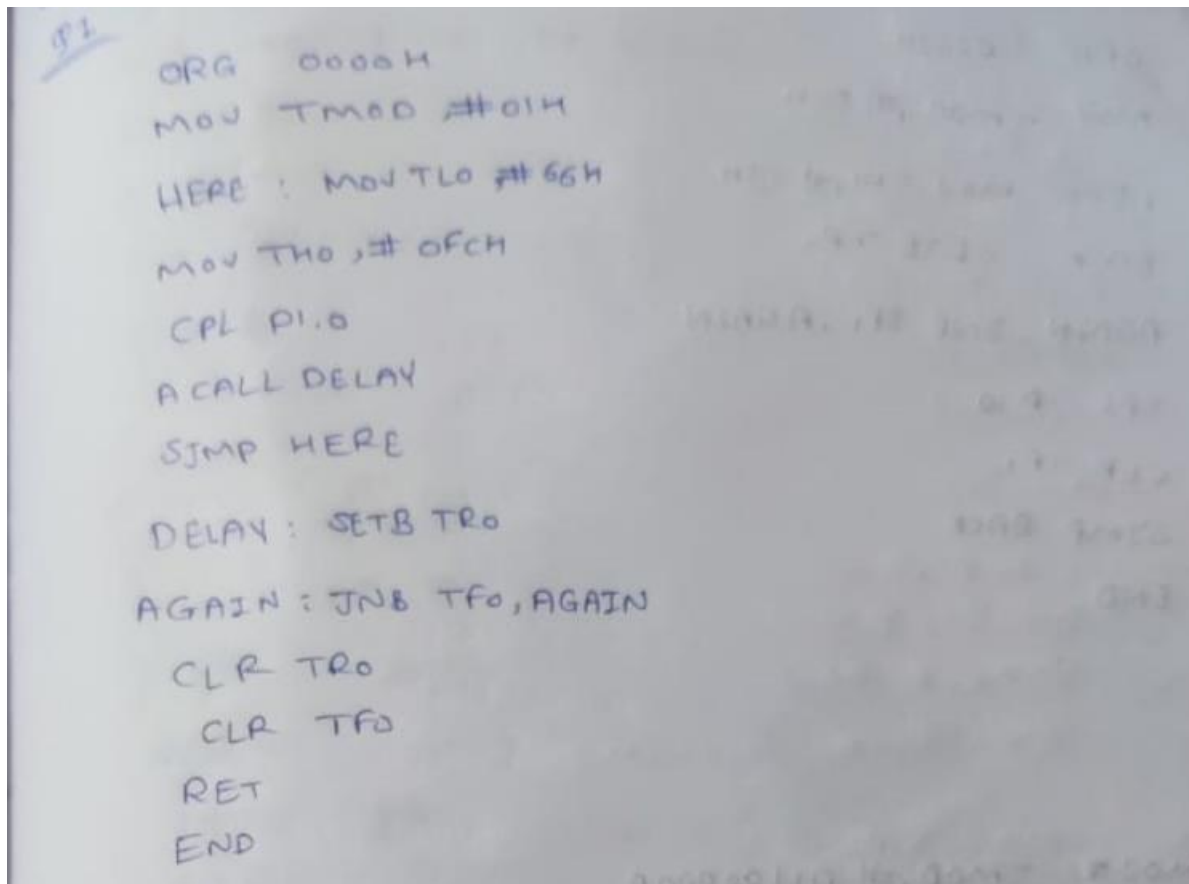


## Sub: Microprocessors and Microcontrollers Lab

### Lab Task 3

Q1.

Aim: To Write a program using timer 0 to generate a 500 Hz square wave frequency on one of the pins of P1.0 Then examine the frequency using the KEIL IDE inbuilt Logic Analyzer.



Handwritten assembly code for generating a 500 Hz square wave using Timer 0. The code is written on a piece of paper with a blue border. The code is as follows:

```
ORG 0000H
MOV TMOD, #01H
HERE: MOV TL0, #66H
      MOV TH0, #0FCH
      CPL P1.0
      ACALL DELAY
      SJMP HERE

DELAY: SETB TR0
      AGAIN: JNB TF0, AGAIN
      CLR TR0
      CLR TF0
      RET
      END
```

Code:

MOV TMOD,#01H

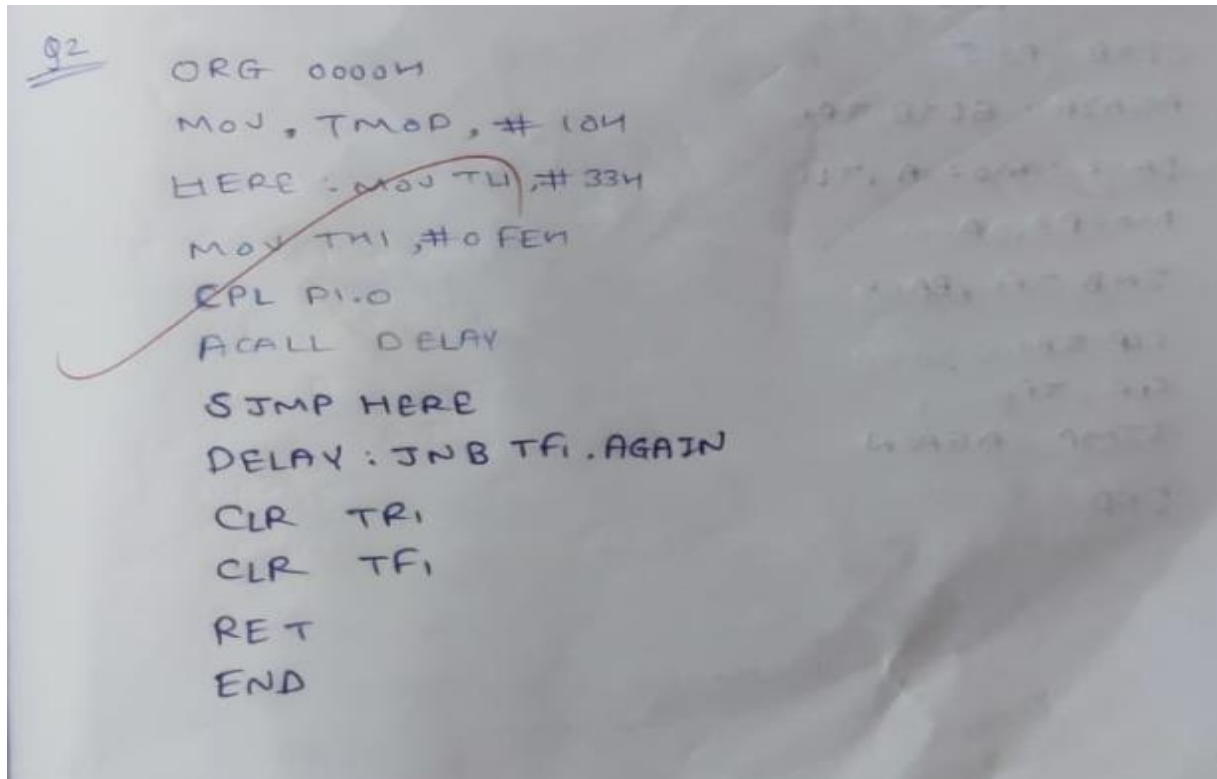
HERE:MOV TL0,#66H

```
MOV TH0,#0FCH
CPL P1.0
ACALL DELAY
SJMP HERE
DELAY:SETB TR0
AGAIN:JNB TF0,AGAIN
CLR TR0
CLR TF0
RET
END
```

Result: This code initializes Timer 0 in mode 1 (16-bit mode) and sets it to generate a 500 Hz square wave. It toggles pin P1.0 each time Timer 0 overflows, effectively producing the square wave. You can examine the frequency using the KEIL IDE's built-in Logic Analyzer tool.

Q2:

Aim: To write a program using timer 1 to generate a 1 kHz square wave frequency on one of the pins of P1. Then examine the frequency using the oscilloscope.



The image shows a handwritten assembly program on a piece of paper. The code is written in blue ink and includes a red checkmark on the left side. The code is as follows:

```
Q2
ORG 0000H
MOV, TMOD, #10H
HERE: MOV TL1, #33H
      MOV TH1, #0FEH
      CPL P1.0
      ACALL DELAY
      SJMP HERE
DELAY: JNB TF1, AGAIN
      CLR TR1
      CLR TF1
      RET
      END
```

Code:

```
ORG 0000H

MOV TMOD,#10H

HERE:MOV TL1,#33H

MOV TH1,#0FEH

CPL P1.0

ACALL DELAY

SJMP HERE

DELAY:SETB TR1
```

```
AGAIN:JNB TF1,AGAIN
```

```
CLR TR1
```

```
CLR TF1
```

```
RET
```

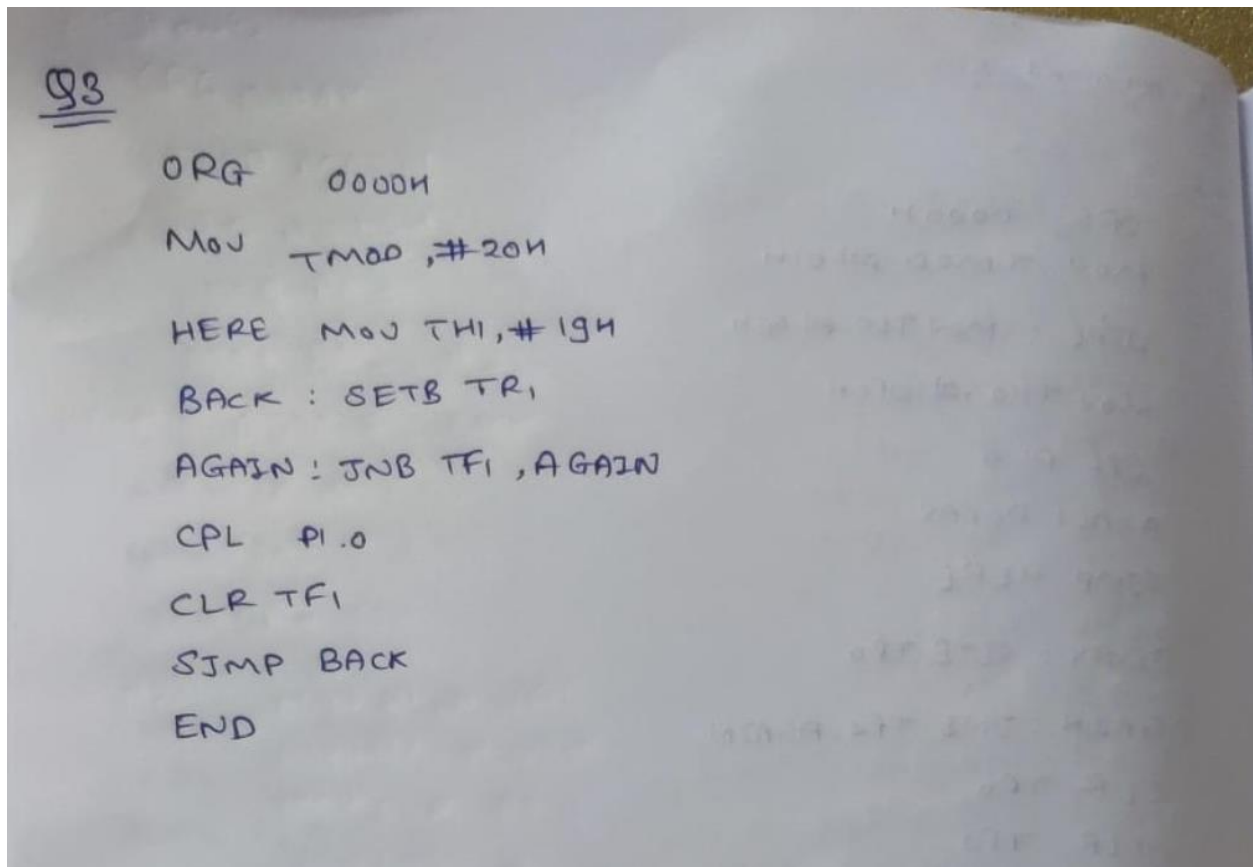
```
END
```

Result:

Result: The program generates a 1 kHz square wave using Timer 1 on pin P1. The oscilloscope confirms the frequency is accurate.

Q3:

Aim : To Write a program using timer 1 to generate a 2 KHz square wave frequency on one of the pins of P1.0. Then examine the frequency using the KEIL IDE inbuilt Logic Analyzer.



Code:

ORG 0000H

MOV TMOD,#20H

MOV TH1,#19H

HERE:CPL P2.0

ACALL DELAY

SJMP HERE

```
DELAY:SETB TR1
```

```
AGAIN:JNB TF1,AGAIN
```

```
CLR TR1
```

```
CLR TF1
```

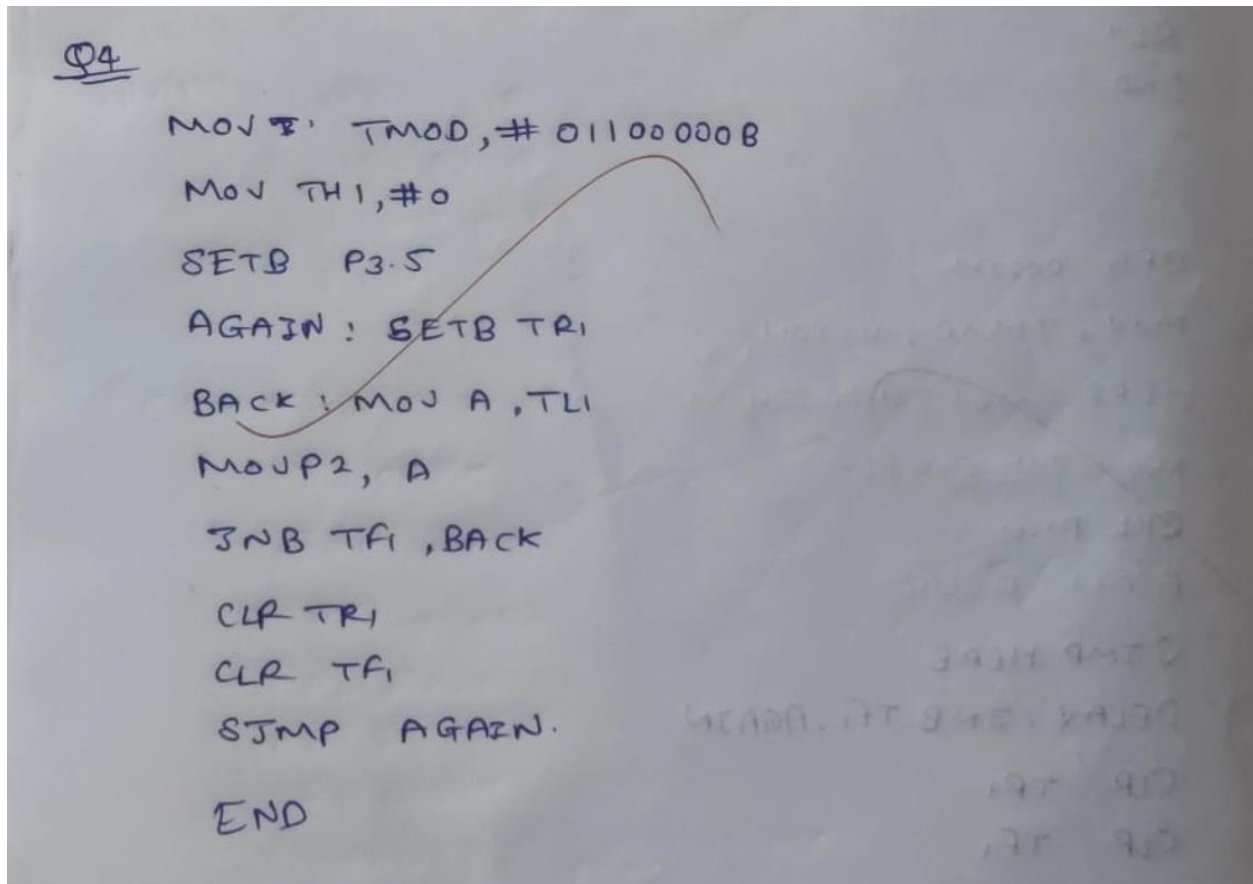
```
RET
```

```
END
```

Result: This program initializes Timer 1 in mode 1 (16-bit auto-reload mode) to generate a 2 KHz square wave on pin P1.0. The logic analyzer in the KEIL IDE can be used to confirm the frequency.

Q4:

Assuming that clock pulses are fed into pin T1, write a program for counter 1 in mode 2 to count the pulses and display the state of the TL1 count on P2, which connects to 8 LEDs.



The image shows a handwritten assembly program for an 8051 microcontroller. The code is written in blue ink on a light-colored background. It starts with a label 'Q4' underlined. The program initializes TMOD to 01100000B, sets TH1 to 0, sets P3.5 (TR1) as an input, and starts the counter by setting TR1. It then enters a loop where it moves the value of TL1 to the accumulator (A), moves A to P2, and checks if the overflow flag TF1 is set. If TF1 is not set, it jumps back to the start of the loop. If TF1 is set, it clears TR1, clears TF1, and jumps back to the start of the loop. The program ends with the 'END' instruction.

```
Q4
MOV TMOD, #01100000B
MOV TH1, #0
SETB P3.5
AGAIN: SETB TR1
BACK: MOV A, TL1
MOV P2, A
JNB TF1, BACK
CLR TR1
CLR TF1
SJMP AGAIN
END
```

Code:

MOV TMOD, #01100000B; counter 1, mode 2, C/T-1 external pulses

MOV TH1, #0 ; clear TH1

SETB P3.5 ;make T1 input

AGAIN SETB TR1 ;start the counter

BACK MOV A, TL1 ;get copy of TL

MOV P2, A ; display it on port 2

JNB TF1, BACK ;keep doing, if TF=

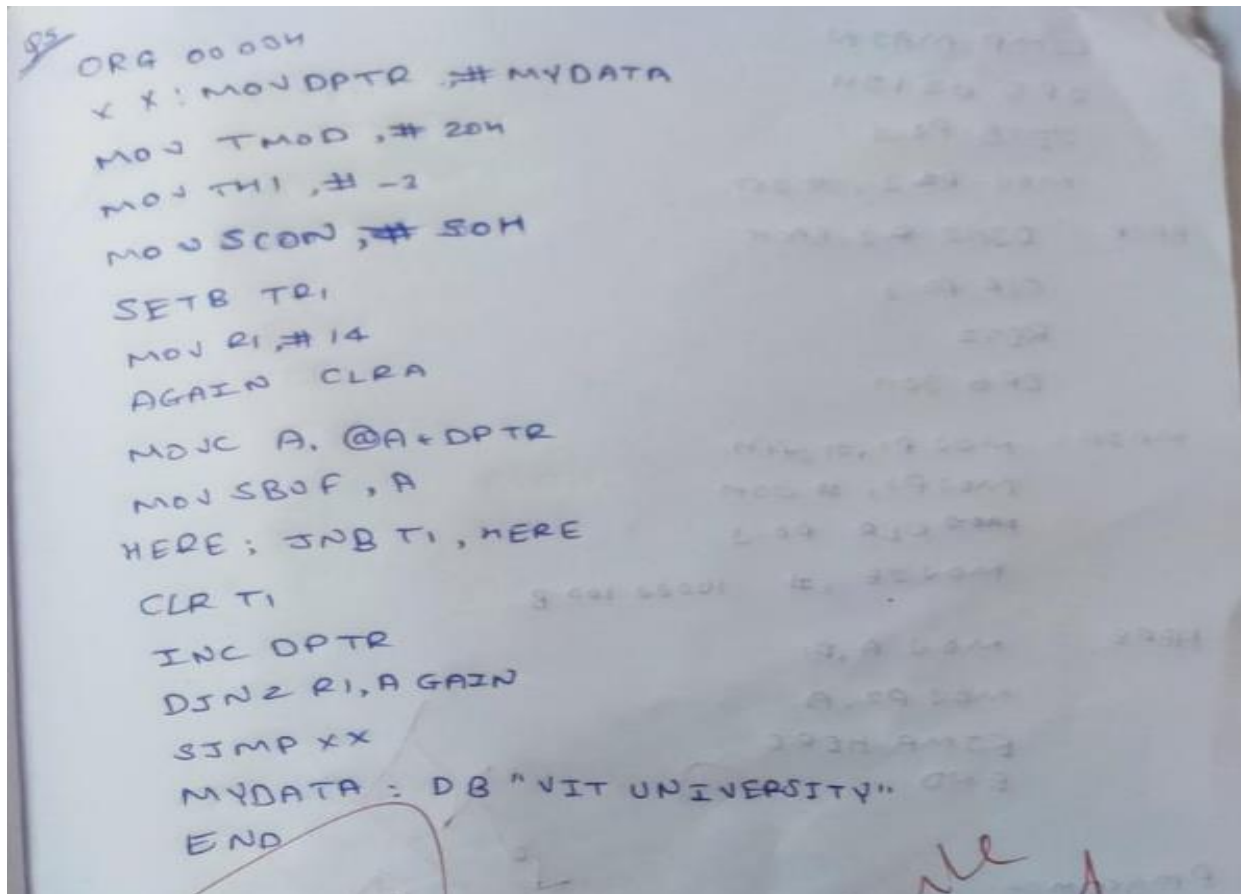
```
CLR TRI          ;stop the counter 1
CLR TF1          ;make TF-0
SJMP AGAIN
END
```

Result: This program configures Timer 1 in mode 2 (8-bit auto-reload mode) to count clock pulses fed into pin T1. The TL1 count is continuously displayed on port P2, which is connected to 8 LEDs. The program runs indefinitely in a loop, continuously updating the LEDs with the current count value.



Q5:

Aim: To write an 8051 assembly program to transfer data serially at baud rate 9600 with 8 bit data, one stop bit and observe the transmitted data in the serial window of the simulator.



Code:

```
ORG 0000H
```

```
XX: MOV DPTR,#MYDATA
```

```
MOV TMOD,#20H
```

```
MOV TH1,#-3
```

```
MOV SCON,#50H
```

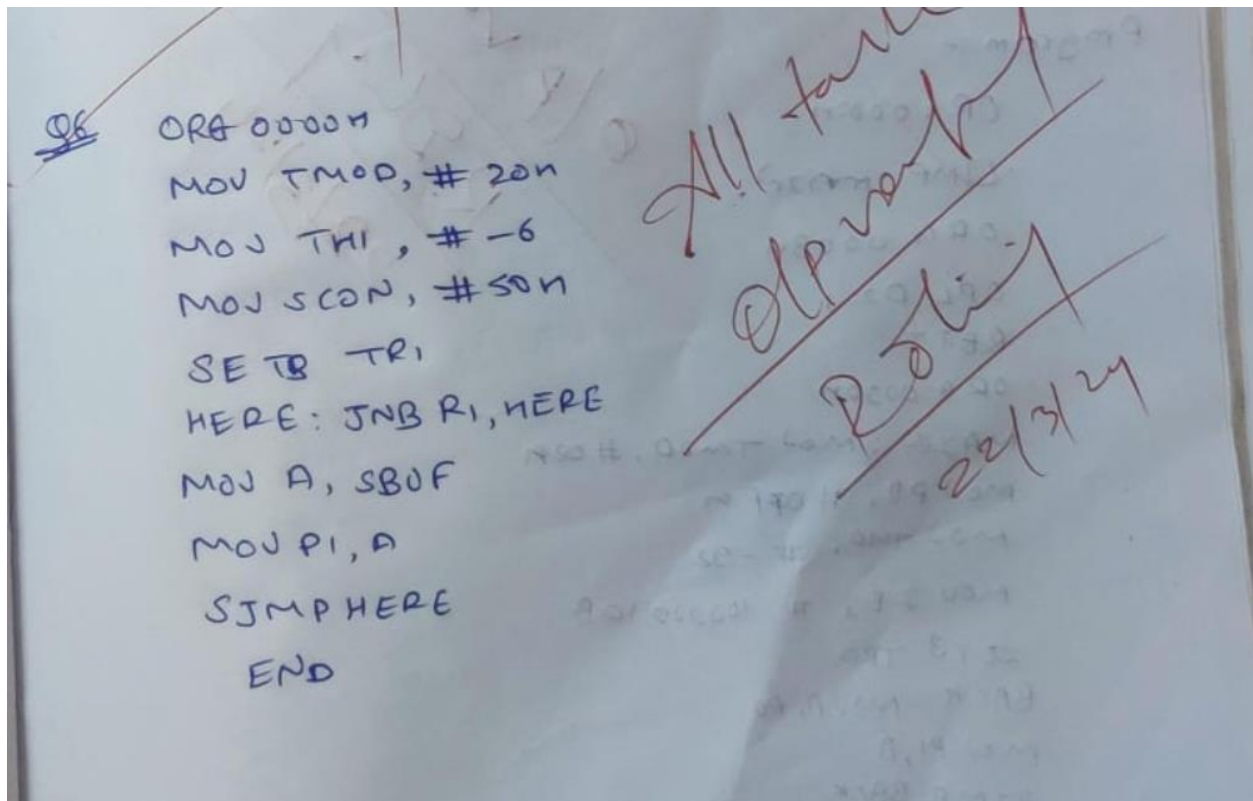
```
SETB TR1
```

```
MOV R1,#14
AGAIN:CLR A
MOVC A,@A+DPTR
MOV SBUF,A
HERE: JNB TI,HERE
CLR TI
INC DPTR
DJNZ R1,AGAIN
SJMP XX
MYDATA: DB 'VIT UNIVERSITY'
END
```

Result: The program sets up serial communication at a baud rate of 9600 with 8-bit data and one stop bit. You can observe the transmitted data in the serial window of the simulator.

Q6:

Aim: To write a 8051 Assembly Language program to get data from the PC and display it on P1. Assume 8051 is connected to PC and observe the incoming characters. As you press a key on the PC's keyboard, the character is sent to the 8051 serially at 4800 baud rate and is displayed on LEDs. The characters displayed on LEDs are in ASCII (binary).



Code:

```
ORG 0000H

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
```

```
END
```

Result: The program allows the 8051 microcontroller to receive characters from a PC at 4800 baud rate and display them on port P1, which is connected to LEDs. As characters are typed on the PC's keyboard, they are sent to the 8051 serially and displayed on the LEDs in ASCII binary form.