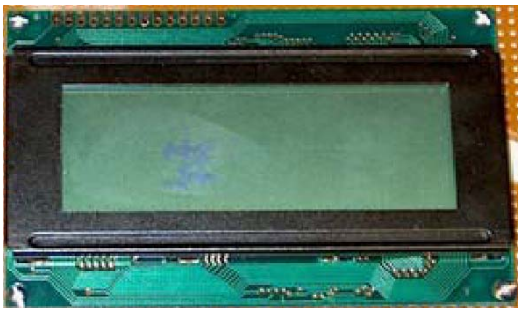
# LCDs

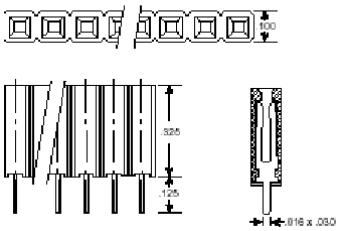
- Liquid Crystal Displays



Male pin strip header

Female pin strip socket





# LCD Lecture Notes

Each LCD display location where a character can be displayed has a unique address in display data RAM (DDRAM). To display a character on the LCD, write the character code to the appropriate address in DDRAM. The DDRAM addresses for the DMC20434 (20x4) LCD are mapped to the LCD panel as in Figure 1. The DDRAM addresses for the DMC16433 (16x4) LCD are similar to the DMC20434.

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F | 10 | 11 | 12 | 13 |
| 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 4A | 4B | 4C | 4D | 4E | 4F | 50 | 51 | 52 | 53 |
| 14 | 15 | 16 | 17 | 18 | 19 | 1A | 1B | 1C | 1D | 1E | 1F | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 54 | 55 | 56 | 57 | 58 | 59 | 5A | 5B | 5C | 5D | 5E | 5F | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 |

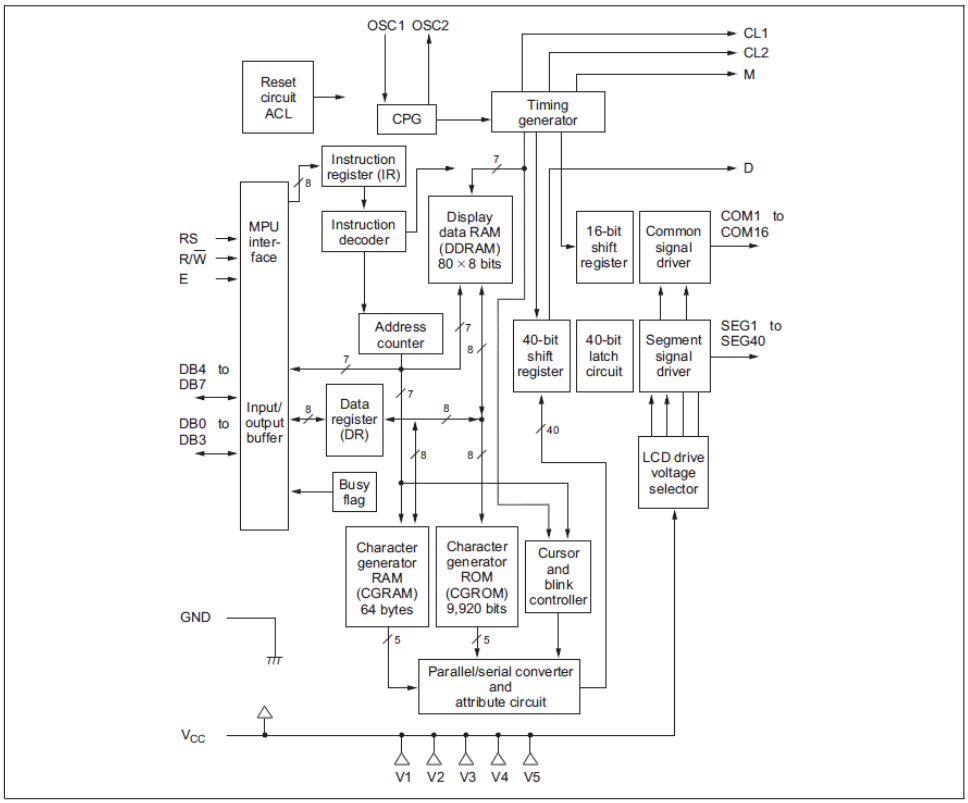
**Figure 1: 20x4 LCD Addresses (given in hexadecimal)**

The DDRAM addresses for the 16x1 LCD are mapped to the LCD panel as follows:

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |

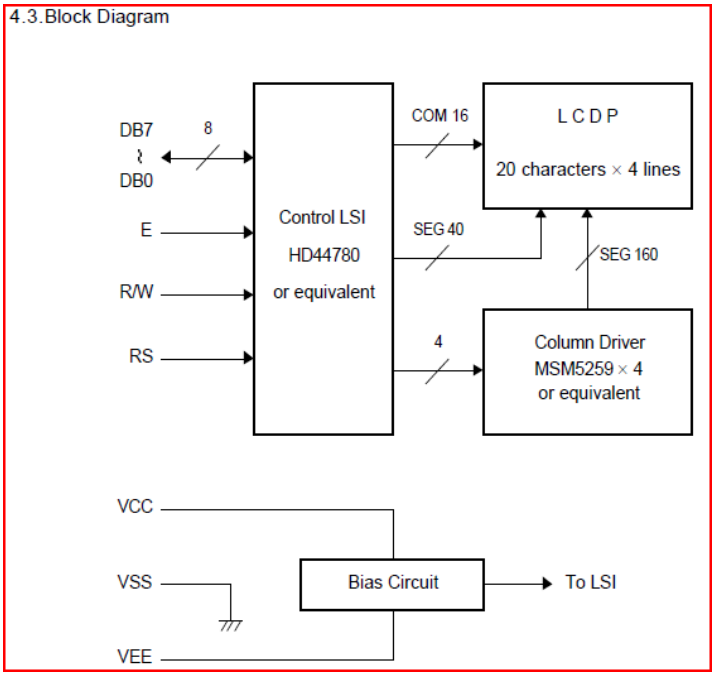
**Figure 2: 16x1 LCD Addresses (given in hexadecimal)**

## HD44780U Block Diagram






# LCD Module Block Diagram



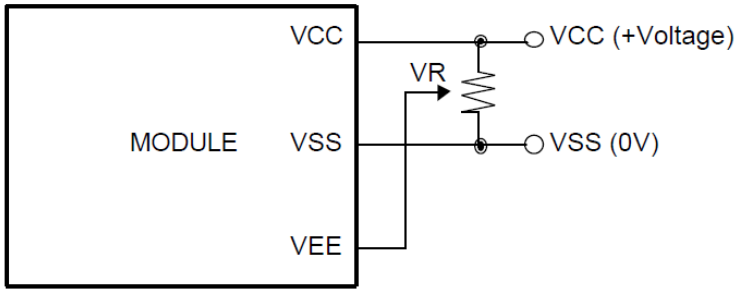
# LCD Module Pin Assignments

| No. | Symbol | Level | Function |
|---|---|---|---|
| 1 | Vss | – | Power Supply (0V, GND) |
| 2 | Vcc | – | Power Supply for Logic |
| 3 | VEE | – | Power Supply for LCD Drive |
| 4 | RS | H / L | Register Select Signal |
| 5 | R/W | H / L | Read/Write Select Signal H : Read L : Write |
| 6 | E | H / L | Enable Signal (No pull-up Resister) |
| 7 | DB0 | H / L | Data Bus Line / Non-connection at 4-bit operation |
| 8 | DB1 | H / L | Data Bus Line / Non-connection at 4-bit operation |
| 9 | DB2 | H / L | Data Bus Line / Non-connection at 4-bit operation |
| 10 | DB3 | H / L | Data Bus Line / Non-connection at 4-bit operation |
| 11 | DB4 | H / L | Data Bus Line |
| 12 | DB5 | H / L | Data Bus Line |
| 13 | DB6 | H / L | Data Bus Line |
| 14 | DB7 | H / L | Data Bus Line |



# LCD Module Contrast Adjustment

It is recommended to apply a potentiometer for the contrast adjust due to the tolerance of the driving voltage and its temperature dependence.



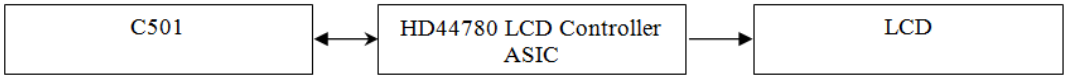

VR=10~20 KΩ

# Notes (1)

Adding an LCD to your embedded system consists of two steps:
1. Designing the hardware interface, ensuring timing requirements are met
2. Designing the LCD driver code

LCD Types: Alphanumeric and Graphic
- Alphanumeric displays allow you to control screen in groups of pixels called character codes.
- Graphic LCDs allow you to control individual pixels on the display, which enables much more detail to be displayed; however, more complex firmware is required in order to control the graphic display.

Processor-LCD interface






# Notes (2)

Optrex LCD modules (with Hitachi HD44780 LCD controller)
- Module pinout has 14-pins, standard interface
- Make sure contrast $V_{EE}$ is set properly (many people find that ground works ok; could use potentiometer)
- Hook up LCD module D0-D7 to Port 0 on C501
- RS selects instruction register or data register
- R/W# selects read or write operation
- Timing requirements for RS and R/W# are identical. Could potentially use processor address lines or general purpose port pins for these two control signals.
- Keep E low except when talking with LCD; otherwise, unwanted LCD changes will occur.

Memory Mapping
- Communicate with I/O with memory read (/RD) and write (/WR) commands
- Use MOVX command to talk with LCD (in C code, use pointers)

LCD interface solutions
- Simple solution exists
- Study the timing waveforms of the LCD controller and of the processor
- Verify timing of your solution; verify E low except when talking with LCD

# Notes (3)

Benefit of using memory mapped I/O: can use standard pointers to access I/O registers

```
LCD.H (pseudocode ideas)
#define  LCD_CMD   ((unsigned char *) 0xADDR1)       //choose an address from your memory map
#define  RD_LCD_INSTR   ((unsigned char *) 0xADDR2)
#define  WR_LCD_INSTR   ((unsigned char *) 0xADDR3)

LCD.C (pseudocode ideas)
        *LCD_CMD = temp;   // write to memory mapped LCD register (using MOVX and DPTR)
        temp = *LCD_CMD;   // read from memory mapped LCD register (using MOVX and DPTR)
```

Review initialization sequence for HD44780

- HD44780 can communicate using either an 8-bit or a 4-bit interface. 4-bit interface good for interfacing to microcontrollers with few I/O pins. HD44780 comes up in 4-bit mode and must be switched to 8-bit mode. Lower 4 bits of the Function Set instruction are don't cares during first part of initialization.
- Can use DDS delay(x) library function. Delays for x milliseconds. You can always wait longer than the minimum - you can wait twice as long as specified, if not in a hurry.
- Must remember to turn LCD display on (D bit).

Must poll the busy flag (BF) or ensure LCD has had enough time to complete previous command. Read the register containing the busy flag and mask it off. Busy flag polling only works after first few initialization commands have completed.

```
while (busy_flag == 1);  // wait for LCD ready (pseudocode)
```



# LCD Controller AC Characteristics

2.3.AC Characteristics

Vcc=5.0V±10%

| Parameter | Symbol | Conditions | Min. | Max. | Units |
|---|---|---|---|---|---|
| Enable Cycle Time | $t_{CYC}$ | Fig.1, 2 | 500 | – | ns |
| Enable Pulse Width | $PW_{EH}$ | Fig.1, 2 | 230 | – | ns |
| Enable Rise/Fall Time | $t_{Er}$, $t_{Ef}$ | Fig.1, 2 | – | 20 | ns |
| Address Setup Time | $t_{AS}$ | Fig.1, 2 | 40 | – | ns |
| Address Hold Time | $t_{AH}$ | Fig.1, 2 | 10 | – | ns |
| Write Data Setup Time | $t_{DSW}$ | Fig.1 | 80 | – | ns |
| Write Data Hold Time | $t_{DHW}$ | Fig.1 | 10 | – | ns |
| Read Data Delay Time | $t_{DDR}$ | Fig.2 | – | 160 | ns |
| Read Data Hold Time | $t_{DHR}$ | Fig.2 | 5 | – | ns |

# LCD Controller Bus Timing

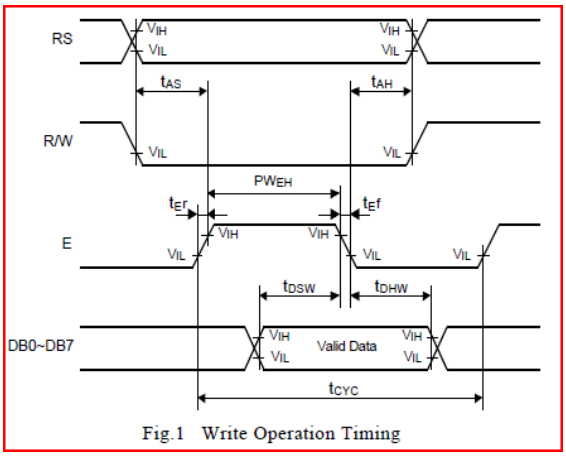


Fig.1 Write Operation Timing

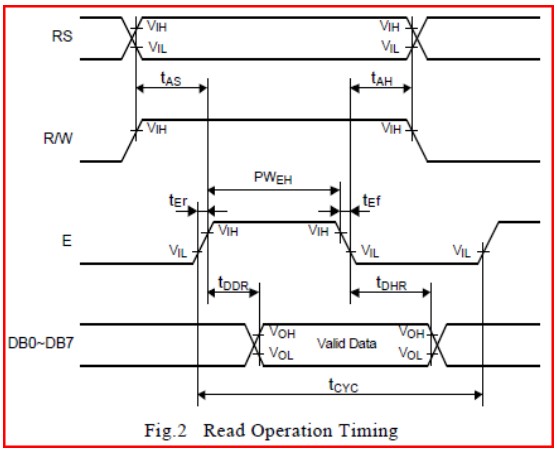


Fig.2 Read Operation Timing



# LCD Controller Write Timing

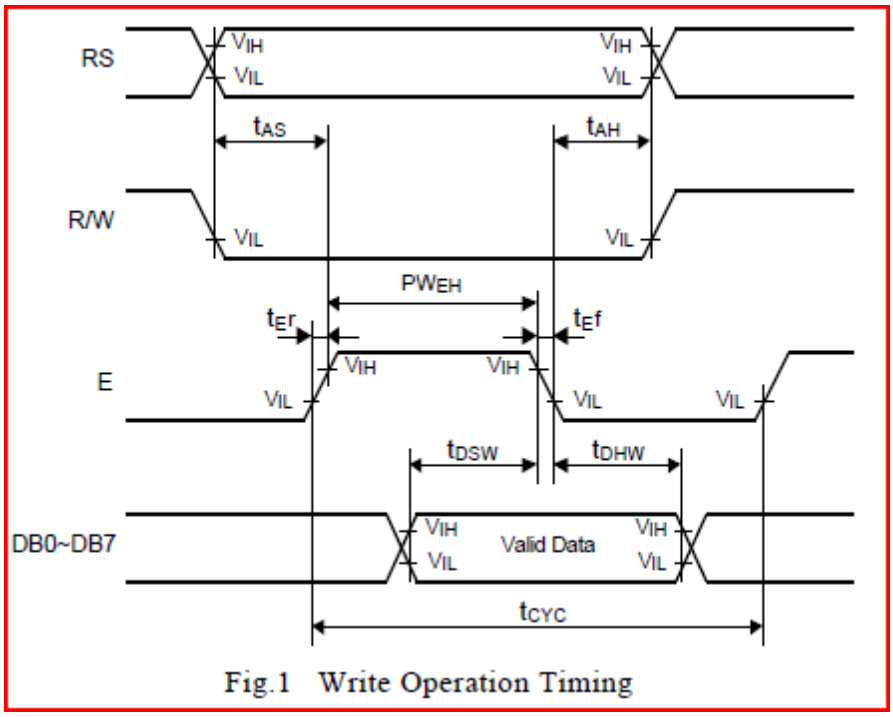


Fig.1 Write Operation Timing

# LCD Controller Read Timing

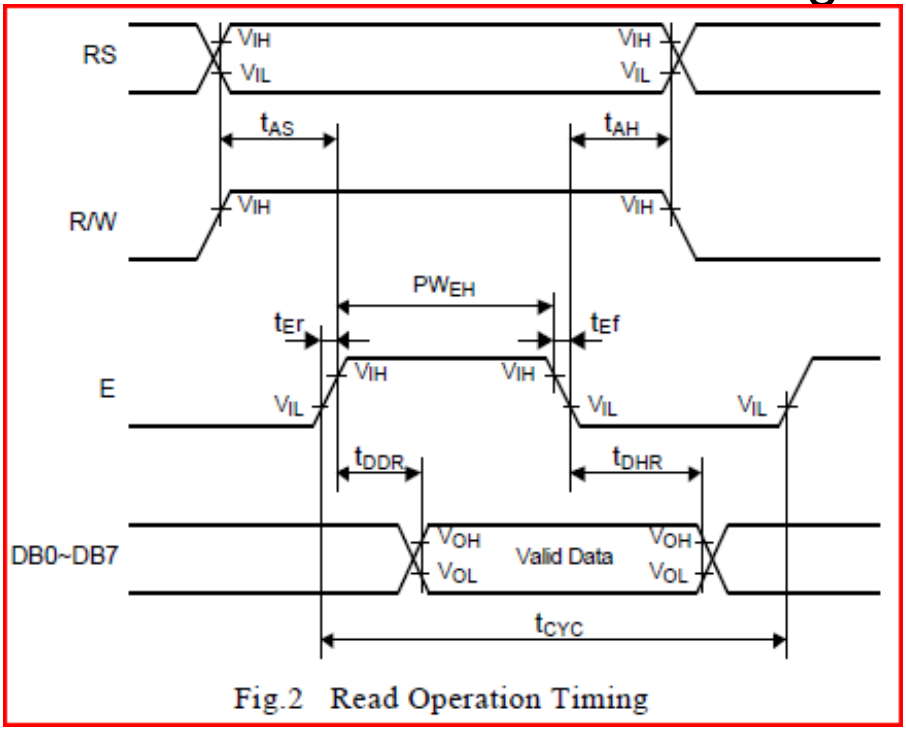



Fig.2 Read Operation Timing

 Source: Optrex DMC20434 LCD Module Data Sheet 

# Basic Processor XRAM Write Cycle

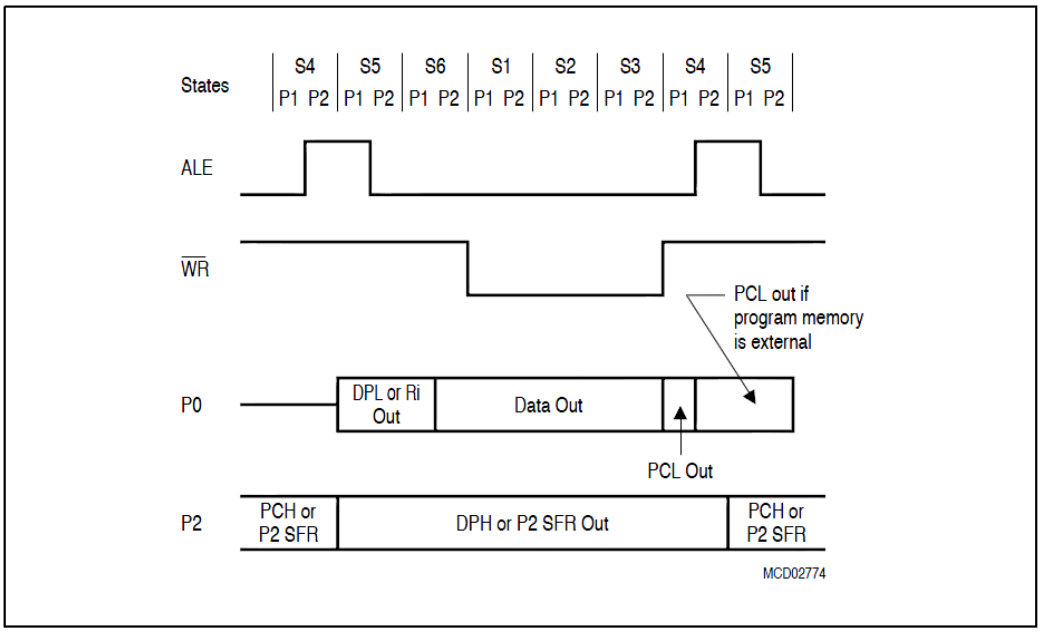



**Figure 3-4**
**External Data Memory Write Cycle**

 Source: Siemens C500 Architecture and Instruction Set Manual 

# Basic Processor XRAM Read Cycle

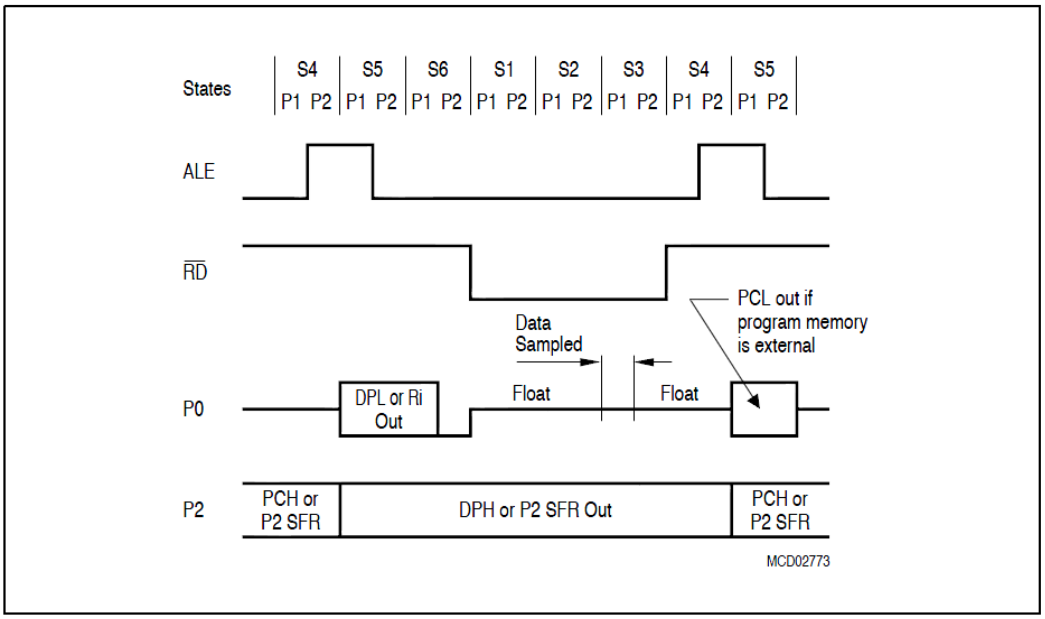



**Figure 3-3**
**External Data Memory Read Cycle**



**Table 4 Correspondence between Character Codes and Character Patterns (ROM Code: A00)**

| Upper 4 Bits / Lower 4 Bits | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| xxxx0000 | CG RAM (1) | | | 0 | @ | P | ` | p | | | | ー | タ | ミ | α | p |
| xxxx0001 | (2) | | | ! | 1 | A | Q | a | q | | | 。 | ア | チ | ム | ä | q |
| xxxx0010 | (3) | | | " | 2 | B | R | b | r | | | 「 | イ | ツ | メ | ß | θ |
| xxxx0011 | (4) | | | # | 3 | C | S | c | s | | | 」 | ウ | テ | モ | ε | ∞ |
| xxxx0100 | (5) | | | $ | 4 | D | T | d | t | | | 、 | エ | ト | ヤ | μ | Ω |
| xxxx0101 | (6) | | | % | 5 | E | U | e | u | | | ・ | オ | ナ | ユ | σ | ü |
| xxxx0110 | (7) | | | & | 6 | F | V | f | v | | | ヲ | カ | ニ | ヨ | ρ | Σ |
| xxxx0111 | (8) | | | ' | 7 | G | W | g | w | | | ァ | キ | ヌ | ラ | g | π |
| xxxx1000 | (1) | | | ( | 8 | H | X | h | x | | | ィ | ク | ネ | リ | √ | x̄ |
| xxxx1001 | (2) | | | ) | 9 | I | Y | i | y | | | ゥ | ケ | ノ | ル | ⁻¹ | y |
| xxxx1010 | (3) | | | * | : | J | Z | j | z | | | ェ | コ | ハ | レ | j | 千 |
| xxxx1011 | (4) | | | + | ; | K | [ | k | { | | | ォ | サ | ヒ | ロ | x | 万 |
| xxxx1100 | (5) | | | , | < | L | ¥ | l | \| | | | ャ | シ | フ | ワ | ¢ | 円 |
| xxxx1101 | (6) | | | - | = | M | ] | m | } | | | ュ | ス | ヘ | ン | ₹ | ÷ |
| xxxx1110 | (7) | | | . | > | N | ^ | n | → | | | ョ | セ | ホ | ゛ | ñ | |
| xxxx1111 | (8) | | | / | ? | O | _ | o | ← | | | ッ | ソ | マ | ° | ö | |

Note: The user can specify any pattern for character-generator RAM.

**Table 6 Instructions**

| Instruction | Code RS | R/$\overline{W}$ | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 | Description | Execution Time (max) (when $f_{cp}$ or $f_{OSC}$ is 270 kHz) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Clear display | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Clears entire display and sets DDRAM address 0 in address counter. | |
| Return home | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | — | Sets DDRAM address 0 in address counter. Also returns display from being shifted to original position. DDRAM contents remain unchanged. | 1.52 ms |
| Entry mode set | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | I/D | S | Sets cursor move direction and specifies display shift. These operations are performed during data write and read. | 37 µs |
| Display on/off control | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | C | B | Sets entire display (D) on/off, cursor on/off (C), and blinking of cursor position character (B). | 37 µs |
| Cursor or display shift | 0 | 0 | 0 | 0 | 0 | 1 | S/C | R/L | — | — | Moves cursor and shifts display without changing DDRAM contents. | 37 µs |
| Function set | 0 | 0 | 0 | 0 | 1 | DL | N | F | — | — | Sets interface data length (DL), number of display lines (N), and character font (F). | 37 µs |
| Set CGRAM address | 0 | 0 | 0 | 1 | ACG | ACG | ACG | ACG | ACG | ACG | Sets CGRAM address. CGRAM data is sent and received after this setting. | 37 µs |
| Set DDRAM address | 0 | 0 | 1 | ADD | ADD | ADD | ADD | ADD | ADD | ADD | Sets DDRAM address. DDRAM data is sent and received after this setting. | 37 µs |
| Read busy flag & address | 0 | 1 | BF | AC | AC | AC | AC | AC | AC | AC | Reads busy flag (BF) indicating internal operation is being performed and reads address counter contents. | 0 µs |





**Table 6 Instructions (cont)**

| Instruction | Code RS | R/$\overline{W}$ | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 | Description | Execution Time (max) (when $f_{cp}$ or $f_{OSC}$ is 270 kHz) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Write data to CG or DDRAM | 1 | 0 | Write data | | | | | | | | Writes data into DDRAM or CGRAM. | 37 µs<br>$t_{ADD}$ = 4 µs* |
| Read data from CG or DDRAM | 1 | 1 | Read data | | | | | | | | Reads data from DDRAM or CGRAM. | 37 µs<br>$t_{ADD}$ = 4 µs* |
| | I/D = 1: Increment<br>I/D = 0: Decrement<br>S = 1: Accompanies display shift<br>S/C = 1: Display shift<br>S/C = 0: Cursor move<br>R/L = 1: Shift to the right<br>R/L = 0: Shift to the left<br>DL = 1: 8 bits, DL = 0: 4 bits<br>N = 1: 2 lines, N = 0: 1 line<br>F = 1: 5 × 10 dots, F = 0: 5 × 8 dots<br>BF = 1: Internally operating<br>BF = 0: Instructions acceptable | | | | | | | | | | DDRAM: Display data RAM<br>CGRAM: Character generator RAM<br>ACG: CGRAM address<br>ADD: DDRAM address (corresponds to cursor address)<br>AC: Address counter used for both DD and CGRAM addresses | Execution time changes when frequency changes<br>Example:<br>When $f_{cp}$ or $f_{OSC}$ is 250 kHz,<br>$37\ \mu s \times \frac{270}{250} = 40\ \mu s$ |

Note: — indicates no effect.

* After execution of the CGRAM/DDRAM data write or read instruction, the RAM address counter is incremented or decremented by 1. The RAM address counter is updated after the busy flag turns off. In Figure 10, $t_{ADD}$ is the time elapsed after the busy flag turns off until the address counter is updated.

# Initialization by Instruction

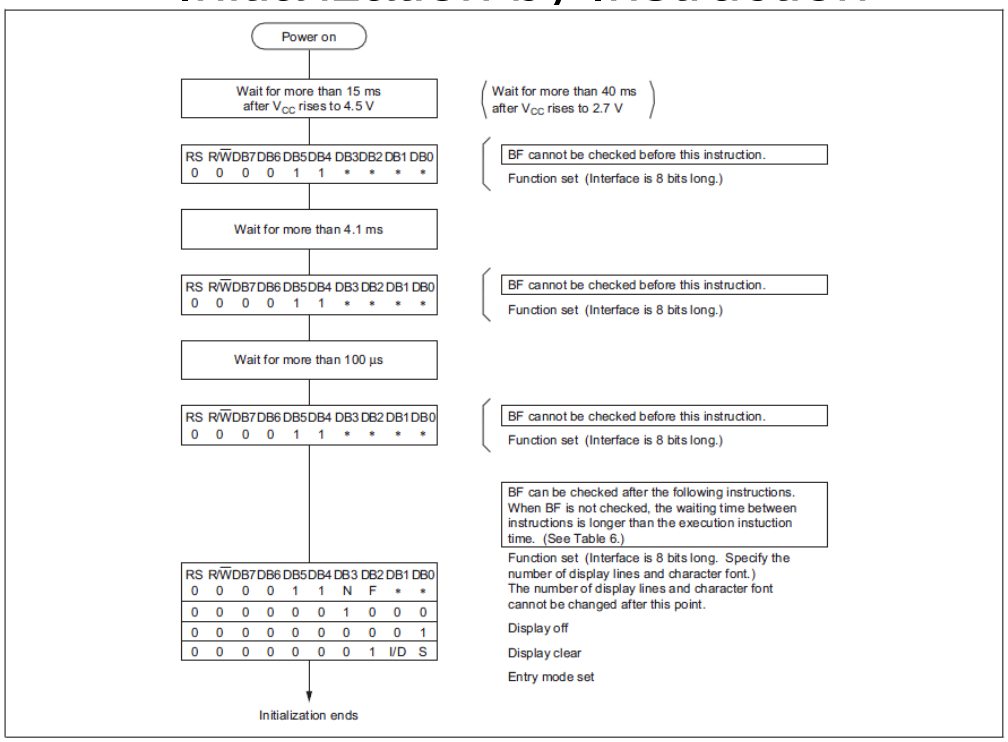



Figure 25 8-Bit Interface



## HD44780U

**Table 5 Relationship between CGRAM Addresses, Character Codes (DDRAM) and Character Patterns (CGRAM Data)**

For 5 × 8 dot character patterns

| Character Codes (DDRAM data) 7 6 5 4 3 2 1 0 High Low | CGRAM Address 5 4 3 2 1 0 High Low | Character Patterns (CGRAM data) 7 6 5 4 3 2 1 0 High Low | |
|---|---|---|---|
| 0 0 0 0 * 0 0 0 | 0 0 0 | * * * 1 1 1 1 0 | Character pattern (1) |
| | 0 0 1 | 1 0 0 0 1 | |
| | 0 1 0 | 1 0 0 0 1 | |
| | 0 1 1 | 1 1 1 1 0 | |
| | 1 0 0 | 1 0 1 0 0 | |
| | 1 0 1 | 1 0 0 1 0 | |
| | 1 1 0 | 1 0 0 0 1 | |
| | 1 1 1 | * * * 0 0 0 0 0 | Cursor position |
| 0 0 0 0 * 0 0 1 | 0 0 1 | * * * 1 0 0 0 1 | Character pattern (2) |
| | | 0 1 0 1 0 | |
| | | 1 1 1 1 1 | |
| | | 0 0 1 0 0 | |
| | | 1 1 1 1 1 | |
| | | 0 0 1 0 0 | |
| | | 0 0 1 0 0 | |
| | | * * * 0 0 0 0 0 | Cursor position |
| | 0 0 0 | * * * | |
| | 0 0 1 | | |
| 0 0 0 0 * 1 1 1 | 1 1 1 | | |
| | 1 0 0 | | |
| | 1 0 1 | | |
| | 1 1 0 | | |
| | 1 1 1 | * * * | |

# Custom Characters

DDRAM contains the information that is to be displayed on the LCD screen. The LCD controller is responsible for converting character codes stored in the DDRAM into pixels on the screen. The HD44780 LCD controller can display character patterns consisting of either blocks of 5x8 pixels or blocks of 5x10 pixels.

Predefined character patterns are stored in CGROM (character generator ROM)
Custom character patterns can be created and stored in CGRAM (character generator RAM)

To write characters to the screen:
1. Set DDRAM address
2. Write data (character code) to DDRAM

If using autoincrement, continue sending more data. Remember to poll busy flag between characters.
Polling busy flag just before writing a character is more efficient than polling immediately after writing a character.

To create custom user-defined characters:
1. Set CGRAM address (6 bit address has 3 bits for character code, 3 bits for row number)
2. Write data to CGRAM (most significant 3 bits set to 0, least significant 5 bits contain row data)