Lucrarea nr. 4 Aplicaţie de identificare radio (RFID - Radio frequency identification)

1. Obiectivul lucrării

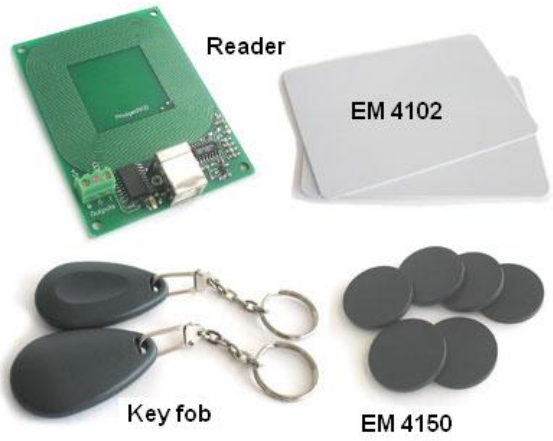
Lucrarea îşi propune studierea unui exemplu de implementare a unei aplicaţii RFID pe platformă de tip microcontrolor. Se analizează un alt mediu de programare pentru microcontroloare care include un compilator "C" dedicat pentru familia de microcontroloare PIC.

2. Consideraţii teoretice

În diverse aplicaţii se solicită identificarea automată a persoanelor sau produselor pe baza unor coduri ataşate. Cele mai utilizate ssteme de codificare şi recunoaştere automată sunt cele pe bază de coduri de bare şi respectiv cele de tip RFID. În primul caz codurile sunt marcaje pe suport de hârtie usor de identificat pe cale optică. Marcajele se pot tipări pe o linie sau pe două directii. Codurile sunt formate dintr-un număr limitat de cifre şi o dată aplicate nu pot fi modificate. Pentru a fi identificat codul trebuie să fie adus în zona de vizibilitate a cititorului optic.

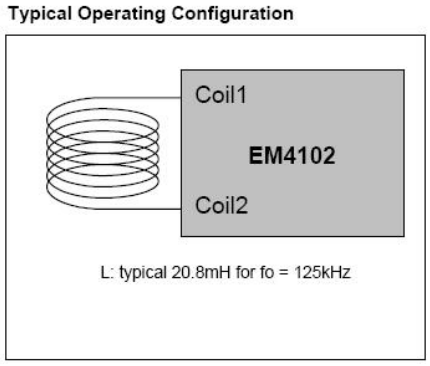
Mult mai versatile sunt etichetele (tag-urile sau transponderele) de tip RFID care pot fi identificate şi uneori chiar modificate prin unde radio. Din punct de vedere principial un tag RFID este un microcircuit dotat cu o antenă radio încorporat într-un suport de plastic, hârtie, etc. Cititorul RFID emite o undă radio care alimentează tag-ul RFID şi în acelaşi timp solicită codul înscris în tag. În preajma unui cititor, tagul emite o undă radio care conţine codul tag-ului respectiv. De obicei codul conţine o secvenţă de lungime predefinită de biţi (40 , 64, 96 biti de date). In cazul tag-urilor mai evoluate sunt permise operaţii de ştergere şi rescriere repetate, iar capacitatea de stocare a datelor este semnificativ mai mare.

Astfel de circuite de identificare se pot utiliza pentru evidenţa automată a miscării produselor sau pentru identificarea persoanelor autorizate să acceseze un anumit perimetru. Există implementări de sisteme de autentificare a persoanelor bazate pe tag-uri RFID ce permit stocarea datelor relevante ale persoanei (date demografice sau informaţii medicale).



In principiu pentru a construi un cititor RFID sunt necesare următoarele elemente:

- o antenă construită sub forma unei bobine
- un emiţator-receptor radio specializat pe standardul RFIF
- un circuit "inteligent" pentru interpretarea mesajelor şi transmiterea acestora pe cale serială la un calculator; de obicei se foloseşte un microcontrolor



Tag-urile RFID sau Transpoderele contin un circuit integrat si o bobina pe post de antenă. Ele pot fi active sau pasive. Cele active contin propria sursa de energie folosita in pastrarea informatiei in memorie si comunicare. Cele pasive se bazeaza pe energia electrica a frecventei purtatoare a semnalului radio generat de cititor. Acestea contin o memorie de tip EEPROM pentru a pastra informatia in timpul in care transponderul (tag-ul) nu se afla in aria unei antene.

In functie de complexitate tag-urile au drepturi de citire, scriere, protectie cu parola, criptare, protectie la suprapunerea mai multor tag-uri pe aceeasi antena, rescriere multiplă, etc. Cele incluse in kit-ul ce se studiază în acest laborator sunt: EM 4102 – citire, EM4150 – citire, scriere, parola pe

32 biti. Date suplimentare despre aceste dispozitive pot fi obţinute din foile de catalog ale circuitelor (www.emmicroelectronic.com/webfiles/Product/RFID/DS/EM4102_DS.pdf, www.emmicroelectronic.com/webfiles/Product/RFID/DS/EM4150_DS.pdf)

Antena este un element cheie al sistemului RFID. Dimensiunea, forma si calitatea antenei influenteaza distanta comunicarii. Odata cu dimensiunea creste si raza de actiune a acesteia. Este un echilibru intre limita maxima de constructie a antenei si limitele de siguranta ale parametrilor tensiunilor care vor alimenta circuitul integrat al tag-urilor.

Microcontrolorul gestionează comunicaţia cu transpoderele şi cu calculatorul gazdă.. Acesta genereaza o frecventa purtatoare catre antena care prin fluctuatii de camp detecteaza prezenta transponderelor la o distanta, ce poate să varieze de la câţiva centimetri la câţiva zeci de centimetri (aprox. 5-10cm pentru kitul utilizat în laborator). Comunicaţia cu calculatorul gazdă se face pe un canal serial asincron de tip RS232 sau RS485. Folosirea standardului RS485 permite conectarea mai multor dispozitive de tip RFID sau cu alte functionalităţi pe acelaşi tronson de cablu; printr-un sistem de adresare calculatorul gazdă, care joacă rolul de master poate citi succesiv dispozitivele slave conectate în circuit. Deoarece un calculator obisnuit de tip PC nu are o interfaţă RS485, ci doar una de tip RS232, trebuie sa se utilizeze un circuit de conversie RS485-RS232. In kit-ul studiat acest convertor este realizat cu un microcontrolor PIC în care s-a înscris un program de conversie.

In schemele anexate se pot identifica componentele cititorului RFID şi ale convertorului RS232-485. Date despre cele două standarde de comunicaţie seriale RS232 şi RS485 pot fi obţinute de pe Internet (www.lammertbies.nl/comm/info/RS-232_specs.html, www.lammertbies.nl/comm/info/RS-485.html**)**

Un cititor RFID functioneaza pe mai multe game de frecventa: 50-500 kHz, 13.6 MHz si 0.9-2.5 GHz. Cele de frecventa mica sunt mai folosite datorita costului redus raportat la o performată satisfacatoare.

Transponderele EM 4102 (Read-only)

Acestea contin 64 biti de informatie structurati astfel : 9 biti de header, 40 bits de date(D), 10 biti de paritate pe rand (P), 4 biti de paritate pe coloana. (C) si un bit de 0 de stop (S0).

1	1	1	1	1	1	1	1	1	9 biţi header
8 biţi de versiune sau Customer ID				D00	D01	D02	D03	P0	
				D10	D11	D12	D13	P1	
				D20	D21	D22	D23	P2	
				D30	D31	D32	D33	P3	
32 biţi de date				D40	D41	D42	D43	P4	
				D50	D51	D52	D53	P5	
				D60	D61	D62	D63	P6	
				D70	D71	D72	D73	P7	
				D80	D81	D82	D83	P8	
				D90	D91	D92	D93	P9	10 biţi de paritate pe
				PC0	PC1	PC2	PC3	S0	

4 biţi de paritate pe coloană

Cei 40 biti de date sunt structurati dupa cum arata tabelul de mai jos: 8 biti alocati pentru campul Customer ID si 4 secvente de 8 biti pentru Tag Id, care împreună (32 de biti) pot codifica peste 4 miliarde de id-uri unice.

Customer ID	Tag Id	Tag Id	Tag Id	Tag Id

Atunci cand tag-ul EM 4102 este pozitionat in zona cititorului RFID, care emite un semnal radio de 125kHz, va transmite în mod continuu secventa de 40 de biti. Secvenţa va fi preluată şi interpretată de cititor.

Pentru evaluarea unui sistem de tip RFID se foloseşte un sistem de dezvoltare RFID oferit de firma CCS Inc. (Custom Computer Services, http://www.ccsinfo.com/product_info.php?products_id=RFIDkit). Firma este specializată în instrumente şi platforme de dezvoltare pentru microcontroloare, în speţă circuite din familia PIC.

Sistemul de dezvoltare este compus din următoarele componente:

1. placă “Cititor RFID”
2. placă de conversie RS232-RS485
3. modul de programare şi depanare ICD U40 (In circuit debugger, USB)
4. alimentator de reţea
5. trei tipuri de transpondere (tag-uri RFID)
6. cabluri de legătură: RS232, RS485, USB şi de depanare (DEP)

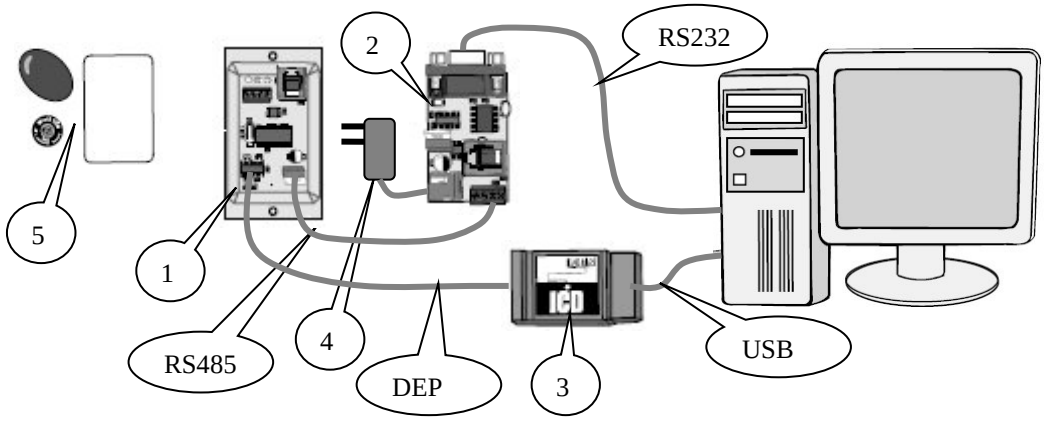


Figura 1. Sistemul de dezvoltare RFID

Pentru dezvoltare aplicaţiei se foloseste mediul PCW IDE şi compilatorul PICC. Mediul se va instala de pe CD-ul ataşat sistemului de dezvoltare. De asemenea se va instala driver-ul pentru modulul ICD-U40, în momentul în care circuitul este cuplat la o intrare USB a calculatorului gazdă. Pentru detalii prvind modul de instalare se va consulta manualul de utilizare al sistemului de dezvoltare.

Se realizează schema din figura 1, iar alimentatorul se cuplează la retea. Sistemul este pregătit pentru realizarea, încărcarea şi execuţia primului program scris în limbajul PICC (variantă de limbaj C pentru circuitele PIC). Programul aprinde şi stinge un LED pe placa ţintă, echivalent cu un program “Hello world” pentru un PC.

După lansarea mediului PCW IDE se parcurg următorii paşi:

a. Editarea programului

– se închid eventualele fişiere deschise prin: *File>Close All* (File= prima icoana din meniu)

- se creează un nou fişier sursă cu: *File>New* şi se atribuie numele de *EX3.C*

- se copiază programul din Anexa1 denumit EX3.C în noul fişier creeat; programul cuprinde:

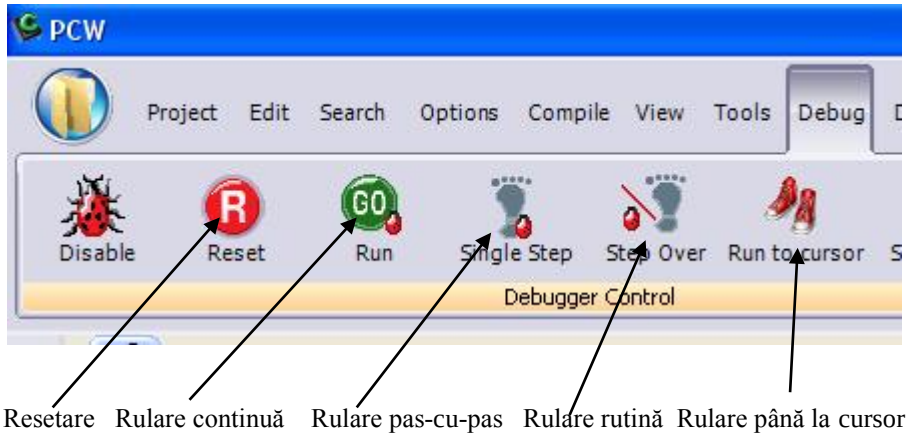
  - directiva “include <16F876A.h>” pentru a include declaraţiile specifice pentru circuitul cu care se lucrează
  - linia de configurare a “fuzibilelor”; detalii despre configurarea fuzibilelor se obţin prin *View>Valid Fuses*
  - un set de 3 declaraţii care fac legătura dintre pinii microcontrolorului şi cele 3 LEDuri de pe placă

- secvenţa de program care aprinde şi stinge la infinit unul dintre LEDuri; pentru temporizare se foloseşte rutina “delay_ms()” iar pentru setarea şi resetarea ieşirii pentru LED se folosesc funcţiile: *output_low*() şi respectiv *output_high()*
- se compilează programul prin: *Compile>Compile*; compilarea se va face cu opţiunea 14 Bit
-se salvează fişierul prin: File>Save All

b. Creerea unui proiect
- se construieste un nou proiect prin: *Project>Create*
- -se alege fişierul sursă fişierul creeat anterior
- se setează tipul de dispozitiv pentru care se generează proiectul: *PIC16F876A*
- - se apasă butonul *Apply*

c. Încărcarea şi lansarea programului
- programul se descarcă prin: *Tools>ICD* sau prin *Compile>Program Chip>ICD*
- pe placă se observă aprinderea şi stingerea periodică aLEDului verde

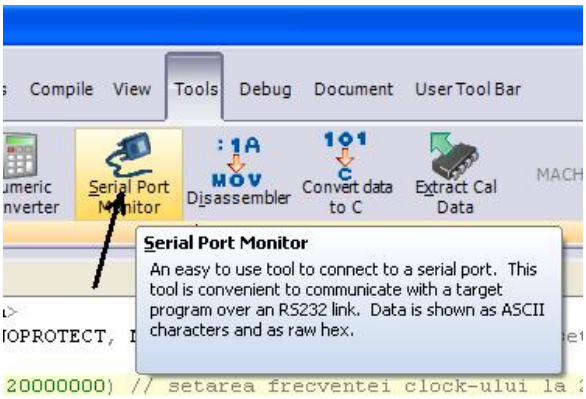
d. Rularea programului în regim de depanare (debugger)
- programul se descarcă şi se rulează prin: *Debug* sau *Compile>Debug>PCW*
- rularea programului se poate face în regim continuu sau pas-cu-pas, conform schemei de mai jos; prin apăsarea butonului RUN pe placă se observă aprinderea şi stingerea LEDului.



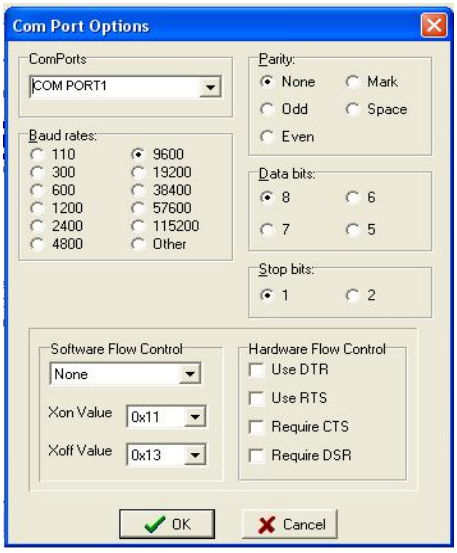
TOCMAI ATI REALIZAT PRIMUL PROGRAM IN PICC COMPILER!

Următorul program testat (ex5.c) realizează comunicaţia dintre placa ţintA şi PC prin interfaţa serială asincronă RS232. Se 6inchide proiectul anterior şi se creează unul nou în mod similar cu cazul precedent. In fişierul sursa se va copia exemplul 5 din Anexă.

Pentru monitorizarea canalului serial se porneşte un utilitar (similar cu HyperTerminal din Windows) prin secvenţa: *Tools>Serial Port Monitor*. Inainte de a continua, va trebui sa conectam de aceasta daca si cablul serial la interfata RS232 de la PC pentru a putea facilita



comunicarea cu sistemul nostru RFID.



Atentie, CABLUL SERIAL SE VA CONECTA INAINTE DE ALIMENTAREA DE LA RETEA A PLACII

Se configurează apoi port serial pe care se va efectua comunicaţia, conform schemei alăturate. Programul va astepta citirea unui caracter de la monitorul serial (un caracter asci sau un cod hex). Pentru o mai buna siguranta, se recomanda trimiterea lor pe casuta de editare HEX dupa modelulu: Hex: 47 0d 0a ceea ce înseamnă în codificare ASCII: G, CR, LF. Dupa cum reiese si din codul sursa, la apasarea tastei G se va aprinde LEDul verde, R LEDul rosu, si O ambele LEDuri..

După aceste exerciţii premergătoare se poate trece la testarea programului care implementează funcţia de citire RFID.

3. Desfăşurarea lucrării:

3.1. Se vor studia protocoalelor seriale RS232 si RS485 pe baza documentaţiei de la link-urile indicate în paragraful precedent; identificaţi diferentele dintre cele 2 standarde.

3.2 Rulaţi exemplele de program descrise mai sus, iar apoi modificaţi programele prin adăugarea de noi funcţionalităti (ex: aprinderea mai multor LEDuri, într-o anumită secvenţă).

3.3 Folosind exemplele de mai sus, scrieti un program care citeste un sir de n caractere, şi în funcţie de textul transmis (şi identificat de program) va aprinde diferite combinaţii de LEDuri. Programul va rula in ciclu infinit.

3.4 Se va scrie un program care citeste pachetul de date trimis de un transponder de tip em 4102 si il va transmite la PC pentru afisarea pe ecran; se va folosi programul Ex7 din anexa, care are urmatoarele elemente:

-directivele de includere a unor module specifice pentru tag-uri, convertorul RS232-485 şi unele utilitare

```
#include <em4095.c> //controls the reader IC
#include <em4102.c> // allows reading 4102 transponders
#include <rs485.c> // protocol de comunicatii
#include <utilities.c>
```

- in programul principal (main) apar următoarele funcţii :

```
rf_init();        //initializare cititor RFID
rf_powerUp();     //alimentarea antenei
rs485_init();     //iniţializarea conversiei RS232-485
```

- directive de declarare a variabilelor:
  - int8 – intreg pe 8 biti
  - int32 – intreg pe 32 biti
- funcţii utile:
  - int 32 make32(int a, int b, int c, int d) // împachetare date pe 32 biţi (conform structurii pachetului de date trimis de EM //4102)
  - sprintf(sir_de_caractere,”mesaj”); // copiază parametrul 2 în şirul de caracterde dat de parametrul 1

- boolean read_4102(msg) - > returneaza true daca pachetul de 40 biti din tag a fost salvat in varibila msg.
- char rs485getc() – asteapta citirea unui character de la tastatura pe consola de SIOW (searial port monitor)
- void rs485send(msg) // trimite pe consola mesajul indicat de “msg”
- itoa(customerCode,10, msg) //conversie int to ascii

3.5 Testati codul ex8.c in cadrul unui nou proiect. Remarcati similitudinea cu programul realizat la problema 3.3; de această dată codul nu se va mai citi de la tastatura ci va fi trimis de tag-ul RFID apropiat de antenă.

Anexea 1 Programe

Programul Ex3

```
#include <16F876A.h>
#fuses HS, NOWDT, NOPROTECT, NOLVP, NOBROWNOUT, PUT // biti de setare

#USE delay(clock = 20000000) // setarea frecventei clock-ului la 20Mhz

#define GREEN_LED  PIN_C3 // redenumirea pinului 4 din PORTC
#DEFINE YELLOW_LED PIN_C4 // redenumirea pinului 5 din PORTC
#DEFINE RED_LED    PIN_C5 // redenumirea pinului 6 din PORTC

void main()
{
  while (true)
    {
      output_low(GREEN_LED); // in logica negativa, 0 -> led aprins
      delay_ms(1000); // tine ledul aprins 1000 ms = 1 sec
      output_high(GREEN_LED); // 1 -> led stins
      delay_ms(1000);
    }
}
```

Programul Ex5

```
#include "rfid.h"
#define RS485_ID          0x11
#define RS485_USE_EXT_INT  FALSE
#define ADAPTER_RS485_ID   0X7F
#include <rs485.c>

 int msg[32] ;
typedef enum {OFF, GREEN, RED } Ledcolor;

 void twoColorLed (Ledcolor color) {
                switch (color) {
```

```
                case OFF: {
                          output_low(PIN_A3);
                          output_low(PIN_A5);
                          break;
                         }
                case GREEN:{
                          output_high(PIN_A3);
                          output_low(PIN_A5);
                          break;
                         }

                case RED:{
                          output_low(PIN_A3);
                          output_high(PIN_A5);
                          break;
                         }

                  }
              }
// trimite un sir de caractere consolei SIOW de comunicare seriala
void RS485send(char * s){
               int8 size;
               for(size = 0 ; s[size]!= '\0'; ++size)
                    rs485_wait_for_bus(FALSE);
               while (!rs485_send_message(ADAPTER_RS485_ID, size, s)) {
                  delay_ms(RS485_ID);
               }
              }
              /* citeste un caracter din monitorul comnuicari seriale
                 SIOW. Din casuta de editare HEX se va trimite codul hex
                 al cifrelor G , R sau O urmate de caracterele 0D 0A.
                  Ex: 47 0d 0a, pt caracterul G
              */
              char RS485getc(){
                rs485_get_message(msg, TRUE);
                return msg[2];
              }

 void main(void) {
                output_low(GREEN_LED); //SHOW POWER IS ON
                rs485_init();   // initializare comunicare seriala
                            // + conversie rs485 -> rs232

                while (true) {
                  sprintf(msg, "(O)ff , (G)reen, (R)ed \n\r");
                  RS485send(msg);
                  switch(toupper(RS485getc())) {
                    case 'O': twoColorLed(OFF); break;
                    case 'G': twoColorLed(GREEN); break;
                    case 'R': twoColorLed(RED); break;
```

```
    }
   }
  }
```

Programul Ex7

```
#include "rfid.h"
#include <em4095.c> //controls the reader IC
#include <em4102.c> // allows reading 4102 transponders
#define ADAPTER_RS485_ID   0X7F

#include <rs485.c>
int8 msg[32];
 // trimite un sir de caractere consolei SIOW de comunicare seriala
 void RS485send(char * s){
                 int8 size;
                 for(size = 0 ; s[size]!= '\0'; ++size)
                      rs485_wait_for_bus(FALSE);
                 while (!rs485_send_message(ADAPTER_RS485_ID, size, s)) {
                    delay_ms(RS485_ID);
                 }
                }
                /* citeste un caracter din monitorul comnuicari seriale
                   SIOW. Din casuta de editare HEX se va trimite codul hex
          al cifrelor G , R sau O urmate de caracterele 0D 0A.
                    Ex: 47 0d 0a, pt caracterul G
                */
                char RS485getc(){
                  rs485_get_message(msg, TRUE);
                  return msg[2];
                }

void main(){
                   int8 customerCode;
                   int32 tagNum;
                   rs485_init();   // initializare comunicare seriala
                               // + conversie rs485 -> rs232

                   rf_init();         //initialize the RF reader
                   rf_powerUp();      //power up the antenna
                   rs485_init();     // initialize rs485 comunication
                   output_low(GREEN_LED);// show the board is powered and ready
                   sprintf(msg, "Citeste card \n\r");
                    RS485send(msg);

                   for (;;){
                     if (read_4102(msg)) {
                       customerCode = msg[0];
                       tagNum = make32(msg[1], msg[2],msg[3],msg[4]);
```

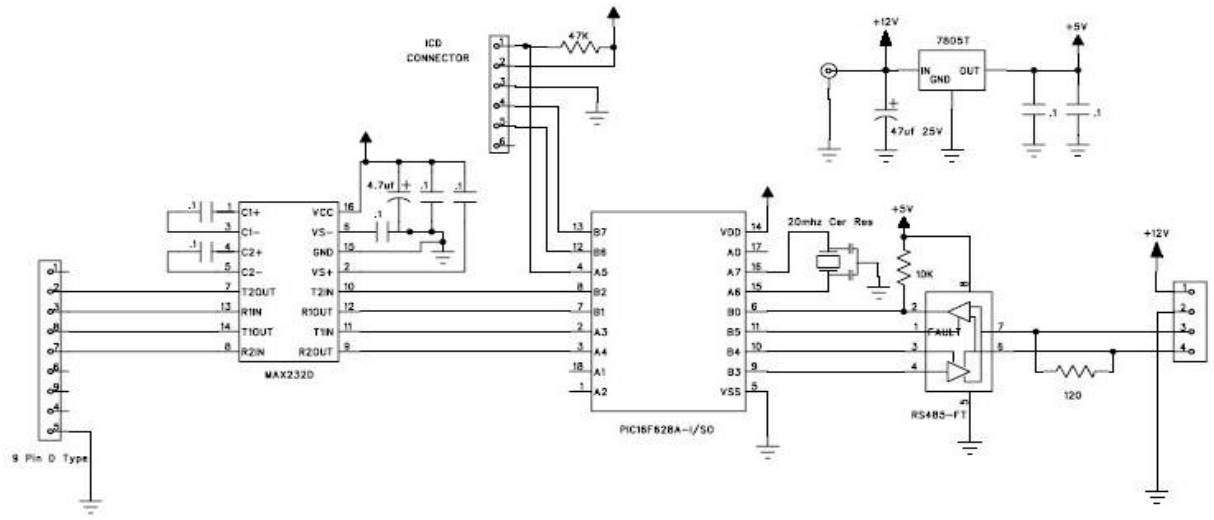
```
                sprintf(msg,"customer code: %u\n\r ", customerCode);
                RS485send(msg);

                sprintf(msg,"Tag number: %lu\n\r ", tagNum);
                RS485send(msg);
              }
            }//END FOR
          }//END MAIN
```

Programul Ex8

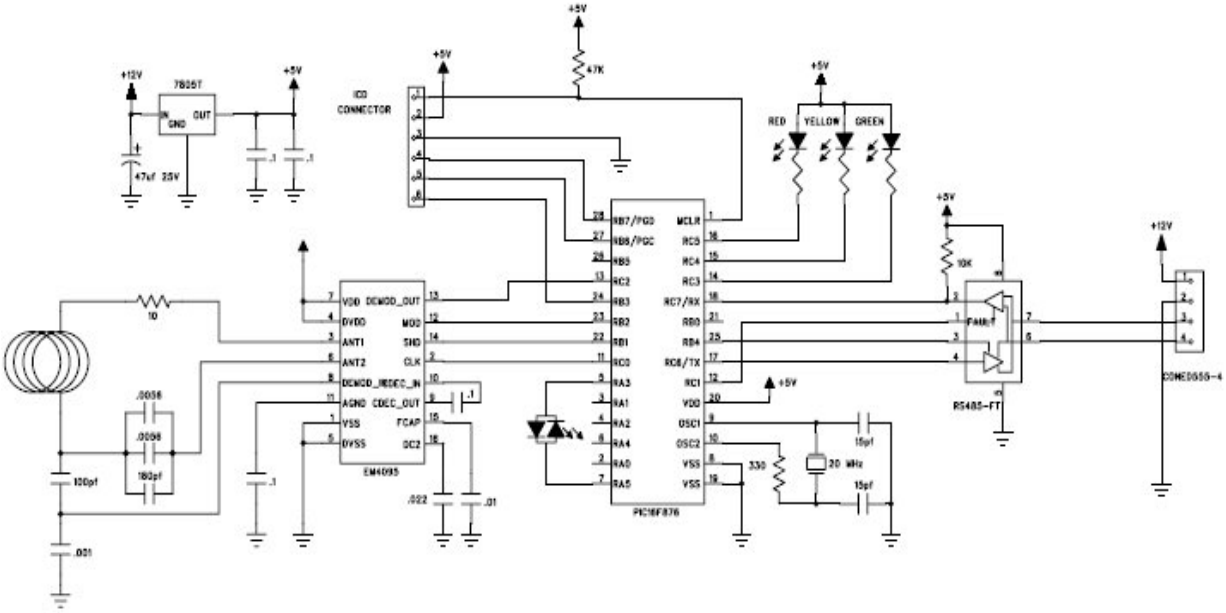
```
#include "rfid.h"
#include <em4095.c>
#include <em4102.c>
#include <rs485.c>
#include <stdlib.h>

int8 msg[32];
#include "utilities.c"

void main(){
  int8 customerCode, code;
  int32 tagNum, tag_ID;

  rf_init();
  rf_powerUp();
  rs485_init();
  output_low(YELLOW_LED);
  /*
  sprintf(msg, "Enter the customer code: ");
  RS485send(msg);
  code = RS485getInt();
    output_low(GREEN_LED);
    delay_ms(1000);
    output_high(GREEN_LED);

  sprintf(msg, "\n\n\r");
  RS485send(msg);

  sprintf(msg, "Enter the tag ID: ");
  Rs485send(msg);
  tag_ID = Rs485getI32();
    output_low(GREEN_LED);
    delay_ms(1000);
    output_high(GREEN_LED);

  sprintf(msg,"\n\n\rScanning...");
  Rs485send(msg);*/
  code = 176 ;
  tag_ID = 531324;
```

```
  for(;;){
    if (read_4102(msg)){

      customerCode =msg[0];
      tagNum = make32(msg[1], msg[2], msg[3], msg[4]);
     //itoa(customerCode,10, msg);//send code
      //RS485send(msg);
    //sprintf(msg, "\n\n\r");
      //RS485send(msg);
     //itoa(tagNum,10, msg);
     // RS485send(msg); // send tag id
       if ( customerCode == code && tagNum == tag_ID ){
         output_high(RED_LED);
         output_low(GREEN_LED);
       }else{
         output_high(GREEN_LED);
         output_low(RED_LED);
       }
       delay_ms(2000);
       output_high(GREEN_LED);
       output_high(RED_LED);
    }
  }
}
```

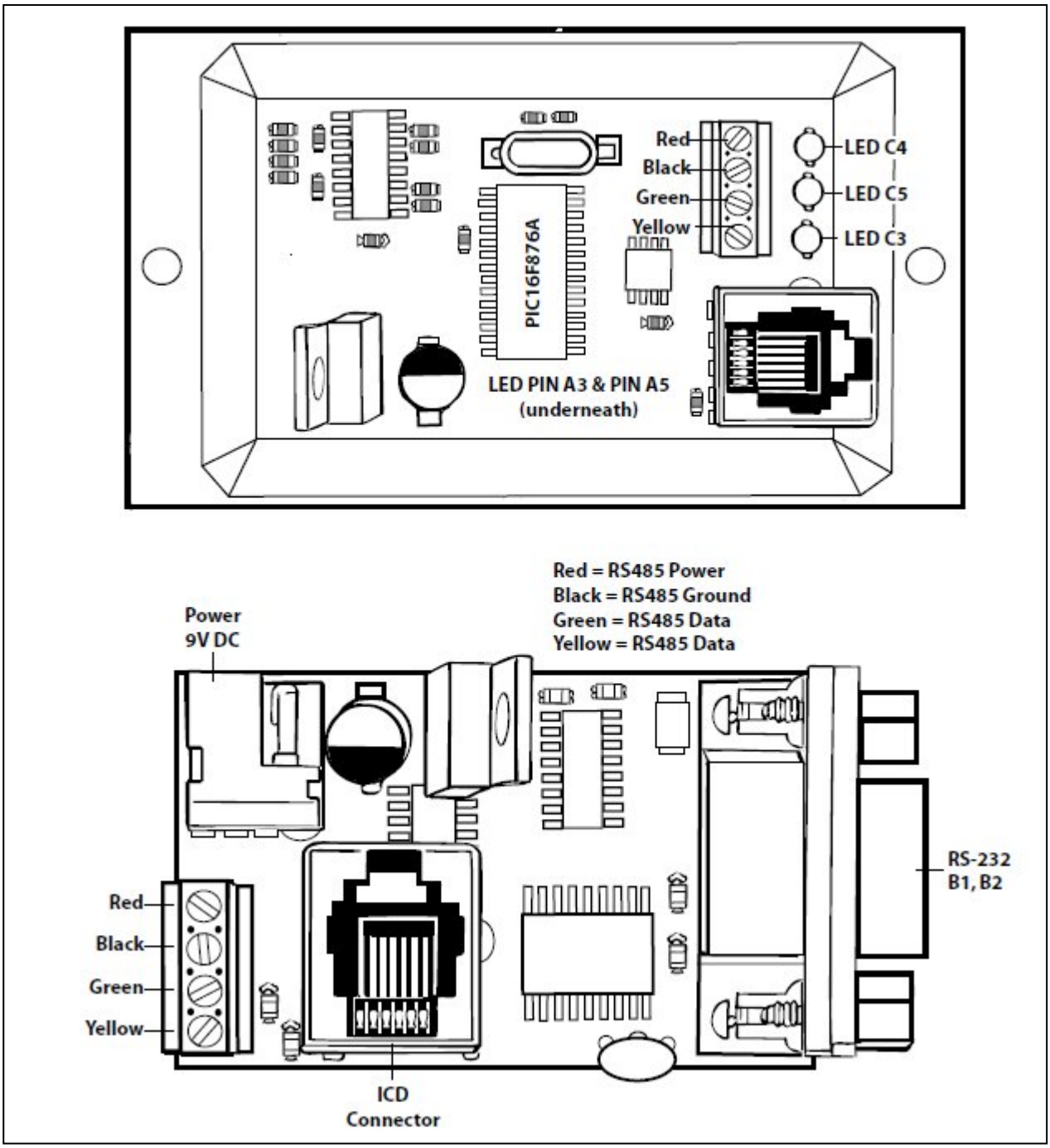
Anexa 2 Scheme electrice şi desene de amplasare



Scema circuitului de conversie RS232-RS485



Schema Cititorului RFID



Schemele de amplasare a componentelor pe cele două plăci