



LABORATORIO SISTEMI IOT

Realizzato da:

Michał Horowski
Danilo Palladino
Vittorio Dedi

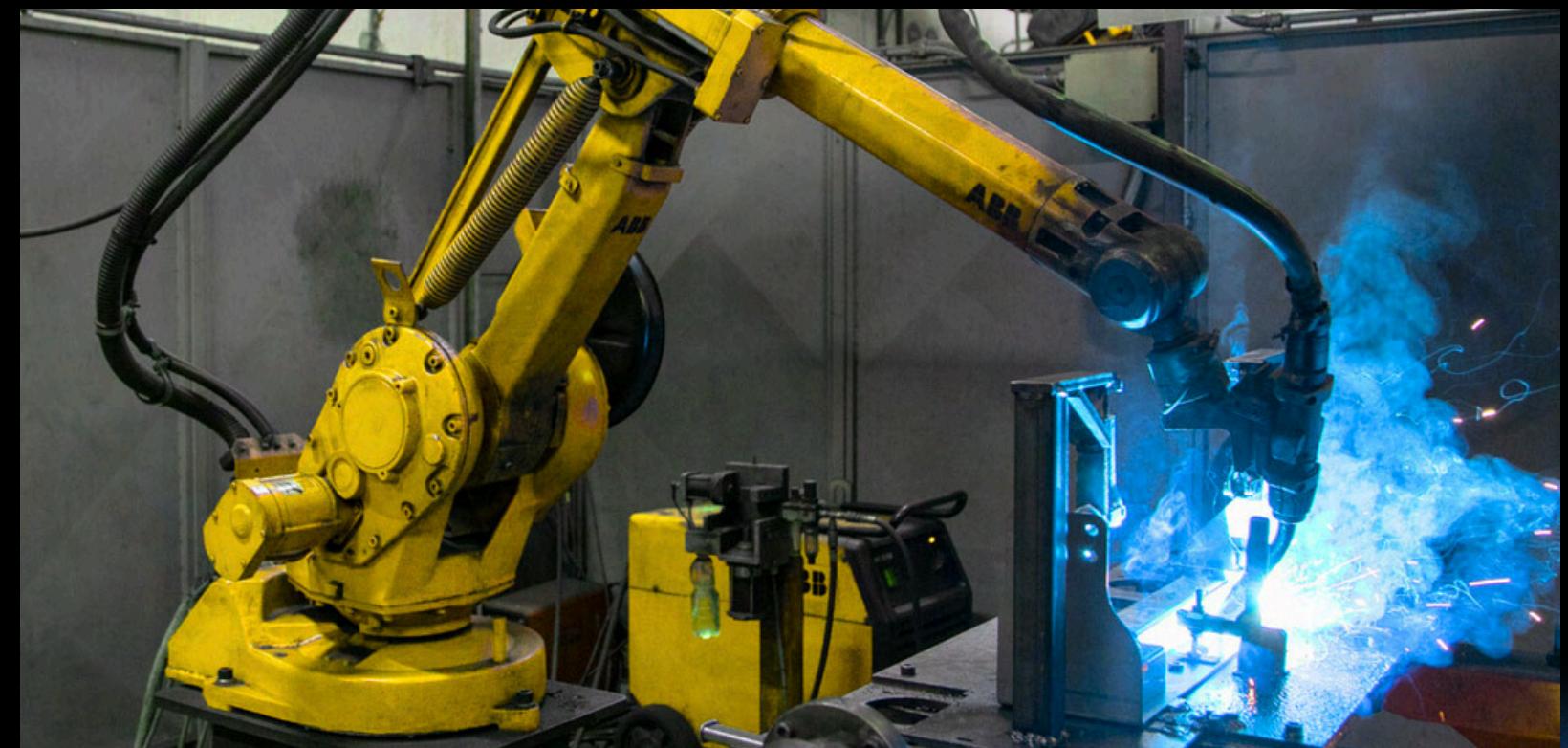
Introduzione e Progettazione

Idea e Obiettivo

Il progetto P.A.N. è nato con l'idea di sviluppare un sistema IoT distribuito che simuli il funzionamento di un braccio robotico saldatore industriale.

Il sistema si basa sulla comunicazione wireless tra due schede BBC micro:bit utilizzando componenti del kit Keyestudio.

Il progetto che presentiamo oggi rappresenta un proof-of-concept in scala ridotta di un impianto automatizzato per l'ambiente industriale con controllo e monitoraggio remoto.



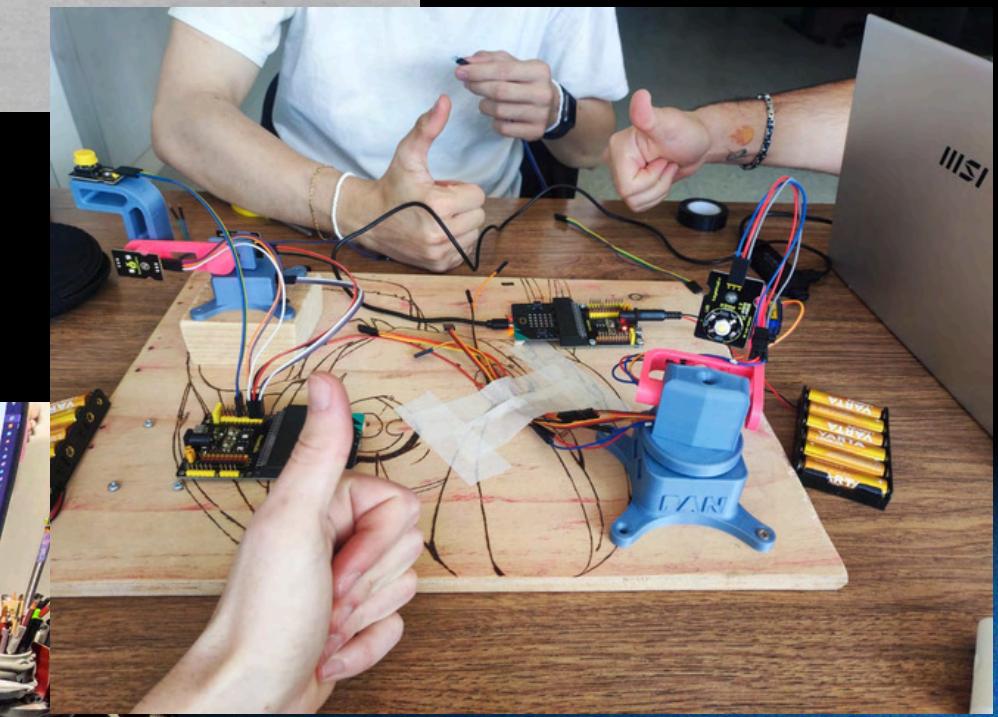
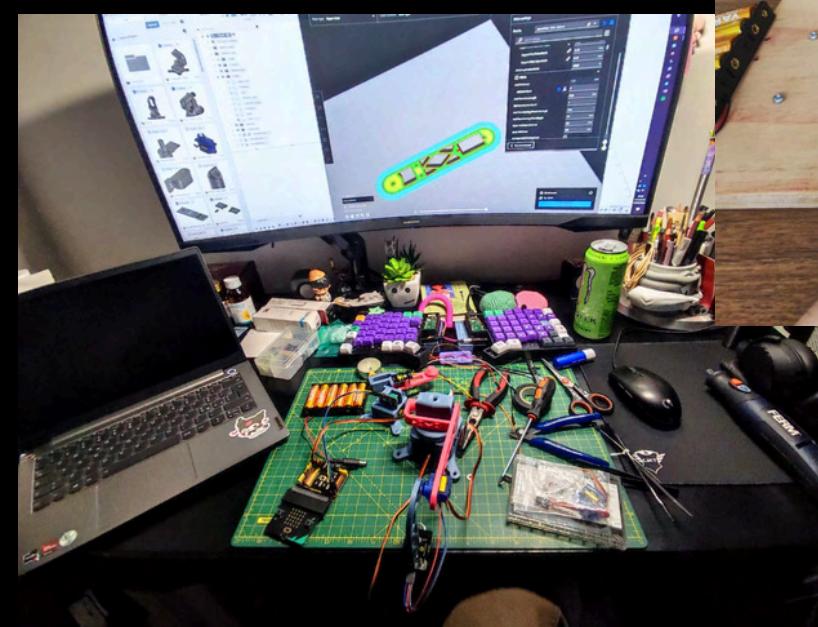
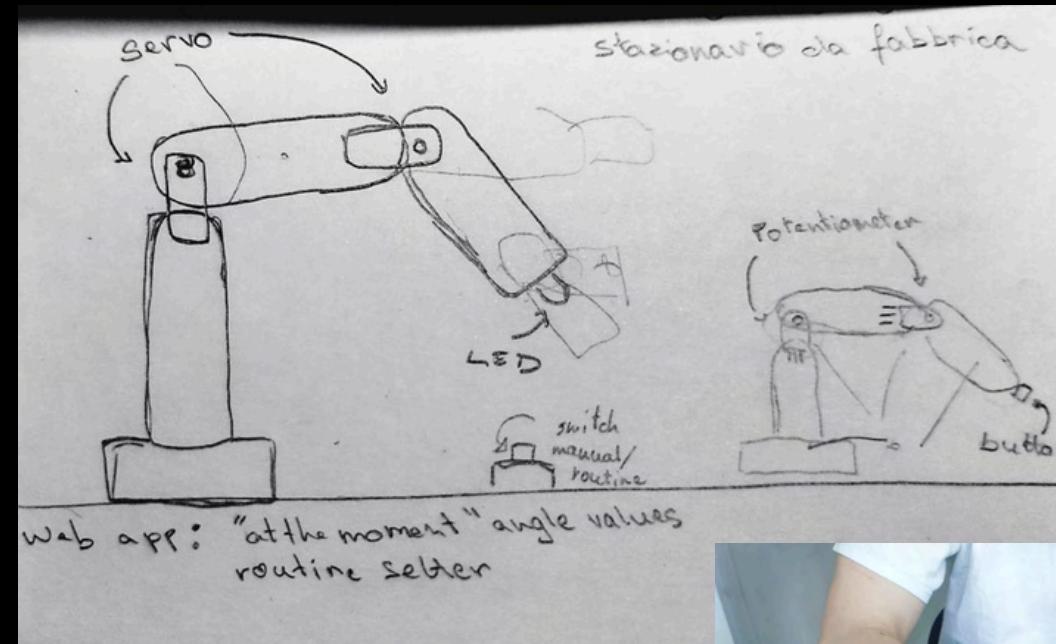
Idea e Obiettivo

La prima scheda, configurata come fosse un joystick ha tre potenziometri e permette all'utente di impartire comandi di movimento.

La seconda scheda riceve questi comandi e li traduce in movimenti fisici attraverso tre servomotori che controllano le articolazioni del braccio robotico.

Tutti i dati operativi vengono registrati in tempo reale e visualizzati attraverso una dashboard web interattiva che permette il monitoraggio completo del sistema.

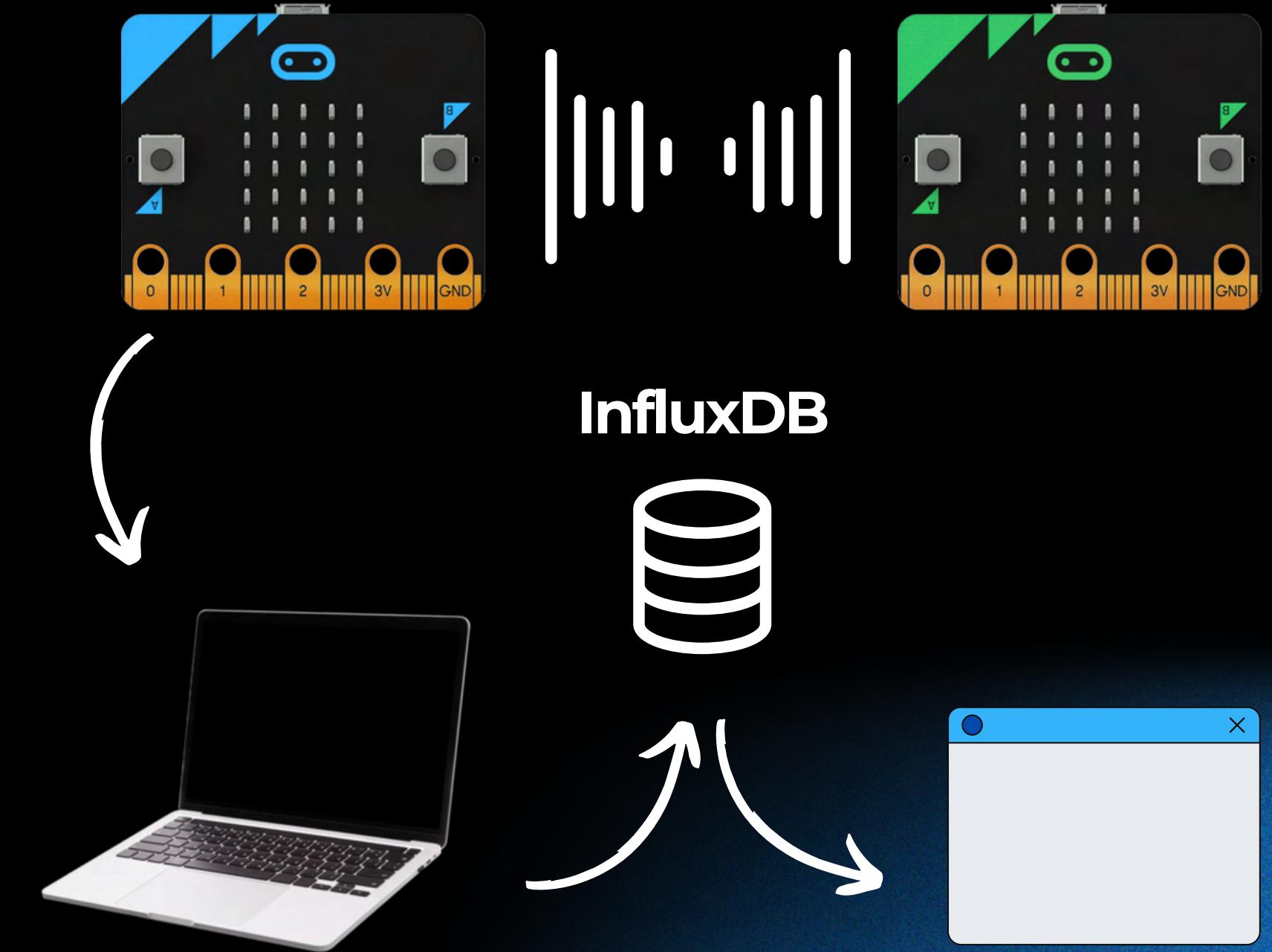
Il sistema ha avuto diverse iterazioni di prototipi diversi in design e codice



Struttura del sistema

Il sistema è basato su due nodi micro:bit comunicanti via radio. La scheda trasmittente integra tre potenziometri analogici collegati ai pin P0, P1 e P2. Poi abbiamo collegato un pulsante digitale sul pin P5, che funge da comando per l'attivazione della simulazione di saldatura (accensione del led).

La scheda ricevente controlla tre servomotori standard collegati ai pin P8, P12 e P16. Infine abbiamo un LED collegato al pin P14. Entrambe le schede utilizzano shield Keyestudio per facilitare i collegamenti e pacchi batteria esterni per garantire alimentazione stabile e indipendente.

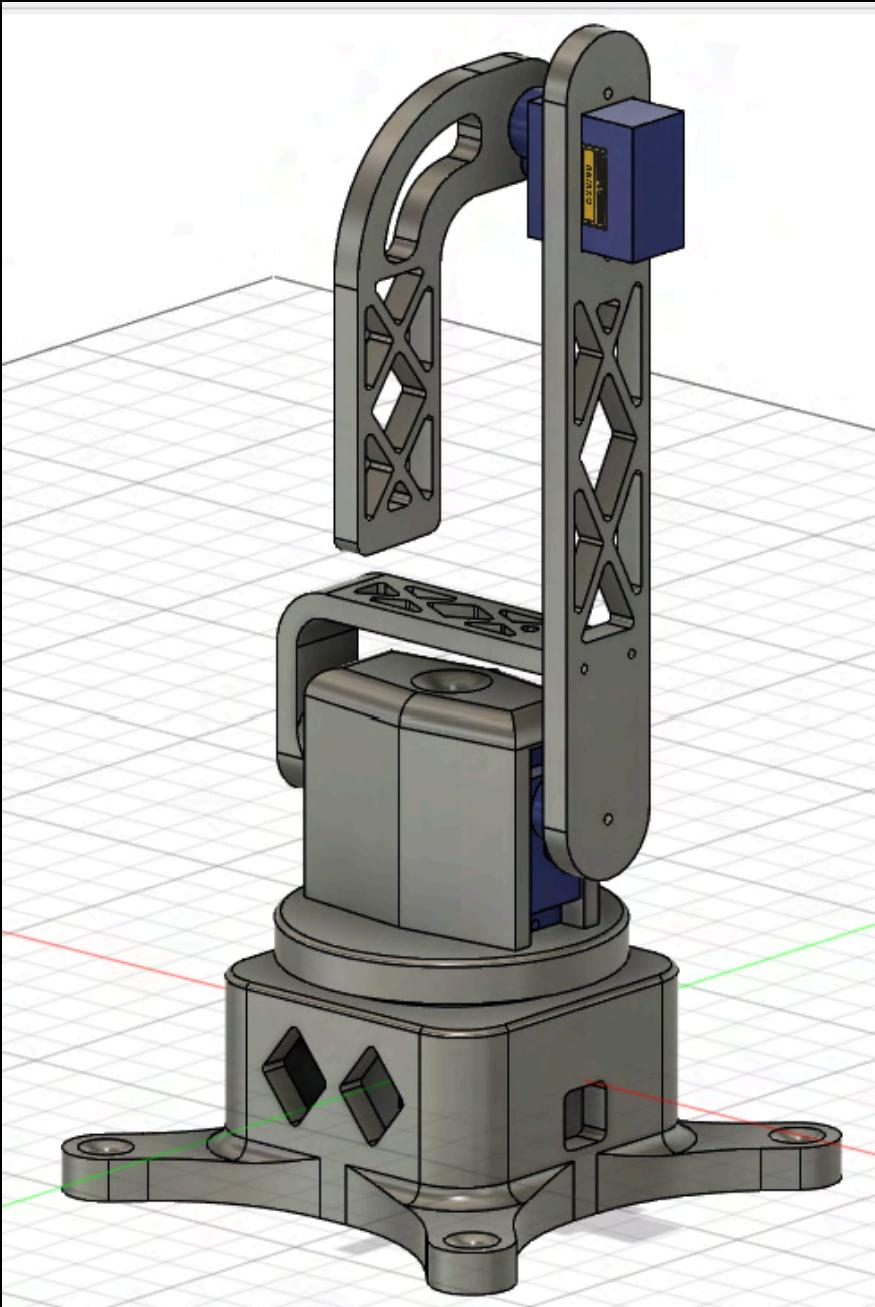


Progettazione del Sistema

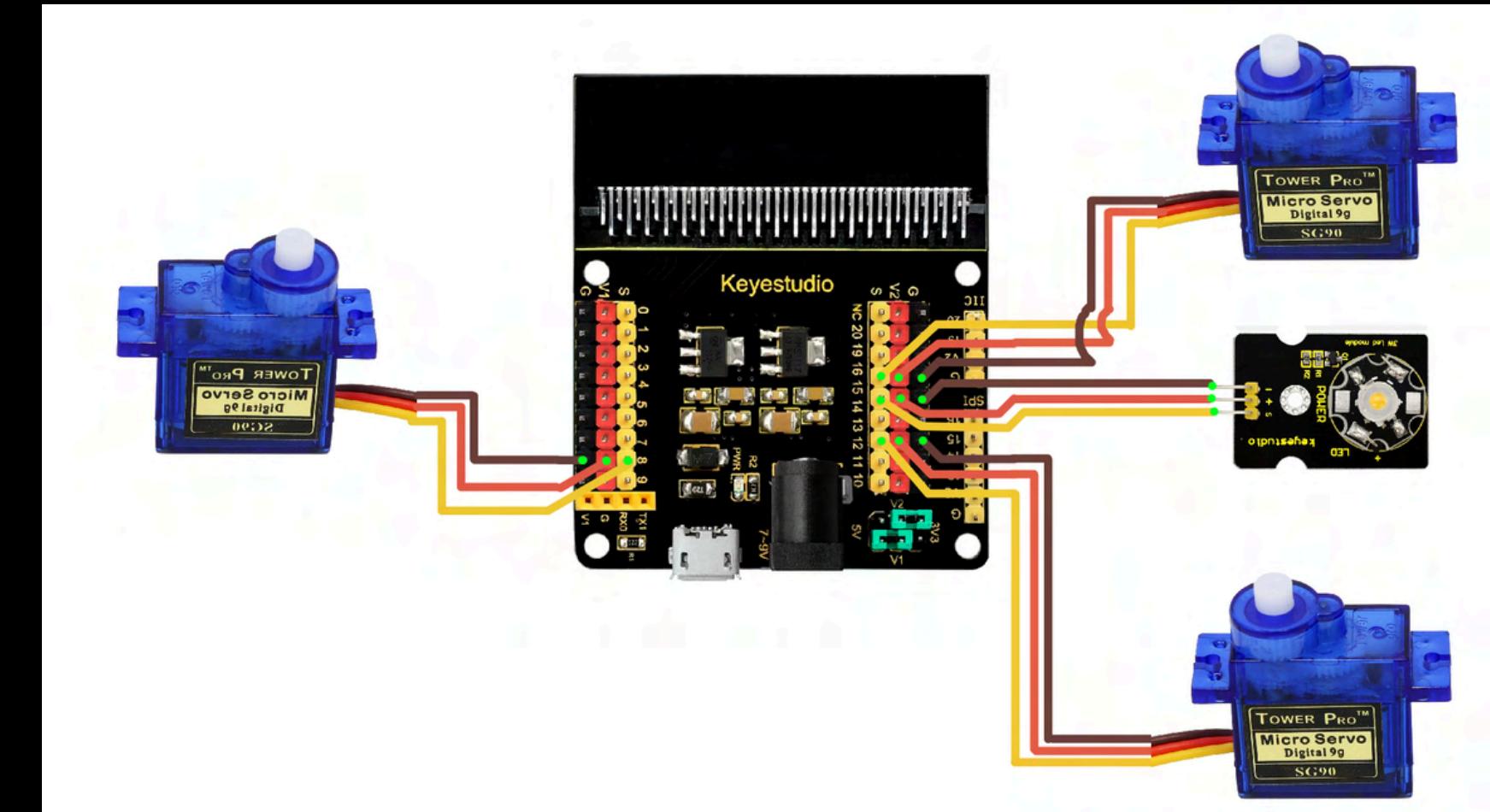
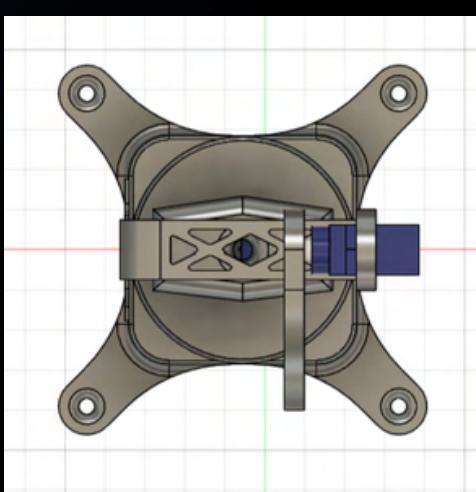
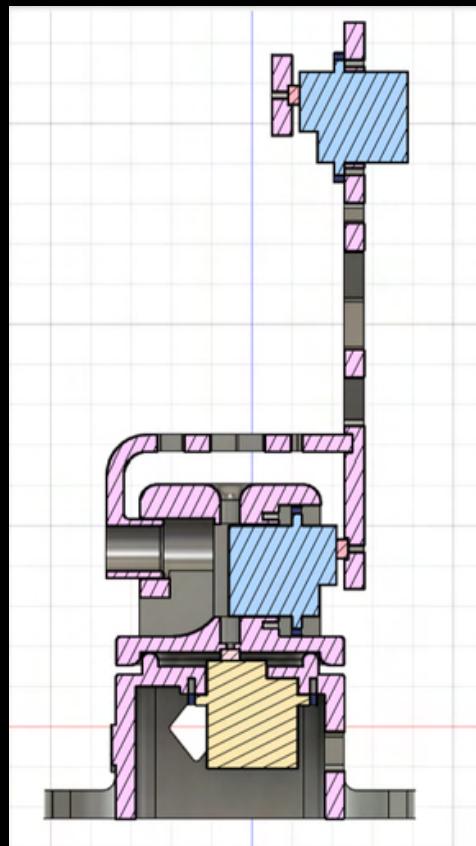
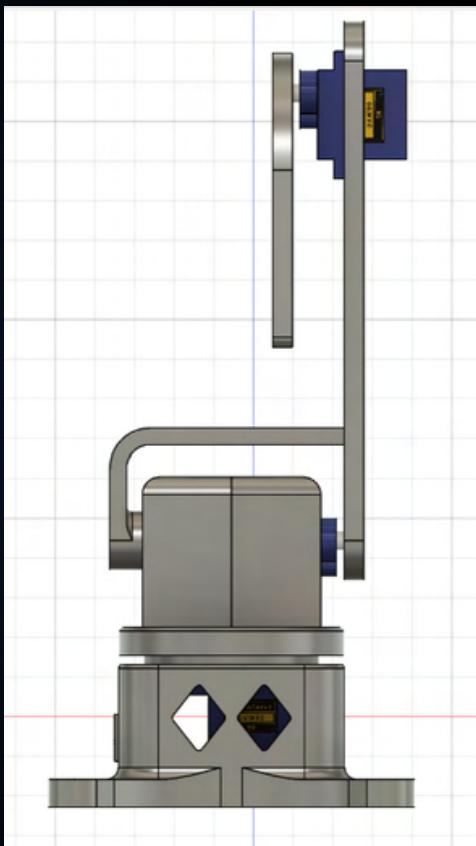
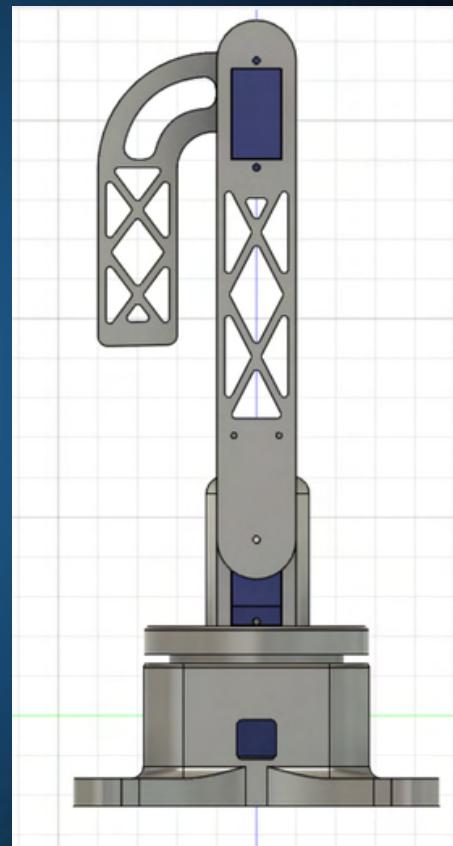
La struttura fisica del braccio robotico è stata progettata utilizzando Fusion 360, questa si compone di tre sezioni principali: la base rotante che ospita il primo servomotore, lo snodo centrale che permette l'elevazione del braccio, e il terminale superiore "ad angolo retto" che controlla l'orientamento finale della parte terminale.

La base è composta da due parti che permettono una rotazione scorrevole e lo snodo centrale è supportato da una staffa per stabilizzare la leva del braccio

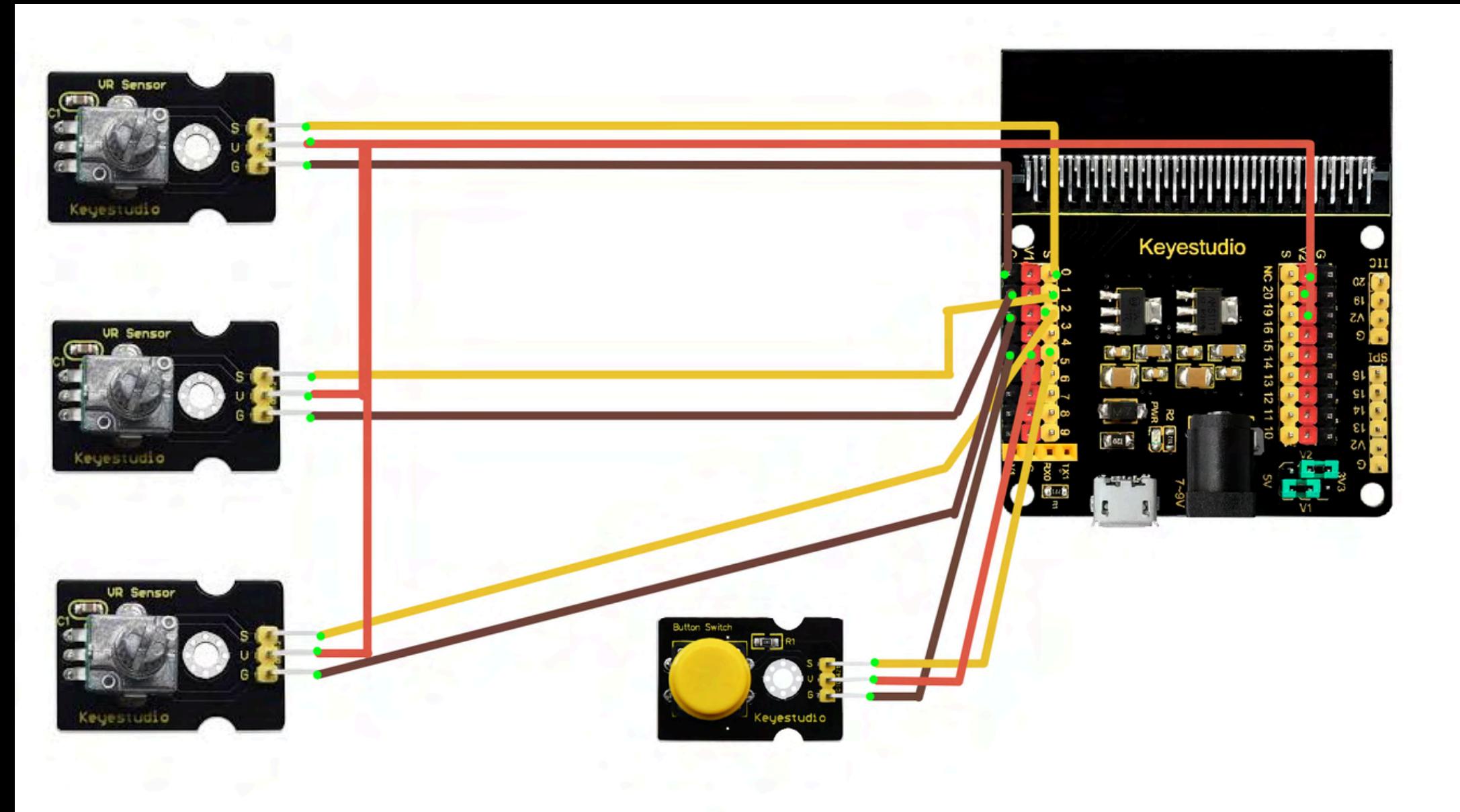
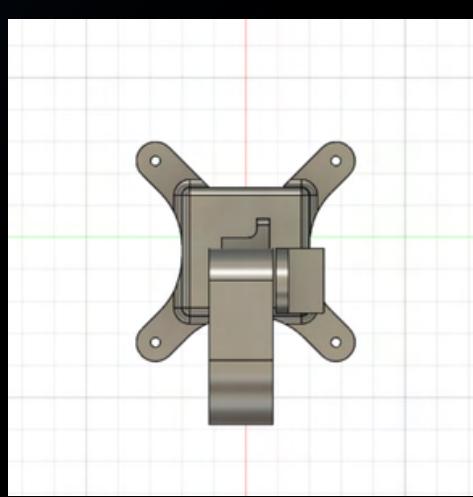
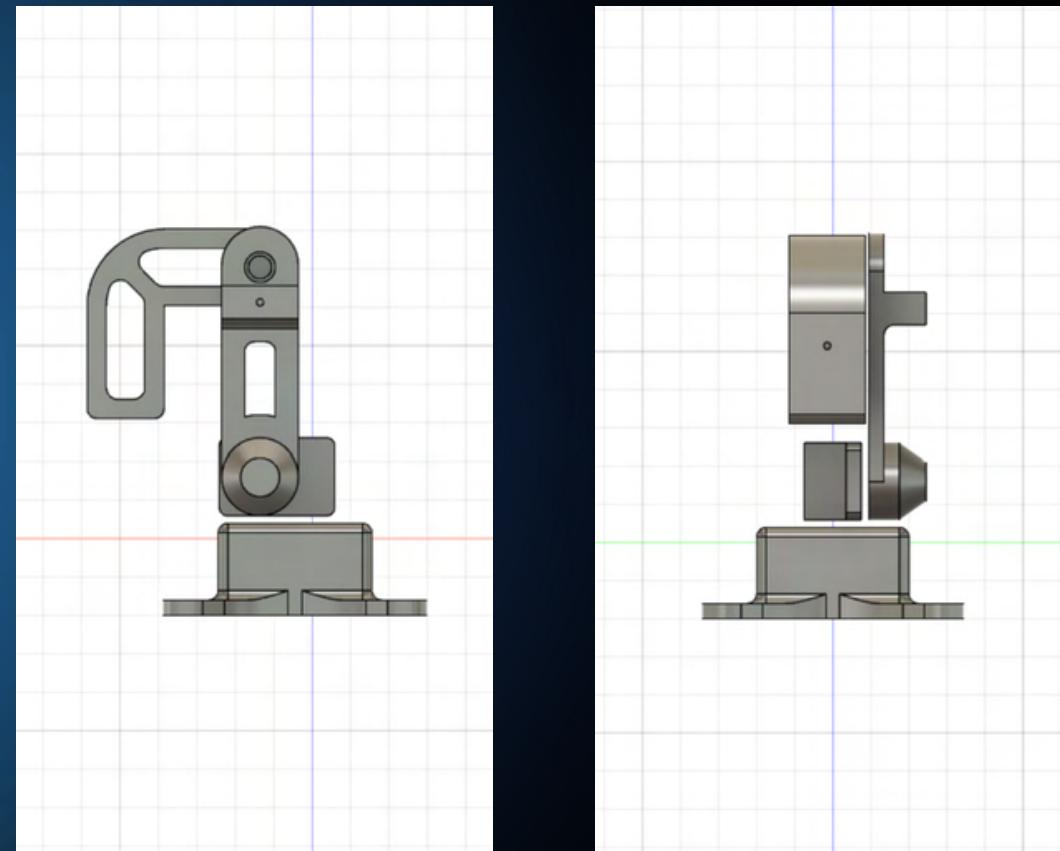
Il design permette l'accesso diretto ai componenti elettronici semplificando le operazioni di manutenzione e la canalina rende possibile un cablaggio pulito e funzionale. Le parti sono state stampate in PLA con colori azzurro e rosa fluorescente.



Braccio P.A.N.



Controller P.A.N.



Comunicazione Radio

La comunicazione tra le schede utilizza il modulo radio integrato del micro:bit, configurato sul canale 42 alla massima potenza disponibile per garantire affidabilità.

Il protocollo sviluppato utilizza messaggi testuali strutturati nel formato "P1:valore1,P2:valore2,P3:valore3,BTN:stato" per trasmettere simultaneamente tutti i dati di controllo.

Per ottimizzare l'utilizzo della banda radio ed evitare congestione, la trasmissione avviene solo quando vengono rilevate variazioni significative nei valori dei potenziometri, con un timeout massimo di 100 millisecondi per garantire aggiornamenti regolari anche in condizioni statiche.

Programmazione

Modulo Trasmittente

Il software della scheda trasmittente, come quello della ricevente è implementato in MicroPython e segue una logica ciclica caratteristica della scheda con cui stiamo lavorando. All'inizio viene inizializzata e configurata la configurazione radio. Il loop principale esegue letture continue dei tre potenziometri analogici, ottenendo valori nel range 0-1023. Questi vengono trasmessi nel pacchetto dati.

Modulo Trasmittente (Flow Chart)

INIZIALIZZA sistema trasmittente:

CONFIGURA radio(canale=42, potenza=7)

CONFIGURA potenziometri su PIN0, PIN1, PIN2

CONFIGURA pulsante su PIN5

IMPOSTA pull-down per pulsante

LOOP principale trasmittente:

LEGGI valori potenziometri (0-1023)

LEGGI stato pulsante (0/1)

SE cambiamenti rilevati O timeout intervallo:

CREA messaggio "P1:val1,P2:val2,P3:val3,BTN:stato"

TRASMETTI via radio

AGGIORNA display con potenziometro più alto

LOG valori trasmessi

ATTENDI 100ms

Modulo Ricevente

Nel codice della scheda ricevente viene implementato un sistema di ascolto continuo del canale radio dedicato. Il software include funzioni di conversione per tradurre i valori dei potenziometri (0-1023) in angoli servo appropriati (0-180 gradi).

La funzione **pot_to_angle()** implementa una logica di mappatura inversa che limita il range massimo a 512 per evitare sovraccarichi meccanici sui servomotori, applicando la formula $180 - (\text{valore}/512)*180$ per ottenere un comportamento intuitivo del controllo.

Modulo Ricevente

La conversione successiva in segnali PWM (Pulse Width Modulation) utilizza la funzione `angle_to_pwm()` che mappa linearmente gli angoli nel range PWM 26-128 appropriato per i servomotori standard.

Il sistema include piccoli meccanismi di sicurezza: un timeout di connessione infatti monitora continuamente la ricezione di dati, e in caso di perdita del segnale radio per oltre 5 secondi, tutti i servomotori vengono automaticamente posizionati in una configurazione sicura a 180 gradi e il LED viene spento.

Tutti i dati ricevuti e le azioni intraprese vengono trasmessi via porta seriale per consentire il monitoring esterno.

Modulo Ricevente (Flow Chart)

INIZIALIZZA sistema ricevente:

CONFIGURA radio(canale=42, potenza=7)

CONFIGURA servo su PIN8, PIN12, PIN16 (periodo PWM 20ms)

CONFIGURA LED su PIN14

POSIZIONA tutti servo a 180° (stato iniziale)

FUNZIONI conversione:

pot_to_angle(valore_pot):

LIMITA valore_pot a max 512

RITORNA $180 - (\text{valore_pot}/512)*180$ // Inversione angolo

angle_to_pwm(angolo):

RITORNA PWM tra 26-128 per angoli 0-180°

Modulo Ricevente (Flow Chart)

LOOP principale ricevente:

SE messaggio radio ricevuto:

PARSA "P1:val,P2:val,P3:val,BTN:val"

PER ogni servo (1,2,3):

angolo = pot_to_angle(valore_potenziometro)

pwm = angle_to_pwm(angolo)

APPLICA PWM al servo

AGGIORNA timeout connessione

LOG dati ricevuti e applicati

SE timeout connessione superato:

POSIZIONA tutti servo a posizione sicura (180°)

SPEGNI LED

MOSTRA stato disconnesso

Interfaccia Web

Sistema di Data Acquisition e Database

Il componente software più importante del sistema, senza il quale non sarebbe possibile la costruzione dell'app web è rappresentato dallo script Python **data_logger_script.py** che si occupa dell'acquisizione, elaborazione e archiviazione dei dati. Lo script stabilisce una connessione seriale con la scheda ricevente e implementa un parser basato su espressioni regolari per estrarre informazioni strutturate dal flusso di dati.

Il sistema di parsing riconosce pattern specifici come "**RX: P1=valore → A1=angolo**" ed estrae automaticamente tutti i vari parametri: valori dei potenziometri, angoli calcolati dei servomotori, stato del pulsante e del LED.

Sistema di Data Acquisition e Database

Questi dati grezzi vengono poi elaborati per calcolare informazioni come percentuali relative, valori PWM corrispondenti, stati di attivazione dei servo e conteggi di servomotori attivi.

L'archiviazione avviene su **InfluxDB**, un database ottimizzato per serie temporali che permette di effettuare query efficienti su grandi volumi di dati.

I dati vengono organizzati in measurement distinti: "**servo_system**" per i dati dei servomotori, "**controls**" per pulsante e LED, e measurement individuali per ciascun servo con tag appropriati per facilitare le query.

Applicazione Web

L'interfaccia web è costruita su un'architettura che utilizza **Flask** come framework backend integrato con **Socket.IO** per comunicazioni real-time. Il server espone una serie di endpoint REST API che permettono l'accesso ai dati passati e real-time memorizzati in InfluxDB.

L'endpoint **/api/latest** fornisce i dati più recenti del sistema con latenza inferiore a 100 millisecondi, mentre **/api/history** permette il recupero di serie temporali per la generazione di grafici passati. L'endpoint **/api/measurements** implementa campionamento dei dati ogni 30 secondi per popolare tabelle di riepilogo senza sovraccaricare l'interfaccia utente.

Il sistema **WebSocket** mantiene connessioni persistenti con tutti i client connessi, trasmettendo aggiornamenti ogni 2 secondi. Questo approccio elimina

Frontend

Mentre l'interfaccia utente è sviluppata con tecnologie web standard (HTML5, CSS3, JavaScript) e utilizza la libreria Chart.js per visualizzazioni grafiche.

La dashboard è organizzata in tre sezioni principali

.

La sezione "**Overview**" presenta lo stato real-time del sistema con indicatori visuali per ogni servomotore, mostrando simultaneamente valori angolari, percentuali di attivazione, valori PWM e stato operativo.

La sezione "**Charts**" offre visualizzazioni temporali interattive. I grafici vengono aggiornati automaticamente senza perdita di continuità.

La sezione "**Table**" presenta dati campionati in formato tabellare con funzionalità di filtering, sorting e export CSV per analisi offline.

The screenshot displays the 'Servo Controller Dashboard' interface. On the left, a sidebar menu includes 'Servo Control' (selected), 'Dashboard' (highlighted in blue), 'Grafici', 'Tabella Dati', and 'Impostazioni'. The main area features a title 'Servo Controller Dashboard' and a subtitle 'Controllo Real-time di 3 Servo + Potenziometri + LED'. At the top right are four status indicators: 'MESSAGGI/ORA' (252), 'SERVO ATTIVI' (1), 'PULSANTE' (LIBERO), and 'LED' (SPENTO). Below these are three servo controls labeled SERVO 1, SERVO 2, and SERVO 3. Each control panel shows the current angle (77°, 0°, 0°), potentiometer value (POT: 29%, 59%, 68%), PWM value (69, 26, 26), and a green/red button labeled 'ATTIVO' or 'STOP'.

Servo Control

Dashboard

Grafici

Tabella Dati

Impostazioni

Servo Controller Dashboard

Controllo Real-time di 3 Servo + Potenziometri + LED

SERVO 1

77 °

POT: 29% PWM: 69

SERVO 2

0 °

POT: 59% PWM: 26

SERVO 3

0 °

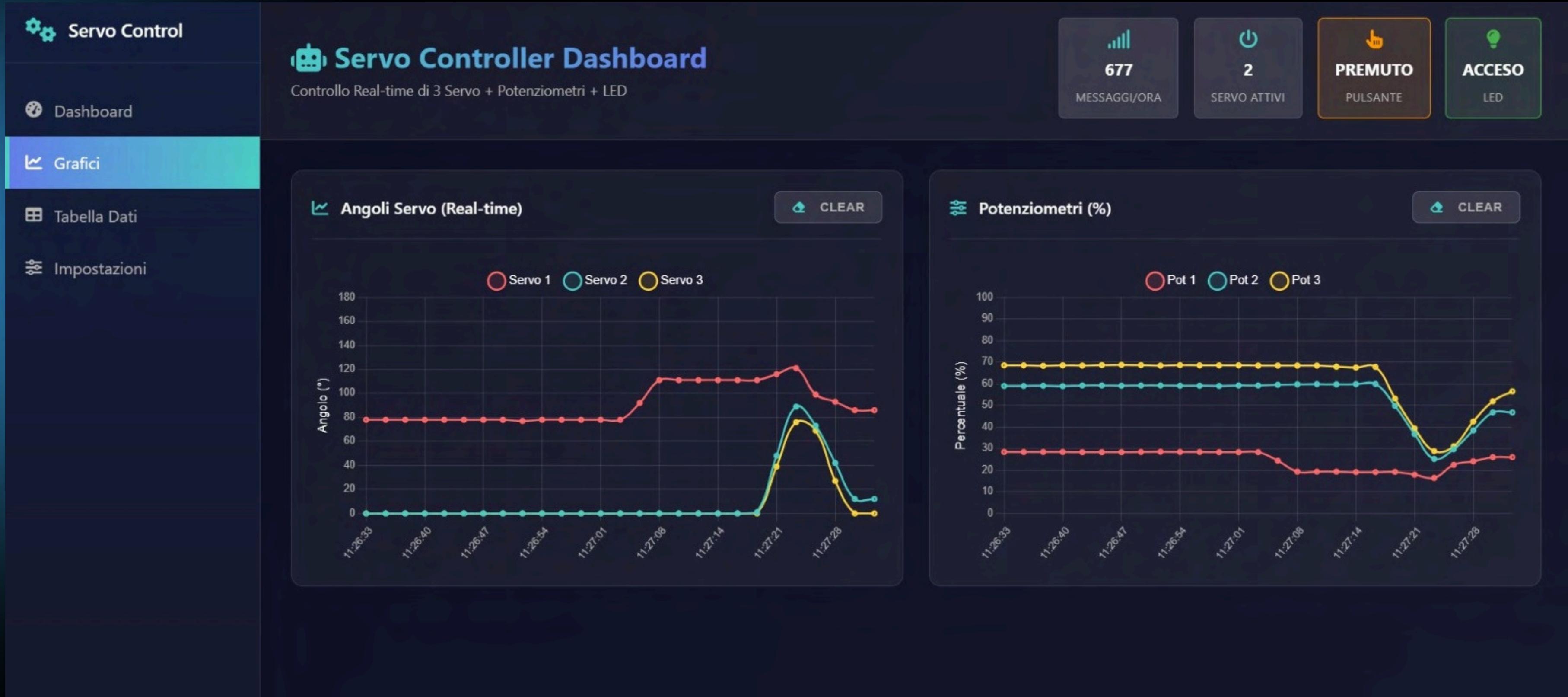
POT: 68% PWM: 26

MESSAGGI/ORA: 252

SERVO ATTIVI: 1

PULSANTE: LIBERO

LED: SPENTO



Servo Control

Servo Controller Dashboard
Controllo Real-time di 3 Servo + Potenziometri + LED

MESSAGGI/ORA: 863
SERVO ATTIVI: 3
PULSANTE: LIBERO
LED: SPENTO

Storico Misurazioni

AGGIORNA ESPORTA CSV 50 record

Ora	Data	S1 (°)	S2 (°)	S3 (°)	P1 (%)	P2 (%)	P3 (%)	BTN	LED	Attivi
09:27:49	30/05/2025	128°	17°	3°	14.6%	45.4%	49.2%	PREMUTO	ACCESO	3
09:27:30	30/05/2025	86°	12°	12°	26.1%	46.6%	46.8%	LIBERO	SPENTO	3
09:27:00	30/05/2025	78°	0°	0°	28.3%	59.2%	68.5%	LIBERO	SPENTO	1
09:26:30	30/05/2025	78°	0°	0°	28.4%	59.1%	68.5%	LIBERO	SPENTO	1
08:09:30	30/05/2025	91°	14°	0°	24.7%	46.1%	68.5%	LIBERO	SPENTO	2
08:09:00	30/05/2025	95°	23°	22°	23.6%	43.6%	44%	LIBERO	SPENTO	3
08:08:30	30/05/2025	91°	15°	56°	24.8%	45.9%	34.4%	LIBERO	SPENTO	3
08:08:00	30/05/2025	46°	0°	69°	37.2%	58%	31%	LIBERO	SPENTO	2
08:07:30	30/05/2025	45°	0°	123°	37.4%	58.5%	15.9%	LIBERO	SPENTO	2
08:07:00	30/05/2025	45°	0°	123°	37.4%	54.7%	15.9%	PREMUTO	ACCESO	2
08:06:30	30/05/2025	46°	0°	138°	37.3%	54.5%	11.7%	LIBERO	SPENTO	2

Conclusioni

Difficoltà Incontrate

Durante lo sviluppo sono emerse diverse sfide tecniche, tra queste abbiamo la **comunicazione radio** che inizialmente mostrava instabilità con perdita di pacchetti.

Inoltre la gestione del flusso dati seriali ha presentato diverse complessità dovute a rumore e interruzioni nella trasmissione.

Una sfida particolarmente complessa è stata la **conversione matematica** dei valori trasmessi via radio. I potenziometri forniscono valori analogici nel range 0-1023, che devono essere mappati in angoli servo compresi tra 0 e 180 gradi.

La conversione diretta lineare risultava inadeguata per questo è stata implementata una funzione di mappatura che limita il range massimo di movimento a 512 unità per evitare sollecitazioni meccaniche eccessive sui servomotori.

**GRAZIE PER
L'ATTENZIONE**