Μικρο υπολογιστές
σειρά ασκήσεων 3

Ζαρίφης Νικόλαος 0312178
Ιωάννης Ορούτζογλου 03112124

Ασκήσεις 1-4:

1:

```
LOOP1:
LDA 2000H
MVI B,00H ;zero to B
MVI D,08H ;if A is zero (8 loops)
LOOP2:
DCR D
JZ LOOP1Z
RRC ; rotate to find the first 1
JC LEDON
MOV C,A
MOV A,B
RLC ;rotate because you wont light this led
ORI 01H ; dont light first led
MOV B,A
MOV A,C
JMP LOOP2
LOOP1Z:
MVI A,00H ;if a is zero
CMA
STA 3000H
JMP LOOP1
LEDON:
MOV A,B ;put B to a to See which led to light
STA 3000H
JMP LOOP1

RST 1
END
```

2:

```
IN 10H
LOOP1:
CALL KIND ; read keyboard
MOV B,A ;save
CPI 00H
JZ LIGHT ; if it is zero dont light anything
MVI A,00H
ORI 01H ; put the first 1 to find which led will light
LOOP2:
DCR B
JZ LIGHT
RLC ;while not zero push left the bit
JMP LOOP2

LIGHT:
CMA
STA 3000H
```

```
JMP LOOP1


RST 1
END
3:
IN 10H
MVI B,00H
LINE0:
MVI A,FEH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,86H ;KODIKOS  INSTR STEP
JZ LED
CPI 05H
MVI C,85H ; code FETCH PC
JZ LED

LINE1:
MVI A,FDH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,84H ;KODIKOS  RUN
JZ LED
CPI 05H
MVI C,80H ; code FETCH register
JZ LED
CPI 03H
MVI C,82H ; fetch address
JZ LED

LINE2:
MVI A,FBH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,00H ;KODIKOS  0
JZ LED
CPI 05H
MVI C,83H ; code store/incr
JZ LED
CPI 03H
MVI C,86H ; INCR
JZ LED

LINE3:
MVI A,F7H ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,01H ;KODIKOS 1
JZ LED
CPI 05H
MVI C,02H ; code 2
JZ LED
CPI 03H
```

```
MVI C,03H ; code 3
JZ LED

LINE4:
MVI A,EFH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,04H ;KODIKOS 4
JZ LED
CPI 05H
MVI C,05H ; code 5
JZ LED
CPI 03H
MVI C,06H ; code 6
JZ LED

LINE5:
MVI A,DFH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,07H ;KODIKOS 7
JZ LED
CPI 05H
MVI C,08H ; code 8
JZ LED
CPI 03H
MVI C,09H ; code 9
JZ LED

LINE6:
MVI A,BFH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,0AH ;KODIKOS A
JZ LED
CPI 05H
MVI C,0BH ; code B
JZ LED
CPI 03H
MVI C,0CH ; code C
JZ LED

LINE7:
MVI A,7FH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
MVI C,0DH ;KODIKOS D
JZ LED
CPI 05H
MVI C,0EH ; code E
JZ LED
CPI 03H
MVI C,0FH ; code F
JZ LED
```

```
MVI C,00H

LED:
LXI H,0B04H
MOV A,C ;fere minima ston A
ANI 0FH ;first 4 bytes to store
MOV M,A
MOV A,C
ANI F0H
INX H
MOV M,A
LXI D,0B00H
CALL STDM
CALL DCD
MOV B,C
JMP LINE0
RST 1
END

4:
IN 10H
MVI B,00H

LINE0:
MVI A,FEH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H ;KODIKOS  INSTR STEP
JZ LEDINST
CPI 05H ; code FETCH PC
JZ LEDPC

LINE1:
MVI A,FDH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H
;KODIKOS  RUN
JZ LEDRUN
CPI 05H
; code FETCH register
JZ LEDREG
CPI 03H ; fetch address
JZ LEDADR

LINE2:
MVI A,FBH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H ;KODIKOS  0
JZ LED0
CPI 05H ; code store/incr
JZ LEDINCR
CPI 03H  ; INCR
JZ LEDINST

LINE3:
MVI A,F7H ;epilogi grammis
STA 2800H
```

```
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H ;KODIKOS 1
JZ LED1
CPI 05H ; code 2
JZ LED2
CPI 03H ; code 3
JZ LED3

LINE4:
MVI A,EFH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H  ;KODIKOS 4
JZ LED4
CPI 05H ; code 5
JZ LED5
CPI 03H ; code 6
JZ LED6

LINE5:
MVI A,DFH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H  ;KODIKOS 7
JZ LED7
CPI 05H  ; code 8
JZ LED8
CPI 03H  ; code 9
JZ LED9

LINE6:
MVI A,BFH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H ;KODIKOS A
JZ LEDA
CPI 05H ; code B
JZ LEDB
CPI 03H ; code C
JZ LEDC

LINE7:
MVI A,7FH ;epilogi grammis
STA 2800H
LDA 1800H ;diabasma stilis
ANI 07H ;mask number
CPI 06H  ;KODIKOS D
JZ LEDD
CPI 05H ; code E
JZ LEDE
CPI 03H ; code F
JZ LEDF



LED0:
MVI A,30H
STA 2800H ; second digit
```

```
MVI A,C0H ; 0 code
STA 3800H ; this code is the same for all 0-F , i will just cp it

MVI A,10H
STA 2800H ;first digit
MVI A,C0H ; zero
STA 3800H
JMP RR ;reset digits


LED1:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,F9H ; one
STA 3800H
JMP RR ;reset digits

LED2:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,A4H ; two
STA 3800H
JMP RR ;reset digits

LED3:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,B0H ; three
STA 3800H
JMP RR ;reset digits

LED4:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,99H ; zero
STA 3800H
JMP RR ;reset digits

LED5:
MVI A,30H
STA 2800H
MVI A,C0H
```

```
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,92H ; five
STA 3800H
JMP RR ;reset digits

LED6:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,82H ; six
STA 3800H
JMP RR ;reset digits

LED7:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,F8H ; seven
STA 3800H
JMP RR ;reset digits

LED8:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,80H ; eight
STA 3800H
JMP RR ;reset digits

LED9:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,98H ; nine
STA 3800H
JMP RR ;reset digits

LEDA:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H
```

```
MVI A,10H
STA 2800H ;first digit
MVI A,88H ; A
STA 3800H
JMP RR ;reset digits

LEDB:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,83H ; b
STA 3800H
JMP RR ;reset digits

LEDC:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,A3H ; c
STA 3800H
JMP RR ;reset digits

LEDD:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,A1H ; d
STA 3800H
JMP RR ;reset digits

LEDE:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,86H ; e
STA 3800H
JMP RR ;reset digits

LEDF:
MVI A,30H
STA 2800H
MVI A,C0H
STA 3800H

MVI A,10H
STA 2800H ;first digit
```

```
MVI A,8EH ; f
STA 3800H
JMP RR ;reset digits


LEDINST:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,82H ; 6
STA 3800H
JMP RR ;reset digits

LEDPC:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,92H ; 5
STA 3800H
JMP RR ;reset digits

LEDRUN:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,99H ; 4
STA 3800H
JMP RR ;reset digits

LEDREG:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,C0H ; 0
STA 3800H
JMP RR ;reset digits

LEDADR:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,A4H ; 2
```

```
STA 3800H
JMP RR ;reset digits

LEDINCR:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,B0H ; 3
STA 3800H
JMP RR ;reset digits

LEDDECR:
MVI A,30H
STA 2800H
MVI A,80H ;eight
STA 3800H

MVI A,10H
STA 2800H ;first digit
MVI A,F9H ; 1
STA 3800H
JMP RR ;reset digits

RR:
MVI A,FFH
STA 3800H
JMP LINE0

RST 1
END
```

5:Πρώτος μΥ. (που στέλνει)

```
MVI B,00H ;counter for bytes

READY:
MVI A,FFH; ; we want to sent 1 so A= 11XXXXXX
SIM ; sent bit
RIM  ;receive bit
RLC ; check if is 1
JNC READY

YES:
; we sent nothing so , the other feveice will receive the 0 bit
LDAX H
OUT DATA
INX H
ICR B
JC ENDA ; if B=256 ->C=1 B=0
WAIT:
RIM ; wait for 0 bit
RLC
JNC READY ;byte was sent ,go for the next
JMP WAIT ; wait to complete the transfer
ENDA:
END
```

2ος μΥ

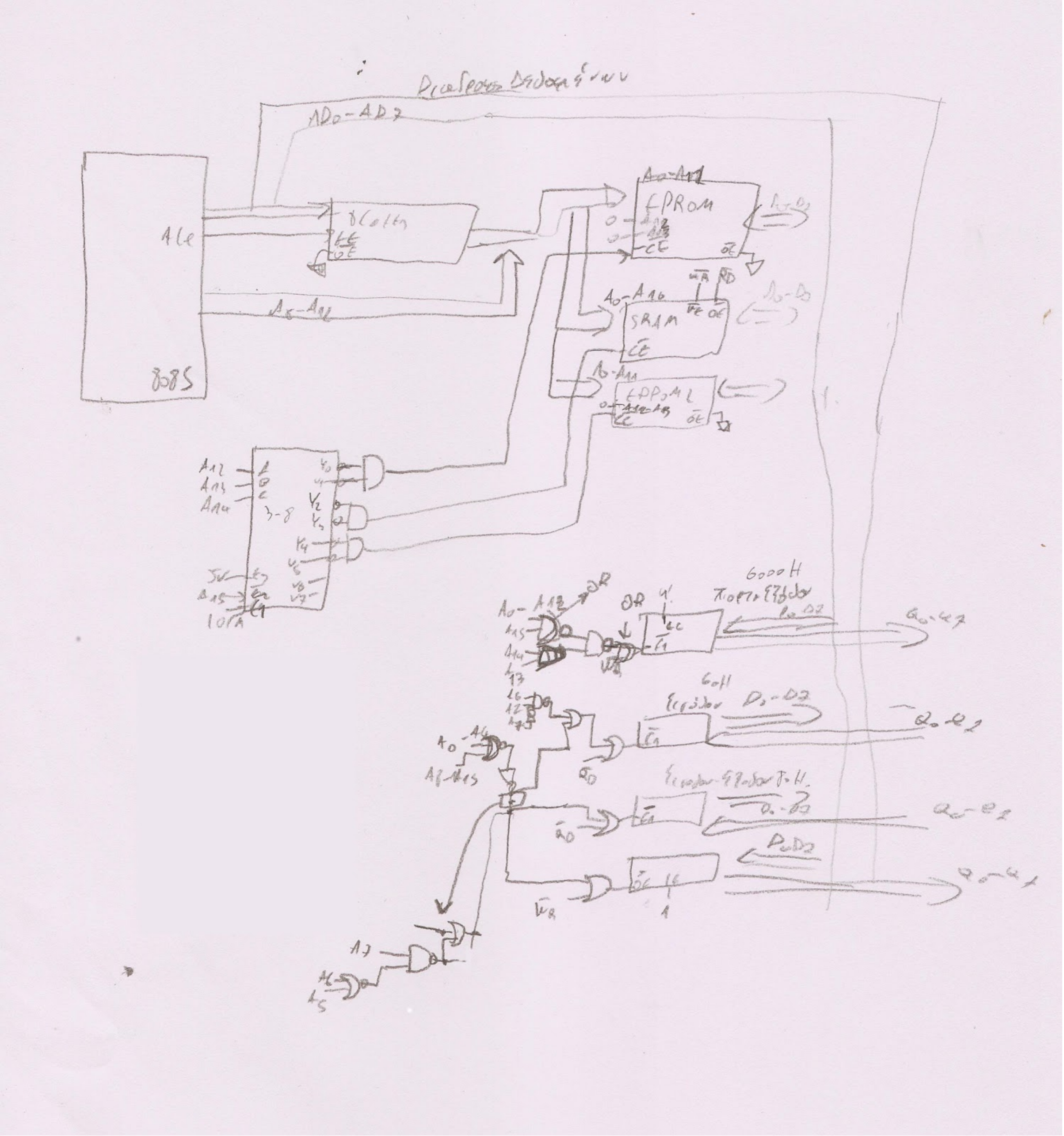
```
MVI B,00H
READY:
RIM  ;receive bit
RLC ; check if is 1
JNC READY

YES:
MVI A,FFH ; sent 1
SIM
RIM
RLC
JZ REC ; receive data
JMP YES
REC:
IN DATA ; get data
STAX H ; save data to memory
INX H ; next cell
ICR B
JC ENDA
EI ; next data
JMP READY
ENDA:
END
```

Για το χρόνο, αν υποθέσουμε

6:

Type	Start	End
EPROM1	0**000** 0000 0000 0000	0**001** 1111 1111 1111
RAM	0**010** 0000 0000 0000	0**011** 1111 1111 1111
EPROM2	0**100** 0000 0000 0000	0**101** 1111 1111 1111
έξοδος	0**110** 0000 0000 0000	0**110** 0000 0000 0000
είσοδος	**0000** 0000 **011**0 0000	**0000** 0000 **011**0 0000
είσοδος εξόδου	**0000** 0000 **100**0 0000	**0000** 0000 **100**0 0000

Έχοντας το memory map της εκφώνησης , χρησιμοποιούμε 2 EPROM αν κι θα μπορούσαμε με μια απλά χρησιμοποιώντας ένα bit που θα δείχνει πιο μέρος χρησιμοποιούμε συγκεκριμένα το bit-13 της μνήμης. Κι έχουμε το ακόλουθο σχήμα, έχουμε χρησιμοποιήση αναγώγες από άλγευρα boole δλδ A-NOT B-NOT NOR -> AND κλπ. Επίσεις προτίμησα να χρησιμοποιήσω πύλες για είσοδο/έξοδο αφού είναι πιο φθηνές κι δεν χρειαζόμαστε παραπάνω από τρεις εξόδους όπως πριν.



7:

```
SWAP MACRO Q,R
MOV A,B
MOV C,B
MOV B,A ;simple swap using A as temporary reg
ENDM

FILL MACRO ADDR,L,K
LXI H,ADDR
MVI A,L
CPI 00H ; if L is zero
JZ ENDA
```

```
LOOP1:
MVI M,K ; store byte
INR M ; next adress
DCR A
JZ ENDA ;did we ended fill?
JMP LOOP1

ENDA:
ENDM

RHLL MACRO n
MVI A,n ;
CPI 00H ;
JZ ENDR ;if it is zero
MVI B,n ;counter

LOOP2:
MOV A,L ; take from L
RAL ; rotate with carry
MOV L,A ;store
MOV A,H ;again with H
RAL ;
DCR B ;decrease counter
JZ ENDR
JMP LOOP2
ENDR:
ENDM
```