**Simulation of Traffic Light using STM32 Nucleo board**

Aar Mhan Phyo

Student ID: 0268bk9c(PSB)

22072426(UH)

Module: Digital Electronics

Module code: 4FTC2047

Date of Submission: 3rd July,2023

**Introduction**

The objective of this project is to control a set of LEDs in a traffic light style using the NUCLEO-F401RE microcontroller. The project also includes an additional component, a 7-segment display, as my extra work. The microcontroller will read the state of a push-button switch and operate the LEDs accordingly. The 7-segment display will be utilized to show a countdown during the red LED phase. Additionally, the project will implement a feature where pressing the button twice will switch off the green LED and activate a blinking amber LED in an infinite loop.

**Hardware Setup**

The hardware setup for this project includes the following components:

* NUCLEO-F401RE microcontroller
* Breadboard
* Red, green, and yellow LEDs
* Push-button switch
* Resistors for current limiting

The LEDs are connected to the breadboard with appropriate current-limiting resistors to prevent excessive current flow.

1.Connect the microcontroller to the breadboard:

* + Connect the 3.3V pin (VCC) of the microcontroller to the positive rail of the breadboard.
  + Connect the GND pin of the microcontroller to the ground rail of the breadboard.

2.Connect the LEDs to the microcontroller:

* Connect the anode (longer leg) of the red LED to a digital output pin (e.g., PC\_10) of the microcontroller.
* Connect the cathode (shorter leg) of the red LED to one end of a current-limiting resistor (typically 220-470 ohms).
* Connect the other end of the resistor to the ground rail of the breadboard.
* Repeat the above steps for the green LED (connect to PC\_3) and the yellow LED (connect to PC\_12).

3.Connect the 7-segment display to the microcontroller:

* Connect the segment a of the 7-segment display to a digital output pin (e.g., PC\_8) of the microcontroller.
* Connect segment b to PC\_6, segment c to PC\_5, segment d to PA\_12, segment e to PA\_11, segment f to PC\_0, segment g to PC\_4, and the decimal point (dp) to PC\_9.
* Connect the common cathode or anode of the 7-segment display to the ground rail or the positive rail of the breadboard, respectively.

**Software Implementation**

The software implementation is carried out using the mbed platform and the provided mbed library. The code is written in C++ and follows a structured approach to control the LEDs and interact with the button.

* Pin Definitions: The first step is to define the pin assignments for the LEDs (red, green, and yellow) and the button. This is done using the DigitalOut and DigitalIn objects, which allow us to control the pins and read the button state.
* 7-Segment Display Function: To facilitate the countdown functionality, a function is implemented to display numbers on the 7-segment display. This function takes an input number and maps the corresponding pin states to display the desired number.
* Initialization: The code initializes variables and states required for the operation. This includes setting the initial state of LEDs, initializing button states and variables, and configuring the 7-segment display pins.
* Main Loop: The program enters an infinite loop to continuously monitor the button state and control the LEDs accordingly.
* Button Interaction: Inside the loop, the current state of the button is read. If the button is pressed and the button was not previously pressed, the following actions are executed:
  + Increment the button press count.
  + If the button press count is 2, turn off the green LED, turn on the yellow LED, and enter an infinite loop for blinking the amber LED.
  + Otherwise, turn on the green LED for 3 seconds, followed by the yellow LED for 3 seconds, and then the red LED for 3 seconds.
  + During the red LED phase, countdown from 3 to 1 on the 7-segment display using the displayNumber() function.
* Button Reset: If the button is not pressed, the button pressed flag is reset to allow for detecting subsequent button presses.
* Loop Continuation: The loop continues indefinitely, repeating the button interaction and LED control process.

**Result**

The implemented code successfully controls the LEDs and interacts with the button on the NUCLEO-F401RE board. The LEDs respond accordingly based on button presses. When the button is pressed once, the green LED turns on for 3 seconds, followed by the yellow LED for 3 seconds, and then the red LED for 3 seconds. Simultaneously, the 7-segment display counts down from 3 to 1. If the button is pressed twice, the green LED turns off, and the amber LED starts blinking in an infinite loop.

**Source code:**

| #include "mbed.h"  // Define the LED pins connected to the breadboard DigitalOut **redLED**(PC\_10); DigitalOut **yellowLED**(PC\_12); DigitalOut **greenLED**(PC\_3);  // Define the button pin DigitalIn **button**(PC\_13);  // Define the LED pin on the microcontroller DigitalOut **ld2**(LED1);  // Optional start // Define 7-segment pins DigitalOut **a**(PC\_8); DigitalOut **b**(PC\_6); DigitalOut **c**(PC\_5); DigitalOut **d**(PA\_12); DigitalOut **e**(PA\_11); DigitalOut **f**(PC\_0); DigitalOut **g**(PC\_4); DigitalOut **dp**(PC\_9);  // Function to display a number on the 7-segment display  void **displayNumber**(int num) { switch (num) { case 0: a = b = c = d = e = f = 1; g = 0; break; case 1: b = c = 1; a = d = e = f = g = 0; break; case 2: a = b = g = e = d = 1; c = f = 0; break; case 3: a = b = c = d = g = 1; e = f = 0; break; case 4: b = c = f = g = 1; a = d = e = 0; break; case 5: a = c = d = f = g = 1; b = e = 0; break; case 6: a = c = d = e = f = g = 1; b = 0; break; case 7: a = b = c = 1; d = e = f = g = 0; break; case 8: a = b = c = d = e = f = g = 1; break; case 9: a = b = c = d = f = g = 1; e = 0; break; default: a = b = c = d = e = f = g = 0; break; } } // end  int **main**() { bool buttonPressed = false; bool greenLedOn = true; greenLED = 1; int buttonPressCount = 0;  // Check the initial state of the button bool prevButtonState = button; while (1) { bool currentButtonState = button;  if (currentButtonState && !prevButtonState && !buttonPressed) { buttonPressed = true; buttonPressCount++;  if (buttonPressCount == 2) { greenLED = 0; // Turn off green LED yellowLED = 1; // Turn on yellow LED  // Blink the amber LED in an interval of 1 second while (1) { yellowLED = !yellowLED; // Toggle the amber LED state wait\_us(1000000); // Wait for 1 second (1000000 microseconds) } } else { greenLED = 1; // Turn on green LED wait\_us(3000000); // Wait for 3 seconds (3000000 microseconds)  greenLED = 0; // Turn off green LED yellowLED = 1; // Turn on yellow LED wait\_us(3000000); // Wait for 3 seconds (3000000 microseconds)  yellowLED = 0; // Turn off yellow LED ld2 = 1; redLED = 1; // Turn on red LED  // Count down from 3 to 1 on the 7-segment display // This loop is for 7 segments for (int i = 3; i >= 1; i--) { displayNumber(i); wait\_us(1000000); // Wait for 1 second (1000000 microseconds) }  redLED = 0; // Turn off red LED ld2 = 0; greenLED = 1;  // Reset the 7-segment display to display nothing displayNumber(-1); } } else if (!currentButtonState) { buttonPressed = false; } prevButtonState = currentButtonState; } }  **Future Improvements: The code can be enhanced in several ways:**  1.Implement debouncing for the button to ensure reliable button press detection.  2.Add error handling and robustness checks to handle unexpected scenarios.  3.Expand the functionality by incorporating additional sensors or actuators.  4.Improve the user interface by adding feedback mechanisms such as sound or display messages.  **Conclusion:**  In conclusion, the developed code provides a functional solution for controlling LEDs and interacting with a button on the NUCLEO-F401RE microcontroller board. The addition of the 7-segment display allows for a visual countdown when the red LED is turned on. The code can serve as a basis for further projects requiring LED control and button interactions, and it demonstrates the versatility and capabilities of the NUCLEO-F401RE board.Overall, this project has opened up a world of possibilities for me in the realm of embedded systems. |
| --- |