Aim: - To write C++ code of the given following tasks.

Software Used: -

Cloud Keil Studio IDE and Tera Term

Theory: -

The mbed NXP LPC11U24 Microcontroller in particular is designed for prototyping low-cost USB devices, battery powered applications and 32-bit ARM® Cortex™-M0 based designs. It is packaged as a small DIP form-factor for prototyping with through-hole PCBs, stripboard and breadboard, and includes a built-in USB FLASH programmer.

Specifications of LPC11U24:

* NXP LPC11U24 MCU
* Low power ARM® Cortex™-M0 Core
* 48MHz, 8KB RAM, 32KB FLASH
* USB Device, 2xSPI, I2C, UART, 6xADC, GPIO
* Prototyping form-factor
* 40-pin 0.1" pitch DIP package, 54x26mm
* 5V USB, 4.5-9V supply or 2.4-3.3V battery
* Built-in USB drag 'n' drop FLASH programmer
* mbed.org Developer Website
* Lightweight Online Compiler
* High level C/C++ SDK
* Cookbook of published libraries and projects

API Used:

* Syntax used for digital output:
* DigitalOut (PinName pin)
* For LED blinking we’ve used:
* DigitalOut variable(LEDn); where n= 1,2,3,4
* For delay:
* wait(t); where ‘t’ is in seconds

Task 1: - Write a C++ code with mbed APIS to generate a PWD signal with duty cycle 50%. Verify using buzzer/led and digital oscilloscope.

Code: -

#include "mbed.h"

PwmOut led(PC\_8);

int main()

{

    //specify period first

    led.period(1.0f);

    led.write(0.2f); //set duty cycle, relative t period

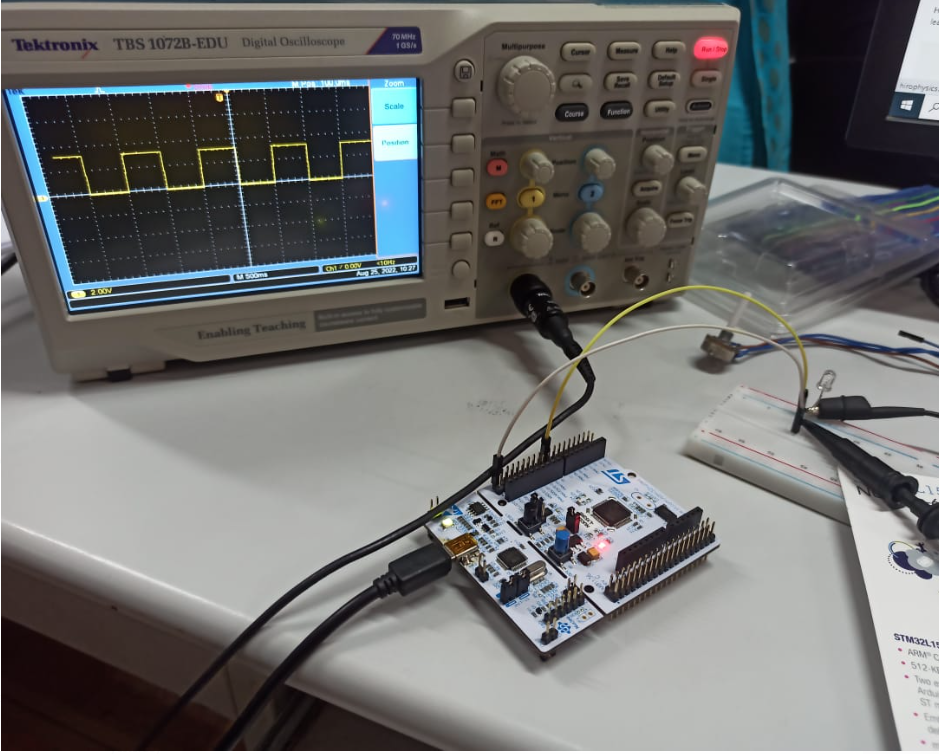
    //led = 0.5f;      //shorthand for led.write();

    //led.pulsewidth\_ms(50);

    while(1); //used to generate continous waveform

}

Output: -



Task 2: - Write a C++ code with mbed APIs generate a PWM signal with duty cycle 20%. Verify the output using LED, Digital oscilloscope and PC Monitor.

Code: -

#include "mbed.h"

#include "mbed2/299/drivers/PwmOut.h"

PwmOut led(PC\_8);

Serial pc(USBTX,USBRX);

int main()

{

    led.period(1.0f);

    led.write(0.2f);

    while(1)

    {

        pc.printf("High");

        wait(0.2);

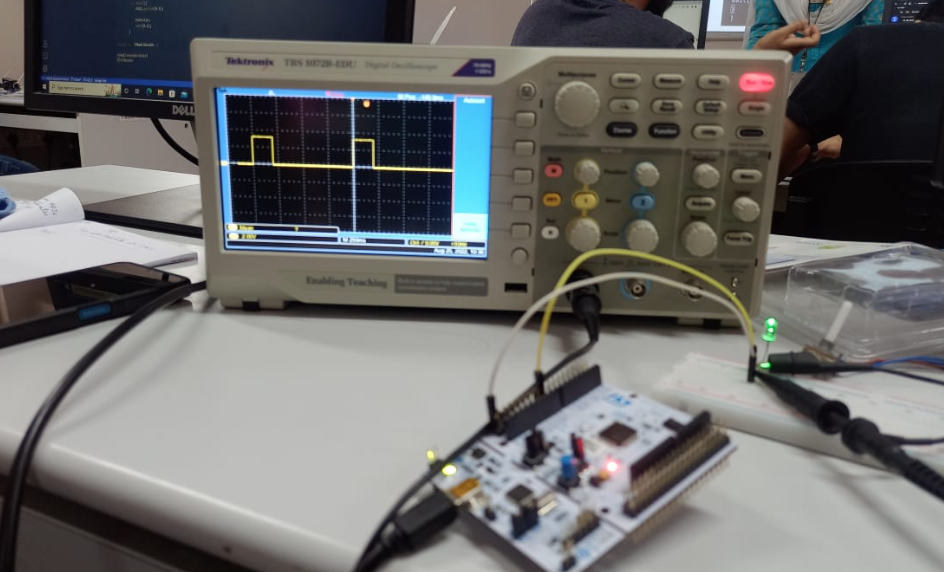
        pc.printf("Low");

        wait(0.8);

    }

}

Output: -



Task 3: - Controlling the Led brightness by using POT on the analog pin while the PWM period remains the same.

Code: -

#include "mbed.h"

PwmOut PWM1(PC\_8);

AnalogIn Ain(PC\_5);

int main()

{

    while(1)

    {

        PWM1.period(0.5);

        //set PWM period to 10ms

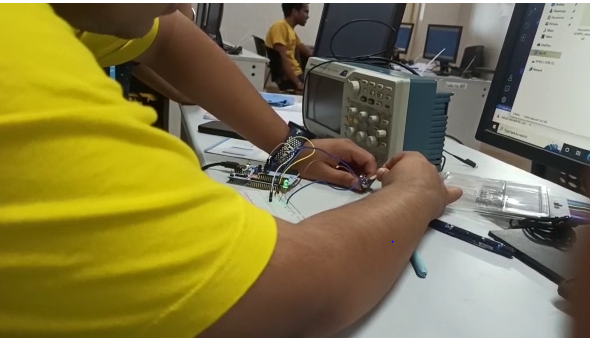
        PWM1 = Ain;

        wait(0.1);

    }

}

Output: -



Task 4: - Generate a PWM signal to increase the brightness of LED when pressed “I” and then decrease the brightness when pressed “d”.

Code: -

#include "mbed.h"

PwmOut PWM1(PC\_8);

PwmOut led(PC\_8);

Serial pc(USBTX, USBRX);

DigitalOut myled(LED1);

int main()

{

    float brightness = 0.0f;

    while(1)

    {

        pc.printf("Press i(to increase) and d(to decrease) the brightness of LED: ");

        char y = pc.getc();

        if(y == 'd'){

            brightness -= 0.1f;

        }

        else if(y == 'i'){

            brightness += 0.1f;

        }

        else{

            pc.printf("Please enter a valid character!");

        }

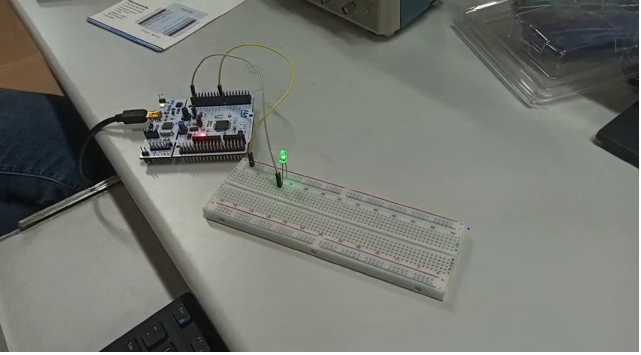
        led.write(brightness);

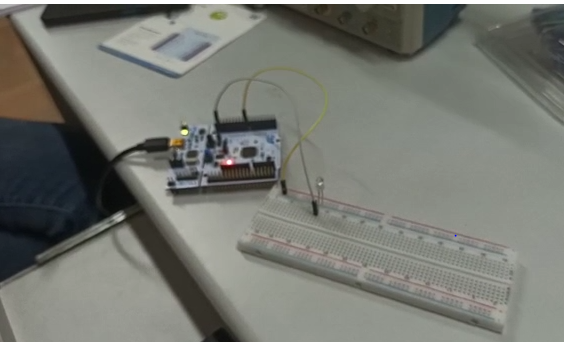
        wait(0.1);

    }

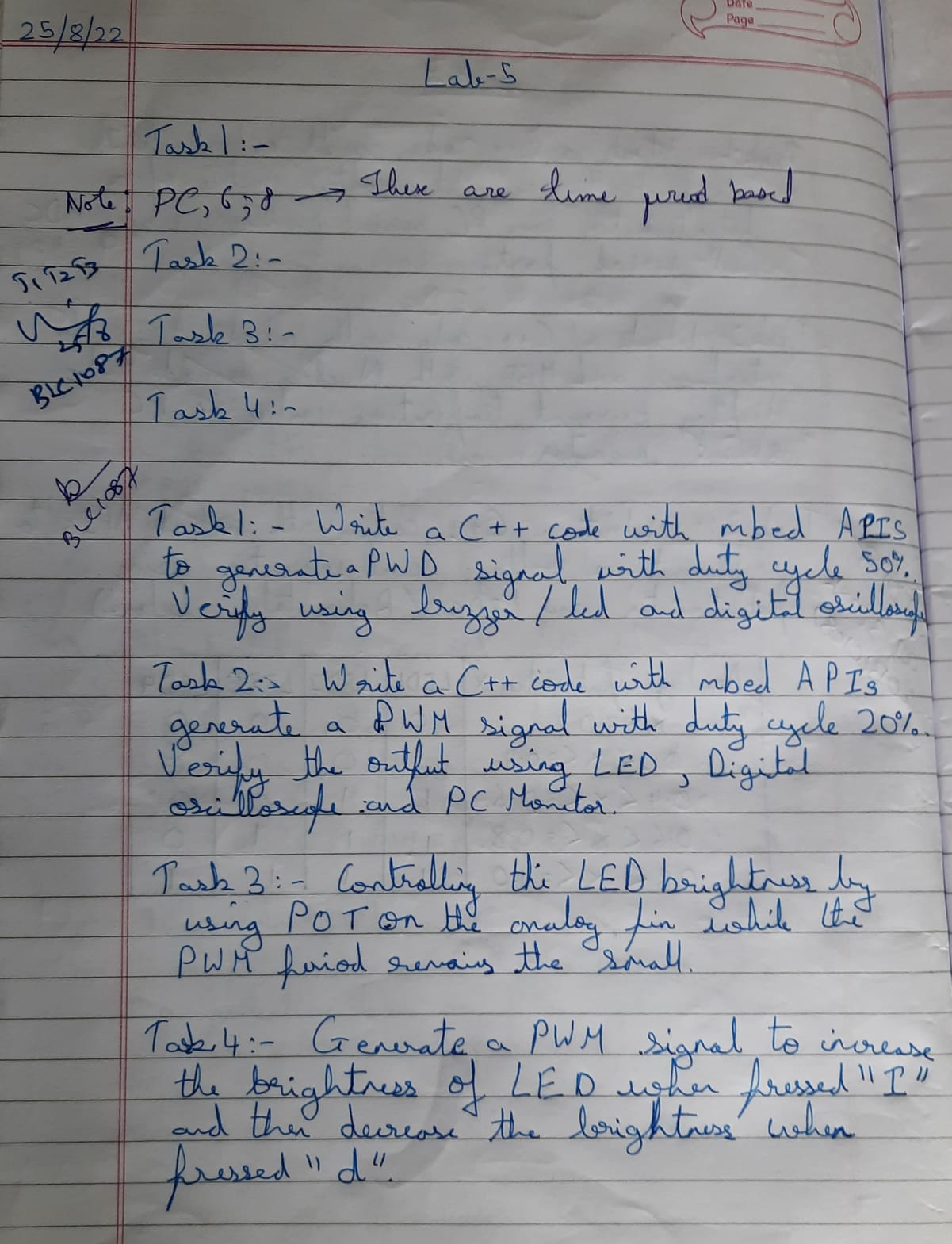
}

Output: -





Verification Status: -



Result: -

Successfully understood and performed all the given tasks.