**TASK\_14\_4 (**Write a driver to control a 16x2 LCD using an 8 bit interface (using eight data lines). Implement functions like lcd\_init, lcd\_set\_cursor(row, col), lcd\_write(row, col), lcd\_clear**)**

**Project Structure:**

The project consists of three main files:

1. LCD.h - Header file
2. LCD.c - Source file with function implementations
3. T\_14\_4.c - Main file containing the main function

**1. LCD.h**

This header file contains the function prototypes, macro definitions, and pin configurations for controlling the LCD.

|  |
| --- |
| #ifndef LCD\_H |
| #define LCD\_H |
| #include <reg51.h> *// Include the header file for the 8051 microcontroller* |
| *// Define the register select, read/write, and enable pins* |
| sbit rs **=** P2^6; *// RS pin is connected to pin 6 of port 2* |
| sbit rw **=** P2^5; *// RW pin is connected to pin 5 of port 2* |
| sbit en **=** P2^7; *// EN pin is connected to pin 7 of port 2* |
| *// Define the data port for the LCD* |
| #define text P0 *// Data port is connected to port 0* |
| *// Function prototypes* |
| void lcd\_init(); *// Function to initialize the LCD* |
| void lcd\_cmd(unsigned char cmd); *// Function to send a command to the LCD* |
| void lcd\_data(unsigned char dat); *// Function to send data to the LCD* |
| void lcd\_set\_cursor(unsigned char row, unsigned char col); *// Function to set the cursor position* |
| void lcd\_write(unsigned char row, unsigned char col, unsigned char dat); *// Function to write data at a specific position* |
| void lcd\_clear(); *// Function to clear the display* |
| #endif *// LCD\_H* |

**Explanation:**

* **#ifndef LCD\_H / #define LCD\_H / #endif:** These lines prevent multiple inclusions of the header file.
* **#include <reg51.h>:** This includes the header file for the 8051 microcontroller.
* **Pin Definitions:**
  + sbit rs = P2^6;: The RS (Register Select) pin is connected to bit 6 of port 2.
  + sbit rw = P2^5;: The RW (Read/Write) pin is connected to bit 5 of port 2.
  + sbit en = P2^7;: The EN (Enable) pin is connected to bit 7 of port 2.
* **#define text P0:** The data port for the LCD is defined as port 0.
* **Function Prototypes:** These are the declarations of the functions that will be implemented in LCD.c.

**2. LCD.c**

This source file contains the implementation of the functions declared in LCD.h.

|  |
| --- |
| #include "LCD.h" |
| *// Delay function to create a short delay* |
| void delay() { |
| int i, j; |
| **for**(i **=** 0; i **<** 100; i**++**) { *// Outer loop* |
| **for** (j **=** 0; j **<** 100; j**++**); *// Inner loop* |
| } |
| } |
| *// Define lcd\_delay function to create a delay for LCD timing* |
| void lcd\_delay() { |
| delay(); *// Call the delay function*} |
| *// Function to send a command to the LCD* |
| void lcd\_cmd(unsigned char cmd) { |
| text **=** cmd; *// Put the command on the data port* |
| rs **=** 0; *// Select command register (RS=0)* |
| rw **=** 0; *// Select write operation (RW=0)* |
| en **=** 1; *// Generate a high-to-low pulse on the enable pin* |
| lcd\_delay(); *// Wait for the command to be processed* |
| en **=** 0; *// End the enable pulse* |
| } |
| *// Function to send data to the LCD* |
| void lcd\_data(unsigned char dat) { |
| text **=** dat; *// Put the data on the data port* |
| rs **=** 1; *// Select data register (RS=1)* |
| rw **=** 0; *// Select write operation (RW=0)* |
| en **=** 1; *// Generate a high-to-low pulse on the enable pin* |
| lcd\_delay(); *// Wait for the data to be processed* |
| en **=** 0; *// End the enable pulse* |
| } |
| *// Function to initialize the LCD* |
| void lcd\_init() { |
| lcd\_cmd(0x38); *// Set LCD to 8-bit mode, 2 lines, 5x7 matrix* |
| lcd\_delay(); *// Wait for the command to be processed* |
| lcd\_cmd(0x0C); *// Display ON, cursor OFF* |
| lcd\_delay(); *// Wait for the command to be processed* |
| lcd\_cmd(0x06); *// Entry mode, auto increment with no shift* |
| lcd\_delay(); *// Wait for the command to be processed* |
| lcd\_cmd(0x01); *// Clear the display* |
| lcd\_delay(); *// Wait for the command to be processed* |
| } |
| *// Function to set the cursor position* |
| void lcd\_set\_cursor(unsigned char row, unsigned char col) { |
| unsigned char pos; |
| **if** (row **==** 0) |
| pos **=** 0x80 **+** col; *// Set cursor position for the first line* |
| **else** **if** (row **==** 1) |
| pos **=** 0xC0 **+** col; *// Set cursor position for the second line* |
| lcd\_cmd(pos); *// Send command to set the cursor position* |
| } |
| *// Function to write data at a specific position* |
| void lcd\_write(unsigned char row, unsigned char col, unsigned char dat) { |
| lcd\_set\_cursor(row, col); *// Set cursor to specified position* |
| lcd\_data(dat); *// Write data at that position* |
| } |
| *// Function to clear the display* |
| void lcd\_clear() { |
| lcd\_cmd(0x01); *// Clear display screen* |
| lcd\_delay(); *// Wait for the command to be processed* |
| } |

**Explanation:**

* **#include "LCD.h":** This includes the header file where function prototypes and pin configurations are defined.
* **Delay Function:**
  + void delay(): This function creates a delay by using nested loops.
* **void lcd\_delay():** This function calls the delay() function to create an appropriate delay for LCD timing.
* **Command Function:**
  + void lcd\_cmd(unsigned char cmd): This function sends a command to the LCD.
    - text = cmd;: Places the command on the data port.
    - rs = 0;: Sets RS to 0 to select the command register.
    - rw = 0;: Sets RW to 0 to select the write operation.
    - en = 1;: Generates a high-to-low pulse on the enable pin.
    - lcd\_delay();: Waits for the command to be processed.
    - en = 0;: Ends the enable pulse.
* **Data Function:**
  + void lcd\_data(unsigned char dat): This function sends data to the LCD.
    - text = dat;: Places the data on the data port.
    - rs = 1;: Sets RS to 1 to select the data register.
    - rw = 0;: Sets RW to 0 to select the write operation.
    - en = 1;: Generates a high-to-low pulse on the enable pin.
    - lcd\_delay();: Waits for the data to be processed.
    - en = 0;: Ends the enable pulse.
* **Initialization Function:**
  + void lcd\_init(): This function initializes the LCD.
    - lcd\_cmd(0x38);: Sets the LCD to 8-bit mode, 2 lines, 5x7 matrix.
    - lcd\_cmd(0x0C);: Turns on the display, cursor off.
    - lcd\_cmd(0x06);: Sets the entry mode, auto increment with no shift.
    - lcd\_cmd(0x01);: Clears the display.
* **Cursor Function:**
  + void lcd\_set\_cursor(unsigned char row, unsigned char col): This function sets the cursor position on the LCD.
    - if (row == 0) pos = 0x80 + col;: Sets cursor position for the first line.
    - else if (row == 1) pos = 0xC0 + col;: Sets cursor position for the second line.
    - lcd\_cmd(pos);: Sends the command to set the cursor position.
* **Write Function:**
  + void lcd\_write(unsigned char row, unsigned char col, unsigned char dat): This function writes data to a specific position on the LCD.
    - lcd\_set\_cursor(row, col);: Sets the cursor to the specified position.
    - lcd\_data(dat);: Writes data at that position.
* **Clear Function:**
  + void lcd\_clear(): This function clears the display.
    - lcd\_cmd(0x01);: Sends the command to clear the display.
    - lcd\_delay();: Waits for the command to be processed.

**3. T\_14\_4.c**

This main file contains the main function demonstrating how to use the LCD functions.

|  |
| --- |
| #include "LCD.h" *// Include the LCD header file* |
| int main() { |
| **while**(1){ |
| lcd\_init(); *// Initialize the LCD* |
| lcd\_clear(); *// Clear the LCD* |
| *// Example usage* |
| lcd\_write(0, 0, 'H'); *// Write 'H' at the beginning of the first line* |
| lcd\_write(0, 1, 'e'); *// Write 'e' at the second position of the first* line |
| lcd\_write(0, 2, 'l'); *// Write 'l' at the third position of the first line* |
| lcd\_write(0, 3, 'l'); *// Write 'l' at the fourth position of the first line* |
| lcd\_write(0, 4, 'o'); *// Write 'o' at the fifth position of the first line* |
| lcd\_set\_cursor(1, 0); *// Set cursor to the beginning of the second line* |
| lcd\_write(1, 0, 'W'); *// Write 'W' at the beginning of the second line* |
| lcd\_write(1, 1, 'o'); *// Write 'o' at the second position of the second line* |
| lcd\_write(1, 2, 'r'); *// Write 'r' at the third position of the second line* |
| lcd\_write(1, 3, 'l'); *// Write 'l' at the fourth position of the second line* |
| lcd\_write(1, 4, 'd'); *// Write 'd' at the fifth position of the second line* |
| *// while(1); // Infinite loop to keep the microcontroller running* |
| } |
| } |

**Explanation:**

* **#include "LCD.h":** This includes the header file where function prototypes and pin configurations are defined.
* **int main():** The main function where the program execution begins.
  + **while(1):** An infinite loop to keep the microcontroller running.
  + **lcd\_init();:** Initializes the LCD.
  + **lcd\_clear();:** Clears the LCD display.
  + **Example Usage:**
    - lcd\_write(0, 0, 'H');: Writes 'H' at the beginning of the first line.
    - lcd\_write(0, 1, 'e');: Writes 'e' at the second position of the first line.
    - lcd\_write(0, 2, 'l');: Writes 'l' at the third position of the first line.
    - lcd\_write(0, 3, 'l');: Writes 'l' at the fourth position of the first line.
    - lcd\_write(0, 4, 'o');: Writes 'o' at the fifth position of the first line.
    - lcd\_set\_cursor(1, 0);: Sets cursor to the beginning of the second line.
    - lcd\_write(1, 0, 'W');: Writes 'W' at the beginning of the second line.
    - lcd\_write(1, 1, 'o');: Writes 'o' at the second position of the second line.
    - lcd\_write(1, 2, 'r');: Writes 'r' at the third position of the second line.
    - lcd\_write(1, 3, 'l');: Writes 'l' at the fourth position of the second line.
    - lcd\_write(1, 4, 'd');: Writes 'd' at the fifth position of the second line.

**Summary:**

* **LCD.h** defines the pins and function prototypes.
* **LCD.c** implements the functions to control the LCD.
* **T\_14\_4.c** is the main program that demonstrates how to use the LCD functions to display "Hello" on the first line and "World" on the second line of a 16x2 LCD. The infinite loop ensures the microcontroller keeps running, continuously refreshing the display with the same message.

This project initializes and controls an LCD using an 8-bit interface, sending commands and data to display characters on the screen.