LCD Display 4-bit Interface

Introduction

1. The difference between 4 bit and 8 bit operation is that data is sent out as nibbles instead of as one byte.

2. D3:0 are not used on the LCD when the module is operated in 4-bit mode and D7:4 are used to transfer nibbles to/from the LCD module. Note: D7 is the MSB.

3. Commands and data are still 8 bits long, but are transferred as mentioned above as two 4-bit nibbles on data bus lines D7:4.

3. The most significant nibble should be transferred first, followed by the least significant nibble.

4. The optimal contrast for the LCD is 3.3 - 3.7V.

Device Pin-out

Pin Number	Symbol	Type	Function
1	Vss	-	GND
2	Vcc	-	+5V
3	Vee	Analog Input	contrast adjust, see #4 above
4	RS	Input	0 = command input/output, 1 = data input/output
5	R/W	Input	0 = write to LCD module, 1 = read from LCD module
6	EN	Input	enable signal (data strobe)
7	DB4	I/O	data bit, not used in 4 bit mode
8	DB5	I/O	data bit, not used in 4 bit mode
9	DB6	I/O	data bit, not used in 4 bit mode
10	DB7	I/O	data bit, not used in 4 bit mode

Initialization for 4-bit operation

The module powers up in 8-bit mode. The initial start-up instructions are sent in 8-bit mode, with the lower four bits (which are not connected) of each instruction as don't cares.

<POWER ON>

<Wait at least 15ms>

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	n/c	n/c	n/c	n/c

<Wait at least 4.1ms>

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	n/c	n/c	n/c	n/c

<Wait at least 100us>

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	n/c	n/c	n/c	n/c

<Wait 4.1ms>

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	0	n/c	n/c	n/c	n/c

After the fourth instruction shown above, which switches the module to 4-bit operation, the control bytes are sent on consecutive enable cycles (no delay is required between nibbles). The most significant nibble is sent first, followed immediately by the least significant nibble. See the next page for details.

4 Bit Mode Operation (Initialization of Control Bytes)

All transfers must take place in pairs, 1 nibble at a time. The remainder of the initialization sequence could be carried out by issuing calls to an LCDWriteCommand subroutine which should wait the specified time shown below after executing a command or for BF to become 0.

<Wait 40us> ;wait for the previous command to finish

RS	R/W	DB7	DB6	DB5	DB4	;Function Set Command
0	0	0	0	1	0	
0	0	N	F	X	X	;where N = Number of `lines'
						; 0 for 1/8 duty cycle -- 1 `line'
						; 1 for 1/16 duty cycle -- 2 `lines'
						;F = font
						; 0 for 5x8 dot matrix
						; 1 for 5x11 dot matrix
						;X = don't care

<Wait 40us or till BF=0>

RS	R/W	DB7	DB6	DB5	DB4	;Display Off, Cursor Off, Blink Off
0	0	0	0	0	0	
0	0	1	0	0	0	

<Wait 40us or till BF=0>

RS	R/W	DB7	DB6	DB5	DB4	;Clear Screen & Returns the Cursor Home
0	0	0	0	0	0	
0	0	0	0	0	1	

<Wait 1.64ms or till BF=0>

RS	R/W	DB7	DB6	DB5	DB4	;Inc cursor to the right when writing
0	0	0	0	0	0	; and don’t shift screen
0	0	0	1	1	0	

<Wait 40us or till BF=0>

<Initialization Complete you must now turn the display back on and begin writing characters>