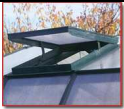




Fiche guide M4	TS SI		P.P.E. Mini serre	
Mesure	2h			
	Mise en œuvre d'une horloge temps réel (HTR)			

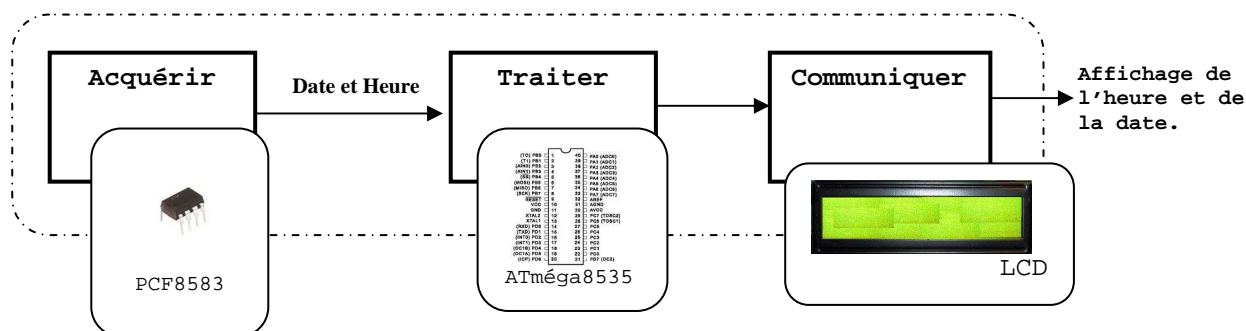
Nom :	Classe :	Groupe :
-------	----------	----------

Objectif : Dater les grandeurs physiques mesurées dans la serre.

<p>Matériels Carte ATMELSSI V1 + Alimentation 10V.</p> <p>Logiciels CodeVisionAvr.</p> <p>Documentation Schémas de la carte ATMELSSI. Documentation technique du circuit HTR I2C PCF8583 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

Les mesures d'humidité, de température et de luminosité doivent être datées avant d'être exploitées par un logiciel permettant de suivre leur évolution dans la serre. L'heure et la date courante doivent également être affichées sur le LCD* de la carte SSI. 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 une horloge temps réel PCF8583. La fonction « Traiter » est assurée par un programme implanté dans le microcontrôleur (ATMEGA8535). La fonction « Communiquer » est remplie par un afficheur LCD (processeur Hitachi).

L'ensemble des structures matérielles étant réunies sur la carte « ATMELSSI », 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 la structure du programme avec les fonctions nécessaires à la mise en œuvre de l'horloge temps réel 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 dater les mesures faites dans la mini serre.

*LCD : Display Liquid Crystal

	FGM4	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** »

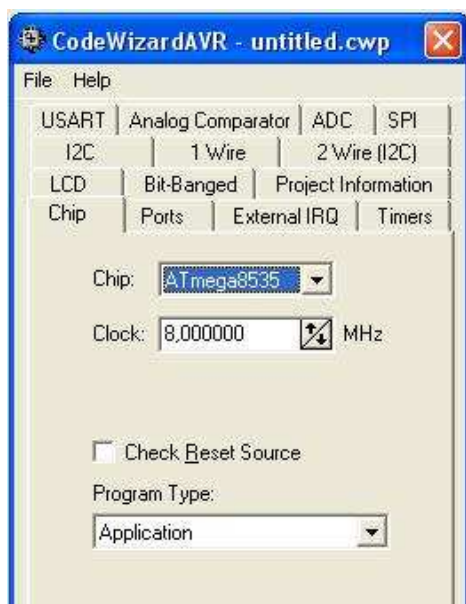


Ici « **Yes** »



(2) Sélection du composant cible

La boîte du « **Magicien** » s'ouvre comme ci-dessous. Choisissez le « **Chip** » **ATMEGA8535** et réglez le signal d'horloge « **Clock** » à 8Mhz.

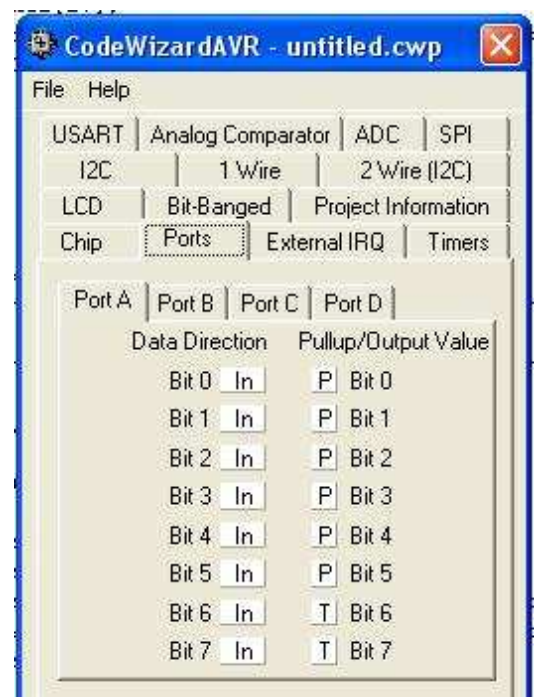


(3) Configuration du port A

- Sélectionnez l'onglet « **Port A** ».

Les boutons-poussoirs de la carte ATMEL SSI sont connectés au port A du microcontrôleur. (Voir le schéma structurel « **ATMELSSI V1** »)

- Configurez le Port A comme ci-contre.



(4) Configuration du bus I²C

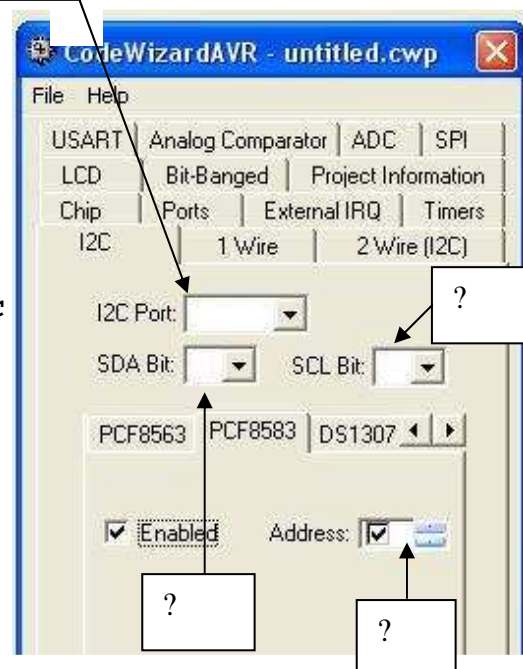
- Sélectionnez l'onglet « I²C ».

Pour que le microcontrôleur ATMEGA8535 communique avec l'horloge temps réel PCF8583, il faut configurer le bus I²C. Pour cela, il vous devez :

- Identifiez la position des lignes SCL et SDA sur le microcontrôleur (voir le schéma de la carte **ATMELSSI V1**) et configurez les champs « I2C Port », « SDA Bit » et « SCL Bit » dans la boîte de dialogue.

- Sélectionnez l'onglet **PCF8583** en cochant « **Enable** ».

Réglez le champ « **Adresse** » du PCF8583.

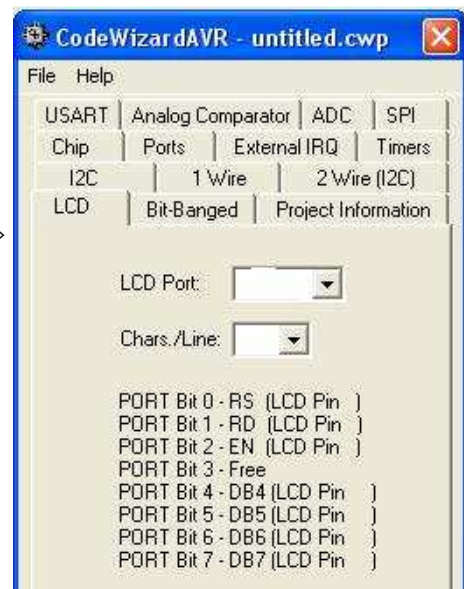


(5) Choix de l'affichage

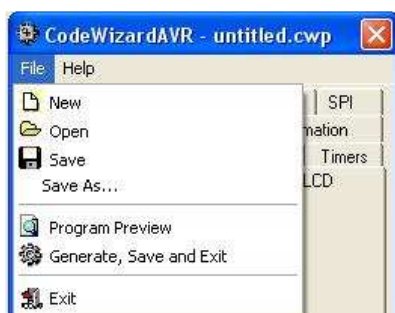
- Sélectionnez l'onglet « LCD ».

Après avoir étudié le schéma de la carte « **ATMELSSI V1** », déterminez sur quel port est connecté l'afficheur LCD et le nombre de caractères par ligne de cet afficheur.

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



(6) 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>

// I2C Bus functions
#asm
.equ __i2c_port=0x12
.equ __sda_bit=3
.equ __scl_bit=7
#endasm
#include <i2c.h>
```

```
// PCF8583 Real Time Clock functions
#include <pcf8583.h>

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

.....Et plus bas .....

// I2C Bus initialization
i2c_init();

// PCF8583 Real Time Clock initialization
rtc_init(0,0);

// LCD module initialization
lcd_init(16);
```

Fermez la fenêtre.

Sélectionnez

→ File

→ Generate, save and Exit

Donnez le nom **HTR** 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 : Acquisition et affichage de l'heure et de la date

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

```
// Declare your global variables here

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 du à la « richesse » des bibliothèques de fonctions fournies avec le cross-compileur CodeVisionAVR.

Par exemple, la lecture de l'heure fournie par l'horloge temps réel s'effectue avec la fonction :

```
rtc_get_time(adresse_HTR,&heures,&minutes,&secondes,&dizieme);
```

adresse_HTR correspond à l'adresse du composant (état logique de A0). **Après la lecture, &heures, &minutes, &secondes, &dizieme** contiennent respectivement l'heure, les minutes, les secondes et les dixièmes de seconde. Cette fonction est située dans la bibliothèque pcf8583. On accède à cette fonction par une référence au fichier <pcf8583.h>.

La lecture de la date fournie par l'horloge temps réel s'effectue avec la fonction

```
rtc_get_date(adresse_HTR,&jour,&mois,&annee);
```

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 rajouterez également une référence à la bibliothèque `ssi` (nécessaire pour accéder aux fonctions `Lire_BP` et `Affiche_LCD`) à la bibliothèque `delay` (nécessaire pour réaliser une temporisation).

(2) Partie exécutive du programme à réaliser

Le programme à réaliser se limite à la suite des actions ci-dessous:

Lire (Date et Heure)
Traiter (Régler la date et l'heure)
Ecrire (Afficher date et Heure)

o Lecture de la date et de l'heure

La lecture de la date et de l'heure fournie par l'horloge temps réel s'effectue avec les fonctions :

```
rtc_get_time(adresse_HTR,&heures,&minutes,&secondes,&dizieme);
rtc_get_date(adresse_HTR,&jour,&mois,&annee);
```

Complétez le fichier source C de votre projet comme ci-dessous.

```
while (1)
{
    BP = Lire_BP();
// ----- Lecture et affichage de la date et de l'heure -----
    switch (etat)
    {
        // Début switch (etat)
        case Horloge: rtc_get_time(adresse_HTR,&heures,&minutes,&secondes,&dizieme);
                        if ((etat_1 != etat) || (secondes_1 != secondes))
                        {
                            rtc_get_date(adresse_HTR,&jour,&mois,&annee);
                            sprintf(display_buffer_ligne0," %-u / %-u / %-i",jour,mois,annee);
                            sprintf(display_buffer_ligne1,"%-uh / %-umn / %-us ",heures,
                                minutes, secondes);
                            Affiche_LCD(display_buffer_ligne0,display_buffer_ligne1);
                            secondes_1 = secondes;
                        }
    }
}
```

```

        if (BP==ENTR)etat = Reglage_HTR;
            else etat = Horloge;
        etat_1 = Horloge;

break;

```

o **Traitement (Réglage de la date et de l'heure)**

Le réglage de la date et de l'heure s'effectue avec la fonction :

```

int regle_HTR(unsigned char adresse,unsigned char Heure_ou_Date, char H_J,
char mn_mois, int sec_annee);

```

Cette fonction, destinée à la carte ATMEL SSI, est contenue dans la bibliothèque SSI.

On donne sa description ci-dessous.

Rôle : Règle la date et l'heure d'une HTR PCF8583 située à adresse (0 ou 1)

Commentaires : La fonction permet de régler l'heure si Heure_ou_Date = 0 ou la date si Heure_ou_Date = 1. Les paramètres H_J, mn_mois, sec_annee servent de base aux réglages.

Celui-ci s'effectue avec les BP :

- INC pour augmenter une valeur,
- DEC pour diminuer une valeur,
- OK pour valider la valeur modifiée et passer au réglage suivant
- EChap pour transférer la valeur réglée et sortir de la fonction

Librairie utilisée : PCF8583.h

Entrées :

```

    adresse : adresse du boitier (0 ou 1 selon A0)
    Si Heure_ou_Date = 0 (réglage de l'heure)
        alors -1 < H_J < 24, -1 < mn_mois < 60,
            -1 < sec_annee < 60
        sinon -1 < H_J < 32, 0 < mn_mois < 13,
            1999 < sec_annee < 2050
    Fin si

```

Sortie : la fonction renvoie 0
*/

Complétez le fichier source C de votre projet comme ci-dessous.

```
// ----- Traitement -----
case Reglage_HTR : if (etat_1 != etat)
{
    sprintf(display_buffer_ligne0,"  Reglage HTR");
    sprintf(display_buffer_ligne1,"SET>H ETR>Date ");
    Affiche_LCD(display_buffer_ligne0,display_buffer_ligne1);
}

switch(BP)
{ // Les noms des BP sont contenus dans la bibliothèque ssi.h
    case SET : Heure_ou_Date = 0;
                H_J = 12;
                mn_mois = 30;
                sec_annee = 0;
                regle_HTR(adresse_HTR,Heure_ou_Date,H_J,mn_mois,sec_annee);
                etat = Horloge;

                break;

    case ENTR : Heure_ou_Date = 1;
                H_J = 15;
                mn_mois = 6;
                sec_annee = 2008;
                regle_HTR(adresse_HTR,Heure_ou_Date,H_J,mn_mois,sec_annee);
                etat = Horloge;

                break;

    default:    etat = Reglage_HTR;
}
    etat_1 = Reglage_HTR;
break;

} //fin du switch
} // fin du while
```

(3) Partie déclarative du programme à réaliser

Deux zones de variables doivent être complétées :

Complétez le fichier source C du projet comme ci-dessous :

```
// Declare your global variables here
//-----
char display_buffer_ligne0[17];          // tampon ligne 0 de l'afficheur
char display_buffer_ligne1[17];          // tampon ligne 1 de l'afficheur

unsigned char heures, minutes, secondes, secondes_1, dixieme;
                                           //stockage de l'heure
unsigned char Heure_ou_Date, H_J, mn_mois;
unsigned int sec_annee;
```

```
// Declare your local variables here
//-----
unsigned char jour,mois, BP;
unsigned int annee;
unsigned char etat = Horloge, etat_1 = Reglage_HTR;
```

pour terminer !

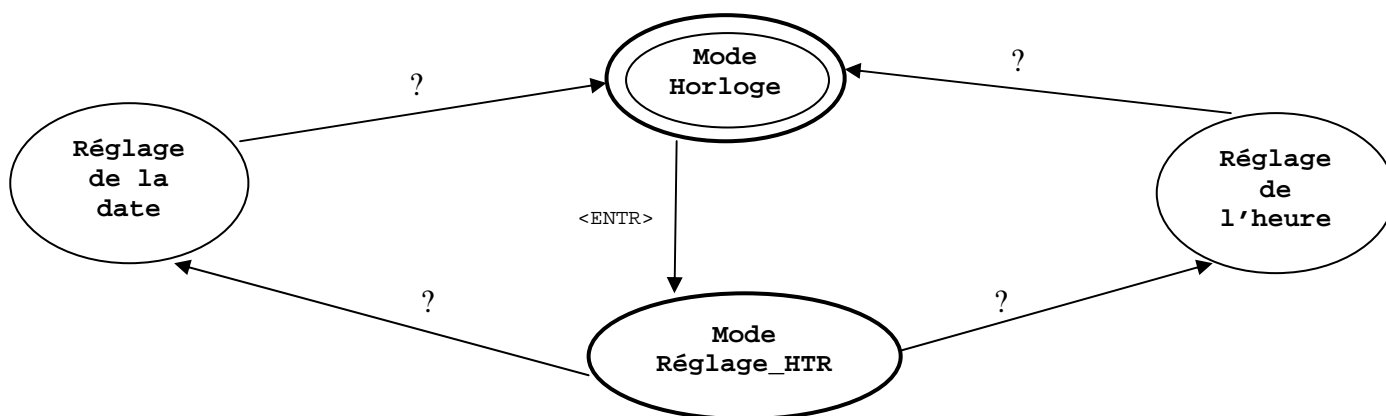
```
#include <mega8535.h>
#include <ssi.h>          // contient les fonctions Lire_BP,Affiche_LCD etc..
#include <stdio.h>         // contient la fonction sprintf
#include <delay.h>         // contient la fonction delay_ms

//Etats du graphe
#define Horloge 0
#define Reglage_HTR 1

//Adresse HTR
#define adresse_HTR 0
```

(4) Analyse et modélisation du programme proposé

Le comportement du programme proposé peut être modélisé par le graphe des transitions ci-dessous.



On donne en **annexe 1** un exemple de graphe des transitions et sa traduction algorithmique.

Analysez le programme de la page précédente et **complétez** les conditions de transitions dans le graphe ci-dessus.

Modifiez le graphe ci-dessus et le programme pour que le passage du mode Réglage_HTR au mode Horloge se fasse avec le bouton-poussoir <ECHAP>.

Notez ci-dessous la modification à apporter au programme.

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"

D) Test du programme

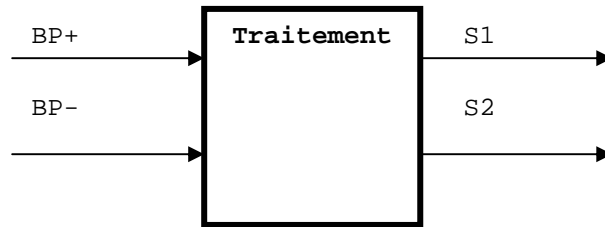
Appel prof

Pour faire vérifier le fonctionnement

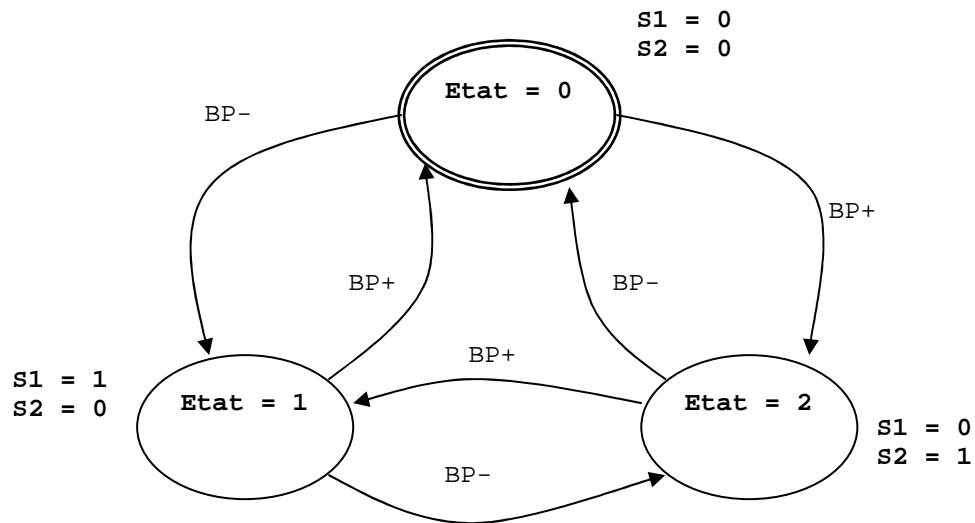


Annexe 1 : Exemple de graphe des transitions et sa traduction algorithmique

Une fonction « Traitement » active deux sorties binaires (S1, S2) selon l'état de deux boutons-poussoirs (BP+, BP-).



L'évolution de S1 et S2 est décrite par le graphe des transitions ci-dessous.



Algorithme Traitement

variables

BP+, BP-, S1, S2 : bit ;

état : octet ;

début prog

état ← 0 ; S1 ← 0 ; S2 ← 0 ;

début boucle

lire(BP+, BP-) ;

Evolution des sorties

Evolution de la variable d'état

selon (état)

0 : S1 ← 0 ; S2 ← 0 ;

si (BP+ = 1) alors état = 2 ;

sinon si (BP- = 1) alors état ← 1 ;
sinon état ← 0 ;

fin si

fin si

fin selon

1 : S1 ← 1 ; S2 ← 0 ;

si (BP+ = 1) alors état ← 0 ;

sinon si (BP- = 1) alors état ← 2 ;
sinon état ← 1 ;

fin si

fin si

fin selon

2 : S1 ← 0 ; S2 ← 1 ;

si (BP+ = 1) alors état ← 1 ;

sinon si (BP- = 1) alors état ← 0 ;
sinon état ← 2 ;

fin si

fin si

fin selon

écrire(S1, S2) ;

fin boucle

fin prog