LABORATORY SKILLS ASSESSMENT (Affective)

Affective Activity: Lab Report + Viva Voce

S. No	Criteria	Performance	Description	Marks
1	**Lab Report Completeness & Organization (4)**	Level 4	Report is complete, well-organized, properly presented, and demonstrates responsible completion of the assigned laboratory work.	4
		Level 3	Report is complete with minor omissions or organizational issues and demonstrates satisfactory responsibility for the assigned work.	3
		Level 2	Report is partially complete with several omissions and demonstrates limited responsibility for the assigned work.	2
		Level 1	Report has major omissions or poor organization and demonstrates minimal responsibility for the assigned work.	1
2	**Keypad/Display Output & Conclusion (2)**	Level 2	The keypad/display output and conclusion are correctly presented and discussed, demonstrating ownership of the assigned work.	2
		Level 1	The keypad/display output and conclusion are presented with minor errors or omissions.	1
3	**Viva Voce (4)**	Level 4	Clearly explains the assigned experiment, keypad scanning, display coding, program explanation, working, and results, demonstrating individual responsibility, participation, and effective communication.	4
		Level 3	Explains most of the assigned experiment and results correctly, with minor difficulty, and demonstrates good individual participation.	3
		Level 2	Explains basic aspects of the assigned experiment with some prompting and demonstrates basic participation.	2
		Level 1	Demonstrates limited understanding or involvement and requires substantial prompting to explain the assigned work.	1
Total Obtained				**/10**

The student has not submitted the report and has not appeared in the Viva Voce.

Criteria	Description	Marks
Lab Report and Results Not Submitted	Student has not submitted the report and results file.	**0**
Absent in Viva Voce	Student is absent from the viva voce.	**0**

Instructor's Signature: _____ Date: _____

Experiment No 7

Liquid Crystal Display (LCD) Interface with STM32

Objectives:

- To learn the basics of 16x2 LCD and its interface with STM32F103 Board
- Controlling the LCD by using the library of LCD 16x2.

16x2 LCD:

Liquid Crystal Display (LCD) is very commonly used electronic display module and having a wide range of applications such as calculators, laptops, mobile phones etc. 16×2 character LCD display is very basic module which is commonly used in electronics devices and projects. 16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like, 8×1, 8×2, 10×2, 16×1, etc. But the most used one is the 16×2 LCD, hence we are using it here. It can display 2 lines of 16 characters. Each character is displayed using 5×7 or 5×10 pixel matrix. Interfacing 16×2 LCD with STM32F103 using Keil uVision5 is a bit complex because there are no powerful libraries in Keil. To solve this problem we will be using an open source LCD library which includes commonly used features, you just need to include therequired files and use them.

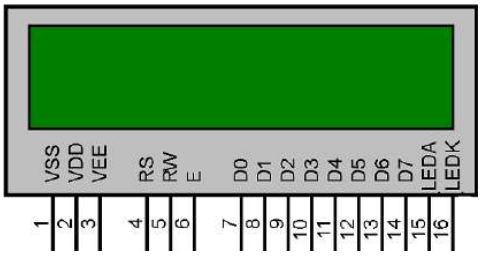


Figure-1: 16x2 LCD Schematic



Figure-2: 16x2 LCD

The pin configuration of 16x2 LCD is shown below:

NAME	FUNCTION
VSS	Ground Pin
VDD	+5V Input Pin
VEE	Contrast Adjust Pin
RS	Register Select
R/W	Read/Write Pin
E	Enable Pin
D0-D7	Data Pins (8 Pins)
LED A	Anode Pin (+5V)
LED K	Cathode Pin (GND)

The LCD can work in two different modes, namely the 4-bit mode and the 8-bit mode. In 4-Bit mode we send the data nibble by nibble, first upper nibble and then lower nibble. For those of

you who don‟t know what a nibble is: a nibble is a group of four bits, so the lower four bits (D0-D3) of a byte form the lower nibble while the upper four bits (D4-D7) of a byte form the higher nibble. This enables us to send 8- Bit data.

Whereas in 8-Bit mode we can send the 8-bit data directly in one stroke since we use all the 8 data lines.

Here in this project we will use the most commonly used mode which is 4-Bit mode. In 4-Bit mode we can save 4 pins of microcontroller and also reduce the wiring overhead.

16x2 also uses HEX code to take any command, there are many hex commands for LCD like to move the cursor, select the mode, shift the control to second line etc.

Keypad Interface:

Embedded system applications usually require large number of buttons connected (ex. calculator,cell phone). Keypad is one of the input devices that are commonly used in embedded systems which require large number of buttons.
4x3 matrix keypads have 12 buttons. We can connect these buttons directly to the GPIO pins of STM32F103, but it will eat up precious GPIO pins (If we have 12 buttons, then we need 12 GPIO pins). To avoid this trouble, keypad uses a technique that will save GPIO pins. In this technique, buttons are connected in a matrix (row/column) style.



Figure 1: 4x3 Keypad

4x4 matrix keypads have 16 buttons. We can connect these buttons directly to the GPIO pins of STM32F103, but it will take 16 GPIO pins so the 4x4 keypad uses the matrix (row/column) technique just like in the 4x3 keypad.

The internal connection of the buttons in 4x3 matrix keypad is like this:

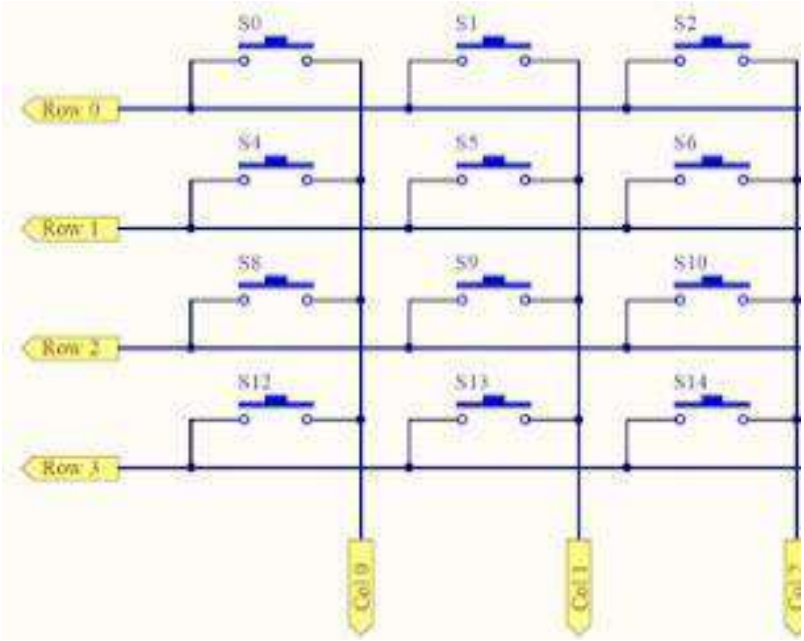


Figure 2: 4x3 Keypad Internal Circuitry

From the image above, for example if S8 is pressed, then Col-0 and Row 2 is connected. We can detect this by using a microcontroller. The microcontroller must scan the keypad by writing the columns and reading the rows.

To connect this keypad to the microcontroller, we need 7 GPIO pins. The column pins will be connected to the GPIO pins as open-drain or push-pull output. The row pins will be connected to the GPIO pins as inputs. At first, the column pins will be set to logic 1, and the other columnpins will be set to logic 0 condition. So, when no button pressed in this column, then all row pins will receive logic 0. When there is a button pressed on this column, then the corresponding row pin for that button will receive logic 1. The microcontroller can read the rows and decide what to display on the 16x2 LCD Display.

Required GPIO Configurations for Keypad:

```
void configureKeyPadpins(void)
{  GPIO_InitTypeDef myLED;
   // Enable clock for GPIOA
RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
   // GPIO Configuration for Columns (PA13, PA14, PA15)
   myLED.GPIO_Speed =
   GPIO_Speed_2MHz;
   myLED.GPIO_Mode  =
   GPIO_Mode_Out_PP;
   myLED.GPIO_Pin   = GPIO_Pin_13 |
   GPIO_Pin_14 | GPIO_Pin_15;
   GPIO_Init(GPIOA, &myLED);
   // GPIO Configuration for Rows (PA8, PA9, PA10, PA11)
   myLED.GPIO_Mode  = GPIO_Mode_IPU;
// better than floating
   myLED.GPIO_Pin   = GPIO_Pin_8 |
   GPIO_Pin_9 | GPIO_Pin_10 | GPIO_Pin_11;
   GPIO_Init(GPIOA, &myLED);  }
```

Required GPIO Configurations for LCD:

LCD_Interface.c | lcd16x2.h | lcd16x2.c | delay.h | delay.c

```
22
23  /** Port and pin definition for 4-bit mode ---------------------
24  // LCD control lines (must be on the same port)
25  #define LCD16X2_RCC_GPIO_CONTROL  RCC_APB2Periph_GPIOB
26  #define LCD16X2_GPIO_CONTROL      GPIOB
27  #define LCD16X2_GPIO_RS           GPIOB
28  #define LCD16X2_GPIO_RW           GPIOB
29  #define LCD16X2_GPIO_EN           GPIOB
30  // LCD data lines (must be on the same port)
31  #define LCD16X2_RCC_GPIO_DATA     RCC_APB2Periph_GPIOA
32  #define LCD16X2_GPIO_DATA         GPIOA
33  #define LCD16X2_GPIO_D4           GPIOA
34  #define LCD16X2_GPIO_D5           GPIOA
35  #define LCD16X2_GPIO_D6           GPIOA
36  #define LCD16X2_GPIO_D7           GPIOA
37  // Pin definition
38  #define LCD16X2_PIN_RS  GPIO_Pin_13
39  #define LCD16X2_PIN_RW  GPIO_Pin_14
40  #define LCD16X2_PIN_EN  GPIO_Pin_15
41  #define LCD16X2_PIN_D4  GPIO_Pin_0    // 4-bit mode LSB
42  #define LCD16X2_PIN_D5  GPIO_Pin_1
43  #define LCD16X2_PIN_D6  GPIO_Pin_2
44  #define LCD16X2_PIN_D7  GPIO_Pin_3    // 4-bit mode MSB
45
```

Libraries to include:

```
1  #include "stm32f10x.h"
2  #include "stm32f10x_rcc.h"
3  #include "stm32f10x_gpio.h"
4  #include "delay.h"
5  #include "lcd16x2.h"
6
7  void configureKeyPadpins(void);
8
9  void SetColumnsLow(void);
10 void SetColumn_1(void);
11 void SetColumn_2(void);
12 void SetColumn_3(void);
13
14 void KeyPad_Scan(void);
15
16 int Col_1=0;
17 int Col_2=0;
18 int Col_3=0;
19
20 GPIO_InitTypeDef myLED;
21
```

Main Function:

```
22  int main(void)
23  {
24      DelayInit();
25
26      // LCD initialization
27      lcd16x2_init(LCD16X2_DISPLAY_ON_CURSOR_ON_BLINK_OFF);
28
29      configureKeyPadpins();
30
31      lcd16x2_puts("Soban Ahmed");
32      lcd16x2_gotoxy(7,2);
33      DelayMs(10);
34
35      SetColumnsLow();
36
37
38      while (1)
39      {
40          SetColumn_1();
41          DelayMs(1);
42          KeyPad_Scan();
43
44          SetColumn_2();
45          DelayMs(1);
46          KeyPad_Scan();
47
48          SetColumn_3();
49          DelayMs(1);
50          KeyPad_Scan();
51      }
52  }
```

Set Column Functions:

```
164  void SetColumnsLow(void)
165  {
166    GPIO_WriteBit(GPIOA, GPIO_Pin_13,0);      // Col-1 = 0
167    GPIO_WriteBit(GPIOA, GPIO_Pin_14,0);      // Col-2 = 0
168    GPIO_WriteBit(GPIOA, GPIO_Pin_15,0);      // Col-3 = 0
169
170    Col_1=0;
171    Col_2=0;
172    Col_3=0;
173  }
174  void SetColumn_1(void)
175  {
176    GPIO_WriteBit(GPIOA, GPIO_Pin_13,1);      // Col-1 = 1
177    GPIO_WriteBit(GPIOA, GPIO_Pin_14,0);      // Col-2 = 0
178    GPIO_WriteBit(GPIOA, GPIO_Pin_15,0);      // Col-3 = 0
179
180    Col_1=1;
181    Col_2=0;
182    Col_3=0;
183  }
184  void SetColumn_2(void)
185  {
186    GPIO_WriteBit(GPIOA, GPIO_Pin_13,0);      // Col-1 = 0
187    GPIO_WriteBit(GPIOA, GPIO_Pin_14,1);      // Col-2 = 1
188    GPIO_WriteBit(GPIOA, GPIO_Pin_15,0);      // Col-3 = 0
189
190    Col_1=0;
191    Col_2=1;
192    Col_3=0;
193  }
194  void SetColumn_3(void)
195  {
196    GPIO_WriteBit(GPIOA, GPIO_Pin_13,0);      // Col-1 = 0
197    GPIO_WriteBit(GPIOA, GPIO_Pin_14,0);      // Col-2 = 0
198    GPIO_WriteBit(GPIOA, GPIO_Pin_15,1);      // Col-3 = 1
199
200    Col_1=0;
201    Col_2=0;
202    Col_3=1;
203  }
```

Keypad Scan Function:

```
127  void KeyPad_Scan(void)
128  {
129      if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_11) == 1)
130      {
131        lcd16x2_cursor_shift_left();
132        if(Col_1==1)
133          lcd16x2_puts("1");
134        if(Col_2==1)
135          lcd16x2_puts("2");
136        if(Col_3==1)
137          lcd16x2_puts("3");
138      }
139      if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_10) == 1)
```

Programming in KEIL:

To program the interfaces in Keil, the first step is to create a new uVision Project and selectingthe device which is STM32F103C8.

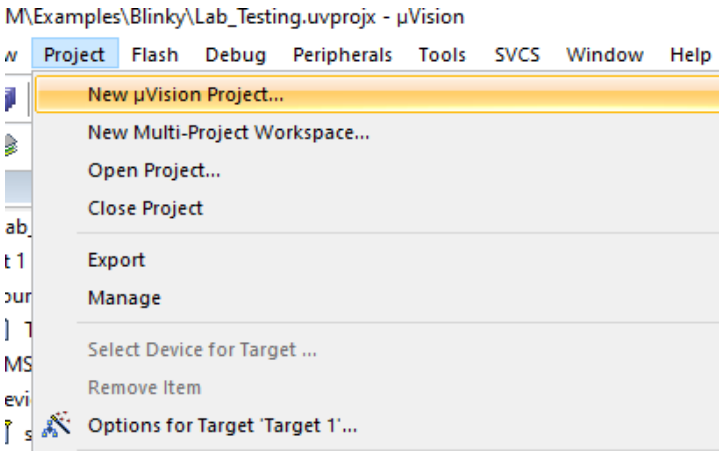


Figure 3: New Project Creation

After saving the project in your desired directory, the device will be selected in the „SelectDevice for Target“ dialogue box.

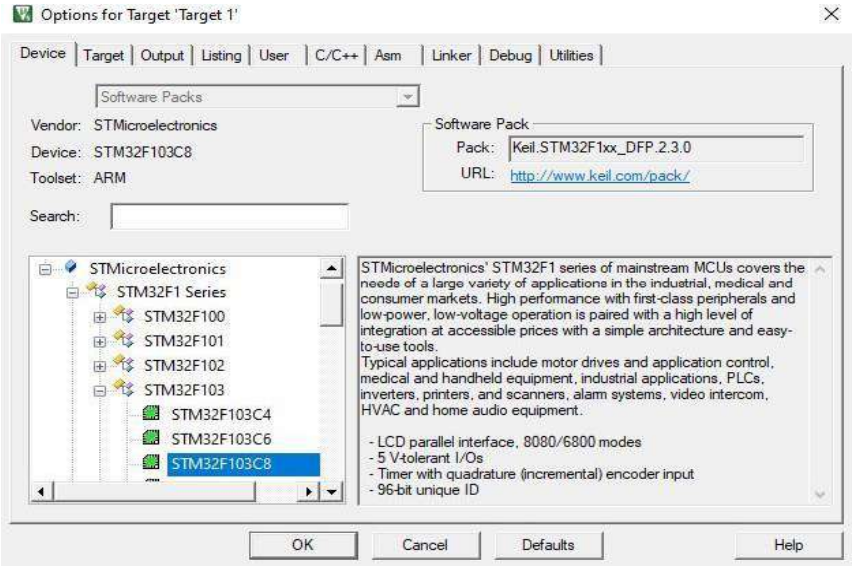


Figure 4: Device Selection

After that, include the libraries as shown in the figure below:

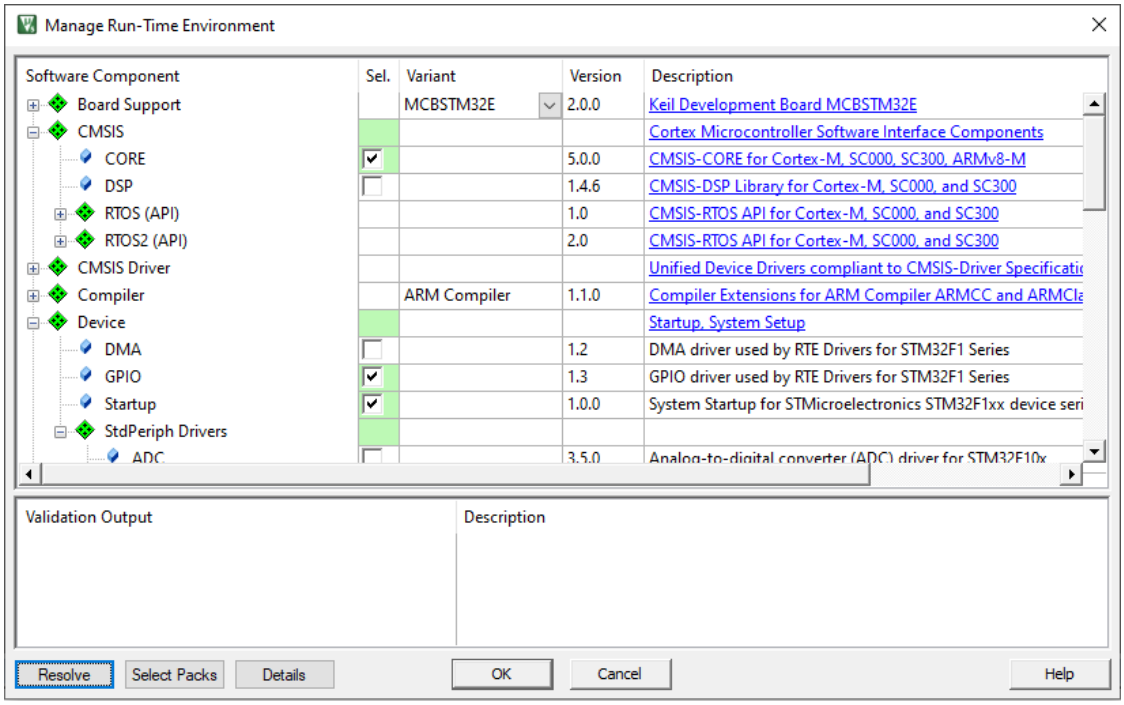


Figure 5: Adding required libraries

Then add a new item into your source group.

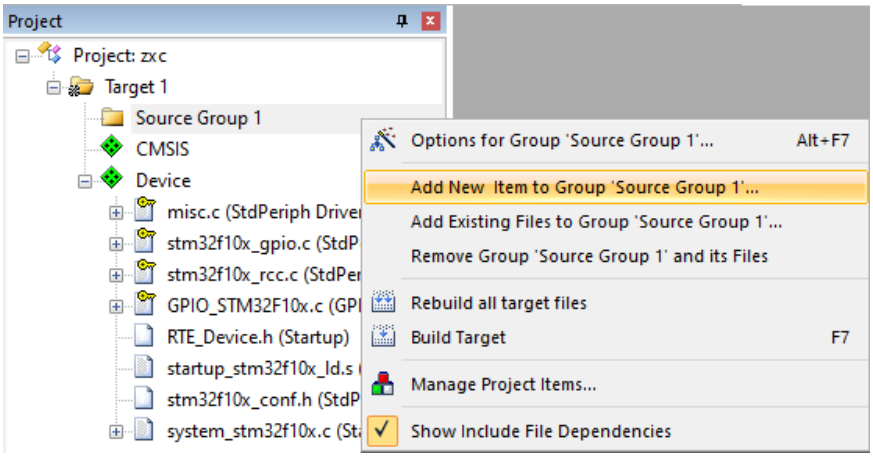


Figure 6: Adding new file

Select the “C File (.c)” option, name your C file and save it in your desired directory.

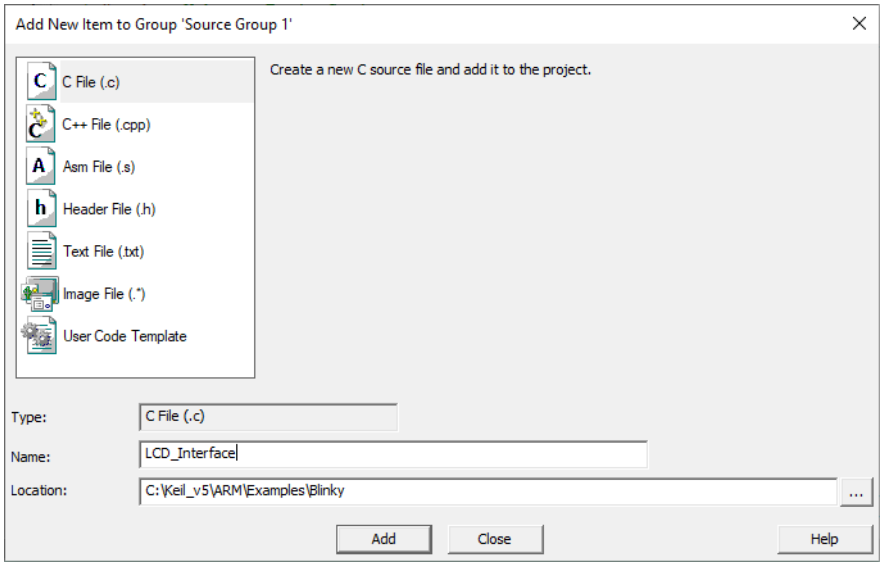


Figure 7: Adding C File

After adding the new C file add the required header files as shown below.

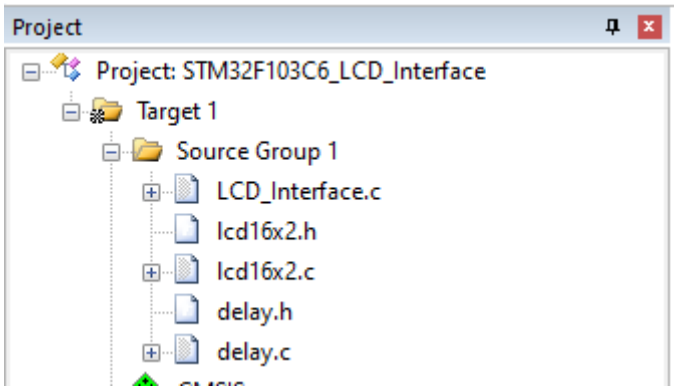


Figure 8: Required Header files

Lab Task (Psychomotor)

Activity 01:

Write C-Program to interface 16x2 LCD with STM32F103 board. Program will write your name in the first row of the LCD and it will also read the keys pressed by the user on the keypad and display on the second row of LCD. Implement the design using STM32F103 board and show the results to lab instructor.

Activity 02:

Write a C-Program that can multiply and divide two numbers and displays the result in second row of LCD. The question statement will be displayed on the first row of LCD. After completion of one calculation the LCD screen must be cleared before doing the second calculation. User will press the keypad buttons for inputs (* for multiplication and # for division).

Lab Task (Cognitive)

Question 1:

A 16x2 character LCD (HD44780-compatible) is interfaced with the STM32F4 in 4-bit mode. Line 1 DDRAM starts at 0x00 and line 2 DDRAM starts at 0x40.

a) Determine the command bytes (hexadecimal) needed to place the cursor at (i) line 2, first column and (ii) line 1, sixth column (column index 5). **[4 Marks]**

b) Describe how the data byte 0x4A is sent in 4-bit mode and state which character is displayed. **[3 Marks]**

c) State the role of the RS pin (0 and 1) and of the E pin in a write operation. **[3 Marks]**