```
/***************************** C SOURCE FILE ********************************
**
**  Project:    Basic Menu Testing for UART0 Driver for CL2bm1
**  Filename:   Basic(U1)_Menu.c
**  Version:    1.0
**  Date:       v30.1.18
**  Modified    R.Oliva  - Include interrupt routines in C separate file (test)
**
*******************************************************************************
**
**
*******************************************************************************
**
**  VERSION HISTORY:
**  ----------------
**  initial Version:    1.0
**  Date:       30.1.18
**  Revised by: R.Oliva
**  Description:
**      -
**      - Newer versions see top.
**
**
**
*******************************************************************************/

#include "Basic(U1)_Menu.h"

/*******************************************************************************
**
** DEFINITIONS
**
*******************************************************************************/

// ***************************************************************************
// ** For Menus - 12.9.2010
char input_str[26];                     // 23.9.06 Input strings from COM0
char instr1[26];                        // For Get_String()
char *p_input_str;
char str[26];  // For replacing printfs() 30.1.2018

char Station_name[26];                  // Dummy station name 18.12.2017
float VBTH_PlusK = 25.8;                // Dummy float value 18.12.2017
//  For Menus.. 2.8.2010
unsigned char MSG_opt = 1;
char INTERV_TIME[15];
uint8_t Flag_Print_METEO = 0;
uint8_t NBin_com = 0;
uint8_t Interval_ST = 0;



//
*****************************************************************************************************************
******
//                                                  get_string1()
// This functions is for STRINGS from COMPORT1
// After call, global string: instr1[] contains the String read from COM1..
// rev 30.1.2018


int get_string1(void)
{
    // char instr1[26]; - set global..
    char c;
    int i;
    c = 0;
```

```
    instr1[c] = getchar1();
    // putchar1(instr1[c]); 29.10.06 Test Disabling putchar()
    while((isprint(instr1[c]) == 1) && (c < 25))
    {
        c++;
        instr1[c] = getchar1();
        // putchar(instr1[c]); 29.10.06 Test Disabling putchar()
        delay_ms(30); // Equiv to printf..
        //printf("-%d-",c);
    }
    // now that we have the number, null terminate the string
    // (after checking for the Escape!)
    if (instr1[c] == 0x1B)
        return 256;
    c++;
    instr1[c] = '\0';
    //strncpy(s,instr,25);
    return 0;
}


//
*****************************************************************************************************
******
//                                                    get1_val()
// This functions works OK for getting values from COMPORT1
// rev 30.1.2018

int get_val1(unsigned char *uc)
{
    char instr[6];
    char c;
    int i;
    uint8_t *Tempor;

    //Tempor = (unsigned char)(&RTCYear-2000); // Added for compatibility 2.8.2010
    c = 0;
    instr[c] = getchar1();
    putchar1(instr[c]);
    while((isdigit(instr[c]) == 1) && (c < 4))
    {
        c++;
        instr[c] = getchar1();
        putchar1(instr[c]);
    }
    // now that we have the number, null terminate the string
    // (after checking for the Escape!)
    if (instr[c] == 0x1B)
        return 256;
    c++;
    instr[c] = '\0';
    i = atoi(instr);
    // Strange condition - required Tempor calculation 2.8.2010
    if (i > 255) //&& (uc !=Tempor)
        i = 255;
    *uc = (unsigned char) i;
    return 0;
}




//
*****************************************************************************************************
******
//                                                    Check_UART1_Menu()
// Menu de ensayo para Driver UART1_Dr1 . c/.h tomado de PWRC2, limitado en funciones.
```

```
// Primera version 30.01.2018 - Es un menú para lazo,
//    utiliza MSG_option_local como parámetro, y modifica la variable de estado
//    global del Menu, llamada MSG_opt. Para verificar la lectura de un caracter, lee el valor
//    de rx_counter1 (modificado por ISR Rx presente en UART1_Dr1) y luego get_char1() alternativo
//    del mismo driver.
//  Variables globales a definir:
//            unsigned char MSG_opt = 1;
//            unsigned int NBin_com = 0;
//            unsigned char Interval_ST = 0;
//
*****************************************************************************************************
******

void Check_UART1_Menu( unsigned char MSG_option_local)
{
char c;
int i;
char a;
unsigned long c3;
unsigned int c2;
unsigned char UTemp, i_eep, pointer;
unsigned char TempYr;  // TempYr 00 to 99, since get_val will not compile for uints..
uint8_t temp_ubyte =0;     // For Vmin y otros.. 28-9-2010
float ftmp;
int FDeb;
uint8_t ch1;

    if (MSG_option_local == 1)
    {
    puts1("\n\r CL2bm1 - Inicializacion (Presione ? para comandos):\n\r" );
    puts1("\t?\t- Listar comandos.\r");
    puts1("\t1\t- Menu de Setup Basico \r");
    puts1("\t2\t- Menu de Setup Extendido \r");
    puts1("\t3\t- Lanzar Prueba c/param. actuales (nuevo archivo)\r" );
    puts1("\t4\t- Re-Lanzar Prueba (archivo en curso)\r" );
    puts1("\t5\t- Configurar Canales \r");   // v9c Agregado 29.7.2014
    // puts1("\t6\t- Otros Coeficientes - Manual \r");
    #ifdef TEST_EV
    printf("\n\r Archivo de Eventos: %s", SYS.ee_EV_FNAME);  // Added for testing..
    #endif
    // puts1("\t5\t- Volver a Modo Detenido\r");
    MSG_opt = 2;   // Set to next Menu level.. (global)
    }

  if (MSG_option_local == 3)
    {
    puts1("\n\r  CL2bm1 - Menu Setup Basico (Presione ? para comandos):\n\r" );
    puts1("\t?\t- Listar comandos.\r");
    puts1("\tC\t- Setear Fecha y Hora\r");
    puts1("\tc\t- Leer Fecha y Hora Actual.\r");
    puts1("\tA\t- Fijar Tiempo Almacenamiento a 1minuto ..\r" );
    puts1("\tB\t- Fijar Tiempo Almacenamiento a  5minutos ..\r" );
    puts1("\tE\t- Fijar Tiempo Almacenamiento a 10minutos ..\r" );
    puts1("\tD\t- Fijar Tiempo Almacenamiento a 15minutos ..\r" );
    puts1("\tF\t- Mostrar Seteos Actuales de la Prueba ..\r" );
    puts1("\tN\t- Ingresar Nombre Estacion  (25c max)\r" );
    puts1("\tn\t- Leer Nombre Estacion\r" );
    puts1("\tI\t- Ingresar ID Estacion (25c max)\r" );
    puts1("\ti\t- Leer ID Estacion.\r"    );
    puts1("\tW\t- Ingresar Limites DIR Excluidos (min, max º) \r" );
    puts1("\tw\t- Leer Limites DIR Excluidos (min, max º) \r" );
    puts1("\tV\t- Ingresar Nivel de Tension de Prueba (N/A/B) \r" );
    puts1("\tv\t- Leer Nivel de Tension de Prueba (N/A/B) \r" );
    puts1("\tM\t- Modificar Niveles de Tension Nominales \r" );
    puts1("\tm\t- Leer Niveles de Tension Nominales \r" );
    puts1("\tS\t- Espacio disponible en SD y Nº Dias \r" );
    puts1("\ts\t- Listar Archivos en SD y tamaño \r" );
```

```
    puts1("\tp\t- Mostrar e ingresar Nro Bins p/Compl.\r");       // 23.2.2007 -- added.
    puts1("\tx\t- Volver a Menu anterior \r\n");
    MSG_opt = 4;   // Set to next Menu level..
    }

    if (MSG_option_local == 5)
    {
    puts1("\n\r  CL2bm1 - Menu Setup Extendido (Presione ? para comandos):\n\r");
    puts1("\tF\t- Ingresar Fabricante del Equipo (25c max)\r");
    puts1("\tf\t- Leer Fabricante del Equipo.\r"    );
    puts1("\tB\t- Ingresar Direccion Modbus de la Estacion (1-255).\r");
    puts1("\tb\t- Leer Direccion Modbus de la Estacion.\r"    );
    puts1("\tT\t- Mostrar Valores Setup Extendido.\r"    );
    puts1("\tg\t- ON/OFF impresion Valores Float.\r");
    puts1("\ty\t- Test de Watchdog (RESET!)\r");
    puts1("\tx\t- Volver a Menu anterior \r");
    MSG_opt = 6;   // Set to next Menu level..
    }

    while(rx_counter1)         // echo port 0..
    {
        c = getchar1();
        // putchar(c);
        if(MSG_option_local == 2) {
          switch (c)
            {
            case '?':
             puts1("\t?\t- Listar comandos.\r");
             puts1("\t1.\t- Menu de Setup Prueba \r");
             puts1("\t2.\t- Menu Setup Extendido \r");
             puts1("\t3.\t- Lanzar Prueba c/param. actuales (nuevo archivo)\r");
             puts1("\t4.\t- Re-Lanzar Prueba (archivo en curso)\r");
             puts1("\t5.\t- Configurar Canales \r");   // v9c Agregado 29.7.2014
             //puts1("\t6.\t- Otros Coeficientes - Manual \r");
             puts1("\tP.\t- Impresion METEO ON/OFF\r");
             break;
            case '1':           // Send to Setup Menu presentation..
             MSG_opt = 3;
             //EVT_LogEvent(EVT_ENTER_TSTSETUP);
             break;
            case '2':            //  Send to Hardware Menu..
             MSG_opt = 5;
             break;
             // COM1 Testing-Meteo..
             // Toggle debug printing flag
            case 'P':   // Change PrintFlag...
              if (Flag_Print_METEO) {
                      Flag_Print_METEO = 0;
                      puts1("\n\r Impresion METEO - OFF");  // Testing..
                      }
              else {
                      Flag_Print_METEO = 1;
                      puts1("\n\r Impresion METEO -  ON");  // Testing..
                     }
              break;
            case '3':
                   puts1("\n\r Lanza Prueba..");
                   //Command_3_Function();
                   break;
            case '4':
                   puts1("Reinicia Prueba...\n\r");
                   break;
            case '5':    // v9c - Cambiado a configuracion de canales..
                     puts1("Configurar canales...\n\r");
                     break;
            default:
                   puts1("\n\rComando no reconocido...\n\r");
```

```
          }  // End this switch..
        }    // end if '2'

    if(MSG_option_local == 4) {        // Enter Setup mode..
     switch (c)
        {
        case '?':
            puts1("\n\r  CL2bm1 Menu de Setup Prueba (Presione ? para comandos):\n\r" );
            puts1("\t?\t- Listar comandos.\r");
            puts1("\tC\t- Setear Fecha y Hora\r");
            puts1("\tc\t- Leer Fecha y Hora Actual.\r");
            puts1("\tA\t- Fijar Tiempo Almacenamiento a 1minuto ..\r");
            puts1("\tB\t- Fijar Tiempo Almacenamiento a  5minutos ..\r");
            puts1("\tE\t- Fijar Tiempo Almacenamiento a 10minutos ..\r");
            puts1("\tD\t- Fijar Tiempo Almacenamiento a 15minutos ..\r");
            puts1("\tF\t- Mostrar Seteos Actuales de la Prueba ..\r");
            puts1("\tN\t- Ingresar Nombre Estacion  (25c max)\r");
            puts1("\tn\t- Leer Nombre Estacion\r");
            puts1("\tI\t- Ingresar ID Estacion (25c max)\r");
            puts1("\ti\t- Leer ID Estacion.\r"    );
            puts1("\tW\t- Ingresar Limites DIR Excluidos (min, max º) \r");
            puts1("\tw\t- Leer Limites DIR Excluidos (min, max º) \r");
            puts1("\tV\t- Ingresar Nivel de Tension de Prueba (N/A/B) \r");
            puts1("\tv\t- Leer Nivel de Tension de Prueba (N/A/B) \r");
            puts1("\tM\t- Modificar Niveles de Tension Nominales \r");
            puts1("\tm\t- Leer Niveles de Tension Nominales \r");
            puts1("\tS\t- Espacio disponible en SD y Nº Dias \r");
            puts1("\ts\t- Listar Archivos en SD y tamaño \r");
            puts1("\tp\t- Mostrar e ingresar Nro Bins p/Compl.\r");       // 23.2.2007 -- added.
            puts1("\tx\t- Volver a Menu anterior \r\n");
               break;

        case 'p':  // Read and modify min. Bins for completion..
                puts1("\n\r Nro Minimo Bins para Completar (10=def.)= 10.."); // Moved to TSTAT
                16.6.2012
                // monitor for cancel!
                puts1("\n\r Ingresar Nvo. Valor[5-250][ENTER o ESC]:\n\r");
                i = get_val1(&temp_ubyte);
                if (i != 256) {
                      if((temp_ubyte >4) && (temp_ubyte<251)){
                            NBin_com = (unsigned int)temp_ubyte;
                            sprintf(str,"\n\r Nvo Valor NBins_min = %d..", NBin_com );
                            puts1(str);  //30.1.2018 to send to COM1
                            }
                      else {puts1("\n\r Fuera de Rango!");}
                }
                break;

        case 'c':  // Read Real Time Settings..
            puts1("\n\r Fecha/Hora interna:");
            rtc_get_timeNdate(&RTCHour, &RTCMin, &RTCSec, &RTCDay, &RTCMonth, &RTCYear);
            sprintf(str,"%02d/%02d/%04d-%02d:%02d:%02d ",
            RTCDay,RTCMonth,RTCYear,RTCHour,RTCMin,RTCSec);
            puts1(str);  //30.1.2018 to send to COM1
            break;

        case 'C':  // Modify Real time settings..
            puts1("\n\r Ingresar Fecha y hora como:   dd/mm/aa hh:mm:ss[ENTER]\n\r" );
            // monitor for cancel! - get_val only works for UBYTES, so for Yr use TmpYr.. 2.8.2010
            // Then convert as RTCYear = 2000 + TempYr;
            i = get_val1(&RTCDay);
            if (i != 256)
                i = get_val1(&RTCMonth);
            if (i != 256)
                i = get_val1(&TempYr);
            if (i != 256)
                i = get_val1(&RTCHour);
```

```
        if (i != 256)
            i = get_val1(&RTCMin);
        if (i != 256)
            i = get_val1(&RTCSec);
        if (i != 256)
        {
        RTCYear = 2000 + TempYr;  // Convert byte to  uint. 2.8.2010
        if(rtc_set_time(RTCHour,RTCMin,RTCSec)) {puts1("Error in RTC_set_time..!!\n\r");}
        else {puts1("OK Time.\n\r");}
        delay_ms(100);          // Added 10.3.2010 - Change to 1/2 value with ints..30-9-10
        if(rtc_set_date(RTCDay,RTCMonth,RTCYear)){ puts1("Error in RTC_set_date..!!\n\r");}
        else { puts1("OK Date.\n\r");}
        // rtc_set_time(hour,minute,sec);
        // rtc_set_date(date,month,year);
        }
        break;
     // Interval Tiime changing  -- 28-9.10
     // #define   INT_10MIN  1
     // #define   INT_30MIN  2
     // #define   INT_5MIN   3
     // #define   INT_1MIN   4
    case 'A':
        #asm("cli")
        Interval_ST = INT_1MIN;
        #asm("sei")
        strcpyf(INTERV_TIME, "1 Minuto");
        puts1("\n\rFijado Tiempo Almacenamiento a 1 minuto ..\r");
        break;
    case 'B':
        #asm("cli")
        Interval_ST = INT_5MIN;
        #asm("sei")
        strcpyf(INTERV_TIME, "5 Minutos");
        puts1("\n\rFijado Tiempo Almacenamiento a 5 minutos ..\r");
        break;

    case 'E':
        #asm("cli")
        Interval_ST = INT_10MIN;
        #asm("sei")
        strcpyf(INTERV_TIME, "10 Minutos");
        puts1("\n\rFijado Tiempo Almacenamiento a 10 minutos ..\r");
        break;
    case 'D':
        #asm("cli")
        Interval_ST = INT_30MIN;
        #asm("sei")
        strcpyf(INTERV_TIME,"30 Minutos");
        puts1("\n\rFijado Tiempo Almacenamiento a 30 minutos ..\r");
        break;

    case 'F':   // Show all parameters in normal setup..
        puts1("\n\r Parametros de la Prueba:");
        #define TEST_1ST_PART
        #ifdef TEST_1ST_PART
        #asm("cli")
        UTemp = Interval_ST;
        #asm("sei")
        if(UTemp == INT_1MIN){
                 puts1("\n\rTiempo Alm. = 1 minuto ..\n\r");
                 }
        else if (UTemp == INT_5MIN){
                 puts1("\n\rTiempo Alm. = 5 minutos ..\n\r");
                 }
        else if (UTemp == INT_10MIN){
                 puts1("\n\rTiempo Alm. = 10 minutos ..\n\r");
                 }
```

```
        else if (UTemp == INT_30MIN){
                puts1("\n\rTiempo Alm. = 15 minutos ..\n\r");
                }
        else{
                sprintf(str, "\n\rLeido %d: Incorrecto..", UTemp);
                puts1(str); //30.1.18 to com1
              }
        puts1("\n\r Archivo a usar: Datos.csv");   // f_namestr..
        sprintf(str, "\n\r Nombre Estacion %s ", Station_name);
        puts1(str); //30.1.18 to com1

        puts1("\n\r ID Prueba ID01 ");
        // printf("\n\r Intervalo Alm.: %s \n\r", INTERV_TIME);
        #endif
        #define TEST_2ND_PART
        #ifdef TEST_2ND_PART
        //if (PTST.ee_VTestLVL == 2) sprintf(tempstr, "V.BAJO, %2.2f, [V]",PTST.ee_VBLow_TH);
        //else if (PTST.ee_VTestLVL == 1) sprintf(tempstr, "V.ALTO, %2.2f,
        [V]",PTST.ee_VBHi_TH);
        //else sprintf(tempstr, "NOMINAL, %2.2f, [V]",PTST.ee_VBNom_TH);
        //printf("\n\r Nivel Umbral medio de Tensión de la prueba:, %s", tempstr);
        //printf("\n\r Valor VTH en RAM:, %2.2f,[V]",PTST.ee_VBNom_TH);   // Used to be VBTH
        15-6-2012
        //printf("\n\r Valor Porcentaje K (5%%nom): %d",PTST.ee_VBNom_percentage);
        // 5.11.06 Following computed within startup or initialization..
        // float  VBTH_PlusK              // Will store = ee_VB + 5%
        // float  VBTH_MinusK             // Will store = ee_VB - 5%
        // sprintf(tempstr, "+K%%, %2.2f, -K%%, %2.2f [V]",VBTH_PlusK,VBTH_MinusK);
            // 15.6.2012 Printout
        // VBTH_PlusK = VBTH (1+ VBNOM_PERCENTAGE/100)
        // VBTH_MinusK = VBTH (1- VBNOM_PERCENTAGE/100)
        // 15.6.2012 Assign accordingly..
        puts1("\n\r Nivel de Tension Prueba: NOM");
        puts1("\n\r Umbrales promedio 1min de Tensión:");
        printf("\n\r V_Nivel +K: %2.2f [V]", VBTH_PlusK);
        puts1("\n\r V_Nivel -K: 24.5 [V]");
        #endif
        // Horas REq..
        puts1("\n\r Horas de Prueba requeridas:, 12");
        puts1("\n\r>");
        break;

    case 'N':
        puts1("\n\rIngresar Nombre Estacion. (Max.25c + ENTER, ESC sale)\n\r");
        // p_input_str = &input_str[0];
        if (get_string1() == 0){
            //if(scanf("%s,\n", input_str) == 1){
            //       if(strlen(input_str) > 24) input_str[24]=0;   // Null terminate if too
            big..
           sprintf(str,"\n\r Ingresado OK: %s \n\r", instr1);
           puts1(str);
           for(i_eep=0; i_eep<25;i_eep++){
                 Station_name[i_eep] =  instr1[i_eep];   // EEPROM value read..
                 if(instr1[i_eep]==0) break;
                   }  // No strcpy for eeprom...
              puts1(" .. y copiado a EEPROM..\n\r");
         }     // ind if scanf = 1..
        else{
        puts1("\n\rError al ingresar nombre!\n\r");
        }
        break;

        // elim 17.2.2006
    case 'n':
        sprintf(str,"\n\r (copia en RAM): %s \n\r", Station_name);
        puts1(str);
        break;
```

```
      case 'I':            // read
          puts1("\n\rIngresar ID Estacion. (Max.25c + ENTER, ESC sale)\n\r");
          puts1("\n\r Ingresado OK: TST01 \n\r");
          break;
      case 'i':            // send the existing ID
          puts1("\n\r ID de Estacion TST01 \n\r");
          break;

      case 'W':  //Ingresar Wexc max,min
          puts1("\n\r  DIRECCIONES Excluidas de la prueba..");
          break;

      case 'w':            // show the existing excl. directions
          puts1("\n\r  Read DIRECCIONES Excluidas de la prueba..");
          break;

      case 'V':
          puts1("\n\r Cambiar Nivel de Tension de la Prueba:,");
          break;
      case 'v':
          puts1("\n\r Nivel Actual de Tension Prueba:");
          puts1("\n\r V_Nivel +K: %2.2f [V]");
          puts1("\n\r V_Nivel -K: 24.5 [V]");
          break;

      case 's':
          puts1("\n\r Archivos en Tarjeta SD:");
          break;

      case 'x':
          MSG_opt = 1; // Back to main menu..
          break;
      default:
           puts1("\n\rOpcion no reconocida..\n\r");
          } // End switch..
    }       // end if

  if(MSG_option_local == 6) {        // Enter Hardware mode..
    switch (c)
      {
      case '?':
          puts1("\n\r  CL2bm1 - Setup Extendido (Presione ? para comandos):\n\r");
          puts1("\tF\t- Ingresar Fabricante del Equipo (25c max)\r");
          puts1("\tf\t- Leer Fabricante del Equipo.\r"    );
          puts1("\tB\t- Ingresar Direccion Modbus de la Estacion (1-255).\r");
          puts1("\tb\t- Leer Direccion Modbus de la Estacion.\r"    );
          puts1("\tT\t- Mostrar Valores Setup Extendido.\r"    );
          puts1("\tg\t- ON/OFF impresion Valores Float.\r");
          puts1("\ty\t- Test de Watchdog (RESET!)\r");
          puts1("\tx\t- Volver a Menu anterior \r");
          break;

      case 'b':   // Read Modbus address
          puts1("\n\r Direccion Modbus de la Estacion 64 \n\r" );
          break;

      case 'B':   // New Modbus address
          puts1("\n\r Ingresar Nueva Direccion Modbus: 1-127 [ENTER]\n\r");
          break;

      case 'F':
          puts1("\n\rIngresar Fabricante Aerogenerador: (Max.25c + ENTER, ESC sale)\n\r");
          break;

          // elim 17.2.2006
      case 'f':
```

```
                puts1("\n\r Fabricante: EOLUX \n\r");
                break;


            case 'g':
                puts1("\n\r AD Values Print ON");  // Testing..
                break;

            //  Show all selections v16.6.2012 - Added Other Coefs.29-10-2012
            case 'T':
                puts1("\n\r Valores Setup Extendido:\n\r");
                break;

            case 'y': // 17.3.2007 - Test WDOG
                puts1("\n\r WDOG_test..");
                WD_ON_Flag = 1;                 // Set to 1 inhibits wdog petting.. 17.3.2007 v12e
                break;

            case 'x':
                MSG_opt = 1; // Back to main menu..
                break;
            default:
                putchar1(c);            // echo local until UDP echo is enabled by first reception
            }  // End switch..
        }      // End if.....
    }      // End while..

}           // End function...



/***********************************************
** Very simple puts1() implementation
** 30.1.2018 - 100 char limited
**
***********************************************/

void puts1(char *str)
{
    int i;
    for(i=0; i<100; i++)
    {
        if(str[i]=='\0')
        {
            putchar1('\n');
            break;
        }
        putchar1(str[i]);

    }

}
```