



## **Dossier de Projet ER-EDP**

*Wattmètre 48V 50A Programmation afficheur LCD*

Université François-Rabelais de Tours  
Institut Universitaire de Technologie de Tours  
Département Génie Électrique et Informatique Industrielle

UNIVERSITE FRANCOIS-RABELAIS  
TOURS



Institut Universitaire de Technologie

Département  
GENIE ELECTRIQUE ET  
INFORMATIQUE INDUSTRIELLE

## **Dossier de Projet ER-EDP**

*Wattmètre 48V 50A Programmation afficheur LCD*

WARG Jérémy – MOREAU Benjamin  
2ème Année \ GP P1  
2007-2009

Enseignants  
M. LEQUEU Thierry

# Sommaire

Introduction.....	4
1. Présentation du projet.....	5
1.1. Cahier des charges.....	5
1.2. Schéma synoptique de niveau 1.....	6
1.3. Schéma synoptique de niveau 2.....	7
1.4. Planning prévisionnel et réel.....	8
2. Ajout sur le projet.....	8
2.1. Capteur de température.....	8
2.2. Bouton poussoir.....	9
3. Programmation.....	9
3.1. Mesure de température.....	9
3.2. Mesure de courant.....	10
3.3. Mesure de tension.....	10
3.4. Calcul de la puissance.....	11
3.5. Calcul de l'énergie.....	12
3.6. Affichage.....	12
4. Essais et mesures.....	13
Conclusion.....	15
Index des illustrations.....	16
Bibliographie.....	17
Annexe 1 : Programme complet.....	18

## Introduction

Dans le cadre du module ER-EDP, nous avons choisis de réaliser un projet concernant la programmation d'un afficheur LCD pour un wattmètre 48 V 50 A. Le wattmètre est déjà réalisé, il faut ajouter un bouton poussoir ainsi qu'un capteur de température.

Nous allons devoir programmer l'ATméga pour afficher diverses valeurs, comme la tension, le courant, ou encore la puissance.

La première partie de ce dossier traitera du cahier des charges avec le planning prévisionnel et réel, ainsi que les synoptiques globales et détaillés.

Les ajouts effectués sur le projet sont consignés dans la partie deux. Ces ajouts sont le bouton poussoir ainsi que la sonde de température.

La programmation sera expliquée dans la partie trois, avec le détail des diverses parties.

Enfin, les essais et mesures seront dans la dernière parties.

# 1. Présentation du projet

## 1.1. Cahier des charges

Le wattmètre sera installé entre une alimentation et des batteries en chargement pour les karting du club e-kart.

Le wattmètre possède un afficheur LCD commandé par un Atmégas 8535. Celui-ci est déjà inséré dans un boîtier avec un capteur de courant installé sur les fils d'alimentation de la batterie. Il faut créer le capteur de température à partir d'un typon déjà conçu que l'on branchera directement sur les ports prévues à cet effet.[1]

I	:	1	5	0	A				U	:	5	0	V		
P	u	i	s	s	a	n	c	e	:	8	0	0	0	W	
E	n	e	r	g	i	e	:	1	0	0	0	0	K	W	s
T	e	m	p	:	+	2	3	,	5	°	C				

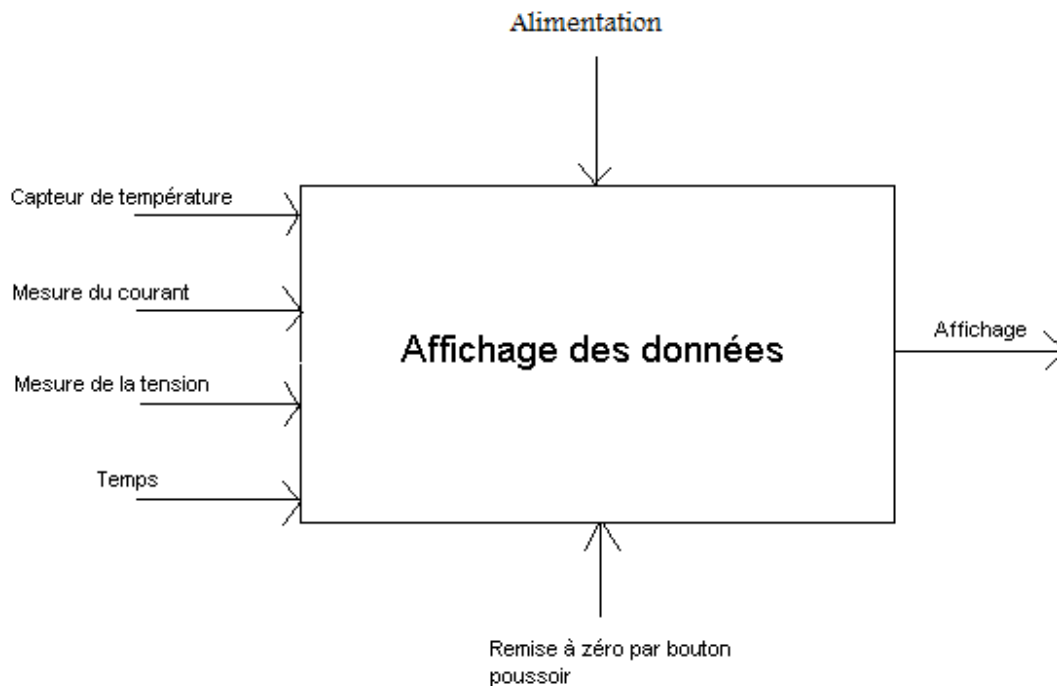
*Illustration 1: Objectif à obtenir sur l'afficheur LCD*

On affichera sur l'écran LCD l'intensité, la tension, la puissance, l'énergie et la température.

La mesure de courant est réalisée avec un capteur de type HAS. Le capteur de température est une sonde LM75

L'écran LCD (MC1604-séries) contient 4 lignes et 16 colonnes, ce qui suffira pour afficher les différentes informations.

## 1.2. Schéma synoptique de niveau 1



*Illustration 2: Synoptique de premier niveau*

La remise à zéro par bouton poussoir réinitialisera le temps pour le calcul de l'énergie.

Ici l'affichage des données est traité par l'ATméga, il enverra les informations à l'afficheur LCD qui se contentera d'afficher les données qu'il recevra. Il faudra donc bien faire attention à ce qu'en sortie de l'Atméga, les données ne soit pas dans un format incompréhensible.

### 1.3. Schéma synoptique de niveau 2

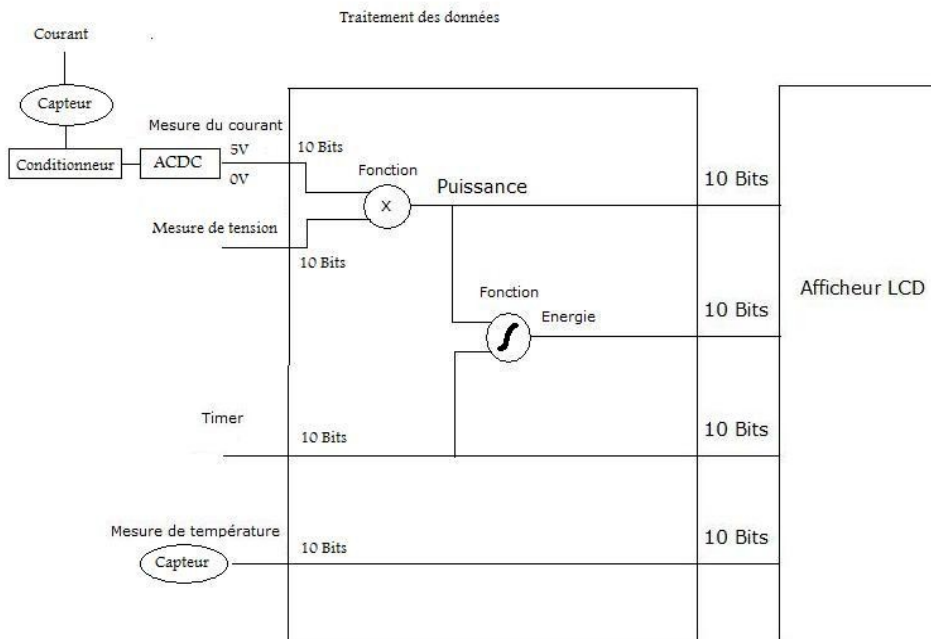


Illustration 3: Synoptique de second niveau

Le timer sera donné par le quartz qui est intégré dans l'ATméga.

On remarque qu'à partir de l'information du courant et de la tension, l'ATméga pourra calculer la puissance. A partir de cette même puissance, il en déduira l'énergie qu'il calculera à l'aide du timer interne de l'ATméga.

### 1.4. Planning prévisionnel et réel

Semaine	4	5	5	6	7	8	9	10	11	12
Cahier des charges et analyse du projet présent	■	■					■	■		
Prog* du signal de puissance (tension et courant)		■				■	■	■		
Prog* du signal d'énergie (puissance et timer)			■				■	■	■	
Prog* du signal du capteur de température et timer			■	■			■	■		
Prog* de la RAZ par bouton poussoir				■	■		■	■		
Prog* et conversion des signaux vers afficheur					■	■	■	■		
Implantations du programme et Tests							■	■	■	■
Installation du BP		■					■	■	■	
Rédaction			■	■	■	■	■	■	■	■

■ Planning prévisionnel    ■ Planning réel

\* Prog = Programmation

Illustration 4: Planning final

## 2. Ajout sur le projet

A partir de la maquette déjà réalisé nous avons apporté diverses modifications qui seront nécessaires au bon fonctionnement du programme.

### 2.1. Capteur de température

L'ajout du capteur de température nécessite de le créer à partir d'un typon conçu par M. LEQUEU. Le capteur est une sonde LM75 délivrant une tension qui sera directement traitée et comprise par l'ATméga.

La mesure de température s'effectue à partir de la sonde LM75. Elle envoi directement l'information sous forme de tension à l'ATméga, qui va la traiter et l'envoyer sur l'afficheur.

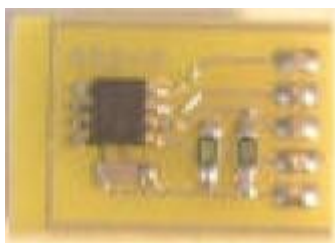


Illustration 5: Carte mesure de température

La carte à été réalisée à partir du typon de M. LEQUEU. Celle-ci est composée de trois composants, la sonde LM7574 et de deux résistances.[2]

## 2.2. Bouton poussoir

Le bouton poussoir à été rajouter sur la maquette déjà existante. Celui-ci va permettre une remise à zéro du temps calculé nécessaire au calcul de l'énergie.

Il a fallut percer le boitier pour installer le bouton poussoir avec retour. Sur la maquette d'origine, il à été prévu un bornier pour pouvoir accueillir l'ajout du bouton poussoir.

## 3. Programmation

### 3.1. Mesure de température

Pour afficher la température, on programme :

```
{
    //Mesure température

    temp=lm75_temperature_10(0);
    signe='+';
    if(temp<0)
    {
        signe='-';
        temp=-temp;
    }
    sprintf(tampon,"%c%i.%u\xdfC",signe,temp/10,temp*10);

    lcd_gotoxy(5,3);
    lcd_puts(tampon);
}
```

*Illustration 6: Programme mesure température*

L'information de température acquise à partir de la sonde est directement décodée par l'ATméga (« lm75\_temperature\_10(0)).

Si la valeur de la variable temp est inférieur à 0,on insère un caractère (+ ou -).

La valeur est alors affichée par la fonction lcd\_puts.

### 3.2. Mesure de courant

```
        //mesure courant
    {
        Ibatp=((read_adc(1)*4)/818.4)*50/4; //de 0 à 818
        Ibatm=((read_adc(2)*4)/818.4)*50/4;
        if((Ibatp)>0)
        {
            sprintf(tampon,"%+dA ",Ibatp);
        }
        else
        {
            sprintf(tampon,"%-dA",Ibatm);
        }
        lcd_gotoxy(2,0);
        lcd_puts(tampon);
    }
}
```

*Illustration 7: Programme mesure courant*

Le courant est relevé avec la fonction `read_adc`. On la multiplie par 50 pour les cinquante spires. On crée deux variables, dans lesquelles les valeurs du courant sont insérées. L'une d'elle servira lorsque la valeur du courant est positive, l'autre lorsque celle-ci sera négative.

### 3.3. Mesure de tension

```
        //mesure de tension
    {
        i=read_adc(0);
        Vbat=0.071*i;
        Vbat=((float)i*(float)i)*0.00014395)-(0.1052532*(float)i)+51.870777;
        sprintf(tampon,"%dV",Vbat);
        lcd_gotoxy(12,0);
        lcd_puts(tampon);
    }
}
```

*Illustration 8: Programme mesure tension*

La tension est relevée par la fonction `read_adc(0)` et est affichée.

### 3.4. Calcul de la puissance

```
        //affichage de la puissance

    {
        if((Ibatp)>0)
        {
            if(PIND.3==0)
            {
                E=0;
                P=0;
            }
            else
            {
                P=Wbat*Ibatp;
            }
        }
        else
        {
            if(PIND.3==0)
            {
                E=0;
                P=0;
            }
            else
            {
                P=Wbat*Ibatm;
            }
        }

        sprintf(tampon,"%d W",P);
        lcd_gotoxy(10,1);
        lcd_puts(tampon);
    }
}
```

*Illustration 9: Programme mesure puissance*

Ici, la remise à zéro s'effectue par le bouton poussoir, qui est connecté à l'entrée 3. La RAZ s'effectue avec un état bas.

Pour calculer la puissance, nous prenons les variables déjà existantes de la tension et de l'intensité pour appliquer la loi d'Ohm. L'affichage est toujours réalisé avec les mêmes fonctions.

### 3.5. Calcul de l'énergie

```
//Affichage de l'énergie
{
    (float)E=(float)E+(P*0.1/100);
    sprintf(tampon,"%dKWs",E/10);
    lcd_gotoxy(8,2);
    lcd_puts(tampon);
}
```

*Illustration 10: Programme mesure énergie*

Ici, l'énergie est calculée par la formule suivante :

$$W = W + (P \times t)$$

t étant le temps et P la puissance.

### 3.6. Affichage

Pour afficher les paramètres sur l'écran LCD, on utilise deux fonctions principales :

- La fonction `lcd_gotoxy` (numéro de colonne, numéro de ligne) va placer sur l'écran à la bonne place les valeurs souhaitées.
- La fonction `lcd_puts` (tampon) va placer la variable tampon sur l'afficheur à l'endroit qu'on lui a indiqué précédemment.

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

lcd_gotoxy(0,0);
lcd_putsf("i:--");
lcd_gotoxy(10,0);
lcd_putsf("U:--");
lcd_gotoxy(0,1);
lcd_putsf("Puissance:---");
lcd_gotoxy(0,2);
lcd_putsf("Energie:---");
lcd_gotoxy(0,3);
lcd_putsf("Temp:----"C");
```

*Illustration 11: Programme initialisation afficheur LCD*

Ce morceau de programme sert à initialiser l'écran LCD en y mettant en place les termes nécessaires à l'information des données affichées.

## 4. Essais et mesures

Nous avons effectué une mesure pour vérifier la valeur que nous retourne le capteur de courant.

Un tableau des résultats à été dressé :

HAS 50-S avec 50 spires      mesure du 13 fevrier

Courant	Tension capteur	Valeur affichée
0	0	1,5
0,11	0,45	24,5
0,2	0,84	43
0,5	2,1	103
1	4,2	204

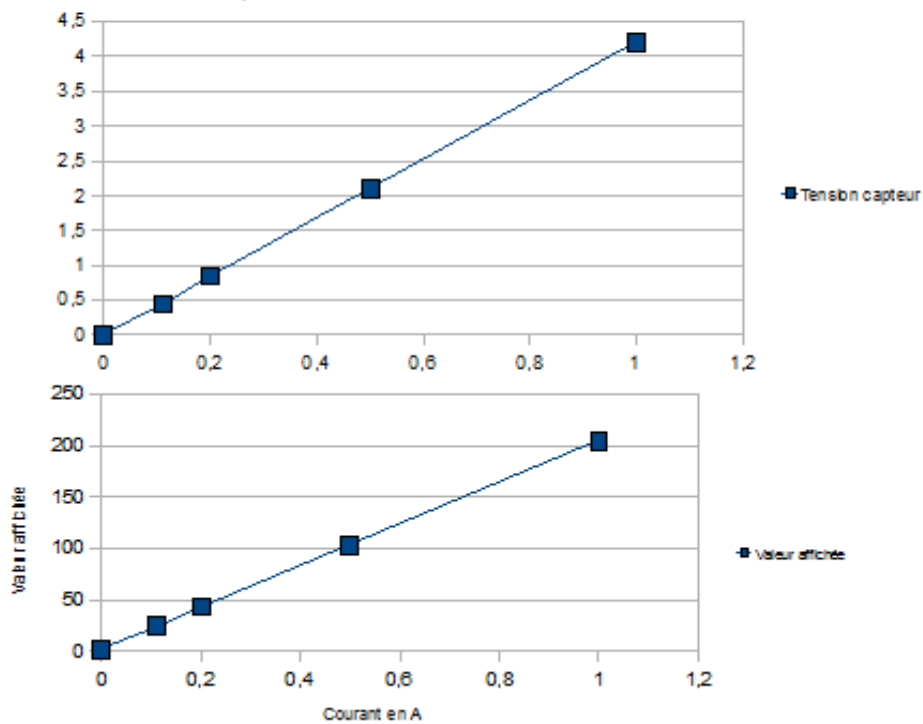
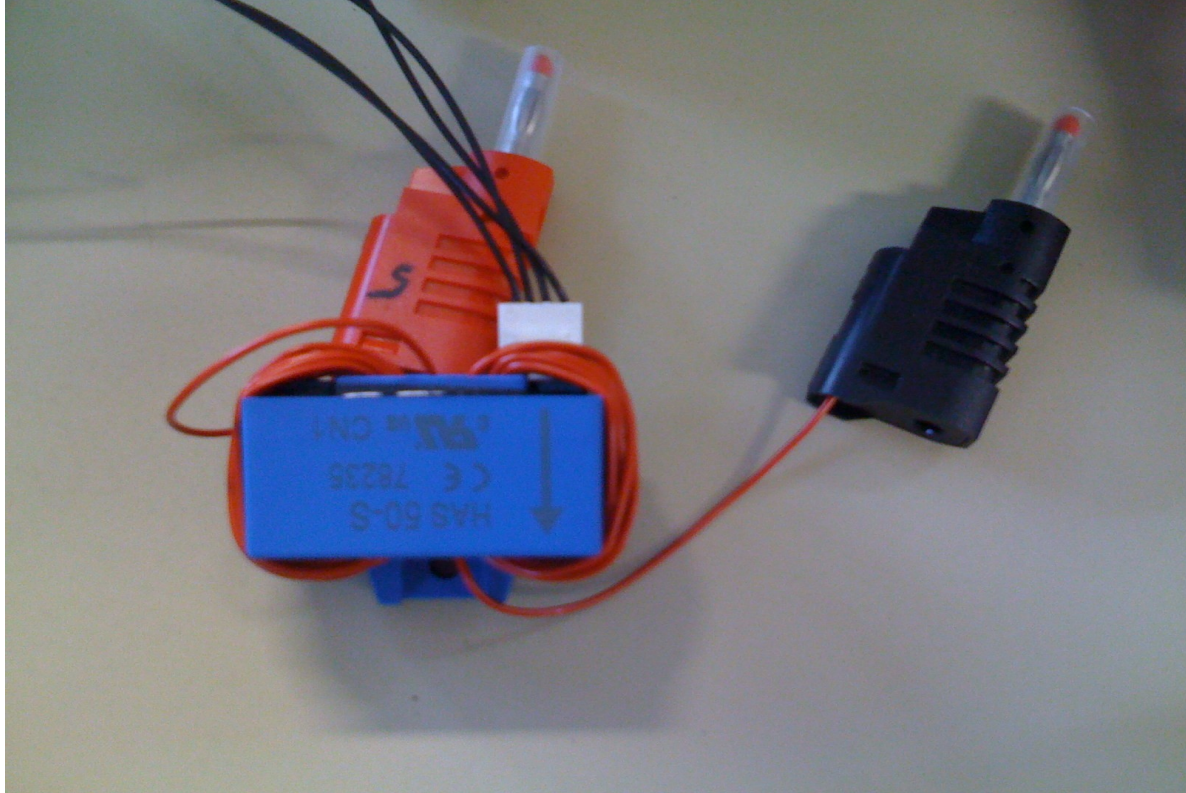


Illustration 12: Mesure pour le test du capteur de courant HAS 50-S

On remarque que le capteur de courant délivre une tension en fonction du courant. A partir de la tension mesurée, on vérifie la valeur affichée. Les tableaux ci-dessus montre que les valeurs sont bien proportionnel, et on peut donc penser que les mesures sont correctes.[3]

*Illustration 13: Capteur de courant HAS 50-S*



Pour simuler la mesure de courant avec des intensités importantes, on effectue un bobinage pour que le capteur « voit » un champ plus important.

## Conclusion

Ce projet nous a permis d'étudier le fonctionnement du Wattmètre 48V 50A. L'étude c'est principalement portée sur la programmation de l'At-Méga 8535 car nous avons repris un projet déjà existant de l'année dernière. Les principales fonctions réalisées sont les suivantes: mesure courant, tension, puissance, énergie et la température.

L'étude du projet complet nous a beaucoup apporté, en termes d'autonomie, de réflexion et de recherches de solutions appropriées à un problème donné. Nous avons dû nous répartir le travail et nous concerter lors de prises de décision comme la sélection d'une solution par rapport à un vaste panel de possibilités.

Cela nous a également permis de faire une utilisation avancée d'un logiciel de programmation informatique professionnel (Code Vision AVR), ce qui pourra se révéler utile lors de nos stages ou notre entrée dans le monde professionnel.

## Index des illustrations

Illustration 1: Objectif à obtenir sur l'afficheur LCD.....	5
Illustration 2: Synoptique de premier niveau.....	6
Illustration 3: Synoptique de second niveau.....	7
Illustration 4: Planning final.....	8
Illustration 5: Carte mesure de température.....	8
Illustration 6: Programme mesure température.....	9
Illustration 7: Programme mesure courant.....	9
Illustration 8: Programme mesure tension.....	10
Illustration 9: Programme mesure puissance.....	10
Illustration 10: Programme mesure énergie.....	11
Illustration 11: Programme initialisation afficheur LCD.....	11
Illustration 12: Mesure pour le test du capteur de courant HAS 50-S.....	12
Illustration 13: Capteur de courant HAS 50-S.....	13

## Bibliographie

- [1] **LEQUEU Thierry**, "*ATMEL*", [En ligne]. <<http://www.thierry-lequeu.fr/data/AT-MEGA-8535L.pdf>> (Page consultée le 23/01/2009).
- [2] **LEQUEU Thierry**, "*LM 2574*", [En ligne]. <<http://www.thierry-lequeu.fr/data/LM2574.pdf>> (Page consultée le 06/02/2009).
- [3] **LEQUEU Thierry**, "*Current Transducer HAS 50-S*", [En ligne]. <<http://www.thierry-lequeu.fr/data/HAS-50-600-S.pdf>> (Page consultée le 13/02/2009).

# Annexe 1 : Programme complet

/\*\*\*\*\*\*

This program was produced by the  
CodeWizardAVR V1.24.2c Professional  
Automatic Program Generator  
© Copyright 1998-2004 Pavel Haiduc, HP InfoTech s.r.l.  
<http://www.hpinfotech.ro>  
e-mail:office@hpinfotech.ro

Project :

Version :

Date : 27/03/2009

Author : Jérémy WARG

Company : IUT GEII TOURS

Comments:

Chip type : ATmega8535

Program type : Application

Clock frequency : 16,000000 MHz

Memory model : Small

External SRAM size : 0

Data Stack size : 128

---

// Declare your global variables here

void main(void)

{

// Declare your local variables here

unsigned char tampon[20];

int i,lbatm,lbatp,Vbat;

```

unsigned int P,P1=0,E;
int temp;
char signe;

// LM75 Temperature Sensor initialization
// thyst: 35°C
// tos: 40°C
// O.S. polarity: 0
lm75_init(0,35,40,0);

// LCD module initialization
lcd_init(16);

lcd_gotoxy(0,0);
lcd_putsf("i:--");
lcd_gotoxy(10,0);
lcd_putsf("U:--");
lcd_gotoxy(0,1);
lcd_putsf("Puissance:---");
lcd_gotoxy(0,2);
lcd_putsf("Energie:---");
lcd_gotoxy(0,3);
lcd_putsf("Temp:----°C");

/* switch to writing in Display RAM */

while (1)
{
    {
        //Mesure température

```

```

temp=lm75_temperature_10(0);
    signe='+';
    if(temp<0)
        {
            signe='-';
            temp=-temp;
        }
sprintf(tampon,"%c%i.%u\\xdfC",signe,temp/10,temp%10);

    lcd_gotoxy(5,3);
    lcd_puts(tampon);

}

    //mesure courant
{
    lbatp=((read_adc(1)*4)/818.4)*50/4; //de 0 à 818
    lbatm=((read_adc(2)*4)/818.4)*50/4;
    if((lbatp)>0)
    {
        sprintf(tampon,"+%dA ",lbatp);
    }
    else
    {
        sprintf(tampon,"-%dA",lbatm);
    }
    lcd_gotoxy(2,0);
    lcd_puts(tampon);
}

//mesure de tension

```

```

{
  i=read_adc(0);
  Vbat=0.071*i;
  Vbat=(((float)i*(float)i)*0.00014395)-(0.1052532*(float)i)+51.870777;
  sprintf(tampon,"%dV",Vbat);
  lcd_gotoxy(12,0);
  lcd_puts(tampon);
}
//affichage de la puissance

```

```

{
if((Ibatp)>0)
{
  if(PIND.3==0)
  {
    E=0;
    P=0;
  }
  else
  {
    P=Vbat*Ibatp;
  }
}
else
{
  if(PIND.3==0)

```

```

        {
            E=0;
            P=0;
        }
        else
        {
            P=Vbat*Ibatm;
        }
    }

    sprintf(tampon,"%d W",P);
    lcd_gotoxy(10,1);
    lcd_puts(tampon);

}

//Affichage de l'énergie

{

(float)E=(float)E+(P*0.1/100);
sprintf(tampon,"%dKWs",E/10);
lcd_gotoxy(8,2);
lcd_puts(tampon);
}

}

}

```