```
/*
 * lcdUtil_xmega.asm
 *
 *  Created: 8/21/2011 8:00:20 PM
 *   Author: hanway
 */
    .equ    lcdPORT = PORTK_OUT ; LCD data bus and control signals
    .equ    lcdPORT_DIR = PORTK_DIR ; LCD port direction control
    .def    tmp = r20
    .equ    lcdE    = 0x02      ; mask for E signal
    .equ    lcdRS   = 0x04      ; mask for RS signal
    .equ    lcdRW   = 0x08      ; mask for R/W signal
; ------------------------------------------------------------------------------------------
; This subroutine outputs the command in r16 to the LCD controller and wait for 50 us to
; make sure most of the LCD instructions have finished execution.
; ------------------------------------------------------------------------------------------
cmd2LCD:
    lds tmp,lcdPORT
    andi    tmp,~(lcdRW+lcdRS); select IR register and write operation
    sts lcdPORT,tmp ;   "
    ori tmp,lcdE    ; pull E high
    sts lcdPORT,tmp ;   "
    mov tmp,r16     ; place cmd in tmp
    andi    tmp,0xF0    ; mask out the lower 4 bits
    ori tmp,lcdE    ; combine cmd upper 4 bits, R/W, RS, and E
    sts lcdPORT,tmp ; output to LCD port
    nop
    nop
    nop
    nop
    nop
    nop
    lds tmp,lcdPORT
    andi    tmp,~(lcdE) ; pull E low
    sts lcdPORT,tmp ; to terminate write of upper 4 cmd bits
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    ori tmp,lcdE    ; pull E high
    sts lcdPORT,tmp ;   "
    swap    r16     ; output lower 4 cmd bits, E, RS, and R/W bits
    andi    r16,0xF0    ;   "
    ori r16,lcdE    ;   "
    sts lcdPORT,r16 ;   "
    nop
    nop
    nop
    nop
    nop
    nop
    lds tmp,lcdPORT ; pull E low to terminate write of lower 4 cmd bits
    andi    tmp,~(lcdE) ;   "
    sts lcdPORT,tmp ;   "
    call    delay50us   ; wait until command is completed
    ret
; ------------------------------------------------------------------------------------------
```

```
; The following subroutine initializes the LCD controller to operate with 2-row display, 5 x 8
; font, 4-bit interface, cursor move right, enable display, cursor, and cursor blinking.
; ------------------------------------------------------------------------------------------------
openLCD:
    ldi tmp,0xFF    ; configure PORTK for output
    sts lcdPORT_DIR,tmp ;   "
    ldi r16,3       ; wait for LCD finish internal initialization
    call    delayby100ms    ;   "
    ldi r16,0x33
    call    cmd2LCD
    ldi r16,5
    call    delayby1ms
    ldi r16,0x32
    call    cmd2LCD
    ldi r16,2
    call    delayby1ms
    ldi r16,0x28    ; select 4-bit data, 2-line display, 5 x 8 font
    call    cmd2LCD     ;   "
    ldi r16,0x0F    ; turn on display, cursor, and cursor blinking
    call    cmd2LCD     ;   "
    ldi r16,0x01    ; clear screen, move cursor to home position
    call    cmd2LCD     ;   "
    ldi r16,2       ; wait for the last command to complete
    call    delayby1ms  ;   "
    ldi r16,0x06    ; move cursor right
    call    cmd2LCD     ;   "
    ret
; ------------------------------------------------------------------------------------------------
; The following subroutine outputs a character passed in r16 to the LCD screen.
; ------------------------------------------------------------------------------------------------
putc2LCD:
    lds tmp,lcdPORT ; set RS signal high to select DR register
    ori tmp,lcdRS   ;   "
    sts lcdPORT,tmp ;   "
    andi    tmp,~lcdRW  ; pull R/W low to select write operation
    sts lcdPORT,tmp ;   "
    ori tmp,lcdE    ; pull E high
    sts lcdPORT,tmp ;   "
    mov tmp,r16     ; get a copy of data byte
    andi    tmp,0xF0    ; mask out lower 4 bits
    ori tmp,lcdRS+lcdE  ; combine upper 4 data bits, RS, and E
    sts lcdPORT,tmp ; and output to LCD
    nop         ; extend E pulse to more than 250 us
    nop         ;   "
    nop         ;   "
    nop         ;   "
    nop         ;   "
    nop         ;   "
    lds tmp,lcdPORT ; pull E low to terminate write of upper 4 data bits
    andi    tmp,~lcdE   ;   "
    sts lcdPORT,tmp ;   "
    nop         ; extend E low interval so that tC is longer than 500 us
    nop         ;   "
    nop         ;   "
    nop         ;   "
    nop         ;   "
    nop         ;   "
    nop         ;   "
    ori tmp,lcdE    ; pull E high
    sts lcdPORT,tmp ;   "
```

```
    swap    r16     ; swap upper and lower nibbles of data byte
    andi    r16,0xF0    ; mask out the upper data nibble
    ori r16,lcdRS+lcdE  ; output lower 4 data bits, RS, and E
    sts lcdPORT,r16 ;   "
    nop         ; extend E pulse so that it is longer than 250 us
    nop         ;   "
    nop         ;   "
    nop         ;   "
    nop         ;   "
    lds r16,lcdPORT ; pull E low to terminate write of lower 4 data bits
    andi    r16,~lcdE   ;   "
    sts lcdPORT,r16 ;   "
    call    delay50us   ; wait until data write is completed
    ret
; ---------------------------------------------------------------------------------------------
; The next subroutine outputs a string in program memory pointed to by Z
; ---------------------------------------------------------------------------------------------
putSC2LCD:
loopSC: lpm r16,z+
    cpi r16,0
    breq    doneCS
    call    putc2LCD
    jmp loopSC
doneCS: ret
; ---------------------------------------------------------------------------------------------
; The next subroutine outputs a string in data memory pointed to by Z.
; ---------------------------------------------------------------------------------------------
putSD2LCD:
loopSD: ld  r16,z+
    cpi r16,0
    breq    doneDS
    call    putc2LCD
    jmp loopSD
doneDS: ret
```