```
/***************************** C SOURCE FILE ********************************
**
**  Project:    Basic Menu Testing for UART0 Driver for CL2bm1
**  Filename:   Basic_Menu.c
**  Version:    1.0
**  Date:       v18.12.2017
**  Modified    R.Oliva  - Include interrupt routines in C separate file (test)
**
*******************************************************************************
**
**
*******************************************************************************
**
**  VERSION HISTORY:
**  ----------------
**  initial Version:    1.0
**  Date:       17.12.2017
**  Revised by: R.Oliva
**  Description:
**      -
**      - Newer versions see top.
**
**
**
*******************************************************************************/

#include "Basic_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 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_string()
// This functions is for STRINGS from COMPORT0
// After call, global string: instr1[] contains the String read from COM0..
// 29.10.06 Test Disabling putchar()
// 21-9-2010 Testing with v11..


int get_string(void)
{
    // char instr1[26]; - set global..
    char c;
    int i;
    c = 0;
    instr1[c] = getchar();
    // putchar(instr1[c]); 29.10.06 Test Disabling putchar()
    while((isprint(instr1[c]) == 1) && (c < 25))
    {
        c++;
        instr1[c] = getchar();
        // putchar(instr1[c]); 29.10.06 Test Disabling putchar()
        delay_ms(30); // Equiv to printf..
        //printf("-%d-",c);
    }
```

```
76          // now that we have the number, null terminate the string
77          // (after checking for the Escape!)
78          if (instr1[c] == 0x1B)
79              return 256;
80          c++;
81          instr1[c] = '\0';
82          //strncpy(s,instr,25);
83          return 0;
84      }
85
86
87      // ********************************************************************************************************
88      //                                                 get_val()
89      // This functions works OK for getting values from COMPORT0
90      // putchar(instr[c]);    29.10.06 Test Disabling putchar()
91      // 2.8.2010 Compiler does not cast from uint argument to uchar..
92
93      int get_val(unsigned char *uc)
94      {
95          char instr[6];
96          char c;
97          int i;
98          uint8_t *Tempor;
99
100         //Tempor = (unsigned char)(&RTCYear-2000); // Added for compatibility 2.8.2010
101         c = 0;
102         instr[c] = getchar();
103         putchar(instr[c]);
104         while((isdigit(instr[c]) == 1) && (c < 4))
105         {
106             c++;
107             instr[c] = getchar();
108             putchar(instr[c]);
109         }
110         // now that we have the number, null terminate the string
111         // (after checking for the Escape!)
112         if (instr[c] == 0x1B)
113             return 256;
114         c++;
115         instr[c] = '\0';
116         i = atoi(instr);
117         // Strange condition - required Tempor calculation 2.8.2010
118         if (i > 255) //&& (uc !=Tempor)
119             i = 255;
120         *uc = (unsigned char) i;
121         return 0;
122     }
123
124
125
126
127     // ********************************************************************************************************
128     //                                                 Check_UART0_Menu()
129     // Menu de ensayo para Driver UART0_Dr1 . c/.h tomado de PWRC2, limitado en funciones.
130     // Primera version 18.12.2017 - Es un menú para lazo,
131     //    utiliza MSG_option_local como parámetro, y modifica la variable de estado
132     //    global del Menu, llamada MSG_opt. Para verificar la lectura de un caracter, lee el valor
133     //    de rx_counter0 (modificado por ISR Rx presente en UART0_Dr1) y luego get_char() alternativo
134     //    del mismo driver.
135     //  Variables globales a definir:
136     //            unsigned char MSG_opt = 1;
137     //            unsigned int NBin_com = 0;
138     //            unsigned char Interval_ST = 0;
139     // ********************************************************************************************************
140
141     void Check_UART0_Menu( unsigned char MSG_option_local)
142     {
143     char c;
144     int i;
145     char a;
146     unsigned long c3;
```

```
147   unsigned int c2;
148   unsigned char UTemp, i_eep, pointer;
149   unsigned char TempYr;  // TempYr 00 to 99, since get_val will not compile for uints..
150   uint8_t temp_ubyte =0;     // For Vmin y otros.. 28-9-2010
151   float ftmp;
152   int FDeb;
153   uint8_t ch1;
154
155       if (MSG_option_local == 1)
156       {
157       printf("\n\r CL2bm1 - Inicializacion (Presione ? para comandos):\n\r");
158       putsf("\t?\t- Listar comandos.\r");
159       putsf("\t1\t- Menu de Setup Basico \r");
160       putsf("\t2\t- Menu de Setup Extendido \r");
161       putsf("\t3\t- Lanzar Prueba c/param. actuales (nuevo archivo)\r");
162       putsf("\t4\t- Re-Lanzar Prueba (archivo en curso)\r");
163       putsf("\t5\t- Configurar Canales \r");   // v9c Agregado 29.7.2014
164       // putsf("\t6\t- Otros Coeficientes - Manual \r");
165       #ifdef TEST_EV
166       printf("\n\r Archivo de Eventos: %s", SYS.ee_EV_FNAME);  // Added for testing..
167       #endif
168       // putsf("\t5\t- Volver a Modo Detenido\r");
169       MSG_opt = 2;   // Set to next Menu level.. (global)
170       }
171
172     if (MSG_option_local == 3)
173       {
174       printf("\n\r  CL2bm1 - Menu Setup Basico (Presione ? para comandos):\n\r");
175       putsf("\t?\t- Listar comandos.\r");
176       putsf("\tC\t- Setear Fecha y Hora\r");
177       putsf("\tc\t- Leer Fecha y Hora Actual.\r");
178       putsf("\tA\t- Fijar Tiempo Almacenamiento a 1minuto ..\r");
179       putsf("\tB\t- Fijar Tiempo Almacenamiento a  5minutos ..\r");
180       putsf("\tE\t- Fijar Tiempo Almacenamiento a 10minutos ..\r");
181       putsf("\tD\t- Fijar Tiempo Almacenamiento a 15minutos ..\r");
182       putsf("\tF\t- Mostrar Seteos Actuales de la Prueba ..\r");
183       putsf("\tN\t- Ingresar Nombre Estacion  (25c max)\r");
184       putsf("\tn\t- Leer Nombre Estacion\r");
185       putsf("\tI\t- Ingresar ID Estacion (25c max)\r");
186       putsf("\ti\t- Leer ID Estacion.\r"    );
187       putsf("\tW\t- Ingresar Limites DIR Excluidos (min, max °) \r");
188       putsf("\tw\t- Leer Limites DIR Excluidos (min, max °) \r");
189       putsf("\tV\t- Ingresar Nivel de Tension de Prueba (N/A/B) \r");
190       putsf("\tv\t- Leer Nivel de Tension de Prueba (N/A/B) \r");
191       putsf("\tM\t- Modificar Niveles de Tension Nominales \r");
192       putsf("\tm\t- Leer Niveles de Tension Nominales \r");
193       putsf("\tS\t- Espacio disponible en SD y N° Dias \r");
194       putsf("\ts\t- Listar Archivos en SD y tamaño \r");
195       putsf("\tp\t- Mostrar e ingresar Nro Bins p/Compl.\r");       // 23.2.2007 -- added.
196       putsf("\tx\t- Volver a Menu anterior \r\n");
197       MSG_opt = 4;   // Set to next Menu level..
198       }
199
200       if (MSG_option_local == 5)
201       {
202       printf("\n\r  CL2bm1 - Menu Setup Extendido (Presione ? para comandos):\n\r");
203       putsf("\tF\t- Ingresar Fabricante del Equipo (25c max)\r");
204       putsf("\tf\t- Leer Fabricante del Equipo.\r"    );
205       putsf("\tB\t- Ingresar Direccion Modbus de la Estacion (1-255).\r");
206       putsf("\tb\t- Leer Direccion Modbus de la Estacion.\r"    );
207       putsf("\tT\t- Mostrar Valores Setup Extendido.\r"    );
208       putsf("\tg\t- ON/OFF impresion Valores Float.\r");
209       putsf("\ty\t- Test de Watchdog (RESET!)\r");
210       putsf("\tx\t- Volver a Menu anterior \r");
211       MSG_opt = 6;   // Set to next Menu level..
212       }
213
214       while(rx_counter0)         // echo port 0..
215       {
216           c = getchar();
217           // putchar(c);
218           if(MSG_option_local == 2) {
219             switch (c)
220               {
221               case '?':
222                putsf("\t?\t- Listar comandos.\r");
223                putsf("\t1.\t- Menu de Setup Prueba \r");
```

```
224              putsf("\t2.\t- Menu Setup Extendido \r");
225              putsf("\t3.\t- Lanzar Prueba c/param. actuales (nuevo archivo)\r");
226              putsf("\t4.\t- Re-Lanzar Prueba (archivo en curso)\r");
227              putsf("\t5.\t- Configurar Canales \r");   // v9c Agregado 29.7.2014
228              //putsf("\t6.\t- Otros Coeficientes - Manual \r");
229              putsf("\tP.\t- Impresion METEO ON/OFF\r");
230              break;
231             case '1':           // Send to Setup Menu presentation..
232              MSG_opt = 3;
233              //EVT_LogEvent(EVT_ENTER_TSTSETUP);
234              break;
235             case '2':            //  Send to Hardware Menu..
236              MSG_opt = 5;
237              break;
238              // COM1 Testing-Meteo..
239              // Toggle debug printing flag
240             case 'P':   // Change PrintFlag...
241               if (Flag_Print_METEO) {
242                       Flag_Print_METEO = 0;
243                       printf("\n\r Impresion METEO - OFF");  // Testing..
244                       }
245               else {
246                       Flag_Print_METEO = 1;
247                       printf("\n\r Impresion METEO -  ON");  // Testing..
248                      }
249               break;
250             case '3':
251                    putsf("\n\r Lanza Prueba..");
252                    //Command_3_Function();
253                    break;
254             case '4':
255                    putsf("Reinicia Prueba...\n\r");
256                    break;
257             case '5':    // v9c - Cambiado a configuracion de canales..
258                     putsf("Configurar canales...\n\r");
259                     break;
260             default:
261                    printf("\n\rComando no reconocido...\n\r");
262              }  // End this switch..
263            }    // end if '2'
264
265          if(MSG_option_local == 4) {        // Enter Setup mode..
266           switch (c)
267              {
268              case '?':
269                  printf("\n\r  CL2bm1 Menu de Setup Prueba (Presione ? para comandos):\n\r")
    ;
270                  putsf("\t?\t- Listar comandos.\r");
271                  putsf("\tC\t- Setear Fecha y Hora\r");
272                  putsf("\tc\t- Leer Fecha y Hora Actual.\r");
273                  putsf("\tA\t- Fijar Tiempo Almacenamiento a 1minuto ..\r");
274                  putsf("\tB\t- Fijar Tiempo Almacenamiento a  5minutos ..\r");
275                  putsf("\tE\t- Fijar Tiempo Almacenamiento a 10minutos ..\r");
276                  putsf("\tD\t- Fijar Tiempo Almacenamiento a 15minutos ..\r");
277                  putsf("\tF\t- Mostrar Seteos Actuales de la Prueba ..\r");
278                  putsf("\tN\t- Ingresar Nombre Estacion  (25c max)\r");
279                  putsf("\tn\t- Leer Nombre Estacion\r");
280                  putsf("\tI\t- Ingresar ID Estacion (25c max)\r");
281                  putsf("\ti\t- Leer ID Estacion.\r"    );
282                  putsf("\tW\t- Ingresar Limites DIR Excluidos (min, max °) \r");
283                  putsf("\tw\t- Leer Limites DIR Excluidos (min, max °) \r");
284                  putsf("\tV\t- Ingresar Nivel de Tension de Prueba (N/A/B) \r");
285                  putsf("\tv\t- Leer Nivel de Tension de Prueba (N/A/B) \r");
286                  putsf("\tM\t- Modificar Niveles de Tension Nominales \r");
287                  putsf("\tm\t- Leer Niveles de Tension Nominales \r");
288                  putsf("\tS\t- Espacio disponible en SD y N° Dias \r");
289                  putsf("\ts\t- Listar Archivos en SD y tamaño \r");
290                  putsf("\tp\t- Mostrar e ingresar Nro Bins p/Compl.\r");       // 23.2.2007
    -- added.
291                  putsf("\tx\t- Volver a Menu anterior \r\n");
292                     break;
293
294              case 'p':  // Read and modify min. Bins for completion..
295                       printf("\n\r Nro Minimo Bins para Completar (10=def.)= %d..",10 ); //
    Moved to TSTAT 16.6.2012
296                       // monitor for cancel!
```

```
297                     printf("\n\r Ingresar Nvo. Valor[5-250][ENTER o ESC]:\n\r");
298                     i = get_val(&temp_ubyte);
299                     if (i != 256) {
300                           if((temp_ubyte >4) && (temp_ubyte<251)){
301                                 NBin_com = (unsigned int)temp_ubyte;
302                                 printf("\n\r Nvo Valor NBins_min = %d..", NBin_com );
303                                 }
304                           else {printf("\n\r Fuera de Rango!");}
305                     }
306                     break;
307
308             case 'c':  // Read Real Time Settings..
309                 printf("\n\r Fecha/Hora interna:");
310                 rtc_get_timeNdate(&RTCHour, &RTCMin, &RTCSec, &RTCDay, &RTCMonth, &RTCYear)
    ;
311                 printf("%02d/%02d/%04d-%02d:%02d:%02d ", RTCDay,RTCMonth,RTCYear,RTCHour,
    RTCMin,RTCSec);
312                 break;
313
314             case 'C':  // Modify Real time settings..
315                 printf("\n\r Ingresar Fecha y hora como:   dd/mm/aa hh:mm:ss[ENTER]\n\r");
316                 // monitor for cancel! - get_val only works for UBYTES, so for Yr use TmpYr
    .. 2.8.2010
317                 // Then convert as RTCYear = 2000 + TempYr;
318                 i = get_val(&RTCDay);
319                 if (i != 256)
320                     i = get_val(&RTCMonth);
321                 if (i != 256)
322                     i = get_val(&TempYr);
323                 if (i != 256)
324                     i = get_val(&RTCHour);
325                 if (i != 256)
326                     i = get_val(&RTCMin);
327                 if (i != 256)
328                     i = get_val(&RTCSec);
329                 if (i != 256)
330                 {
331                 RTCYear = 2000 + TempYr;  // Convert byte to  uint. 2.8.2010
332                 if(rtc_set_time(RTCHour,RTCMin,RTCSec)) {printf("Error in RTC_set_time..!!\
    n\r");}
333                 else {printf("OK Time.\n\r");}
334                 delay_ms(100);          // Added 10.3.2010 - Change to 1/2 value with ints.
    .30-9-10
335                 if(rtc_set_date(RTCDay,RTCMonth,RTCYear)){ printf("Error in RTC_set_date..!
    !\n\r");}
336                 else { printf("OK Date.\n\r");}
337                 // rtc_set_time(hour,minute,sec);
338                 // rtc_set_date(date,month,year);
339                 }
340                 break;
341              // Interval Tiime changing  -- 28-9.10
342              // #define   INT_10MIN  1
343              // #define   INT_30MIN  2
344              // #define   INT_5MIN   3
345              // #define   INT_1MIN   4
346             case 'A':
347                 #asm("cli")
348                 Interval_ST = INT_1MIN;
349                 #asm("sei")
350                 strcpyf(INTERV_TIME, "1 Minuto");
351                 putsf("\n\rFijado Tiempo Almacenamiento a 1 minuto ..\r");
352                 break;
353             case 'B':
354                 #asm("cli")
355                 Interval_ST = INT_5MIN;
356                 #asm("sei")
357                 strcpyf(INTERV_TIME, "5 Minutos");
358                 putsf("\n\rFijado Tiempo Almacenamiento a 5 minutos ..\r");
359                 break;
360
361             case 'E':
362                 #asm("cli")
363                 Interval_ST = INT_10MIN;
364                 #asm("sei")
365                 strcpyf(INTERV_TIME, "10 Minutos");
366                 putsf("\n\rFijado Tiempo Almacenamiento a 10 minutos ..\r");
```

```
367                 break;
368             case 'D':
369                 #asm("cli")
370                 Interval_ST = INT_30MIN;
371                 #asm("sei")
372                 strcpyf(INTERV_TIME,"30 Minutos");
373                 putsf("\n\rFijado Tiempo Almacenamiento a 30 minutos ..\r");
374                 break;
375 
376             case 'F':   // Show all parameters in normal setup..
377                 printf("\n\r Parametros de la Prueba:");
378                 #define TEST_1ST_PART
379                 #ifdef TEST_1ST_PART
380                 #asm("cli")
381                 UTemp = Interval_ST;
382                 #asm("sei")
383                 if(UTemp == INT_1MIN){
384                          putsf("\n\rTiempo Alm. = 1 minuto ..\n\r");
385                          }
386                 else if (UTemp == INT_5MIN){
387                          putsf("\n\rTiempo Alm. = 5 minutos ..\n\r");
388                          }
389                 else if (UTemp == INT_10MIN){
390                          putsf("\n\rTiempo Alm. = 10 minutos ..\n\r");
391                          }
392                 else if (UTemp == INT_30MIN){
393                          putsf("\n\rTiempo Alm. = 15 minutos ..\n\r");
394                          }
395                 else{
396                          printf("\n\rLeido %d: Incorrecto..", UTemp);
397                        }
398                 printf("\n\r Archivo a usar: Datos.csv");   // f_namestr..
399                 printf("\n\r Nombre Estacion %s ", Station_name);
400                 printf("\n\r ID Prueba ID01 ");
401                 // printf("\n\r Intervalo Alm.: %s \n\r", INTERV_TIME);
402                 #endif
403                 #define TEST_2ND_PART
404                 #ifdef TEST_2ND_PART
405                 //if (PTST.ee_VTestLVL == 2) sprintf(tempstr, "V.BAJO, %2.2f, [V]",PTST.
    ee_VBLow_TH);
406                 //else if (PTST.ee_VTestLVL == 1) sprintf(tempstr, "V.ALTO, %2.2f, [V]",
    PTST.ee_VBHi_TH);
407                 //else sprintf(tempstr, "NOMINAL, %2.2f, [V]",PTST.ee_VBNom_TH);
408                 //printf("\n\r Nivel Umbral medio de Tensión de la prueba:, %s", tempstr);
409                 //printf("\n\r Valor VTH en RAM:, %2.2f,[V]",PTST.ee_VBNom_TH);   // Used
    to be VBTH 15-6-2012
410                 //printf("\n\r Valor Porcentaje K (5%%nom): %d",PTST.ee_VBNom_percentage);
411                 // 5.11.06 Following computed within startup or initialization..
412                 // float  VBTH_PlusK              // Will store = ee_VB + 5%
413                 // float  VBTH_MinusK             // Will store = ee_VB - 5%
414                 // sprintf(tempstr, "+K%%, %2.2f, -K%%, %2.2f [V]",VBTH_PlusK,VBTH_MinusK);
415                     // 15.6.2012 Printout
416                 // VBTH_PlusK = VBTH (1+ VBNOM_PERCENTAGE/100)
417                 // VBTH_MinusK = VBTH (1- VBNOM_PERCENTAGE/100)
418                 // 15.6.2012 Assign accordingly..
419                 printf("\n\r Nivel de Tension Prueba: NOM");
420                 printf("\n\r Umbrales promedio 1min de Tensión:");
421                 printf("\n\r V_Nivel +K: %2.2f [V]", VBTH_PlusK);
422                 printf("\n\r V_Nivel -K: 24.5 [V]");
423                 #endif
424                 // Horas REq..
425                 printf("\n\r Horas de Prueba requeridas:, 12");
426                 putsf("\n\r>");
427                 break;
428 
429             case 'N':
430                 putsf("\n\rIngresar Nombre Estacion. (Max.25c + ENTER, ESC sale)\n\r");
431                 // p_input_str = &input_str[0];
432                 if (get_string() == 0){
433                     //if(scanf("%s,\n", input_str) == 1){
434                     //       if(strlen(input_str) > 24) input_str[24]=0;   // Null
    terminate if too big..
435                   printf("\n\r Ingresado OK: %s \n\r", instr1);
436                   for(i_eep=0; i_eep<25;i_eep++){
437                         Station_name[i_eep] =  instr1[i_eep];   // EEPROM value read..
438                         if(instr1[i_eep]==0) break;
```

```
439                         }  // No strcpy for eeprom...
440                       putsf(" .. y copiado a EEPROM..\n\r");
441                   }     // ind if scanf = 1..
442                 else{
443                 putsf("\n\rError al ingresar nombre!\n\r");
444                 }
445                 break;
446
447                 // elim 17.2.2006
448             case 'n':
449                 printf("\n\r (copia en RAM): %s \n\r", Station_name);
450                 break;
451
452             case 'I':            // read
453                 putsf("\n\rIngresar ID Estacion. (Max.25c + ENTER, ESC sale)\n\r");
454                 printf("\n\r Ingresado OK: TST01 \n\r");
455                 break;
456             case 'i':            // send the existing ID
457                 printf("\n\r ID de Estacion TST01 \n\r");
458                 break;
459
460             case 'W':  //Ingresar Wexc max,min
461                 printf("\n\r  DIRECCIONES Excluidas de la prueba..");
462                 break;
463
464             case 'w':            // show the existing excl. directions
465                 printf("\n\r  Read DIRECCIONES Excluidas de la prueba..");
466                 break;
467
468             case 'V':
469                 printf("\n\r Cambiar Nivel de Tension de la Prueba:,");
470                 break;
471             case 'v':
472                 printf("\n\r Nivel Actual de Tension Prueba:");
473                 printf("\n\r V_Nivel +K: %2.2f [V]", VBTH_PlusK);
474                 printf("\n\r V_Nivel -K: 24.5 [V]");
475                 break;
476
477             case 's':
478                 printf("\n\r Archivos en Tarjeta SD:");
479                 break;
480
481             case 'x':
482                 MSG_opt = 1; // Back to main menu..
483                 break;
484             default:
485                  printf("\n\rOpcion no reconocida..\n\r");
486                 } // End switch..
487           }       // end if
488
489         if(MSG_option_local == 6) {        // Enter Hardware mode..
490           switch (c)
491             {
492             case '?':
493                 printf("\n\r  CL2bm1 - Setup Extendido (Presione ? para comandos):\n\r");
494                 putsf("\tF\t- Ingresar Fabricante del Equipo (25c max)\r");
495                 putsf("\tf\t- Leer Fabricante del Equipo.\r"    );
496                 putsf("\tB\t- Ingresar Direccion Modbus de la Estacion (1-255).\r");
497                 putsf("\tb\t- Leer Direccion Modbus de la Estacion.\r"    );
498                 putsf("\tT\t- Mostrar Valores Setup Extendido.\r"    );
499                 putsf("\tg\t- ON/OFF impresion Valores Float.\r");
500                 putsf("\ty\t- Test de Watchdog (RESET!)\r");
501                 putsf("\tx\t- Volver a Menu anterior \r");
502                 break;
503
504             case 'b':   // Read Modbus address
505                 printf("\n\r Direccion Modbus de la Estacion 64 \n\r" );
506                 break;
507
508             case 'B':   // New Modbus address
509                 printf("\n\r Ingresar Nueva Direccion Modbus: 1-127 [ENTER]\n\r");
510                 break;
511
512             case 'F':
513                 putsf("\n\rIngresar Fabricante Aerogenerador: (Max.25c + ENTER, ESC sale)\n
    \r");
```

```
514                 break;
515
516                 // elim 17.2.2006
517             case 'f':
518                 printf("\n\r Fabricante: EOLUX \n\r");
519                 break;
520
521
522             case 'g':
523                 printf("\n\r AD Values Print ON");  // Testing..
524                 break;
525
526             //  Show all selections v16.6.2012 - Added Other Coefs.29-10-2012
527             case 'T':
528                 printf("\n\r Valores Setup Extendido:\n\r");
529                 break;
530
531             case 'y': // 17.3.2007 - Test WDOG
532                 printf("\n\r WDOG_test..");
533                 WD_ON_Flag = 1;                // Set to 1 inhibits wdog petting.. 17.3.
    2007 v12e
534                 break;
535
536             case 'x':
537                 MSG_opt = 1; // Back to main menu..
538                 break;
539             default:
540                 putchar(c);            // echo local until UDP echo is enabled by first
    reception
541             }  // End switch..
542         }      // End if.....
543     }      // End while..
544
545 }           // End function...
546
```