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 SYSTEM DESIGN

LAB RECORD

(lab slot L49+L50)

Submitted By

22BEC1205

JAYAKRISHNAN MENON

Submitted To

Dr. S.Muthulakshmi

DATE: 27/03/2025

Slot: L49+L50
Date: 27/03/2025

LAB – 11: SPI

AIM:

To understand the working of Nucleo64-STM32L152RE Board and to perform the following tasks:

Lab Task-1: Read input form master and display on slave.
Lab Task-2: Exchange of Button press state between master and slave.
Lab Task-3: Half Adder.

SOFTWARE REQUIRED:

ARM Keil Studio (MBED Online compiler)

Hardware Required:

Micro USB cable, 2 Nos. NUCLEO64-STM32L152 Board, LEDs, Jumper Wires (M-F and M- M), Breadboards.

Procedure:

1. Log in to ARM Keil Studio.
2. Go to File → New → Mbed Project.
3. Select mbed2-example-blinky from the Example project drop-down.
4. Name your project and click Add project.
5. Open main.cpp from the project folder and edit the code as needed.
6. Check the Problems tab for errors.
7. If error-free, connect the Nucleo Board via Micro USB.
8. Click the Play icon to upload and run the code on the board.

PROGRAM:

Lab Task 1: Read input form master and display on slave.

CODE:

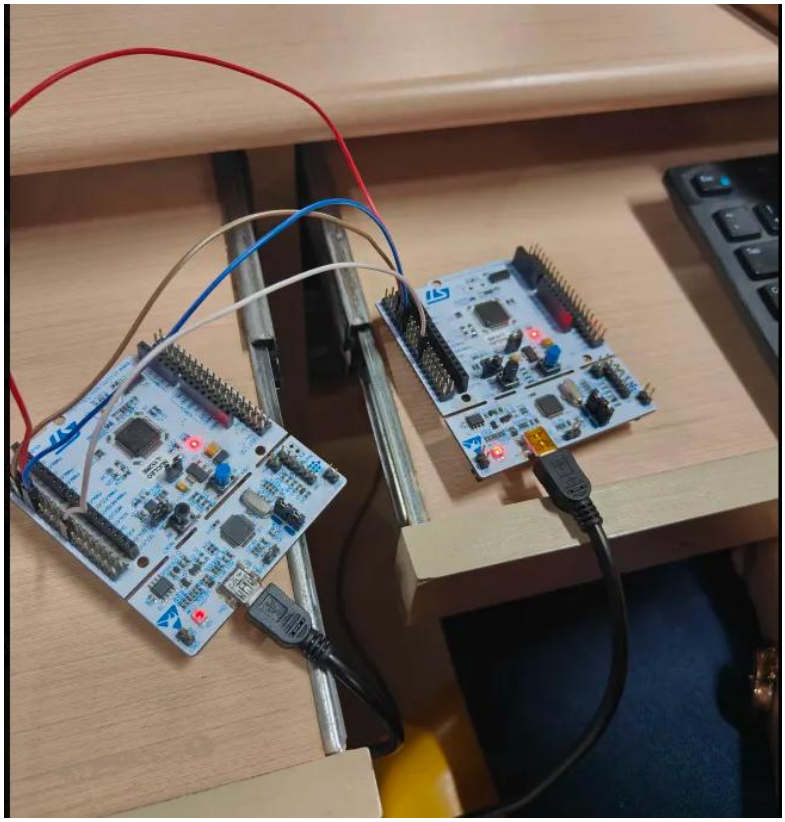
MASTER:

```
#include "mbed.h"

SPI spi(PB_15, PB_14, PB_13);
DigitalOut cs(PB_12);
Serial pc(USBTX, USBRX);

int main() {
char send_val;
pc.printf("Press any key to start...\n");

while (1) {
send_val = pc.getc();
pc.printf("%c", send_val);

cs = 0;
spi.write(send_val);
cs = 1;
wait(0.01);
}
}
```

SLAVE:

```
#include "mbed.h"

SPISlave spi(PB_15, PB_14, PB_13, PB_12);
Serial pc(USBTX, USBRX);

int main() {
char recd_val; pc.printf("Received word is...\n");

while (1) {
if (spi.receive()) { recd_val = spi.read();
pc.printf("%c", recd_val);
}

}
}
```

OUTPUT:



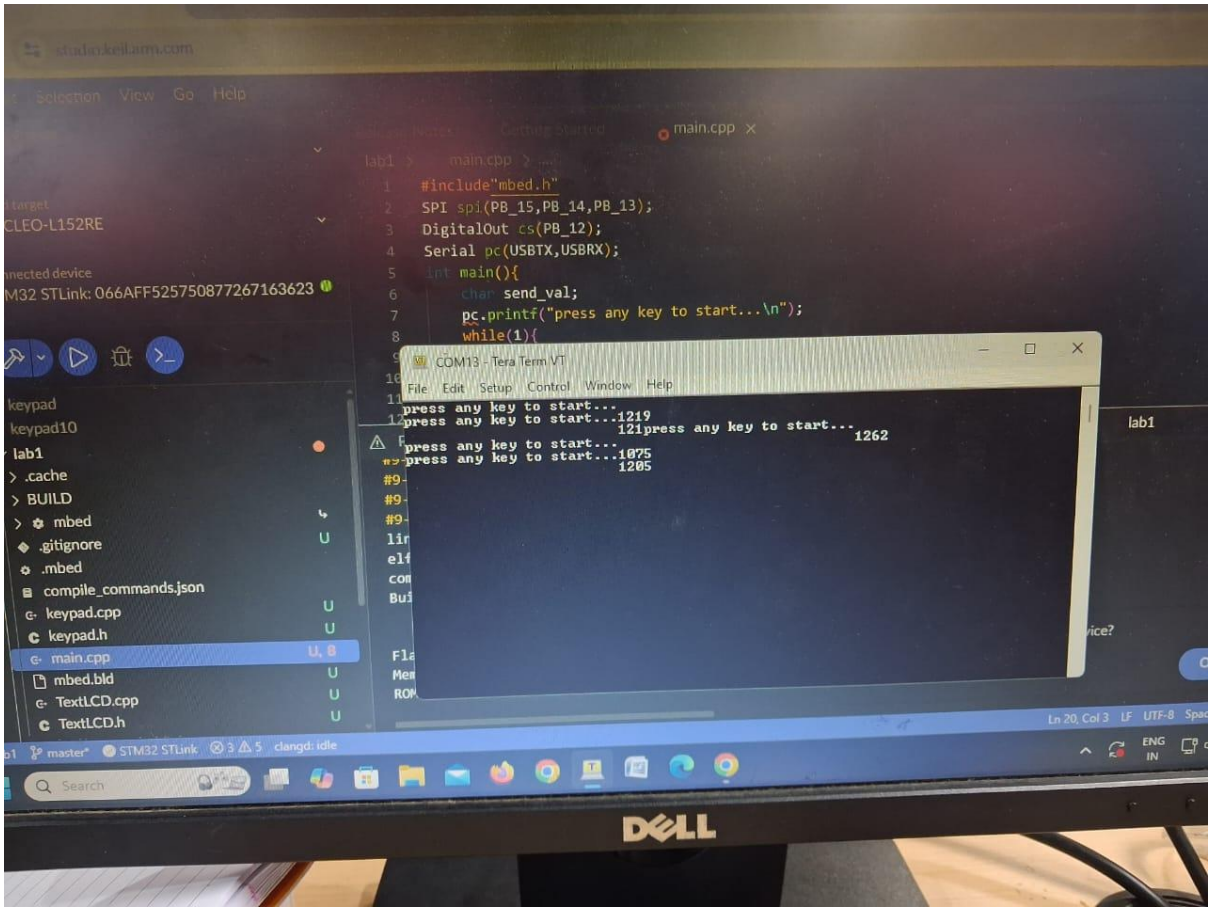


Fig.1: Reading input form master and display on slave.

Lab Task-2: Exchange of Button press state between master and slave.

CODE:

MASTER:

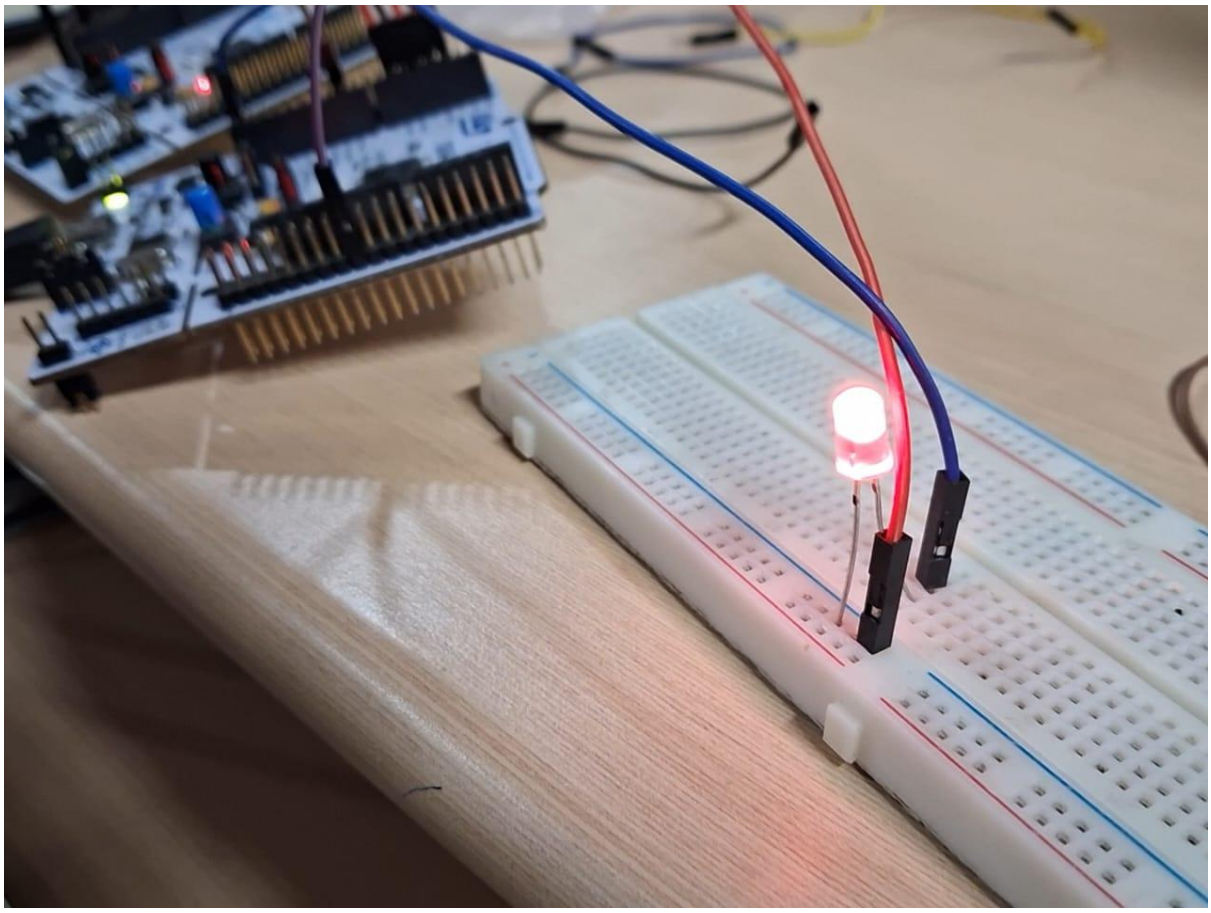
```
#include "mbed.h"
SPI ser_port(PB_15, PB_14, PB_13);
DigitalOut cs(PB_12);
Serial pc(USBTX, USBRX);

DigitalOut led(PC_8);
DigitalIn switch_ip(PC_4);

int switch_word_tx;
char tx_val;

int main() {
```

```
while (1) {
switch_word_tx = 0x00;
if (switch_ip == 1)
switch_word_tx = switch_word_tx | 0x01;
cs = 0;
tx_val = ser_port.write(switch_word_tx);
cs = 1;
wait(1);
led = 0;
tx_val = tx_val & 0x01;

if (tx_val == 1) led = 1;
}
}
```

SLAVE:

```
#include "mbed.h"

SPISlave ser_port(PB_15, PB_14, PB_13, PB_12);
Serial pc(USBTX, USBRX);
DigitalOut led(PC_8);
DigitalIn switch_ip(PC_4);

int switch_word_rx;
char rx_val;

int main() {
while (1) {
switch_word_rx = 0x00;

if (switch_ip == 1) {
switch_word_rx = switch_word_rx | 0x01;
}

if (ser_port.receive()) {
rx_val = ser_port.read();

ser_port.reply(switch_word_rx);
}

led = 0;
rx_val = rx_val & 0x01;

if (rx_val == 1) { led = 1;
}
}
}
```

OUTPUT:



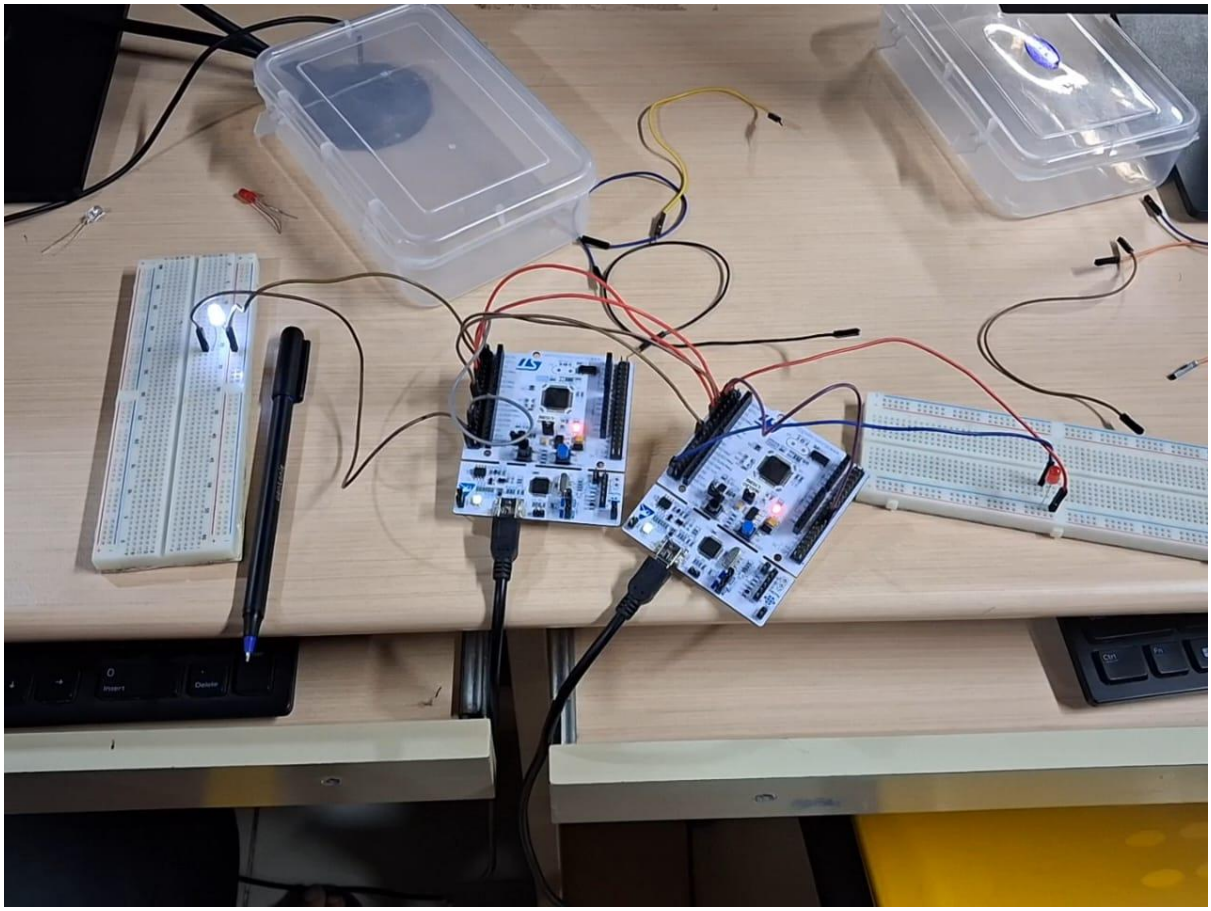


Fig.2: Button Press state between master and slave

Lab Task-3: Half Adder.

CODE:

MASTER:

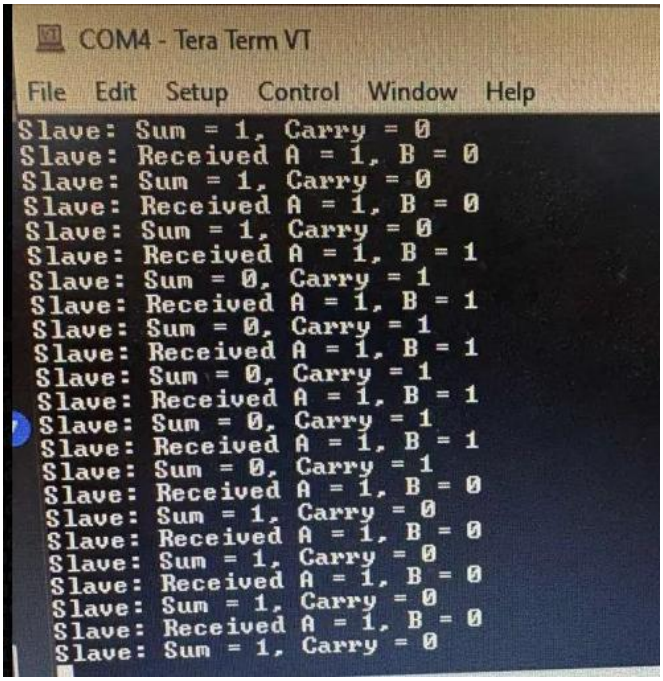
```
#include "mbed.h"

SPI ser_port(PB_15, PB_14, PB_13);
DigitalOut cs(PB_12);
Serial pc(USBTX, USBRX);
DigitalOut sum_led(PC_8);
DigitalIn switch_A(PC_4);

char received_sum;
```

```
int A;
int main() {
ser_port.format(8, 0);
ser_port.frequency(1000000);
cs = 1;
while (1) {
A = switch_A.read();
pc.printf("Master: A = %d\r\n", A);
cs = 0;
wait_us(10);
received_sum = ser_port.write(A);
wait_us(10);
cs = 1;
pc.printf("Master Received Sum: %x\r\n", received_sum);
sum_led = received_sum;
wait(0.5);
}
}
```

SLAVE:

```
#include "mbed.h"

SPISlave ser_port(PB_15, PB_14, PB_13, PB_12);

Serial pc(USBTX, USBRX);
DigitalOut carry_led(PC_8);
DigitalIn switch_B(PC_4);
char A, B, sum, carry;
int main() {
while (1) {
if (ser_port.receive()) {
A = ser_port.read();
B = switch_B.read();
pc.printf("Slave: Received A = %d, B = %d\r\n", A, B);
sum = A ^ B;
carry = A & B;
ser_port.reply(sum);
carry_led = carry;
pc.printf("Slave: Sum = %d, Carry = %d\r\n", sum, carry);
}
}
}
```

OUTPUT:



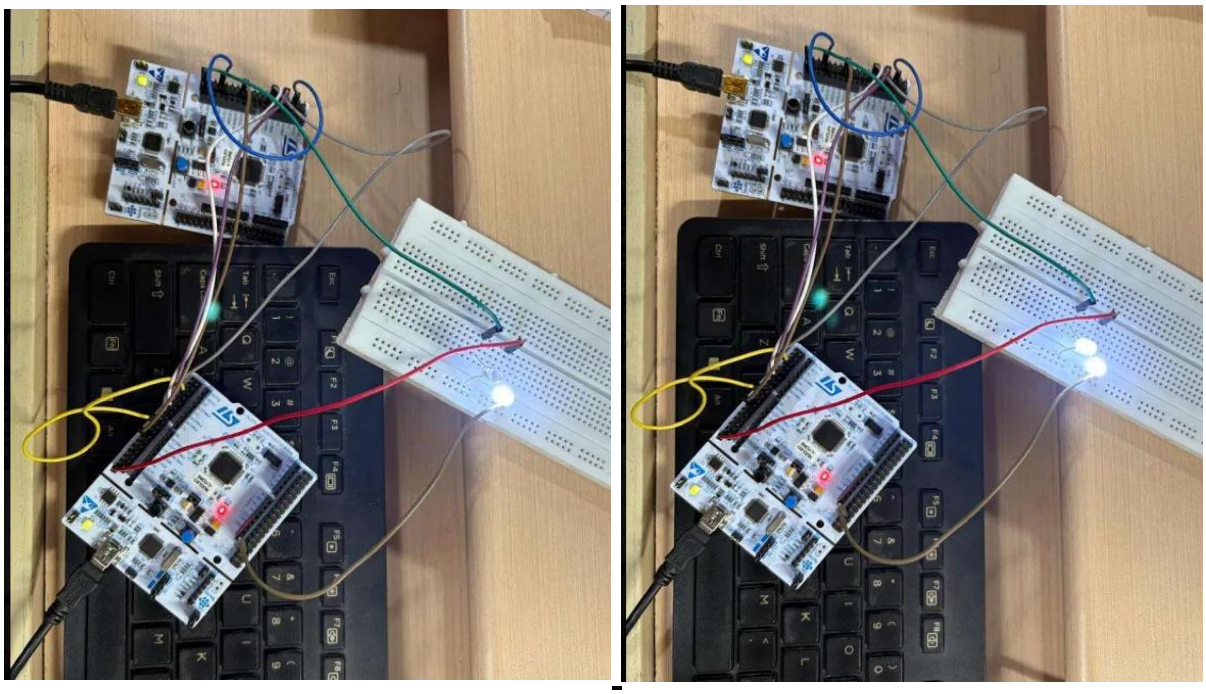
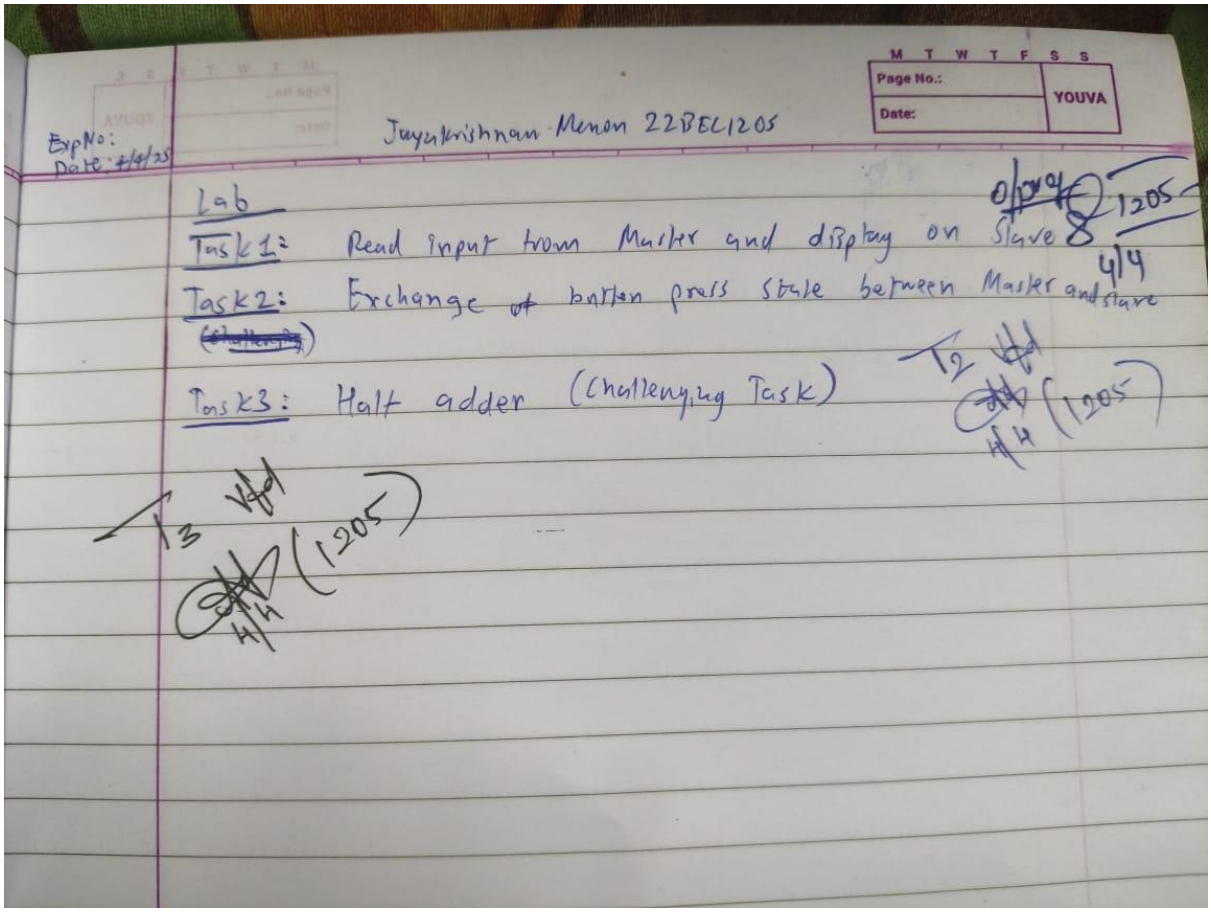


Fig.3: Half Adder Logic

OUTPUT VERIFICATION:



RESULT:

Thus, we have successfully implemented and executed using Nucleo64- STM32L152RE Board and tera term application to perform the following tasks.

- Read the input from Master and display on Slave.
- Exchange of button press state between Master and slave.
- Half Adder Implementation using SPI Protocol.