How to control HD44780-based Character-LCD

(Industry-Standard-Character-LCD)
Code-examples for PIC16C84

If you viewing these pages in someone elses frame,
click here to view *"Peer's LCD Pages"* in a new browser window

Last updated: 19/02/11

4. PIC example
- 4.1. Basic control software
  - 4.1.1. Requirements / features
  - 4.1.2. Global declarations
    - 4.1.2.1. Register declarations
    - 4.1.2.2. Literal declarations
    - 4.1.2.3. Procedure declarations / library interface
  - 4.1.3. Code
    - 4.1.3.1. LCD initialisation
    - 4.1.3.2. Busy flag
    - 4.1.3.3. Clear display
    - 4.1.3.4. Cursor home
    - 4.1.3.5. Entry mode
    - 4.1.3.6. Display mode
    - 4.1.3.7. Set character generator RAM address
    - 4.1.3.8. Set display data RAM address
    - 4.1.3.9. Get address counter contents
    - 4.1.3.10. Write character
    - 4.1.3.11. Write command
    - 4.1.3.12. Delay loops
  - 4.1.4. Availability
- 4.2. Advanced control software
  - 4.2.1. User defined characters
  - ...
- 4.3. Used hardware
  - 4.3.1 Controller
  - 4.3.2 LCD hardware interface
- 4.4. Development environment
  - 4.4.1. Software
  - 4.4.2. Hardware

4. PIC example

TOC

4.1. Basic control software

Microchip's AN587 was used as a basis for this code.

WARNING:
Microchip's AN587 has major errors in the ***read from*** LCD code sequences.
The routines on this page use the correct *read from* LCD code sequences.

TOC

4.1.1. Requirements / features

- HD44780-based (industry-standard) character-LCD, all software in this chapter is based on it's instruction-set.
- PIC16C84 running on a 4MHz crystal, some code is based on this frequency.
- 8-bit interface between microcontroller and LCD-module.

TOC

4.1.2. Global declarations

To get things working.

TOC

4.1.2.1. Register declarations

Purpose:
- Tells MPASM which ports and registers (files) to use.

Code:

```
LCD_DATA EQU      PORTB             ; LCD data lines interface
LCD_DATA_TRIS     EQU      TRISB
LCD_CTRL EQU      PORTA             ; LCD control lines interface

LCD_TEMP EQU      0x020             ; LCD subroutines internal use

DELAY             EQU      0x023             ; Used in DELAYxxx routines
X_DELAY           EQU      0x024             ; Used in X_DELAYxxx routines
```

TOC

4.1.2.2. Literal declarations

Purpose:
- Literal declarations (Equates) used in the code.

Code:

```
; PORTA control bits
LCD_E             EQU      2                 ; LCD Enable control line
LCD_RW            EQU      1                 ; LCD Read/Write control line
LCD_RS            EQU      0                 ; LCD Register-Select control line
```

TOC

4.1.2.3. Procedure declarations / library interface

Since MPLIB and MPLINK are not yet available, no declarations are needed.

TOC

4.1.3. Code

TOC

4.1.3.1. LCD initialisation

Purpose:
- LCD initialisiation code to be executed after power-up (i.e.: *before* any other subrou
- Should be modified to your needs (i.e. display type, cursor on/off, etc.)

Code:

```
LCDINIT
                                        ; Busy-flag is not yet valid
                  CLRF     LCD_CTRL ; ALL PORT output should output Low.
                                        ; power-up delay
                  MOVLW    0x01E
                  CALL     X_DELAY500        ; 30 * 0.5mS = 15mS
                                        ; Busy Flag should be valid from here
                  MOVLW    0x038             ; 8-bit-interface, 2-lines
                  CALL     LCDPUTCMD
                  MOVLW    0x000             ; disp.off, curs.off, no-blink
                  CALL     LCDDMODE
                  CALL     LCDCLEAR
                  MOVLW    0x004             ; disp.on, curs.off
                  CALL     LCDDMODE
                  MOVLW    0x002             ; auto-inc (shift-cursor)
                  CALL     LCDEMODE
                  RETURN
```

TOC

4.1.3.2. Busy flag

Purpose:
- Tests if the LCD is busy. Returns when LCD busy-flag is inactive.

Code:

```
LCDBUSY
                  BSF      STATUS,RP0        ; Select Register page 1
                  MOVLW    0x0FF             ; Set PORTB for input
                  MOVWF    LCD_DATA_TRIS
                  BCF      STATUS, RP0       ; Select Register page 0
                  BCF      LCD_CTRL, LCD_RS; Set LCD for command mode
                  BSF      LCD_CTRL, LCD_RW; Setup to read busy flag
                  BSF      LCD_CTRL, LCD_E   ; LCD E-line High
                  MOVF     LCD_DATA, W       ; Read busy flag + DDram address
                  BCF      LCD_CTRL, LCD_E   ; LCD E-line Low
                  ANDLW    0x80              ; Check Busy flag, High = Busy
                  BTFSS    STATUS, Z
                  GOTO     LCDBUSY
LCDNOTBUSY        BCF      LCD_CTRL, LCD_RW
                  BSF      STATUS, RP0       ; Select Register page 1
                  MOVLW    0x000
                  MOVWF    LCD_DATA_TRIS     ; Set PORTB for output
                  BCF      STATUS, RP0       ; Select Register page 0
                  RETURN
```

TOC

4.1.3.3. Clear display

Purpose:
- Clears display and returns cursor to home position (upper-left corner).

Code:

```
LCDCLEAR
                MOVLW   0x001
                CALL    LCDPUTCMD
                RETURN
```

TOC

4.1.3.4. Cursor home

Purpose:

```
- Returns cursor to home position.
- Returns display to original position (when shifted).
```

Code:

```
LCDHOME
                MOVLW   0x002
                CALL    LCDPUTCMD
                RETURN
```

TOC

4.1.3.5. Entry mode

Purpose:

```
- Sets entry mode of the LCD
- Required entry mode must be set in W
  b0    : 0 = no display shift, 1 = display shift
  b1    : 0 = auto-decrement, 1 = auto-increment
  b2-b7 : don't care
```

Code:

```
LCDEMODE
                ANDLW   0x003           ; Strip upper bits
                IORLW   0x004           ; Function set
                CALL    LCDPUTCMD
                RETURN
```

TOC

4.1.3.6. Display mode

Purpose:

```
- Sets display control
- Required entry mode must be set in W
  b0    : 0 = cursor blink off, 1 = cursor blink on (if b1 = 1)
  b1    : 0 = cursor off,  1 = cursor on
  b2    : 0 = display off, 1 = display on (display data remains in DD-RAM)
  b3-b7 : don't care
```

Code:

```
LCDDMODE
                ANDLW   0x007           ; Strip upper bits
                IORLW   0x008           ; Function set
                CALL    LCDPUTCMD
                RETURN
```

TOC

4.1.3.7. Set character generator RAM address

Purpose:

```
- Sets the Character-Generator-RAM address. CGRAM data is read/written after this setti
- Required CGRAM address must be set in W
   b0-5  : required CGRAM address
   b6-7  : don't care
```

Code:

```
LCDSCGA
                ANDLW    0x03F            ; Strip upper bits
                IORLW    0x040            ; Function set
                CALL     LCDPUTCMD
                RETURN
```

TOC

4.1.3.8. Set display data RAM address

Purpose:

```
- Sets the Display-Data-RAM address. DDRAM data is read/written after this setting.
- Required entry mode must be set in W
   b0-6  : required DDRAM address
   b7    : don't care
```

Code:

```
LCDSDDA
                IORLW    0x080            ; Function set
                CALL     LCDPUTCMD
                RETURN
```

TOC

4.1.3.9. Get address counter contents

Purpose:

```
- Returns address counter contents, used for both DDRAM and CGRAM.
- RAM address is returned in W
```

Code:

```
LCDGADDR
                BSF      STATUS,RP0       ; Select Register page 1
                MOVLW    0x0FF            ; Set PORTB for input
                MOVWF    LCD_DATA_TRIS
                BCF      STATUS, RP0      ; Select Register page 0
                BCF      LCD_CTRL, LCD_RS; Set LCD for command mode
                BSF      LCD_CTRL, LCD_RW; Setup to read busy flag
                BSF      LCD_CTRL, LCD_E  ; LCD E-line High
                MOVF     LCD_DATA, W      ; Read busy flag + RAM address
                BCF      LCD_CTRL, LCD_E  ; LCD E-line Low
                ANDLW    0x07F            ; Strip upper bit
                BCF      LCD_CTRL, LCD_RW
                BSF      STATUS, RP0      ; Select Register page 1
                MOVLW    0x000
                MOVWF    LCD_DATA_TRIS    ; Set PORTB for output
                BCF      STATUS, RP0      ; Select Register page 0
                RETURN
```

TOC

4.1.3.10. Write character

Purpose:

```
- Sends character to LCD
- Required character must be in W
```

Code:

```
LCDPUTCHAR
                MOVWF   LCD_TEMP ; Character to send is in W
                CALL    LCDBUSY          ; Wait for LCD to be ready
                BCF     LCD_CTRL, LCD_RW; Set LCD in read mode
                BSF     LCD_CTRL, LCD_RS; Set LCD in data mode
                BSF     LCD_CTRL, LCD_E  ; LCD E-line High
                MOVF    LCD_TEMP, W
                MOVWF   LCD_DATA ; Send data to LCD
                BCF     LCD_CTRL, LCD_E  ; LCD E-line Low
                RETURN
```

TOC

4.1.3.11. Write command

Purpose:
- Sends command to LCD
- Required command must be in W

Code:

```
LCDPUTCMD
                MOVWF   LCD_TEMP ; Command to send is in W
                CALL    LCDBUSY          ; Wait for LCD to be ready
                BCF     LCD_CTRL, LCD_RW; Set LCD in read mode
                BCF     LCD_CTRL, LCD_RS; Set LCD in command mode
                BSF     LCD_CTRL, LCD_E  ; LCD E-line High
                MOVF    LCD_TEMP, W
                MOVWF   LCD_DATA ; Send data to LCD
                BCF     LCD_CTRL, LCD_E  ; LCD E-line Low
                RETURN
```

TOC

4.1.3.12. Delay loops

Purpose:
- Used in LCDINIT subroutine
- Required delay factor must be in W
  (Could be coded more efficient, but this approach gives more flexibility)

Code:

```
;************************************* a 500uS delay @ 4MHz X-tal
DELAY500 MOVLW    D'165'            ; +1             1 cycle
                  MOVWF   DELAY             ; +2             1 cycle
DELAY500_LOOP     DECFSZ  DELAY, F ; step 1 1 cycle
                  GOTO    DELAY500_LOOP     ; step 2 2 cycles
DELAY500_END      RETURN                    ; +3             2 cycles

;************************************* a delay of 'W' * 500mS
X_DELAY500        MOVWF   X_DELAY           ; +1             1 cycle
X_DELAY500_LOOP   CALL    DELAY500 ; step1          wait 500uSec
                  DECFSZ  X_DELAY, F        ; step2          1 cycle
                  GOTO    X_DELAY500_LOOP   ; step3          2 cycles
X_DELAY500_END    RETURN                    ; +2             2 cycles
```

TOC

4.1.4. Availability

LCD-PIC.ZIP: an example using some of the above subroutines (all subroutines are included).
Source is coded for a 4*20 LCD, adjust it to your needs!

Shows the following screen on a 4*20 LCD:

```
        ----------------------
        |This is on line :  0|
        |This is on line :  1|
        |This is on line :  2|
        |This is on line :  3|
        ----------------------
```

Shows the following screen on a 2*40 LCD:

```
        ----------------------------------------
        |This is on line :  0This is on line :  2|
        |This is on line :  1This is on line :  3|
        ----------------------------------------
```

Shows the following screen on a 2*20 LCD:

```
        ----------------------
        |This is on line :  0|
        |This is on line :  1|
        ----------------------
```

TOC

4.2. Advanced control software

TOC

4.2.1. User defined characters

Purpose:

```
After several requests a quick explanation on how to implement
user-defined characters:

First you'll need to make a pixel definition for the characters
you want to use.  This is the pixel definition for an underlined
'0' (char code 0x30) based on a 5x7 dots character definition:

      |   bits    | byte
  row | 76543210  | value
  ----------------------
  000 |     xxx   | 0x0E
  001 |    x   x  | 0x11
  010 |    x  xx  | 0x13
  011 |    x x x  | 0x15
  100 |    xx  x  | 0x19
  101 |    x   x  | 0x11
  110 |     xxx   | 0x0E
  111 |    xxxxx  | 0x1F


The byte values need to be loaded into CGRAM address 00cccrrr
(binary), where:
-  ccc = user-defined character number (0...7)
-  rrr = row number of the user defined character (0...7)

Once that's done you can write character codes 0...7 to the
desired LCD character position, just like you do with
'normal' characters.

User-defined character definitions may be changed 'on-the-fly'.
```

Code:

```
(More detailed code may be published some day)
```

TOC

Mail me your ideas!
TOC

4.3. Used hardware

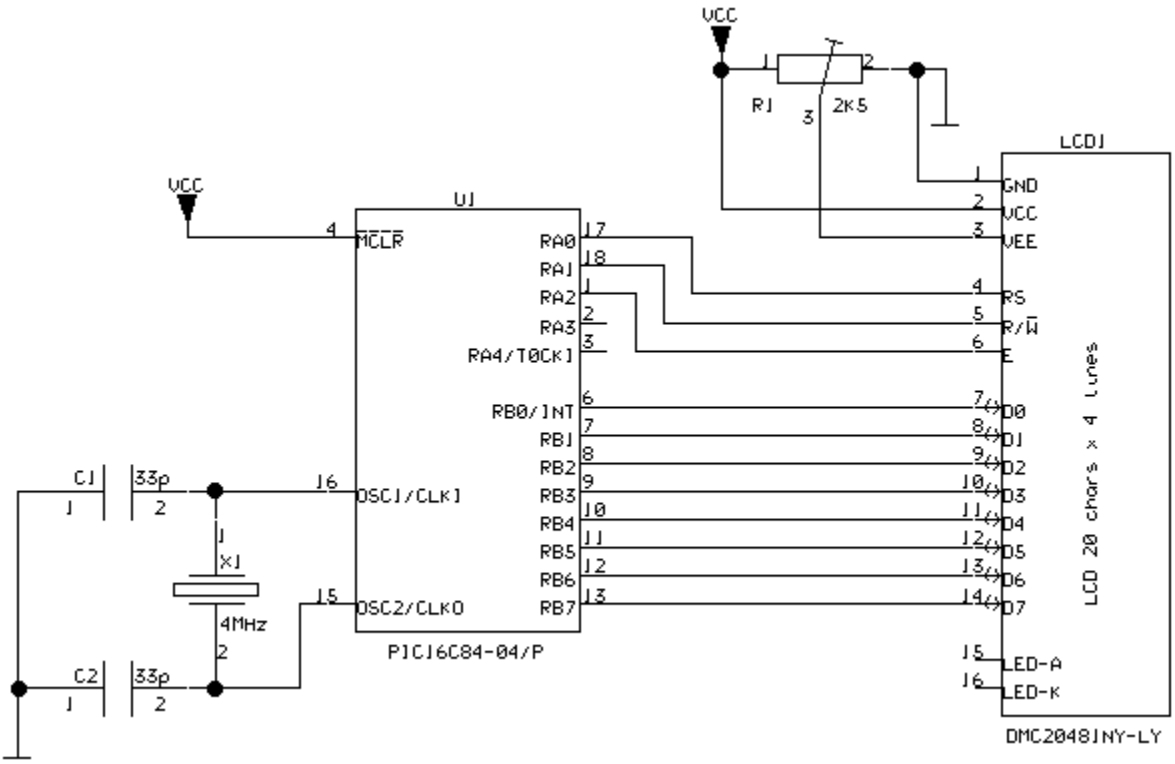
TOC

4.3.1 Controller

- A PIC16C84 is used to control the LCD.
- 8-bit data interface between controller and LCD.

TOC

4.3.2 LCD hardware interface



Shift-Click to download gif.

TOC

4.4. Development environment

TOC

4.4.1. Software

- Assembler: MPASM V1.30
- Programmer software: PICSTART 16B1 V5.00.00

TOC

4.4.2. Hardware

- Programmer PICSTART 16B1 (firmware V2.00)

TOC

[General info] [8051 example] [PIC example] [Misc. examples] [Manuf./Distrib.] [Home]