VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

B. Tech – Electronics & Communication Engineering

BECE403E – EMBEDDED SYSTEMS DESIGN

LAB RECORD

(lab slot L27+L28)

Submitted By

22BEC1205 – Jayakrishnan Menon

Submitted To

Dr. S.Muthulakshmi

DATE: 03/01/2025

Slot: L27+L28

Date: 03/01/2025

LAB – 02: Working with LEDs & Switches

AIM: To understand the interfacing of Nucleo64-STM32L152RE Board with LEDs and Switches, and also to perform the following tasks:

Lab Task-1: Write a C++ program with mbed APIs to display Hexadecimal Counting Pattern from 0 to 15 by blinking LEDs. Implement and verify this logic on the STM32 Nucleo-64 board using Keil Studio Cloud IDE.

Lab Task-2: Write a C++ program with mbed APIs to blink one LED at a time serially in a group of 4 LEDs using switch.
If switch is LOW, blink the LEDs one at a time from left to right (1000, 0100, 0010, 0001 pattern) with 0.5 Sec delay between each.
If switch is HIGH, blink the LEDs one at a time from right to left (0001, 0010, 0100, 1000 pattern) with 0.5 Sec delay between each.

Lab Task-3: Write a C++ program with mbed APIs to design a traffic light controller system for a four lane junction (North, South, East, West) to coordinate the traffic moves.

SOFTWARE REQUIRED: ARM Keil Studio (Mbed Online Compiler)

HARDWARE REQUIRED: Micro USB cable, NUCLEO64-STM32L152 Board, LEDs, Jumper Wires (M-F and M-M), Breadboard

PROCEDURE:

1. Go to ARM Keil Studio (https://studio.keil.arm.com) and log in
2. Select File → New → Mbed Project
3. Click the Example project drop-down list and select “mbed2-example-blinky”
4. In Project name field, provide the name of the new project and click Add project
5. Double click on the “main.cpp” file from the newly created project folder
6. Modify the code in the editor window as per the logic of your application
7. Check for any errors in the program under the “Problems” tab of the panels window
8. If no errors, connect the Nucleo Board to the computer using Micro USB Cable
9. Click Play icon (Run project) to upload and start the code execution on the board.

PROGRAMS:

Lab Task 1: Write a C++ program with mbed APIs to display Hexadecimal Counting Pattern from 0 to 15 by blinking LEDs. Implement and verify this logic on the STM32 Nucleo-64 board using Keil Studio Cloud IDE.

Code:

```
#include "mbed.h"
BusOut myled(PC_4,PB_13,PB_14,PB_15);
int i;
int main() {
  while(1) {
      myled=0x00;
      for(i=0;i<16;i++){
          myled=myled+1;
          wait(0.5);
      }
  }
}
```

Output:

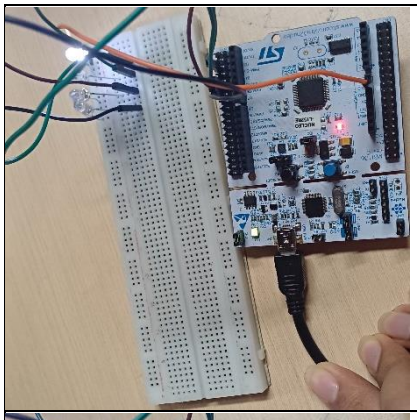


Fig. 1.1: Hexadecimal Counting Pattern 0x01 displayed for 0.5 seconds

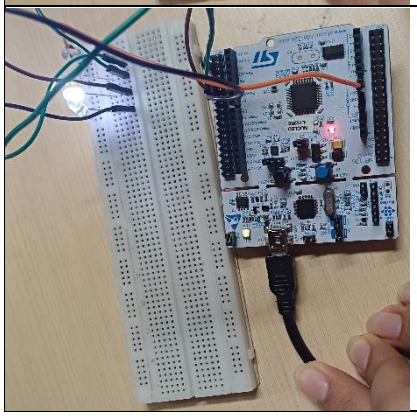
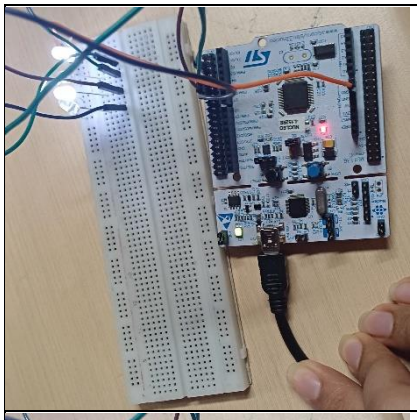
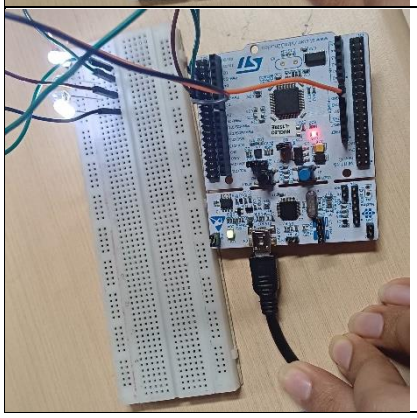


Fig. 1.2: Hexadecimal Counting Pattern 0x04 displayed for 0.5 seconds

	**Fig. 1.3:** Hexadecimal Counting Pattern 0x05 displayed for 0.5 seconds
	**Fig. 1.4:** Hexadecimal Counting Pattern 0x09 displayed for 0.5 seconds

Lab Task 2: Write a C++ program with mbed APIs to blink one LED at a time serially in a group of 4 LEDs using switch. If switch is LOW, blink the LEDs one at a time from left to right (1000, 0100, 0010, 0001 pattern) with 0.5 Sec delay between each. If switch is HIGH, blink the LEDs one at a time from right to left (0001, 0010, 0100, 1000 pattern) with 0.5 Sec delay between each.

Code:

```
#include "mbed.h"
DigitalIn switch1(PC_13);
BusOut myled(PC_4,PB_13,PB_14,PB_15);

int main() {
  while(1) {
      if(switch1==0){
          myled=8;
          wait(0.5);
          myled=4;
          wait(0.5);
          myled=2;
          wait(0.5);
          myled=1;
          wait(0.5);
      }
      else{
          myled=1;
          wait(0.5);
          myled=2;
          wait(0.5);
          myled=4;
          wait(0.5);
          myled=8;
          wait(0.5);
      }
  }
}
```

Output:

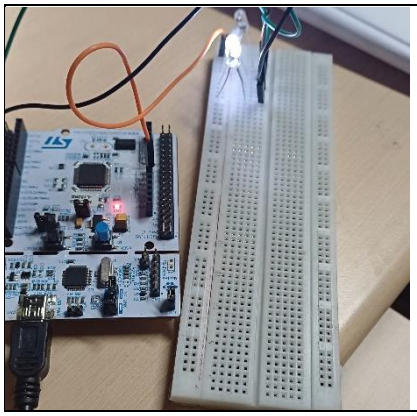
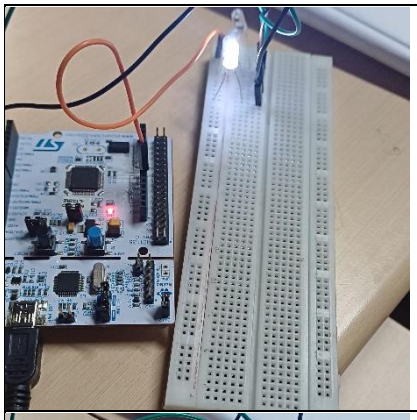
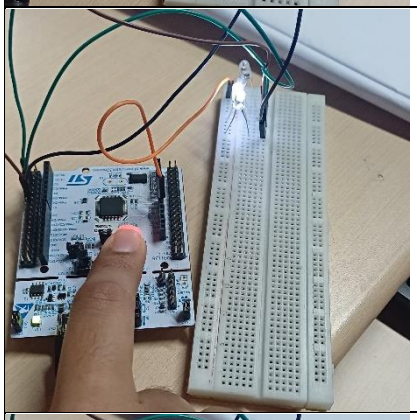
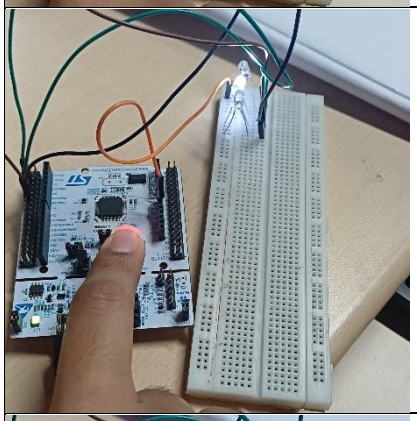
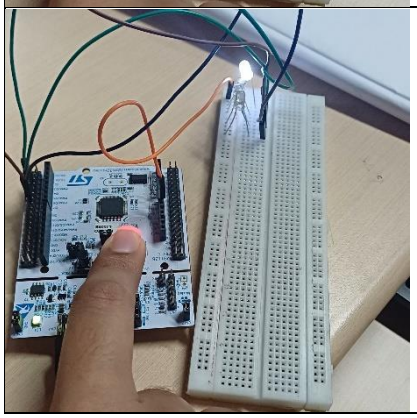


Fig. 2.1: 3rd LED from the top is in ON state and the rest are in OFF state for 0.5 seconds (The Button is not pressed so the order of blinking is top to down)

	**Fig. 2.2:** 4th LED from the top is in ON state and the rest are in OFF state for 0.5 seconds (The Button is not pressed so the order of blinking is top to down)
	**Fig. 2.3:** 3rd LED from the top is in ON state and the rest are in OFF state for 0.5 seconds (The Button is pressed and the order of blinking is reversed, Bottom to up)
	**Fig. 2.4:** 2nd LED from the top is in ON state and the rest are in OFF state for 0.5 seconds (The Button is pressed and the order of blinking is reversed, Bottom to up)
	**Fig. 2.5:** 1st LED from the top is in ON state and the rest are in OFF state for 0.5 seconds (The Button is pressed and the order of blinking is reversed, Bottom to up)

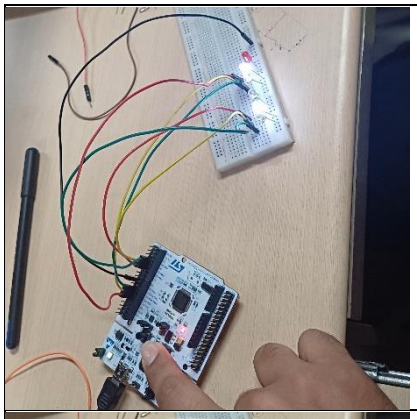
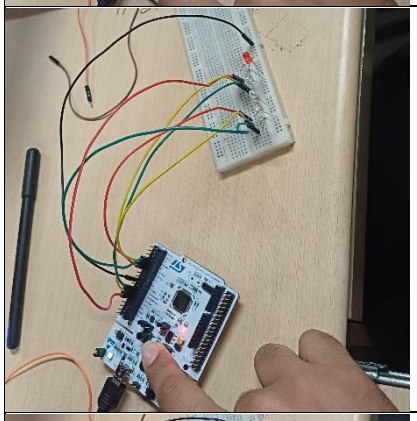
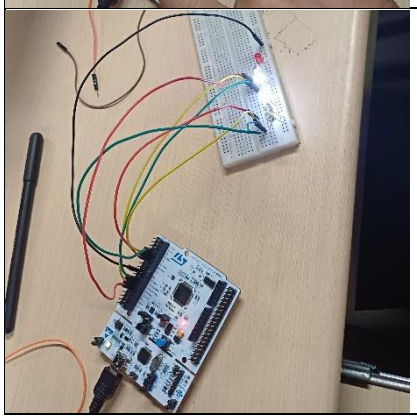
Lab Task 3 (Challenging Task): Write a C++ program with mbed APIs to design a traffic light controller system for a four lane junction (North, South, East, West) to coordinate the traffic moves.

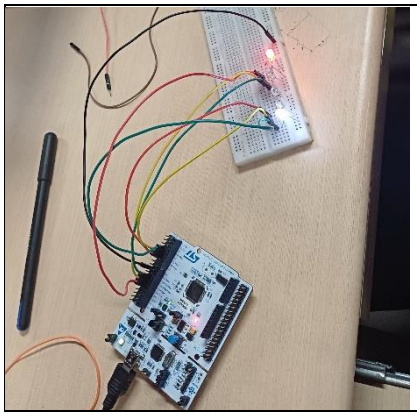
Code:

```
#include "mbed.h"
DigitalOut r1(PB_4);
DigitalOut y1(PB_5);
DigitalOut g1(PB_3);
DigitalOut r2(PA_5);
DigitalOut y2(PA_6);
DigitalOut g2(PA_7);
DigitalIn switch1(PC_13);
int main() {
  while(1) {
    if(switch1==1){
          r1=1;
          y1=0;
          g1=0;
          r2=1;
          y2=0;
          g2=0;
          wait(1);
          r2=0;
          y2=1;
          g2=0;
          wait(2);
          r2=0;
          y2=0;
          g2=1;
          wait(15);
          r1=1;
          y1=0;
          g1=0;
          r2=1;
          y2=0;
          g2=0;
          wait(1);
          r1=0;
          y1=1;
          g1=0;
          wait(2);
          r1=0;
          y1=0;
          g1=1;
          wait(15);
      }
    else{
          r1=0;
          r2=0;
          g1=0;
```

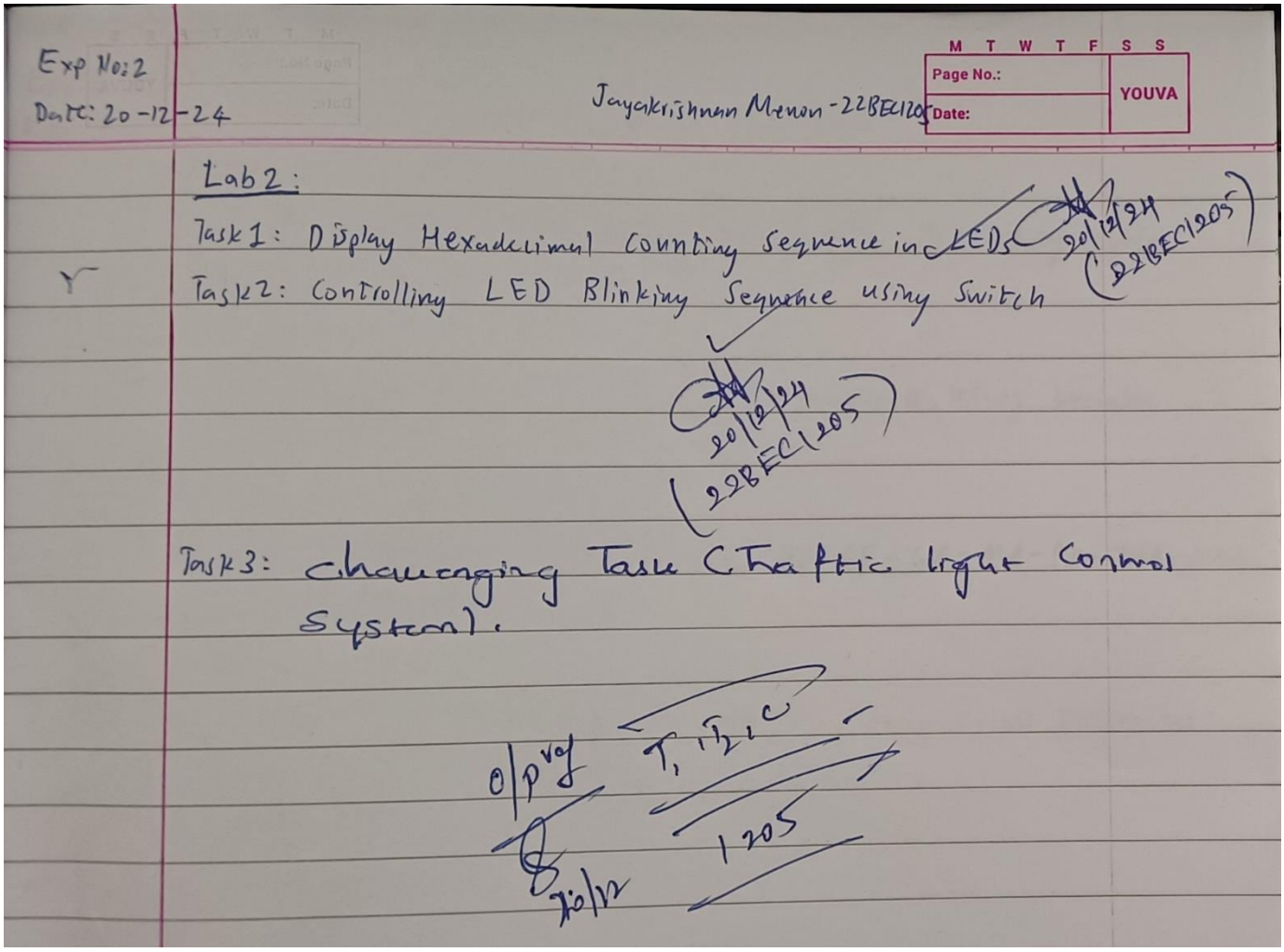
```
            g2=0;
            y1=1;
            y2=1;
            wait(2);
            r1=0;
            r2=0;
            g1=0;
            g2=0;
            y1=0;
            y2=0;
            wait(2);
        }
    }
}
```

Output:

	**Fig. 3.1:** The Button is pressed; hence the device is in the late-night mode. The LEDs corresponding to the YELLOW lights are in the ON state for the period of 2 seconds.
	**Fig. 3.2:** The Button is pressed; hence the device is in the late-night mode. The LEDs corresponding to the YELLOW lights are in the OFF state for the period of 2 seconds.
	**Fig. 3.3:** The Button is not pressed; hence the device is in the normal signaling operation. In this picture, the LED corresponding to the GREEN light (of the N-S lane) is in the ON state for the period of 15 seconds and the LED corresponding to the RED light (of the E-W lane) is in the ON state for the entire duration. This is to ensure that accidents don't happen on the intersection of the lanes.

	**Fig. 3.3:** The Button is not pressed; hence the device is in the normal signaling operation. In this picture, the LED corresponding to the GREEN light (of the E-W lane) is in the ON state for the period of 15 seconds and the LED corresponding to the RED light (of the N-S lane) is in the ON state for the entire duration. This is to ensure that accidents don't happen on the intersection of the lanes.

OUTPUT VERIFICATION:



INFERENCE:

1. Using the "BusOut" API, we can combine a number of "DigitalOut" pins to write values on all of them using a single statement. Its Syntax is: "BusOut Identifier(PinNames)".

2. The “identifier” of a “BusOut” can be assigned values, inside the main function. Through this, the user can assign different output values to the Output Bus, based on the requirements. It is possible to assign hexadecimal values (eg: 0x08) or decimal values (eg: 8) to it.
3. Using the “DigitalIn” API, we can read the value of a digital input pin. Using this, input devices like switches can be interfaced with the Development Board. Its Syntax is: “DigitalIn Identifier(PinName)”.
4. The Nucleo64-STM32L152RE Board has an inbuilt switch which is accessible to the user, through the pin “PC_13”
5. Programmable Delays can be implemented through the use of the “wait()” function.
6. A program can be set to run indefinitely using the “while(1)” loop

RESULT:

Thus, the interfacing of the Nucleo64-STM32L152RE Board with LEDs and Switches was understood and the tasks were performed successfully.