Reaction Game 2.0

Titolo del progetto: Reaction Game 2.0
Alunni: Dyuman Bulloni, Erik Stalliviere
Classe: Info 3
Anno scolastico: 2017/2018
Docente responsabile: Francesco Mussi

1 Introduzione

1.1 Informazioni sul progetto

Allievi:
- Dyuman Bulloni, Responsabile Software, Responsabile Organizzazione
- Erik Stalliviere, Responsabile Hardware

Docente Responsabile: Francesco Mussi.
Scuola: Scuola d'Arti Mestieri Trevano.
Sezione: Informatica.
Materia: Modulo 306.
Data Inizio Progetto: 16.03.2018
Data Consegna Progetto: 18.05.2018

1.2 Abstract

This documentation explains how continue the project "ReactionGame" and finish the main elements. In particular, the two seven segments displays, the new modalities and the php code will be show step by step.

1.3 Scopo

Lo scopo di questo progetto consiste nel riprendere il punto di arrivo della prima versione del progetto ReactionGame (che aveva lo scopo di ricreare una BATAK machine, un gioco per allenare i propri riflessi, maggiori dettagli nella versione 1.0 del progetto) e portare a compimento tutti gli elementi mancati o non completati appunto in questa prima versione del lavoro. Le cose che con questo progetto dobbiamo portare a termine sono l'aggiunta dei due display sette segmenti per la visualizzazione di tempo e punteggio, una pagina di login che comprende una classifica dei punteggi per modalità, il completamento di tutte e 23 le modalità inizialmente previste e l'implementazione di una pedana di gioco.

Analisi

1.4 Analisi del dominio

Per maggiori informazioni, vedere il seguente paragrafo nella documentazione della versione 1.0 del progetto.

1.5 Analisi dei costi e benefici

1.5.1.1 Versione 1.0

Categoria	**Costo**
Telaio BATAK	700 CHF
12 Bottoni led	70 CHF
Display 7 segmenti	2 CHF x2
Display LCD	10 CHF
Cavi	10 CHF
Arduino MEGA	40 CHF
Ethernet Shield 2	15 CHF
Personale	94 ore * 4 persone * 50 CHF = 18 800
TOT	**19 649 CHF**

1.5.1.2 Versione 2.0

Categoria	Costo
Arduino UNO	20 CHF
Personale	72 ore * 2 persone * 50 CHF = 7 200
TOT PARZIALE	**7 220 CHF**
TOT V 1.0 + V 2.0	**26 869 CHF**

Analisi e specifica dei requisiti

ID: REQ-01	
Nome	Vecchie funzioni, Software
Priorità	1
Versione	2.0
Note	
Sotto requisiti	
001	Le modalità mancanti (5,6,9,10,20,21,22,23) devono essere implementate e integrate perfettamente col codice principale.
002	La classifica dei punteggi per modalità deve essere implementata e funzionante.

ID: REQ-02	
Nome	Vecchie funzioni, Hardware
Priorità	1
Versione	2.0
Note	
Sotto requisiti	
001	I due display 7 segmenti devono essere correttamente collegati e mostrare gli elementi descritti nelle varie modalità (solitamente tempo e punteggio).
002	La pedana deve essere analizzata, trovata la soluzione ideale da applicare e agire completandola nella pratica.

ID: REQ-03	
Nome	Nuove funzioni
Priorità	1
Versione	2.0
Note	
Sotto requisiti	
001	Il collegamento tra Arduino e PC non sarà più tramite ethernet, ma sviluppato tramite un Fishino con il collegamento Wireless.

1.6 Pianificazione

1.7 Analisi dei mezzi

1.7.1 Software

Arduino 1.8.1.10
Fritzing 0.9.3b
Microsoft Word 2013
Gantt Project 2.8.5
PowerPoint 2013
HeidiSQL
Notepad++ 7.5
Google Chrome 64.0
SourceTree
WinRAR 5.50
Foxit Reader 8.3.1

1.7.2 Hardware

2 PC, uno per collaboratore
Arduino MEGA.
Arduino UNO
Shield Ethernet 2
2 display 7 segmenti
1 display LCD
1 Pedana

Per gli elementi necessari alla costruzione del telaio, consultare la documentazione 1.0 del progetto.

2 Progettazione

2.1 Design dei dati e database

Per la visualizzazione del design utilizzato, consultare la documentazione 1.0 del progetto.

2.2 Design delle interfacce

Qui di seguito verranno mostrate e spiegate tutte le modalità attraverso un piccolo schema con relativa spiegazione.
Nella Batak i numeri a destra sono pari (0, 2, 4, 6, 8) e a sinistra i dispari (1, 3, 5, 7, 9) quello sopra "#", mentre quello sotto è "@". Il riquadro in basso corrisponde al display LCD, non così fondamentale da dover essere descritto.

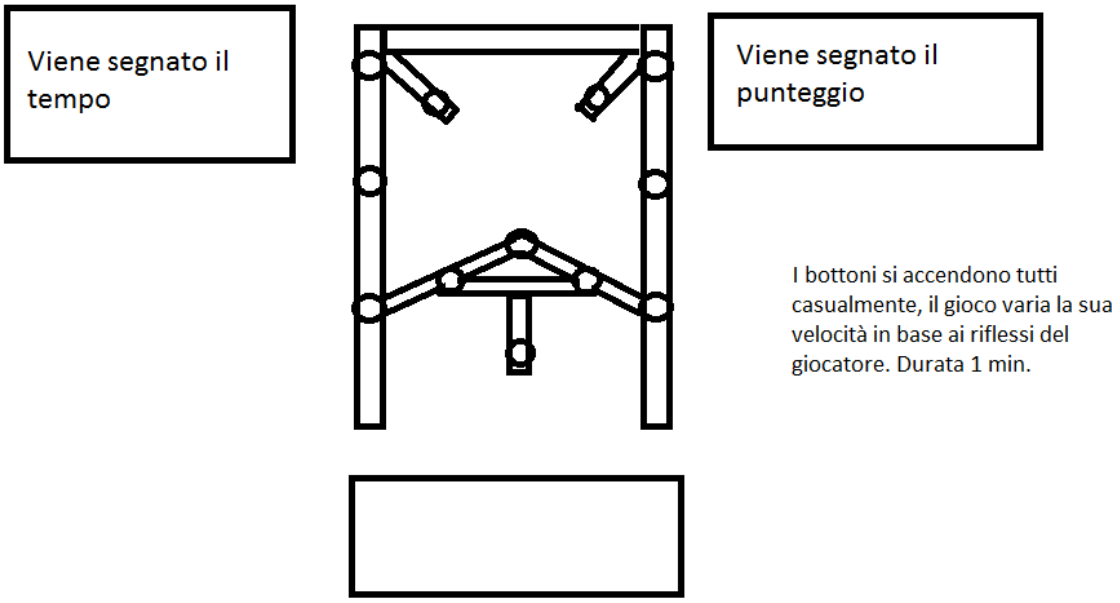


Figura 1: Modalià 1, Cumulativo 60[s], senior

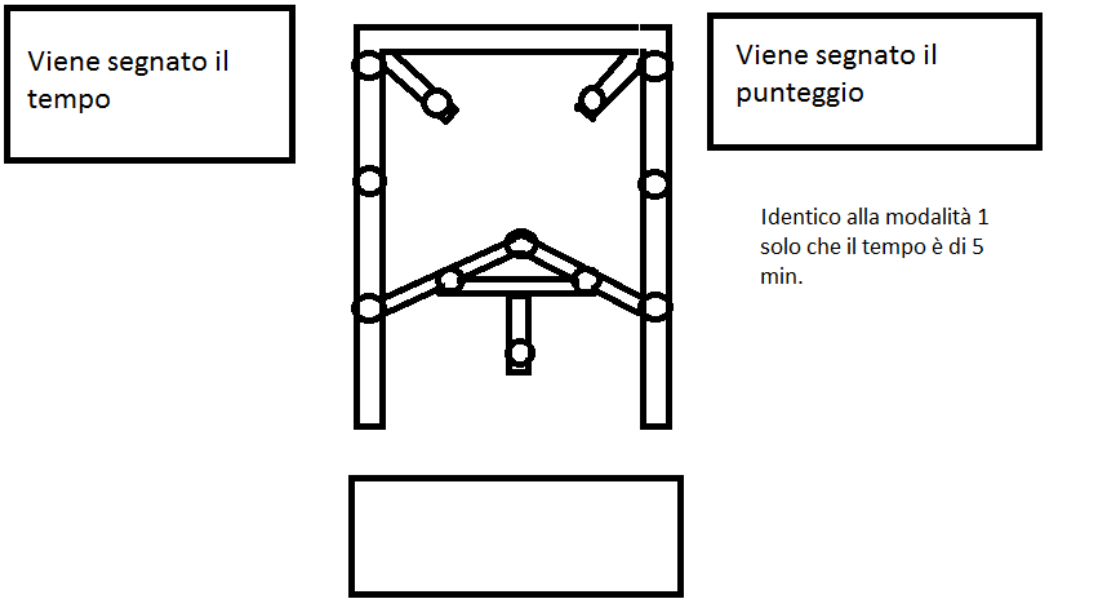


Figura 2: Modalità 2, Maratona 300[s], senior

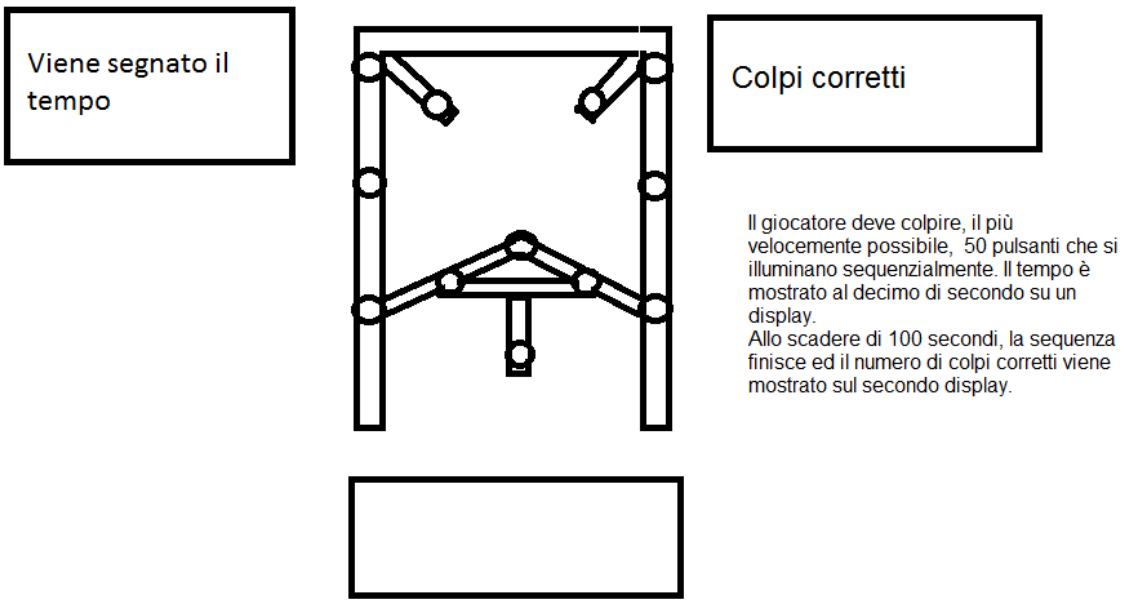


Figura 3: Modalità 3, Corsa 50 pulsanti, senior

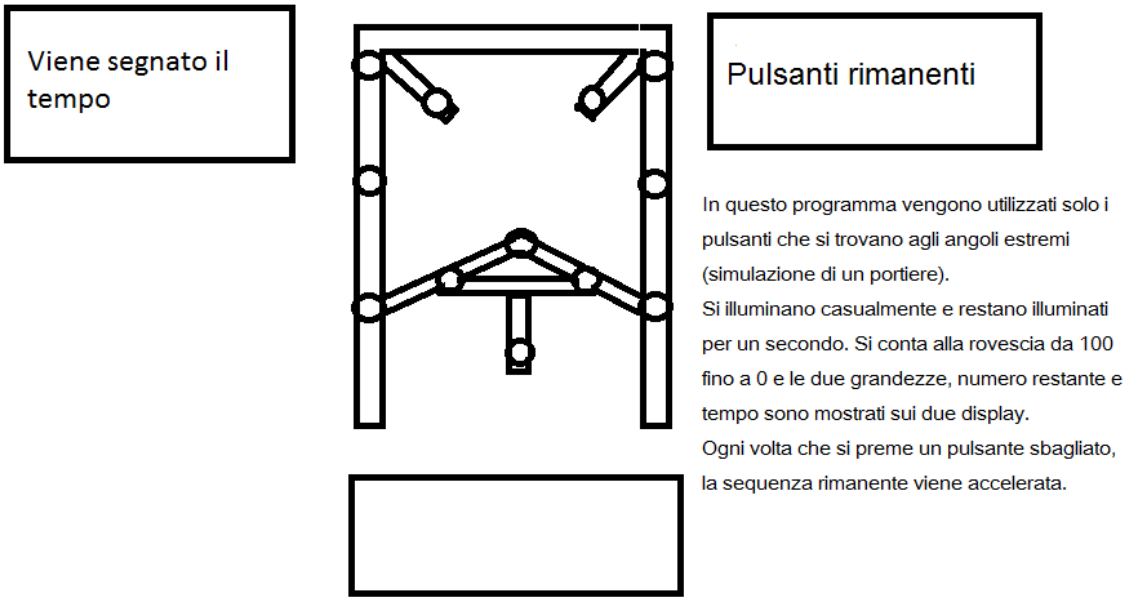


Figura 4: Modalità 4, Stretching angolare (100 pulsanti), senior

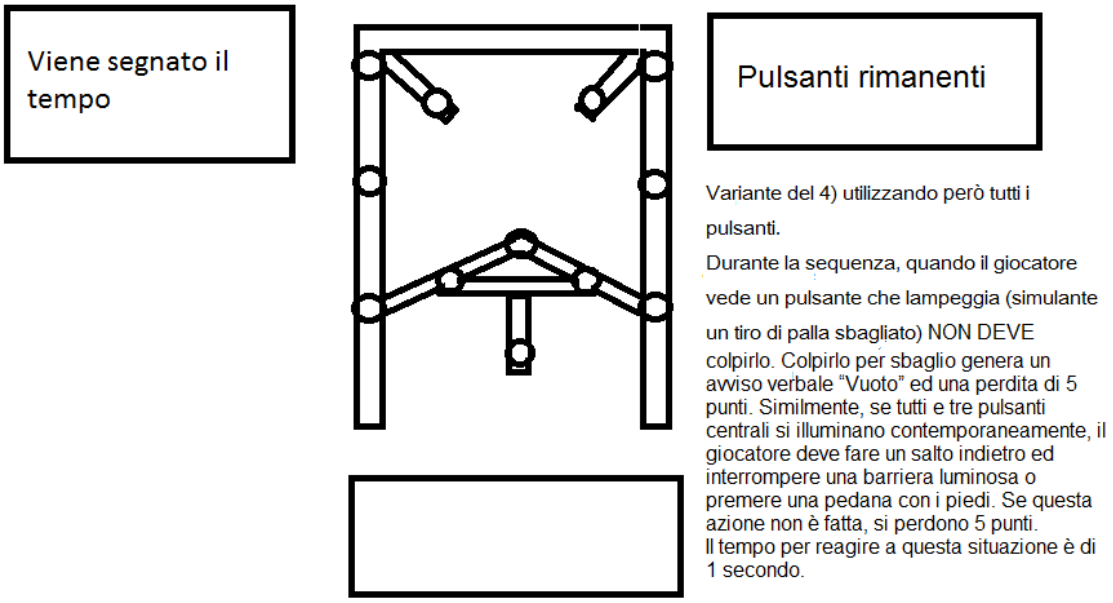


Figura 5: Modalità 5, senior

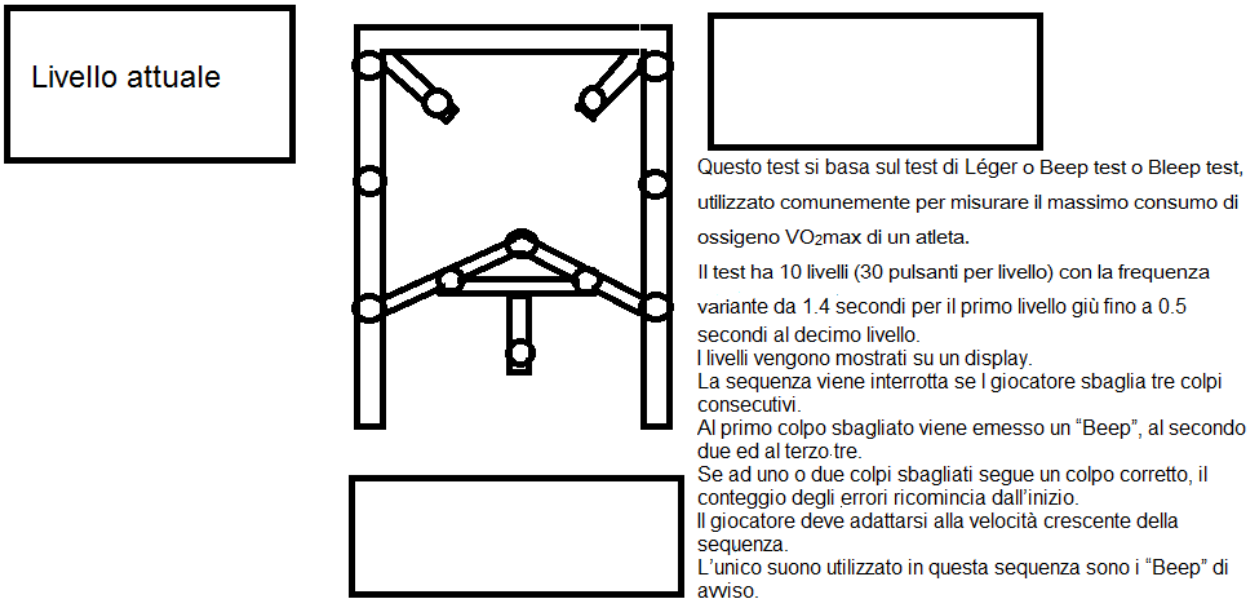


Figura 6: Modalità 6, Test di Legéro o Bleep test, senior

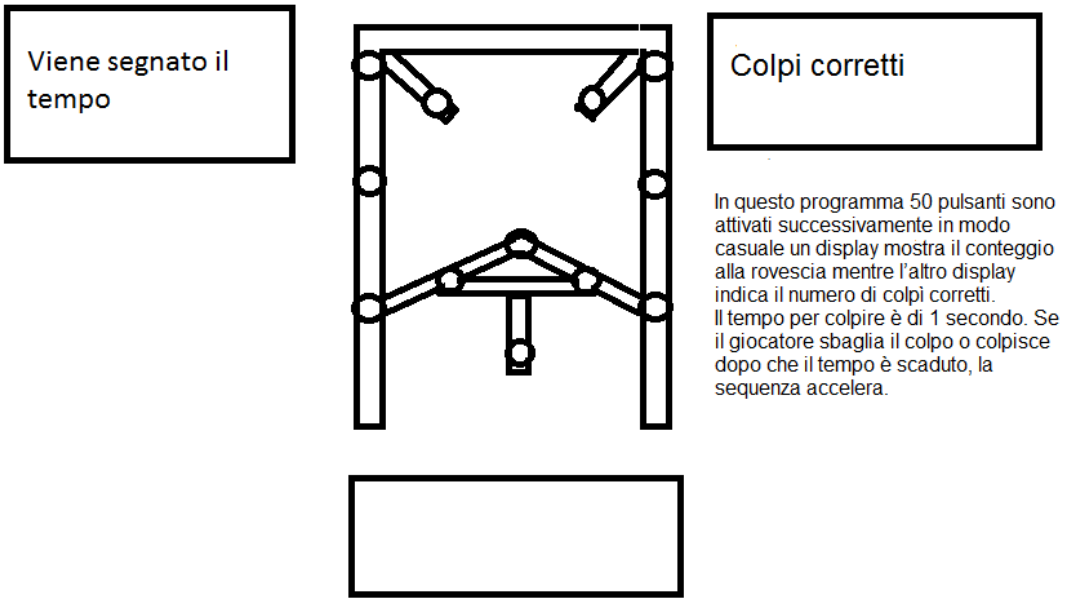


Figura 7: Modalità 7, 50 pulsanti temporizzati, senior

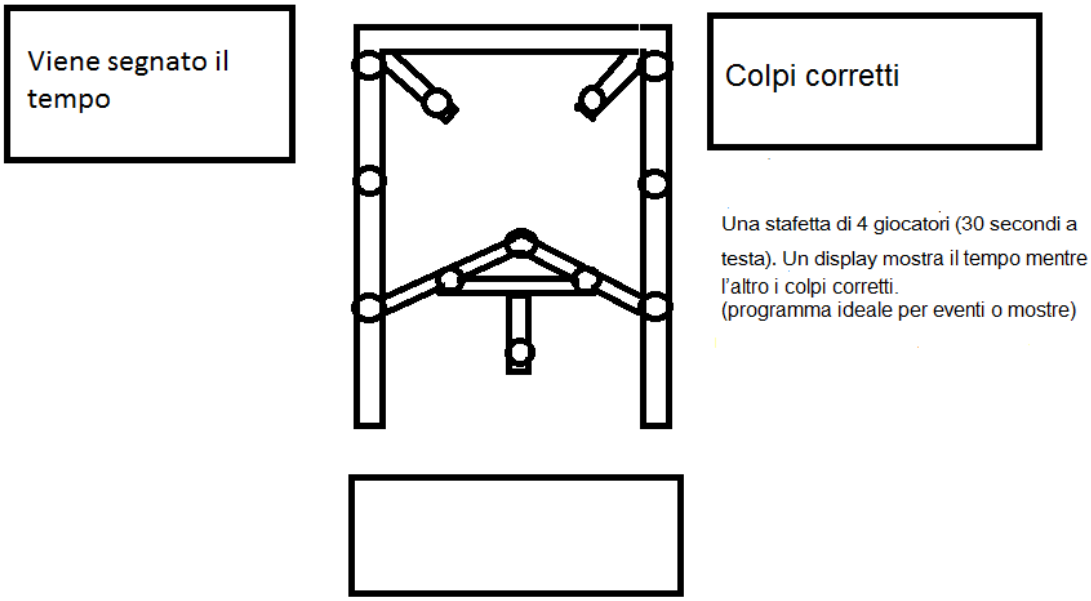


Figura 8: Modalità 8, Staffetta 4 giocatori, senior

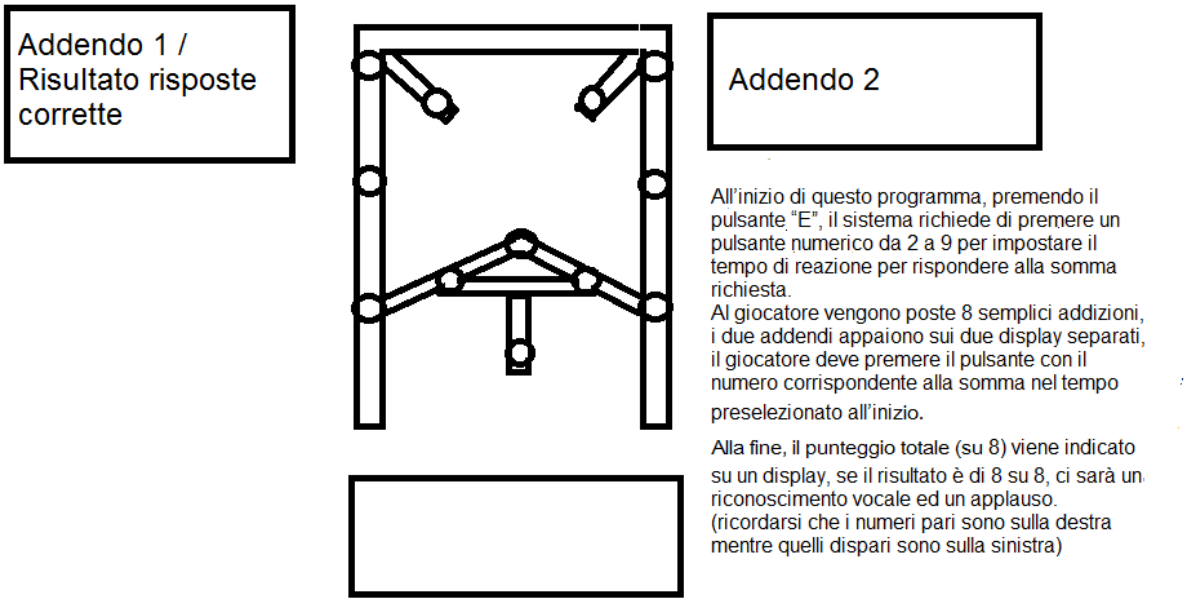


Figura 9: Modalità 9, Reazione somma matematica

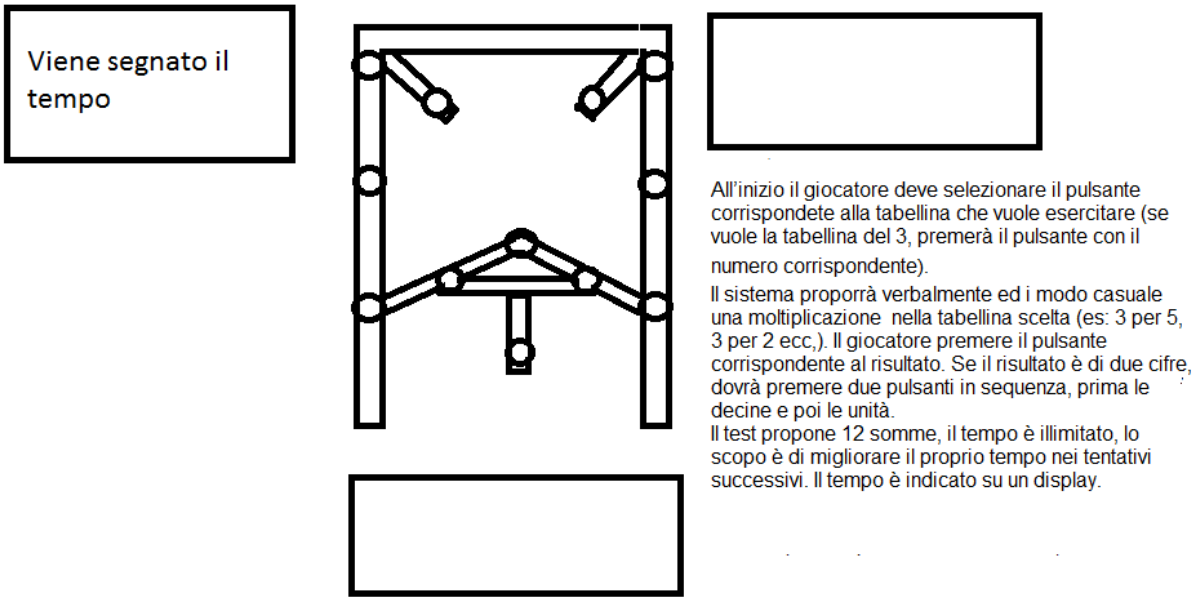


Figura 10: Modalità 10, Tabelline test velocità

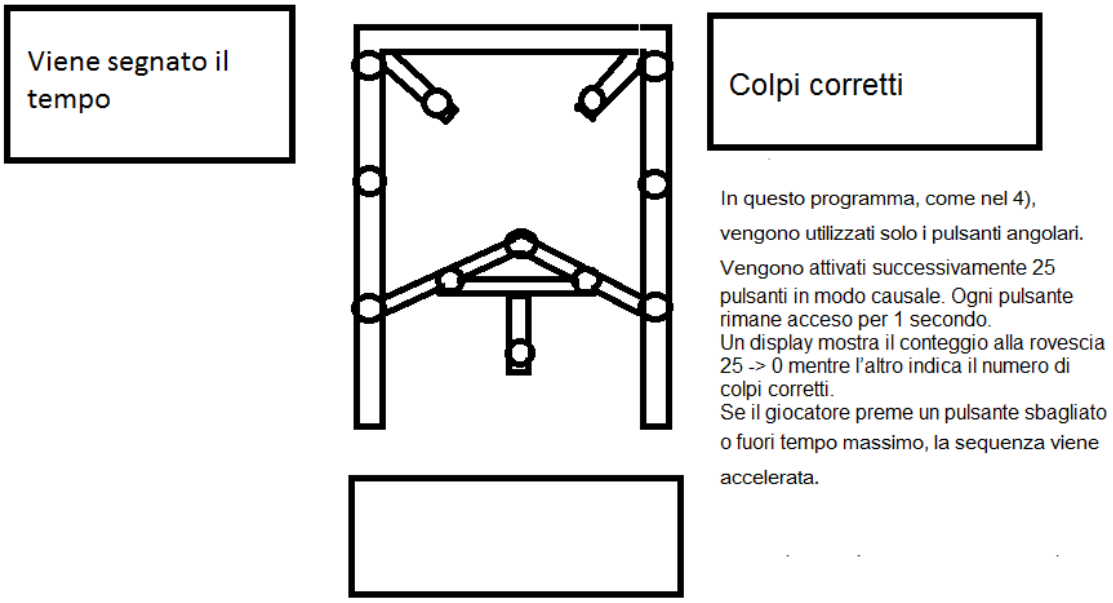


Figura 11: Modalità 11, Stretching angolare (25 pulsanti)

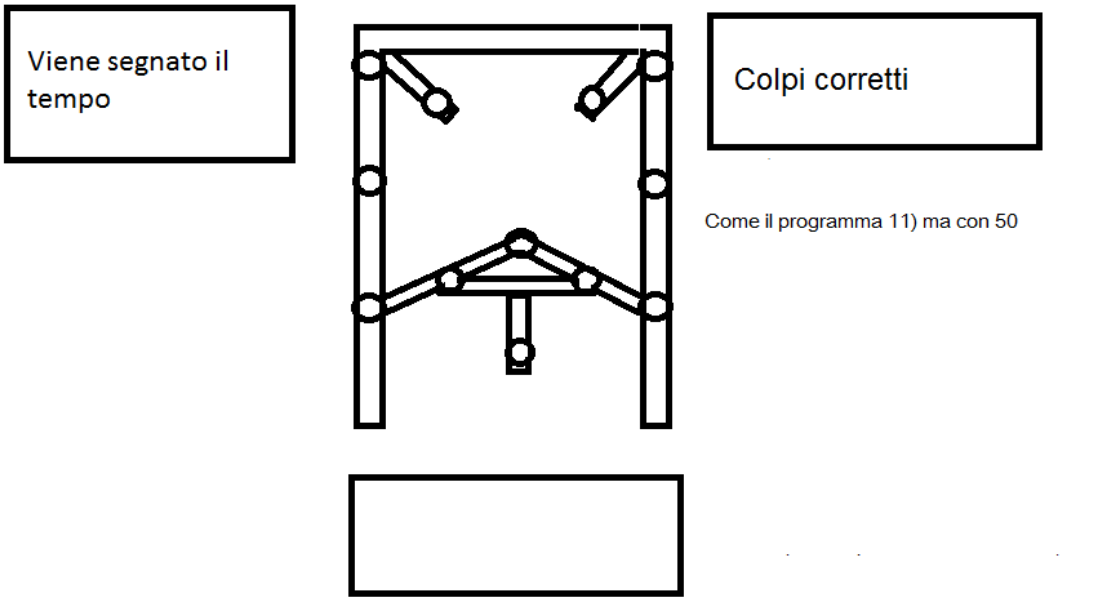


Figura 12: Modalità 12, Stretching angolare (50 pulsanti)

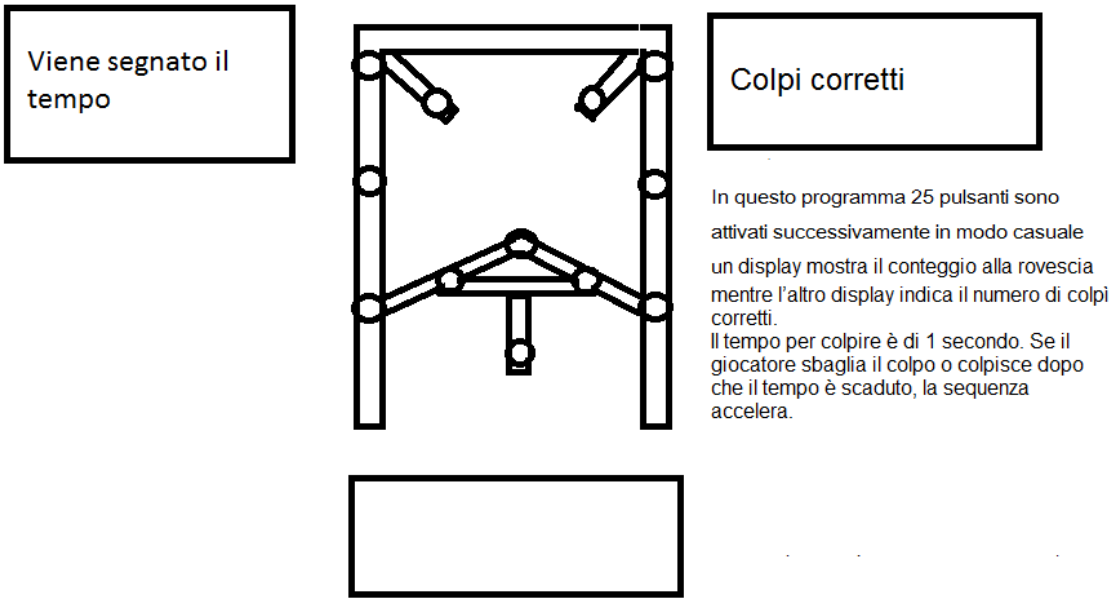


Figura 13: Modalità 13, 25 pulsanti temporizzati, junior

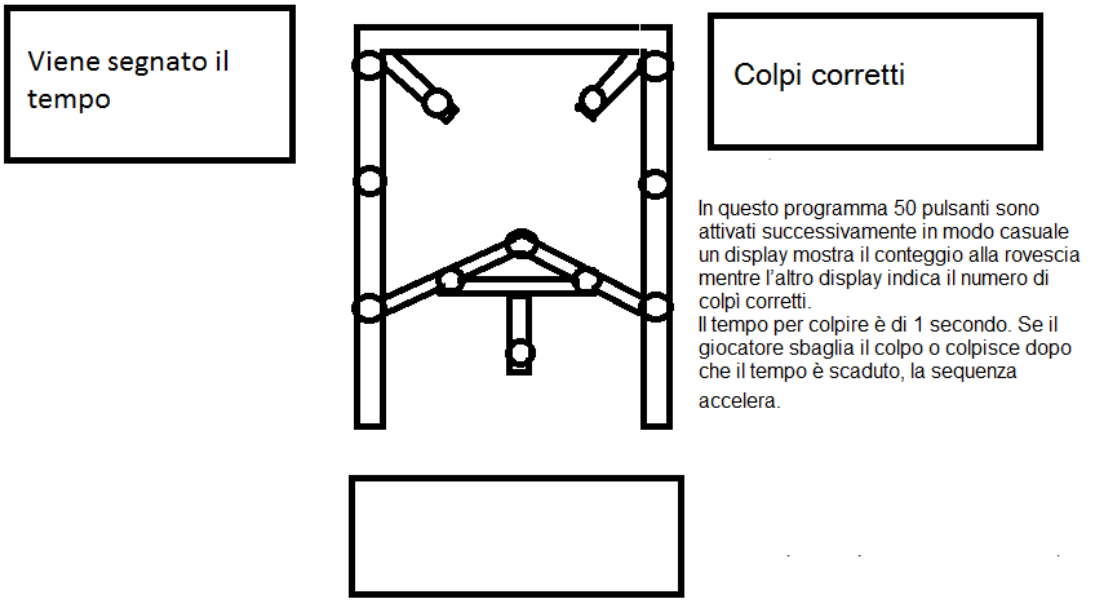


Figura 14: Modalità 14, 50 pulsanti temporizzati, junior

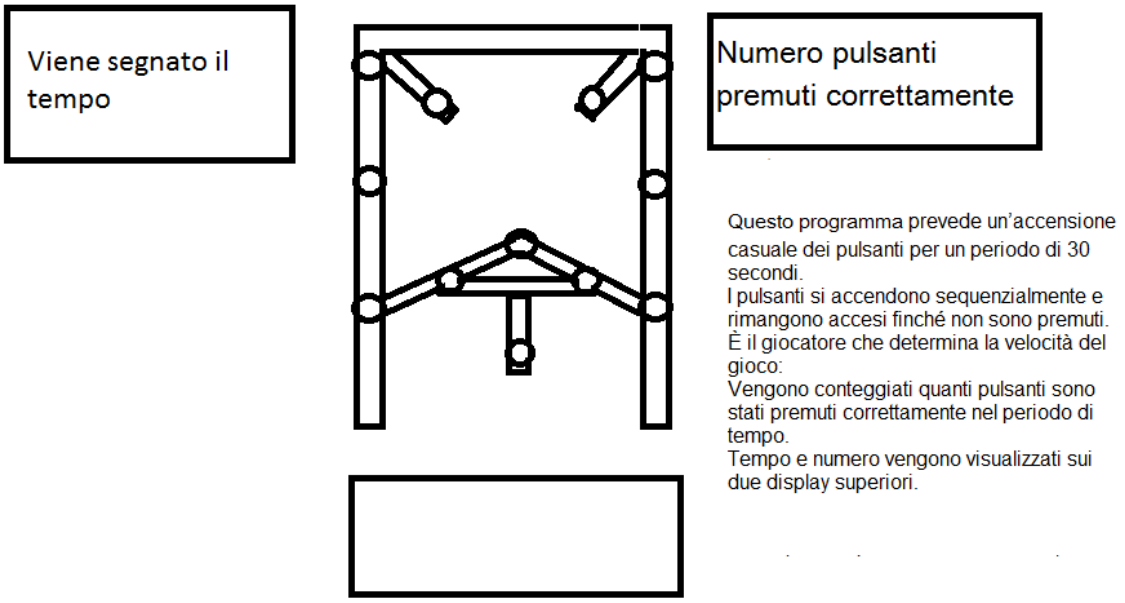


Figura 15: Modalità 15, Cumulativo 30[s], junior

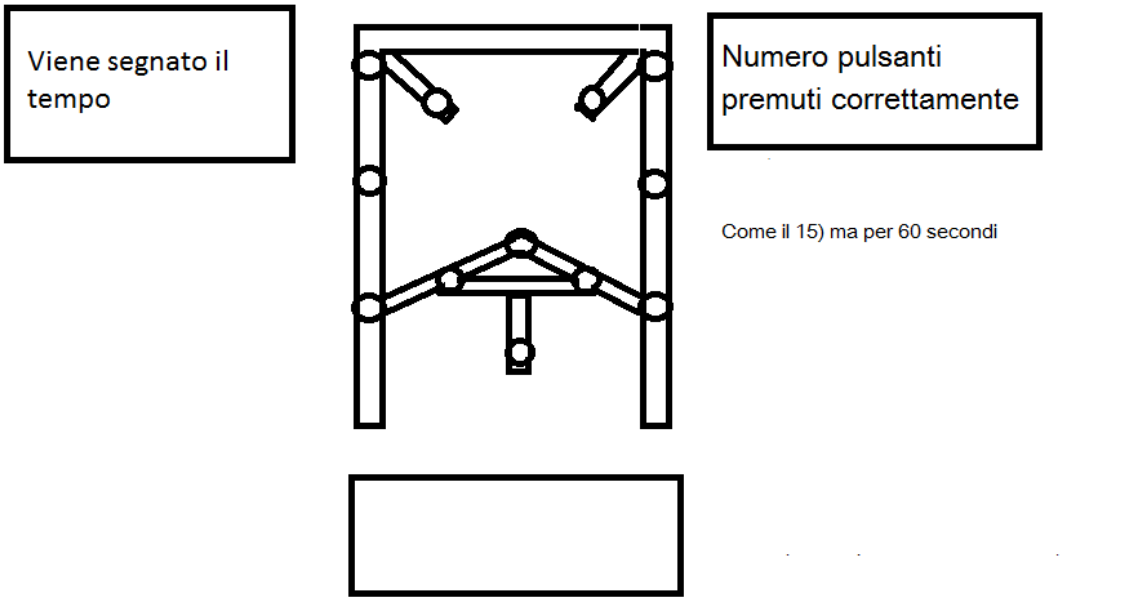


Figura 16: Modalità 16, Cumulativo 60[s], junior

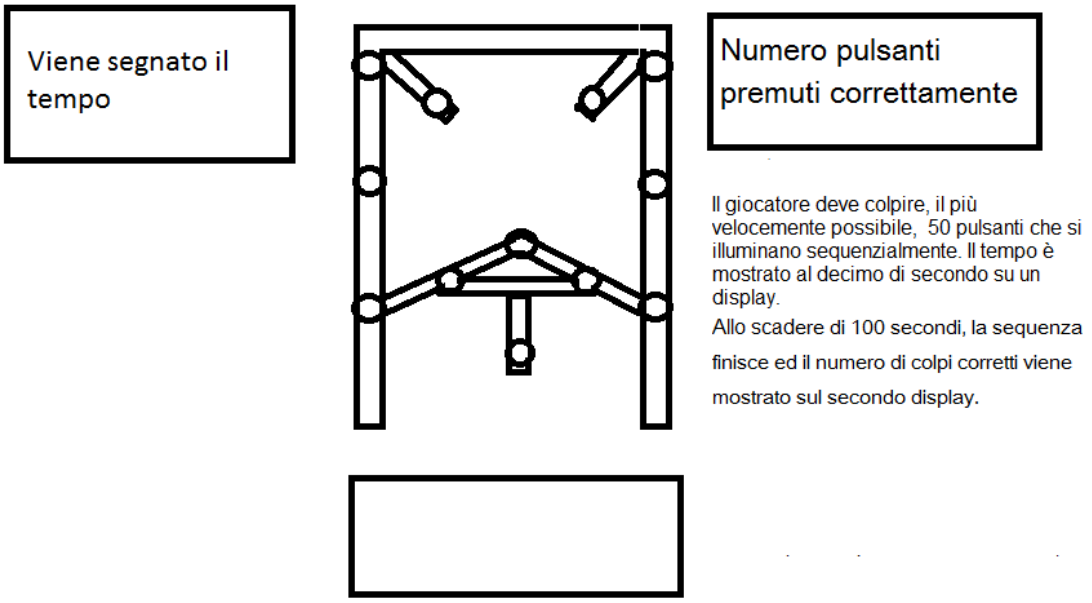


Figura 17: Modalità 17, Corsa 25 pulsanti, junior

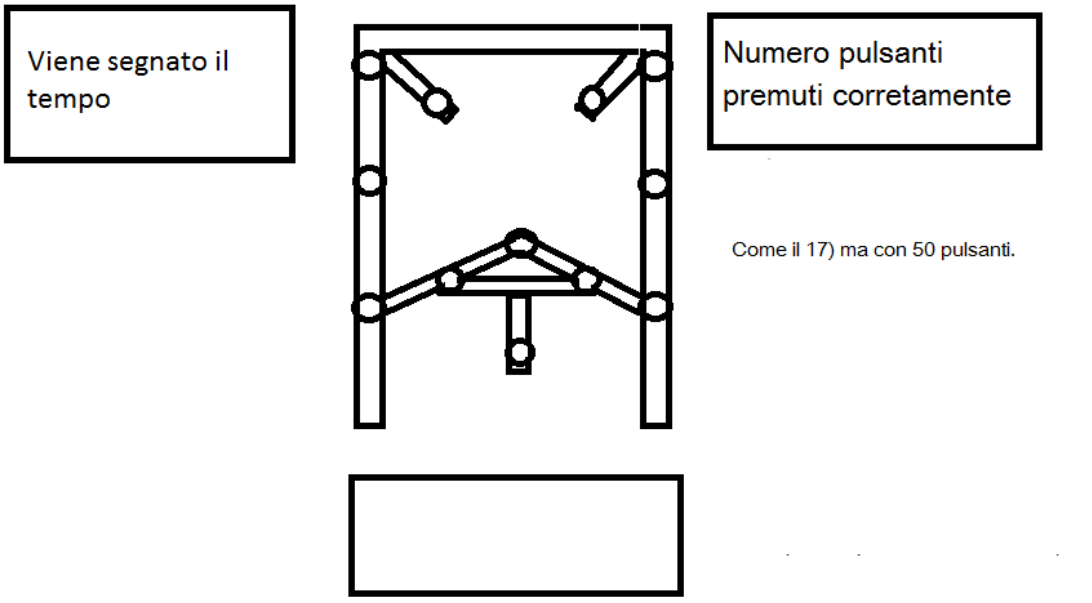


Figura 18: Modalità 18, Corsa 50 pulsanti, junior

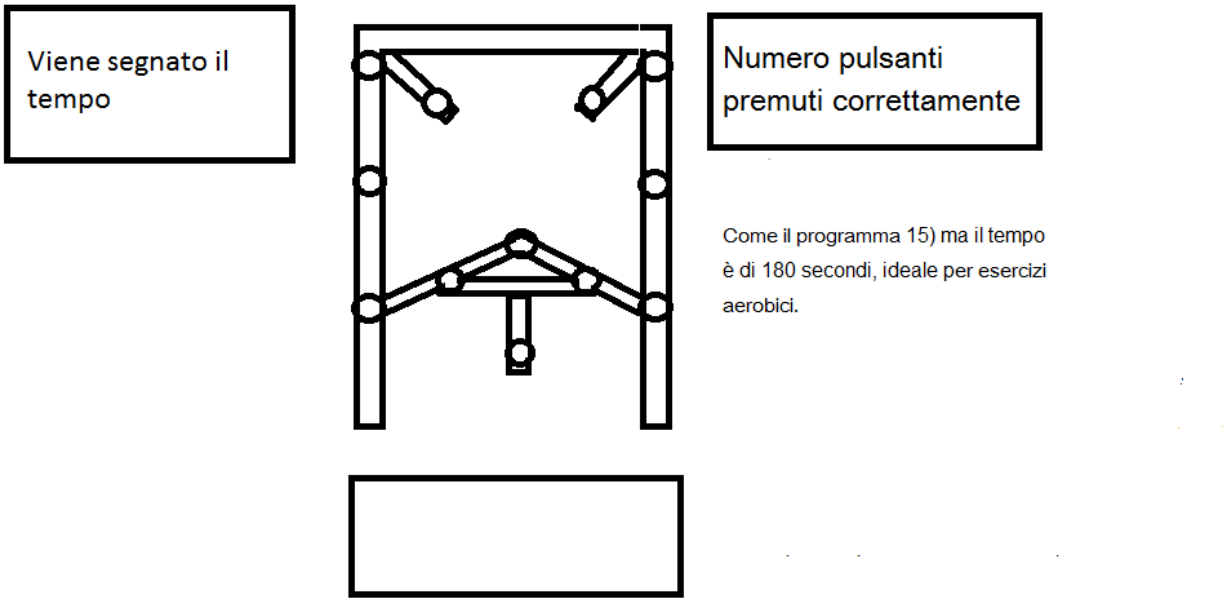


Figura 19: Modalità 19, Maratona 180[s], junior

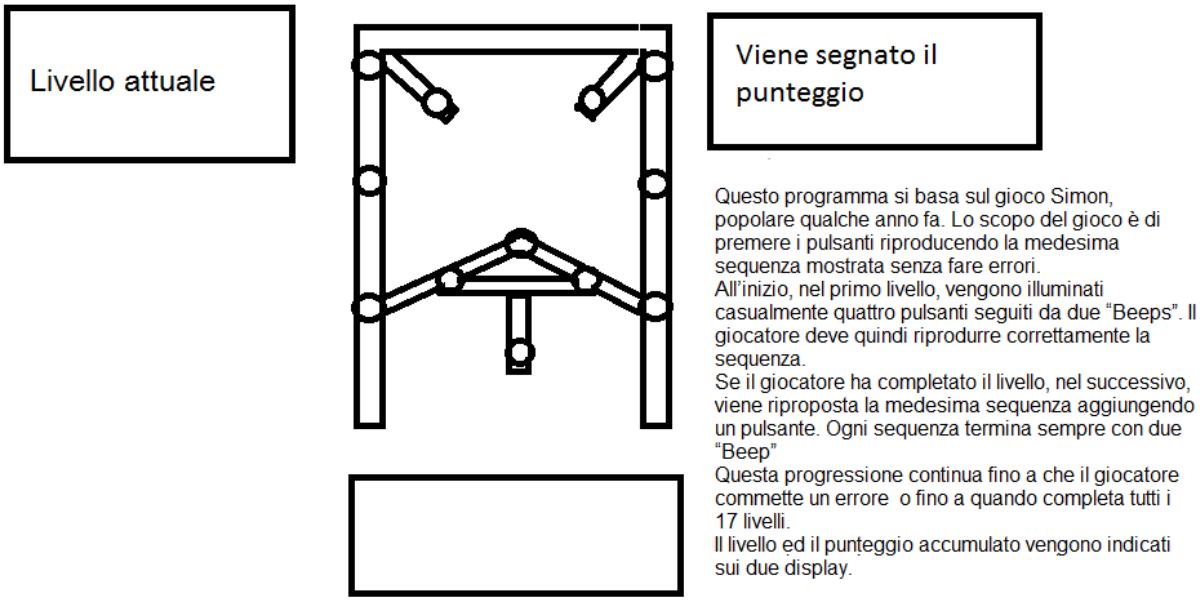


Figura 20: Modalità 20, Semplice gioco Simon

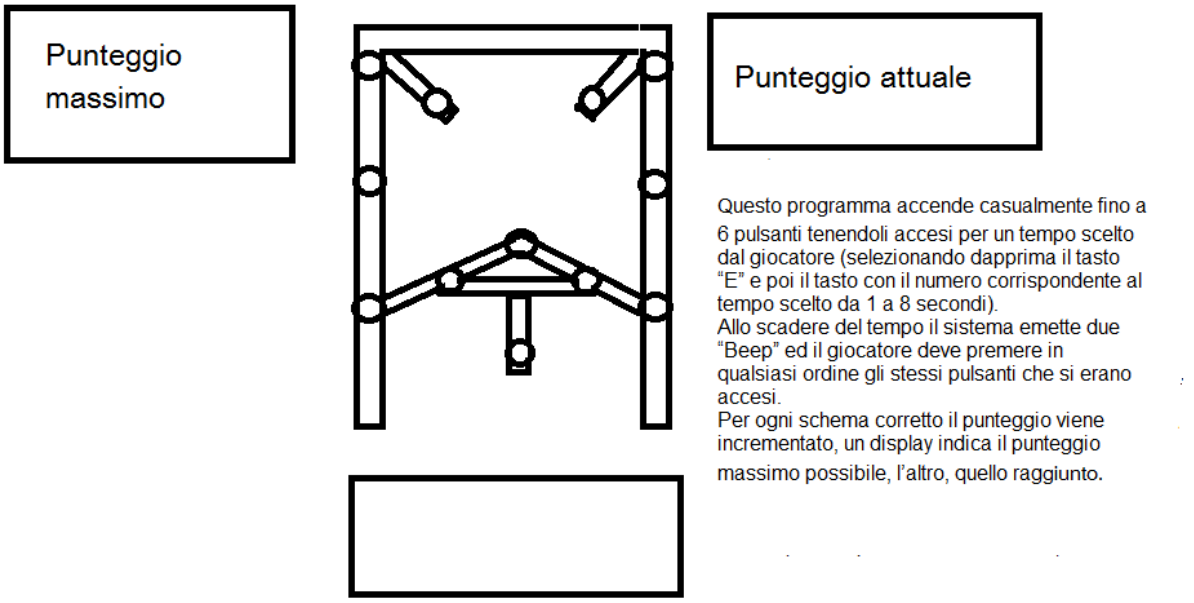


Figura 21: Modalità 21, Flash test

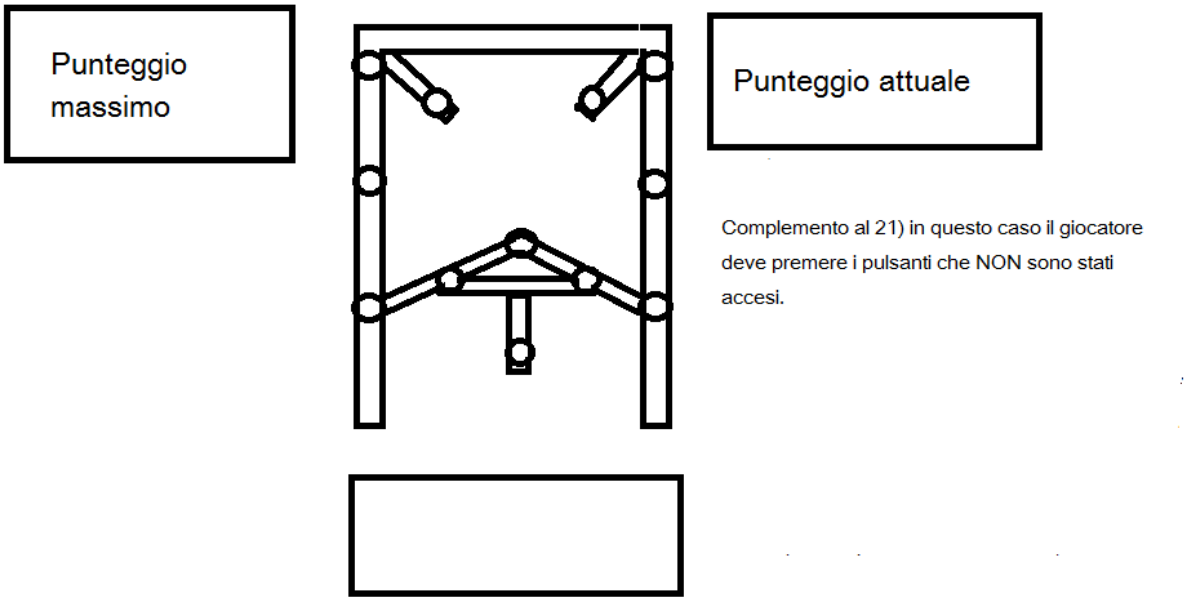


Figura 22: Modalità 22, Anti flash test

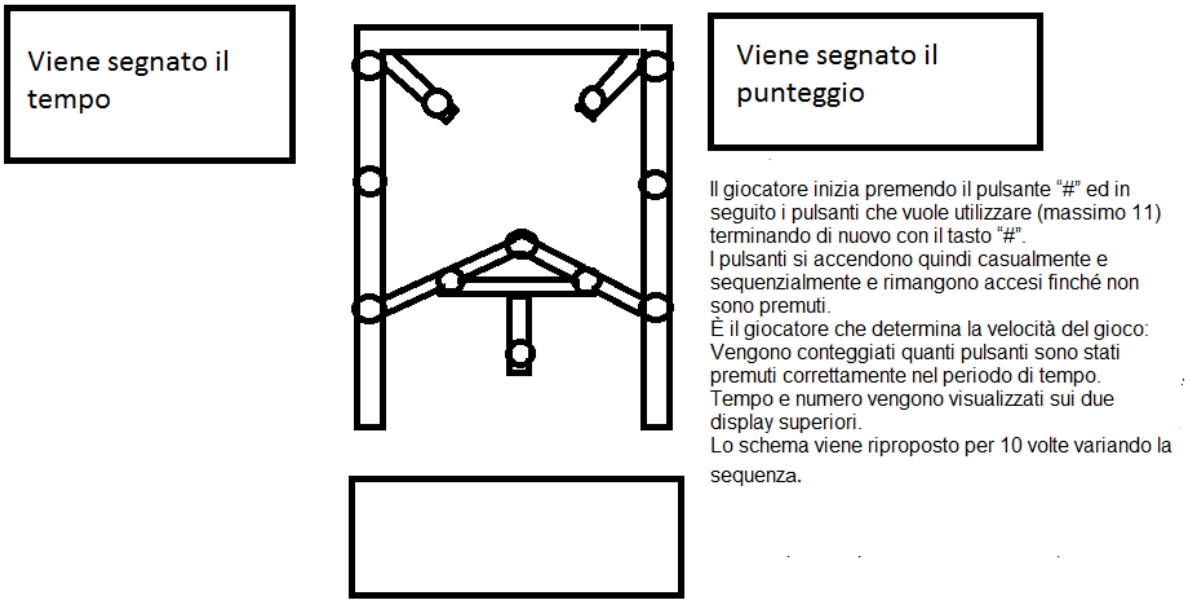


Figura 23: Modalità 23, Reazione veloce

3 Implementazione

Cambiamenti codice modalità

La conoscenza e l'esperienza necessaria alla creazione delle modalità è semplicemente il fatto che sia stata la stessa persona a farsi carico della riuscita di tutte le modalità.
Le categorie descritte nella prima versione del progetto sono rimaste, con le seguenti aggiunte:
Alla Categoria 3 è stata aggiunta la modalità 5, visto il comportamento molto simile di quest'ultima con le restanti della categoria. È stata gestita in modo che se il numero dei bottoni indicato alla chiamata della funzione sia 11, esegua i dovuti cambiamenti che permettano di giocare la modalità 5. Nello specifico, la possibilità che ci siano bottoni che lampeggino e che quindi non vadano premuti e l'accensione dei 3 bottoni centrali (che formano un triangolo) che indicano la necessità di premere la pedana, che come verrà descritto in seguito, è stata temporaneamente sostituita con il pulsante "#".
Le restanti modalità, nonostante per alcune avessimo anche codice relativamente funzionante, che con un po' di tempo saremmo riusciti a sistemare (cosa fatta con modalità 21 e 22), per altre abbiamo ritenuto più adatto rifarle completamente da 0, causa una logica disordinata ed evitabile (modalità 9 e 23). La modalità 6, 10 e 20, così come le 4 sopracitate, sono state tutte implementate in modo singolo, con una categoria a loro completamente dedicata.

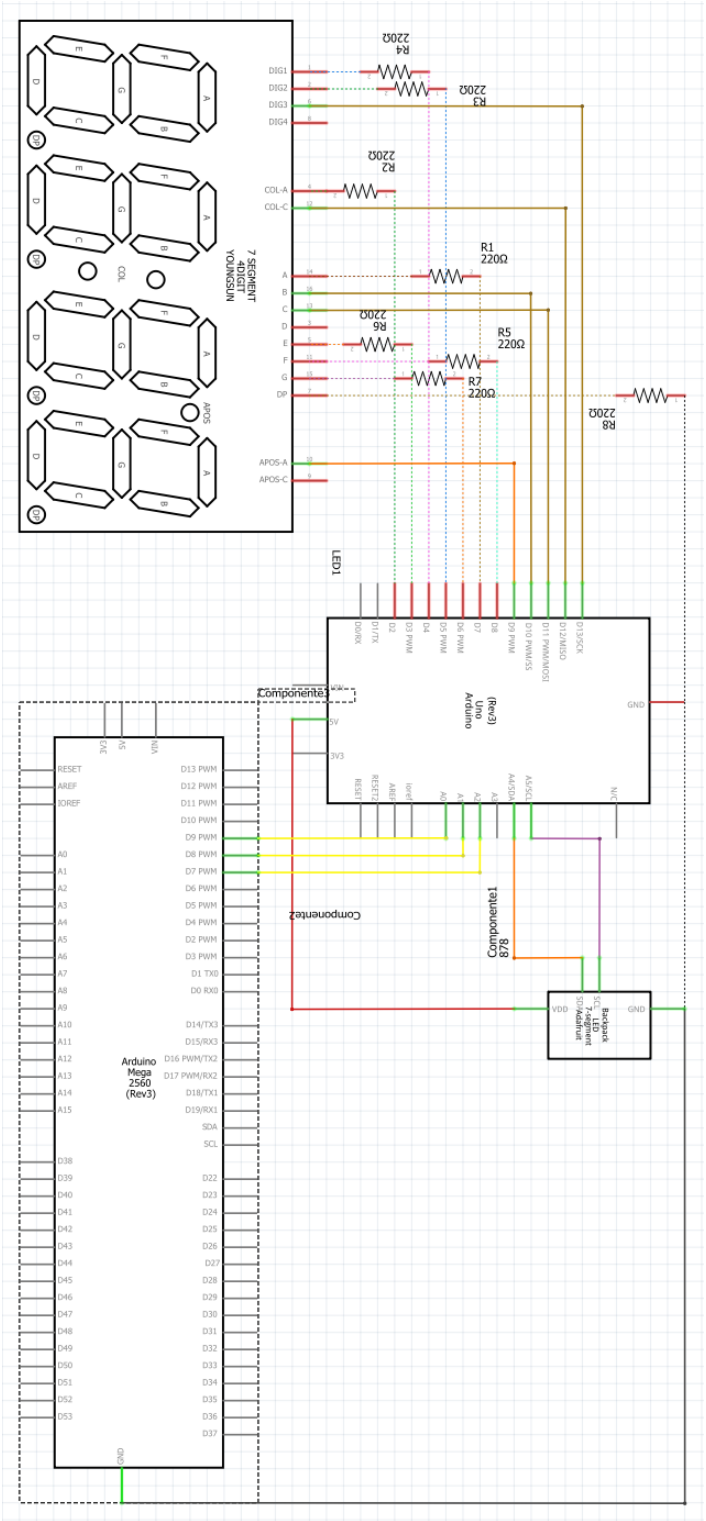
Collegamento Arduino → PHP → DB

Il collegamento tra questi ultimi non è stato modificato dalla prima versione del progetto (e quindi trovata descritta nel dettaglio nella documentazione di quest'ultima), con l'eccezione di un piccolo miglioramento nella gestione della pagina php principale (index.php) e per il fatto che la connessione da Arduino al Server (PC) sia una sola, per punteggio e modalità, invece che dividere i due aggiornamenti rendendo il codice inutilmente più macchinoso. La classifica è stata aggiunta direttamente nella pagina di gioco della persona che ha eseguito l'accesso. Questa visualizza appunto l'ordine decrescente dei punteggi avuti per la modalità giocata, con tanto di riga colorata di rosso per la partita appena terminata oppure di verde, per qualsiasi altra partita giocata dalla persona loggata per quella modalità.

Parte fisica

Per quanto riguarda la parte fisica del progetto, le uniche cose cambiate sono la presenza di due display 7 segmenti. La pedana, tanto discussa in corso di progetto sul come realizzarla, alla fine non è mai stata implementata. L'idea iniziale consisteva in un semplice gioco di molle che avrebbe permesso di chiudere un circuito e far così leggere il segnale. Si è poi arrivati all'ipotesi di rendere il tutto wireless o Bluetooth, per non usare dei cavi, scomodi nel caso si volesse variare la posizione della pedana, con un Fishino che permettesse di appunto inviare il segnale della pedana (che alla fine non è nient'altro che un output booleano) all'Arduino MEGA. Nonostante stessimo procedendo in questa direzione, dopo qualche settimana persa a implementare questa idea, è poi stato convenuto dal Docente Barchi che sarebbe stato molto meglio utilizzare delle resistenze flessibili per la creazione della pedana. Anche in questo caso, dopo un po' di tempo percorso in questa direzione, si è arrivati alla conclusione della fragilità delle suddette resistenze, che avrebbero rischiato di non resistere a diversi tipi di peso. Ormai il tempo a disposizione era agli sgoccioli e una volta tornati all'idea precedente, con l'utilizzo di un Fishino, abbiamo potuto constatare che il tempo per far arrivare il pezzo principale mancava e che le idee sulla pedana erano diventate troppe e sconclusionate per poterne effettivamente tirare fuori qualcosa in una settimana. È stata così sostituita, almeno in questa versione del progetto, in un semplice pulsante da premere (il "#", per la precisione).

Display 7 Segmenti



In questo schema si può vedere come i due Arduini sono collegati, l'Arduino Mega 2560 comunica attraverso tre pin (7,8 e 9) i quali possono inviare i dati analogici. Abbiamo collegato anche il "GND" in comune.
Il collegamento al modulo Adafruit con i pin "A4" (SDA), "A5" (SCL), 5[V] (VCC) e "GND", i quali servono per fare una comunicazione I2C.

Inoltre nello schema si può vedere anche un sette segmenti a 4 digits, per il quale non ho trovato il pezzo corretto. Infatti questo ha 16 pin e non 14, ma ho indicato i collegamenti corretti (come se fosse un sette segmenti a 14 pin). Ho utilizzato sette resistenze per i rispettivi segmenti.

```
1  int start = 3;
2  int finish = 5;
3
4  int button = 8;
5  int increment = 6;
6
7  void setup() {
8    pinMode(start, OUTPUT);
9    pinMode(finish, OUTPUT);
10
11   pinMode(increment, OUTPUT);
12   pinMode(button, INPUT);
13
14   analogWrite(start, 1023);
15   analogWrite(finish, 0);
16   start = millis()/1000;
17 }
18
19 void loop() {
20
21   if(digitalRead(button) == HIGH){
22     analogWrite(increment, 1023);
23   }
24   else{
25     analogWrite(increment, 0);
26   }
27   if((millis()/1000 - start) >= 20){
28     analogWrite(finish, 1023);
29     analogWrite(start, 0);
30     delay(3000);
31
32   }
33 }
```

Figura 24: Codice Arduino 2560 ProvaMaster

Questo codice è stato usato per l'Arduino Mega 2560. Il codice è relativamente semplice, si creano 4 variabili con le quali si fa la comunicazione e la ricezione della pressione del bottone. Nel loop viene fatto il controllo del tempo passato per far finire il programma e la gestione dell'incremento del punteggio.

```arduino
#include <Wire.h> // Enable this line if using Arduino Uno, Mega, etc.
#include <Adafruit_GFX.h>
#include "Adafruit_LEDBackpack.h"

Adafruit_7segment matrix = Adafruit_7segment();

int pinFinish = A0;
int pinStart = A1;

int increment = A2;

const int dig4 = 13;
const int dig3 = 12;
const int dig2 = 11;
const int dig1 = 10;

int dueP = 9;

int tempo[] = {5,4,2,3,6,7,8};

int punteggio = 0;

int start;
int ora;
```

Figura 25: Creazione variabili Arduino Uno

Queste sono le variabili globali dell'Arduino Uno con il quale si comandano i sette segmenti. Gli "#include" servono per importare le librerie necessarie della "Adafruit" per comandare il sette segmenti attraverso il relativo modulo. I tre pin (pinFinish, pinStart e increment) servono per ricevere i dati dal Mega. I restanti servono per le modalità e il sette segmenti.

```
26 void setup() {
27   pinMode(dig4, OUTPUT);
28   pinMode(dig3, OUTPUT);
29   pinMode(dig2, OUTPUT);
30   pinMode(dig1, OUTPUT);
31
32   for(int i = 0; i < sizeof(tempo); i++){
33     pinMode(tempo[i], OUTPUT);
34   }
35   pinMode(dueP, OUTPUT);
36
37   digitalWrite(dig4, HIGH);
38   digitalWrite(dig3, HIGH);
39   digitalWrite(dig2, HIGH);
40   digitalWrite(dig1, HIGH);
41
42   pinMode(pinStart, INPUT);
43   pinMode(pinFinish, INPUT);
44   pinMode(increment, INPUT);
45
46   matrix.begin(0x70);
47 }
48
49 void loop() {
50   if(analogRead(pinStart) > 1000 && analogRead(pinFinish) < 30){
51     start = millis()/1000;
52     while(true){
53       boolean finished = false;
54
55       if(analogRead(pinFinish) > 1000){
56         finished = true;
57       }
58
59       timer(start, finished);
60       if(analogRead(pinStart) < 30 && analogRead(pinFinish) > 1000){
61         punteggio = 0;
62         start = 0;
63         ora = 0;
64         timer(start, true);
65         break;
66       }
67     }
68   }
69 }
```

Figura 26: Metodi “setup” e “loop” Arduino Uno

Le prime 17 righe (27 - 44) servono ad istanziare le variabili per il sette segmenti senza modulo, poi le tre istanze sotto sono per i pin di ricezione del Master. Il “matrix.begin(0x70)” serve a “selezionare” la porta di uscita per il collegamento I2C con sette Segmenti con modulo.
Mentre nel loop c’è il controllo se è stata scelta una modalità e parte la partita con il timer e un “while(true)” dove si controlla se ha ricevuto il segnale dal Master di bloccarsi e poi di finire.

```
116 void stampa(int nrDisplay, int arrayP[]){
117
118   switch (nrDisplay){
119
120   case 0:
121     digitalWrite(arrayP[4], HIGH);
122     digitalWrite(arrayP[6], HIGH);
123     digitalWrite(arrayP[2], HIGH);
124     digitalWrite(arrayP[0], HIGH);
125     digitalWrite(arrayP[1], HIGH);
126     digitalWrite(arrayP[5], HIGH);
127     digitalWrite(arrayP[3], LOW);
128     break;
129
130   case 1:
131     digitalWrite(arrayP[4], LOW);
132     digitalWrite(arrayP[6], HIGH);
133     digitalWrite(arrayP[2], HIGH);
134     digitalWrite(arrayP[0], LOW);
135     digitalWrite(arrayP[1], LOW);
136     digitalWrite(arrayP[5], LOW);
137     digitalWrite(arrayP[3], LOW);
138     break;
```

Figura 27: Metodo stampa

etodo che serve a fare la stampa del numero per il digit selezionato. L'immagine mostra solo il codice necessario per i numeri 0 e 1, come esempio, i restanti sono stati ritenuti superflui e prolissi da aggiungere.

```
79 void timer(int inizio, boolean finish){
80   digitalWrite(dueP, HIGH);
81
82
83     if(!finish){
84       ora = (millis()/1000);
85     }
86     for(int dig = 4; dig > 0; dig--){
87       numero(dig);
88
89       matrix.print(punteggio);
90       matrix.writeDisplay();
91
92       if(dig == 4){
93         stampa((ora -start)%10, tempo);
94       }
95       else if(dig == 3){
96         stampa(((ora -start)/10)%6, tempo);
97       }
98       else if(dig == 2){
99         stampa(((ora -start)/60)%10, tempo);
100      }
101      else{
102        stampa((((ora -start)/60)/10)%6, tempo);
103      }
104
105      delayMicroseconds(5500);
106
107      if(analogRead(increment) > 1000){
108        punteggio++;
109      }
110
111      spegni();
112    }
113
```

Figura 28: Metodo timer

Metodo che serve a selezionare il digit e il numero corretto, sempre nel metodo controlla se è stato incrementato il punteggio attraverso l'input.

4 Test

Test Case: **Riferimento**:	TC-001	**Nome:**	Funzionamento bottoni telaio
Descrizione:	Controllare che tutti i bottoni funzionino e che vengano rilevati tramite il loro numero corrispettivo.		
Prerequisiti:	• Telaio BATAK con gli 11 bottoni funzionanti. • Display LCD collegato al dispositivo Arduino all'interno del telaio.		
Procedura:	1. Premere "@" per far partire la scelta delle modalità. 2. Premere bottone per bottone, verificando che il numero della modalità selezionata sia quello premuto. 3. Il numero della modalità usa un Sistema di Shifting, ergo, il numero appena premuto diventerà l'unità, mentre quello che prima era unità diventa decina.		
Risultati attesi:	Il numero dell'unità del display LCD dovrebbe essere uguale a quello del bottone appena premuto.		

Test Case: **Riferimento**:	TC-002	**Nome:**	Funzionamento led telaio
Descrizione:	Controllare che tutti i led funzionino e che corrispondano al bottone corretto.		
Prerequisiti:	• Telaio BATAK con gli 11 bottoni funzionanti.		
Procedura:	1. Premere "@" per far partire la scelta delle modalità e selezionare la modalità 23. 2. Premere bottone per bottone, verificando che una volta premuto quest'ultimo si accenda, e che venendo ripremuto si spenga.		
Risultati attesi:	I led dovrebbero tutti corrispondere al bottone premuto.		

Test Case:	TC-003	**Nome:**	Funzionamento Aggiornamento Classifica
Riferimento:	REQ-01		
Descrizione:	Controllare che I punteggi del giocatore vengano inviati a php e mostrati in tabella.		
Prerequisiti:	• Telaio BATAK. • Ethernet2 collegato all'Arduino MEGA e Cavo Ethernet. • Server Apache e Server MySQL funzionanti (consigliato XAMPP).		
Procedura:	1. Collegare tramite cavo Ethernet il PC e Ethernet2. 2. Dare al nostro computer l'ip fisso 192.168.5.17. 3. Accendere I due server, Apache e MySQL. 4. Inserire all'interno del PC, nella cartella htdocs, la cartella reactiongame e la cartella reactiongameform, trovabili sul GitHub del progetto sotto 2.0/codice. 5. Entrare nella pagina locale localhost/reactiongameform/index.php. 6. Creare o collegarsi con un account valido. 7. Far partire una modalità qualsiasi dalla BATAK ed eseguire qualche punto per controllare che la scrittura dei punteggi vada a buon fine. 8. Una volta finita la modalità, controllare la pagina a cui si era loggati. Dovrebbe essere comparsa una scritta e una classifica con punteggio e modalità.		
Risultati attesi:	Il punteggio e la modalità corrispondono a quelli realmente avuti giocando.		

4.1 Risultati Test

Test	**Risultato**
Funzionamento bottoni telaio	Positivo
Funzionamento led telaio	Positivo
Funzionamento Aggiornamento Classifica	Positivo

4.2 Mancanze/limitazioni conosciute

La pedana e il collegamento Wireless invece che Ethernet, inizialmente entrambi previsti per questa seconda versione del progetto, sono stati scartati per mancanza dei pezzi primari da utilizzare (pedana e Fishino). Volendo potremmo definire come mancanza anche la scarsa giocabilità di alcune modalità, che non essendo state rianalizzate assieme ai responsabili risultano a dir poco impossibili, abbiamo trovato. Tuttavia abbiamo ritenuto questo fattore non importante e per questo non gli è stato dedicato neanche del tempo.

5 Consuntivo

6 Conclusioni

Il prodotto ha dato i risultati sperati, con il completamento di tutti i requisiti primari e del progetto in sé. Ha molte possibilità di sviluppo, come il miglioramento delle cose abbandonate in corso del lavoro, e sarebbe potuto sicuramente andare meglio nel discorso 7 segmenti, visto l'enorme mole di tempo che ci ha fatto perdere.

6.1 Sviluppi futuri

Nuove modalità, migliorare la giocabilità di molte di quelle già presenti, una pedana effettiva e il collegamento al pc per l'aggiornamento punteggio sono gli esempi più lampanti di miglioramento che si potrebbe dare ulteriormente al progetto. Vanno poi aggiunte anche alcuni elementi che già nella scorsa versione del lavoro erano considerati non ottimali ma che sono stati ignorati per questa seconda versione. L'esempio migliore sono i led dei bottoni, che illuminano fin troppo poco, rendendo tedioso notare anche solo il fatto che siano illuminati, specialmente con certi tipi di luminosità. Ultimo ma non per importanza,

6.2 Considerazioni personali

Questo progetto ci ha mostrato esattamente come un solo elemento (due display 7 segmenti) possa risultare così problematico da bloccare tutto il procedimento del lavoro, finendo per spingere il completamento di quest'ultimo agli ultimi attimi, cosa non poco stressante e che per ora non siamo ancora riusciti ad evitare. Ciò nondimeno, il lavoro ci ha permesso di portare a compimento un progetto nato parecchi mesi fa assieme a due colleghi che per questa seconda fase del progetto purtroppo non ci hanno potuto seguire. Il senso di soddisfazione nel vedere un lavoro completato come si deve, dopo tanti sforzi, serate e perfino un weekend rimasti a lavorarci su come ore supplementari pensiamo possa dimostrarci come tenessimo anche noi stessi al lavoro, di come lo considerassimo già allora come una nostra creazione, qualcosa di cui siamo pienamente responsabili, sia nei pregi che nei difetti.

7 Bibliografia

7.1 Sitografia

- https://www.arduino.cc/en/Guide/
  - Ethernet2
  - Fishino
  - ITC
    - Arduino.
- https://www.w3schools.com/
  - Javascript
  - Php
    - W3Schools

http://www.hobbytronics.co.uk/arduino-4digit-7segment, codice prova sette segmenti senza modulo Adafruit
https://cdn-shop.adafruit.com/datasheets/865datasheet.pdf, datasheet sette segmenti senza modulo

8 Allegati

Allegato A: I3_Diari_ReactionGame2.pdf
Allegato B: I3_Documentazione_ReactionGame.pdf