Part-4
GROUP-6
E/21/302
E/21/087

```
loadi 0 0x06
loadi 1 0x01
loadi 2 0x03
loadi 3 0x08
sub   0 0 1
beq   0x01 0 2
j     0xFD
add   4 0 3
```

Explanation:
This code implements a synchronous loop that repeatedly decrements the value in register R0 by 1 until it becomes equal to the value in register R2 (which is 3). Here's what happens:

1.Initial Setup:

- R0 is initialized to 6

- R1 is 1 (used for decrementing)

- R2 is 3 (target value)

- R3 is 8 (used for the final addition)

2.Loop Execution:

- The sub instruction decreases R0 by 1 in each cycle.

- The beq instruction checks whether R0 == R2 .

- If not equal, the j 0xFD instruction (which is a relative jump of -3) sends the control back to the sub instruction.

- This loop continues until R0 becomes 3.

3.Exit Condition:

- Once R0 == R2, the beq becomes true and the jump is skipped.

- Then, it proceeds to execute the add instruction, computing R4 = R0 + R3 = 3 + 8 = 11.

4. Result:

The final result is stored in R4 as 11.

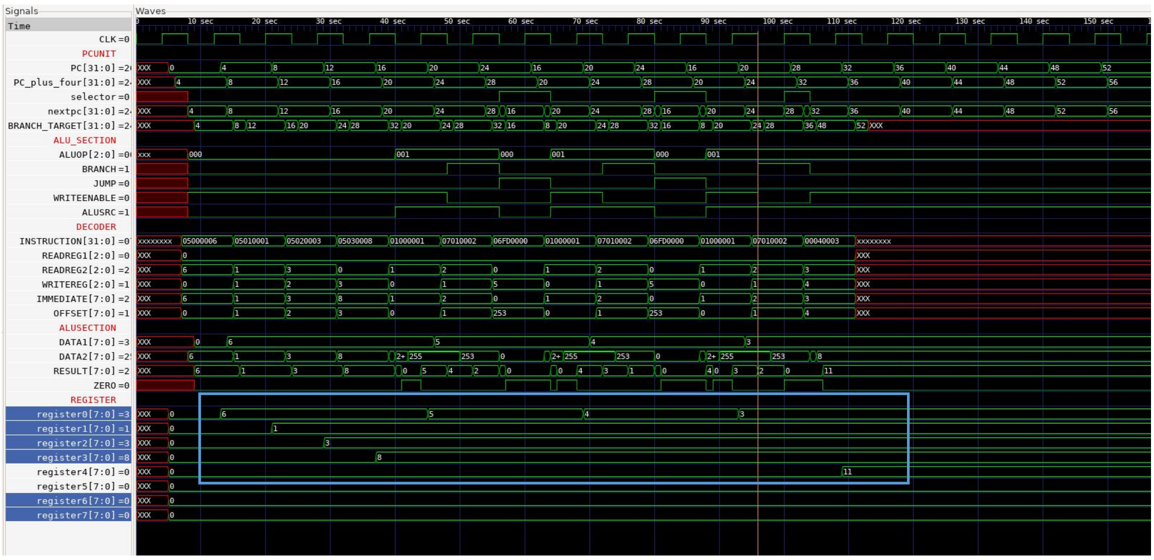
GTKWave Form :
You will observe the value of R0 decrementing synchronously: 6 → 5 → 4 → 3.

Each decrement corresponds to one loop cycle.

Once R0 becomes 3, the loop exits, and the add instruction is triggered.

You can also track R4, which is updated to 11 after the loop ends.

This waveform visually confirms the synchronized decrement and conditional jump behavior until the match condition (R0 == R2) is met.


Signals
Waves
Time
CLK =0
PCUNIT
PC[31:0]
PC_plus_four[31:0]
selector =0
nextpc[31:0]
BRANCH_TARGET[31:0]
ALU_SECTION
ALUOP[2:0]
BRANCH =1
JUMP =0
WRITEENABLE =0
ALUSRC =1
DECODER
INSTRUCTION[31:0]
READREG1[2:0] =0
READREG2[2:0] =2
WRITEREG[2:0] =1
IMMEDIATE[7:0] =2
OFFSET[7:0] =1
ALUSECTION
DATA1[7:0] =3
DATA2[7:0]
RESULT[7:0] =2
ZERO =0
REGISTER
register0[7:0] =3
register1[7:0] =1
register2[7:0] =3
register3[7:0] =8
register4[7:0] =0
register5[7:0] =0
register6[7:0] =0
register7[7:0] =0