****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**19CSE303 – EMBEDDED SYSTEMS**

**BATCH: 2022-2026**

**TOPIC: Bluetooth-Controlled LCD Display System with STM32 and HC-05 Module**

|  |  |  |
| --- | --- | --- |
| **01** | **ANANTHKRISHNAN C K** | **CB.EN.U4CSE22308** |
| **02** | **PAVITHRAA S** | **CB.EN.U4CSE22343** |
| **03** | **VIVEK RAJASEKHAR** | **CB.EN.U4CSE22356** |
| **04** | **GAURANG VARMA** | **CB.EN.U4CSE22365** |

**Objective**: This project uses an STM32 microcontroller, an HC-05 Bluetooth module, and a 16x2 LCD to display messages received over Bluetooth from an Android phone. The system processes command-based instructions to manage the LCD display content, including text display, clearing, and scrolling.

**System Overview**

The system comprises the following components:

1. **STM32 Microcontroller**: Serves as the central controller for UART-based communication and LCD control.
2. **HC-05 Bluetooth Module**: Allows wireless communication with a mobile device.
3. **16x2 LCD Display**: Displays messages received over Bluetooth.
4. **GPIO Pins**: Configured for LCD control and general purpose
5. **Android Phone:** Acts as the input device, sending commands and messages to the microcontroller.

The software is written in C using the STM32 HAL library . Worked with ? IDE

**Hardware Setup**

1. **UART Communication**:

**UART Protocol for Bluetooth Communication**

* **Configuration**: UART is configured at a 9600 baud rate for seamless data transfer with the HC-05 Bluetooth module.
* **Interrupt Handling**: Data reception via UART is managed using interrupts, enabling the STM32 to process incoming data without active polling.
* **Message Parsing**: Commands are parsed to allow various display options (e.g., clearing the screen, scrolling text).
* **Challenges**:
  + Implementing reliable message parsing.
  + Handling asynchronous data received over UART.

1. **LCD Pins**:

LCD is connected in a 4-bit mode

D7->B10

D6->B2

D5->B1

D4->B0

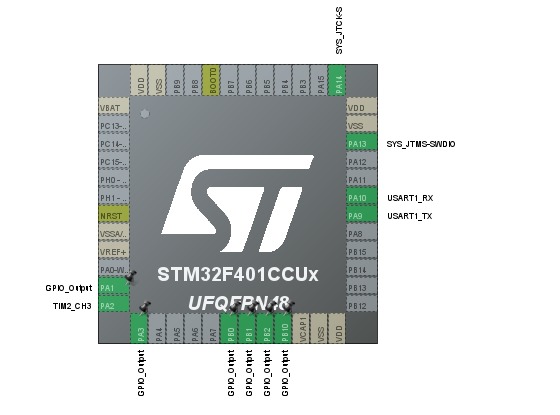
Control pins for the LCD are connected as follows:

**RS**: PA1

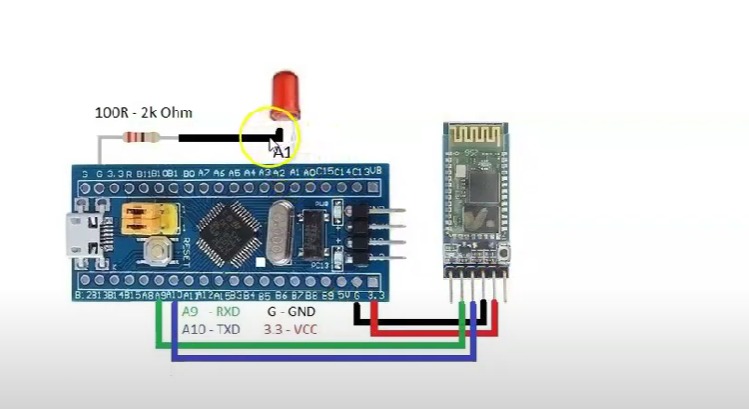
**E**: PA3

LCD ports are initialized through Lcd\_create() with appropriate configurations.

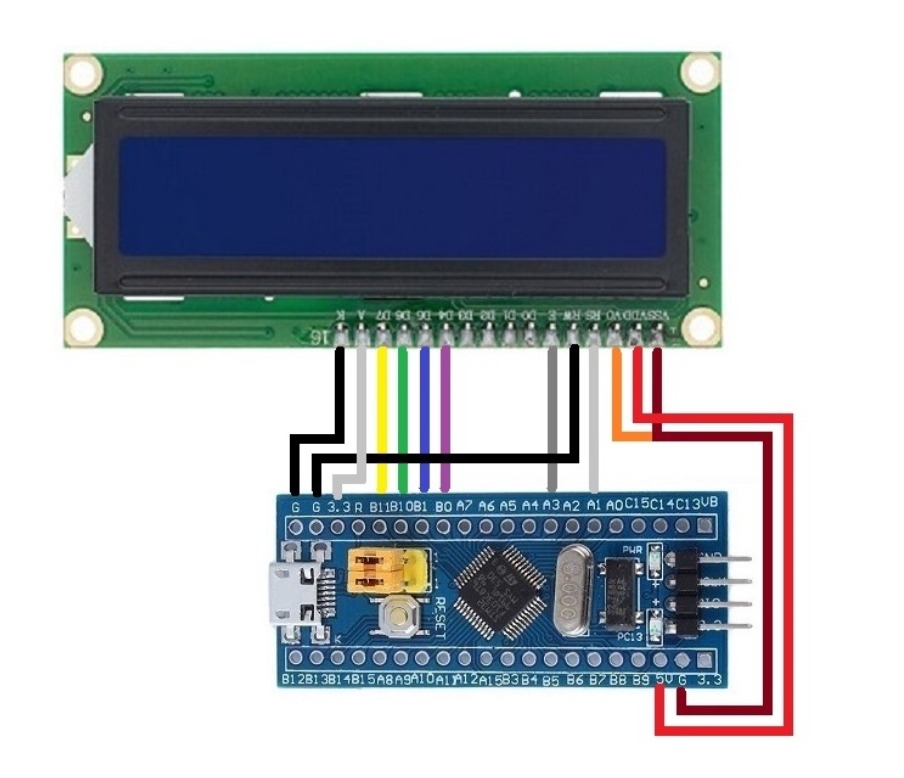
**Configurations**



**Circuit Diagram**



**LCD Connection**



**Software Components and Functions**

**Code Structure**

* **Main Program:** Initializes peripherals, sets up UART and LCD, and runs a continuous loop to process commands.
* **UART Callback:** Handles data reception and triggers message parsing.
* **Message Parser:** Interprets incoming commands (e.g., line display, screen clearing, scroll initiation).
* **Scroll Text Function : scroll\_text()** Manages timing and scroll positioning for lengthy messages.

**1. Initialization and Main Loop**

* **System Initialization**:
  + HAL\_Init(), SystemClock\_Config(), MX\_GPIO\_Init(), and MX\_USART1\_UART\_Init() functions initialize the MCU, GPIO, and UART.
* **LCD Initialization**:
  + Lcd\_create() sets the LCD in 4-bit mode.
  + Lcd\_clear() initializes the LCD screen to display the default "input" message.
* **UART Interrupt-based Reception**:
  + HAL\_UART\_Receive\_IT() initiates an interrupt-based UART reception.
  + HAL\_UART\_RxCpltCallback() processes incoming data by storing it in a buffer until a newline character ('\n') is received, indicating the end of a message.

**2. Processing Commands**

The process\_command() function interprets messages based on command prefixes:

* **CLR Command**: Clears the screen and displays "Screen Cleared" for 2 seconds before resetting.
* **LINE1 and LINE2 Commands**: Display text on specific lines:
  + "1 " prefix places the message on line 1.
  + "2 " prefix places the message on line 2.
* **SCROLL Command**: Initiates scrolling for messages longer than 16 characters.
  + If a message is shorter than 16 characters, it is displayed without scrolling.
  + scroll\_text() implements the scrolling logic with a configurable interval, creating a continuous loop of the text.

**3. Scroll Function**

The scroll\_text() function displays messages longer than 16 characters by scrolling through the text, (If the entire message is shorter than 16 characters, it is displayed without scrolling.) with these key features:

* **Display Buffer**: A 16-character buffer (display\_buffer) displays a portion of the message.
* **Scrolling Control**: scroll\_interval controls the delay between shifts. (Here 300ms)
* **Continuous Scrolling**: Resets the scroll position to the start after reaching the end.

**4. Error Handling**

In case of a UART or LCD error, Error\_Handler() disables interrupts and enters an infinite loop, allowing for error monitoring and debugging.

**Conclusion**

This project successfully demonstrates an STM32-based Bluetooth-controlled LCD system, capable of receiving and displaying messages. The modular code structure and clear command-based controls allow for easy system expansion. Future improvements could include:

* Adding support for more commands (e.g., changing scroll speed).
* Using a more advanced user interface for error reporting.
* Implement SPI or I2C protocols to support more peripherals.