

Une des façons de simuler le fonctionnement de cette installation consiste en l'utilisation d'un programme représenté dans le fichier `COMMUNIC_RS232.c` du code-source 4.11. Cet exemple illustre l'utilisation de la fonction **Software_UART** de bibliothèque du compilateur *mikroC PRO*.

Dans le schéma de simulation donné par la figure 4.9, la visualisation de chaque octet envoyé et reçu via la communication série est affichée à l'aide d'un programme standard de PROTEUS appelé *VIRTUAL TERMINAL*.

Avant de compiler le fichier `COMMUNIC_RS232.c`, il est nécessaire de cocher **Software_UART** dans le gestionnaire de bibliothèque du compilateur mikroC PRO.

Code-source 4.11

```

/*****
                                COMMUNICATION SERIE RS232
                                =====
Cet exemple illustre l'utilisation de fonction Software_UART de la
bibliothèque du compilateur mikroC PRO.
Microcontrôleur: 16F887
Oscillateur : HS, 10.0000 Mhz
Fichier: COMMUNIC_RS232.c
Date: Juillet 2012
*****/
char error, byte_read;           // Variables auxiliaires

void main()
{
    ANSEL = 0;                    // Configuration E/S en numérique
    ANSELH = 0;
    TRISC = 0x00;                 /* Configurer PORTB en sortie
                                   (signalisation d'erreur)*/
    PORTC = 0;                    // Pas d'erreur

    // Initialiser Soft_UART en 9600 bds

    error = Soft_UART_Init(&PORTC, 7, 6, 9600, 1);
    if (error > 0)
    {
        PORTC = error;            // Initialisation d'erreur
        while(1);                 // Stop programme
    }
    while(1)
    {
        // Boucle sans fin

        // Lire octet, puis tester drapeau d'erreur

        byte_read = Soft_UART_Read(&error);
        if (error)                // Si l'erreur a été détectée
            PORTC= error;         // signale error sur le PORTB
        else
            Soft_UART_Write (byte_read) ; /* Si l'erreur n'a pas été détectée,
                                             l'octet lu en retour */
    }
}

```

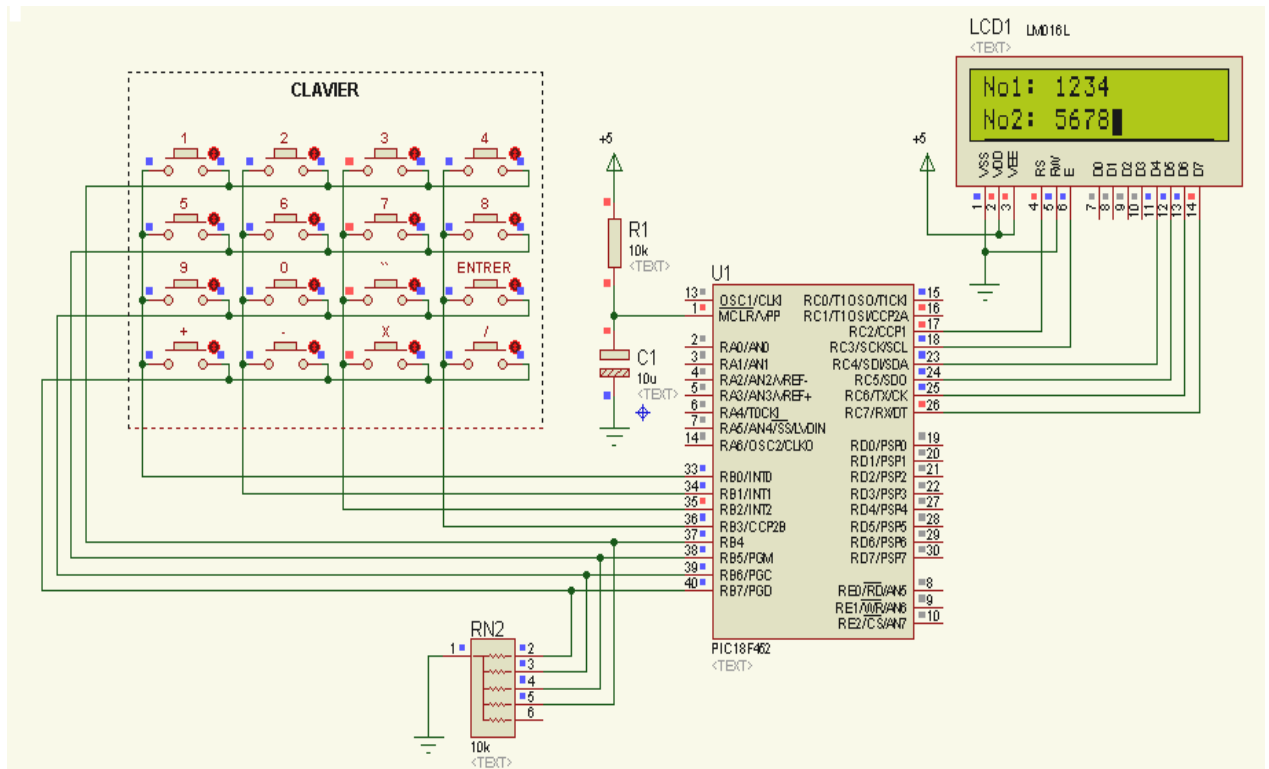


Fig.4.13. Schéma de simulation pour l'exemple 14

Exemple 15 Calculatrice avec l'interface RS232

Dans cet exemple, le PC est connecté au microcontrôleur via un câble RS232. Le projet c'est une simple calculatrice où les données sont envoyées au microcontrôleur en utilisant le clavier de l'ordinateur et ensuite ils sont affichés sur monitor de PC.

L'exemple de calcul :

CALCULATRICE PROGRAMME

Entrer Premier Nombre : 12

Entrer Deuxième nombre : 2

Entrer Opération : +

Résultat : 14

Le fichier de ce projet appelé *CALCUL_RS232.c* est donné dans le code-source 4.16.

Avant de compiler le fichier *CALCUL_RS232.c* il est nécessaire de cocher les fonctions **Conversions** et **Software_UART** dans le gestionnaire de bibliothèque du compilateur mikroC PRO.

Code-source 4.16

```
/******
                        CALCULATRICE AVEC L'INTERFACE RS232
                        =====
Le projet est une simple calculatrice. L'utilisateur saisit les données à l'aide
du clavier du PC. Les résultats sont affichés sur l'écran du PC.
Les opérations suivantes peuvent être effectuées:
+,-,*,/,.
Dans ce programme, la communication série est traitée par le logiciel
Soft_UART et le port série est configuré pour fonctionner avec 9600 Baud.
Broches du port RC6 et RC7 sont utilisés pour série TX et RX respectivement.
Microcontrôleur: 18F452
Oscillateur : HS, 10.0000 Mhz
Fichier: CALCUL_RS232.c
Date: Septembre 2012
*****/
#define Enter 13
#define Plus '+'
#define Minus '-'
#define Multiply '.'
#define Divide '/'

// Retour de chariot et saut de ligne à USART

void Newline()
{
    Soft_UART_Write(0x0D);          // Envoyer retour chariot
    Soft_UART_Write(0x0A);          // Envoyer saut de ligne
}

// Envoyer un texte à port série

void Text_To_Usart(unsigned char *m)
{
    unsigned char i;
    i = 0;
    while(m[i] != 0)
    {
        Soft_UART_Write(m[i]);      // Envoyer TEXT vers le port série
        i++;
    }
}

// Start du programme principale

void main()
{
    unsigned char MyKey, i,j,error,kbd[5],op[12];
    unsigned long Calc, Op1, Op2,Key;
    unsigned char msg1[] = " CALCULATRICE PROGRAMME";
    unsigned char msg2[] = " Entrer Premier Nombre : ";
    unsigned char msg3[] = " Entrer Deuxieme Nombre : ";
    unsigned char msg4[] = " Enter Operation : ";
    unsigned char msg5[] = " Resultat = ";

    // Configurer port série

    Soft_UART_Init(PORTC,7,6,9600,0); // TX=RC6, RX=RC7, Baud=9600
```

```
// Program loop
for(;;)                                // Boucle sans fin
{
    MyKey = 0;
    Op1 = 0;
    Op2 = 0;
    Newline();                          // Envoyer saut de ligne
    Newline();                          // Envoyer saut de ligne
    Text_To_Usart(msg1);                 // Envoyer TEXT
    Newline();                          // Envoyer saut de ligne
    Newline();                          // Envoyer saut de ligne
    // Pour premier nombre
    Text_To_Usart(msg2);                 // Envoyer message 2
    do                                  // Obtenir le premier numéro
    {
        do                             // Si un nombre est tapé
        {
            MyKey = Soft_UART_Read(&error); // Envoyer ce nombre
            while (error);
            if(MyKey == Enter)break;      // Si le clavier ENTER est pressé
            Soft_UART_Write(MyKey);      // Sauter à nouvelle ligne
            Key = MyKey - '0';
            Op1 = 10*Op1 + Key;           // Premier nombre de Op1
        } while(1);
        Newline();
    }
    // Pour deuxième nombre
    Text_To_Usart(msg3);                 // Envoyer message 3
    do                                  // Obtenir le deuxième nombre
    {
        do
        {
            MyKey = Soft_UART_Read(&error); // Envoyer le deuxième nombre
            while(error);
            if(MyKey == Enter) break;    // Si le clavier ENTER est pressé
            Soft_UART_Write(MyKey);      // Sauter à nouvelle ligne
            Key = MyKey - '0';
            Op2 = 10*Op2 + Key;           // Deuxième nombre de Op2
        } while(1);
        Newline();
    }
    // Pour opération
    Text_To_Usart(msg4);
    do
    {
        do
        {
            MyKey = Soft_UART_Read(&error);
            while(error);
            if(MyKey == Enter)break;
            Soft_UART_Write(MyKey);
            Key = MyKey;
        } while(1);
    }

    // Perform the operation

    Newline();
    switch(Key)                          // Calcule
    {
        case Plus:
            Calc = Op1 + Op2;             // Si Addition
            break;
        case Minus:
            Calc = Op1 - Op2;             // Si Soustraction
            break;
        case Multiply:
```

```

        Calc = Op1 * Op2;           // Si Multiplication
        break;
    case Divide:
        Calc = Op1 / Op2;           // Si Diviser
        break;
    }
    LongToStr(Calc, op);             // Convertir en ligne
// Remove leading blanks
    j=0;
    for(i=0;i<=11;i++)
    {
        if(op[i] != ' ')             // Si un espace
        {
            kbd[j]=op[i];
            j++;
        }
    }
    Text_To_Usart(msg5);
    for(i=0; i<j;i++)Soft_UART_Write(kbd[i]); // Afficher le resultat
}
}

```

Le schéma de simulation pour ce projet est représenté sur la figure 4.14

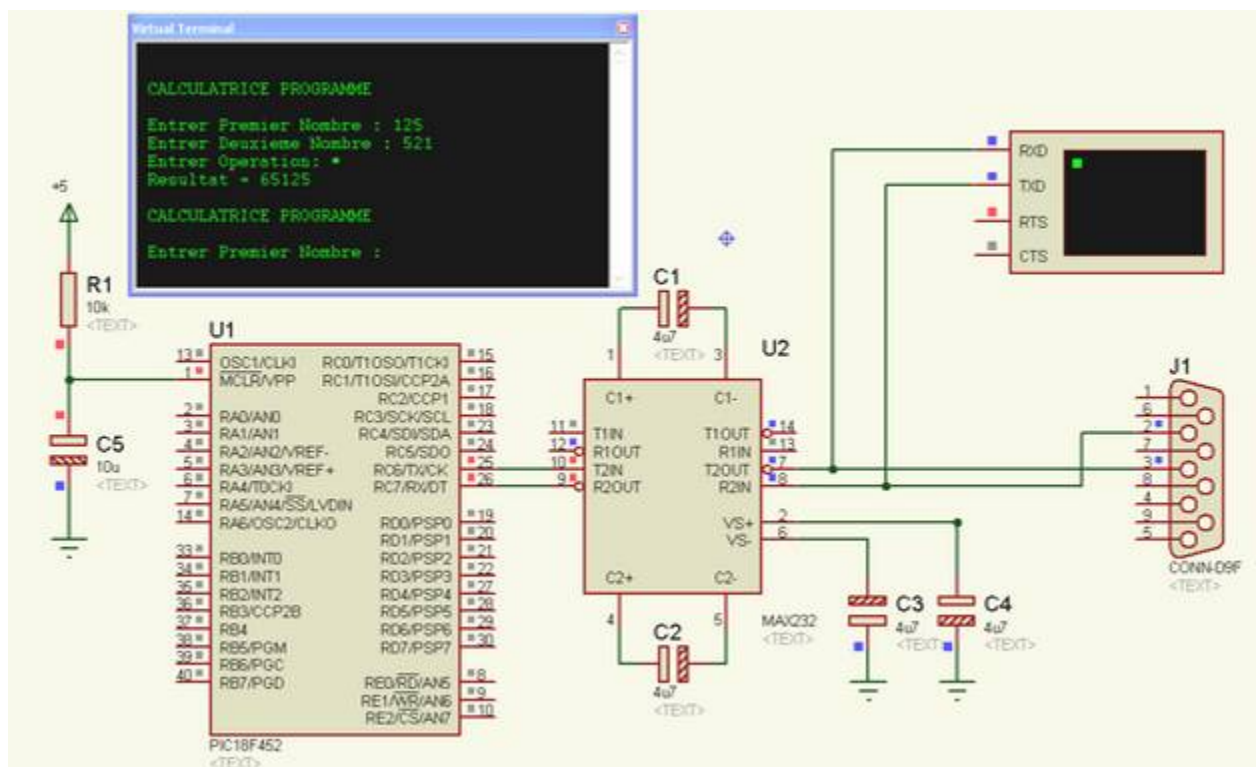


Fig. 4.14. Schéma de câblage pour la simulation de l'exemple 15