Εργαστήριο Μικροϋπολογιστών – 1η Σειρά Ασκήσεων

Κυριακόπουλος Γιώργος – el18153
Τζελέπης Σεραφείμ – el18849

1η Άσκηση:

```
START:

    LXI B,03E8H      ; move 1000 to BC for 1 second delay
    LDA 2000H        ; read input
    ANI 0FH          ; keep 4 LSBs
    INR A            ; increase A to store x+1
    MOV E,A          ; move x+1 to E
RESTART:
    MVI A,00H        ; start counting upwards from 0
UP:
    CALL MSBCHECK    ; call MSB check routine
    CMA              ; complement A because of LED's negative logic
    STA 3000H        ; output A'
    CALL DELB        ; call the 1 second delay routine
    CMA              ; restore A
    INR A            ; increase A
    CMP E            ; compare A with x
    JNZ UP           ; if A is not equal(less) than x keep counting
    MOV A,E          ; keep 4 LSBs
DOWN:
    CALL MSBCHECK    ; call MSB check routine
    DCR A            ; decrement A to count down
    CMA              ; complement A because of LED's negative logic
    STA 3000H        ; output A'
    CALL DELB        ; call the 1 second delay routine
    CMA              ; restore A
    CPI 00H          ; compare A with 0
    JNZ DOWN         ; if A is not equal(greater) than 0 keep counting
    LDA 2000H        ; read input for new x
    ANI 0FH          ; keep 4 LSBs
    INR A            ; increase A
    MOV E,A          ; move new x+1 to E
    JMP RESTART      ; restart the counting
```

```
MSBCHECK:
    PUSH PSW          ; push A and flags
LOOPER:
    LDA 2000H         ; read input
    RAL               ; rotate left to check MSB
    JNC LOOPER        ; if MSB is 0 repeat
    POP PSW           ; pop A and flags
    RET
END
```

2η Άσκηση:

```
    IN 10H
START:                  ; max value = 16*F + F = FF
    CALL KIND           ; read x from the keyboard
    CPI 00H             ; if x < 0 read again
    JC START
    CPI 10H             ; if x > F read again
    JNC START
    RLC
    RLC
    RLC
    RLC
    MOV B,A             ; move 16*x to B
    CALL KIND           ; read y from the keyboard
    CPI 00H             ; if y < 0 read again
    JC START
    CPI 10H             ; if y > F read again
    JNC START
    ADD B               ; A = 16*x + y
    LXI H,0A02H         ; initialize address for the displays
    MVI C,00H           ; initialize c to count hundreds
DIV1:
    INR C               ; increment the hundreds counter
    SUI 64H             ; subtract 100 from A
    JNC DIV1            ; if A isn't negative repeat
    DCR C               ; decrement C to fix the number of the hundreds
    MOV M,C             ; store hundreds to 0A02H
    ADI 64H             ; add 100 back to restore A to get tens
    MVI C,00H           ; reset c to count tens
DIV2:
    INR C               ; increment the tens counter
    SUI 0AH             ; subtract 10 from A
    JNC DIV2            ; if A isn't negative repeat
    DCR C               ; decrement C to fix the number of the hundreds
    DCX H               ; decrement memory pointer
    MOV M,C             ; store tens to 0A01H
    ADI 0AH             ; add 10 back to restore A to get ones
    DCX H               ; decrement memory pointer
    MOV M,A             ; store ones in 0A00H

    LXI H,0A05H         ; initialize address for the empty displays
    MVI M,10H           ; load the space symbol
```

```
        DCX H               ; decrement memory pointer
        MVI M,10H           ; load the space symbol
        DCX H               ; decrement memory pointer
        MVI M,10H           ; load the space symbol

        LXI D,0A00H         ; load D with the memory address
        CALL STDM           ; call the routine to prepare the display
        CALL DCD            ; output to the 7-segment displays

        JMP START           ; repeat
    END
```

3η Άσκηση:

```
START:
    MVI A,01H        ; initiliaze A to LSB LED
    MVI D,01H        ; initiliaze D to LSB LED
    LXI B,01F4H      ; move 500 to BC for 0.5 second delay
    CMA              ; complement A because of LED's negative logic
    STA 3000H        ; output A'
    CMA              ; restore A
    CALL DELB        ; call the 0.5 second delay routine
    CALL MSBCHECK    ; call MSB check routine
WAIT:
    CALL LSBCHECK    ; call LSB check routine
    MOV A,E          ; move LSB to A
    CPI 00H          ; compare with 0
    JZ WAIT          ; loop
TOLEFT:
    MOV H,E          ; update H with last LSB
    MOV A,D          ; restore A
    CPI 80H          ; compare with far-left position
    JZ SWITCHR       ; if the train is far left change direction
    RLC              ; move the train left
    CMA              ; complement A because of LED's negative logic
    STA 3000H        ; output A'
    CMA              ; restore A
    CALL DELB        ; call the 0.5 second delay routine
    CALL MSBCHECK    ; call MSB check routine
    CALL LSBCHECK    ; call LSB check routine
    MOV D,A          ; backup A
    MOV A,E          ; move LSB to A
    CPI 00H          ; compare with 0
    JNZ TOLEFT       ; if it's equal to 0 change direction
    CMP H            ; compare with last LSB
    JZ TOLEFT        ; if it's same as last LSB don't change direction
TORIGHT:
    MOV H,E          ; update H with last LSB
    MOV A,D          ; restore A
    CPI 01H          ; compare with far-right position
    JZ SWITCHL       ; if the train is far right change direction
    RRC              ; move the train right
    CMA              ; complement A because of LED's negative logic
    STA 3000H        ; output A'
    CMA              ; restore A
```

```asm
        CALL DELB        ; call the 0.5 second delay routine
        CALL MSBCHECK    ; call MSB check routine
        CALL LSBCHECK    ; call LSB check routine
        MOV D,A          ; backup A
        MOV A,E          ; move LSB to A
        CPI 00H          ; compare with 0
        JNZ TORIGHT      ; if it's equal to 0 change direction
        CMP H            ; compare with last LSB
        JZ TORIGHT       ; if it's same as last LSB don't change direction
        JMP TOLEFT       ; change direction

MSBCHECK:
        PUSH PSW         ; push A and flags
LOOPER:
        LDA 2000H        ; read input
        RAL              ; rotate left to check MSB
        JNC LOOPER       ; if MSB is 0 repeat
        POP PSW          ; pop A and flags
        RET

LSBCHECK:
        PUSH PSW         ; push A and the flags
        LDA 2000H        ; read input
        ANI 01H          ; isolate the lsb of A
        MOV E,A          ; move LSB to E
        POP PSW          ; pop A and the flags
        RET

SWITCHR:
        CALL DELB        ; call the 0.5 second delay routine
        JMP TORIGHT      ; change direction

SWITCHL:
        CALL DELB        ; call the 0.5 second delay routine
        JMP TOLEFT       ; change direction
END
```