ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΟΙΚΙΝΩΝΙΩΝ

Αρχιτεκτονική Υπολογιστών Εργαστήριο

Τελική Εργασία Εξαμήνου

Στοιχεία Φοιτητή

Ονοματεπώνυμο: Κατόπης Δημήτριος

ΑΜ:2124

Email:int02124@uoi.gr

1η Άσκηση – Wiring Arduino

Σε αυτό το κύκλωμα χρησιμοποιούμε ένα Arduino το οποίο σε συνεργασία με έναν αισθητήρα Ultrasonic Sensor HC-Sr04 μετράει την απόσταση αντικείμενων . Σε συνδυασμό με 8 led , παίρνουμε πληροφορίες σχετικά με την απόσταση η οποία μετρείται σε εκατοστά και ανάλογα με την απόσταση αναβοσβήνουν τα led καθώς όσο μικραίνει η απόσταση αλλάζει ο ρυθμός με τον οποίο αναβοσβήνουν και ο προειδοποιητικός ήχος γίνεται όλο και πιο έντονος (με την βοήθεια ενός αισθητήρα piezo ) .

|  |  |
| --- | --- |
| Απόσταση | LED |
| 1-5 cm | LED 1 |
| 6-7 cm | LED 2 |
| 7-8 cm | LED 3 |
| 8-10 cm | LED 4 |
| 10-11 cm | LED 5 |
| 11-15 cm | LED 6 |
| 15-20 cm | LED 7 |
| >20 cm | LED 8 |

Χρησιμοποιούμε : 1. Arduino Uno

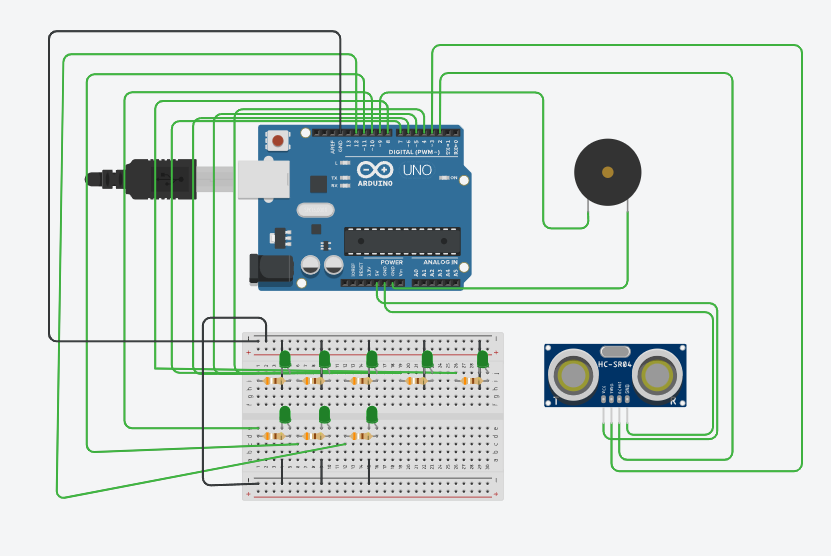
2. Breadboard

3.8 led

4. R=390Ω για κάθε λεντ

5. Ultrasonic sensor hc-sr04

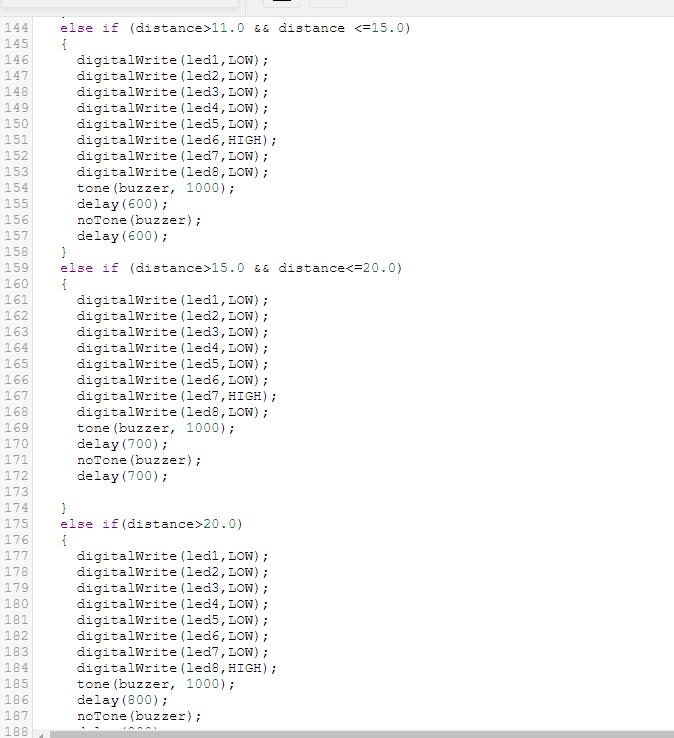
6. 1 piezo



Εικόνα που περιέχει κείμενο

Περιγραφή που δημιουργήθηκε αυτόματαΕικόνα που περιέχει κείμενο

Περιγραφή που δημιουργήθηκε αυτόματαΕικόνα που περιέχει πίνακας

Περιγραφή που δημιουργήθηκε αυτόματα

2η Άσκηση- Assembly Mips

Εδώ γράφουμε έναν κώδικα σε γλώσσα μηχανής (assembly mips ) το οποίο υλοποιεί μια αριθμομηχανή .

Η αριθμομηχανή κάνει τις εξής πράξεις

1.Πρόσθεση

2.Αφαίρεση

3.Πολλαπλασιασμό

4.Διαίρεση

5.Ύψωση σε δύναμη του 2 του 1ου αριθμού που έδωσε ο χρήστης

6.Ύψωση σε δύναμη του 2 του 2ου αριθμού που έδωσε ο χρήστης

7.Έξοδος

Ο χρήστης αρχικά εισάγει δύο αριθμούς και μετά μπορεί να επιλέξει ποια ενέργεια θα κάνει πατώντας τον κατάλληλο αριθμό.

# Προσθήκη δύο αριθμών απο τον χρήστη

        .data

                value: .word 0, 0, 0

                msg1:  .asciiz "Sum="

                msg2:  .asciiz "afairesi="

                msg3:  .asciiz "pollaplasioasmos="

                msg4:  .asciiz "Diairesi="

                msg5:  .asciiz "Ypsosi se dunami tou 2 ston 1o aritmo="

                msg55:  .asciiz "Ypsosi se dunami tou 2 ston 2o aritmo="

                msg6:  .asciiz "Lathos epilogi, dwse ksana\n"

                msg0: .asciiz "\nDwse 2 arithmous\n"

                msg00:  .asciiz "Dwse epilogi\n1.Προσθεση\n2.Αφαιρεση\n3.Πολλαπλασιασμος\n4.Διαιρεση\n5.Υψωση σε δυναμη του 2 του 1ου αριθμου\n6.Υψωση σε δυναμη του 2 του 2ου αριθμου\n7.EXIT\n"

        .text           # Μέρος text προγράμματος

        .globl main # Κύριο πρόγραμμα

main:

while:

        li $v0, 4

        la $a0, msg0

        syscall

        la $t0, value # Φόρτωση διεύθυνσης του value για αρχικοποίηση των καταχωρητών

        li $v0, 5         # Άμεση φόρτωση στο καταχωρητή $v0 την τιμή 5 (syscall για read\_int)

        syscall           # Κλήση syscall

        sw $v0, 0($t0)  # Αποθήκευση του περιεχομένου του καταχωρητή $v0 στην θέση 0 του καταχωρητή $t0

# Σε αυτό το σημείο έχουμε διαβάσει την πρώτη τιμή

        li $v0, 5         # Άμεση φόρτωση στο καταχωρητή $v0 την τιμή 5 (syscall για read\_int)

        syscall           # Κλήση syscall

        sw $v0, 4($t0)     # Αποθήκευση του περιεχομένου του καταχωρητή $v0 στην θέση 4 του καταχωρητή $t0

# TIP: αποθηκεύουμε +4 θέσεις επειδή η μνήμη μας είναι οργανωμένη σε byte

# Σε αυτό το σημείο έχουμε διαβάσει την δεύτερη τιμή

epilogi:

        li $v0, 4

        la $a0, msg00

        syscall

        la $t4,value

        li $v0,5

        syscall

        sw $v0,16($t4)

        lw $t5,16($t4)

        bne $t5,1,else

# Τώρα αρκεί να φορτώσουμε τα δεδομένα μας για να κάνουμε την πρόσθεση

        lw $t1, 0($t0)  # Φόρτωση των δεδομένων απο την θέση 0 του καταχωρητή $t0 και αποθήκευση αυτου στον καταχωρητή $t1

        lw $t2, 4($t0)  # Φόρτωση των δεδομένων απο την θέση 4 του καταχωρητή $t0 και αποθήκευση αυτου στον καταχωρητή $t2

        add $t3, $t1, $t2       # Πρόσθεση των τιμών των καταχωρητών $t1 και $t2 και αποθήκευση αυτου στο καταχωρητή $t3

        sw $t3, 8($t0)  # Αποθήκευση του περιεχομένου του καταχωρητή $t3 στην θέση 8 του καταχωρητή $t0

# Σε αυτο το σημείο έχει ολοκληρωθεί η πρόσθεση

# Εμφάνιση αποτελέσματος:

   li $v0, 4   # Άμεση φόρτωση στο καταχωρητή $v0 την τιμή 4 (syscall για print\_string)

   la $a0, msg1  # Φόρτωση της διέυθυνσης του msg1 στο καταχωρητή $a0 (argument to print\_string call)

   syscall  # Κλήση syscall

# Μέχρι εδώ έχουμε εμφανίσει το μήνυμα Sum=

   li $v0, 1   # Άμεση φόρτωση στο καταχωρητή $v0 την τιμή 1 (syscall για print\_int)

   move $a0, $t3 # Φόρτωση στο καταχωρητή $a0 το περιεχόμενο του $t3

   syscall     # Κλήση syscall

   j while

else:

        bne $t5,2,else1

        lw $t1, 0($t0)

        lw $t2, 4($t0)

        sub $t3, $t1, $t2

        sw $t3, 8($t0)

        li $v0, 4

        la $a0, msg2

        syscall

        li $v0, 1

        move $a0, $t3

        syscall

        j while

else1:

        bne $t5,3,else2

        lw $t1, 0($t0)

        lw $t2, 4($t0)

        mul  $t3, $t1, $t2

        sw $t3, 8($t0)

        li $v0, 4

        la $a0, msg3

        syscall

        li $v0, 1

        move $a0, $t3

        syscall

        j while

else2:

        bne $t5,4,else3

        lw $t1, 0($t0)

        lw $t2, 4($t0)

        div  $t3, $t1, $t2

        sw $t3, 8($t0)

        li $v0, 4

        la $a0, msg4

        syscall

        li $v0, 1

        move $a0, $t3

        syscall

        j while

else3:

        bne $t5,5,else4

        lw $t1, 0($t0)

        lw $t2, 4($t0)

        mul  $t3, $t1, $t1

        sw $t3, 8($t0)

        li $v0, 4

        la $a0, msg5

        syscall

        li $v0, 1

        move $a0, $t3

        syscall

        j while

else4:

        bne $t5,6,else5

        lw $t1, 0($t0)

        lw $t2, 4($t0)

        mul  $t3, $t2, $t2

        sw $t3, 8($t0)

        li $v0, 4

        la $a0, msg55

        syscall

        li $v0, 1

        move $a0, $t3

        syscall

        j while

else5:

        bne $t5,7,else6

        li  $v0, 10 # Άμεση φόρτωση στο καταχωρητή $v0 την τιμή 10 (syscall για exit)

        syscall  # Κλήση syscall

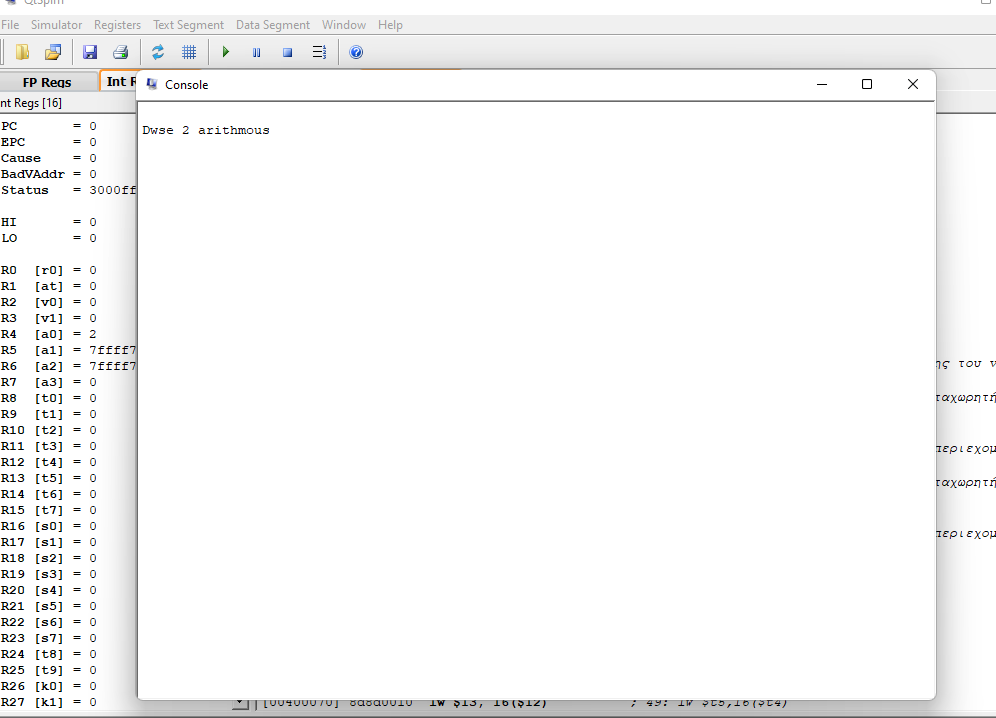
else6:  li $v0, 4

        la $a0, msg6

        syscall

        j epilogi

Μέσω του Qtspim τρέχουμε τον κώδικα



Εικόνα που περιέχει κείμενο

Περιγραφή που δημιουργήθηκε αυτόματα

