ΣχολήΗλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Εργαστήριο Μικροϋπολογιστών

ΠΡΩΤΟ ΓΕΝΙΚΟ ΘΕΜΑ 8085

Λαμπράκος Χρήστος, Α.Μ: 03112062
Μανδηλαράς Νικηφόρος, Α.Μ: 03112012
Σπαθαράκης Δημήτριος, Α.Μ: 03113523
Ομάδα C05
Έβδομο Εξάμηνο

Παραδοτέα: 15/11/2015

Πρώτο Γενικό Θέμα 8085 Ζήτημα 5

Η αριθμομηχανή μας χωρίζεται σε 4 βασικές συναρτήσεις. Μία που διαβάζει δύο μονοψήφιους δεκαεξαδικούς αριθμούς, τους αποθηκεύει και τους απεικονίζει στα δύο αριστερά ψηφία (συνάρτηση RDEM). Μία που περιμένει για το πάτημα του A ή του F και κάνει πολλαπλασιασμό ή πρόσθεση των δύο αριθμών αντίστοιχα (συνάρτηση Praxi). Μία που κάνει μετατροπή σε από δεκαεξαδικό σε δεκαδικό σύστημα (η οποία είναι ακριβώς ίδια με την προηγούμενη αναφορά, συνάρτηση BCD). Και τέλος μία, μία που ελέγχει αν πατήθηκε INCR ή DECR, και ανάλογα μηδενίζει ή προσθέτει το αποτέλεσμα της προηγούμενης πράξης σε έναν accumulator (συνάρτηση Accum). Το πρόγραμμα μας είναι συνεχής λειτουργίας.
Η συνάρτηση RDEM ξεκινάει βάζοντας το «κενό» σε όλες τις θέσεις μνήμης που θα χρησιμοποιηθούν για τις αναπαραστάσεις στο display. Κατόπιν χρησιμοποιώντας έναν μετρητή, τον B , διαβάζει μέσω της ρουτίνας KIND δύο αριθμούς που ελέγχει αν είναι έγκυροι (δέχεται μόνο ψηφία αριθμών), τους αποθηκεύει στους καταχωρητές H,L και κατόπιν τους εμφανίζει στις δύο αριστερότερες θέσεις του Display. Επίσης τσεκάρει αν έχει δοθεί κάποιο νούμερο (από την προηγούμενη συνάρτηση, αφού είναι συνεχής λειτουργίας) και τότε διαβάζει μόνο τον δεύτερο αριθμό.
Μετά η συνάρτηση PRAXI, περιμένει να διαβάσει από το πληκτρολόγιο A ή F και ανάλογα κάνει πρόσθεση ή πολλαπλασιασμό. Η πρόσθεση είναι εύκολη αλλά για τον πολλαπλασιασμό μπαίνει σε μία Loop που αθροίζει τον H με τον εαυτό του L φορές. Το αποτέλεσμα είναι στον καταχωρητή E.
Ακολουθεί η συνάρτηση BCD, εδώ όλο το ζήτημα ήταν η μετατροπή του μέτρου του αριθμού σε μορφή BCD, ώστε να απεικονιστεί κάθε ένα από τα ψηφία του. Για το σκοπό αυτό έγινε χρήση του διαγράμματος ροής που συνοδεύει την εκφώνηση της προηγούμενης άσκησης, σύμφωνα με το οποίο αφαιρούμε διαρκώς δέκα από τον αριθμό, προσθέτοντας δεκάδες σε ένα μετρητή, μέχρις ότου ο αριθμός να γίνει μικρότερος του 10--οπότε και έχουμε φτάσει στις μονάδες. Τα σχόλια του κώδικα δείχνουν σε ποιους καταχωρητές αποθηκεύσαμε το εκάστοτε μέγεθος. Το αποτέλεσμα τυπώνεται στις δεξιές θέσεις του DISPLAY.
Τέλος η συνάρτηση ACCUM, καλεί αρχικά την KIND. Επειδή η χρήση του συσσωρευτή δεν είναι αναγκαστική αν διαβαστεί αριθμός τον αποθηκεύει και σταματάει (οπότε η πρώτη συνάρτηση θα διαβάσει μόνο έναν αριθμό). Αν από την KIND διαβαστεί τον DECR τότε μηδενίζεται ο accumulator B. Αλλιώς αθροίζεται σε δεκαεξαδική μορφή από τον καταχωρητή E (που είχε το αποτέλεσμα σε hex). Σε κάθε περίπτωση το αποτέλεσμα εμφανίζεται στα δύο αριστερότερα ψηφία. Όλες οι αναπαραστάσεις αριθμών έγιναν μέσω των ρουτίνων STDM, DCD με αντίστοιχα PUSH,POP πριν και μετά για να μην αλλοιώνονται οι καταχωρητλές.

```
BEGIN:      IN 10H
      MVI H,2AH                 ;just a flag
      MVI B,00H                 ;accumulator initialization

LOOPA:      CALL RDEM                 ;read 2 numbers
      CALL PRAXI                ;choose between sum and multiplication
      CALL BCD                  ;show the result
      CALL ACCUM                ;check accumulator function
      JMP LOOPA
;=============================================================================
RDEM: PUSH D
      PUSH B
      PUSH PSW
      MVI A,10H                 ;blank code
      LXI D,0B00H               ;display address
      MVI B,04H                 ;counter
KILL: STAX D                    ;initialize all digits as blank
      INX D
      DCR B
      JNZ KILL
      MVI B,02H                 ;B counts the digits read
      MOV A,H
      CPI 2AH                   ;has a first number been given?
      JZ GET                    ;if not, read both numbers now!
      DCR B
      LXI D,0B05H               ;if yes, be sure to light it up!
      STAX D
BGET: LXI D,0B04H
      MVI A,10H
      STAX D
GET:  LXI D,0B00H
      PUSH H
      CALL STDM
      CALL DCD                  ;update display
INPUT:      CALL KIND                 ;wait for input
      POP H
      MOV C,A
      MOV A,B                   ;check which number do we wait for
      CPI 01H
      JZ UPS
NUPS: LXI D,0B04H
      MVI A,10H
      STAX D
      LXI D,0B05H               ;first number goes to the left
      JMP LOAD
UPS:  LXI D,0B04H               ;and second to the right
LOAD: MOV A,C                   ;retrieve input
      STAX D
      DCR B                     ;update counter
      JZ SEC                    ;if we are finished, go out
      MOV H,A                   ;else save first number
      JMP GET
SEC:  MOV L,A                   ;save second number
      LXI D,0B00H
      PUSH H
      CALL STDM
      CALL DCD
      POP H
      POP PSW
      POP B
      POP D
```

```
RET
;===============================================================================
PRAXI: PUSH B                    ;swsimo kataxwrhtwn kai shmaiwn
       PUSH PSW
INP:   PUSH H
       CALL KIND
       POP H
       CPI 0AH                   ;elegxos an edwse A
       JZ PROS                   ;an nai, pros8esi
       CPI 0FH                   ;allios elegxos gia F
       JNZ INP                   ;an oute auto, perimene allo patima
       JMP MLT                    ;allios pollaplasiase
PROS:  MOV A,H
       ADD L                     ;prosthesh
       JMP TELOS
MLT:   CPI 0FH                   ;elegxos an edwse F
       JNZ INP
       MVI A,00H
AD1:   ADD H                     ;L diadoxikes prostheseis tou H
       DCR L
       JNZ AD1
TELOS: MOV E,A                   ;apotelesma ston kataxwrith E
       POP PSW
       POP B                     ;epanafora timwn
       RET
;===============================================================================
ACCUM: PUSH PSW
AGAIN: PUSH B
       PUSH D
       PUSH H
       CALL KIND
       POP H
       POP D
       POP B
       CPI 83H
       JZ GOINCR
       CPI 81H
       JZ GODECR
       CPI 00H             ;did we press a number instead?
       JC AGAIN
       CPI 0FH
       JNC AGAIN           ;if not, wait for new input
       MOV H,A             ;if yes, save the number
       JMP FNSH
DISPLAY:
         MOV A,B
         MOV L,A
         ANI 0FH           ;keep right half of code
         STA 0B04H         ;save it in fifth digit
         MOV A,L
         ANI F0H           ;likewise for left half
         RRC
         RRC
         RRC
         RRC
         STA 0B05H
         PUSH D
         PUSH H
         PUSH B
LIGHT:    LXI D,0B00H
         CALL STDM
         CALL DCD
         POP B
```

```
         POP H
POP D
         JMP FNSH
GODECR: MVI B,00H          ;CALL PRINT HEX NUMBERS
        MVI H,2AH
        JMP DISPLAY
GOINCR:
        MOV A,B
        ADD E
OLE:    MOV B,A            ;CALL PRINT HEX NUMBERS
        MVI H,2AH
        JMP DISPLAY
FNSH:  POP PSW
       RET
;===============================================================================
BCD:   PUSH H
       PUSH B
       PUSH PSW
       MOV A,E
KAT3:  MVI H,00H     ;decades here
       MVI L,00H     ;units here
       MVI C,00H     ;...and finally the hundreds
LOOPB: CPI 0AH
       JNC KAT
KAT2:  MOV L,A       ;number was less than 10, so this
       JMP OUTB      ;is the units' value
KAT:   INR H         ;number was greater than 10, keep
       SUI 0AH       ;another decade and uppdate
       JMP LOOPB
OUTB:  MOV A,H
       CPI 14H
       JC HUNDR
       MVI C,02H
       SUI 14H
       MOV H,A
HUNDR: CPI 0AH       ;if there were more than 10 decades...
       JC GOON
       MVI C,01H     ;...hundreds go to one,
       MOV A,H
       SUI 0AH       ;and subtract 10 from the decades
       MOV H,A
GOON:  MOV A,C       ;this just save the BCD values...
       STA 0B02H
       MOV A,H       ;...in the right...
       STA 0B01H
       MOV A,L       ;...place.
       STA 0B00H
       PUSH D
       LXI D,0B00H   ;light em upp
       CALL STDM
       CALL DCD
       POP D
       POP PSW
       POP B
       POP H
       RET
;===============================================================================
       END
```