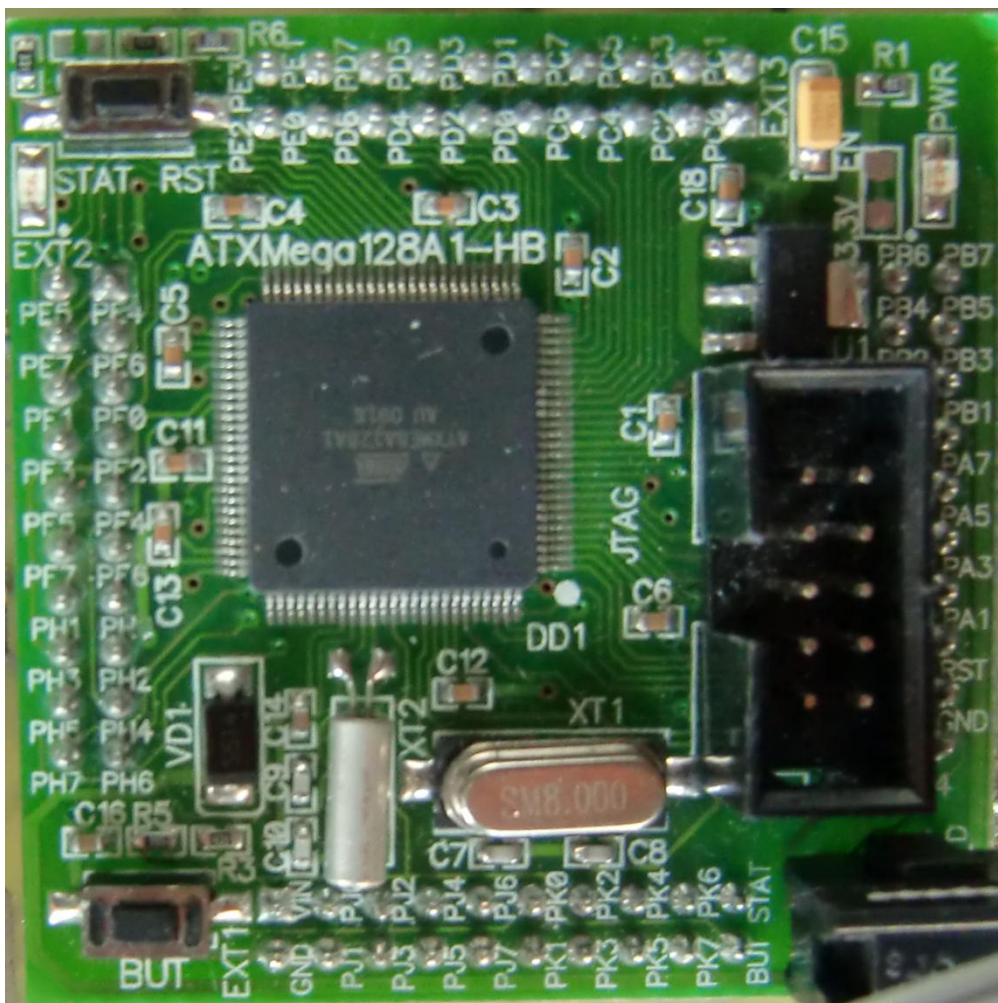


Le Club Robotique de l'INSA Strasbourg présente :

# LA PROGRAMMATION DE L'ATXMEGA 128A1



**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

## Présentation :

L'ATXMéga est microcontrôleur récent fabriqué par Atmel. Nous l'avons utilisé en 2011 pour l'asservissement du robot et la commande de la pince.

En voici les caractéristiques principales :

- 128 ko de mémoire de programme et 8 ko de RAM

- horloge de 32 MHz maximum

- 8 Timers

- 4 modules UART

- Décodeur de quadrature intégré

Cette dernière caractéristique le rend très intéressant dans toutes les applications utilisant un codeur pour asservir un moteur en vitesse ou en position.

De plus, ce microcontrôleur est disponible pré soudé sur une plaquette intégrant régulation 3,3V (tension d'alimentation de la puce), ce qui permet de s'affranchir des problèmes liés au découplage.

C'est pour cela que nous l'utilisons pour l'asservissement de notre robot.

La plaquette ainsi que la documentation associée est disponible en suivant ce lien :

[http://www.bravekit.com/development\\_boards/atmel\\_avr\\_boards/Xmega\\_ATXmega128A1\\_board\\_JTAG\\_PDI](http://www.bravekit.com/development_boards/atmel_avr_boards/Xmega_ATXmega128A1_board_JTAG_PDI)

Le but de ce document est de vous permettre de le mettre en œuvre.

Dans un premier temps, je vous donnerai le mode d'emploi pour installer la suite logicielle nécessaire pour la programmation.

Ensuite, je vous présenterai les différents périphériques ainsi que leur programmation.

## La suite logicielle :

Nous utilisons l'IDE Eclipse associé à différents plug-in.

1) Téléchargez Eclipse à l'adresse suivante :

<http://www.eclipse.org/downloads/>

Vous devez prendre la version Eclipse IDE for C/C++ Developers

Si vous n'avez pas Java installé sur votre ordinateur, il faudra le télécharger et l'installer pour pouvoir utiliser Eclipse.

Voir <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Il vous faudra le JRE.

Une fois téléchargé, installez-le en suivant les instructions.

2) Décompressez le fichier compressé eclipse.

Nom	Modifié le	Type	Taille
configuration	14/04/2011 19:46	Dossier de fichiers	
dropins	18/02/2011 03:20	Dossier de fichiers	
features	14/04/2011 19:46	Dossier de fichiers	
p2	14/04/2011 19:46	Dossier de fichiers	
plugins	15/04/2011 09:54	Dossier de fichiers	
readme	14/04/2011 19:46	Dossier de fichiers	
.eclipseproduct	29/07/2010 10:37	Fichier ECLIPSEPR...	1 Ko
artifacts.xml	18/02/2011 03:20	Document XML	63 Ko
eclipse.exe	22/12/2010 13:08	Application	52 Ko
eclipse.ini	18/02/2011 03:20	Paramètres de co...	1 Ko
eclipse.exe	22/12/2010 13:08	Application	24 Ko
epl-v10.html	25/02/2005 17:53	Document HTML	17 Ko
notice.html	27/04/2010 15:23	Document HTML	9 Ko

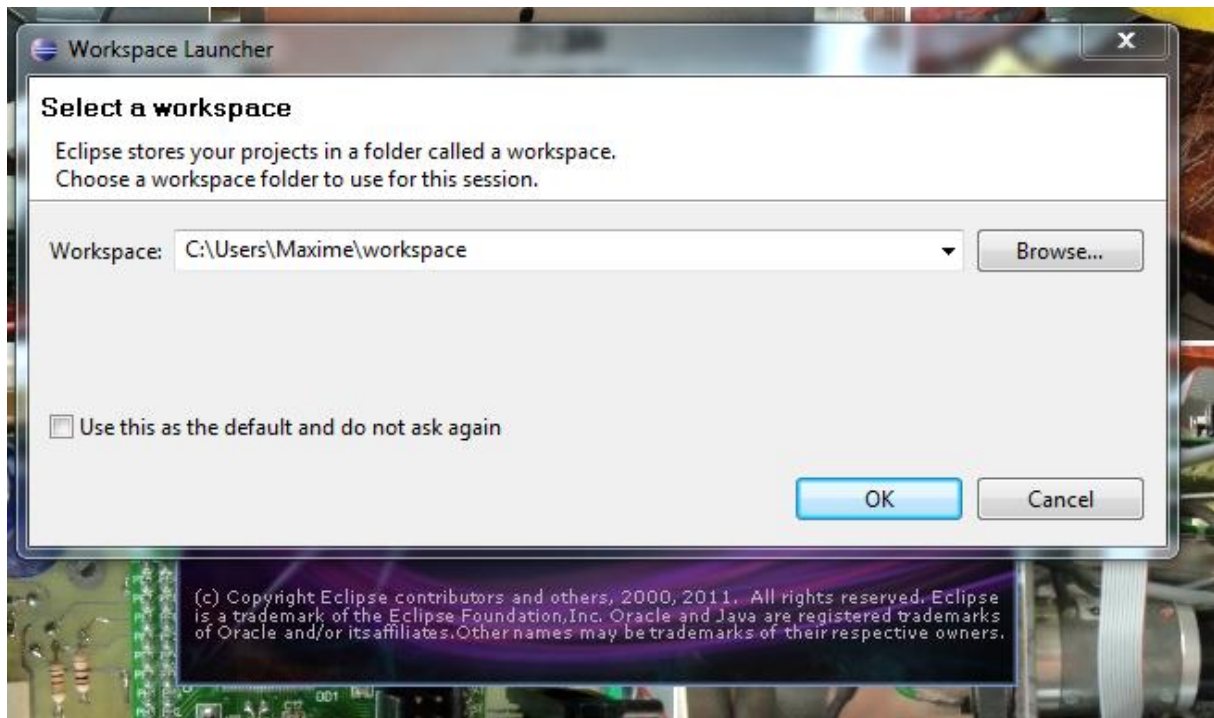
3) Lancez Eclipse en double-cliquant sur « eclipse.exe ».

**Club Robotique de l'INSA Strasbourg**

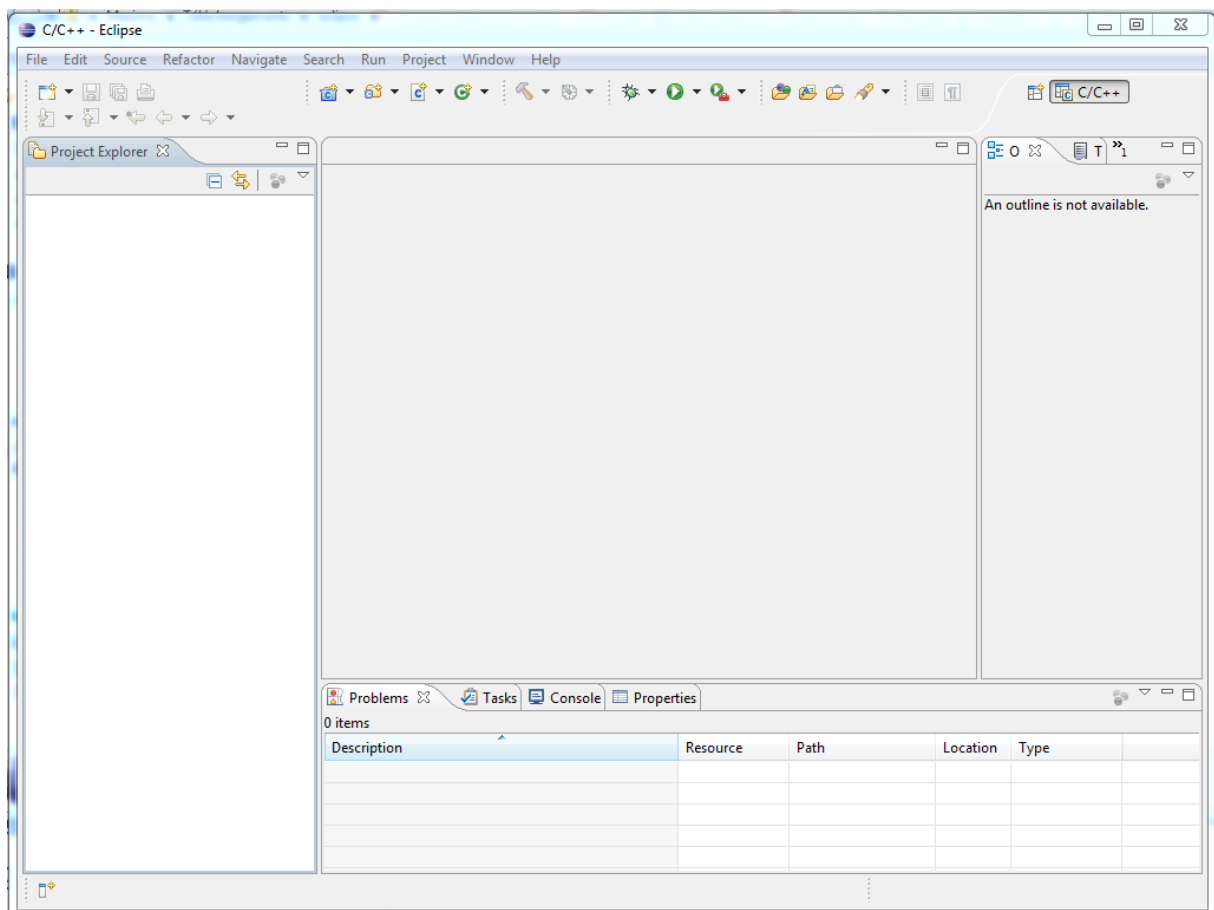
24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

4) Tous vos projets sont rassemblés dans un dossier appelé workspace. Eclipse vous le demandera à chaque démarrage.



5) Vous arriverez ensuite sur la page d'accueil d'Eclipse.



**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

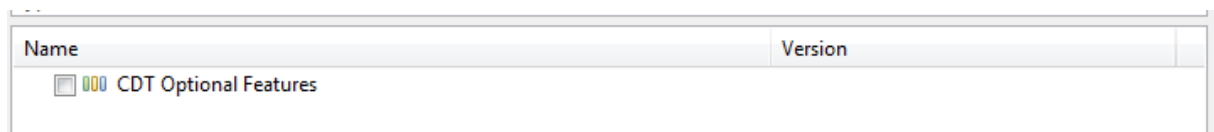
Pour pouvoir programmer notre microcontrôleur, nous devons installer l'AVR Eclipse plug-in.

Pour cela, cliquez sur Help->Install new Software et entrez l'adresse

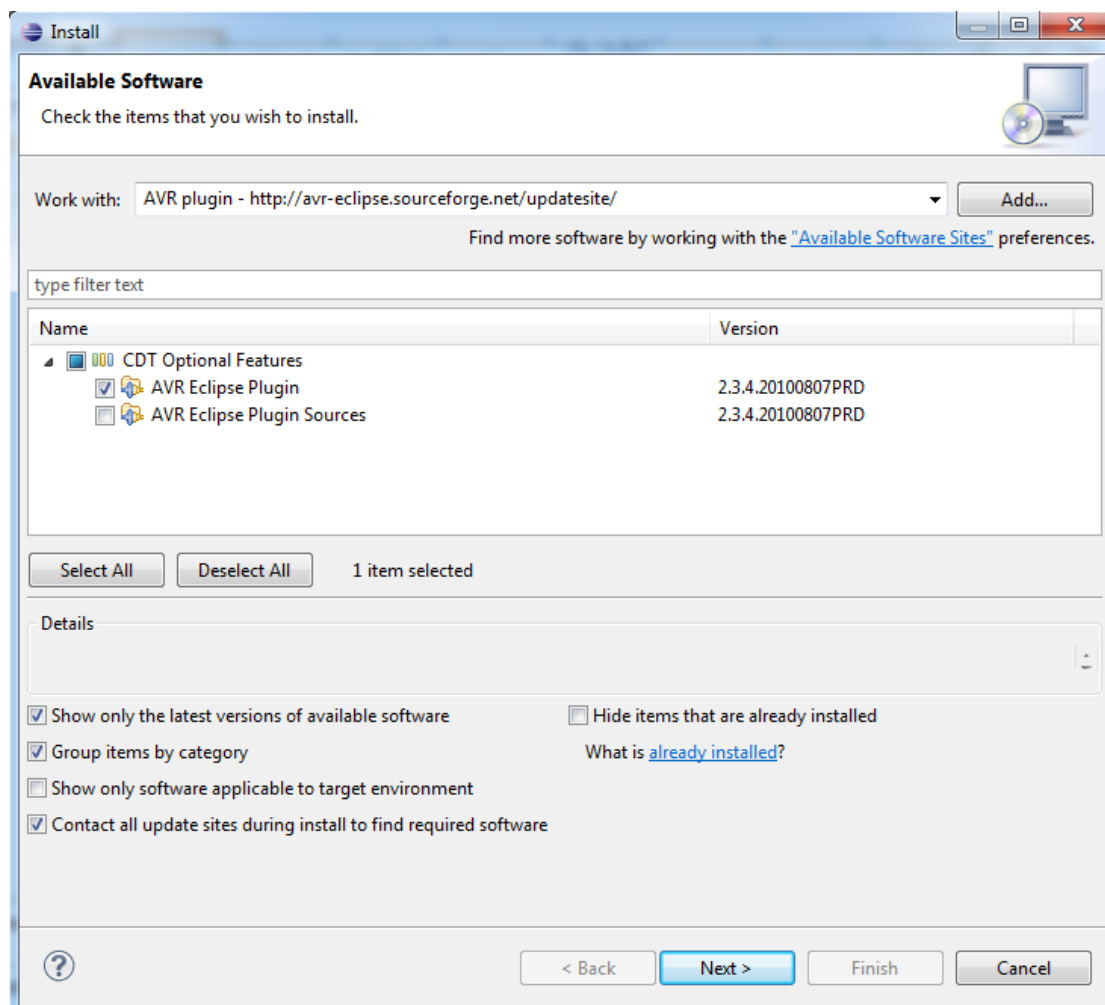
<http://avr-eclipse.sourceforge.net/updatesite/> dans le champ « work with ».

Cliquez ensuite sur « Add » entrez le nom « AVR Eclipse Plugin » dans le champ name puis cliquez sur OK.

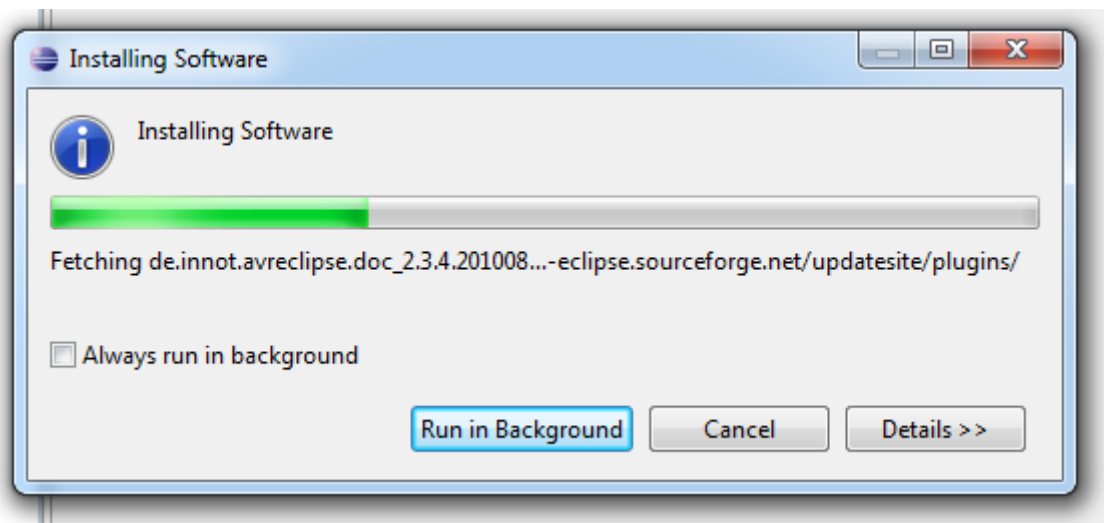
Attendez ensuite d'avoir ceci :



Cochez la case, cliquez sur la petite flèche à côté puis sélectionnez « AVR Eclipse Plugin ».



Cliquez sur Next, suivez les étapes jusqu'à ce l'installation commence.



6) Pour le programmeur, vous avez le choix. Nous utilisons l'AVR ISP mkII.

Il est disponible par exemple à cette adresse :

<http://fr.farnell.com/atmel/atavrisp2/programmeur-avr-mcu-isp/dp/1135517>

Pour l'utiliser, nous avons besoin de WinAVR, qui est disponible ici :

<http://sourceforge.net/projects/winavr/files/WinAVR/>

Prenez la version la plus récente. Il s'agit du premier élément. Cliquez dessus et téléchargez WinAVR-20100110-install.exe .

Ne pas installer WinAVR dans un chemin comportant des parenthèses. WinAvr installé dans le dossier Program Files(x86) ne fonctionnera pas!

Une fois installé, redémarrez Eclipse. Vous devriez à présent voir cette icône



en haut à gauche de la fenêtre d'Eclipse.

## Le premier programme

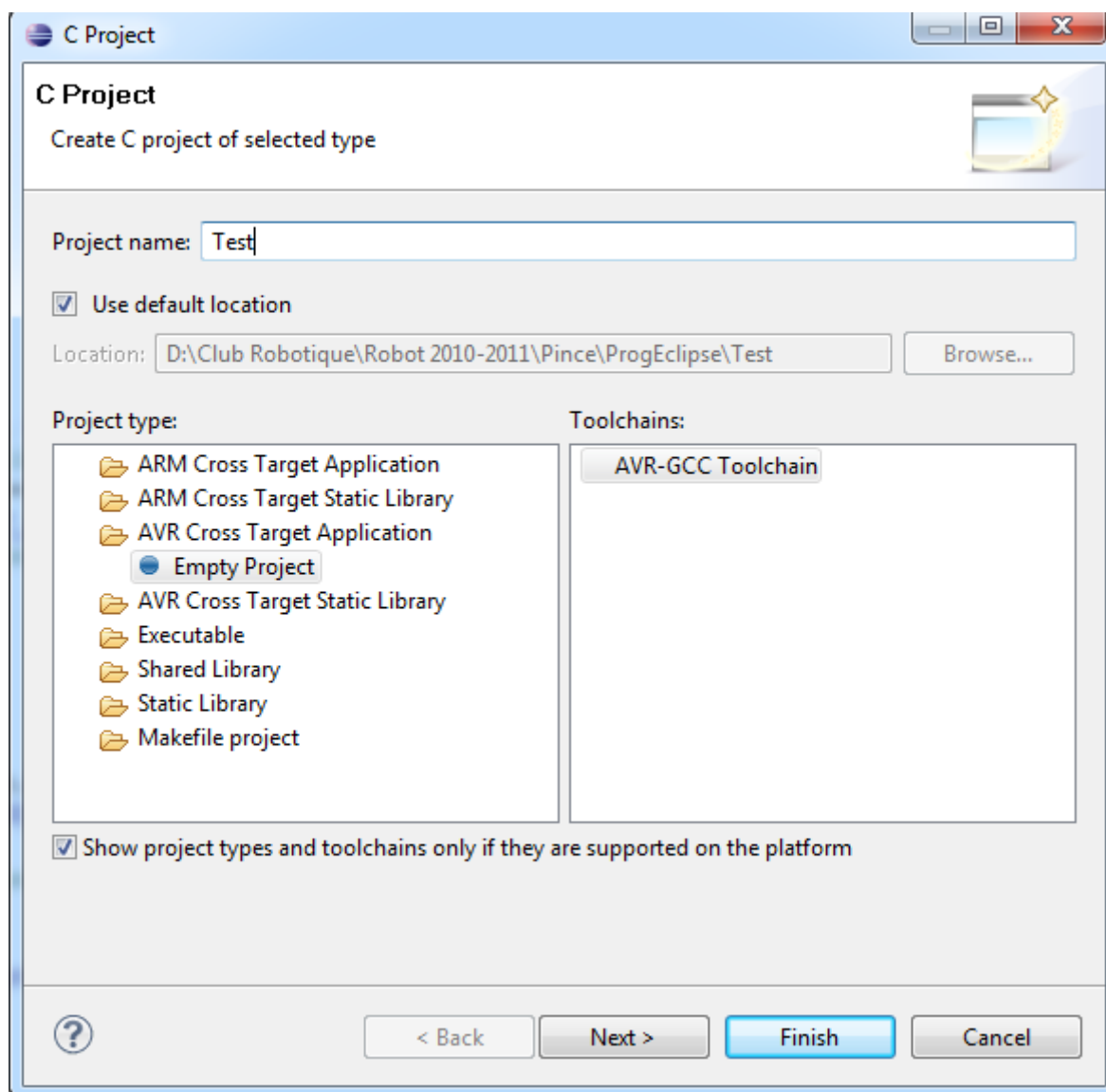
Nous allons maintenant écrire un programme de test. Le but est de faire clignoter la LED sur la platine.

Nous allons commencer par créer un projet :

File -> New -> C Project

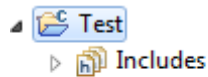
Dans Project name, entrez le nom du projet, par exemple Test.

Dans Project type, sélectionnez le dossier AVR Cross Target Application puis Empty Project.





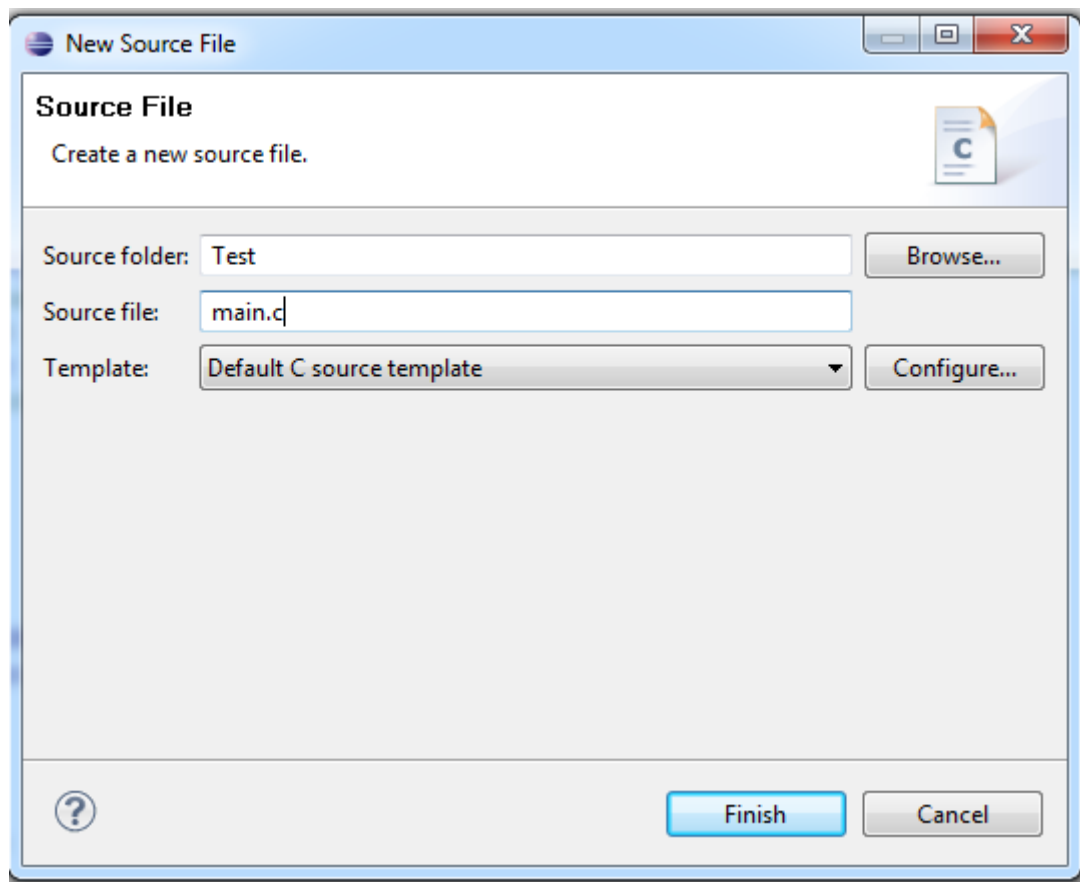
A gauche, dans project explorer, vous devez avoir ceci :



Le project explorer affiche tous les fichiers du projet.

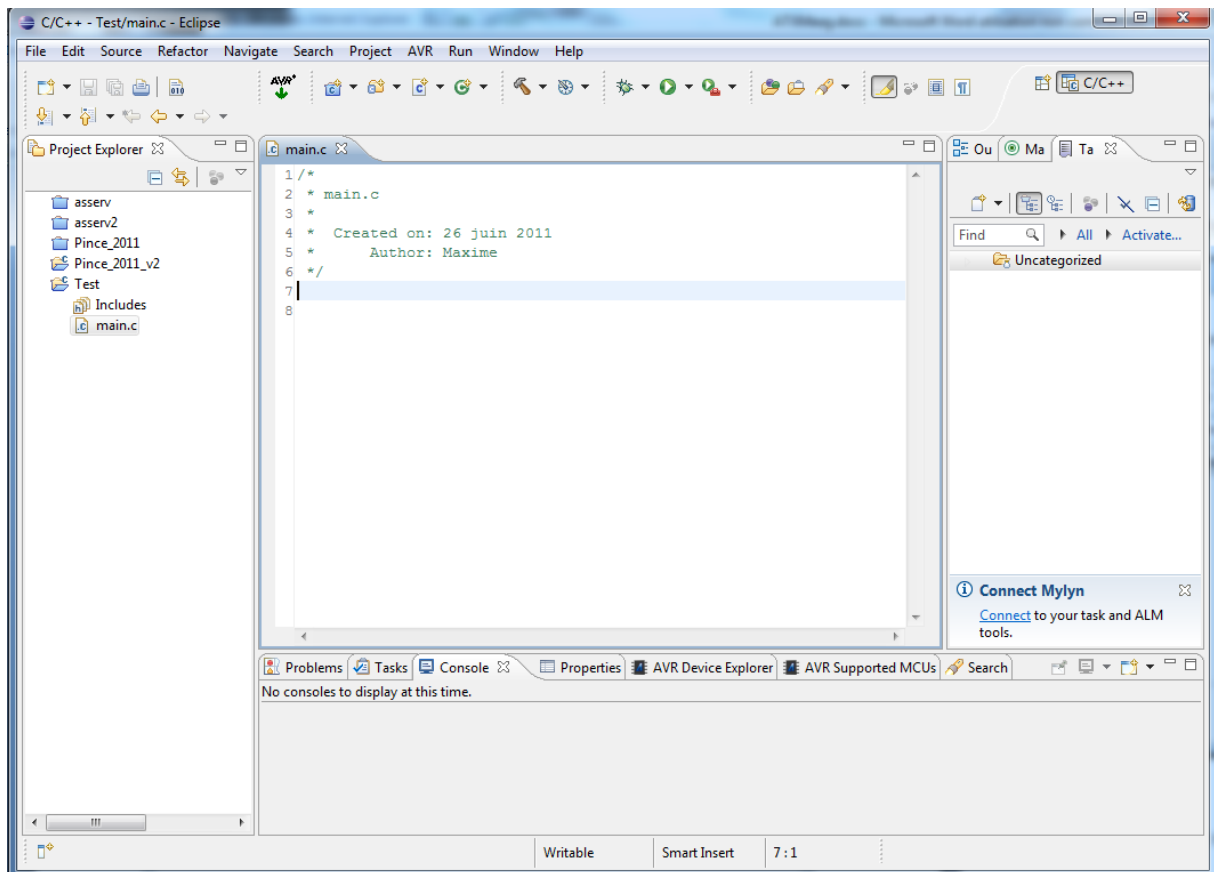
Cliquez avec le bouton droit dessus, new -> C source file.

Entrez les informations suivantes, puis cliquez sur Finish.



Votre écran doit ressembler à ceci (sans autant de projets à gauche !) :





C'est parti, on peut maintenant taper notre premier programme.

Juste une petite chose : Je vous conseille de télécharger les notes d'application de Atmel. Elles expliquent comment utiliser les différents périphériques et fournit les drivers qui nous seront utiles.

Sur cette page,

[http://www.atmel.com/dyn/products/documents.asp?category\\_id=163&family\\_id=607&subfamily\\_id=1965](http://www.atmel.com/dyn/products/documents.asp?category_id=163&family_id=607&subfamily_id=1965) vous trouverez une liste de documents. Le petit CD signifie que des exemples de programmes et des drivers sont téléchargeables et associés au document.

Commencez par télécharger celle-ci : Il s'agit du driver des entrées-sorties.



**AVR1313: Using the XMEGA IO Pins and External Interrupts**  
(9 pages, revision A, updated 2/08)  
This application note describes the highly configurable XMEGA I/O pins and external interrupts. A driver interface written in C is included as well.

Par la suite, quand vous utiliserez d'autres périphériques, télécharger ce qui lui correspond.

**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

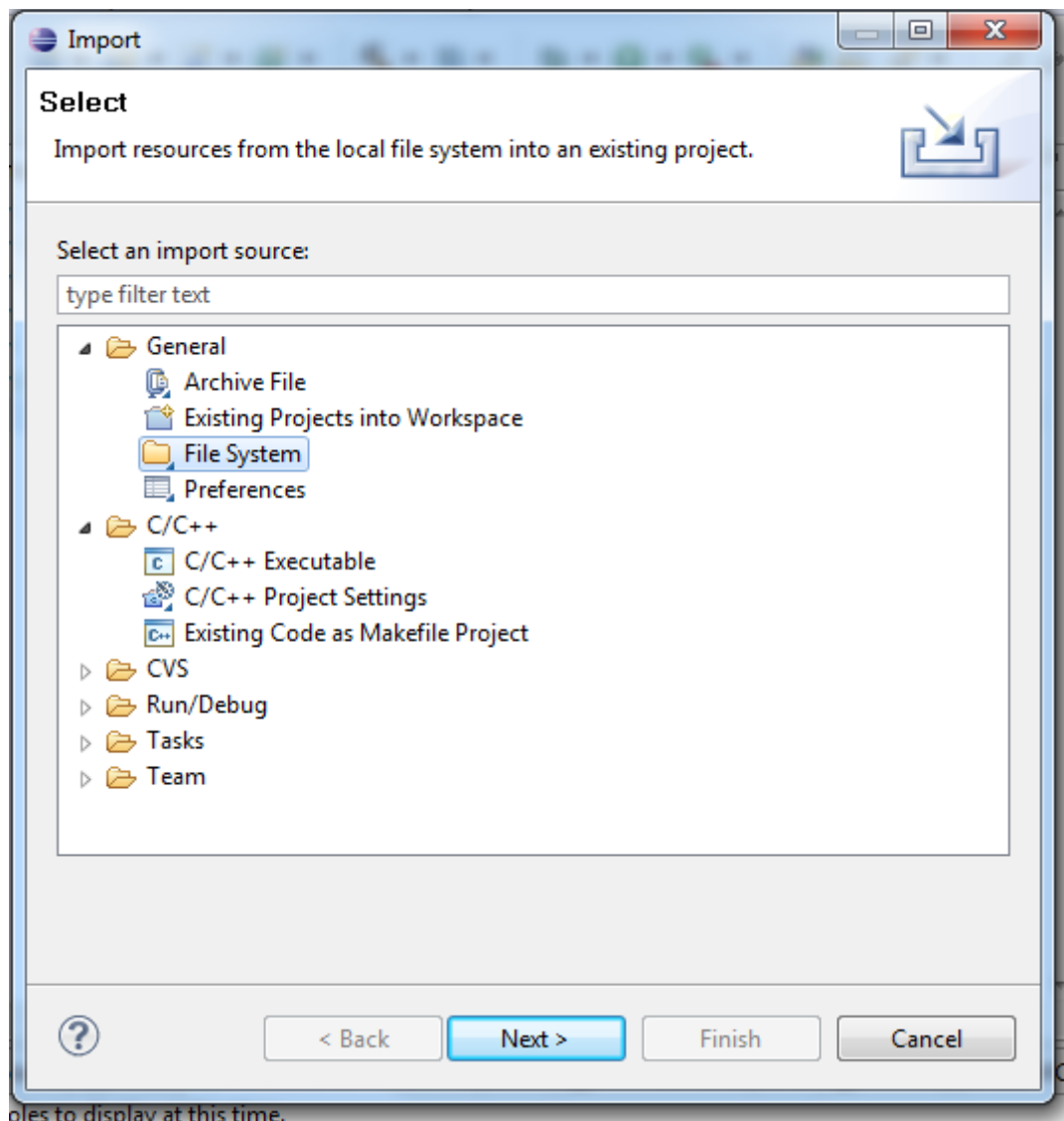
<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

Une fois les différents fichiers extraits, nous allons rajouter les drivers au projet pour pouvoir les utiliser.

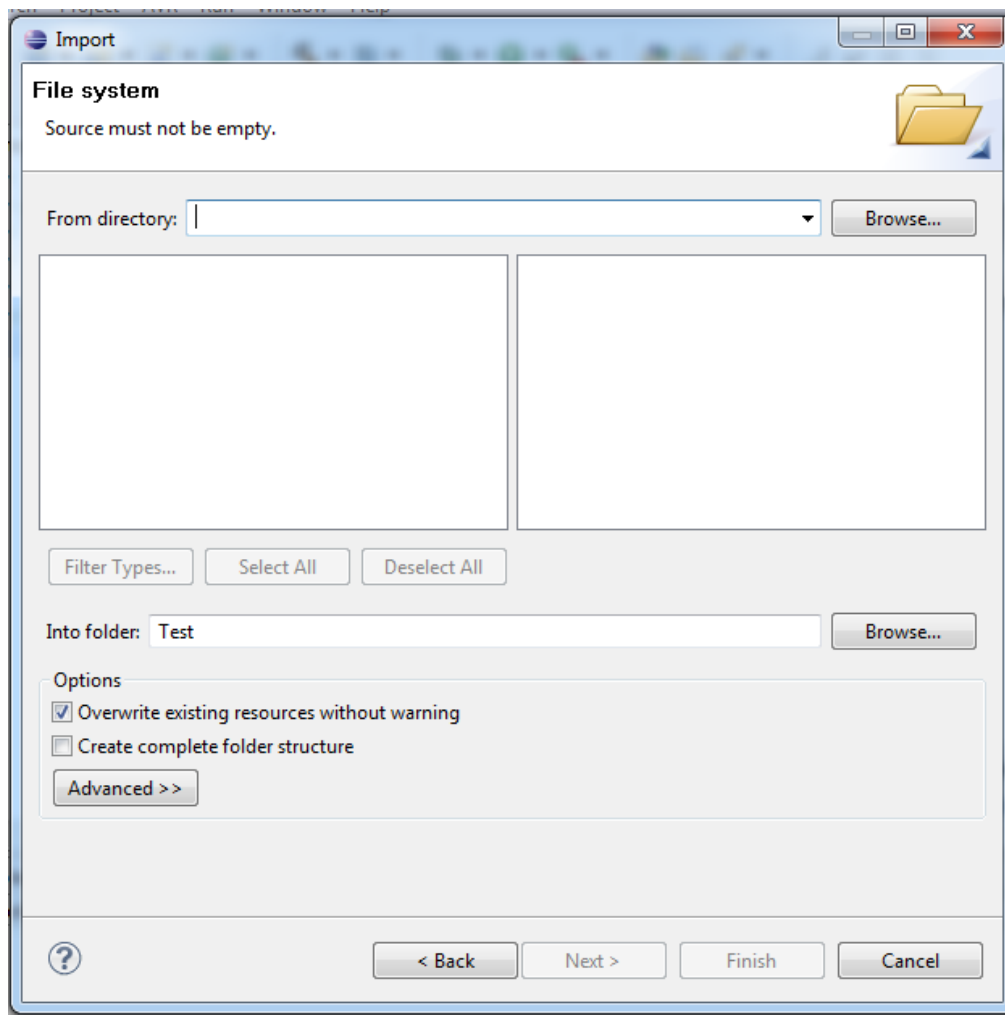
Pour cela :

Clic droit sur Test puis cliquez sur Import.

Cette fenêtre s'ouvre.



Sélectionnez File System puis cliquez sur Next.



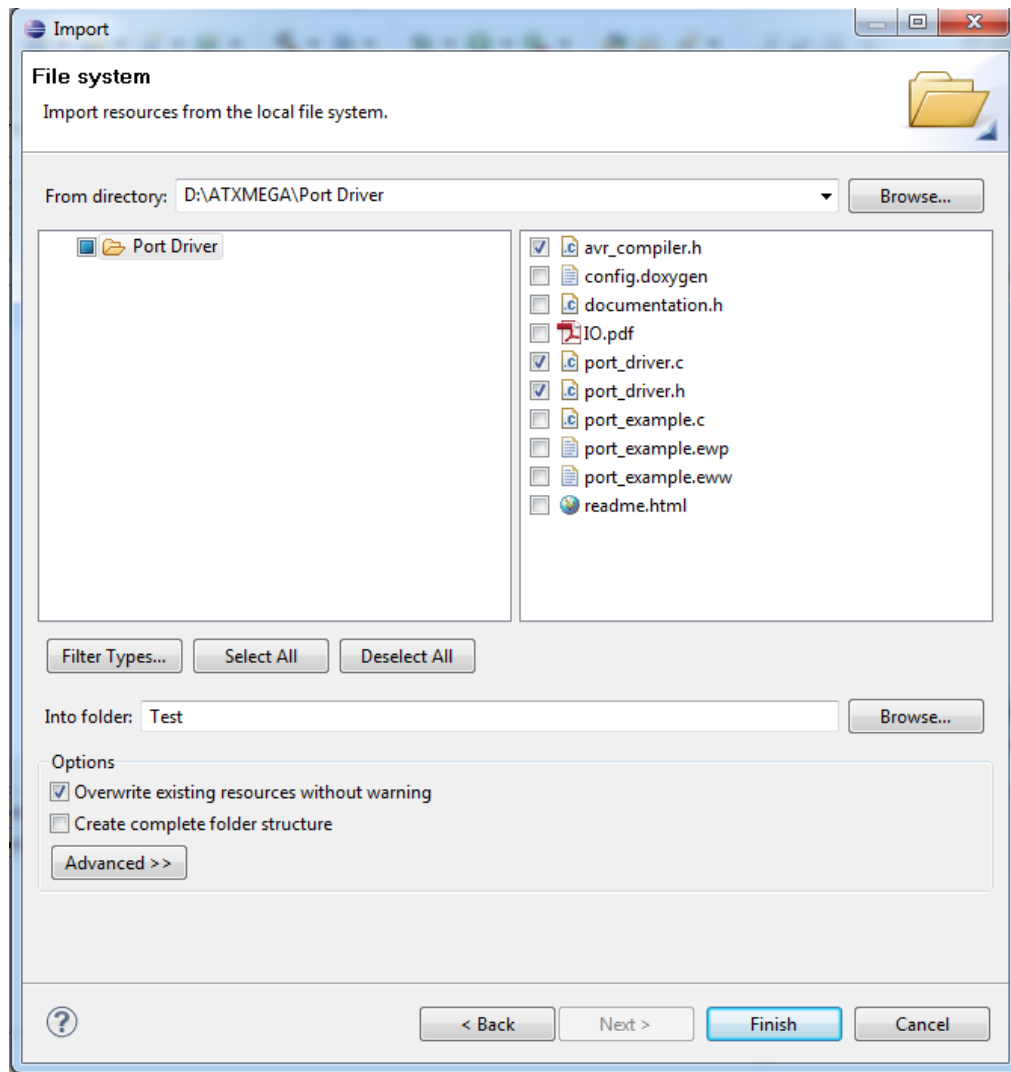
Cliquez sur Browse puis allez chercher le dossier contenant les fichiers téléchargés.

Nous avons besoin de 3 fichiers :

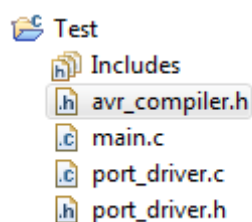
- port\_driver.c et port\_driver.h qui sont, comme le nom l'indique, les drivers pour utiliser les ports d'entrées-sorties de la puce.

- avr\_compiler.h contient des définitions liées au processeur et au compilateur. On en a besoin pour notre programme.

Notez la présence de IO.pdf qui explique tout le fonctionnement des entrées-sorties de notre processeur.



Cliquez sur Finish. Votre Project explorer doit ressembler à ceci :



Nous allons maintenant écrire notre premier programme. Je ne le détaille pas ici. Il s'agit juste d'un test pour vérifier l'installation des différents logiciels et du matériel.

Copiez-collez ceci dans main.c :

```
#include "avr_compiler.h"
#include <avr/io.h>
#include "port_driver.h"

int main(void)
{
    unsigned int i=0; //Variables ce comptage (de 0 à 65535)


    //Configuration de la patte de la LED en sortie
    PORT_SetPinAsOutput(&PORTQ, 0b00000100); //La LED est branchée sur la
    //patte 2 du PORTQ

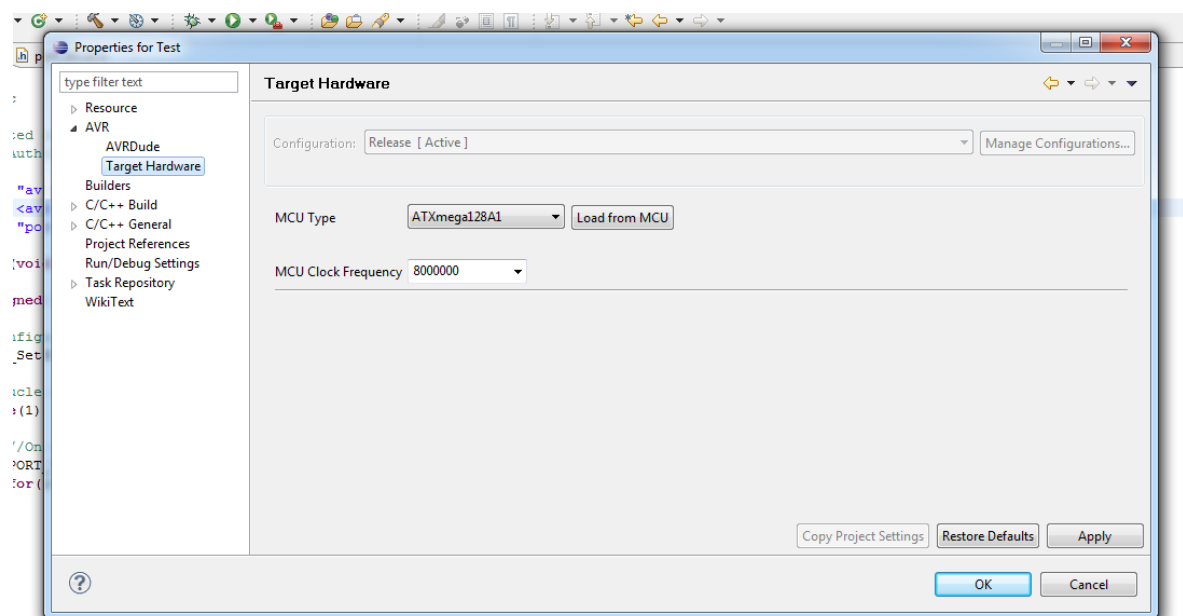
    //Boucle infinie
    while(1)
    {
        //On change l'état de la LED(allumé devient éteint et
        //inversement)
        PORT_TogglePins(&PORTQ, 0b00000100);
        for(i=0;i<10000;i++);
    }
}
```

Nous allons maintenant dire au compilateur quel processeur nous utilisons. En effet, ces logiciels permettent de programmer des dizaines de processeurs Atmel.

Pour cela, nous allons à nouveau ouvrir la fenêtre des propriétés du projet :

Clic droit sur le nom du projet dans le project explorer->Proprieties

Cliquez ensuite sur la petite flèche à côté de AVR :  et enfin sur Target Hardware. Vous obtenez la fenêtre suivante :



**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

MCU Type : Sélectionnez ATXmega128A1 dans le menu déroulant.

MCU Clock Frequency : 8000000

Pour terminer, cliquez sur OK.

Maintenant arrive un grand moment : Nous allons compiler le programme, c'est-à-dire transformer les fichiers textes (.c et .h) en instructions exécutables par le microcontrôleur.

**Avant toute compilation, il faut impérativement sauvegarder tous les fichiers que vous avez modifiés si vous voulez que vos modifications soient prises en compte.**

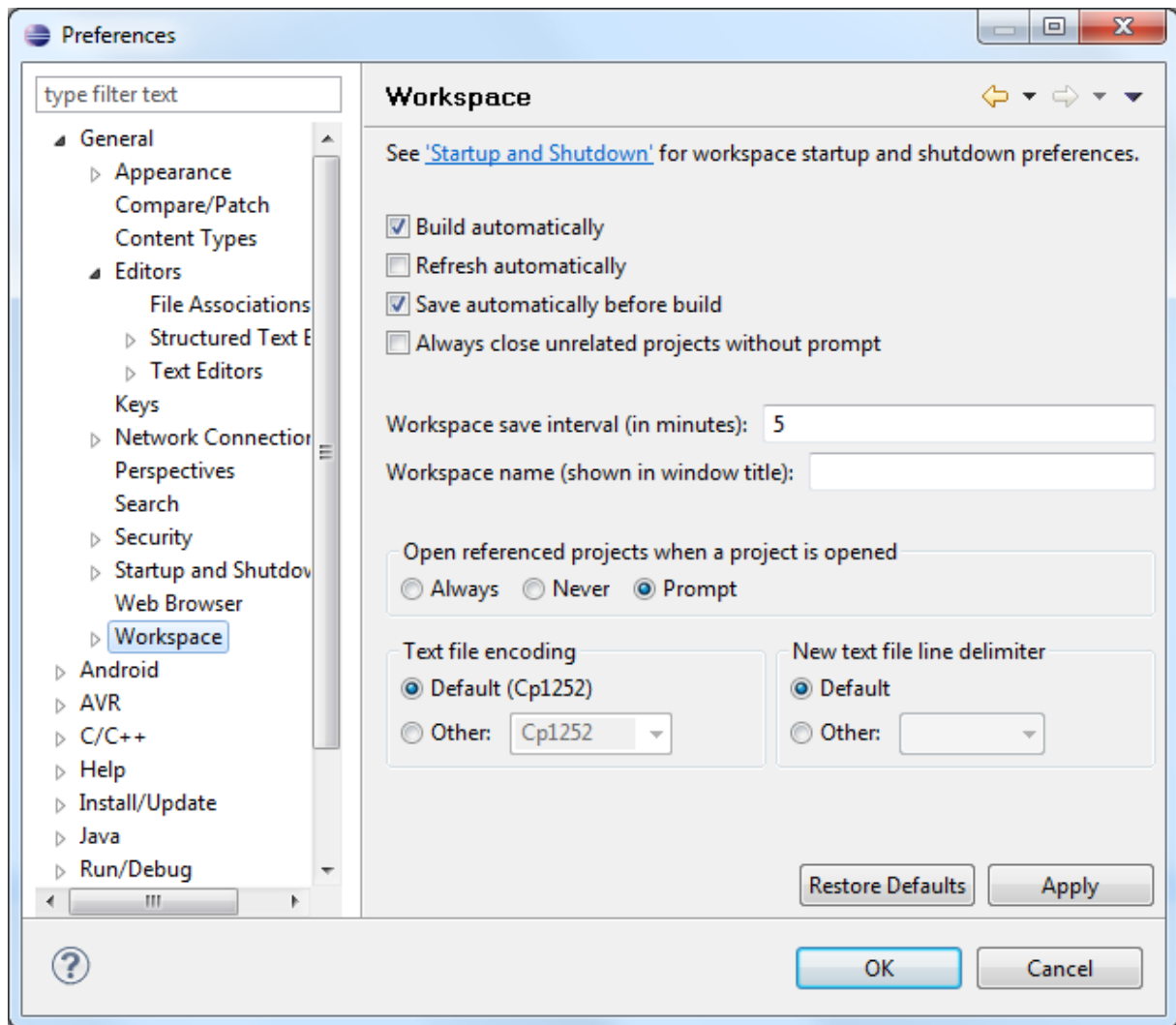
Pour cela, deux icônes existent :  sauvegarde uniquement le fichier affiché,




sauvegarde tous les fichiers modifiés.

Eclipse propose quand même de sauvegarder automatiquement avant la compilation en activant l'option concernée :

Allez dans Window->Preferences->General puis cliquez sur l'onglet Workspace.  
Cocher la case « Save automatically before build »

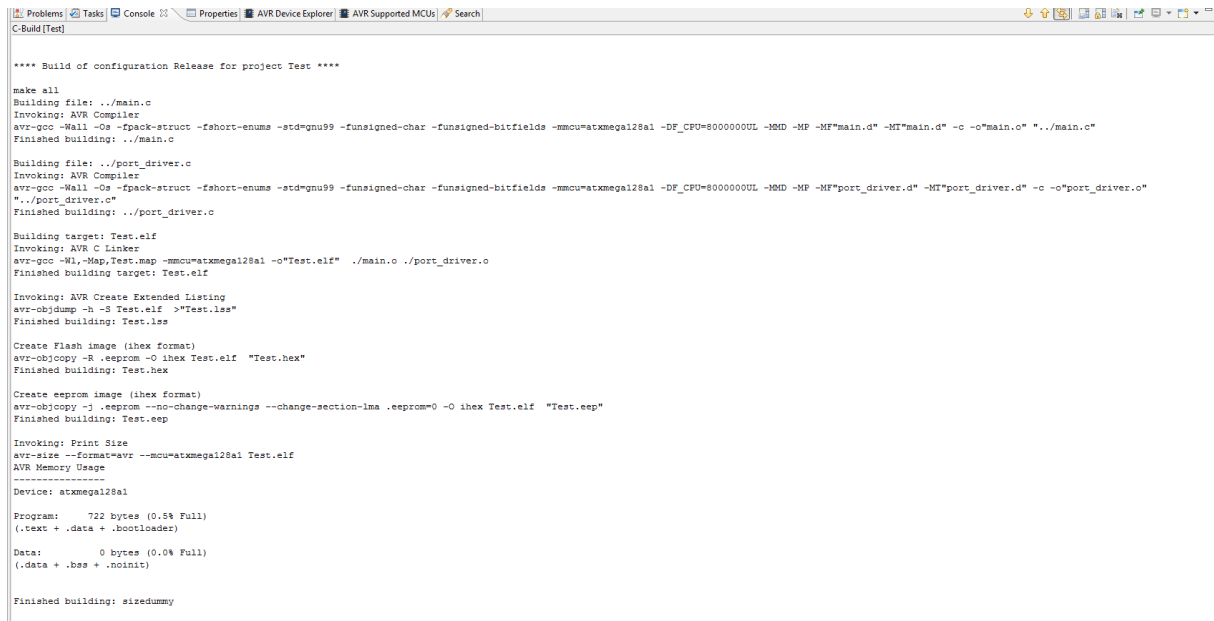


Cliquez sur le marteau en haut de l'IDE (  ).

Du texte doit défiler en bas de l'écran. Il s'agit d'informations sur l'avancement de la compilation.

Une fois terminée, la console doit ressembler à cela :





```
**** Build of configuration Release for project Test ****

make all
Building file: ../main.c
Invoking: AVR Compiler
avr-gcc -Wall -Os -fpack-struct -fshort-enums -std=gnu99 -funsigned-char -funsigned-bitfields -mmcu=atxmega128a1 -DF_CPU=8000000UL -MD -MP -MF"main.d" -MT"main.d" -c -o"main.o" ../main.c
Finished building: ../main.o

Building file: ../port_driver.c
Invoking: AVR Compiler
avr-gcc -Wall -Os -fpack-struct -fshort-enums -std=gnu99 -funsigned-char -funsigned-bitfields -mmcu=atxmega128a1 -DF_CPU=8000000UL -MD -MP -MF"port_driver.d" -MT"port_driver.d" -c -o"port_driver.o"
../port_driver.c
Finished building: ../port_driver.o

Building target: Test.elf
Invoking: AVR C Linker
avr-gcc -Wl,-Map,Test.map -mmcu=atxmega128a1 -o"Test.elf" ../main.o ../port_driver.o
Finished building target: Test.elf

Invoking: AVR Create Extended Listing
avr-objdump -h -S Test.elf >"Test.lss"
Finished building: Test.lss

Create Flash image (ihex format)
avr-objcopy -R .eeprom -O ihex Test.elf "Test.hex"
Finished building: Test.hex

Create eeprom image (ihex format)
avr-objcopy -j .eeprom --no-change-warnings --change-section-lma .eeprom=0 -O ihex Test.elf "Test.eep"
Finished building: Test.eep

Invoking: Print Size
avr-size --format=avr --mcu=atxmega128a1 Test.elf
AVR Memory Usage
-----
Device: atxmega128a1

Program:   722 bytes (0.5% Full)
(.text + .data + .bootloader)

Data:      0 bytes (0.0% Full)
(.data + .bss + .noinit)

Finished building: sizedummy
```

Le plus important est ceci : Signifie que la compilation est réussie.

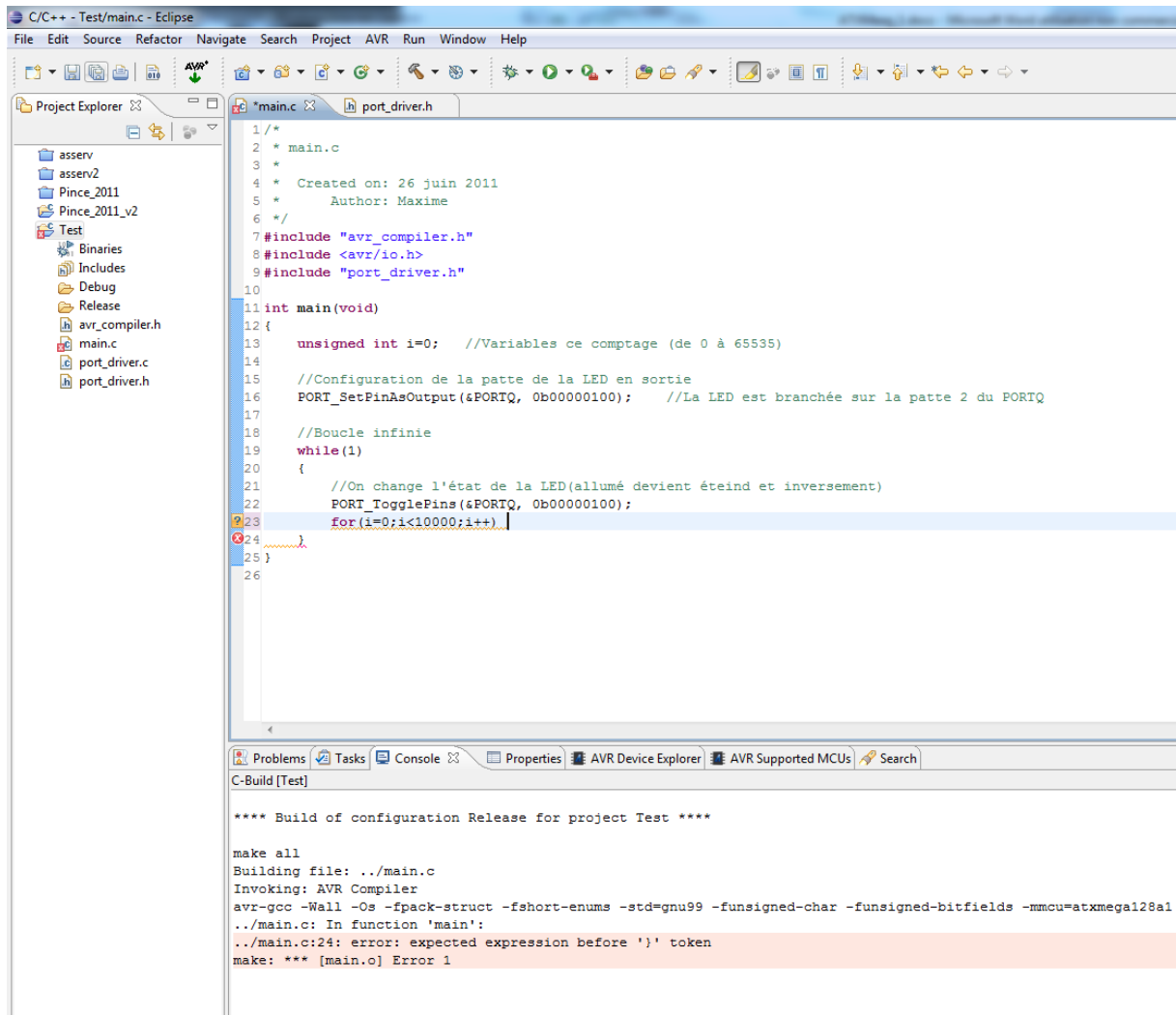
```
Device: atxmega128a1

Program:   722 bytes (0.5% Full)
(.text + .data + .bootloader)

Data:      0 bytes (0.0% Full)
(.data + .bss + .noinit)

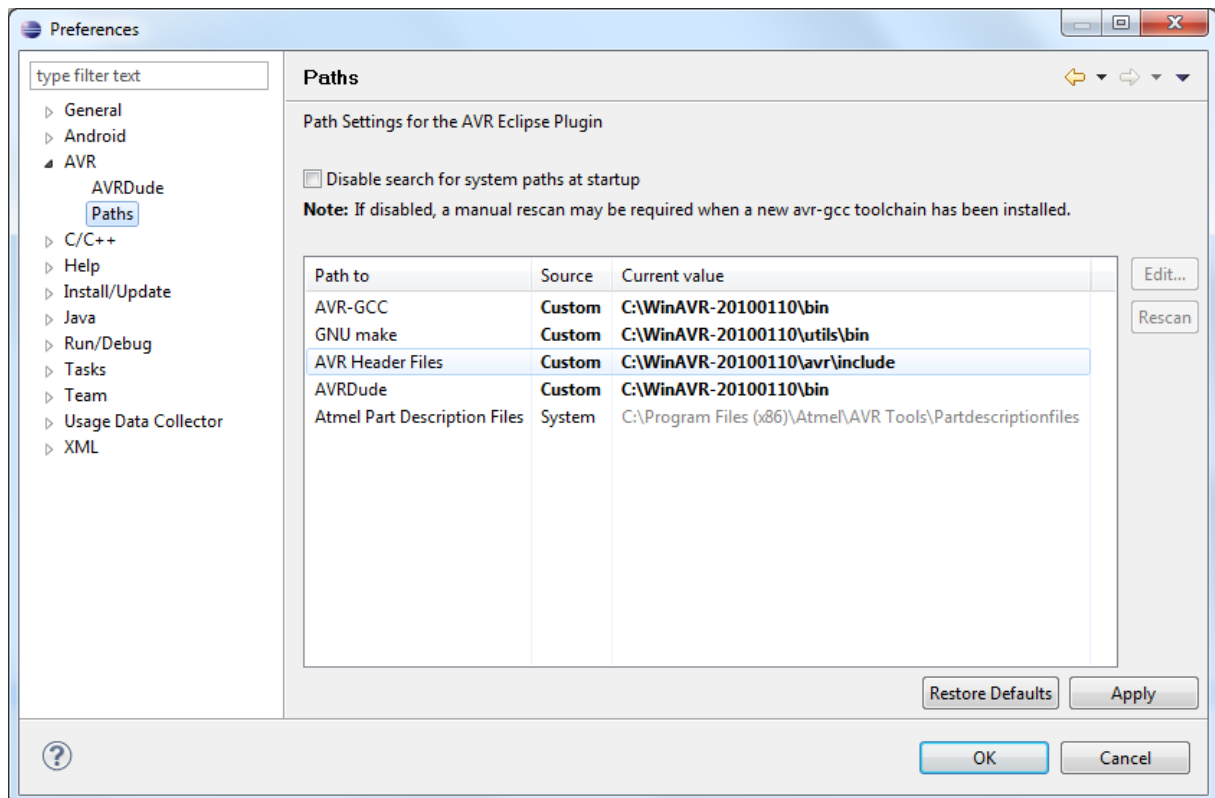
Finished building: sizedummy
```

Si une erreur est détectée, sa description s’affichera dans la console et au moins une croix rouge apparait dans le project explorer : Voici un exemple en cas d’erreur :



Ici, il s'agit d'une erreur de syntaxe : j'ai juste supprimé le point-virgule après le for (i=0 ;i<1000 ;i++).

Si la compilation ne se déclenche pas, vérifier les chemins d'accès vers les utilitaires WinAVR : Window->Preferences->AVR->Paths



## Le programmeur

Maintenant, le programme est prêt à être transféré dans le microcontrôleur.

Cette opération est réalisée par un programmeur. Il relie le PC à la puce.



Nous utilisons au club l'AVR ISP2 en photo ci-dessus.

Branchez le programmeur à l'ordinateur par USB.

Le PC va chercher le pilote. Vous devez l'installer manuellement : Annulez la recherche de Windows update et sélectionnez le dossier où vous avez installé WinAVR au début.

**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

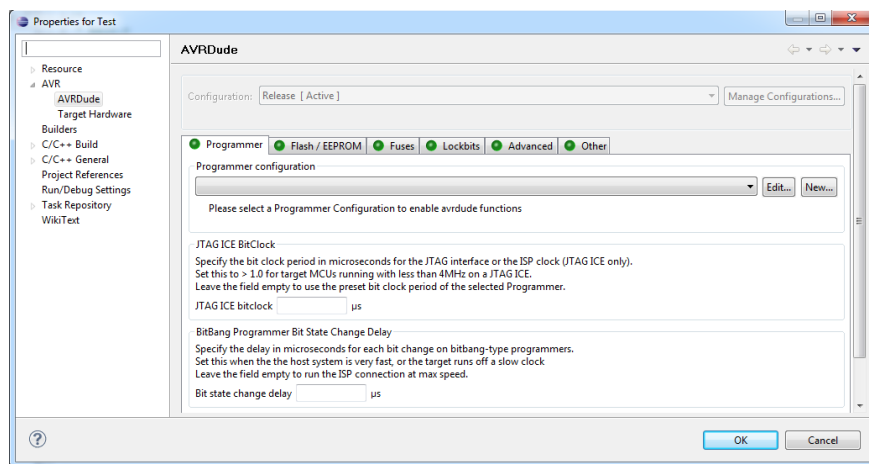
<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

Si Windows se plaint de la non-signature du pilote, installez le quand-même. Si après cela, le périphérique n'est pas reconnu, téléchargez le .zip sur la même page que ce document. Exécutez ensuite install\_x86 ou install\_x64 selon votre version de Windows.

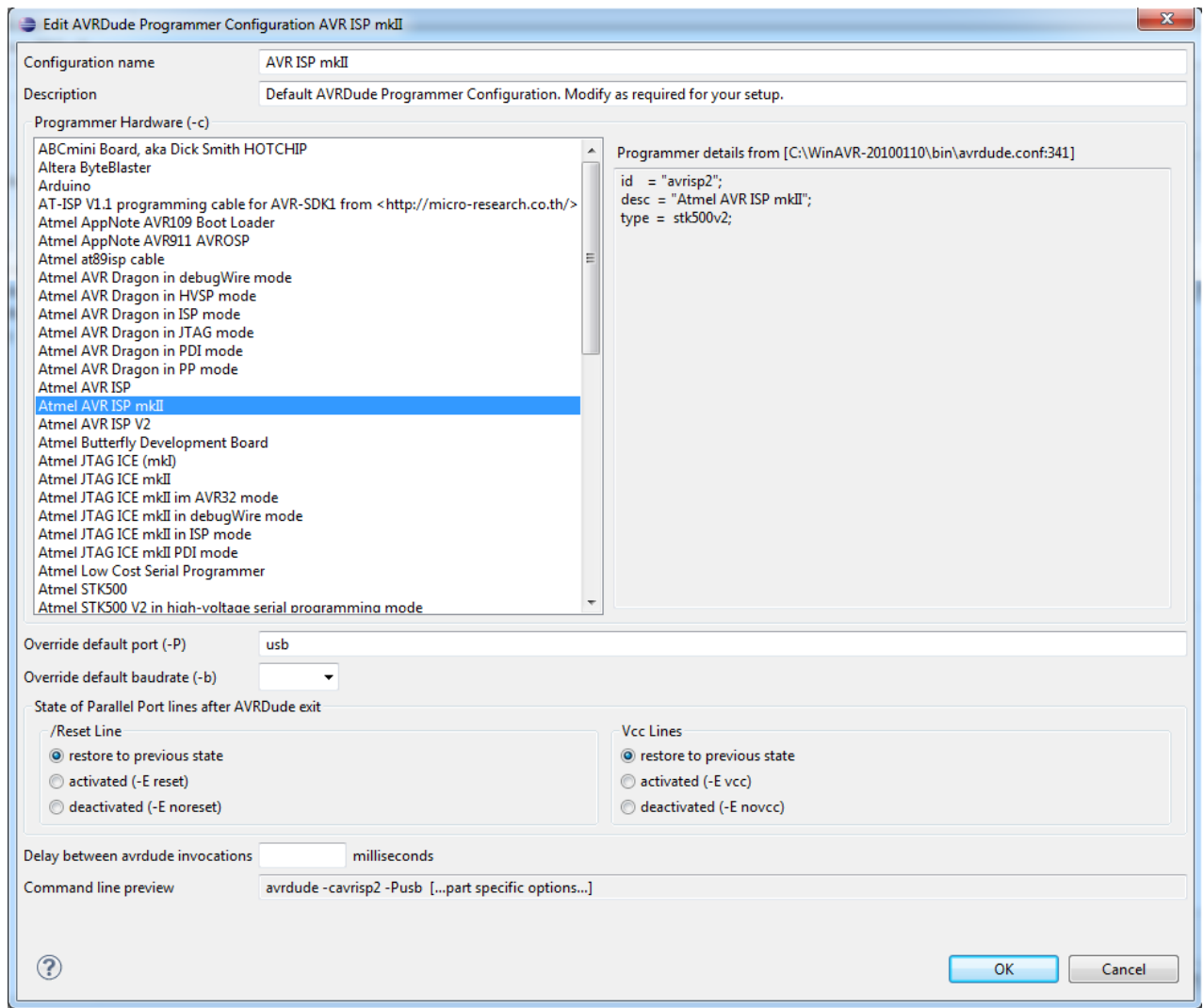
Ensuite nous allons régler Eclipse et AVRdude pour utiliser ce programmeur :

Comme lors du choix du processeur, ouvrez les propriétés du projet.

Allez en suite dans la rubrique AVR, cliquez sur la flèche et enfin sur AVRdude

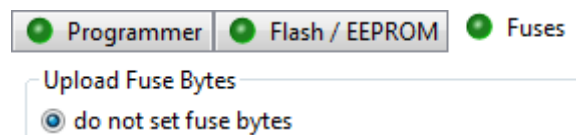


A côté du menu déroulant « Programmer configuration », cliquez sur new et entrez ces paramètres :

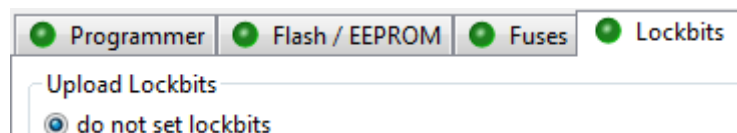


Ensuite validez en cliquant sur OK.

Dans l'onglet Fuses Bytes, sélectionnez « do not set fuses bytes »



Dans l'onglet lockbits, idem :



Validez le tous en cliquant sur OK.

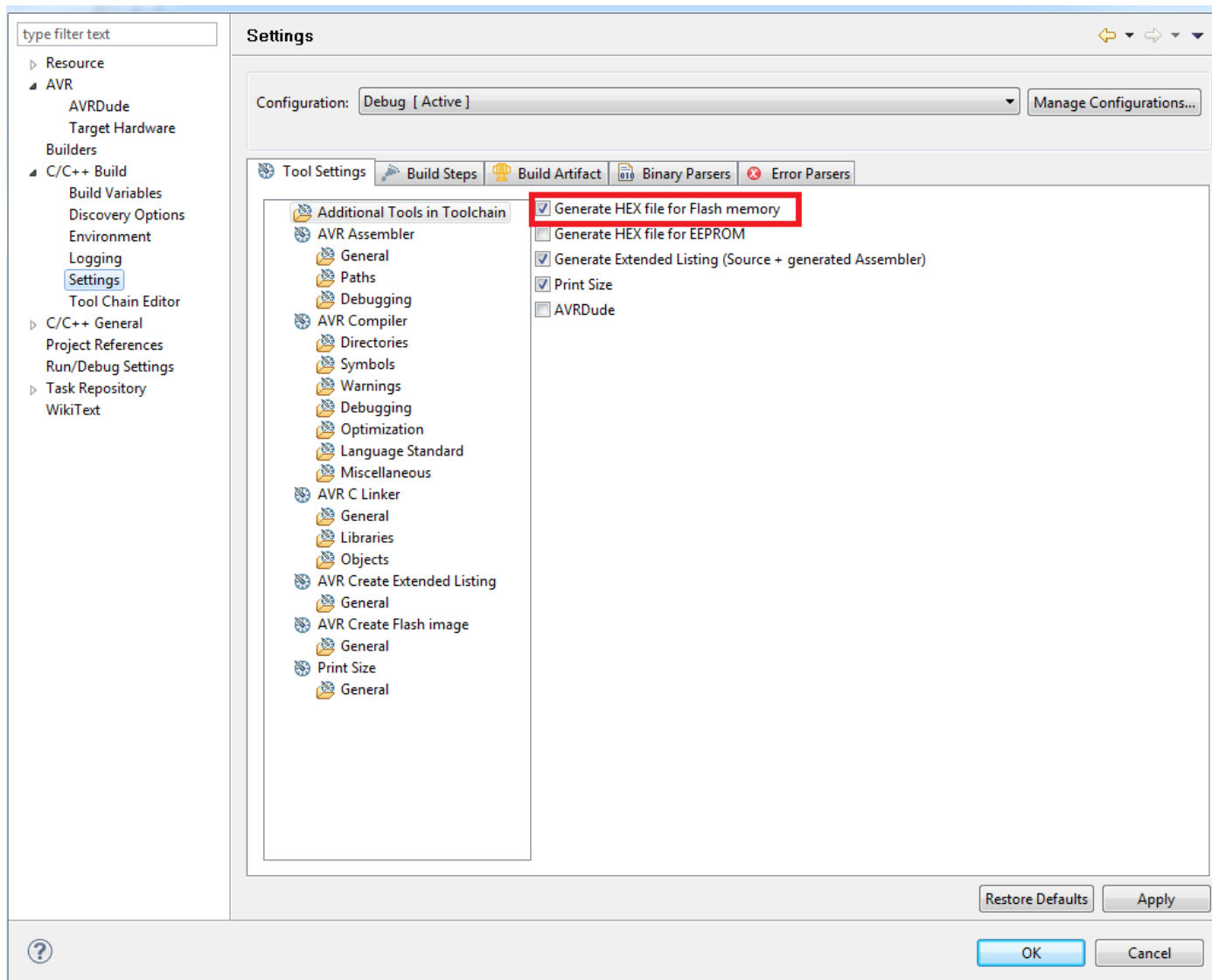
A présent, il faut paramétrer le linker pour que le fichier généré lors de la compilation soit un .hex et non un .elf.

**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

Pour cela, ouvrez les propriétés du projet -> C/C++ Build -> Settings.



Cochez « Generate HEX file for Flash memory » comme sur l'image puis cliquez sur OK.

Ensuite, recompilez le projet. Dans la console, le message « Finished building: Asserv.hex » doit apparaître.

## Le Final

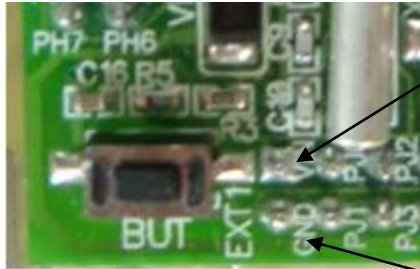
**Club Robotique de l'INSA Strasbourg**

24, Boulevard de la Victoire – 67084 STRASBOURG Cedex

<http://www.insa-strasbourg.fr/club-robotique/> – [club-robotique@insa-strasbourg.fr](mailto:club-robotique@insa-strasbourg.fr) - 06 80 52 20 77

Moment crucial, nous allons maintenant programmer le microcontrôleur (enfin) !

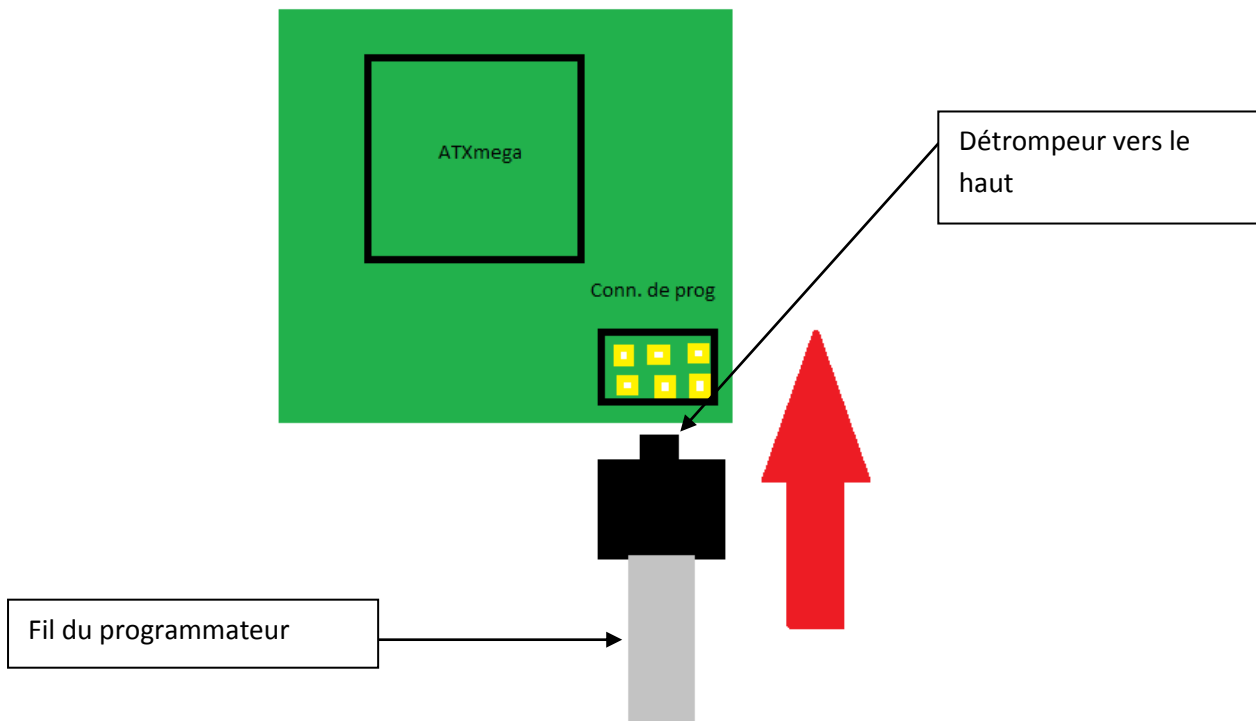
\*) L'alimentation du module : Le programmeur n'alimente pas la carte. Elle a besoin d'une alimentation externe pour pouvoir fonctionner, qui se branche entre VIN et GND. Elle se fait par le circuit imprimé de l'application.



**VIN(+) 12V Max**

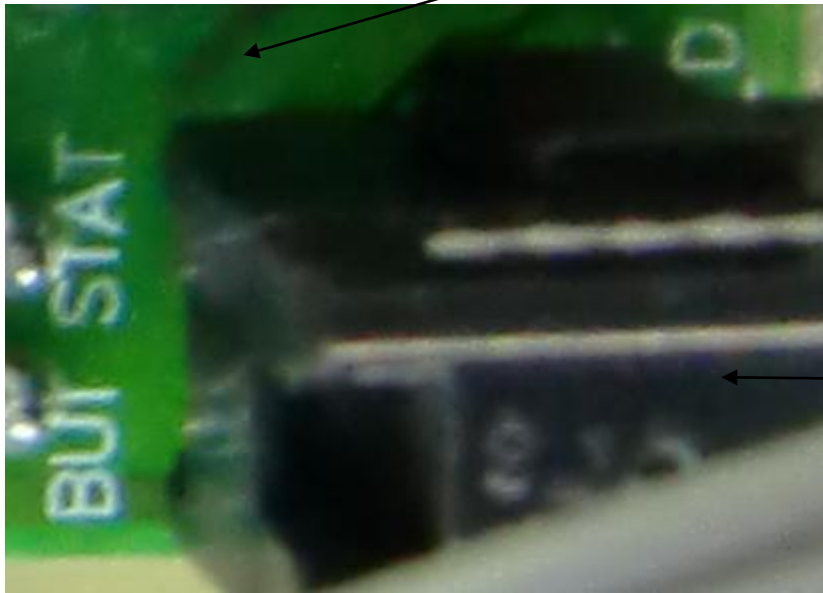
**GND(-)**

\*) Connexion du programmeur :





Platine



Connecteur du  
programmeur



Une fois tous les éléments connectés, cliquez là-dessus et le programme est envoyé dans l'ATXMega. Dès que la programmation est terminée, le programme s'exécute immédiatement. C'est gagné !

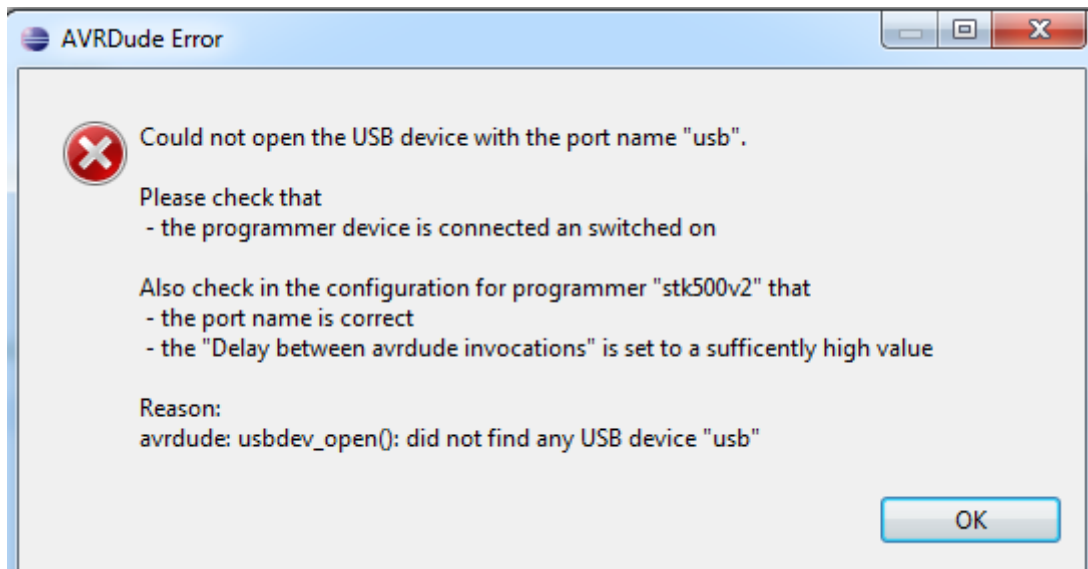
En cas d'erreur, ne vous énervez pas tout de suite, dans un grand nombre de cas, il s'agit d'une simple faute. (Erreur lors du suivi des instructions, oubli d'un ; ...)

Vérifiez bien vos réglages :

-> En cas de problèmes de compilation, si vous avez une erreur du type « ; or } or... expected before... » C'est que vous avez fait une erreur de syntaxe. Le compilateur ne reconnaît pas ce que vous avez écrit.

-> « could not find definition of... » Signifie qu'un mot utilisé n'est pas reconnu par le compilateur : faute de frappe dans les instructions, variables non déclarées...

-> Si jamais vous ne pouvez pas programmer, qu'une fenêtre de ce type apparaît lorsque vous tentez de programmer, vérifiez les réglages du programmeur.



Si tout est bon mais que ça ne fonctionne pas, allez dans le gestionnaire des périphériques et vérifiez que le programmeur est identifié dans une catégorie « Win32 ». Mettez le pilote à jour ou carrément réinstallez le périphérique. N'oubliez pas : Vous devez sélectionner le pilote dans le dossier WINAvr !