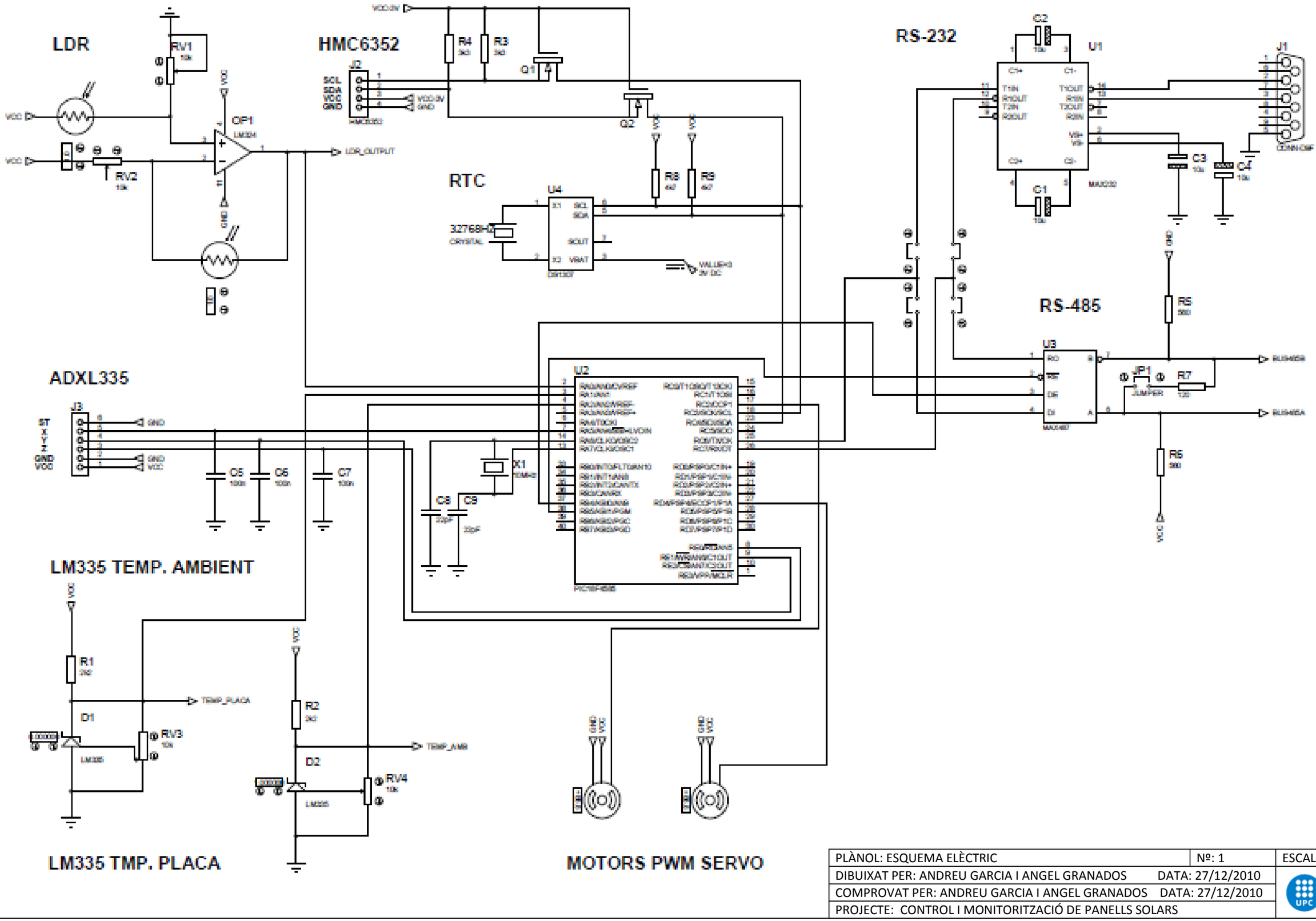
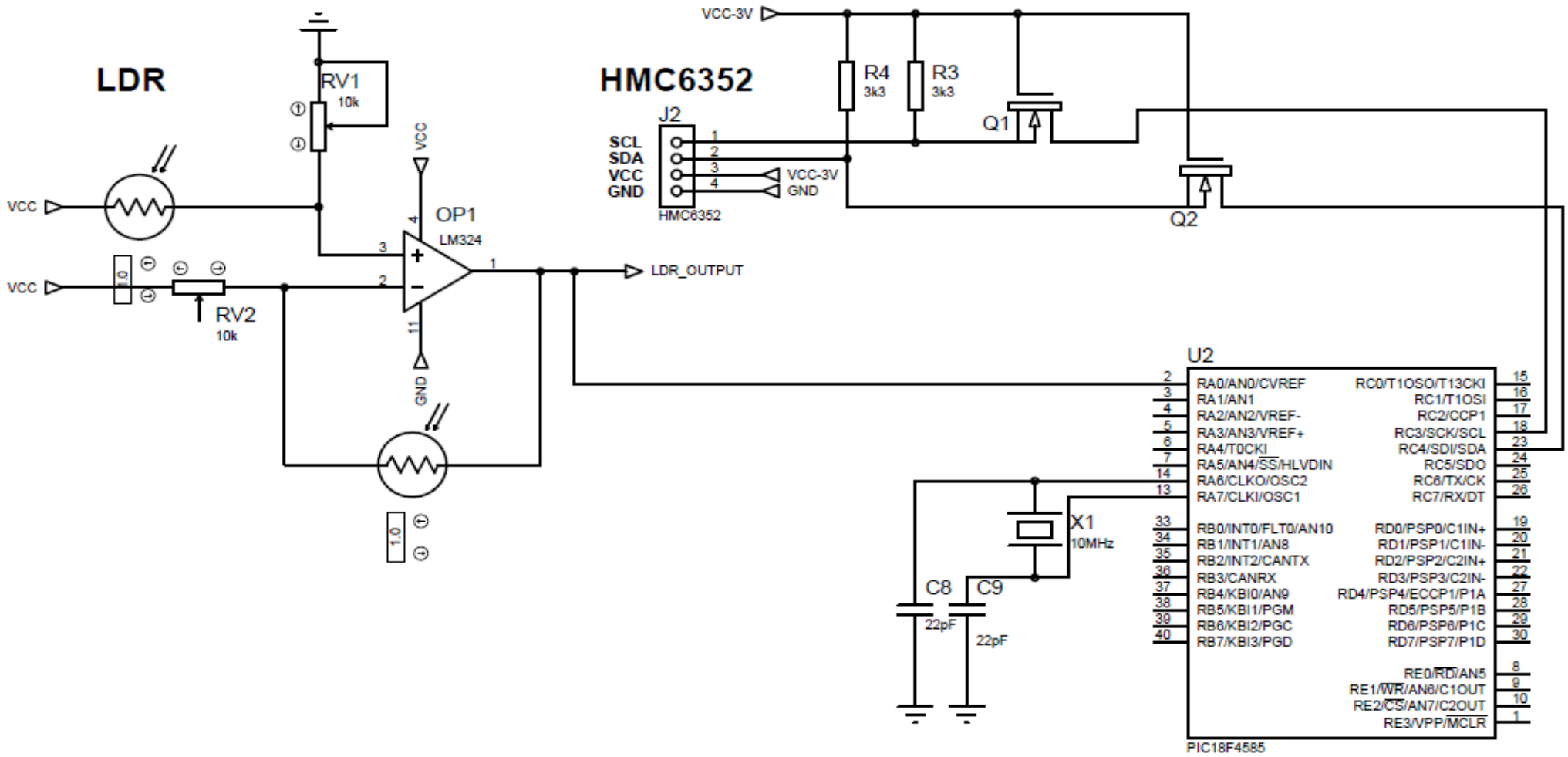
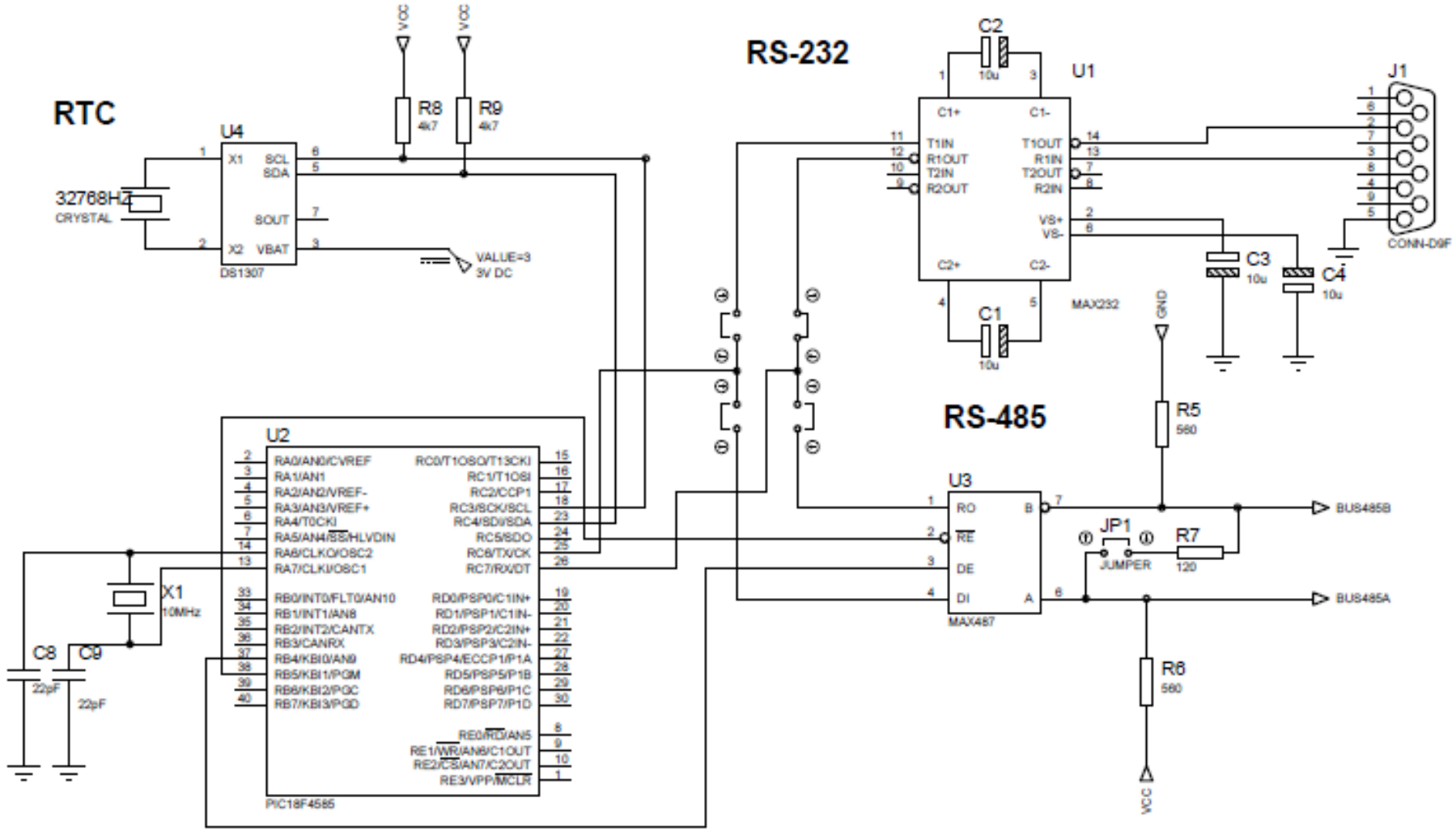
ÍNDEX MEMÒRIA



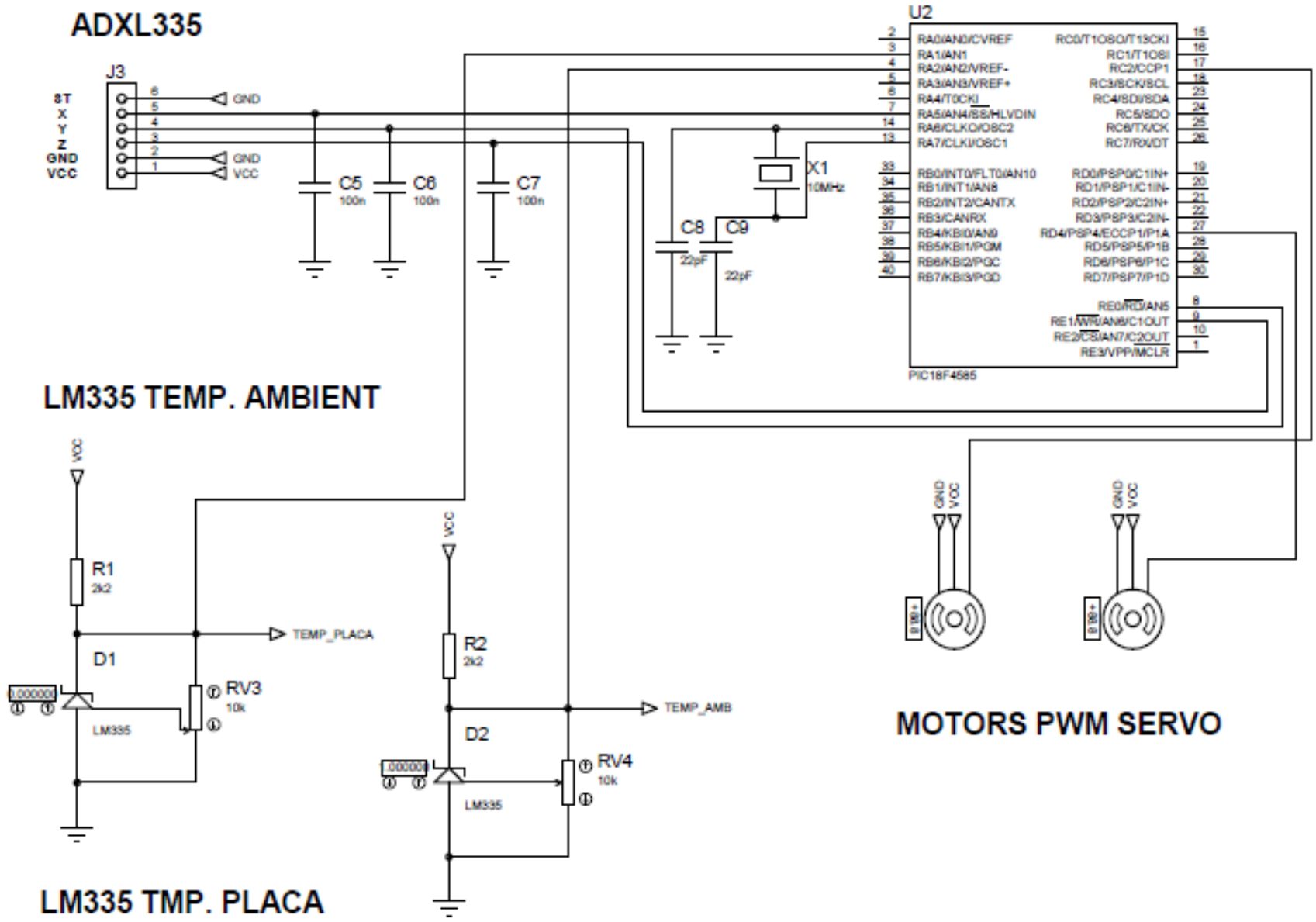
PLÀNOL: ESQUEMA ELÈCTRIC	Nº: 1	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS	DATA: 27/12/2010	
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS	DATA: 27/12/2010	
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



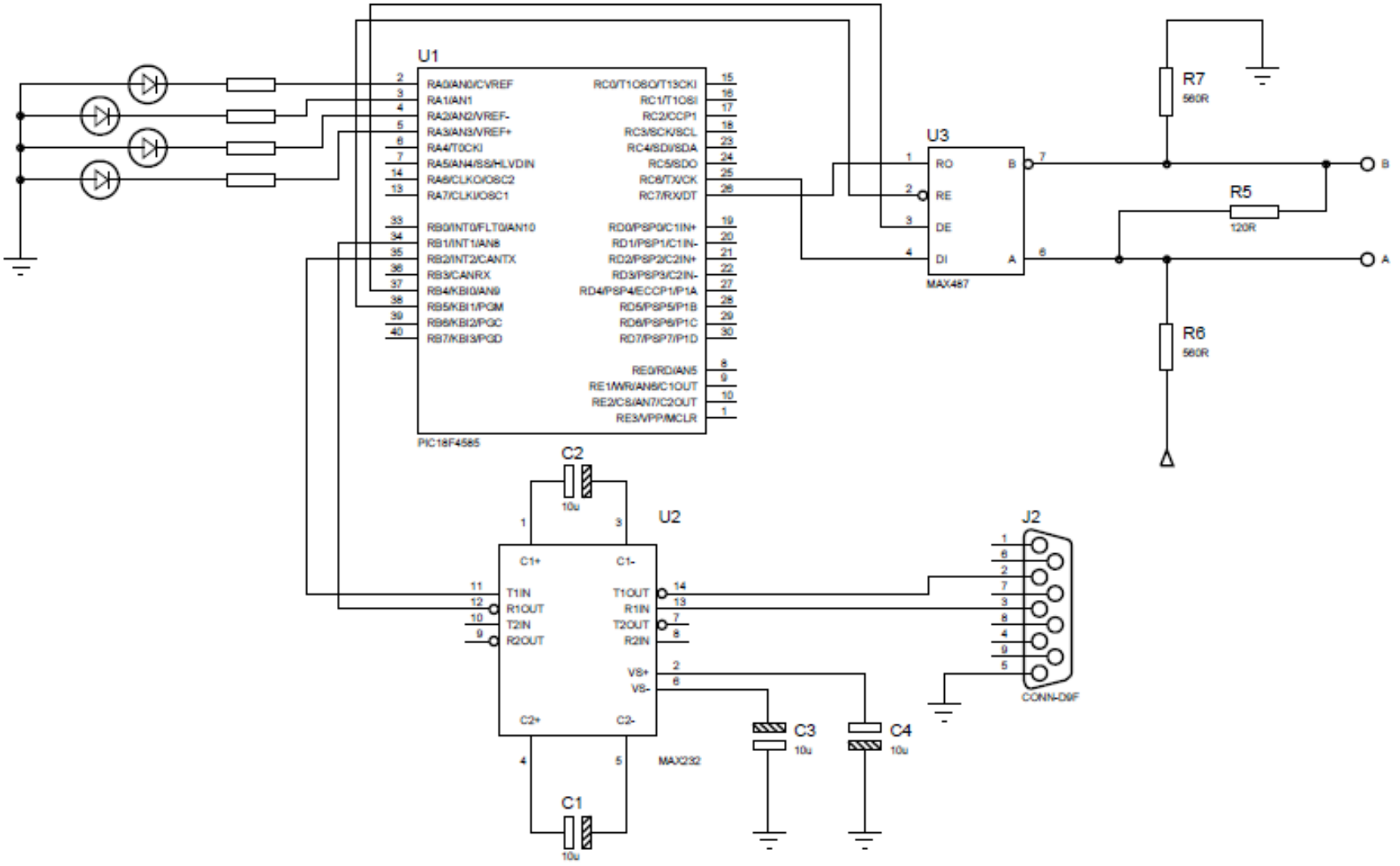
PLÀNOL: ESQUEMA ELÈCTRIC BRÚIXOLA I LDR	Nº: 2	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



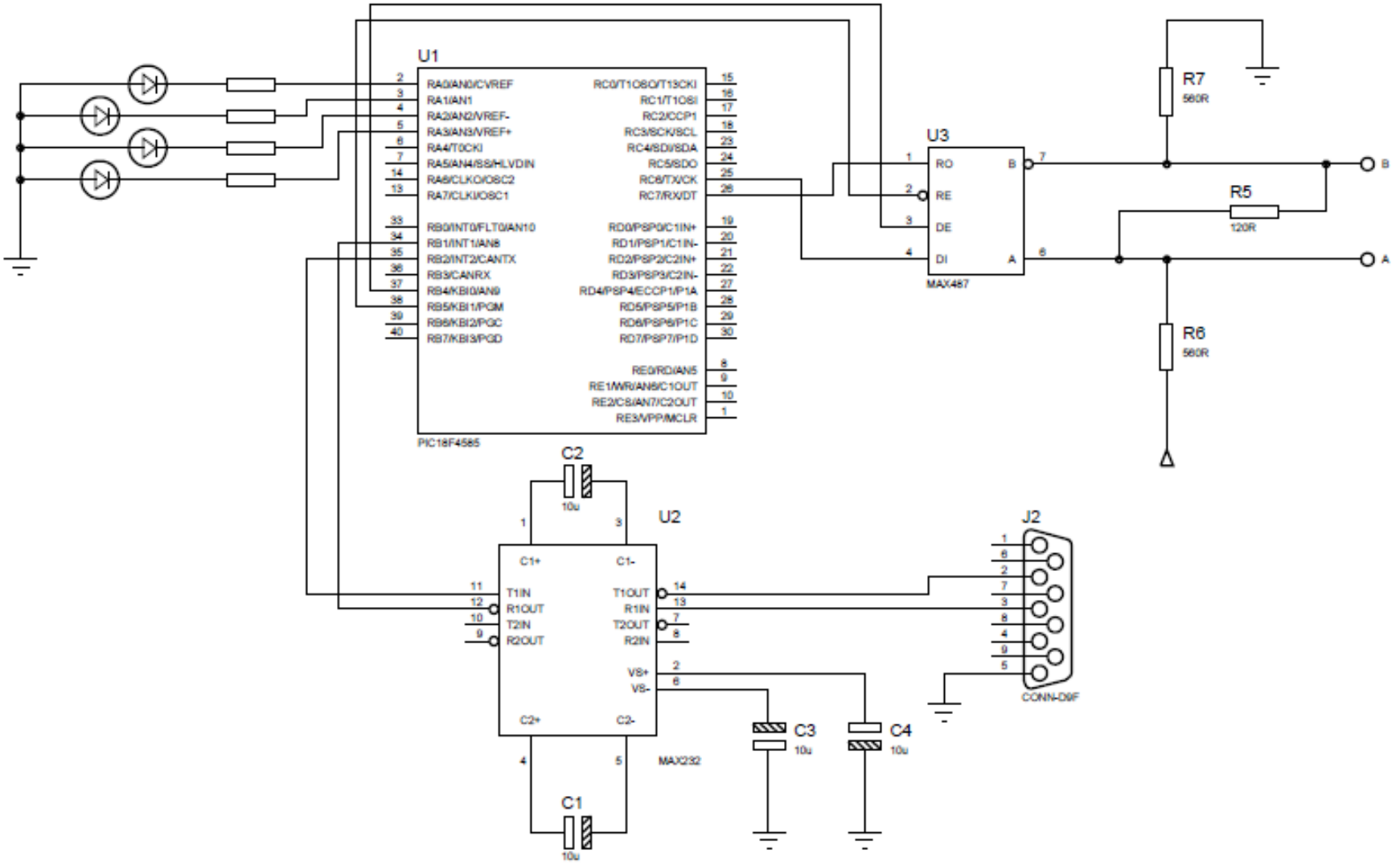
PLÀNOL: ESQUEMA ELÈCTRIC RTC I SISTEMA DE COMUNICACIONS	Nº: 3	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



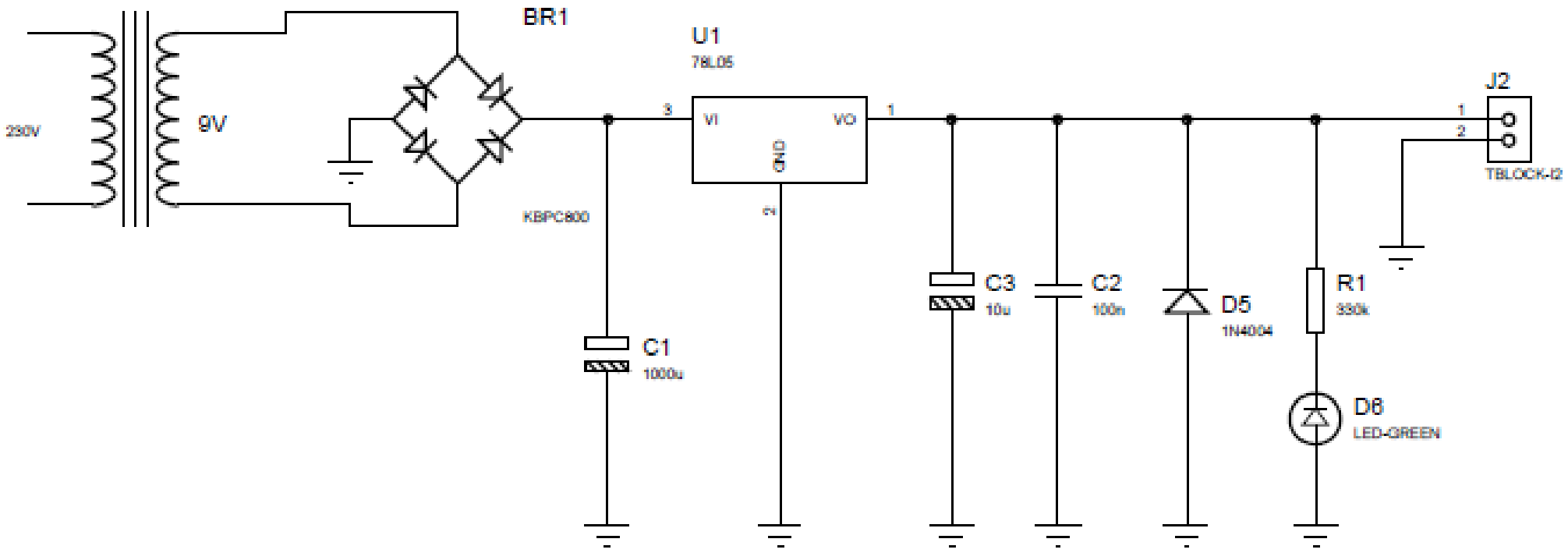
PLÀNOL: ESQUEMA ELÈCTRIC MOTORS I SENSORS TEMPERATURA I ACELEROMÈTRE	Nº: 4	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



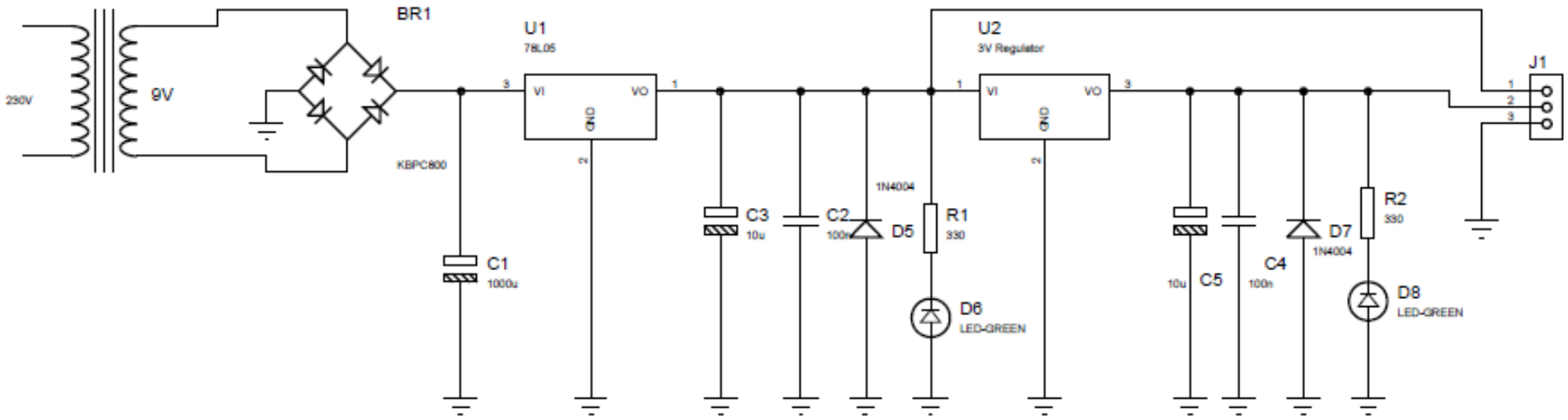
PLÀNOL: ESQUEMA ELÈCTRIC PLACA MASTER	Nº: 5	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



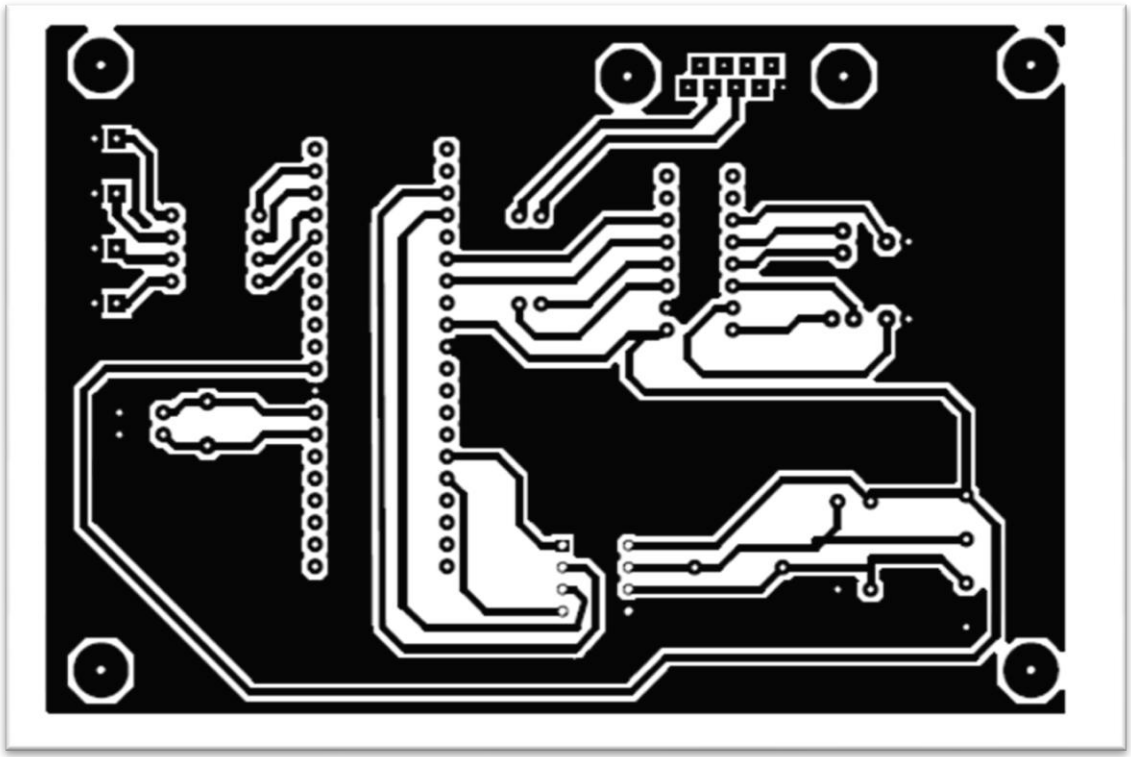
PLÀNOL: ESQUEMA ELÈCTRIC PLACA MASTER	Nº: 6	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



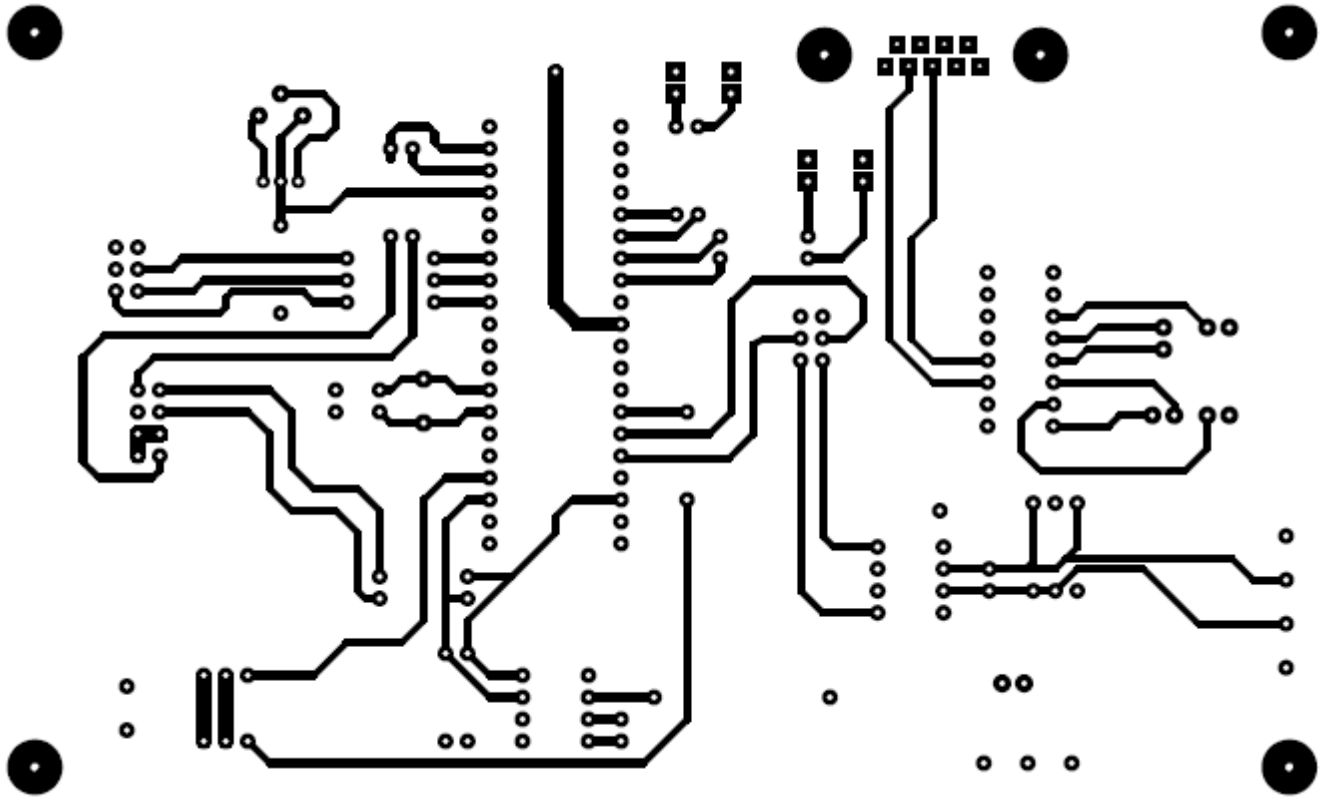
PLÀNOL: ESQUEMA ELÈCTRIC FONT D'ALIMENTACIÓ SERVOMOTORS	Nº: 7	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



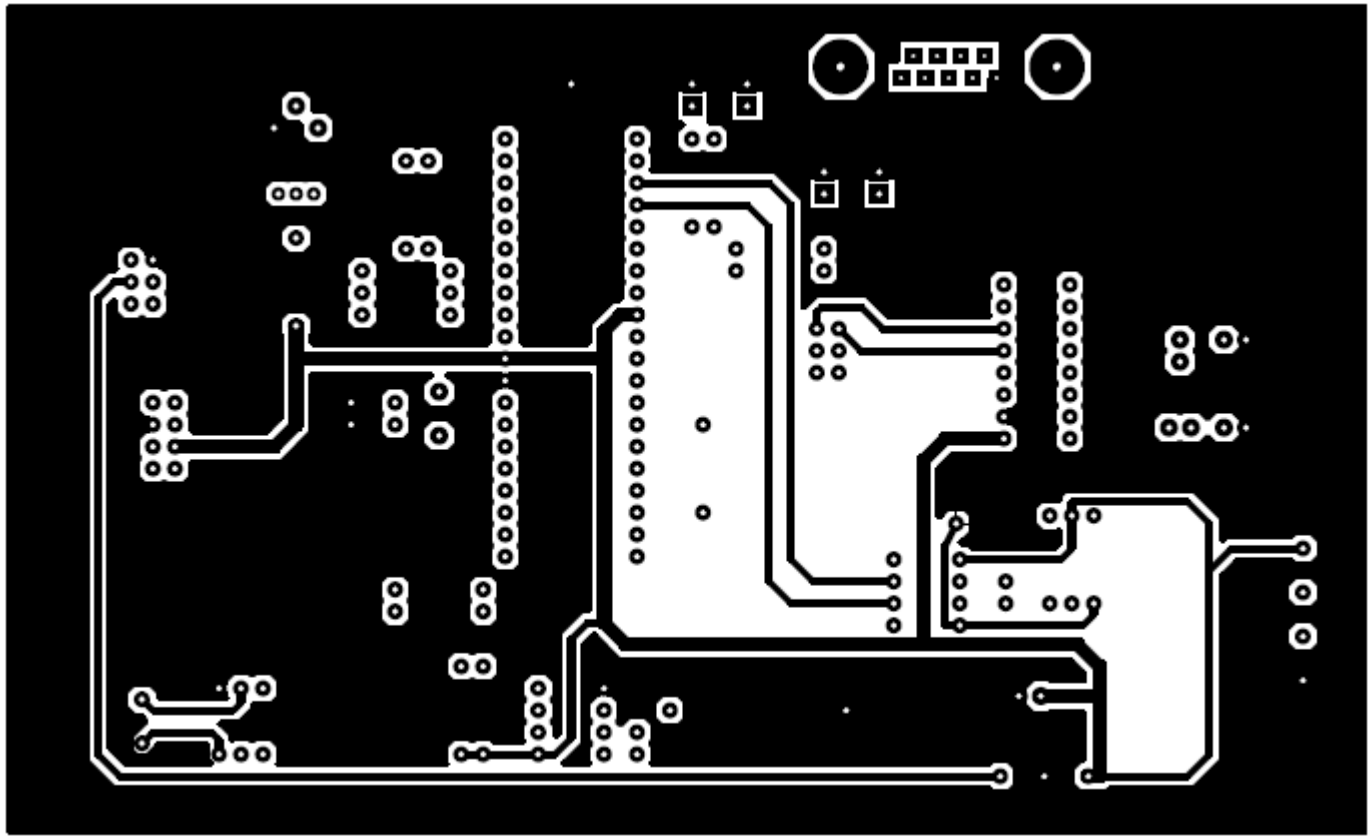
PLÀNOL: ESQUEMA ELÈCTRIC FONT D'ALIMENTACIÓ PLACA PRINCIPAL	Nº: 8	ESCALA: -
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



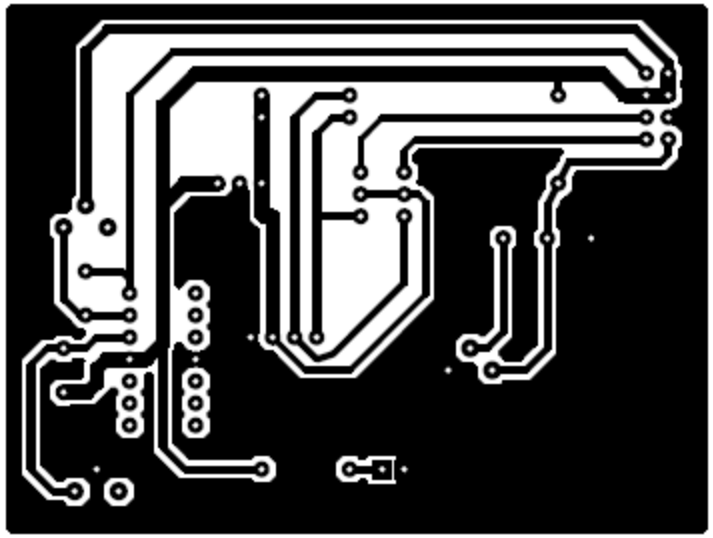
PLÀNOL: LAYOUT PLACA PRINCIPAL	Nº: 9	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



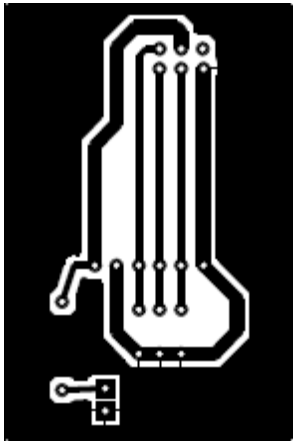
PLÀNOL: LAYOUT PLACA PRINCIPAL BOTON COPPER	Nº: 10	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



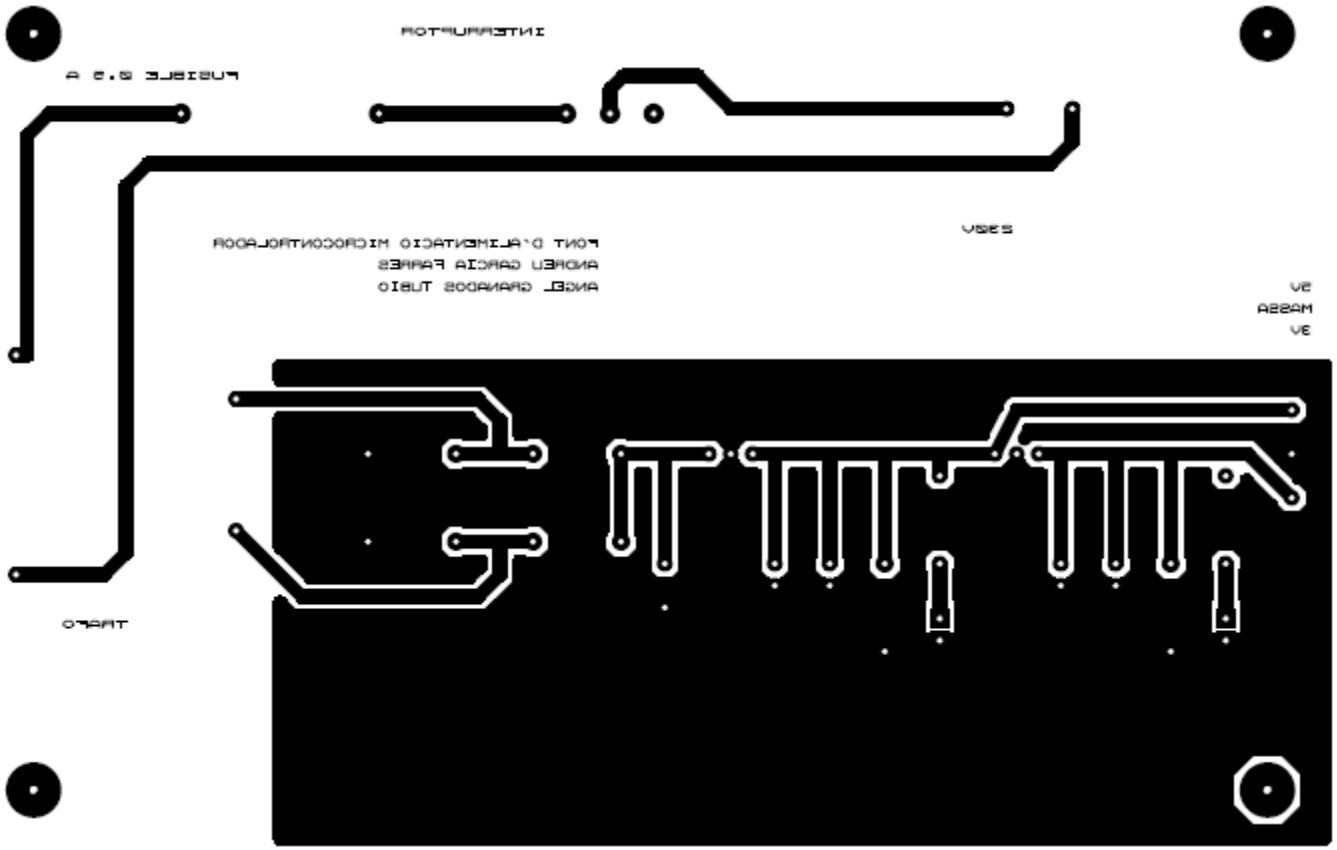
PLÀNOL: LAYOUT PLACA PRINCIPAL TOP COPPER	Nº: 11	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



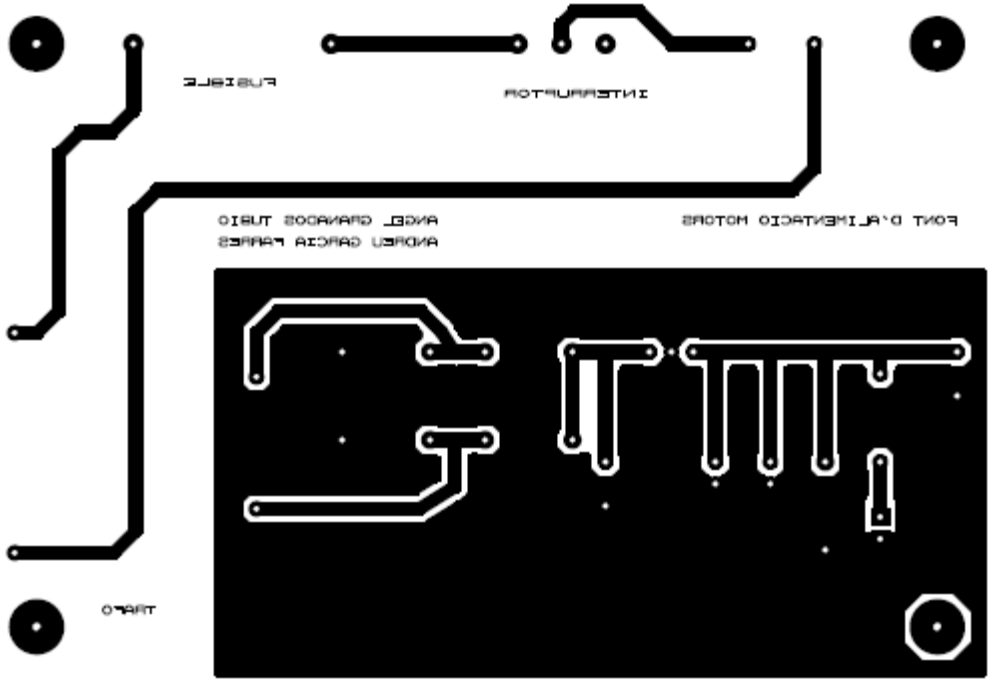
PLÀNOL: LAYOUT PLACA BRÚIXOLA I SENSORS LDR I TMP PLACA	Nº: 12	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



PLÀNOL: LAYOUT PLACA ACELERÒMETRE	Nº: 13	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



PLÀNOL: LAYOUT PLACA ALIMENTACIO PRINCIPAL	Nº: 14	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		



PLÀNOL: LAYOUT PLACA ALIMENTACIO SERVOMOTORS	Nº: 15	ESCALA: 1:1
DIBUIXAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
COMPROVAT PER: ANDREU GARCIA I ANGEL GRANADOS DATA: 27/12/2010		
PROJECTE: CONTROL I MONITORITZACIÓ DE PANELLS SOLARS		

CAPÍTOL 2: CODI CCS PLACA PRINCIPAL

2.1. Programa principal

```
#include <18f4585.h>
#device HIGH_INTS = TRUE
#device ADC = 10
#fuses HS,H4,NOMCLR,NOWDT
#use delay(clock=40000000)
#include <math.h>                                   //Llibreria Funcions matematiques
#include <i2c.c>                                    //Llibreria Bruixola
#include <ds1307.c>                                 //Llibreria RTC
#include <ecuacions.c>                              //Llibreria Ecuacions
#include <rs485.c>                                  //Llibreria Drivers rs485

#use rs232 (baud=9600,xmit=pin_c6, rcv=pin_c7, bits=8,STREAM = PC)

#BYTE TRISA = 0xF92                                 //DEFINICIO ENTRADES I SORTIDES
#BYTE TRISC = 0xF94
#BYTE TRISB = 0xF93
#BYTE TRISE = 0xF96
#BYTE TRISD = 0xF95

#BYTE PORTA = 0xF80                                 //DEFINICIO DIRECCIÓ PORTS
#BYTE PORTB = 0xF81
```

```
#BYTE PORTC = 0xF82
#BYTE PORTD = 0xF83
#BYTE PORTE = 0xF84


#bit  LED1 = 0xF81.0                                                    //PINS DEL MICRO
#bit  LED3 = 0xF81.2
#bit  LED5 = 0xF81.1
#bit  LED6 = 0xF81.3
#bit  PWM_PIN=0xF82.2
#bit  PWM_PIN2=0xF83.4

//////////////////////////// CONSTANTES ////////////////////////////////////////////////////////////

int const lenbuff=32;                                                   // Longitud de buffer, Ajustar

//////////////////////////// VARIABLES EN RAM ///////////////////////////////////////////////////////

int xbuff=0x00;                                                          // Índice: siguiente char en cbuff
char cbuff[lenbuff];                                                    // Buffer
char rcvchar=0x00;                                                      // último carácter recibido
char flag = 0;

///////////////// DECLARACION FUNCION INTERRUPCION RS232/////////////////////////////////////

void inicbuff(void);                                                                    // Borra buffer
int addcbuff(char c);                                                   // añade carácter recibido al buffer

//////////////////DEFINICIO DE VARIBALES GLOBALES/////////////////////////////////////////////////

int8  day, month, yr, hrs, min, sec,dow ;
int16 e=0;
int16 contador = 0;
int16 contador2 = 0;
int16 duty_cicle = 40;
int16 contador3 = 0;
int16 duty_cicle2 = 40;
int16 dada = 0;
int16 lectura_bruixola = 0;
float altura_medida = 0;
float altura_sol = 0;
int16 dif1 = 0;
int16 dif2 = 0;

////////////////////////////////////INT TIMER 1/////////////////////////////////////////////////////
#int_TIMER1 HIGH
void timer1_interrupt() {          //Generació senyal PWM motors
  set_timer1(65336);
  contador++;
  contador2++;
  contador3++;
  LED1=1;
  if (contador == 808)
    {contador=0;
     contador2=0;
     contador3=0;
     PWM_PIN=1;
     PWM_PIN2=1;}
  if (contador2 >= duty_cicle)
    PWM_PIN=0;
```

```
  if (contador3 >= duty_cicle2)
    PWM_PIN2=0;
  LED1=0;
}

//////////////////////////////////// INT RDA  ////////////////////////////////////////////////////////////

#INT_RDA
void serie_interrupt () {
   rcvchar=0x00;                                                    // Inicializo carácter recibido
   if(kbhit())
   {                                                          // Si hay algo pendiente de recibir ...
     rcvchar=getc();                                                          // lo descargo y ...
     addcbuff(rcvchar);                                              // lo añado al buffer y ...
     flag = 1;
   }
}
void inicbuff(void){                                                          // Inicia a \0 cbuff
 int i;
 int count=32;
 for(i=0;i<count;i++){                                          // Bucle que pone a 0 todos los
   cbuff[i]=0x00;                                                  // caracteres en el buffer
 }
 xbuff=0x00;                                                  // Inicializo el índice de siguiente
                                                                                  // carácter
}
int addcbuff(char c){                                          // Añade a cbuff ----------------------
     cbuff[xbuff++]=c;                                          // Añade carácter recibido al Buffer
 }

//////////////////////////////////LECTURA ADC ////////////////////////////////////////////////////////////

void llegir_adc ()                  // FEM 50 MOSTRES
  {
  int32 entrada_adc = 0;
  int8 j = 0;
  do{
   entrada_adc += read_adc();        // INCREMENTEN EL VALOR LLEGIT
   j = j+1;
  }
  while (j<50);
  dada = entrada_adc/50;             // OBTENIM MITJANA PER REDUIR VARIACIONS
  entrada_adc = 0;                     // TORNEM A INICIALITZAR VAR PER GUARDAR LECTURA
TEMPORAL
  }

//////////////////////////////CALCUL EQUACIONS ASTRONÒMIQUES/////////////////////////////////////

void calcul_ecuacions ()
  {
  dia_julia = calcul_dia_julia(day,month);
  angle_diari = calcul_angle_diari(dia_julia);
  declinacio = calcul_declinacio(dia_julia);
  ET = ecuacio_temps(angle_diari);
  w = calcul_angle_solar_horari (hrs, min, sec, ET, longitud);
  altura_solar = calcul_altura_solar(hrs,min,sec,w,declinacio,latitud);
  altura_sol = 90-altura_solar;
  altura_sol=90-altura_sol;
    if (altura_sol<0)
       {altura_sol=0;}
```

```
  angle_azimut = (calcul_angle_azimut(hrs,min,sec,w,altura_solar,declinacio,latitud));
  }



//////////////////////////PROGRAMA PRINCIPAL/////////////////////////////////////////////////////

void main()

{  //VARIABLES LOCALS DEL PROGRAMA PRINCIPAL
  int8 vent = 0;
  int8 j=0;
  int16 lectura_ldr = 0, lectura_temp = 0, lectura_temp2 = 0, accelY = 0, accelZ = 0;
  int16 LDR = 0, TEMP = 0, TEMP2 = 0;
  int8 ldr_high=0,temp_low=0,temp_high=0,temp_low2=0,temp_high2=0,ldr_low=0;
  int8 buffer[16];
  char A;
  disable_interrupts(global);                          // DESABILITACIO INTERRUPCIONS
  TRISA = 0xFF;
  TRISD = 0x80;                                                  // 1 ENTRADA ; 0 SORTIDA
  TRISC = 0x98;                                    // DEFINIM ENTRADES I SORTIDES PORTS
  TRISB = 0x02;                                                        // UTILITZATS

  // CONFIGURACIO PORTS ANALOGICS
  setup_adc_ports(AN0_TO_AN8);                              // AN0 es donde mides ADC
  setup_adc_ports(VSS_VDD);                                   // Tensions de refrència
  setup_adc(ADC_CLOCK_DIV_8);                                   // Clock rellotge ADC

  // INITZACIO RTC
  ds1307_init(DS1307_OUT_ON_DISABLED_HIHG|DS1307_OUT_ENABLED| S1307_OUT_8_KHZ);

  // VALORS INICIALS DUTY CICLE
  duty_cicle=85;
  duty_cicle2=80;
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);                  //CONFIGURACIO TIMER 1
  set_timer1(65336);                                           //CARREGA TIMER 1
  PWM_PIN = 1;
  PWM_PIN2 = 1;
  e=47;

  calcul_ecuacions ();
  enable_interrupts(INT_TIMER1);                  // HABILITACIO INTERRUPCIO TIMER1
  enable_interrupts(INT_RDA);                      // HABILITACIO INTERRUPCIO SERIE
  enable_interrupts(global);                        // HABILITACIO DE INTERRUPCIONS

do
    {
    if (flag ==1) {                          //COMPROBA SI A SALTAT LA INTERRUPCIÓ SÈRIE

       if (cbuff[0]=='A')
       {
       ds1307_set_date_time(cbuff[1],cbuff[2],cbuff[3],1,cbuff[4],cbuff[5],cbuff[6]);
       }
       else
         {
         if (cbuff[0]=='B')
         {
         latitud = cbuff[1]+ (float)cbuff[2]/60 + (float)cbuff[3]/3600;
         longitud = cbuff[4]+ (float)cbuff[5]/60 + (float)cbuff[6]/3600;
         }
```

```
        }
      inicbuff();
      flag = 0;
   }

   e++;

   if(e==100)
   {
   LED3=1;
   delay_ms(500);                                  // CÀLCUL ECUACIONS I REGULACIÓ POSICIÓ
   calcul_ecuacions ();
   e = 0;
   LED3=0;
   if (lectura_ldr>=300)
     {
     if (altura_medida>(altura_sol+10))
        {
        duty_cicle=duty_cicle-2;
        delay_ms(50);
        }

     if (altura_medida<(altura_sol-10))
        {
        duty_cicle=duty_cicle+2;
        delay_ms(50);
        }

     if (altura_medida>(altura_sol+5))
        {
        duty_cicle=duty_cicle-1;
        LED5=1;
        delay_ms(100);
        }
        else
          {
          if (altura_medida<(altura_sol-5))
          {
          duty_cicle=duty_cicle+1;
          LED5=0;
          delay_ms(100);
          }
          else
          duty_cicle=duty_cicle;
          }
     }
     else
        duty_cicle =  80;

      angle_azimut=(angle_azimut*-1)+180;

      if(angle_azimut<180)
        { angle_azimut=angle_azimut+180;}
       else
        {angle_azimut=angle_azimut-180;}

if (angle_azimut<=90)
 {
  if(lectura_bruixola<=90)
  {
```

```
      if (lectura_bruixola>(angle_azimut+20))
        {
        duty_cicle2=duty_cicle2-3;
        delay_ms(100);
        }
        else
          {
          if (lectura_bruixola<(angle_azimut-20))
          {duty_cicle2=duty_cicle2+3;
          delay_ms(100);
          }
          }

      if (lectura_bruixola>(angle_azimut+4))
        {
        duty_cicle2=duty_cicle2-1;
        LED6=1;
        delay_ms(100);
        }
        else
          {
          if (lectura_bruixola<(angle_azimut-4))
          {duty_cicle2=duty_cicle2+1;
          LED6=0;
          delay_ms(100);
          }
          else
          duty_cicle2=duty_cicle2;
          }
 }

   else
   {
   duty_cicle2=duty_cicle2+5;
   }
 }
else
{
if(angle_azimut>=270)
{
 if (lectura_bruixola<=90)
 {
 duty_cicle2=duty_cicle2-5;
 }
 else
 {
      if (lectura_bruixola>(angle_azimut+20))
        {
        duty_cicle2=duty_cicle2-3;
        delay_ms(100);
        }
      if (lectura_bruixola<(angle_azimut-20))
          {duty_cicle2=duty_cicle2+3;
          delay_ms(100);
          }
      if (lectura_bruixola>(angle_azimut+4))
        {
        duty_cicle2=duty_cicle2-1;
        LED6=1;
```

```
        delay_ms(100);
        }
      if (lectura_bruixola<(angle_azimut-4))
         {duty_cicle2=duty_cicle2+1;
         LED6=0;
         delay_ms(100);
         }
         else
         duty_cicle2=duty_cicle2;
 }
}
else
{
duty_cicle2=duty_cicle2;
}
}
  }
  LED1 = 1;

  //LECTURA DELS SENSORS CONTROL I MONITORITZACIÓ
  set_adc_channel(0);                                          // SENSOR LDR
  delay_ms(10);
  llegir_adc();
  LDR = dada;
  lectura_ldr=(900.0*LDR)/1023;
  ldr_high = make8(lectura_ldr,1);
  ldr_low = make8(lectura_ldr,0);

  set_adc_channel(1);                                          // SENSOR TEMPERATURA
  delay_ms(10);                                                // SENSIBILITAT 10mV/K
  llegir_adc();
  TEMP = dada;
  lectura_temp =((500.0*TEMP)/1023)-273;
  temp_high = make8(lectura_temp,1);
  temp_low = make8(lectura_temp,0);

  vent = rand()%30+1;                                          //SIMULACIÓ SENSOR VENT

  set_adc_channel(2);                                          // SENSOR TEMPERATURA
  delay_ms(10);                                                // SENSIBILITAT 10mV/K
  llegir_adc();
  TEMP2 = dada;
  lectura_temp2 =((500.0*TEMP2)/1023)-273;
  temp_high2 = make8(lectura_temp2,1);
  temp_low2 = make8(lectura_temp2,0);

  ds1307_get_date(day,month,yr,dow);
  ds1307_get_time(hrs,min,sec);                                // OBTENCIO HORA
  lectura_bruixola = HMC6352_read_heading()/10;                // LECTURA BRÚIXOLA


  set_adc_channel(5);                                          // SENSOR ACCELERACIO
  delay_ms(10);
  llegir_adc();
  accelY = dada;

  set_adc_channel(4);
  delay_ms(10);
  llegir_adc();
  accelZ = dada;
```

```
    altura_medida = atan2 ((float) accelZ, (float)accelY);          //CALCUL ANGLE ALTURA SOLAR
    altura_medida = (altura_medida-0.68)*450;


       //PREPARACIÓ ARRAY BITS ENVIO PER PORT SÈRIE O MAX485
       A='A';
       buffer[0]='S';
       buffer[1]=ldr_low;
       buffer[2]=ldr_high;
       buffer[3]=temp_low;
       buffer[4]=temp_high;
       buffer[5]=vent;
       buffer[6]=temp_low2;
       buffer[7]=temp_high2;
       buffer[8]='B';
       buffer[9]=day;
       buffer[10]=month;
       buffer[11]=yr;
       buffer[12]=hrs;
       buffer[13]=min;
       buffer[14]=sec;
       buffer[15]='F';

while (j<16)                                                        //ENVIO PEL PORT
    {
       putc(buffer[j]);
       j++;
    }
    j=0;
    LED1=0;
  }
  while (true);
  }
```

2.2. Llibreria equacions

```
//////////////////////////////////////////////////////////////////////////////////////////////
///////////////                  ECUACIONS.C                        ////////////////////
/////////////////           CALCULS ECUACIONS ASTRONOMIQUES         ////////////////////
//////////////////////////////////////////////////////////////////////////////////////////////
///// calcul_dia_julia (int8 dia, int8 mes)////////////////////////////////////////////////////
///// calcul_angle_diari (int16 dia_julia///////////////////////////////////////////////////////
///// calcul_declinacio (float dia_julia) //////////////////////////////////////////////////////
///// calcul_angle_solar_horari (int8 hora, int minutos, int8 segons, float ET, float longitud)//////
///// calcul_angle_azimut (float altura_solar, float declinacio, float latitud)/////////////////////////
///// calcul_altura_solar (float w, float declinacio, float latitud)//////////////////////////////////
//////////////////////////////////////////////////////////////////////////////////////////////

#include <math.h>                                          //Llibreria matemàtiques


float longitud = 2.128941;                                 // En graus a EUTIB
float latitud = 41.388033;                                 // En graus a EUETIB

int8 dia = 0;                                              //VARIABLES A UTILITZAR
int8 mes = 0;
```

```
int8 hora = 0;
int8 minutos = 0;
int8 segons = 0;
int16 dia_julia = 0;
float angle_diari = 0;
float declinacio = 0;
float ET = 0;
float w = 0;
float altura_solar = 0;
float angle_azimut = 0;

int16 calcul_dia_julia (int8 dia, int8 mes) {                    // CALCUL DIA JULIA
  switch (mes) {
       case 1: dia_julia = dia;
              break;
       case 2: dia_julia = dia + 31;
              break;
       case 3: dia_julia = dia + 59;
              break;
       case 4: dia_julia = dia + 90;
              break;
       case 5: dia_julia = dia + 120;
              break;
       case 6: dia_julia = dia + 151;
              break;
       case 7: dia_julia = dia + 181;
              break;
       case 8: dia_julia = dia + 212;
              break;
       case 9: dia_julia = dia + 243;
              break;
       case 10: dia_julia = dia + 273;
               break;
       case 11: dia_julia = dia+304;
               break;
       case 12: dia_julia = dia + 334;
               break; }
  return (dia_julia);}

float calcul_angle_diari (int16 dia_julia) {                     //CALCUL ANGLE DIARI
  angle_diari = ((2*PI*(dia_julia-1))/365);
  return (angle_diari);}

float calcul_declinacio (float dia_julia) {                      //CALCUL DECLINACIÓ
  float x = 0;
  float dia = 0;

  dia = dia_julia;
  x = (284.0+dia)*360.0/365.0;
  x = x * PI/180.0;

  declinacio = 23.45*sin(x);
  return (declinacio);}

float ecuacio_temps (float angle_diari) {                        //CALCUL EQ. DEL TEMPS
  ET = 229.18* (0.000075 + 0.001868*cos(angle_diari) + 0.0320077*sin(2*angle_diari)-
0.014615*cos(2*angle_diari)-0.04089*sin(2*angle_diari));
  return (ET);}

float calcul_angle_solar_horari (int8 hora, int8 minutos, int8 segons, float ET, float longitud) {
```

```
  float TO = 0;

  TO = ((hora*60.0) + minutos + (segons/60.0));
  w = TO + ET + 4*(0-longitud);
  return (w);}

float calcul_altura_solar (int8 hora,int8 minutos, int8 segons, float w, float declinacio, float latitud)
{
  float decli = 0;
  float lati = 0;
  float wrad = 0;
  float altura_s =0;
  float hor = 0;
  float min = 0, sec = 0;
  min = minutos;
  sec = segons;
  hor = hora + min/60 + sec/36000;
  wrad  =  (w*(15/60)*(PI/180))+(15*PI/180*hor)-  PI;    //PASSEM DE MINUTS A RADIANS
// 1 H = 15º
  decli = declinacio * PI/180;       // PASSEM A RADIANS
  lati = latitud * PI/180;          // PASSEM A RADIANS
  altura_s = asin ((sin(decli)*sin(lati))+ (cos(decli)*cos(lati)*cos(wrad)) );
  altura_solar= altura_s*180/PI;

  return (altura_solar);
}

float calcul_angle_azimut (int8 hora, int8 minutos, int8 segons,float w,float altura_solar, float
declinacio, float latitud) {
  float decli = 0;
  float lati = 0;
  float azi = 0;
  float altura = 0;
  float wrad = 0;
  float hor = 0;
  float min = 0, sec = 0;
  min = minutos;
  sec = segons;
  hor = hora + min/60 + sec/36000;
  wrad = (w*(15/60)*(PI/180))+(15*PI/180*hor)- PI;
  altura = altura_solar*PI/180;
  decli = declinacio * PI/180;      // PASSEM A RADIANS
  lati = latitud * PI/180;         // PASSEM A RADIANS
  azi = asin ( (cos(decli)*sin(wrad)/cos (altura)));
  angle_azimut=(-1)*(azi*180/PI);
  return (angle_azimut);
}
```

2.3. Llibreria RTC

```
/////////////////////////////////////////////////////////////////////////////////////////////////
///////////////                DS1307.C                      /////////////////////////
///////////////                Driver for Real Time Clock    /////////////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
/////////////// ds1307_init()                                 //////////////////////////
//////////////- Enable oscillator without clearing the seconds register -   ////////////////////
///             used when PIC loses power and DS1307 run from 3V BAT        ////////////////////
///           - Disable squarewave output                                   ////////////////////
/// ds1307_set_date_time(day,mth,year,dow,hour,min,sec)    Set the date/time  //////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
/// ds1307_get_date(day,mth,year,dow)                      Get the date       /////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
/// ds1307_get_time(hr,min,sec)                            Get the time       /////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
#define RTC_SDA  PIN_C4
#define RTC_SCL  PIN_C3
#use i2c(master, sda=RTC_SDA, scl=RTC_SCL)
#define DS1307_ALL_DISABLED         0b00000000 // All disabled
#define DS1307_OUT_ON_DISABLED_HIHG 0b10000000 // Out to Hight on Disable Out
#define DS1307_OUT_ENABLED          0b00010000 // Out Enabled
#define DS1307_OUT_1_HZ             0b00000000 // Freq. Out to 1 Hz
#define DS1307_OUT_4_KHZ            0b00000001 // Freq. Out to 4.096 Khz
#define DS1307_OUT_8_KHZ            0b00000010 // Freq. Out to 8.192 Khz
#define DS1307_OUT_32_KHZ           0b00000011 // Freq. Out to 32.768 Khz

BYTE bin2bcd(BYTE binary_value);
BYTE bcd2bin(BYTE bcd_value);

void ds1307_init(int val)
{
      byte seconds = 0;
      i2c_start();
      i2c_write(0xD0);
      i2c_write(0x00);
      i2c_start();
      i2c_write(0xD1);
      seconds = bcd2bin(i2c_read(0));
      i2c_stop();
      seconds &= 0x7F;
      delay_us(3);
      i2c_start();
      i2c_write(0xD0);
      i2c_write(0x00);
      i2c_write(bin2bcd(seconds));
      i2c_start();
      i2c_write(0xD0);
      i2c_write(0x07);
      i2c_write(val);
      i2c_stop();
}

void ds1307_set_date_time(BYTE day, BYTE mth, BYTE year, BYTE dow, BYTE hr, BYTE min, BYTE
sec)
{
  sec &= 0x7F;
  hr &= 0x3F;
  i2c_start();
  i2c_write(0xD0);          // I2C write address
```

```
  i2c_write(0x00);          // Start at REG 0 - Seconds
  i2c_write(bin2bcd(sec));    // REG 0
  i2c_write(bin2bcd(min));    // REG 1
  i2c_write(bin2bcd(hr));     // REG 2
  i2c_write(bin2bcd(dow));    // REG 3
  i2c_write(bin2bcd(day));    // REG 4
  i2c_write(bin2bcd(mth));    // REG 5
  i2c_write(bin2bcd(year));   // REG 6
  i2c_write(DS1307_OUT_8_KHZ);// REG 7 - Disable squarewave output pin
  i2c_stop();
}

void ds1307_get_date(BYTE &day, BYTE &mth, BYTE &year, BYTE &dow)
{
  i2c_start();
  i2c_write(0xD0);
  i2c_write(0x03);                  // Start at REG 3 - Day of week
  i2c_start();
  i2c_write(0xD1);
  dow  = bcd2bin(i2c_read() & 0x7f);   // REG 3
  day  = bcd2bin(i2c_read() & 0x3f);   // REG 4
  mth  = bcd2bin(i2c_read() & 0x1f);   // REG 5
  year = bcd2bin(i2c_read(0));        // REG 6
  i2c_stop();
}

void ds1307_get_time(BYTE &hr, BYTE &min, BYTE &sec)
{
  i2c_start();
  i2c_write(0xD0);
  i2c_write(0x00);          // Start at REG 0 - Seconds
  i2c_start();
  i2c_write(0xD1);
  sec = bcd2bin(i2c_read() & 0x7f);
  min = bcd2bin(i2c_read() & 0x7f);
  hr  = bcd2bin(i2c_read(0) & 0x3f);
  i2c_stop();
}

BYTE bin2bcd(BYTE binary_value)
{
  BYTE temp;
  BYTE retval;
  temp = binary_value;
  retval = 0;
  while(1)
  {
    if(temp >= 10)
    {
      temp -= 10;
      retval += 0x10;
    }
    else     {
      retval += temp;
      break;
    }
  }
  return(retval);
}

BYTE bcd2bin(BYTE bcd_value)
{
  BYTE temp;
  temp = bcd_value;
  temp >>= 1;
```

```
  temp &= 0x78;
  return(temp + (temp >> 2) + (bcd_value & 0x0f));
}
```

2.4. Llibreria Brúixola

```
/////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////              I2C.C                          //////////////////////////////
//////////////////          BRUIXOLA DRIVER LECTURA                           /////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
////////////////// int16 HMC6352_read_heading(void) //LECTURA BRUIXOLA/////////////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////////////////////////////////////
/////////   ** AMPLIACIÓ: VE FUNCIO PER REALITZAR AJUST AUTOMATIC SENSE TINDRE    ///////
////////////////////////////          AFEGIR CAP OFSET MANUAL**                  //////////////
/////////////////////////////////////////////////////////////////////////////////////////////////


#use i2c(master, sda=PIN_C4, scl=PIN_C3,Force_HW)

#define HMC6352_I2C_WRITE_ADDRESS 0x42            // DIRECCIO PER ESCRIURE
#define HMC6352_I2C_READ_ADDRESS  0x43            // DIRECCIO PER LLEGIR
#define HMC6352_GET_DATA_COMMAND 0x41             // DIRECCIO PER QUE FACI CALCUL

int8 lsb;                                         // DEFINICIO VARIABLES
int8 msb;                                         // VARIABLE QUE OBTE MICRO 2xint8

int16 HMC6352_read_heading(void)                  // LECTURA SENSOR
{

i2c_start();                                      // INICI ESCRITURA
i2c_write(HMC6352_I2C_WRITE_ADDRESS);             // ORDRE DE CALCULAR ANGLE
i2c_write(HMC6352_GET_DATA_COMMAND);              // ORDRE DE CALCULAR LECTURA
i2c_stop();                                       // FI ESCRITURA

delay_ms(7);                                      // S'HA D'ESPERAR 6 ms PER SEGURETAT
                                                  // ESPEREM 1 ms + PER SEGURETAT
i2c_start();                                      // INICI ESCRITURA
i2c_write(HMC6352_I2C_READ_ADDRESS);              // DIRECCIO LESCTURA
msb = i2c_read();                                 // LLEGIR LECTURA
lsb = i2c_read(0);
i2c_stop();
return (make16(msb,lsb));                         // RETURN la orientacio de la placa
}
```

2.5. Llibreria RS-485

```
///////////////////////////////////////////////////////////////////////////////////////////
///////////////////////                    RS485.c                    //////////////////////////
///////////////////////////////////////////////////////////////////////////////////////////
///////////////////////   This file contains drivers for RS-485 communication   ////////////////
///////////////////////////////////////////////////////////////////////////////////////////
/////////////// int1 rs485_get_message(int* data_ptr, int wait)            ////////////////////
///////////////    * Get a message from the RS485 bus                      ////////////////////
///////////////    * Address can be 1 - 33.                                ////////////////////
///////////////    * A 1 will be returned if a parity check error occurred. ///////////////////
///////////////////////////////////////////////////////////////////////////////////////////
/////////////// int1 rs485_send_message(int to, int len, int* data)        ////////////////////
///////////////    * Send a message over the RS485 bus                     ////////////////////
///////////////    * Inputs:                                               ////////////////////
////////////////        to    - Destination address                        ////////////////////
////////////////        len   - Message length                             ////////////////////
//////////          *data - Pointer to message                             ////////////////////
//////////////     * Returns TRUE if successful, FALSE if failed           ////////////////////
///////////////////////////////////////////////////////////////////////////////////////////
/////////////// void rs485_wait_for_bus(int1 clrwdt)                       ////////////////////
///////////////    * Wait for the RS485 bus to open. Normally used before  ////////////////////
//////////////////      sending a message to avoid collisions.             ////////////////////
///////////////    * Pass in TRUE to restart the watch dog timer to prevent ///////////////////
////////////////      the device from resetting.                           ////////////////////
/////////////////     * This is not needed if sending an immediate response ///////////////////
///////////////////////////////////////////////////////////////////////////////////////////

#ifndef RS485_DRIVER
#define RS485_DRIVER

#ifndef RS485_ID
#define RS485_ID  0x10                                  // The device's RS485 address or ID
#endif

#ifndef RS485_USE_EXT_INT
#define RS485_USE_EXT_INT FALSE                         // Select between external interrupt
#endif                                                  // or asynchronous serial interrupt


#if(RS485_USE_EXT_INT == FALSE)
  #ifndef RS485_RX_PIN
  #define RS485_RX_PIN       PIN_C7                     // Data receive pin
  #endif

  #ifndef RS485_TX_PIN
  #define RS485_TX_PIN       PIN_C6                     // Data transmit pin
  #endif

  #ifndef RS485_ENABLE_PIN
  #define RS485_ENABLE_PIN   PIN_B4                     // Controls DE pin.  RX low, TX high.
  #endif

  #ifndef RS485_RX_ENABLE
  #define RS485_RX_ENABLE    PIN_B5                     // Controls RE pin.  Should keep low.
  #endif

  #use rs232(baud=9600, xmit= RS485_TX_PIN, rcv=RS485_RX_PIN, enable =
RS485_ENABLE_PIN, bits = 9, long_data, errors, stream=RS485)
```

```
  #use rs232(baud=9600, xmit=RS485_TX_PIN, rcv=RS485_RX_PIN,
enable=RS485_ENABLE_PIN, bits=9, long_data, force_sw, multi_master, errors,
stream=RS485_CD)

  #if getenv("AUART")
    #define RCV_OFF() {setup_uart(FALSE);}
  #else
    #define RCV_OFF() {setup_uart(FALSE);}
  #endif
#else
  #ifndef RS485_RX_PIN
  #define RS485_RX_PIN      PIN_B0                                  // Data receive pin
  #endif

  #ifndef RS485_TX_PIN
  #define RS485_TX_PIN      PIN_B3                                  // Data transmit pin
  #endif

  #ifndef RS485_ENABLE_PIN
  #define RS485_ENABLE_PIN   PIN_B4                          // Controls DE pin.  RX low, TX high.
  #endif

  #ifndef RS485_RX_ENABLE
  #define RS485_RX_ENABLE    PIN_B5                          // Controls RE pin.  Should keep low.
  #endif

  #use rs232(baud=9600, xmit=RS485_TX_PIN, rcv=RS485_RX_PIN,
enable=RS485_ENABLE_PIN, bits=9, long_data, errors, stream=RS485)
  #use rs232(baud=9600, xmit=RS485_TX_PIN, rcv=RS485_RX_PIN,
enable=RS485_ENABLE_PIN, bits=9, long_data, multi_master, errors, stream=RS485_CD)

  #define RCV_OFF() {disable_interrupts(INT_EXT);}
#endif

#define RS485_wait_time 20                                        // Wait time in milliseconds

#bit    rs485_collision = rs232_errors.6

#ifndef RS485_RX_BUFFER_SIZE
#define RS485_RX_BUFFER_SIZE  40
#endif

int rs485_state, rs485_ni, rs485_no;
int rs485_buffer[RS485_RX_BUFFER_SIZE];

// Purpose:    Enable data reception
// Inputs:     None
// Outputs:    None
void RCV_ON(void) {
  #if (RS485_USE_EXT_INT==FALSE)
    while(kbhit(RS485)) {getc();} // Clear RX buffer. Clear RDA interrupt flag. Clear overrun error
flag.
    #if getenv("AUART")
      setup_uart(UART_ADDRESS);
      setup_uart(TRUE);
    #else
      setup_uart(TRUE);
    #endif
  #else
    clear_interrupt(INT_EXT);
    enable_interrupts(INT_EXT);
  #endif
}

// Purpose:    Initialize RS485 communication. Call this before
```

```
//            using any other RS485 functions.
// Inputs:     None
// Outputs:    None
void rs485_init() {
  RCV_ON();
  rs485_state=0;
  rs485_ni=0;
  rs485_no=0;
  #if RS485_USE_EXT_INT==FALSE
  enable_interrupts(INT_RDA);
  #else
  ext_int_edge(H_TO_L);
  enable_interrupts(INT_EXT);
  #endif
  enable_interrupts(GLOBAL);
  output_low(RS485_RX_ENABLE);
}

// The index for the temporary receive buffer
int8 temp_ni;

// Purpose:    Add a byte of data to the temporary receive buffer
// Inputs:     The byte of data
// Outputs:    None
void rs485_add_to_temp(int8 b) {
  // Store the byte
  rs485_buffer[temp_ni] = b;

  // Make the index cyclic
  if(++temp_ni >= RS485_RX_BUFFER_SIZE)
  {
    temp_ni = 0;
  }
}

// Purpose:    Interrupt service routine for handling incoming RS485 data
#if (RS485_USE_EXT_INT==FALSE)
#int_rda
#else
#int_ext
#endif
void incomming_rs485() {
  int16 b;
  static int8  cs,state=0,len;
  static int16 to,source;

  b=fgetc(RS485);
  cs^=(int8)b;

  switch(state) {
    case 0:  // Get from address
      temp_ni=rs485_ni;
      source=b;
      cs=b;
      rs485_add_to_temp(source);
      break;

    case 1:  // Get to address
      to=b;
      #if (getenv("AUART")&&(RS485_USE_EXT_INT==FALSE))
        setup_uart(UART_DATA);
      #endif
      break;

    case 2:  // Get len
```

```
        len=b;
        rs485_add_to_temp(len);
        break;

      case 255:   // Get checksum
        if  ((!cs)&&(bit_test(to,8))&&(bit_test(source,8))&&((int8)to==RS485_ID))  {   // If  cs==0,
then checksum is good
          rs485_ni=temp_ni;
        }

        #if (getenv("AUART")&&(RS485_USE_EXT_INT==FALSE))
          setup_uart(UART_ADDRESS);
        #endif

        state=0;
        return;

      default: // Get data
        rs485_add_to_temp(b);
        --len;
        break;
    }
    if ((state>=3) && (!len)) {
      state=255;
    }
    else {
      ++state;
    }
}

// Purpose:    Send a message over the RS485 bus
// Inputs:     1) The destination address
//             2) The number of bytes of data to send
//             3) A pointer to the data to send
// Outputs:    TRUE if successful
//             FALSE if failed
// Note:       Format:  source | destination | data-length | data | checksum
int1 rs485_send_message(int8 to, int8 len, int8* data) {
    int8 try, i, cs;
    int1 ret = FALSE;

    RCV_OFF();
    #if RS485_USE_EXT_INT
      disable_interrupts(GLOBAL);
    #endif

    for(try=0; try<5; ++try) {
      rs485_collision = 0;
      fputc((int16)0x100|rs485_id, RS485_CD);
      fputc((int16)0x100|to, RS485_CD);
      fputc(len, RS485_CD);

      for(i=0, cs=rs485_id^to^len; i<len; ++i) {
        cs ^= *data;
        fputc(*data, RS485_CD);
        ++data;
      }

      fputc(cs, RS485_CD);
      if(!rs485_collision) {
        ret = TRUE;
        break;
      }
      delay_ms(RS485_ID);
    }
```

```
  RCV_ON();
  #if RS485_USE_EXT_INT
    enable_interrupts(GLOBAL);
  #endif

  return(ret);
}


// Purpose:    Wait for wait time for the RS485 bus to become idle
// Inputs:     TRUE - restart the watch dog timer to prevent reset
//             FALSE - watch dog timer not restarted
// Outputs:    None
void rs485_wait_for_bus(int1 clrwdt)
{
  int16 i;

  RCV_OFF();
  for(i=0; i <= (rs485_wait_time*20); ++i)
  {
    if(!input(RS485_RX_PIN))
      i = 0;
    else
      delay_us(50);

    if(clrwdt)
      restart_wdt();
  }
}

// Purpose:    Get a message from the RS485 bus and store it in a buffer
// Inputs:     1) A pointer to a buffer to store a message
//             2) TRUE  - wait for a message
//                FALSE - only check if a message is available
// Outputs:    TRUE if a message was received
//             FALSE if wait is FALSE and no message is available
// Note:       Data will be filled in at the pointer as follows:
//             FROM_ID  DATALENGTH  DATA...
int1 rs485_get_message(int* data_ptr, int1 wait)
{
  while(wait && (rs485_ni == rs485_no)) {}

  if(rs485_ni == rs485_no)
    return FALSE;
  else {
    int n;
    n = rs485_buffer[(rs485_no+1)%sizeof(rs485_buffer)] + 2;

    for(; n>0; --n)
    {
      *data_ptr = rs485_buffer[rs485_no];
      if(++rs485_no >= sizeof(rs485_buffer))
      {
        rs485_no = 0;
      }
      ++data_ptr;
    }
    return TRUE;
  }
}

#endif
```

CAPÍTOL 3: PROGRAMACIÓ VISUAL C#

3.1. Programa inicialització

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace WindowsFormsApplication1
{
    static class Program
    {
        /// <summary>
        /// Punto de entrada principal para la aplicación.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

3.2. Programació objectes i events

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;



namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {

        public UInt16 LDR = 0;
        public Int16 TMP = 0, TMP_PLACA = 0;
        public byte ldr_high = 0, ldr_low = 0, tmp_low = 0, tmp_high = 0,
vent = 0, tmp_placa_low = 0, tmp_placa_high =
0,dia=0,mes=0,any=0,hora=0,minut=0,segons=0;
        public byte[] buffer = new byte [8];
        public byte[] envio = new byte[7];
        public byte dia_semana = 0;
        public int [] vent_graf = new int [1175];
        public int[] temp_graf = new int[1175];
        public int[] temp_placa_graf = new int[1175];
        public int[] ldr_graf = new int[1175];
        public int i = 0;
        public string hour, minuts, seconds,day, month, year;


        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            serialPort1.Open();
            button2.Enabled = true;
            button1.Enabled = false;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            serialPort1.Close();
            button1.Enabled = true;
            button2.Enabled = false;
        }

        private void serialPort1_DataReceived(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
        {
```

```
            int picBoxWidth = pictureBox1.Size.Width;
            int picBoxHeight = pictureBox1.Size.Height;
            int halfWidth = pictureBox1.Size.Width / 2;
            int halfHeight = pictureBox1.Size.Height / 2;
            Graphics objGraphic = this.pictureBox1.CreateGraphics();
            Graphics objGraphic2 = this.pictureBox2.CreateGraphics();
            Graphics objGraphic3 = this.pictureBox3.CreateGraphics();
            Graphics objGraphic4 = this.pictureBox4.CreateGraphics();
            int bytes = serialPort1.BytesToRead;
            serialPort1.Read(buffer, 0, bytes);
            Pen pen = new Pen(Color.Black);
            Pen pen1 = new Pen(Color.Red);
            Pen pen2 = new Pen(Color.Orange);
            Pen pen3 = new Pen(Color.Brown);
            Pen pen4 = new Pen(Color.Blue);

            if (buffer[0] == 'S')
            {
                ldr_low = buffer[1];
                ldr_high = buffer[2];
                tmp_low = buffer[3];
                tmp_high = buffer[4];
                vent = buffer[5];
                tmp_placa_low = buffer[6];
                tmp_placa_high = buffer[7];
            }
            else
                if (buffer[0] == 'B')
                {
                    dia = buffer[1];
                    mes = buffer[2];
                    any = buffer[3];
                    hora = buffer[4];
                    minut = buffer[5];
                    segons = buffer[6];
                    dia_semana = buffer[7];
                }

            /*day = Convert.ToString(dia);
            month = Convert.ToString(mes);
            year = Convert.ToString(any);
            hour = Convert.ToString(hora);
            minuts = Convert.ToString(minut);
            seconds = Convert.ToString(segons);
            */

            LDR = Convert.ToUInt16((ldr_low) | (ldr_high << 8));
            TMP = Convert.ToInt16((tmp_low) | (tmp_high << 8));
            TMP_PLACA = Convert.ToInt16((tmp_placa_low) | (tmp_placa_high
<< 8));
            textBox1.Text = Convert.ToString(LDR);
            textBox2.Text = Convert.ToString(TMP);
            textBox3.Text = Convert.ToString(vent);
            textBox4.Text = Convert.ToString(TMP_PLACA);
            label5.Text = Convert.ToString(dia);
            label6.Text = Convert.ToString(mes);
            label7.Text = Convert.ToString(any);
            label8.Text = Convert.ToString(hora);
            label9.Text = Convert.ToString(minut);
            label10.Text = Convert.ToString(segons);

            if (i < 20)
            {
```

```
                do
                {
                    temp_graf[i] = 420 - (TMP * 400 / 140)-40*400/140;
                    vent_graf[i] = 420 - (vent * 400 / 60);
                    ldr_graf[i] = 420 - (LDR * 400 / 1000);
                    temp_placa_graf[i] = 420 - (TMP_PLACA * 400 / 140)-
40*400/140;
                    i = i + 1;
                }
                while (i < 20);
            }
            else
            {

                i = i + 1;
                temp_graf[i] = 420 - (TMP * 400 / 140)-40*400/140;
                vent_graf[i] = 420 - (vent * 400 / 60);
                ldr_graf[i] = 420 - (LDR * 400 / 1000);
                temp_placa_graf[i] = 420 - (TMP_PLACA * 400 / 140)-
40*400/140;

                if (radioButton3.Checked == true)
                {
                    objGraphic3.DrawLine(pen2, (i - 1), vent_graf[i - 1],
i, vent_graf[i]);
                    objGraphic3.DrawLine(pen, 20, 420, 1150, 420);
                    objGraphic3.DrawLine(pen, 20, 420, 20, 20);
                    System.Drawing.Drawing2D.GraphicsState graph =
objGraphic3.Save();
                    objGraphic3.Restore(graph);
                }
                else
                {
                    if (radioButton2.Checked == true)
                    {
                        objGraphic2.DrawLine(pen1, (i - 1), temp_graf[i -
1], i, temp_graf[i]);
                        objGraphic2.DrawLine(pen, 20, 286, 1150, 286);
                        objGraphic2.DrawLine(pen, 20, 420, 20, 20);
                        System.Drawing.Drawing2D.GraphicsState graph2 =
objGraphic2.Save();
                        objGraphic2.Restore(graph2);
                    }
                    else
                    {
                        if (radioButton1.Checked == true)
                        {
                            objGraphic.DrawLine(pen3, (i - 1), ldr_graf[i -
1], i, ldr_graf[i]);
                            objGraphic.DrawLine(pen, 20, 420, 1150, 420);
                            objGraphic.DrawLine(pen, 20, 420, 20, 20);
                            System.Drawing.Drawing2D.GraphicsState graph =
objGraphic.Save();
                            objGraphic.Restore(graph);
                        }
                        else
                        {
                            objGraphic4.DrawLine(pen4, i - 1,
temp_placa_graf[i - 1], i, temp_placa_graf[i]);
                            objGraphic4.DrawLine(pen, 20, 286, 1150, 286);
                            objGraphic4.DrawLine(pen, 20, 420, 20, 20);
                            System.Drawing.Drawing2D.GraphicsState graph4 =
objGraphic4.Save();
```

```
                            objGraphic4.Restore(graph4);
                        }
                    }
                }
            }


            if (i >= 1000)
                {
                    i = 0;
                    objGraphic.Clear(Color.White);
                    objGraphic2.Clear(Color.White);
                    objGraphic3.Clear(Color.White);
                    objGraphic4.Clear(Color.White);
                }
            //Array.Clear(buffer, 0, buffer.Length);


        }

        private void button3_Click(object sender, EventArgs e)
        {

            envio[0] = Convert.ToByte('A');
            envio [1] = Convert.ToByte(dateTimePicker1.Value.Day);
            envio [2] = Convert.ToByte(dateTimePicker1.Value.Month);
            envio [3] =
Convert.ToByte(Convert.ToUInt16(dateTimePicker1.Value.Year)-2000);
            envio [4] = Convert.ToByte(dateTimePicker2.Value.Hour);
            envio [5] = Convert.ToByte(dateTimePicker2.Value.Minute);
            envio [6] = Convert.ToByte(dateTimePicker2.Value.Second);

            serialPort1.Write(envio, 0, envio.Length);


        }

        private void button4_Click(object sender, EventArgs e)
        {
            Close();
        }

        private void radioButton1_CheckedChanged(object sender, EventArgs
e)
        {
            if (radioButton1.Checked == true)
                {
                    pictureBox1.Visible = true;
                    pictureBox2.Visible = false;
                    pictureBox3.Visible = false;
                    pictureBox4.Visible = false;
                    label12.Visible = true;
                    label11.Visible = false;
                    label13.Visible = false;
                    label14.Visible = false;
                    label18.Visible = true;
                    label17.Visible = true;
                    label21.Visible = true;
                    label22.Visible = true;
                    label19.Visible = false;
                    label15.Visible = false;
                    label20.Visible = false;
                    label16.Visible = false;
```

```
                label23.Visible = false;
                label22.Visible = false;

            }
        }

        private void radioButton2_CheckedChanged(object sender, EventArgs
e)
        {
            if (radioButton2.Checked == true)
            {
                pictureBox1.Visible = false;
                pictureBox2.Visible = true;
                pictureBox3.Visible = false;
                pictureBox4.Visible = false;
                label12.Visible = false;
                label11.Visible = true;
                label13.Visible = false;
                label14.Visible = false;

                label18.Visible = false;
                label17.Visible = false;
                label21.Visible = false;
                label22.Visible = false;
                label19.Visible = false;
                label15.Visible = true;
                label20.Visible = true;
                label16.Visible = true;
                label23.Visible = true;
                label22.Visible = false;

            }
        }

        private void radioButton3_CheckedChanged(object sender, EventArgs
e)
        {
            if (radioButton3.Checked == true)
            {
                pictureBox1.Visible = false;
                pictureBox2.Visible = false;
                pictureBox3.Visible = true;
                pictureBox4.Visible = false;
                label12.Visible = false;
                label11.Visible = false;
                label13.Visible = true;
                label14.Visible = false;

                label18.Visible = false;
                label17.Visible = true;
                label21.Visible = false;
                label22.Visible = false;
                label19.Visible = true;
                label15.Visible = false;
                label20.Visible = false;
                label16.Visible = false;
                label23.Visible = false;
                label22.Visible = true;

            }
        }
```

```
        private void radioButton4_CheckedChanged(object sender, EventArgs
e)
        {
            pictureBox1.Visible = false;
            pictureBox2.Visible = false;
            pictureBox3.Visible = false;
            pictureBox4.Visible = true;
            label12.Visible = false;
            label11.Visible = false;
            label13.Visible = false;
            label14.Visible = true;
            label18.Visible = false;
            label17.Visible = false;
            label21.Visible = false;
            label22.Visible = false;
            label19.Visible = false;
            label15.Visible = true;
            label20.Visible = true;
            label16.Visible = true;
            label23.Visible = true;
            label22.Visible = false;
        }

        private void button5_Click(object sender, EventArgs e)
        {
            envio[0] = Convert.ToByte('B');
            envio[1] = Convert.ToByte(textBox5.Text);
            envio[2] = Convert.ToByte(textBox6.Text);
            envio[3] = Convert.ToByte(textBox7.Text);
            envio[4] = Convert.ToByte(textBox10.Text);
            envio[5] = Convert.ToByte(textBox8.Text);
            envio[6] = Convert.ToByte(textBox9.Text);

            serialPort1.Write(envio, 0, envio.Length);
        }



    }
}
```