

Proteus et Arduino

Tutorial 2 – Simulation en pas à pas

MULTIPOWER – Avril 2015

Réalisé avec Proteus V8.2



Objectif : Ce tutorial fait suite au tutorial 1 sur Proteus et Arduino. Son objectif est de vous faire découvrir le potentiel de l'environnement de débogage intégré à Proteus VSM.

Etape 1 : Les fonctions Setup et Loop

Les deux fonctions **setup()** et **loop()**, toujours présentes dans les projets Arduino, ont un rôle bien précis.

La fonction **setup()** est utilisée pour initialiser le contexte du projet ainsi que les valeurs de départ des variables. Cette fonction n'est exécutée qu'une seule fois au lancement.

La fonction **loop()**, est une boucle sans fin qui contient les actions à entreprendre. C'est cette fonction qui contrôle votre hardware.

```
main.ino x
1  /* Main.ino file generated by New Project wizard
2  *
3  * Created:   Thu Sep 5 2013
4  * Processor: ATmega328P
5  * Compiler:  Arduino AVR
6  */
7
8  #include <LiquidCrystal.h>      /* FICHER A INCLURE */
9
10 LiquidCrystal lcd (8,9,4,5,6,7); /* DEFINITION DES LIENS */
11
12 void setup()
13 {
14     lcd.begin (16,2);           /* lcd.begin(cols, rows) - LCD DE 16 colonnes, 2 lignes */
15     lcd.setCursor (0,0);       /* Curseur sur la première colonne de la première ligne */
16     lcd.print ("SYSTEME PRET"); /* Affiche le texte 'Programme' */
17 }
18
19 int ma_var = 0;
20
21 void loop()
22 {
23     lcd.clear();
24     lcd.setCursor (0,1);       /* Curseur sur la première colonne de la seconde ligne */
25     lcd.print (ma_var );      /* Affiche le contenu de la variable ma_var */
26     delay (1000);             /* Attente de 1000 ms soit 1 seconde */
27     ma_var = ( ma_var + 1 ) % 10; /* Compte de 0 à 9 grâce au modulo 10 */
28 }
29
```

Notre programme de test

ACTIONS :

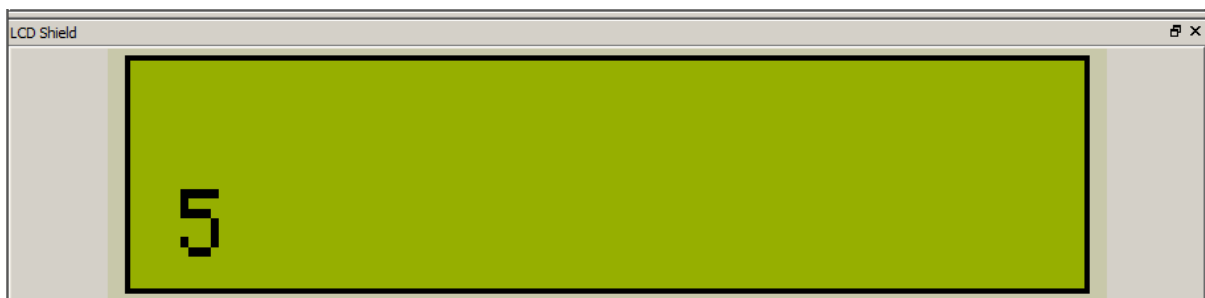
- 1) Ajoutez le contenu de la fonction loop() de la page précédente.
- 2) Construisez le programme via le premier bouton en haut et à gauche.



- 3) Lancez l'exécution du programme via le premier bouton Play en bas et à gauche.



Le LCD affiche chaque seconde une valeur qui s'incrémente de 0 à 9, puis reprend à 0.



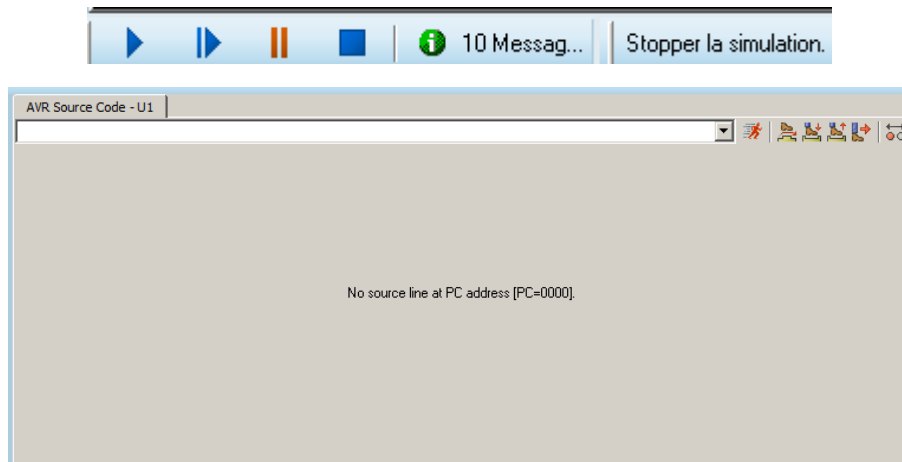
A ce stade, les fonctions setup() et loop() sont renseignées et le programme s'exécute correctement sans arrêt et sans fin.

Arrêtez le programme via le bouton STOP - 4ème bouton en bas et à gauche



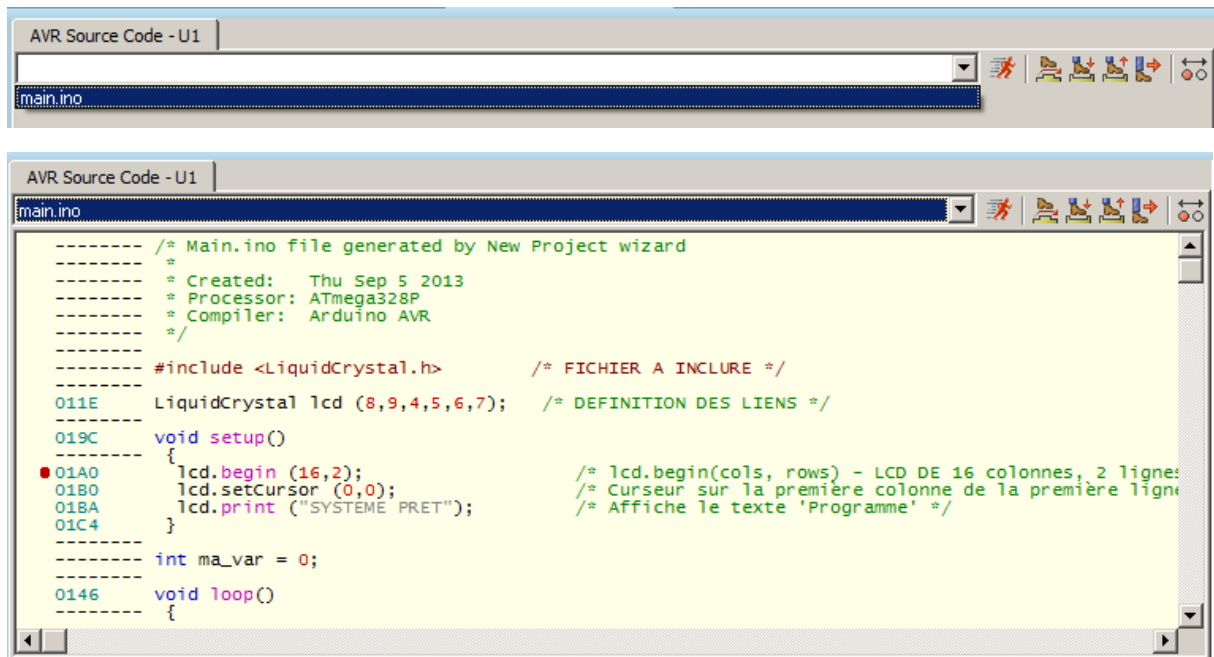
Etape 2 : Placer un point d'arrêt.

Pour placer un point d'arrêt, nous devons lancer le programme en PAUSE par un clic sur le troisième bouton ci-dessous (orange). Comme le programme a été arrêté précédemment, l'exécution commence au temps 0 et stoppe immédiatement.



**A ce stade, le programme est lancé et stoppé.
Le code source n'est pas visible.**

Il faut sélectionner le fichier source **main.ino** grâce à la liste déroulante.



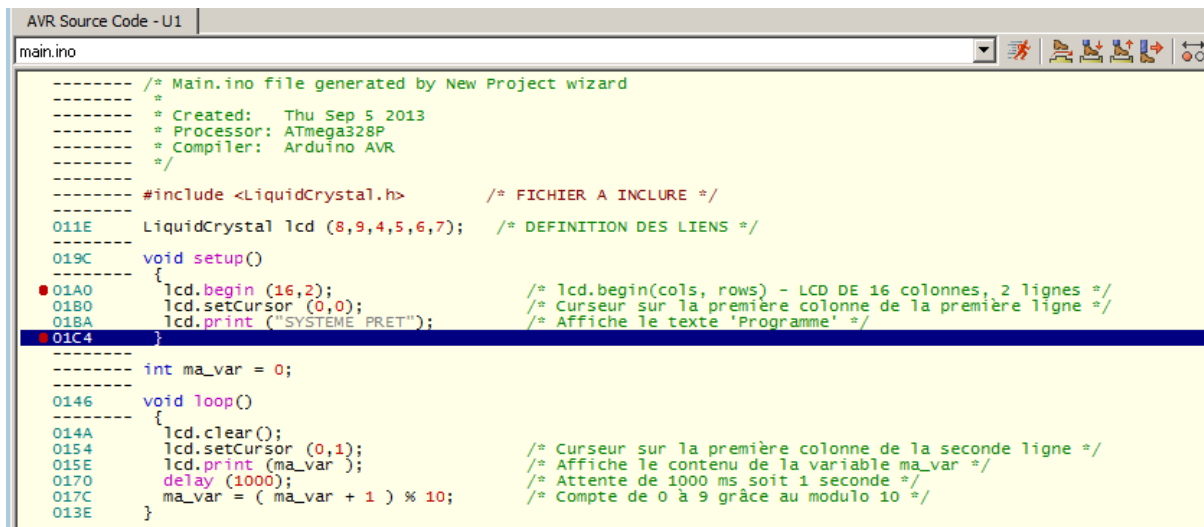
Le point rouge à gauche de l'adresse 01A0 est un point d'arrêt automatique placé au lancement.

Etape 3 : Ajoutez un point d'arrêt à la ligne 01C4.

Il existe plusieurs façons de placer un point d'arrêt sur une ligne.

Méthode 1 : double-cliquez sur la ligne en question.

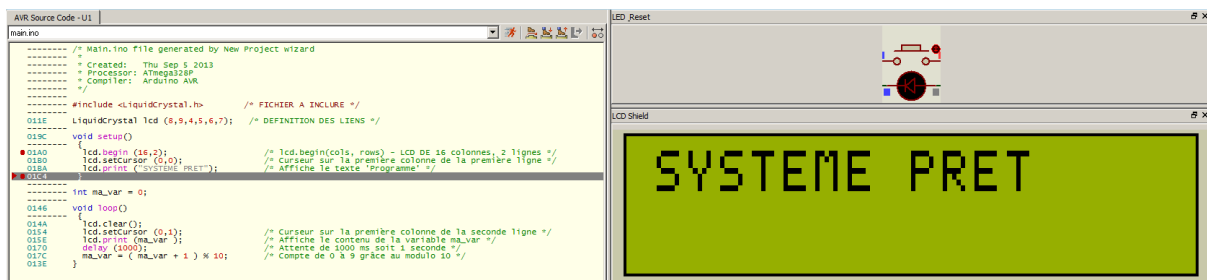
Méthode 2 : placez la souris sur la ligne et faites un clic droit pour accéder au menu contextuel afin de placer un point d'arrêt (**breakpoint**).



```
AVR Source Code - U1
main.ino
----- /* Main.ino file generated by New Project wizard
----- *
----- * Created: Thu Sep 5 2013
----- * Processor: ATmega328P
----- * Compiler: Arduino AVR
----- */
----- #include <LiquidCrystal.h> /* FICHER A INCLURE */
011E LiquidCrystal lcd (8,9,4,5,6,7); /* DEFINITION DES LIENS */
019C void setup()
----- {
01A0 { lcd.begin (16,2); /* lcd.begin(cols, rows) - LCD DE 16 colonnes, 2 lignes */
01B0 lcd.setCursor (0,0); /* Curseur sur la première colonne de la première ligne */
01BA lcd.print ("SYSTEME PRET"); /* Affiche le texte 'Programme' */
01C4 }
-----
----- int ma_var = 0;
-----
0146 void loop()
----- {
014A lcd.clear();
0154 lcd.setCursor (0,1); /* Curseur sur la première colonne de la seconde ligne */
015E lcd.print (ma_var); /* Affiche le contenu de la variable ma_var */
0170 delay (1000); /* Attente de 1000 ms soit 1 seconde */
017C ma_var = ( ma_var + 1 ) % 10; /* Compte de 0 à 9 grâce au modulo 10 */
013E }
```

Point d'arrêt positionné à la ligne 01C4, fin de la fonction setup()

Relancez l'exécution via le bouton Play ; vous observerez que le programme s'arrête à nouveau à l'emplacement du point d'arrêt.



Etat à la ligne 01C4

A ce stade, vous savez placer un point d'arrêt.

Etape 4 : Exécuter un programme en pas à pas.

Vous pouvez également poursuivre l'exécution du programme ligne par ligne.

Pour ce faire utilisez le premier bouton à gauche

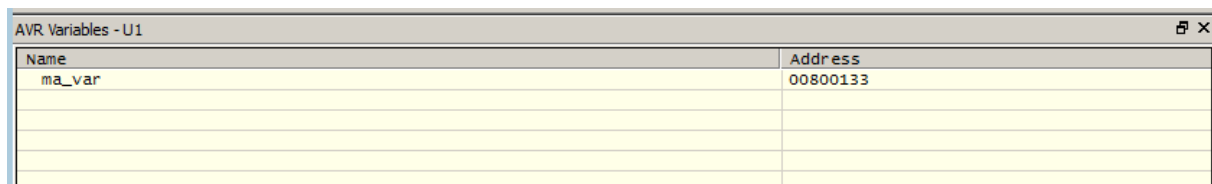


A chaque clic le programme avance d'une ligne !

Etape 5 : Visualiser la valeur d'une variable.

Nous souhaitons afficher l'état de la variable `ma_var`.

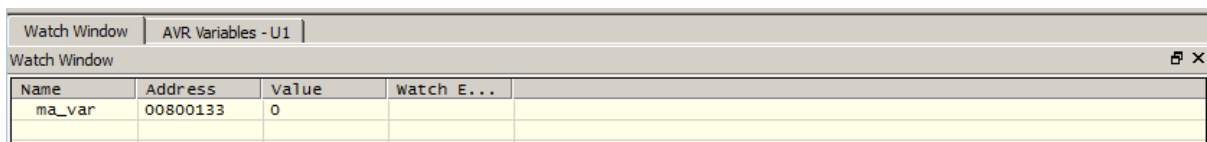
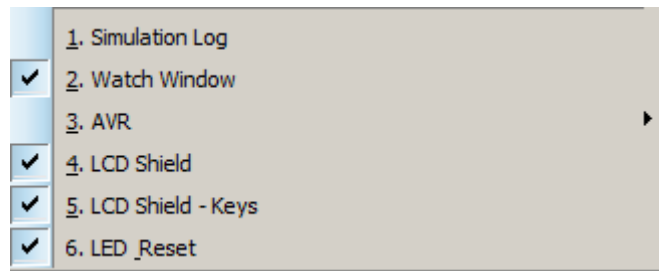
1 : Faites un clic droit sur la ligne `ma_var` et choisissez 'Add to watch window'.



The screenshot shows a window titled "AVR Variables - U1". It contains a table with two columns: "Name" and "Address". The variable `ma_var` is listed with the address `00800133`.

Name	Address
<code>ma_var</code>	<code>00800133</code>

2 : Dans le menu Débogage, cochez la ligne fenêtre Watch.



The screenshot shows the "Watch Window" window. It contains a table with four columns: "Name", "Address", "Value", and "Watch E...". The variable `ma_var` is listed with the address `00800133` and the value `0`.

Name	Address	Value	Watch E...
<code>ma_var</code>	<code>00800133</code>	<code>0</code>	

A ce stade, vous visualisez la valeur de la variable `ma_var`.