MINI PROJECT

# **Interfacing 16×2 LCD with**

# **Raspberry Pi**

**PRESENTED BY: -**

445 - Iftekhar Ansari

446 - Sunil Gaikwad

452 - Abhay Parocha

455 - Shivam Yadav

**GUIDED BY: -**

## Sanjay Deshmukh

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING YEAR 2020-2021

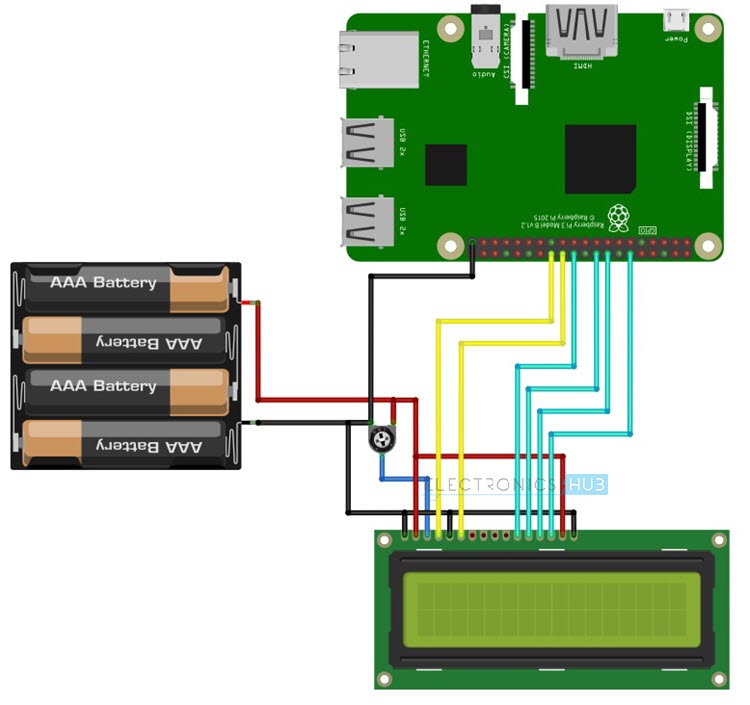
# **Abstract: -**

In this project, you can see all the steps for Interfacing a 16×2 LCD with Raspberry Pi like circuit diagram, components, working, Python Program and explanation of the code.

Even though the Raspberry Pi computer is capable of doing many tasks, it doesn’t have a display for implementing it in simple projects. A 16×2 Alphanumeric Character LCD Display is a very important types of display for displaying some basic and vital information.

A 16×2 LCD is one of the most popular display modules among hobbyists, students and even electronics professionals. It supports 16 characters per row and has two such rows. Almost all the 16×2 LCD Display Modules that are available in the market are based on the Hitachi’s HD44780 LCD Controller.

# **Circuit diagram: -**

****

**Component: -**

* Raspberry Pi 3 Model B (any Raspberry Pi)
* 16 x 2 LCD Module
* 10 KΩ Potentiometer
* Mini Breadboard
* Connecting wires (Jumper wires)
* 5V – 2A Power Supply
* Miscellaneous (Computer, Ethernet Cable, etc.

### **Circuit Design:-**

The design of the circuit for Interfacing 16×2 LCD with Raspberry Pi is very simple. First, connect pins 1 and 16 of the LCD to GND and pins 2 and 15 to 5V supply.

Then connect a 10KΩ Potentiometer to pin 3 of the LCD, which is the contrast adjust pin. The three control pins of the LCD i.e. RS (Pin 4), RW (Pin 5) and E (Pin 6) are connected to GPIO Pin 7 (Physical Pin 26), GND and GPIO Pin 8 (Physical Pin 24).Now, the data pins of the LCD.

Since we are configuring the LCD in 4 – bit mode, we need only 4 data pins (D4 to D7). D4 of LCD is connected to GPIO25 (Physical Pin 22), D5 to GPIO24 (Physical Pin 18), D6 to GPIO24 (Physical Pin 16) and D7 to GPIO18 (Physical Pin 12).

# **Working: -**

The working of project for Interfacing 16×2 LCD with Raspberry Pi is very simple. After making the connections as per the circuit diagram, login to your Raspberry Pi using SSH Client like Putty in Windows.

Alternatively, you can use software like Proteus. (NOTE: I’ve used Proteus Software for accessing the Simulation of Raspberry Pi’s Desktop on my personal computer). I’ve created a folder named “lcd\_rpi” on the desktop of the Raspberry Pi. So, I’ll be saving my Python Program for Interfacing 16 x 2 LCD with Raspberry Pi in this folder.

Using “cd” commands in the terminal, change to this directory. After that, open an empty Python file with name “lcdPi.py” using the following command in the terminal. Now, copy the above code and paste it in the editor. It is important to properly use the Tab characters as they help in grouping the instructions in Python.

Save the file and close the editor. To test the code, type the following command in the terminal. If everything is fine with your connections and Python Program, you should be able to see the text on the 16×2 LCD.

To exit the program, simply press CRTL+C in the terminal.

**Code:-**

#!/usr/bin/python

import RPi.GPIO as GPIO

import time

# Define GPIO to LCD mapping

LCD\_RS = 7

LCD\_E = 8

LCD\_D4 = 25

LCD\_D5 = 24

LCD\_D6 = 23

LCD\_D7 = 18

# Define some device constants

LCD\_WIDTH = 16 # Maximum characters per line

LCD\_CHR = True

LCD\_CMD = False

LCD\_LINE\_1 = 0x80 # LCD RAM address for the 1st line

LCD\_LINE\_2 = 0xC0 # LCD RAM address for the 2nd line

# Timing constants

E\_PULSE = 0.0005

E\_DELAY = 0.0005

def main():

# Main program block

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM) # Use BCM GPIO numbers

GPIO.setup(LCD\_E, GPIO.OUT) # E

GPIO.setup(LCD\_RS, GPIO.OUT) # RS

GPIO.setup(LCD\_D4, GPIO.OUT) # DB4

GPIO.setup(LCD\_D5, GPIO.OUT) # DB5

GPIO.setup(LCD\_D6, GPIO.OUT) # DB6

GPIO.setup(LCD\_D7, GPIO.OUT) # DB7

# Initialise display

lcd\_init()

while True:

# Send some test

lcd\_string("WELCOME",LCD\_LINE\_1)

lcd\_string("SIR / MAM ",LCD\_LINE\_2)

time.sleep(3) # 2 second delay

# Send some text

lcd\_string("1234567890123456",LCD\_LINE\_1)

lcd\_string("abcdefghijklmnop",LCD\_LINE\_2)

time.sleep(2) # 2 second delay

# Send some text

lcd\_string("GROUP 13",LCD\_LINE\_1)

lcd\_string("EXTC",LCD\_LINE\_2)

time.sleep(2)

# Send some text

lcd\_string("RPI PROJECT",LCD\_LINE\_1)

lcd\_string("THANK YOU",LCD\_LINE\_2)

time.sleep(2)

def lcd\_init():

# Initialise display

lcd\_byte(0x33,LCD\_CMD) # 110011 Initialise

lcd\_byte(0x32,LCD\_CMD) # 110010 Initialise

lcd\_byte(0x06,LCD\_CMD) # 000110 Cursor move direction

lcd\_byte(0x0C,LCD\_CMD) # 001100 Display On,Cursor Off, Blink Off

lcd\_byte(0x28,LCD\_CMD) # 101000 Data length, number of lines, font size

lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

time.sleep(E\_DELAY)

def lcd\_byte(bits, mode):

# Send byte to data pins

# bits = data

# mode = True for character

# False for command

GPIO.output(LCD\_RS, mode) # RS

# High bits

GPIO.output(LCD\_D4, False)

GPIO.output(LCD\_D5, False)

GPIO.output(LCD\_D6, False)

GPIO.output(LCD\_D7, False)

if bits&0x10==0x10:

GPIO.output(LCD\_D4, True)

if bits&0x20==0x20:

GPIO.output(LCD\_D5, True)

if bits&0x40==0x40:

GPIO.output(LCD\_D6, True)

if bits&0x80==0x80:

GPIO.output(LCD\_D7, True)

# Toggle 'Enable' pin

lcd\_toggle\_enable()

# Low bits

GPIO.output(LCD\_D4, False)

GPIO.output(LCD\_D5, False)

GPIO.output(LCD\_D6, False)

GPIO.output(LCD\_D7, False)

if bits&0x01==0x01:

GPIO.output(LCD\_D4, True)

if bits&0x02==0x02:

GPIO.output(LCD\_D5, True)

if bits&0x04==0x04:

GPIO.output(LCD\_D6, True)

if bits&0x08==0x08:

GPIO.output(LCD\_D7, True)

# Toggle 'Enable' pin

lcd\_toggle\_enable()

def lcd\_toggle\_enable():

# Toggle enable

time.sleep(E\_DELAY)

GPIO.output(LCD\_E, True)

time.sleep(E\_PULSE)

GPIO.output(LCD\_E, False)

time.sleep(E\_DELAY)

def lcd\_string(message,line):

# Send string to display

message = message.ljust(LCD\_WIDTH," ")

lcd\_byte(line, LCD\_CMD)

for i in range(LCD\_WIDTH):

lcd\_byte(ord(message[i]),LCD\_CHR)

if \_\_name\_\_ == '\_\_main\_\_':

try:

main()

except KeyboardInterrupt:

pass

finally:

lcd\_byte(0x01, LCD\_CMD)

lcd\_string("Goodbye!",LCD\_LINE\_1)

GPIO.cleanup()

### **APPLICATION: -**

* By interfacing 16×2 LCD with Raspberry Pi, we can have a simple display option for our raspberry Pi which can display some basic information like Date, Time, Status of a GPIO Pin, etc.
* Many simple and complex application of Raspberry Pi like weather station, temperature control, robotic vehicles, etc. needs this small 16×2 LCD Display.

### **Limitations:-**

* The 16×2 LCD Module can only display simple alphanumeric characters.
* Even though some special characters and custom characters can be displayed, information which is graphic intensive cannot be displayed.