Experiment-5

Aim: Hands-on experimentation of 16x2 LCD interfacing with ATMega32 programming in C.

Objectives: After successfully completion of this experiment students will be able to,

- Use C language for ATMega32 microcontroller programming on AVRStudio.
- Experiment of interfacing 16x2 LCD with ATMega32 AVR Development Board. How to interface any external device with the development board.

Equipment required:

- Windows7 or later based host computer
- ATMega32 Development board
- USBasp Programmer
- Jumper Wires
- 16x2 LCD

Software required:

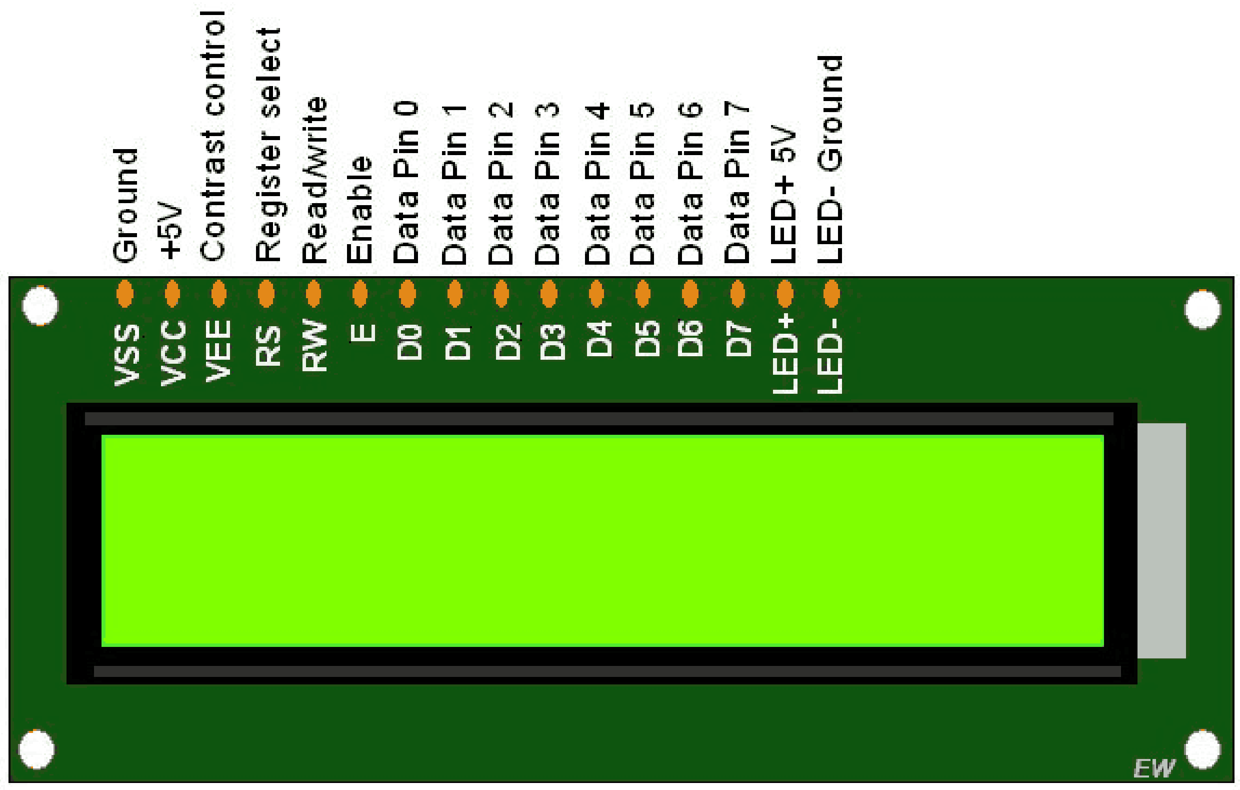
- AVR Studio7 installation setup
- USBasp driver installation setup

Theory:

16x2 LCD Introduction:
LCDs (Liquid Crystal Displays) are used for displaying status or parameters in embedded systems.
LCD 16x2 is a 16 pin device which has 8 data pins (D0-D7) and 3 control pins (RS, RW, EN). The remaining 5 pins are for supply and backlight for the LCD.

The control pins help us configure the LCD in command mode or data mode. They also help configure read mode or write mode and also when to read or write.

LCD 16x2 can be used in 4-bit mode or 8-bit mode depending on the requirement of the application. In order to use it, we need to send certain commands to the LCD in command mode and once the LCD is configured according to our need, we can send the required data in data mode.

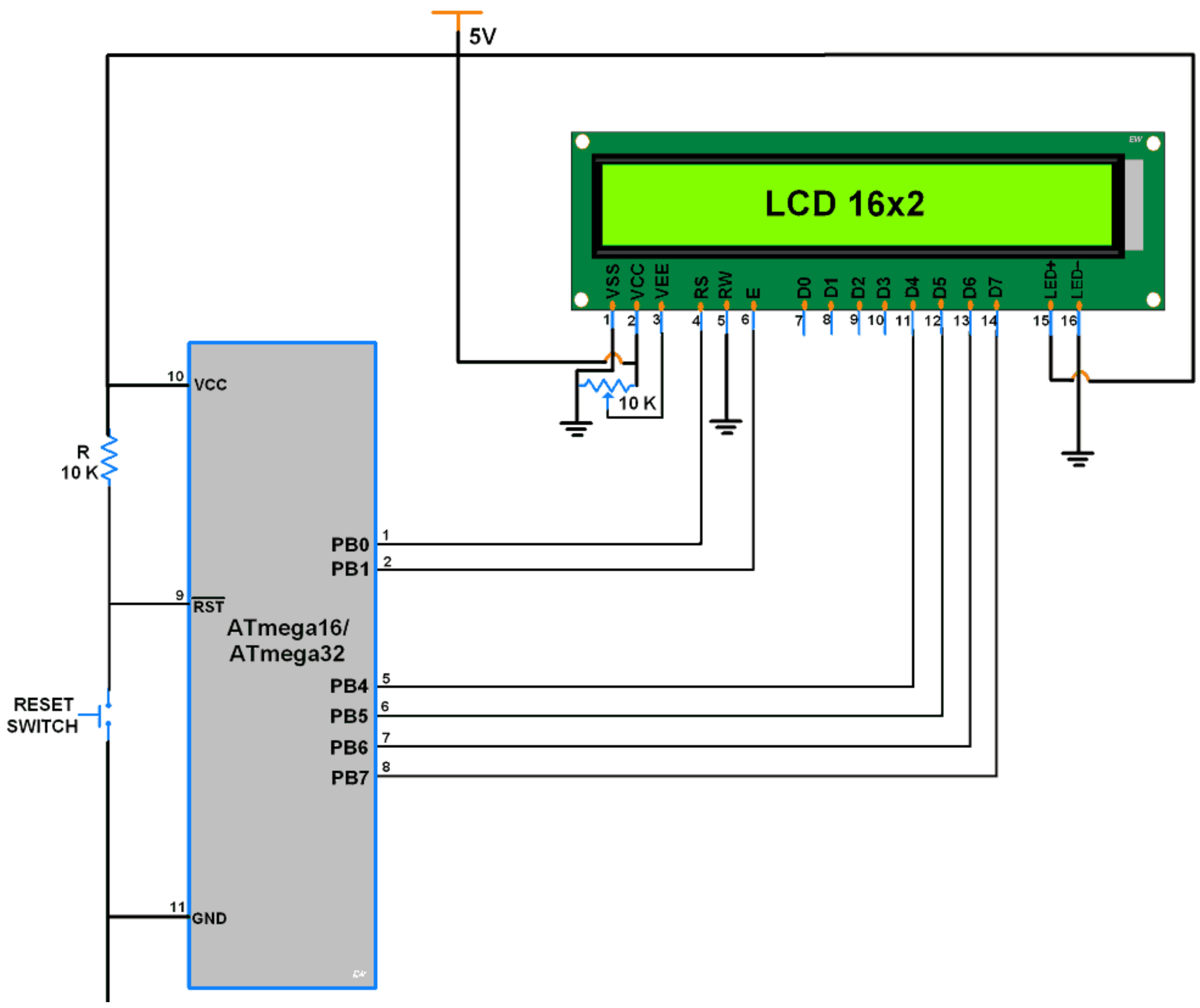


4-bit Mode

- In 4-bit mode, data/command is sent in a 4-bit (nibble) format.
- To do this 1st send a Higher 4-bit and then send a lower 4-bit of data/command.
- Only 4 data (D4 - D7) pins of 16x2 of LCD are connected to the microcontroller and other control pins RS (Register select), RW (Read/write), E (Enable) is connected to other GPIO Pins of the controller.

Therefore, due to such connections, we can save four GPIO pins which can be used for another application.

Interfacing Diagram



LCD Rs(Register select) Pin

Register select selects the controller registers. It switches between Command and data register.

- Command Register
- Data Register

CommandRegister

When we send commands to lcd these commands go to Command register and are processed there. Commands with their full description are given in the picture below. When Rs=0 command register is selected.

DataRegister

When we send Data to lcd it goes to data register and is processed there. When Rs=1 data register is selected.

LCD RW(Read/Write) Pin

Rw pin is used to read and write data to data and command registers. When Rw=1 we can read data from lcd. When Rw=0 we can write to lcd.

LCD En(Enable) Pin

When we select the register Rs(Command and Data) and set Rw(read – write) and placed the raw value on 8-data lines/4-data lines, now its time to execute the instruction. By instruction i mean the 8-bit data/4-bit data or 8-bit command/4-bit command present on Data lines of lcd. For sending the final data/command present on the data lines we use this enable pin. Usually it remains en=0 and when we want to execute the instruction we make it high en=1 for some mills seconds. After this we again make it ground en=0.

LCD V0 of contrast set pin

To set lcd display sharpness use this pin. Best way is to use variable resistor such as potentiometer a variable current makes the character contrast sharp. Connect the output of the potentiometer to this pin. Rotate the potentiometer knob forward and backward to adjust the lcd contrast.

NOTE:
we can not send an integer, float, long, double type data to lcd because lcd is designed to display a character only. Only the characters that are supported by the HD44780 controller. See the HD44780 data sheet to find out what characters can we display on lcd. The 8 data pins on lcd carries only Ascii 8-bit code of the character to lcd. How ever we can convert our data in character type array and send one by one our data to lcd. Data can be sent using lcd in 8-bit or 4-bit mode. If 4-bit mode is used, two nibbles of data (First high four bits and then low four bits) are sent to complete a full eight-bit transfer. 8-bit mode is best used when speed is required in an application and at least ten I/O pins are available. 4-bit mode requires a minimum of seven bits. In 4-bit mode, only the top 4 data pins (4-7) are used.

What are the general steps of interfacing LCD module with microcontrollers?
1)Do the port initialization and pin selection
2)Send the command to LCD

3) Send the data to LCD

4) String

5) Delay

6) LCD Initialization

7)Main logic

Programming steps of LCD16x2 4-bit mode with AVR Atmega32

Initialization

1. Wait for 15ms, Power-on initialization time for LCD16x2.
2. Send 0x02 command which initializes LCD 16x2 in 4-bit mode.
3. Send 0x28 command which configures LCD in 2-line, 4-bit mode, and 5x8 dots.
4. Send any Display ON command (0x0E, 0x0C)
5. Send 0x06 command (increment cursor)

Now we successfully initialized LCD & it is ready to accept data in 4-bit mode to display.

To send command/data to 16x2 LCD we have to send higher nibble followed by lower nibble. As 16x2 LCD's D4 - D7 pins are connected as data pins, we have to shift the lower nibble to the right by 4 before transmitting.

Command write function

1. First, send a Higher nibble of command.
2. Make RS pin low, RS=0 (command reg.)
3. Make RW pin low, RW=0 (write operation) or connect it to ground.
4. Give High to Low pulse at Enable (E).
5. Send lower nibble of command.
6. Give High to Low pulse at Enable (E).

Data write function

1. First, send a Higher nibble of data.
2. Make RS pin high, RS=1 (data reg.)
3. Make RW pin low, RW=0 (write operation) or connect it to ground.
4. Give High to Low pulse at Enable (E).
5. Send lower nibble of data.
6. Give High to Low pulse at Enable (E).

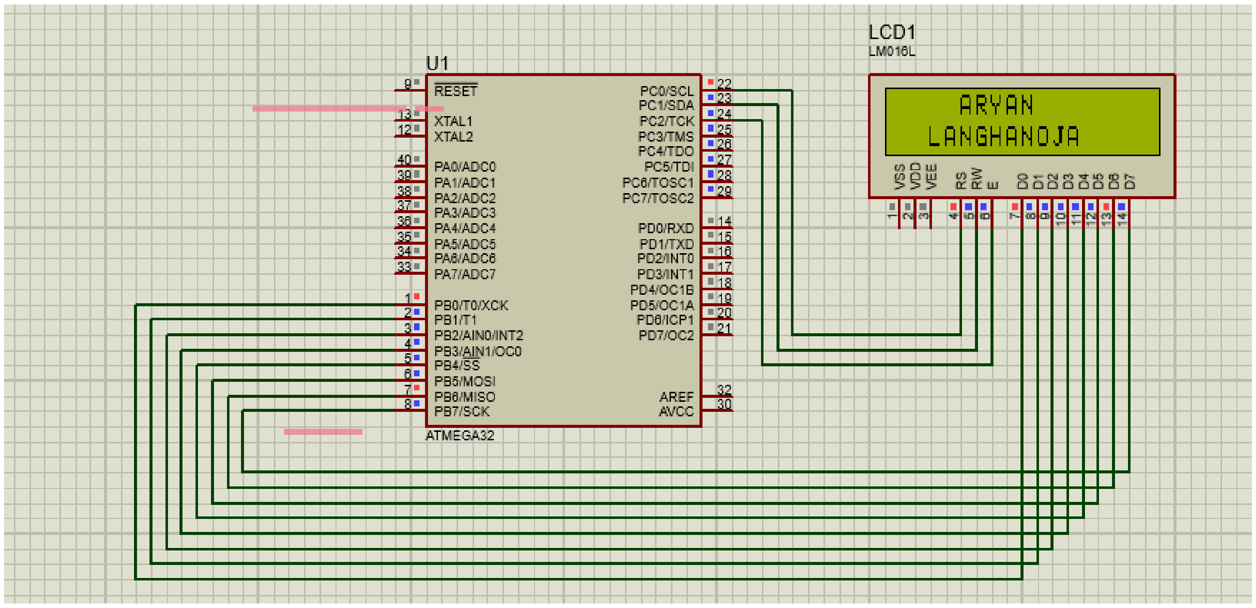
CODE:

```
#include <avr/io.h>
```

```
#include <util/delay.h>

#define F_CPU 1000000UL

void LCD_Command(unsigned char Command) {
        PORTB = Command;
        PORTC &= ~(0x01);
        PORTC &= ~(0x02);
        PORTC |= (0x04);
        _delay_ms(1); // Increased delay for commands
        PORTC &= ~(0x04);
}

void LCD_Data(unsigned char Data) {
        PORTB = Data;
        PORTC |= (0x01);
        PORTC &= ~(0x02);
        PORTC |= (0x04);
        _delay_ms(1); // Increased delay for data
        PORTC &= ~(0x04);
}

void LCD_Init() {
        _delay_ms(50); // wait for LCD to power up
        LCD_Command(0x38); // Initialize 8-bit mode
        LCD_Command(0x0C); // Display ON, Cursor OFF
        LCD_Command(0x06); // Increment cursor
        LCD_Command(0x01); // Clear LCD
        _delay_ms(2); // Delay for LCD to clear
        LCD_Command(0x80); // Move cursor to beginning of first line
}

void LCD_SetCursor(uint8_t row, uint8_t column) {
        uint8_t position = 0x80; // Base address for the first line

        if (row == 1) // If second row
        position = 0xC0; // Base address for the second line

        position += column - 1; // Adjust position for the desired column
        LCD_Command(position); // Set cursor position
}

int main(void) {
        DDRB = 0xFF;
        DDRC = 0xFF;
        LCD_Init();
```

```
    char First_Name[5] = "ARYAN";
    char Last_Name[10] = "LANGHANOJA";

        LCD_SetCursor(0, 5);
    for (int i = 0; i < 5; i++) {
            LCD_Data(First_Name[i]);
    }

        LCD_SetCursor(1, 3);
    for (int i = 0; i < 10; i++) {
            LCD_Data(Last_Name[i]);
    }

        while (1);

        return 0;
}
```

OUTPUT:



CONCLUSION:

By Doing this experiment I came to know LCD Programming.

Experiment-5 Post Lab Exercise

Student Name: Aryan Langhanoja
Enrollment No: 92200133030

Answer the following questions:

1) What is the function of following pins in 16x2 LCD:

   (i) RS

   (ii) R/W

   (iii) E

- **RS: -** This pin is used to select between data register (when RS is high) and command register (when RS is low).
- **R/W :-** This pin controls the direction of data flow.
- **E :-** This pin is used to enable the LCD module.

2) What is the difference between 4 bit and 8 bit LCD mode? Which mode should be used in which kind of applications? Which mode we have used in the experiment?

In 4-bit mode:

- Only 4 data lines (D4-D7) are used to transmit data.
- Data is sent to the LCD in two 4-bit nibbles.
- Requires fewer GPIO pins.

In 8-bit mode:

- All 8 data lines (D0-D7) are used for data transmission.
- Data is sent to the LCD in one 8-bit byte.
- Provides faster data transfer compared to 4-bit mode.
- Requires more GPIO pins.

3) Why we have connected the R/W pin to the GND in our experiment?
   - Because we have to only write the data to the LCD.
4) How to interface 16x2 LCD with our AVR development board? Why did not we connect VSS, VCC, and VEE? (HINT: Please put the SS of the exercise we did in lab of studying avr board lab manual and finding the schematic diagram of LCD with ATMEGA32 chip)

- Data lines (D0-D7) of the LCD connected to the data pins (PORT or PIN) of the AVR microcontroller.
- RS (Register Select), R/W (Read/Write), and E (Enable) pins of the LCD connected to specific pins of the AVR microcontroller.
- Optionally, you may connect control lines such as backlight control pins or contrast adjustment pins if needed.

- VSS (Ground): This pin connects to the ground (0V) of the power supply.
- VCC (Supply Voltage): This pin connects to the positive supply voltage (typically +5V or +3.3V).
- VEE (Contrast Adjustment): This pin is used to adjust the contrast of the LCD. By varying the voltage applied to this pin, you can adjust the contrast of the displayed characters.

5) What should we do if the display in LCD is not visible properly? Which pin of LCD should we look into for this problem?

   ➢ If the display on the LCD is not clear, adjust the VEE pin using a potentiometer. Connect the potentiometer between VCC and GND, with the wiper connected to the VEE pin. This helps optimize the contrast for better visibility.