**Logo

Description automatically generatedΕθνικό Μετσόβιο Πολυτεχνείο**

**Σχολή: Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών**

**Συστήματα Μικροϋπολογιστών (6ο εξάμηνο)**

**3η Ομάδα Ασκήσεων**

**Δημήτριος Καλαθάς - el18016**

**Δημήτριος Καλέμης - el18152**

Ασκήσεις Προσομοίωσης

Άσκηση 1

|  |
| --- |
| Πρόγραμμα 1 σε Assembly |
| MVI C,06H ;COUNTER  IN 10H  MAIN:  LXI H,0A00H  CLEAR:  MVI M,10H ;CLEAR THE SCREEN WITH SPACE  INX H  DCR C  JNZ CLEAR    MVI A,0DH ;ENABLE 6,5 RST  SIM  EI    INF:  LXI H,0A02H  MVI M,00H  INX H  MVI M,00H  WAIT:  CALL DISP ;SHOW 00 IN SCREEN    JMP WAIT ;WAIT UNTIL INTERAPT    INTR\_ROUTINE: ;INTERAPT ROUTINE START  MVI E,3CH ;SET TIMER(E) = 60  LXI B,0064H ;(BC) = 100, 100 ms  EI  INIT:  MVI D,0AH ;(D) = 10, (D)\*(BC) = 1 sec  CALL NEXT\_SEC ;COUNT SEC    MVI A,00H  STA 3000H ;OPEN LEDS LED  L1: CALL DISP ;SHOW IN SCREEN  CALL DELB ;DELAY 100 ms  DCR D  JNZ L1 ;Total delay = 1 sec    L2: CALL DISP  DCR D ;EXTRA DELAY  JNZ L2    DCR E ;60..0SEC  JNZ INIT  MVI A,FFH  STA 3000H  JMP INF  NEXT\_SEC:  PUSH PSW  PUSH B  PUSH H    MVI B,FFH  MOV A,E  L3:  INR B  SUI 0AH  JNC L3 ;μέχρι να γίνει αρνητικό  ADI 0AH ;προσθέτουμε 10. Τώρα:  ;(Β) = δεκάδες  ;(Α) = μονάδες  LXI H,0A02H  MOV M,A ;1ο 7-seg-disp = μονάδες  INX H  MOV M,B ;2ο 7-seg-disp = δεκάδες  POP H  POP B  POP PSW  RET  DISP:  PUSH PSW  PUSH D  LXI D,0A00H ;GO TO block 0A00H - 0A05H  CALL STDM  CALL DCD  POP D  POP PSW  RET  END |

Άσκηση 2

|  |
| --- |
| Πρόγραμμα 2 σε Assembly |
| MAIN: IN 10H  MVI A,0DH  SIM ;RST 6,5 ONLY ENABLE  EI  MVI B,06H  LXI H,0A00H  CLEAR: ;CLEAR SCREEN  MVI M,10H  INX H  DCR B  JNZ CLEAR  MVI D,32H ;K1  MVI E,50H ;K2    PUSH D ;SAVE D  LXI D,0A00H ;6 SPACE IN SCREEN  CALL STDM  POP D  CALL DCD  WAIT:  JMP WAIT ;WAIT INTERRUPT  INTR\_ROUTINE:  CALL KIND ;PUT LSB A  STA 0A00H  MOV B,A  CALL KIND ;PUT MSB A  STA 0A01H  RLC  RLC  RLC  RLC  ADD B ;THE WHOLE NUMBER    ;NUMBER<=32H  CMP D  JZ DOWN32  JC DOWN32  JMP CHECK50  DOWN32: ;3RD LSB  MVI A,04H  JMP END  CHECK50: ;32H<NUMBER<=50H  CMP E  JZ DOWN50  JC DOWN50  JMP OTHER  DOWN50:  MVI A,02H ;2ND LSB  JMP END  OTHER: ;50H<NUMBER  MVI A,01H ;1ST LSB  JMP END  END:  CMA  STA 3000H ;OPEN 1 LED  PUSH D  LXI D,0A00H ;READ FROM HERE  CALL STDM  POP D  EI    SCREEN: ;NUMBER IN SCREEN  CALL DCD  JMP SCREEN  END |

Θεωρητικές Ασκήσεις

Άσκηση 3

**Α)**

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)**

FILL MACRO RP, X, K

**PUSH** PSW

**PUSH** H

**MOV** A,X

**LXI** H,RP

START:

**MVI** M,K

**INR** M

**DCR** A

**JNZ** START

**POP** H

**POP** PSW

ENDM

**Γ)**

RHLR MACRO n

**PUSH** PSW

**PUSH** B

**MVI** A,n

**CPI** 00H

**JZ** FINISH

**MVI** B,n

START:

**MOV** A,L

**RAR**

**MOV** L,A

**MOV** A,H

**RAR**

**MOV** H,A

**DCR** B

**JNZ** START

FINISH:

**POP** B

**POP** PSW

ENDM

Άσκηση 4

Η διακοπή συμβαίνει στο μέσο της εντολής **CALL 0880H**, άρα θα ολοκληρωθεί η εκτέλεση της τρέχουσας εντολής: η τρέχουσα τιμή του μετρητή προγράμματος **(0800H)** αποθηκεύεται στην στοίβα, ο δείκτης στοίβας ανεβαίνει 2 θέσεις πάνω και στον μετρητή προγράμματος καταχωρείται η διεύθυνση **0880H**. Έπειτα σώζεται η τιμή του μετρητή προγράμματος και η κατάσταση του 8085 και εκτελείται η ρουτίνα εξυπηρέτησης της διακοπής RST 7.5. Αυτό σημαίνει ότι η τιμή του μετρητή προγράμματος (**0880H)** αποθηκεύεται ξανά στην στοίβα, ο δείκτης στοίβας ανεβαίνει άλλες 2 θέσεις πάνω και στον μετρητή προγράμματος καταχωρείται η διεύθυνση της διακοπής για να εκτελεστεί η σχετική ρουτίνα. Όταν ολοκληρωθεί η εκτέλεση της ρουτίνας εξυπηρέτησης της διακοπής, η διεύθυνση που βρίσκεται στην κορυφή της στοίβας **(0880H)** επανέρχεται στον μετρητή προγράμματος, ο δείκτης στοίβας κατεβαίνει 2 θέσεις κάτω και εκτελείται η ρουτίνα που αρχίζει από τη διεύθυνση **0880H**, σύμφωνα με την εντολή **CALL 0880H**. Όταν ολοκληρωθεί η εκτέλεση και της τελευταίας ρουτίνας, η διεύθυνση στην κορυφή της στοίβας **(0800H)** επαναφέρεται στον μετρητή προγράμματος, ο δείκτης στοίβας κατεβαίνει άλλες 2 θέσεις κάτω και συνεχίζεται η εκτέλεση του προγράμματος από τη διεύθυνση **0801H.**

Η όλη διαδικασία φαίνεται σχηματικά στον παρακάτω πίνακα, όπου δίνονται τα περιεχόμενα του μετρητή προγράμματος και της στοίβας αρχικά (1), μετά την εκτέλεση της εντολής **CALL 0880H** (2), μετά την πραγματοποίηση της διακοπής RST 7.5 (3), μετά την εκτέλεση της ρουτίνας εξυπηρέτησης της διακοπής (4) και μετά την εκτέλεση της ρουτίνας που καλεί η εντολή **CALL 0880H** (5).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | | **2** | | **3** | | **4** | | **5** | |
| PC | 0800H | PC | 0880H | PC | (RST 7.5) | PC | 0880H | PC | 0800H |
| SP | 00H | SP | 00H | SP | 80H | SP | 00H | SP | 00H |
| SP+1 | 30H | SP+1 | 08H | SP+1 | 08H | SP+1 | 08H | SP+1 | 30H |
|  |  | SP+2 | 00H | SP+2 | 00H | SP+2 | 00H |  |  |
|  |  | SP+3 | 30H | SP+3 | 08H | SP+3 | 30H |  |  |
|  |  |  |  | SP+4 | 00H |  |  |  |  |
|  |  |  |  | SP+5 | 30H |  |  |  |  |

Άσκηση 5

PORT\_IN EQU 20H

MVI A,0DH ;Αρχικοποίηση μάσκας διακοπών

SIM

LXI H,0000H

MVI C,64D

EI

ADDR:

MOV A,C ;LOOP UNTIL READ ALL DATA

CPI 00H ;

JNZ ADDR ;infinite loop

DI ;DISABLE ANY FUTURE INTERRUPTS

DAD H ;3 ολισθήσεις αριστερά και παίρνουμε το

DAD H ;ακέραιο μέρος του μέσου όρου στον H, ενώ

DAD H ;το δεκαδικό μέρος του μ.ο. στον L

HLT

0034:

JMP INT\_6.5

INT\_6.5:

PUSH PSW ;store flags and accumulator in stack

MOV A,C

ANI 00000001B ;check for odd interrupt

JPO READ\_MSB ;if odd interrupt go to READ\_MSB

IN PORT\_IN

ANI 00001111B ;Get only X0-X3

MOV B,A

JMP READ\_LSB

READ\_MSB:

IN PORT\_IN

ANI 00001111B ;Get only X0-X3

RLC ;Convert them to MSB of total number

RLC ;

RLC ;

RLC ;

ORA B ;get whole number (add 4 LSBs to 4 MSBs)

MVI D,00H

MOV E,A

DAD D ;add number to total sum stored in HL

READ\_LSB:

DCR C ;reduce counter by 1

POP PSW ;restore accumulator and flags

EI

RET ;go to infinite loop

PORT\_IN EQU 20H

LXI H,0000H

MVI C,64D

A1: ;loop while PORT\_IN = 0

IN PORT\_IN

MOV B,A

RLC

JNC A1

A2: ;loop while PORT\_IN = 1

IN PORT\_IN

RLC

JC A2

READ: ;Read number only when Data Ready goes from

; 0 to 1 and again 0

MOV A,C ;check if C gets zero and go to FINISH if so

CPI 00H

JZ FINISH

PUSH PSW ;store flags and accumulator in stack

MOV A,C

ANI 00000001B ;check for odd interrupt

JPO READ\_MSB ;if odd interrupt go to READ\_MSB

IN PORT\_IN

ANI 00001111B ;Get only X0-X3

MOV B,A

JMP READ\_LSB

READ\_MSB:

IN PORT\_IN

ANI 00001111B ;Get only X0-X3

RLC ;Convert them to MSB of total number

RLC ;

RLC ;

RLC ;

ORA B ;get whole number (add 4 LSBs to 4 MSBs)

MVI D,00H

MOV E,A

DAD D ;add number to total sum stored in HL

READ\_LSB:

DCR C ;reduce counter by 1

POP PSW ;restore accumulator and flags

JUMP A1

FINISH:

DAD H ;3 ολισθήσεις αριστερά και παίρνουμε το

DAD H ;ακέραιο μέρος του μέσου όρου στον H, ενώ

DAD H ;το δεκαδικό μέρος του μ.ο. στον L

HLT