```
/*****************************************************
CL2bm1 Test Menus for UART1_Dr1 R.Oliva
L&R Ingeniería 2010-2018
Project : Test_UART1_Dr1  with AtMega1284P 14.76MHz
File:     Tst_Uart1Menu_main.c
Version : 1  (14.76MHz Test system)
Date    : 30.01.2018
Author  : Rafael Oliva
Company : L&R Ingenieria


Chip type               : ATmega1284P
Program type            : Application
AVR Core Clock frequency: 14,745600 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 4096
**
** Version 30.01.2018
** Call Simplified ExternalTimer Interrupt, pin_change_isr3()
** based on 0.5Hz signal 30.01.2018
** External_Timer_Init();
*****************************************************/


#include <mega1284p.h>
#include <delay.h>
#include <string.h>
#include <stdio.h>
#include <stdarg.h>
#include <stdlib.h>
#include <string.h>
#include <io.h>


// Added for LCD- 19-12-2017
// #include "..\..\LCD\LCD4x20(2010)\inc\lcd_cl2_3.h"

// Added for TWI - 30.01.2018
#include "..\..\TWI\TWI12A\inc\twi-cl2_12a.h"

// Testing UART1_DR1 - 30.01.2018
#include "..\inc\Uart1_dr1.h"

// Added for testing -  Initialize_CL2_Simple()
#include "Tst_Uart1Menu_main.h"

// Added for testing Menus
#include "Basic(U1)_Menu.h"


// PB.0 down --> initialize RTC 19.11.17
#define KBD_LEFT_ARROW          PINB.0

/* OLD definitions..USART Baud rate */
#define BAUD_RATE 19200
#define BAUD_INIT (_MCU_CLOCK_FREQUENCY_/(BAUD_RATE*16L)-1)


// Made Global 18.12.17 to control WDOG..
bit WD_ON_Flag = 0;    // Default pets the dog
// ***********************************************
// ** For EXT_IRQ (INT3) with 1.0Hz from CKOUT
// ** Added 19.12.2017
// ***********************************************
unsigned char sm2_1sec = 0;        // 12.2017 - 1Second @0.5sec interval count 1-3
unsigned char Flag_Sec_Change = 0; // For second change signalling 19.12.17
```

```
int8_t rval = 0;                    // Debug CKOUT
int8_t rval1 = 0;

/*****************************************************************************
**
**  Initialize_CL2_simple() Depends on processor used and board.
**  Simple Version TAKES AWAY Interrupts /  modified 19.11.2017
**  For LCD and RTC testing..
**  Current for Mega1284 uses AtMega1284P and CL2bm1
**  COM0 at 19200,N,8,1 - traditional
**  COM1 not used
**  Port C and one pin PortD used for LCD.
**  Timer0 not used,
**  Timer1 not used
**  ADC not set
**
*****************************************************************************/

void Initialize_CL2_simple(void)
{
  CLI();      /* disable all interrupts */


 // Input/Output Ports initialization

  // Port A initialization
  // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
  // State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
  PORTA=0x00;
  DDRA=0x00;

  // Port B initialization
  // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
  // State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
  PORTB=0x00;
  DDRB=0x00;

  // Port C initialization
  // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
  // State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
  // Debug: set Bit C.7 to 1 3.2.09
  // v0.3 6-2-09 Set PC.2-7 as outputs - For LCD..
  PORTC=0x00;
  DDRC=0xFC;

  // Port D initialization
  // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
  // State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
  // v0.3 6-2-09 Set Pd.7 as output - For LCD 0x80
  // V1284p-7 Set Pd.5./.6 to outputs too for OLED and BacklightLCD 24.7.10 0xE0
  PORTD=0x00;
  //DDRD=0x80;
  DDRD=0xE0;

// Following from SDCard4.c initialization..19.11.2017

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0A output: Disconnected
// OC0B output: Disconnected
TCCR0A=(0<<COM0A1) | (0<<COM0A0) | (0<<COM0B1) | (0<<COM0B0) | (0<<WGM01) | (0<<WGM00);
TCCR0B=(0<<WGM02) | (0<<CS02) | (0<<CS01) | (0<<CS00);
TCNT0=0x00;
OCR0A=0x00;
OCR0B=0x00;
```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Disconnected
// OC1B output: Disconnected
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) | (0<<WGM10);
TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2A output: Disconnected
// OC2B output: Disconnected
ASSR=(0<<EXCLK) | (0<<AS2);
TCCR2A=(0<<COM2A1) | (0<<COM2A0) | (0<<COM2B1) | (0<<COM2B0) | (0<<WGM21) | (0<<WGM20);
TCCR2B=(0<<WGM22) | (0<<CS22) | (0<<CS21) | (0<<CS20);
TCNT2=0x00;
OCR2A=0x00;
OCR2B=0x00;

// Timer/Counter 3 initialization
// Clock source: System Clock
// Clock value: Timer3 Stopped
// Mode: Normal top=0xFFFF
// OC3A output: Disconnected
// OC3B output: Disconnected
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer3 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR3A=(0<<COM3A1) | (0<<COM3A0) | (0<<COM3B1) | (0<<COM3B0) | (0<<WGM31) | (0<<WGM30);
TCCR3B=(0<<ICNC3) | (0<<ICES3) | (0<<WGM33) | (0<<WGM32) | (0<<CS32) | (0<<CS31) | (0<<CS30);
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x00;
OCR3AL=0x00;
OCR3BH=0x00;
OCR3BL=0x00;

// Timer/Counter 0 Interrupt(s) initialization
TIMSK0=(0<<OCIE0B) | (0<<OCIE0A) | (0<<TOIE0);

// Timer/Counter 1 Interrupt(s) initialization
TIMSK1=(0<<ICIE1) | (0<<OCIE1B) | (0<<OCIE1A) | (0<<TOIE1);

// Timer/Counter 2 Interrupt(s) initialization
```

```
TIMSK2=(0<<OCIE2B) | (0<<OCIE2A) | (0<<TOIE2);

// Timer/Counter 3 Interrupt(s) initialization
TIMSK3=(0<<ICIE3) | (0<<OCIE3B) | (0<<OCIE3A) | (0<<TOIE3);

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
// Interrupt on any change on pins PCINT0-7: Off
// Interrupt on any change on pins PCINT8-15: Off
// Interrupt on any change on pins PCINT16-23: Off
// Interrupt on any change on pins PCINT24-31: Off
EICRA=(0<<ISC21) | (0<<ISC20) | (0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);
EIMSK=(0<<INT2) | (0<<INT1) | (0<<INT0);
PCICR=(0<<PCIE3) | (0<<PCIE2) | (0<<PCIE1) | (0<<PCIE0);

// USART0 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
// USART0 Transmitter: On
// USART0 Mode: Asynchronous
// USART0 Baud Rate: 19200
UCSR0A=(0<<RXC0) | (0<<TXC0) | (0<<UDRE0) | (0<<FE0) | (0<<DOR0) | (0<<UPE0) | (0<<U2X0) | (0<<MPCM0);
UCSR0B=(0<<RXCIE0) | (0<<TXCIE0) | (0<<UDRIE0) | (1<<RXEN0) | (1<<TXEN0) | (0<<UCSZ02) | (0<<RXB80)
| (0<<TXB80);
UCSR0C=(0<<UMSEL01) | (0<<UMSEL00) | (0<<UPM01) | (0<<UPM00) | (0<<USBS0) | (1<<UCSZ01) |
(1<<UCSZ00) | (0<<UCPOL0);
UBRR0H=0x00;
UBRR0L=0x2F;

/* initialize the USART0 TX, 8N1, Baud rate: 19200 */
UCSR0A=0;
UCSR0B=1<<TXEN0;
UCSR0C=(1<<UCSZ01)|(1<<UCSZ00);
UBRR0H=BAUD_INIT>>8;
UBRR0L=BAUD_INIT&0xFF;

// USART1 initialization
// USART1 disabled
UCSR1B=(0<<RXCIE1) | (0<<TXCIE1) | (0<<UDRIE1) | (0<<RXEN1) | (0<<TXEN1) | (0<<UCSZ12) | (0<<RXB81)
| (0<<TXB81);

// Analog Comparator initialization
// Analog Comparator: Off
// The Analog Comparator's positive input is
// connected to the AIN0 pin
// The Analog Comparator's negative input is
// connected to the AIN1 pin
ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);
ADCSRB=(0<<ACME);
// Digital input buffer on AIN0: On
// Digital input buffer on AIN1: On
DIDR1=(0<<AIN0D) | (0<<AIN1D);

// ADC initialization
// ADC disabled
ADCSRA=(0<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) | (0<<ADPS1) |
(0<<ADPS0);


// This kept from old Initialize_CL2() function..
  // ******************************
  // Two Wire Bus initialization
  // Bit Rate:
  // 17.2.09 Changed TWBR to 0x0c for 184kHz..
  // if XTAL = 7.3728e6 means:
```

```
  // Bit Rate: 184.320 kHz
  // ** NOTE: For Bit Rate: 115.200 kHz
  // ** then CV-Wiz selectede TWBR=0x18;
  // TWBR=0x0C;
  // Two Wire Bus Slave Address: 0h
  // General Call Recognition: Off
  // TWAR=0x00;
  // Generate Acknowledge Pulse: Off
  // TWI Interrupt:  OFF
  // TWCR=0x05 - would be for TWI IRQ ON;
  // TWCR=0x04; // Same as v32..
  // TWSR=0x00;

  // Two Wire Bus initialization for XTal 14.756MHz 22.3.2012
  // Bit Rate: 184.320 kHz
  TWBR=0x20;
  // Two Wire Bus Slave Address: 0h
  // General Call Recognition: Off
  TWAR=0x00;
  // Generate Acknowledge Pulse: Off
  // TWI Interrupt: Off
  TWCR=0x04;
  TWSR=0x00;


  // Watchdog Timer initialization - CV-Wiz (07.2010)
  // Watchdog Timer Prescaler: OSC/2k
  // Watchdog Timer interrupt: Off
  // Re-eanble 30-9-2010
  // b4 WDCE, b3 WDE
  // b5 b2 b1 b0 WDP3-WDP0 Prescaler
  #pragma optsize-
  #asm("wdr")
  WDTCSR=0x18;    // 0001 1000
  //WDTCSR=0x08;    // 0000 1000  -- Prescaler in 16ms
  WDTCSR=0x28;    // 0010 1000  -- Prescaler in 1000 = 4sec
  #ifdef _OPTIMIZE_SIZE_
  #pragma optsize+
  #endif

  // Global enable interrupts
  // SEI();      /* re-enable interrupts 19.11.2017 */

}


/*****************************************************************************
**
**  External_Timer_Init(void)  - 19.12.2017
**  Uses CKOUT as source on PD.4 pin, CL2bm1
**  Used together with pin_change_isr3
*****************************************************************************
**/

void External_Timer_Init(void){        // 19.12.2017 CKOUT driven Interrupt
  // External Interrupt(s) initialization
  // INT0: Off
  // INT1: Off
  // INT2: Off
  // Interrupt on any change on pins PCINT0-7: Off
  // Interrupt on any change on pins PCINT8-15: Off
  // Interrupt on any change on pins PCINT16-23: Off
  // Interrupt on any change on pins PCINT24-31: On
  EICRA=0x00;
  EIMSK=0x00;
  PCMSK3=0x10;
  PCICR=0x08;
```

```
  PCIFR=0x08;
}

/*****************************************************************************
**
** Simplified External PD.4 interrupt ISR (level change, OSC_OUT set to 1Hz)
** Called on each level change or 2Hz (0.5sec) intervals..
** 24.10.2012 Pin change 24-31 interrupt service routine - fist phase add
**     OLED change on level transition..
** Revision - 19.12.2017
**
*****************************************************************************/

 interrupt [PC_INT3] void pin_change_isr3(void)
 {
    // New values for 2Hz transition calls.. 24.10.2012
    // Calculations see Cuaderno#2, SoftwPWRC2 Additions 10.2012
    // 24.10.2012 Counts transitions 1 to 3
    #define SECOND2_TOUT 3


    // First step - toggle LCD every 0.5 Hz
    if(OLED) OLED=0; // Add a 1 sec life LED.. 24-10-2012
    else OLED= 1;

    if(WD_ON_Flag == 0){ #asm("wdr")}

    sm2_1sec ++;

    if(sm2_1sec == SECOND2_TOUT) {
        sm2_1sec = 1;
        Flag_Sec_Change = 1;//
     } // All solved in 1sec- interval..

   }




void main(void)
{
// Declare your local variables here
// 19.11.2017 for Display & COM showing..
char s[25];
// 19.11.2017 for Display & COM showing..
int8_t RTC_result = 0;  // Defined for RTC
// Local for new functions (Old RTC funtions use int8_t)
uint8_t uRTCHour = 0;
uint8_t uRTCMin = 0;
uint8_t uRTCSec = 0;
uint8_t uRTCDay = 0;
uint8_t uRTCMonth = 0;
uint16_t uRTCYear = 0;


// *****************************************************
// * Change to same notation in CL2_Basis4
// * 19-11-2017
  #pragma optsize-
  CLKPR=0x80;
  CLKPR=0x00;
  #ifdef _OPTIMIZE_SIZE_
  #pragma optsize+
  #endif
```

```
// 18.12.207 Most initialization sent to Initialize_CL2() funtion
Initialize_CL2_simple();

// Call special COM1 init
USART1_Init();

// Call Simplified ExternalTimer Interrupt, pin_change_isr3()
// based on 0.5Hz signal 19.12.2017
External_Timer_Init();


/* globally enable interrupts */
#asm("sei")

// *****************************************************************************
// 16-5-2017 - Added - for life detection..
// BACK_LIGHT = 1;
// OLED = 1;
// delay_ms(500); // 0.5sec
// OLED = 0;

// *****************************************************************************
// 19-11-2017 - Added For LCD starting..
// *****************************************************************************
sprintf(s,"Test UART1_Menu..\n\r");
puts1(s);
//init_display();
//clear_display();
//set_LCD_cur(0,0);
//disp_cstr("TestUARTMenu2017"); // Modificación de etiqueta 19.11.17
//set_LCD_cur(1,0);
//disp_cstr("*             *"); // Cambio
delay_ms(1000);

// *******************************************************
// RTC Initialize   -19.11.2017
// *******************************************************
// #define OPTION_CKOUT_01HZ   0x83
// set Clockout
// set_LCD_cur(0,0);
  rval1 = 4;
  rval = rtc_set_ckoutfreq(rval1);
  sprintf(s, "RTC Ckout result: %d",rval);
  puts1(s);
  puts1("CKout_1Hz       ");
  delay_ms(2000);

puts1("RTC inicializando..\n\r");
//  RTC Read..
RTC_result = rtc_get_timeNdate(&RTCHour, &RTCMin, &RTCSec, &RTCDay, &RTCMonth, &RTCYear);

   // clock battery dead.. reset to something sane
   // Set to error 16.2.09
   // Back to TWI_SUCCESS 17.2.09
   // Force initialize.. 4.1.2012
   // RTC_result = 187;
   if((RTC_result != TWI_SUCCESS)|(KBD_LEFT_ARROW == 0))
    {
        puts1("RTC-Now initialized!\n\r");
        // set_LCD_cur(0,0);
        puts1("RTes RTC Init.. \n\r");
            // Probable requirement of Variables, not constants in rtc_set_time/date..
            // Verify 19.11.2017
            RTCHour = 22;
            RTCMin  = 41;
            RTCSec  = 0;
            rtc_set_time(RTCHour,RTCMin,RTCSec);
```

```
            delay_ms(1000);
            RTCDay   = 19;
            RTCMonth = 11;
            RTCYear  = 2017;
            rtc_set_date(RTCDay,RTCMonth,RTCYear);
            delay_ms(1000);
    }
    else
       {
       puts1("RTC-OK!\n\r");
       //set_LCD_cur(0,0);
       //disp_cstr("RTC OK read..   ");
       }

  rtc_get_timeNdate(&RTCHour, &RTCMin, &RTCSec, &RTCDay, &RTCMonth, &RTCYear);
  // Added  Display RTC on LCD
  sprintf(s,"%02d/%02d/%04d-%02d:%02d:%02d ", RTCDay,RTCMonth,RTCYear,RTCHour,RTCMin,RTCSec);
  //set_LCD_cur(1,0);
  puts1(s);

  delay_ms(2000);
  // Test new functions a) Time
  RTC_result = (int8_t)(rtc_get_time(&uRTCHour, &uRTCMin, &uRTCSec));
  sprintf(s,"RTC new function ReadTime Access: %d",   RTC_result);
  puts1(s);
  delay_ms(2000);
  sprintf(s,"Time:%02d:%02d:%02d   ",uRTCHour,uRTCMin,uRTCSec);
  // set_LCD_cur(1,0);
  // disp_str(s);
  puts1(s);
  delay_ms(2000);

  rtc_get_date(&uRTCDay, &uRTCMonth, &uRTCYear);
  sprintf(s,"Date:%02d/%02d/%04d ", uRTCDay,uRTCMonth,uRTCYear);
  //set_LCD_cur(1,0);
  //disp_str(s);
  puts1(s);
  delay_ms(2000);




  puts1("Testing UART1 Menu Driver 1:\r\n");

  // set_LCD_cur(0,0);
  // disp_cstr("Menu Test..     ");


delay_ms(2000); //  2sec
// ***************************************************************************

while (1)
      {
      // Place your code here
      // 16-5-2017 - Added - for Incorrect Mounting..
      // if(WD_ON_Flag == 0){ #asm("wdr")} sent to ISR 19.12.2017
      // OLED = 1;    // sent to ISR..19.12.2017
      // delay_ms(500); //  0.5 blinking
      // OLED = 0;
      delay_ms(50);    //  Minimal delay 19.12.17
      Check_UART1_Menu(MSG_opt);

      #ifdef SEND_OUT_TIME
       if (Flag_Sec_Change == 1){
         Flag_Sec_Change = 0;
         RTC_result = (int8_t)(rtc_get_time(&uRTCHour, &uRTCMin, &uRTCSec));
         sprintf(s,"Time:%02d:%02d:%02d \n\r ",uRTCHour,uRTCMin,uRTCSec);
```

```
          //set_LCD_cur(1,0);
          puts1(s);
        }
      #endif


      } // en while(1)
}
```