Experiment No : 8

Aim : SERIAL COMMUNICATION PORT PROGRAMMING

Theory :

Since IBM PC/compatible computers are so widely used to communicate with 8051-based systems, serial communications of the 8051 with the COM port of the PC will be emphasized. To allow data transfer between the PC and an 8051 system without any error, we must make sure that the baud rate of the 8051 system matches the baud rate of the PC‟s COM port.

Baud rate in the 8051: The 8051 transfers and receives data serially at many different baud rates. Serial communications of the 8051 is established with PC through the COM port. It must make sure that the baud rate of the 8051 system matches the baud rate of the PC's COM port/ any system to be interfaced. The baud rate in the 8051 is programmable. This is done with the help of Timer. When used for serial port, the frequency of timer tick is determined by (XTAL/12)/32 and 1 bit is transmitted for each timer period (the time duration from timer start to timer expiry). The Relationship between the crystal frequency and the baud rate in the 8051 is that the 8051 divides the crystal frequency by 12 to get the machine cycle frequency which is shown in figure1. Here the oscillator is XTAL = 11.0592 MHz, the machine cycle frequency is 921.6 kHz. 8051's UART divides the machine cycle frequency of 921.6 kHz by 32 once more before it is used by Timer 1 to set the baud rate. 921.6 kHz divided by 32 gives 28,800 Hz. Timer 1 must be programmed in mode 2, that is 8-bit, auto-reload.

SBUF (serial buffer) register: It is an 8 bit register used solely for serial communication in the 8051.A byte of data to be transferred via the TxD line must be placed in the SBUF register. SBUF holds the byte of data when it is received by the RxD line. It can be accessed like any other register The fifth bit is TB8 which is used by modes 2 and 3 for the 8-bit transmission. When mode 1 is used the pin TB8 should be cleared. The sixth bit RB8 is used by modes 2 and 3 for the reception of bit 8. It is used by mode1 to store the stop bit. The seventh bit is TI which is the Transmit Interrupt. When 8051 finishes the transfer of the 8-bit character, it sets TI to "1" to indicate that it is ready to transfer the next character. The TI is raised at the beginning of the stop bit. The last bit is the RI which is the receive interrupt. When 8051 receives a character,the UART removes start bit and stop bit. The UART puts the 8-bit character in SBUF. RI is set to „1‟ to indicate that a new byte is ready to be picked up in SBUF.RI is raised halfway through the stop bit 1.3 Steps to send data serially: 1.Set baud rate by loading TMOD register with the value 20H, this indicating timer 1 in mode 2 (8-bit auto-reload) to set baud rate 2. The TH1 is loaded with proper values to set baud rate for serial data transfer 3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8- bit data is framed with start and stop bits 4. TR1 is set to 1 to start timer 1 5. TI is cleared by CLR TI instruction 6. The character byte to be transferred serially is written into SBUF register 7.

The TI flag bit is monitored with the use of instruction JNB TI,xx to see if the character has been transferred completely 8. To transfer the next byte, go to step 5
Check the TI flag bit, we know whether or not 8051 is ready to transfer another byte. TI flag bit is raised by the 8051 after transfer of data. TI flag is cleared by the programmer by instruction like “CLR TI”. When writing a byte into SBUF, before the TI flag bit is raised, it may lead to loss of a portion of the byte being transferred. 1.4 Steps to receive data serially: 1. Set baud rate by loading TMOD register with the value 20H, this indicating timer 1 in mode 2 (8-bit auto-reload) to set baud rate 2. The TH1 is loaded with proper values to set baud rate 3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8- bit data is framed with start and stop bits 4.

TR1 is set to 1 to start timer 1 5. RI is cleared by CLR RI instruction 6. The RI flag bit is monitored with the use of instruction JNB RI,xx to see if an entire character has been received yet 7. When RI is raised, SBUF has the byte; its contents are moved into a safe place 8. To receive next character, go to step 5

Complete the following 8051 assembly programs in EDSIM using serial communication mode1 with crystal frequency of 11.0592MHz

1) Write a program for the 8051 to transfer letter "A" serially at 2400 baud only once

2) Write a program for the 8051 to transfer the word "YES" serially at 2400 baud continuously

3) Write a program for the 8051 to transfer letter "A" serially at 4800 baud only one

4) Write a program for the 8051 to transfer letter "A" serially at 4800 baud only once by doubling the 2400 baud

5) Write a program for the 8051 to transfer letters 'a','b','c' stored at memory location 30H, 31H, 32H respectively at 4800 baud. At 33H store '0', when 'o' is encountered stop

6) Write a program for the 8051 to receive letter 'K' serially at 2400 baud.

7) Write a program for the 8051 to receive letter 'K' serially at 4800 baud.

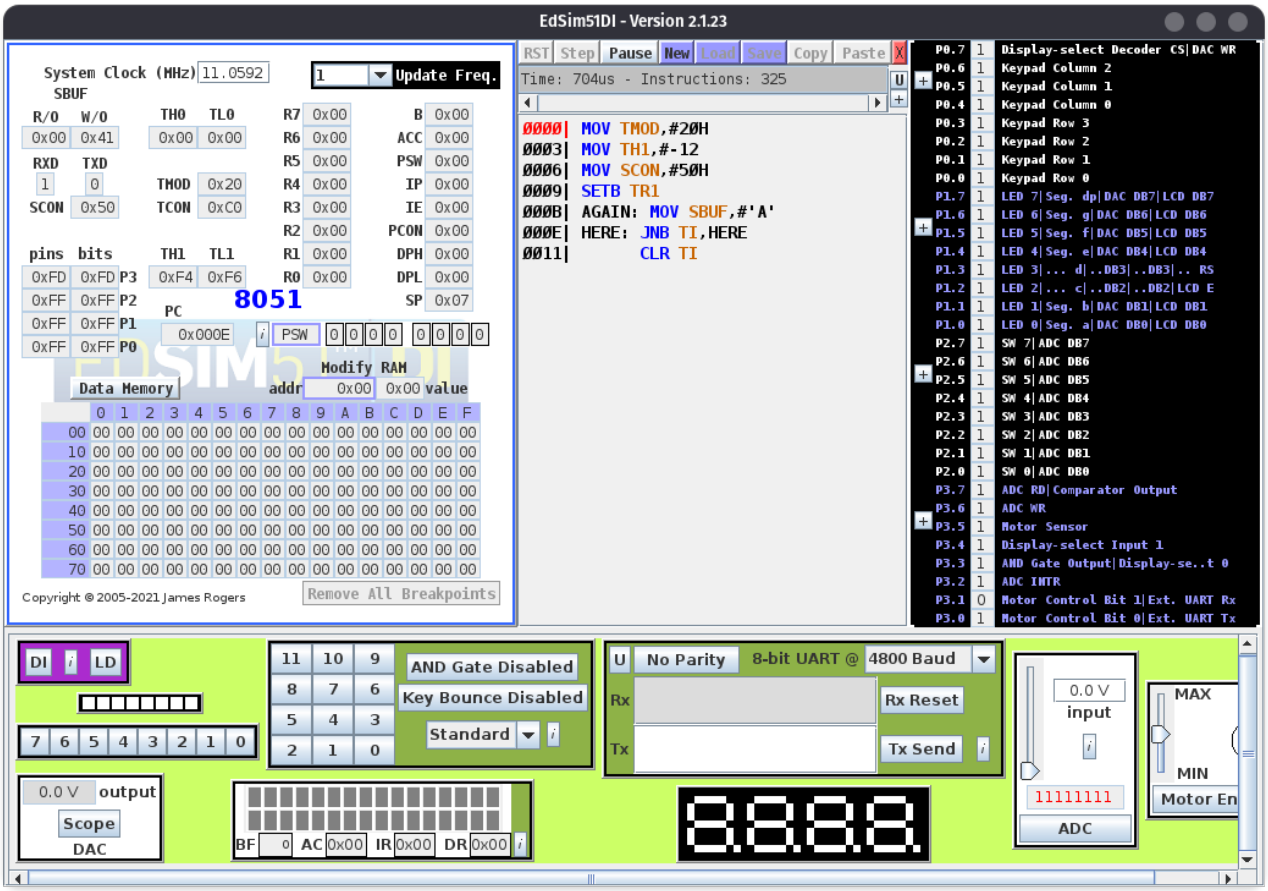
8) Write a program for the 8051 to receive letter 'K' serially at 4800 baud by doubling 2400 baud. Transmission.

9) Write a program for the 8051 to receive 'GOA' serially at 2400 baud and store it in memory location starting at 30H

10) Write a program for the 8051 to receive 'GOA' serially at 4800 baud and store it in memory location starting at 30H. If the character 'D' appears the reception should stop

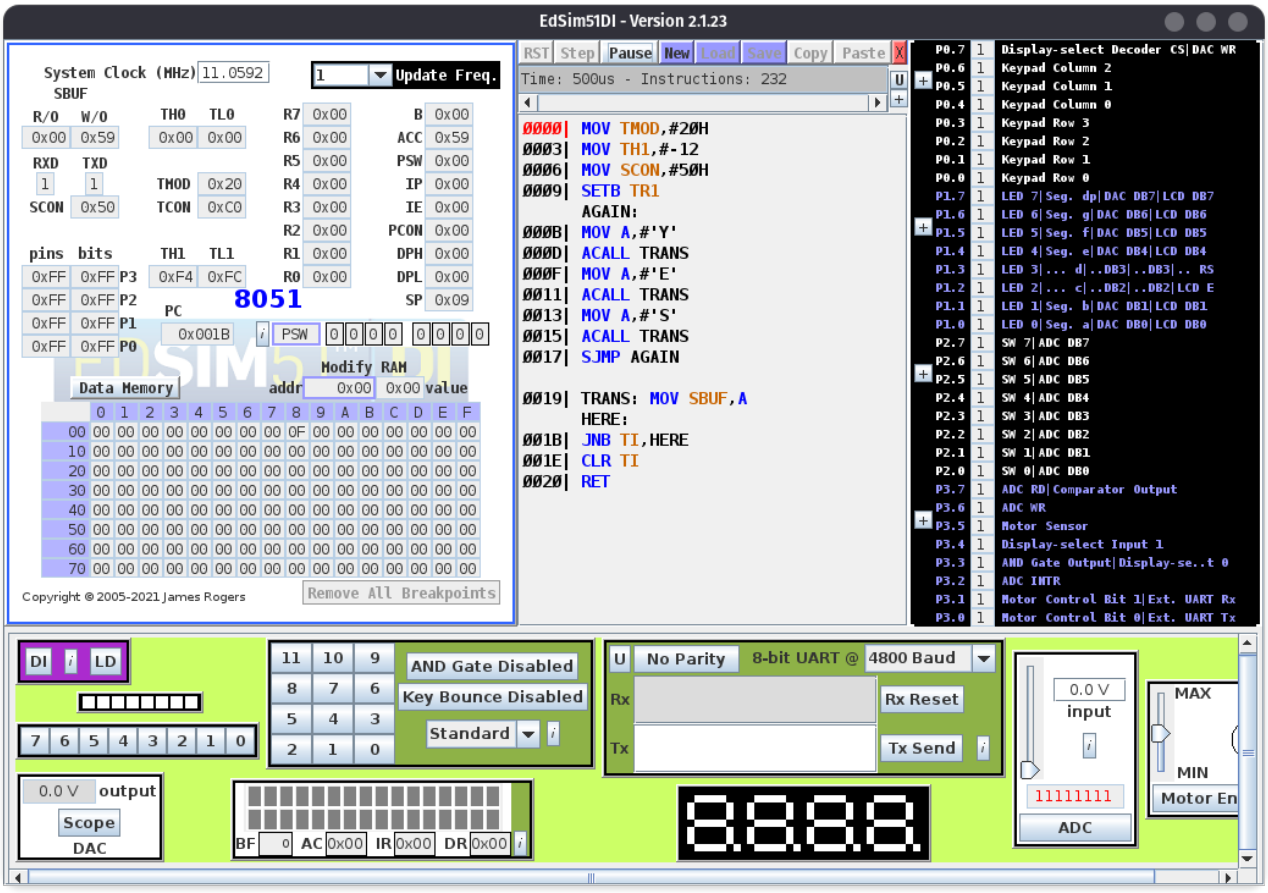
1.

```
MOV TMOD,#20H
MOV TH1,#-12
MOV SCON,#50H
SETB TR1
AGAIN: MOV SBUF,#'A'
HERE:JNB TI,HERE
CLR TI
```



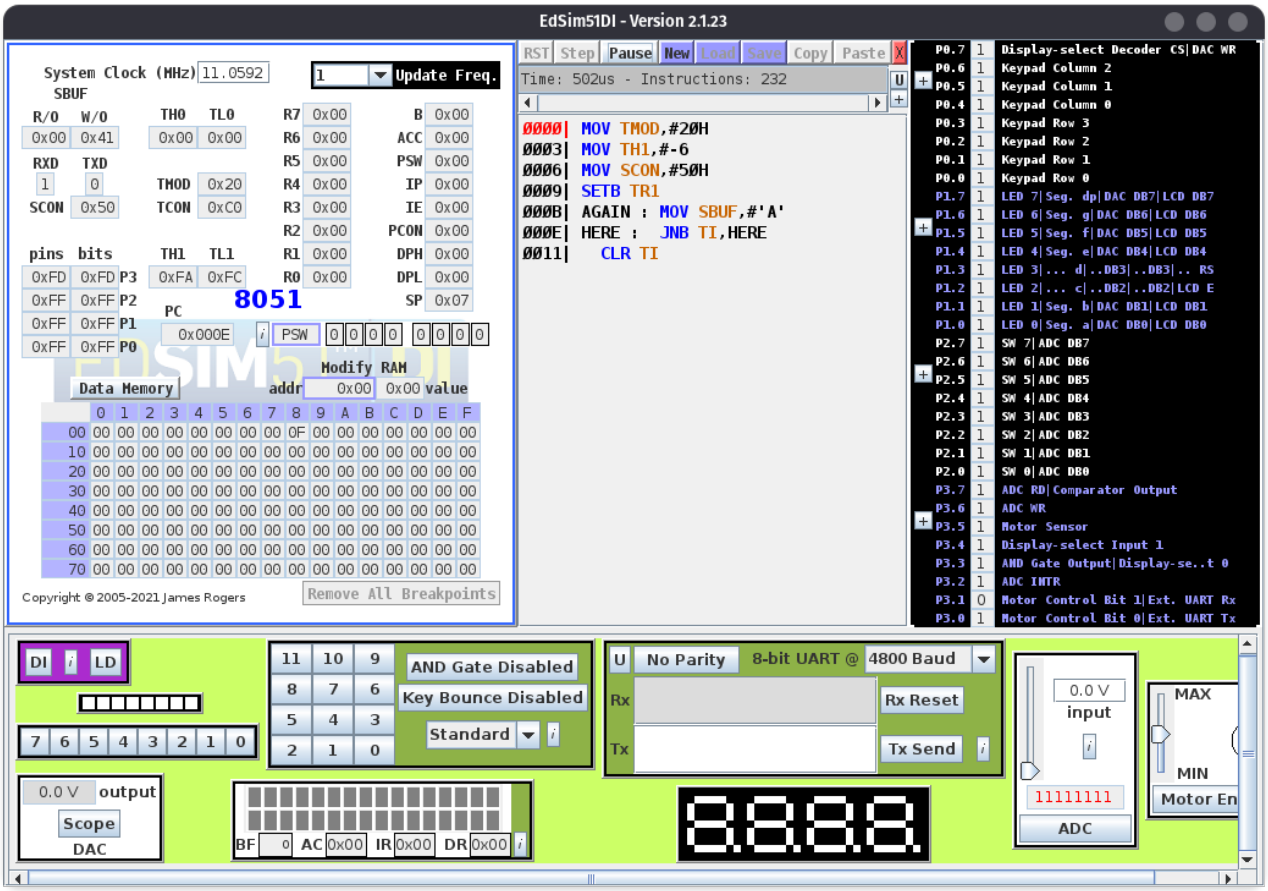
2)

```
MOV TMOD,#20H
MOV TH1,#-12
MOV SCON,#50H
SETB TR1
AGAIN:
MOV A,#'Y'
ACALL TRANS
MOV A,#'E'
ACALL TRANS
MOV A,#'S'
ACALL TRANS
SJMP AGAIN

TRANS: MOV SBUF,A
HERE:
JNB TI,HERE
CLR TI
RET
```



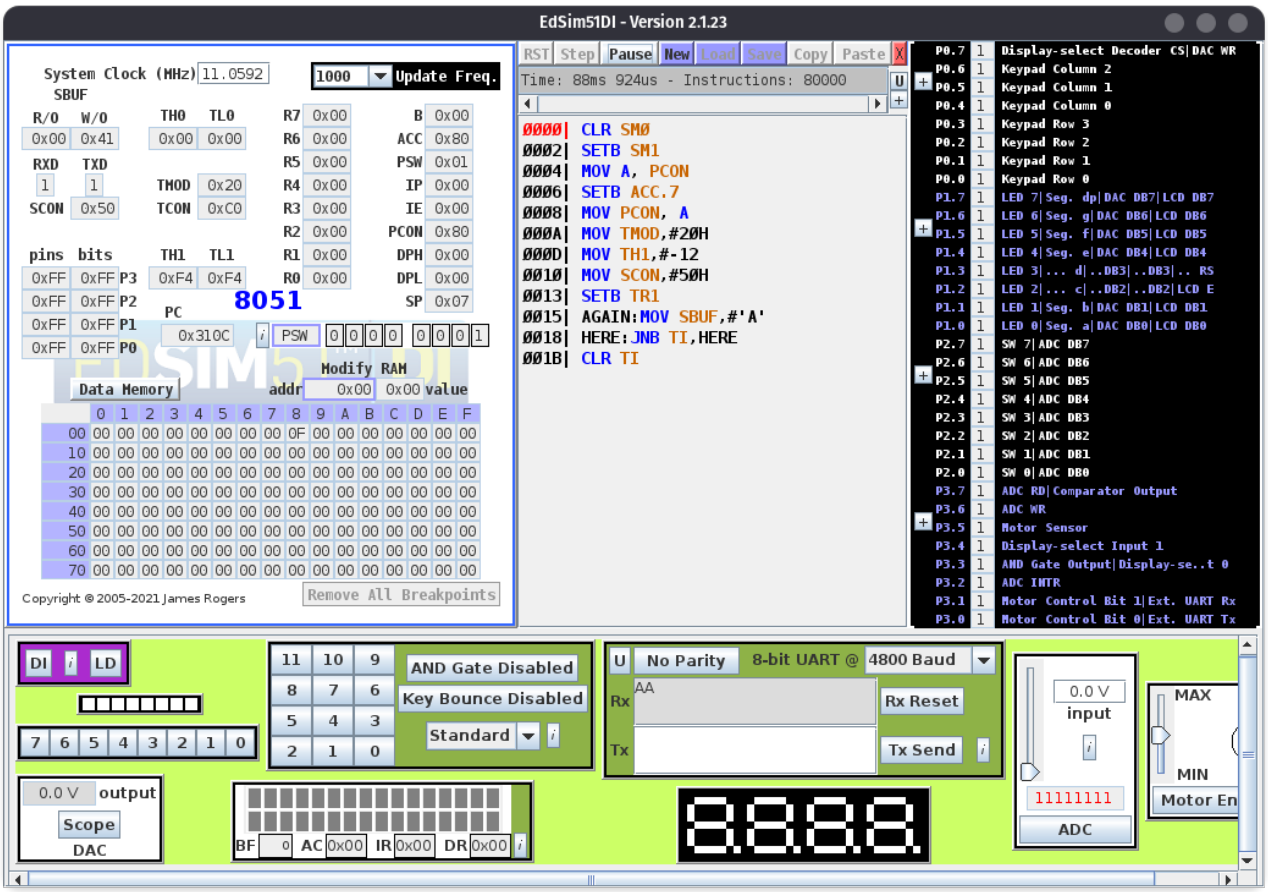
3)

```
MOV TMOD,#20H
MOV TH1,#-6
MOV SCON,#50H
SETB TR1
AGAIN : MOV SBUF,#'A'
HERE :  JNB TI,HERE
        CLR TI
```



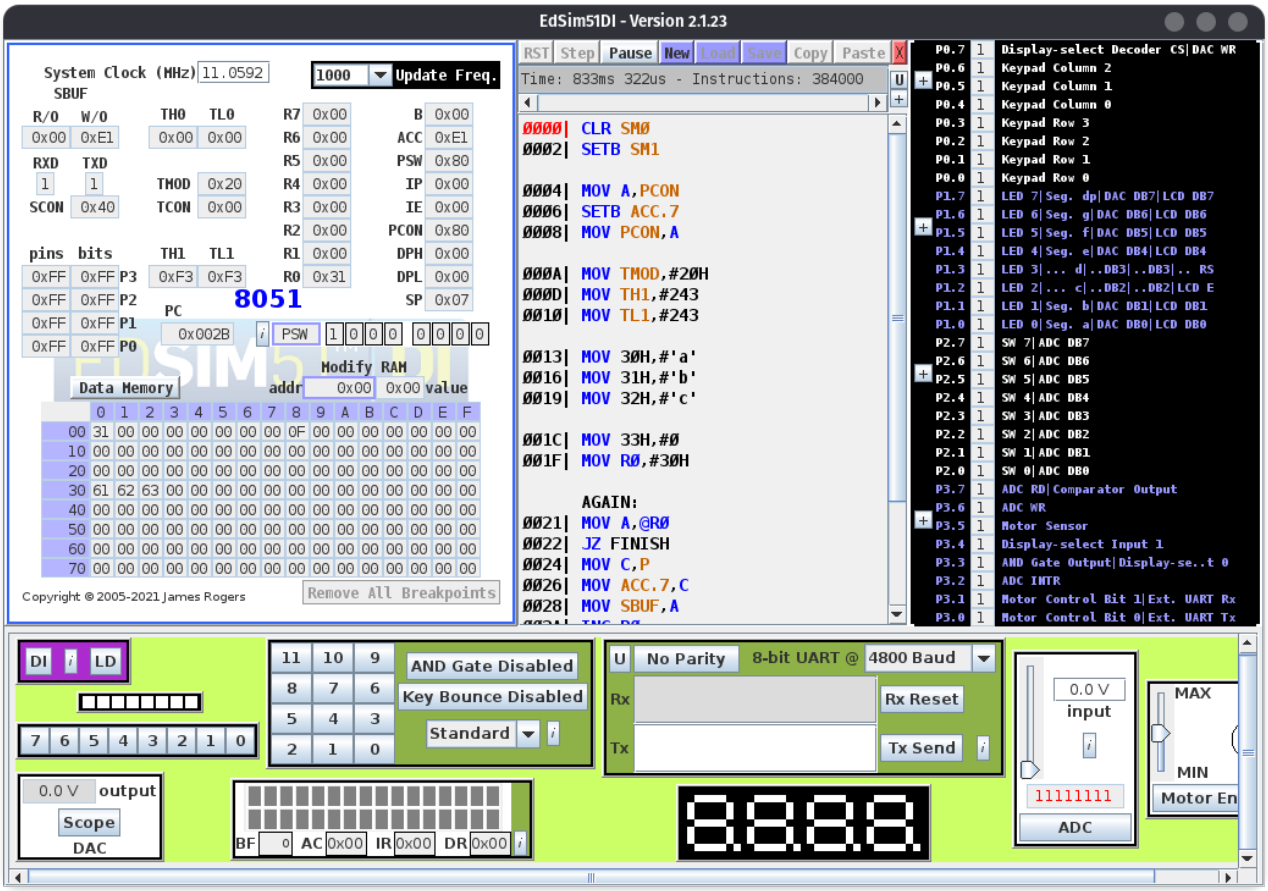
4.

```
CLR SM0

SETB SM1

MOV A, PCON

SETB ACC.7

MOV PCON, A

MOV TMOD,#20H

MOV TH1,#-12

MOV SCON,#50H

SETB TR1

AGAIN:MOV SBUF,#'A'

HERE:JNB TI,HERE

CLR TI
```



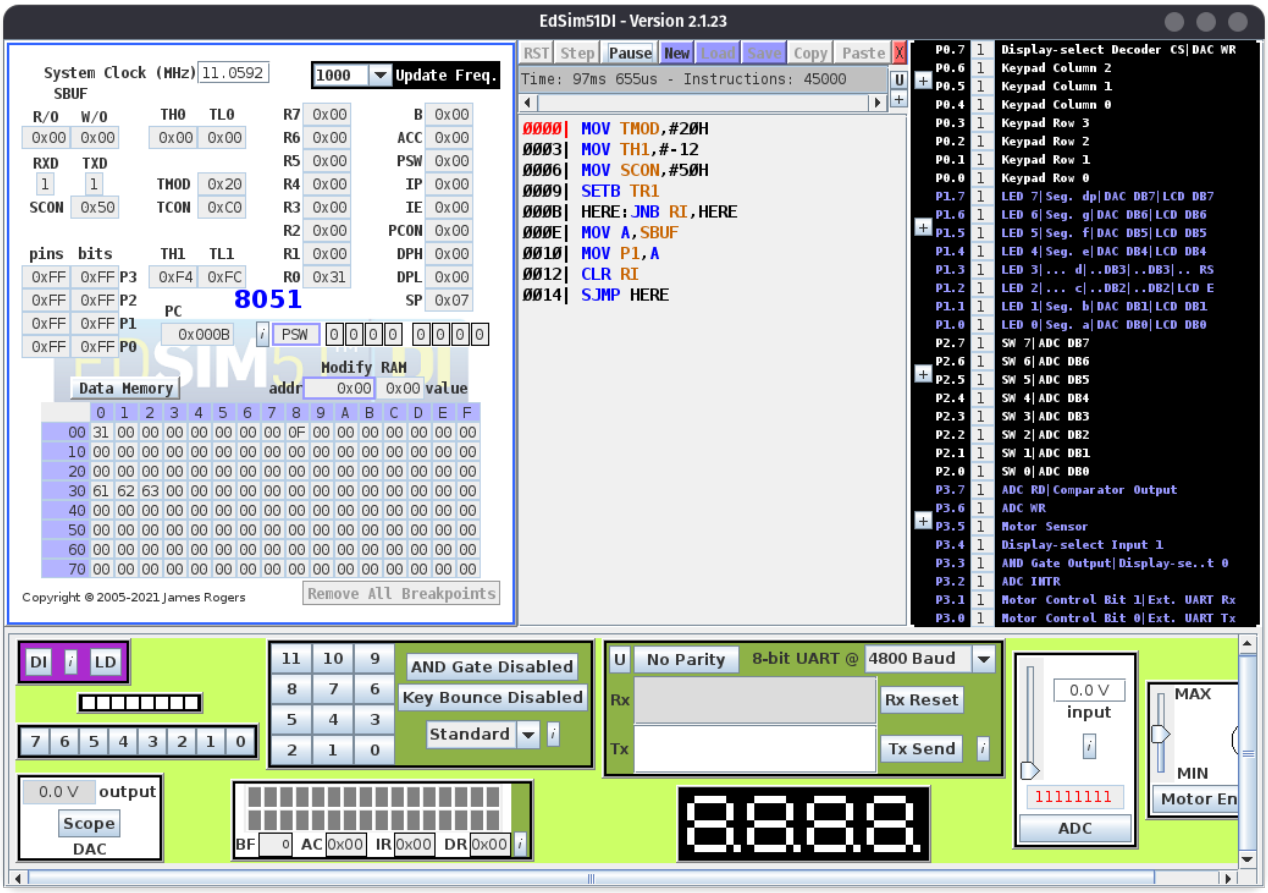
5.

```
CLR SM0
SETB SM1
MOV A,PCON
SETB ACC.7
MOV PCON,A
MOV TMOD,#20H
MOV TH1,#243
MOV TL1,#243
MOV 30H,#'a'
MOV 31H,#'b'
MOV 32H,#'c'
MOV 33H,#0
MOV R0,#30H
AGAIN: MOV A,@R0
JZ FINISH
MOV C,P
MOV ACC.7,C
MOV SBUF,A
INC R0
JNB TI,$
CLR TI
JMP AGAIN
FINISH:JMP $
```


EdSim51DI - Version 2.1.23
System Clock (MHz) 11.0592
1000
Update Freq.
SBUF
R/O W/O
0x00 0xE1
RXD TXD
1 1
SCON 0x40
TH0 TL0
0x00 0x00
TMOD 0x20
TCON 0x00
TH1 TL1
0xF3 0xF3
R7 0x00
R6 0x00
R5 0x00
R4 0x00
R3 0x00
R2 0x00
R1 0x00
R0 0x31
B 0x00
ACC 0xE1
PSW 0x80
IP 0x00
IE 0x00
PCON 0x80
DPH 0x00
DPL 0x00
SP 0x07
pins bits
0xFF 0xFF P3
0xFF 0xFF P2
0xFF 0xFF P1
0xFF 0xFF P0
PC
0x002B
8051
PSW 1 0 0 0 0 0 0 0
Modify RAM
Data Memory
addr 0x00 0x00 value
Copyright © 2005-2021 James Rogers
Remove All Breakpoints
RST Step Pause New Load Save Copy Paste
Time: 833ms 322us - Instructions: 384000
0000| CLR SM0
0002| SETB SM1
0004| MOV A,PCON
0006| SETB ACC.7
0008| MOV PCON,A
000A| MOV TMOD,#20H
000D| MOV TH1,#243
0010| MOV TL1,#243
0013| MOV 30H,#'a'
0016| MOV 31H,#'b'
0019| MOV 32H,#'c'
001C| MOV 33H,#0
001F| MOV R0,#30H
AGAIN:
0021| MOV A,@R0
0022| JZ FINISH
0024| MOV C,P
0026| MOV ACC.7,C
0028| MOV SBUF,A
P0.7 Display-select Decoder CS|DAC WR
P0.6 Keypad Column 2
P0.5 Keypad Column 1
P0.4 Keypad Column 0
P0.3 Keypad Row 3
P0.2 Keypad Row 2
P0.1 Keypad Row 1
P0.0 Keypad Row 0
P1.7 LED 7|Seg. dp|DAC DB7|LCD DB7
P1.6 LED 6|Seg. g|DAC DB6|LCD DB6
P1.5 LED 5|Seg. f|DAC DB5|LCD DB5
P1.4 LED 4|Seg. e|DAC DB4|LCD DB4
P1.3 LED 3|... d|..DB3|..DB3|.. RS
P1.2 LED 2|... c|..DB2|..DB2|LCD E
P1.1 LED 1|Seg. b|DAC DB1|LCD DB1
P1.0 LED 0|Seg. a|DAC DB0|LCD DB0
P2.7 SW 7|ADC DB7
P2.6 SW 6|ADC DB6
P2.5 SW 5|ADC DB5
P2.4 SW 4|ADC DB4
P2.3 SW 3|ADC DB3
P2.2 SW 2|ADC DB2
P2.1 SW 1|ADC DB1
P2.0 SW 0|ADC DB0
P3.7 ADC RD|Comparator Output
P3.6 ADC WR
P3.5 Motor Sensor
P3.4 Display-select Input 1
P3.3 AND Gate Output|Display-se..t 0
P3.2 ADC INTR
P3.1 Motor Control Bit 1|Ext. UART Rx
P3.0 Motor Control Bit 0|Ext. UART Tx
DI LD
7 6 5 4 3 2 1 0
11 10 9
8 7 6
5 4 3
2 1 0
AND Gate Disabled
Key Bounce Disabled
Standard
U No Parity 8-bit UART @ 4800 Baud
Rx
Tx
Rx Reset
Tx Send
0.0 V output
Scope
DAC
BF 0 AC 0x00 IR 0x00 DR 0x00
0.0 V input
11111111
ADC
MAX
MIN
Motor En

6.

```
MOV TMOD,#20H
MOV TH1,#-12
MOV SCON,#50H
SETB TR1
HERE:JNB RI,HERE
MOV A,SBUF
MOV P1,A
CLR RI
SJMP HERE
```



7.

MOV TMOD,#20H

MOV TH1,#-6

MOV SCON,#50H

SETB TR1

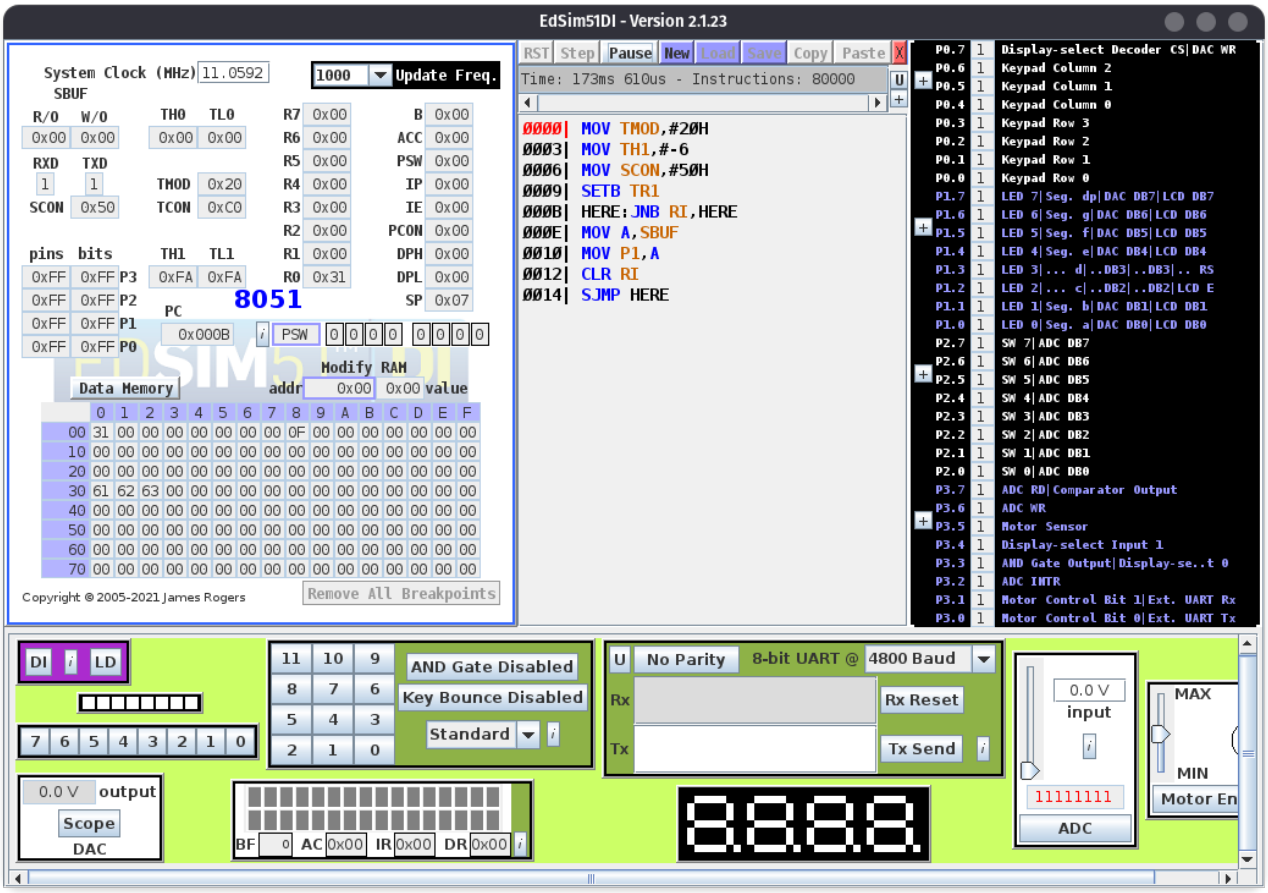
HERE:JNB RI,HERE

MOV A,SBUF

MOV P1,A

CLR RI

SJMP HERE



8.

MOV A, PCON

SETB ACC.7

MOV PCON, A

MOV TMOD,#20H

MOV TH1,#-12

MOV SCON,#50H

SETB TR1

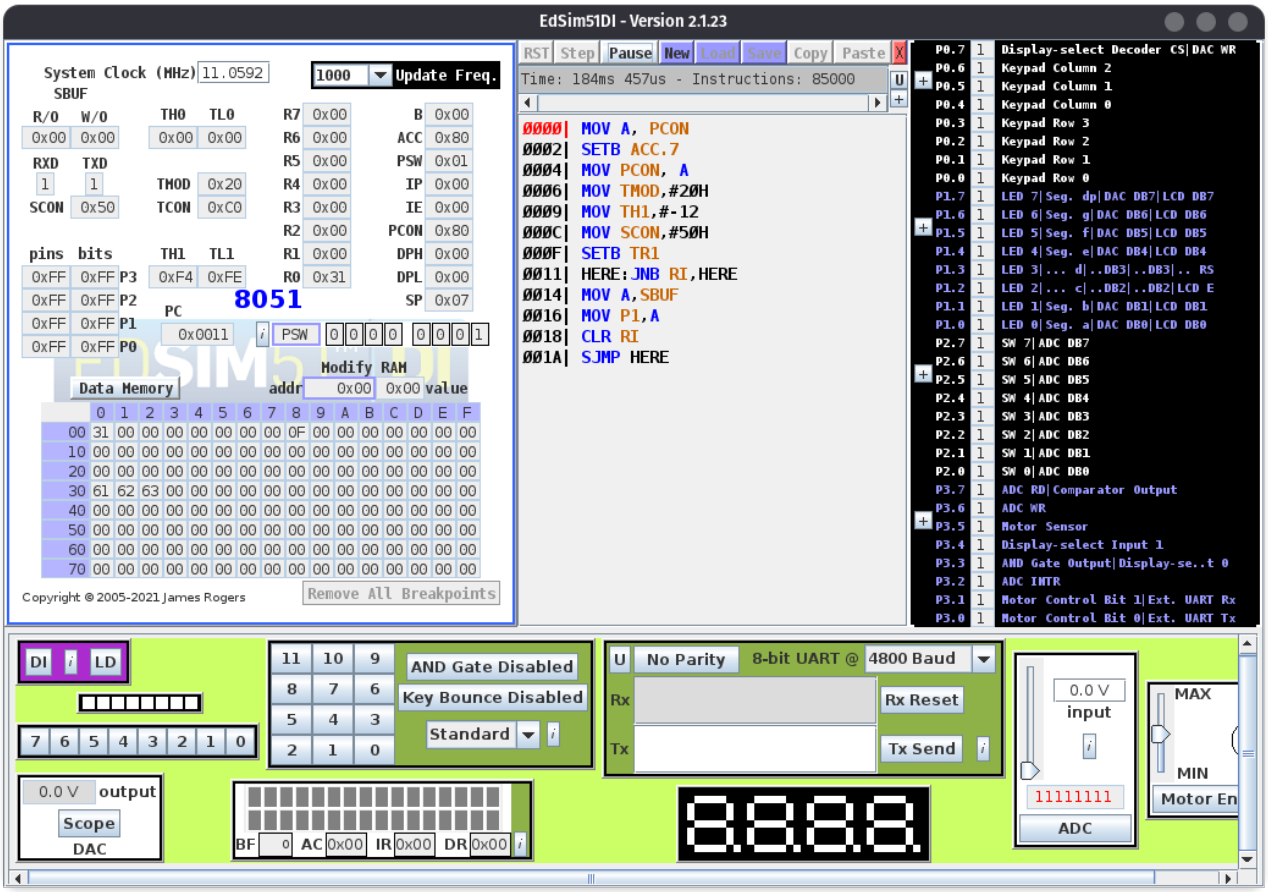
HERE:JNB RI,HERE

MOV A,SBUF

MOV P1,A

CLR RI

SJMP HERE



9.

MOV A, PCON

SETB ACC.7

MOV PCON, A

MOV TMOD,#20H

MOV TH1,#-12

MOV SCON,#50H

SETB TR1

HERE:JNB RI,HERE

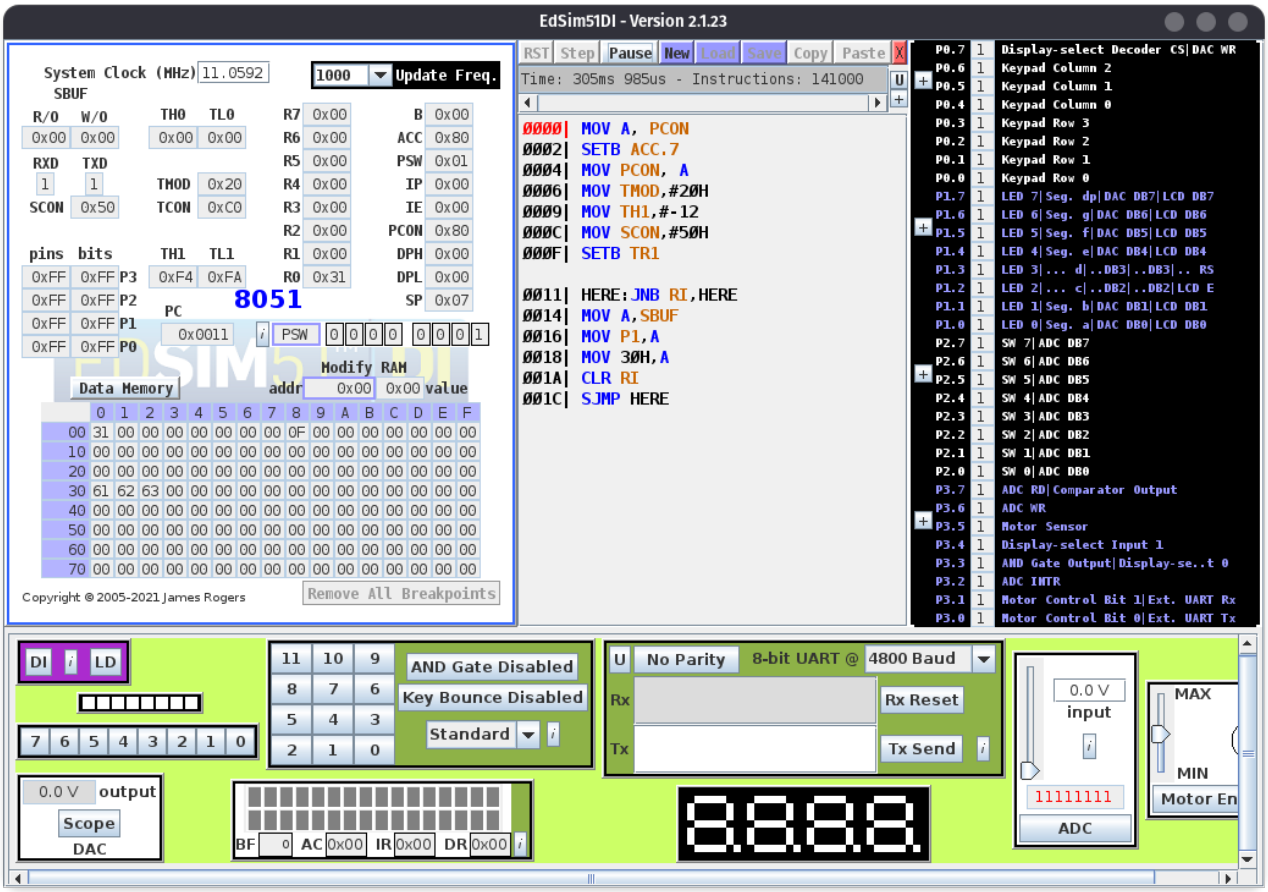
MOV A,SBUF

MOV P1,A

MOV 30H,A

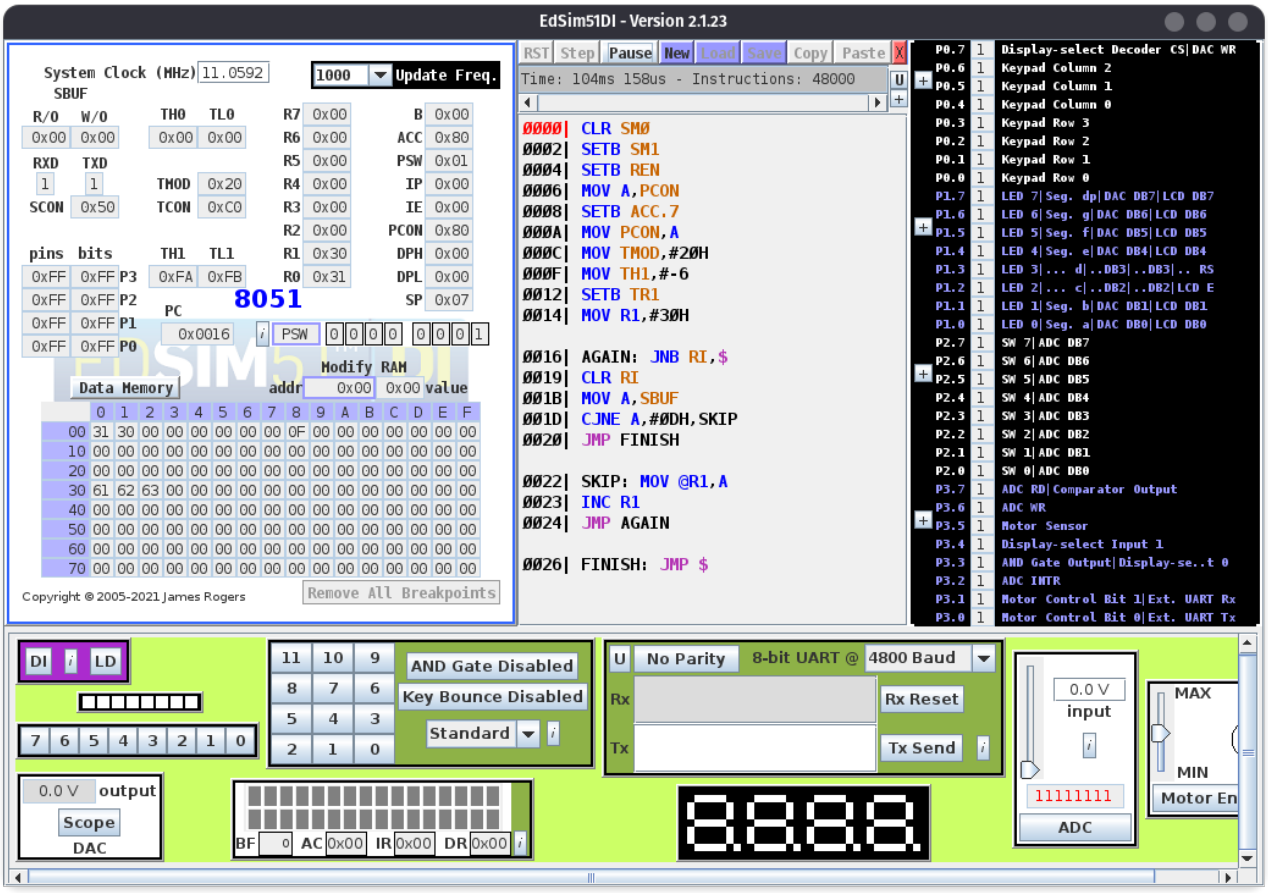
CLR RI

SJMP HERE



10.

```
CLR SM0
SETB SM1
SETB REN
MOV A,PCON
SETB ACC.7
MOV PCON,A
MOV TMOD,#20H
MOV TH1,#-6
SETB TR1
MOV R1,#30H
AGAIN: JNB RI,$
CLR RI
MOV A,SBUF
CJNE A,#0DH,SKIP
JMP FINISH
SKIP: MOV @R1,A
INC R1
JMP AGAIN
FINISH: JMP $
```



Conclusion : Serial communication port programming was studied and the codes were implemented sucessfully.

Deepraj Bhosale

181105016