Experiment No : 11

Aim : Logical Operations

Theory :

Procedure :

1. Set Port 0, bits 1,3,5, and 7, to one; set the rest to zero.
2. Clear bit 3 of RAM location 22h without affecting any other bit.
3. Invert the data on the port 0 pins and write the data to port 1.
4. Swap the nibbles of R0 and R1 so that the low nibble of R0 swaps with the high nibble of R1 and the high nibble of R0 swaps with the low nibble of R1.
5. Complement the lower nibble of RAM location 2Ah.
6. Make the low nibble of R5 the complement of the high nibble of R6.
7. Make the high nibble of R5 the complement of the low nibble of R6.
8. Move bit 6 of R0 to bit 3 of port 3.
9. Move bit 4 of RAM location 30h to bit 2 of A.
10. XOR a number with whatever is in A so that the result is FFh.
11. Store the most significant nibble of A in both nibbles of register R5; for example, if A = B6h, then R5 = BBh.
12. Store the least significant nibble of A in both nibbles of RAM address 3Ch; for example, if A = 36h, then 3Ch = 66h.
13. Set the carry flag to one if the number in A is even; set the carry flag to zero if the number in A is odd.
14. Treat registers R0 and R1 as 16-bit registers, and rotate them one place to the left; bit 7 of R0 becomes bit 0 of R1, bit 7 of R1 becomes bit 0 of R0, and so on.
15. Repeat Problem 14 but rotate the registers one place to the right.
16. Rotate the DPTR one place to the left; bit 15 becomes bit 0.
17. Repeat problem 16 but rotate the DPTR one place to the right.
18. Shift register B one place to the left; bit 0 becomes a zero, bit 6 becomes bit 7, and so on. Bit 7 is lost.

19) Consider two numbers 35H and FFH. Perform the following logical operations on them:
a) AND
b) OR
c) XOR
d) Complement the two numbers
e) Compare the two numbers to check equality
20) Write a code to determine if register A contains the value 99H. If so, make R1=FFH; otherwise, make R1=0

21) Assume that P1 is an input port connected to a temperature sensor. Write a program to read the temperature and test it for the value 75. According to the test results, place the temperature value into the registers indicated by the following.

If T = 75 Then A = 75
If T<75 Then R1 = T
If T>75 Then R2 = T
22) Write a program to continuously monitor P1 for the value 31H. It should get out of the monitoring only if P1=63H
23) Assume internal RAM locations 40 H- 44H contain the daily temperature for five days as shown below. Search to see if any of the values equals 65. If value 65 does exist in the table give its location to R4; otherwise make R4=0.
40H=67
41H= 79
42H= 69
43H = 65 44H =62

Programs :

	label	**Mneumonic**	Operation
1)		MOV A,#0AAH MOV P0,A	Set A=10101010B Transfer to P0
2)		CLR 13H	Clear bit 3 of RAM location 22h
3)		MOV A,P0 CPL A MOV P1,A	Transfer P0 to A Complement A Transfer A to P1
4)		MOV A,R0 SWAP A MOV R2,A MOV A,R1 SWAP A MOV R0,A MOV A, R2 MOV R1,A	Get data at A=81h Exchange nibbles (A=18h) Store at R0 Get data at A=2Ah Exchange nibbles (A=A2h) Exchange between 2 register Transfer R2 contents into acc Store at R1
5)		MOV A,#0Fh XRL 2Ah,A	Get higher nibble Complement higher nibble
6)		MOV A,R6 XRL A,#0F0H SWAP A MOV R1,#05H XCHD A,@R1	Get data at A=25h Complement higher nibble Exchange nibble Use R1 as pointer Exchange nibbles

7)		MOV A,R5 SWAP A MOV R0,#06H XCHD A,@R0 XRL A,#0FH SWAP A MOV R5,A	Get data at A=25h Swap bits Use R0 as pointer Exchange nibbles Complement lower bit Swap bit Transfer acc contents into R5
8)		MOV 20H,R0 MOV C,06H MOV P3.3,C	Get data at bit address Copy carry flag Move to 3rd bit
9)		MOV 20H,30H MOV C,04H MOV ACC.2,C	Get data at bit address Copy carry flag Move to 3rd bit
10)		MOV R0,A XRL A,#3FH XRL A,R0	Get data at R0 (A XOR 3FH) Forming N (A XOR N) Result 3Fh
11)		ANL A,#0F0H MOV R5,A SWAP A ORL A,R5 MOV R5,A	Mask lower bit Transfer acc contents into R5 Exchange nibbles OR operation Load contents into R5
12)		ANL A, #0FH MOV 3CH, #00H ORL A, 3CH	Mask lower bit Send it to 3CH OR operation
		MOV 3CH,A SWAP A ORL A,3CH MOV 3CH,A	Transfer Acc contents to 3CH Xchange upper&lowernibbles OR operation Finally load content in 3CH
13)		MOV R0,A RRC A MOV A,R0	Get data at R0 Rotate the bits in acc Transfer R0 to acc
14)		MOV A, R0 RLC A MOV R0,	Move R0 contents into acc Rotate acc 1 bit Transfer acc contents to

		A MOV A, R1 RLC A MOV R1, A MOV A, R0 MOV 0E0H, C; MOV R0, A	R0 Move R1 contents into acc Rotate acc 1 bit Transfer acc contents to R1 Final adjustments Transfer carry to acc Transfer acc contents into R0

15)		MOV A, R0
		MOV C, ACC.0
		MOV A, R1
		RRC A
		MOV R1,
		A MOV A,
		R0 RRC A
		MOV R0,
		A
			Move R0 contents into acc
			Set carry flag
			Move R1 contents into acc
			Rotate acc 1 bit
			Move acc contents into R1
			Final adjustments
			Rotate R1 1 bit
			Transfer acc contents to R0

16)	HERE	CLR C MOV A,DPL RLC A MOV A,DPH RLC A MOV DPH,A MOV A,DPL RLC A MOV DPL,A HERE:SJMP HERE	Set carry flag to 0 Load low bytes into acc Rotate bits in A Load high bytes into acc Rotate bits in A Transfer acc data to DPH Load low bytes to acc Rotate bits in A Transfer acc data to DPL
17)	HERE	CLR C MOV A,DPL RRC A MOV A,DPH RRC A MOV DPH,A MOV A,DPL RRC A MOV DPL,A HERE:SJMP HERE	Set carry flag to 0 Load low bytes into acc Rotate bits in A Load high bytes into acc Rotate bits in A Transfer acc data to DPH Load low bytes to acc Rotate bits in A Transfer acc data to DPL
18)		MOV A,B RL A CLR	Clear carry flag Tranfer bits of B to A Set bit 0 to 0 Tranfer bits back to B

		ACC.0 MOV B,A	

19)		MOV A,#35H
		MOV
		R0,#0FFH
		ANL A,R0
		MOV 20H,A
		
		MOV
		A,#35H
		ORL A,R0
		MOV 21H,A
		
		MOV
		A,#35H
		XRL A,R0
		MOV
		22H,A
		
		MOV
		A,#35H
		CPL A
		MOV 23H,A
		MOV
		A,#0FFH
		CPL A
		MOV 24H,A
		
		MOV A,#35H
		CJNE A,#0FFH,25H
			Set A=35H
			Set R0 = 0FFH
			And
			Operation
			Store at 20H
			
			Set A=35
			OR Operation
			Store at 21H
			
			Set A=35
			XOR Operation
			Store at 22H
			
			Set A=35
			Complement
			A Store at 23H
			Set A = 0FFH
			Complement
			A Store at 24H
			
			Set A=35
			Compare equality

20)	NEXT	MOV R1,#0 CJNE A,#99H,NEXT MOV R1,#0FFH	Clear R1 If A≠99, then jump otherwise R1=0FFH
21)	OVER NEXT EXIT	MOV P1,#0FFH MOV A,P1 CJNE A,#75,OVER SJMP EXIT JNC NEXT MOV R1,A SJMP EXIT MOV R2,A	Make P1 an input port Read P1 port Jump if A≠75 A=75 A<75, save A R1 A>75, save A in R2
22)	HERE	MOV P1,#0FFH MOV A,P1 CJNE A,63,HERE	Make P1 an input Get P1 Keep monitoring

23)	
	
	BACK
	
	NEXT
	
	EXIT
		MOV R4,#0
		MOV
		R0,#40H
		MOV R2,#05
		MOV A,#65
		CJNE A,@R0,NEXT
		MOV R4,R0
		SJMP
		EXIT INC
		R0
		DJNZ R2,BACK
			R4=0
			Load
			pointer
			Load
			counter
			A=65, value searched for
			compare ram data with
			65 If 65, save address
			Exit
			Increment pointer
			Keep check until count=0

Conclusion : Logical operations were studied and the code were implemented successfully.

Deepraj Bhosale
181105016