



ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ (Α.Π.Θ.)

ΗΥ0901 ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ ΚΑΙ ΠΕΡΙΦΕΡΕΙΑΚΑ

Εργασία Εργαστηρίου: Ἀσκησή Με Σιλό Αποθήκευσης

*Αντωνιάδης Δημήτριος (8462): akdimitri@auth.gr
Δημητριάδης Βασίλειος (8404): dimvasdim@auth.gr*

Περιεχόμενα

1 Εισαγωγή.	2
2 Λειτουργία προγράμματος.	3
2.1 Θεωρητική περιγραφή της λειτουργίας του προγράμματος.	3
2.2 Τλοποίηση του προγράμματος.	4
2.2.1 Τυποποιήσεις και ορισμοί.	4
2.2.2 Διάνυσμα RESET.	5
2.2.3 TIMER0 OVERFLOW VECTOR	6
2.2.4 MAIN.	8
2.2.5 PAUSE.	8
2.2.6 ALARM.	8
2.2.7 TIM2_OVF.	8
3 Αποτελέσματα.	9
4 Επίλογος.	11

1 Εισαγωγή.

Το παρόν έγγραφο αποτελεί την αναφορά της εργασίας εργαστηρίου που πραγματοποιήθηκε στο πλαίσιο του μαθήματος *Μικροεπεξεργαστές και Περιφερειακά*. Σκοπός της εργασίας ήταν ο προγραμματισμός του μικροελεγκτή **ATmega16** προκειμένου αυτός να ελέγχει και να ρυθμίζει μία δομή αποθήκευσης, η οποία περιλαμβάνει 3 σιλό αποθήκευσης. Πιο συγκεκριμένα, το ζητούμενο ήταν ο αποδοτικός διαμερισμός των αποθηκευμένων υλικών από το Silo 0 στα Silo 1 και Silo 2 μέσω μίας γραμμής μεταφοράς.

Παραπάνω έγινε μία σύντομη αναφορά του σκοπού της εργασίας. Στην επόμενη (2^η) ενότητα παρουσιάζεται η λειτουργία του προγράμματος και στην τρίτη (3^η) ενότητα τα αποτελέσματα λειτουργικότητας του προγράμματος. Τέλος, η τέταρτη (4^η) ενότητα αποτελεί τον επίλογο της εργασίας αυτής.

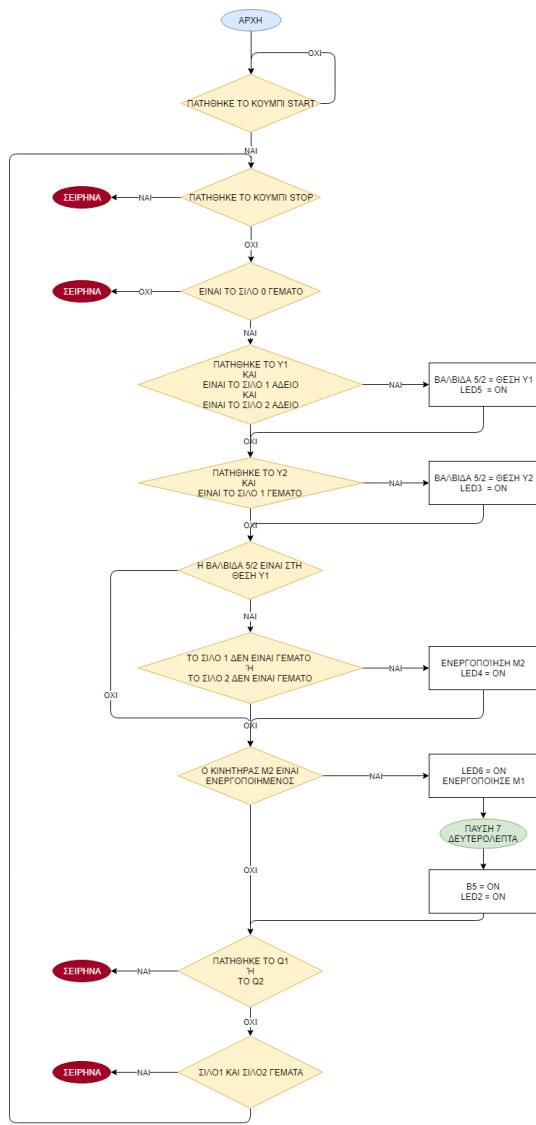
Ο πηγαίος κώδικας του προγράμματος βρίσκεται στο αρχείο **silo.asm**. Επιπλέον, στο φάκελο υποβολής περιλαμβάνεται και το αρχείο **README.txt** το οποίο περιγράφει τον πηγαίο κώδικα.

Το σύνολο των αρχείων της εργασίας μπορεί να βρεθεί στον παρακάτω σύνδεσμο: <https://github.com/akdimitri/Microprocessors-and-Peripherals-Lab-Project.git>

2 Λειτουργία προγράμματος.

2.1 Θεωρητική περιγραφή της λειτουργίας του προγράμματος.

Η λειτουργία του προγράμματος είναι σχετικά απλή και φαίνεται στο παρακάτω διάγραμμα.



Σχήμα 1: Διάγραμμα ροής κυρίως προγράμματος.

τήρας την κατάλληλη ταχύτητα και ανάβει το LED2.

Τέλος, αν έχει πατηθεί κάποιο από τα κουμπιά που προσομοιώνουν τα θερμικά Q1, Q2 ή έχουν γεμίσει και τα δύο σιλό (Σιλό 1 και Σιλό 2), τότε ηγείται η σειρήνα.

Καθώς εκτελείται το παραπάνω πρόγραμμα, ο μικροελεγκτής ανά τακτά χρονικά διαστήματα, διαχορίζει τη ροή του προγράμματος προκειμένου να διαβάζει τις στάθμες των τριών σιλό.

Αρχικά, το πρόγραμμα περιμένει να πατηθεί το START button. Αν πατηθεί τότε εκτελείται το πρόγραμμα.

Από τη στιγμή που έχει πατηθεί το START button το πρόγραμμα μπαίνει σε ένα συνεχές loop. Το πρώτο πράγμα που ελέγχει το πρόγραμμα ενόσω βρίσκεται σε αυτό το συνεχές loop είναι το γεγονός της αναγνώρισης του πατήματος του STOP button. Αν πατηθεί το STOP button τότε ενεργοποιείται η σειρήνα και εκτελείται η ρουτίνα που την εξυπηρετεί, αλλιώς συνεχίζει κανονικά η εκτέλεση του προγράμματος.

Ο επόμενος έλεγχος αφορά το σιλό αποθήκευσης (Σιλό 0), αν το σιλό αποθήκευσης είναι άδειο, τότε ενεργοποιείται η σειρήνα, αλλιώς συνεχίζει η εκτέλεση του προγράμματος.

Στη συνέχεια ελέγχεται αν πατήθηκε το κουμπί προσομοίωσης για τη θέση Y1. Τότε αν το σύλο 1 είναι άδειο και το σύλο 2 είναι άδειο, η βαλβίδα 5/2 τοποθετείται στη θέση Y1 και ανάβει το LED5.

Το επόμενο πράγμα που ελέγχει το πρόγραμμα είναι αν πατήθηκε το κουμπί προσομοίωσης Y2 και το σιλό 1 είναι γεμάτο. Τότε μετακινείται η βαλβίδα στη θέση Y2 και το LED3 ανάβει.

Τώρα, αν η βαλβίδα 5/2 βρίσκεται στη θέση Y1 και το σιλό 1 ή το σιλό 2 δεν είναι γεμάτο ενεργοποιείται ο κινητήρας M2 και ανάβει το LED4.

Στη συνέχεια, ελέγχεται αν ο κινητήρας M2 είναι ενεργοποιημένος. Τότε ανάβει το LED6 και ενεργοποιείται ο M1. Το πρόγραμμα περιμένει 7 δευτερόλεπτα για να πιάσει ο κινητήρας M1.

2.2 Υλοποίηση του προγράμματος.

Η υλοποίηση του προγράμματος βασίστηκε στο Datasheet του ATmega16. [1] Η ταχύτητα του επεξεργαστή ορίστηκε ώστε $clk = 4 \text{ MHz}$. Όλες οι ρυθμίσεις που παρουσιάζονται παρακάτω έχουν γίνει σύμφωνα με τον παραπάνω ορισμό της ταχύτητας του επεξεργαστή.

2.2.1 Τυποποιήσεις και ορισμοί.

Για την εκτέλεση του προγράμματος χρησιμοποιούνται 2 καταχωρητές οι οποίοι ονομάζονται στο πρόγραμμα

- COntr ol REgister A (COREA), ο οποίος αντιστοιχεί στον καταχωρητή R30.
- COntr ol REgister B (COREB), ο οποίος αντιστοιχεί στον καταχωρητή R31.

Τα bits των καταχωρητών αυτών μεταχειρίζονται ως εξής:
COREA:

B1	B2	B3	B4	A1	Y1	Y2	-
7	6	5	4	3	2	1	0

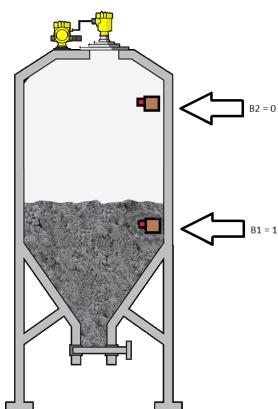
COREB:

M1	M2	-	-	-	B5	-	-
7	6	5	4	3	2	1	0

Σημειώνεται ότι για τα bits B1, B2, B3, B4, A1, το 1 σημαίνει ότι το αποθηκευμένο υλικό στο σιλό έχει φτάσει τη στάθμη αυτή, ενώ το 0 ότι ακόμη δεν έχει φτάσει τη στάθμη αυτή. (Σχήμα 2)

Επιπλέον, κατά την εκτέλεση του προγράμματος χρησιμοποιούνται τρία Interrupt Vectors. Τα διανύσματα αυτά βρίσκονται σύμφωνα με το [1] στις εξής διεύθυνσεις και ανακατευθύνουν τη ροή του προγράμματος στο σημείο όπου έχει τοποθετηθεί το αντίστοιχο Label:

- \$0000 RESET
- \$0008 TIM0_OVF
- \$0012 TIM0_OVF



Σχήμα 2: Παράδειγμα τιμών καταχωρητών.

Το κομμάτι του κώδικα που εκτελεί το πρόγραμμα τοποθετείται κάτω από τη διεύθυνση \$002A όπως προτείνεται στο [1].

2.2.2 Διάνυσμα RESET.

Το διάνυσμα αυτό εκτελείται στην εκκίνηση του προγράμματος και είναι υπεύθυνο για την πραγματοποίηση των ρυθμίσεων του μικροελεγκτή ώστε να εκτελεστεί το πρόγραμμα. Το διάνυσμα RESET επιπλέον εκτελείται με το πάτημα του κουμπιού RESET που βρίσκεται στην πλακέτα ή με κλήση του διανύσματος RESET κατά τη διάρκεια εκτέλεσης του προγράμματος.

Στο συγκεκριμένο πρόγραμμα το διάνυσμα RESET είναι υπεύθυνο για την τοποθέτηση του Stack Pointer στην κορυφή της RAM [1], για την αρχικοποίηση του ADC, τον ορισμό των INPUT-OUTPUT PINS και την ενεργοποίηση του TIMER0 OVERFLOW VECTOR. Επιπλέον, είναι υπεύθυνο για την απενεργοποίηση του TIMER2 OVERFLOW VECTOR και του BUZZER.

Για την αρχικοποίηση του ADC τίθεται ο ADCSRA με την εξής τιμή:

ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
1	0	0	0	0	1	0	1

Το ADEN ενεργοποιεί τον ADC. Ακόμη, με τα bits: ADPS2, ADPS1, ADPS0, ο prescaler τίθεται σε clk/32, ώστε να δουλεύει σε συχνότητα 125 kHz. Αυτό γίνεται διότι στο datasheet αναφέρεται ότι ο ADC δουλεύει καλύτερα σε συχνότητες 50-200 kHz. Μία μετατροπή ζεκινάει όταν τίθεται το bit ADSC σε 1. [1] Η διαδικασία της μετατροπής περιγράφεται αναλυτικά στην υποενότητα που περιγράφει τον TIM0_OVF.

Επίσης, στο διάνυσμα αυτό ενεργοποιείται ο TIMER0 OVERFLOW. Συγκεκριμένα, στον καταχωρητή TIMSK τίθεται σε 1 το bit 0, δηλαδή το TOIE0. Με τον τρόπο αυτό ενεργοποιείται το Interrupt Vector TIM0_OVF. Ακόμη, ο καταχωρητής TCCR0 τίθεται με την εξής τιμή:

FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00
0	0	0	0	0	0	0	1

Με τον παραπάνω ορισμό του καταχωρητή TCCR0, ο TIMER/COUNTER0 τίθεται σε Normal Mode με Prescaler ίσο με clk/1 = 4 MHz.

Ακόμη, στο διάνυσμα αυτό απένεργοποιείται ο TIMER2 OVERFLOW. Συγκεκριμένα, στον καταχωρητή TIMSK τίθεται σε 0 το bit 6, δηλαδή το TOIE2. Με τον τρόπο αυτό απενεργοποιείται το Interrupt Vector TIM2_OVF. Ακόμη, ο καταχωρητής TCCR0 ορίζεται να ισούται με την τιμή 0.

Τέλος, στο διάνυσμα RESET ορίζονται τα PINS του PORTD ως είσοδοι και αυτά του PORTB ως έξοδοι. Το PORTC:0 τίθεται ως έξοδος και τίθεται στην τιμή 0 (BUZZER). Η τιμή του καταχωρητή R28 τίθεται στην τιμή 0. Ο καταχωρητής αυτός χρησιμοποιείται στο διάνυσμα TIM2_OVF και περιγράφεται στην αντίστοιχη υποενότητα αναλυτικά. Οι καταχωρητές COREA, COREB αρχικοποιούνται στην τιμή 0 με σκοπό να λάβουν τις κατάλληλες τιμές κατά την κλήση του TIM0_OVF.

Το διάνυσμα RESET φαίνεται παρακάτω σε επίπεδο κώδικα.

```

1 .ORG 0x02A
2 RESET:
3   CLI           ; Disable Interrupts
4   LDI R16, HIGH(RAMEND)      ; Main program start
5   OUT SPH, R16              ; Set stack pointer to top of RAM
6   LDI R16, LOW(RAMEND)
7   OUT SPL, R16
8
9   LDI COREA, 0b0000000000    ; Initialize CONTROL REGISTER A
10  LDI COREB, 0b0000000000    ; Initialize CONTROL REGISTER B
11
12  SBI DDRC, 0               ; PORTC:0 OUTPUT
13  CBI PORTC, 0
14
15  LDI R16, 0b11111111       ; Define PORTB -> OUTPUT
16  OUT DDRB, R16
17  OUT PORTB, R16            ; TURN OFF LEDS
18
19  LDI R16, 0b00000000       ; Define PORTD -> INPUT
20  OUT DDRD, R16
21  LDI R16, 0b11111111       ; Activate PULL UP Resistors
22  OUT PORTD, R16
23
24  LDI R16, 0b10000101       ; ENABLE ADC CONVERTER, SET PRESCALING TO CLK
25   /32 -> 125 kHz
26  OUT ADCSRA, R16
27
28  LDI R16, 0b00011111       ; ENABLE PULL UP RESISTORS OF PA0-PA4
29  OUT PORTA, R16
30
31  IN R17, TIMSK
32  ANDI R17, 0b10111111      ; Disable TIM2_OVF Interrupt
33  OUT TIMSK, R17
34  LDI R17, 0                 ; Stop TIMER2
35  OUT TCCR2, R17
36  LDI R28, 0                 ; Initialize counter of TIM2_OVF times
37
38  LDI R16, 0xFA             ; Set COUNTER0 to 0xFA to FORCE interrupt in a
39   few steps
40  OUT TCNT0, R16
41  LDI R16, 0b00000001       ; Enable TIM0_OVF Interrupt
42  IN R17, TIMSK
43  OR R17, R16
44  OUT TIMSK, R17
45  LDI R16, 0b00000001       ; Start COUNTER0, Normal Mode, Prescaler =
46   clk/1 -> 4 MHz
47  OUT TCCRO, R16
48  SEI

```

2.2.3 TIMER0 OVERFLOW VECTOR

Στο κύριο πρόγραμμα εντοπίζεται υπό το Label **TIM0_OVF**. Κατα τη διάρκεια εκτέλεσης της MAIN, κάθε 64 μικροδευτερόλεπτα καλείται το διάνυσμα **TIM0_OVF**. Αυτό είναι υπεύθυνο για την ανάγνωση της στάθμης των σιλό και ανάλογα τον προσδιορισμό των bits στον καταχωρητή COREA.

Η στάθμη των σιλό προσομοιώνεται με 5 ποτενσιόμετρα τα οποία είναι συνδεδεμένα στα bits:0-4

του PORTA. Στο διάνυσμα TIM0_OVF χρησιμοποιείται ο ADC για να αναγνωσθεί η τιμή των ποτενσιομέτρων. Αν η τιμή αυτή είναι μεγαλύτερη από μία συγκεκριμένη τιμή, τότε το αντίστοιχο bit που αντιπροσωπεύει τη στάθμη αυτή τίθεται σε 1, αλλιώς τίθεται στην τιμή 0.

Ενδεικτικά δίνεται παρακάτω η διαδικασία για την ανάγνωση μίας τιμής και συγκεκριμένα του A1. Η διαδικασία είναι παρόμοια και για την ανάγνωση των υπόλοιπων τιμών.

Για την πραγματοποίηση της μετατροπής ο καταχωρητής ADMUX τίθεται με την εξής τιμή:

REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0
1	1	1	0	0	0	0	0

Τα bits: REFS1, REFS0 χρησιμοποιούνται για την επιλογή της τάσης αναφοράς με σκοπό την πραγματοποίηση της μετατροπής. Στην συγκεκριμένη περίπτωση έχει επιλεχθεί εσωτερική τάση 2.56 Volts με εξωτερικό πυκνωτή στο PIN: AREF. Το bit: ADLAR τίθεται σε 1 ώστε το αποτέλεσμα, το οποίο αποθηκεύεται στους καταχωρητές ADCH, ADCL, να είναι Left Adjusted. Τέλος, τα 5 πρώτα bits χρησιμοποιούνται για την επιλογή του PIN που θα διαβάσει ο μετατροπέας, όπου στην προκειμένη περίπτωση έχει επιλεχθεί το PIN:0 του PORTA.

Για την εκκίνηση της μετατροπής ενεργοποιείται το bit:ADSC του ADCSRA. Τα υπόλοιπα bits του ADCSRA παραμένουν όπως αυτά περιγράφηκαν στην υποενότητα αναφορικά με το διάνυσμα RESET. Η Μετατροπή ολοκληρώνεται όταν το bit ADSC γίνει 0.

```

1 READ_PA0:
2   LDI R18, 0b11100000      ; INTERNAL VREF 2.56 V. ADLAR = 1 (LEFT
3     ADJUSTED) ->READ ADCH, PA0
4   OUT ADMUX, R18
5
5 LDI R18, 0b11000101      ; ENABLE ADC CONVERTER, START CONVERSION, SET
6   PRESCALING TO CLK/32 -> 125 kHz
7   OUT ADCSRA, R18
8 PAO_CONVERSION:
9   SBIC ADCSRA, 6           ; IF ADCSRA:6 (ADSC) IS CLEARED THEN THE
10    CONVERSION HAS BEEN COMPLETED
11   JMP PAO_CONVERSION
12
12 STORE_PA0:
13   IN R19, ADCH            ; ADLAR == 1 THEN IT IS LEFT ADJUSTED
14
14 CPI R19, 0x0F            ; IF(R19 > 0x0F) Then
15   BRSR SET_A1             ; SET A1
16 CLEAR_A1:
17   CBR COREA, 0b00001000
18   JMP READ_PA1
19 SET_A1:
20   SBR COREA, 0b00001000

```

Στον παρακάτω αλγόριθμο περιγράφεται σε μορφή ψευδογλώσσας η λειτουργία του TIM0_OVF.

Algorithm 1: TIMER0 OVERFLOW VECTOR

Data: Potentiometers Values

Result: Define B1,B2,B3,B4,A1 Values in COREA

- 1 Save SREG
 - 2 Read PA0-4 using ADC
 - 3 Set/Clear COREA: 7-3 /* B1, B2, B3, B4, A1 */
 - 4 Restore SREG
 - 5 Return
-

2.2.4 MAIN.

Η συνάρτηση MAIN υλοποιήθηκε σύμφωνα με τη θεωρητική περιγραφή που παρουσιάσθηκε στην ενότητα 2.1 και παρουσιάζεται στο παρακάτω διάγραμμα. Η συνάρτηση αυτή κατα την εκτέλεση της διαχόπτεται από το διάνυσμα TIM0_OVF. Επιπλέον, κατά την πρώτη ενεργοποίηση του M2 καλείται η συνάρτηση PAUSE η οποία απλά χρονοτρίβει για 7 δευτερόλεπτα έως ότου η γραμμή μεταφορά αποκτήσει την κατάλληλη ταχύτητα.

2.2.5 PAUSE.

Η καθυστέρησή 7 δευτερολέπτων πραγματοποιείται με τη συνάρτηση PAUSE, η οποία με 3 καταχωρητές εκτελεί 28 000 000 κύκλους ρολογιού στα 4 MHz.

2.2.6 ALARM.

Η συνάρτηση MAIN περιλαμβάνει ορισμένα σενάρια κατα τα οποία καλείται η συνάρτηση ALARM. Η συνάρτηση αυτή απενεργοποιεί το Interrupt Vector TIM0_OVF και ενεργοποιεί το Interrupt Vector TIM2_OVF. Επίσης, η συνάρτηση αυτή ενεργοποιεί το buzzer. Τέλος, περιμένει το πάτημα του κουμπιού ACKNOWLEDGEMENT και εκκινεί ξανά το πρόγραμμα από το VECTOR RESET. Η συνάρτηση ALARM όσο εκτελείται διαχόπτεται από την TIM2_OVF κάθε 0.5 δευτερόλεπτα η οποία είναι υπεύθυνη να αναβοσβήνει το LED0.

Για την απενεργοποίηση του TIM0_OVF το bit TOIE0 τίθεται 0 και ο καταχωρητής TCCR0 τίθεται να είναι ίσο με 0.

Για την ενεργοποίηση του TIM2_OVF το bit TOIE2 του TIMS τίθεται 1 και ο καταχωρητής TCCR2 ισούται με:

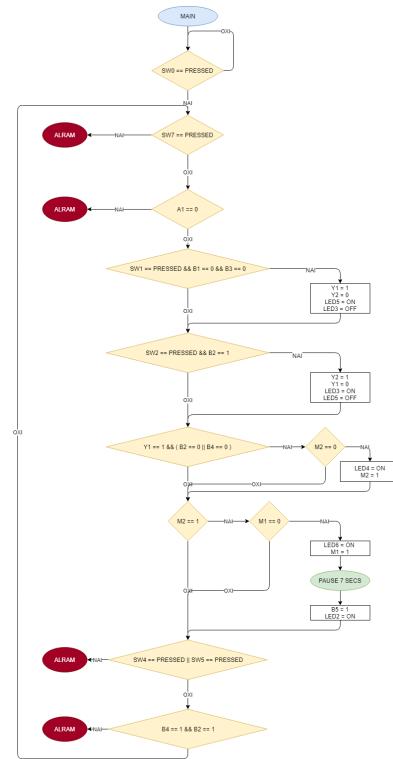
FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20
0	0	0	0	0	1	1	1

Συνεπώς, ο prescaler του TIMER2 ισούται με 1024. Στην επόμενη υποενότητα, αναλύεται η λογική με βάση την οποία αναβοσβήνει το LED0 κάθε 0.5 δευτερόλεπτα.

2.2.7 TIM2_OVF.

Στο διάνυσμα RESET αρχικοποιήθηκε ο καταχωρητής R28 ίσο με 0. Ο καταχωρητής αυτός μετράει πόσες φορές έχει κληθεί το διάνυσμα TIM2_OVF. Το διάνυσμα αυτό καλείται κάθε $(4MHz/1024/256)^{-1} = 0.065$ δευτερόλεπτα.

Μόλις ο καταχωρητής R28 γίνει ίσος μέ 7, δηλαδή ύστερα από χρόνο $7 * 0.065 = 0.45$ δευτερόλεπτα, το διάνυσμα TIM2_OVF αναστρέφει την τιμή στην έξοδο του PIND:0. Δηλαδή κάθε 0.5 δευτερόλεπτα περίπου αναβοσβήνει το LED0. Στη συνέχεια, μηδενίζει τον μετρητή R28 και επιστρέφει στην κανονική ροή.



Σχήμα 3: Συνάρτηση MAIN.

```

1  TIM2_OVF:
2    IN R27, SREG           ; SAVE STATUS REGISTER IN STACK
3    PUSH R27
4
5    INC R28                ; INCREMENT times TIM2_OVF counter
6    CPI R28, 7             ; If( counter != 7)
7    BRNE FINISH_TIM2_OVF   ; Then exit
8    ; Else
9  REINITIALIZE_TEMPORARY_COUNTER: ; Toggle LED0
10   LDI R28, 0
11
12 TOGGLE_LED0:
13   SBIC PORTB, 0
14   JMP TURN_ON_LED0
15 TURN_OFF_LED0:           ; TOGGLE LED0 EVERY 0.5 SECOND
16   SBI PORTB, 0
17   JMP FINISH_TIM2_OVF
18 TURN_ON_LED0:
19   CBI PORTB, 0
20
21 FINISH_TIM2_OVF:
22   POP R27
23   OUT SREG, R27
24   RETI

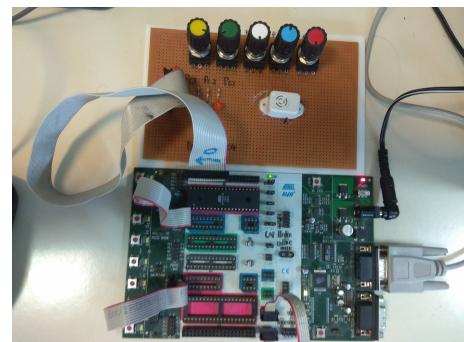
```

3 Αποτελέσματα.

Το πρόγραμμα ελέγχθηκε στην πλακέτα STK500 του εργαστηρίου και ήταν πλήρως λειτουργικό.

Στο σχήμα 4 φαίνεται στην πλακέτα ότι το πρόγραμμα αναμένει το πάτημα του κουμπιού START διότι κανένα LED δεν είναι αναμμένο. Ακόμη, το ποτενσιόμετρο PA0 έχει στραφεί ώστε A1 = 1.

Στο σχήμα 5 φαίνεται να έχει πατηθεί το START και το πρόγραμμα να τρέχει διότι το LED7 είναι αναμμένο. Το επόμενο βήμα είναι να τοποθετηθεί η βαλβίδα 5/2 στη θέση Y1.



Σχήμα 4: Αναμονή για START button.

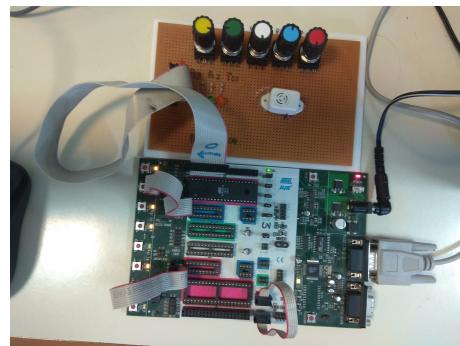


Σχήμα 5: Running.

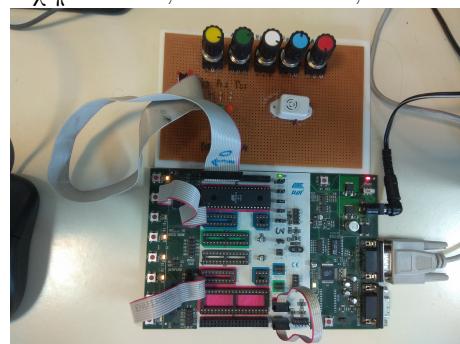
Στο σχήμα 6 φαίνεται ότι η βαλβίδα 5/2 βρίσκεται στη θέση Y1 (LED5 = ON), οι κινητήρες M1, M2 λειτουργούν (LED4 = ON, LED6 = ON) και ότι η γραμμή μεταφοράς έχει πιάσει την απαραίτητη ταχύτητα (LED2 = ON).

Στο σχήμα 7 φαίνεται ότι η βαλβίδα 5/2 βρίσκεται στη θέση Y2 (LED3 = ON), οι κινητήρες M1, M2 λειτουργούν (LED4 = ON, LED6 = ON) και ότι η γραμμή μεταφοράς έχει πιάσει την απαραίτητη ταχύτητα (LED2 = ON). Ακόμη, τα ποτενσιόμετρα που προσομοιώνουν τα B1, B2 είναι στραμμένα ώστε να δηλώνουν ότι το σιλό 1 είναι γεμάτο.

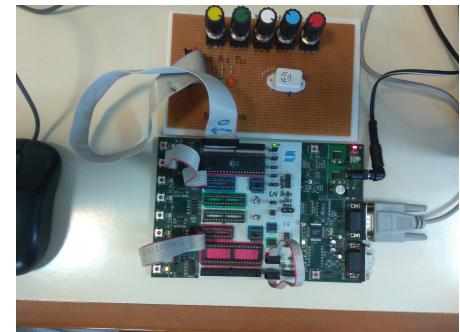
Στο σχήμα 8 φαίνεται ότι όλα τα ποτενσιόμετρα έχουν στραφεί ώστε να φαίνεται ότι τα σιλό είναι γεμάτα. Η συνθήκη αυτή προκαλεί την κατάσταση ALARM. Το παραπάνω γεγονός επιβεβαιώνεται από τη συγκεκριμένη φωτογραφία αφού το LED0 αναβοσβήνει.



Σχήμα 6: M1, M2 ON και Y1, B5 = 1.



Σχήμα 7: M1, M2 ON και Y2, B5 = 1.



Σχήμα 8: ALARM.

4 Επίλογος.

Συνοψίζοντας, στο έγγραφο αυτό παρουσιάστηκε η εργασία που πραγματοποιήθηκε στο πλαίσιο του εργαστηρίου του μαθήματος *Μικροεπεξεργαστές και Περιφερειακά*. Στην αναφορά αυτή, επεξηγήθηκε η λύση της εργασίας και οι παράγοντες που οδήγησαν στη λύση αυτή. Τέλος, το πρόγραμμα που υλοποιήθηκε δοκιμάστηκε στη δοκιμαστική πλατφόρμα του εργαστηρίου και απεδείχθη πλήρως λειτουργικό.

Αναφορές

[1] ATmega16 Datasheet. <http://ww1.microchip.com/downloads/en/devicedoc/doc2466.pdf>.