Συστήματα Μικροϋπολογιστών

3η Ομάδα Ασκήσεων

Ομάδα:

Κατσιάνης Κωνσταντίνος - 03120183

Κουρής Γεώργιος - 03120116

6ο Εξάμηνο - Μάιος 2023

Άσκηση 1:

Το πρόγραμμα για το ερώτημα είναι το εξής:

```
       MVI A,10H     ;move value 10H to the accumulator
       STA 0B00H     ;set up right-most digit in display (1st segment)
       STA 0B01H     ;set up seconf right most digit in display (2nd segment
       STA 0B02H     ;only 0B00H and 0B01H are used
       STA 0B03H     ;the others are set up to light them out
       STA 0B04H     ;if they're not they will display zeros
       STA 0B05H
       MVI A,0DH     ;move value 0DH to the accumulator
       SIM           ;set interupt mask
       EI            ;enable interupts
BEGIN:
       JMP BEGIN     ;jumpt to BEGIN
INTR_ROUTINE:
       POP H         ;pop out 2-Bytes from the top of the stack through H
       EI            ;enable interrupts
       MVI A,00H     ;move value 0 to the accumulator
       STA 3000H     ;turn on the LEDs (all will light up)
       MVI H,06H     ;count down, starting from 60
       MOV A,H       ;move value of H to the accumulator
       DCR A         ;decrease accumulator by one
       STA 0B01H     ;save value to the 2nd segment (the tens)
ONES:
       MVI A,09H     ;move value 09H to the accumulator (9 seconds)
USE_DISPLAY:
       STA 0B00H     ;save value to the 1st segment
       CALL DISPLAY  ;show display
       DCR A         ;decrease accumulator by one
       CPI 00H       ;compare accumulator to 0
       JNZ USE_DISPLAY ;if A != 0 jump to LIGHTS_ON (9 sec have yet to pass)
       CALL DISPLAY_ZERO    ;call function to display 0
       DCR H         ;decrease H by one
       JZ EXIT       ;if H == 0 jump to EXIT
       MOV A,H       ;move value of H to the accumulator
       DCR A         ;decrease accumulator by one
       STA 0B01H     ;save value to the second segment
       JMP ONES      ;jump to seconds
EXIT:
       MVI A,FFH     ;move value FFH to the accumulator
       STA 3000H     ;turn on the LEDs (because of value of A they're off)
       JMP BEGIN     ;jump to BEGIN
DISPLAY:
       LXI B,0064H   ;load adress 0064H to B (100ms)
       LXI D,0B00H   ;load adress 0B00H to D
       PUSH PSW      ;push the value of program status word in memory stack
```

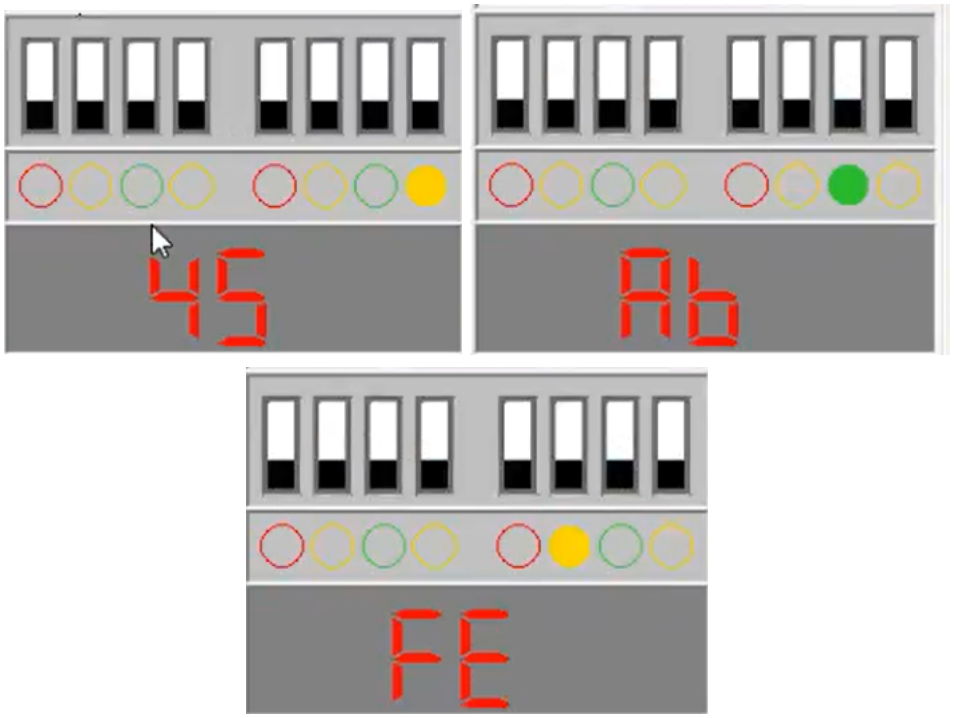
```
        PUSH H          ;push value of H in memory stack
        PUSH D          ;push value of D in memory stack
        PUSH B          ;push value of B in memory stack
        CALL STDM       ;call function STDM
        MVI A,28H       ;move value 28H (40) to the accumulator
                        ;normally, it should have been 0AH (10) so that
                        ;we had 10*100ms. Unfortunately, the emulator
                        ;runs too fast so we had to quadruple the delay
                        ;to have one second
1SEC:
        CALL DCD        ;call function DCD
        CALL DELB       ;call function DELB for delay
        DCR A           ;decrease accumulator by one
        CPI 00H         ;compare accumulator to 0
        JNZ 1SEC        ;if A != 0 jump to 1SEC
        POP B           ;pop out 2-Bytes from the top of the stack through B
        POP D           ;pop out 2-Bytes from the top of the stack through D
        POP H           ;pop out 2-Bytes from the top of the stack through H
        POP PSW         ;pop out 2-Bytes from the top of the stack through PSW
        RET             ;return from the subroutine
DISPLAY_ZERO:
        MVI A,00H       ;move value 0 to the accumulator
        STA 0B00H       ;save value to the 1st segment
        CALL DISPLAY    ;call function DISPLAY
        CALL DELB       ;call function DELB for delay
        RET             ;return from the subroutine

END
```

Άσκηση 2:

Το πρόγραμμα για το ερώτημα είναι το εξής:

```
        IN 10H          ;make content op the port available to the accumulator
        MVI A,0DH       ;move value 0DH to the accumulator
        SIM             ;set interupt mask (RST 6.5)
        EI              ;enable interupt
        MVI B,06H       ;move value 06H to B
        LXI H,0A00H     ;load address 0A00H to register H

J1:     MVI M,10H       ;move value 10H to M
        INX H           ;increment value of register H
        DCR B           ;decrease value of B by one
        JNZ J1          ;jump to J1 if B != 0
        MVI D,70H       ;move value 70H (=112) to D (=K1)
        MVI E,BBH       ;move value BBH (=187) to E (=K2)
        PUSH D          ;store value of D to the stack
        LXI D,0A00H     ;load address 0A00H to register H
        CALL STDM       ;call subroutine STDM
        POP D           ;pop the value from the stack to D
        CALL DCD        ;call subroutine DCD

J2:     JMP J2          ;jump to J2 (until we have input)

INTR_ROUTINE:           ;interupt routine
        CALL KIND       ;call subroutine kind
        STA 0A03H       ;store value of accumulator to address 0A05H
        RLC             ;rotate accumulator left
        RLC             ; -//-
        RLC             ; -//-
        RLC             ; -//-
        MOV B,A         ;move value of accumulator to B
        CALL KIND       ;call subroutine kind
        STA 0A02H       ;store value of accumulator to address 0A04H
        ADD B           ;add value of B to the accumulator

        CMP D           ;compare D with accumulator
        JZ CASE1        ;jump to CASE1 if D = A
        JC CASE1        ;jump to CASE1 if D > A
        JMP CASE2A      ;else jump to CASE2A (if D < A)

CASE1:                  ;case number is in [00H, K1]
        MVI A,01H       ;move value 01H to accumulator (lsb LED)
        JMP DISPLAY     ;jump to DISPLAY (display it)
```

```
CASE2A:
       CMP E          ;compare E with accumulator
       JZ CASE2B      ;jump to CASE1 if E = A
       JC CASE2B      ;jump to CASE1 if E > A
       JMP CASE3      ;else jump to CASE2A (if E < A)

CASE2B:               ;case number is in (K1, K2]
       MVI A,02H      ;move value 02H to accumulator (second lsb LED)
       JMP DISPLAY    ;jump to DISPLAY (display it)

CASE3:                ;case number is in (K2, FFH]
       MVI A,04H      ;move value 04H to accumulator (third lsb LED)

DISPLAY:
       CMA            ;find complement of A (LEDs on at 0)
       STA 3000H      ;light the LEDs

       PUSH D         ;store value of D to the stack
       LXI D,0A00H    ;load address 0A00H to register D
       CALL STDM      ;call subroutine STDM
       POP D          ;pop the value from the stack to D
       EI             ;enable interupt

LOOP_DCD:
       CALL DCD       ;call subroutine DCD
       JMP LOOP_DCD   ;jump to LOOP_DCD

END
```



Άσκηση 3:

(α)

Το πρόγραμμα για το ερώτημα (α) είναι το εξής:

```
SWAP Nible MACRO Q
        PUSH PSW

        MOV A,Q
        RLC
        RLC
        RLC
        RLC

        MOV Q,A
        MOV A,M

        RRC
        RRC
        RRC
        RRC
        MOV M,A

        POP PSW
ENDM
```

(β)

Το πρόγραμμα για το ερώτημα (β) είναι το εξής:

```
FILL MACRO RP, X, K
        PUSH PSW
        PUSH H
        MOV H,R
        MOV L,P
        LOOP1:
        MVI M,K
        INX H
        DCR X
        JNZ LOOP1
        POP H
        POP PSW
ENDM
```

(γ)

Το πρόγραμμα για το ερώτημα (γ) είναι το εξής:

```
RHLR MACRO
        PUSH PSW
        PUSH B

        MVI A,1
        CPI 00H
        JZ END
        MVI B,1
BEGIN:
        MOV A,H
        RAR
        MOV H,A
        MOV A,L
        RAR
        MOV L,A
        DCR B
        JNZ BEGIN
END:
        POP B
        POP PSW
ENDM
```

Άσκηση 4:

Καθώς εκτελούμε την εντολή CALL 0900H έχουμε μία διακοπή τύπου hardware interrupt, συγκεκριμένα RST 5.5 (maskable interrupt στο pin 9). Αρχικά, θα ολοκληρωθεί η τρέχουσα εντολή και θα αποθηκευτεί στον σωρό η διεύθυνση της εντολής που θα είχαμε μετά την CALL 0900H. Ο δείκτης του σωρού θα ανέβει 2 θέσεις πάνω και ο PC θα πάρει την τιμή 0900H. Στη συνέχεια η τιμή του PC αποθηκεύεται στον σωρό (0900H) και θα έχουμε εκτέλεση της ρουτίνας εξυπηρέτησης του hardware interrupt RST 5.5. Ο δείκτης του σωρού θα ανέβει άλλες 2 θέσεις, δηλαδή. Το PC θα πάρει επίσης την address 002CH.

Αφού τελειώσει η ρουτίνα εξυπηρέτησης RST 5.5, θα πάρουμε από τον σωρό την προηγούμενη τιμή του PC (0900H), η οποία θα αποθηκευτεί σε αυτόν και προφανώς ο δείκτης σωρού κατεβαίνει 2 θέσεις. Μετά την ολοκλήρωση της εκτέλεσης της ρουτίνας στη διεύθυνση 0900H, θα πάρουμε από τον σωρό και τη διεύθυνση της εντολής μετά την CALL 0900H, που είναι η 0903H. Η τιμή θα αποθηκευτεί στο PC και ο δείκτης του σωρού θα κατέβει άλλες 2 θέσεις.

Εντολές	1		2		3		4		5	
PC	PC	0840H	PC	0900H	PC	002CH	PC	0900H	PC	0903H
SP	-	-	SP	09H	SP	09H	SP	09H	-	-
			SP+1	03H	SP+1	00H	SP+1	03H		
					SP+2	09H				
					SP+3	03H				

Άσκηση 5:

(α)

```
        MVI A,0DH
        SIM
        LXI H,0000H
        MVI C,40H
        EI
JUMP_1:
        MOV A,C
        CPI 00H
        JNZ JUMP_1
        DI
        DAD H
        DAD H
        DAD H
        HLT
0034:
        JMP RST6.5
RST6.5:
        PUSH PSW
        MOV A,C
        ANI 01H
        JPO MSB4
        IN PORT_IN
        ANI 0FH
        MOV B,A
        JMP LSB4
MSB4:
        IN PORT_IN
        ANI 0FH
        RLC
        RLC
        RLC
        RLC
        ORA B
        MVI D,00H
        MOV E,A
        DAD D
        DCR C
LSB4:
        POP PSW
        EI
        RET
END
```

(β)

```
        LXI H,0000H
        MVI C,40H
BEGIN:
        IN PORT_IN
        ANI 80H
        JP BEGIN
        MOV A,C
        ANI 01H
        JPO MSB4
        IN PORT_IN
        ANI 0FH
        MOV B,A
        JMP LSB4
MSB4:
        IN PORT_IN
        ANI 0FH
        RLC
        RLC
        RLC
        RLC
        ORA B
        MVI D,00H
        MOV E,A
        DAD D
LSB4:
        DCR C
        JZ JUMP_1
CHECK:
        IN PORT_IN
        ANI 80H
        JM CHECK
        JMP BEGIN
JUMP_1:
        DAD H
        DAD H
        DAD H
        HLT
END
```