Ε.Μ.Π. - ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧ. & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ & ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

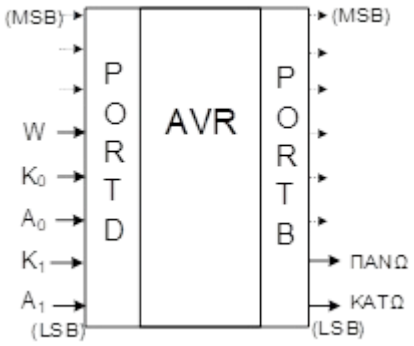
ΑΘΗΝΑ 25. 6. 2021

ΓΡΑΠΤΗ ΕΞΕΤΑΣΗ ΣΤΟ ΜΑΘΗΜΑ "***Συστήματα Μικροϋπολογιστών***"

(ΘΕΜΑ 2ο – ΣΥΝΟΛΟ 4.5 Μονάδες)

**Έναρξη 12:30 - ΔΙΑΡΚΕΙΑ 60' + 10' Παράδοση: 13:40'**

**ΟΝΟΜΑΤΕΠΩΝΥΜΟ: Θοδωρής Παπαρρηγόπουλος el18040**




**ΘΕΜΑ 2ο:** (4.5 ΜΟΝΑΔΕΣ): Σε ένα μικροελεγκτή AVR Mega16 που αξιοποιεί μία θύρα εισόδου και μία εξόδου, όπως φαίνεται στο διπλανό σχήμα, να υλοποιηθεί ένα σύστημα οδήγησης ενός ανελκυστήρα δυο θέσεων (ισογείου και 1ου ορόφου). Η κίνηση προς το ισόγειο ή τον 1ο όροφο ελέγχεται από τους εξωτερικούς διακόπτες (Push-Buttons) *K*0 και *K*1 αντίστοιχα καθώς και από έναν εσωτερικό διακόπτη (Push-Button) *W*. Για να δοθεί εντολή από τους διακόπτες αυτούς, προϋπόθεση είναι το βαγόνι να είναι σταματημένο στο ισόγειο ή στον 1ο όροφο. Όταν κινείται πρέπει να σταματάει από το πρόγραμμα με βάση τους αισθητήρες *A*0 και *A*1 που είναι τερματικοί διακόπτες και οι οποίοι δίνουν λογικό 1 αυτόματα όταν ο θάλαμος φτάνει στο ισόγειο ή στον 1ο όροφο αντίστοιχα. Υποθέτουμε ότι κατά την εκκίνηση του συστήματος, ο θάλαμος πρέπει να βρίσκεται στο ισόγειο, αλλιώς πριν δεχτεί οποιαδήποτε εντολή να μεταφέρεται σε αυτή τη θέση αυτόματα.

Αναλυτικά, αν ο θάλαμος φτάσει στο ισόγειο, τότε πρέπει να σταματάει η κίνησή του και να ελέγχονται οι διακόπτες *K*1 και *W*. Αν ένας από αυτούς είναι ενεργοποιημένος (=1) τότε έχουμε κίνηση προς τα πάνω. Αντίστοιχα αν ο θάλαμος φτάσει στον 1ο όροφο, τότε πρέπει να σταματάει η κίνησή του και να ελέγχονται οι διακόπτες *K*0 και *W*. Αν ένας από αυτούς είναι ενεργοποιημένος (=1) τότε έχουμε κίνηση προς τα κάτω. Δώστε το αντίστοιχο πρόγραμμα σε assembly και σε C.

(Assembly: 2.5 ΜΟΝΑΔΕΣ και C: 2 ΜΟΝΑΔΕΣ)

**C:**

```
#include <avr/io.h>
#include <mega16.h>

char A1,A0;
char K1,K0;
char W;
char input;
char output;

void init() {
   while (1) {
      input = PIND; // read input
      input = PIND; // read input
      A1 = input & 0x01; // Get A1 digit
      K1 = input & 0x02 >> 1; // Get K1 digit
      A0 = input & 0x04 >> 2; // Get A0 digit
      K0 = input & 0x08 >> 3; // Get K1 digit
      W  = input & 0x16 >> 4; // Get W digit
      if (A1 == 1) {
         output = 0x01; // steile sto isogio
      } else {
         output = 0x00;
         break;
      }
   }
```

```
}
int main() {
   DDRD = 0x00; // set D as input
   DDRB = 0xFF; // set B as output

   init();

   while (1) {
      input = PIND; // read input
      A1 = input & 0x01; // Get A1 digit
      K1 = input & 0x02 >> 1; // Get K1 digit
      A0 = input & 0x04 >> 2; // Get A0 digit
      K0 = input & 0x08 >> 3; // Get K1 digit
      W  = input & 0x16 >> 4; // Get W digit
      if ( (A1 == 1 && A0 == 1) | (A1 == 0 && A0 == 0) ) {
         output = 0x03; // 11 as output indicates error
      }
      else if (A1 == 1) {
         if (W == 1 || K0 == 1) {
            output = 0x01;
         } else {
            output = 0x00;
         }
      } else if (A0 == 1) {
         if (W == 1 || K0 == 1) {
            output = 0x02;
         } else {
            output = 0x00;
         }
      }
      PORTB = output;
   }
}
```

**Assembly:**

```
ser r24 ; set register r24 = 0xFF
out DDRB, r24 ;B is output port
clr r24 ; clear register to r24 = 0x00
out DDRB, r24 ; A is input port

ldi r17, 0x01; isogio check buttons
sbis PIND, 2 ; isogio check buttons
rjmp init ; else isiogio

K1:sbis PIND, 1 ; isogio check buttons
        rjmp W1
        ldi r16, 0x02
        out PORTB, r16
        rjmp A1

W1: sbis PIND, 4 ; first floor and W
        rjmp K1
        ldi r16, 0x02
        out PORTB, r16
        rjmp A1

init: sbis PIND,2 ; if floor zero
        out PORTB, r17

A0: : sbis PIND, 2  ; if first floor send 0 else check buttons
```

```
        rjmp A1
        ldi r16, 0x00
        out PORTB, r16
        rjmp K1

A1: sbis PIND, 0 ; if first floor and read A1
        rjmp A0
        ldi r16, 0x00
        out PORTB, r16

K0:sbis PIND, 3 ; if first floor and read K0
        rjmp W2
        out PORTB, r17
        rjmp init

W2: sbis PIND, 4 ; if first floor and read W
        rjmp K0
        out PORTB, r17
        rjmp init
```