R2Pi0

Tosatto Davide Riccardo Grespan

11 febbraio 2017

Indice

1 Obiettivi del progetto

L'obiettivo del progetto è di creare una riproduzione in scala di R2-D2, noto personaggio robotico della serie di Star Wars.

Figura 1: R2D2

La ricostruzione comprende:

1. movimento sul piano attraverso ruote (come nel film)
2. riproduzione di suoni simili a quelli originali attraverso un buzzer, per dare quel tocco di retro che non guasta mai
3. luci led e finto proiettore, comunque emulato da un led
4. rilevamento degli ostacoli e corrispondente arrabbiatura del robot se lo mandiamo a tutta velocità verso uno di essi
5. funzione follow che gli fa mantenere una certa distanza dall'oggetto (o dalla persona) che lo precede
6. un rudimentale controllo vocale
7. esoscheletro in carta che riproduce le fattezze del robot

Questa prima versione non avrà la possibilità di ruotare la testa e cambiare inclinazione, questo per mancanza di tempo e di componenti (ulteriori motori, ulteriori driver, contatti girevoli per le luci che stanno sulla testa)

2 Componenti e scelte progettuali

2.1 Scheda di controllo

Iniziamo dal cuore del progetto: *la scheda di controllo.*

Le opzioni erano sostanzialmente due: Raspberry Pi0 o Arduino Uno.

Figura 2: Raspberry Pi0

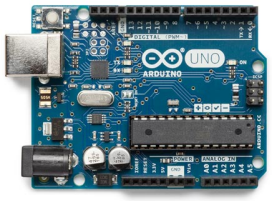

Figura 3: Arduino Uno

Per la maggior parte dei compiti le due schede erano sostanzialmente intercambiabili, in particolare:

1. *GPIO - General Purpose Input Output*: entrambi i dispositivi hanno tutte le interfacce che ci servono, ossia qualche I/O digitale e uscite PWM. Entrambi non possono dare in uscita elevate correnti, ma questo non importa perché il controllo di ogni componente è sempre mediato da transistor esterni;

2. *Alimentazione*: entrambi i dispositivi possono essere alimentati da usb, rendendo quindi possibile l'uso di una normale batteria esterna da telefono per l'alimentazione;

3. *Potenza di calcolo*: entrambi i dispositivi in esame hanno la potenza di calcolo necessaria per svolgere ognuno dei compiti necessari agli scopi del progetto.

Le seguenti motivazioni ci hanno fatto propendere per Raspberry Pi0:

1. *Semplicità di sviluppo*: grazie al fatto di avere a disposizione un sistema operativo completo, possiamo utilizzare linguaggi di più alto livello rispetto al C fornito da Arduino. Questo ci permette di creare codice meglio organizzato e più facilmente espandibile in futuro. La nostra scelta per il linguaggio è ricaduta su *Python*, in quanto semplice, conciso ed efficace;

2. *Memoria a disposizione*: grazie alla memoria disponibile, sensibilmente superiore a quella di Arduino, ci è permesso di integrare più suoni. Inoltre, in futuro, sarà sempre possibile migliorare la qualità degli stessi.

3. *Facilità di connessione*: Raspberry Pi0 risulta più facile da connettere al mondo in quanto basta collegare un adattatore wifi o bluetooth all'USB. Arduino invece richiede appositi shield, costosi e non utilizzabili su altre schede, almeno non senza opportune modifiche

4. *Prezzo*: Raspberry Pi0 costa 5$, molto meno di un Arduino, e, nonostante questo, offre infinite possibilità in più in quanto propone un sistema completo di tutto.

Certo Raspberry Pi0 ha dalla sua un grande svantaggio: il consumo.

Arduino a vuoto consuma circa 50mA, mentre Raspberry Pi0 arriva a 65mA in idle, ma si alza parecchio se il carico di lavoro aumenta, questo perchè il processore di Raspberry Pi0 è molto più performante di quello di Arduino, inoltre, montando un sistema operativo completo, Raspberry Pi0 ha anche diverso overhead in campo di potenza computazionale e consumo, cosa non trascurabile.

Chiaramente poi i dispositivi di comunicazione comportano un innalzamento dei consumi, ma questo su entrambe le schede.

In conclusione, i vantaggi nell'utilizzo di Raspberry Pi0 ci sono sembrati schiaccianti.

2.2 Comunicazione

Anche qui, due opzioni iniziali: Bluetooth o WiFi.

Abbiamo optato per una connessione Bluetooth per i seguenti motivi:

1. *Consumo*: il Bluetooth risulta avere un consumo energetico inferiore rispetto al WiFi, portandoci ad avere una durata della batteria leggermente superiore (non abbiamo dati precisi)

2. *Semplicità d'uso*: il Bluetooth è pensato per connessioni punto a punto, ossia esattamente la funzione che serve a noi. Il Wifi, invece, non è pensato per quello scopo e quindi relegherebbe all'utilizzo nei soli spazi dotati di rete WiFi oppure alla configurazione di una rete ad-hoc che risulta comunque un'operazione non banale e spesso mal funzionante (soprattutto su Linux)

3 Schemi circuitali

3.1 Controllo motori

Per il controllo dei motori abbiamo deciso di utilizzare una versione leggermente modificata del ponte H. I nomi dei componenti sono presi da fig. 4. É ovviamente necessario uno di questi circuiti per ogni motore installato.

Passiamo alla rapida spiegazione del funzionamento del circuito. Il circuito ha due input: IN1 e IN2. Il comportamento è il seguente:

1. Se solo uno dei due fra IN1 e IN2 è attivo, nel motore scorre corrente, il verso dipende da quale dei due è alto.

2. Se sia IN1 che IN2 sono a 0, il motore resta completamente scollegato dal circuito, permettendogli di muoversi per inerzia

3. Se sia IN1 che IN2 sono a 1, il motore ha entrambi i terminali a massa. Questo chiude la bobina del motore creando un cortocircuito. Così facendo se si tenta di muovere il motore si originerà all'interno della bobina una corrente che genererà a sua volta un campo magnetico che si opporrà al movimento del motore. Questo ha come risultato un effetto frenante.

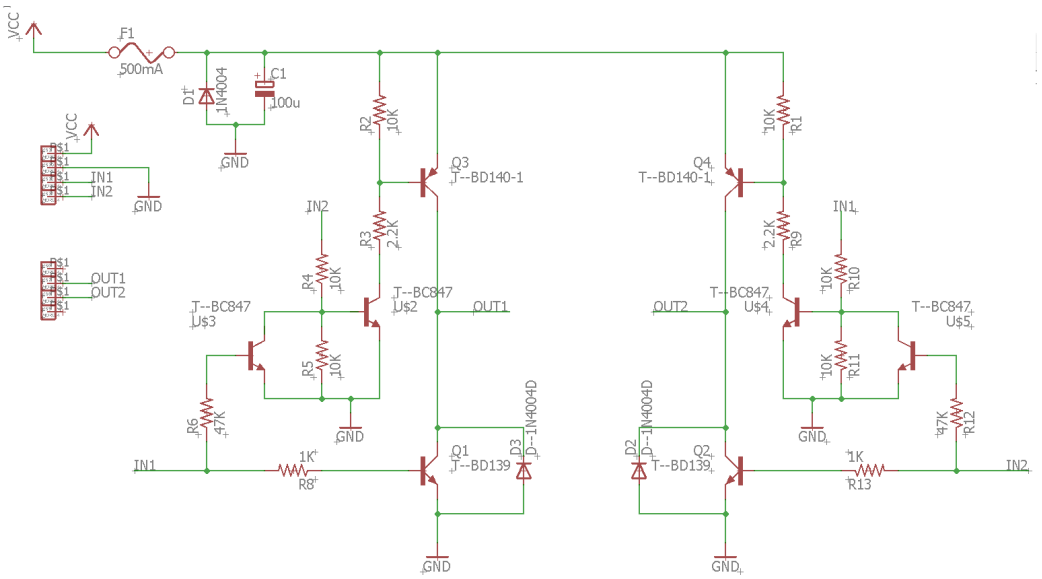

Figura 4: Schema del circuito di controllo dei motori

Passiamo alla disamina dei componenti e delle loro funzioni. Tutti i transistori sono BJT. Questa scelta è stata fatta perché i BJT permettono di fare passare correnti elevate e soprattutto hanno basse cadute di potenziale in saturazione, ossia circa 0.3V tra collettore ed emettitore. Mantenere cadute di potenziale basse è essenziale perché il tutto è alimentato a soli 5V.

I quattro transistor principali di potenza sono Q1, Q2, Q3 e Q4. La circuiteria è fatta in modo tale che né Q1 e Q3, né Q2 e Q4 conducano *mai* insieme. Questo avvenimento infatti produrrebbe un cortocircuito che danneggerebbe l'hardware.

Da misurazioni, si è visto che i motori consumano meno di 200mA l'uno, e si è dunque pensato di inserire per sicurezza un fusibile da 500mA, F1, sufficientemente permissivo da far funzionare il motore, ma abbastanza restrittivo per evitare di bruciare l'alimentatore in caso di corto, che comunque, salvo manomissioni, non dovrebbe avvenire.

Abbiamo poi altri quattro transistor che servono ad assicurare l'assenza di corti e ad implementare la funzionalità di cortocircuitare la bobina con entrambi gli ingressi a 1.
La funzione logica realizzata è quella mostrata in tabella 1

Da notare che U$2 e U$4 hanno un partitore resistivo sulla base. Questo

IN1	IN2	Q1	Q2	Q3	Q4
off	off	off	off	off	off
off	on	off	on	on	off
on	off	on	off	off	on
on	on	on	on	off	off

Tabella 1: Funzione logica implementata nel circuito

serve a ritardare la loro accensione, infatti gli input non sono realmente squadrati ma aumentano gradualmente con una certa pendenza, seppur elevata. Questo permette di ritardare l'attivazione dei transistor sopracitati. Infatti se sulla base servono 0.7V per permettere l'accensione, con il partitore è richiesto che IN2 sia già ad almeno 1.4V. Questo accorgimento serve quando attiviamo contemporaneamente IN1 e IN2, per dare il tempo a U$3 e U$1 di inibire l'accensione di Q3 e Q4 impedendo quindi cortocircuiti.