Date: 21/02/2021

Student's name	Anuj Shah
Roll Number	18104B0024
Name of Professor	Professor Mohit Gujar

Experiment number	2
Experiment title	Block transfer and Block exchange
Hardware requirement	-
Software requirement	Keil uVision5

Aim	To write a program to copy 10 numbers stored from 30H memory location to memory location starting from 40H
Theory	Our primary objective in these programs is to transfer the data stored in memory locations (0x30, 0x31, 0x32, ..., 0x3A) to the memory locations (0x40, 0x41, 0x42, ..., 0x4A). There are two ways to do this: • Block transfer, which carries out one-directional flow of data; that is, only from (0x30, 0x31, 0x32, ..., 0x3A) to (0x40, 0x41, 0x42, ..., 0x4A). • Block exchange, which carries out two-directional flow of data; that is, both from (0x30, 0x31, 0x32, ..., 0x3A) to (0x40, 0x41, 0x42, ..., 0x4A); and also from (0x40, 0x41, 0x42, ..., 0x4A) to (0x30, 0x31, 0x32, ..., 0x3A). Both the programs make heavy use of Pointers (Indirect addressing). Both the programs make use of a conditional jump statement called DJNZ (Decrement register, and Jump if that register is Not Zero). The block exchange program also maked use of the "XCH A,*register*" command, which is used to exchange the value of register A with the value contained in *register*. Source: https://www.win.tue.nl/~aeb/comp/8051/set8051.html#51xch
Algorithm / Flowchart	***Block transfer:*** 1. Because we want to transfer 10 numbers (10 in decimal = 0xA in hex) from address 0x30 to address 0x40, that is why we write the commands "MOV R3,#0AH", "MOV R0,#30H" and "MOV R1,#40H" 2. "MOV A,@R0" copies the contents of address 0x30 to register A 3. "MOV @R1,A" copies the contents of register A to address 0x40 4. "INC R0" changes 0x30 to 0x31 5. "INC R1" changes 0x40 to 0x41 6. "DJNZ R3,COUNTDOWN" changes 0xA to 0x9, and because 0x9 is not

zero, that is why our program jumps back to step 2.

7. Steps 2-6 are repeated continuously, until R3 finally becomes 0x0.

Thus, our program *copies* the contents from address 0x30 to address 0x40; from 0x31 to 0x41; and so on, until 0x3A to 0x4A.

Block exchange:

1. Because we want to transfer 10 numbers (10 in decimal = 0xA in hex) from address 0x30 to address 0x40, that is why we write the commands "MOV R3,#0AH", "MOV R0,#30H" and "MOV R1,#40H"
2. "MOV A,@R0" copies the contents of address 0x30 to register A
3. "XCH A,@R1" exchanges the contents of register A with the contents at address 0x40
4. "MOV @R0,A" copies the contents of register A to address 0x30
5. "INC R0" changes 0x30 to 0x31
6. "INC R1" changes 0x40 to 0x41
7. "DJNZ R3,COUNTDOWN" changes 0xA to 0x9, and because 0x9 is not zero, that is why our program jumps back to step 2.
8. Steps 2-6 are repeated continuously, until R3 finally becomes 0x0.

Thus, our program *exchanges* the contents of addresses 0x30 and 0x40; the contents of 0x31 and 0x41; the contents of 0x32 and 0x42; and so on, until 0x3A and 0x4A.

Source: https://www.keil.com/support/man/docs/is51/is51_mov.htm

Program

Block transfer:

block_transfer.asm

```
        ORG 0000H

        MOV R3,#0AH
        MOV R0,#30H
        MOV R1,#40H

COUNTDOWN:
        MOV A,@R0
        MOV @R1,A
        INC R0
        INC R1
        DJNZ R3,COUNTDOWN

        END
```

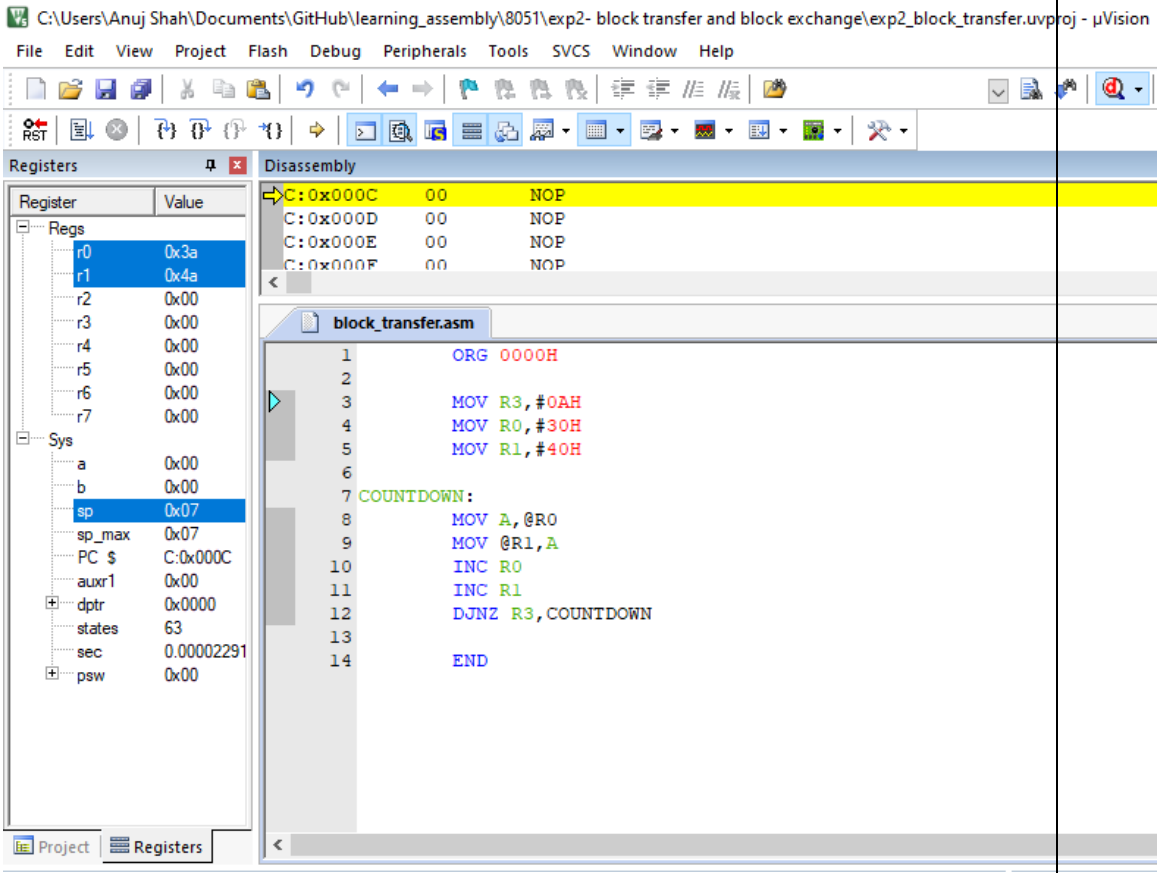
Block exchange:

block_exchange.asm

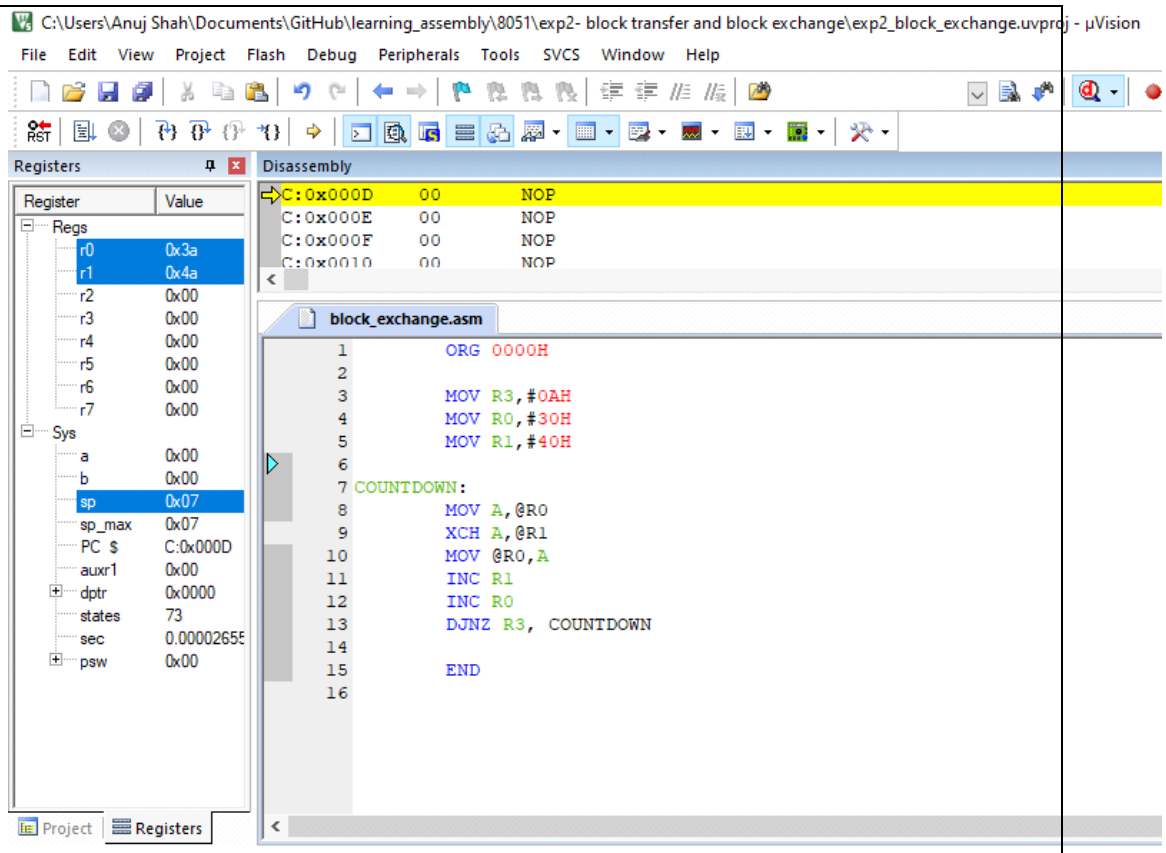
```
        ORG 0000H

        MOV R3,#0AH
        MOV R0,#30H
        MOV R1,#40H

COUNTDOWN:
        MOV A,@R0
        XCH A,@R1
        MOV @R0,A
        INC R1
        INC R0
        DJNZ R3, COUNTDOWN

        END
```

Results / Output

Block transfer:



Block exchange:



Conclusion	We learned some interesting features of 8051 assembly language in this experiment: • Indirect addressing (pointers) • Conditional jumps (eg. DJNZ) and count-downs • XCH, a slight variant of the MOV keyword

Faculty Sign Grade received