# Program No.: 5

| | | | |
|---|---|---|---|
| **Reg.No** | 19BEC0358 | | |
| **Student Name** | ARPIT PATAWAT | | |
| **Course Code** | ECE3003 | **Slot & Semester** | L35+L36<br>FALL ~ 2021-22 |
| **Course Name** | Microcontroller and its applications | | |
| **Program Title** | INTERRUPT PROGRAM | | |
| **Date of Exp.** | 08-11-2021 | **Date of Submission** | 21-11-2021 |
| **Faculty** | A. Karthikeyan | | |

SCHOOL OF ELECTRONICS ENGINEERING
VIT, VELLORE

# Question

1. Write an 8051 program to get data from port P0 and send it to port P1continuously while an interrupt will do the following: Timer 0 will toggle the P2.1 bit every 200 microseconds.
2. Write an 8051 program to get data from a single bit of P1.2 and send it to P1.7
   continuously while an interrupt will do the following:
   A serial interrupt service routine will receive data from a PC and display it on P2 ports.
   9600 BAUD RATE
3. Write a program using interrupts to do the following:
   (a) Receive data serially and sent it to P0,
   (b) Have P1 port read and transmitted serially, and a copy given to P2,
   (c) Make timer 0 generate a square wave of 5kHz frequency on P3.2.
   Assume that XTAL-11.0592. Set the baud rate at 4800.

# TASK 1) -

**Aim**: To write an 8051 ALP get data from port 0 and send to port 1 also using interrupt to toggle P2.1 bit every 200uS using keil software and to verify the result manually.

**Tools Required - Keil Micro vision Software**

**Algorithm:**

1. At the main memory address, set timer mode, load the count value and start timer
2. Make P0 as an input
3. Enable the timer 0 interrupt
4. At the address of interrupt vector table for timer interrupt, toggle the P2.1
5. Take data from port 0 and move it back to port1(main program)
6. Repeat step 5 on loop

**Program:**

| Label | Mnemonics | Operands | addressing mode used | Machine cycle Required | Memory Byte Required | Type of Instruction | Comments | Flags getting affected by the Instruction. |
|---|---|---|---|---|---|---|---|---|
| | ORG | 0000H | | | | Assembler directive | Defining origin of the program | NONE |
| | LJMP | MAIN | | 2 | 3 | Program branching | Make a long jump to main label | NONE |
| | ORG | 000BH | | | | Assembler directive | Timer 0 interrupt vector table | NONE |
| | CPL | P2.1 | Immediate | 1 | 1 | Logical | Toggle P2.1 | NONE |
| | RETI | | | | | Program branching | Return from ISR | NONE |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ORG | 0030H | | | | Assembler directive | Main function | NONE |
| MAIN: | MOV | TMOD, #02H | Immediate | 1 | 2 | Data transfer | Timer 0 mode 2 is selected | NONE |
| | MOV | P0, #0FFH | Immediate | 1 | 2 | Data transfer | Making port 0 as input | NONE |
| | MOV | TH0,-#92H | Immediate | 1 | 2 | Data transfer | Load -92 value to TH0 | NONE |
| | MOV | IE, #10000010B | Immediate | 1 | 2 | Data transfer | Enable the timer 0 interrupt | NONE |
| | SETB | TR0 | Register | 1 | 2 | Boolean | Start timer 0 | NONE |
| BACK: | MOV | A, P0 | Register | 1 | 1 | Data transfer | Take data from port 0 and move to A | NONE |
| | MOV | P1, A | Register | 1 | 1 | Data transfer | Take data from port A and move to P1 | NONE |
| | SJMP | BACK | | 2 | 2 | Program branching | Short hump to back | NONE |
| | END | | | | | Assembler directive | End of the program | NONE |

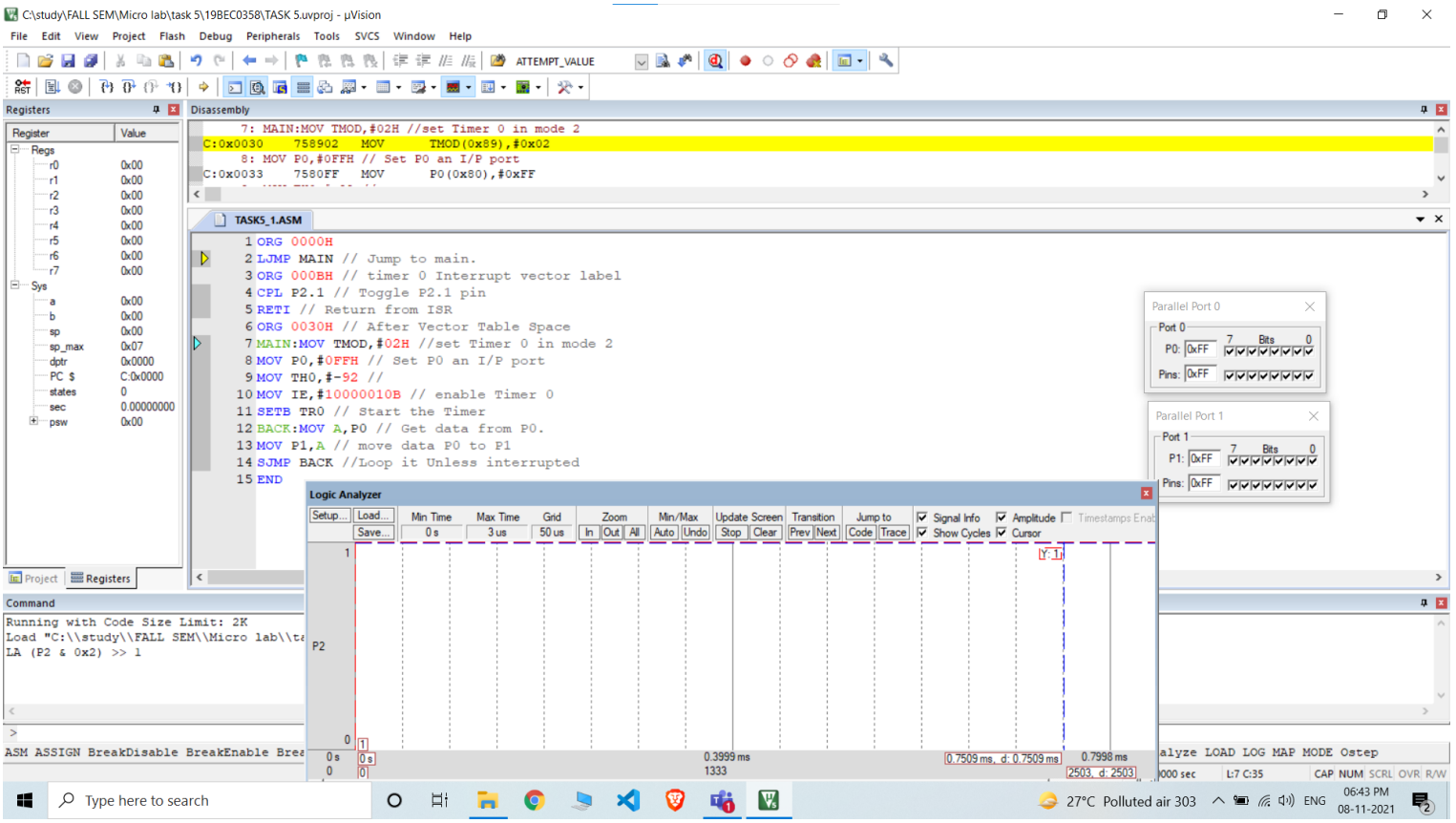
**Output:** the output can be seen in port 1 and a wave on P2.1 with time period 200uS

**Manual Calculation: total time period = 200us, timer period for half of the pulse = 100us**

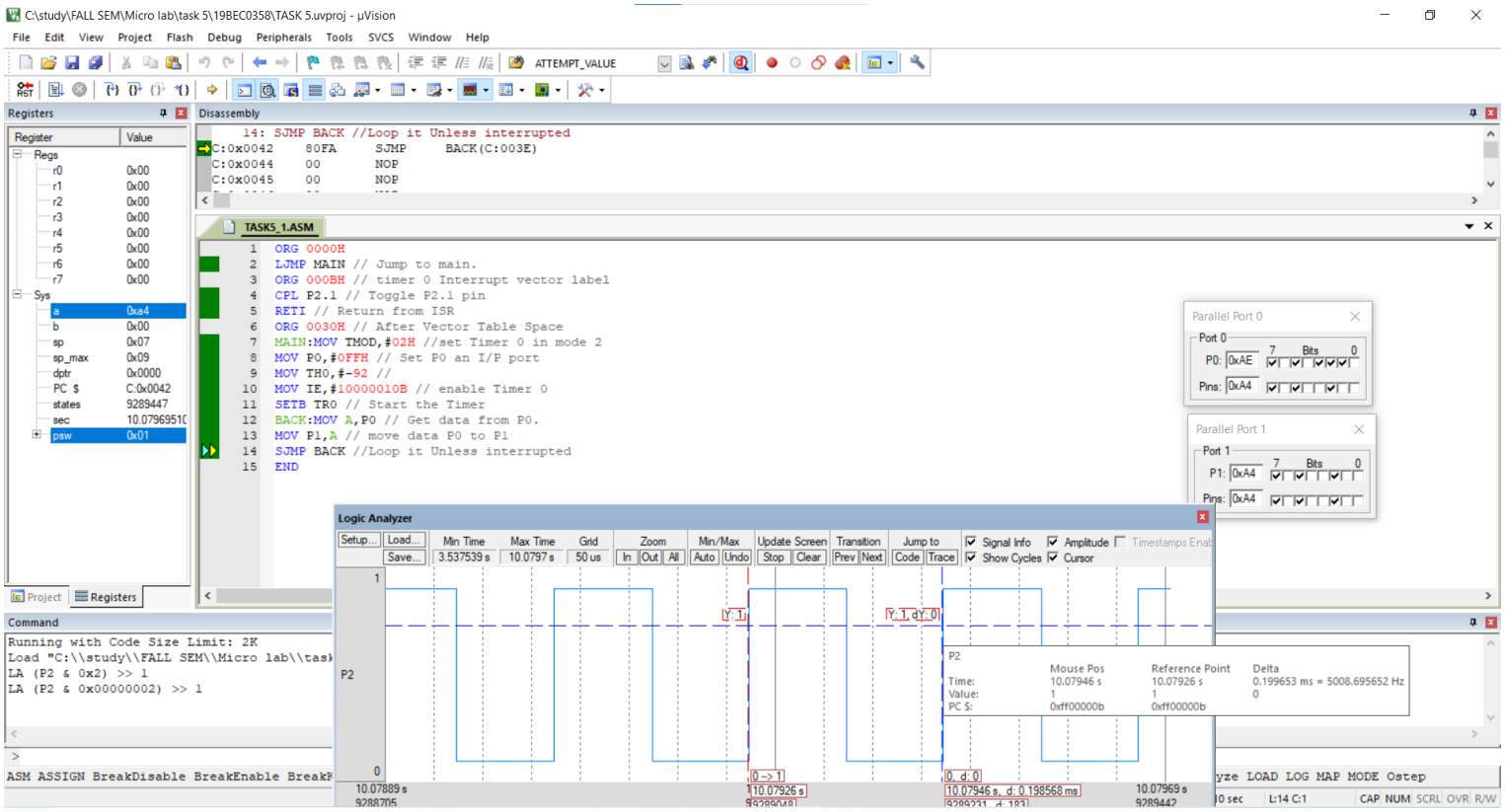
**Count value required = 100us/1.085us = 92.16 → 92 so count value loaded to TH0 = -92**

## Results and Observations

Print Screen of the Program and registers before execution:




Print Screen of the Program and registers after execution:




## Inferences:

1.P0 will get the data and send it continuously to P1.

2.at P2.1 a wave will be generated with timer period of 200us

**Result:** the 8051 ALP to exchange data at ports and generating a pulse at P2.1using Keil software is executed and the results are verified Manually.

# TASK 2) -

**Aim**: To write an 8051 ALP get data from P1.2 and send to P1.7 also using interrupt to receive data from PC and display it on Port 2 using keil software and to verify the result manually.

**Tools Required - Keil Micro vision Software**

**Algorithm:**

1.make P1.2 as input

2.make timer 1 mode 2 for serial communication

3.set baud rate

4.enable serial communication interrupt

5.At the address of interrupt vector table for Serial interrupt, take data from PC and send it to Port 2

6.Take data from P1.2 and move it back to P1.7

7.Repeat step 6 on loop

**Program:**

| Label | Mnemonics | Operands | addressing mode used | Machine cycle Required | Memory Byte Required | Type of Instruction | Comments | Flags getting affected by the Instruction. |
|---|---|---|---|---|---|---|---|---|
| | ORG | 0000H | | | | Assembler directive | Defining origin of the program | NONE |
| | LJMP | MAIN | | 2 | 3 | Program branching | Make a long jump to main label | NONE |
| | ORG | 0023H | | | | Assembler directive | Serial interrupt vector table | NONE |
| | LJMP | SERIAL | | 2 | 3 | | | NONE |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ORG | 0030H | | | | Assembler directive | Main function | NONE |
| MAIN: | SETB | P1.2 | | | | | | NONE |
| | MOV | TMOD, #20H | Immediate | 1 | 2 | Data transfer | Timer 1 mode 2 is selected | NONE |
| | MOV | TH1,-#03H | Immediate | 1 | 2 | Data transfer | Load -03 value to TH1 | NONE |
| | MOV | SCON, #50H | Immediate | 1 | 2 | Data transfer | Serial mode 1 with receive enabled | NONE |
| | MOV | IE, #10010000B | Immediate | 1 | 2 | Data transfer | Enable the Serial interrupt | NONE |
| | SETB | TR1 | Register | 1 | 2 | Boolean | Start timer 1 | NONE |
| BACK: | MOV | C, P1.2 | Register | 1 | 1 | Data transfer | Take data from port 1.2 and move to carry flag | Carry flag |
| | MOV | P1.2, C | Register | 1 | 1 | Data transfer | Take data from Carry flag and move to P1.2 | Carry flag |
| | SJMP | BACK | | 2 | 2 | Program branching | Short hump to back | NONE |

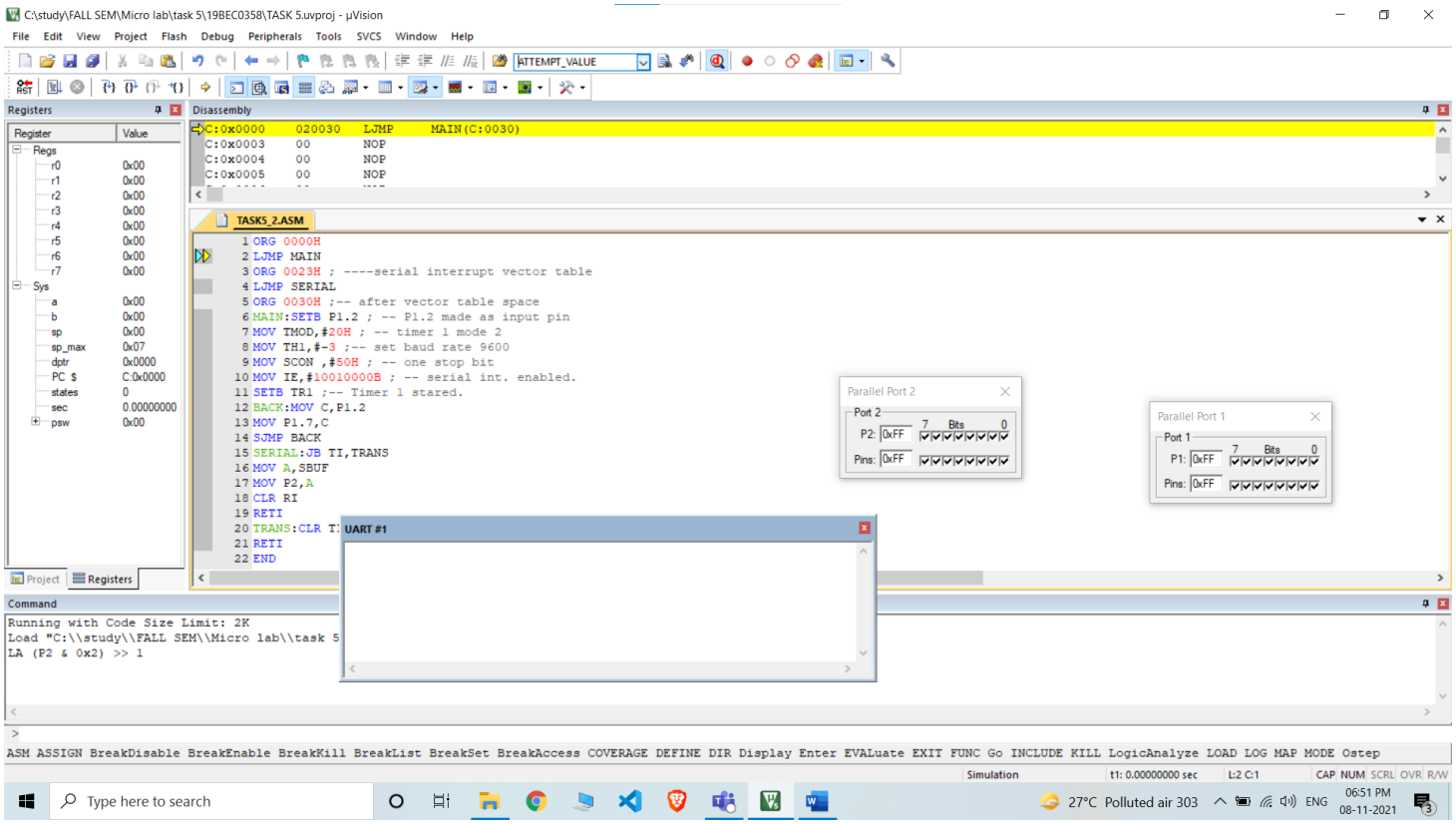
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| SERIAL: | JB | TI, TRANS | | 2 | 3 | Boolean | Jump to trans label if data is transferred | NONE |
| | MOV | A, SBUF | Register | 1 | 1 | Data transfer | Take data from SBUF and mov to A | NONE |
| | MOV | P2,A | Register | 1 | 1 | Data transfer | Take data from A and move to P2 | NONE |
| | CLR | RI | Register | 1 | 2 | Boolean | Clear RI so that next data can be received | NONE |
| | RETI | | | | | Program branching | Return from ISR | NONE |
| TRANS | CLR | TI | Register | 1 | 2 | Boolean | Clear TI so that next data can be transferred | NONE |
| | RETI | | | | | Program branching | Return from ISR | NONE |
| | END | | | | | Assembler directive | End of the programme | NONE |

**Output:** the output can be seen in P1.7 and port 2

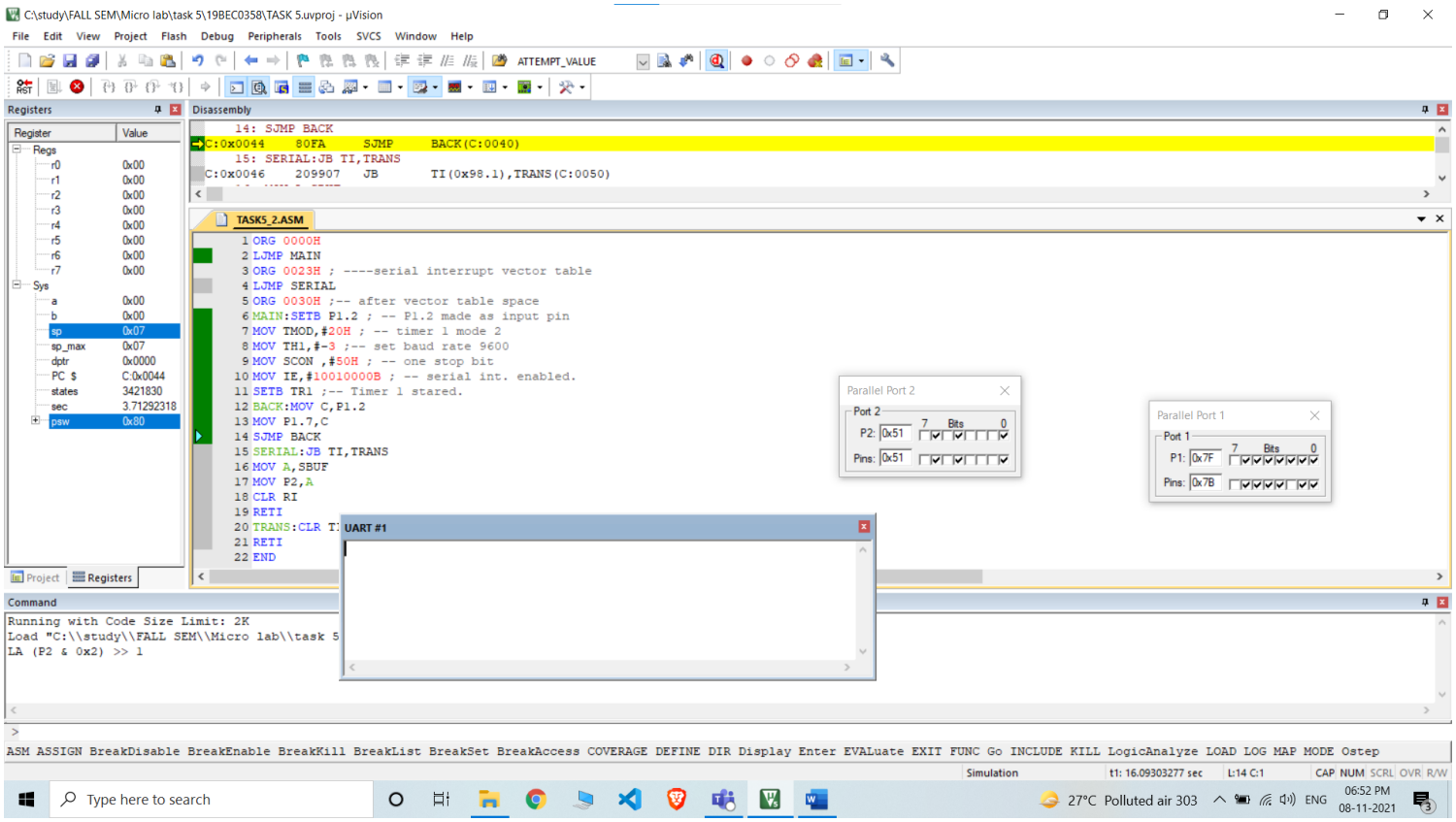
**Manual Calculation:** **28800 / 9600 = 3**

# Results and Observations

Print Screen of the Program and registers before execution:




Print Screen of the Program and registers after execution:



## Inferences:

1.P1.7 is changing according to P1.2

2.when a key is pressed, the corresponding ASCII value can be seen in Port 2

**Result:** the 8051 ALP to exchange data at ports and sending data from PC to port 2 using Keil software is executed and the results are verified Manually.

# TASK 3) -

**Aim**: To write an 8051 ALP to get data from PC, send it to P0 then take data from P1 and transfer serially along with giving copy to P2 then finally generate a square wave at P3.2 using keil software and to verify the result manually.

**Tools Required - Keil Micro vision Software**

**Algorithm:**

1.make Port 1 as input

2.set timer 1 mode 2 for serial communication and timer 0 mode 1 for square wave

3.enable serial and timer 0 interrupt

4.set baud rate and start both the timers

5.for the main function – take data from P1 and send it to P2 as well as UART

6.for the timer interrupt- compliment P3.2 then set the value of TH1 and TL0 as for timer 1 we need to reload the value every time the its rolls down

7.for the Serial interrupt – receive data from UART and sent it to A

**Program:**

| Label | Mnemonics | Operands | addressing mode used | Machine cycle Required | Memory Byte Required | Type of Instruction | Comments | Flags getting affected by the Instruction. |
|---|---|---|---|---|---|---|---|---|
| | ORG | 0000H | | | | Assembler directive | Defining origin of the program | NONE |
| | LJMP | MAIN | | 2 | 3 | Program branching | Make a long jump to main label | NONE |
| | ORG | 000BH | | | | Assembler directive | Timer 0 interrupt vector table | NONE |
| | CPL | P3.2 | Immediate | 1 | 1 | Logical | Toggle P3.2 | NONE |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | MOV | TH0, #0FFH | Immediate | 1 | 1 | Data transfer | Load value of TH0 | NONE |
| | MOV | TL0, #0A4H | Immediate | 1 | 1 | Data transfer | Load value of TL0 | NONE |
| | RETI | | | | | Program branching | Return from ISR | NONE |
| | ORG | 0023H | | | | Assembler directive | Serial interrupt vector table | NONE |
| | LJMP | SERIAL | | 2 | 3 | | | NONE |
| | ORG | 0030H | | | | Assembler directive | Main function | NONE |
| MAIN: | MOV | P1, #0FFH | Immediate | 1 | 2 | Data transfer | Making port 0 as input | NONE |
| | MOV | TMOD, #00100001B | Immediate | 1 | 2 | Data transfer | Timer 1 mode 2 + Timer 0 mode 1 is selected | NONE |
| | MOV | TH1,-#06H | Immediate | 1 | 2 | Data transfer | Load -03 value to TH1 baud rate = 4800 | NONE |
| | MOV | SCON, #50H | Immediate | 1 | 2 | Data transfer | Serial mode 1 with receive enabled | NONE |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | MOV | IE, #10010010B | Immediate | 1 | 2 | Data transfer | Enable the Serial + timer 0 interrupt | NONE |
| | SETB | TR1 | Register | 1 | 2 | Boolean | Start timer 1 | NONE |
| | SETB | TR0 | Register | 1 | 2 | Boolean | Start timer 0 | NONE |
| BACK: | MOV | A, P1 | Register | 1 | 1 | Data transfer | Take data from port 1 and move to A | NONE |
| | MOV | SBUF, A | Register | 1 | 1 | Data transfer | Take data from A and move to SBUF | NONE |
| | MOV | P2, A | Register | 1 | 1 | Data transfer | Take data from A and move to P2 | NONE |
| HERE | JNB | TI, HERE | | 2 | 3 | Boolean | Jump to HERE label until data is not transferred | NONE |
| | CLR | TI | Register | 1 | 2 | Boolean | Clear TI so that next data can be transferred | NONE |
| | SJMP | BACK | | 2 | 2 | Program branching | Short hump to back | NONE |
| SERIAL: | MOV | A, SBUF | Register | 1 | 1 | Data transfer | Take data from | NONE |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SBUF and mov to A | |
| | MOV | P0,A | Register | 1 | 1 | Data transfer | Take data from A and move to P0 | NONE |
| | CLR | RI | Register | 1 | 2 | Boolean | Clear RI so that next data can be received | NONE |
| | RETI | | | | | Program branching | Return from ISR | NONE |
| | END | | | | | Assembler directive | End of the programme | NONE |

**Output:** the output can be seen in P0 , UART, P2 and at P3.2

**Manual Calculation: 28800 / 4800 = 6 = baud rate**

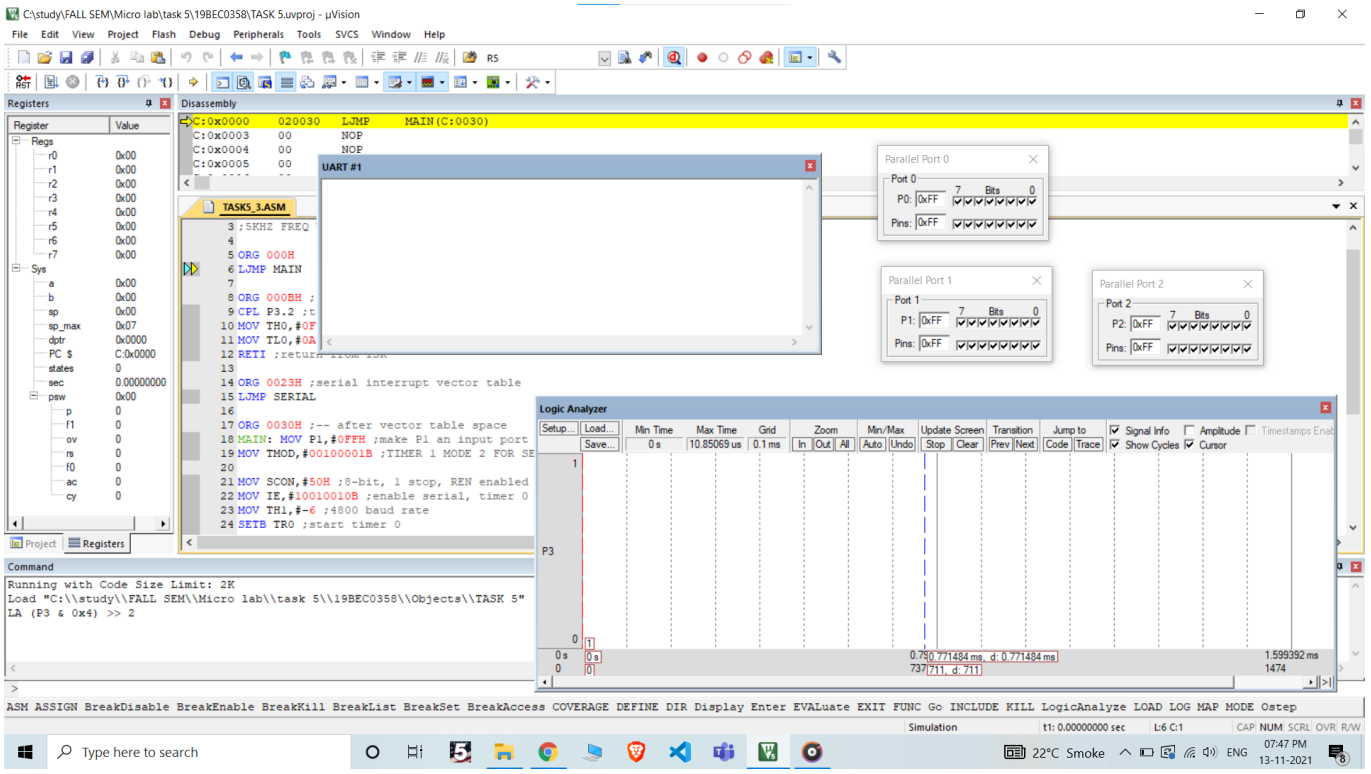
**Frequency = 5kHZ → timer = 1/f = 0.2ms and for half pulse → 0.1ms**

**Count= 0.1ms/1.085us = 92.16 → 92**

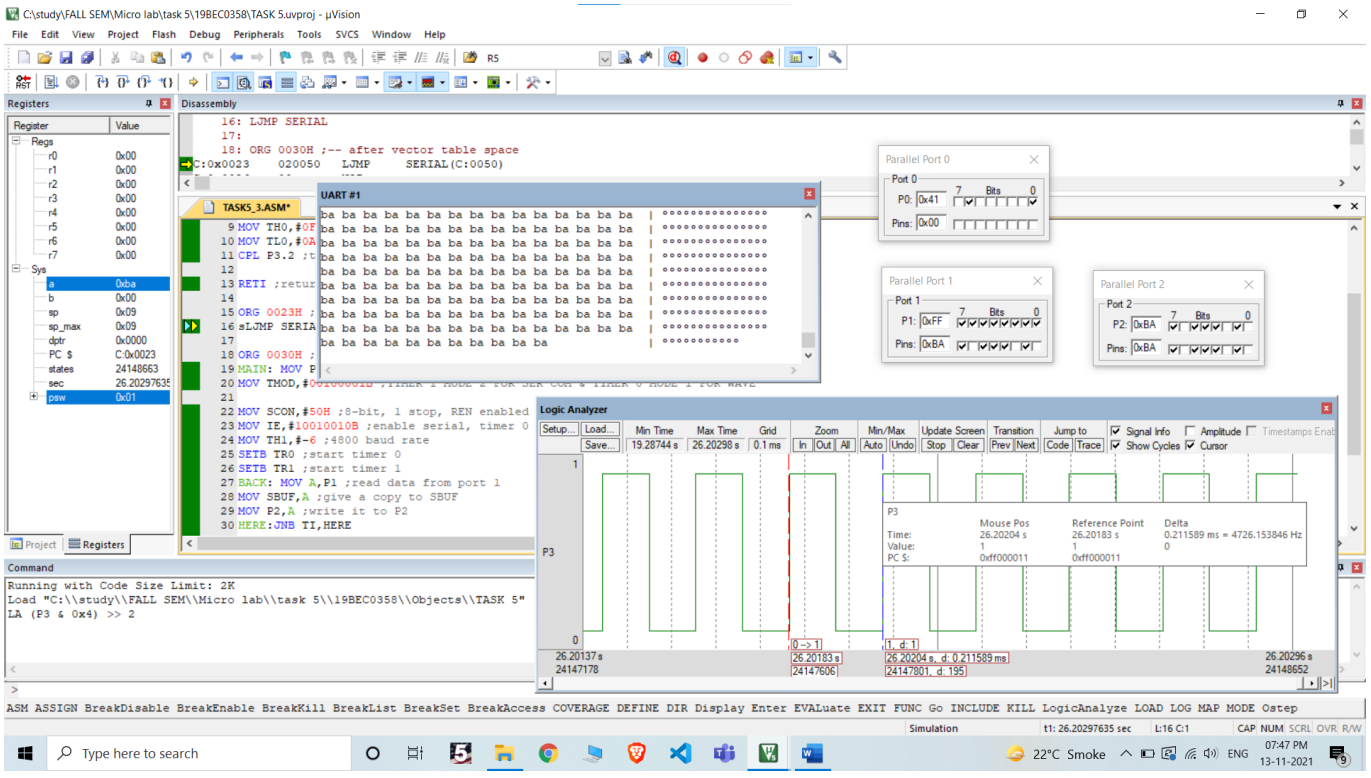
**65536 – 92 = 65444 = FFA4 In Hex**

# Results and Observations

Print Screen of the Program and registers before execution:




Print Screen of the Program and registers after execution:



## Inferences:

1.P0 changes with the ASCII value whenever a key is pressed in the UART

2.when P1 value changes then it is reflected in P2 as well as UART

3.a square wave of frequency of 5Khz is generated at P3.2

**Result:** the 8051 ALP to transfer ports data, generating square wave and displaying data to UART using Keil software is executed and the results are verified Manually.

----------xxxxx----------