




Fiche guide M3	TS SI		P.P.E. Mini serre	
Mesure	2h			
 Lycée Polyvalent PIERRE EMILE MARTIN	Mesurer et afficher un taux humidité			

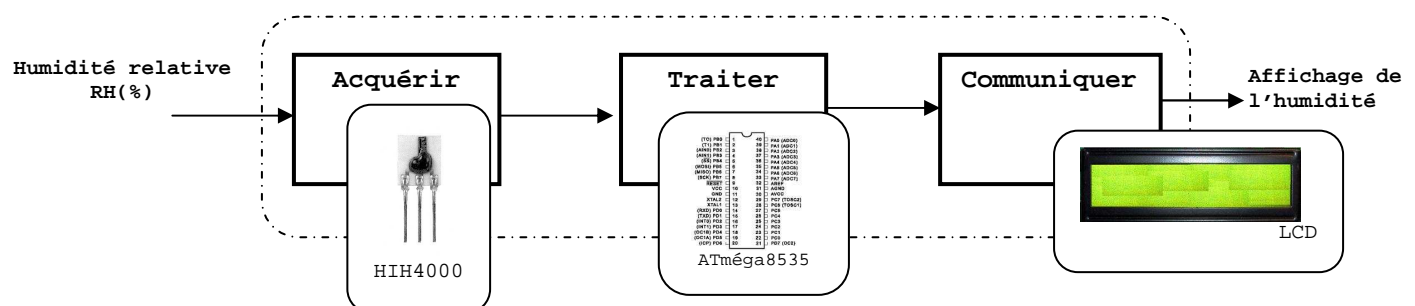
Nom :	Classe :	Groupe :
-------	----------	----------

Objectif : Etre capable de mesurer l'humidité dans la mini serre et l'afficher sur un LCD.

<p>Matériels Carte ATMESSSI V1 + Module capteurs + Alimentation 10V.</p> <p>Logiciels CodeVisionAvr.</p> <p>Documentation Schémas de la carte ATMESSSI et de la carte Capteurs. Documentation technique du capteur d'humidité HIH4000 et de l'afficheur LCD à processeur Hitachi.</p> <p>Le présent document et la documentation technique des composants sont <u>téléchargeables</u> sur le site <u>WebGE</u> à l'adresse http://p.mariano.free.fr/ (rubrique PPE)</p>
--

A) Présentation

On souhaite afficher l'humidité ambiante sur un LCD*. Pour cela, on propose de mettre en œuvre une structure correspondant à la chaîne d'information donnée ci-dessous.





La fonction « Acquérir » est réalisée par un capteur d'humidité (HIH4000) et le convertisseur analogique numérique du microcontrôleur (ATmega8535). La fonction « Traiter » est assurée par un programme implanté dans le microcontrôleur. La fonction « Communiquer » est remplie par un afficheur LCD (à processeur Hitachi).

L'ensemble des structures matérielles étant réunies sur la carte « ATMESSSI » et la carte « Capteurs », votre travail va se limiter à la réalisation du logiciel à implanter dans le microcontrôleur.

Pour cela, vous allez créer et configurer un projet avec le magicien du cross-compileur **CodeVisionAVR**. Puis, vous complèterez cette structure avec les fonctions nécessaires à la mise en œuvre du capteur et de l'afficheur.

La suite de ce document décrit le travail à réaliser étape par étape. A la fin de cette activité, vous serez capable de mesurer et d'afficher l'humidité relative dans la mini serre.

*LCD : Display Liquid Crystal

 Lycée Polyvalent PIERRE EMILE MARTIN	FGM3	Mesures		PPE Mini Serre	1
--	------	---------	---	----------------	---

B) Travail demandé

Etape 1 : Création et configuration d'un projet

Lancez le logiciel CodeVisionAVR.

(1) Création d'un nouveau projet

Dans la barre d'outils : « **File** » puis « **New** » pour obtenir la boîte de dialogue ci-contre.
Cochez « **Project** » puis clic sur « **Ok** »

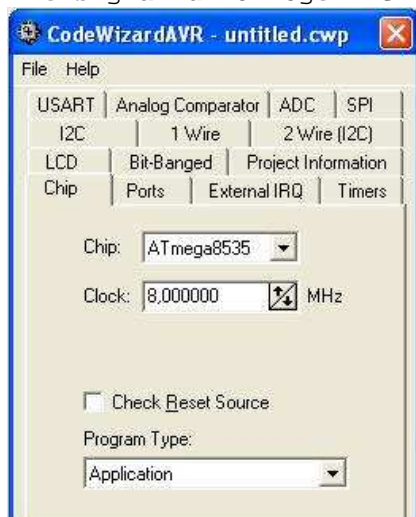


Ici « **Yes** »



(2) Sélection du composant cible

La boîte du « **Magicien** » ci-dessous s'ouvre. Choisissez le « **Chip** » ATmega8535 et réglez le signal d'horloge « **Clock** » à 8Mhz.

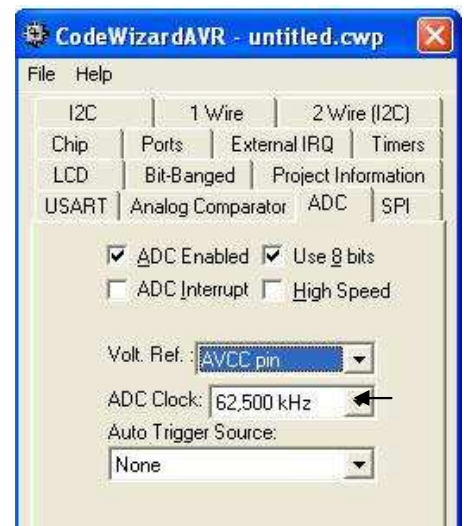


(3) Configuration du convertisseur analogique numérique

Sélectionnez l'onglet « **ADC** ».

Il est nécessaire de cocher la case « **ADC Enabled** » et de régler l'horloge « **ADC Clock** » à 62,5kHz pour utiliser le convertisseur analogique numérique du microcontrôleur.

Précisez également que le résultat de la conversion doit être fourni sur **8 bits**.



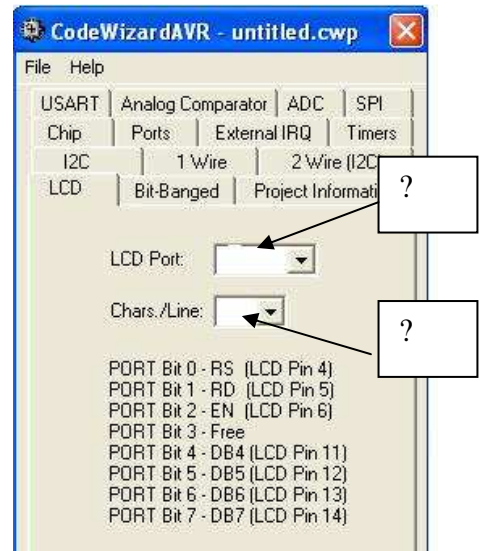
(4) Choix de l'affichage

Sélectionnez l'onglet « LCD ».

Pour obtenir l'organisation ci-contre, il est nécessaire de remplir le champ « LCD Port ».

En étudiant le schéma de la carte « ATMELSSI V1 », et la documentation de l'afficheur LCD, **déterminez** sur quel port est connecté l'afficheur et son nombre de caractères par ligne.

Configurez les champs « LCD Port » et « Chars./Line » de la boîte de dialogue « LCD ».



(5) Enregistrement du projet



Sélectionnez « Program Preview ».

Si le projet est correctement configuré, le début du fichier source du programme doit correspondre au texte ci-dessous.

```
#include <mega8535.h>

// Alphanumeric LCD Module functions
#asm
.equ __lcd_port=0x15
#endasm
#include <lcd.h>

#define ADC_VREF_TYPE 0x60
// Read the 8 most significant bits
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
{
    ADMUX=adc_input/ADC_VREF_TYPE;
    // Start the AD conversion
    ADCSRA|=0x40;
    // Wait for the AD conversion to complete
    while ((ADCSRA & 0x10)==0);
    ADCSRA|=0x10;
    return ADCH;
}
```

Fermez la fenêtre.

Sélectionnez

→ File

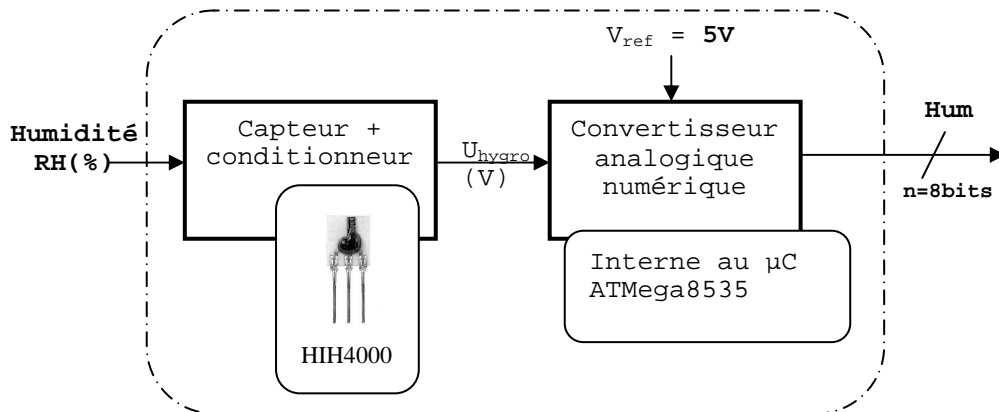
→ **Generate, save and Exit**

Donnez le nom **Humide** à votre projet (3 fois) pour créer les trois fichiers à la base du projet. (.c, .prj, .cwp)

ATTENTION : Le Magicien ne peut plus être utilisé pour modifier votre projet. Voir le prof pour d'éventuelles corrections.

Etape 2 : Etude de la fonction « Acquérir »

La fonction « Acquérir » peut être représentée par le schéma ci-dessous :

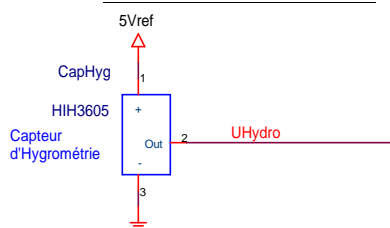


Remarque : Hum est une variable de type entier non signée codée sur un octet.

Le programme à réaliser doit tenir compte des **coefficients** introduits par les structures associées aux boîtes fonctionnelles ci-dessus. Ces coefficients sont déterminés dans l'étude qui suit.

- **Expression de la tension U_{hygro} en fonction de l'humidité ambiante (notée RH)**
L'humidité est mesurée par un capteur HIH4000. Cette structure délivre une différence de potentiel U_{hygro} proportionnelle à l'humidité relative ambiante.

o Schéma structurel



Capteur d'hygrométrie HIH4000



- out +

- o **Spécifications techniques du capteur d'humidité**
Tension d'alimentation : 5 V c.c.
Plage de mesure : 0 à 100%
Précision : +/- 2%
Stabilité : +/- 1% RH (à 50% sur 5 ans)
Temps de réponse : 15 s (air en déplacement lent)
Tension de sortie : 0,8 à 3,9 V c.c.
Température d'utilisation : -40°C à +85°C

La droite nommée « **Typical best fit straight line** » située dans la documentation du capteur d'humidité HIH4000 permet d'exprimer U_{hygro} en fonction de l'humidité.

U_{hygro} peut se mettre sous la forme $U_{hygro(V)} = a.RH + b$.

Déterminez les coefficients a et b à partir de la documentation du capteur.

- **Expression de Hum en fonction de l'humidité ambiante notée RH**
Mettez l'expression de Hum sous la forme $Hum = a' \cdot RH + b'$ sachant que
 $Hum = (2^n / V_{ref}) \cdot U_{hygro}$

Note : Les coefficients a' et b' seront utilisés dans le programme à réaliser.

Etape 3 : Mesure et affichage de l'humidité

Dans ce paragraphe, vous allez compléter la **partie déclarative**...

```
void main(void)
{
// Declare your local variables here
```

...et la partie **exécutive** du programme.

```
while (1)
{
// Place your code here

};
```

On rappelle que la **partie déclarative** d'un programme est la zone dans laquelle sont **créées les variables** alors que la **partie exécutive** est la zone de **traitement** de ces variables.

Malgré la complexité des structures à mettre en œuvre, le programme à réaliser reste relativement simple. Ceci est dû à la « richesse » des **bibliothèques de fonctions** fournies avec le cross-compileur CodeVisionAVR.

Par exemple, la lecture de l'information fournie par le capteur de luminosité s'effectue avec la fonction :

read_adc(?)

? correspond à l'entrée sélectionnée sur le port A du microcontrôleur.
 Cette fonction a été introduite dans le programme par le Magicien (voir page 3 de ce document).

L'écriture sur l'afficheur LCD nécessite la fonction `sprintf()`. Cette fonction est située dans la bibliothèque `stdio` accessible par une **référence** au fichier `<stdio.h>`.

On fait référence à une bibliothèque avec la directive **`#include`**.

(1) Déclaration des bibliothèques de fonctions utilisées dans le programme

Vous **devez rajouter** une référence à la bibliothèque `stdio` à la suite de `#include <mega8535.h>` dans le fichier source. Vous ajouterez également une référence à la bibliothèque `stdlib` (nécessaire pour convertir un nombre réel en une chaîne de caractères) et à la bibliothèque `delay` (nécessaire pour réaliser une temporisation).

(2) Partie exécutive du programme à réaliser

Le programme à réaliser peut être résumé par les actions ci-dessous :

Lire (humidité)
Traitement (Appliquer les coefficients introduits par les structures à la mesure d'humidité)
Ecrire (humidité)

o Acquisition de l'humidité

L'humidité est lue par l'expression ci-dessous :

`Humid = read_adc(?);`

? correspond au numéro de la broche du Port A sur laquelle est connectée le capteur. Voir le schéma de la carte capteurs.

Complétez le fichier source C comme ci-dessous et remplacez ? par le numéro de la broche du Port A connectée au capteur d'humidité et ?? par la valeur de la temporisation.

```
sprintf(display_buffer,"Mesure humidite");
lcd_puts(display_buffer);
delay_ms(??); // Attente de 15s avant la première mesure
while (1)
{
// ----- Lecture de l'humidité -----

    Hum = read_adc(?);                // A compléter

};
```

o Traitement (prise en compte des coefficients a' et b')

La valeur à afficher n'est pas Hum mais RH. **Exprimez** RH en fonction de a', b' et Hum.

Complétez le fichier source C comme ci-dessous. Remplacez ? dans l'expression de RH.

```
while (1)
{
// ----- Lecture de l'humidité -----

    Hum = read_adc(?);                // A compléter

// ----- Traitement -----

    RH = (float)((Hum - ?)/ ?); // A compléter: Calcul de RH

};
```

o Affichage de l'humidité

La valeur décimale RH à afficher est convertie en une chaîne de caractères par la fonction :

`ftoa(RH,1,Chaine_Humidite);`

Cette chaîne est placée dans une zone mémoire appelée buffer avant d'être envoyée à l'afficheur. Ceci est réalisé par l'expression ci-dessous.

`sprintf(display_buffer_ligne1,"RH=%-s-c",Chaine_Humidite,0x25);`

Le contenu du buffer est ensuite envoyé à l'afficheur avec la commande ci-dessous

```
lcd_puts(display_buffer);
```

Complétez le fichier source C comme ci-dessous et remplacez ?? pour que la temporisation ait lieu toutes les **quinze secondes**.

```
while (1)
{
// ----- Lecture de l'humidité -----

Hum = read_adc(?); // A compléter

// ----- Traitement -----

RH = (float)((Hum - ?)/ ?); // A compléter: Calcul de l'humidité
ftoa(RH,1,Chaine_Humidite);

// ----- Affichage -----

sprintf(display_buffer,"RH=%-s%-c",Chaine_Humidite,0x25);
lcd_clear();
lcd_puts(display_buffer);
delay_ms(?); // A compléter : Mesure de l'humidité faite toute
// les 15 secondes

};
```

(3) Partie déclarative du programme à réaliser

Le programme ci-dessous utilise trois types de variables :

- **Hum** : entier de type octet,
- **RH** : réel
- **display_buffer, Chaine_Humidite** : tableaux de caractères

Pour être reconnues, ces variables doivent être **déclarées** avant leur utilisation.

Complétez le fichier source C comme ci-dessous :

```
void main(void)
{
// Declare your local variables here
// -----
//type          nom          Commentaires
// -----
unsigned char Hum = 0; // Image de l'humidité [0, 255]
float RH; // Valeur de l'humidité [0, 100%]
unsigned char display_buffer[17]; // Tampon ligne 0 de l'afficheur
unsigned char Chaine_Humidite[]="00.0";
```

C) Programmation du composant

Configurez le projet

- > Project
- > Configure
 - > Sélectionnez l'onglet "After Make"
 - > Cochez "Program the Chip"
 - > ok

Programmez le composant



- > Icône "Make the Project"
- > "Program"

Appel prof

Pour faire vérifier le fonctionnement

Annexe 1 : Câblage de la carte capteurs

Les cavaliers correspondant aux entrées des capteurs doivent être positionnés en 1-2

