```
;
; MIOS Custom LCD Driver Example for character LCDs
;
; NOTE: this is just a template for LCDs which are different to
;       HD44780 (that is natively supported by MIOS -> LCD type #0)
;       Note also that this driver only supports a single CLCD
;       and no free definable enable (E) line
;
; ==========================================================================
;
;  Copyright (C) 2003  Thorsten Klose (tk@midibox.org)
;  Licensed for personal non-commercial use only.
;  All other rights reserved.
;
; ==========================================================================

;; ---------------------------------------------------------------------------
;;  Following system variables are given by MIOS and can be directly
;;  accessed by the driver. The addresses are defined in mios.h and
;;  should not be changed
;;
;;  MIOS_GLCD_BUFFER         a 8 byte buffer for data transfers
;;  MIOS_LCD_OPTION1         contains the first LCD option given by MIOS_LCD_TypeSet
;;  MIOS_LCD_OPTION2         contains the second LCD option given by MIOS_LCD_TypeSet
;;  MIOS_LCD_CURSOR_POS      the current cursor pos of characters (GLCD: multiplied by width)
;;  MIOS_GLCD_GCURSOR_X      for GLCDs: the current X position of graphical cursor
;;  MIOS_GLCD_GCURSOR_Y      for GLCDs: the current Y position of graphical cursor
;;  MIOS_GLCD_FONT_WIDTH     for GLCDs: the fontwidth given by MIOS_GLCD_FontInit
;;  MIOS_GLCD_FONT_HEIGHT    for GLCDs: the fontheight given by MIOS_GLCD_FontInit
;;  MIOS_GLCD_FONT_X0        for GLCDs: the first byte within a char entry
;;  MIOS_GLCD_FONT_OFFSET    for GLCDs: the byte offset between the characters
;;  MIOS_GLCD_FONT_PTRL      for GLCDs: pointer to the character table, low-byte
;;  MIOS_GLCD_FONT_PTRH      for GLCDs: pointer to the character table, high-byte
;;  MIOS_LCD_TIMEOUT0        can be used for timeout loops
;;  MIOS_LCD_TIMEOUT1        can be used for timeout loops
;;  MIOS_GLCD_TMP1           can be used as temporary buffer
;;  MIOS_GLCD_TMP2           can be used as temporary buffer
;;  MIOS_GLCD_TMP3           can be used as temporary buffer
;;  MIOS_GLCD_TMP4           can be used as temporary buffer
;;  MIOS_LCD_Y0_OFFSET       Y0 offset of LCD
```

```
;;  MIOS_LCD_Y1_OFFSET       Y1 offset of LCD
;;  MIOS_LCD_Y2_OFFSET       Y2 offset of LCD
;;  MIOS_LCD_Y3_OFFSET       Y3 offset of LCD
;;  MIOS_LCD_CURSOR_POS_REAL unmapped cursor position which has been set
with MIOS_LCD_CursorSet
;;
;;  Note: the addresses are located in an upper bank and therefore have to
;;        be accessed with the BANKED flag. Also the BSR has to be justified
;;        before using the registers
;;  Example:
;;  SET_BSR MIOS_LCD_OPTION1    ; sets BSR to the bank where MIOS_LCD_*
;;                  ; has been located. You don't need to
;;                  ; change the BSR for the other LCD registers
;;      movf    MIOS_LCD_OPTION1, W, BANKED ; get LCD option #1
;;
;;  Important: to allow a proper interaction with MIOS applications, you are
;;  only allowed to modify MIOS_PARAMETER[123], the mutliplication registers
;;  and FSR1. You are not allowed to change TMP[1-5] or FSR0
;;  if you need some temporary registers, use the given addresses above or
;;  locate them to addresses which are not used by the application
;; ----------------------------------------------------------------------
-

;; Pins of LC-Display
USER_LCD_LAT_D      EQU    LATB    ; Pin B.7-0
USER_LCD_PORT_D EQU     PORTB
USER_LCD_TRIS_D EQU     TRISB

USER_LCD_LAT_RW EQU     LATD
USER_LCD_PIN_RW EQU     6               ; Pin D.6
USER_LCD_LAT_RS EQU     LATD
USER_LCD_PIN_RS EQU     5               ; Pin D.5

USER_LCD_LAT_E  EQU     LATD        ; Pin D.7
USER_LCD_PIN_E  EQU     7

;; new names for CLCD registers
USER_LCD_STATUS     EQU    MIOS_GLCD_TMP1
USER_LCD_SC_CTR     EQU    MIOS_GLCD_TMP3

#define USER_LCD_STATUS_LCD0_DISABLED   0    ; bit0:    if set, first LCD
disabled
#define USER_LCD_STATUS_LCD1_DISABLED   1    ; bit1:    if set, second LCD
disabled -- not provided by this driver!
#define USER_LCD_STATUS_CUR_DISABLED    2    ; bit2:    if set, currently
selected LCD disabled
#define USER_LCD_STATUS_CUR_LCD     3    ; bit3: if cleared: current LCD is
first LCD, else second LCD

;; ----------------------------------------------------------------------
-
```

```
;;  This function is called by MIOS when the custom LCD should be
initialized
;;  In:  MIOS_LCD_OPTION1 - contains the first LCD option given by
MIOS_LCD_TypeSet
;;       MIOS_LCD_OPTION2 - contains the second LCD option given by
MIOS_LCD_TypeSet
;;  Out: -
;; ------------------------------------------------------------------------
-
USER_LCD_Init
    ;; notify that no graphical LCD is connected
    bcf    MIOS_BOX_CFG0, MIOS_BOX_CFG0_USE_GLCD

    movlw    0xf9        ; set only TRISE[2:1] as output
    andwf    TRISE, F

    ; (Initialization of Ports: done in Init_Ports)
    SET_BSR    MIOS_LCD_TIMEOUT1
    clrf    USER_LCD_STATUS, BANKED

    movlw    100            ; 100 ms delay
    call    MIOS_Delay

        bcf     USER_LCD_LAT_RW, USER_LCD_PIN_RW    ; LCD_WRITE
        bcf     USER_LCD_LAT_RS, USER_LCD_PIN_RS    ; USER_LCD_PIN_RS_0

    ;; initialize LCD
    movlw    0x38
    movwf    USER_LCD_LAT_D
    rcall    USER_LCD_Strobe_Set
    rcall    USER_LCD_Strobe_Clr
    movlw    50            ; 50 ms delay
    call    MIOS_Delay
    rcall    USER_LCD_Strobe_Set
    rcall    USER_LCD_Strobe_Clr
    movlw    50            ; 50 ms delay
    call    MIOS_Delay
    rcall    USER_LCD_Strobe_Set
    rcall    USER_LCD_Strobe_Clr

    movlw    0x08            ; Display Off
    rcall    USER_LCD_Cmd
    movlw    0x0c            ; Display On
    rcall    USER_LCD_Cmd
    movlw    0x06            ; Entry Mode
    rcall    USER_LCD_Cmd
    movlw    0x01            ; Clear Display
    call    USER_LCD_Cmd
    bcf    MIOS_LCD_TIMEOUT1, 7, BANKED    ; everything ok, make sure that
LCD_TIMEOUT, bit 7 is cleared
    movlw    0x38            ; without these lines the LCD will not work
```

```
    rcall    USER_LCD_Cmd        ; correctly after a second USER_LCD_Init
    movlw    0x0c
    rcall    USER_LCD_Cmd
    movlw    0x00           ; set cursor to zero pos
    rgoto    USER_LCD_CursorSet

;; ----------------------------------------------------------------------
-
;;  FUNCTION: USER_LCD_Data
;;  DESCRIPTION: sends a data value to the LCD display.<BR>
;;  On CLCDs: branch directly to USER_LCD_PrintChar<BR>
;;  On GLCDs: ignore this function!
;;  IN:  data which should be sent
;;  OUT: -
;; ----------------------------------------------------------------------
-
USER_LCD_Data
    ;; store byte in data latch
    movwf    USER_LCD_LAT_D
    ;; store bits 3:2 into port E bits 2:1
    rrncf     WREG, 0, 0    ;; shift right
    movwf    LATE        ;; store in port E latch
    rlncf    WREG, 0, 0    ;; shift left, leave WREG as it was!

    ;; wait until display unbusy
    rcall    USER_LCD_WaitUnbusy

    ;; exit if current LCD not available due to timeout
    BIFSET    USER_LCD_STATUS, USER_LCD_STATUS_CUR_DISABLED, BANKED, return

    ;; select data register
        bsf     USER_LCD_LAT_RS, USER_LCD_PIN_RS

    ;; activate write
        bcf     USER_LCD_LAT_RW, USER_LCD_PIN_RW

    ;; strobe and exit
    rcall    USER_LCD_Strobe_Set
    rgoto    USER_LCD_Strobe_Clr



;; ----------------------------------------------------------------------
-
;;  FUNCTION: USER_LCD_Cmd
;;  DESCRIPTION: sends a command to the LCD display.<BR>
;;  On CLCDs: use this function to decode the HD44780 commands if
required<BR>
;;  On GLCDs: ignore this function!
;;  IN:  command which should be sent
;;  OUT: -
;; ----------------------------------------------------------------------
```

```
-
USER_LCD_Cmd
    ;; store byte in data latch
    movwf    USER_LCD_LAT_D
    ;; store bits 3:2 into port E bits 2:1
    rrncf     WREG, 0, 0    ;; shift right
    movwf    LATE        ;; store in port E latch
    rlncf    WREG, 0, 0    ;; shift left, leave WREG as it was!

    ;; wait until display unbusy
    rcall    USER_LCD_WaitUnbusy

    ;; exit if current LCD not available due to timeout
    BIFSET    USER_LCD_STATUS, USER_LCD_STATUS_CUR_DISABLED, BANKED, return

    ;; select command register
        bcf     USER_LCD_LAT_RS, USER_LCD_PIN_RS

    ;; activate write
        bcf     USER_LCD_LAT_RW, USER_LCD_PIN_RW

    ;; strobe and exit
    rcall    USER_LCD_Strobe_Set
    rgoto    USER_LCD_Strobe_Clr


;; ----------------------------------------------------------------------------
-
;;  This function is NOT called by MIOS, but only used by the custom driver
;;  to wait until the LCD is unbusy
;;  In:  -
;;  Out: -
;; ----------------------------------------------------------------------------
-
USER_LCD_WaitUnbusy
    ;; exit if current LCD not available due to timeout
    BIFSET    USER_LCD_STATUS, USER_LCD_STATUS_CUR_DISABLED, BANKED, return

    ;; turn off output drivers
    movlw    0xf3        ; set all except TRISB[3:2] as input
    iorwf    USER_LCD_TRIS_D, F
    movlw    0x06        ; set only TRISE[2:1] as input
    iorwf    TRISE, F

    ;; select command register
        bcf     USER_LCD_LAT_RS, USER_LCD_PIN_RS

    ;; poll busy bit
    clrf    MIOS_LCD_TIMEOUT0, BANKED
    clrf    MIOS_LCD_TIMEOUT1, BANKED
```

```
        bsf     USER_LCD_LAT_RW, USER_LCD_PIN_RW    ; LCD_READ
USER_LCD_WaitUnbusy_Loop
    rcall    USER_LCD_Strobe_Clr
    incf    MIOS_LCD_TIMEOUT0, F, BANKED
    skpnz
    incf    MIOS_LCD_TIMEOUT1, F, BANKED
    bz    USER_LCD_WaitUnbusy_Disable    ; leave loop when LCD_TIMEOUT =
0xff. Up to now bit 7 is set and the LCD
                        ; busy routine will never be called again
    rcall    USER_LCD_Strobe_Set
    IFSET    USER_LCD_PORT_D, 7, rgoto USER_LCD_WaitUnbusy_Loop
    rcall    USER_LCD_Strobe_Clr

USER_LCD_WaitUnbusy_End
    ;; turn on output drivers again
    movlw    0x0c        ; set all except TRISB[3:2] as output
    andwf    USER_LCD_TRIS_D, F
    movlw    0xf9        ; set only TRISE[2:1] as output
    andwf    TRISE, F
    return

USER_LCD_WaitUnbusy_Disable
    ;; disable currently selected LCD
    btfss    USER_LCD_STATUS, USER_LCD_STATUS_CUR_LCD, BANKED
    bsf    USER_LCD_STATUS, USER_LCD_STATUS_LCD0_DISABLED, BANKED
    btfsc    USER_LCD_STATUS, USER_LCD_STATUS_CUR_LCD, BANKED
    bsf    USER_LCD_STATUS, USER_LCD_STATUS_LCD1_DISABLED, BANKED
    rgoto    USER_LCD_WaitUnbusy_End


;; ----------------------------------------------------------------------
-
;;  This function is NOT called by MIOS, but only used by the custom driver
;;  to set the strobe line to logic-1
;;  In:  -
;;  Out: -
;; ----------------------------------------------------------------------
-
USER_LCD_Strobe_Set
    ;; (code for variable E output removed)
    bsf    USER_LCD_LAT_E, USER_LCD_PIN_E
    nop
    nop
    nop
    nop
    nop
    return

;; ----------------------------------------------------------------------
-
;;  This function is NOT called by MIOS, but only used by the custom driver
```

```
;;  to set the strobe line to logic-0
;;  In:  -
;;  Out: -
;; ------------------------------------------------------------------------
-
USER_LCD_Strobe_Clr
    ;; (code for variable E output removed)
    nop
    nop
    nop
    nop
    nop
    bcf    USER_LCD_LAT_E, USER_LCD_PIN_E
    return


;; ------------------------------------------------------------------------
-
;;  This function is called by MIOS when the custom LCD should be cleared
;;  In:  MIOS_LCD_OPTION1 - contains the first LCD option given by
MIOS_LCD_TypeSet
;;       MIOS_LCD_OPTION2 - contains the second LCD option given by
MIOS_LCD_TypeSet
;;  Out: -
;; ------------------------------------------------------------------------
-
USER_LCD_Clear
    movlw    0x01
    call    USER_LCD_Cmd
    BIFSET    MIOS_LCD_Y2_OFFSET, 7, BANKED, rgoto USER_LCD_Clear2
    BIFSET    MIOS_LCD_Y3_OFFSET, 7, BANKED, rgoto USER_LCD_Clear2
    return
USER_LCD_Clear2
    bsf    MIOS_LCD_CURSOR_POS, 7, BANKED
    movlw    0x01
    call    USER_LCD_Cmd
    bcf    MIOS_LCD_CURSOR_POS, 7, BANKED
    return

;; ------------------------------------------------------------------------
-
;;  This function is called by MIOS when the cursor should be changed
;;  In:  MIOS_LCD_OPTION1 - contains the first LCD option given by
MIOS_LCD_TypeSet
;;       MIOS_LCD_OPTION2 - contains the second LCD option given by
MIOS_LCD_TypeSet
;;       MIOS_GLCD_CURSOR_X - horizontal cursor position (for GLCDs)
;;       MIOS_GLCD_CURSOR_Y - vertical cursor position   (for GLCDs)
;;       MIOS_LCD_CURSOR_POS - character cursor position  (for CLCDs)
;;  Out: -
;; ------------------------------------------------------------------------
```

```
-
USER_LCD_CursorSet
    SET_BSR    MIOS_LCD_CURSOR_POS
    movf    MIOS_LCD_CURSOR_POS, W, BANKED
    iorlw    0x80
    rgoto    USER_LCD_Cmd

;; ---------------------------------------------------------------------------
-
;;  This function is called by MIOS when a character should be print
;;  In:  WREG - character
;;       all other MIOS_*LCD_* registers
;;  Out: GLCDs should justify the X/Y cursor position
;; ---------------------------------------------------------------------------
-
USER_LCD_PrintChar
    rgoto    USER_LCD_Data


;; ---------------------------------------------------------------------------
-
;;  FUNCTION: USER_LCD_SpecialCharInit
;;  DESCRIPTION: see MIOS_CLCD_SpecialCharInit
;;  IN:  number of special character (0-7) in WREG
;;       pointer to special char pattern in TBLPTR (consists of 8
;;          entries for every character-line)
;;  OUT: TBLPTR has to be set to next table entry (TBLPTR+=8)
;; ---------------------------------------------------------------------------
-
USER_LCD_SpecialCharInit
    ;; transfer special character to display
    swapf    WREG, F
    rrf    WREG, W
    andlw    0x38
    iorlw    0x40
    rcall    USER_LCD_Cmd

    SET_BSR    USER_LCD_SC_CTR
    clrf    USER_LCD_SC_CTR, BANKED
USER_LCD_SpecialCharInitLoop
    tblrd*+
    movf    TABLAT, W
    rcall    USER_LCD_Data
    incf    USER_LCD_SC_CTR, F, BANKED
    BIFCLR    USER_LCD_SC_CTR, 3, BANKED, rgoto USER_LCD_SpecialCharInitLoop

    goto    USER_LCD_CursorSet
```