Universitatea politehnica Timisoara
Facultatea de Electronica , Telecomunicatii si tehnologia Informatiei
Departamentul de Electronica Aplicata

PROIECT CU PROCESOARE

MODUL DE MONITORIZARE STARE INCARCARE BATERIE AUTO CU LCD.

Coordonator:

Drd.ing. Sorin Popescu

Studenti:

Ostan Lidia-Simona

Igna Gheorghe

Jucut Marius-Adrian

Echipa 6 /EA-1.4

2017

Universitatea politehnica Timisoara
Facultatea de Electronica , Telecomunicatii si tehnologia Informatiei
Departamentul de Electronica Aplicata

Cuprins

1.Tema de proiectare

1.1Descriere

Tema proiectului consta in realizarea unui modul de monitorizare stare incarcare baterie auto cu LCD.

Afisarea rezultatelor se realizeaza prin intermediul unui afisaj LCD

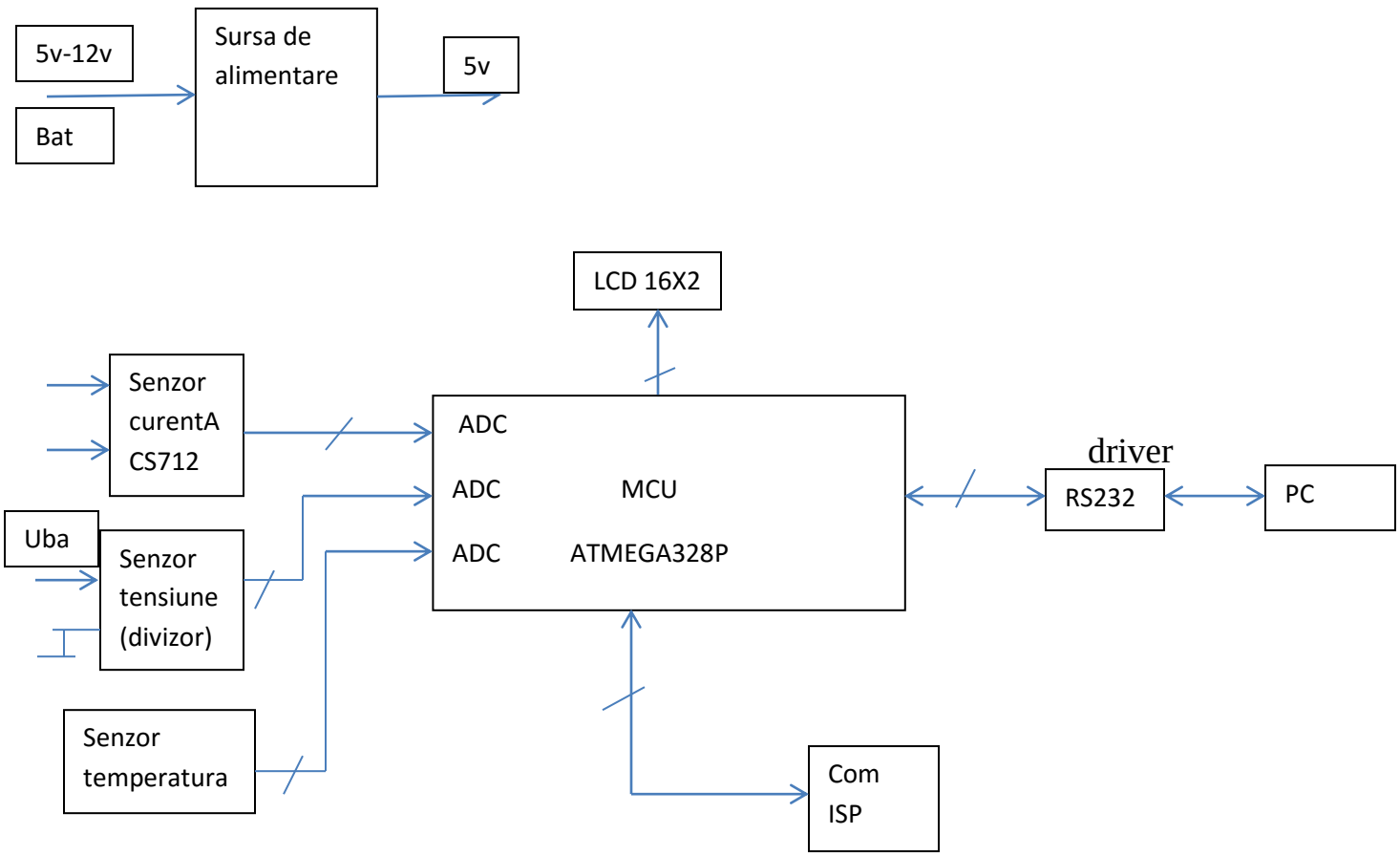
Pentru realizarea acestuia am ales un microcontroller Atmega328p care respecta cerinta impusa de catre proiect si anume aceea ca memoria de tip Flah sa poata fi programata cu ajutorul tehnologiei ISP.

Atmega328p are urmatoarele caracteristici:

Producator	ATMEL
Tip circuit integrat	microcontroller AVR
Organizare memorie flah	32kx8bit
Capacitate memorie EEPROM	1024B
Capacitate memorie SRAM	2048B
Carcasa	DIP28
Secvente sincronizare	20MHz
Numar intrari/iesiri	23
Numar canale PWM	6
Numar timere 8 biti	2
Numar Timere 16biti	1
Montare	THT
Tensiune de lucru	1.8…5.5V

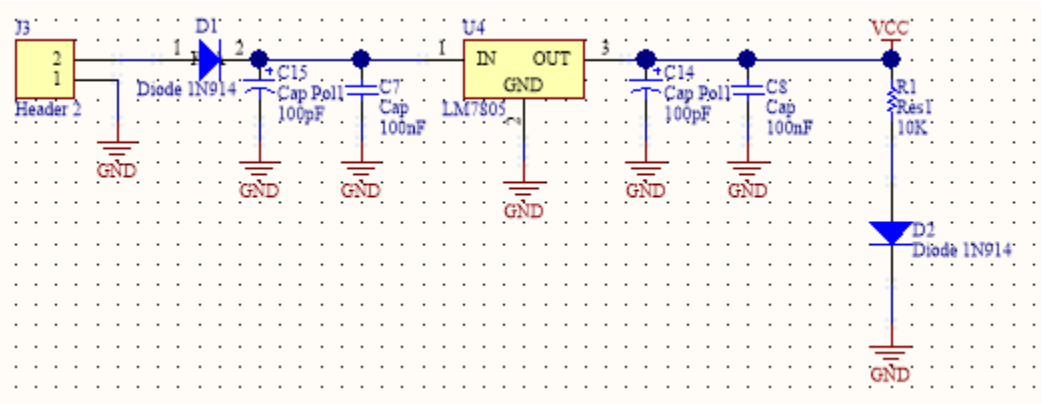
Memoria flah poate fi reprogramata folosind o interfata seriala SPI printr-un program de memorie conventional nevolatil.

2.1 Structura



2.2 Decriere blocuri functionale

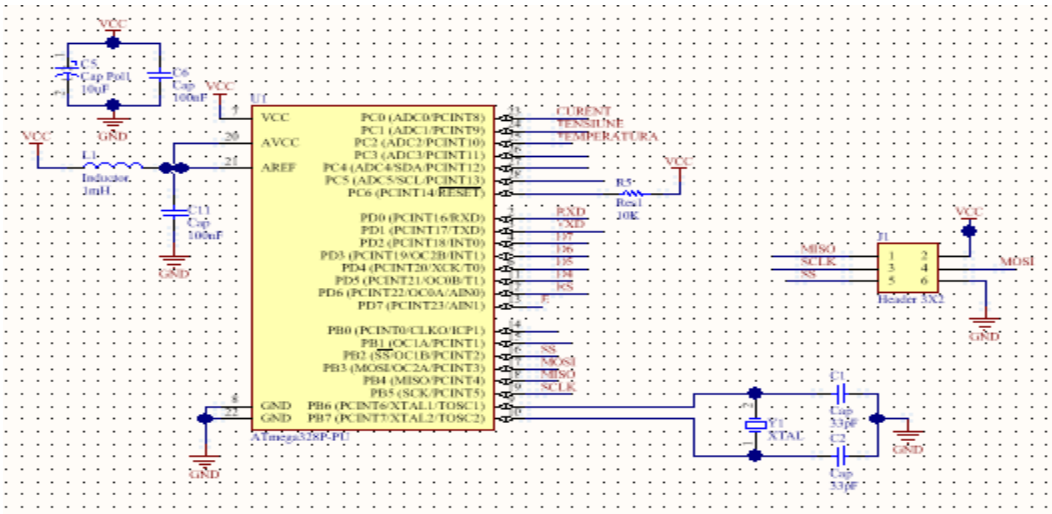
Blocul de alimentare asigura alimentarea intregului circuit cu o tensiune de 5V constanta ,transformata de la valoarea de 12V la valoarea de 5V.Pentru acest lucru se foloseste un regulator LM7805.



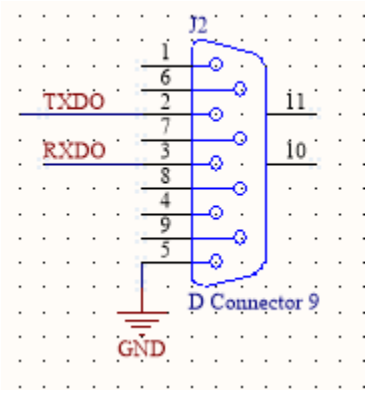
Blocul microcontroller si blocul ISP reprezinta creierul proiectului si interfata prin intermediul caruia acesta poate fi programat.In memoria flash a microcontrolerului se incarca programul software necesar indeplinirii functiei circuitului nostrum si

anume citirea datelor venite de la senzorul de temperature,tensiune si curent, si transmiterea acestora catre afisajul LCD pentru a permite vizualizarea datelor ca catre operatorul uman .

Blocul ISP cuprinde un conector cu sase pini cu ajutorul caruia se realizeaza programarea seriala amicrocontrolerului.

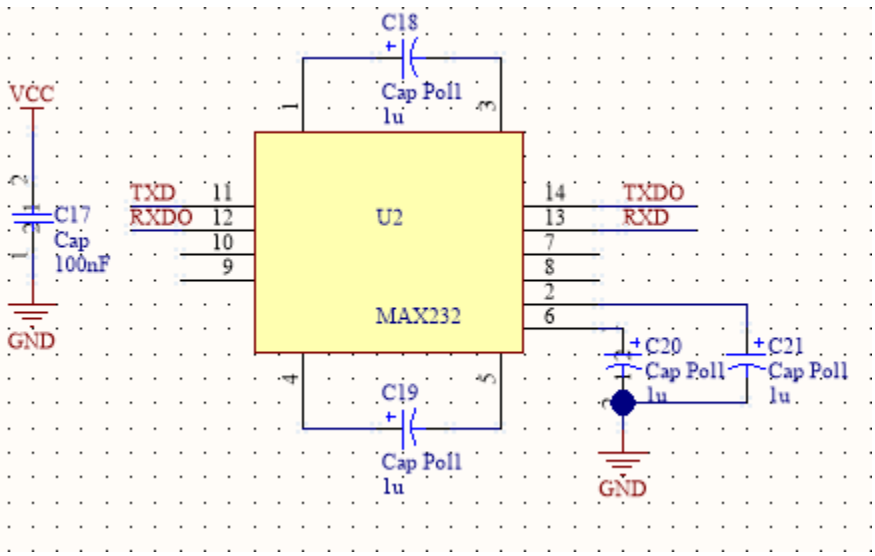


Blocul DB9 realizeaza conexiunea cu PC.Aceasta este relizata prin intermediul conectorului DB9

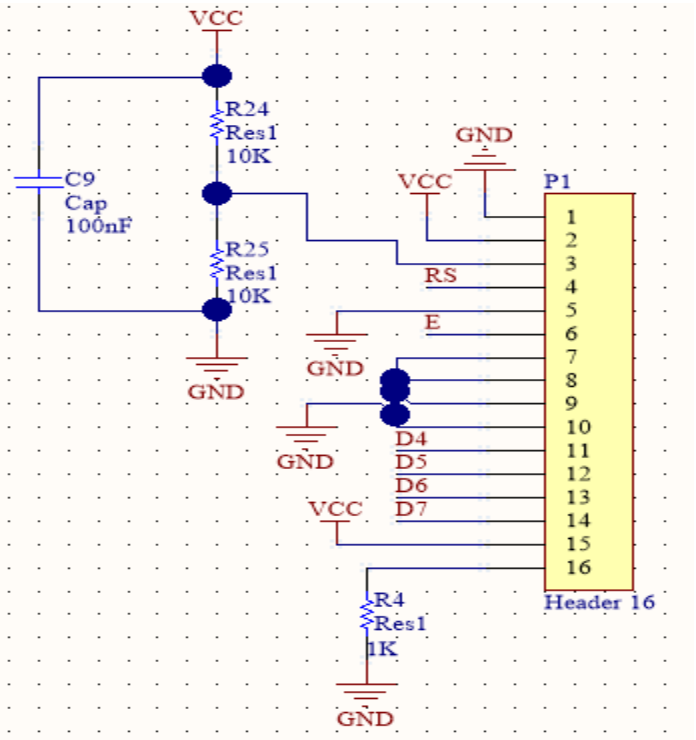


Blocul driver RS232 realizeaza conexiunea intre microcontroller si PC.In spatele acestui bloc se afla un circuit integrat denumit MAX232.Acesta converteste semnalele de la un port serial RS232 in semnale TTL utilizabile in circuitele logice.Circuitul MAX232 este conceput pentru alimentare la 5V si contine un dublor si un invertor de tensiune ce foloseste condensatoare commutate pentru a

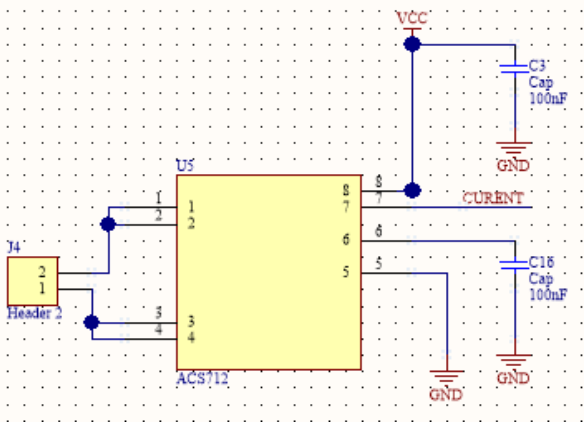
obtine o tensiune de +10Vsi -10V, necesare petru compatibilitatea cu semnalele standard RS232.



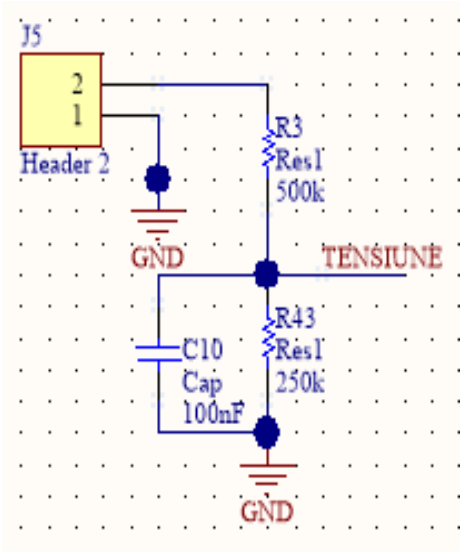
Pentru afisarea tensiuni ,curentului si temperaturii se foloseste un LCD 16x2 simplu. Acesta se conecteaza la microcontroller prin 6 semnale , 2 de control si 4 de date: RS,EN,D4,D5,D6,D7.



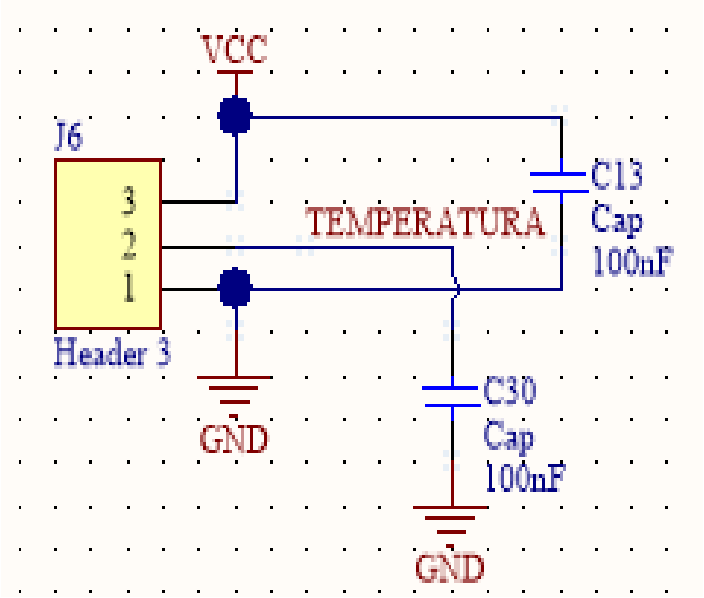
Senzorul de curent se monteaza in serie cu sarcina in circuitul extern si masoara curentul absorbit de sarcina de maxim 20A.Principiul de functionare se bazeaza pe efectul Hall, care are o iesire analogical de la care microcontroller-ul citeste tensiunea,pe care o converteste in curent.



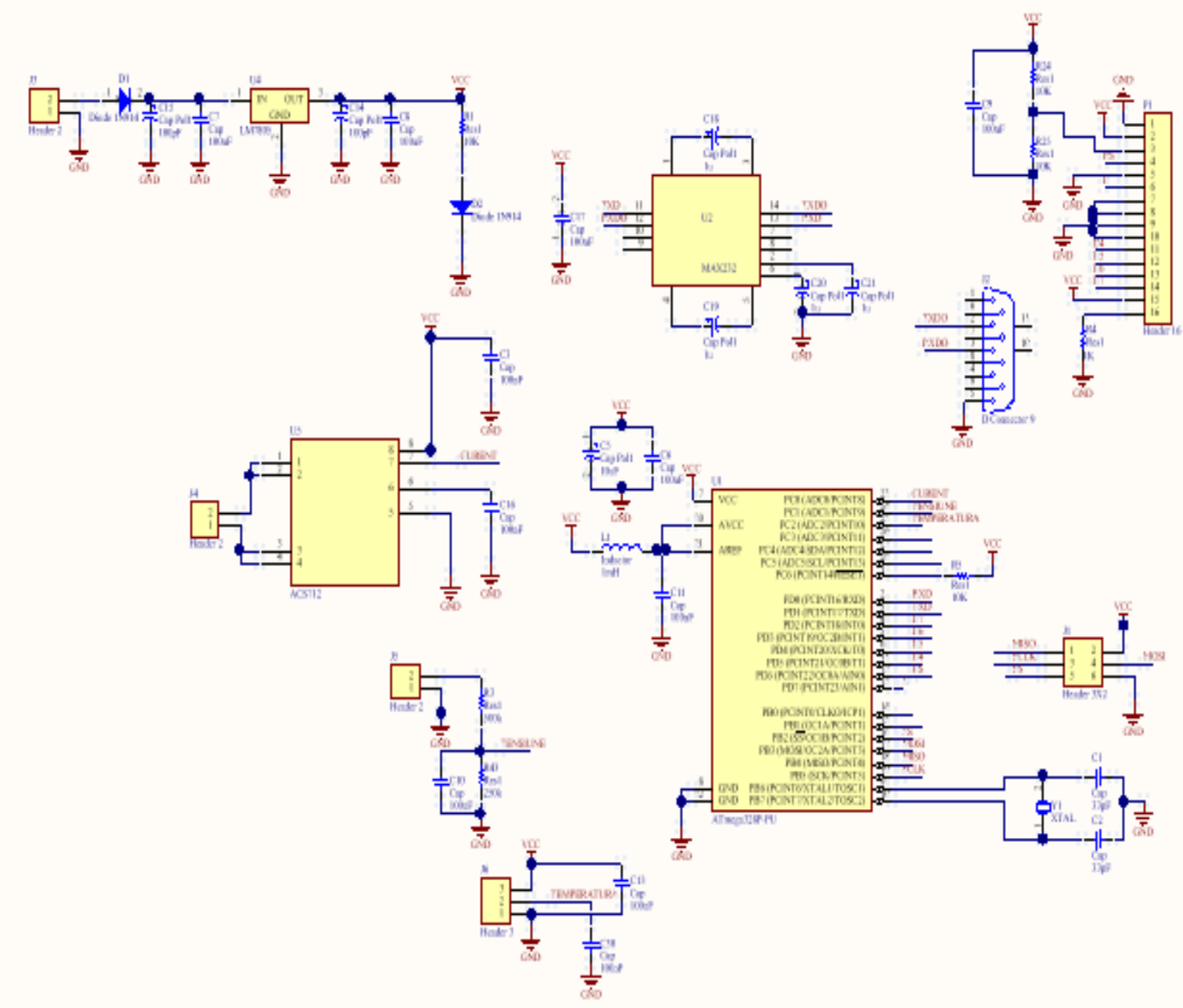
Senzorul de tensiune este format dintr-un divisor rezistiv de doua rezistente .Pentru conexiunea cu circuitul extern se foloseste o regleta.



Senzorul de temperature este un DS18B20 cu protocol ONEWIRE.Poate masura temperature de la -55 pana la 180 grade Celsius



3. Schema electronica



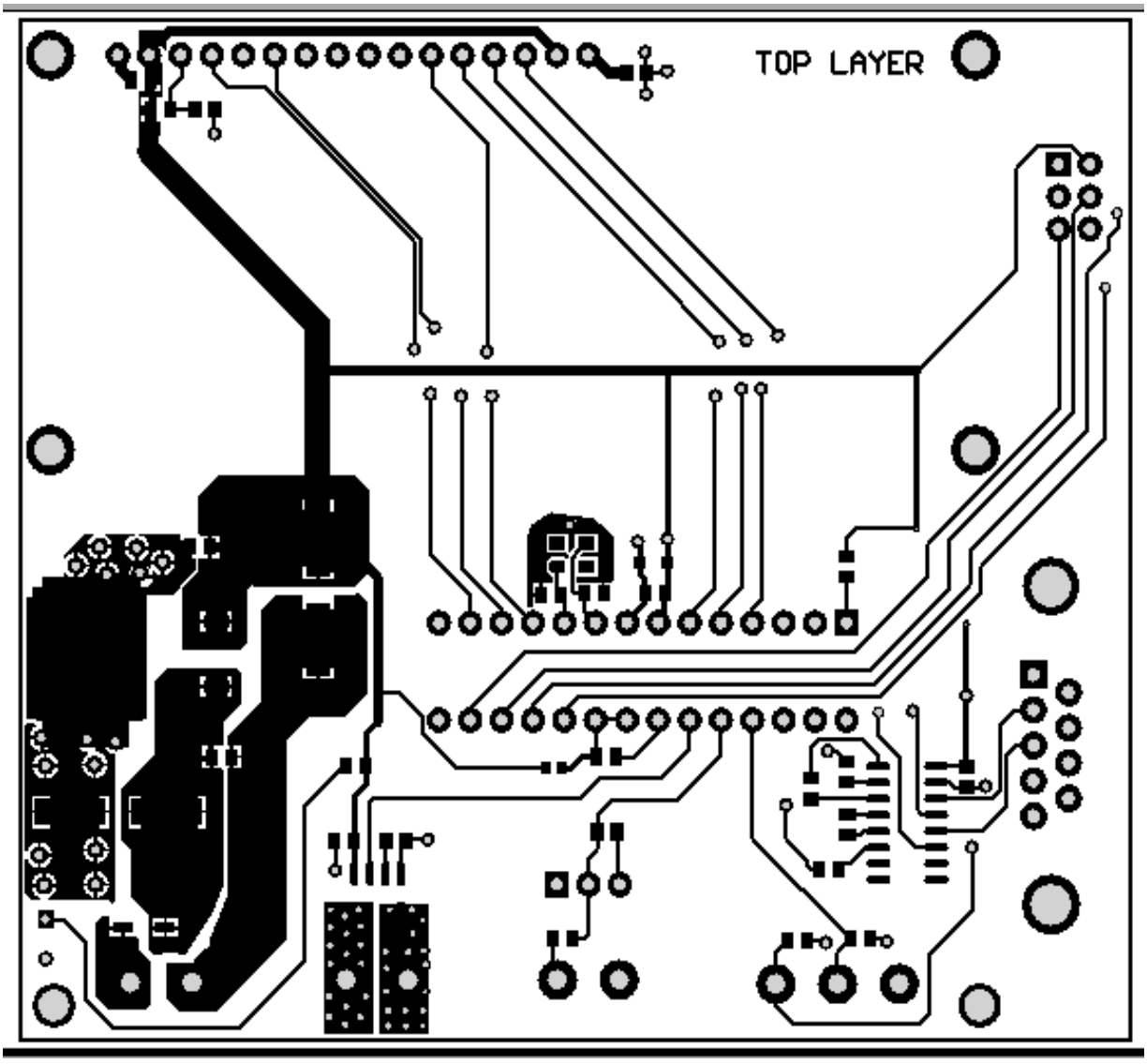
4. Lista de componente

Lista cu necesarul de componente a fost intocmita utilizand site-urile TME ,FARNELL .Pentru urmarirea listei de componente cu tot necesarul a se vedea anexa 1.

5.Circuitul imprimat

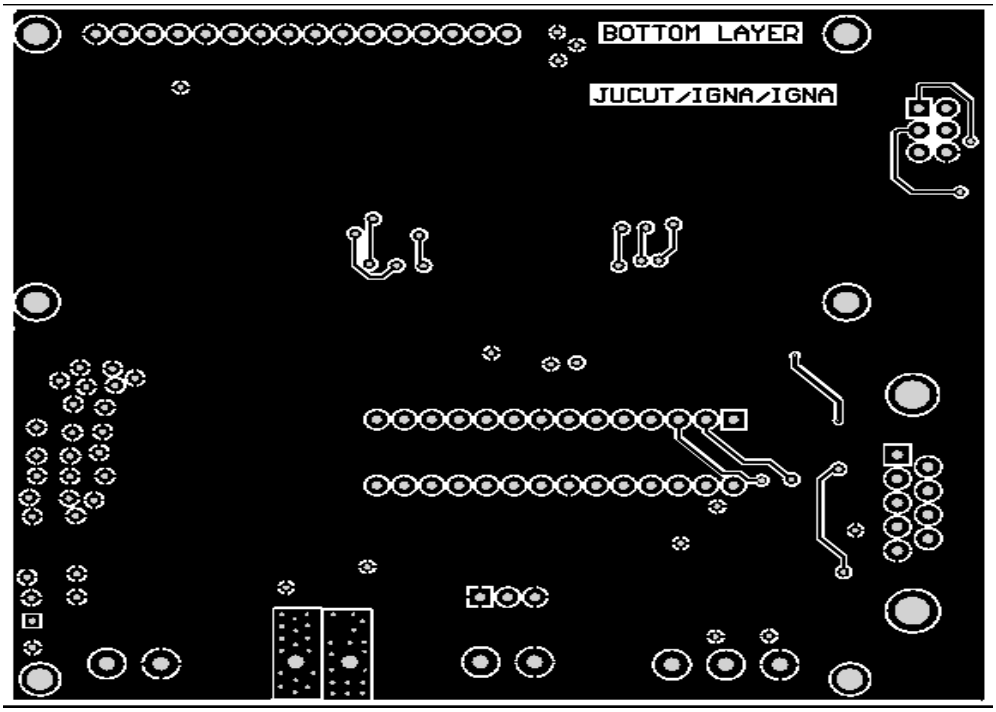
Este realizat pe doua layere Top si Botton,componentele sunt amplasate pe un singur layer acesta fiind Top.

Top layer

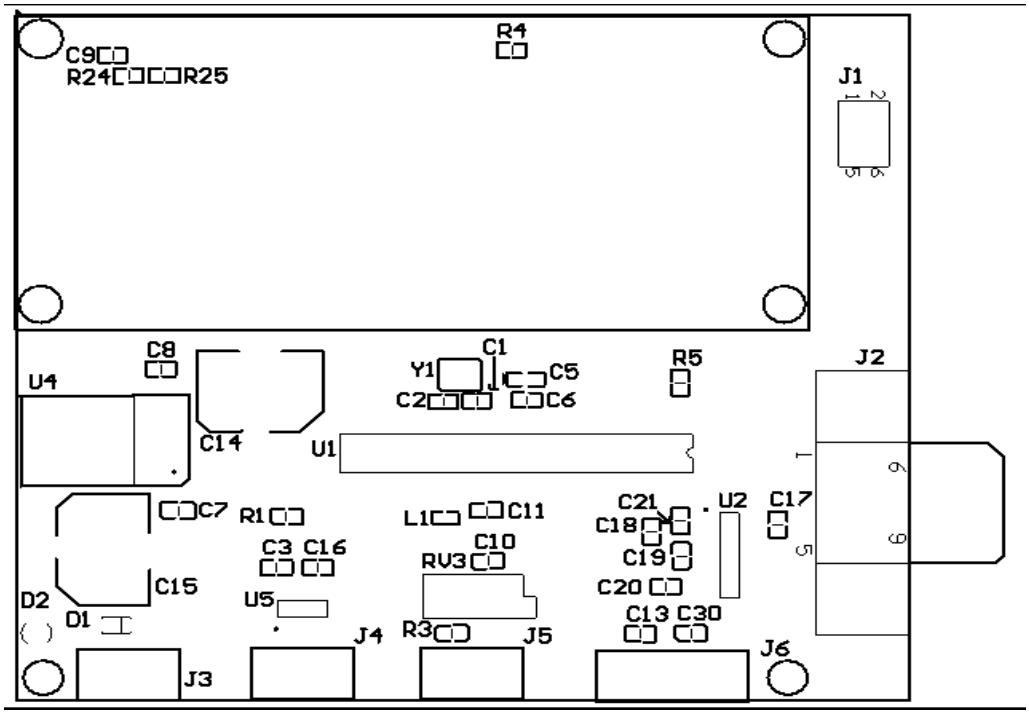


Universitatea politehnica Timisoara
Facultatea de Electronica , Telecomunicatii si tehnologia Informatiei
Departamentul de Electronica Aplicata

Bottom layer

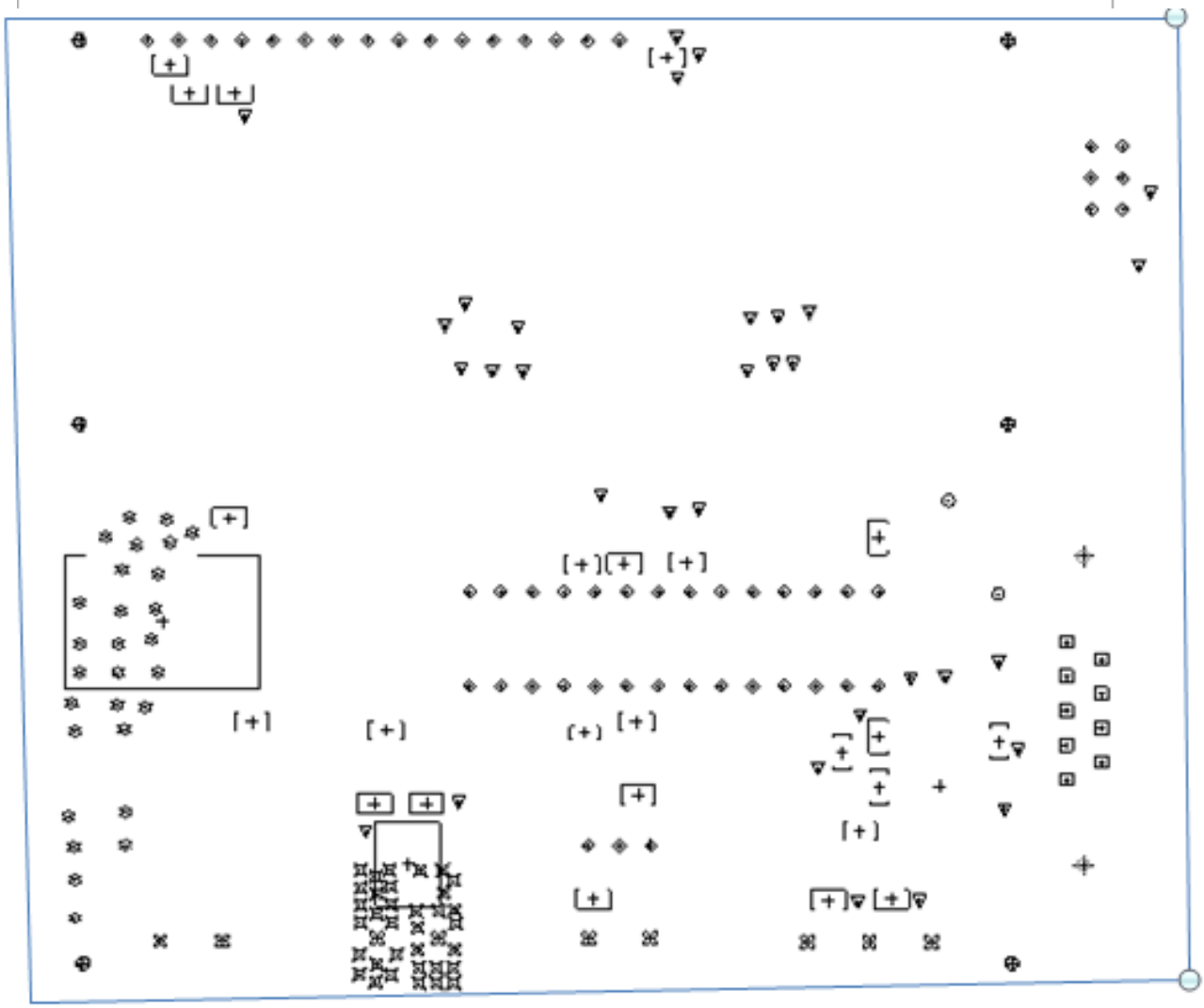


Silscreen top



Universitatea politehnica Timisoara
Facultatea de Electronica , Telecomunicatii si tehnologia Informatiei
Departamentul de Electronica Aplicata

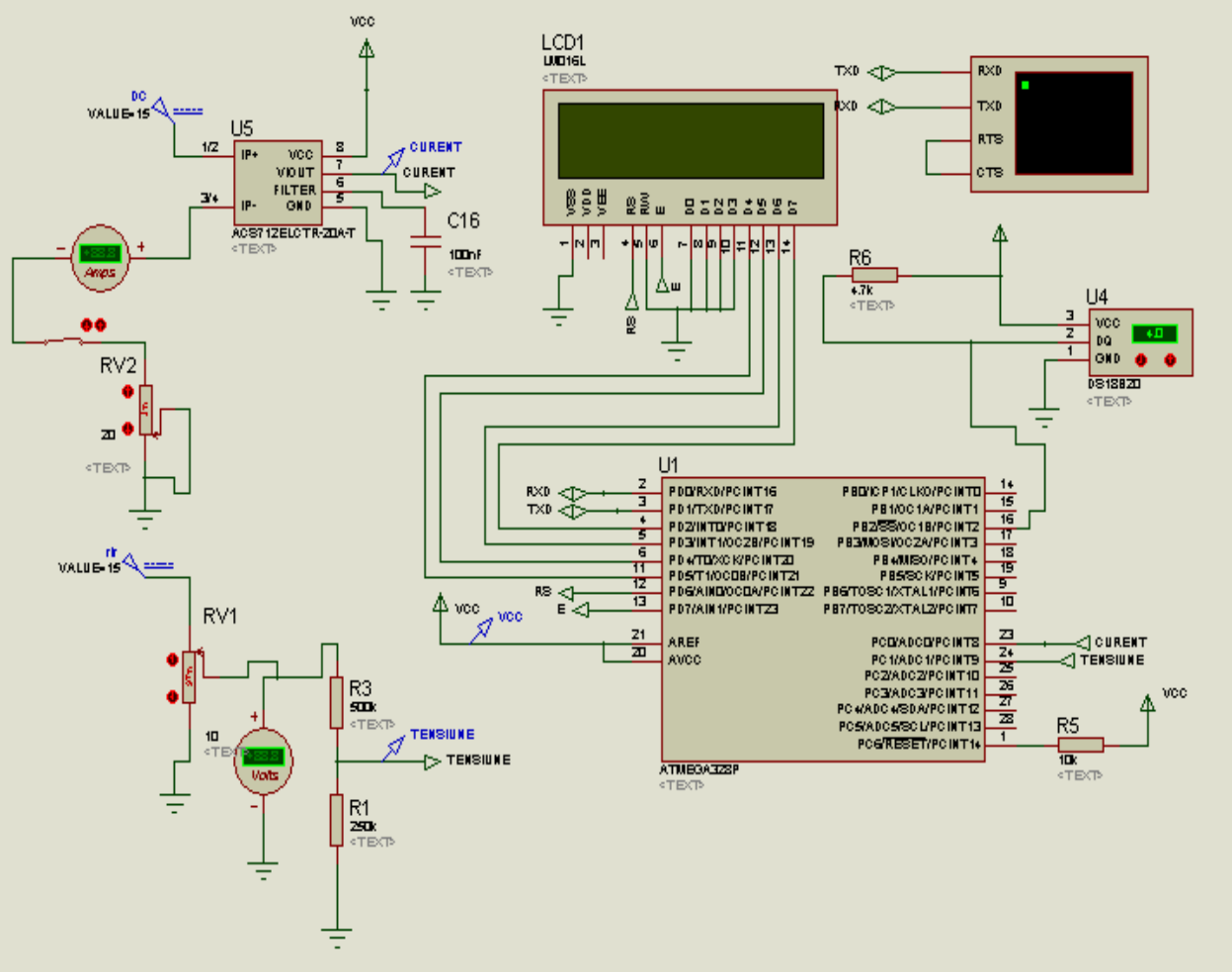
```
CAMtasticDXP Drill Report
Serial Number:0000-00-00000
Drill File: C:\Users\ghita\Desktop\cam.drl
Format: 2.3
Units: Metric (mm.)
Step & Repeat Codes Used: NO


TOOL        SIZE        COUNT
====        ======      =====

 1       0.3000       2
 2       0.5000       34
 3       0.6000       32
 4       0.7000       28
 5       1.0000       53
 6       1.1000       9
 7       1.6000       9
 8       2.5000       6
 9       3.3000       2
```



6. Simularea circuitului

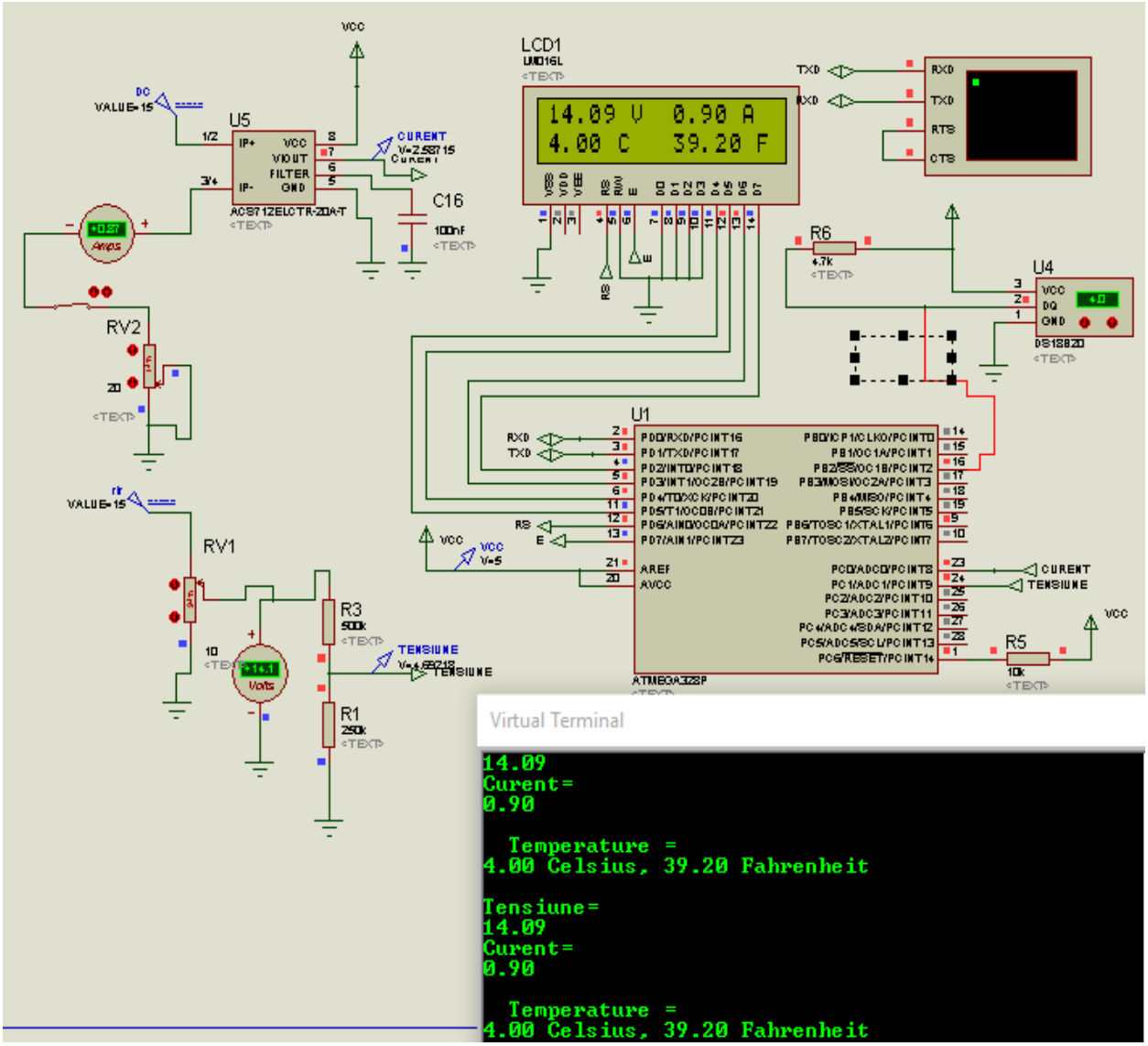
Schema de simulare realizata in Proteus:

Pentru a simula variatia de tensiune si curent s-au folosit potentiometer din biblioteca de simulare,Rv1 si Rv2 iar pentru a vedea ceea ce sriem pe serial folosim un Virtual Terminal.



Universitatea politehnica Timisoara
Facultatea de Electronica , Telecomunicatii si tehnologia Informatiei
Departamentul de Electronica Aplicata

Se ataseaza schema de simulare in momentul rularii. Se observa ca pe LCD se afiseaza ceea ce s-a dorit comform cerintei proiectului cu toti parametri doriti.Acest lucru este evidentiant si cu ajutorul multimetrelor utilizate.



7. Codul sursa

```
#include <OneWire.h>
#include <LiquidCrystal.h>
// OneWire DS18S20, DS18B20, DS1822 Temperature Example
//
// http://www.pjrc.com/teensy/td_libs_OneWire.html
//
// The DallasTemperature library can do all this work for you!
// http://milesburton.com/Dallas_Temperature_Control_Library

OneWire  ds(10);  // on pin 10 (a 4.7K resistor is necessary)
LiquidCrystal lcd(6, 7, 5, 4, 3, 2);
const int Pintensiune = A1; // senzor tensiune la A1
const int Pincurent = A0; // senzor curent la A0
int mVperAmp = 100;
int ACSoffset = 2500;
double Voltage = 0;
double Amps = 0;
int curent=0;
 float sumatens =0;
  double sumacurent = 0;
 float tensiune1=0;
  double curent1;
  float tensiune2;
void setup(void) {
  Serial.begin(9600);
  lcd.begin(16, 2);
  lcd.print("monitorizare");
```

```
  lcd.setCursor(0, 1);
  lcd.print("baterie auto ");
  delay(700);
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("CODE BY IGNA GH");
  Serial.begin(9600);
  delay(700);
  lcd.clear();
}

void loop(void) {
  byte i;
  byte present = 0;
  byte type_s;
  byte data[12];
  byte addr[8]={0X28,0X30,0XC5,0XB8,0,0,0,0X8E};
  float celsius, fahrenheit;

//  if ( !ds.search(addr)) {
//    Serial.println("No more addresses.");
//    Serial.println();
//    ds.reset_search();
//    delay(250);
//    return;
//  }

//  Serial.print("ROM =");
//  for( i = 0; i < 8; i++) {
//    Serial.write(' ');
```

```
//    Serial.print(addr[i], HEX);
//
//
//  }
//   Serial.print(" ");
//     Serial.print(addr[1], HEX);

  if (OneWire::crc8(addr, 7) != addr[7]) {
     // Serial.println("CRC is not valid!");
      return;
  }
 //Serial.println();

  // the first ROM byte indicates which chip
  switch (addr[0]) {
    case 0x10:
     // Serial.println("  Chip = DS18S20");  // or old DS1820
      type_s = 1;
      break;
    case 0x28:
    //  Serial.println("  Chip = DS18B20");
      type_s = 0;
      break;
    case 0x22:
    //  Serial.println("  Chip = DS1822");
      type_s = 0;
      break;
    default:
     // Serial.println("Device is not a DS18x20 family device.");
      return;
```

```
  }

  ds.reset();
  ds.select(addr);
  ds.write(0x44, 1);        // start conversion, with parasite power on at the
end

  delay(250);     // maybe 750ms is enough, maybe not
  // we might do a ds.depower() here, but the reset will take care of it.

  present = ds.reset();
  ds.select(addr);
  ds.write(0xBE);         // Read Scratchpad

 // Serial.print("  Data = ");
 // Serial.print(present, HEX);
 // Serial.print(" ");
  for ( i = 0; i < 9; i++) {           // we need 9 bytes
    data[i] = ds.read();
   // Serial.print(data[i], HEX);
   // Serial.print(" ");
  }
 // Serial.print(" CRC=");
  //Serial.print(OneWire::crc8(data, 8), HEX);
  //Serial.println();
asa();
  // Convert the data to actual temperature
  // because the result is a 16 bit signed integer, it should
  // be stored to an "int16_t" type, which is always 16 bits
  // even when compiled on a 32 bit processor.
```

```
int16_t raw = (data[1] << 8) | data[0];
if (type_s) {
  raw = raw << 3; // 9 bit resolution default
  if (data[7] == 0x10) {
    // "count remain" gives full 12 bit resolution
    raw = (raw & 0xFFF0) + 12 - data[6];
  }
} else {
  byte cfg = (data[4] & 0x60);
  // at lower res, the low bits are undefined, so let's zero them
  if (cfg == 0x00) raw = raw & ~7;  // 9 bit resolution, 93.75 ms
  else if (cfg == 0x20) raw = raw & ~3; // 10 bit res, 187.5 ms
  else if (cfg == 0x40) raw = raw & ~1; // 11 bit res, 375 ms
  //// default is 12 bit resolution, 750 ms conversion time
}
celsius = (float)raw / 16.0;
fahrenheit = celsius * 1.8 + 32.0;

if(curent1>0.1 && tensiune2<14.4)
{
lcd.setCursor(0, 1);
lcd.print(celsius);
lcd.print(" C");
lcd.setCursor(9, 1);
lcd.print(fahrenheit);
lcd.print(" F");
}
Serial.println("Tensiune=");
Serial.println(tensiune2);
Serial.println("Curent=");
```

```
 Serial.print(curent1);
 Serial.println();
 Serial.println("    ");
  Serial.println("  Temperature = ");
  Serial.print(celsius);
  Serial.print(" Celsius, ");
  Serial.print(fahrenheit);
  Serial.println(" Fahrenheit");
  Serial.println("    ");
  delay(500);

}



void asa(void)
{

  sumatens=0;
     for (int i=1; i <= 5; i++)
     {
       float tensiune = analogRead(A1);  //citim tensiunea de la pinul A1
       tensiune1 = 3*5*tensiune/1023.0 ;  //transfromam tensiunea
masurata la pinii MCU in tensiunea reala nedivizata
       sumatens = sumatens + tensiune1;
       }                              //facem o suma de 5 masuratori pentru
precizie
     tensiune2 = sumatens / 5.0 ;
    lcd.clear();
    lcd.setCursor(0, 0);
```

```
  lcd.print(tensiune2);        //afisam tensiune pe ecran
  //Serial.println(tensiune2);  //scriem tensiunea pe serial
   lcd.print(" V");



   curent = analogRead(Pincurent);// masuram tensiunea de la intrarea
analogica A0
 Voltage = ((curent)/1023.0) * 5000; // se converteste in mV
 Amps = ((Voltage - ACSoffset) / mVperAmp);  //se calculeaza curentul
masurat de senzor
  curent1 =(1)*Amps ;
  lcd.setCursor(9, 0);
   lcd.print(curent1);  //afisam curentul masurat plus unitatea de masura
   lcd.print(" A");
   delay(10);
   if(curent1<0.1)
   {
     lcd.setCursor(0, 1);
     lcd.print("legati bateria");
   }
   if(tensiune2>14.4)
   {
     lcd.setCursor(0, 1);
     lcd.print("baterie incarcata");
   }

  }
```

8. Bibliografie

8.1 https://www.arduino.cc/en/Tutorial/HelloWorld

8,2 https://forum.arduino.cc/index.php?topic=400304.0

8.3 http://forum.arduino.cc/index.php?topic=141030.0

8.4 http://www.ti.com/lit/ds/symlink/max232.pdf