
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE E SCIENZE MATEMATICHE

---

Corso di laurea in
INGEGNERIA INFORMATICA E DELL'INFORMAZIONE

# Progettazione e realizzazione della struttura meccanica e del sistema di controllo di un robot esapode

Relatore:
Prof. Andrea Garulli

Candidato:
Alessandro Paghi

Anno Accademico 2015 – 2016

*Ringraziamenti*

Desidero ringraziare il Prof. Andrea Garulli per la disponibilità e la professionalità mostrata durante tutto il periodo di tempo dedicato alla realizzazione del prototipo e della documentazione scritta.

Ringrazio in particolar modo il Dott. Prof. Ing. Nicola Donatucci, docente di Elettronica presso ITI Sarrocchi, per essere stato il mio mentore nella disciplina elettronica ed avermi trasmesso per essa amore e passione, per avermi guidato durante la progettazione del sistema elettronico di bordo e, in particolar modo, per lo splendido e sentito rapporto che mai avrei creduto poter essere instaurato fra professore ed alunno.

Un ringraziamento al Prof. Angelo Gagliani, docente di Sistemi Informatici presso ITI Sarrocchi, per essere stato il mio mentore nella disciplina informatica ed avermi trasmesso per essa amore e passione.

Infine, ringrazio di cuore Gabriele Corsi, amico, compagno di avventure, collega e socio, per aver curato la struttura meccanica come in pochi avrebbero saputo fare, essermi stato vicino nei momenti difficili ed avermi spronato a dare sempre il meglio di me stesso.

# Indice generale

# Capitolo 1: Introduzione

Il presente lavoro riguarda la progettazione e la realizzazione della struttura meccanica e del sistema di controllo di un robot esapode. In particolare, è stata analizzata una struttura semplificata del robot esapode; sono state disegnate e realizzate le parti meccaniche; è stata progettata l'elettronica di bordo e realizzata la scheda elettronica contenente i processori adibiti al controllo del robot; è stato implementato un algoritmo per il controllo dei motori e infine sono stati progettati e testati alcune semplici strategie di controllo per lo svolgimento di movimenti elementari.

Per quanto riguarda la struttura meccanica e il sistema di bordo, essi sono stati progettati in modo da poter rendere il dispositivo in grado di svolgere una vasta gamma di movimenti e poter implementare una grande varietà di algoritmi di controllo.

In questo capitolo vengono introdotti i robot esapodi e le ragioni per cui la ricerca sui sistemi biologici è di ispirazione per la robotica.

## 1.1 Perché una struttura esapode?

Uno dei principali motivi per cui viene sviluppato un robot dotato di gambe è perché esso può arrampicarsi e superare un'ampia gamma di ostacoli che metterebbero invece in seria difficoltà un veicolo dotato di ruote.
È pur vero che sarebbe possibile costruire ruote più grandi per aggirare questo problema.
Inoltre costruire un sistema dotato di ruote è considerato molto più semplice che costruire un sistema dotato di gambe. Tuttavia un sistema munito di gambe è molto vantaggioso laddove sussista uno spazio di manovra ridotto od un terreno di lavoro particolarmente irregolare, il quale non consenta lo spostamento di veicoli con ruote.
Esempi di situazioni in cui è preferibile utilizzare un sistema dotato di gambe rispetto ad un sistema dotato di ruote includono cave, miniere, foreste e particolari edifici in costruzione.
Uno dei maggiori vantaggi dei robot muniti di gambe sta nel fatto che essi possono muoversi in spazi veramente ridotti in qualsiasi direzione e ruotare su se stessi di un qualsiasi angolo. Un altro indubbio vantaggio sta nel fatto che un sistema dotato di movimento tramite gambe può superare pericoli presenti nel terreno, quali buche o ostacoli.
Questi vantaggi sono condivisi dai robot indipendentemente dal numero di gambe che essi possiedono.
Tuttavia, i robot esapodi, e quelli con più di sei gambe, sono "staticamente stabili", mentre bipedi e quadrupedi non lo sono. La stabilità statica è la capacità di rimanere in piedi senza cambiare continuamente l'ingresso di controllo e si ottiene mantenendo sempre tre gambe di supporto al corpo.
Senza la stabilità statica un robot deve essere "dinamicamente stabile" oppure cadrebbe a terra.
Un quadrupede può essere considerato staticamente stabile se ogni gamba passa almeno ¾ del suo tempo appoggiata a terra, ma questa costrizione limita notevolmente la velocità di movimento dell'intero sistema. Risulta quindi che avere sei gambe permette di avere stabilità statica con il massimo utilizzo di tutte le gambe.

## 1.2 Ispirazione biologica

Dall'osservazione di sistemi biologici sono state derivate particolari strutture fisiche per i robot e metodi di controllo degli stessi.

Gli animali sono la prima ispirazione biologica nella ricerca robotica. Essi sono i migliori esempi di adattamento a luoghi complessi, luoghi in cui i robot sarebbero teoricamente in grado di operare.

Nell'analisi di un sistema biologico non è sempre necessario studiare l'intero apparato, ma, più semplicemente, ricavare informazioni solo su quello che può esserci effettivamente utile.

Se si suppone, per esempio, che la struttura fisica di una specie particolare sia ottimale per l'obiettivo che si vuol perseguire, risulta essere utile basare la struttura del dispositivo sulla struttura della specie studiata.

Inoltre è importante analizzare i metodi sensoriali che un animale utilizza per ricavare informazioni sul mondo che lo circonda, al fine di replicare una tecnologia simile per i robot.

# Capitolo 2: Struttura e cinematica

Questo capitolo analizza la struttura semplificata di un robot esapode, basata su segmenti e circonferenze.

## 2.1 Studio del corpo e dei giunti

Per semplificare la rappresentazione si è supposto il corpo rappresentabile perfettamente da una circonferenza centrata nell'origine degli assi e le gambe rappresentabili da semplici segmenti.

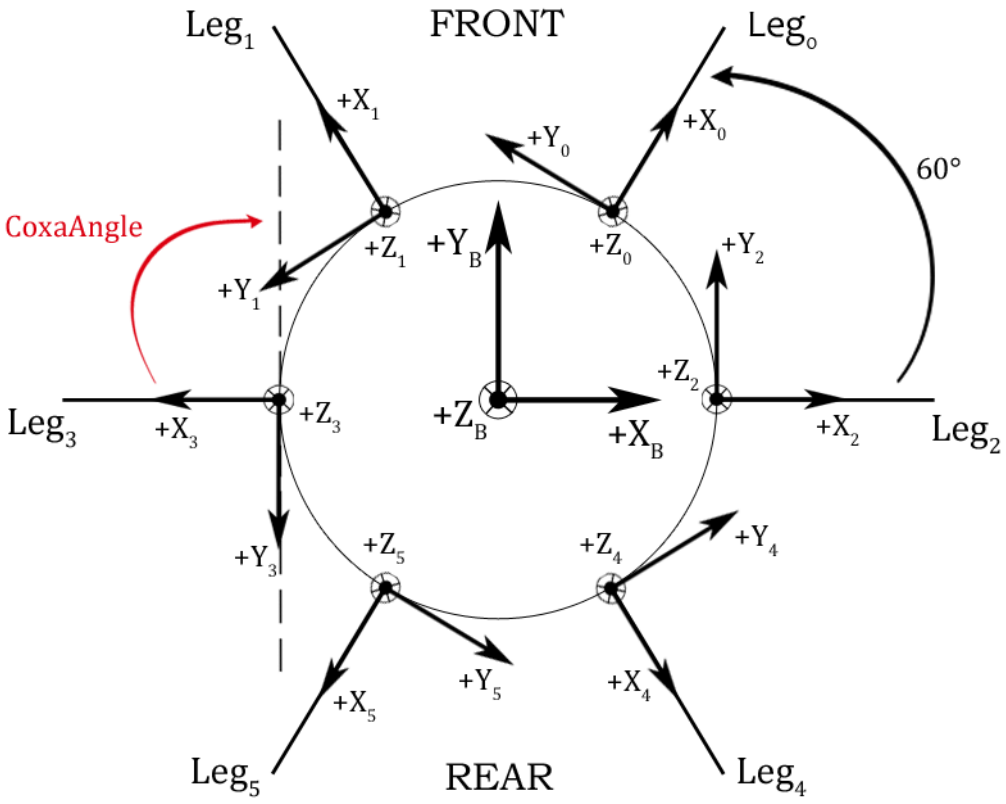



Figura 2.1: Modello semplificato del corpo

Posizionando ogni gamba a 60° di distanza dall'altra, la struttura dell'assieme risulta essere simmetrica rispetto all'asse X e all'asse Y.
Questo, permette notevoli semplificazioni nello studio del movimento di ogni singolo arto, consentendo di realizzare lo studio della cinematica inversa per una singola gamba.

Per l'analisi di un singolo arto è utile ricordare che esso è composto da:

- Un'anca: connette la struttura della gamba al corpo centrale;
- Un femore: connette l'anca alla tibia;
- Una tibia: è la parte finale della gamba e permette all'intera struttura di poggiare a terra.

Da notare che esistono più sistemi di riferimento: il sistema di riferimento Body $(X_B, Y_B, Z_B)$ e i sei sistemi di riferimento Leg $(X_i, Y_i, Z_i)$, i quali risultano essere sia traslati che ruotati rispetto al sistema di riferimento Body.

Studiando l'articolazione della gamba, si è supposto che ogni singola componente sia rappresentabile da un segmento.

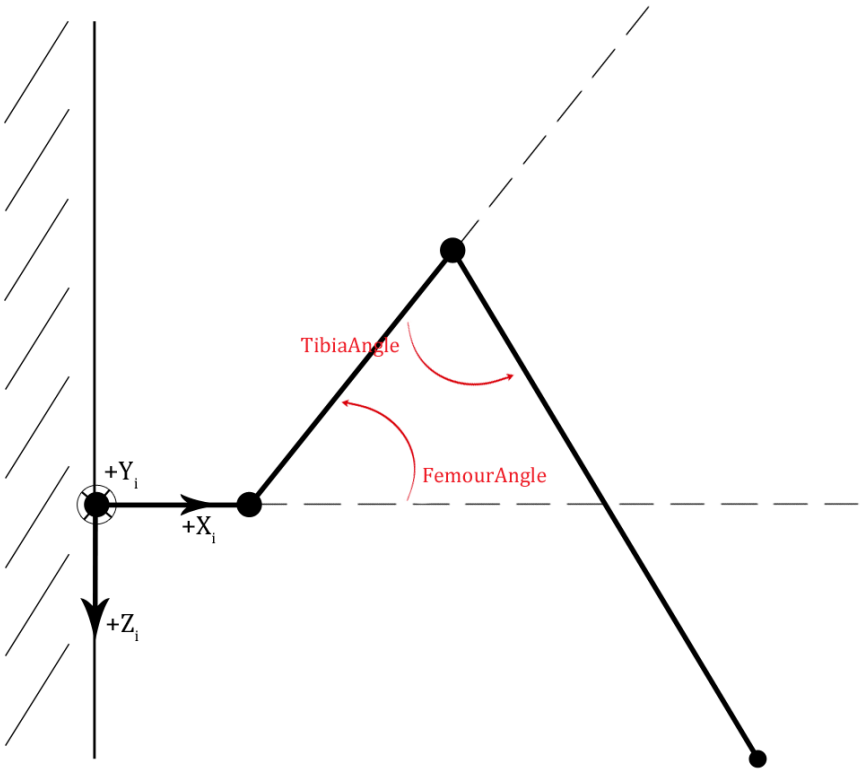



Figura 2.2: Modello semplificato della gamba

Si suppone, inoltre, che gli angoli indicati in figura, sino compresi in questi intervalli di valori:

- $-90° < CoxaAngle < 90°$;
- $-30° < FemourAngle < 90°$;
- $0° < TibiaAngle < 180°$;

## 2.2 Cinematica inversa per una gamba

Nel seguito vengono analizzate le equazioni che gestiscono il movimento dei 3 giunti.

La cinematica inversa è il processo di determinazione delle variabili di un oggetto articolato e flessibile (in questo caso gli angoli dei giunti), che soddisfino il raggiungimento di una posa desiderata, in base al posizionamento delle sole estremità (in questo caso il punto di appoggio P).

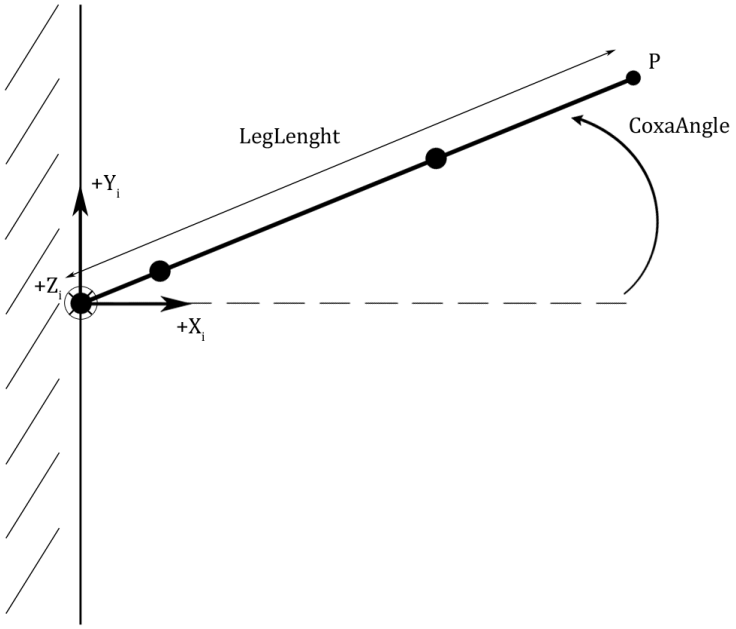



Figura 2.3: Modello per la cinematica della gamba

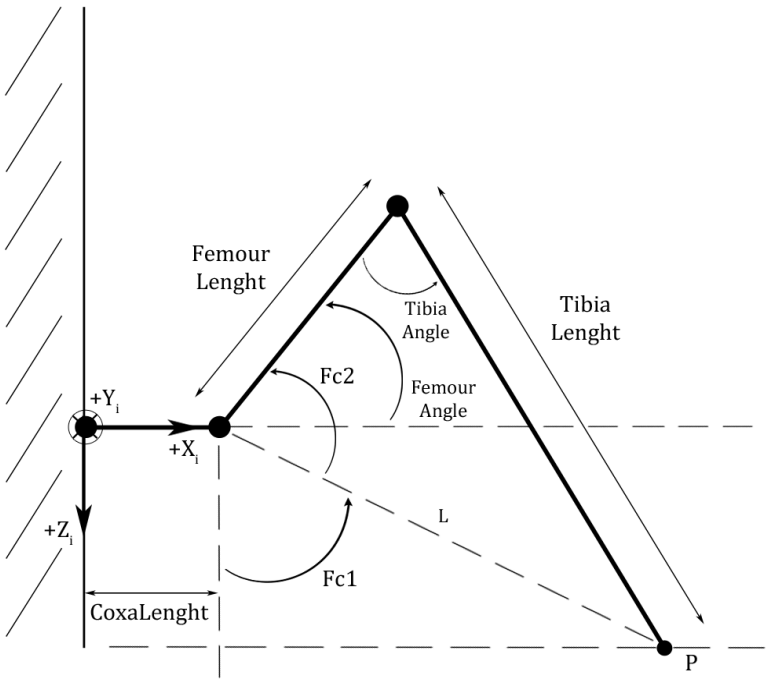



Figura 2.4: Modello per la cinematica della gamba

Supponiamo che $P(x, y, z)$ sia il punto che desideriamo che il nostro arto raggiunga; risulta che:

$$\frac{y}{x} = \tan(CoxaAngle)$$

$$CoxaAngle = \tan^{-1}\left(\frac{y}{x}\right)$$

$$LegLenght = \sqrt{x^2 + y^2}$$

$$L = \sqrt{(LegLenght - CoxaLenght)^2 + z^2}$$

$$Fc1 = \cos^{-1}\left(\frac{z}{L}\right)$$

$$Fc2 = \cos^{-1}\left(\frac{TibiaLenght^2 - FemourLenght^2 - L^2}{-2 * FemourLenght * L}\right)$$

$$FemourAngle = (Fc1 + Fc2) - \frac{\pi}{2}$$

$$TibiaAngle = \cos^{-1}\left(\frac{L^2 - TibiaLenght^2 - FemourLenght^2}{-2 * TibiaLenght * FemourLenght}\right)$$

## 2.3 Cinematica inversa per il corpo

### 2.3.1 Traslazioni del corpo

Nella geometria euclidea, una traslazione è una trasformazione affine dello spazio euclideo, che sposta tutti i punti di una distanza fissa nella stessa direzione.

Se si indica con $P_o(x_o, y_o, z_o)$ il punto pretraslazione, con $P_n(x_n, y_n, z_n)$ il punto postraslazione e con $[t_x; t_y; t_z]$ il vettore traslazione, definiti nel sistema di riferimento Body, risulta che:

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

### 2.3.2 Rotazioni del corpo

In matematica, e in particolare in geometria, una rotazione è una trasformazione del piano o dello spazio euclideo che sposta gli oggetti in modo rigido e che lascia fisso almeno un punto, nel caso del piano, o una retta, nel caso dello spazio. I punti che restano fissi nella trasformazione formano più in generale un sottospazio: quando questo insieme è un punto o una retta, si chiama rispettivamente il centro e l'asse della rotazione.

Matrice di rotazione attorno a $X_B$ di un angolo $\boldsymbol{\theta}_x$:

$$R_{X_B}(\boldsymbol{\theta}_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\boldsymbol{\theta}_x & -\sin\boldsymbol{\theta}_x \\ 0 & \sin\boldsymbol{\theta}_x & \cos\boldsymbol{\theta}_x \end{bmatrix}$$

Matrice di rotazione attorno a $Y_B$ di un angolo $\boldsymbol{\theta}_y$:

$$R_{Y_B}(\boldsymbol{\theta}_y) = \begin{bmatrix} \cos\boldsymbol{\theta}_y & 0 & \sin\boldsymbol{\theta}_y \\ 0 & 1 & 0 \\ -\sin\boldsymbol{\theta}_y & 0 & \cos\boldsymbol{\theta}_y \end{bmatrix}$$

Matrice di rotazione attorno a $Z_B$ di un angolo $\boldsymbol{\theta}_z$:

$$R_{Z_B}(\boldsymbol{\theta}_z) = \begin{bmatrix} \cos\boldsymbol{\theta}_z & -\sin\boldsymbol{\theta}_z & 0 \\ \sin\boldsymbol{\theta}_z & \cos\boldsymbol{\theta}_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

La matrice di rotazione finale risulta il prodotto delle tre, ovvero:

$$R_{(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z) = R_{X_B}(\boldsymbol{\theta}_x) * R_{Y_B}(\boldsymbol{\theta}_y) * R_{Z_B}(\boldsymbol{\theta}_z) =$$

$$= \begin{bmatrix} \cos\boldsymbol{\theta}_z \cos\boldsymbol{\theta}_y & -\cos\boldsymbol{\theta}_y \sin\boldsymbol{\theta}_z & \sin\boldsymbol{\theta}_y \\ \sin\boldsymbol{\theta}_x \sin\boldsymbol{\theta}_y \cos\boldsymbol{\theta}_z + \cos\boldsymbol{\theta}_x \sin\boldsymbol{\theta}_z & -\sin\boldsymbol{\theta}_x \sin\boldsymbol{\theta}_y \sin\boldsymbol{\theta}_z + \cos\boldsymbol{\theta}_x \cos\boldsymbol{\theta}_z & -\sin\boldsymbol{\theta}_x \cos\boldsymbol{\theta}_y \\ -\cos\boldsymbol{\theta}_x \sin\boldsymbol{\theta}_y \cos\boldsymbol{\theta}_z + \sin\boldsymbol{\theta}_x \sin\boldsymbol{\theta}_z & \cos\boldsymbol{\theta}_x \sin\boldsymbol{\theta}_y \sin\boldsymbol{\theta}_z + \sin\boldsymbol{\theta}_x \cos\boldsymbol{\theta}_z & \cos\boldsymbol{\theta}_x \cos\boldsymbol{\theta}_y \end{bmatrix}$$

Se si indica con $P_o(x_o, y_o, z_o)$ il punto prerotazione, con $P_n(x_n, y_n, z_n)$ il punto postrotazione e con $R_{(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z)$ la matrice di rotazione, definiti nel sistema di riferimento Body, risulta che:

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = R_{(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z) * \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix}$$

### 2.3.3 Rototraslazioni del corpo

Una rototrslazione è una trasformazione geometrica dello spazio eclideo composta da una rotazione ed un traslazione.

Se si indica con $P_o(x_o, y_o, z_o)$ il punto prima della rototraslazione, con $P_n(x_n, y_n, z_n)$ il punto dopo la rotoraslazione, con $[t_x; t_y; t_z]$ il vettore traslazione e con $R_{(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z)$ la matrice di rotazione, definiti nel sistema di riferimento Body, risulta che:

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = R_{(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z) * \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

### 2.3.4 Cambio di sistema di riferimento

La procedura descritta nel paragrafo precedente permette di rototraslare un punto, a patto che esso sia definito nel sistema di riferimento Body.
Quando si vuole muovere il corpo del robot è necessario, quindi, applicare una rototraslazione ai sei punti di origine degli arti. Questo permette l'individuazione di sei nuovi punti di orgine dati dalla rototraslazione applicata alle vecchie origini, definiti nel sistema di riferimento Body.

C'è da notare, però, che i punti di appoggio non sono definiti nel sistema di riferimento Body $(X_B, Y_B, Z_B)$, ma nel sistema di riferimento Leg $(X_i, Y_i, Z_i)$.
È necessario, perciò, un cambio di sistema di riferimento, in maniera tale che le sei nuove origini possano essere definite nei sistemi di riferimento locali.

Se si indica con $O_{i_B}(x_B, y_B, z_B)$ la nuova origine definita nel sistema Body $(X_B, Y_B, Z_B)$, con $O_{i_L}(x_L, y_L, z_L)$ la nuova origine definita nel sistema di riferimento Leg $(X_i, Y_i, Z_i)$, con $R_{O\,(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z)$ la matrice di rotazione che permette al sistema di riferimento Body di portarsi allo stesso angolo del sistema di riferimeto Leg, con $[t_{O_X}; t_{O_Y}; t_{O_Z}]$ il vettore traslazione che permette al sistema di riferimento Body di sovrapporsi al sistema di riferimento Leg, risulta che:

$$O_{i_L}(x_L, y_L, z_L) = R_{O\,(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z) * \left( O_{i_B}(x_B, y_B, z_B) - \begin{bmatrix} t_{O_X} \\ t_{O_Y} \\ t_{O_Z} \end{bmatrix} \right)$$

### 2.3.5 Procedura per il movimento del corpo

Questo paragrafo riassume la procedura che è necessario seguire per poter rototraslare il corpo di un robot esapode.

1) Definizione di $t_x, t_y, t_z$, coordinate di traslazione del centro del corpo rispetto alla posizione iniziale;
2) Definizione di $\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z$, angoli di rotazione del corpo rispetto alla posizione iniziale;
3) Calcolo della matrice di rotazione $R_{(X_B,Y_B,Z_B)}(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\theta}_z)$;
4) Calcolo dei sei nuovi punti di origine $O_{i_B}(x_B, y_B, z_B)$ degli arti, applicando la rototraslazione ai punti di origine iniziali;
5) Calcolo dei sei punti di origine $O_{i_L}(x_L, y_L, z_L)$ nel sistema di rifermento Leg, applicando il cambiamento di sistema di riferimento;
6) Calcolo del nuovo punto di appoggio P tramite le formule:

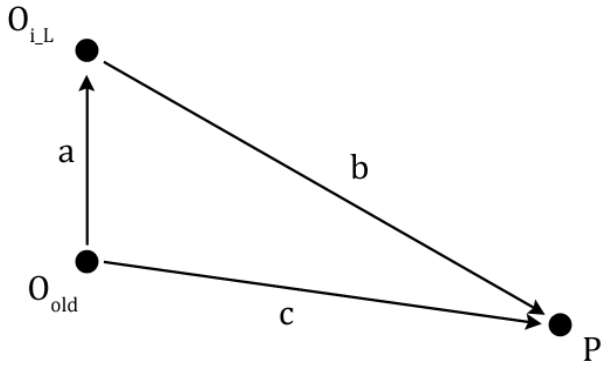



Figura 2.5: Modello per il calcolo dei nuovo punto di appoggio

$$\bar{c} = \bar{a} + \bar{b}$$
$$\bar{b} = \bar{c} - \bar{a}$$

Dove i vettori $\bar{a}, \bar{b}, \bar{c}$ sono definiti come in figura 2.5, nella quale $O_{i_L}(x_L, y_L, z_L)$ è il punto di origine dopo la rototraslazione definito rispetto al sistema di riferimento Leg, $O_L(0,0,0)$ è il punto di origine precedente definito rispetto al sistema di riferimento Leg, e $P(x_P, y_P, z_P)$ è il punto di appoggio definito nel sistema di riferimento Leg.

# Capitolo 3: Struttura meccanica

## 2.4 Metodo realizzativo e materiale utilizzato

Per realizzare la particolare struttura meccanica ideata è stato necessario ricorrere alla stampa 3D.

Per stampa 3D si intende la realizzazione di oggetti tridimensionali mediante produzione additiva, partendo da un modello 3D digitale. Il modello digitale viene prodotto con software dedicati e successivamente elaborato per essere poi realizzato, strato dopo strato, attraverso una stampante 3D.

Le stampanti 3D:

- Sono generalmente più veloci, più affidabili e più semplici da usare rispetto ad altre tecnologie per la produzione additiva;
- Offrono la possibilità di stampare e assemblare parti composte da diversi materiali con diverse proprietà fisiche e meccaniche in un singolo processo di costruzione. Le tecnologie di stampa 3D avanzate creano modelli che emulano molto da vicino l'aspetto e le funzionalità dei prototipi;

L'intera struttura è stata costruita in acido polilattico (PLA) che è il polimero dell'acido lattico.

Il PLA, a differenza di altri materiali come l'ABS, risulta essere estrudibile a temperature minori (200°-230°) rispetto ai concorrenti filamenti e non necessita che il piatto della stampante sia riscaldato.

## 2.5 Motori

Per quanto riguarda i giunti, sono stati realizzati utilizzando dei motori servo.

Un servomotore si presenta come un piccolo contenitore di materiale plastico da cui fuoriesce un perno in grado di ruotare di un angolo compreso fra 0° e 180°, la cui posizione rimane stabile.

Per ottenere la rotazione del perno è utilizzato un motore in DC e un meccanismo di demoltiplica che consente di aumentare la coppia in fase di rotazione.

La rotazione del motore è effettuata tramite un circuito di controllo interno in grado di rilevare l'angolo di rotazione raggiunto dal perno tramite un potenziometro resistivo e di bloccare, quindi, il motore sul punto desiderato.

Il modello utilizzato è un Hitec HS-53 alimentato con batteria a 6 V.

Figura 3.1: Modello CAD del servo motore Hitec HS-53

Caratteristiche tecniche:

| | |
|---|---|
| Alimentazione | 4.8 – 6.0 V |
| Coppia a 6 V | 1.5 Kg/cm |
| Velocità a 6 V | 0.13 sec/60 deg |
| Ingranaggi | Nylon |
| Dimensioni | 22.8 x 11.6 x 22.6 mm |
| Peso | 8 gr |

## 2.6 Parti meccaniche

Questa sezione analizza nel dettaglio le parti progettate su ambiente di sviluppo CAD, al fine di creare una struttura robusta ed esteticamente accattivante.
Le parti realizzate al fine di produrre il robot risultano essere:

- Tibia;
- Femore;
- Scocca reggi-servo (anca);
- Corpo inferiore;
- Corpo superiore.

### 2.6.1 Tibia

Progettata al fine di creare un unico corpo insieme alla scocca del servomotore, risulta essere un elemento molto robusto che può essere fissato al servo con il solo utilizzo di due viti. All'interno dell'elemento è stata ricavata una finestra che ne riduce il peso ma non ne pregiudica la solidità.

Figura 3.2: Modello CAD della tibia

### 2.6.2 Femore

Risulta essere una forma abbastanza complessa che permette il fissaggio fra l'elemento e il servo grazie a delle asole in cui vengono inseriti gli ancoraggi del motore. Facendo ciò è stato possibile creare delle giunzioni molto semplici ma allo stesso tempo solide.
Anche all'interno di questo elemento è stata ricavata una finestra che ne riduce il peso ma non ne pregiudica la solidità.

Figura 3.3: Modello CAD del femore

### 2.6.3 Scocca reggi-servo

La versatilità della stampante 3D è risultata fondamentale nella creazione di questo componente grazie alla possibilità di inserire materiale di supporto, successivamente rimovibile, per la creazione di particolari forme.
Al fine di connettere femore ed anca, è stato creato un piano con 2 protuberanze per lato, su cui poggiano i due servo fissati, rispettivamente, mediante due viti.

Figura 3.4: Modello CAD della scocca reggi-servo (anca)

### 2.6.4 Corpo superiore e inferiore

Nell'elemento superiore sono state riutilizzate le asole precedentemente descritte per il femore, che in questo caso permettono il fissaggio del gruppo gamba col corpo.
Al fine di ridurre il peso e di permettere i vari collegamenti dei motori con una potenziale elettronica presente fra corpo inferiore e superiore, è stata creata una finestra circolare centrata sulla scocca.

Figura 3.5: Modello CAD del corpo superiore

L'elemento inferiore è stato invece progettato per essere fissato al corpo superiore in modo da irrobustire la struttura finale e fornire un appoggio alla batteria di alimentazione. La forma di questo particolare oggetto non impedisce alcuna libertà di movimento alle gambe e risulta inoltre essere di facile montaggio grazie all'ausilio di sole sei viti.

Figura 3.6: Modello CAD del corpo inferiore

## 2.7 Assemblaggio

La struttura finale si presenta come un oggetto estremamente robusto, esteticamente accattivante e di facile montaggio, che vede l'utilizzo di sole viti, senza la necessità di bulloni.

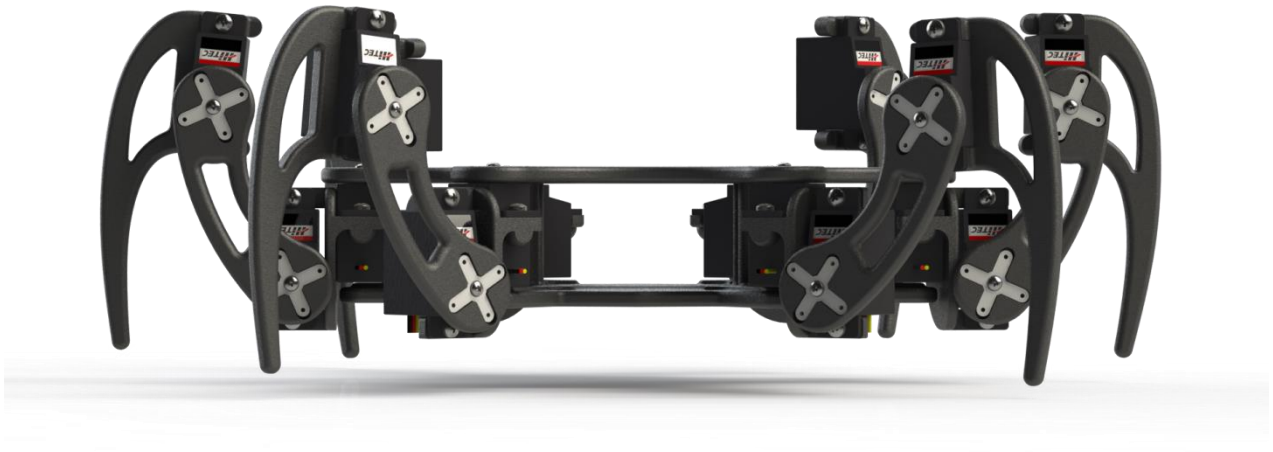


Figura 3.7: Modello CAD dell'assieme, vista frontale

Figura 3.8: Modello CAD dell'assieme, vista laterale dx

Figura 3.9: Modello CAD dell'assieme, vista superiore

Figura 3.10: Modello CAD dell'assieme, vista in prospettiva

# Capitolo 4: Elettronica di bordo

## 4.1 Microcontrollore

Il cuore della scheda elettronica progettata è il controllore Microchip modello PIC18F4550.

Il processore, che presenta una frequenza operativa massima pari a 48 MHz (12 MIPS), viene fornito di una memoria di programmazione flash pari a 32KB, una memoria RAM di 2048B e una memoria EEPROM di 256B.

Presenta 35 porte I/O, 13 convertitori A/D a 10 bit e supporta protocolli PWM, SPP, SPI, I2C ed EAUSART.

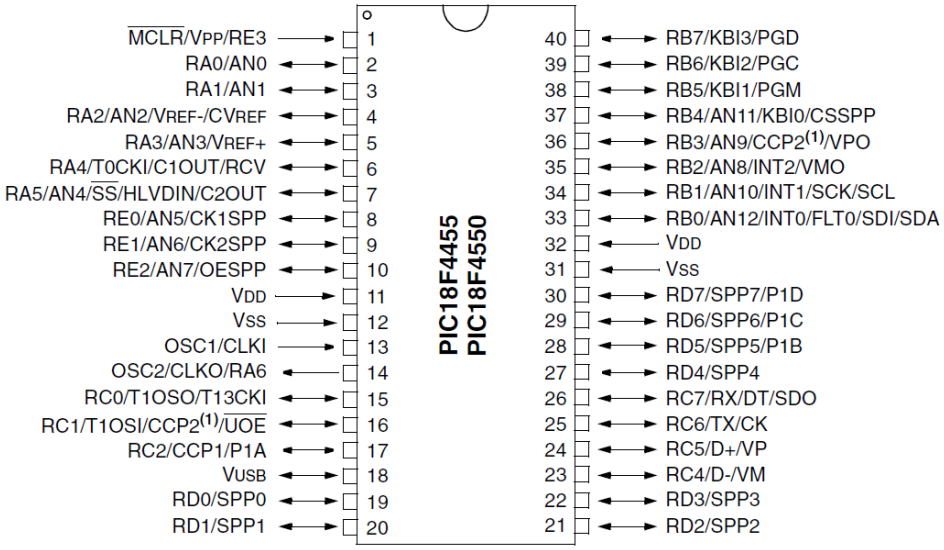



Figura 4.1: Socket PIC18F4550 40 pin, foro passante

La famiglia PIC18 fornisce alte performance ad un prezzo economico e garantisce la durata nel tempo sia del dispositivo che del programma memorizzato al suo interno. PIC18F4550 è dotato anche di un'ampia gamma di features che riducono significativamente il consumo di potenza durante lo svolgimento delle operazioni. Inoltre utilizza lo standard di comunicazione USB 2.0 sia in modalità low-speed che in modalità full-speed e dispone di diverse opzioni di settaggio dell'oscillatore, fornendo all'utente un ampio range di scelte durante lo sviluppo delle proprie applicazioni.

## 4.2 Boot Loader

Il boot loader è un programma che, in fase di avvio, carica il kernel dalla memoria secondaria alla memoria primaria, permettendone l'esecuzione e il successivo avvio del sistema.
In definitiva, non è nient'altro che una porzione di programma che permette di caricare ed eseguire il codice applicazione senza fare uso di un programmatore dedicato.
Il microcontrollore si interfaccia col computer mediante un canale di comunicazione dedicato, in questo caso USB, ed è possibile programmarlo mediante il file .hex prodotto nell'ambiente di sviluppo.

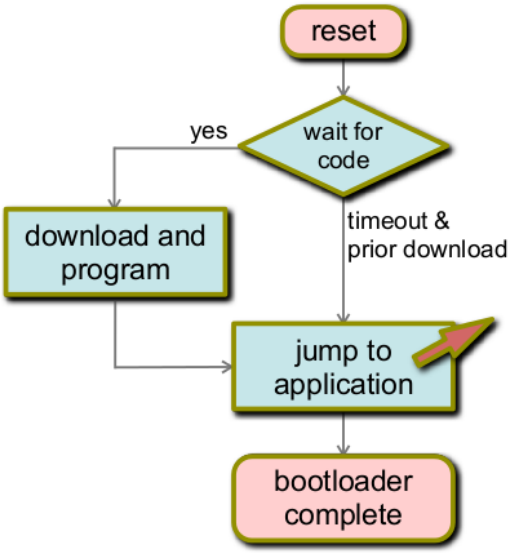
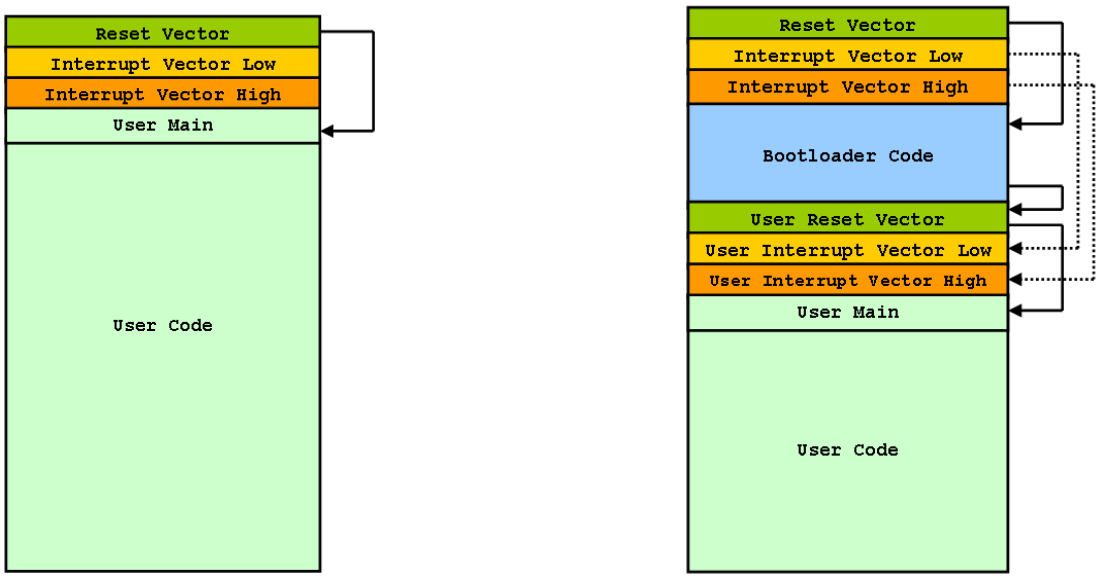



Figura 4.2: Algoritmo di funzionamento del boot loader

Non appena avviene il reset del dispositivo, il boot loader si attiva e per un certo periodo di tempo va a monitorare una linea di I/O. A seconda dello stato logico del pin, il dispositivo entra in modalità programmazione, modalità nella quale è possibile caricare un nuovo programma sul controllore, oppure in modalità applicazione, modalità nella quale è possibile eseguire tale programma.

Il boot loader  e il codice applicazione risiedono nella stessa area di memoria, per cui la memoria deve essere allocata in maniera tale che i due programmi non si sovrappongano.




(a) (b)

Figura 4.3: (a) Memoria mappata senza boot loader;
(b) Memoria mappata con boot  loader

Nel caso di boot loader, la memoria deve essere rimappata e solitamente si riserva al programma loader un'area di memoria immediatamente successiva ai vettori di interruzione.
In questo modo, al reset del dispositivo, il boot loader verrà avviato al posto del codice utente, che invece si trova in un'area di memoria a indirizzo maggiore.
Anche i vettore di interrupt devono essere rimappati in modo da puntare a locazioni di memoria successive al boot loader.

Il boot loader utilizzato viene definito attraverso l'acronimo HID (Human Interface Device). Questo significa che il programma loader in questione è stato scritto rispettando le specifiche USB-HID, che fanno sì che esso possa essere visto dal sistema operativo Windows come una generica periferica USB, non richiedendo dunque ulteriori driver se non quelli già presenti nel sistema operativo.

## 4.3 Ripartizione dei processi di elaborazione

Per suddividere i processi di elaborazione, sono stati inseriti sulla scheda elettronica due controllori.
Il primo (Master) ha il compito di elaborare i dati relativi alla cinematica e allo studio del moto, risolvere quindi le equazioni proposte nella sezione “Struttura semplificata” ed elaborare gli angoli che i giunti devono raggiungere.
Il secondo (Slave) risulta essere invece un Motor Driver: riceve dal processore primario le informazioni riguardanti gli angoli dei giunti e fornisce ai motori i riferimenti desiderati mantenendoli fino a nuovo comando.
In due controllori comunicano attraverso una linea seriale (EUSART) condivisa.

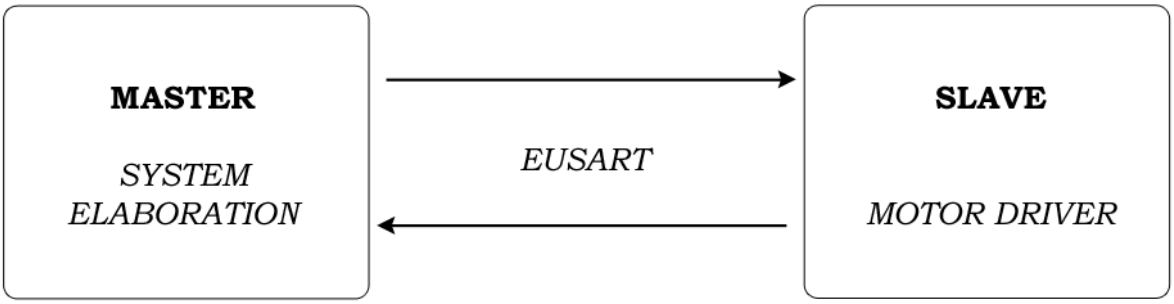



Figura 4.4: Comunicazione delle informazioni master-slave

Una volta che i dati sono stati elaborati dal Master, essi vengono suddivisi in pacchetti da 8 bit. Ogni volta che il Master invia un pacchetto, aspetta l’ACK dallo Slave prima dell’invio del pacchetto successivo. Lo Slave, quando riceve un dato, controlla in prima istanza che su esso non siano presenti errori e successivamente invia l’ACK al Master.
Considerato che per ogni coppia motore-angolo sono necessari 4 pacchetti (2 per motore e 2 per angolo), risulta chiaro che per ogni posizionamento del robot esapode sono necessari 72 pacchetti, per un totale di 144 interazioni Master-Slave.
Questa soluzione permette di ridurre il numero di interrupts su ogni dispositivo in caso di mantenimento prolungato di una certa posizione, a discapito di un maggior tempo necessario per far cambiare posizione all’esapode, ottimizzando quindi l’elaborazione e riducendo il tempo di esecuzione di ogni singolo programma.

# 4.4 Scheda elettronica

Di seguito sono riportati gli schemi che hanno portato alla realizzazione della scheda elettronica.

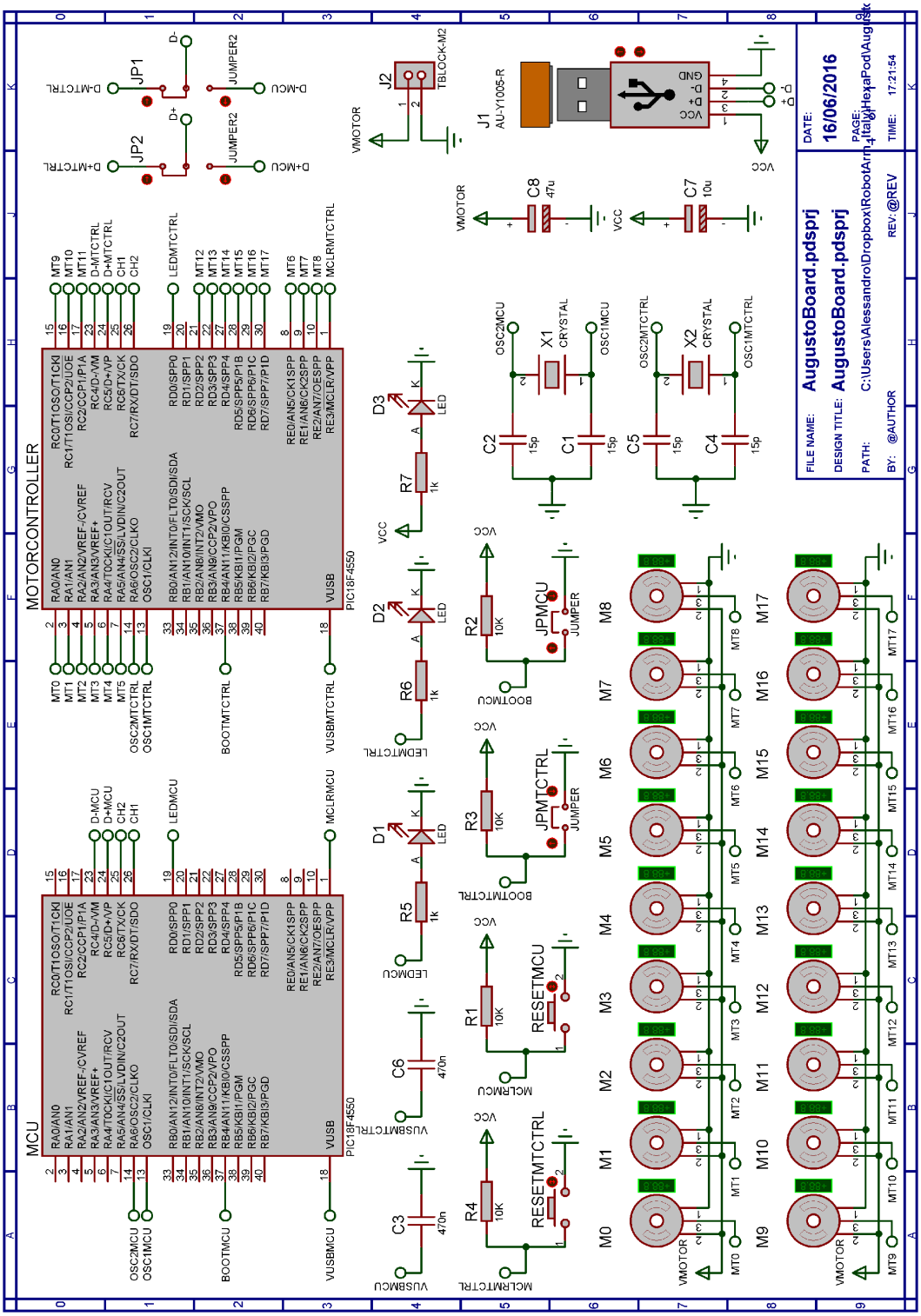



Figura 4.5: Schema circuitale della scheda elettronica progettata

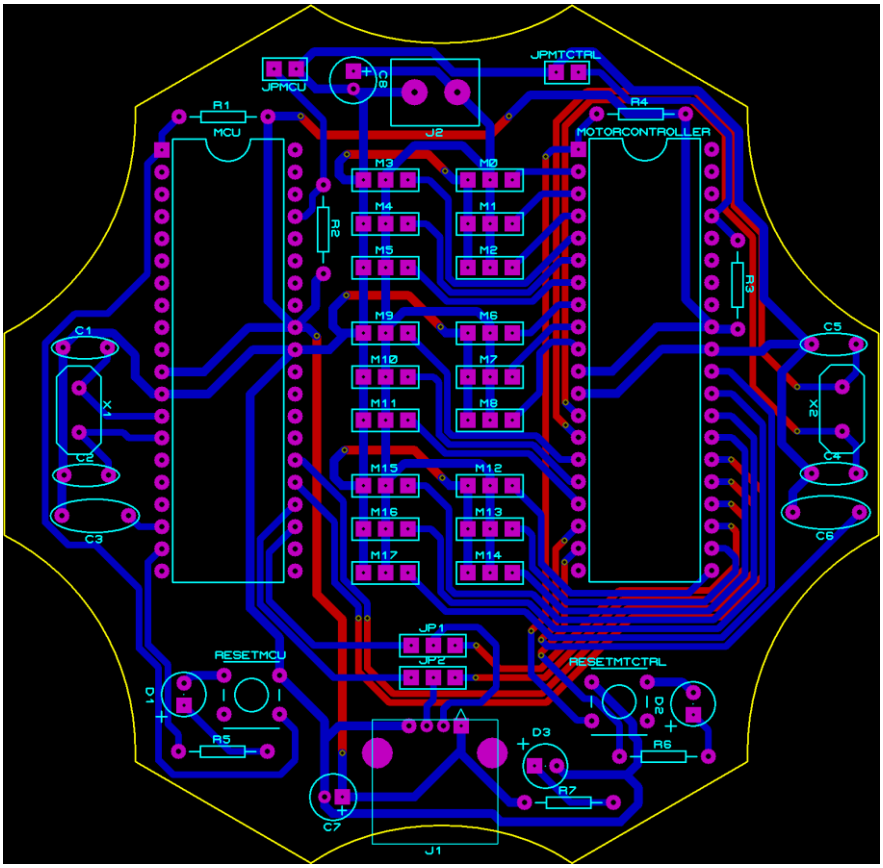



Figura 4.6: Sbroglio della scheda elettronica progettata

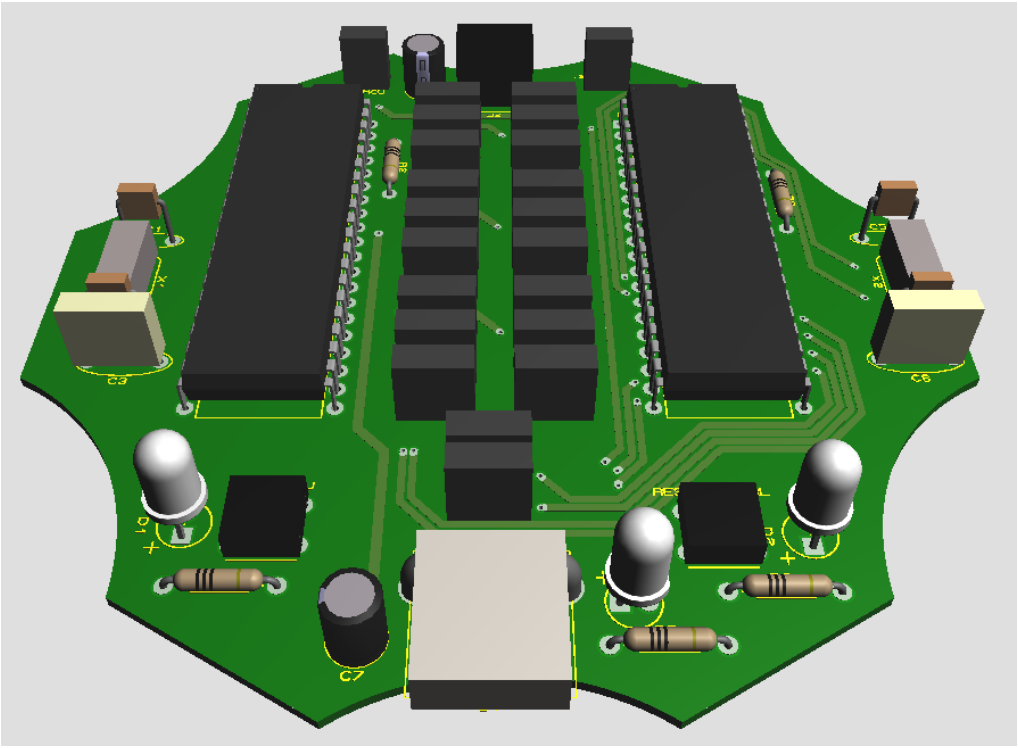


Figura 4.7: Visualizzazione tridimensionale della scheda elettronica progettata

# Capitolo 5: Algoritmo di gestione dei motori

## 5.1 Codifica dei segnali di controllo

Al fine di posizionare i servo motori nella posizione desiderata, viene utilizzato uno dei due microcontrollori presenti sulla scheda elettronica progettata.
Il controllore, che prende il nome di "Motor Driver", riceve una serie di dati dal sistema di elaborazione, li interpreta e ne ricava informazioni utili per il posizionamento dei giunti.
I dati si presentano sotto forma di pacchetti di dimensione fissa di 8 bit.
Una volta ricomposto l'intero messaggio, il microcontrollore è in possesso di tutte le informazioni necessarie per posizionare i servo: ID motore e angolo di posizionamento.
Subentra, in questo momento, una routine che consente di tradurre l'angolo da raggiungere in un'onda quadra con un certo duty cycle che dovrà poi essere utilizzata per pilotare elettronicamente il motore.
Infine vengono ordinate le onde quadre per duty cycle crescente e si attiva una nuova routine che consente il posizionamento dei servo e il loro mantenimento nella posizione indicata.
Il mantenimento nella posizione voluta viene raggiunto semplicemente continuando ad inviare al servo lo stessa onda quadra con lo stesso duty cycle.
Una volta che il duty cycle dell'onda quadra varia, anche l'angolo raggiunto dal servo varierà.

# 5.2 Timer

Per la realizzazione dell'onda quadra, si utilizza un dispositivo presente all'interno del controllore chiamato Timer.

Il modulo Timer consente l'esecuzione di processi a specifici intervalli di tempo opportunamente programmati. Si tratta di dispositivi di conteggio che funzionano indipendentemente dalla CPU, la quale è libera di effettuare altre operazione mentre il timer conta.

Il Microchip PIC18F4550 presenta 4 moduli timer al suo interno: Timer0, Timer1, Timer2 e Timer3.

Il modulo utilizzato per la realizzazione dell'onda quadra è stato il Timer0.

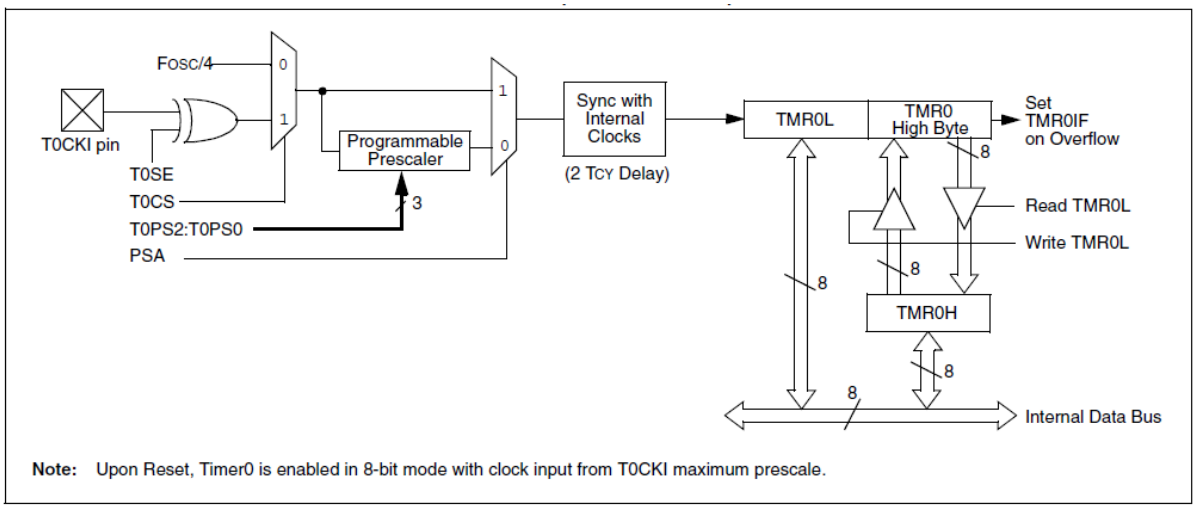



Figura 5.1: Schema a blocchi del Timer0 in modalità 16bit

Timer0 può operare sia come timer che come contatore. In modalità timer, il modulo incrementa il proprio valore ad ogni colpo di clock.

Le caratteristiche principali di questo timer sono:

- Possibilità di lavorare sia con registri a 8 bit che con registri a 16 bit;
- Possibilità di usare un Prescaler per ridurre la frequenza di lavoro;
- Possibilità di selezionare la sorgente di clock;
- Possibilità di generare un interrupt nel caso di overflow al termine del conteggio.

Il registro del timer verrà incrementato fino a raggiungere il suo valore massimo, che a 16 bit risulta essere 65535, dopodiché andrà in overflow e si azzererà generando un interrupt.
Prima di proseguire oltre si fa cenno all'espressione utilizzata per tradurre l'angolo da raggiungere nel corrispondente impulso quadrato, in particolar modo, nel livello alto di tale impulso:

$$l_{HIGH} = {}^{degree * (l_{HIGH}^{MAX} - l_{HIGH}^{min})}/_{180} + l_{HIGH}^{min}$$

La frequenza di lavoro del timer, risulta essere:

$$f_{TIMER} = {}^{f_{OSC}}/_{4 * Pre}$$

Dove $f_{OSC}$ è la frequenza di utilizzo del PIC, in questo caso 48 MHz, e $Pre$ invece è il Prescaler utilizzato, in questo caso 8.
Nel conteggio, il tempo che intercorre tra un incremento e l'altro è dato da:

$$t_{TIMER} = {}^{1}/_{f_{TIMER}}$$

In conclusione, risulta che il tempo necessario per raggiungere l'overflow è:

$$t_{OVR} = (65535 - X) * t_{TIMER}$$

Da cui si ricava che $X$, valore iniziale del conteggio, è:

$$X = 65536 - {}^{t_{OVR}}/_{t_{TIMER}}$$

$X$ è la variabile che è necessario modificare per ottenere ritardi di una specifica durata.

Per la realizzazione di un'onda quadra di periodo 20ms con, con DC=25%, sarà sufficiente generare due ritardi:

- Il primo, di 5ms, in cui si pone il pin di uscita a livello logico H. Da qui risulta che $t_{OVR}^{H} = 5\text{ms}$;
- Il secondo, di 15ms, in cui si pone il pin di uscita a livello logico L. Da qui risulta che $t_{OVR}^{L} = 15\text{ms}$;

e ripetere tale sequenza di operazione infinite volte nel tempo.

## 5.3 Pilotaggio dei motori

Per il mantenimento dei servo motori nella posizione desiderata, gli impulsi quadrati vengono ordinati per duty cycle crescente.
Questo consente di generare una serie di interrupts di durata temporale sempre positiva dati dalla differenza dei livelli logici alti dei singoli impulsi.
Nella seguente trattazione, l'ID utilizzato per descrivere un motore non è l'ID di riferimento fisico, ma semplicemente l'indice che tale componente assume nel vettore motori ordinati.
In particolar modo accade che:

- Si pone alto il pin relativo al motore 0 e si ritarda l'esecuzione di una certa fase;
- Si pone alto il pin relativo al motore 1 e si ritarda l'esecuzione di una certa fase;
- …e così via fino al motore 17;
- Si pone alto il pin relativo al motore 17 e si calcola la differenza fra il tempo fin ora ritardato e il livello logico alto relativo al motore 0. Una volta calcolato, il timer viene programmato con tale ritardo;
- Raggiunto l'overflow, viene posto basso il pin relativo al motore 0, si calcola la differenza di livello logico alto fra motore 1 e motore 0. Una volta calcolato, il timer viene programmato con tale ritardo;
- Raggiunto l'overflow, viene posto basso il pin relativo al motore 1, si calcola la differenza di livello logico alto fra motore 2 e motore 1. Una volta calcolato, il timer viene programmato con tale ritardo;
- …e così via fino al motore 17;
- Raggiunto l'overflow, viene posto basso il pin relativo al motore 17 e si ritarda l'esecuzione del livello logico basso del motore 17;
- Raggiunto l'overflow, la procedura ricomincia;

Questa procedura permette quindi di mantenere in posizione i motori del robot esapode, ma può essere ritenuta valida per mantenere in posizione motori adibiti a molteplici funzionamenti.

Tramite la sequenza di operazioni indicata risulta facile capire che il numero di motori controllabili dipende dalla fase inserita fra un motore e l'altro, che nel caso sopra citato risulta essere 10µs.
Per esempio, con una fase di 10 µs ed un livello logico alto minimo pari a 600µs, risulta che il numero massimo di motori controllabili è 60.

In generale risulta:

$$N_{motor} = {}^{l_{HIGH}^{min}}/_{\varphi}$$

Dove $\varphi$ è la fase utilizzata.

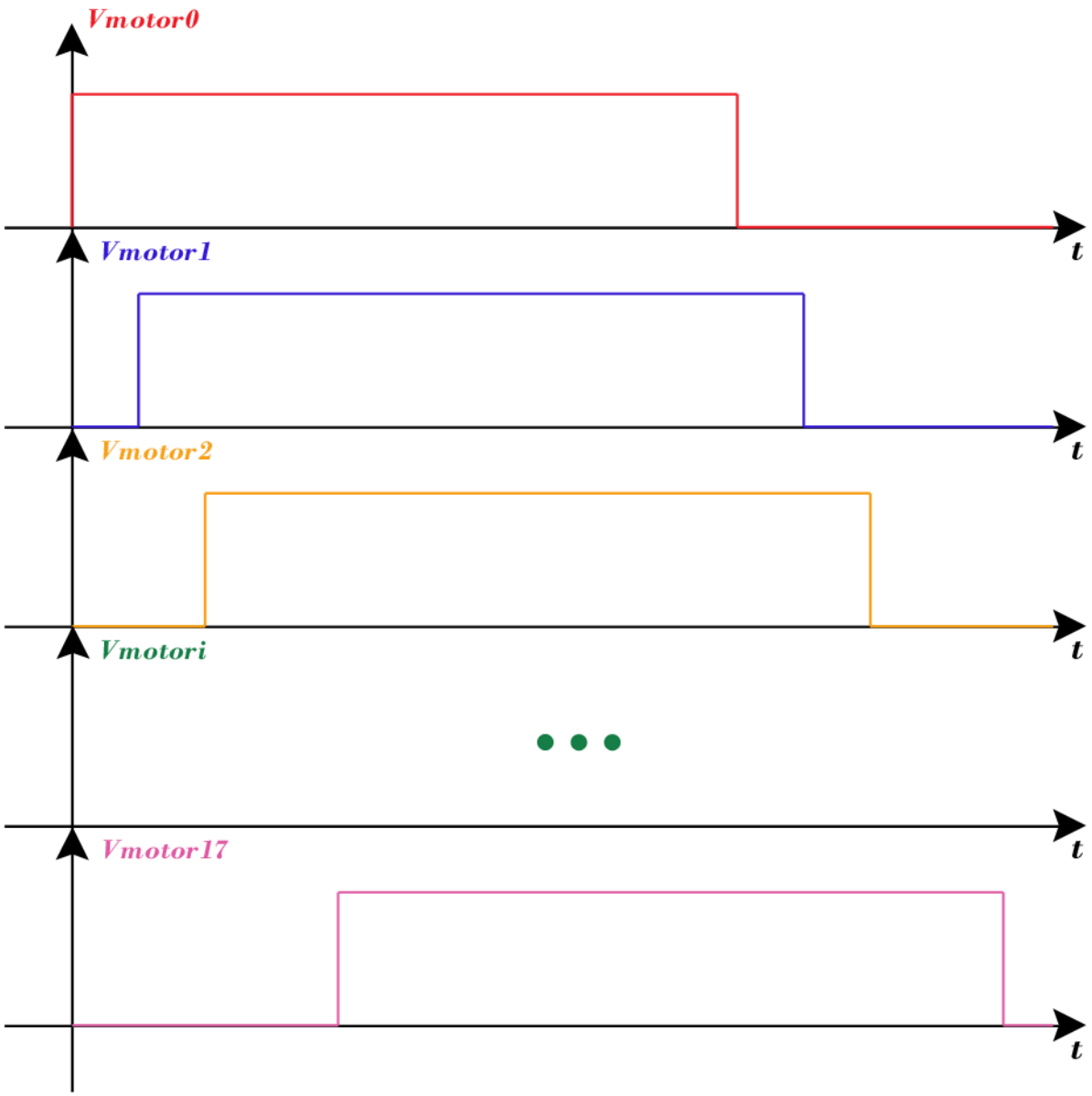



Figura 5.2: Schema di generazione delle onde quadre

# Bibliografia

[1] Michael Fielding, “Omnidirectional Gait Generating Algorithm for Hexapod Robot”, University of Canterbury, June 2002.

[2] Mohammad Mahdi Agheli Hajiabadi, “Analytic Workspace, Kinematics, and Foot Force Based Stability of Hexapod Walking Robots”, Worcester Polytechnic Institute, May 2003.

[3] Microchip, “PIC18F2455/2550/4455/4550 Data Sheet”, 2006.

[4] Microchip, “MPLAB X IDE User’s Guide”, 2015.

[5] Microchip, “MPLAB XC8 C Compiler User’s Guide”, 2015.