

## Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Titolo del progetto:</b>	Sistema didattico per Arduino con libreria per attuatori e relativa documentazione
<b>Alunno/a:</b>	Mattia Lazzaroni – Mattia Toscanelli
<b>Classe:</b>	Info I3AC
<b>Anno scolastico:</b>	2018/2019
<b>Docente responsabile:</b>	Adriano Barchi – Luca Muggiasca – Francesco Mussi – Massimo Sartori

1.	Introduzione.....	3
1.1	Informazioni sul progetto.....	3
1.2	Abstract .....	3
1.3	Scopo .....	3
Analisi.....		4
1.4	Analisi del dominio .....	4
1.5	Analisi e specifica dei requisiti .....	4
1.6	Analisi dei costi .....	6
1.7	Pianificazione .....	6
1.8	Analisi dei mezzi .....	8
1.8.1	Software .....	8
1.8.2	Hardware.....	8
	Arduino Digispark .....	9
2	Progettazione .....	10
2.1	Design degli schemi elettrici.....	10
2.1.1	Schema elettrico potenziometro .....	10
2.1.2	Schema elettrico ServoMotore .....	11
2.1.3	Schema elettrico Ultrasuoni .....	12
2.2	Design delle librerie.....	13
2.2.1	Libreria Potenziometro.....	13
2.2.2	Libreria ServoMotore .....	14
2.2.3	Libreria Ultrasuoni.....	15
3	Implementazione .....	16
3.1	Libreria Potenziometro .....	16
3.1.1	Schema circuito Potenziometro .....	16
3.1.2	Creazione libreria.....	16
3.1.3	Sviluppo libreria Potenziometro .....	17
3.2	Libreria ServoMotore .....	18
3.2.1	Schema circuito ServoMotore.....	18
3.2.2	Sviluppo libreria ServoMotore.....	19
3.3	Libreria Ultrasuoni .....	20
3.3.1	Schema circuito Ultrasuoni .....	20
3.3.2	Sviluppo libreria Ultrasuoni .....	21
3.4	Esempi di utilizzo librerie.....	21
4	Test.....	22
4.1	Protocollo di test.....	22
4.2	Risultati test.....	24
4.3	Mancanze/limitazioni conosciute.....	24
5	Consuntivo.....	24
6	Conclusioni .....	26
6.1	Sviluppi futuri.....	26
6.2	Considerazioni personali.....	26
7	Bibliografia.....	26
7.1	Sitografia .....	26
8	Allegati.....	27

## 1. Introduzione

### 1.1 Informazioni sul progetto

**Titolo:** Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

**Allievi:** Mattia Lazzaroni, impiegato nello svolgimento del progetto.

Mattia Toscanelli, impiegato nello svolgimento del progetto.

**Classe:** I3AC

**Docenti:** Adriano Barchi, Luca Muggiasca, Francesco Mussi, Massimo Sartori

**Sezione scuola:** Scuola arti e mestieri Trevano

**Materia:** Modulo 306 – Progetti

**Data inizio:** 14.11.2018

**Fine:** 08.02.2019

### 1.2 Abstract

*This documentation is intended to help those who want to approach the world of electronics and programming. Within the document you can find all the parts of the project analysis, the initial design, the parts of code used for the implementation, all the various test cases and final conclusions.*

### 1.3 Scopo

Lo scopo del progetto è quello di realizzare un prodotto didattico dedicato agli allievi di terza media che arriveranno nelle giornate di porte aperte Promtec. Il nostro progetto consentirà all'utente di ambientarsi con la piattaforma elettronica Arduino (nel nostro caso mini DigiSpark) grazie a delle guide semplici ed intuitive che spiegheranno come utilizzarlo e come programmarlo. Per far capire al meglio il concetto di programmazione il prodotto offre delle librerie create da noi, contenenti dei metodi a cui sono stati attribuiti nomi semplici ed intuitivi.

Al nostro gruppo sono stati assegnati i seguenti componenti:

- Potenziometro
- ServoMotore
- Ultrasuoni

## Analisi

### 1.4 Analisi del dominio

Bisogna trovare un metodo che aiuti i ragazzi di terza media, giunti alla scuola arti e mestieri Trevano durante le porte aperte Promtec, ad approciarsi al mondo della programmazione. All'inizio della giornata lo studente riceverà un DigiSpark (un Arduino molto piccolo ma che svolge le stesse funzioni). L'obiettivo è quello di creare delle librerie per quest'ultimo contenenti metodi che semplificano la stesura del codice. Tutto ciò viene accompagnato con dei manuali utente in cui sono scritte dettagliate descrizioni su come funziona la libreria e su come devono venir utilizzati i vari metodi con tanto di esempi. Alla fine della giornata il ragazzo tornerà a casa con il DigiSpark, un piccolo manuale d'uso e sicuramente con delle conoscenze in più per quanto riguarda la programmazione.

### 1.5 Analisi e specifica dei requisiti

Il committente deve ricevere un DigiSpark e una guida per capirne il funzionamento. Il prodotto deve svolgere determinate azioni, come far accendere un led tramite un potenziometro. Queste funzioni sono implementate tramite il software Arduino 1.8.7. L'utente dovrebbe consultare la guida per capire come montare il circuito e come usare il proprio Arduino e le librerie presenti in esso da noi implementate. L'allievo consultando la guida dovrà capire tutto ciò che deve fare tramite una spiegazione semplice ed efficace, anche se egli non si è mai applicato prima nell'elettronica. Come minimo il prodotto deve far sì che tutte le librerie presenti funzionino e raggiungano il risultato atteso al 100%, senza bug o mancanze. Ovviamente il circuito dovrà essere sicuro e senza la presenza di rischi che possano ferire un visitatore. Quando l'utente finirà la sua attività avrà la possibilità di tenere il lavoro da lui svolto come ricordo della sua visita.

ID: REQ-001	
Nome	Saldamento dei pin DigiSpark
Priorità	1
Versione	1.0
Note	-
Sotto requisiti	
001	Si necessita un modulo da 6 pin e uno da 3 pin femmina.
002	Si necessita di un saldatore.
003	Si necessita dello stagno.

ID: REQ-002	
<b>Nome</b>	Creazione della libreria Potenziometro
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Per poter implementare le librerie bisogna completare il requisito REQ-001.
<b>Sottorequisiti</b>	
001	Il codice deve essere ben indentato.
002	Il codice deve essere commentato.
003	I nomi dei metodi devono essere di facile comprensione da parte dell'utente.
004	Si devono creare 3 codici d'esempio sull'utilizzo della libreria.
005	La libreria deve contenere un metodo che semplifichi la lettura del valore dal Potenziometro.

ID: REQ-003	
<b>Nome</b>	Creazione della libreria ServoMotore
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Per poter implementare le librerie bisogna completare il requisito REQ-001.
<b>Sottorequisiti</b>	
001	Il codice deve essere ben indentato.
002	Il codice deve essere commentato.
003	I nomi dei metodi devono essere di facile comprensione da parte dell'utente.
004	Si devono creare 3 codici d'esempio sull'utilizzo della libreria.
005	La libreria deve contenere dei metodi che semplificano il muovimento dell'elica del ServoMotore.

ID: REQ-004	
<b>Nome</b>	Creazione della libreria Ultrasuoni
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Per poter implementare le librerie bisogna completare il requisito REQ-001.
<b>Sottorequisiti</b>	
001	Il codice deve essere ben indentato.
002	Il codice deve essere commentato.
003	I nomi dei metodi devono essere di facile comprensione da parte dell'utente.
004	Si devono creare 3 codici d'esempio sull'utilizzo della libreria.
005	La libreria deve contenere un metodo che semplifichi la distanza letta dal sensore Ultrasuoni.

ID: REQ-005	
<b>Nome</b>	Creazione della guida
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Per poter scrivere la guida bisogna completare i requisiti REQ-002, REQ-003, REQ-004.
<b>Sottorequisiti</b>	
<b>001</b>	Bisogna spiegare come installare i driver della scheda Digispark.
<b>002</b>	Deve essere comprensibile a tutti gli utenti.
<b>003</b>	Deve contenere la spiegazione di tutte le librerie.
<b>004</b>	Deve avere degli esempi per aiutare la comprensione dell'utente.

## 1.6 Analisi dei costi

Ci siamo divisi i lavori:

- Mattia Toscanelli si occupa di creare delle librerie molto semplici in modo da aiutare gli allievi di terza media e si occupa di documentare tutti i passaggi eseguiti.
- Mattia Lazzaroni si occupa di generare la guida delle librerie dove vengono spiegati i vari componenti utilizzati e i metodi semplificati e si occupa di eseguire i vari test case.

Categoria	Ore di lavoro	Costo all'ora	Costo totale
Manodopera Mattia Lazzaroni	70 h	62 CHF	4340 CHF
Manodopera Mattia Toscanelli	70 h	62 CHF	4340 CHF

## 1.7 Pianificazione

Abbiamo inserito due pietre miliari, una a fine progettazione, cioè quando il progetto inizierà a diventare concreto; l'altra a fine progetto, quando potrà essere consegnato al cliente. La fase d'implementazione sarà quella in cui dedicheremo più tempo, in particolare l'attività "Sviluppo librerie", in quanto necessita di maggiore cura e inoltre per funzionare correttamente richiede di effettuare molti test.

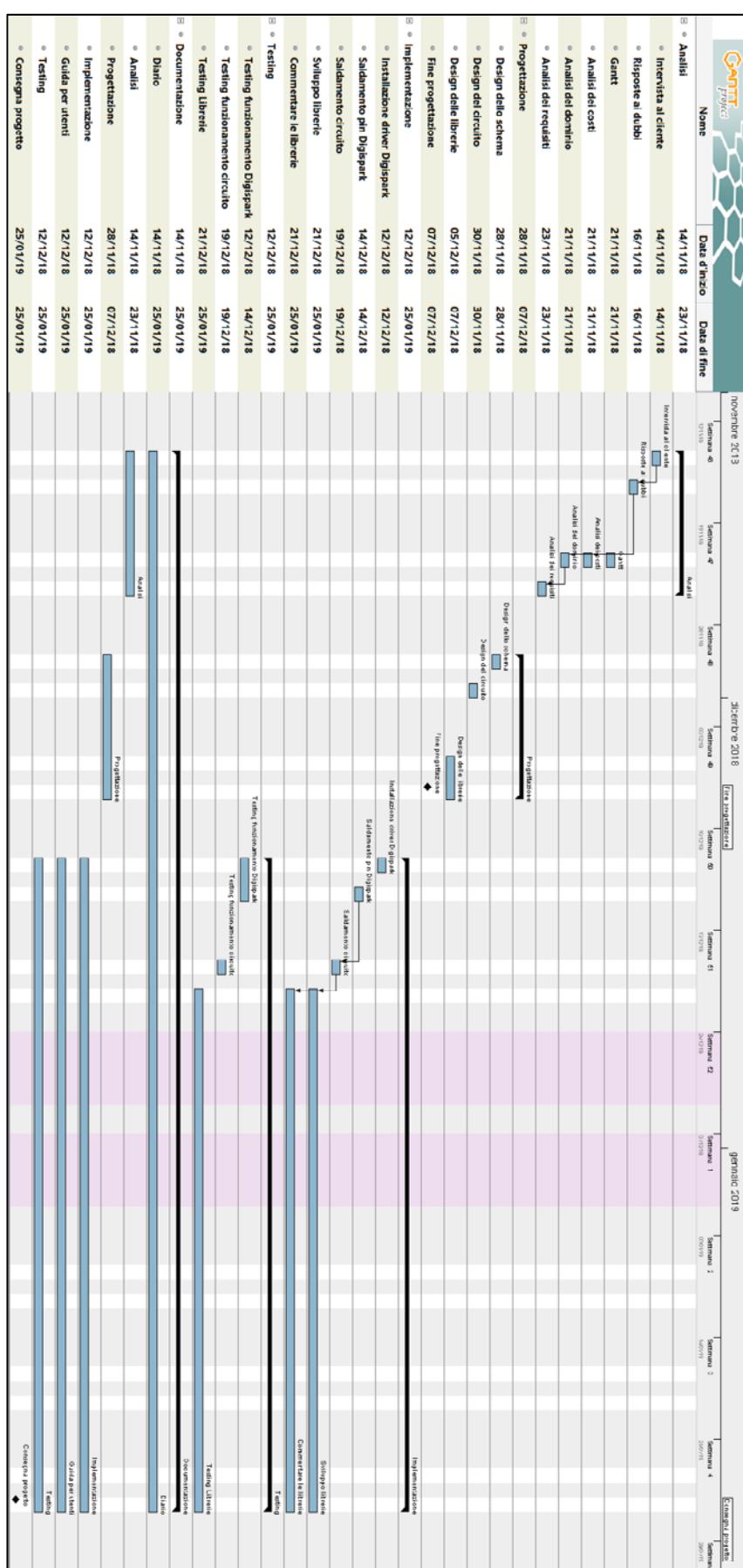


Figura 1, Gantt di pianificazione

## 1.8 Analisi dei mezzi

Per la realizzazione di questo progetto si ha bisogno di:

- Un arduino o simile
- Un PC con performance in grado di far girare il software di Arduino.

### 1.8.1 Software

I software utilizzati per la realizzazione di questo progetto sono:

- Word 2016
- Fritzing 0.9.3b
- Arduino 1.8.7
- Power Point 2016
- GanttProject 2.8.9
- Google Chrome 71.0.3578.98

### 1.8.2 Hardware

I componenti hardware utilizzati per la realizzazione del progetto sono:

- Digispark USB Development Board
- Caratteristiche computer Mattia Toscanelli:
  - Modello: Huawei Matebook X Pro
  - Processore: i7 8550u
  - RAM: 8GB LPDDR3 2133 MHz
  - GPU: NVIDIA MX150
  - SSD: 512 GB
- Caratteristiche computer Mattia Lazzaroni:
  - Modello: Acer Aspire E 15
  - Processore: i7 7500u
  - RAM: 16GB LPDDR4 2133 MHz
  - GPU: NVIDIA 940MX
  - SSD: 256 GB
- Digispark USB Development Board
  - Alimentazione tramite USB 5V oppure da sorgente estera 7-35V (consigliati 12V)
  - Regolatore On-board da 5V 500mA
  - Porta di collegamento a computer USB
  - 6 pin I/O (se il programma comunica attivamente tramite USB 2 pin vengono utilizzati per la porta USB, altrimenti è possibile utilizzare tutte 6, anche se si sta programmando via USB)
  - Memoria Flash da 8k (circa 6k dopo il bootloader)
  - Pin di I2C e SPI
  - 3 pin PWM (altri possibili con il software PWM)
  - 4 pin ADC
  - LED di alimentazione
  - LED di test/stato

## Arduino Digispark

Il **Digispark** è una scheda di sviluppo di piccole dimensioni e possiede un microcontrollore. Questo microcontrollore è simile a quello presente su un classico Arduino, soltanto che quello del Digispark costa meno ed è un po' più scarso. Nell'immagine qua sotto, sulla sinistra, si può vedere il Digispark sui due fronti. Come si può notare la scheda è già assemblata, tranne per i due connettori che si possono saldare facilmente. Lo scopo di questa scheda è quello di realizzare dei dispositivi di controllo o degli automatismi. Digispark dà la possibilità di usare l'IDE (software per la programmazione) di Arduino, che è gratuito e accessibile a tutti. Se si vuole collegare la scheda a dei componenti bisogna utilizzare i vari pin presenti sul Digispark. Nella parte inferiore troviamo tre pin. Il primo è il pin dell'alimentazione ed è indicato con "5V", il secondo pin è il pin della massa (o del -) ed è indicato con "GND" che sta per "Ground" (tradotto in italiano "Terra") e il terzo è il pin per alimentare la scheda da un alimentazione esterna ed è indicato con "Vin". I restanti 6 che si trovano sulla parte destra (dal P0 al P5) sono i pin "programmabili". Questi possono scrivere (cioè emettere corrente) o leggere (cioè ricevere corrente), inoltre possono assumere due tipologie diverse: Digitali, cioè che lavorano soltanto con 0 (niente corrente) e 1 (corrente) oppure Analogici, cioè che lavorano con un intervallo di corrente che va da 0 a 1023. Tutti i pin possono assumere la funzione digitale sia di lettura sia di scrittura, ma non tutti i pin possono assumere la funzione analogica. Infatti per quanto riguarda la scrittura analogica possono eseguirla solamente i pin: P0, P1 e P4. Per quanto riguarda la lettura analogica possono eseguirla solamente i pin: P2, P3, P4, P5. Normalmente la lettura analogica su Digispark ha una struttura diversa da quella che si può pensare infatti per leggere dal pin P2 bisogna scrivere 1, il pin P5 bisogna scrivere 0, il pin P4 bisogna scrivere 2 e per il pin P3 bisogna scrivere 3. Però grazie alle nostre librerie tutta questa struttura strana si può anche dimenticare, ma basta seguire la tabella qui di seguito:

Pin	Digital Read/Write	Analog Read	Analog Write
0	X		X
1	X		X
2	X	X	
3	X	X	
4	X	X	X
5	X	X	

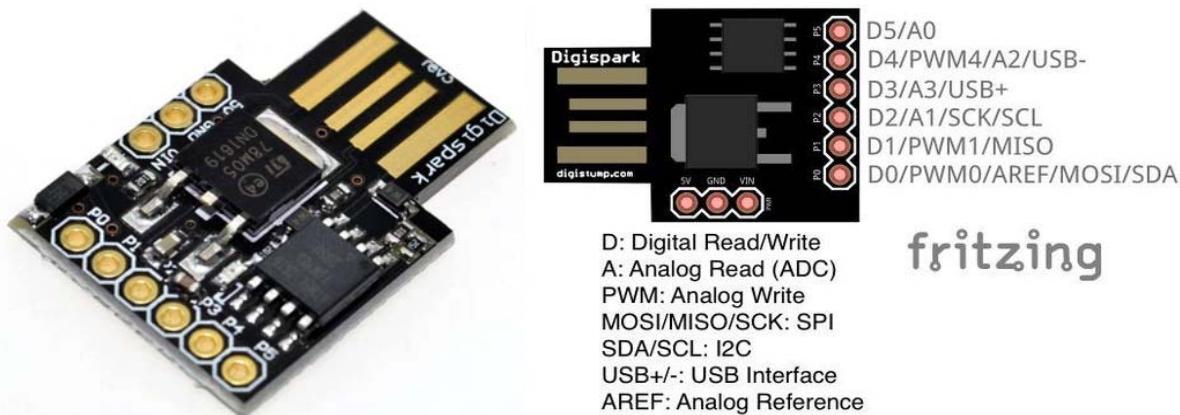


Figure 2, Scheda Digispark

## 2 Progettazione

### 2.1 Design degli schemi elettrici

#### 2.1.1 Schema elettrico potenziometro

Il funzionamento di questo circuito è molto semplice. Per farlo si necessita un diodo rosso, una resistenza da  $180\Omega$  e un potenziometro da  $10k\Omega$ . Lo scopo di questo circuito è il cambiamento di stato del LED grazie alla rotazione del potenziometro. Per effettuare questo circuito bisogna collegare il terminale 3 del potenziometro (vedi Figura 2) a un pin analogico del Digispark (in questo caso il P2). Per avere un cambiamento di luminosità del LED bisogna collegare anch'esso a un pin analogico, nel nostro caso al pin P0.

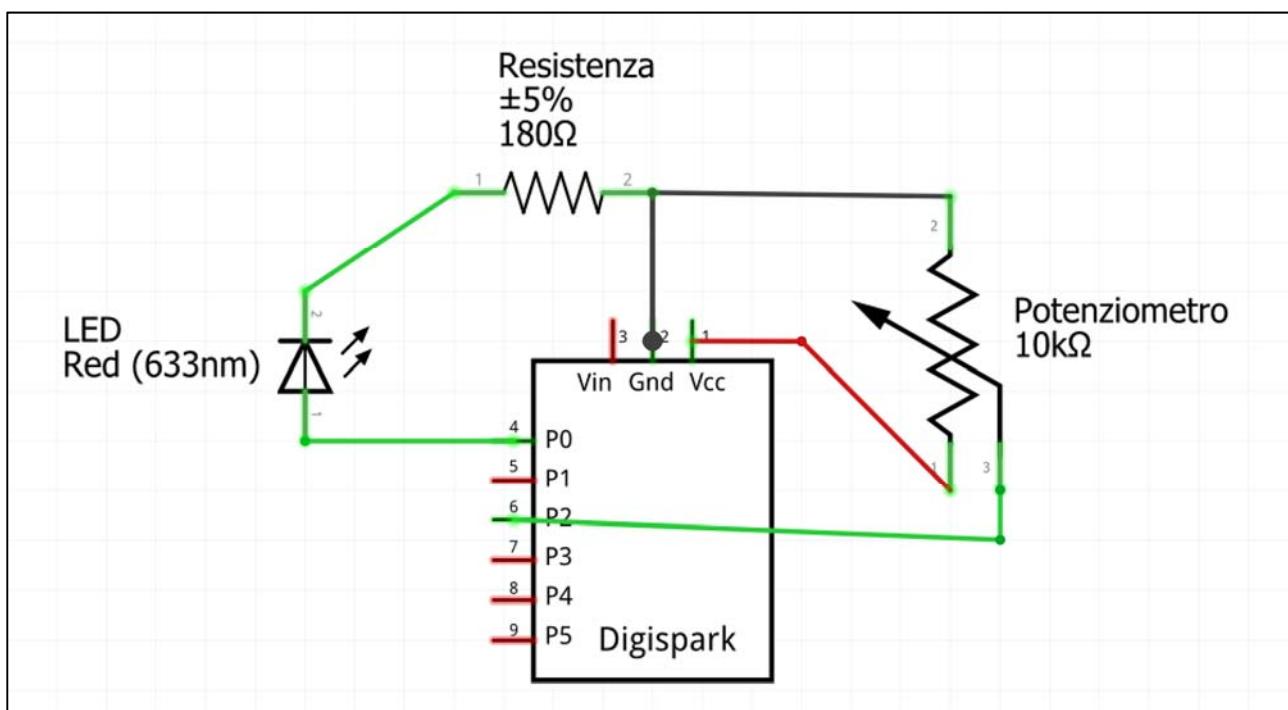


Figura 3, Schema elettrico del Potenziometro

### 2.1.2 Schema elettrico ServoMotore

Per fare questo circuito si necessita un pulsante e un ServoMotore. Lo scopo di questo circuito è il movimento del Servomotore grazie alla pressione o meno del pulsante. Per effettuare questo circuito bisogna collegare il terminale di pulse del ServoMotore (vedi Figura 3) a un pin analogico del Digispark (in questo caso il pin P3). In seguito bisogna collegare a un pin del Digispark uno dei due terminali 1 del pulsante (vedi Figura 3) per capire quando fa cambiare lo stato del ServoMotore.

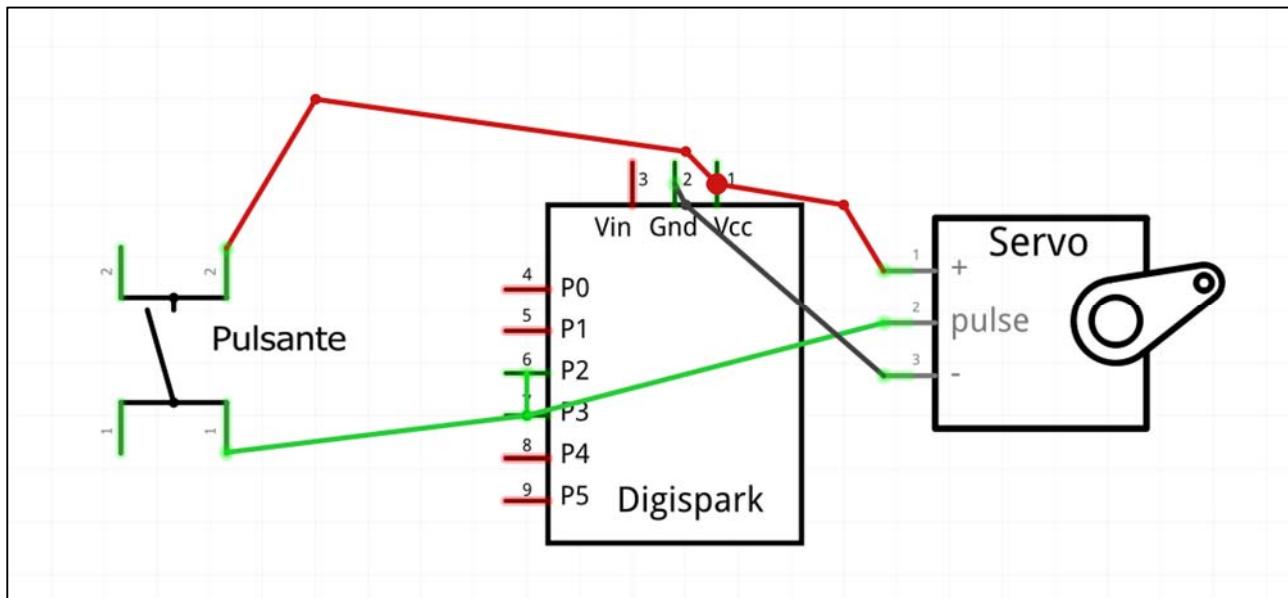


Figura 4, Schema elettrico del ServoMotore

### 2.1.3 Schema elettrico Ultrasuoni

Per fare questo circuito si necessita un sensore Ultrasuoni e un LED. Lo scopo di questo circuito è il cambiamento di stato del LED in base a quello che legge il sensore Ultrasuoni. Per effettuare questo circuito bisogna collegare il terminale di echo dell'Ultrasuoni (vedi Figura 4) a un pin analogico del Digispark (in questo caso il pin P2). In seguito bisogna collegare il pin di trig ad qualsiasi pin del Digispark (in questo caso il pin P3).

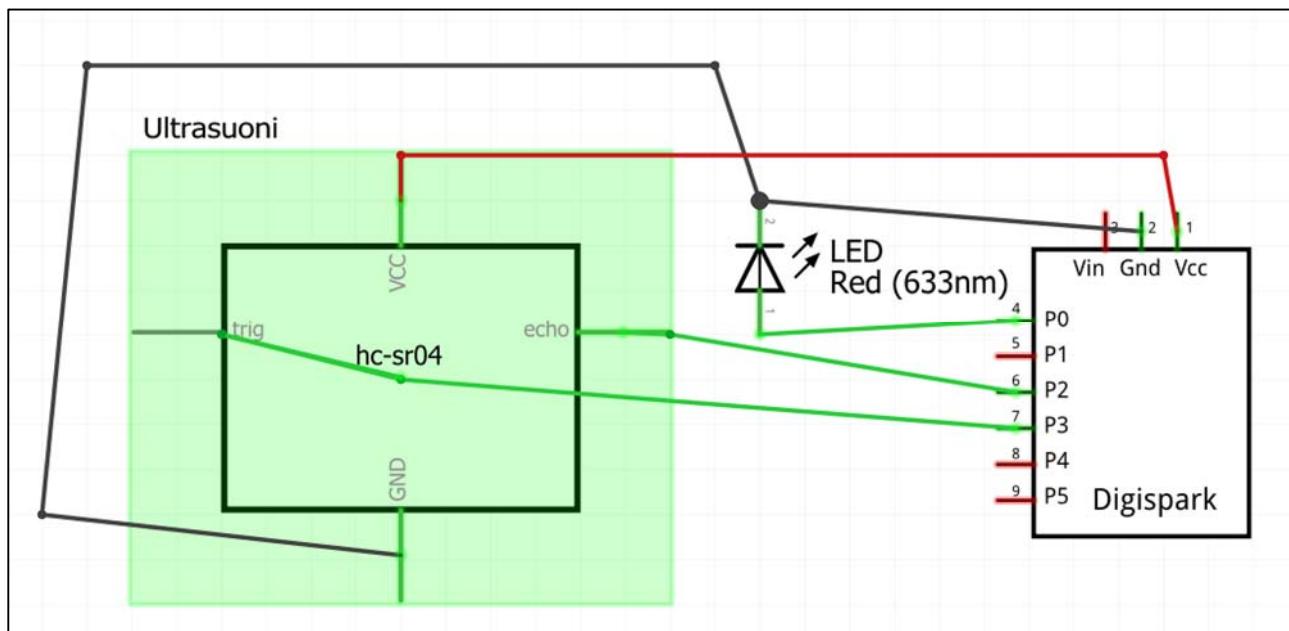


Figura 5, Schema elettrico dell'Ultrasuoni

## 2.2 Design delle librerie

### 2.2.1 Libreria Potenziometro

Questa libreria semplifica l'utilizzo di un Potenziometro. Contiene un metodo che permette la lettura del Potenziometro in modo rapido e in modo più semplificato.

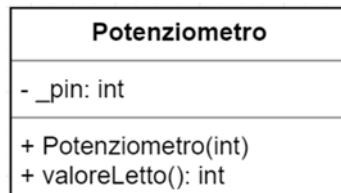


Figura 6, Libreria del potenziometro

Nome	Tipo	Descrizione
_pin	Variabile	L'attributo _pin conterrà il numero del pin a cui sarà collegato il terminale di lettura del potenziometro.
Potenziometro(int)	Metodo costruttore	È il metodo costruttore della classe Potenziometro. È composto da un solo parametro e quest'ultimo corrisponde al terminale di lettura del potenziometro.
valoreLetto(): int	Metodo	Questo metodo ritorna il valore letto dal pin analogico in cui è collegato il terminale del potenziometro. Il valore viene letto e convertito in un intervallo che parte da 0 fino a 255.

### 2.2.2 Libreria ServoMotore

Questa libreria semplifica l'utilizzo del ServoMotore. Contiene dei metodi che ottimizzano l'utilizzo del ServoMotore, come il settaggio della velocità e l'ottenimento della posizione.



Figura 7, Libreria del ServoMotore

Nome	Tipo	Descrizione
_pin	Variabile	L'attributo _pin conterrà il numero del pin a cui sarà collegato il terminale di lettura del ServoMotore.
_velocita	Variabile	L'attributo _velocita conterrà la velocità del ServoMotore. Questo valore può variare da -100 a 100.
_posizione	Variabile	L'attributo _posizione conterrà la posizione dell'elica del ServoMotore. Questo valore parte da 0° fino a 180°
POS_MINIMA	Variabile costante	L'attributo POSIZIONE_MINIMA conterrà la posizione minima dell'elica del ServoMotore (cioè la posizione corrispondente a 0°).
POS_MAXIMA	Variabile costante	L'attributo POSIZIONE_MAXIMA conterrà la posizione massima dell'elica del ServoMotore (cioè la posizione corrispondente a 180°).
ServoMotore(int,int)	Metodo costruttore	È il metodo costruttore della classe ServoMotore. È composto da due parametri: il primo esprime il pin a cui è collegato il ServoMotore e il secondo esprime la velocità dell'elica.
settaVelocita(int)	Metodo	Questo metodo permette di settare la velocità al ServoMotore. All'interno di esso viene controllata la velocità passata come parametro ed eventualmente viene corretta.
ottieniPosizione()	Metodo	Questo metodo ritorna la posizione dell'elica del ServoMotore.
avviaServo()	Metodo	Questo metodo fa muovere il ServoMotore.

### 2.2.3 Libreria Ultrasuoni

Questa libreria semplifica l'utilizzo dell'Ultrasuoni. Contiene un metodo che permette di sapere velocemente la velocità letta dall'Ultrasuoni.

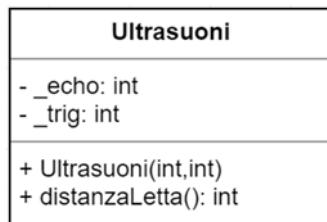


Figura 8, Libreria dell'Ultrasuoni

Nome	Tipo	Descrizione
_echo	Variabile	L'attributo _echo conterrà il numero del pin a cui sarà collegato il terminale di echo.
_trig	Variabile	L'attributo _echo conterrà il numero del pin a cui sarà collegato il terminale di trig.
Ultrasuoni(int, int)	Metodo costruttore	È il metodo costruttore della classe Ultrasuoni. È composto da due parametri: il primo esprime il pin di echo e il secondo il pin di trig.
distanzaLetta(): int	Metodo	Questo metodo ritorna la distanza letta dall'ultrasuoni e quest'ultima viene convertita in centimetri.

### 3 Implementazione

#### 3.1 Libreria Potenziometro

##### 3.1.1 Schema circuito Potenziometro

La prima libreria che ci è stata assegnata è quella del potenziometro. Innanzitutto ho preparato con Fritzing lo schema del circuito.

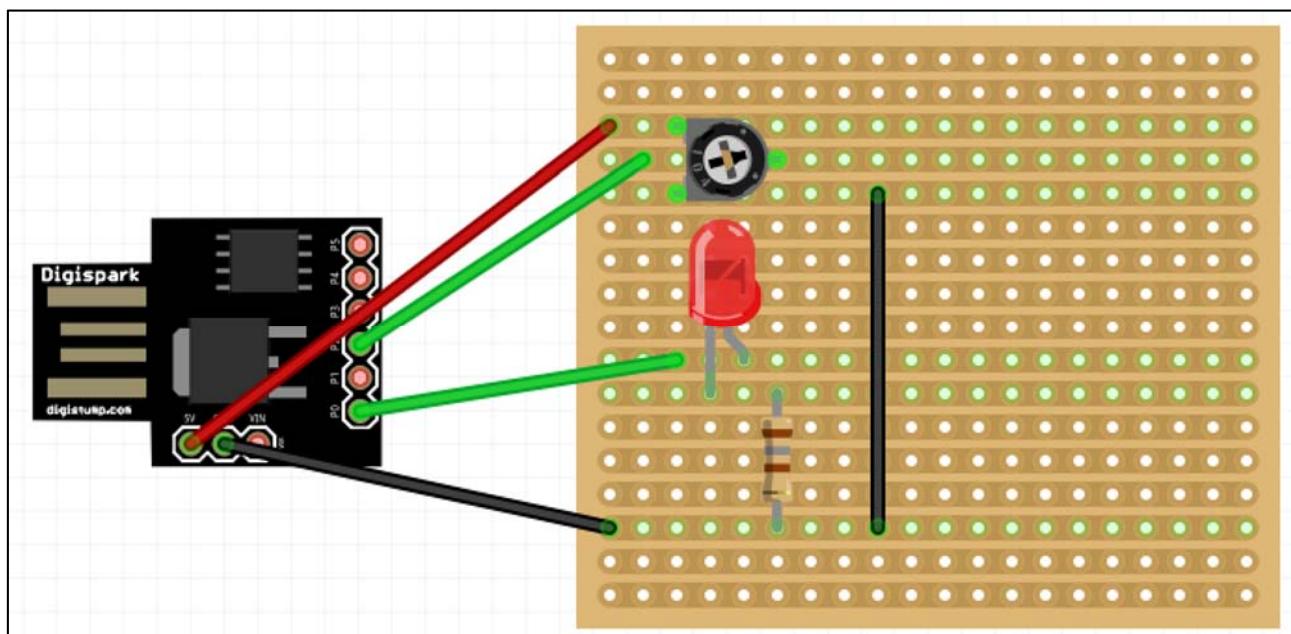


Figura 9, Schema del circuito con potenziometro e led.

Per iniziare abbiamo collegato il VCC del Digispark al terminale superiore del potenziometro, mentre il terminale inferiore del potenziometro è collegato a un cavo che giunge alla massa. Il pin 2 del Digispark è collegato al terminale centrale del potenziometro così da poter leggere il suo valore in modo analogico. Successivamente abbiamo congiunto il pin 0 del Digispark al anodo del led rosso, mentre il catodo giunge alla resistenza di  $180\Omega$  che a sua volta si reca alla massa.

##### 3.1.2 Creazione libreria

Essendo che non avevamo mai scritto una libreria per arduino ci siamo documentati su come la si faceva. Abbiamo scoperto che per effettuarle bisogna creare due file distinti per ognuna. Il primo di questi è un .h e ha la funzione di esprimere le variabili e le funzioni per il secondo tipo di file, come una vera e propria interfaccia. Il secondo file è un .cpp, in questo file dobbiamo includere l'interfaccia definita dal .h e "riempire" i metodi che poi verranno utilizzati. Questi due file vengono scritti con il linguaggio C.

Per includere un .h bisogna fare nel seguente modo:

```
#include "Potenziometro.h"
```

### 3.1.3 Sviluppo libreria Potenziometro

Abbiamo creato come prima cosa il file Potenziometro.h e abbiamo stabilito che metodi sarebbero stati utili per la semplificazione dell'utilizzo del componente. Questo è il risultato:

```
#ifndef Potenziometro_h
#define Potenziometro_h

#include "Arduino.h"

class Potenziometro{
public:
    Potenziometro (int pin);
    int valoreLetto();
private:
    int _pin;
};

#endif
```

La libreria è composta da 2 metodi pubblici e un singolo attributo privato. L'attributo `_pin` contiene il pin a cui è collegato il digispark e quindi dove verrà letto il valore del potenziometro.

Il metodo costruttore ha un singolo parametro e questo specifica appunto il numero del pin a cui viene collegato il potenziometro. Il pin deve essere di tipo analogico e essendo che si dovrà leggere si dovrà fare attenzione alla mappatura di Digispark in quanto è diversa da quella classica, infatti il pin2 corrisponde al 1, il pin3 corrisponde al 3, il pin4 corrisponde al 2 e il pin5 corrisponde al 0. Questo viene gestito dentro al metodo costruttore ridefinito nel file .ccp in questo modo:

```
if(pin == 2){
    _pin = 1;
}else if(pin == 4){
    _pin = 2;
}else if(pin == 5){
    _pin = 0;
}else{
    _pin = 3;
}
```

Infine c'è il metodo `valoreLetto()` che semplifica la lettura del potenziometro, infatti i valori analogici solitamente vanno da 0 a 1023 e questo metodo ristabilisce il range che parte da 0 e arriva fino a 255.

## 3.2 Libreria ServoMotore

### 3.2.1 Schema circuito ServoMotore

La seconda libreria che ci è stata assegnata è quella del ServoMotore. Come nel primo caso abbiamo preparato lo schema del circuito con Fritzing.

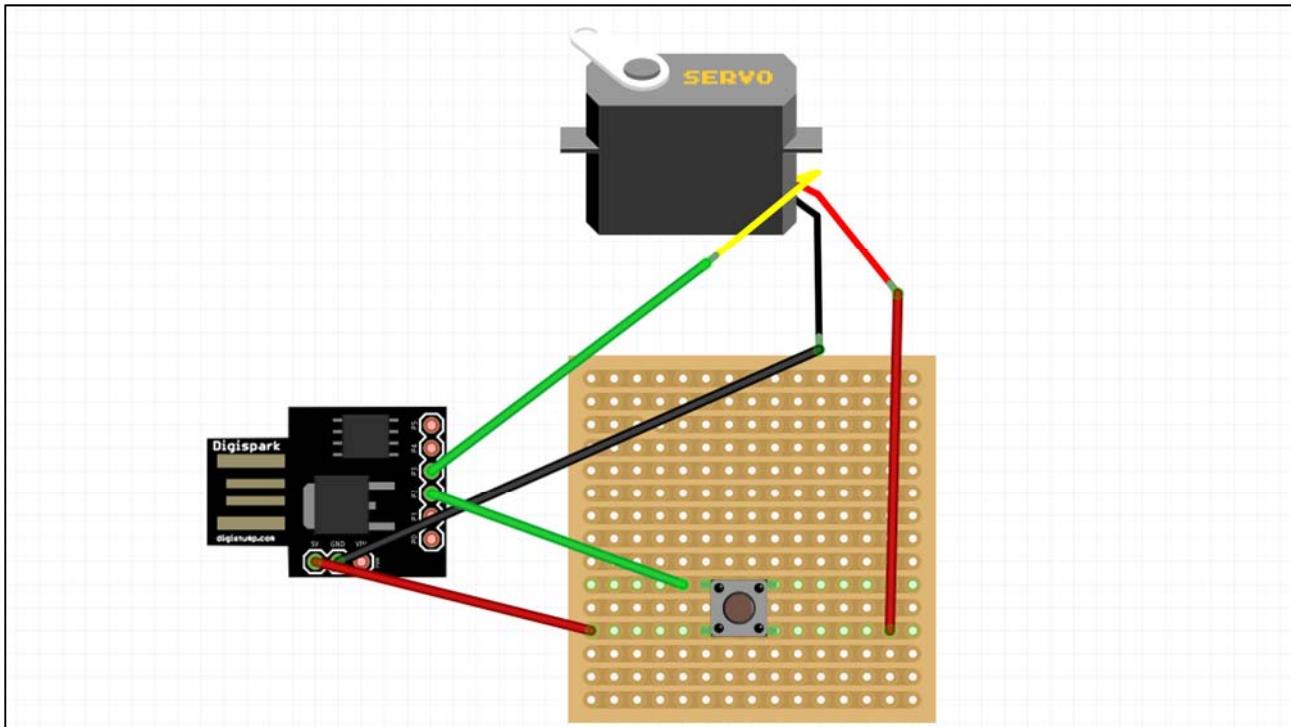


Figura 10, Schema del circuito con ServoMotore

Per iniziare abbiamo collegato il VCC del Digispark ai terminali inferiori del pulsante e al cavo di alimentazione rosso del ServoMotore, mentre al GND del Digispark abbiamo collegato il cavo di massa nero del ServoMotore. Il pin 2 del Digispark è collegato ai terminali superiori del pulsante così quanto quest'ultimo viene premuto o rilasciato si può leggere il suo valore in modo digitale. Successivamente abbiamo congiunto il pin 3 del Digispark al cavo di scrittura del ServoMotore in modo analogico.

### 3.2.2 Sviluppo libreria ServoMotore

Abbiamo creato come per la libreria del potenziometro il file contenente l'interfaccia del .cpp dove abbiamo stabilito i metodi che sarebbero stati utili per la semplificazione dell'utilizzo del componente. Questo è il risultato:

```
#ifndef ServoMotore_h
#define ServoMotore_h

#include "Arduino.h"

class ServoMotore {
public:
    ServoMotore(int pin, int velocita);
    void settaVelocita(int velocita);
    int ottieniPosizione();
    void avviaServo();
private:
    int _pin;
    const int POS_MINIMA = 544;
    const int POS_MAXIMA = 2400;
    int _velocita;
    int _posizione;
};

#endif
```

A differenza della prima libreria che abbiamo creato questa seconda è stata quello che ci ha richiesto più tempo in quanto più complessa. Esisteva già una libreria che permetteva l'utilizzo di questo componente però come ci è stato detto da un supervisore abbiamo provveduto a crearne una tutta nostra. La libreria è composta da 4 metodi pubblici e 5 attributi privati. Partendo con gli attributi, la variabile `_pin`, come nella precedente libreria del potenziometro, conterrà il pin a cui è collegato il cavo di scrittura del ServoMotore. Questo deve essere analogico ma in questo caso non si eseguirà la mappatura descritta nel capitolo 3.1.2 essendo che bisognerà scrivere e non leggere. Poi abbiamo creato due variabili costanti (`POS_MINIMA`, `POS_MAXIMA`) che esprimono, come dice il termine, la posizione minima dell'elica del ServoMotore e viceversa. Poi abbiamo creato una variabile `_velocita` in cui sarà descritta la velocità dell'elica e per fine una variabile `_posizione` dove appunto conterrà la posizione dell'elica.

Passando con i metodi, il costruttore ha due parametri, cioè il pin a cui è collegato il cavo di scrittura del ServoMotore e la velocità dell'elica da settare come partenza ed infine viene settata anche la posizione iniziale (quella minima).

Poi abbiamo creato il metodo `settaVelocità(int)` in cui si può settare appunto la velocità dell'elica del ServoMotore nel corso del programma. Inoltre all'interno del metodo vengono eseguiti dei controlli sul valore passato come parametro ed eventualmente corretti con un numero che va da -100 a 100.

Come penultimo metodo abbiamo creato `ottieniPosizione()` che ritorna la posizione dell'elica in gradi (da 0 a 180), come descritto nell'istruzione qui di seguito:

```
return map(_posizione, POS_MINIMA, POS_MAXIMA, 0, 180);
```

In pratica grazie alla funzione `map()` possiamo ridefinire il range in uno più accettabile.

Infine abbiamo creato il metodo `avviaServo()` che appunto sarà l'istruzione che farà muovere il ServoMotore con le caratteristiche settate precedentemente.

### 3.3 Libreria Ultrasuoni

#### 3.3.1 Schema circuito Ultrasuoni

L'ultima libreria che ci è stata assegnata è quella dell'Ultrasuoni. Come nelle altre due libreria abbiamo preparato come prima cosa lo schema in Fritzing

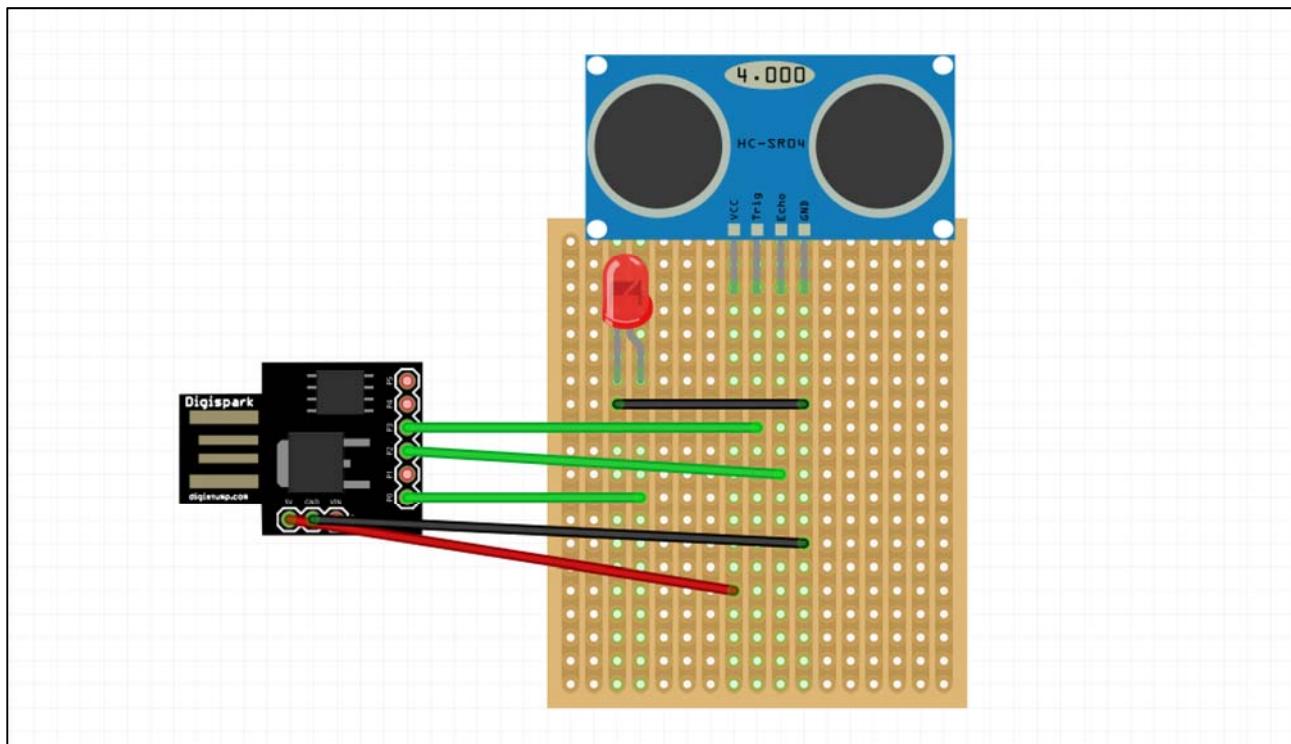


Figura 11, Schema del circuito con Ultrasuoni

Per iniziare abbiamo collegato il VCC del Digispark al terminale VCC dell'Ultrasuoni, mentre abbiamo collegato il pin di GND del Digispark al terminale GND dell'Ultrasuoni e al catodo del LED. Il pin 2 digitale del Digispark è collegato al terminale di trig dell'Ultrasuoni, questo serve per inviare un impulso alto per almeno 10 microsecondi cosicché il sensore invii il ping sonoro e aspetti il ritorno delle onde riflesse. Per poi rispondere sul terminale di Echo del sensore, collegato al pin 2 sul Digispark, con un impulso alto della durata corrispondente a quella di viaggio delle onde sonore. Infine abbiamo collegato al pin 0 del Digispark l'anodo del LED il quale cambiare stato in funzione della lettura del sensore Ultrasuoni.

### 3.3.2 Sviluppo libreria Ultrasuoni

Abbiamo creato come prima cosa il file Ultrasuoni.h e abbiamo stabilito che metodi sarebbero stati utili per la semplificazione dell'utilizzo del componente. Questo è il risultato:

```
#ifndef Ultrasuoni_h
#define Ultrasuoni_h

#include "Arduino.h"

class Ultrasuoni {
public:
    Ultrasuoni(int echo, int trig);
    int distanzaLetta();
private:
    int _echo;
    int _trig;
};

#endif
```

La libreria è composta da 2 metodi pubblici e un 2 attributi privati. A differenza delle due librerie precedenti in questa libreria teniamo conto di due pin e non uno solo, infatti nella variabile `_echo` verrà contenuto il numero del pin a cui è collegato il terminale di echo del sensore ultrasuoni e nella variabile `_trig` verrà contenuto il numero del pin a cui è collegato il terminale di trig.

Il metodo costruttore ha due parametri, il pin di echo e il pin di trig i quali specificano appunto il numero dei pin a cui vengono collegati.

Infine abbiamo creato il metodo `distanzaLetta()` che ha questa serie di istruzioni:

```
digitalWrite(_trig, LOW);
delayMicroseconds(2);
digitalWrite(_trig, HIGH);
delayMicroseconds(10);
digitalWrite(_trig, LOW);
long durata = pulseIn(_echo, HIGH);
return (durata * 0.034 / 2);
```

In pratica viene mandato l'impulso di trigger, cioè quello descritto nel capitoletto precedente, ed in seguito viene letta la durata del viaggio delle onde sonore, per poi essere divisa per 2 (avanti + indietro).

## 3.4 Esempi di utilizzo librerie

Per concludere abbiamo creato per ogni libreria una serie di 3 esempi, così da avere un totale di 9 esempi. Ogni gruppo della libreria presenta un esempio semplice, uno intermedio e per finire uno più difficile. Questi non ci hanno creato particolarmente problemi in quanto per persone del nostro livello sono cose abbastanza basilari. Il primo esempio di ogni libreria viene descritto nel manuale utente.

## 4 Test

### 4.1 Protocollo di test

<b>Test Case:</b>	TC-001	<b>Nome:</b>	Verifica di funzionamento della scheda Digispark
<b>Riferimento:</b>	REQ-001		
<b>Descrizione:</b>	Verifica se i pin installati sulla scheda Digispark sono funzionanti.		
<b>Prerequisiti:</b>	Arduino IDE deve essere presente sulla macchina. È necessario un LED rosso.		
<b>Procedura:</b>	<ol style="list-style-type: none"> <li>1. Aprire il software “Arduino.exe”.</li> <li>2. Collegare l'anodo del LED sul pin 0 della scheda Digispark e il catodo sul pin GND sempre della scheda Digispark.</li> <li>3. Recarsi nella navbar del software Arduino, premere sul <b>File → Examples → 01.Basics → Blink</b></li> <li>4. Modificare tutti i campi con scritto “LED_BUILTIN” con “0”.</li> <li>5. Premere il pulsante in alto con la freccetta verso destra e aspettare il messaggio nella console “Plug in device now... (will timeout in 60 seconds)”.</li> <li>6. Inserire la scheda Digispark nel computer e attendere il risultato.</li> </ol>		
<b>Risultati attesi:</b>	Il LED rosso che lampeggia ininterrottamente.		

<b>Test Case:</b>	TC-002	<b>Nome:</b>	Verifica del funzionamento della libreria
<b>Riferimento:</b>	REQ-002 REQ-003 REQ-004		
<b>Descrizione:</b>	Verifica se la libreria è stata posizionata nella corretta posizione.		
<b>Prerequisiti:</b>	-		
<b>Procedura:</b>	<ol style="list-style-type: none"> <li>1. Copiare dalla root del progetto la cartella “Potenziometro”.</li> <li>2. Cliccare il pulsante <b>Windows Start</b></li> <li>3. Digitare la parola “Arduino”</li> <li>4. Al primo risultato fare tasto destro → <b>Apri percorso file</b></li> <li>5. Cercare il file “Arduino.exe” → tasto destro → <b>Apri percorso file</b></li> <li>6. Entrare nella cartella <b>libraries</b></li> <li>7. Inserire la scheda Digispark nel computer e attendere il risultato.</li> </ol>		
<b>Risultati attesi:</b>	Il LED rosso che lampeggia ininterrottamente.		

**Sistema didattico per Arduino**

<b>Test Case:</b>	TC-003	<b>Nome:</b>	Verifica di funzionamento della libreria del Potenziometro
<b>Riferimento:</b>	REQ-002		
<b>Descrizione:</b>	Verifica se la libreria del Potenziometro funziona correttamente.		
<b>Prerequisiti:</b>	La cartella deve essere inserita all'interno della cartella di <b>Arduino</b> , più precisamente nella sottocartella <b>libraries</b> .		
<b>Procedura:</b>	<ol style="list-style-type: none"> <li>Eseguire il montaggio visto nel capitolo <b>3.1.1</b> (<a href="#">Schema circuito Potenziometro</a>).</li> <li>Avviare il primo esempio di codice della libreria Potenziometro trovatosi nella root del progetto nella cartella "Potenziometro esempi" → "PotenziometroBlink" → "PotenziometroBlink.ino".</li> <li>Premere il pulsante in alto con la freccetta verso destra e aspettare il messaggio nella console "Plug in device now... (will timeout in 60 seconds)".</li> <li>Inserire la scheda Digispark nel computer e attendere il risultato.</li> </ol>		
<b>Risultati attesi:</b>	Il codice compilato correttamente e il circuito funzionante.		

<b>Test Case:</b>	TC-004	<b>Nome:</b>	Verifica di funzionamento della libreria del ServoMotore
<b>Riferimento:</b>	REQ-003		
<b>Descrizione:</b>	Verifica se la libreria del ServoMotore funziona correttamente.		
<b>Prerequisiti:</b>	La cartella deve essere inserita all'interno della cartella di <b>Arduino</b> , più precisamente nella sotto cartella <b>libraries</b> .		
<b>Procedura:</b>	<ol style="list-style-type: none"> <li>Eseguire il montaggio visto nel capitolo <b>3.2.1</b> (<a href="#">Schema circuito ServoMotore</a>).</li> <li>Avviare il primo esempio di codice della libreria Potenziometro trovatosi nella root del progetto nella cartella "ServoMotore esempi" → "ServoMotoreBase" → "ServoMotoreBase.ino".</li> <li>Premere il pulsante in alto con la freccetta verso destra e aspettare il messaggio nella console "Plug in device now... (will timeout in 60 seconds)".</li> <li>Inserire la scheda Digispark nel computer e attendere il risultato.</li> </ol>		
<b>Risultati attesi:</b>	Il codice compilato correttamente e il circuito funzionante.		

<b>Test Case:</b>	TC-005	<b>Nome:</b>	Verifica di funzionamento della libreria del Ultrasuoni
<b>Riferimento:</b>	REQ-004		
<b>Descrizione:</b>	Verifica se la libreria del Ultrasuoni funziona correttamente.		
<b>Prerequisiti:</b>	La cartella deve essere inserita all'interno della cartella di <b>Arduino</b> , più precisamente nella sotto cartella <b>libraries</b> .		
<b>Procedura:</b>	<ol style="list-style-type: none"> <li>Eseguire il montaggio visto nel capitolo <b>3.3.1</b> (<a href="#">Schema circuito Ultrasuoni</a>).</li> <li>Avviare il primo esempio di codice della libreria Potenziometro trovatosi nella root del progetto nella cartella "Ultrasuoni esempi" → "UltrasuoniBase" → "UltrasuoniBase.ino".</li> <li>Premere il pulsante in alto con la freccetta verso destra e aspettare il messaggio nella console "Plug in device now... (will timeout in 60 seconds)".</li> <li>Inserire la scheda Digispark nel computer e attendere il risultato.</li> </ol>		
<b>Risultati attesi:</b>	Il codice compilato correttamente e il circuito funzionante.		

<b>Test Case:</b>	TC-006	<b>Nome:</b>	Verifica della guida
<b>Riferimento:</b>	REQ-006		
<b>Descrizione:</b>	Verifica se la guida è comprensibile a tutti gli utenti.		
<b>Prerequisiti:</b>	-		
<b>Procedura:</b>	<ol style="list-style-type: none"> <li>1. Verificare se è presente la spiegazione su come installare i driver della scheda Digispark.</li> <li>2. Verificare se è comprensibile per qualsiasi persona.</li> <li>3. Verificare se sono presenti delle spiegazioni sulle librerie.</li> <li>4. Verificare se sono presenti degli esempi per aiutare la comprensione.</li> </ol>		
<b>Risultati attesi:</b>	La guida contenente tutto.		

#### 4.2 Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l'errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l'errore con eventuali ipotesi di correzione.

Nome Requisito	Risultato
TC-001	Riuscito
TC-002	Riuscito
TC-003	Riuscito
TC-004	Riuscito
TC-005	Riuscito
TC-006	Riuscito

#### 4.3 Mancanze/limitazioni conosciute

Il progetto è stato concluso a buon fine, cioè tutto ciò che è stato richiesto è stato fatto. Abbiamo però riscontrato diverse limitazioni per quanto riguarda la scheda Digispark. Il primo ed evidente problema è la quantità di pin a disposizione che offre la scheda, infatti ne dispone solamente 6 e questo significa che non si possono progettare ed eseguire dei circuiti con più di 6 componenti. Un secondo svantaggio è l'assenza di del *Serial monitor*, cioè lo strumento presente nell'IDE di Arduino che permette di dare un *feedback* al programmatore mostrandogli dei dettagli di test del suo programma. Questo è stato risolto utilizzando un'altra scheda Arduino (Arduino MEGA) la quale presentava questa funzione. Un ultimo problema che abbiamo riscontrato è l'alimentazione di più componenti contemporaneamente, infatti quando si andavano a collegare più LED in diversi pin distinti quest'ultimi si illuminavano molto poco.

#### 5 Consuntivo

---

Il Gantt ha ricevuto molte modifiche per quanto riguarda la fase di Implementazione e a sua volta anche la fase di Testing. Questo è successo perché durante i primi due mesi i nostri supervisori erano poco disponibili per proporci domande inerenti a nostri dubbi sul progetto.

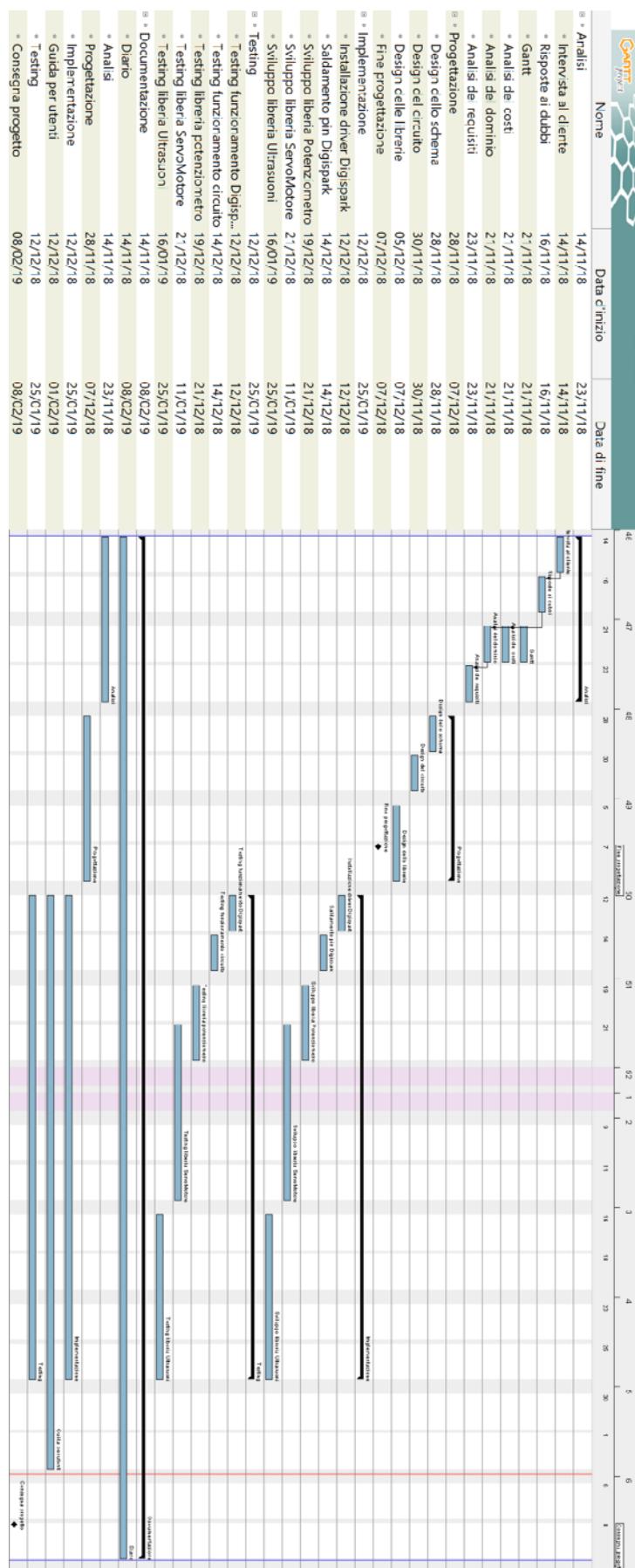


Figura 12, Gantt consuntivo

## 6 Conclusioni

Per realizzare questo progetto abbiamo speso molto ore, nonostante ciò non crediamo che si sia trattato di una perdita di tempo dato che questo progetto ci ha aiutati a migliorare le nostre capacità di lavorare in team e ci ha permesso di imparare le basi di linguaggi a noi prima quasi sconosciuti come C++ e C. Sicuramente questo progetto non servirà a cambiare il mondo, tuttavia nel suo piccolo potrà aiutare i giovani ragazzi che vogliono avvicinarsi al mondo della programmazione e dell'elettronica. Arduino essendo un linguaggio abbastanza semplice può essere molto utile per gli inesperti, per poi procedere con l'apprendimento di linguaggi più complicati. I risultati che abbiamo ottenuto non sono difficilmente realizzabili, ma comunque ci riteniamo soddisfatti del lavoro svolto e delle nozioni acquisite.

### 6.1 Sviluppi futuri

Nel futuro si potrebbero realizzare altre librerie con nuovi attuatori oppure aggiungere dei metodi alle librerie presenti. Inoltre si potrebbe produrre un manuale ancora più semplificato per chi non sa neanche che cosa sia l'informatica o l'elettronica.

### 6.2 Considerazioni personali

Da questo progetto abbiamo imparato le funzionalità di alcuni componenti che prima non conoscevamo bene, come per esempio il sensore ad ultrasuoni. Inoltre abbiamo appreso come sviluppare delle librerie usando un C++ e C, seppur usando del codice molto basilare. Abbiamo anche capito che non è così facile seguire un Gantt e rispettare le scadenze che ci eravamo prefissati, infatti spesso ci è capitato di non finire una fase in tempo oppure di finirla prima del previsto. Sicuramente questo progetto ha aiutato a migliorare il nostro modo di lavorare in team e ci riteniamo soddisfatti del percorso svolto.

## 7 Bibliografia

### 7.1 Sitografia

- <https://www.adrirobot.it/arduino/digispark/digispark.htm>, Scheda Digispark, 30-11-2018.
- <https://github.com/digistump/DigistumpArduino/releases/download/1.6.7/Digistump.Drivers.zip>, Digistump Arduino Release 1.6.7, 30-11-2018.
- <https://www.instructables.com/id/Digispark-DIY-The-smallest-USB-Arduino/>, Digispark DIY: the Smallest USB Arduino, 07-12-2018
- <http://digistump.com/wiki/digispark>, Getting Started with your Digispark or Digispark Pro, 12-12-2018.
- <http://fritzing.org/download/>, Fritzing, 12-12-2018.
- [http://www.radiofo.it/files/articoli/sku207366\\_programmare\\_att.pdf](http://www.radiofo.it/files/articoli/sku207366_programmare_att.pdf), Scheda tecnica Attinity85

31-01-2010

- [http://www.dmf.unisalento.it/~denunzio/allow\\_listing/ARDUINO/HC-SR04-GDN.pdf](http://www.dmf.unisalento.it/~denunzio/allow_listing/ARDUINO/HC-SR04-GDN.pdf), Manuale s'uso Sensore Ultrasuoni HC-SR04, 02-02-2019
- <https://www.dummies.com/programming/electronics/become-familiar-leds/>, Become familiar with leds, 03-02-2019.
- <https://www.dummies.com/programming/electronics/components/electronics-components-resistors/>, Electronics components: resistors 03-02-2019.
- <https://www.dummies.com/programming/electronics/what-is-a-potentiometer/>, What is a potentiometer?, 03-02-2019
- <https://www.keyence.com/ss/products/sensor/sensorbasics/ultrasonic/info/>, What is an Ultrasonic Sensor?, 04-02-2019
- <https://en.wikipedia.org/wiki/Servomotor>, Servomotor, 04-02-2019
- <http://www.learningaboutelectronics.com/Articles/What-is-a-normally-closed-push-button>, What is a Normally Closed Push Button?, 04-02-2019

## **8 Allegati**

---

- Diari di lavoro
- Librerie
- Codice d'esempio
- QdC
- Scheda d'utilizzo
- Schemi in Fritzing

# Diari



# **PROGETTO | Diario di lavoro - 14.11.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 14.11.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 13:45	Come prima cosa ci è stato presentato il nuovo progetto da consegnare entro il 25.01.2019 e ci è stato consegnato il Quaderno dei Compiti 2. Successivamente abbiamo determinato le coppie per svolgere il progetto. La nostra coppia è Mattia Lazzaroni e Mattia Toscanelli.
13:45 - 14:30	In seguito abbiamo iniziato a preparare su GitHub la struttura del nostro progetto progetto.
14:30 – 14:45	Per finire Abbiamo scritto il primo diario di lavoro.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 16.11.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 16.11.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Nelle prime due ore abbiamo svolto il secondo test teorico.
15:00 - 16:30	In seguito abbiamo posto le domande preparate settimana scorsa e ci sono state date maggiori informazioni riguardo il secondo progetto.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 21.11.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 21.11.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso della lezione abbiamo aggiunto nella documentazione lo scopo del progetto ed abbiamo realizzato il gantt della pianificazione.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 23.11.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 23.11.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Nelle prime due ore abbiamo svolto il capitolo "1.4 – Analisi del dominio" della documentazione.
15:00 - 16:30	In seguito abbiamo inserito nella documentazione il capitolo "1.8 – Analisi dei mezzi" ed abbiamo cominciato l'analisi e specifica dei requisiti.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 28.11.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 28.11.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso delle due ore abbiamo terminato il capitolo 1.5 "Analisi e specifica dei requisiti" e il capitolo 1.8 "Analisi dei mezzi".

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# PROGETTO | Diario di lavoro - 30.11.2018

---

Mattia Lazzaroni – Mattia Toscanelli

**Canobbio, 30.11.2018**

## Lavori svolti

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

Orario	Lavoro svolto
13:15 – 14:45	Durante le prime due ore abbiamo installato i driver per la scheda Digispark tramite la guida: <a href="https://www.adrirobot.it/arduino/digispark/digispark.htm">https://www.adrirobot.it/arduino/digispark/digispark.htm</a> e ne abbiamo testato il funzionamento.
15:00 - 16:30	In seguito abbiamo realizzato il design dello schema e il design del circuito.

## Problemi riscontrati e soluzioni adottate

## Punto della situazione rispetto alla pianificazione

## Programma di massima per la prossima giornata di lavoro

---

# **PROGETTO | Diario di lavoro - 05.12.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 05.12.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso della lezione abbiamo cominciato il design delle librerie e saldato il nostro Digispark.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 07.12.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 07.12.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Nel corso delle prime ore abbiamo sistemato lo schema del circuito secondo le informazioni forniteci dai professori.
15:00 - 16:30	Successivamente abbiamo saldato il secondo Digispark tramite un saldatore e dello stagno. Infine abbiamo testato il funzionamento del circuito con il potenziometro e il led.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 12.12.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 12.12.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso delle due ore abbiamo testato il funzionamento del nostro Digispark con gli esempi basici del software di Arduino (Blink, Fade).

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 14.12.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 14.12.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso delle prime due ore abbiamo continuato la realizzazione delle librerie .
15:00 - 16:30	Successivamente, oltre al continuo della realizzazione delle librerie, abbiamo finito lo schema del circuito e lo abbiamo adattato nella documentazione. Al termine della lezione abbiamo mandato per email lo schema realizzato ai professori.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 19.12.2018**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 19.12.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso della lezione abbiamo iniziato l'implementazione della seconda libreria, realizzata per il servo motor. In seguito abbiamo iniziato a realizzare gli esempi questa libreria.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# **PROGETTO | Diario di lavoro - 09.01.2019**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 09.01.2018**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso delle due ore abbiamo realizzato l'UML delle librerie e abbiamo continuato il manuale.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

## **Programma di massima per la prossima giornata di lavoro**

---

# PROGETTO | Diario di lavoro - 11.01.2019

Mattia Lazzaroni – Mattia Toscanelli

**Canobbio, 11.01.2019**

## Lavori svolti

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

Orario	Lavoro svolto
13:15 – 14:45	Durante il corso delle prime due ore abbiamo cominciato a progettare la terza libreria ed abbiamo continuato la realizzazione del manuale per la seconda libreria.
15:00 - 16:30	Nelle seconde due ore abbiamo continuato ancora lo sviluppo degli esempi per il manuale ed abbiamo iniziato a svolgere la terza libreria.

## Problemi riscontrati e soluzioni adottate

## Punto della situazione rispetto alla pianificazione

Al momento siamo ad un punto che coincide con la nostra pianificazione.

## Programma di massima per la prossima giornata di lavoro

Nella prossima lezione vorremmo terminare tutte le librerie e il manuale, così da continuare con la documentazione.

# **PROGETTO | Diario di lavoro - 16.01.2019**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 16.01.2019**

## **Lavori svolti**

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso delle due ore abbiamo finito di descrivere gli esempi della seconda libreria nel manuale ed abbiamo continuato la realizzazione della terza libreria.

## **Problemi riscontrati e soluzioni adottate**

Abbiamo riscontrato una problema con un led che è bruciato e abbiamo dovuto cambiarlo.

## **Punto della situazione rispetto alla pianificazione**

Al momento ci troviamo in una situazione che coincide con la nostra pianificazione.

## **Programma di massima per la prossima giornata di lavoro**

Il programma per la prossima giornata di lavoro è quello di finire la terza libreria, iniziare ad aggiungere questa terza libreria nel manuale ed infine portarsi più avanti possibili con la documentazione.

---

# PROGETTO | Diario di lavoro - 18.01.2019

---

Mattia Lazzaroni – Mattia Toscanelli

**Canobbio, 18.01.2019**

## Lavori svolti

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

Orario	Lavoro svolto
13:15 – 16:30	Durante il corso della lezione abbiamo proceduto con la realizzazione delle terza ed ultima libreria ed abbiamo continuato la stesura della documentazione.

## Problemi riscontrati e soluzioni adottate

In una libreria abbiamo utilizzato “serial.write()” e questo metodo stampava dei caratteri strani. Per evitare questo problema abbiamo utilizzato “serial.print()”.

## Punto della situazione rispetto alla pianificazione

Siamo un attimo indietro rispetto alla pianificazione dato che oggi dovevamo finire la terza libreria.

## Programma di massima per la prossima giornata di lavoro

Nella prossima giornata di lavoro finiremo le terza libreria.

---

# PROGETTO | Diario di lavoro - 23.01.2019

---

Mattia Lazzaroni – Mattia Toscanelli

**Canobbio, 23.01.2019**

## Lavori svolti

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

Orario	Lavoro svolto
13:15 – 14:45	Nel corso della lezione abbiamo cominciato a sistemare il manuale ed abbiamo continuato a realizzare la progettazione nella documentazione.

## Problemi riscontrati e soluzioni adottate

Ci siamo accorti di aver capito male come realizzare la scheda d'utilizzo, dunque come già sopracitato abbiamo cominciato a risistemare il manuale.

## Punto della situazione rispetto alla pianificazione

Visto questo ultimo imprevisto ci troviamo un po' in ritardo coi tempi.

## Programma di massima per la prossima giornata di lavoro

Nella prossima giornata di lavoro dovremmo continuare la riscrittura del manuale e continuare la documentazione.

---

# **PROGETTO | Diario di lavoro - 25.01.2019**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 25.01.2019**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 16:30	Durante il croso delle due ore abbiamo ripreso a sistemare il manuale e abbiamo continuato a svolgere la documentazione.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

Al momento siamo leggermente in ritardo rispetto a quanto pianificato ma abbiamo intenzione di recuperare il lavoro a casa.

## **Programma di massima per la prossima giornata di lavoro**

Nella prossima giornata di lavoro vorremmo giungere alla conclusione del manuale e iniziare a svolgere la parte di implementazione nella documentazione.

---

# **PROGETTO | Diario di lavoro - 30.01.2019**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 30.01.2019**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 16:30	Durante il corso della lezione abbiamo continuato a svolgere il manuale e la parte di implementazione nella documentazione. Inoltre abbiamo cercato un modo alternativo per disegnare gli schemi elettrici ma senza risultati.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

Al momento ci troviamo in ritardo rispetto alla pianificazione ma dato che la consegna del progetto è stata rimandata siamo in tempo a finire il tutto.

## **Programma di massima per la prossima giornata di lavoro**

Nella prossima giornata di lavoro vorremmo finire il manuale definitivamente e la parte di implementazione della doc.

---

# **PROGETTO | Diario di lavoro - 01.02.2019**

---

**Mattia Lazzaroni – Mattia Toscanelli**

**Canobbio, 01.02.2019**

## **Lavori svolti**

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 14:45	Durante il corso delle lezione abbiamo finito completamente il manuale la parte di implementazione della documentazione.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

Al momento siamo quasi riusciti a recuperare il “ritardo” rispetto alla pianificazione.

## **Programma di massima per la prossima giornata di lavoro**

Nella prossima giornata di lavoro finiremo i test case e tutte le conclusioni della documentazione

---

# **PROGETTO | Diario di lavoro - 06.02.2019**

---

Mattia Lazzaroni – Mattia Toscanelli

**Canobbio, 06.02.2019**

## **Lavori svolti**

---

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

<b>Orario</b>	<b>Lavoro svolto</b>
13:15 – 16:30	Nella lezione di oggi abbiamo finalmente concluso la documentazione, inserendo i test case che mancavano e abbiamo inserito i punti mancanti nel manuale. Inoltre abbiamo controllato e corretto la grammatica di tutte le librerie e della documentazione.

## **Problemi riscontrati e soluzioni adottate**

## **Punto della situazione rispetto alla pianificazione**

Al momento abbiamo recuperato i ritardi e siamo in linea con la pianificazione.

## **Programma di massima per la prossima giornata di lavoro**

Nella prossima giornata di lavoro realizzeremo la presentazione.

---

---

## 1 INFORMAZIONI GENERALI

<b>Allievo/i</b>	Nome:	Cognome:
		
<b>Luogo di lavoro</b>	Aula 417 (ex A-413) – Scuola d'Arti e Mestieri Trevano	
<b>Orientamento</b>	<input checked="" type="checkbox"/> 88602 Informatica aziendale	
<b>Docente responsabile</b>	Nome:	Cognome:
		
<b>Periodo</b>		
<b>Orario di lavoro</b>	Secondo orario scolastico 1° semestre	
<b>Numero di ore lezione</b>		
<b>Pianificazione (in H o %)</b>	Analisi: 10%	
	Implementazione: 40%	
	Test: 20%	
	Documentazione: 30%	

---

## 2 PROCEDURA

- L'allievo realizza il lavoro autonomamente o con il gruppo al quale è assegnato, sulla base del quaderno dei compiti ricevuto il 1 ° giorno.
- Il quaderno dei compiti è presentato, commentato e discusso con l'allievo. Con la sua firma, l'allievo accetta il lavoro proposto.
- L'allievo ha conoscenza della scheda di valutazione all'inizio del lavoro.
- L'allievo è responsabile dei suoi dati.
- In caso di problemi gravi, l'allievo avverte immediatamente il docente responsabile.
- L'allievo ha la possibilità di chiedere aiuto, ma deve menzionarlo nella documentazione.
- Alla fine del tempo a disposizione per la realizzazione del progetto, l'allievo deve inviare via email il progetto al docente responsabile. In parallelo, una copia cartacea della documentazione dovrà essere fornita sempre al docente responsabile. Quest'ultima deve essere in tutto identica alla versione elettronica.

---

### 3 TITOLO

Sistema didattico per Arduino con libreria per attuatori e relativa documentazione

---

### 4 HARDWARE E SOFTWARE DISPONIBILE

1 PC della scuola + programmi concordati con i formatori

---

### 5 PREREQUISITI

Conoscenza Arduino

---

### 6 DESCRIZIONE DEL PROGETTO

Si tratta di sviluppare una nuovo prodotto didattico da utilizzare con gli allievi del terzo anno delle scuole medie che vengono a visitare la scuola durante le giornate di porte aperte Promtec.

Il prodotto da sviluppare è composto da un Arduino USB (mini DigiSpark), dei componenti elettronici da usare come attuatori, delle librerie/esempi di codice per l'utilizzo dei componenti e una guida che aiuti l'utente all'uso dell'Arduino e al montaggio del circuito.

In particolare:

- Verificare che tutto i componenti del progetto siano funzionanti.
- Creare per ogni combinazione di attuatori una libreria con del codice d'esempio.
- Per ogni combinazione di attuatori, creare almeno tre differenti esempi di codice.
- Tutto il codice deve essere ben commentato, in modo che gli allievi possano utilizzare/modificare il prodotto in maniera facile e veloce.
- Preparare una procedura di test, per controllare che tutti gli Arduino possano funzionare con tutte le combinazioni di attuatori che verranno proposti.
- Produrre una guida/scheda d'utilizzo, per l'installazione dell'Arduino e per ogni tipologia di attuatori, da consegnare insieme al prodotto durante il corso.
-

---

## 7 RISULTATI FINALI

L'allievo è responsabile della consegna al docente responsabile di:

- Una pianificazione iniziale (entro la prima settimana)
- Una documentazione della pianificazione e progettazione
- Una documentazione dell'implementazione e test
- Un diario di lavoro giornaliero entro le 18:00
- *Le librerie prodotte, gli schemi elettrici di collegamento ed i rispettivi Sketch di applicazione delle singole librerie con le rispettive spiegazioni*

---

## 8 PUNTI TECNICI SPECIFICI VALUTATI

La griglia di valutazione definisce i criteri generali secondo cui il lavoro dell'allievo sarà valutato (documentazione, diario, rispetto dei standard, qualità, ...).

Inoltre, il lavoro sarà valutato sui seguenti 7 punti specifici (punti da A14 a A20):

1. 159, *Analisi del problema (programmazione)*
2. 124, *Ipotesi di test, casi di test*
3. 185, *Rilevamento di errori*
4. 228, *Manuale utente*
5. 229, *Valutazione*
6. 164, *Codifica: Gestione degli errori*
7. 123, Commenti del codice sorgente

---

## 9 FIRMA

**Allievo**

(luogo e data)

**Docente responsabile**

(luogo e data)