COA LAB
Experiment - 5

Write an assembly language for the following conditional statements:

1. If R02 is greater than R01, R03 is set to 8. (Use R01 as the first operand and R02 as the second operand).
2. If R01 = 0, R03 is set to 5, else R03 is set to R01 plus 1.
3. A loop that repeats 5 times where R02 is incremented by 2 every time the loop repeats.
4. A loop that repeats while R04 is > 0. Set the initial value of R04 to 8.
5. A loop that repeats until R05 is > R09. Set the initial values of R05 to 0 and R09 to 12.

Program no. 1

Algorithm:

1. Load the value 15 into R01
2. Load the value 5 into R02
3. Compare R01 and R02 with the CMP instruction
4. If R02 is greater than R01, jump to the ADD8 label
5. If R02 is not greater than R01, halt the program with the HLT instruction
6. At the ADD8 label, load the value 8 into R03
7. Halt the program with the HLT instruction

Assembly Language code:

```
MOV #15, R01 ;Load the value 15 into R01
MOV #5, R02 ;Load the value 5 into R02
CMP R01, R02 ;Compare R01 and R02
JGT ADD8 ;If R02 is greater than R01, jump to the ADD8 label
HLT ;If R02 is not greater than R01, halt the program
ADD8: ;Label for the ADD8 instruction
MOV #8, R03 ;Load the value 8 into R03
HLT ;Halt the program
```

Result:

Case 1: R02 > R01

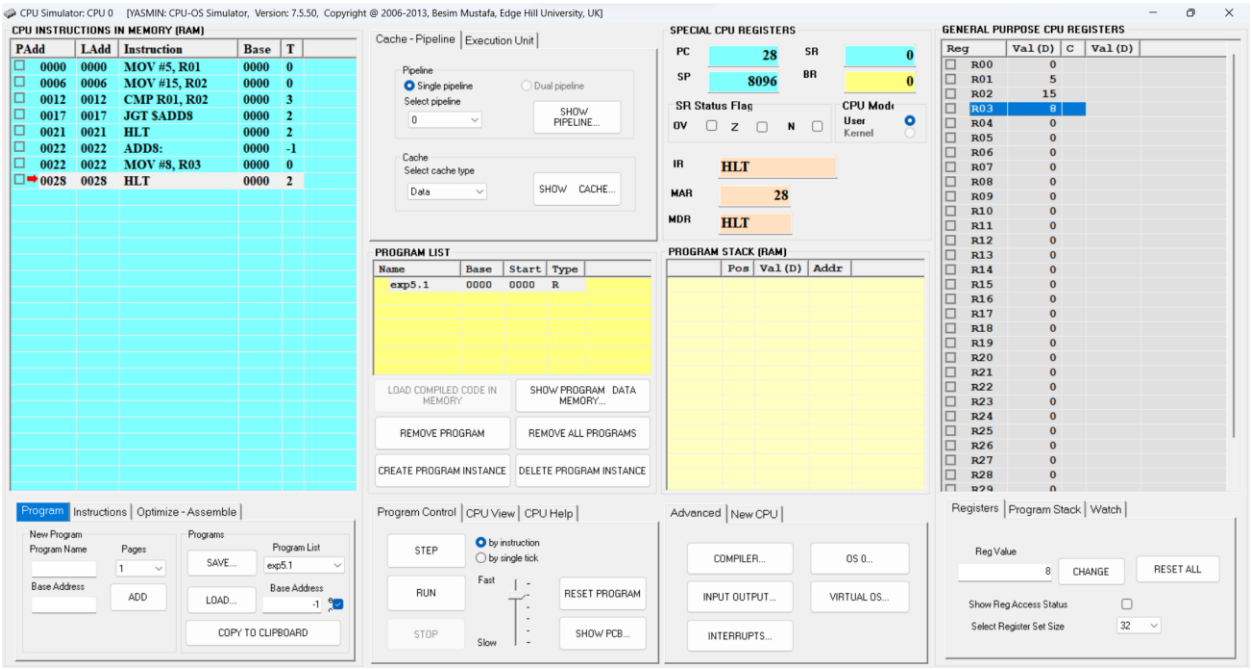

Fig.1: CPU Simulator Window

Case 2: R02 < R01

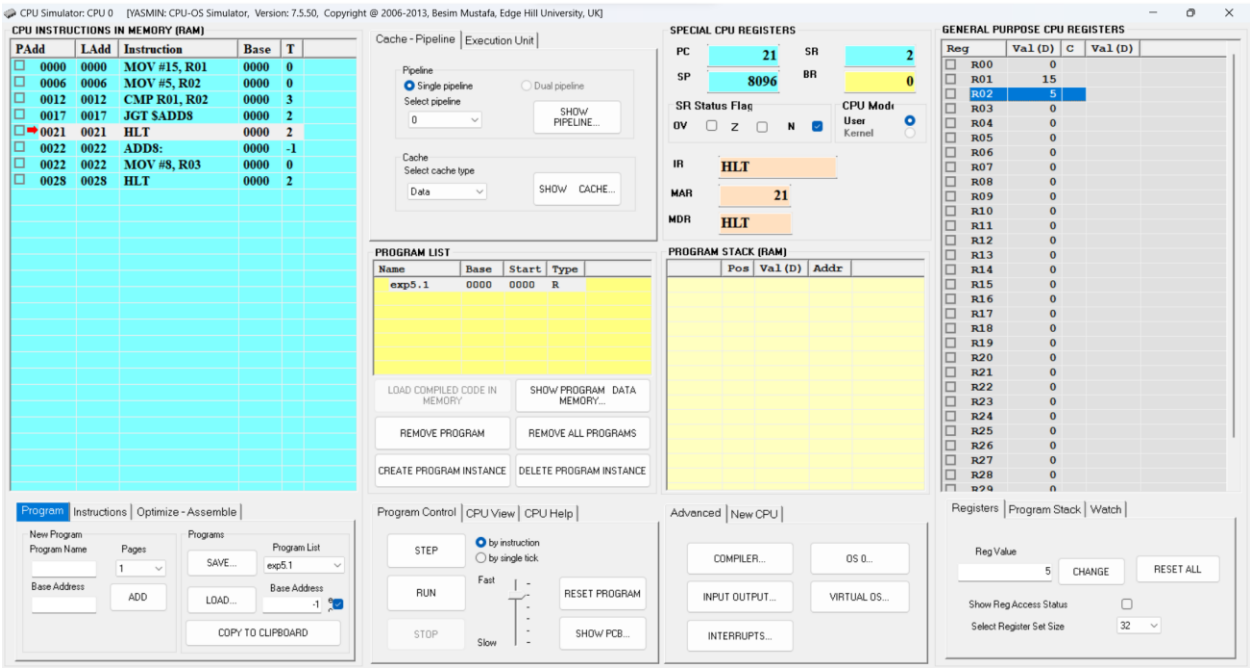

Fig.2: CPU Simulator Window

Program no. 2

Algorithm:

1. Initialize R01 to 0
2. Compare R01 with 0

3. If R01 equals 0, go to IF statement
4. If R01 does not equal 0, go to ELSE statement
5. At IF statement, set R03 to 5
6. At ELSE statement, add 1 to R01 and set R03 to R01
7. Halt the program.

Assembly Language code:

```
MOV #0, R01 ;Initialize R01 to 0
CMP #0, R01 ;Compare R01 with 0
JEQ $IF ;If R01 equals 0, go to IF statement
JNE $ELSE ;If R01 does not equal 0, go to ELSE statement
$IF: ;At IF statement, set R03 to 5
MOV #5, R03
HLT ;Halt the program
$ELSE: ;At ELSE statement, add 1 to R01 and set R03 to R01
ADD #1, R01
MOV R01, R03
HLT ;Halt the program
```

Result:

Case 1: R01 = 0

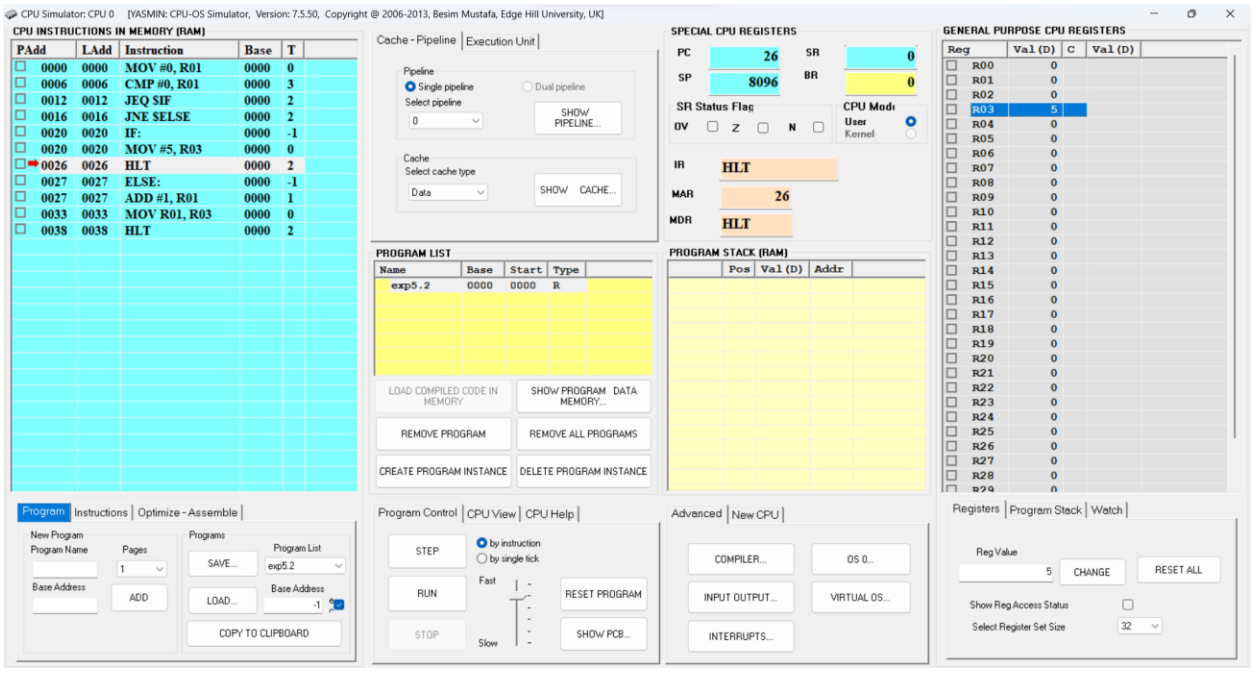

Fig.3: CPU Simulator Window

Case 2: R01 != 0

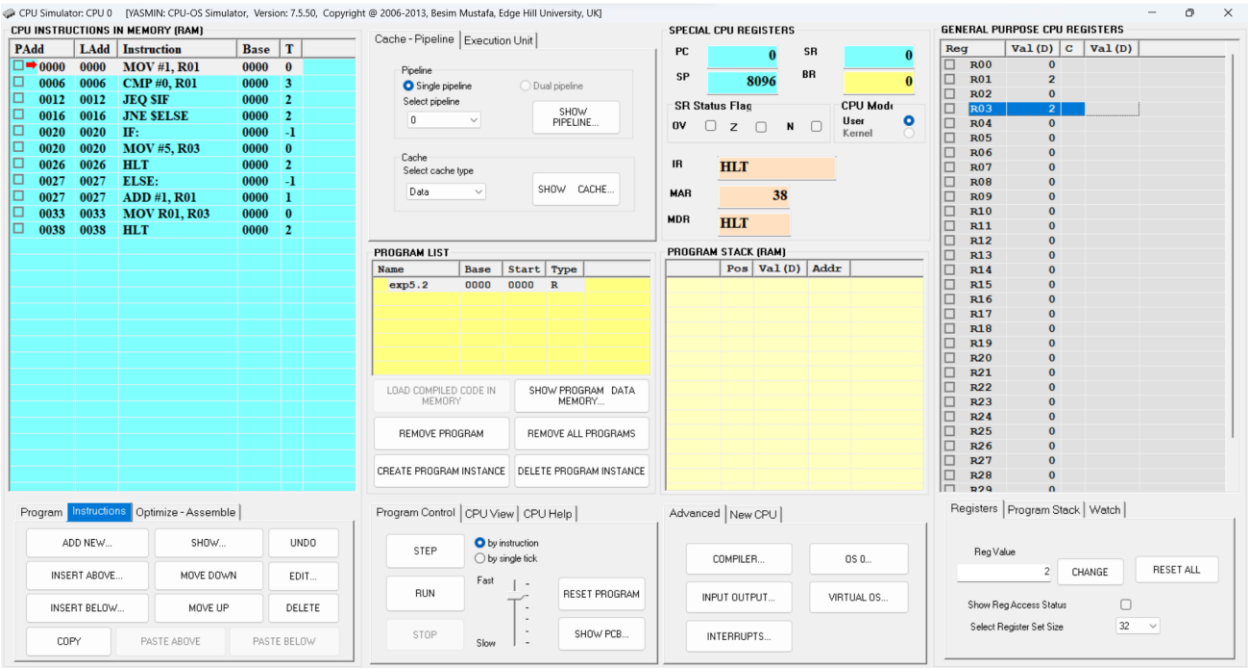

Fig.4: CPU Simulator Window

Program no. 3

Algorithm:

1. Initialize the value of R01 to 1.
2. Initialize the value of R02 to 5.
3. Repeat the following steps until R01 is equal to 5:
    a. Increment R02 by 2.
    b. Increment R01 by 1.
4. Halt the program.

Assembly Language code:

MOV #1, R01 *;move the value 1 into register R01*
MOV #5, R02 *;move the value 5 into register R02*
LOOP: *;label the start of the loop as "LOOP"*
ADD #2, R02 *;add the value 2 to register R02*
ADD #1, R01 *;add the value 1 to register R01*
CMP #5, R01 *;compare the value in register R01 to 5*
JNE $LOOP *;jump to "LOOP" if R01 is not equal to 5*
HLT *;halt the program*

Result:

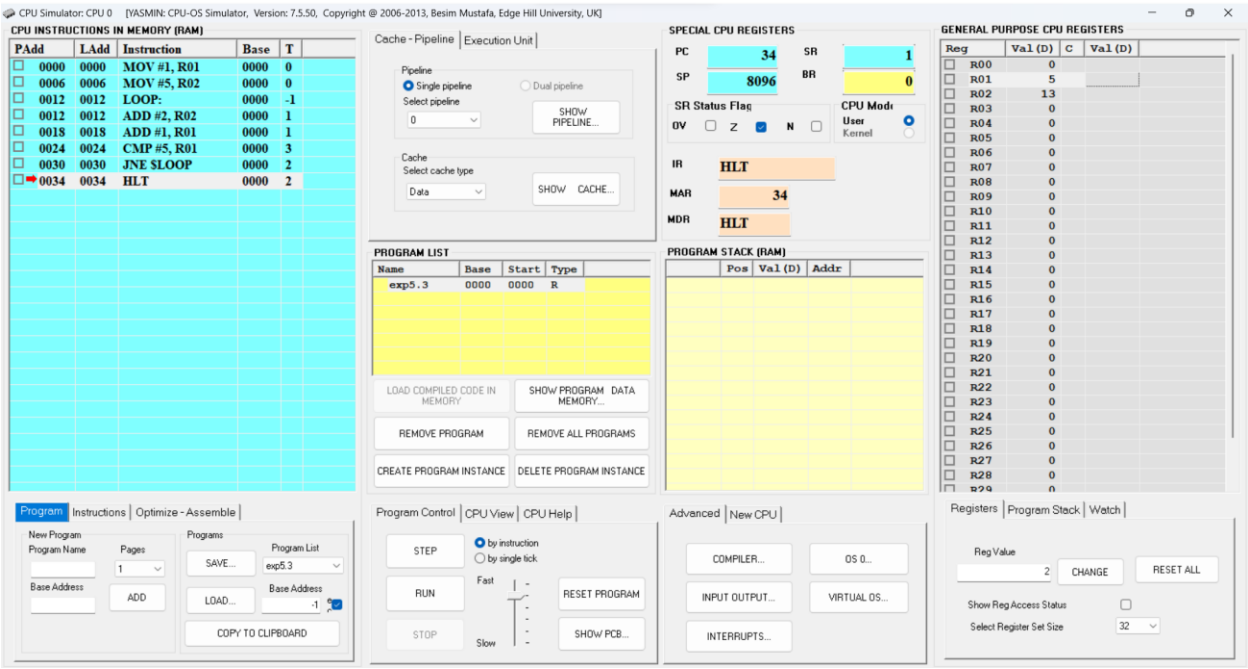

Fig.5: CPU Simulator Window

Program no. 4

Algorithm:

1. Initialize R04 to 8.
2. Start the loop: "LOOP"
3. Subtract 1 from R04.
4. Compare R04 to 0.
5. If R04 is not equal to 0, go back to the loop.
6. If R04 is equal to 0, halt the program.

Assembly Language code:

MOV #8, R04 *;Move value 8 into register R04*
LOOP: *;Label for the start of the loop*
SUB #1, R04 *;Subtract value 1 from the contents of R04*
CMP #0, R04 *;Compare the contents of R04 with value 0*
JNE LOOP *;If R04 is not equal to 0, jump back to the label LOOP*
HLT *;Halt the program execution.*

Result:

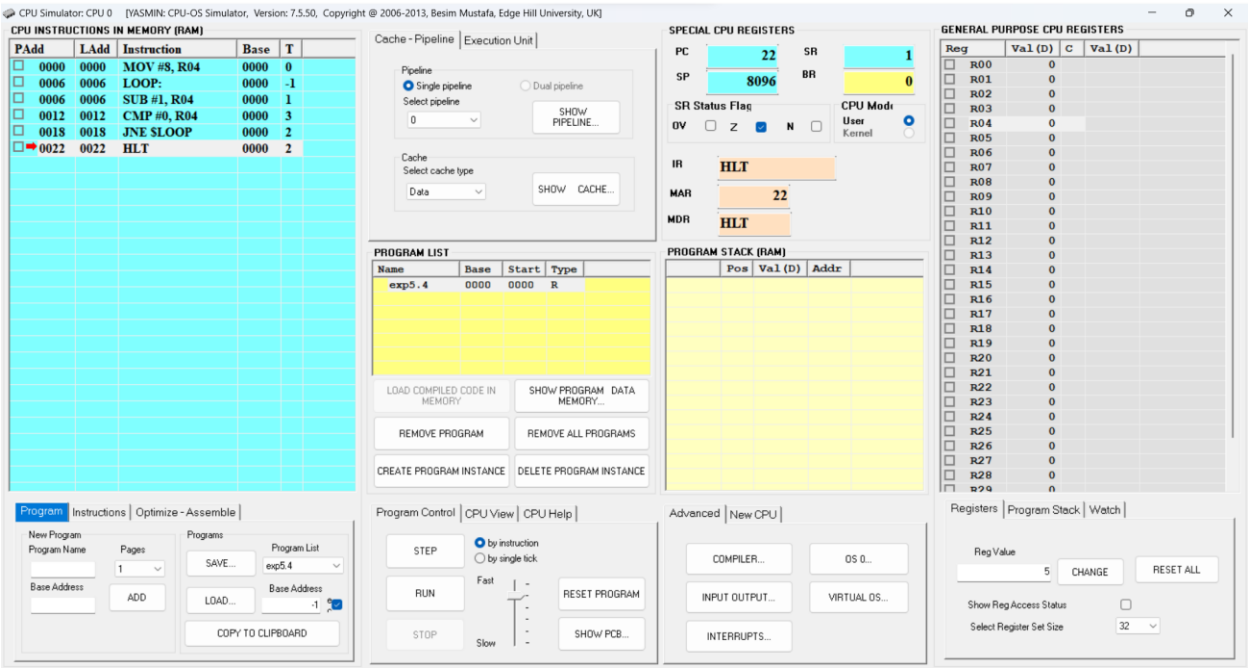

Fig.6: CPU Simulator Window

Program no. 5

Algorithm:

1. Initialize R05 to 0 and R09 to 12.
2. Start the loop with the label "LOOP".
3. Decrement R09 by 1.
4. Compare R05 and R09. If R05 is greater than R09, go to step 7.
5. If R05 is not greater than R09, repeat from step 3.
6. End the loop.
7. Halt the program.

Assembly Language code:

MOV #0, R05 *;Initialize R05 to 0*
MOV #12, R09 *;Initialize R09 to 12*
LOOP: *;Start the loop*
SUB #1, R09 *;Decrement R09 by 1*
CMP R05, R09 *;Compare R05 and R09*
JGT $LOOP *;If R05 is greater than R09, jump to the end of the loop*
HLT *;Halt the program*

Result:

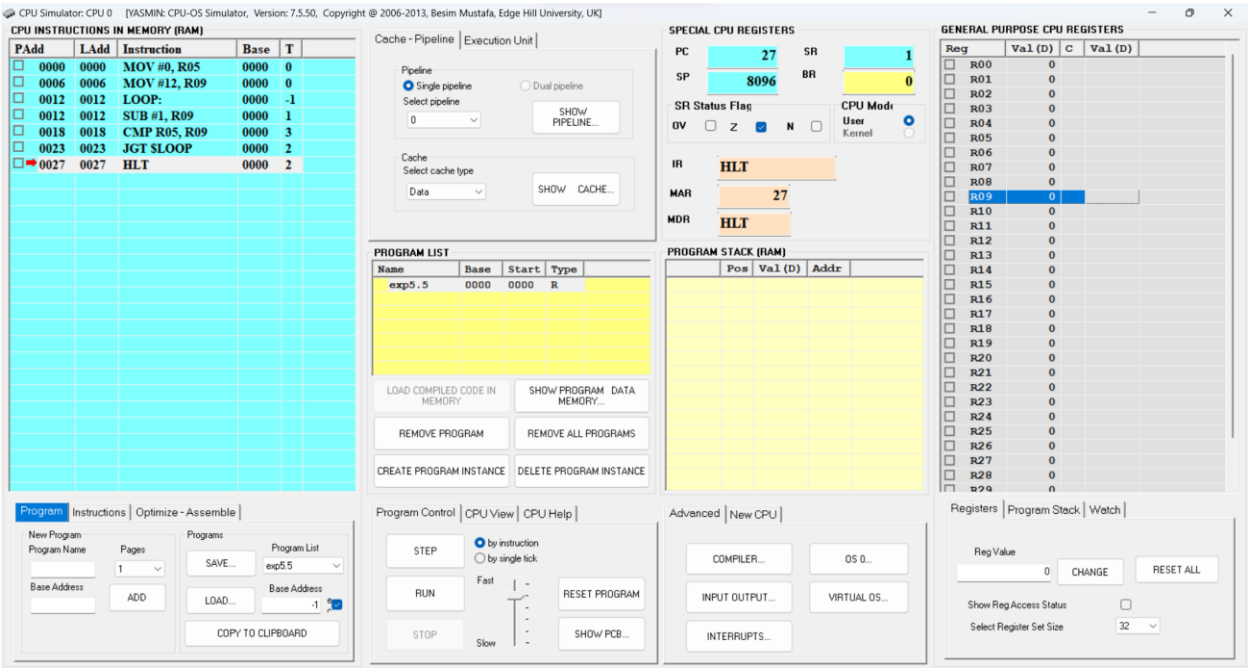

Fig.7: CPU Simulator Window