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

ER-ISI4
Wattmètre énergimètre 48V 50A

BRUNAUD Cyril
DELAUNAY Francis
2e Année Groupe Q2

Thierry LEQUEU
Inès DACOSTA

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

ER-ISI4
Wattmètre énergimètre 48V 50A

BRUNAUD Cyril
DELAUNAY Francis
2e Année Groupe Q2

Thierry LEQUEU
Inès DACOSTA

SOMMAIRE

INTRODUCTION

Lors du semestre 4, dans le cadre des projets tutorés en ER-ISI, nous avons choisi de réaliser la programmation d'un Wattmètre énergimètre servant au club E-Kart dont M.LEQUEU fait parti. Ce Wattmètre sera utilisé pendant la charge de la batterie du Kart permettant de surveiller divers informations comme la tension, la puissance ou encore la température.

Lors de cette étude, nous allons plus particulièrement étudier les différentes mesures de courant, tension, température ainsi que le micro-contrôleur ATMEGA 8535 et sa programmation.

Durant ce projet, étant donné que la carte était déjà réalisée, nous nous sommes plus particulièrement penchés sur programmation du microcontrôleur.

1.PRÉSENTATION DU PROJET

1.1. Cahier des charges

La carte électronique avec sa programmation a pour but d'afficher sur un écran LCD la tension, le courant, la température, la puissance et l'énergie du kart pendant sa charge. Elle sera mise dans un boîtier permettant d'avoir une meilleure protection.

La mesure du courant est réalisée à l'aide d'un capteur de type HAS et la mesure de température se fait grâce à une sonde LM75.

Le micro-contrôleur utilisé est un ATMEGA 8535.

L'affichage se fera avec un écran LCD de référence MC1604C-SERIES possédant 4 lignes x 16 caractères.

Le choix finale pour l'affichage des données est le suivant :

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: Afficheur LCD 16x4 caractères[1]

Mesure de la température

La température du moteur est mesurée grâce à la sonde de température LM75 qui possède 8 broches et fournit directement la température sans l'aide de composants externes. Ce composant peut fonctionner sous des températures allants de -55°C à +125°C. Il nécessite d'être alimenté en 5V.

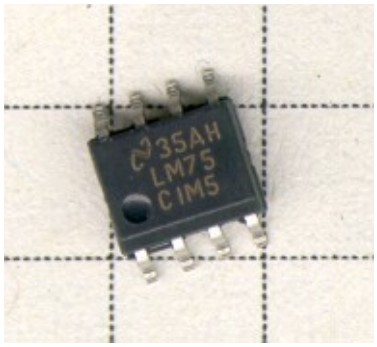


Illustration 3: Sonde de température LM75[2]

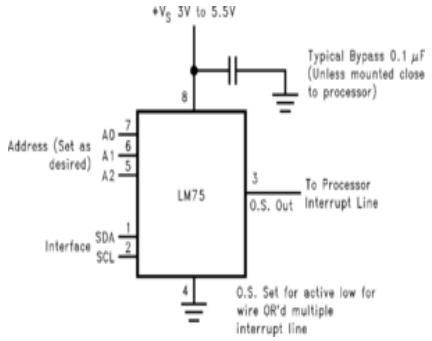


Illustration 2: Schéma LM75[3]

Mesure du courant

Le courant est mesuré par un capteur de type HAS. Dans notre projet, nous avons utilisé un capteur HAS 200 pouvant mesurer jusqu'à 200A. Ce capteur renvoi au micro-contrôleur une tension comprise entre -4V et +4V (soit -4V pour -200A et +4V pour +200A).

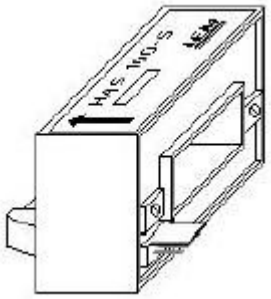
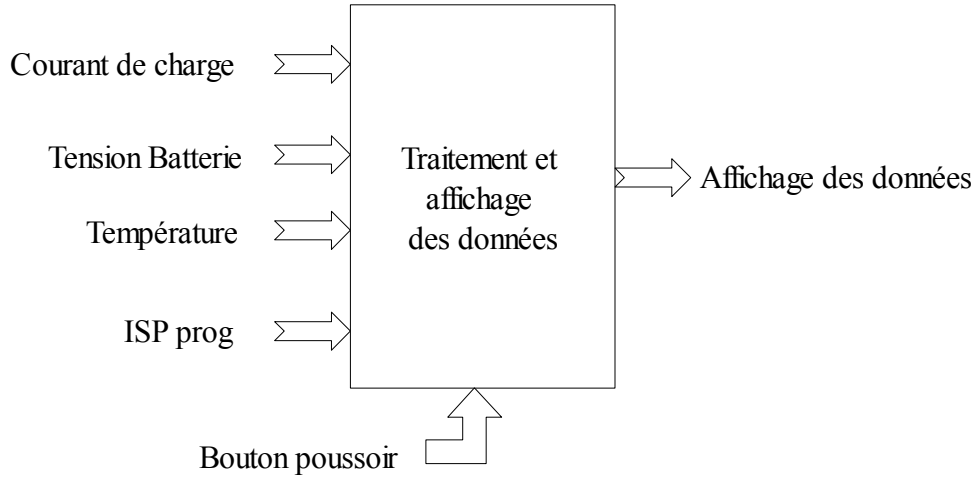


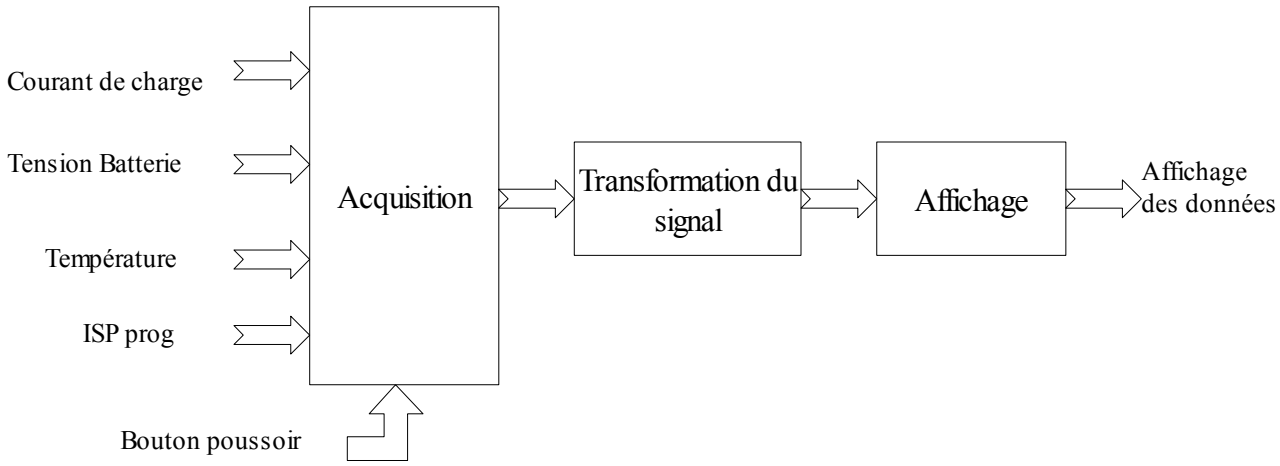
Illustration 4: Capteur courant HAS200[4]

1.2.Schéma fonctionnel de niveau 1

La carte électronique permet le traitement de grandeurs électriques venant des différents capteurs pour afficher les informations correspondantes sur l'afficheur LCD.



1.3.Schéma fonctionnel de niveau 2



Ce schéma représente la chaîne de traitement de la carte. L'acquisition et la transformation du signal est réalisée par l'ATMEGA 8535 avec la programmation. L'affichage est fait grâce à un afficheur LCD.

1.4.Analyse fonctionnelle

- **Acquisition :** Effectuée par le microcontrôleur ATMEGA 8535.
- **Transformation du signal :** Effectuée par l'ATMEGA 8535.
- **Affichage :** Effectuée par l'afficheur 16x4 caractères.

2.ÉTUDE DE LA CARTE

2.1.Alimentation

L'alimentation de la carte est réalisée en 2 parties. Premièrement il y a un régulateur TEN 5-4823 où la tension d'alimentation pouvant être comprise entre 36V et 75V y est directement appliquée. Ce régulateur sort 2 valeurs de tensions (+15V et -15V) servants à alimenter le capteur de courant. La tension positive va aussi être appliquée à l'entrée du régulateur LM2574M (régulateur pouvant accepter une tension d'entrée comprise entre 7V et 60V) qui va donner une tension de sortie de +5V permettant d'alimenter le micro-contrôleur ainsi que l'écran LCD.

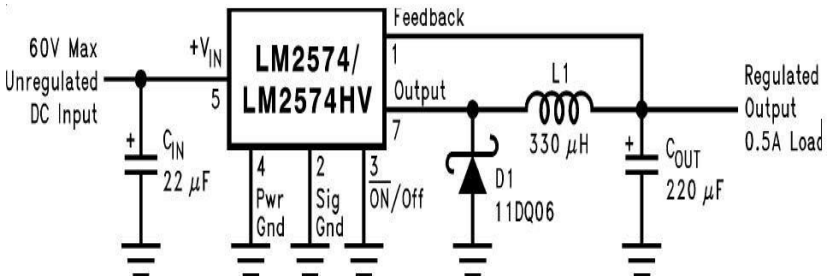


Illustration 5: Schéma régulateur LM2574M[5]

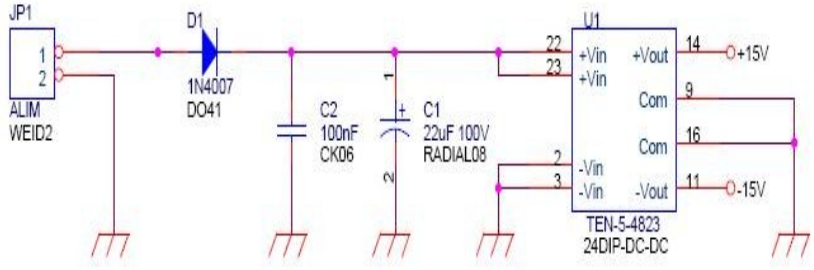


Illustration 6: Schéma régulateur TEN 5-4823[6]

Les condensateurs placées en entrée permettent de limiter les variations de tension en entrée et donc permettre au régulateur de fonctionner correctement. Le condensateur Cout placé en sortie du LM2574M permet de lisser la tension de sortie pour qu'elle soit constante.

La diode D1 est une diode de protection pour éviter toute détérioration du régulateur en cas d'inversion de la tension.

Sur l'entrée FeedBack du LM2574M, il est nécessaire d'appliquer une tension de 1,21V afin de vérifier la bonne régulation du composant.

2.2.Microcontrôleur

Le microcontrôleur utilisé pour la réalisation de notre carte est un ATMEGA 8535 du fabricant Atmel. C'est ce composant qui va accueillir le programme gérant les entrées-sorties afin d'afficher sur l'écran LCD les informations venants des capteurs. Il est alimenté en +5V à partir de la patte 10 (VCC).

Illustration 7: ATMEGA 8535[7]

Le connecteur CON ISP est utilisé pour flasher l'EEPROM relié sur la patte 9 (/Reset) afin de mettre le programme désiré.

Le quartz cadencé à 16Mhz est relié sur les entrées 12 (XTAL2) et 13 (XTAL1). Il est utilisé afin de remplacer l'horloge interne du microcontrôleur qui est cadencé à une fréquence moins élevée.

Le PORTA variant de la patte 33 (PA7) à 40 (PA0) est utilisé comme entrée analogique sur laquel nous y appliquons la tension de la batterie ainsi que les tensions venants du capteur de courant. Le PORTC est un port numérique relié à l'afficheur LCD. Le connecteur CON ISP permettant la programmation du microcontrôleur est relié sur le PORTB qui est lui aussi numérique. Le dernier port numérique est le PORTD où est connecté le bouton poussoir permettant la remise à zéro de l'énergie.

2.3.Mesure du courant

Si le courant mesuré est positif, le capteur renvoi une tension positive. On a donc sur l'AOP suiveur en haut du schéma sa sortie qui est positive et ainsi Ibatp positif. Sur l'AOP inverseur, la sortie est négative et donc les diodes de protections rendent la tension Ibatm égale à 0. Il en est de même si le courant est négatif avec Ibatm égal à 0 et Ibatp positif.

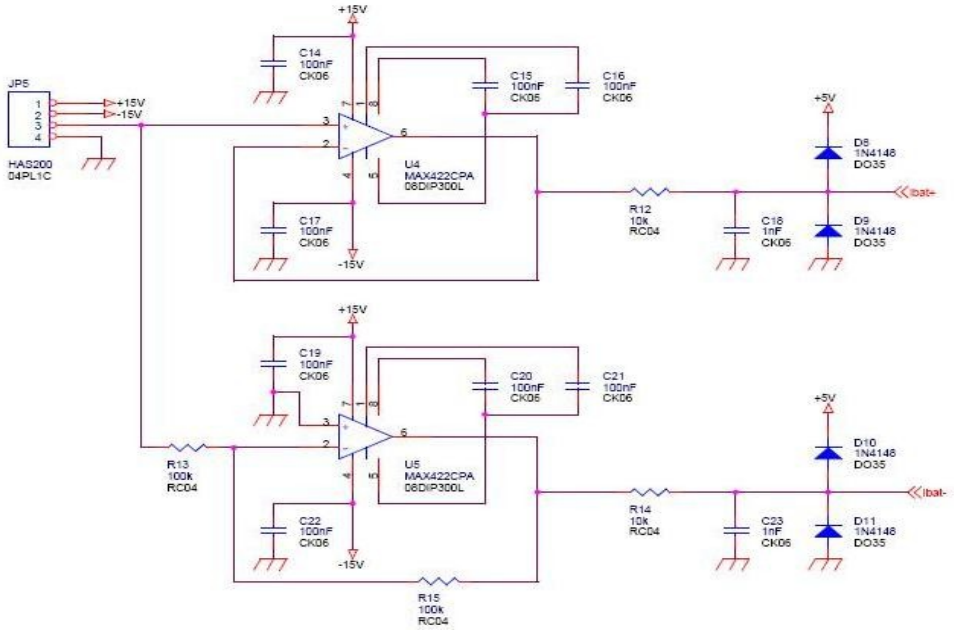


Illustration 8: Schéma mesure du courant[8]

2.4.Mesure de la tension

En entrée de la carte, nous appliquons une tension pouvant aller de 36V à 75V. Voici le tableau contenant différentes mesures en fonction de la tension d'alimentation.

Vbat (V)	36	40	45	50	55	60	65
VADC0 (V)	2,71	3	3,33	3,61	3,84	4,02	4,16
Nthéorique	491	545	613	682	750	818	886
Npratique	544	602	668	726	771	807	835
G	0,08	0,08	0,07	0,07	0,07	0,07	0,06

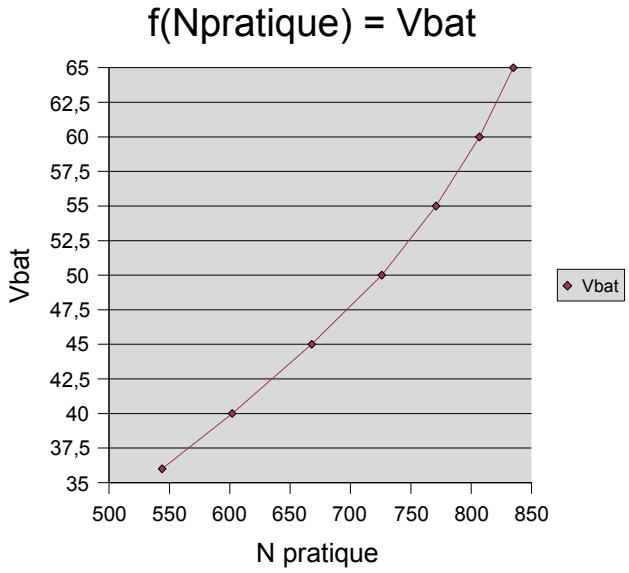
• Vbat : Tension d'alimentation de la carte.

• VADC0 : Tension mesurée sur la patte 40 (ADC0) du microcontrôleur.

•Nthéorique : Valeur théorique codée sur 10bits (0 à 1023) correspondant à la tension VADC0.

• Npratique : Valeur pratique codée sur 10bits (0 à 1023) correspondant à la tension VADC0.

• G : Gain du pont diviseur.



Nous pouvons voir que la valeur obtenue codée sur 10bits en fonction de la tension d'alimentation n'est pas exactement linéaire. La fonction correspondante à cette courbe est la suivante :

$$f(x)=0.00014395*x^2-0.1052532*x+51.870777$$

2.5. Test de l'alimentation

Objectif du test

L'objectif de ce test est de vérifier le bon fonctionnement de l'alimentation de notre carte réalisée à l'aide du régulateur TEN 5-4823. Le test sera directement fait sur la carte réalisée par M. LEQUEU.

Schéma de mesure

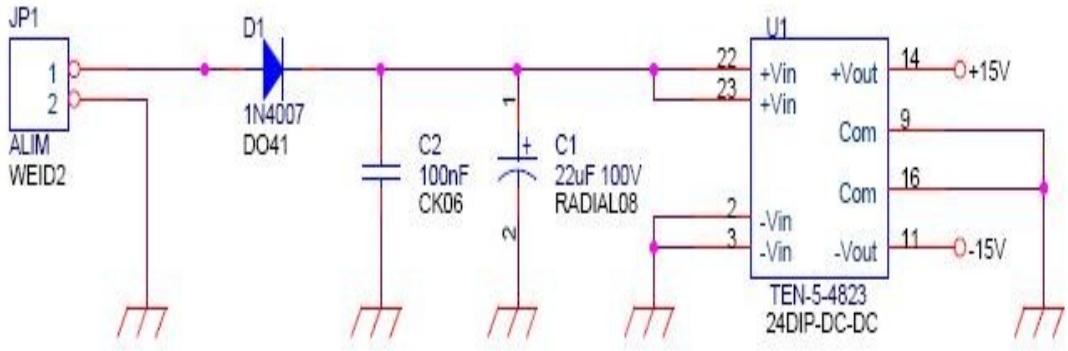


Illustration 9: Schéma de l'alimentation réalisé sous ORCAD[6]

Préparation du test

Point d'entrée (JP1) : La tension d'alimentation appliquée à JP1 est réalisée à l'aide du régulateur TEN 5-4823 décrit précédemment.

Procédure de test

Appareil de mesure nécessaire : un oscilloscope

Appliquer une tension d'entrée sur JP1 pouvant varier de 36V à 75V et la faire varier jusqu'aux valeurs limites.

Critères d'évaluation

Les tensions de sorties +Vout doit être égale à +15V et -Vout égale à -15V quelque soit la tension appliquée en entrée sur JP1. La diode électroluminescente doit également s'allumer.

Rapport du test

Résultat du test : **Accepté**

Compte-rendu

Les tensions qui ont été obtenue en sortie sur +Vout et -Vout sont bien égales à +15V et -15V pour toute la plage de tension d'entrée (36V à 75V).

3.PROGRAMMATION

Avant de débuter la programmation, quelques configurations sont nécessaires dans CodeVision AVR afin de permettre au logiciel de faire toute une partir d'initialisation.

• Il faut premièrement commencer par configurer grâce au CodeWizard le microcontrôleur ATMega 8535, il faut également lui donner la valeur de la fréquence du quartz qui dans notre cas est de 16MHz.

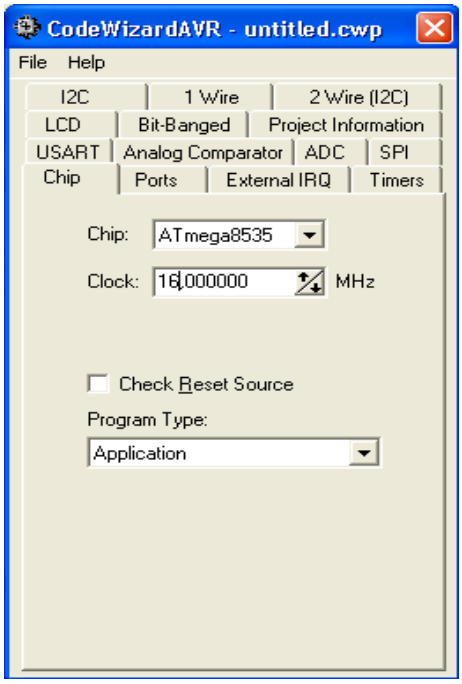


Illustration 10: Fenêtre n°1 de configuration

• Il faut ensuite configurer le port du microcontrôleur relié à l'écran LCD.

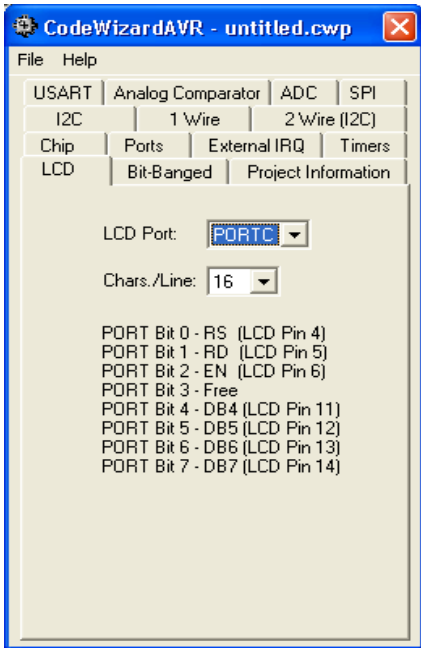


Illustration 11: Fenêtre n°2 de configuration

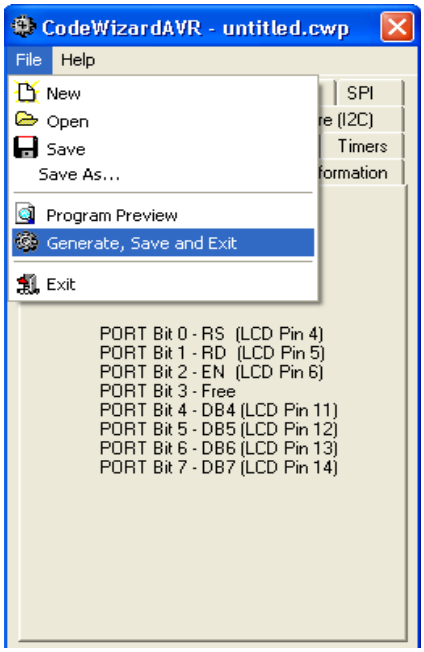


Illustration 12: Fenêtre n°3 de configuration

• Une fois ces étapes terminées, il faut faire 'Generate, Save and Exit' et nous obtenons la fenêtres de programmation suivante :

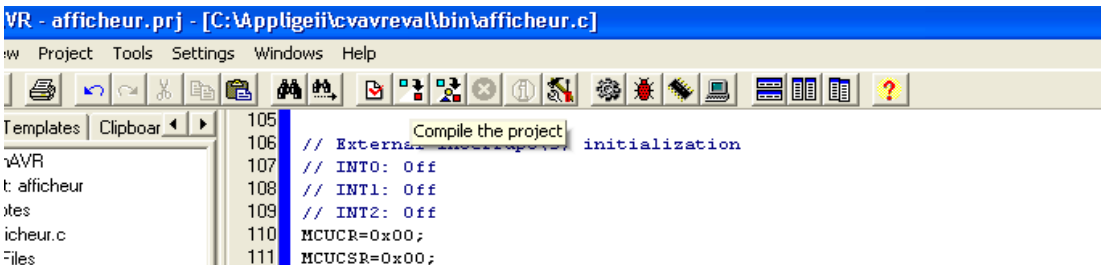


Illustration 13: Fenêtre de programmation

• La programmation peut ensuite démarrer.

3.1.Affichage de la température

```
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);
}
```

Nous mettons dans la variable 'temp' la température reçue à l'aide de la fonction 'lm75_temperature_10(0)', ensuite nous affichons le signe + ou – suivant si la température est positive ou négative.

3.2.Affichage du courant

```
While(1)
{     // Mesure Courant
       Ibatp=((read_adc(1)*4)/818.4)*200/4;  // de 0 à 818
       Ibatm=((read_adc(2)*4)/818.4)*200/4;
        if((Ibatp)>0)
        {
          sprintf(tampon,"+%dA  ",Ibatp);
        }
        else
        {
          sprintf(tampon,"-%dA  ",Ibatm);
        }
        lcd_gotoxy(2,0);
        lcd_puts(tampon);
}
```

Les entrées du microcontrôleur peuvent recevoir une tension comprise entre 0 et 5V. Cette tension est codée sur 10 bits (de 0 à 1023). Le capteur de courant fournit une tension de +4V en valeur absolue ce qui correspond à 818 en décimal. La fonction 'read_adc' récupère la valeur de la tension fournit par le capteur. Grâce à un produit en croix, nous mettons dans 'Ibatp' et 'Ibatm' la valeur du courant positive et négative. Ensuite le signe + ou – est ajouté suivant si 'Ibatp' est positif ou non.

3.3.Affichage de la tension

```
While(1)
{     // Mesure 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
 }
```

La valeur codée sur 10 bits de la tension est mise dans 'i' qui est ensuite multipliée par le gain (0,071). Enfin dans 'Vbat', nous mettons la valeur finale de la tension d'alimentation à l'aide de la formule trouvée précédemment.

3.4.Affichage de la puissance

```
While(1)
{   //Calcul puissance
      if((Ibatp)>0)
       {
        P1=P;
        P=Vbat*Ibatp;
       }
       else
       {
       P1=P;
        P=Vbat*Ibatm;
       }
      sprintf(tampon,"%d W",P);
       lcd_gotoxy(10,1);
      lcd_puts(tampon);    }
```

La puissance est calculée par la formule 'P=Vbat*Ibatp' si 'Ibatp' est positif et 'P=Vbat*Ibatm' si 'Ibatp' est négatif. Nous mettons la valeur de 'P' dans 'P1' pour pouvoir faire l'affichage de l'énergie décrite ci-après.

3.5.Affichage de l'énergie

```
While(1)
{    //calcul énergie
       (float)E=(float)E+((P1+P)/2.0)*0.1/100;
       sprintf(tampon,"%dKWs",E/10);
       lcd_gotoxy(8,2);
       lcd_puts(tampon);
}
```

L'énergie est le calcul de la moyenne de la valeur précédente de la puissance avec la valeur actuelle multiplié par un Δt.

Afin d'afficher l'énergie en Kws nous avons divisé l'énergie par 1000. Cependant, nous nous sommes aperçus qu'en divisant cette valeur par 1000, celle-ci était trop petite. Nous avons donc choisi de ne diviser que par 100 l'energie et d'afficher ensuite la valeur divisée par 10, ainsi la variable ne s'en trouve pas modifiée et nous observons bien sur l'afficheur l'énergie en Kws.

3.6.Détection bouton poussoir

```
While(1)
{
//Détection bouton poussoir
        if(PIND.3==0)
        {
         delay_ms(100);
         E=0;
         P=0;
        lcd_gotoxy(0,2);
        lcd_putsf("Energie:       ");
        }
}
```

PIND.3 correspond à la patte 3 du PORTD du microcontrôleur sur laquelle est relié le bouton poussoir. S'il y a un appui sur le bouton, la valeur de l'énergie 'E' et de la puissance 'P' est remise à zéro.

CONCLUSION

Nous avons donc étudié lors de ce projet le développement d'un wattmètre-énergimètre. Nous avons effectué ce projet pendant une période de 8 semaines. Ceci nous a permis de mettre en œuvre et d'approfondir les techniques utilisées durant le semestre 3 et de trouver nos propres solutions aux problèmes rencontrés. De plus, nous avons aussi réalisé la programmation d'un microcontrôleur en langage C à l'aide du logiciel CodeVision AVR.

Index des illustrations

Bibliographie

1: , Afficheur LCD 16x4 caractères, 08/03/2008, http://www.conrad.fr

3: , Schéma sonde LM75, 02/03/2008, http://www.national.com/mpf/LM/LM75.html

2: , Sonde de température LM75, 08/03/2008,
http://www.sprut.de/electronic/pic/programm/thermo75/lm75.jpg

4: , Capteur de courant HAS200, 02/03/2008, http://www.lem.com/docs/products/has50_600s_e.pdf

5: , Schéma régulateur LM5274M, 24/02/2008, http://www.radiospares.fr

6: , Schéma régulateur TEN 5-4823, 24/02/2008, http://www.thierry-lequeu.fr/

7: , ATMEGA 8535, 22/02/2008,
http://shop.embedit.de/gen_image.php?img=mega32pi2.jpg&type=fv

8: , Schéma mesure du courant, 20/02/2008, http://www.thierry-lequeu.fr/

Annexe 1

Planning

Semaine n°	4	5	6	7	8	9	10	11	12
Compréhension du projet Cahier des charges									
Étude et test du capteur de courant									
Schéma sous ORCAD									
Réalisation du typon sous LAYOUT									
Réalisation de la carte									
Test de la carte									
Programmation									
Finalisation + implémentation									
Rédaction du rapport									

Planning prévisionnel
Planning réel

Annexe 2

Schéma électrique de la carte (partie 1)

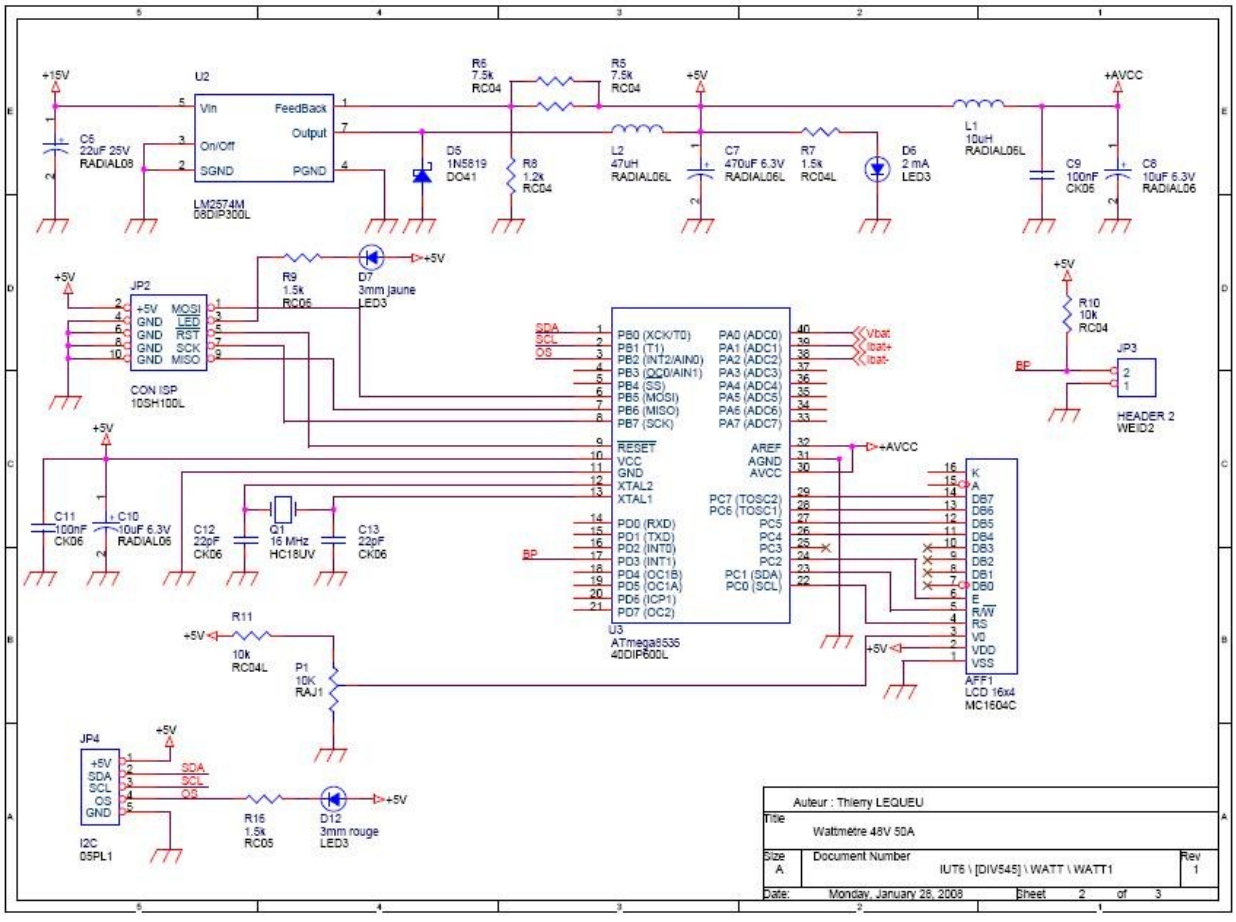


Schéma électrique de la carte (partie 2)

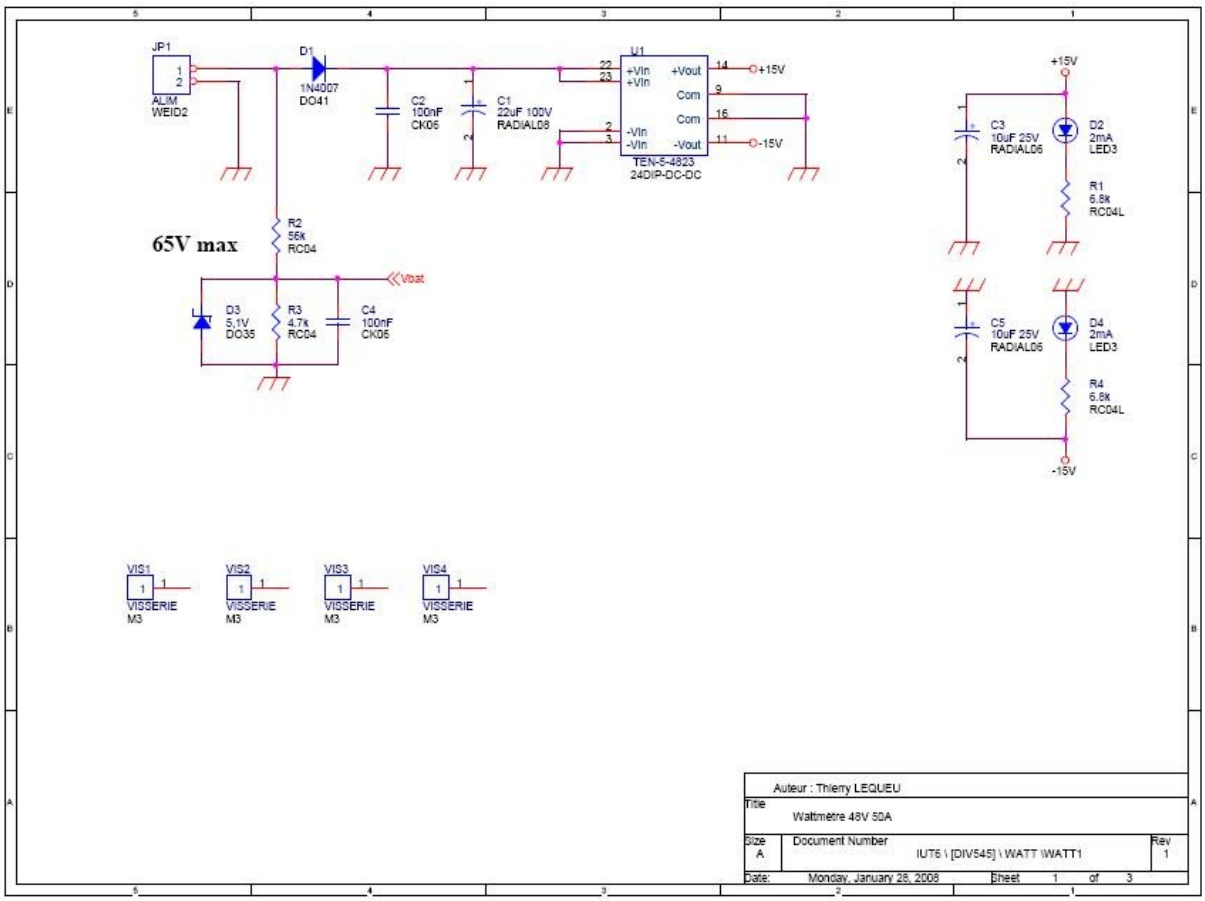
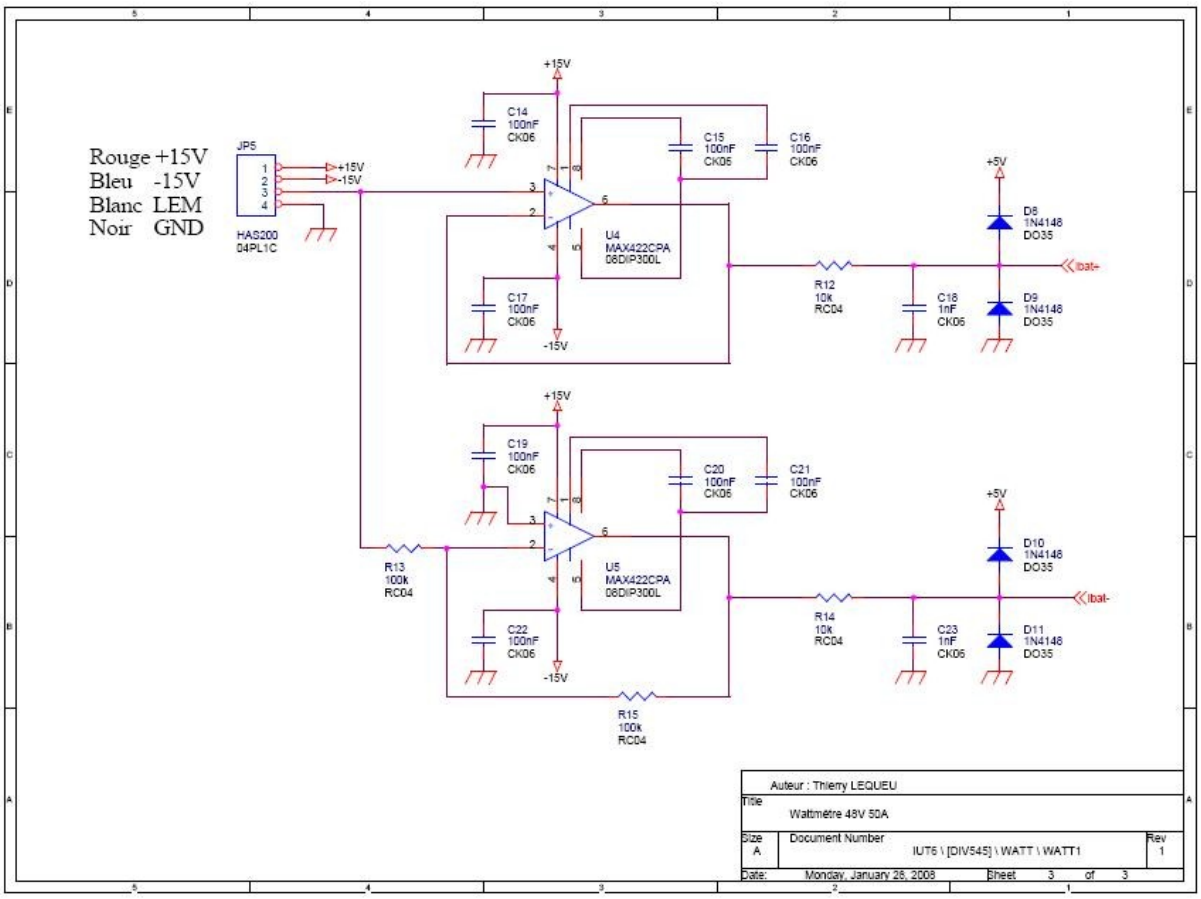


Schéma électrique de la carte (partie 3)



Annexe 3

Programme

```
/*****************************************************
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    : 13/02/2008
Author  : F4CG
Company : F4CG
Comments:


Chip type           : ATmega8535
Program type        : Application
Clock frequency     : 16,000000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 128
*****************************************************/

#include <mega8535.h>


// I2C Bus functions
#asm
   .equ __i2c_port=0x18 ;PORTB
```

```
   .equ __sda_bit=0
   .equ __scl_bit=1
#endasm
#include <i2c.h>

#include<delay.h>
#include<stdio.h>

// LM75 Temperature Sensor functions
#include <lm75.h>

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

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

#include<stdio.h>
#include<delay.h>
```

```
// Declare your global variables here

void main(void)
{
// Declare your local variables here
  unsigned char tampon[20];
  unsigned int i,Ibatm,Ibatp,Vbat;
  unsigned int  P,P1=0,E;
  int temp;
  char signe;

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;
```

```
// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
```

```
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
// Analog Comparator Output: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 1000,000 kHz
// ADC Voltage Reference: AREF pin
// ADC High Speed Mode: Off
// ADC Auto Trigger Source: None
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x84;
SFIOR&=0xEF;

// I2C Bus initialization
i2c_init();

// 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)
     {  //Détection bouton pousoir
        if(PIND.3==0)
        {
         delay_ms(100);

         E=0;
         P=0;
        lcd_gotoxy(0,2);
        lcd_putsf("Energie:       ");
        }
```

```
// Mesure Courant
   Ibatp=((read_adc(1)*4)/818.4)*200/4;  // de 0 à 818
   Ibatm=((read_adc(2)*4)/818.4)*200/4;

    if((Ibatp)>0)
    {
      sprintf(tampon,"+%dA  ",Ibatp);
    }
    else
    {
      sprintf(tampon,"-%dA  ",Ibatm);
    }
    lcd_gotoxy(2,0);
    lcd_puts(tampon);


// 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 Tension
   i=read_adc(0);
   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

    //Calcul puissance

        if((Ibatp)>0)
         {
          P1=P;
          P=Vbat*Ibatp;
         }
         else
         {
          P1=P;
          P=Vbat*Ibatm;
         }

        sprintf(tampon,"%d W",P);

         lcd_gotoxy(10,1);
         lcd_puts(tampon);

         //calcul énergie

         (float)E=(float)E+((P1+P)/2.0)*0.1/100;

         sprintf(tampon,"%dKWs",E/10);

         lcd_gotoxy(8,2);
         lcd_puts(tampon);

         delay_ms(100);

     };
}
```