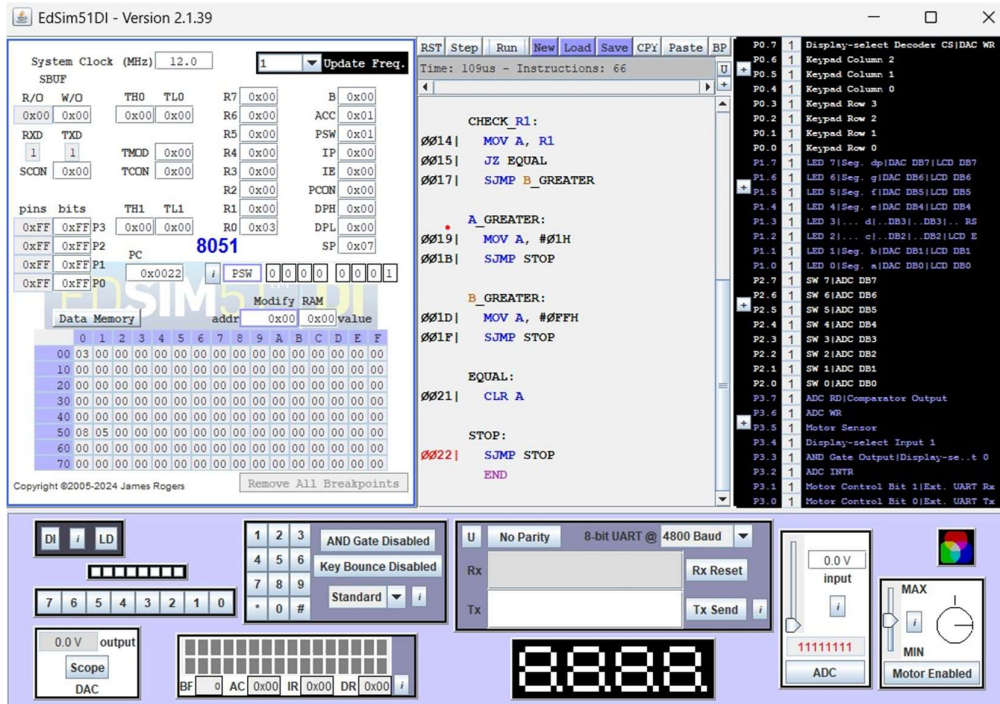
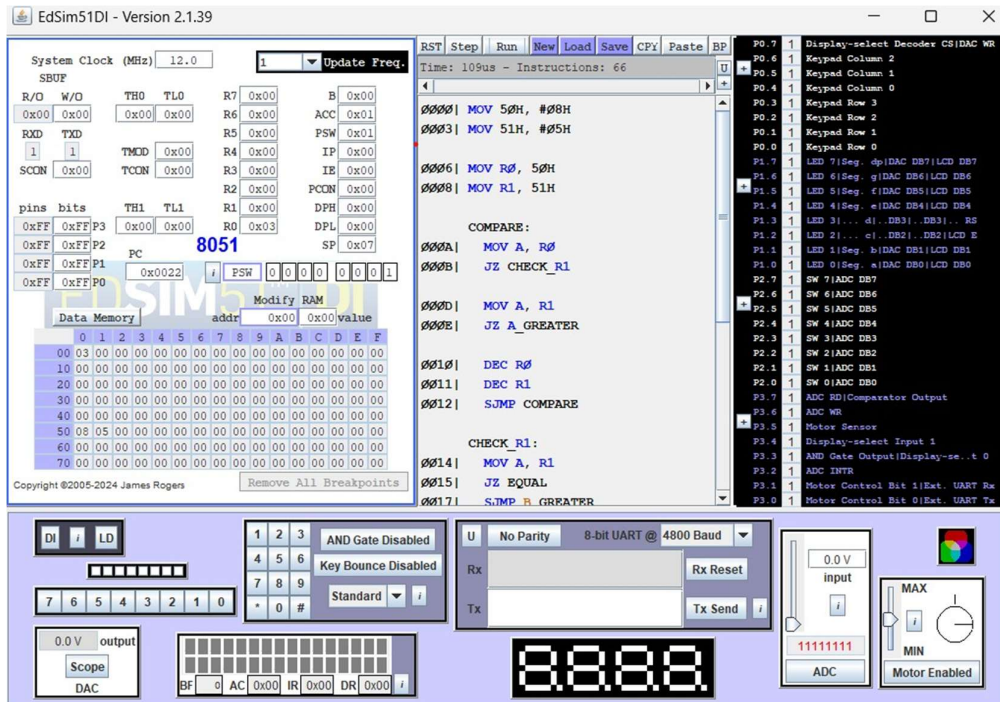




- Write an 8051 Assembly Language Program to compare two unsigned numbers stored at internal RAM locations 50H and 51H without using CJNE, DJNZ, or SUBB. Only MOV, INC, DEC, JZ, JNZ, CLR, SETB, ANL, and ORL instructions are allowed. Store the result such that 01H indicates 50H > 51H, 00H indicates equality, and FFH indicates 50H < 51H.



- A student claims that two assembly programs are equivalent because they access the same RAM address. Using one program with direct addressing and another with indirect addressing, demonstrate why this claim is incorrect by showing a case where both reference the same base address but produce different outputs. Explain how the difference in addressing modes causes this behavior.

EdSim51DI - Version 2.1.39

System Clock (MHz): 12.0

Update Freq. 1

Time: 25us - Instructions: 13

```

0000 MOV 30H, #40H
0003 MOV 40H, #55H

; PROGRAM 1
0006 MOV A, 30H ;DIRECT ADDRESSING
0009 STOP: SJMP STOP
END

```

PC: 0x0008

Data Memory:

Addr	Value
00	00
01	00
02	00
03	00
04	00
05	00
06	00
07	00
08	00
09	00
0A	00
0B	00
0C	00
0D	00
0E	00
0F	00
10	00
11	00
12	00
13	00
14	00
15	00
16	00
17	00
18	00
19	00
1A	00
1B	00
1C	00
1D	00
1E	00
1F	00
20	00
21	00
22	00
23	00
24	00
25	00
26	00
27	00
28	00
29	00
2A	00
2B	00
2C	00
2D	00
2E	00
2F	00
30	40
31	00
32	00
33	00
34	00
35	00
36	00
37	00
38	00
39	00
3A	00
3B	00
3C	00
3D	00
3E	00
3F	00
40	55
41	00
42	00
43	00
44	00
45	00
46	00
47	00
48	00
49	00
4A	00
4B	00
4C	00
4D	00
4E	00
4F	00
50	00
51	00
52	00
53	00
54	00
55	00
56	00
57	00
58	00
59	00
5A	00
5B	00
5C	00
5D	00
5E	00
5F	00
60	00
61	00
62	00
63	00
64	00
65	00
66	00
67	00
68	00
69	00
6A	00
6B	00
6C	00
6D	00
6E	00
6F	00
70	00

Output: 8888

EdSim51DI - Version 2.1.39

System Clock (MHz): 12.0

Update Freq. 1

Time: 17us - Instructions: 9

```

0000 MOV 30H, #40H
0003 MOV 40H, #55H

; PROGRAM 2
0006 MOV R0, 30H
0009 MOV A, @R0 ;INDIRECT ADDRESSING
000B STOP: SJMP STOP
END

```

PC: 0x0009

Data Memory:

Addr	Value
00	00
01	00
02	00
03	00
04	00
05	00
06	00
07	00
08	00
09	00
0A	00
0B	00
0C	00
0D	00
0E	00
0F	00
10	00
11	00
12	00
13	00
14	00
15	00
16	00
17	00
18	00
19	00
1A	00
1B	00
1C	00
1D	00
1E	00
1F	00
20	00
21	00
22	00
23	00
24	00
25	00
26	00
27	00
28	00
29	00
2A	00
2B	00
2C	00
2D	00
2E	00
2F	00
30	40
31	00
32	00
33	00
34	00
35	00
36	00
37	00
38	00
39	00
3A	00
3B	00
3C	00
3D	00
3E	00
3F	00
40	55
41	00
42	00
43	00
44	00
45	00
46	00
47	00
48	00
49	00
4A	00
4B	00
4C	00
4D	00
4E	00
4F	00
50	00
51	00
52	00
53	00
54	00
55	00
56	00
57	00
58	00
59	00
5A	00
5B	00
5C	00
5D	00
5E	00
5F	00
60	00
61	00
62	00
63	00
64	00
65	00
66	00
67	00
68	00
69	00
6A	00
6B	00
6C	00
6D	00
6E	00
6F	00
70	00

Output: 8888



- Write an 8051 Assembly Language Program to generate the last four digits of your mobile number using logical instructions (ANL, ORL, CLR). Do not directly load the complete four-digit number. The final result must be available in the Accumulator (A).

The screenshot displays the Proteus 8051 simulator interface. The central window shows an assembly program designed to generate the last four digits of a mobile number using logical instructions (ANL, ORL, CLR). The program is as follows:

```

ORG 0000H
0000| CLR A           ; A = 00H
; --- Construct digit 7 (upper n
0001| ORL A, #70H      ; A = 0111 0000
; --- Construct digit 9 (lower n
0003| ORL A, #09H      ; A = 0111 1001
; --- Mask to ensure correctness
0005| ANL A, #7FH      ; Ensure valid
0007| HERE: SJMP HERE ; Stop executi
END
  
```

The left panel shows the 8051 register file and memory. The right panel shows the I/O devices, including a display-select decoder, keypad, and various sensors. The bottom panel shows the physical components of the simulator, including a keyboard, a display showing '8888', and various control buttons.

The screenshot displays the EdSim51DI - Version 2.1.39 software interface. The interface is divided into several sections:

- Top Left:** System configuration including System Clock (12.0 MHz), SBUF, I/O pins (R/O, W/O, TMO, TLO, R6, B, R7, ACC, R5, PSW, R4, IP, R3, IE, R2, PCON, R1, DPH, R0, DPL, SP), and a Data Memory table.
- Top Right:** Assembly code editor showing instructions like MOV, INC, SJMP, and their hex values.
- Bottom Left:** I/O devices including a 4x4 keypad, a 7-segment display, a DAC, and a motor control unit.
- Bottom Right:** A status bar showing the current instruction and the state of various hardware components like the display, keypad, and motor.