## Objective: AVR Microcontroller LCD Programming In C.

**Task-1:-**  Interface 16x2 LCD with ATMEGA32 in 8-bit mode. Write a program to display your name. Your first name should be displayed from the 5th position of the 1st ROW and your surname from the 3rd position of the 2nd row.

**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]);

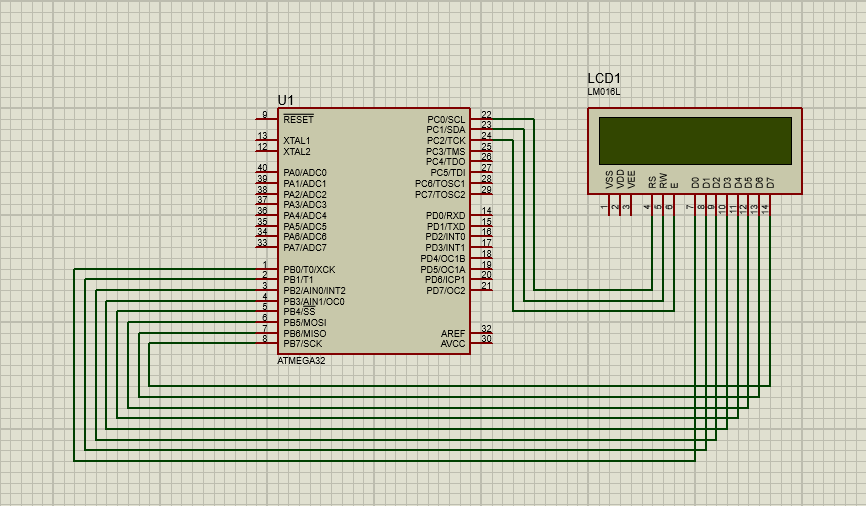
}

while (1);

return 0;

}

**Circuit:-**



**Task-2 :-**  Interface 16x2 LCD with ATMEGA32 in 4-bit mode to do the same task as mentioned in problem 1.

**Code :-**

#include <avr/io.h>

#include <util/delay.h>

#define F\_CPU 1000000UL

void LCD\_Command(unsigned char Command) {

unsigned char A , B ;

A = Command & 0xF0 ;

PORTB = A | 0x04 ;

\_delay\_ms(100);

PORTB &= ~(0x04) ;

B = Command << 4 ;

PORTB = B | 0x04 ;

\_delay\_ms(100) ;

PORTB &= ~(0x04) ;

}

void LCD\_Data(unsigned char Data) {

unsigned char A,B ;

A = Data & 0xF0 ;

PORTB = 0x05 | A ;

\_delay\_ms(100) ;

PORTB &= ~(0x04) ;

B = Data << 4 ;

PORTB = 0x05 | B ;

\_delay\_ms(100) ;

PORTB &= ~(0x04) ;

}

void LCD\_SetCursor(uint8\_t row, uint8\_t column) {

uint8\_t position = 0x80;

if (row == 1)

position = 0xC0;

position += column - 1;

LCD\_Command(position);

}

int main(void) {

DDRB = 0x0F ;

LCD\_Command(0x02);

LCD\_Command(0x28);

LCD\_Command(0x0E);

LCD\_Command(0x06);

LCD\_Command(0x01);

LCD\_Command(0x80);

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]);

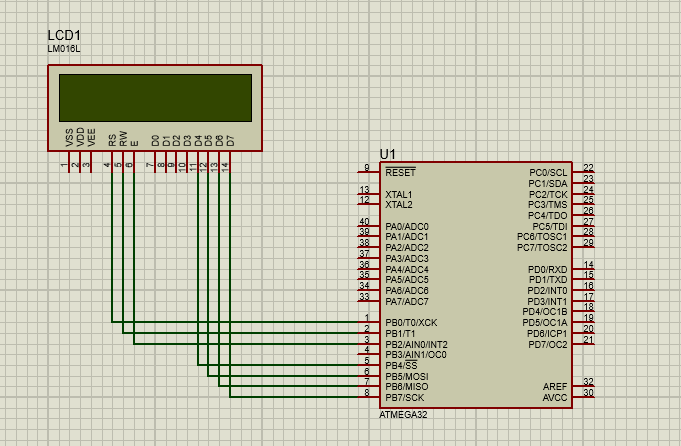
}

while (1);

return 0;

}

**Circuit:-**

****

**Task-3 :-**   Write a program to display scrolling text in 16x2 LCD. Assume the string to displayed as per your choice.

**Code :-**

#include <avr/io.h>

#include <util/delay.h>

#define LCD\_DATA\_PORT PORTD

#define LCD\_CTRL\_PORT PORTC

#define LCD\_RS PC0

#define LCD\_RW PC1

#define LCD\_EN PC2

void LCD\_Command(unsigned char Command) {

LCD\_DATA\_PORT = (LCD\_DATA\_PORT & 0x0F) | (Command & 0xF0);

LCD\_CTRL\_PORT &= ~(1 << LCD\_RS); // RS low for command

LCD\_CTRL\_PORT &= ~(1 << LCD\_RW); // RW low for write

LCD\_CTRL\_PORT |= (1 << LCD\_EN); // Enable high

\_delay\_us(1);

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // Enable low

\_delay\_us(100);

LCD\_DATA\_PORT = (LCD\_DATA\_PORT & 0x0F) | ((Command << 4) & 0xF0);

LCD\_CTRL\_PORT |= (1 << LCD\_EN); // Enable high

\_delay\_us(1);

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // Enable low

\_delay\_ms(2);

}

void LCD\_Data(unsigned char Data) {

LCD\_DATA\_PORT = (LCD\_DATA\_PORT & 0x0F) | (Data & 0xF0);

LCD\_CTRL\_PORT |= (1 << LCD\_RS); // RS high for data

LCD\_CTRL\_PORT &= ~(1 << LCD\_RW); // RW low for write

LCD\_CTRL\_PORT |= (1 << LCD\_EN); // Enable high

\_delay\_us(1);

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // Enable low

\_delay\_us(100);

LCD\_DATA\_PORT = (LCD\_DATA\_PORT & 0x0F) | ((Data << 4) & 0xF0);

LCD\_CTRL\_PORT |= (1 << LCD\_EN); // Enable high

\_delay\_us(1);

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // Enable low

\_delay\_ms(2);

}

void LCD\_Init(void) {

LCD\_DATA\_PORT = 0xFF; // Initialize LCD data port as output

LCD\_CTRL\_PORT |= (1 << LCD\_EN) | (1 << LCD\_RW) | (1 << LCD\_RS); // Initialize control port pins as output

\_delay\_ms(20);

LCD\_Command(0x33); // Initialization sequence

LCD\_Command(0x32); // Initialization sequence

LCD\_Command(0x28); // 4-bit mode, 2 lines, 5x8 font

LCD\_Command(0x0C); // Display on, cursor off, blink off

LCD\_Command(0x06); // Increment cursor

LCD\_Command(0x01); // Clear display

\_delay\_ms(2);

}

void LCD\_Clear(void) {

LCD\_Command(0x01); // Clear display

\_delay\_ms(2);

}

void LCD\_GotoXY(unsigned char x, unsigned char y) {

unsigned char address = 0;

if (y == 0) {

address = 0x80 + x; // Line 1

} else if (y == 1) {

address = 0xC0 + x; // Line 2

}

LCD\_Command(address);

}

void LCD\_Puts(const char \*str) {

while (\*str) {

LCD\_Data(\*str++);

}

}

// right to left

void LCD\_Scroll\_Text(const char \*text) {

int len = 0;

while (text[len] != '\0') {

len++;

}

// Calculate the starting position

int start\_position = len > 16 ? len - 16 : 0;

// Add spaces at the beginning to adjust the starting position

char scrolled\_text[2 \* len + 6]; // Double the length of the text for scrolling effect

for (int i = 0; i < start\_position; i++) {

scrolled\_text[i] = ' ';

}

// Copy the text to scrolled\_text

for (int i = 0; i < len; i++) {

scrolled\_text[start\_position + i] = text[i];

}

while (1) {

for (int i = 0; i < len + start\_position + 3; i++) {

LCD\_Clear();

LCD\_GotoXY(0, 0);

LCD\_Puts(scrolled\_text + i);

\_delay\_ms(500); // Adjust the delay as needed for desired scrolling speed

}

}

}

int main(void) {

DDRD = 0xFF; // Port D as output

DDRC |= (1 << LCD\_EN) | (1 << LCD\_RW) | (1 << LCD\_RS); // Control pins as output

LCD\_Init();

LCD\_Clear();

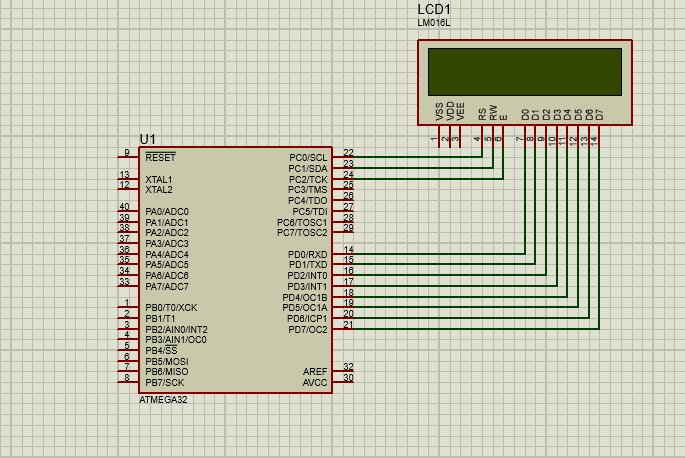
LCD\_Scroll\_Text("HELLO FROM ARYAN LANGHANOJA");

while (1) {

}

return 0;

}

**Circuit :-**

**Task-4 :-**  In continuation with problem 3, take an input from two switches connected on PD0 and PD1, and show the scrolling of text from right to left and left to right.

**Code :-**

#include <avr/io.h>

#include <util/delay.h>

#define LCD\_DATA\_PORT PORTD // Data port connected to PORT D

#define LCD\_CTRL\_PORT PORTC // Control port connected to PORT C

#define LCD\_RS PC0 // Register select pin

#define LCD\_EN PC1 // Enable pin

void lcd\_command(unsigned char cmnd) {

LCD\_DATA\_PORT = cmnd;

LCD\_CTRL\_PORT &= ~(1 << LCD\_RS); // RS = 0 for command

LCD\_CTRL\_PORT |= (1 << LCD\_EN); // EN = 1 for H-to-L pulse

\_delay\_us(1);

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // EN = 0 to latch command

\_delay\_us(100);

}

void lcd\_data(unsigned char data) {

LCD\_DATA\_PORT = data;

LCD\_CTRL\_PORT |= (1 << LCD\_RS); // RS = 1 for data

LCD\_CTRL\_PORT |= (1 << LCD\_EN); // EN = 1 for H-to-L pulse

\_delay\_us(1);

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // EN = 0 to latch data

\_delay\_us(100);

}

void lcd\_init() {

LCD\_CTRL\_PORT |= (1 << LCD\_RS) | (1 << LCD\_EN); // Set RS and EN as output

LCD\_DATA\_PORT = 0xFF; // Set data port as output

LCD\_CTRL\_PORT &= ~(1 << LCD\_RS); // RS = 0 for command

LCD\_CTRL\_PORT &= ~(1 << LCD\_EN); // EN = 0

\_delay\_ms(20); // Initialization routine

lcd\_command(0x38); // 2 line, 5x8 matrix

lcd\_command(0x0E); // Display on, cursor blinking

lcd\_command(0x01); // Clear LCD

\_delay\_ms(2);

}

void lcd\_string(char \*str) {

while (\*str)

lcd\_data(\*str++);

}

void lcd\_scroll\_left(char \*str) {

int i, len;

len = strlen(str);

for (i = 0; i < len; i++) {

lcd\_command(0x1C); // Shift display left

\_delay\_ms(500);

}

}

void lcd\_scroll\_right(char \*str) {

int i, len;

len = strlen(str);

for (i = 0; i < len; i++) {

lcd\_command(0x18); // Shift display right

\_delay\_ms(500);

}

}

int main() {

DDRB &= ~(1 << DDB0) & ~(1 << DDB1); // Set PB0 and PB1 as input

PORTB |= (1 << PB0) | (1 << PB1); // Enable pull-up resistors for PB0 and PB1

DDRD = 0xFF; // Set PORTD as output

DDRC = 0xFF; // Set PORTC as output

lcd\_init();

char message[] = "Scrolling Text";

while (1) {

if (!(PINB & (1 << PB0))) {

// If PB0 is high, scroll text

lcd\_scroll\_left(message);

} else if (!(PINB & (1 << PB1))) {

// If PB1 is high, scroll text in the opposite direction

lcd\_scroll\_right(message);

} else {

// If neither PB0 nor PB1 is high, display static text

lcd\_command(0x01); // Clear LCD

lcd\_command(0x80); // Set cursor to the beginning of the first line

lcd\_string(message);

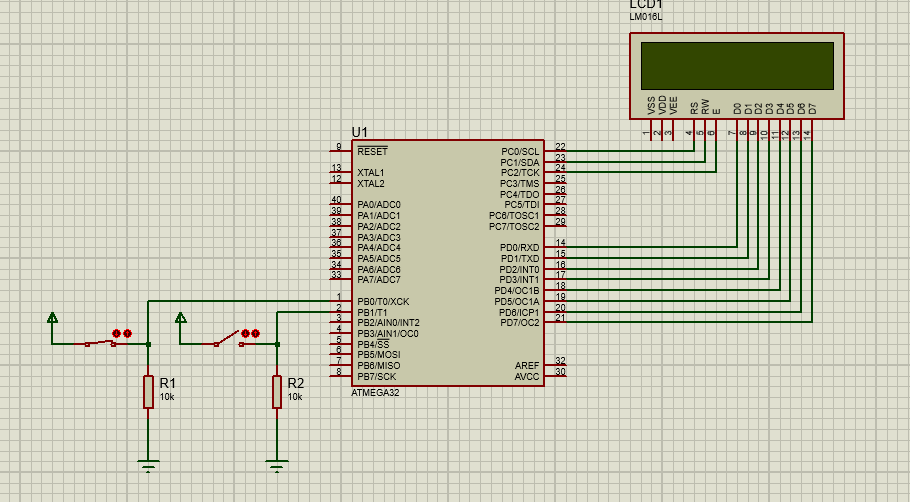
}

}

return 0;

}

**Circuit :-**



**Task-5 :-**  Display the coustom characters of your choice using 16x2 LCD and ATMEGA32.

**Code :-**

#define F\_CPU 8000000UL

#include <avr/io.h>

#include <util/delay.h>

#define rs PB2

#define en PB3

void lcd\_init();

void dis\_cmd(char);

void dis\_data(char);

void lcdcmd(char);

void lcddata(char);

void lcd\_create\_custom\_char();

void lcd\_display\_custom\_char();

int main(void)

{

DDRB = 0xFF;

lcd\_init();

lcd\_create\_custom\_char();

lcd\_display\_custom\_char();

while (1)

{

// Your main code here

}

}

void lcd\_init()

{

dis\_cmd(0x02);

dis\_cmd(0x28);

dis\_cmd(0x01);

dis\_cmd(0x0C);

dis\_cmd(0x06);

dis\_cmd(0x80); // Set DDRAM address to 0 (cursor at home position)

}

void dis\_cmd(char cmd\_value)

{

char cmd\_value1;

cmd\_value1 = cmd\_value & 0xF0;

lcdcmd(cmd\_value1);

cmd\_value1 = ((cmd\_value << 4) & 0xF0);

lcdcmd(cmd\_value1);

}

void dis\_data(char data\_value)

{

char data\_value1;

data\_value1 = data\_value & 0xF0;

lcddata(data\_value1);

data\_value1 = ((data\_value << 4) & 0xF0);

lcddata(data\_value1);

}

void lcdcmd(char cmdout)

{

PORTB = cmdout;

PORTB &= ~(1 << rs);

PORTB |= (1 << en);

\_delay\_ms(1);

PORTB &= ~(1 << en);

}

void lcddata(char dataout)

{

PORTB = dataout;

PORTB |= (1 << rs);

PORTB |= (1 << en);

\_delay\_ms(1);

PORTB &= ~(1 << en);

}

void lcd\_create\_custom\_char()

{

// Set CGRAM address to 0

dis\_cmd(0x40);

// Define custom character pattern (Heart shape)

dis\_data(0b00000);

dis\_data(0b01010);

dis\_data(0b11111);

dis\_data(0b11111);

dis\_data(0b01110);

dis\_data(0b00100);

dis\_data(0b00000);

dis\_data(0b00000);

// Set DDRAM address to 0x80 (cursor at home position)

dis\_cmd(0x80);

\_delay\_ms(100);

}

void lcd\_display\_custom\_char()

{

// Display the custom character at the beginning of the first line

dis\_data(0); // Display custom character at position (0, 0)

}

**Circuit :-**