LAB TASK 12

# SERIAL COMMUNICATION BETWEEN MICROCONTROLERS

# **TASK 1:**

## **SENDER MICROCONTROLLER CODE**

# include <reg51.H>

void Ldelay()

{

unsigned int j;

for(j=0;j<32800;j++);

}

unsigned char WhichNumberPressed()

{

// read the pins from port 0 and return the pressed character

P1 = 0xF0;

asd:

P1 = 0xFE;

if (P1 == 0x0EE) return '7';

if (P1 == 0x0DE) return '8';

if (P1 == 0x0BE) return '9';

if (P1 == 0x07E) return '/';

P1 = 0xFD;

if (P1 == 0x0ED) return '4';

if (P1 == 0x0DD) return '5';

if (P1 == 0x0BD) return '6';

if (P1 == 0x07D) return '\*';

P1 = 0xFB;

if (P1 == 0x0EB) return '1';

if (P1 == 0x0DB) return '2';

if (P1 == 0x0BB) return '3';

if (P1 == 0x07B) return '-';

P1 = 0xF7;

if (P1 == 0x0E7) return ' ';

if (P1 == 0x0D7) return '0';

if (P1 == 0x0B7) return '=';

if (P1 == 0x077) return '+';

if (P1 == 0x0F7 || P1 == 0x0FB || P1 == 0x0FD || P1 == 0x0FE ) goto asd;

}

void senddata(unsigned char dat)

{

SCON = 0x50; /\* SCON: mode 1, 8-bit UART, enable rcvr \*/

TMOD = 0x20; /\* TMOD: timer 1, mode 2, 8-bit reload \*/

TH1 = -3;

TR1 = 1;

SBUF = dat;

while (TI == 0);

TI = 0;

}

unsigned char pressed;

void main()

{

while(1)

{

pressed = WhichNumberPressed();

senddata(pressed);

Ldelay();

}

}

## **RECEIVER MICROCONTROLLER CODE**

# include <REGX51.H>

# define LCD\_data P1

# define LCD\_rs P2\_0

# define LCD\_rw P2\_1

# define LCD\_en P2\_2

void msdelay() // Function for creating delay in milliseconds.

{

unsigned int j;

for(j=0;j<500;j++);

}

void LCD\_command(unsigned char var)

{

LCD\_data = var; //send command

LCD\_rs = 0; //Selected command register

LCD\_rw = 0; //We are writing in instruction register

LCD\_en = 1;

msdelay();

LCD\_en = 0; //Enable H->L

}

void LCD\_senddata(unsigned char var)

{

LCD\_data = var; //send data

LCD\_rs = 1; //Selected data register

LCD\_rw = 0; //We are writing

LCD\_en = 1;

msdelay();

LCD\_en = 0; //Enable H->L

}

void LCD\_init()

{

LCD\_command(0x38); //Function set: 2 Line, 5x7 dots

LCD\_command(0x0F); //Display on, Curson blinking command

LCD\_command(0x01); //Clear LCD

LCD\_command(0x06); // auto increment cursor shift (cursor to right)

}

unsigned char a;

unsigned char Received()

{

TMOD = 0x20;

TH1 = -3;

SCON = 0x50;

TR1 = 1;

while (RI == 0);

a = SBUF;

RI = 0;

return a;

}

unsigned char pressed;

void main()

{

again:

LCD\_init();

P0 = 0x00;

while(1)

{

pressed = Received();

if (pressed == ' ')

goto again;

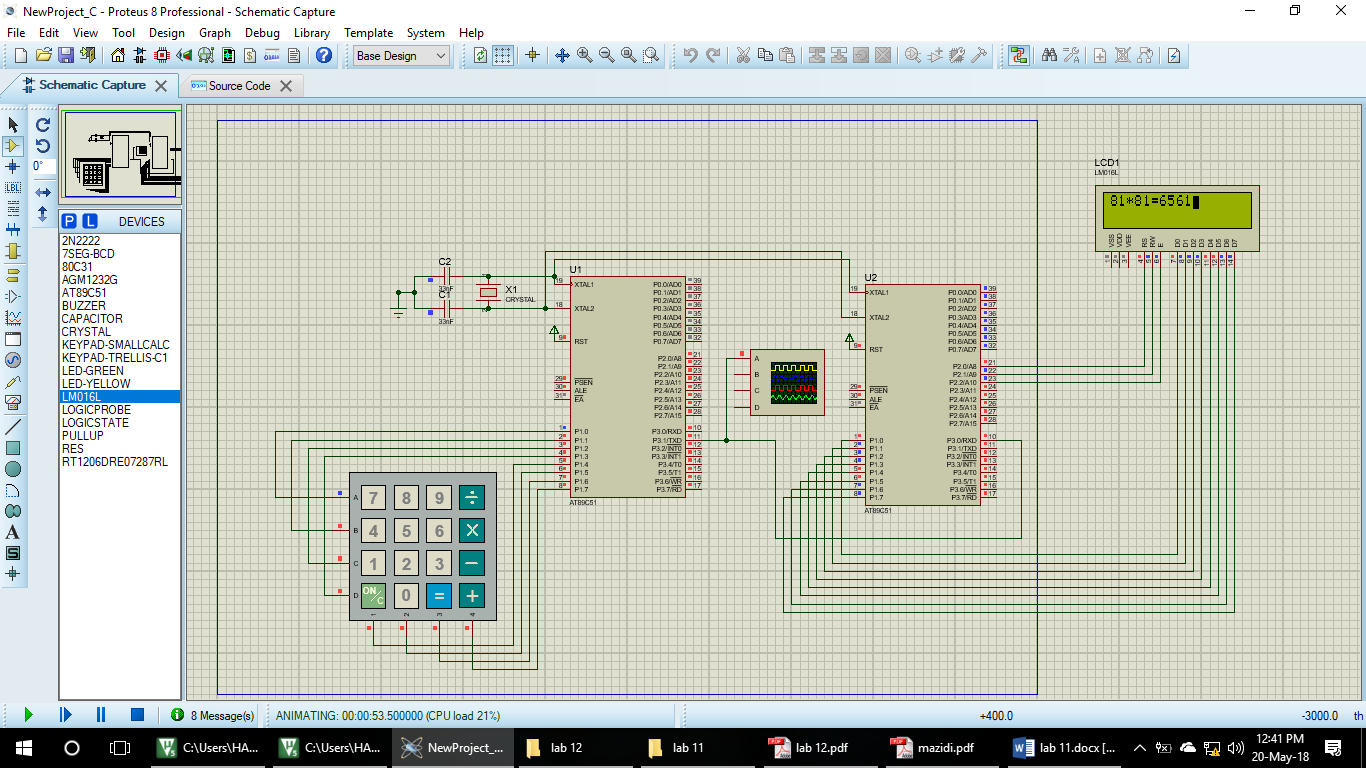
else

LCD\_senddata(pressed);

}

}

## **OUTPUT**



# **TASK 2:**

The flow charts of both, transmission and receiving algorithms which we have done in the Task number 1 of this Lab Task are attached with the lab report.

# **TASK 3:**

The **Serial Peripheral Interface bus** (**SPI**) is a [synchronous](https://en.wikipedia.org/wiki/Synchronous_circuit) [serial communication](https://en.wikipedia.org/wiki/Serial_communication) interface specification used for short distance communication, primarily in [embedded systems](https://en.wikipedia.org/wiki/Embedded_systems). SPI devices communicate in [full duplex](https://en.wikipedia.org/wiki/Full_duplex) mode using a [master-slave](https://en.wikipedia.org/wiki/Master-slave_(technology)) architecture with a single master. The master device originates the [frame](https://en.wikipedia.org/wiki/Frame_(networking)) for reading and writing. Multiple slave devices are supported through selection with individual [slave select](https://en.wikipedia.org/wiki/Slave_select) (SS) lines.

On the other hand, **I²C** (**Inter-Integrated Circuit**), is a [synchronous](https://en.wikipedia.org/wiki/Synchronous_circuit), [multi-master, multi-slave](https://en.wikipedia.org/wiki/Master/slave_(technology)), [packet switched](https://en.wikipedia.org/wiki/Packet_switching), [single-ended](https://en.wikipedia.org/wiki/Single-ended_signaling), [serial](https://en.wikipedia.org/wiki/Serial_communications) [computer bus](https://en.wikipedia.org/wiki/Computer_bus).

# **Conclusion:**

In this lab we learnt how to serially communicate among different microcontroller devices. In the task 1 of this lab, we used keypad for the input of a number. This number was sent to another microcontroller, which received it and sent it to LCD for the display. This procedure was explained in task 2 with the help of Flow chart.