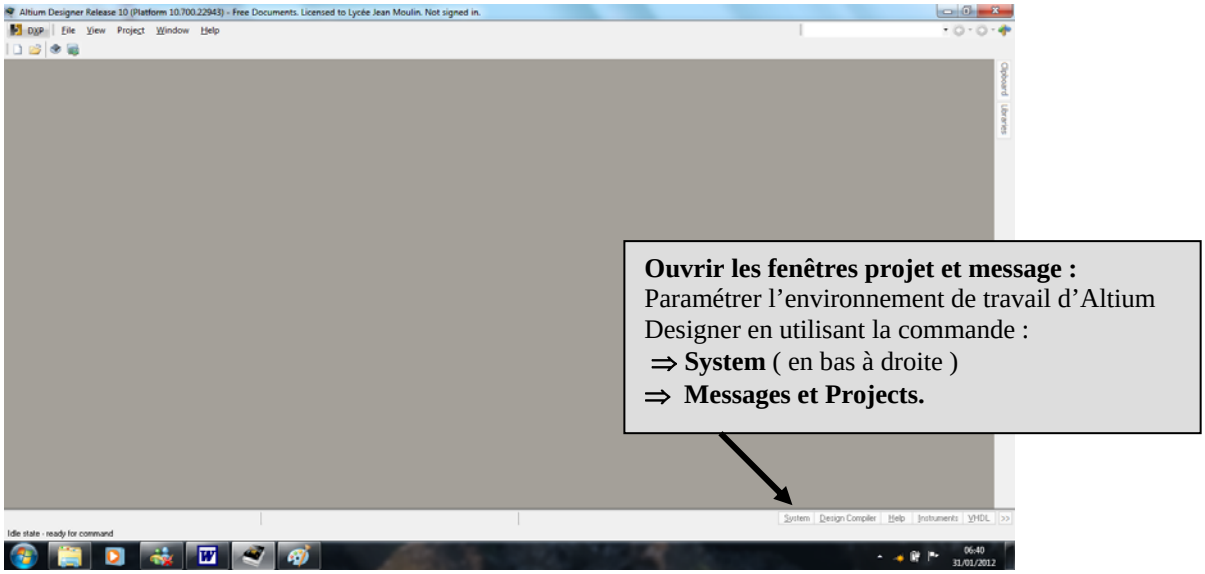
Hello world tel qu'il est proposé en 1978:

```
#include <stdio.h>
main()
{
   printf("hello, world\n");
}
```

ANSI C est une évolution normalisée du C K&R qui reste extrêmement compatible.

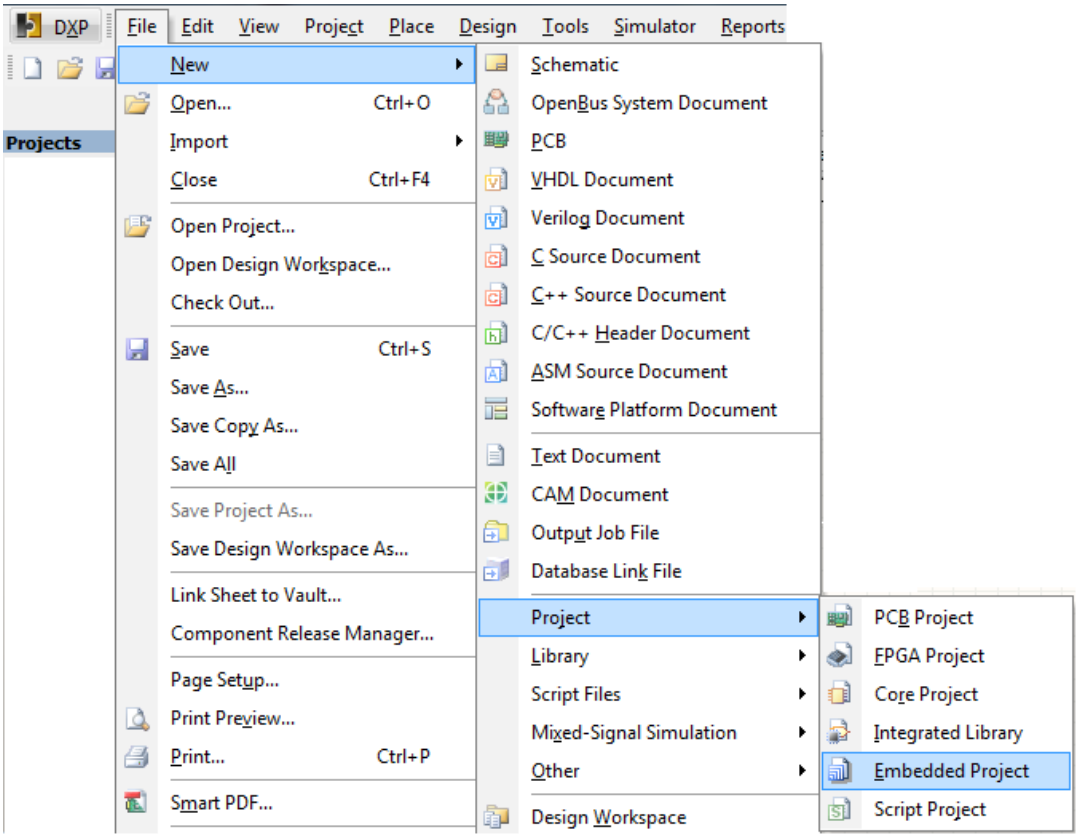
1 Création du projet embarqué

1.1 Repartir d'un environnement vide :



1.2 Ajouter un projet embarqué à votre environnement :

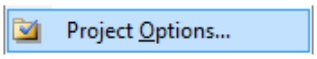
⇒ Créer un nouveau projet ⇒ commande ⇒ **File ⇒ New ⇒ Embedded Projet**.
⇒ Un projet nommé « **Embedded_Projet.PrjEmb »** apparaît dans l'onglet gestion de projet

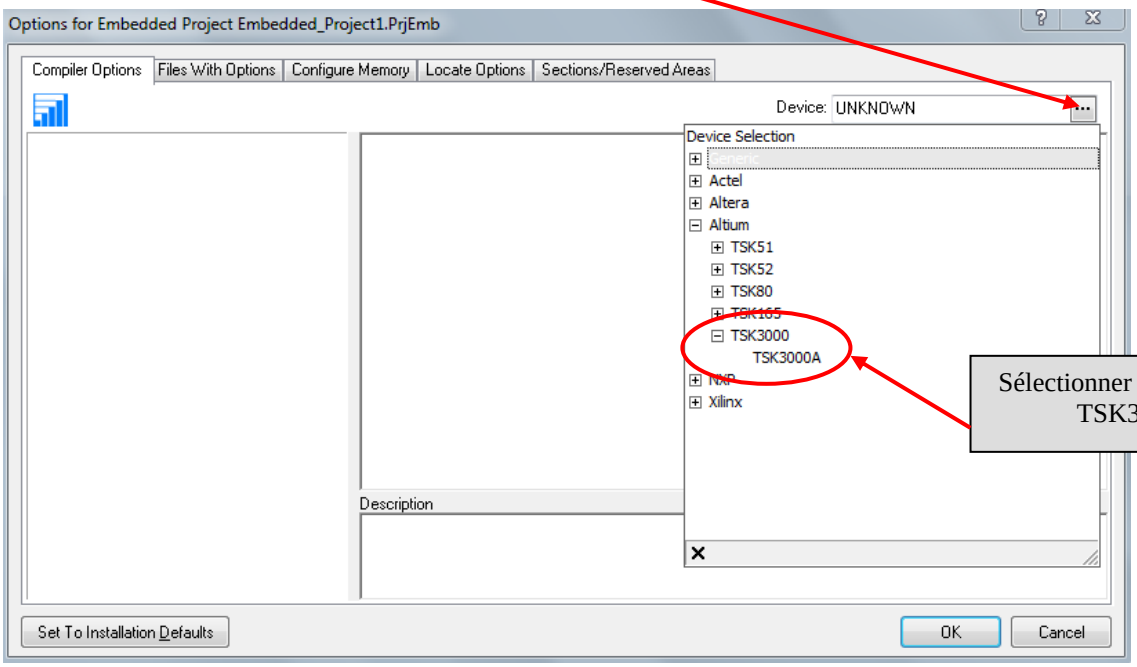


1.3 Définir le processeur auquel est destiné le projet embarqué.

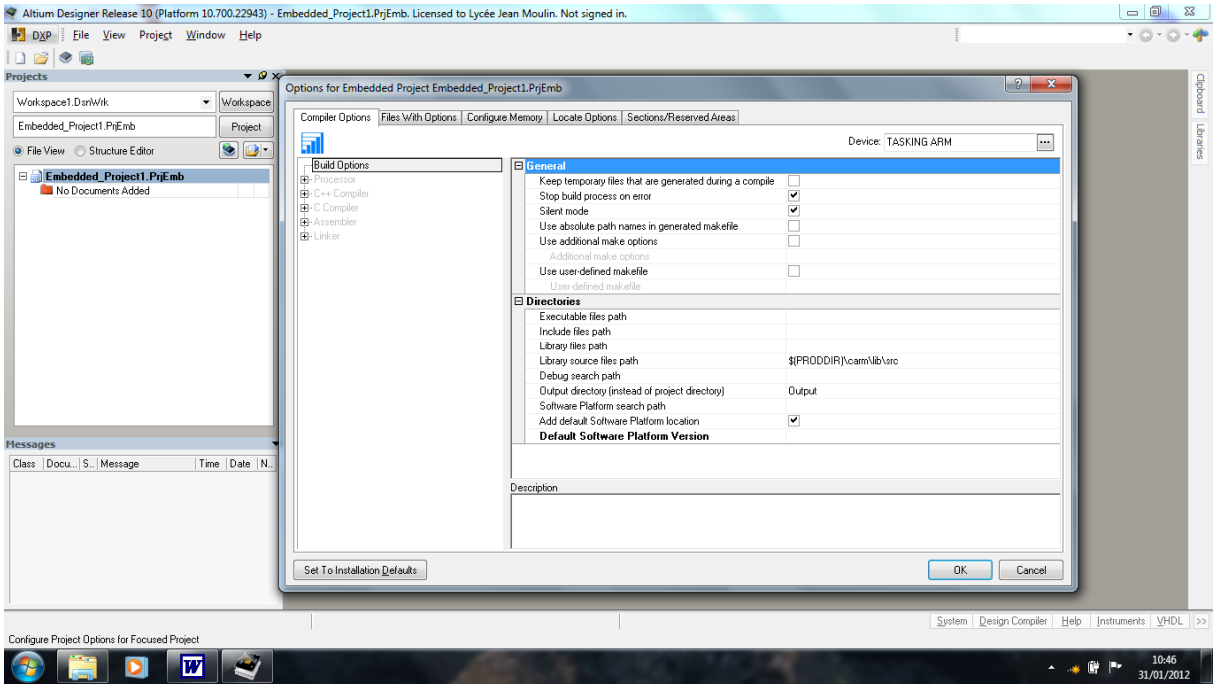
> Remarque : définir le processeur revient à définir le compilateur c associé.

⇒ Cliquez droit sur le projet embarqué
⇒ Cliquez sur Project option
⇒ Déroulez la liste des processeurs disponibles



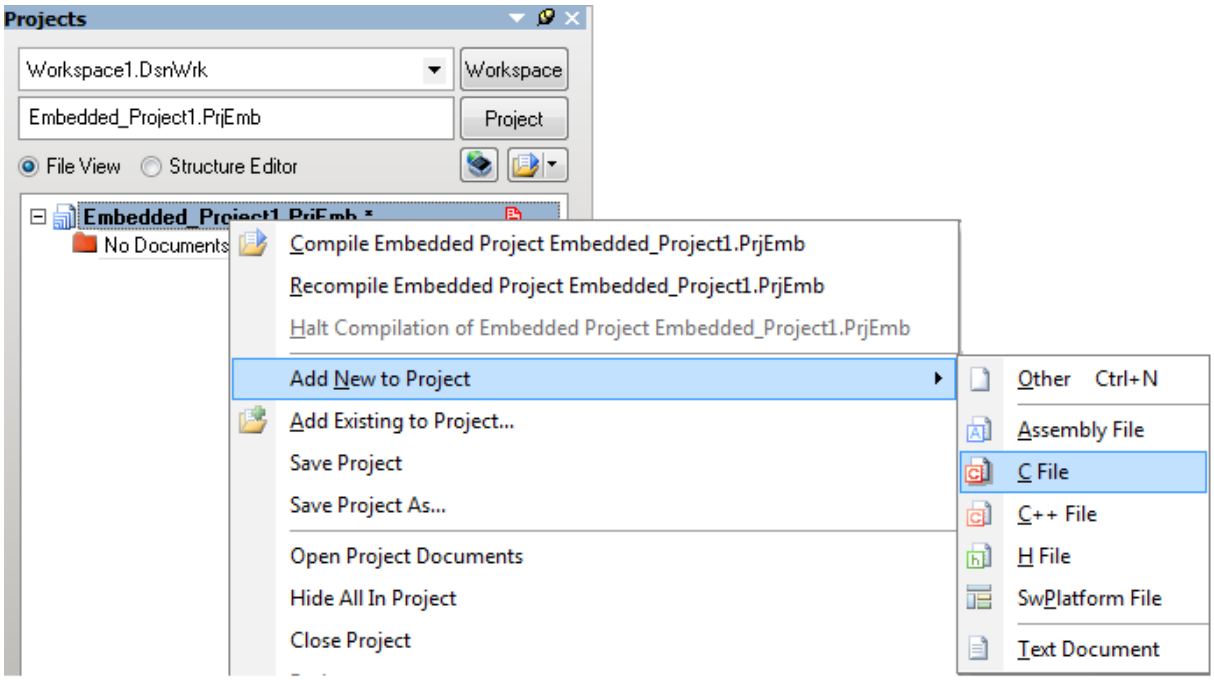


⇒ Vérifiez que les options de compilation suivantes sont sélectionnées :



2 Ecriture du programme C « Hello World ! »

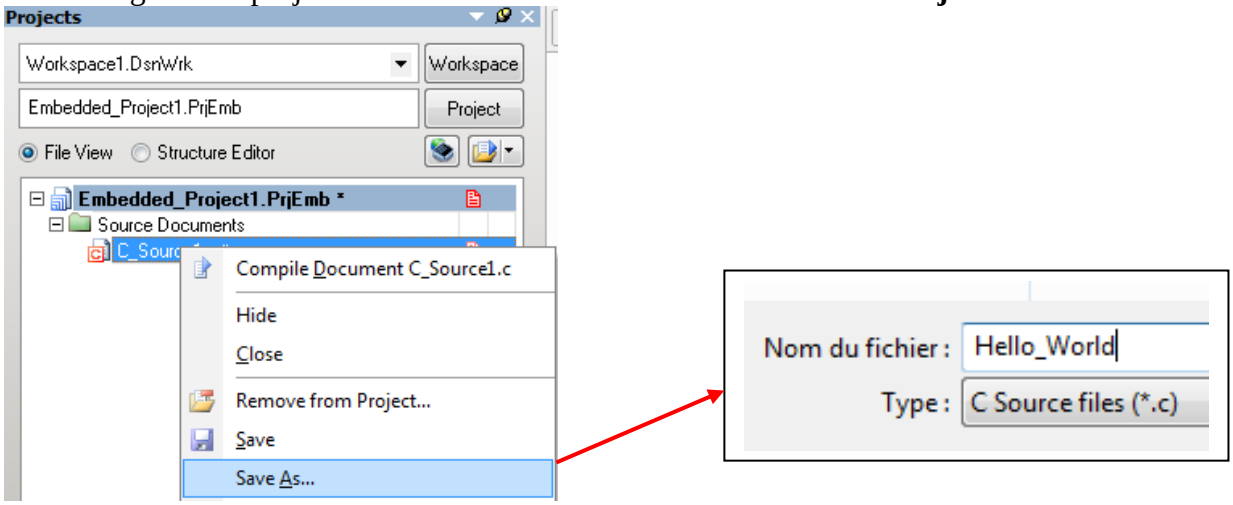
⇒ Cliquez droit sur le projet embarqué et adjoindre un fichier C :
⇒ **Add New to Project ⇒ C File.**



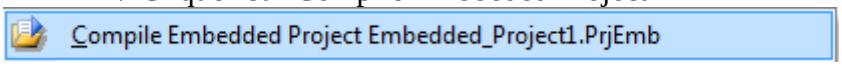
⇒ Dans ce fichier C recopiez le code ci-dessous :

```c
#include <stdio.h>

void printloop(void)

{
    int loop;

    for (loop=0; loop<10; loop++)
    {
        printf("%i\n",loop);
    }
}

void main(void)

{
    printf("Hello World!\n");
    printloop();
}
```

⇒ Sauvegarder ce document : **File ⇒ Save As** « *main.c* » dans le dossier parent du projet.
⇒. Sauvegarder le projet à nouveau : clic droit sur son nom ⇒ **Save Projet**.



⇒ Compilez le projet embarqué : ⇒ Cliquez droit sur le projet
⇒ Cliquez sur Compile Embbeded Project

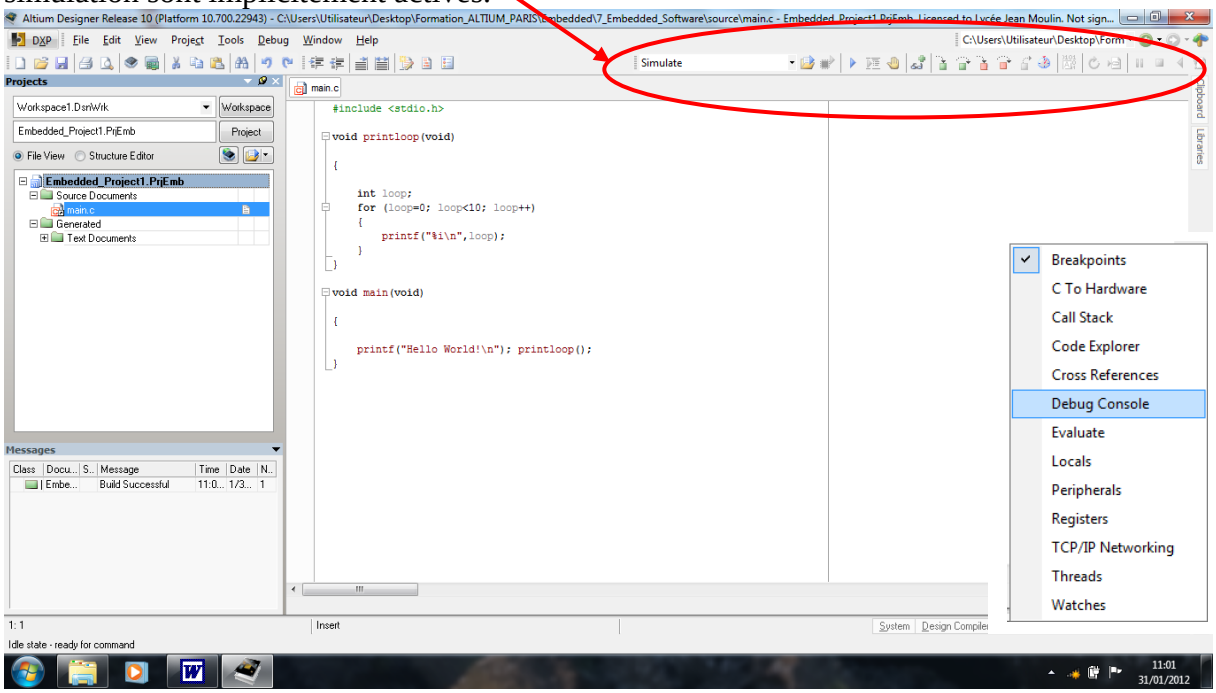


⇒ Si des erreurs apparaissent, corrigez les et recompiler !

3 Mise au point du programme en langage C :

3.1 Simulation du programme avec un point d'arrêt :

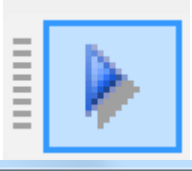
Lorsque vous ouvrez la fenêtre d'édition du programme Hello_World.c les fonctions de simulation sont implicitement actives.

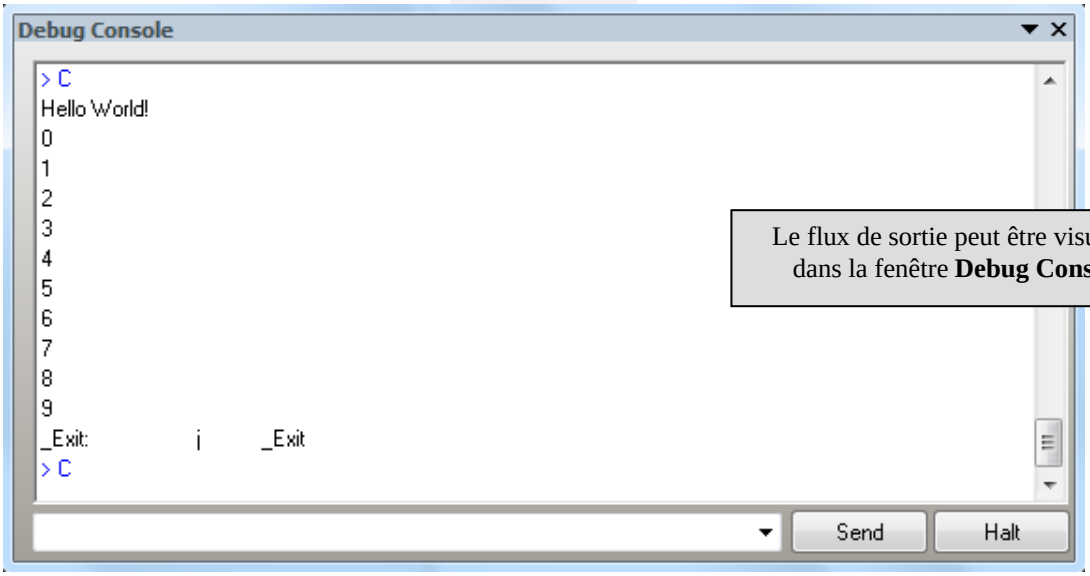


⇒ Ouvrez les fenêtres **breakpoint** et **Debug** à partir de l'onglet **Embedded** (en bas à droite)

Debug Console est la sortie par défaut des fonctions **stdio** donc de la fonction **printf ( )** ;

⇒ Exécution du programme sans point d'arrêt : Run the embbeded program



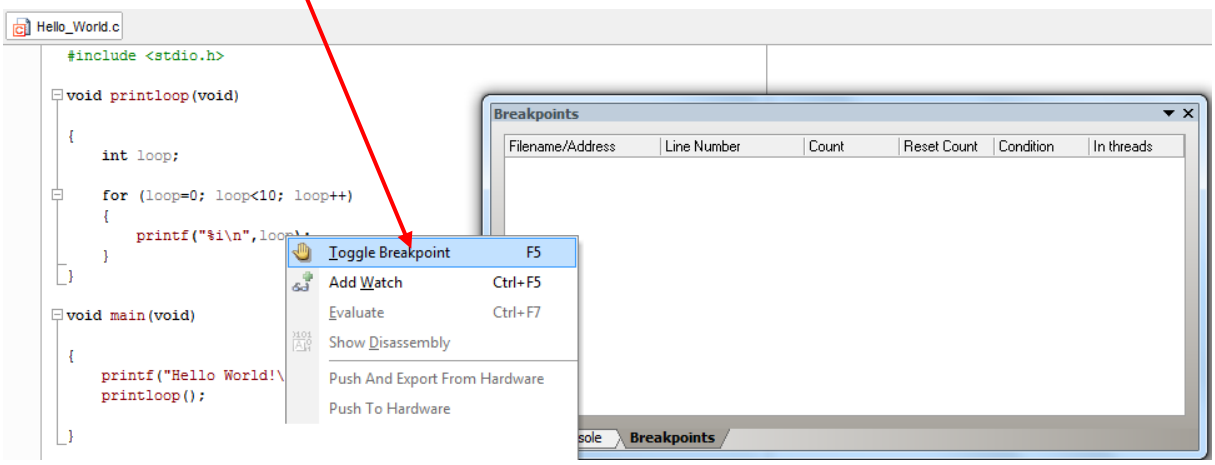


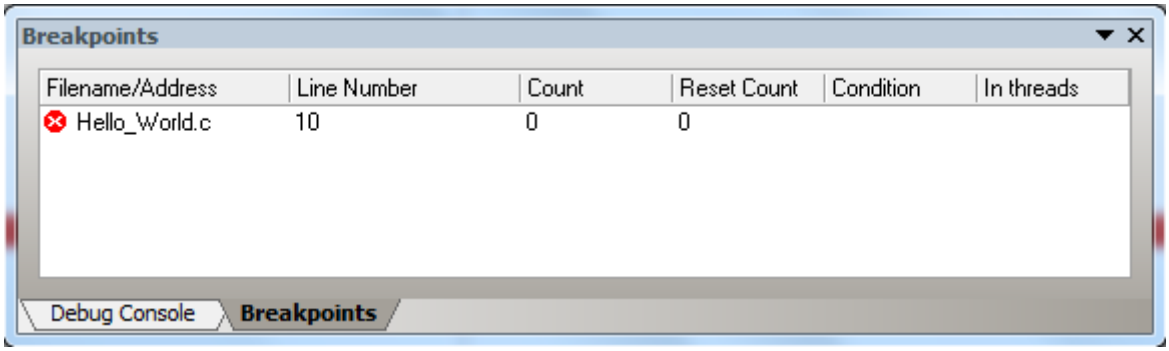
Le flux de sortie peut être visualisé dans la fenêtre **Debug Console**

⇒ Arrêter l'exécution du programme bouton stop, puis effectuer un reset avec retour en début du programme C :

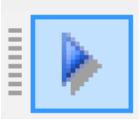
stop :  puis reset et retour en début de programme : 

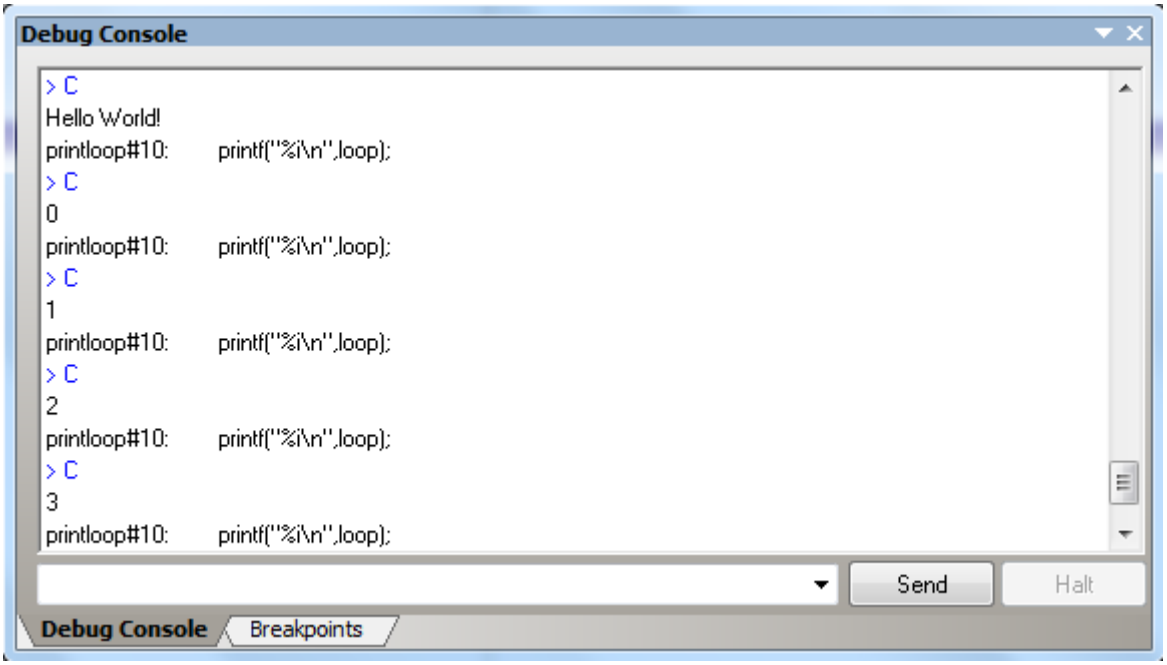
⇒ Pour ajouter un point d'arrêt ligne 10, **(printf(« %i \n, loop)**, placez le curseur dessus
⇒ Cliquez droit
⇒ Toggle Breakpoint





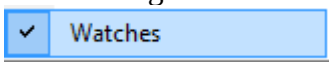
⇒ Exécution du programme avec les points d'arrêts : Run the embbeded program



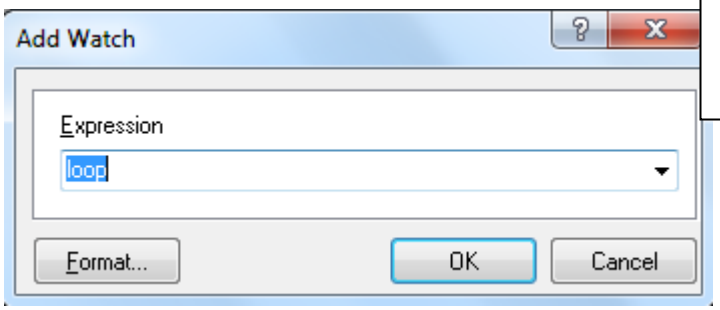


3.2 Visualisation d'une variable :

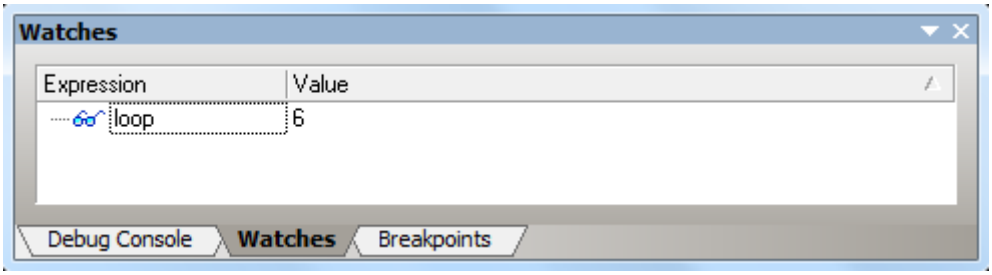
⇒ A partir de l'onglet **Embedded** sélectionnez **Watches**



Insérer une nouvelle image pour préciser comment on ajoute une expression
Afaire < !!!!!

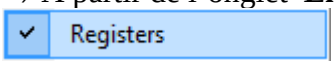


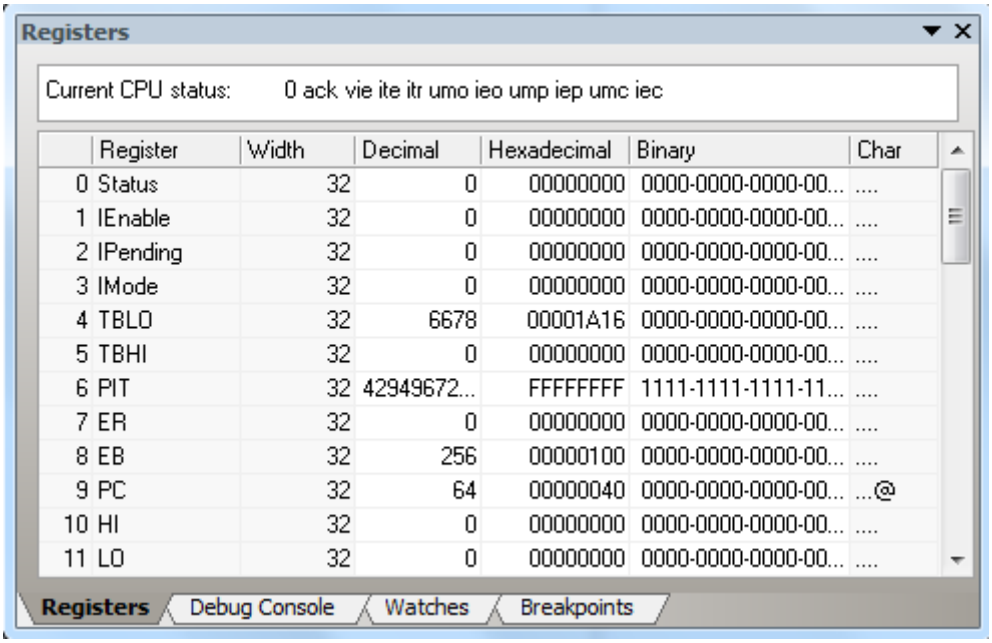
⇒ Visualisation de la variable « loop ».
⇒ Placer un point d'arrêt dans la fonction **«printloop() ; »**.
⇒ Relancer le programme à chaque arrêt : la variable **« loop »** s'incrémente.



3.3 Visualisation des registres SFR du processeur TSK3000A

⇒ A partir de l'onglet **Embedded** sélectionnez **Registers**





3.4 Exécution du programme en mode pas à pas :

⇒ Touche F7 : Step into the current source line 

⇒ Touche F8 : Step to the next source line 

********** Fin du TP5 **********