XMUT202 Digital Electronics

Liquid Crystal Display

Week 14 Lecture 1

Agatha Rachmat

School of Engineering and Computer Science

Victoria University of Wellington

Review - Interrupt

Review

1. What is polling in the 8051?
2. What is an interrupt in the 8051?
3. State one advantage and one disadvantage of polling.
4. State one advantage and one disadvantage of interrupts.

Review

1. What is polling in the 8051?
Polling is a technique where the CPU continuously checks a flag or input pin until a specific condition occurs.
Example: `WAIT: JNB TF0, WAIT`
The CPU repeatedly checks the Timer 0 overflow flag (TF0) until it becomes 1.

2. What is an interrupt in the 8051?
An interrupt is a signal that temporarily stops the current program execution and causes the CPU to execute an Interrupt Service Routine (ISR). After the ISR finishes, the CPU returns to the interrupted program.

3. State one advantage and one disadvantage of polling.
Advantage:
•Simple to program and understand.
Disadvantage:
•Wastes CPU time because the processor continuously checks the condition.

4. State one advantage and one disadvantage of interrupts.
Advantage:
•CPU can perform other tasks until an interrupt occurs.
Disadvantage:
•More complex to program than polling.

Review

5. What is the purpose of the RETI instruction?
6. Write an 8051 polling loop that waits for Timer 1 to overflow.
7. What happens when External Interrupt 1 (INT1) occurs?
8. Compare polling and interrupts.
9. What are the most common assembly codes for polling?
10. Draw Interrupt vector table
11. Write a code to enable INT1 interrupts
12. What does the following code do?

```
MOV A, P1
WAIT: JNZ WAIT
```

Review

5. What is the purpose of the RETI instruction?

RETI (Return from Interrupt):

1.Restores the Program Counter (PC) from the stack.

2.Returns execution to the interrupted program.

3.Signals that interrupt servicing are complete.

6. Write an 8051 polling loop that waits for Timer 1 to overflow.

WAIT: JNB TF1, WAIT

•TF1 = 0 → keep waiting.

•TF1 = 1 → timer overflow occurred, exit loop.

7. What happens when External Interrupt 1 (INT1) occurs?

When INT1 occurs:

1.CPU finishes the current instruction.

2.Saves the Program Counter (PC) on the stack.

3.Jumps to interrupt vector address 0013H.

4.Executes the ISR.

5.Executes RETI.

6.Returns to the main program.

Review

8. Polling vs Interrupt

Polling	Interrupt
CPU continuously checks a flag.	Hardware notifies CPU automatically.
Wastes CPU time.	More efficient CPU usage.
Easier to implement.	More complex to implement.
Suitable for simple systems.	Suitable for multitasking systems.

9. What are the common assembly code for polling?

JB	Jump if bit = 1
JNB	Jump if bit = 0
JZ	Jump if accumulator = 0
JNZ	Jump if accumulator ≠ 0
CJNE	Compare and jump if not equal
DJNZ	Decrement and jump if not zero

Review

10. Interrupt Vector table

- Reset : 00H – 02H
- External interrupt 0 (INT0) : 03H – 0AH
- Timer 0 : 0BH – 012H
- External interrupt 1, (INT1) : 13H – 1AH
- Timer 1 : 1BH – 22H
- Serial port : 22H – 23H

11. Interrupt Vector table

```
MOV IEN0 #10000100B ;Enable interrupts, INT1
```

12. What does the following code do?

```
MOV A, P1
WAIT: JNZ WAIT
```

The program checks Port 1 and keeps looping while the value is not 00H. When all bits of Port 1 become 0, the loop exits and the program continues. This is an example of **polling**.

Today's topic

- LCD in 8051 kit.
- Interfacing with LCD.
- LCD in simulator.
- LCD commands.
- Signal timing.
- Programming LCD.
- Example program.

Liquid Crystal Display (LCD)

Liquid Crystal Display Introduction

As their name suggests, LCDs are electronic devices that can display texts, custom characters, and numbers. Even though we can also display pictures on LCDs. But the results will not be comparable to those of the graphical LCDs (GLCDs). GLCDs can display images with better quality.

LCDs are available in different sizes and features

Liquid Crystal Display (LCD)

LCD in 8051 Microcontroller

An LCD (Liquid Crystal Display) is an output device used with the 8051 microcontroller to display text, numbers, and messages.

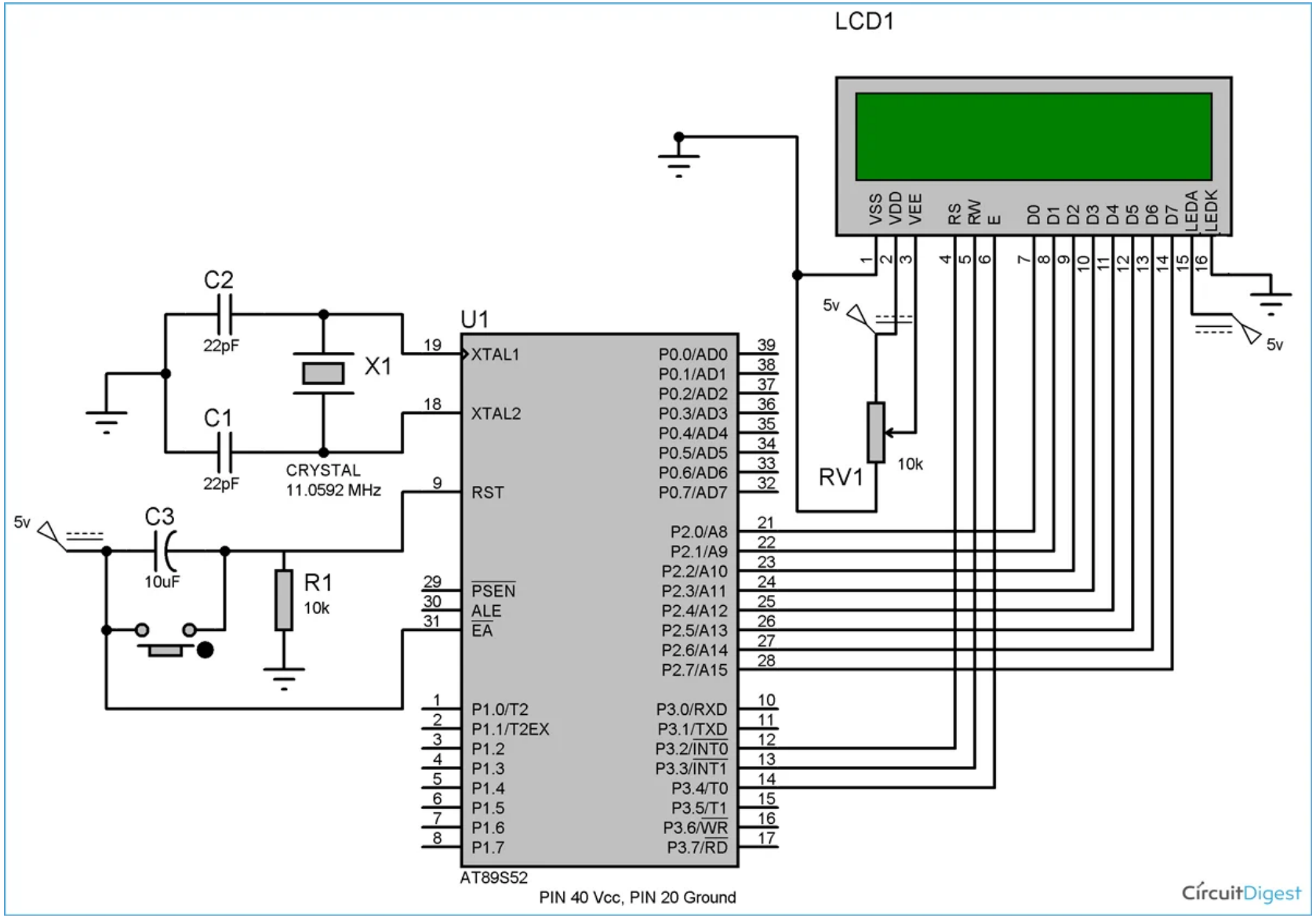
The most common LCD used is the **16×2 LCD**:

- 16 columns × 2 rows
- Can display 32 characters total

Example :

```
HELLO
8051 LCD
```

Liquid Crystal Display (LCD)



Liquid Crystal Display (LCD)

Why use LCD with 8051?

LCDs are used to:

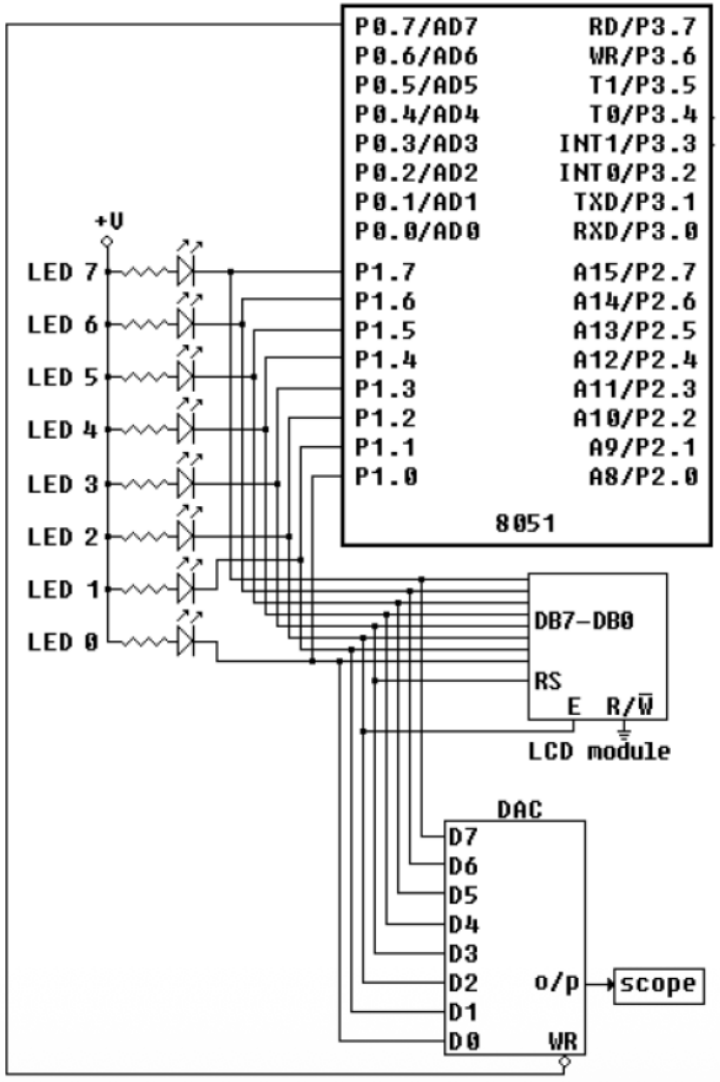
- Show messages
- Display sensor values
- Show menu systems
- Debug programs
- Display timer/count values

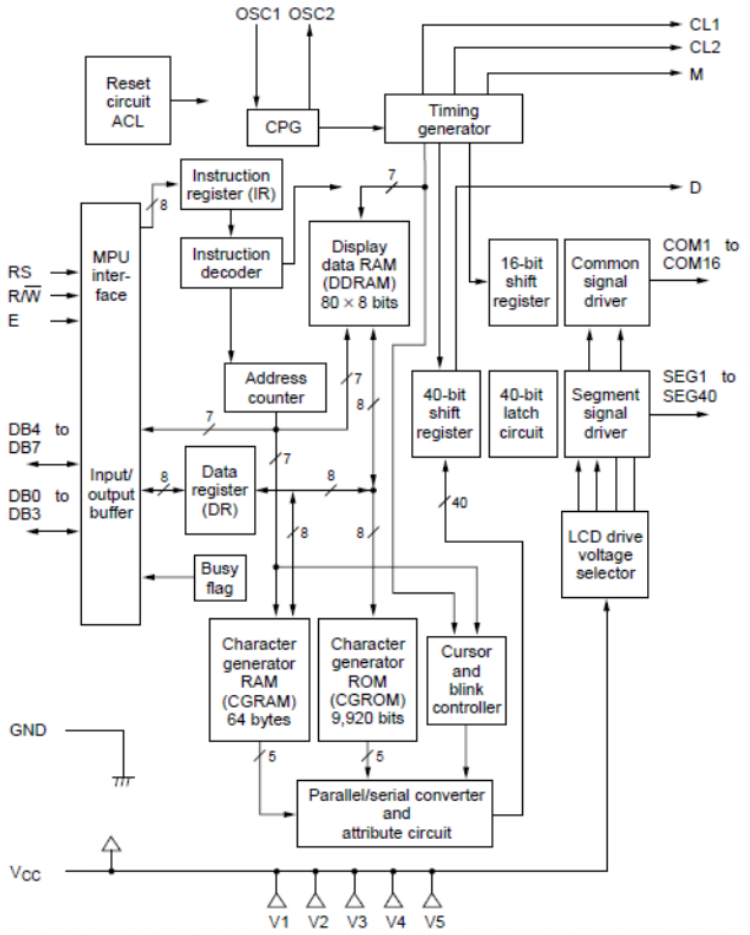
Compared to LEDs or 7-segment displays:

- LCD can display letters and symbols
- Uses less power
- Easier to read

Liquid Crystal Display (LCD)

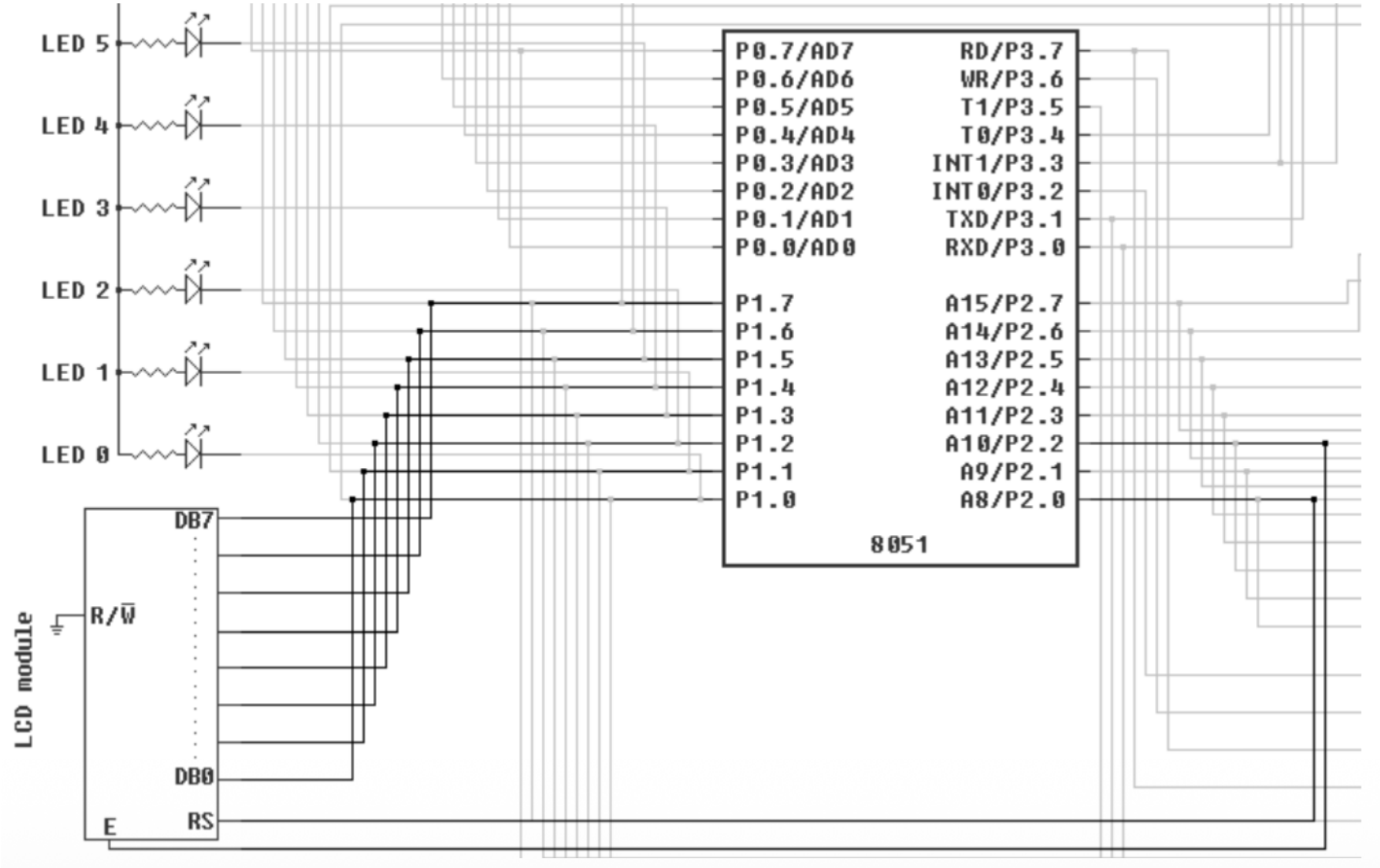
- Schematic circuit and internal circuit of 4-bit mode HD44780U dot-matrix LCD in the 8051 microcontroller lab kit.





Liquid Crystal Display (LCD)

- Schematic circuit of 8 bits mode HD44780U dot-matrix LCD in the 8051 microcontroller lab kit.



Pins in Liquid Crystal Display (LCD)

- LCDs provide a better user interface as they can display ASCII messages.
- Pin Description for LCD, i.e. LCD modules usually have 14 pins.

Pin No.	Name	Function	Description
1	V_{SS}	Power	GND
2	V_{DD}	Power	+5 V
3	V_{EE}	Contrast Adjust	0-5 V
4	RS	Register Select	Signal to select data or command register of the LCD. RS= 0(select command register, for write); Busy flag (address counter for read); RS = 1 select data register (for write)
5	R/W	Read/Write	Signal to read/ write data from/to LCD. RW = 0 (Write to LCD); RW=1 (read from LCD)
6	E	Enable(strobe)	High to low pulse is applied to this pin to enable LCD to accept (latch) data/ command present on its data lines D0-D7
7-14	D0-D7	Data lines DO (Pin7-LSB), D7 (Pin14-MSB)	Bidirectional data lines used to send data/command to LCD; or Read LCD internal registers, D7 is also used as a busy flag, D4-D7 are used in the 4 bit operation

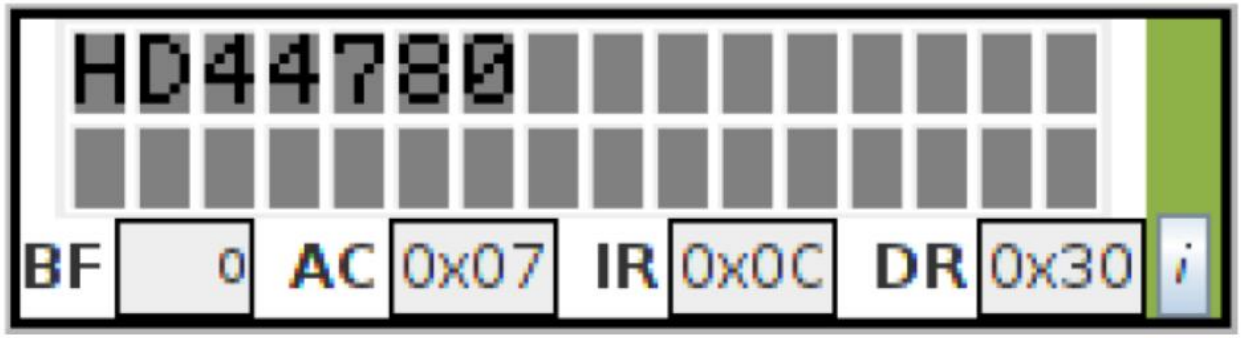
Pins in Liquid Crystal Display (LCD)

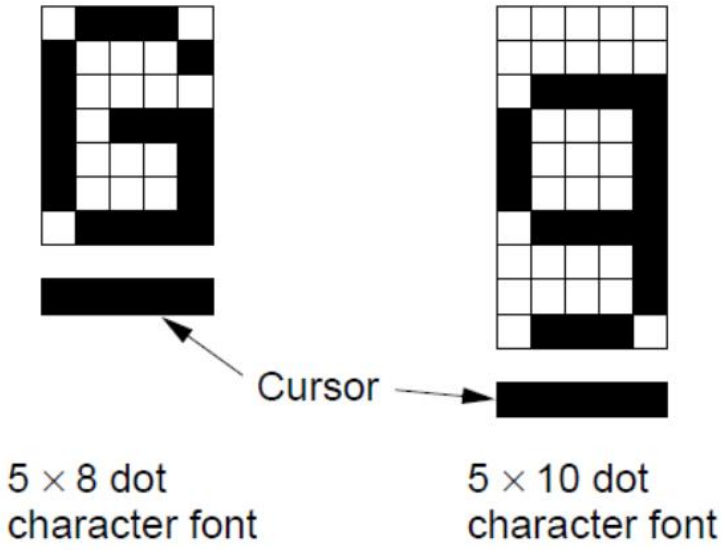
- 16x2 LCD has 16 pins.

Pin No.	Name	Function	Description
1	V_{SS}	Power	GND
2	V_{DD}	Power	+5 V
3	V_{EE}	Contrast Adjust	0-5 V
4	RS	Register Select	Signal to select data or command register of the LCD. RS= 0(select command register, for write); Busy flag (address counter for read); RS = 1 select data register (for write)
5	R/W	Read/Write	Signal to read/ write data from/to LCD. RW = 0 (Write to LCD); RW=1 (read from LCD)
6	E	Enable(strobe)	High to low pulse is applied to this pin to enable LCD to accept (latch) data/ command present on its data lines D0-D7
7-14	D0-D7	Data lines DO (Pin7-LSB), D7 (Pin14-MSB)	Bidirectional data lines used to send data/command to LCD; or Read LCD internal registers, D7 is also used as a busy flag, D4-D7 are used in the 4 bit operation
15 and 16	LED + LED-	Backlight+ Backlight-	

Liquid Crystal Display (LCD)

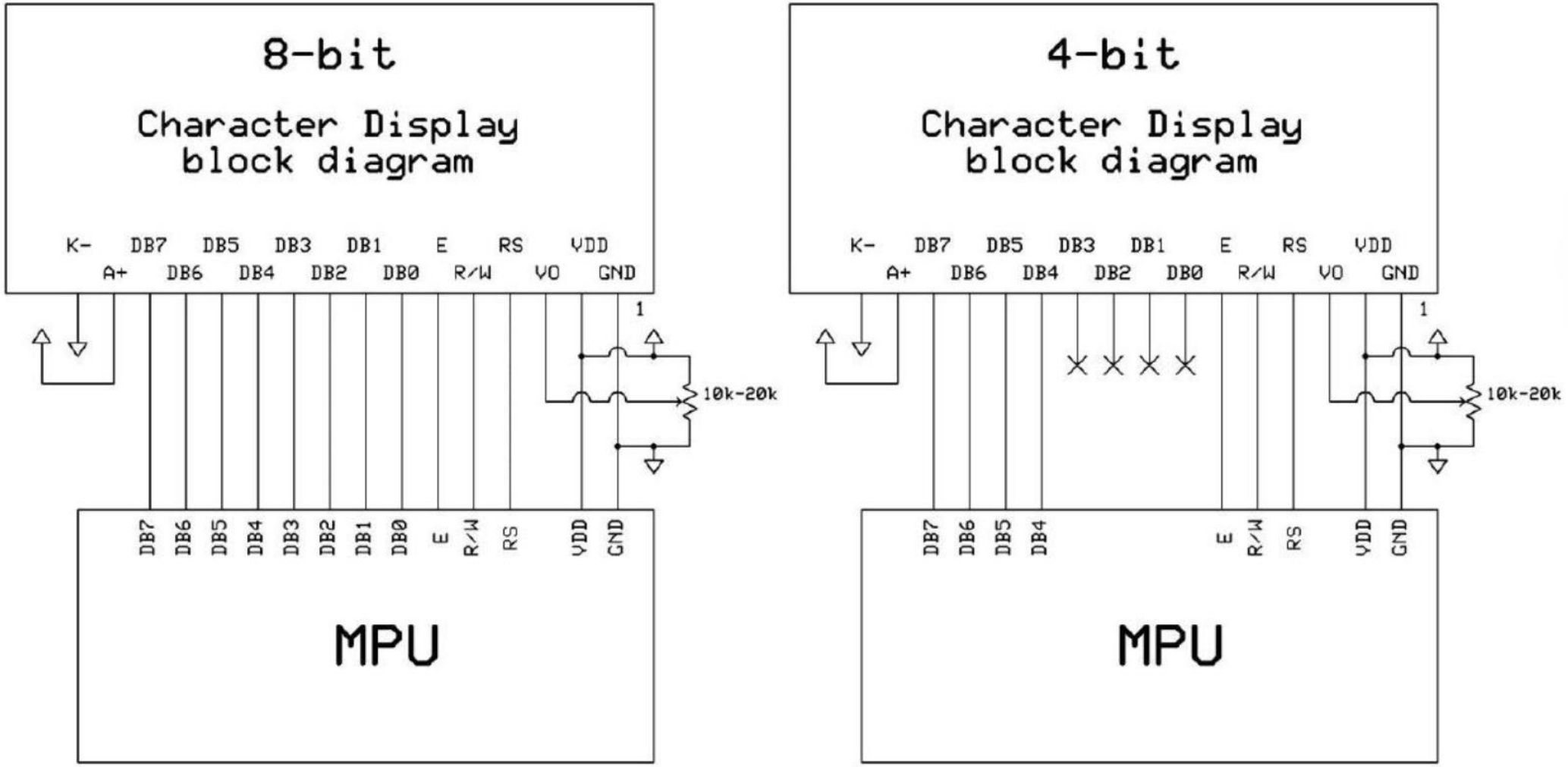
- 4- or 8-bit mode of operation.
- One 8-character line or two 8-character lines.
- Low power supply (2.7 V to 5.5 V).
- 5 x 8 or 5 x 10 dot matrix configuration.
- LCD layout and cursor profile of the LCD in the 8051microcontroller lab kit.





Liquid Crystal Display (LCD)

- 8-bit and 4-bit operations of the LCD.



Liquid Crystal Display (LCD)

Commands to Send Data to LCD Interfacing with 8051

To initialise the LCD with the 8051 microcontroller, the following instructions are to be executed:

Commands	Function
0x38	This is used for 8-bit data initialisation
0x0C	display on, cursor off
0x01	for clearing screen
0x80	force cursor to move to the beginning of 1stline

Liquid Crystal Display (LCD)

Sending Data Command

For displaying any character, whether it is a number, alphabet, or character, we will follow these steps:

- For display data, register select (RS pin) should be high, RS = 1.
- Place a data byte on the data register.
- Pulse the Enable pin (EN pin) from high to low.
- Configure the R/W to write mode. For write mode, R/W = 0.
- Repeat the above steps to send more data.

Liquid Crystal Display (LCD)

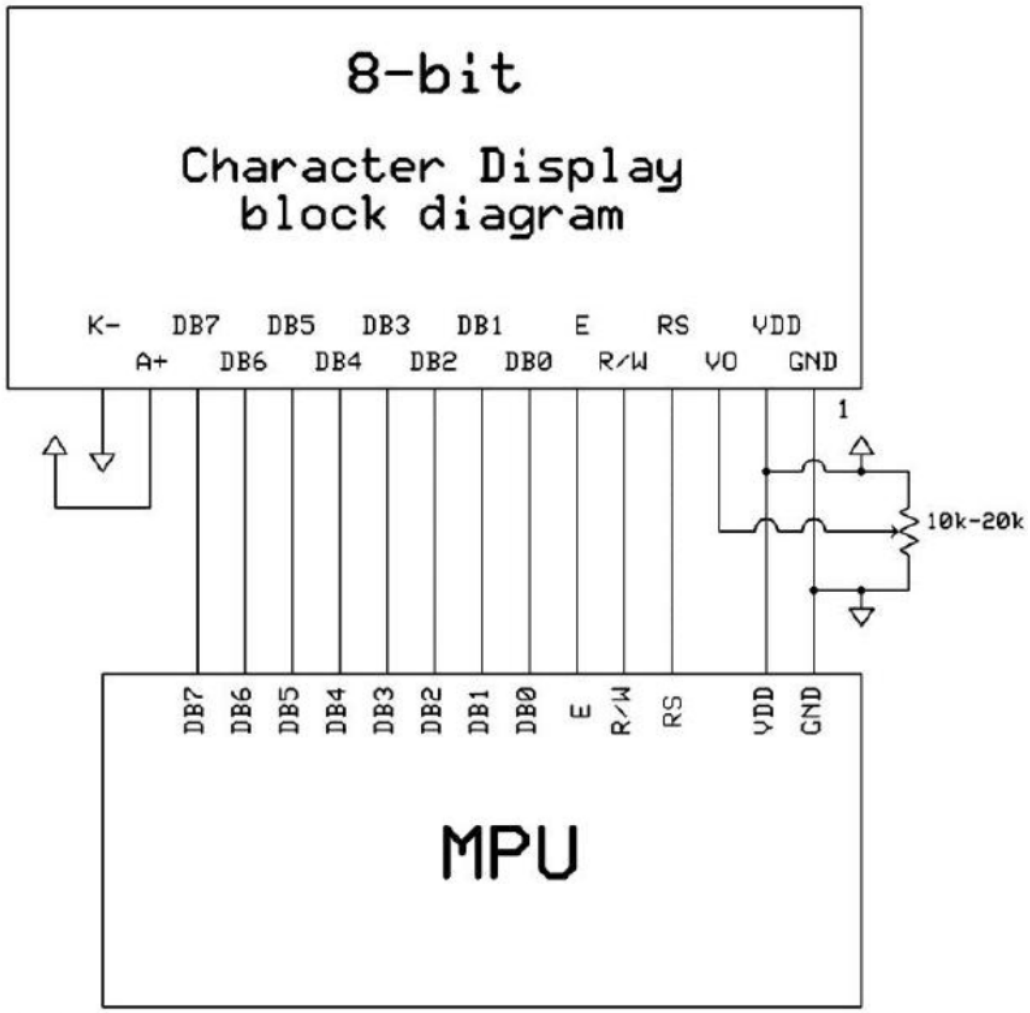
To Send Command

To instruct LCD interfacing with the 8051 microcontroller to perform a specific task, for example, displaying characters in the second row instead of the first, for this the steps are as follows:

- For command mode, the register select (RS pin) should be low, RS = 0.
- Place the data byte on the command register.
- Pulse the Enable pin (EN pin) from high to low.
- Read/Write should be low (write mode), R/W = 0.
- Repeat the above steps to send another command.

Liquid Crystal Display (LCD)

- 8-bit operations of the LCD.



In 8-bit mode, the LCD uses:
`DB0 – DB7`
All 8 data pins are connected to the 8051.

Connection

DB0–DB7 → P2.0–P2.7

LCD Pins	Connected To
DB0–DB7	8051 Port
RS	Control pin
RW	Control pin
EN	Control pin

Liquid Crystal Display (LCD)

- 8-bit operations of the LCD.

How 8-Bit Mode Works

Entire 8-bit data is sent at one time.

Example:

```
MOV P2,#41H
```

Binary:

01000001

LCD receives all 8 bits together.

ASCII 41H = A

LCD displays:

A

Liquid Crystal Display (LCD)

- 8-bit operations of the LCD.

Data Transfer in 8-Bit Mode

Suppose command:

38H

Binary:

00111000

Transfer:

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	1	1	0	0	0

Liquid Crystal Display (LCD)

- 8-bit operations of the LCD.

Advantage	**Explanation**
Faster	Full byte transferred once
Easier programming	Simpler logic
Good for beginners	Easy to understand

Disadvantage	**Explanation**
Uses more pins	8 data pins required
More wiring	Larger circuit

Liquid Crystal Display (LCD)

- 8-bit operations of the LCD.

Example Initialisation in 8-Bit Mode

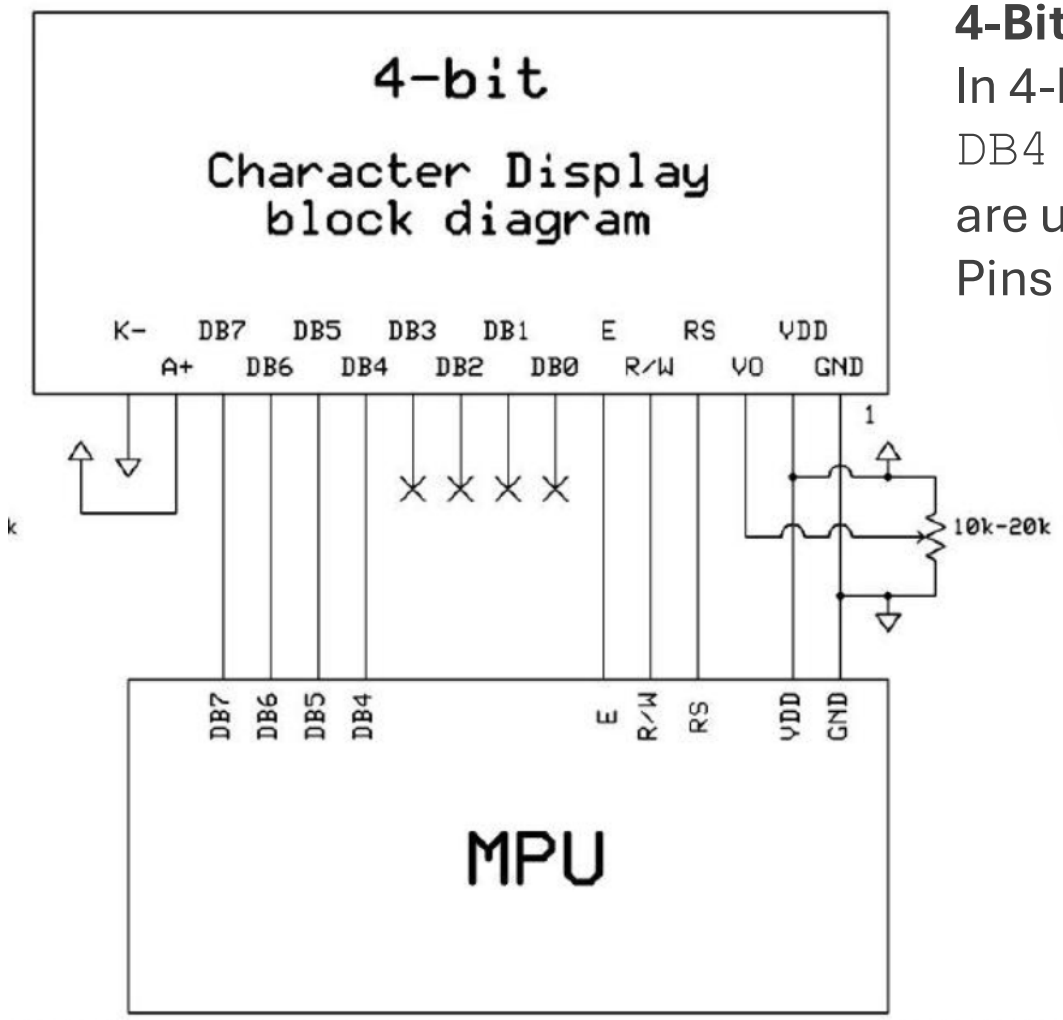
```
MOV A,#38H
ACALL CMD
```

Meaning:

- 8-bit mode
- 2-line display
- 5×7 font

Liquid Crystal Display (LCD)

- 4-bit operations of the LCD.



4-Bit LCD Operation

In 4-bit mode, only:

`DB4 – DB7`

are used.

Pins DB0–DB3 are not connected.

Connection in 4-Bit Mode

LCD Pins	Connected?
DB0–DB3	No
DB4–DB7	Yes

Liquid Crystal Display (LCD)

- 4-bit operations of the LCD.

How 4-Bit Mode Works
An 8-bit value is divided into:
1.Upper nibble
2.Lower nibble
Each nibble = 4 bits.

Example
Suppose:
41H
Binary: 0100 0001

LCD receives:
1.First upper nibble
2.Then lower nibble

Split into:

Nibble	Value
Upper	0100
Lower	0001

Liquid Crystal Display (LCD)

- 4-bit operations of the LCD.

Step-by-Step Transfer
Send Upper Nibble
0100
through:
DB4–DB7
Pulse EN.

Send Lower Nibble
0001
Pulse EN again.
LCD combines both parts internally.

Final data:
01000001 = 41H

Liquid Crystal Display (LCD)

- 4-bit operations of the LCD.

Why Use 4-Bit Mode?

Main reason:

Save microcontroller pins

Very useful when:

•Many devices connected

•Limited I/O ports available

Advantage	Explanation
Uses fewer pins	Only 4 data pins
Smaller wiring	Compact design
Saves ports	More pins for other devices

Disadvantage	Explanation
Slower	Data sent twice
More programming	Extra steps needed

Liquid Crystal Display (LCD)

- 4-bit operations of the LCD.

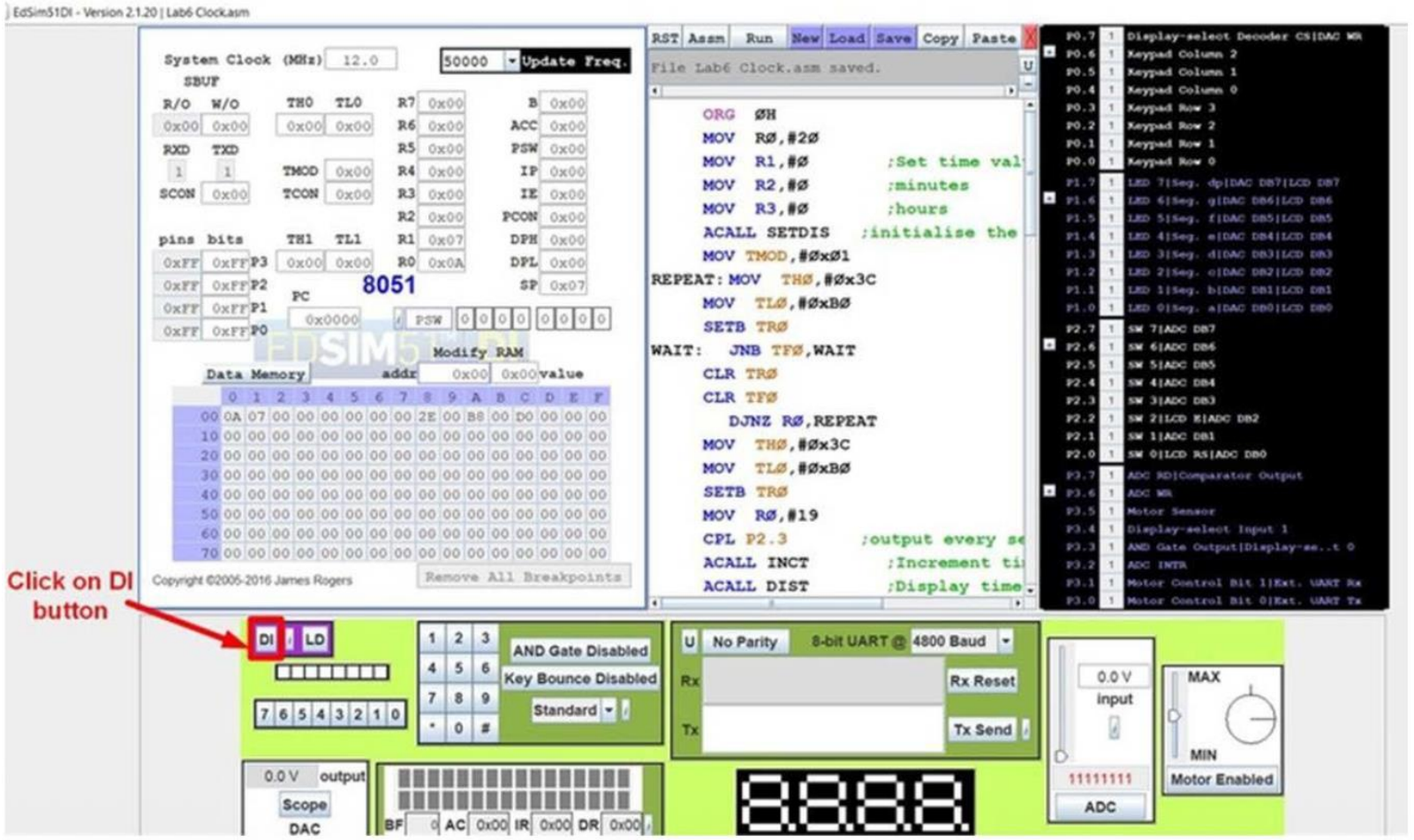
Example of 4-Bit Sending
Suppose command:
28H

Binary:
0010 1000

Send:
1.0010
2.1000
separately.

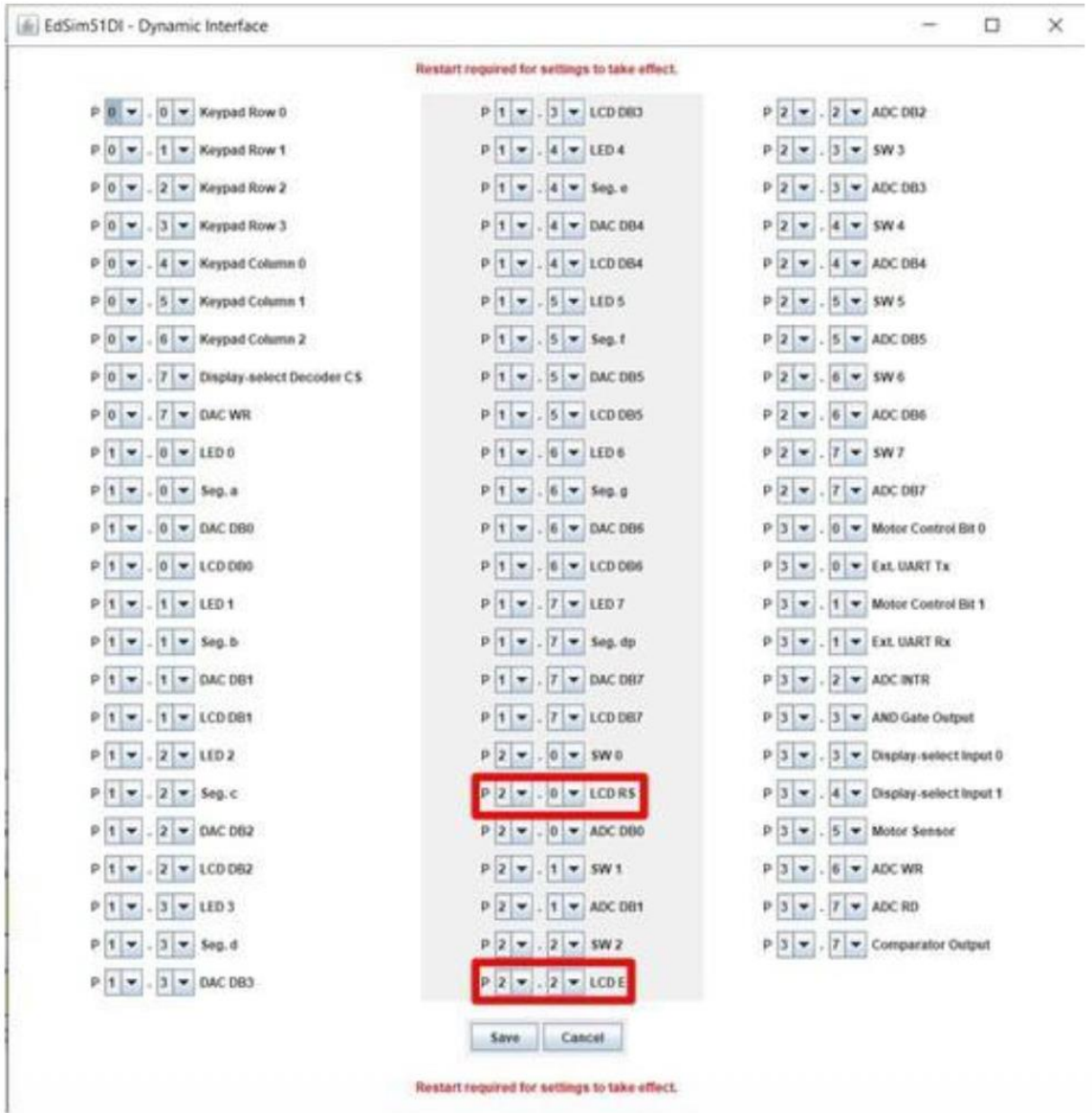
Using Simulator for LCD Interfacing

- In the lab exercise, we used the LCD in 4-bit Mode.
- To use the simulator for the reset of labs, the LCD must be in 8-bit Mode.
- To change mode, RS and E should be remapped to other port pins, using the DI button at the top left of the peripheral panel.



Using Simulator for LCD Interfacing

- Save the new settings and restart the program to see the changes



LCD Commands

Command	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	Command Code (Hex)	Execution Time (Max.) f_{cp} = 250 kHz
Clear display	0	0	0	0	0	0	0	0	0	1	01	1.64 ms
Cursor home	0	0	0	0	0	0	0	0	1	x	02	1.64 ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	04–Shift cursor left	40 µs
											06–Shift cursor right	
											05–Shift display right	
											07–Shift display left	
Display on/off control	0	0	0	0	0	0	1	D	U	B	08–Display off, Cursor off	40 µs
											0A–Display off, Cursor on	
											0C–Display on, Cursor off	
											0E–Display on, Cursor blink off	
											0F–Display on, Cursor blink	
Cursor/Display Shift	0	0	0	0	0	1	D/C	R/L	x	x	10–Shift cursor left	40 µs
											14–Shift cursor right	
											18–Shift display left	
											1C–Shift display right	
Function set	0	0	0	0	1	DL	N	F	x	x	28–2line,5X7matrix,4 line	40 µs
											38–2line,5X7matrix,8 line	40 µs
Set CGRAM address	0	0	0	1	CGRAM address							40 µs
Set DDRAM address	0	0	1	DDRAM address							80– Set cursor at beginning of line 1	40 µs
Read "BUSY" flag (BF)	0	1	BF	DDRAM address								40 µs
Write to CGRAM or DDRAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0		40 µs
Read from CGRAM or DDRAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0		40 µs

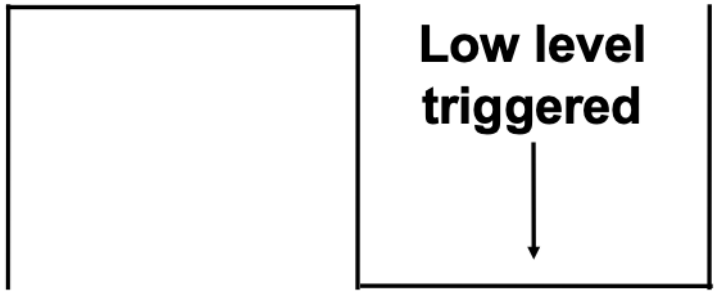
I/D 1 = Increment (by 1)	0 = Decrement (by 1)	R/L 1 = Shift right	0 = Shift left
S 1 = Display shift on	0 = Display shift off	DL 1 = 8-bit interface	0 = 4-bit interface
D 1 = Display on	0 = Display off	N1= Display in two lines	0 = Display in one line
U 1 = Cursor on	0 = Cursor off	F 1 = Character format 5 × 10 dots	0 = 5×7 dots
B 1 = Cursor blink on	0 = Cursor blink off	D/C 1 = Display shift	0 = Cursor shift

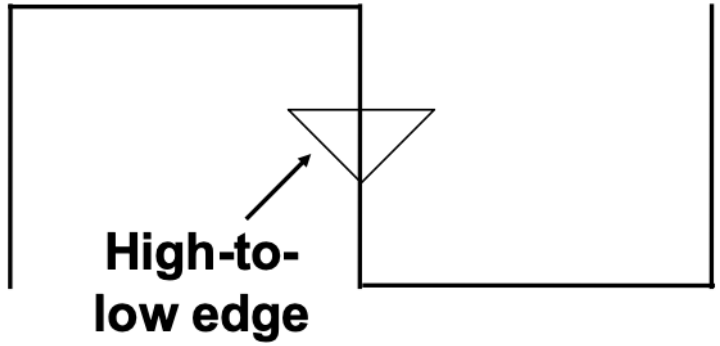
LCD Commands

Instruction	Instruction code										Description	Execution time (f_{osc}= 270 KHZ
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM address to "00H" from AC	1.52ms
Return Home	0	0	0	0	0	0	0	0	1	-	Set DDRAM Address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.52ms
Entry mode Set	0	0	0	0	0	0	0	1	I/D	SH	Sets cursor move direction and specifies display shift. These parameters are performed during data write and read.	37μs
Display ON/ OFF control	0	0	0	0	0	0	1	D	C	B	D=1: Entire display on C=1: Cursor on B=1: Blinking cursor on	37μs
Cursor or Display shift	0	0	0	0	0	1	S/C	R/L	-	-	Sets cursor moving and display shift control bit, and the direction without changing DDRAM data.	37μs
Function set	0	0	0	0	1	DL	N	F	-	-	DL: Interface data is 8/4 bits N: Number of lines is 2/1 F: Font size is 5x11/5x8	37μs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter	37μs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	37μs
Read busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0s
Write data To Address	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	37μs
Read data From RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	37μs

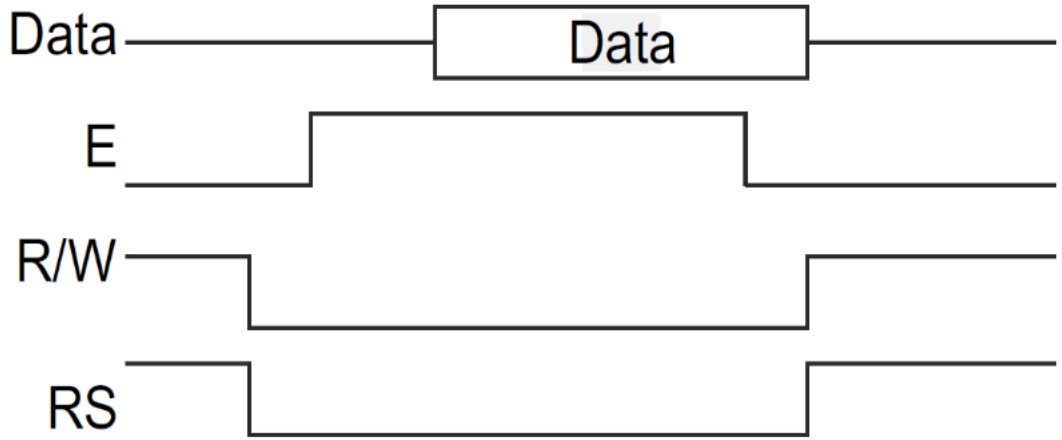
LCD Signal Timing

Level triggering for command and data interfacing



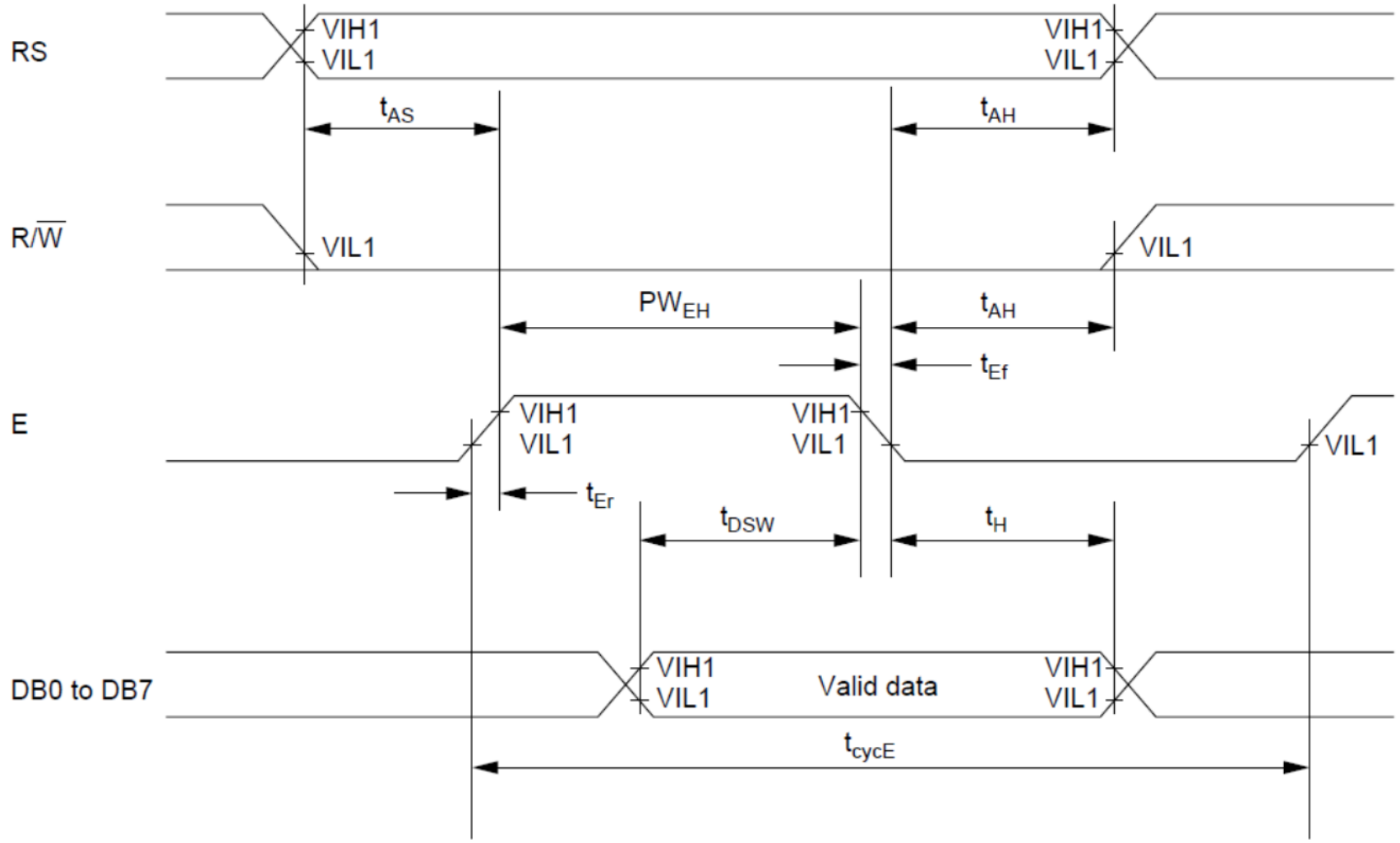


Command and data interfacing with the LCD



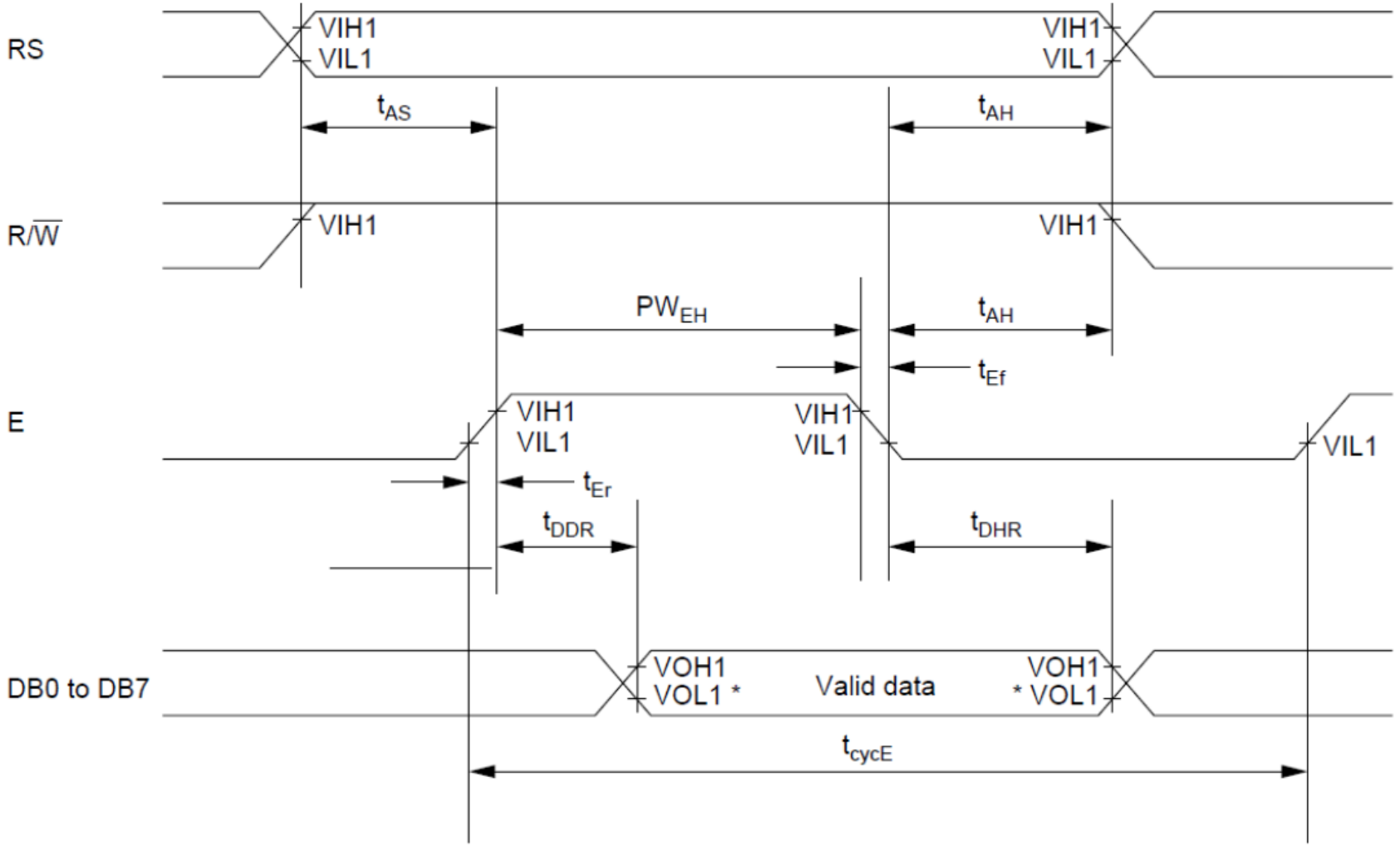
LCD Write Signal Timing

Write operation timing characteristics



LCD Read Signal Timing

Read operation timing characteristics



Note: * VOL1 is assumed to be 0.8 V at 2 MHz operation.

Initialise Set Display of LCD

Initialise the LCD by sending a set of commands to it:

1. Configure the data bus as a 4-bit or 8-bit mode.
2. Select character font (i.e., dots/character).
3. Configure display and cursor type, i.e. display ON or not, cursor blinking or not.
4. Configure display and cursor movement, i.e. left shift or right shift or shift off.
5. Configure display position.
6. Clear display.

The commands are sent by making RS = 0 and R/W = 0, and high to low pulse on E pin, a delay should be provided between two commands to ensure that the LCD has executed the previous command.

Example 1: Initialise Display of LCD

Write an assembly-language subroutine to initialise the LCD display.

```
SETDIS:                 ;display initialisation
MOV A,#38H              ;function set: 8 bits, 2 lines, 5x7 cursor
ACALL COMNWRT
ACALL DELAY
MOV A,#0EH              ;display ctrl: display on, cursor off, no blink
ACALL COMNWRT
ACALL DELAY
MOV A,#01H              ;clear display
ACALL COMNWRT
ACALL DELAY
MOV A,#06H              ;entry mode set: shift cursor right
ACALL COMNWRT
ACALL DELAY
MOV A,#80H              ;set cursor at beginning of line 1
ACALL COMNWRT
ACALL DELAY
RET
```

SetDisplay.asm

Write Command to LCD

- The commands are sent by making RS = 0 and R/W = 0, and a high-to-low pulse on the E pin.
- A delay should be provided between two commands to ensure that the LCD has executed the previous command.

Write Command to LCD

For a standard **16×2 LCD**, each display position has a memory address in the LCD's DDRAM.

First Line

To move the cursor to the **beginning of the first line**:

```
MOV A,#80H
ACALL CMD
```

80H = First row, first column

Second Line

To move the cursor to the **beginning of the second line**:

```
MOV A,#0C0H
ACALL CMD
```

C0H = Second row, first column

Write Command to LCD

Why 80H and C0H?

The LCD command format for setting the cursor is:

1xxxxxxx

The highest bit (bit 7) is 1, indicating a "Set DDRAM Address" command.

Position	DDRAM Address	Command
Line 1, Column 1	00H	80H
Line 1, Column 2	01H	81H
Line 1, Column 3	02H	82H
...	...	...
Line 2, Column 1	40H	C0H
Line 2, Column 2	41H	C1H
Line 2, Column 3	42H	C2H

Write Command to LCD

Why 80H and C0H?

The LCD command format for setting the cursor is:

1xxxxxxx

The highest bit (bit 7) is 1, indicating a "Set DDRAM Address" command.

The command is:

80H + DDRAM Address

Examples:

- First line, first column:
- 80H + 00H = 80H

- Second line, first column:
- 80H + 40H = C0H

Example 2: Write Command to LCD

Write an assembly-language subroutine to send a command to the LCD display.

```
COMNWRT:      ;send command to LCD
MOV P1,A      ;copy reg A to Port 1
CLR P2.0      ;RS=0 for command
CLR P2.1      ;R/W/=0 for write
SETB P2.2     ;E=1 for high pulse
ACALL DELAY   ;give LCD some time
CLR P2.2      ;E=0 for H-to-L pulse
RET

DELAY:
MOV R5,#30    ;short delay

LP1:
DJNZ R5,LP1
RET
```

WriteCommand.asm

Write Data to LCD

- Data characters are sent to the LCD by making RS = 1 and R/W = 0, and high to low pulse on E pin.
- Apply a delay between two consecutive data characters.
- Send ASCII values of the characters to be displayed one character at a time with a delay between them.

Example 3: Write Data to LCD

Write an assembly-language subroutine to send a data to the LCD display.

```
DataWRT:      ;write data to LCD
MOV P1,A      ;copy reg A to Port 1
CLR P2.0      ;RS=1 for command
CLR P2.1      ;R/W/=0 for write
SETB P2.2     ;E=1 for high pulse
ACALL DELAY   ;give LCD some time
CLR P2.2      ;E=0 for H-to-L pulse
RET

DELAY:
MOV R5,#30    ;short delay

LP1:
DJNZ R5,LP1
RET
```

WriteData.asm

Example 4: Write Characters to LCD

Write an assembly-language program to write characters to the LCD:

- Initialise the LCD.
- Write HI to the LCD.

We will use previous LCD interfacing programs e.g. SetDisplay.asm, WriteCommand.asm, WriteData.asm as subroutines in this program.

```
ORG 00H                ;start program
LJMP MAIN              ;jump to main program
ORG 30H                ;bypass interrupt vectors table
MAIN:
ACALL SETDIS           ;initialise LCD
MOV A, #'H'            ;write character H
ACALL DATAWRT          ;write data to LCD
MOV A, #'I'            ;write character I
ACALL DATAWRT          ;write data to LCD
AGAIN: SJMP AGAIN      ;endless loop for nothing
```

WriteCharacters.asm – part 1

Example 4: Write Characters to LCD

```
SETDIS:                 ;display initialisation
MOV A,#38H              ;function set: 8 bits, 2 lines, 5x7 cursor
ACALL COMNWRT
ACALL DELAY
MOV A,#0EH              ;display ctrl: display on, cursor off, no blink
ACALL COMNWRT
ACALL DELAY
MOV A,#01H              ;clear display
ACALL COMNWRT
ACALL DELAY
MOV A,#06H              ;entry mode set: shift cursor right
ACALL COMNWRT
ACALL DELAY
MOV A,#80H              ;set cursor at beginning of line 1
ACALL COMNWRT
ACALL DELAY
RET
```

WriteCharacters.asm – part 2

Example 4: Write Characters to LCD

```
COMNWRT:      ;send command to LCD
MOV P1,A      ;copy reg A to Port 1
CLR P2.0      ;RS=0 for command
CLR P2.1      ;R/W/=0 for write
SETB P2.2     ;E=1 for high pulse
ACALL DELAY   ;give LCD some time
CLR P2.2      ;E=0 for H-to-L pulse
RET

DataWRT:      ;write data to LCD
MOV P1,A      ;copy reg A to Port 1
CLR P2.0      ;RS=1 for command
CLR P2.1      ;R/W/=0 for write
SETB P2.2     ;E=1 for high pulse
ACALL DELAY   ;give LCD some time
CLR P2.2      ;E=0 for H-to-L pulse
RET
```

WriteCharacters.asm – part 3

Example 4: Write Characters to LCD

```
DELAY:
MOV R5,#30    ;short delay
LP1:
DJNZ R5,LP1
RET
```

WriteCharacters.asm – part 4

Example 5: Write Characters to LCD

Explain the function of the following LCD subroutine:

```
DATA:
SETB RS
CLR RW
SETB EN
CLR EN
ACALL DELAY
RET
```

Example 5: Write Characters to LCD

Instruction	Function
SETB RS	Select data register
CLR RW	Select write operation
SETB EN	Enable HIGH
CLR EN	Enable LOW to latch data
ACALL DELAY	Wait for LCD processing
RET	Return from subroutine

Example 5: Write Characters to LCD

Step-by-Step Explanation

1. DATA:

```
DATA:
```

This is a **label** (subroutine name).

The program can jump here using:

```
ACALL DATA
```

Meaning: Execute the DATA subroutine.

2. SETB RS

```
SETB RS
```

Meaning

Set RS pin to logic 1.

```
RS = 1
```

RS determines whether LCD receives:

RS	Mode
0	Command
1	Data

Example 5: Write Characters to LCD

3. CLR RW

```
CLR RW
```

Meaning

Clear RW pin.

RW = 0

Why?

RW selects:

Read (1)

Write (0)

4. SETB EN

```
SETB EN
```

Meaning

Set Enable pin HIGH.

EN = 1

Why?

EN tells LCD:

“Get ready to receive data.”

This begins the enable pulse.

5. CLR EN

```
CLR EN
```

Meaning

Set Enable LOW.

EN = 0

Why?

The LCD reads the data during the HIGH-to-LOW transition of EN.

This pulse tells LCD:

“Now accept the data.”

Example 5: Write Characters to LCD

6. ACALL DELAY

```
ACALL DELAY
```

Meaning

Call the delay subroutine.

Why is the delay needed?

LCD is slower than the 8051.

After receiving data, LCD needs time to:

- Process character
- Store it
- Display it

Without delay:

```
LCD may miss data
```

Example Delay Subroutine

```
DELAY:
MOV R7,#255

D1:
DJNZ R7,D1

RET
```

This creates a small waiting time.

Example 5: Write Characters to LCD

7. RET

```
RET
```

Meaning

Return to the main program.

After finishing DATA subroutine:

- The program goes back
- Continues after ACALL DATA

Example 5: Write Characters to LCD

Full Working Example

Suppose:

```
MOV A,#'H'
ACALL DATA
```

What Happens Internally?

Step 1

Accumulator contains:

'H' = 48H

Step 2

DATA subroutine starts.

Step 3

```
SETB RS
```

LCD knows:

Incoming value is character data

Step 4

```
CLR RW
```

LCD knows: 8051 is writing data

Step 5

```
SETB EN
CLR EN
```

Enable pulse generated.

LCD captures: 48H

Step 6

LCD converts ASCII: 48H → H

Character appears on screen.

Step 7

```
RET
```

Program returns.

Example 5: Write Characters to LCD

Full Working Example

Suppose:

```
MOV A,#'H'
ACALL DATA
```

What Happens Internally?

Step 1

Accumulator contains:

'H' = 48H

Step 2

DATA subroutine starts.

Step 3

```
SETB RS
```

LCD knows:

Incoming value is character data

Step 4

```
CLR RW
```

LCD knows: 8051 is writing data

Step 5

```
SETB EN
CLR EN
```

Enable pulse generated.

LCD captures: 48H

Step 6

LCD converts ASCII: 48H → H

Character appears on screen.

Step 7

```
RET
```

Program returns.

Review

Question 1
What is the function of the RS pin in an LCD?

Question 2
Write an assembly instruction to clear the LCD screen.

Question 3
What is the function of the Enable (EN) pin in LCD interfacing?

Question 4
Write assembly code to display the character 'A' on the LCD.

Question 5
What is the difference between 4-bit mode and 8-bit mode LCD operation?

Review

Question 1 Answer:

RS (Register Select) selects the LCD register:

RS = 0 → Command register

RS = 1 → Data register

Q2 Answer:

```
MOV A,#01H
ACALL CMD
```

Question 3 Answer:

The EN pin generates a pulse that tells the LCD to read and accept the command or data on its pins.

Question 4 Answer:

```
MOV A,#'A'
ACALL DATA
```

Question 5 Answer:

- 8-bit mode uses DB0–DB7 and transfers data in one step.
- 4-bit mode uses DB4–DB7 and transfers data in two steps.

Review

Question 6
What does this code do?

```
MOV A,#80H
ACALL CMD
```

Question 7
Write assembly code to move the cursor to the beginning of the second line.

Question 8
What is the output of the following code?

```
MOV A,#35H
ACALL DATA
```

Question 9
Write assembly code to initialise the LCD in 8-bit, 2-line mode.

Review

Question 6 Answer:
Moves the cursor to the beginning of the first line of the LCD.

Question 7 Answer:

```
MOV A,#0C0H
ACALL CMD
```

Question 8 Answer:
The LCD displays: 5
because ASCII code 35H represents the character '5'.

Question 9 Answer:

```
MOV A,#38H
ACALL CMD
```

Explanation:
38H configures the LCD for:
8-bit interface
2 display lines
5×7 character font

Review

Question 10

Write a complete program to display “HI” on LCD.

Review

Question 10

Write a complete program to display “HI” on LCD.

```
RS BIT P3.0
RW BIT P3.1
EN BIT P3.2

ORG 0000H

; Initialise LCD
MOV A,#38H
ACALL CMD

MOV A,#0EH
ACALL CMD

MOV A,#01H
ACALL CMD

MOV A,#06H
ACALL CMD

; Display H
MOV A,#'H'
ACALL DATA

; Display I
MOV A,#'I'
ACALL DATA

HERE: SJMP HERE
```

Review

Question 10

Write a complete program to display “HI” on LCD.

```
;-----------------
CMD:
MOV P2,A
CLR RS
CLR RW
SETB EN
CLR EN
ACALL DELAY
RET

;-----------------
DELAY:
MOV R7,#255
D1: DJNZ R7,D1
RET

END
```

```
;---------------
DATA:
MOV P2,A
SETB RS
CLR RW
SETB EN
CLR EN
ACALL DELAY
RET
```

Review

Question 11

Use the assembly code from Q10: `CMD` and `DATA`

`Write assembly code to show “OK”`

Review

Question 11

Use the assembly code from Q10: `CMD` and `DATA`
move cursor to second line and display “OK”.

```
MOV A,#0C0H
ACALL CMD

MOV A,#'O'
ACALL DATA

MOV A,#'K'
ACALL DATA
```