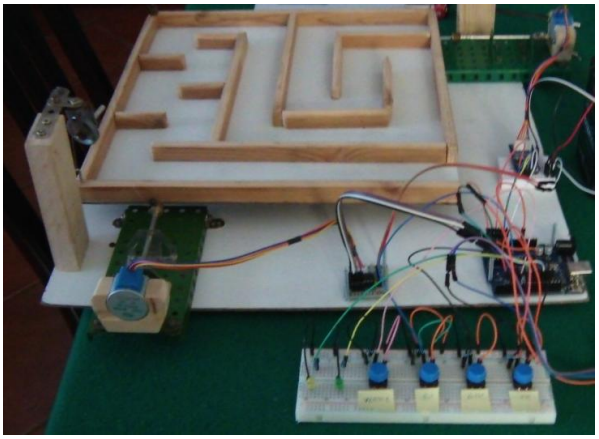
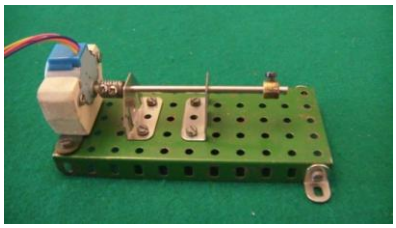
E1 – labirinto – la rivincita dell'esperienza – maze 1- the experience revenge (some notes at end of this section)



Poiche' l'esperienza spesso e' il ricordo dei fallimenti passati, si e' fatto tesoro del fallimento del precedente prototipo "E" (gui la scheda tecnica), per costruire un nuovo labirinto, con una nuova meccanica e soprattutto un nuovo percorso. Sono state abolite le buche, evitabili quasi solo grazie al caso, ed e' stata cambiata la pallina. Non piu' una pesante sfera in metallo ma una leggerissima pallina da ping pong, con una massa meno problematica. Sotto il profilo meccanico il labirinto e' un piano quadrato, con un vertice libero, uno vincolato e gli altri due collegati, tramite filo e carrucola, a due piccoli argani governati da arduino. L'inclinazione e' gestita dai due argani che, sfruttando il vertice vincolato ed il vertice libero, possono inclinare il labirinto in qualunque senso. Il programma e' dotato di una fase "tuning", e cioe' di un pilotaggio manuale degli argani, permessa al solo scopo di individuare le mosse da memorizzare in una tabella che sara' poi utilizzata da arduino. Operativamente il programma consente di manovrare manualmente i motori per portare in bolla il piano del labirinto dopodiche' si ferma in attesa che venga premuto il pulsante di ok. Una volta premuto il pulsante di ok, il sistema esegue le mosse eventualmente gia' memorizzate e poi si ferma di nuovo, azzera i contatori di passi ed inizia ad eseguire e ad evidenziare sul display i movimenti dei motori, richiesti tramite quattro pulsanti che governano le seguenti funzioni:



- seleziona il vertice su cui agire (vertice sinistro oppure destro)
- abbassa il vertice (un passo per ogni pressione del pulsante)
- alza il vertice (un passo per ogni pressione del pulsante)
- ok

Il pulsante di selezione funziona in flip/flop: ogni volta che viene premuto disattiva il vertice attivo e rende operativo il vertice non attivo. I led segnalano il vertice attivo (e cioe' quello su cui agisce il motore): il led verde segnala il vertice di destra mentre il led giallo quello di sinistra.

Grazie al sistema di esecuzione automatica delle mosse memorizzate, il programma, una volta completata la tabella delle mosse, e' in grado di pilotare in autonomia la pallina dal punto di partenza sino al punto di uscita, come d'altronde si vede nel video di presentazione.

Prima di procedere alla compilazione del programma deve essere installata, se non gia' presente, la seguente libreria:

- LiquidCrystal_I2C.h reperibile in https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads

Per installare la libreria e' necessario seguire la procedura illustrata nei precedenti progetti, e sintetizzabile in:

- download della libreria in formato complesso
- installare la nuova libreria andando in IDE-> sketch-> includes Library-> add .zip library
- verificare l'avvenuta installazione (andando in IDE-> sketch-> includes Library-> Contributed library)

Nota: Questo esercizio e questa nota sono parte di una serie che vede protagonisti Arduino ed alcuni dei componenti ad esso collegabili. Per la maggior parte degli esercizi e' anche disponibile un filmato su youtube.

- Esercizi facenti parte della raccolta
- Filmati presenti su youtube
- Informazioni su arduino e sui componenti collegabili (PDF scaricato nell'area di download)
- Breve manuale di programmazione (PDF scaricato nell'area di download)

Per eventuali chiarimenti o suggerimenti sul contenuto di questa scheda scrivere a giocarduino@libero.it

Here some notes about this project, translated by google translator 

Since the experience is often the memory of past failures, we treasured the failure of the previous prototype "E" (here the data sheet), to build a new maze, with a new mechanical and, above all, a new path. They were abolished the holes, avoidable only if lucky, and we changed also the ball. No longer an heavy metal ball, but a very light table tennis ball, with a less problematic mass.

The maze is a square plan, with a free corner, one bounded and the other two connected, via wire and pulley, to two small winches powered by Arduino.

The slope is managed by the two winches that, exploiting the captive summit and the free one, may tilt the maze in any direction. The program has a "tuning" phase: a manual control of the winches, allowed for the sole purpose of identifying the moves to be stored in an array that will then used by Arduino.

Operationally, the program allows you to manually maneuver the engines to bring the maze on level. After that program stops, waiting for the OK button pressing. After pressing, the system performs the already stored moves and then stops again and executes and displays the movement requested by four buttons that govern the following functions:

- Select the corner on which to act (left or right)
- Lowers the corner (a step for each pressing)
- Raises the corner (a step for each pressing)
- ok

The select button operates in flip / flop: each time you press deactivates the active corner and make operational the inactive one. The LEDs indicate the active corner (the one on which the motor acts): the green LED indicates the right corner while the yellow LED the left.

Due to the automatic execution of stored moves, the program, once the "moves" array is completed, is capable to drive independently the ball from the starting point up to the exit point, as also seen in the presentation video.

Before proceeding to program compilation must be installed, if not already done, the library:

- LiquidCrystal_I2C.h found here

For library installation, see process shown in previous projects, and summarized in:

- library download in compressed form;
- Installation via IDE-> sketch-> includes Library-> add .zip library
- After installation please verify the library. It must be present in IDE-> sketch-> includes Library-> Contributed library

Note: This project and this note is part of a series that sees, as main characters, Arduino and some of connectable components. For most projects there is also a video on youtube.

- Projects collection
- Movies on youtube
- About Arduino and components (italian; pdf will be downloaded in your download area
- Quick programming guide (almost english; pdf will be downloaded in your download area)

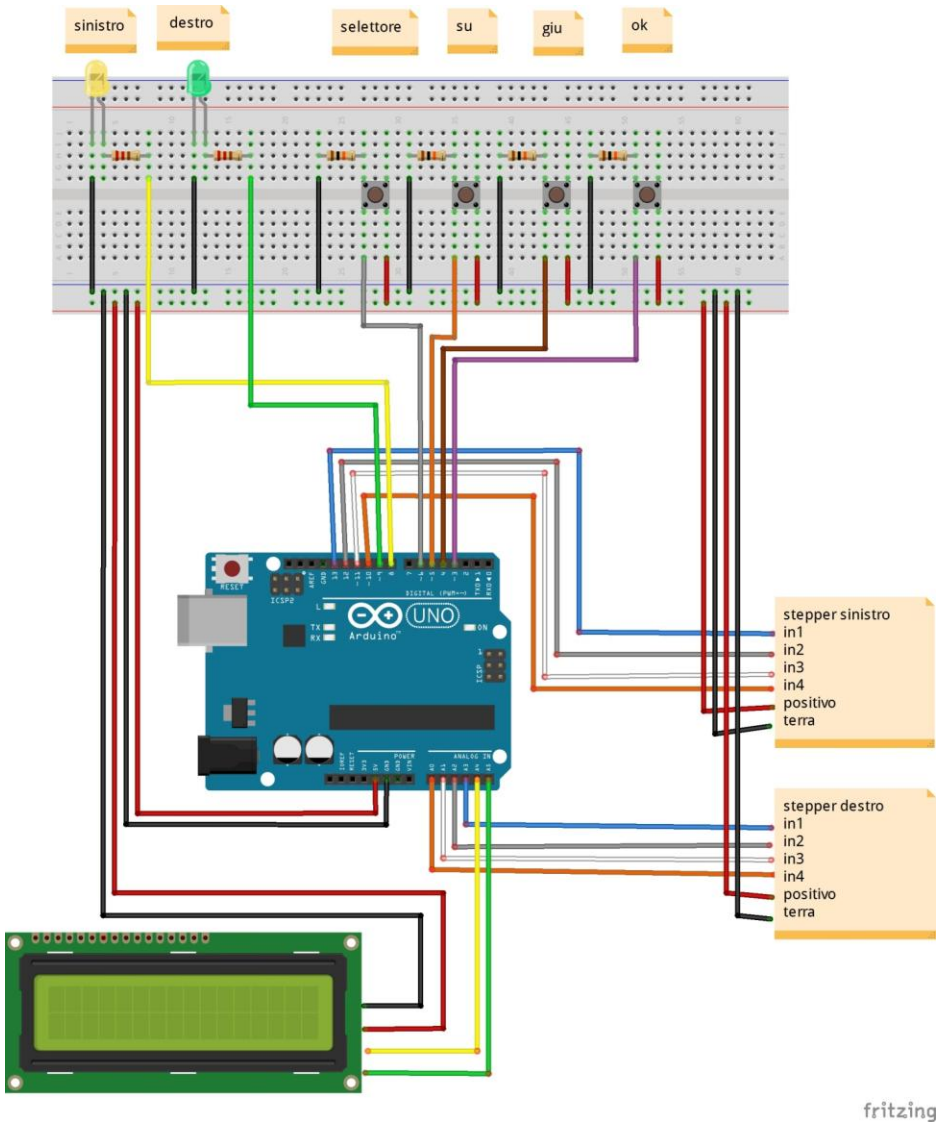
For any questions or suggestions about this note (and on its english translation), please write to giocarduino@libero.it (simple words and short sentences, please)

Materiali

Oltre alla base, al labirinto, agli argani (costruiti con il meccano) ed ai manicotti di collegamento tra motori e argani, sono necessari:

- due motori passo passo con relativi driver
- quattro pulsanti
- quattro resistenze da 10k ohm
- un display lcd a due porte (vedi esercizio 18 bis)
- due led (uno giallo ed uno verde)
- due resistenze da 220 ohm

Schema



Programma

```
/*  Attenzione: facendo il copia/incolla dal PDF all'IDE si perde la formattazione del testo.
* Per rendere piu' facilmente leggibile il programma e' opportuno formattarlo subito dopo il
* trasferimento nell'IDE, premendo CTRL+T.
*
* Questo programma pilota una pallina in un labirinto.
*
* il circuito e' formato da due motori passo passo che manovrano due vertici contrapposti del
* labirinto (il labirinto e' un quadrato con un vertice vincolato), un display lcd, quattro pulsanti
* e due led. Operativamente il sistema consente, in fase iniziale di manovrare i motori per portare
* il labirinto nella posizione di equilibrio dopodiche', alla pressione del pulsante di avvio esegue
* le mosse eventualmente gia' memorizzate e poi si ferma di nuovo in attesa di ulteriori comandi che
* possono essere immessi tramite quattro pulsanti che governano:
*  - la scelta del vertice su cui agire (destro oppure sinistro)
*  - aumento dei gradi di inclinazione
*  - diminuzione dei gradi di inclinazione
*  - ok
* Il programma consente quindi di individuare per ogni tratto del labirinto, la giusta inclinazione
* del piano e registrala su di un foglio per poi riportarla in una tabella (la tabella "mosse").
* Poiche' ad ogni avvio il programma esegue le mosse gia' memorizzate, una volta completata la
* tabella arduino e' in grado di pilotare la pallina fino all'uscita del labirinto.
*
*-----------------------------------------------------------------------------------------------
* Warning: cut&paste from PDF to IDE loses formatting. to restore it press CTRL + T.
*
* This program pilot a ball in a maze
*
* the circuit consists of two stepper motors that maneuver two opposite corners of a maze (the maze
* is a square with a constrained vertex), an LCD display, four buttons and two LEDs. Operationally,
* the system allows, in the initial phase, of maneuvering the motors to move the labyrinth in the
* equilibrium position after that, when you press the start button, executes the moves already
* stored and then stops again pending further commands that can be entered via four buttons for:
*  - Select the corner on which to act (left or right)
*  - Lowers the corner (a step for each pressing)
*  - Raises the corner (a step for each pressing)
*  - ok
*
* The program allows to identify for each section of the labyrinth, the right slope of the plane, to
* record it on a sheet and then bring it back in an array (the  "moves" array). Because each time
* you start, the program performs the already stored moves, once the array is completed, Arduino can
* drive the ball to the exit of the maze.
*-----------------------------------------------------------------------------------------------
*/
#include <Wire.h>                // libreria wire presente, di default, nell'IDE
#include <LiquidCrystal_I2C.h>  // libreria gestione display lcd  a due porte (vedi esercizio 18 bis
//  nota: i terminali SDA e SCL del display devono essere collegati, rispettivamente, alle porte
//  analogiche 4 e 5  di arduino
//--------------------addr, en,rw,rs,d4,d5,d6,d7,bl,blpol
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  // definisce la tipologia del
display

#include <Stepper.h>  // richiama la libreria con le routine di gestione del motore passo passo

Stepper stsinistro(32, 13, 11, 12, 10); // numero di impulsi per un giro dell'albero motore (32)
//                                      ed elenco delle porte di stsinistro, in sequenza di attivazione
/*  nota:  Le porte 13, 12, 11 e 10 di Arduino sono collegate, rispettivamente,  ai pin In1, In2,
In3 ed In4 del driver stsinistro, ma la sequenza di attivazione deve essere In1, In3, In2 ed In4
 per cui la dichiarazione delle porte e' 13, 11, 12, e 10
*/

Stepper stdestro (32, 14, 16, 15, 17); // numero di impulsi per un giro dell'albero motore (32)
// ed elenco delle porte di stdestro, in sequenza di attivazione
/*  nota: le porte 14, 15, 16 e 17 di Arduino (le porte analogiche da A0 ad A3) sono collegate,
rispettivamente, ai pin In1, In2, In3 ed In4 del driver stdestro, ma la sequenza di attivazione
deve essere In1, In3, In2 ed In4  per cui la dichiarazione delle porte e' 14, 16, 15, e 17
*/
//******************** definizione delle costanti ************************
//
int batti          = 3;  // il pulsante di ok e' collegato alla porta 3
int giu            = 4;  // il pulsante di diminuzione dei gradi e' collegato alla porta 4
int su             = 5;  // il pulsante di aumento dei gradi e' collegato alla porta 5
int lato           = 6;  // il pulsante di selezione del vertice (dx oppure sx) e' collegato alla
porta 6
int ledverde    = 9;  // il led verde, che indica operativita' sul vertice destro, e' collegato alla
porta 9
```

Arduino - labirinto 1: la rivincita dell'esperienza – maze 1: the experience revenge

```
int ledgiallo    = 8; // il led giallo, che indica operativita' sul vertice sinistro, e' collegato
alla porta 8
//
//****************definzione delle variabili ****************************
//
int statobatti            = 0; // variabile in cui viene inserito lo stato del pulsante ok
int statogiu              = 0; // variabile in cui viene inserito lo stato del pulsante giu
int statosu               = 0; // variabile in cui viene inserito lo stato del pulsante su
int statolato             = 0; // variabile di memorizzazione del vertice attivo: 1 = destro, 2 =
sinistro
int acquisiscilato        = 0; // variabile in cui viene inserito lo stato del pulsante "selettore"
int impulsidestra         = 0; // variabile in cui viene inserito il numero di impulsi per il motore
destro
int impulsisinistra       = 0; // variabile in cui viene inserito il numero di impulsi per il motore
sinistro
int totimpulsidestra      = 0; // variabile in cui viene inserito il totale impulsi per motore
destro
int totimpulsisinistra    = 0; // variabile in cui viene inserito il  totale impulsi per il motore
sinistro
int switchprimavolta     = 0; // semaforo per separare la fase di equilibratura dalla fase di tuning
int precsinistro          = 0; // variabile di memorizzazione del precedente valore sinistro
int precdestro            = 0; // variabile di memorizzazione del precedente valore destro
int moltiplicatore        = 1; // variabile per  un moltiplicatore di impulsi (per velocizzare le
mosse)
int lapse                 = 500; // variabile di memorizzazione del tempo di attesa dopo ogni mossa
int i                     = 0; // indice di scorrimento della tabella mosse
//
// ************************  tabella mosse  "moves" array *********************************
//sinistra,  destra,  delay
//leftcorner, rightcorner,delay
int mosse [91]          = {
 99,                          // il primo valore della tabella e' inutilizzato
  0,          18,       500, /* 0 impulsi sul vertice sx, 18 impulsi sul vertice dx e 1/2 secondo di
pausa  0 pulses on left corner, 18 pulses on right corner and half second pause */
-60,          18,       500, /* -60 sul vertice dx, 18 sul vertice dx e ½ secondo di pausa… -60 on
left corner, 18 on right corner and half second pause */
 60,         -17,      1500,
 80,         -17,      1500,
-45,         -17,      1000,
-45,          17,      1000,
-20,         -20,      1000,
-45,          17,      1000,
-20,         -20,      1000,
-45,          17,      1000,
-20,         -20,      5000,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,         0,
  0,           0,      2000,
  3,           0,         0,
};

//**
//********************* routine di pulizia del display ************************
//**
void puliscischermo (void)
{
  lcd.setCursor(0,0); // posiziona il cursore all'inizio della prima riga (carattere 0 e riga 0)
  lcd.print("                "); // inserire 16 spazi - insert 16 blanks
  lcd.setCursor(0,1); // posiziona il cursore all'inizio della seconda riga
  lcd.print("                "); // inserire 16 spazi - insert 16 blanks
  delay (100);
}
```

```
//**
//************** routine di azionamento dei motori - operating engines routine *********************
//**
void posizionaservo (void)
{
  digitalWrite (ledverde, HIGH);
  digitalWrite (ledgiallo, LOW);
  stdestro.step(impulsidestra*moltiplicatore);
  impulsidestra = 0;
  digitalWrite (ledverde, LOW);
  digitalWrite (ledgiallo, HIGH);
  stsinistro.step (impulsisinistra*moltiplicatore);
  impulsisinistra = 0;
  digitalWrite (ledgiallo, LOW);
  delay (lapse);
}
//***
//**************** routine di gestione dei lati - selecting corner routine ***********************
//**
void gestionelato (void)
{
  // ****** individua il vertice sul quale agire **********
  acquisiscilato = digitalRead (lato);  // acquisisce il segnale di settaggio del vertice
  if (acquisiscilato == 1)
  {
    if(statolato == 1)                    // se il precedente statolato era 1 (era destro)
    {
      statolato = 2;                      // posiziona su vertice sinistro
      digitalWrite (ledgiallo, HIGH);     // accende il led giallo (opera su vertice sinistro)
      digitalWrite (ledverde, LOW);
      delay (500);
    }
    else                                  // se invece il precedente statolato era 2 (sinistro)
    {
      statolato = 1;                      // posiziona su vertice destra
      digitalWrite (ledverde, HIGH);      // accende il led verde (opera su vertice destro)
      digitalWrite (ledgiallo,LOW);
      delay (500);
    }
  }
  // ******** diminuzione dei gradi di inclinazione - lowers the active corner *********
  statogiu = digitalRead (giu);          // acquisisce lo stato del pulsante giu
  if (statogiu == 1)                     // se e' stato premuto il pulsante giu
  {
    if (statolato == 1)
    {
      impulsidestra = -1;
      totimpulsidestra--;
    }
    if (statolato == 2)
    {
      impulsisinistra = -1;
      totimpulsisinistra--;
    }
  }
  // ********* aumento dei gradi di inclinazione - raises the active corner ************
  statosu = digitalRead (su);           // acquisisce lo stato del pulsante su
  if (statosu == 1)                     // se e' stato premuto il pulsante su
  {
    if (statolato == 1)
    {
      impulsidestra = 1;
      totimpulsidestra++;
    }
    if (statolato == 2)
    {
      impulsisinistra = 1;
      totimpulsisinistra++;
    }
  }
}

//**
//************ routine di visualizzazione impulsi - display pulses routine ***********************
//**
void visualizzaimpulsi (void)
{
  lcd.setCursor(0,0); // posiziona il cursore all'inizio della prima riga
```

```
  lcd.print ("destra = ");
  lcd.print (totimpulsidestra);
  lcd.print (" ");
  lcd.setCursor(0,1); // posiziona il cursore all'inizio della seconda riga
  lcd.print ("sinistra = ");
  lcd.print (totimpulsisinistra);
  lcd.print (" ");
  delay (100);
}
//
//
void setup()
{
  lcd.begin(16,2);              // inizializza il display (16 caratteri per due righe)
  stdestro.setSpeed(500);       // imposta una media velocita' di rotazione dell'albero
  stsinistro.setSpeed(700);     // dei due motori passo passo: 500 impulsi al secondo, pari a
  // 3,125 giri al secondo dell'albero motore e ad una rotazione di 17,5 gradi al secondo
  // del perno in uscita (700 per il motore di sinistra, il cui driver e' meno performante)
  pinMode (batti,    INPUT);    // definisce il pulsante "ok" come unita' di input
  pinMode (giu,      INPUT);    // definisce il pulsante giu come unita' di input
  pinMode (su,       INPUT);    // definisce il pulsante su come unita' di input
  pinMode (lato,     INPUT);    // definisce il pulsante su come unita' di input
  pinMode (ledverde,  OUTPUT); // definisce il led verde come unita' di output
  pinMode (ledgiallo, OUTPUT); // definisce il led giallo come unita' di output
  puliscischermo ();            // lancia la routine di pulizia dello schermo
  lcd.setCursor(0,0);           // posiziona il cursore all'inizio della prima riga
  lcd.print ("metti in piano");
  lcd.setCursor(0,1);           // posiziona il cursore all'inizio della seconda riga
  lcd.print("e premi ok ");
  moltiplicatore = 64;  // inserisce un valore alto nel moltiplicatore, per velocizzare i movimenti
  while (switchprimavolta == 0) // loop di messa in piano del labirinto
  {
    gestionelato();             //lancia la routine di gestione del vertice
    if ((impulsidestra != 0) || (impulsisinistra != 0))  // verifica se c'e qualcosa da fare
      posizionaservo ();   //  se e' cambiata qualche impostazione aziona i motori
    statobatti = digitalRead (batti);
    if (statobatti == HIGH)
    {
      switchprimavolta    = 1;  // setta lo switch primavolta in modo da non ripetere piu' questo
ciclo
      statobatti          = 0;  // rimette a zero il valore del pulsante batti
      totimpulsisinistra  = 0;  // azzera il registro sinistra
      totimpulsidestra    = 0;  // azzera il registro destra
      delay (500);
    }
  }
  puliscischermo ();
  precsinistro = 0;
  precdestro   = 0;
  moltiplicatore = 16;
  //***** esecuzione delle mosse memorizzate in tabella - execute the stored moves ***********
  for (i = 0; i <=30 ; i++)
  {
    lcd.setCursor(0,0);
    lcd.print ("mossa ");
    lcd.print (i);
    lcd.setCursor(0,1);
    lcd.print ("                ");
    lcd.setCursor(0,1);
    lcd.print ("sx: ");
    lcd.print (mosse [i*3+1]);
    lcd.print ("dx: ");
    lcd.print (mosse [i*3+2]);
    delay (100);
    impulsidestra  = mosse [i*3+2] - precdestro;
    precdestro     = mosse [i*3+2];
    impulsisinistra= mosse [i*3+1] - precsinistro;
    precsinistro   = mosse [i*3+1];
    lapse          = mosse [i*3+3];
    posizionaservo ();
  }
  precdestro = 0;
  precsinistro = 0;
  lapse = 50;
  puliscischermo ();
}
//
//
```

```
void loop()
{
  gestionelato();    //lancia la routine di gestione dei vertici
  lapse = 500;
  if ((impulsisinistra != 0) || (impulsidestra != 0) )  // verifica se c'e qualcosa da fare
    posizionaservo ();   //  aziona i motori se e' cambiata qualche impostazione
  visualizzaimpulsi ();
}
```