

Argomenti per Lezioni Arduino Base

Author: Andrea Manni

Copyright: GFDL

Version: 0.1

Appunti e materiali per le lezioni del corso base su Arduino e Raspberry Pi. Questo documento e' da considerarsi come una traccia degli argomenti considerati e non il manuale definitivo delle lezioni.

Indice degli argomenti

1 Il corso	3
1.1 Contesto	3
1.2 Finalita'	3
2 Introduzione	3
2.1 Avvertenze	3
2.2 Blink	4
2.2.1 Pratica	4
2.3 Verifica e compilazione	4
3 Input	4
3.1 Cicli Condizionali	4
3.2 Ottimizzazioni	4
3.2.1 Eventuali	4
4 Analisi ulteriore	5
5 Seriali	5
6 Output	5
6.1 PWM	5
6.2 Funzioni	5
6.3 Piezo	5
6.4 Calibrare l'input	5
6.5 Usare un sensore di luminosita'	5
7 Motori	5
7.1 Servo motori	5
7.2 Motori passo-passo	6
7.3 Motori brushless	6
7.4 Caratteristiche generali	6
8 Elettronica di base	6
8.1 Eventuale: saldature	6
8.2 Approfondimenti	6
9 Linux	6
9.1 Sketch	6
10 Bibliografia	6
10.1 Risorse on line	7
10.2 Utilizzi	7
10.3 Progetti	7

1 Il corso

Il corso di base di Arduino e' rivolto a chi si avvicina per la prima volta al mondo dei makers, proponendosi di fornire le basi di elettronica, informatica e programmazione sia per orientarsi che per una prima interazione nell'innovativo mondo dell'elettronica digitale.

Obiettivo del corso e' partire dalla piattaforma Arduino come primo approccio all'elettronica digitale, lavorare sulle basi di elettronica, informatica e programmazione che poi permetteranno di sviluppare anche su altre soluzioni come RasPi. Le varie tematiche verranno affrontate praticamente dagli studenti con prototipi funzionanti basati su circuiti elettrici nei quali vari tipi di sensori saranno connessi al microcontroller Arduino che elaborera' questi dati per poi interagire con il mondo reale comandando vari tipi di attuatori come motori, luci LED, altoparlanti.

1.1 Contesto

Negli anni recenti molto e' cambiato nel panorama dell'elettronica, la contaminazione con gli ambienti aperti del settore informatico, la disponibilita' di sistemi miniaturizzati a basso costo ha reso disponibili nuove piattaforme come Arduino o RaspBerryPi, soluzioni economiche e flessibili.

Grazie a queste ed altre tecnologie e alla rinnovata cultura di condivisione dei Makers oggi sono alla portata dell'hobbista soluzioni che, per via degli alti costi e dell'esclusivita' delle varie implementazioni, fino a pochi anni fa erano esclusiva dell'automazione industriale.

Il Free Software, Open Source e Open Hardware, Crowdfunding hanno rivoluzionato l'elettronica con una serie di dispositivi che hanno portato la creativita' elettronica nella disponibilita' dei makers, partendo dai micro controller piu' semplici ed economici come Arduino fino a veri e propri micro computer come Raspberry Pi basati su Gnu/Linux. Soluzioni flessibili ed economiche che a loro volta supportano la costruzione di nuovi strumenti come le stampanti 3D, offrendo ad un vasto pubblico la possibilita' di dedicarsi a robotica, domotica, veicoli autonomi. Una nuova generazione di hardware aperto da rendere *smart* con il software libero, oggetti sempre piu' connessi tra loro e ricchi di funzionalita'.

1.2 Finalita'

Tutta questa flessibilita' e disponibilita' di nuove soluzioni puo' pero' disorientare chi si avvicina a queste tecnologie, sia chi e' ai primi passi che coloro che magari hanno precedenti esperienze con solo alcune delle discipline che si amalgamano con Arduino. Scopo del corso e' quindi stabilire delle fondamenta di informatica e elettronica per poi poter crescere sia con Arduino che con le altre tecnologie alla moda.

Durante lo svolgimento del corso i partecipanti impareranno a programmare in Arduino C in ambiente Gnu/Linux utilizzando una breadboard e diversi input/output.

2 Introduzione

- Cos'e' un microcontroller (tutto compreso, memoria calcolo input-output)/ attuatore
- Cenni alle classi di elaboratori
- Cos'e' l'informatica e la programmazione
- Elettronica e elettronica digitale

Perche' l'informatica ha bisogno di un attuatore: interazione con il reale. Elettronica digitale: vantaggi di usare un software (esempio di un bottone, logica booleana): cablatura istantanea, aggiunta di features.

2.1 Avvertenze

Precauzioni per non danneggiare la scheda durante l'uso:

- Applicare materiale isolante (fondo in polistirolo) sotto alla scheda.
- Applicare isolante al connettore USB per staccare il cavo senza statica.

- Solo la porta 13 ha una resistenza integrata, per tutte le altre usare una resistenza da ~300.
- Non usare Arduino come un trasformatore!

2.2 Blink

Analisi di un sketch:

- Struttura (setup, loop, input output)
- Fondamenti di programmazione: i 4 elementi base (fare descrizione completa dopo aver fatto cicli - INPUT).
- Dichiarazione di variabili: LED
- Funzioni per cambio di stato: digitalWrite / delay - output

2.2.1 Pratica

Fare accendere il LED per 1/10 di secondo # Far spegnere il LED per 1/10 di s.

Descrivere una istruzione

- Far accendere il LED per 1/10 di secondo e un secondo

Questa e' una sequenza di istruzioni, accenno alle funzioni, es delay() .

2.3 Verifica e compilazione

Codice sorgente e codice oggetto, compilazione del codice. Controllo formale, preprocessor (#define constantName value).

Eseguire procedure manualmente con makefile, visualizzazione codice oggetto.

3 Input

Esempio con un input, primo esempio con un bottone che fa accendere un LED.

Invertire il circuito del bottone / parametro della luminosita' per spiegare i *vantaggi dell'elettronica digitale* rispetto a elettronica "cablata" (cenni a PWM, varie letture degli eventi di *click*).

3.1 Cicli Condizionali

Eventualmente usare un generatore random (esercizio per testa / croce) per introdurre i cicli condizionali.

3.2 Ottimizzazioni

Note

Questa parte andra' affrontata dopo i motori (servo inclusi).

De-bouncing, multitasking con millis(), identificare il cambio di stato.

Sensori: dilatare i tempi di lettura, utilizzare valori medi di piu' letture (smoothing), calibrazione dei sensori utilizzando seriale e riferimenti.

3.2.1 Eventuali

Trasformazioni di input: da sensore luminoso a piezo: Pitch follower

4 Analisi ulteriore

Introdurre i data types

Cicli iterativi: for - while

5 Seriali

Inviare dati via seriale per debugging.

6 Output

LED, motori, servomotori, piezo, seriale.

6.1 PWM

Esempio con un "byte" di "brightness++" per aumentare la luminosita' di un LED. Introdurre eventualmente i cicli "for" .

6.2 Funzioni

Creare una funzione con un ciclo for per aumentare / diminuire la luminosita' in base a un parametro passato alla funzione.

6.3 Piezo

Emettere suoni e melodie tramite un trasduttore piezoelettrico. - Onde sonore: frequenze e pitch. - Sketch di esempi in Digital

(Sketches in multiple tabs, array). Pitch follower: trasformazione input di un sensore photo -> onde sonore tramite piezo.

6.4 Calibrare l'input

Calibrare l'input di un potenziometro / sensore: identificare valori minimi, massimi, offset e stabilire una formula: "Range = (1024 - offset) * 1024 / (1024 - offset) " .

Utilizzare "map()" per fare la stessa cosa. Caso specifico: inversione tra "1024 <-> 256" : usare un fattore 4.

6.5 Usare un sensore di luminosita'

Utilizzare come sorgente di input la resistenza rilevata da un sensore di luminosita', mappare l'input del sensore su un LED PWM / seriale.

7 Motori

Utilizzo di un motore 5v ~14mAh direttamente su Arduino tramite un transistor e diodo. Variare la velocita' tramite PWM e un ciclo for, utilizzare un potenziometro come input analogico per variare la velocita', trovare il valore minimo di carico per attivare il motore tramite debugging seriale.

7.1 Servo motori

Differenze rispetto a un motore DC, scopi di utilizzo. Caratteristiche: coppia, velocita', peso, alimentazione. Funzionamento: analogici e digitali, riduttori plastici e metallici, bearings.

Sketch Base e Knob. Utilizzare librerie esterne.

- <http://handyboard.com/hb/faq/hardware-faqs/dc-vs-servo/>

7.2 Motori passo-passo

Funzionamento, caratteristiche di utilizzo (consumo - coppia), campi di utilizzo (automazione power tools), differenze rispetto a servo (controllo posizione) e motori normali.

7.3 Motori brushless

Differenze e caratteristiche rispetto ai motori a spazzole. Uso di una ESC, BEC.

7.4 Caratteristiche generali

Potenza, KV, voltaggio utilizzabile, ampere massimi, potenza / peso, efficienza.

8 Elettronica di base

Si dovranno introdurre:

- Legge di Ohm
- Serie e parallelo, in particolare per batterie e resistenze
- Caratteristiche e uso dei LED: come calcolare resistenze necessarie
- Uso di transistor per motori DC

8.1 Eventuale: saldature

Guida di base alle piccole saldature: cavi intrecciato, cavi dritti, PCB. Dissaldare.

8.2 Approfondimenti

Resistenze: resistività in base a sezione e lunghezza: effetti su serie e parallelo. (Fisica) Semiconduttori: resistenze -> diodi -> transistor : cenni sul funzionamento in base ai possibili range di spostamento degli elettroni.

9 Linux

Argomenti specifici per utilizzare integrare Arduino in ambiente Linux.

- Installazione
- Leggere informazioni da seriale (redirezione INPUT, verso un file, screen), mandare informazioni (echo, cat)
- Eventuale: editor alternativi, compilazione e upload manuale (creare un make file).

9.1 Sketch

Installare sketch per Arduino.

- <http://webtechie.be/2014/05/08/scratch-and-arduino-on-linux/>
- <http://s4a.cat/>

10 Bibliografia

Testi consigliati, non richiesti.

- Beginning C for Arduino: Learn C Programming for the Arduino

- Practical Electronics for Inventors
- Arduino Language Reference
- Getting Started With Arduino 2nd Edition
- Arduino for Dummies

10.1 Risorse on line

- <http://arduino.cc/en/Reference/HomePage>
- <http://www.ladyada.net/learn/arduino/index.html>
- <https://learn.adafruit.com/series/learn-arduino>
- <https://wiki.archlinux.org/index.php/arduino>
- <http://arduinoprincipiante.blogspot.it/2013/04/presentazione.html>

10.2 Utilizzi

- Input tramite manipolazione oggetti fisici.
- Stazioni per rilevamento dati: metereologiche, movimento.
- Attuatori per controllo numerico applicato a vari strumenti (stampanti 3D, frese , laser).
- Device per lettura di sensori da utilizzare con smartphones - computer
- Adattatore per sensori verso IoT
- Attuatore per device IoT: arduino - WiFi - rele' = accensione / spegnimento
- RFID per device domestici: smartphone in contesti diversi = diversi profili
- Interfaccia input per tutti gli scenari in cui non si puo' usare un touch screen / tastiera
- Domotica: irrigazione, controllo temperatura ambienti e acqua.
- Robotica: integrazione di apparecchiature di sorveglianza, robot domestici (aspirapolvere) e da giardino (macchine agricole unmanned in miniatura).
- Controllo droni, gyro, GPS, viewpoint. Ardupilot <http://diydrones.com/notes/ArduPilot>
- Prototipi per macchine di dimensioni performance superiori.

10.3 Progetti

- Chorus – United Visual Artists: <http://www.elmsly.com/Chorus-United-Visual-Artists>
- Ardupilot <http://diydrones.com/notes/ArduPilot>