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: -

Display text typed into the MASTER terminal application on to the slave serial terminal.

Code: -

#include "mbed.h"

SPI ser\_port(3 ports);

DigitalOut cs (1 port);

Serial pc(USBTX,USBRX);

char recv; char y;

int main()

{

    pc.printf("word: ");

    while(1)

    {

        y=pc.getc();

        cs=0;

        pc.putc(y);

        recv=ser\_port.write(y);

        cs=1;

    }

}

wait(0.001);

=================================================================================

#include "mbed.h"

#include "mbed2/299/platform/wait\_api.h"

SPIslave ser\_port(PA\_5,PA\_6,PA\_7,PB\_6);//mosl,miso,sclk,ssel

Serial pc(USBTX,USBRX);

char switch\_word; char rec\_value;

int main()

{

    while(1)

    {

        if (ser\_port.receive())

        {

            rec\_value=ser\_port.read();

            pc.printf("%c",rec\_value);

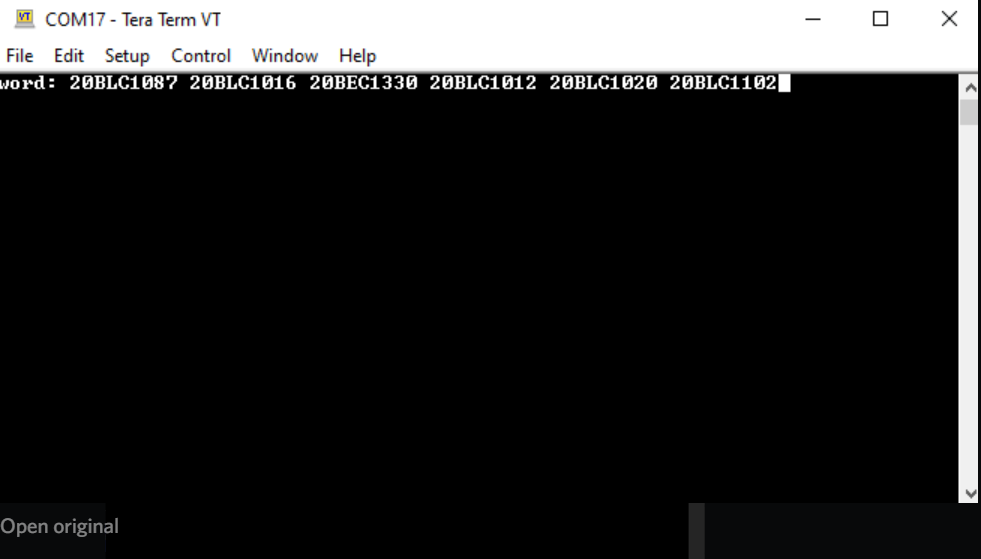
        }

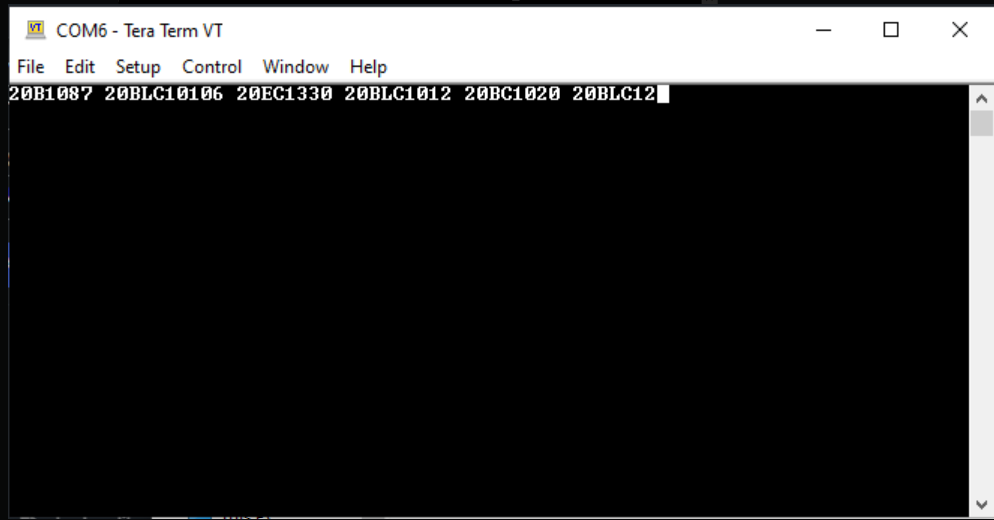
    }

}

wait(0.001);

Output: -





Task 2: -

Set the mbed up as Master, and exchange data with a slave, sending its own switch positions, and displaying those of the slave using LEDs.

Code: -

#include "mbed.h"

SPI ser\_port(p11,p12,p13);

DigitalOut led1(LED1);

DigitalOut led2(LED2);

DigitalOut cs(p14);

DigitalIn switch\_ip1(p7);

DigitalIn switch\_ip2(p8);

char switch\_ip2(p8);

char switch\_word;

char recd\_val;

int main()

{

    while(1)

    {

        switch\_word=0xa0;

        if (switch\_ip1==1)

        {

            switch\_word=switch\_word|0x01;

        }

        if (switch\_ip2==1)

        {

            switch\_word=switch\_word|0x02;

            cs=0;

            recd\_val=ser\_port.write(switch\_word);

            cs=1;

            wait(0.01);

            led1=0;

            led2=0;

            recd\_val=recd\_val&0x03;

            if(recd\_val==1)

            led1=1;

            if(recd\_val==2)

            led2=1;

            if(recd\_val==3){

                led1=1;

                led2=1;

            }

        }

    }

}

=================================================================================

#include "mbed.h"

SPISlave ser\_port(p11,p12,p13,p14);

DigitalOut led1(LED1);

DigitalOut led2(LED2);

DigitalIn switch\_ip1(p5);

DigitalIn switch\_ip2(p6);

char switch\_word;

char recd\_val;

int main()

{

    while(1)

    {

        switch\_word=0xa0;

        if (switch\_ip1==1)

        switch\_word=switch\_word|0x01;

        if(switch\_ip2==1)

        switch\_word=switch\_word|0x02;

        if (ser\_port.receive())

        {

            recd\_val=ser\_port.read();

            ser\_port.reply(switch\_word);

        }

        led1=0;

        led2=0;

        recd\_val=recd\_val&0x03;

        if (recd\_val==1)

        led1=1;

        if(recd\_val==2)

        led2=1;

        if(recd\_val==3)

        {

            led1=1;

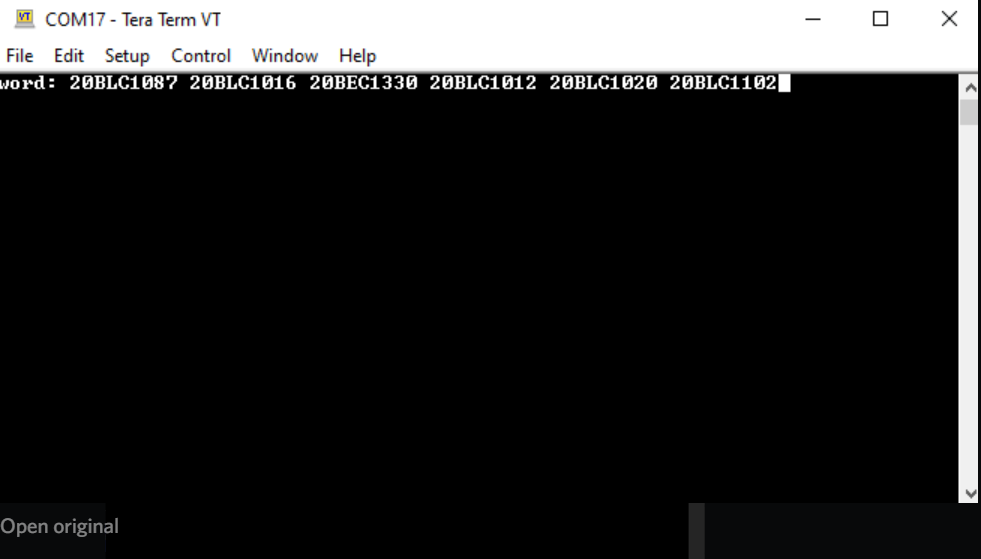
            led2=1;

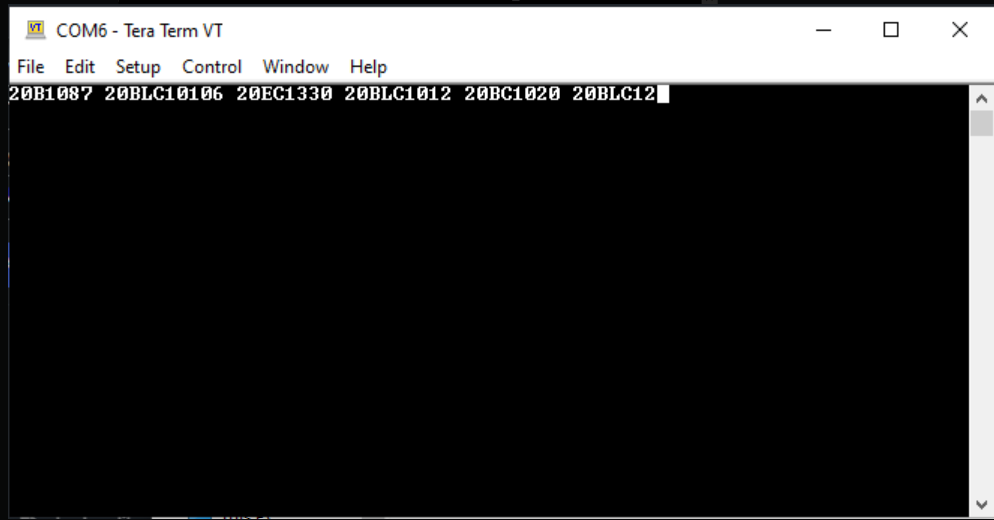
        }

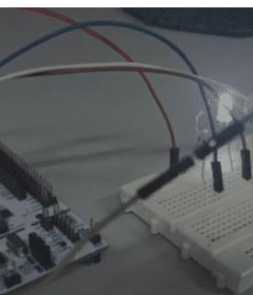
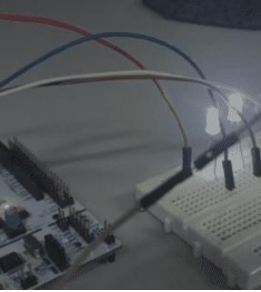
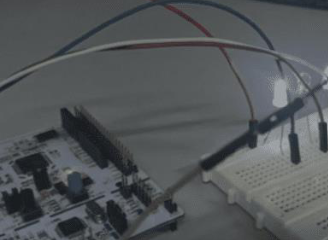
    }

}

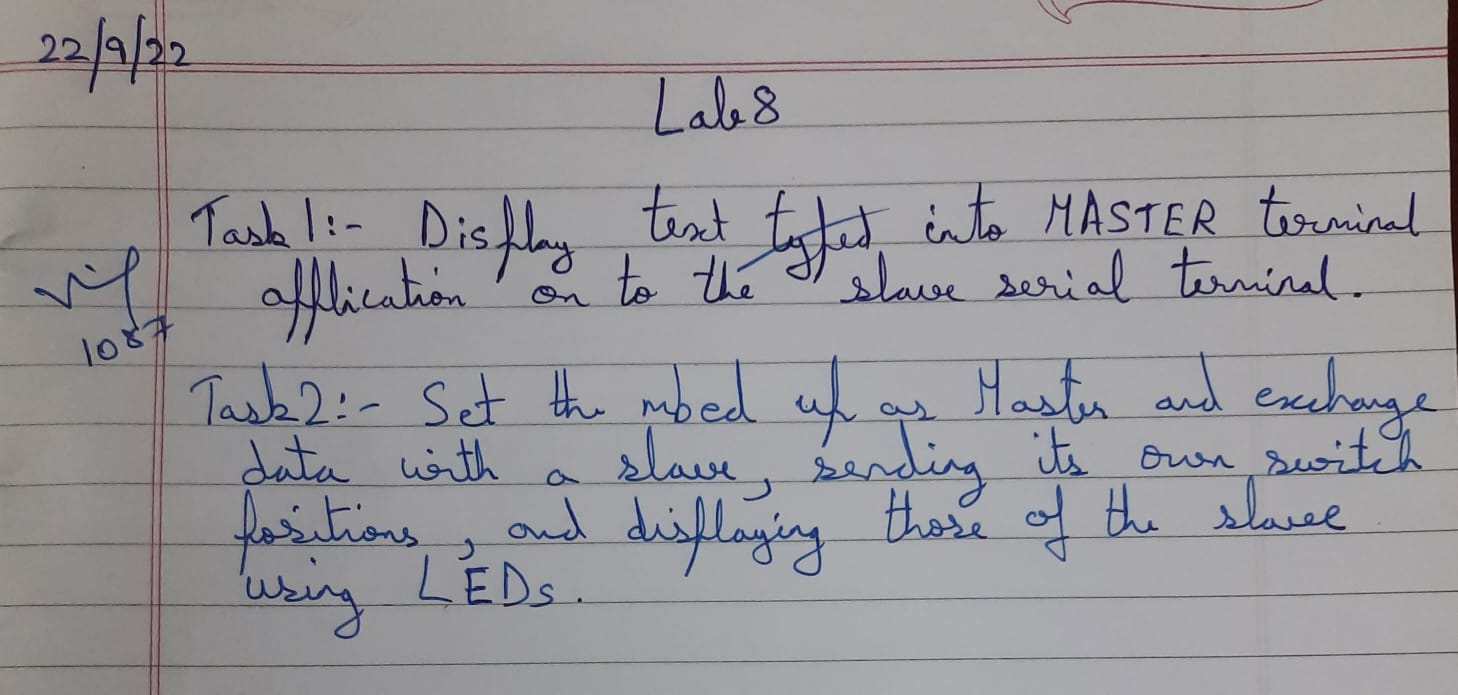
Output: -





Verification Status: -



Result: -

Successfully understood and performed all the given tasks.