R6 - Progettazione hardware, realizzazione del firmware e loro integrazione

Sommario

Indice figure

Indice tabelle

1. Introduzione

Questo documento rappresenta l'output relativo alle attività A3.2 Studio ed identificazione dei componenti hardware (sensori ed elettronica di condizionamento a supporto) e all'attività A3.3 Implementazione/integrazione dell'hardware e firmware presenti nel WP3 dal titolo Studio, progettazione ed implementazione dei componenti hardware e firmware del prototipo IoT del progetto Smart Work Platform - Piattaforma IoT per la manutenzione predittiva, l'efficienza energetica delle macchine industriali ed il benessere dell'uomo nei luoghi di lavoro. Tale progetto in generale ha come obiettivo la realizzazione di una soluzione integrata per favorire la transizione di impianti industriali da tradizionali ad impianti di Industria 4.0. In particolare, si prevede la realizzazione di un dispositivo IoT munito di una piattaforma tecnologica di virtualizzazione e monitoraggio dell'impianto produttivo esistente finalizzata all'ottimizzazione dell'efficienza energetica, alla manutenzione predittiva e al miglioramento del livello di benessere e sicurezza dell'uomo nei luoghi di lavoro.

La soluzione proposta si implementa attraverso i seguenti obiettivi specifici (OS):

- OS1. Implementazione di un dispositivo IoT costituito da sensori in grado di controllare i principali parametri dell'impianto e monitorare l'ambiente di lavoro
- OS2. Implementazione di una piattaforma per il monitoraggio e la virtualizzazione degli impianti industriali dotata di Intelligenza Artificiale (IA) per l'analisi dei dati estratti dai sensori.

Per il dispositivo IoT, sono stati individuati sensori in grado di monitorare parametri fisici legati al:

- Risparmio Energetico (come ad esempio la corrente assorbita per ogni fase, vibrazioni e accelerazioni)
- Qualità dell'ambiente di lavoro e per la salvaguardia della salute degli operatori (come ad esempio sensori per il monitoraggio della temperatura, umidità, CO2, Luminosità, PM2,5, PM10, VOC, Ossigeno, Formaldeide, Presenza di fiamme, Sensori effetto hall, Pressione e frequenza sonora).

Con riferimento alla piattaforma per il monitoraggio e la virtualizzazione degli impianti industriali, si prevede la realizzazione un sistema integrato capace di acquisire i dati dai sensori installati sugli impianti e una piattaforma di virtualizzazione che permetta agli utenti di accedere a aggregazioni di tali dati (grafici, serie temporali storiche) al fine di effettuare un monitoraggio continuo degli impianti sia dal punto di vista energetico che dal punto di vista del loro funzionamento. Ciò consente di identificare eventuali azioni di manutenzione predittiva e favorire il risparmio energetico mediante la virtualizzazione della componentistica in uso.

In particolare nel seguito del documento **le tematiche trattate sono le seguenti:**

- **Progettazione e sviluppo del sistema hardware**
- **Implementazione e realizzazione del firmware ed integrazione con il sistema hardware**

Suddetta attività risulta completata con successo. Durante l'esecuzione di tale attività non si sono riscontrate anomalie o criticità.

2. Progettazione del sistema hardware

L'attività di progettazione e sviluppo del sistema hardware è stata completata con successo, senza alcuna anomalia, rispettando le specifiche tecniche e funzionali previste dal capitolato di progetto. Tale fase ha incluso la progettazione dettagliata del dispositivo IoT, realizzata mediante software CAD e strumenti specifici per la progettazione elettronica. **Per lo sviluppo degli schemi elettrici e dei circuiti stampati è stato utilizzato KiCad, un software open-source ampiamente adottato per la progettazione elettronica professionale.**
Un altro aspetto fondamentale ha riguardato la progettazione meccanica del prototipo finale. Il team ha definito e sviluppato con precisione tutte le componenti meccaniche necessarie per l'assemblaggio del dispositivo, conducendo numerosi test attraverso la stampa di prototipi preliminari. Queste prove hanno permesso di validare le scelte progettuali, ottimizzare il design e migliorare l'ergonomia e l'integrazione tra le varie parti del dispositivo.
Grazie a un processo iterativo di verifica e ottimizzazione, l'attività si è conclusa con il pieno raggiungimento degli obiettivi prefissati.

2.1. Architettura Modulare del Sistema Hardware Prototipale

Il sistema hardware prototipale è stato progettato con un'architettura modulare, finalizzata a garantire flessibilità, scalabilità e facilità di integrazione con ulteriori componenti e funzionalità. L'intero impianto si articola in tre unità principali (board elettroniche), ciascuna delle quali ospita moduli funzionali e sensori specifici, selezionati in base al tipo di misurazione o interfaccia richiesta.

Board 1 – Modulo di Acquisizione Elettrica e Rilevamento Fiamma Questa prima unità costituisce il nucleo base del sistema, dedicato prevalentemente all'acquisizione di grandezze elettriche e alla rilevazione di eventi anomali. Include:

- **Modulo ESP32**: microcontrollore principale responsabile della gestione della comunicazione, dell'elaborazione dei dati e dell'interfacciamento con gli altri moduli.
- **Modulo RTC (Real-Time Clock)**: utilizzato per la sincronizzazione temporale delle misure e per la tracciabilità degli eventi.
- **Modulo SDCard**: per la memorizzazione locale dei dati raccolti, utile per backup o operazioni offline.
- **Sensore di fiamma**: modulo per il rilevamento di radiazioni infrarosse generate da fiamme o scintille.
- **Tre moduli sensore di corrente**: ciascuno dedicato al monitoraggio di linee o dispositivi diversi, con la possibilità di misurare assorbimenti anomali o consumi elettrici.

- **Modulo accelerometro**: integrato per la rilevazione di vibrazioni o movimenti, utile in contesti di diagnostica predittiva o rilevamento incidenti.

Board 2 – Unità Ambientale Multisensore Questa scheda si occupa della raccolta di parametri ambientali di qualità dell'aria e condizioni atmosferiche, ed è dotata dei seguenti elementi:

- **Modulo ESP32**: come per la board 1, svolge il ruolo di unità di controllo.
- **Modulo RTC** e **Modulo SDCard**: replicano le funzioni già presenti nella board 1 per garantire indipendenza funzionale e sincronizzazione.
- **Sensori ambientali**:
  - Ossigeno (O_2)
  - Ozono (O_3)
  - Monossido di Carbonio (CO)
  - Particolato fine (PM2.5)
  - Anidride Carbonica (CO_2)
  - Temperatura e umidità relativa

Questi sensori consentono di realizzare un monitoraggio completo della qualità dell'aria, utile per applicazioni in ambienti industriali, urbani o indoor.

Board 3 – Estensione per Gas e Composti Volatili Questa terza board è fisicamente connessa alla board 2 e amplia ulteriormente la capacità di rilevamento chimico, includendo:

- Sensore di ammoniaca (NH_3)
- Sensore VOC (Composti Organici Volatili)
- Sensore di metano (CH_4)
- Sensore di biossido di azoto (NO_2)

L'aggiunta di questa unità consente una profilazione più dettagliata delle sostanze inquinanti presenti nell'ambiente, rendendo il sistema idoneo per scenari ad alto rischio chimico o con specifici requisiti normativi in termini di sicurezza.

2.2. Progettazione avanzata su PCB

Per la progettazione delle tre board, intesa su circuiti spampati è stato utilizzato kicad. **KiCad** è un software open source altamente professionale per la progettazione di circuiti stampati (PCB - Printed Circuit Boards), largamente impiegato in ambito industriale, accademico e hobbistico. Si tratta di una suite completa per la progettazione elettronica, che consente di passare dalla fase di ideazione del circuito alla generazione dei file di produzione (Gerber, Drill, Pick&Place) necessari alla realizzazione fisica della scheda.

La sua architettura modulare si compone di diversi strumenti principali, integrati tra loro in un flusso di lavoro coerente:

1. **Schematic Editor** È l'ambiente in cui si disegna lo schema elettrico del circuito. Qui si inseriscono i simboli elettronici dei componenti, si definiscono le connessioni logiche (netlist) e si assegnano parametri funzionali. Il software consente anche la gestione di simboli personalizzati, la verifica elettrica (ERC) e l'associazione con le librerie di footprint.
2. **Symbol e Footprint Libraries** KiCad mette a disposizione un vasto database di simboli e footprint (cioè le impronte fisiche dei componenti sulla PCB), continuamente aggiornato dalla community. È possibile creare librerie personalizzate per componenti specifici non inclusi nelle librerie standard.
3. **PCB Editor** Dopo la generazione della netlist o la sincronizzazione diretta dallo schema elettrico, si passa al disegno del layout fisico. Il PCB Editor permette di posizionare i componenti, definire le tracce (piste in rame), piazzare i via (fori di connessione tra layer), disegnare i piani di massa e impostare i layer del circuito (es. doppia faccia, multistrato). Supporta regole di design (DRC), routing manuale e automatico, nonché funzionalità avanzate come il length tuning per circuiti ad alta velocità.
4. **3D Viewer** Una delle funzionalità più apprezzate è il visualizzatore 3D, che consente di ispezionare il circuito in modo tridimensionale, con modelli realistici dei componenti. Questo aiuta a verificare interferenze meccaniche e compatibilità con contenitori o altri moduli.
5. **File di Output per la Produzione** KiCad permette di generare tutti i file necessari per la produzione della scheda:
    - File Gerber per l'incisione delle piste in rame
    - File Drill per i fori di trapano
    - File di posizionamento per le macchine pick-and-place
    - Distinte basi (BOM) in vari formati. I file possono essere visualizzati e verificati direttamente all'interno del software prima dell'invio al produttore.
6. **Gestione delle Regole di Design e Simulazione** Oltre al controllo delle regole elettriche e fisiche, KiCad offre anche il supporto per la simulazione circuitale attraverso l'integrazione con software esterni come SPICE. Questo consente un'analisi funzionale del circuito prima della sua realizzazione fisica.

KiCad si distingue per la sua **flessibilità**, **precisione** e **potenza**, pur mantenendo la filosofia open-source. È multipiattaforma (Windows, Linux, macOS) e gode di un ampio supporto dalla comunità internazionale, con aggiornamenti continui e una documentazione molto ricca.

Di seguito sono riportate le operazioni basilari effettuate per la progettazione delle suddette 3 board. Si riportano anche le immagini relative alla progettazione di ciascuna board.

1. Progettazione dello schema elettrico – Definizione funzionale e logica

La progettazione dello schema elettrico rappresenta il primo e fondamentale passo nella realizzazione di una scheda elettronica con KiCad. Questa fase costituisce il cuore logico del progetto, in quanto permette di definire in maniera chiara e strutturata il funzionamento del circuito. L'obiettivo non è semplicemente quello di disporre simboli elettronici su un'area di lavoro virtuale, ma di tradurre una logica funzionale in una rappresentazione coerente e precisa del sistema. Ogni simbolo inserito nello schema corrisponde a un componente reale e deve essere configurato in tutti i suoi aspetti: viene specificato un nome identificativo, il valore elettrico o elettronico (come una resistenza da 10kΩ, un condensatore da 100nF, ecc.), il footprint ovvero l'impronta fisica che quel componente avrà sul PCB, e se disponibile, anche un modello tridimensionale che ne faciliterà l'integrazione meccanica e la successiva visualizzazione 3D.
Un aspetto particolarmente importante in questa fase è la definizione delle connessioni logiche, ossia la netlist. Questa rappresenta l'insieme delle relazioni tra i vari nodi del circuito e costituisce la base per lo sviluppo del layout fisico della scheda. Lo schema può essere strutturato con elementi di complessità crescente, come bus dati, etichette di rete (netlabel), blocchi gerarchici per la suddivisione funzionale del progetto, e annotazioni univoche per l'identificazione automatica dei componenti. Questa modularità consente di affrontare anche progetti complessi in modo ordinato e scalabile. Una volta completato lo schema, il software consente di eseguire un'analisi preliminare attraverso il controllo delle regole elettriche (ERC), per individuare errori o incongruenze. Solo dopo questa verifica si procede alla generazione della netlist, file cruciale che guiderà la successiva disposizione fisica dei componenti sulla PCB. In questa fase, l'accuratezza e la coerenza dello schema elettrico sono determinanti per la buona riuscita dell'intero progetto.
In sintesi, il processo di realizzazione di una PCB con KiCad parte dalla definizione dello schema elettrico. In questa fase, non si tratta solo di collegare simboli elettronici su un foglio virtuale: ogni simbolo corrisponde a un componente fisico e deve essere correttamente parametrizzato. Ad ogni simbolo vengono associati: il nome, il valore (es. resistenza da 10kΩ), un footprint (impronta fisica del componente), un modello 3D opzionale, e la netlist logica che descrive le connessioni. È possibile includere blocchi gerarchici, bus, netlabel, e

annotazioni univoche, per una gestione modulare e scalabile del progetto. Una volta completato, lo schema viene “compilato” per generare la netlist, che sarà il file chiave da cui deriverà la disposizione fisica dei componenti.

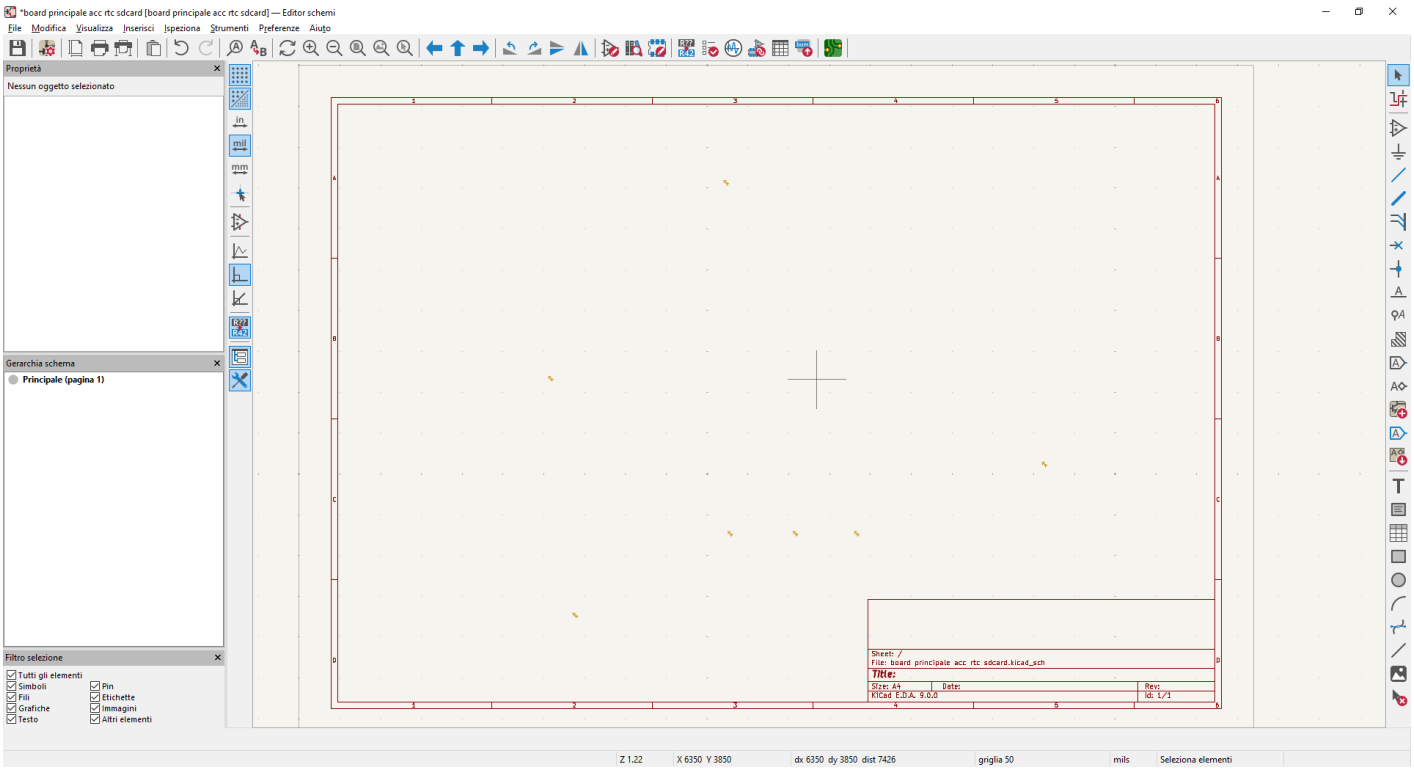


Figura 1: schermata kikad iniziale

2. Posizionamento fisico dei componenti – Ottimizzazione topologica e termica

Il posizionamento fisico dei componenti rappresenta una fase cruciale nella progettazione di una scheda elettronica, in cui entrano in gioco considerazioni non solo di natura logica, ma anche topologica, elettromagnetica e termica. A differenza di quanto si potrebbe pensare, questa attività non ha un valore meramente estetico, bensì funzionale e ingegneristico. Ogni componente deve essere collocato seguendo criteri che mirano a ottimizzare le prestazioni elettriche del circuito, a ridurre le interferenze e a migliorare la gestione del calore generato durante il funzionamento.
In particolare, si presta grande attenzione alla minimizzazione della lunghezza delle tracce che trasportano segnali critici, come linee di clock, segnali digitali ad alta frequenza o linee differenziali, al fine di preservarne

l'integrità e ridurre le problematiche legate alla diafonia e alle perdite. Il layout viene strutturato in modo da seguire un flusso logico coerente, tipicamente dalla sorgente al carico, per semplificare le connessioni e migliorare la leggibilità e la manutenzione del circuito. Parallelamente, si adottano soluzioni progettuali per mitigare fenomeni di accoppiamento capacitivo o induttivo, che potrebbero generare disturbi o interferenze elettromagnetiche (EMI). Un altro aspetto fondamentale è la gestione termica. I componenti che dissipano una quantità significativa di potenza vengono posizionati strategicamente in aree che permettono una migliore conduzione e dispersione del calore, sfruttando piani di rame estesi, vias termici e una disposizione tale da evitare surriscaldamenti localizzati. In questo contesto, la progettazione diventa una vera e propria ottimizzazione multidimensionale, in cui le esigenze elettriche, meccaniche e termiche devono coesistere in equilibrio. Il risultato di questa fase è un layout robusto, efficiente e pronto per essere instradato, come mostrato nella schermata successiva relativa alla disposizione iniziale dei componenti sulla superficie del circuito stampato.

Sintetizzando, il posizionamento dei componenti non è solo un'operazione estetica, ma segue precisi criteri funzionali, elettromagnetici e termici. Si posizionano i componenti in modo tale da minimizzare la lunghezza delle connessioni critiche (es. clock, segnali digitali ad alta frequenza, linee differenziali), mantenere un flusso logico (dalla sorgente verso il carico), e prevenire fenomeni di diafonia, accoppiamenti indesiderati o interferenze EMI. Il layout è influenzato anche dalla dissipazione termica dei componenti: i componenti che generano calore vengono posizionati con abbondante rame libero o vicino a vias termici per favorire la dispersione. Nella schermata seguente si vede lo screen relativo all'aggiunta dei vari componenti su piastra.

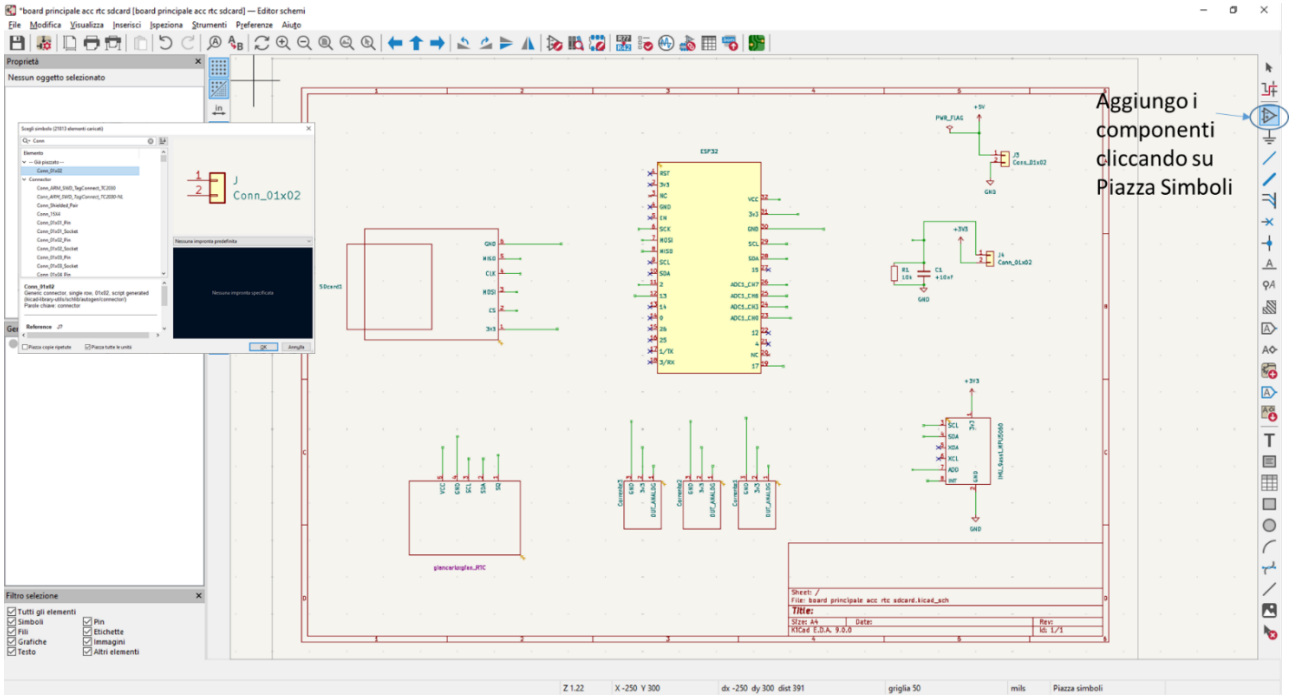


Figura 2: aggiungere componenti

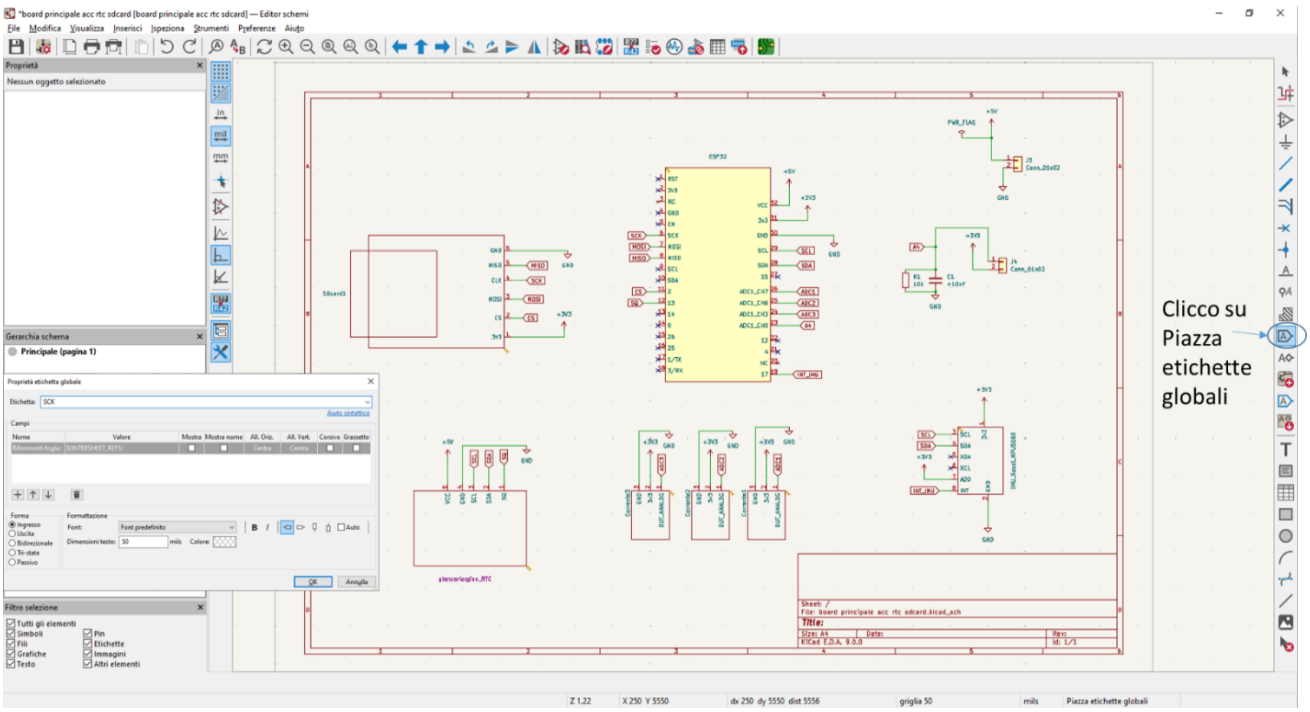


Figura 3: etichette dei componenti

3. Tracciatura delle piste in rame – Routing controllato e ingegnerizzato

Il **routing** è una fase critica: le piste rappresentano i percorsi elettrici effettivi che collegano i pad dei componenti. La tracciatura può avvenire manualmente (preferibile per segnali critici) oppure con il supporto dell'**autorouter** di KiCad (come FreeRouting integrabile). Ogni traccia deve rispettare parametri specifici: larghezza minima (in funzione della corrente), distanza da altre piste o oggetti (per evitare archi elettrici o disturbi), forma (evitare angoli a 90° per ridurre discontinuità di impedenza), e layer (top, bottom, o interni). In caso di segnali ad alta frequenza o linee differenziali (es. USB, Ethernet, HDMI), si utilizzano tecniche di **impedance matching** e **routing simmetrico**, controllando la larghezza della pista e la distanza tra le coppie per garantire caratteristiche elettriche uniformi.

Le piste sono tracciate con attenzione anche in funzione del piano di massa, per ridurre i loop di corrente e contenere le emissioni EMI.

4. Inserimento dei vias e fori – Interconnessione tra layer e meccanica Quando una connessione deve attraversare più strati, vengono utilizzati i **vias**, piccoli fori metallizzati che collegano piste su layer diversi. KiCad permette di configurare diverse tipologie:

- **Vias standard passanti (through-hole vias)**: attraversano tutta la scheda.
- **Vias ciechi (blind vias)**: collegano un layer superficiale a un layer interno.
- **Vias sepolti (buried vias)**: collegano solo strati interni, non visibili dall'esterno. I vias possono essere **tented**, ovvero coperti dalla solder mask per evitare esposizione. Oltre ai vias elettrici, vengono inseriti anche i **fori meccanici**, utilizzati per viti di fissaggio, allineamenti, forature per guide e incastri. Ogni foro viene definito in termini di diametro, rivestimento interno, presenza o meno di pad in rame, e tolleranza dimensionale. Anche i **fori per componenti THT** (through-hole technology) fanno parte di questa fase e vengono definiti con attenzione al footprint, all'orientamento e alla robustezza meccanica.

5. Creazione delle zone in rame – Piani di massa, alimentazione e schermature Le zone in rame (o **poligoni**) vengono utilizzate per realizzare piani di massa, piani VCC o altre alimentazioni. Queste zone migliorano l'integrità del segnale, riducono le EMI, e facilitano la dissipazione termica. Durante la loro definizione, si assegna una net (es. GND) e si configurano le regole di isolamento, il tipo di riempimento (solido, reticolare, termico), e la priorità rispetto ad altre aree. Le zone vengono "ristrette" automaticamente attorno ad altri oggetti in base alle regole DRC, evitando cortocircuiti e assicurando separazione dielettrica adeguata.

Durante la progettazione di un circuito stampato, la creazione delle zone in rame riveste un ruolo strategico, sia per il corretto funzionamento elettrico del sistema sia per la sua affidabilità complessiva. Queste aree, che si estendono come grandi superfici conduttive sulla PCB, vengono impiegate principalmente per realizzare i piani di massa, le alimentazioni principali come il VCC, o eventuali linee dedicate ad altri livelli di tensione. L'utilizzo di tali zone non è soltanto funzionale al trasporto di corrente, ma consente di ottenere numerosi vantaggi in termini di prestazioni e robustezza.

Dal punto di vista dell'integrità del segnale, un piano di massa continuo e ben progettato riduce le impedenze indesiderate e minimizza le riflessioni, soprattutto nei circuiti ad alta frequenza. Allo stesso tempo, queste superfici contribuiscono in modo significativo alla riduzione delle interferenze elettromagnetiche (EMI), agendo da schermo protettivo contro segnali indesiderati che potrebbero compromettere il funzionamento del circuito. Inoltre, poiché il rame possiede un'ottima conducibilità termica, queste zone facilitano la dispersione del calore generato dai componenti elettronici, contribuendo alla gestione termica del sistema.

Durante la definizione di una zona in rame, è fondamentale assegnarle una net specifica, ad esempio GND per la massa, e impostare una serie di parametri come il tipo di riempimento (che può essere solido, a maglia o termico) e il livello di priorità rispetto ad altre aree. Il software, nel rispetto delle regole di progettazione (DRC), provvede automaticamente a modellare i contorni delle zone attorno ai componenti e alle tracce esistenti, garantendo un isolamento elettrico adeguato e prevenendo cortocircuiti. In questo modo, le zone in rame non solo ottimizzano le prestazioni elettriche della scheda, ma rappresentano anche un elemento chiave per la sicurezza e la stabilità operativa del progetto.

6. Design Rule Check (DRC) e Electrical Rule Check (ERC) Queste due fasi rappresentano la verifica strutturale del layout. Il **DRC** verifica che tutte le geometrie rispettino le regole di progetto: larghezze, distanze minime, sovrapposizioni, connessioni mancanti, interferenze tra vias e piste. Il **ERC**, invece, controlla le regole elettriche definite nello schema, come alimentazioni scollegate, net isolate, polarità invertite, oppure uscite multiple collegate tra loro. Qualsiasi violazione genera un report dettagliato, utile per correggere errori prima della produzione. Il Design Rule Check (DRC) e l'Electrical Rule Check (ERC) sono fasi fondamentali nel processo di progettazione di una scheda elettronica, che svolgono una funzione di controllo e verifica della correttezza del layout, rispettivamente dal punto di vista strutturale e elettrico. Questi due strumenti sono essenziali per garantire che il circuito stampato, prima di essere realizzato fisicamente, soddisfi tutte le condizioni necessarie per il corretto funzionamento e la sicurezza del progetto.
Il DRC si concentra sugli aspetti geometrici del layout e verifica che tutte le tracce, vias, componenti e altre caratteristiche fisiche della PCB rispettino le regole di progettazione stabilite. Queste regole riguardano, per esempio, le larghezze minime delle tracce, le distanze tra componenti e piste, la separazione tra vias e tracce, o la corretta disposizione delle aree di alimentazione. In altre parole, il DRC controlla che la geometria della scheda sia ottimizzata per evitare cortocircuiti, sovrapposizioni indesiderate e difetti che potrebbero compromettere la realizzazione della PCB. Un errore comune che il DRC può individuare è, ad esempio, una traccia troppo stretta che potrebbe non sopportare il carico di corrente previsto, o una distanza troppo ridotta tra tracce ad alta tensione e tracce a bassa tensione, che potrebbe causare malfunzionamenti o interferenze.
Il ERC, invece, si occupa degli aspetti elettrici del progetto e verifica che lo schema elettrico, cioè la rappresentazione logica del circuito, non contenga errori legati alla connessione dei componenti. Esso controlla, ad esempio, se tutte le alimentazioni sono correttamente collegate, se non ci sono net isolate che potrebbero essere inutilizzate o mal collegate, se la polarità di componenti come diodi o condensatori è corretta, e se non ci sono conflitti di connessione come uscite multiple collegate tra loro in modo errato. L'ERC è cruciale per evitare che circuiti danneggiati o incompleti vengano prodotti, il che potrebbe portare a circuiti non funzionanti o addirittura pericolosi. Ogni violazione rilevata da questi controlli genera un report dettagliato, che funge da guida per correggere i difetti prima che la produzione della PCB inizi. In questo modo,

DRC ed ERC sono due fasi indispensabili per garantire che il design della PCB sia conforme agli standard richiesti e pronto per la produzione, minimizzando il rischio di errori e migliorando la qualità finale del prodotto.
Per ogni scheda con componenti si fa un check sulla compatibilità elettrica tra i vari componenti inseriti: di seguito la schermata.

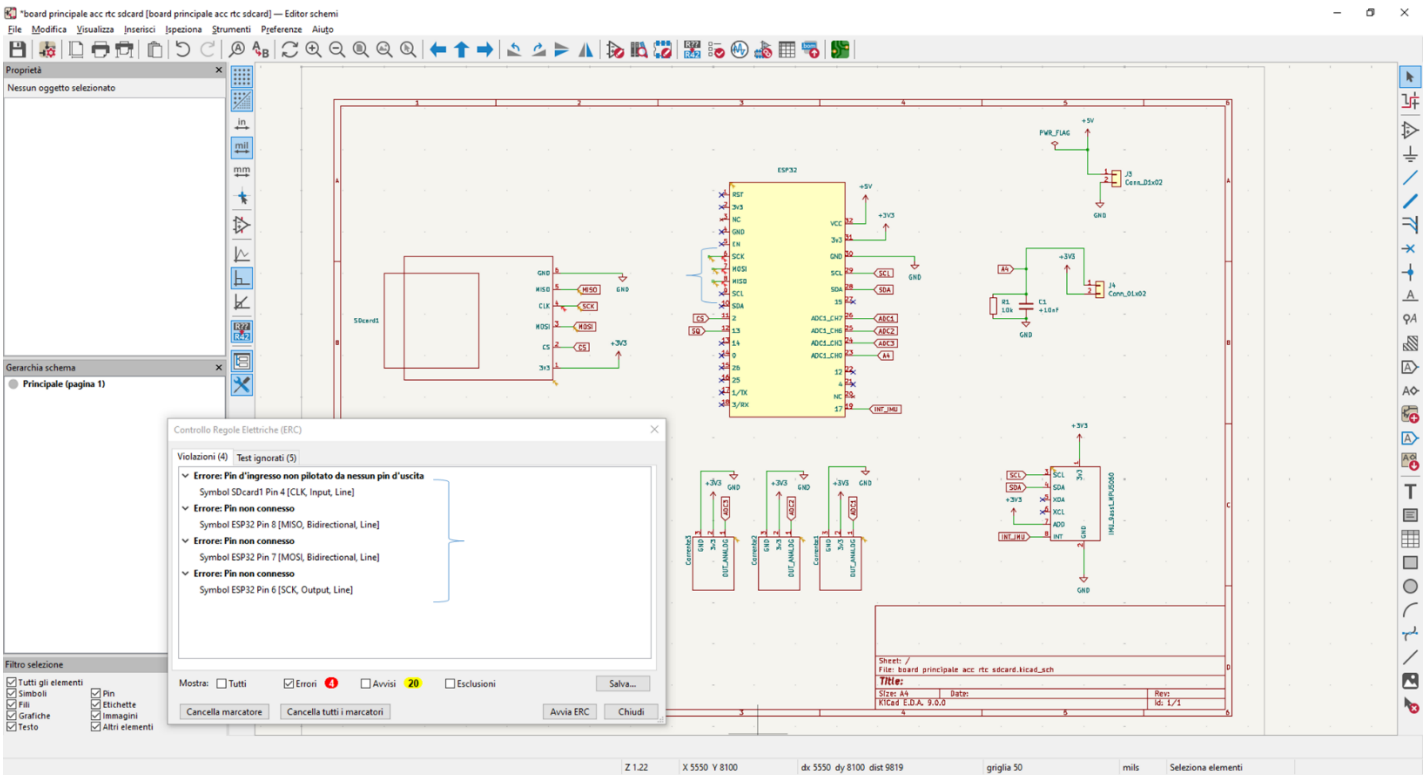


Figura 4: Electrical rule check

7. Rendering 3D – Ispezione visiva realistica della board KiCad offre una modalità di rendering 3D interattiva che consente di ispezionare la board in tempo reale con tutti i componenti posizionati. Questa visualizzazione è utile per verificare collisioni meccaniche, compatibilità con alloggiamenti, orientamento corretto dei connettori, e per realizzare documentazione professionale per il cliente o per l'ufficio tecnico.
Il rendering 3D in KiCad rappresenta una funzionalità avanzata che consente una visione altamente interattiva e realistica della scheda progettata. Questo strumento si rivela di fondamentale importanza nella fase finale del processo di progettazione, poiché consente di visualizzare la board come se fosse già fisicamente realizzata, ma all'interno di un ambiente digitale. A differenza di una semplice visualizzazione bidimensionale, il

rendering 3D offre una rappresentazione tridimensionale dei componenti, delle piste di rame e degli altri dettagli della PCB, permettendo così una valutazione molto più accurata.
Una delle principali utilità di questa funzione è la verifica delle possibili collisioni meccaniche tra i vari componenti montati sulla scheda. Spesso, durante la progettazione, i componenti elettronici possono interferire tra loro o con altre parti del sistema (come scatole o alloggiamenti), e il rendering 3D permette di identificare facilmente questi conflitti prima che si proceda con la produzione. Questo può evitare costosi errori nella fase di assemblaggio o la necessità di dover modificare il layout una volta che il prototipo fisico è stato realizzato.
Inoltre, la visualizzazione 3D aiuta a garantire che l'orientamento dei connettori, dei pin e dei componenti stessi sia corretto. Questo è particolarmente importante per i connettori e i dispositivi con polarità o orientamento specifici, come i connettori USB, i pin header o i componenti SMD (Surface-Mount Devices). La possibilità di ruotare e ispezionare la scheda in tempo reale permette di confermare che tutti i componenti siano posizionati nel verso giusto e che non ci siano errori di assemblaggio che potrebbero compromettere il funzionamento del circuito.
Infine, il rendering 3D non è solo un utile strumento di verifica tecnica, ma è anche uno strumento molto valido per la produzione di documentazione professionale. Le immagini 3D generate possono essere utilizzate per creare presentazioni visive di alta qualità da inviare al cliente, per la documentazione tecnica da fornire agli uffici interni, o per realizzare schede tecniche dettagliate che illustrano la disposizione e le caratteristiche dei componenti sulla scheda. Questa capacità di generare immagini realistiche della board aiuta a comunicare chiaramente la progettazione e a evitare fraintendimenti o problemi legati a interpretazioni errate del layout. In questo modo, il rendering 3D non solo ottimizza il processo di progettazione, ma contribuisce anche alla comunicazione e alla documentazione del progetto in modo professionale e preciso.
Ora di seguito, si riportano gli schemi elettrici progettati su kikad per le tre board.

Sulla board 1 sono presenti i seguenti componenti:

- **Modulo ESP32**: microcontrollore principale responsabile della gestione della comunicazione, dell'elaborazione dei dati e dell'interfacciamento con gli altri moduli.
- **Modulo RTC (Real-Time Clock)**: utilizzato per la sincronizzazione temporale delle misure e per la tracciabilità degli eventi.
- **Modulo SDCard**: per la memorizzazione locale dei dati raccolti, utile per backup o operazioni offline.
- **Sensore di fiamma**: modulo per il rilevamento di radiazioni infrarosse generate da fiamme o scintille.
- **Tre moduli sensore di corrente**: ciascuno dedicato al monitoraggio di linee o dispositivi diversi, con la possibilità di misurare assorbimenti anomali o consumi elettrici.
- **Modulo accelerometro**: integrato per la rilevazione di vibrazioni o movimenti, utile in contesti di diagnostica predittiva o rilevamento incidenti.

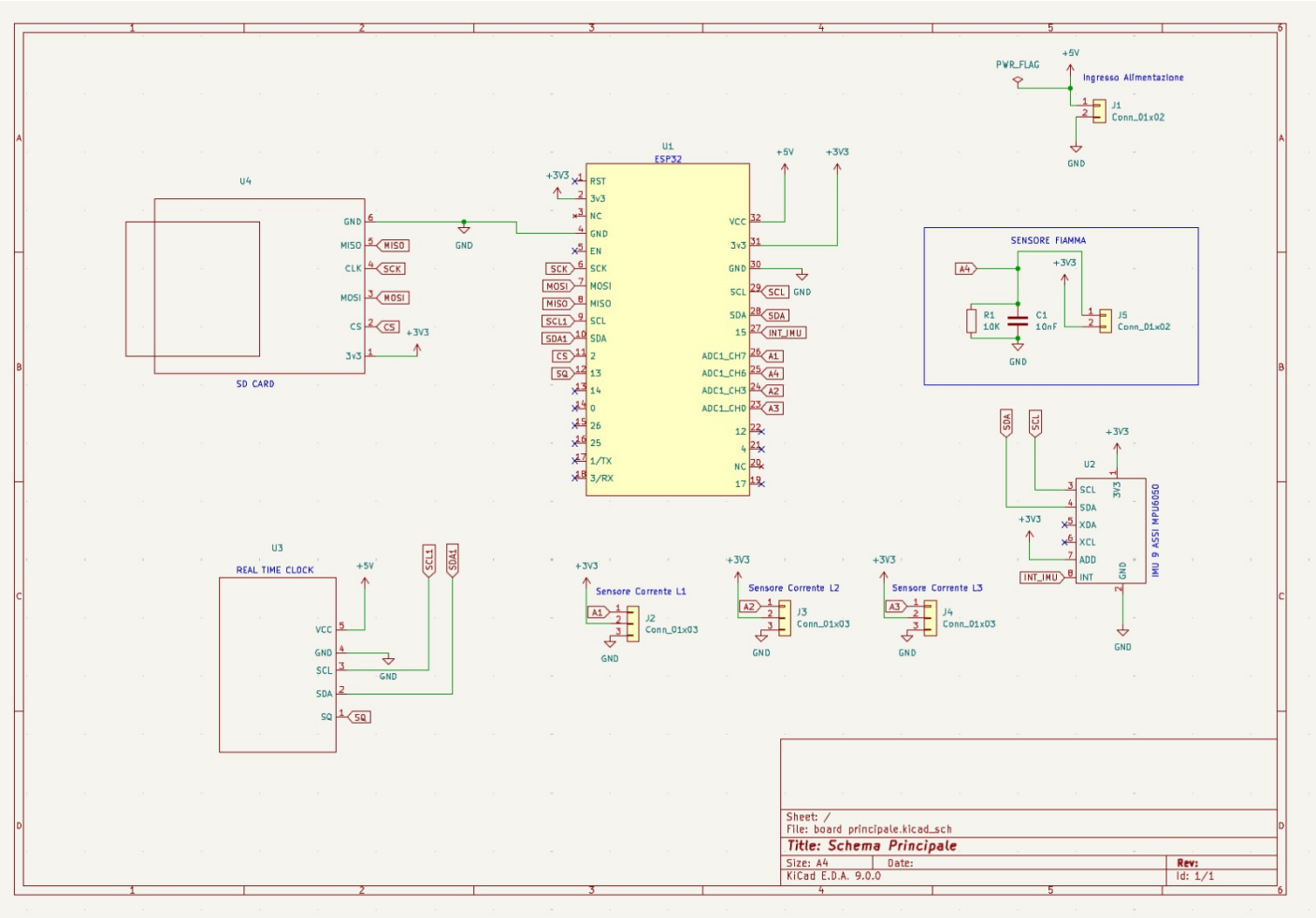


Figura 5: progettazione board 1

Sulla board 2 sono presenti i componenti:

- **Modulo ESP32**: come per la board 1, svolge il ruolo di unità di controllo.
- **Modulo RTC** e **Modulo SDCard**: replicano le funzioni già presenti nella board 1 per garantire indipendenza funzionale e sincronizzazione.
- **Sensori ambientali**:
  - Ossigeno (O_2)
  - Ozono (O_3)
  - Monossido di Carbonio (CO)

- Particolato fine (PM2.5)
- Anidride Carbonica (CO_2)
- Temperatura e umidità relativa

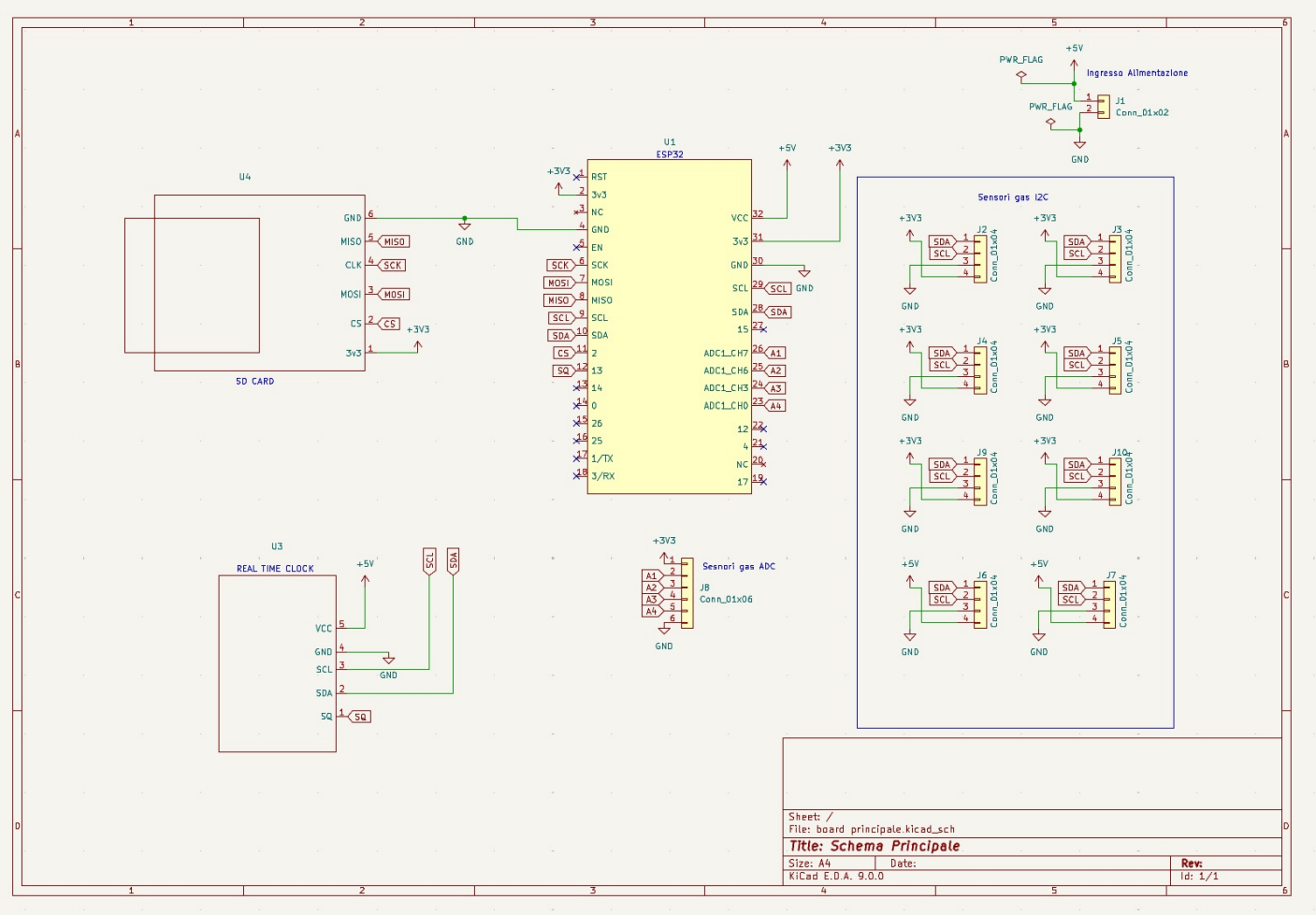


Figura 6: progettazione board 2

La board 3, collegata alla board 2 presenta i seguenti sensori:

- Sensore di ammoniaca (NH_3)
- Sensore VOC (Composti Organici Volatili)
- Sensore di metano (CH_4)
- Sensore di biossido di azoto (NO_2)

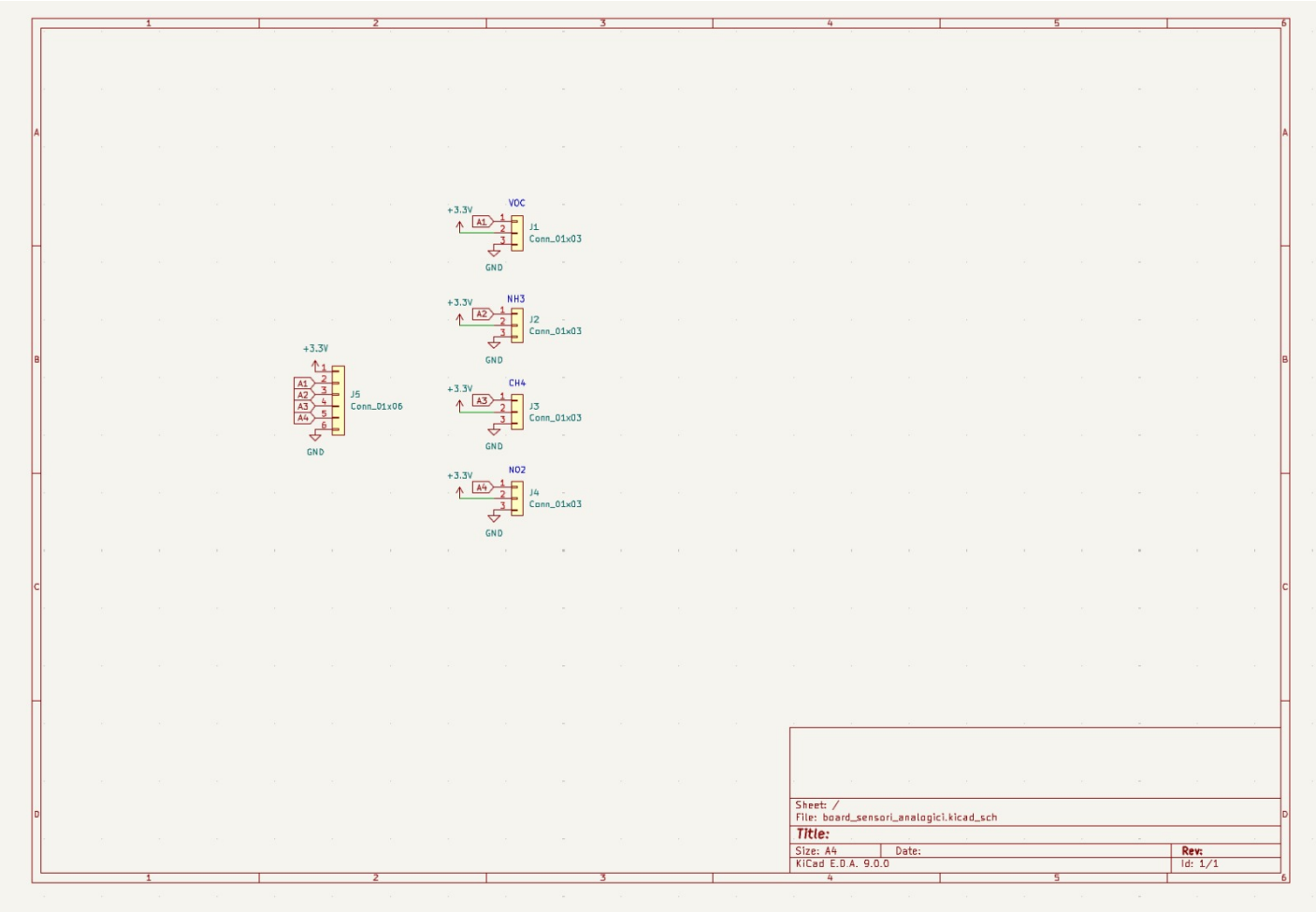


Figura 7: progettazione board 3

8. Generazione dei file di produzione – Gerber, Excellon, BOM, Pick & Place La fase finale consiste nell'esportazione dei file di produzione:

- I **file Gerber** contengono tutte le informazioni geometriche relative a rame, serigrafia, solder mask, contorni.
- I **file Excellon** definiscono la foratura (posizione, diametro, tipo di foro).
- La **BOM (Bill of Materials)** elenca tutti i componenti utilizzati, con codici, quantità e riferimenti.
- I file **Pick & Place** contengono le coordinate esatte dei componenti per il montaggio automatico SMT. Questi file vengono generalmente forniti a un produttore PCB e a un'azienda di assemblaggio.

Di seguito si riportano la versione grafica del file Gerber per ciascuna delle tre board.Da notare che i punti “gialli” sono relativi alle forature.

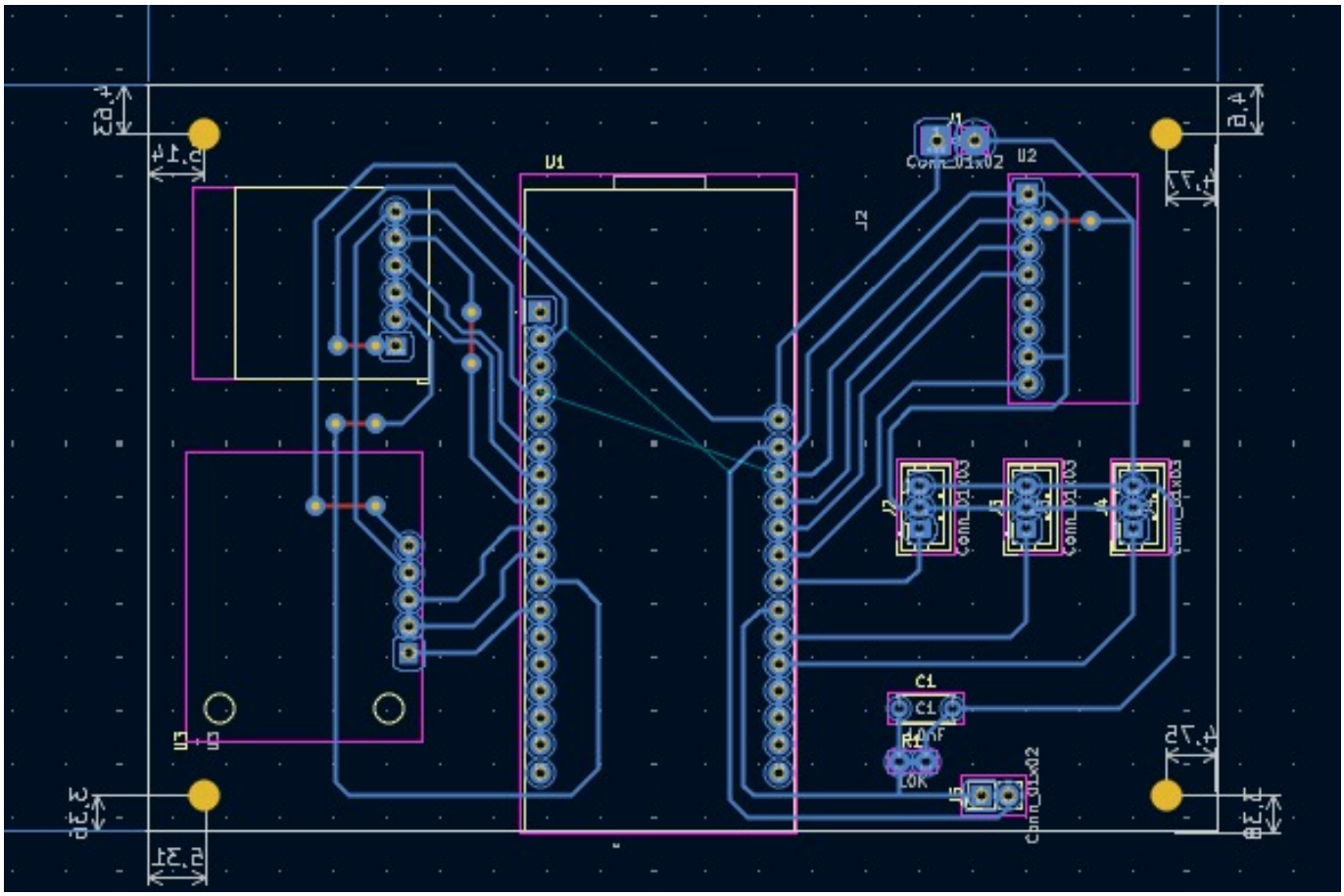


Figura 8: visualizzazione grafica del file Gerber per la board 1

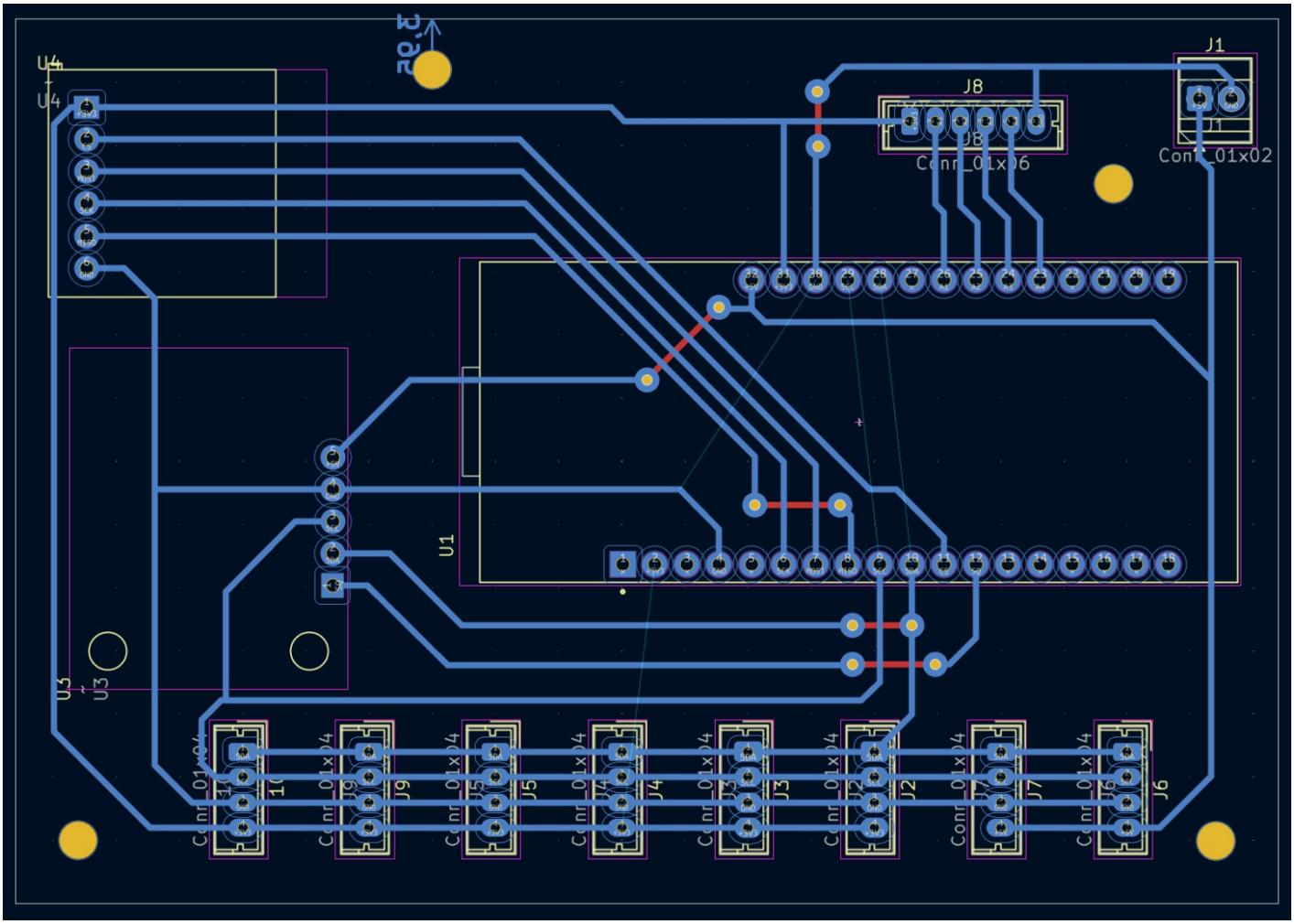


Figura 9: visualizzazione file Gerber per la board 2

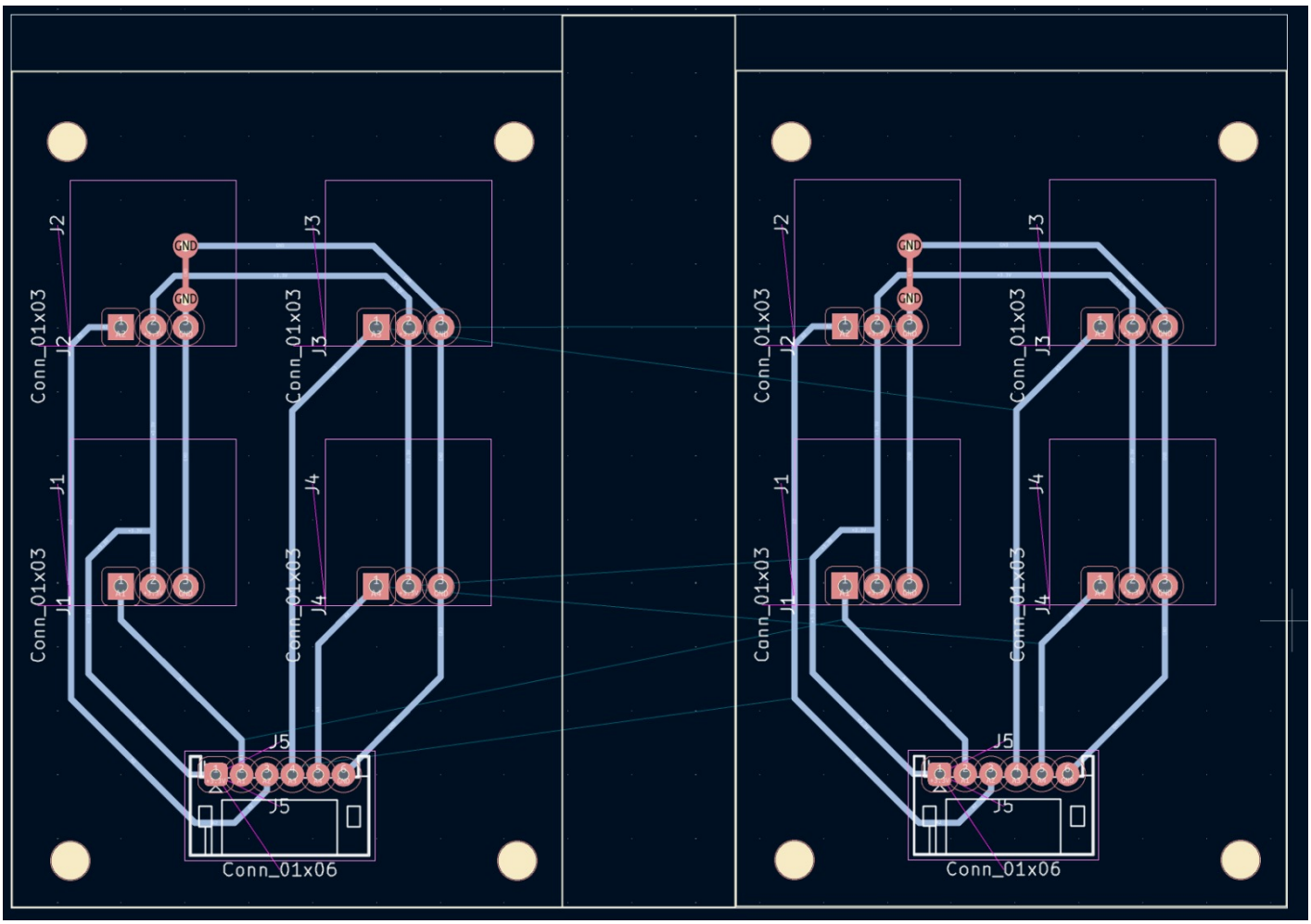


Figura 10: visualizzazione grafica del file Gerber per la board 3

Di seguito si riportano per ciascuna board si mostra il rendering grafico ovvero la visualizzazione grafica con tutti i componenti presenti.

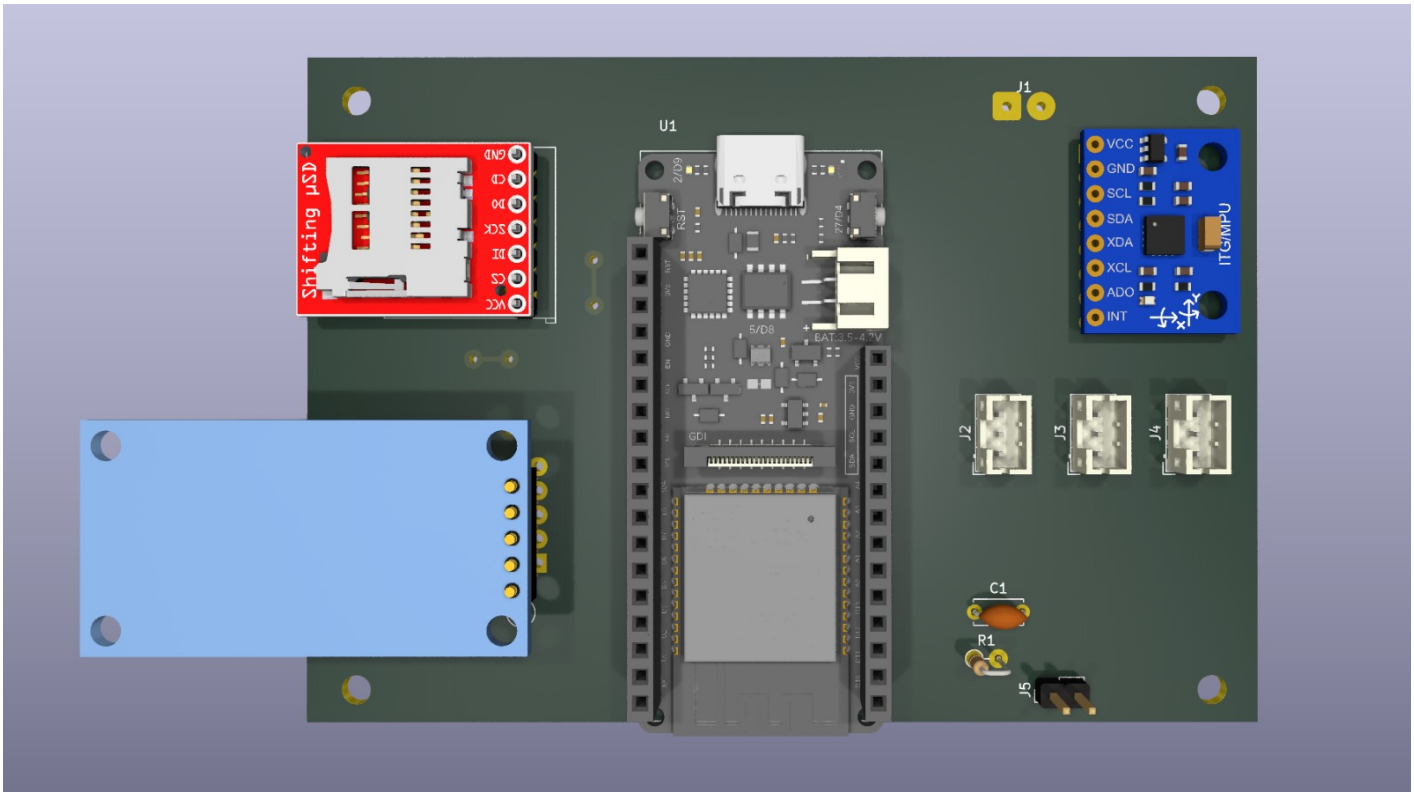


Figura 11: componenti presenti sulla board 1

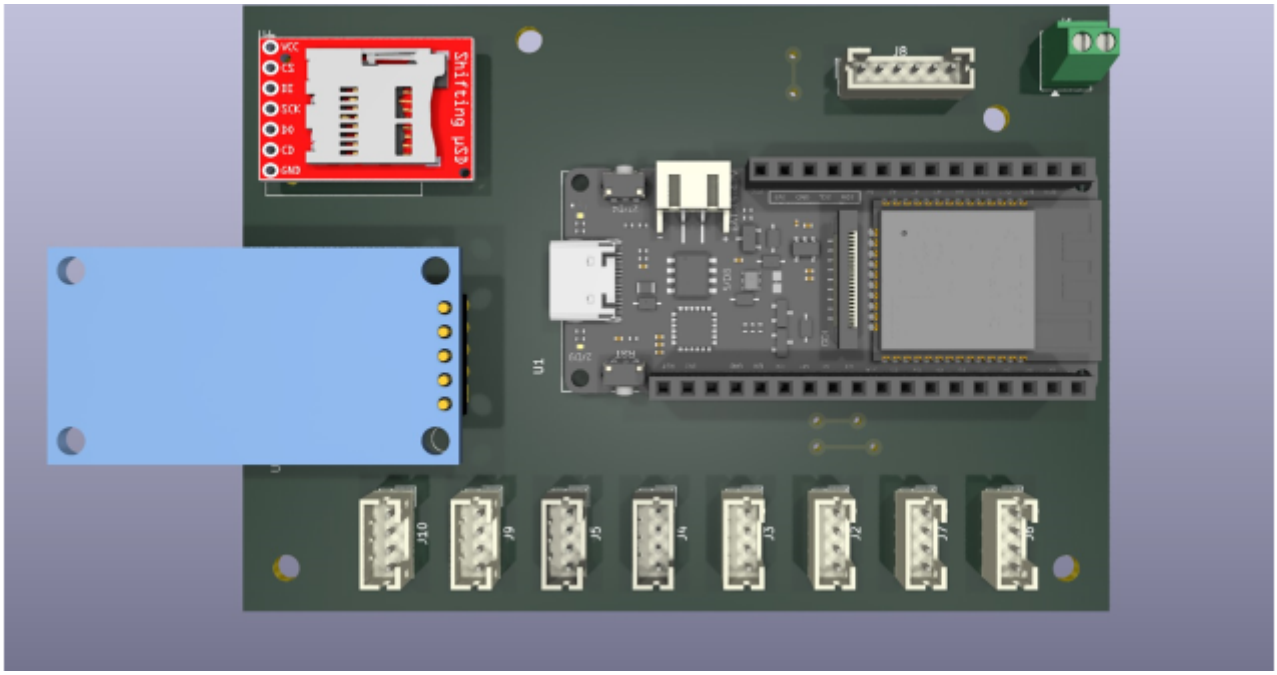


Figura 12: componenti presenti sulla board 2

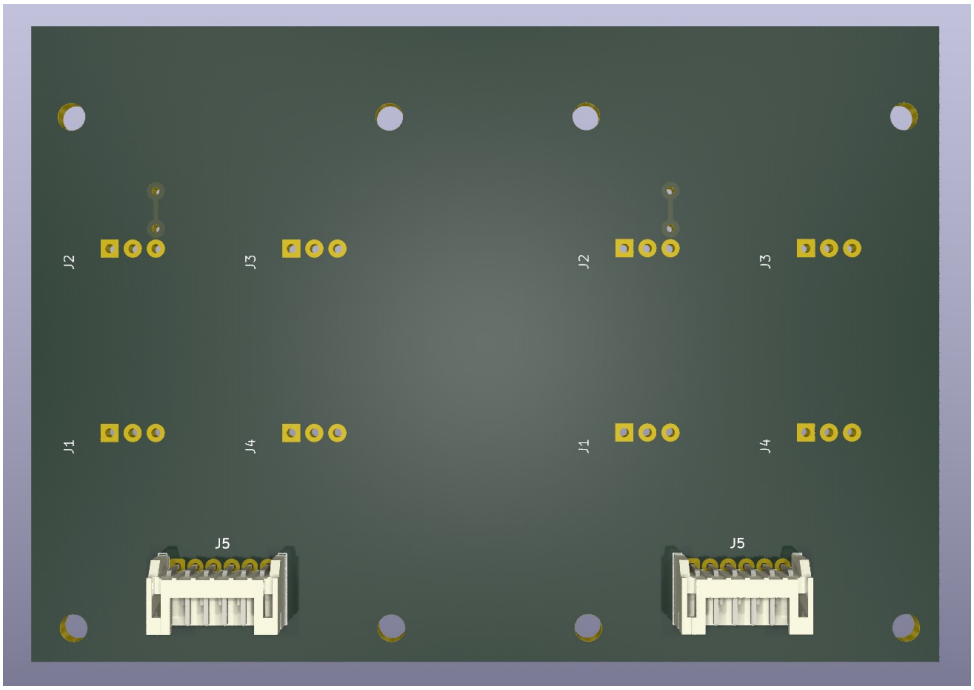


Figura 13: visualizzazione della board 3

2.3. Flusso operativo per la progettazione di PCB propedeutica alla realizzazione

Ecco una descrizione approfondita delle operazioni principali svolte dal software ad esempio FlatCAM (Flat Computer Aided Manufacturing), il quale viene utilizzato per trasformare i file Gerber e Excellon generati da software di progettazione elettronica (come KiCad) in percorsi utensile (toolpaths) adatti alla lavorazione con macchine CNC per la produzione fisica di circuiti stampati (PCB) mediante incisione meccanica.
Anche in questo caso si riportano alcuni screenshot delle più importanti operazioni effettuate. Ovviamente si riportano le immagini relative alle operazioni effettuate per la board 1 in quanto per le altre due board le operazioni ripetute sono simili.

1. Importazione dei file Gerber ed Excellon

La fase iniziale di qualsiasi processo di lavorazione PCB con FlatCAM consiste nell'importazione dei file Gerber e Excellon, che costituiscono la base dati per la creazione dei percorsi utensile destinati alla CNC. I file Gerber, in particolare, descrivono con precisione i diversi layer del circuito stampato, come ad esempio il layer del rame (top e bottom), il soldermask, la serigrafia e i contorni della scheda. Ciascun layer viene convertito da FlatCAM in oggetti geometrici vettoriali, perfettamente sovrapponibili e navigabili all'interno dell'interfaccia grafica del software.
Parallelamente, i file Excellon – generalmente associati all'estensione *.drl o *.txt – contengono le coordinate esatte dei fori da realizzare. Questi fori possono riguardare sia vias (connessioni elettriche tra diversi layer del PCB), sia pin per componenti through-hole, sia elementi meccanici come i fori di fissaggio. Durante l'importazione, FlatCAM consente di impostare parametri come il diametro dei fori, la tipologia di punta da associare e la modalità di interpretazione del file, in particolare per quanto riguarda il formato delle coordinate (ad esempio leading o trailing zero omission, e numero di cifre decimali).
Una volta caricati, tutti i layer e i dati di foratura vengono visualizzati sul workspace di FlatCAM come entità distinte ma combinabili. L'utente può interagire con ciascun layer, verificarne l'allineamento reciproco e correggere eventuali discrepanze nei riferimenti di origine (origin point), assicurandosi che tutte le geometrie siano correttamente registrate rispetto a un comune punto di zero macchina. È inoltre possibile scegliere tra diverse unità di misura (millimetri o pollici) e confermare il corretto riconoscimento del formato numerico dei file importati, un passaggio cruciale per evitare errori di scala nella lavorazione finale.
Questa fase rappresenta un momento chiave per il controllo visivo e geometrico del progetto PCB, in quanto consente all'utente di accertarsi che i dati digitali corrispondano esattamente al layout previsto e che siano pronti per la generazione del G-code ottimizzato per la macchina CNC. Un'accurata importazione è essenziale per garantire la qualità, la precisione e la ripetibilità dell'intero processo di fabbricazione.

La prima operazione fondamentale che FlatCAM esegue consiste nell'importazione dei file Gerber, che descrivono i vari layer della scheda PCB (in particolare il layer del rame, quello dei contorni, e altri opzionali), insieme ai file Excellon, che contengono le coordinate di foratura. Durante questa fase, FlatCAM interpreta le geometrie contenute in questi file e le visualizza come oggetti grafici vettoriali nell'interfaccia. L'utente può verificare l'allineamento tra i diversi layer e assicurarsi che le unità di misura (millimetri o pollici), il formato di output e i riferimenti siano corretti.

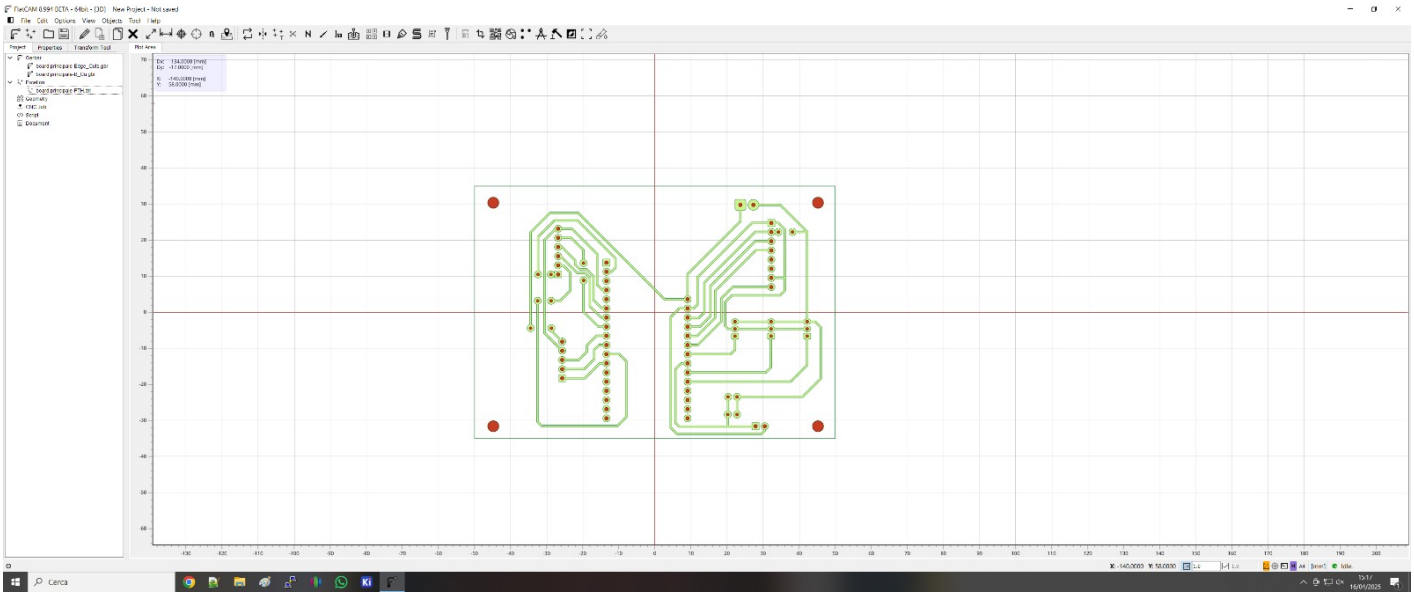


Figura 14: visualizzazione grafica dell'importazione del file Gerber ed Excellor su Flatcam per la board 1

È stato importante impostare l'utensile utilizzare per l'operazione di "scontornatura delle piste":

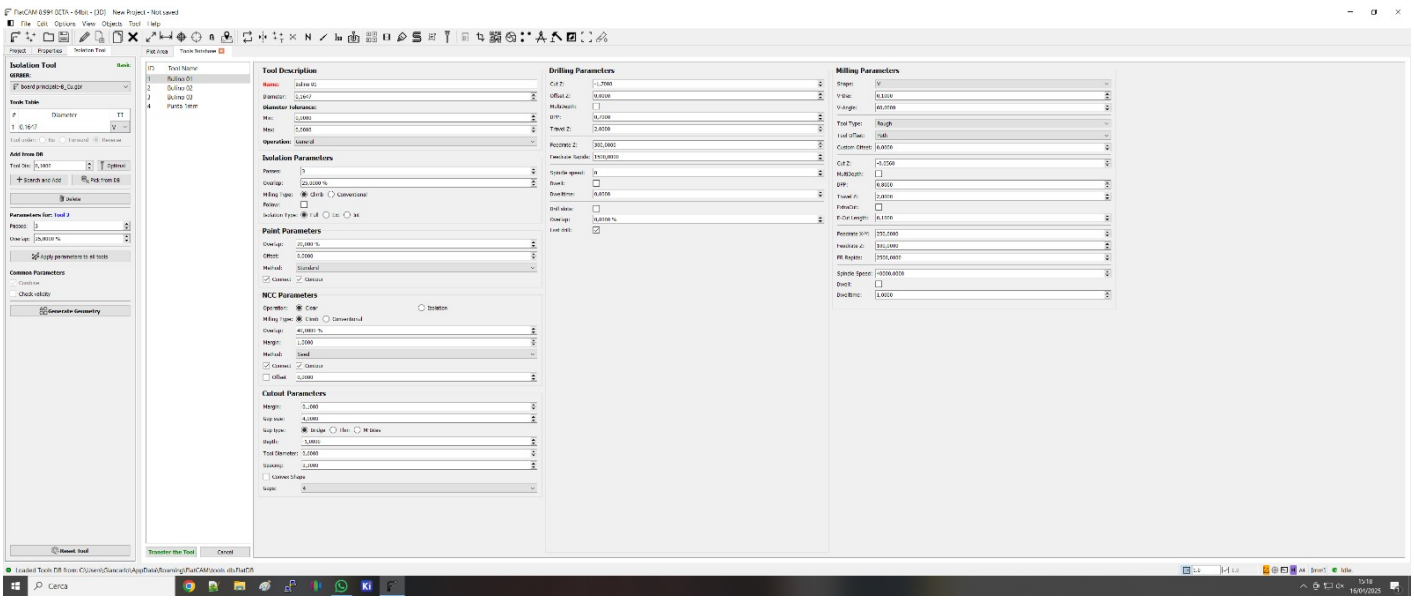


Figura 15:

2. Generazione dei percorsi di isolamento delle piste in rame

Una volta caricati i layer, FlatCAM consente la creazione dei percorsi per la fresatura di isolamento, una tecnica che permette di "scavare" il rame attorno alle piste mantenendole elettricamente isolate tra loro. L'utente seleziona il layer di rame (tipicamente il Top o Bottom Copper) e imposta i parametri dell'utensile, come il diametro effettivo della punta e il numero di passaggi concentrici da eseguire attorno alle tracce. Il software elabora queste informazioni per generare i contorni di fresatura, calcolando percorsi ottimizzati e precisi che garantiscono la continuità elettrica delle piste e l'isolamento tra loro. È possibile anche specificare profondità di lavorazione, velocità di avanzamento, e strategie di compensazione.

Nel processo di preparazione dei file CNC per la realizzazione fisica delle piste in rame su una scheda PCB, si è utilizzato **FlatCAM** come strumento di post-processing, in particolare facendo uso della funzionalità denominata **Isolation Tool**. Questo strumento consente di generare percorsi utensile atti a rimuovere il materiale in modo selettivo attorno alle tracce conduttive, isolandole elettricamente dal resto del piano di rame. Durante questa fase, è stato selezionato un utensile di tipo **bulino** con punta conica da **0.1 mm** e **angolo di 60°**, tipicamente utilizzato per incisioni di precisione su PCB. Tale scelta è cruciale per ottenere un'incisione fine e dettagliata, capace di scontornare accuratamente piste sottili senza intaccare le aree conduttive contigue. Il bulino conico, a differenza delle frese cilindriche, permette una maggiore risoluzione nei dettagli, poiché la punta affilata incide in profondità solo nei punti strettamente necessari, riducendo la possibilità di cortocircuiti accidentali tra tracce vicine.

Il parametro di **profondità di lavorazione** è stato impostato a **-0.056 mm**, ovvero un'incisione molto superficiale ma sufficiente per rimuovere lo strato superficiale di rame e garantire l'isolamento elettrico tra le piste. Tale profondità è calibrata in base allo spessore standard del rame (tipicamente 35 µm per PCB FR4 standard), aggiungendo un margine utile a compensare eventuali irregolarità superficiali del materiale o imperfezioni nel piano della macchina CNC.
Il tool di isolamento ha quindi generato una serie di **percorsi concentrici** attorno a ciascuna traccia conduttiva, seguendo il perimetro esterno e calcolando automaticamente le traiettorie di taglio in base alla geometria delle piste e alle dimensioni dell'utensile. Questi percorsi sono stati esportati in formato **G-Code**, pronti per essere inviati alla macchina CNC per la realizzazione del PCB.
L'utilizzo di FlatCAM in combinazione con questo tipo di utensile permette quindi una produzione prototipale precisa, economica e flessibile, senza necessità di fotoincisione o bagni chimici. Inoltre, l'approccio CNC offre il vantaggio di poter apportare modifiche rapide al layout e rigenerare immediatamente i percorsi utensile aggiornati.

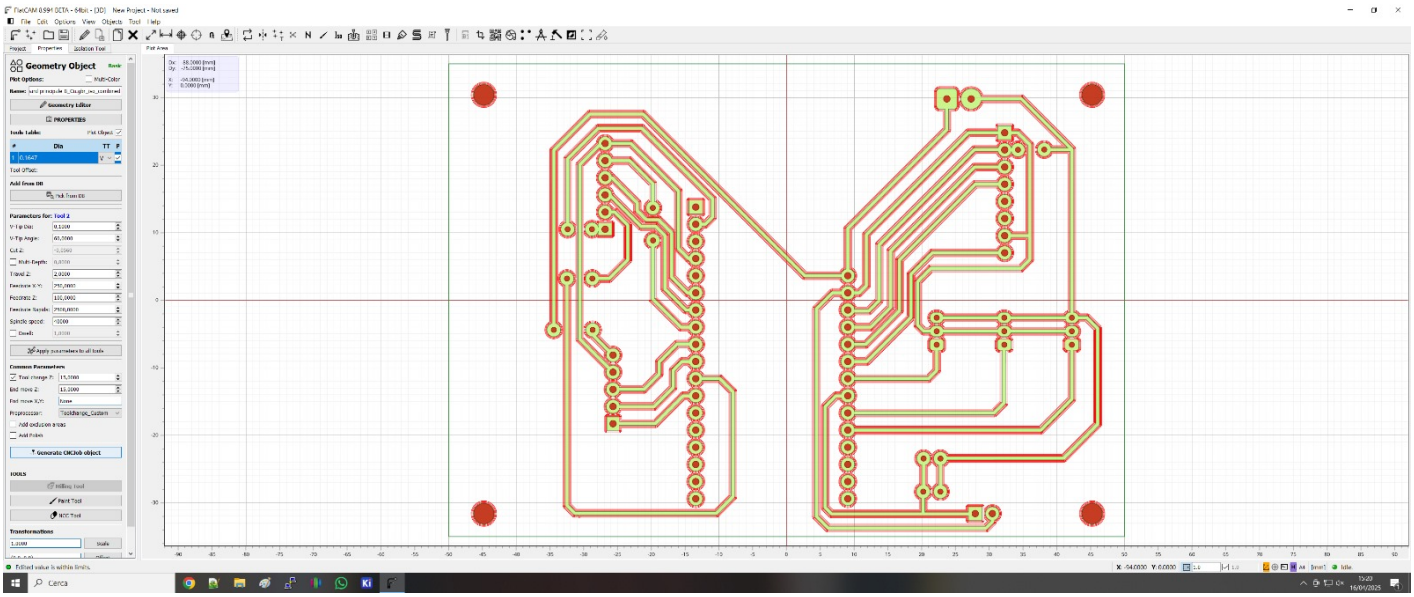


Figura 16: scontornatura piste

3. Generazione del percorso di foratura dei fori passanti

FlatCAM consente la generazione del percorso di foratura dei fori passanti partendo dai dati contenuti nel file Excellon, formato standard prodotto dalla maggior parte dei software CAD per PCB. Questo file descrive in maniera dettagliata la posizione, il diametro e la quantità dei fori necessari per realizzare la scheda. Questi fori

possono avere diverse funzioni: collegamenti elettrici tra i layer (vias), fori per i terminali dei componenti through-hole, oppure fori meccanici per il montaggio della scheda.
Durante l'importazione del file Excellon, FlatCAM consente all'utente di visualizzare graficamente tutti i fori suddivisi per diametro. Per ciascun gruppo di fori con lo stesso diametro, è possibile associare un utensile specifico, generalmente una punta elicoidale o un micro trapano, indicando le sue caratteristiche tecniche.

L'utente può quindi configurare in dettaglio i parametri di lavorazione. Tra questi:

- **La profondità di foratura**, che di norma deve superare leggermente lo spessore del PCB per garantire una foratura completa e pulita.
- **Il passo di discesa (step-down)**, ovvero la profondità massima per ogni passaggio della punta, utile soprattutto nei casi di materiali duri o per evitare sforzi eccessivi sull'utensile.
- **Il feedrate**, cioè la velocità di avanzamento dell'utensile nella Z, parametro fondamentale per bilanciare velocità ed efficienza.
- **L'altezza di sicurezza**, che rappresenta la distanza verticale alla quale l'utensile si solleva tra due fori per evitare collisioni con elementi già lavorati o fissaggi meccanici.

Una volta impostati tutti i parametri, FlatCAM elabora automaticamente il **G-code** per la lavorazione, ottimizzando l'ordine di esecuzione dei fori. L'obiettivo è minimizzare gli spostamenti a vuoto dell'utensile e ridurre i tempi ciclo, mantenendo una traiettoria logica ed efficiente. Questo codice macchina potrà poi essere inviato a una CNC compatibile, garantendo una foratura precisa, affidabile e in linea con le specifiche progettuali.
In sintesi, FlatCAM utilizza i dati contenuti nel file Excellon per individuare i fori da realizzare sul PCB, come quelli destinati alle connessioni tra layer (vias), ai pin dei componenti through-hole o a fori di montaggio. Per ogni diametro identificato, l'utente può assegnare un utensile specifico e configurare parametri come la profondità di foratura, il passo di discesa (step-down), il feedrate e l'altezza di sicurezza. Il software genera quindi il G-code corrispondente, ottimizzando l'ordine dei fori per ridurre gli spostamenti inutili della punta e migliorare l'efficienza della lavorazione.

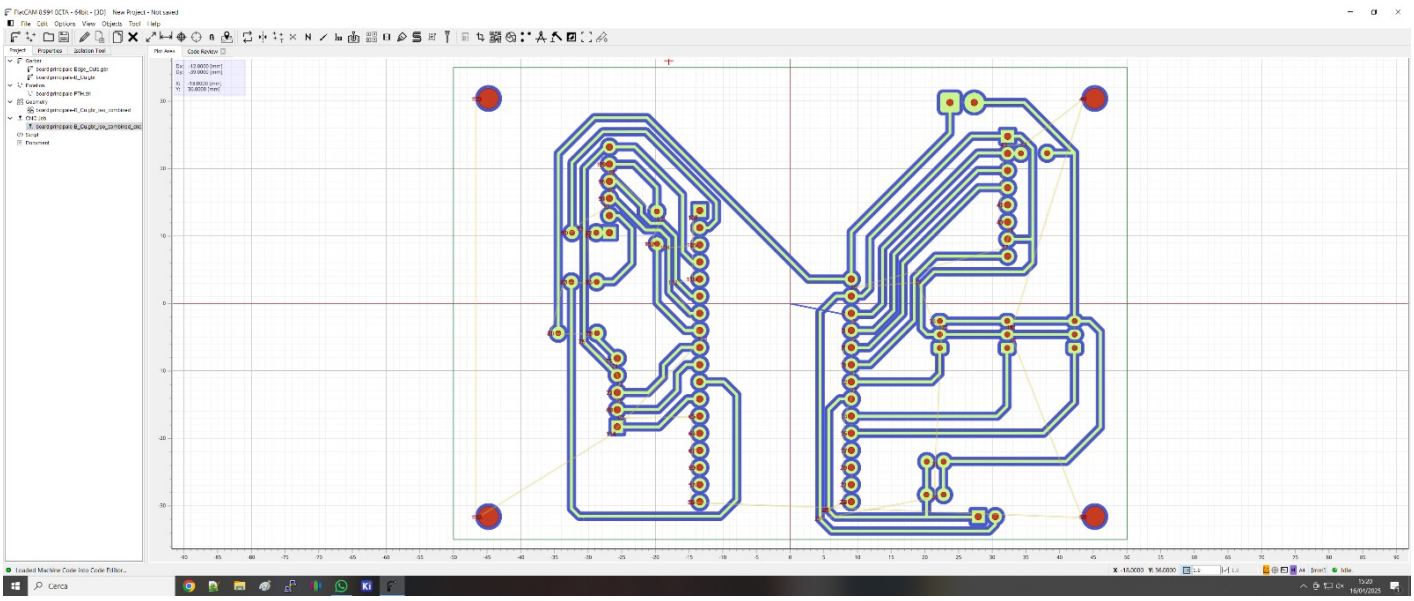


Figura 17: visualizzazione del percorso macchina

4. Generazione del percorso di taglio del profilo del PCB

Per delimitare e separare fisicamente il PCB dal materiale grezzo, FlatCAM elabora il contorno della scheda, solitamente definito nel layer “Edge Cuts”. L’utente seleziona il layer corretto e specifica l’utensile per il taglio, la profondità totale e le passate necessarie per completare la sagomatura. È possibile inserire dei piccoli ponticelli (tabs) che mantengono il PCB ancorato alla basetta, evitando movimenti indesiderati nella fase finale della fresatura. FlatCAM genera il relativo G-code, considerando la compensazione dell’utensile rispetto al bordo effettivo.

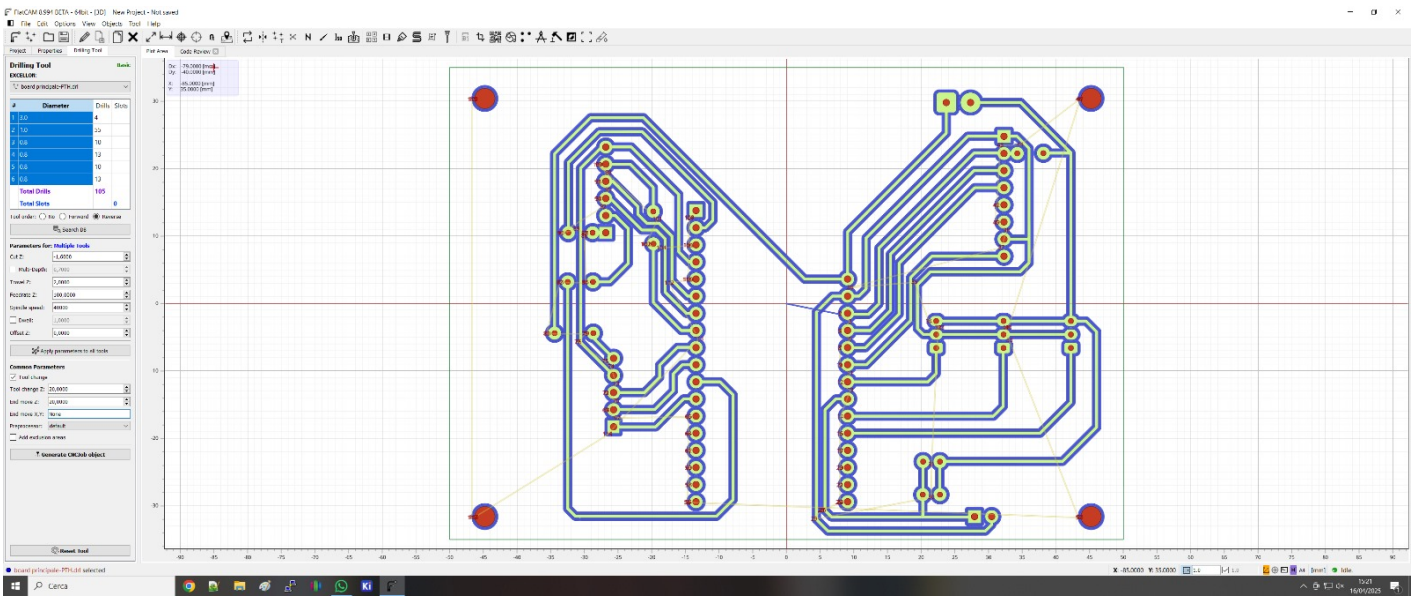


Figura 18: taglio della PCB

Durante la preparazione dei file di lavorazione CNC con **FlatCAM**, una fase fondamentale è rappresentata dalla **programmazione delle operazioni di foratura**, necessarie per la realizzazione dei **via**, dei **fori passanti** per componenti THT (Through-Hole Technology) e per i fori di montaggio meccanico. In questa fase, l'obiettivo è definire con precisione il tipo di utensile da utilizzare, la profondità di perforazione, e la generazione del codice macchina (G-Code) ottimizzato per la lavorazione su PCB.

Nel caso specifico, si è scelto di utilizzare un utensile da foratura con diametro compatibile con i fori previsti nel layout PCB. Il diametro della punta viene selezionato tenendo conto sia della **dimensione dei pad** nel file Gerber sia della **tipologia di componente** che verrà inserito successivamente. Ad esempio, per fori destinati a pin header da 1 mm, si può usare una punta da 0.8 mm o 0.9 mm per garantire un montaggio preciso.

La profondità di lavorazione è stata impostata a **-1.6 mm**, valore scelto per garantire una **perforazione completa** della scheda. Considerando che i PCB FR4 standard hanno uno spessore che può variare da 1.5 mm a 1.6 mm, questa impostazione assicura che la punta attraversi completamente il substrato senza eccessivo superamento che potrebbe danneggiare il piano di appoggio della CNC o compromettere l'integrità della punta. Tale profondità è particolarmente importante in quanto deve essere **coerente con lo spessore reale del PCB**, evitando che la punta fori parzialmente o che eserciti una pressione eccessiva che possa causare delaminazioni.

FlatCAM consente di caricare un file **Excellon** (standard per la foratura nei PCB), che contiene le coordinate e i diametri dei fori. Una volta importato, il software permette di associare a ogni diametro uno specifico

utensile, impostandone parametri come **profondità di taglio (cut Z)**, **profondità di passata (depth per pass)**, **ritrazione dell'utensile (travel Z)** e **velocità di avanzamento**.
Una volta configurati i parametri desiderati, FlatCAM genera i percorsi utensile necessari per ogni punto di foratura e li esporta in formato **G-Code**, pronti per essere letti da una macchina CNC. In fase operativa, l'utensile viene quindi abbassato con precisione in ogni punto specificato, fino alla profondità impostata di **-1.6 mm**, realizzando i fori in modo rapido, preciso e ripetibile.

Questo processo permette una **realizzazione completamente automatizzata** della foratura del PCB, con elevata accuratezza di posizionamento e senza necessità di processi manuali o meccanici successivi. Inoltre, la possibilità di modificare facilmente parametri come il tipo di punta o la profondità consente grande flessibilità nel passaggio da un prototipo a un altro, adattandosi anche a variazioni di spessore del materiale o del design.

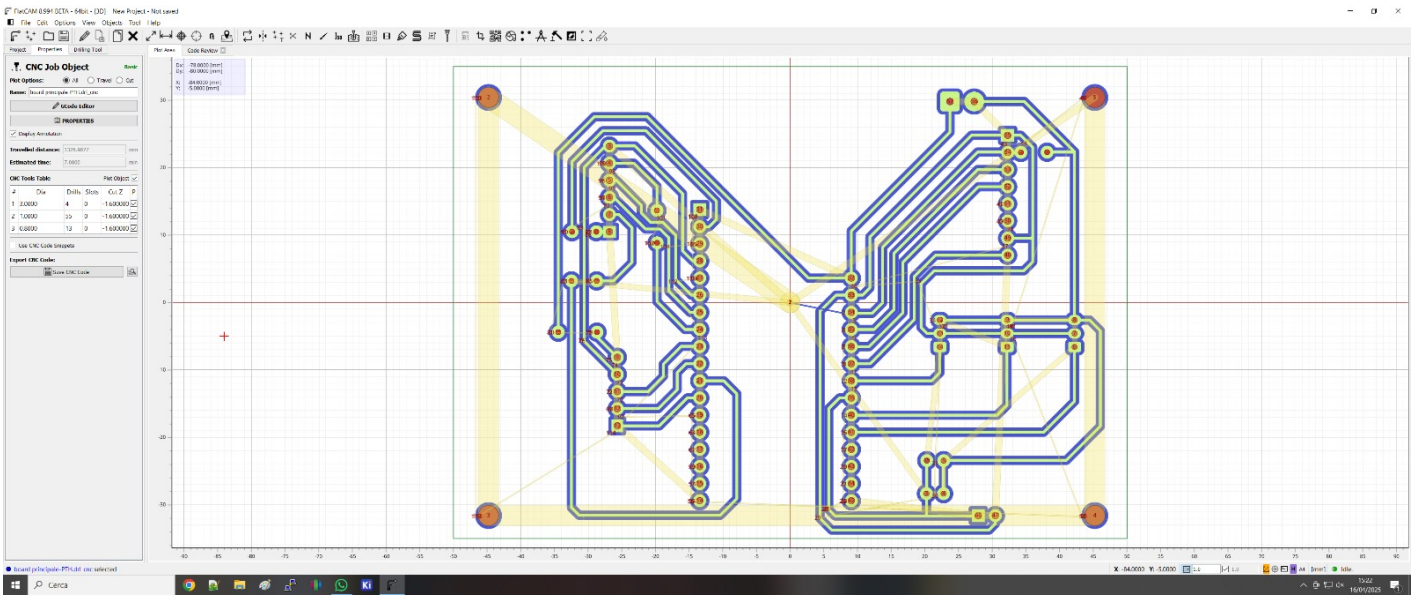


Figura 19:percorso macchina per la Foratura

5. Esportazione dei file G-code per CNC

Una volta definiti tutti i percorsi utensile – isolamento, foratura e contorno – FlatCAM consente l'esportazione dei file in formato G-code, compatibile con la maggior parte dei controllori CNC (GRBL, LinuxCNC, Mach3, ecc.). L'utente può configurare parametri globali come il punto zero della macchina, la velocità di rotazione del mandrino, le accelerazioni, e l'ordine di esecuzione delle operazioni. I file G-code possono essere salvati

separatamente per ogni tipo di lavorazione o unificati in un unico script, a seconda delle capacità della macchina CNC.

In dettaglio, dopo aver completato la definizione di tutti i percorsi utensile – incluse le operazioni di isolamento delle piste in rame, foratura dei pad e realizzazione del contorno esterno della scheda – FlatCAM consente di generare i corrispondenti file G-code necessari per pilotare una macchina CNC. Questa fase è fondamentale poiché traduce le istruzioni grafiche e geometriche in un linguaggio numerico standardizzato, compatibile con i più comuni controllori CNC, come GRBL, LinuxCNC, Mach3 o simili.
L'utente ha la possibilità di personalizzare un'ampia gamma di parametri globali prima dell'esportazione, per garantire una perfetta compatibilità tra il file generato e la macchina CNC utilizzata. Tra questi parametri si includono: la definizione del punto zero della macchina (origin point), che può essere posizionato manualmente o automaticamente in base all'allineamento del grezzo; la velocità di rotazione del mandrino (in RPM), che deve essere adeguata al tipo di utensile e al materiale del PCB; le velocità di avanzamento (feed rate) e le accelerazioni lungo i vari assi (X, Y e Z), per evitare sollecitazioni meccaniche non desiderate.
Inoltre, FlatCAM permette di stabilire l'ordine con cui le operazioni vengono eseguite, ad esempio foratura prima dell'isolamento o viceversa, consentendo una migliore ottimizzazione del processo in base alle caratteristiche specifiche della lavorazione e dell'hardware.
I file G-code risultanti possono essere esportati in modalità separata, cioè uno per ciascun tipo di lavorazione (es. un file per l'isolamento delle piste, uno per la foratura e uno per il contorno), oppure unificati in un unico script sequenziale. Questa seconda modalità è utile nei casi in cui la macchina CNC sia dotata di cambio utensile manuale o automatico, e consenta una lavorazione continua senza interruzioni. In entrambi i casi, l'esportazione avviene in formato .nc o .gcode, e può essere visualizzata in anteprima per una verifica finale prima dell'avvio effettivo della produzione.
Questa flessibilità rende FlatCAM uno strumento altamente efficiente per la preparazione dei file macchina, consentendo una personalizzazione elevata e una piena integrazione nel workflow di fabbricazione PCB tramite CNC.

6. Simulazione e verifica finale dei percorsi

Una volta completata la definizione di tutti i percorsi utensile, FlatCAM offre una funzionalità essenziale: la simulazione grafica dell'intera lavorazione. Questa funzione permette all'utente di esaminare in anteprima, direttamente all'interno dell'interfaccia del software, come si muoveranno gli utensili della macchina CNC in ogni fase del processo, dall'isolamento delle piste alla foratura, fino al contorno della scheda.
Attraverso la visualizzazione dei tracciati, è possibile controllare visivamente che i percorsi non presentino errori come intersezioni indesiderate, passaggi troppo ravvicinati ai bordi, o omissioni nella lavorazione.

Inoltre, FlatCAM consente di effettuare misurazioni precise su distanze critiche, spessori delle piste, distanze minime tra fori o margini, garantendo il rispetto delle tolleranze progettuali.
Durante la simulazione, l'utente può anche valutare i tempi stimati di lavorazione per ciascun percorso utensile, un'informazione utile per pianificare correttamente il ciclo produttivo, soprattutto se si lavora su lotti multipli o su materiali costosi.
Un altro aspetto cruciale di questa fase è la possibilità di individuare potenziali problemi meccanici o errori di configurazione prima che venga eseguito il G-code reale sulla macchina. Per esempio, si possono rilevare movimenti anomali, collisioni con il materiale o la tavola, profondità di taglio errate, oppure sequenze operative inefficienti. FlatCAM, in questo senso, agisce come un ambiente di debug visivo per il processo di fresatura.
Questa simulazione finale non è solo un controllo di sicurezza, ma rappresenta una fase strategica che contribuisce a ridurre gli sprechi di materiale, evitare rotture degli utensili o danneggiamenti alla CNC, aumentando significativamente l'affidabilità e la qualità del processo di produzione delle PCB.
In sintesi, prima di avviare la lavorazione effettiva, FlatCAM consente di visualizzare in anteprima i percorsi utensile direttamente sull'interfaccia grafica. Questo permette di eseguire una simulazione del movimento della CNC, di misurare distanze critiche, di stimare il tempo necessario e di identificare eventuali errori o collisioni potenziali. Questa fase è fondamentale per evitare sprechi di materiale o danneggiamenti all'utensile e alla macchina.

3. Realizzazione di Circuiti Stampati (PCB)

Il centro di lavoro usato per la realizzazione dei circuiti stampati su board è la Datron M10Pro. La Datron M10Pro è un **centro di lavoro a controllo numerico ad alta precisione** (CNC) progettata per applicazioni che richiedono lavorazioni **rapide, accurate e affidabili**. Grazie alla sua struttura rigida e al design ottimizzato per ridurre le vibrazioni, questa macchina garantisce risultati eccellenti in termini di qualità superficiale e tolleranze dimensionali. Dotata di un mandrino ad alta velocità, la **M10Pro è in grado di lavorare una vasta gamma di materiali, tra cui alluminio, plastica, e materiali compositi**. La sua tecnologia avanzata consente velocità di avanzamento elevate senza compromettere la precisione, rendendola ideale per la produzione di componenti di alta qualità nel settore aerospaziale, elettronico, automobilistico e nella micromeccanica. L'integrazione con un sistema di controllo intuitivo e un'interfaccia utente semplificata permette agli operatori di gestire la macchina in modo efficiente, ottimizzando i tempi di lavorazione. La Datron M10 Pro è una macchina potente e altamente precisa che offre rigidità, stabilità termica e accuratezza. **Permette lavorazioni ad alta velocità con mandrino HSK da 3 kW e 40.000** giri/min. Rappresenta l'unico sistema di lavorazione di progettazione tedesca con scale lineari, costruzione termicamente bilanciata e un rapporto tra ingombro **e area di lavorazione di 2:1 per lavori ad alta tolleranza e impegnativi.** Il sistema di misurazione lineare integrato con una risoluzione di 40 nm garantisce una precisione continua, rendendo l'M10 Pro ideale per lotti di diverse dimensioni, numeri piccoli, numeri grandi e materiali high-tech. Ecco di seguito alcune caratteristiche:

- **Scale lineari** – Per precisioni segnalate dai clienti DATRON nell'intervallo **di 4 – 5 micron. Portautensili HSK con runout <3 micron.**
- **Tavolo in granito massiccio** – Il **tavolo in granito massiccio è sostenuto da un telaio in acciaio fuso**. Questa rigidità assicura precisione e qualità dei pezzi.
- **Servoazionamenti digitali** - consentono di raggiungere velocità elevate fino a 30 metri al minuto e un'accelerazione di 5 metri al secondo^2.

Uno dei principali punti di forza della Datron M10Pro è la capacità di lavorare con **strategie di fresatura avanzate che permettono di ottenere superfici lisce e dettagli estremamente precisi.** La sua compatibilità con software CAD/CAM avanzati assicura una perfetta integrazione nei flussi di lavoro digitali moderni, facilitando la programmazione e la gestione delle lavorazioni. Affidabile, veloce e versatile, la Datron M10Pro rappresenta una soluzione ideale per chi necessita di una fresatrice CNC di alta precisione in grado di coniugare prestazioni eccellenti e facilità d'uso.

In basso viene riportata una foto.

Figura 20: Datron M10Pro

La tabella in basso riporta le specifiche tecniche della suddetta CNC.

Tabella 1: specifiche tecniche Datron M10Pro

DATRON M10 Pro Specifications	
Travel	1,020 x 830 x 240 mm (40" x 33" x 9.5") (X, Y, Z)
Spindle	Up to 40,000 RPM spindle
Feed	Up to 30 m/min (1,181 in/min)
Tool Changer	Up to 24 stations automatic tool changer
Control System	DATRON next
Working Area	1,000 x 700 x 200 mm (39" x 27.5" x 7.5")
Rotary axis	Optional 4th axis and 5th axis
Footprint	1,990 x 2,080 x 2,000 mm (78" x 82" x 79") (W x D x H)
Weight	approx. 2 t (3,600 lbs.)

La **Datron M10Pro** è un centro di lavoro CNC a controllo numerico di alta precisione progettato per applicazioni che richiedono lavorazioni rapide, accurate e con tolleranze estremamente rigide. Quando utilizzata per la produzione di circuiti stampati (PCB), questa macchina permette di realizzare prototipi e piccole serie con elevata precisione e qualità delle superfici. La macchina è equipaggiata con **scale lineari**, **mandrino HSK da 3 kW**, e una capacità di avanzamento fino a **30 metri al minuto**, il che consente di ottenere circuiti con tolleranze molto basse.

La procedura per la realizzazione di PCB sulla Datron M10Pro si articola nei seguenti passaggi:

3.1. Preparazione del progetto PCB

Come descritto in precedenza il primo passo consiste nella progettazione elettronica del circuito stampato, che si effettua con un software CAD elettronico specifico per circuiti (ad esempio **KiCAD**, **Altium Designer**, o **Eagle**). Il progetto deve essere esportato nei formati standard come **Gerber** per le piste di rame e **Excellon** per la foratura.

- **File Gerber**: contengono le informazioni sulle tracce, i bordi del PCB e le aree di rame da fresare.
- **File Excellon**: contengono le informazioni sulle forature (fori passanti, vias, pin) necessari per il montaggio dei componenti elettronici.

Dopo l'esportazione dei file, questi vengono importati nel **software CAM** compatibile con la Datron M10Pro, che trasforma i dati progettuali in **percorsi utensile** (toolpath) pronti per il controllo CNC.

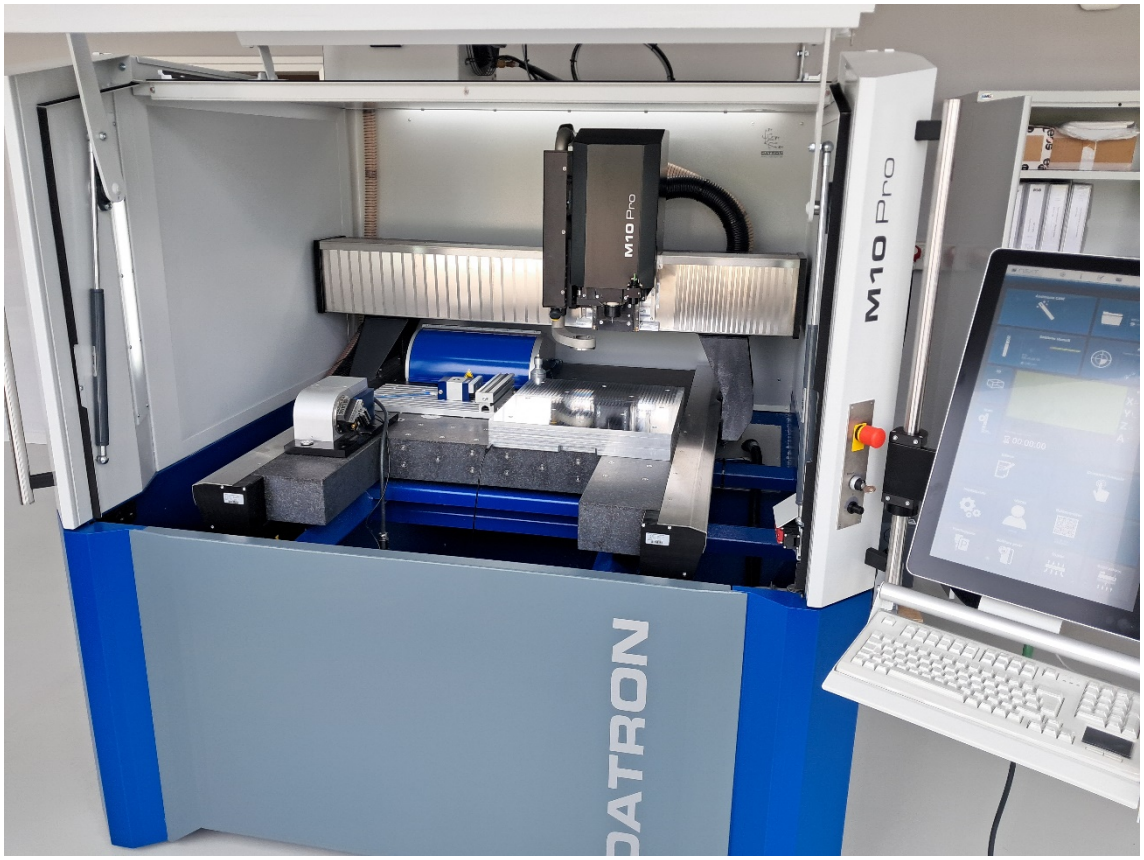


Figura 21: Allestimento della CNC

3.2. Preparazione della macchina e del materiale

- **Materiale di base**: Si seleziona un materiale per PCB, generalmente un laminato **FR4** (fibra di vetro impregnato di resina epossidica) con un rivestimento di rame. Altri materiali compositi, come materiali a base di poliestere o Teflon, possono essere utilizzati in applicazioni specifiche.
- **Posizionamento e fissaggio**: Il laminato viene fissato al piano di lavoro della macchina utilizzando il sistema sottovuoto della **Datron M10Pro**. Questo sistema assicura che il materiale sia completamente stabile durante la lavorazione, evitando vibrazioni che potrebbero compromettere la precisione.
    - Il piano di lavoro della M10Pro è in **granito massiccio**, che offre una rigidità eccezionale e riduce al minimo le deformazioni termiche, garantendo la **precisione dimensionale** del PCB durante tutta la lavorazione.

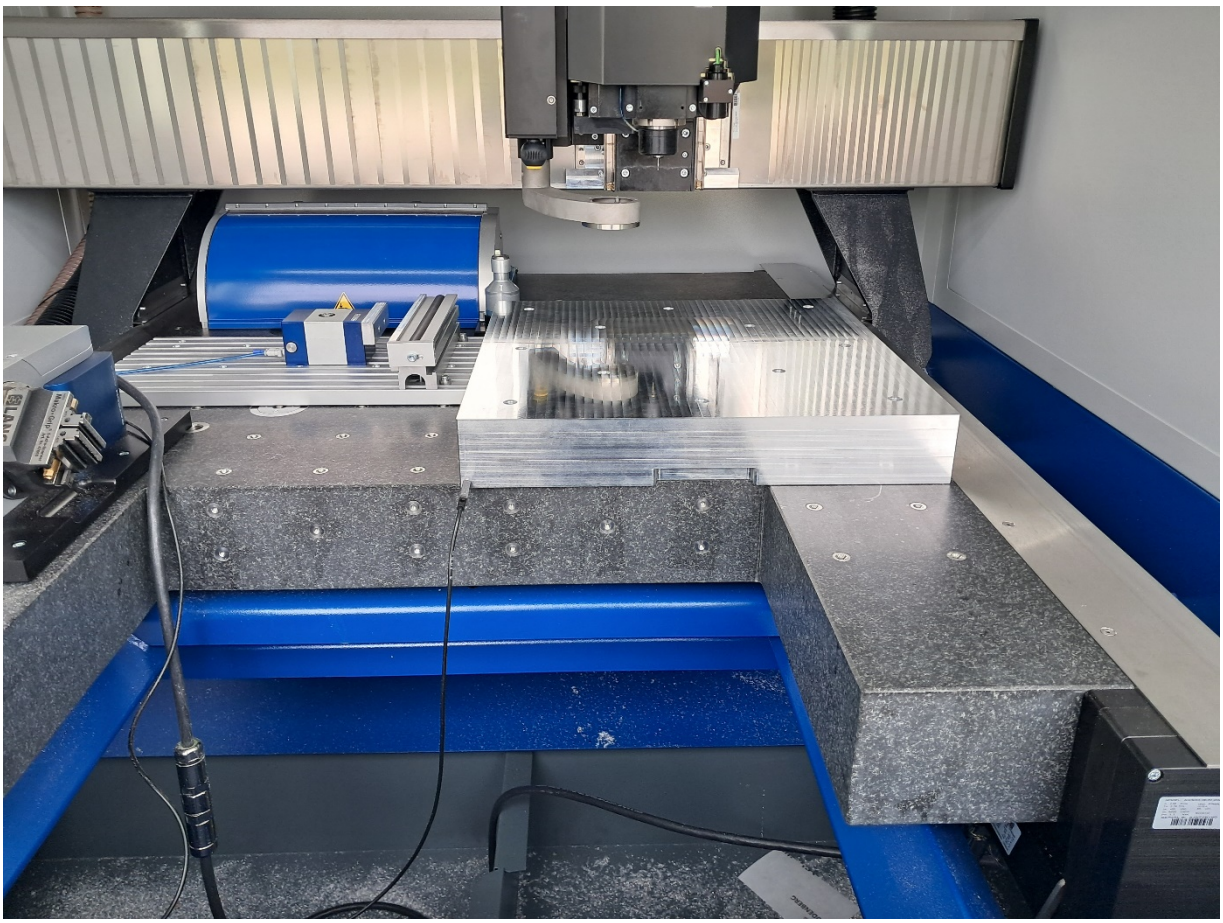


Figura 22: Pulizia della CNC

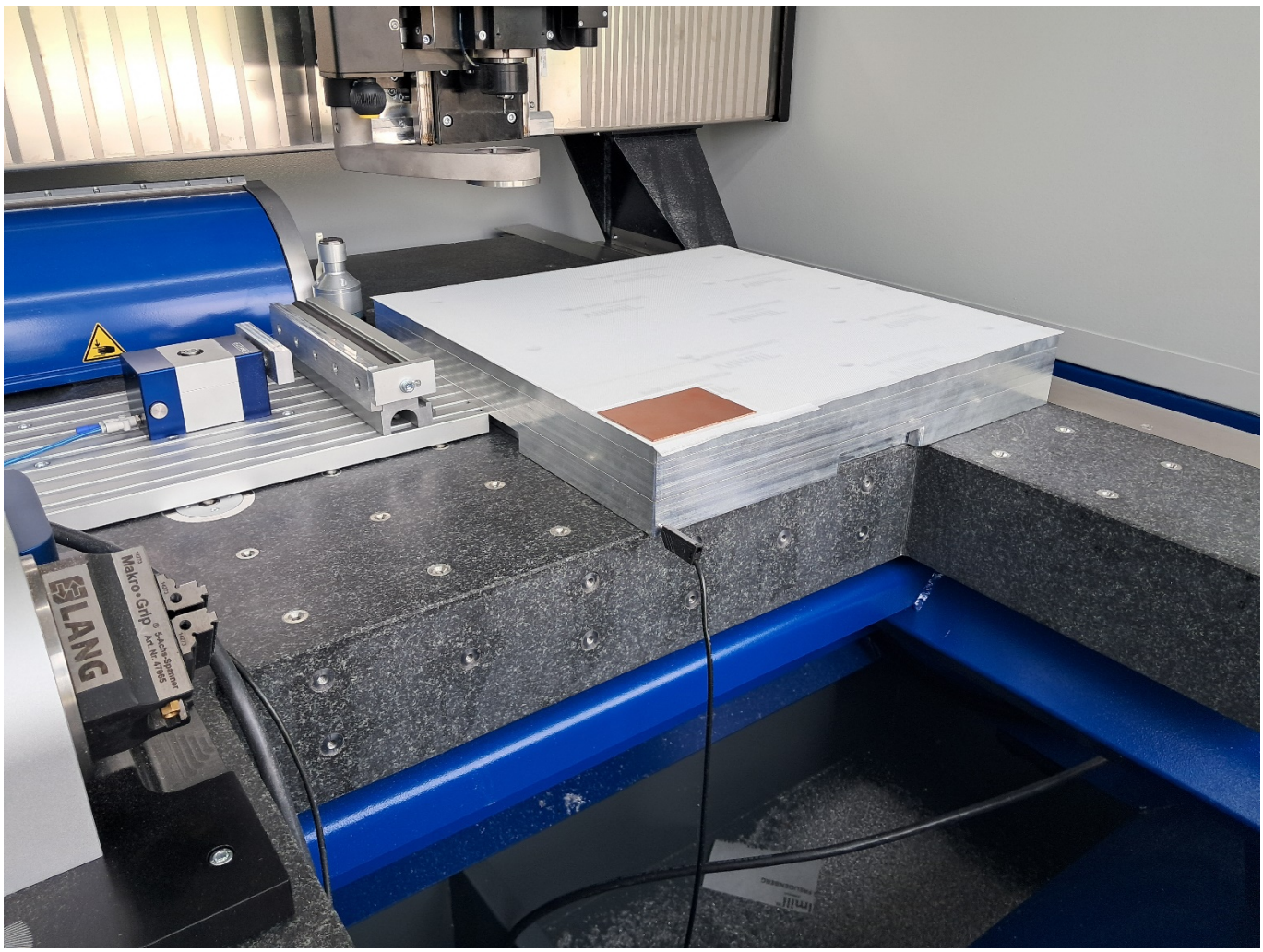


Figura 23: Preparazione della PCB in macchina CNC

3.3. Selezione e configurazione degli utensili

Gli utensili utilizzati per la lavorazione del PCB sono scelti in base alla geometria del circuito e alla tipologia di lavorazione:

- **Fresa per incisione rame**: le frese a punta sottile, generalmente da **0,2 mm** a **0,5 mm**, vengono utilizzate per **incidere** le piste di rame, separando il materiale di rame dal resto della superficie.

- **Utensili per foratura**: vanno da **0,3 mm a 1,5 mm** di diametro, in base alle dimensioni dei fori richiesti nel progetto. I fori sono necessari per **vias** e forature per i componenti elettronici.
- **Utensile di sagomatura**: un utensile adatto per rifinire i bordi del PCB e creare eventuali tasche o scanalature.

La **Datron M10Pro** è equipaggiata con un **portautensili HSK** che garantisce una stabilità eccezionale e una **precisione di centraggio <3 micron** anche durante operazioni ad alta velocità.

3.4. Configurazione dei parametri di lavorazione

La configurazione dei parametri di lavorazione è fondamentale per ottenere un prodotto finale di alta qualità. I principali parametri da considerare sono:

- **Velocità del mandrino**: La M10Pro è dotata di un mandrino da **3 kW** che può raggiungere fino a **40.000 giri/min**. La velocità ideale dipende dal materiale da lavorare e dalle dimensioni dell'utensile. In generale, per la lavorazione del rame, si sceglieranno velocità elevate per garantire un taglio rapido e preciso.
- **Avanzamento e accelerazione**: L'M10Pro supporta velocità di avanzamento fino a **30 metri al minuto** e accelerazioni fino a **5 metri al secondo quadrato**. Questi valori sono fondamentali per ridurre i tempi di lavorazione senza compromettere la qualità.
- **Profondità di incisione**: Le profondità per le piste di rame sono calcolate in base al materiale e alla geometria del progetto. La fresatura deve essere tale da rimuovere completamente il rame in eccesso, ma senza danneggiare il supporto.

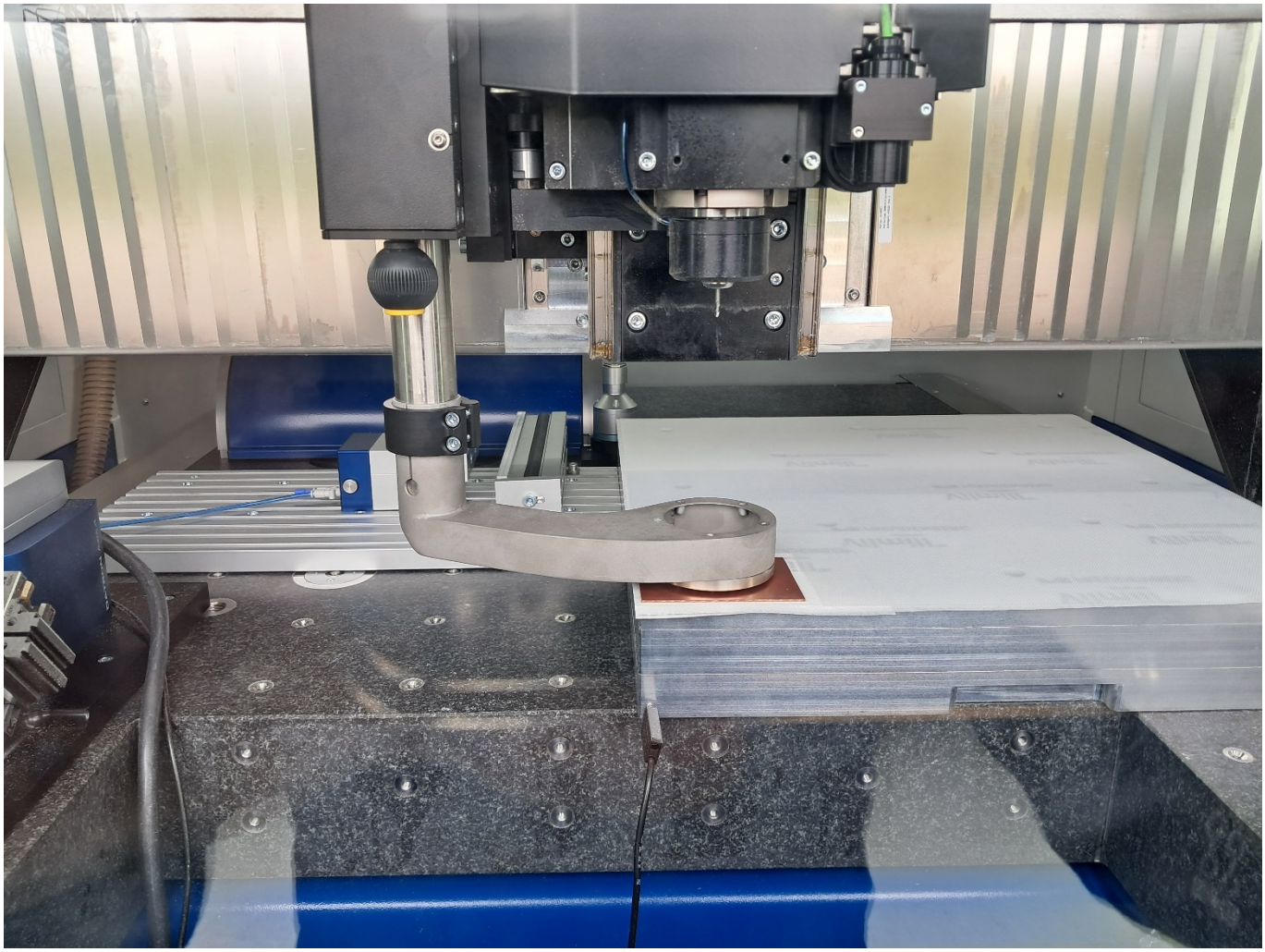


Figura 24: Configurazione dei parametri per la lavorazione

Figura 25: Parametri degli utensili per la lavorazione – visualizzazione su monitor della CNC

3.5. Fase di fresatura

La fresatura avviene in diverse fasi:

1. **Rimozione del materiale in eccesso**: Inizialmente, vengono fresati i bordi e le aree di rame non necessarie.
2. **Incisione delle tracce di rame**: Le piste vengono isolate fresando il rame in eccesso attorno alle tracce progettate. Grazie alle **scale lineari** con **risoluzione 40 nm**, la macchina è in grado di mantenere tolleranze estremamente basse (nell'ordine di pochi micrometri) garantendo un'incisione precisa.
3. Foratura: I fori passanti per vias e componenti elettronici vengono realizzati utilizzando una fresa apposita. Ogni foro viene fresato a una profondità precisa per garantire la continuità elettrica e la giusta adattabilità dei componenti.
4. Finitura e bordatura: Infine, la fresatrice esegue un'operazione di bordatura per rifinire i contorni esterni del PCB.

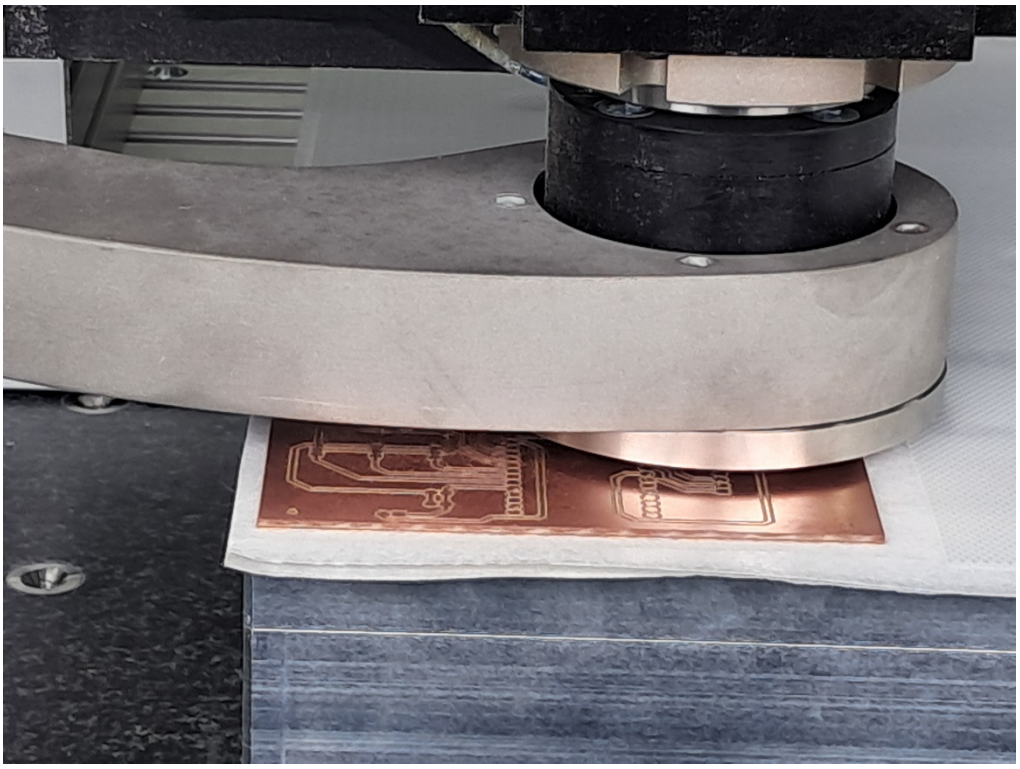


Figura 26: Lavorazione in atto

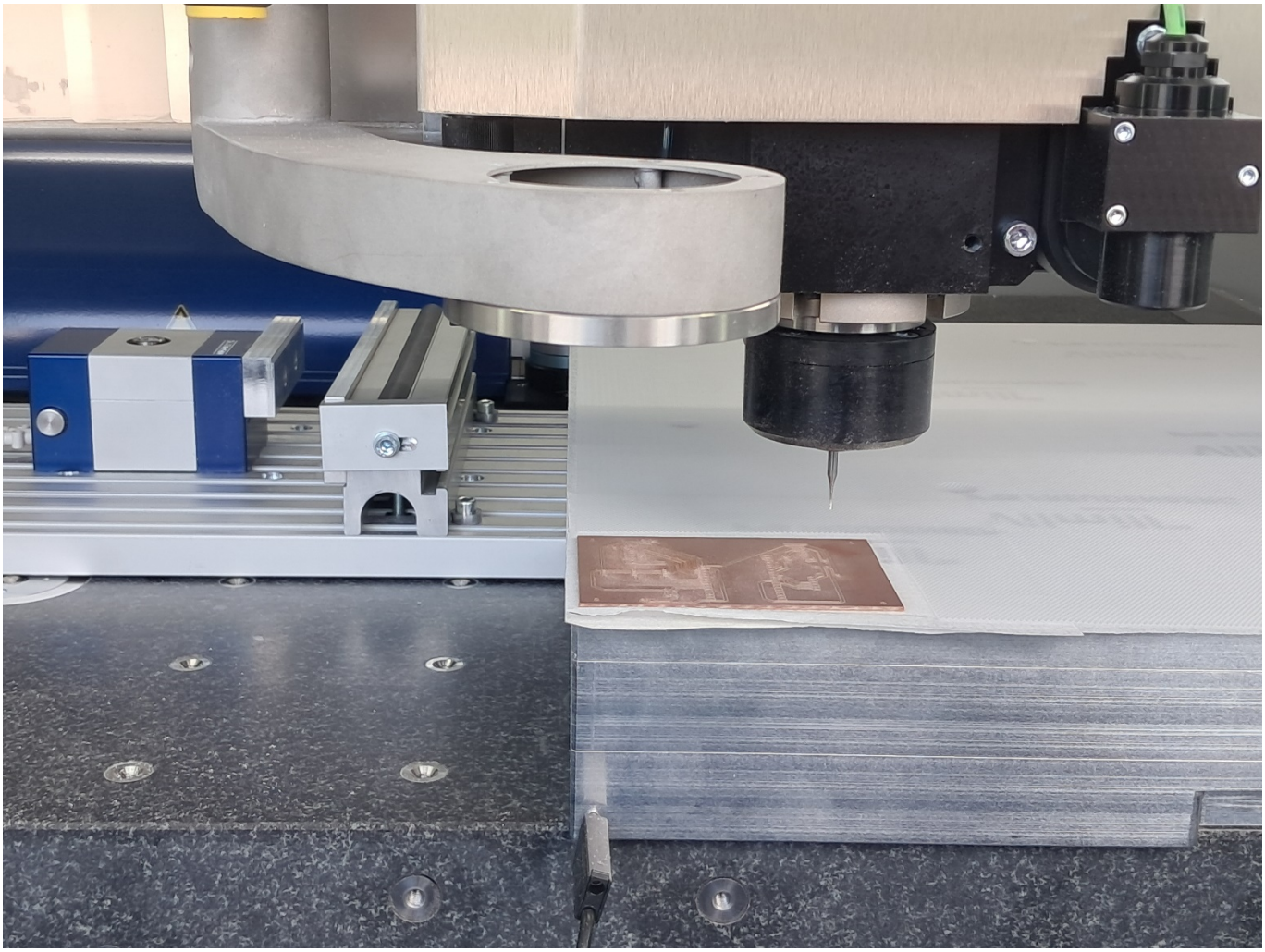


Figura 27: fine della lavorazione della CNC

3.6. Controllo di qualità e validazione

- **Controllo dimensionale**: La macchina, grazie alla sua precisione micrometrica, è in grado di garantire che tutte le piste di rame e i fori siano conformi al progetto. Si effettua un controllo visivo e dimensionale utilizzando strumenti di misura come **micrometri** e **calibri digitali**.
- **Test di continuità elettrica**: Dopo la fresatura, il PCB viene testato per verificare la continuità elettrica delle tracce, assicurandosi che non ci siano cortocircuiti tra le piste.

- **Ispezione visiva**: Il circuito viene ispezionato visivamente per assicurarsi che le tracce siano ben definite e che non ci siano difetti come sovrapposizioni di rame.

3.7. Post-lavorazione e completamento

- **Rimozione delle pellicole protettive**: Se sono state applicate pellicole protettive sul PCB per evitare danni durante la fresatura, queste vengono rimosse.
- **Test di funzionalità**: Se il PCB è destinato a un prototipo funzionale, si monta il circuito e si testano le sue prestazioni elettroniche.

I vantaggi della Datron M10Pro per la realizzazione di PCB sono i seguenti:

- **Alta precisione**: Grazie alle **scale lineari** e alla **rigidità strutturale**, la M10Pro è in grado di eseguire lavorazioni con tolleranze molto basse (fino a 4-5 micron).
- **Velocità di lavorazione**: Le alte velocità di avanzamento e di rotazione del mandrino consentono di ridurre i tempi di produzione senza compromettere la qualità.
- **Flessibilità**: La macchina è ideale per prototipi rapidi, piccole e medie serie, e per la lavorazione di materiali ad alte prestazioni.
- **Integrazione software avanzata**: La compatibilità con i software CAD/CAM più diffusi facilita l'importazione dei file e la creazione dei percorsi utensile.

Questa procedura dettagliata evidenzia come la Datron M10Pro possa essere utilizzata efficacemente per la realizzazione di circuiti stampati di alta qualità, con un processo che coniuga precisione, velocità e affidabilità.

4. Implementazione fisica delle board/schede

L'immagine seguente riporta la scheda realizzata. Le altre due schede o board sono realizzate utilizzando gli stessi step descritti in precedenza. In dettaglio, la **Board 1 – Modulo di Acquisizione Elettrica e Rilevamento Fiamma** costituisce il nucleo base del sistema, dedicato prevalentemente all'acquisizione di grandezze elettriche e alla rilevazione di eventi anomali. Essa include:

- **Modulo ESP32**: microcontrollore principale responsabile della gestione della comunicazione, dell'elaborazione dei dati e dell'interfacciamento con gli altri moduli.
- **Modulo RTC (Real-Time Clock)**: utilizzato per la sincronizzazione temporale delle misure e per la tracciabilità degli eventi.
- **Modulo SDCard**: per la memorizzazione locale dei dati raccolti, utile per backup o operazioni offline.
- **Sensore di fiamma**: modulo per il rilevamento di radiazioni infrarosse generate da fiamme o scintille.
- **Tre moduli sensore di corrente**: ciascuno dedicato al monitoraggio di linee o dispositivi diversi, con la possibilità di misurare assorbimenti anomali o consumi elettrici.
- **Modulo accelerometro**: integrato per la rilevazione di vibrazioni o movimenti, utile in contesti di diagnostica predittiva o rilevamento incidenti.

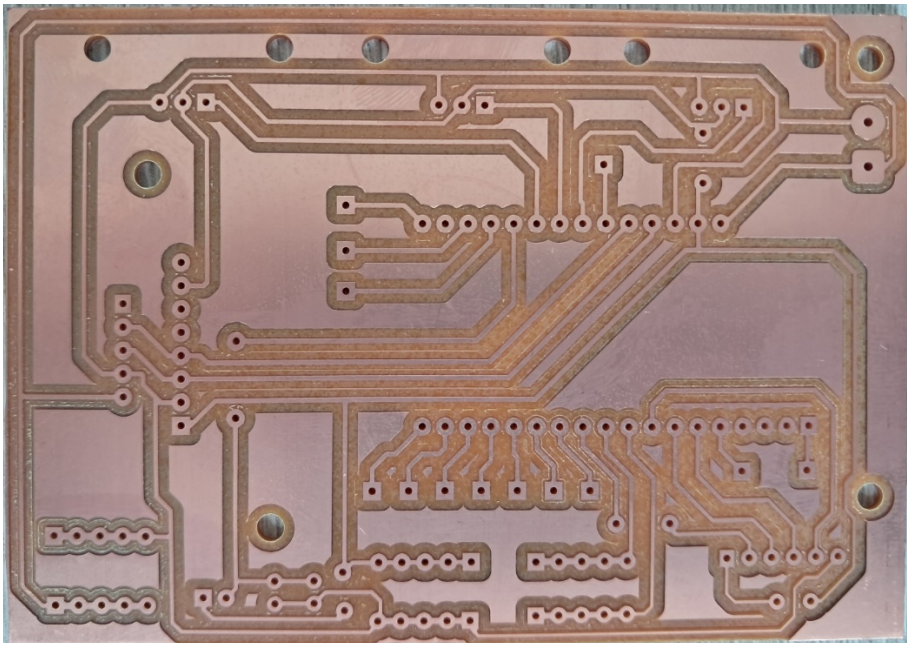


Figura 28: Board 1 realizzata

Di seguito si riportano le immagini della board 2 e della board 3 entrambe realizzate con la stessa strategia precedentemente descritta.

Sulla **Board 2 – Unità Ambientale Multisensore** saranno presenti i seguenti componenti

- **Modulo ESP32**: come per la board 1, svolge il ruolo di unità di controllo.
- **Modulo RTC** e **Modulo SDCard**: replicano le funzioni già presenti nella board 1 per garantire indipendenza funzionale e sincronizzazione.
- **Sensori ambientali**:
    - Ossigeno (O_2)
    - Ozono (O_3)
    - Monossido di Carbonio (CO)
    - Particolato fine (PM2.5)
    - Anidride Carbonica (CO_2)
    - Temperatura e umidità relativa

Mentre la **Board 3 – Estensione per Gas e Composti Volatili** fisicamente connessa alla board 2 include:

- Sensore di ammoniaca (NH_3)
- Sensore VOC (Composti Organici Volatili)
- Sensore di metano (CH_4)
- Sensore di biossido di azoto (NO_2)

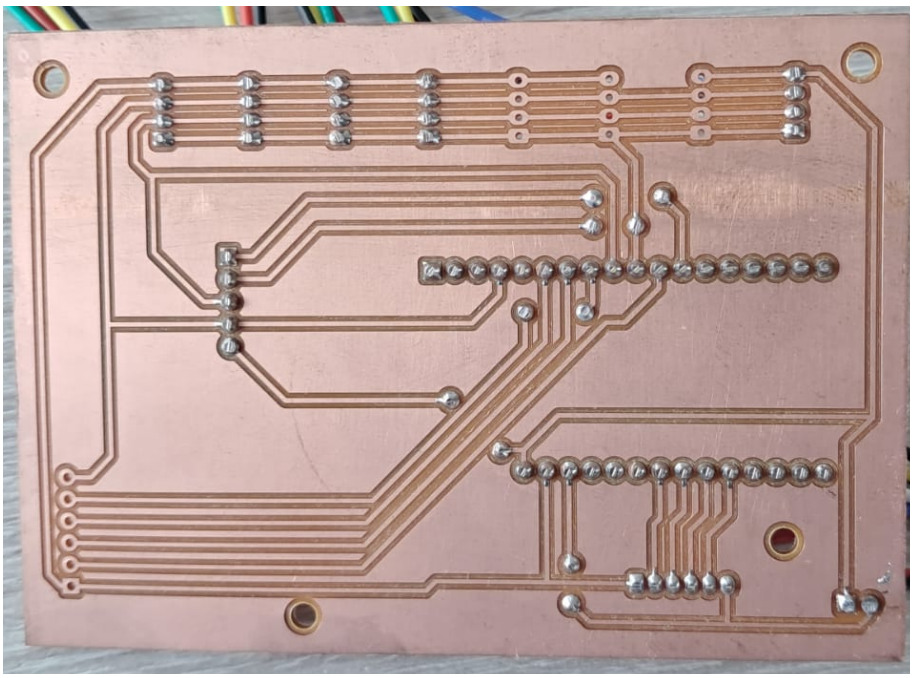


Figura 29: Board 2 realizzata

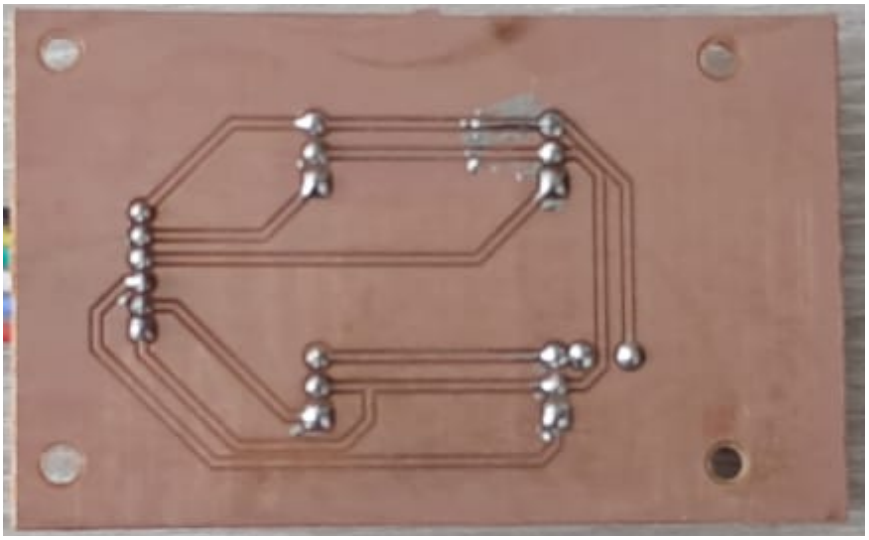


Figura 30: Board 3 realizzata

5. Integrazione tra PCB e componentistica

Di seguito si riporta la descrizione sulla integrazione dei componenti elettronici precedentemente selezionati su ciascuna board realizzata con il sistema CNC.
L'integrazione di componenti elettronici ed elettrici su una scheda realizzata tramite CNC (Computer Numerical Control) richiede una pianificazione precisa e un attento processo di assemblaggio per garantire il corretto funzionamento e l'affidabilità del sistema finale. Ogni componente deve essere posizionato e fissato sulla PCB (Printed Circuit Board) in modo che l'intero sistema funzioni in modo ottimale, con un'attenzione particolare a considerazioni funzionali, elettriche e termiche.
Nel caso della **board 1**, la scheda ospita una serie di componenti integrati che svolgono ruoli cruciali nella gestione dei dati e nell'interfacciamento con l'ambiente esterno. Il cuore di questa architettura è il **Modulo ESP32**, un potente microcontrollore che funge da cervello del sistema. Grazie alle sue capacità di connettività Wi-Fi e Bluetooth, l'ESP32 gestisce la comunicazione tra i vari moduli e interagisce con dispositivi esterni. La sua potenza di elaborazione consente di eseguire calcoli complessi e di gestire l'elaborazione dei dati in tempo reale, un aspetto fondamentale in applicazioni dove la reattività e la velocità sono essenziali.
Accanto al microcontrollore, il **Modulo RTC (Real-Time Clock)** è fondamentale per la sincronizzazione temporale delle operazioni. Questo modulo consente di associare ogni dato raccolto a un preciso timestamp, che diventa essenziale per la tracciabilità degli eventi, per esempio quando si analizzano sequenze di misure o si monitorano eventi in un contesto di controllo o sicurezza. La precisione di un RTC assicura che le operazioni siano temporizzate correttamente, anche in caso di interruzione di alimentazione, grazie alla sua capacità di mantenere il tempo senza dover essere ricalibrato.
Per quanto riguarda la **memorizzazione dei dati**, il **Modulo SDCard** offre una soluzione robusta e flessibile. Permette di archiviare i dati localmente sulla scheda, consentendo operazioni di backup o di salvataggio di dati durante il funzionamento offline. La possibilità di leggere e scrivere su una SDCard amplia notevolmente la capacità di gestione dei dati, e questo modulo è fondamentale per i sistemi che devono conservare grandi quantità di dati per analisi successive o per la registrazione continua di misurazioni.
Il **Sensore di fiamma** integrato sulla PCB è un componente di rilevamento di radiazioni infrarosse emesse dalle fiamme o scintille. Questo modulo è essenziale per la sicurezza del sistema, consentendo di monitorare la presenza di incendi o fiamme in ambienti critici, come impianti industriali, laboratori o aree a rischio. L'accuratezza nel rilevamento delle radiazioni infrarosse consente al sistema di attivare allarmi tempestivi o di eseguire altre azioni correttive in risposta a situazioni di emergenza.
I **Tre moduli sensore di corrente** sono dedicati al monitoraggio delle linee elettriche o dei dispositivi di alimentazione. Questi sensori misurano l'assorbimento di corrente e possono rilevare consumi anomali, come picchi di corrente o guasti nei dispositivi, che potrebbero indicare problemi come cortocircuiti o sovraccarichi.

L'integrazione di più sensori consente di monitorare in modo preciso e indipendente diverse aree o linee, migliorando la gestione energetica e prevenendo possibili danni agli impianti.
Infine, l'**accelerometro** montato sulla scheda ha la funzione di rilevare vibrazioni o movimenti, che sono fondamentali in molte applicazioni industriali e di diagnostica predittiva. Il sensore è utilizzato per monitorare il comportamento dinamico di macchinari o impianti, con l'obiettivo di individuare eventuali anomalie che potrebbero indicare il malfunzionamento di componenti meccanici o la possibilità di guasti imminenti. Inoltre, l'accelerometro è utile in applicazioni di rilevamento di incidenti, dove può fornire dati in tempo reale su eventuali eventi traumatici, come urti o cadute, particolarmente importante in contesti come il monitoraggio di sicurezza.
Il processo di integrazione di questi componenti sulla scheda realizzata con la CNC non si limita alla loro semplice saldatura sulla PCB, ma implica anche una pianificazione strategica dei loro posizionamenti. Ogni modulo deve essere montato in modo da ridurre al minimo le interferenze elettromagnetiche (EMI), ottimizzare i percorsi dei segnali e migliorare la dissipazione del calore, specialmente per i componenti che potrebbero generare un elevato dispendio termico, come l'ESP32 o i sensori di corrente. Inoltre, è fondamentale garantire che il layout permetta una facile connessione ai vari pin e interfacce, e che ogni modulo abbia il suo spazio per evitare conflitti meccanici o difficoltà durante l'assemblaggio.
L'utilizzo di una CNC per la realizzazione della scheda consente di ottenere un'elevata precisione nei fori e nei tagli, necessari per il corretto posizionamento dei componenti e per il flusso ottimale delle tracce di rame. Questo processo migliora la qualità e l'affidabilità della scheda, riducendo al minimo la possibilità di errori durante l'assemblaggio e garantendo che tutte le connessioni elettriche siano realizzate con alta precisione.

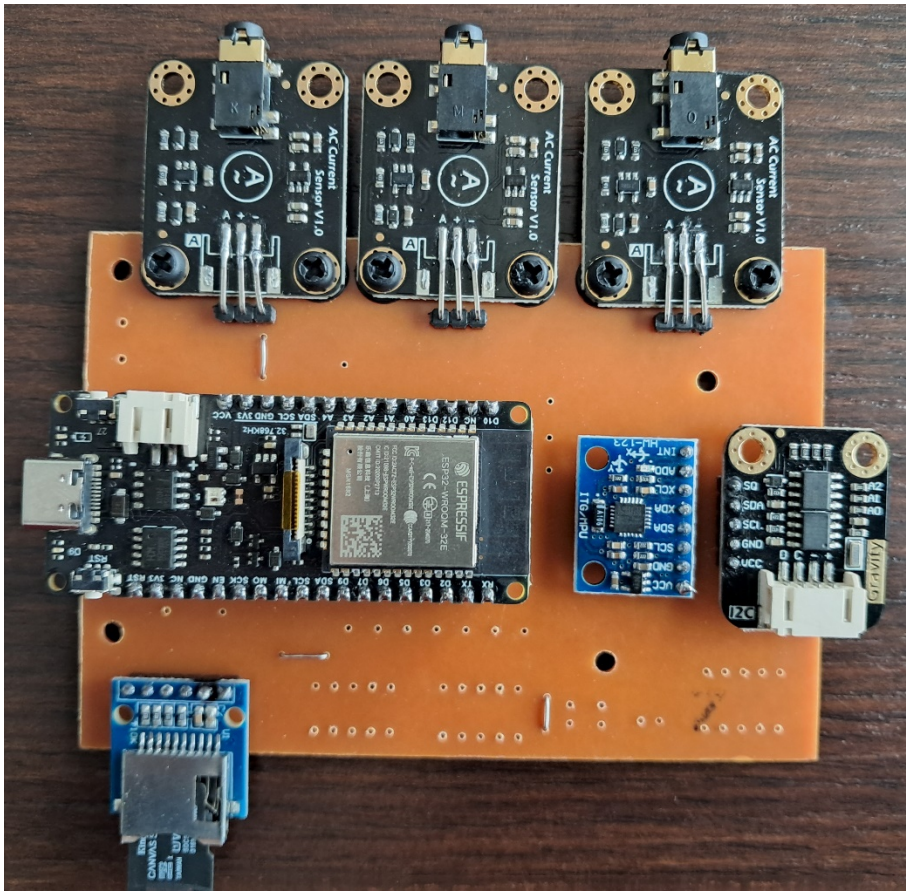


Figura 31: integrazione dei moduli sensori sulla board 1

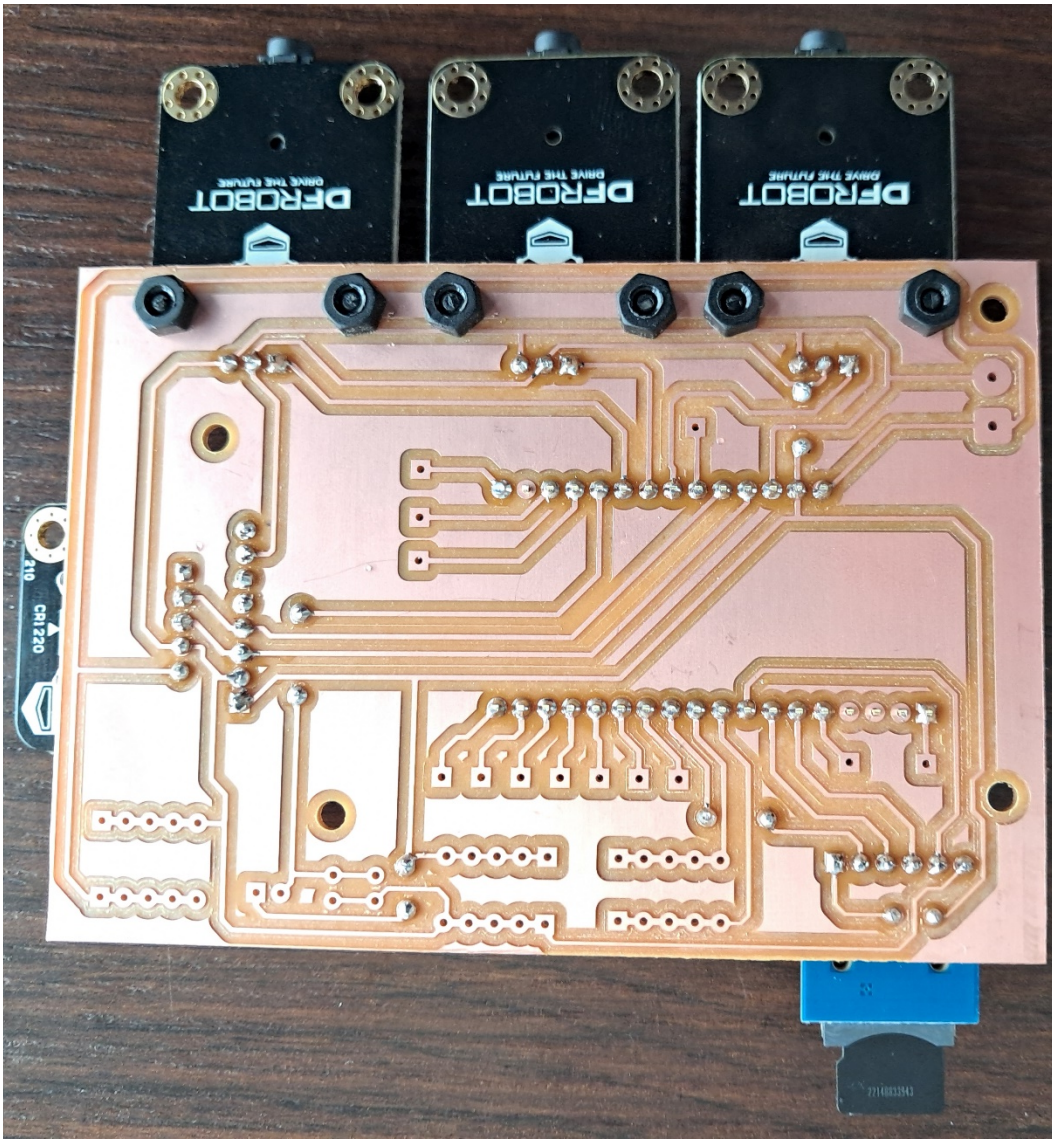


Figura 32: integrazione dei moduli sensori sulla board 1

La **Board 2 – Unità Ambientale Multisensore.** Questa scheda si occupa della raccolta di parametri ambientali di qualità dell'aria e condizioni atmosferiche, ed è dotata dei seguenti elementi:

- **Modulo ESP32**: come per la board 1, svolge il ruolo di unità di controllo.

- **Modulo RTC** e **Modulo SDCard**: replicano le funzioni già presenti nella board 1 per garantire indipendenza funzionale e sincronizzazione.
- **Sensori ambientali**:
    - Ossigeno (O_2)
    - Ozono (O_3)
    - Monossido di Carbonio (CO)
    - Particolato fine (PM2.5)
    - Anidride Carbonica (CO_2)
    - Temperatura e umidità relativa

Questi sensori consentono di realizzare un monitoraggio completo della qualità dell'aria, utile per applicazioni in ambienti industriali, urbani o indoor.
La **Board 3 – Estensione per Gas e Composti Volatili** Questa terza board è fisicamente connessa alla board 2 e amplia ulteriormente la capacità di rilevamento chimico, includendo:

- Sensore di ammoniaca (NH_3)
- Sensore VOC (Composti Organici Volatili)
- Sensore di metano (CH_4)
- Sensore di biossido di azoto (NO_2)

Descriviamo la loro integrazione in modo più dettagliato. L'integrazione dei componenti sulle schede denominate **Board 2 – Unità Ambientale Multisensore** e **Board 3 – Estensione per Gas e Composti Volatili** rappresenta un esempio avanzato di progettazione modulare orientata al monitoraggio ambientale in tempo reale. L'intero sistema è stato sviluppato partendo dalla realizzazione fisica delle schede mediante lavorazione CNC, una scelta che consente un'elevata precisione nell'incisione dei circuiti, nella foratura per il montaggio dei componenti e nella definizione delle connessioni necessarie.
Sulla **Board 2**, il cuore del sistema è il **modulo ESP32**, che svolge il ruolo di unità di controllo, gestendo la comunicazione dei dati, la sincronizzazione delle misure e l'elaborazione locale delle informazioni ambientali acquisite. In stretta integrazione con il microcontrollore troviamo il **modulo RTC (Real-Time Clock)**, fondamentale per la marcatura temporale precisa dei dati raccolti, e il **modulo SDCard**, destinato alla memorizzazione locale dei dataset, permettendo il funzionamento anche in assenza di connessione a infrastrutture di rete. Questa configurazione garantisce piena indipendenza funzionale, replicando l'affidabile architettura della Board 1.

A livello sensoriale, la Board 2 è dotata di una suite completa di sensori ambientali per la rilevazione dei principali parametri atmosferici:

- Sensori di gas come **ossigeno (O_2)**, **ozono (O_3)**, **monossido di carbonio (CO)** e **anidride carbonica (CO_2)** permettono un'analisi dettagliata della qualità dell'aria.
- Il sensore di **particolato fine (PM2.5)** rileva la concentrazione di particelle sospese, essenziali per valutazioni in contesti urbani o industriali.
- I sensori di **temperatura** e **umidità relativa** completano il quadro, fornendo dati ambientali critici per la corretta interpretazione delle misure di gas e particolato.

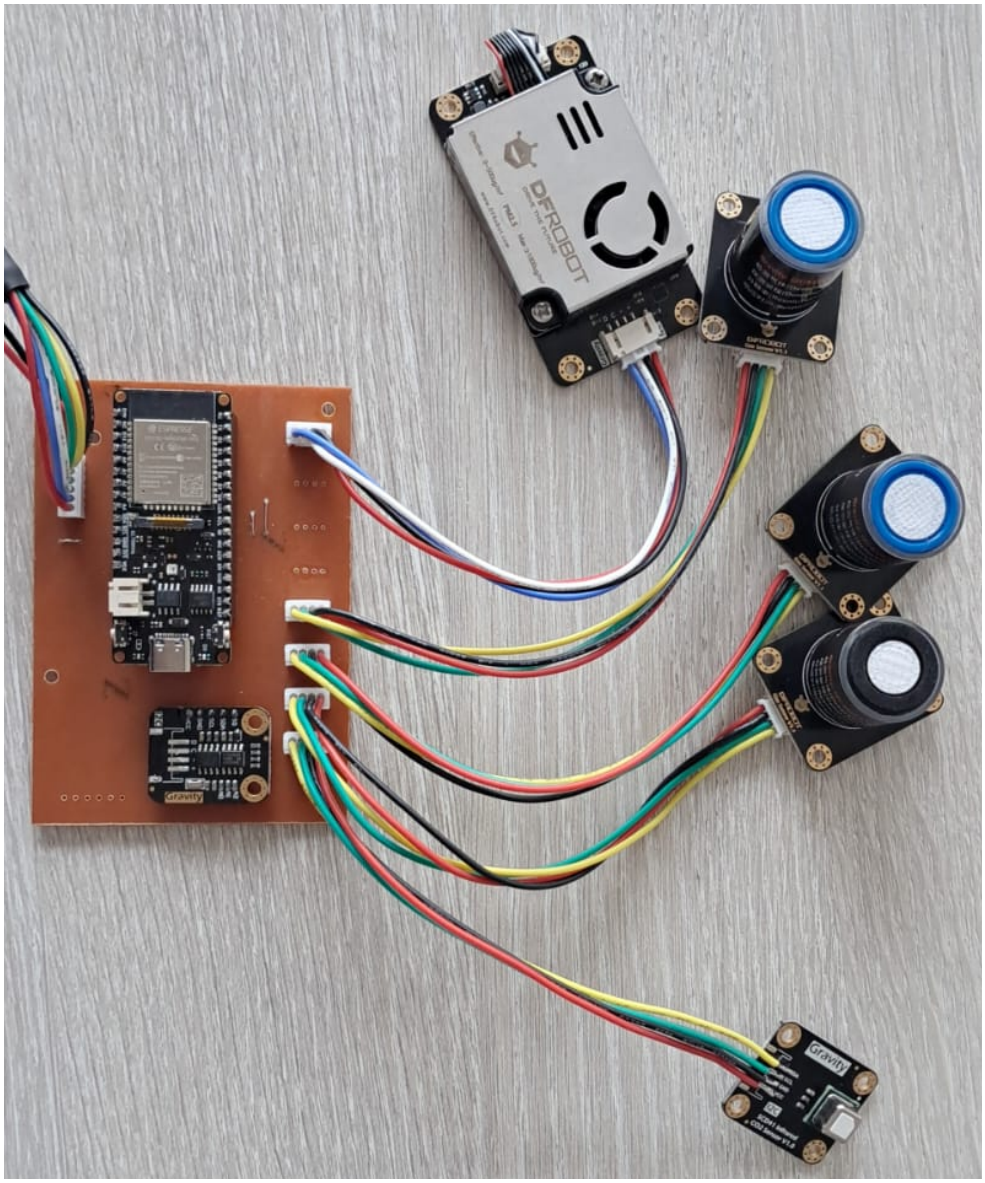


Figura 33: componentistica fissata sulla board 2

La **Board 3**, fisicamente connessa alla Board 2 mediante interfacce appositamente predisposte, costituisce una **estensione specializzata per il rilevamento di gas e composti volatili**. Essa ospita ulteriori sensori di elevata sensibilità, tra cui:

- Un sensore di **ammoniaca (NH_3)** per il monitoraggio di emissioni agricole e industriali,

- Un sensore **VOC** dedicato alla rilevazione di composti organici volatili, parametri fondamentali in ambito indoor e industriale,
- Un sensore di **metano (CH_4)**, mirato al rilevamento di fughe di gas naturale,
- Un sensore di **biossido di azoto (NO_2)**, utile per il monitoraggio delle emissioni da traffico veicolare e impianti industriali.

Dal punto di vista architetturale, l'interconnessione tra le due board è stata progettata per ridurre al minimo le interferenze e mantenere un flusso dati sincrono e stabile. La gestione dell'alimentazione, della comunicazione e della sincronizzazione temporale è centralizzata sull'ESP32 della Board 2, che si occupa anche della supervisione dei dati provenienti dai sensori della Board 3. Il montaggio dei componenti è stato eseguito con attenzione al layout termico e alla minimizzazione della lunghezza delle piste di segnale, per garantire massima affidabilità nella trasmissione dei dati sensibili. Questa integrazione modulare consente di ottenere una piattaforma altamente scalabile, pronta per applicazioni di monitoraggio ambientale professionale, ricerca scientifica o sistemi di allerta precoce in contesti critici.

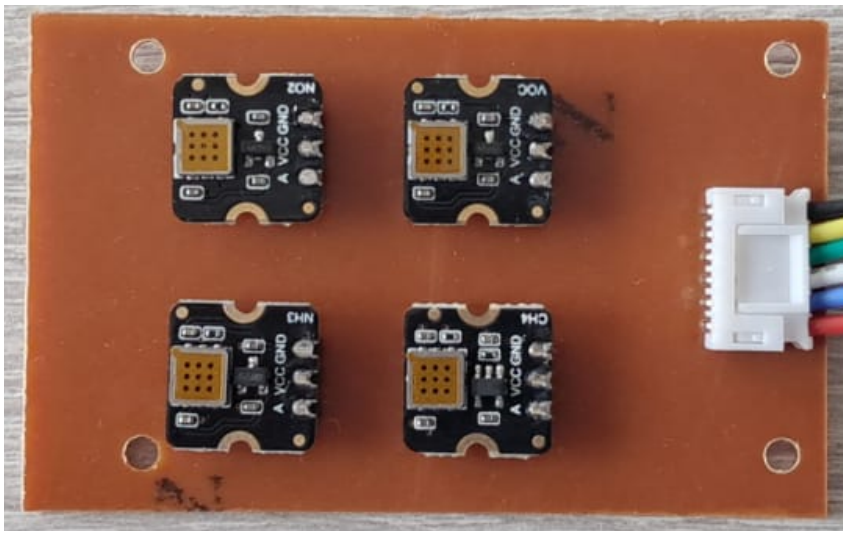


Figura 34: componentistica fissata sulla board 3

Infine nella seguente immagine si può notare le due schede, ovvero la board 2 e la board 3 tra di loro collegate.

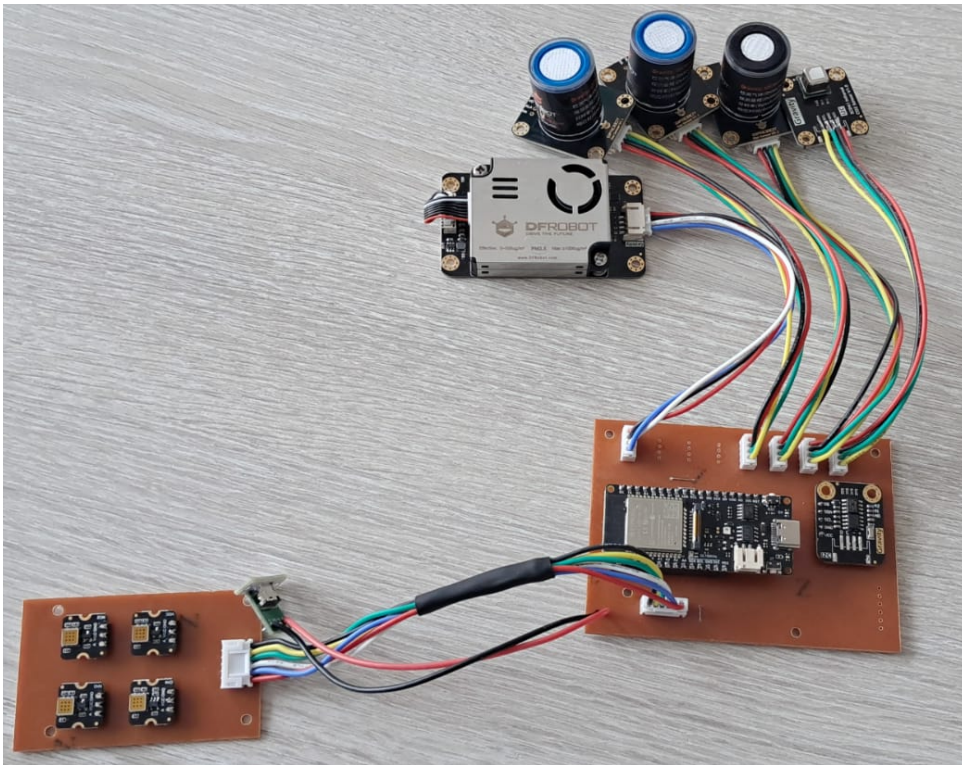


Figura 35: board 2 e board 3 tra di loro collegate

Di seguito viene riportata una **tabella riassuntiva** completa che include:

- **Board 1 – Unità di Controllo e Sensori Primari**
- **Board 2 – Unità Ambientale Multisensore**
- **Board 3 – Estensione Gas e Composti Volatili**

Includerò anche per ciascun componente la funzione principale in modo che la tabella sia chiara ed esaustiva.

Tabella 2: Riepilogo dei componenti sulle board

Board	Componente	Funzione
Board 1 – Unità di Controllo e Sensori Primari	Modulo ESP32	Microcontrollore principale, gestione dati e comunicazioni
	Modulo RTC	Sincronizzazione temporale delle misure

Board	Componente	Funzione
	Modulo SDCard	Memorizzazione locale dei dati raccolti
	Sensore di fiamma	Rilevamento di radiazioni IR generate da fiamme o scintille
	Sensore di corrente 1	Monitoraggio assorbimento linea/dispositivo 1
	Sensore di corrente 2	Monitoraggio assorbimento linea/dispositivo 2
	Sensore di corrente 3	Monitoraggio assorbimento linea/dispositivo 3
	Modulo accelerometro	Rilevamento vibrazioni e movimenti anomali
Board 2 – Unità Ambientale Multisensore	Modulo ESP32	Microcontrollore principale della sezione ambientale
	Modulo RTC	Sincronizzazione temporale delle misure ambientali
	Modulo SDCard	Memorizzazione dati ambientali locali
	Sensore Ossigeno (O_2)	Misura della concentrazione di ossigeno nell'ambiente
	Sensore Ozono (O_3)	Misura della concentrazione di ozono
	Sensore Monossido di Carbonio (CO)	Rilevamento di monossido di carbonio
	Sensore Particolato PM2.5	Rilevamento di particolato fine (polveri sottili)
	Sensore Anidride Carbonica (CO_2)	Misurazione della concentrazione di anidride carbonica
	Sensore Temperatura	Rilevazione della temperatura ambientale
	Sensore Umidità Relativa	Rilevazione dell'umidità ambientale
Board 3 – Estensione Gas e Composti Volatili	Sensore Ammoniaca (NH_3)	Rilevamento di ammoniaca nell'ambiente

Board	Componente	Funzione
	Sensore VOC	Misura di Composti Organici Volatili
	Sensore Metano (CH_4)	Rilevamento di metano e fughe di gas naturale
	Sensore Biossido di Azoto (NO_2)	Misurazione della concentrazione di biossido di azoto

6. Stato dell'arte delle tecnologie di stampa 3D

Nel seguito di questa sezione è opportuno fare una descrizione dello studio dello stato dell'arte delle tecnologie di stampa 3D in modo da far capire al lettore gli enormi vantaggi derivanti dall'uso del 3D printing.

La Stampa 3D nasce come evoluzione della stampa 2D ma non ha molto a che fare con la stampa editoriale o gli stampi industriali. La stampa 3D è un processo produttivo industriale, chiamato anche manifattura additiva, in cui un oggetto viene realizzato aggiungendo strato su strato di materiale, seguendo le istruzioni di un modello digitale. La parola "stampa" è considerata nel senso di "creare" dove l'unico "stampo" è la creatività. Nei paragrafi seguenti sarà descritta nel dettaglio questa tecnologia, dal punto di vista tecnico e funzionale.

6.1. Produzione additiva vs produzione sottrattiva

La produzione di oggetti solidi tridimensionali può avvenire sia attraverso tecniche "sottrattive" sia "additive". Nella manifattura i metodi sottrattivi sono etichettati come "tradizionali". Con la produzione sottrattiva gli oggetti si ottengono tagliando o scavando il materiale da una forma più grande. Le tecnologie più diffuse di questo tipo sono le frese a controllo numerico e il laser cutter.

La produzione additiva invece, prevede di creare un oggetto attraverso la sovrapposizione di strati multipli e sottili di materiale. Le tecniche additive esistenti differiscono secondo il modo in cui gli strati sono depositati e del materiale che può essere usato. I vantaggi principali della produzione additiva sono:

- è possibile realizzare forme veramente complesse e dettagliate senza l'ausilio di stampi o attrezzature (con la produzione sottrattiva non è possibile);
- lo scarto del materiale è minimo, vi è minore spreco e i costi sono inferiori;
- i materiali che possono essere usati sono moltissimi: dal metallo alla ceramica, alla plastica e alla sabbia;
- la produzione additiva è principalmente utilizzata nella prototipazione rapida per produrre modelli e avere un'idea realistica dell'oggetto che si sta progettando prima della sua produzione in serie.

In altri termini, la produzione additiva è un processo di fabbricazione che consente di realizzare oggetti fisici partendo da un modello digitale. Numerose sono le tecnologie e i materiali a disposizione, ma il principio di base è lo stesso: attraverso la sovrapposizione di strati di materiale, uno dopo l'altro, un modello digitale viene trasformato in un oggetto solido tridimensionale [1]. È importante sottolineare che la dicitura "produzione additiva" non si riferisce ad una singola tecnologia ma bensì ad un insieme di processi di produzione, molto diversi tra loro, accomunati da tre fattori:

1. Sono processi di produzione impiegati per la creazione di oggetti tridimensionali.
2. Gli oggetti in questione vengono realizzati attraverso la sovrapposizione di strati successivi di materiale.

3. I prodotti vengono realizzati partendo da un modello 3D digitale.
I processi di produzione additiva si differenziano dai processi di produzione tradizionali, e congiuntamente a questi ultimi fanno parte di una più ampia gamma di processi di fabbricazione a disposizione del settore terziario.

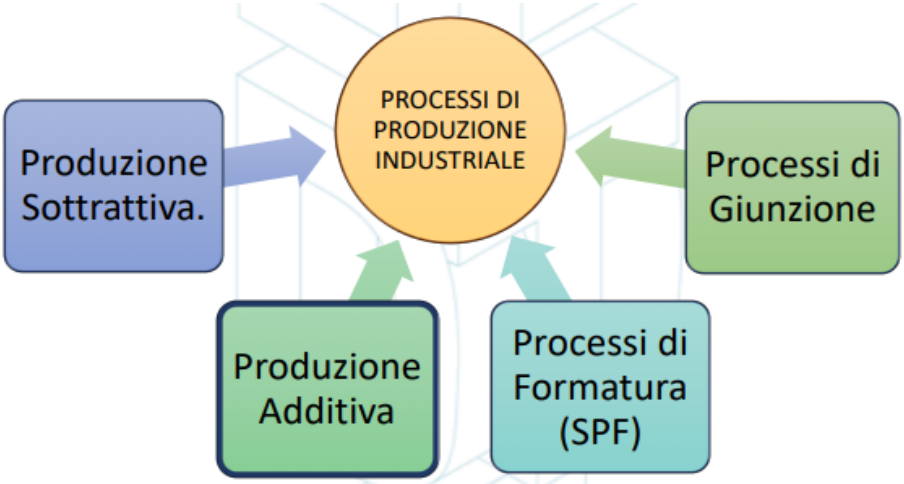


Figura 36: Processi di produzione industriale

Oggi, grazie ai progressi avvenuti è possibile produrre in poco tempo oggetti realistici che non richiedono ulteriori finiture. La produzione additiva, inoltre, assorbe meno energia della produzione sottrattiva, punto chiave per i produttori.

6.2. Mass production vs Mass customization

Lo scenario competitivo in cui le aziende si trovano a operare oggi è evoluto in modo rapido e profondo negli ultimi anni, rendendo la sopravvivenza nel mercato una sfida sempre più difficile.
In particolare, il profilo della domanda è mutato profondamente:

- diminuzione dei volumi: la domanda complessiva, in particolare nei mercati occidentali, è in calo anche per colpa della crisi economica che ha investito il mondo negli ultimi anni;
- aumento della complessità: i prodotti richiesti devono soddisfare requisiti sempre più specifici anche, o addirittura soprattutto, in termine di servizi associati. In questo senso la manifattura smetterà di essere solo la produttrice di oggetti ma si trasformerà in venditore di soluzioni, dove il prodotto fisico sarà solo il tramite verso una gamma di servizi immateriali;
- aumento della personalizzazione: sempre più spesso il cliente richiede la personalizzazione del prodotto, per soddisfare esigenze di unicità. Le aziende saranno costrette a rivoluzionare la propria gamma, cercando di produrre in modo efficiente quantità ridotte di una più ampia varietà di articoli.

In molti settori questo cambiamento ha messo in crisi il paradigma produttivo della "Mass Production" caratterizzata dall'offerta di soli prodotti fisici, dalla realizzazione di grandi serie, dalla standardizzazione dei prodotti e dalla saturazione del mercato.
Per operare nel contesto del mercato di oggi, complesso e in continua evoluzione, sono richieste competenze e strumenti differenti rispetto al passato. Si entra quindi nel nuovo paradigma della "Mass Customization", dove la frammentazione dei mercati, un ciclo di vita dei prodotti sempre più compresso, prestazioni richieste sempre più elevate e una crescente richiesta di personalizzazione sono solo alcune delle sue linee guida.
L'additive manufacturing è uno dei principali strumenti che un'azienda può utilizzare per far fronte a queste nuove sfide. È, infatti, ormai consolidato che queste tecniche siano economicamente valide per la produzione di piccoli lotti di oggetti piccoli e complessi. Sorge però spontanea una domanda: quante sono le unità che devono comporre un lotto per far sì che la stampa 3D risulti conveniente rispetto alla manifattura tradizionale? Purtroppo oggi non si è in grado di dare una risposta univoca, ma si devono effettuare valutazioni tecniche ed economiche della tecnologia in ogni ambito di utilizzo.
Una volta selezionata la tecnica additiva preferibile, dopo l'analisi dei costi principali dovuti a macchina, attrezzature accessorie, materiali e risorse umane, è possibile confrontarne i costi con quelli legati alle tecniche tradizionali. La stampa 3D risulterà conveniente per ridotti volumi produttivi, fino ad un certo break-even point, opportunamente calcolato.

6.3. L'impatto della stampa 3D sul ciclo di vita dei prodotti

Secondo molte ricerche di mercato, la stampa 3D è la più diffusa tra le principali tecnologie emergenti della manifattura 4.0. Definita come una disruptive technology, sarà una delle tecniche che rivoluzioneranno le nostre vite nei prossimi 10-20 anni.
Le opportunità di mercato sono notevoli sia nel campo artigianale dei maker sia nel campo industriale. Per quanto riguarda le aziende, la stampa 3D può avere diversi ruoli:

Figura 37: Ruoli Stampa 3D

Prototipazione
È il primo campo di applicazione che ha portato alla nascita della stampa 3D. Questa fase è finalizzata alla realizzazione di prototipi di prodotti finiti e/o componenti, grazie ai quali è possibile effettuare valutazioni estetiche e funzionali. I benefici principali riguardano la riduzione del time to market, ridotto da mesi a

settimane. Un altro beneficio è la possibilità di stampare diversi componendi in un solo processo, permettendo agli ingegneri di valutare le combinazioni migliori per rispondere alle esigenze del mercato.
Grazie alla stampa 3D, si passa direttamente dalla fase di design a quella di produzione eliminando i passaggi intermedi di realizzazione di utensili e stampi e garantendo quindi la convenienza economica della produzione di piccoli lotti.

Produzione indiretta

Per produzione indiretta si intende la realizzazione della strumentazione necessaria per produrre altri oggetti. Grazie al 3D printing, quindi, un'azienda può produrre internamente gli utensili e le attrezzature necessarie per la produzione vera e propria. I benefici principali sono la maggiore flessibilità, una riduzione dei tempi di produzione e la riduzione dei costi.

Produzione diretta

La stampa 3D garantisce, oltre alla possibilità di ottenere forme complesse e geometricamente difficili da realizzare con le tecniche tradizionali, l'incremento di alcune caratteristiche meccaniche dei prodotti. La manifattura additiva non ha ancora le caratteristiche per poter produrre in un mercato "mass production", ma garantisce la produzione in piccola serie di prodotti ad alto valore aggiunto, oggetti di design, oreficeria, industria automobilistica e aerospaziale, meccanica di precisione, protesi e altri dispositivi medici, che richiedono un alto livello di personalizzazione e complessità, prodotti tipici del Made in Italy.

Parti di ricambio

Le parti di ricambio, nel mercato odierno, sono prodotte e stoccate all'interno di depositi sparsi per il mondo. La domanda è sporadica ma è caratterizzata da numerosi codici talvolta con elevato valore e con alto rischio di obsolescenza. Con il 3D printing i pezzi di ricambio vengono realizzati on-demand. I benefici conseguenti sono molteplici: minori scorte, meno trasporti e minor necessità di magazzini. Anche il modello di business può subire modifiche in questo contesto, perché ad essere venduto potrebbe essere non più il prodotto fisico ma il modello virtuale dello stesso, che potrebbe essere stampato direttamente dall'utente.

6.4. Il processo di produzione

Fondamentalmente il processo di produzione si può articolare in 3 passaggi obbligati:

- Realizzazione modelli 3D;
- Stampa
- Rifinitura

Realizzazione modelli 3D

È la prima fase del processo ed è composta di due passaggi. Il primo passaggio prevede di costruire il modello tramite un software 3D che permette di ottenere rappresentazioni matematiche o modelli tridimensionali dell'oggetto da costruire. Attraverso programmi di tipo CAD o di Animation Modeling Software l'utente disegna a

computer l'oggetto da realizzare. I software usati per la modellazione possono lavorare file con formati di diverso tipo dallo standard .obj.
Un file STL rappresenta un solido la cui superficie è stata discretizzata in triangoli. Esso consiste nella ripetizione di vettori contenenti le coordinate dei tre vertici di ciascun triangolo e l'orientazione della normale alla superficie. Presenta dei vantaggi quali la semplicità, in quanto risulta molto facile da generare e da processare, mentre a suo sfavore presenta una geometria approssimata e la sua struttura dati, che pur risultando semplice, può presentare la ripetizione dello stesso vertice più volte.
Il PLY è un formato file input generato da scanner. I file VRML (o WRL) sono usati come input per tecnologie 3D capaci di stampare a colori.
Questi software hanno la possibilità di poter essere utilizzati per una vasta gamma di applicazioni: per progettare edifici e paesaggi, per fare schizzi e veicoli, per realizzare modelli dettagliati di composti chimici o formazioni geologiche.
La modellazione 3D è stata usata anche per videogiochi ed effetti speciali nei film.
Negli anni '90, i software *3D Studio Max* e *Rhino 3D* guidarono l'evoluzione dei programmi di modellazione 3D. Negli anni il numero dei software e delle applicazioni 3D è incrementato. Oggi il mercato offre una vasta gamma di programmi che si differenziano per capacità, costi e facilita d'uso. I programmi di modellazione 3D più famosi sono:
1. *3D Studio Max*: nasce per il mondo della grafica e dei videogiochi, ma nel tempo si è aggiornato con nuove funzioni che permettono di creare ogni tipo di modello e animazione. Ha un'impronta più artistica che tecnica, ma è in grado di produrre disegni tecnici precisi. È un programma semplice e intuitivo e ha un costo modesto;
2. *Rhinoceros 3D*: conosciuto anche come *Rhino*, nasce appositamente per l'impiegato nel mondo della progettazione e del design industriale. L'utilizzo principale di linee curve lo rende adatto a progettare gioielli, automobili, imbarcazioni, ma anche qualsiasi altro modello per la stampa 3D. Il costo è più alto rispetto a quello precedente, ma ha la possibilità di aggiungere nuove funzioni, disponibili anche gratuitamente;
3. *Inventor*: è il programma CAD per progettazione industriale e meccanica per eccellenza. È un programma professionale per la modellazione solida, con cui disegnare e testare i prodotti ancora prima che vengono realizzati, grazie ad un modulo inserito all'interno del suo sistema. Il prezzo è molto elevato, circa 8000€.
4. *OnShape*: è un programma CAD professionale pensato per la progettazione meccanica, con la caratteristica di essere completamente online. Utilizza il browser come interfaccia e i server in cloud per le elaborazioni più complesse. Al momento di andare in stampa, essendo ancora in versione beta, è utilizzabile ancora in forma gratuita.
Completato il progetto, il secondo passaggio consiste nel trasformare i file in una serie d'istruzioni da comunicare alla stampante (dette G-Code). Un software dedicato (i più famosi sono *Repetier-Host*, *Slic3r*, *Cura* e *Replicator G*) "taglia" il modello virtuale in tanti piani bidimensionali orizzontali che verranno poi stampanti uno sopra l'altro.

Stampa

Nella fase di creazione vera, la stampante legge il G-Code e inizia a stendere gli strati di liquido, polvere o altro materiale per realizzare il modello attraverso quindi una serie di sezioni orizzontali. Tali sezioni si fonderanno per ottenere l'oggetto finale. Uno dei più importanti vantaggi di questa tecnica è proprio la possibilità di riprodurre qualsiasi forma. Ogni modello deve avere almeno un piano di appoggio. Lo spessore di ogni strato normalmente è circa 0,1 mm. Le dimensioni dell'oggetto ovviamente non possono essere maggiori dell'area di lavoro e quindi dipenderanno dal tipo di stampante di cui si è in possesso. La realizzazione di un oggetto oggi può durare ore o addirittura giorni, secondo il metodo usato, della dimensione e della complessità del modello stesso.

Rifinitura

Essendo la produzione di tipo additivo, il risultato sarà di alta qualità. È però possibile scegliere di stampare una versione leggermente più grande dell'oggetto per poi rimuovere il materiale in eccesso e le piccole imperfezioni con un processo sottrattivo. Questo permette di avere una precisione maggiore. In questa fase si rimuovono inoltre i possibili supporti utilizzati per la stampa.

7. Tecnologie abilitative del 3D printing

Da quando è nata questa tecnologia, si sono sviluppate diverse tecniche di produzione additiva. Queste variano in base ai materiali utilizzati (liquidi come nella stereolitografia, fusi come nel Fused Deposition Modeling) e al modo in cui vengono depositati i vari strati uno sopra l'altro.

7.1. Estrusione

Il metodo più conosciuto è il Fused Deposition Modeling (FDM). Brevettato nel 1988 da Scott Crump, fondatore della nota casa produttrice di stampanti 3D Stratasys, viene commercializzato all'inizio degli anni novanta. Scaduto il brevetto di tale tecnologia, nascerà una comunità di sviluppo open source che svilupperà varianti piu economiche rispetto alla Stratasys, denominando la tecnica di stampa Fused Filament Fabrication (FFF), in quanto FDM faceva esclusivamente riferimento al brevetto Stratasys. Il metodo di lavoro di FDM e FFF è però identico: un filamento plastico, di cera o un filo metallico è srotolato da una spirale che fornisce il materiale a un ugello di estrusione che deposita il materiale in modo da costruire strato dopo strato i piani bidimensionali dell'oggetto. L'ugello è riscaldato attraverso dei radiatori che riescono a mantenere la temperatura al sopra del punto di fusione del materiale in modo che questo riesca a fuoriuscire senza intoppi. La piattaforma di lavoro invece è caratterizzata da temperature più basse in modo da permettere al materiale appena depositato un indurimento rapido. Al completamento dello strato, la piattaforma si abbassa di un layer e la testina di estrusione deposita un altro strato di materiale. Lo spessore di ogni strato varia da 0,013 a 0,005 pollici (da 0,33 a 0,127 mm). Uno dei limiti è la difficoltà di realizzare oggetti

cavi, se non attraverso l'utilizzo di supporti che verranno poi eliminati alla fine del processo. Non prevede particolari trattamenti post produzione. È spesso denominata anche Plastic Jet Printing.
Una variante del FDM è lo Stick Deposition Moulding. L'estrusore della stampante è alimentato con appositi bastoncini soggetti a scivolamento, invece che da un filamento circolare. Questa innovazione offre alcuni vantaggi: i bastoni che alimentano automaticamente l'ugello provengono da un magazzino di alimentazione che permette un dosaggio preciso del fuso; inoltre questi bastoncini possono essere facilmente combinati per produrre un oggetto in vari colori e materiali. Altra variante è lo Smooth Curvature Printing (SCP). È un algoritmo che elimina le creste di superficie che a volte appaiono durante la stampa tradizionale vettoriale. Basato sulla tecnologia di controllo del movimento, l'SCP aumenta la velocità di stampa e sostiene un movimento fluido durante il processo in modo tale che le superfici curve risultino notevolmente più lisce e più uniformi.

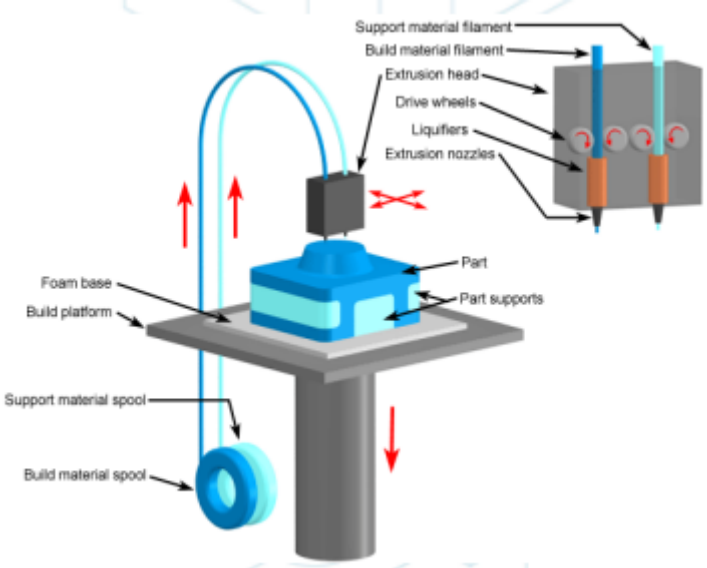


Figura 38: Tecnica di stampa Fused Filament Fabrication (FFF)

Esiste una vasta gamma di materiali che possono essere impiegati nella stampa FDM. La prima distinzione è quella tra materiali industriali e materiali ad uso domestico. I più comuni sono: ABS (Acrilonitrile-Butadiene Stirene) o Cycolac, PLA (Acido Poli lattico) e Nylon (Poliammide), ma è possibile utilizzare miscele di più materiali come plastica e legno o carbonio. Grazie ai numerosi vantaggi che questa tecnologia presenta, la stampa FDM viene spesso utilizzata per la realizzazione di parti di concept, modelli funzionali, prototipi, parti complesse destinate all'uso finale, e utensili di produzione. In particolar modo, la tecnologia FDM è sfruttabile nella produzione in volumi limitati di prototipi per test di forma, idoneità e funzione.

7.2. Tecnologia FDM Vantaggi e Svantaggi

Quando si parla di stampa 3D, una delle principali perplessità ad essa correlate riguarda i costi. L'utilizzo a lungo termine dei materiali può rivelarsi una spesa gravosa, tuttavia coloro che intendono fare uso della tecnologia FDM godono di un vantaggio iniziale; le stampanti FDM sono tra le più economiche e accessibili in commercio, specialmente quelle ad uso domestico. Molte case produttrici offrono modelli già pronti per l'utilizzo, come Maker Bot e Ultimaker, o possono essere costruite con un kit fai da te. Tra gli altri vantaggi, la tecnologia FDM viene considerata una "tecnologia pulita" (o sostenibile), facile da utilizzare e "office-friendly". È anche in grado di produrre geometrie complesse e cavità altrimenti problematiche da realizzare [14]. Per quanto riguarda l'accuratezza, le stampe 3D non raggiungono lo stesso livello di accuratezza e qualità degli altri oggetti che vengono invece prodotti attraverso l'uso della stereolitografia. Detto questo, il risultato è considerato abbastanza qualitativo, a seconda del settore in cui viene applicata la tecnologia. La risoluzione dipende principalmente dalle dimensioni dell'ugello utilizzato. La precisione della macchina dipende dai movimenti dell'estrusore sull'asse X e Y, ma ci sono altri fattori da prendere in considerazione. Ad esempio, la forza di adesione tra gli strati è inferiore rispetto al processo di Stereolitografia. Di conseguenza, il peso degli strati potrebbe schiacciare gli strati inferiori, il che può quindi influenzare e persino compromettere la qualità della stampa stessa. Contrariamente alla SLA, la tecnologia FDM presenta anche una maggiore complessità. Bisogna tener conto del peso e delle dimensioni, ma anche dei vincoli. È molto importante assicurarsi che il prototipo soddisfi le aspettative rappresentate sullo schermo dal modello digitale. I vincoli sono determinati da diversi fattori, in primis dal materiale impiegato, attraverso il quale è possibile prevedere che dimensioni potrà avere il prototipo 15]. Un altro svantaggio è rappresentato dai tempi di stampa, più lunghi rispetto a quelli richiesti dalla Stereolitografia o dalla Sinterizzazione Laser Selettiva, che sarà analizzata successivamente. Inoltre, la finitura superficiale è considerata accettabile, ma non di qualità. Infatti, un prototipo realizzato con stampante FDM richiederà quasi sempre una fase di post-lavorazione.

7.3. Digital Light Processing

I processi che funzionano secondo la tecnica del Digital Light Processing (DLP) sono basati su trasformazioni fotochimiche dei materiali. Una sorgente luminosa convenzionale, come ad esempio una lampada ad arco, attraverso un display a cristalli liquidi o un dispositivo specchio deformabile (DMD) che viene applicato su tutta la superficie della vasca in cui è presente il materiale, illumina il fotopolimero e questo indurisce proprio dove la luce colpisce la superficie. Una volta che lo strato è completato, la piattaforma all'interno della vasca si abbassa di una frazione e lo strato successivo viene tracciato. Questo continua fino a quando l'intero oggetto è completato.
Il metodo più diffuso è la stereolitografia (SLA). Dopo che il progetto CAD viene suddiviso in piani bidimensionali, il software trasmette queste informazioni a una sorgente laser. Appena al di sotto del livello del fluido, resina liquida, è presente una piastra forata. Il raggio laser viene proiettato da un sistema di specchi in modo da scandire la superficie del liquido e permettere di indurire le parti che identificano la sezione dell'oggetto da costruire. Una volta

indurito, la piattaforma si abbassa di uno strato e nuova resina liquida viene esposta allo stesso trattamento, andando a sormontare lo strato precedente. Al termine, il solido viene estratto dalla resina liquida e messo in un forno a luce ultravioletta per completare la polimerizzazione. Un metodo di questa tipologia molto conosciuto è il metodo proprietario dell'Asiga chiamato Sliding Separation (SS). Un metodo che sfrutta la stessa tecnologia è il PolyJet/MultiJet. Questa metodologia assomiglia molto alla stampa a getto d'inchiostro 2D ma, invece di lasciar cadere gocce di inchiostro su un foglio di carta, spruzza fotopolimeri liquidi su un carrello mobile. Si creano così strati sottilissimi (tra i 16 e i 30 micrometri) di materiale che, sottoposti a un fascio di raggi UV, si asciugano velocemente e consentono all'oggetto di essere immediatamente pronto all'uso, senza bisogno di ulteriori lavorazioni. La macchina infine spruzza un liquido gel che ha funzione di supporto per le parti sporgenti o altre più complesse, che viene rimosso facilmente con acqua o mani. Eventuali supporti usati nella costruzione di parti complesse vengono rimossi al termine della stampa.

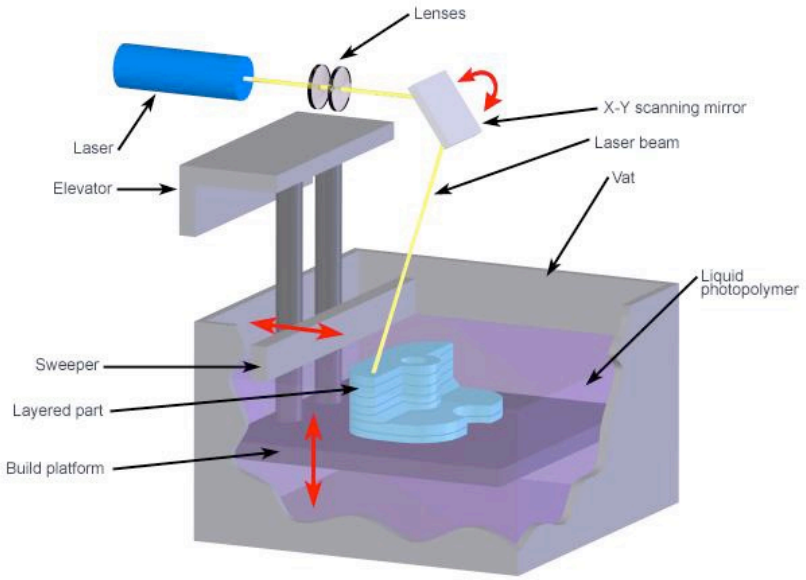


Figura 39: tecnica di produzione Stereolitografica (SLA)

7.4. Tecnologia SLA Vantaggi e Svantaggi

La Stereolitografia è considerata una delle migliori tecnologie di stampa 3D presenti oggi sul mercato. Uno dei maggiori punti di forza riguarda l'alta risoluzione dei dettagli. Permette infatti di stampare oggetti dalle geometrie molto complesse senza comprometterne la qualità. Una delle massime prerogative è la precisione, pertanto la tecnologia SLA è la scelta più azzeccata nel caso in cui forma, adattamento e assemblaggio siano la priorità. Tale precisione è dovuta all'utilizzo delle resine, ovvero, i cosiddetti materiali fotopolimerici. Questi ultimi, dalla consistenza liquida, vengono induriti per mezzo di un raggio laser; offrono una certa libertà in termini di colore,

grado di opacità e rigidità, garantendo tuttavia un'eccellente qualità della superficie. Nonostante la presenza di tecnologie più rapide e recenti sul mercato, la stereolitografia è in grado di funzionare correttamente e in tempi ragionevoli, consentendo di risparmiare tempo anche su quelle parti che richiedono una precisione elevata. I prototipi possono pertanto essere realizzati senza troppe difficoltà rimanendo fedeli al progetto iniziale. Un ulteriore vantaggio derivato dell'uso della stereolitografia e dei materiali in resina, è la personalizzazione, soprattutto per quanto riguarda la scelta del colore. Uno dei metodi più utilizzati è la verniciatura spray. Le varietà disponibili sono quattro: extra opaco, opaco, satinato e lucido, ognuno con un fattore di lucentezza diverso. Più alto è il fattore di lucentezza, più brillante sarà il modello. Quanto più sarà ampia la superficie su cui la vernice viene applicata tanto più sarà visibile la differenza tra i diversi gradi di lucentezza. E' possibile svolgere le operazioni di verniciatura a casa propria o affidarle a degli esperti in base alle proprie esigenze. Uno degli svantaggi più grandi è invece rappresentato dai costi, piuttosto elevati. Nonostante le stampanti vengano considerate più o meno accessibili, i materiali fotopolimerici risultano molto costosi, rendendo l'uso di questa tecnologia poco abbordabile. Inoltre, anche se sono disponibili in diversi colori, c'è ancora una scelta limitata di fotopolimeri. Inoltre, data l'irritabilità e la tossicità delle resine liquide, si consiglia sempre di prendere le dovute precauzioni e di fare uso delle attrezzature e degli strumenti più adeguati. Tra gli altri svantaggi, le stampe stereolitografiche richiedono normalmente una fase di pulitura, con conseguente impiego di tempo e sforzo non indifferenti. Per raggiungere un livello di qualità ottimale è necessario rifinire il prototipo in fase di post-lavorazione. Il livello di qualità desiderato e la scelta del materiale comporteranno un numero maggiore o minore di passaggi. Contrariamente alla Sinterizzazione Laser Selettiva, tecnologia molto simile ad essa, la Stereolitografia richiede l'utilizzo di strutture di supporto, e di conseguenza una quantità maggiore di materiale, che rende tale tecnologia più costosa.

7.5. Fusione di materiale in granuli

Il metodo più famoso è il Selective Laser Selting (SLS). Sviluppato e brevettato da Carl Deckard e Joseph Beaman presso l'Università del Texas, il processo di SLS prevede che uno strato di materiale da costruzione, contenuto in una cartuccia, venga spalmato su una piattaforma e venga livellato con un rullo. Il laser traccia una sezione orizzontale bidimensionale della parte, sinterizzando il materiale da polverulento a indivisibile. Dopo il completamento di ciascuno strato, un apposito pistone si muove verso il basso e abbassa di conseguenza anche lo strato appena realizzato. Viene quindi fatto uscire dalla cartuccia nuovo materiale che subisce la stessa lavorazione, così finché la parte è costruita. Una volta pronto, il modello viene rimosso dalla camera e viene terminato rimuovendo il materiale di scarto e lisciando le superfici visibili. Il fatto che il materiale di costruzione a differenza delle tecnologie precedenti sia solido e l'oggetto sia realizzato immerso completamente nel materiale, porta a vantaggi pratici evidenti in quanto non necessita di strutture di supporto specifiche come avviene per la SLA e il FDM. Il Selective Heat Sintering (SHS) è un metodo del tutto simile al SLS. La differenza sostanziale è il fatto che non viene utilizzato un laser ma una testina di stampa termica meno intensa. Questo ha il vantaggio di ridimensionare le dimensioni della camera di produzione e di conseguenza anche i costi, rendendola così una

soluzione molto più economica. Il Selective Laser Melting (SLM), detto anche Direct Metal Laser Sintering (DMLS), è del tutto identico al metodo SLS ma, a differenza di questo, utilizza come materiali di costruzione polveri metalliche integrali cioè che non contengono materiali bassofondenti, leghe che fondono a temperature inferiori ai 150° C. Per evitare l'ossidazione dei materiali, si utilizza una camera di lavoro con atmosfera inerte. Il laser utilizzato è quindi piu potente e al termine della lavorazione si ottiene un oggetto simile a quelli che si ottengono con la produzione in serie. L'Electron Beam Melting (EBM) è un metodo molto simile al SLM ma, a differenza di questo, per permettere una corretta focalizzazione del fascio di calore (con potenza superiore al laser) sul piano di lavoro crea il vuoto nella camera e questo consente di evitare la formazione di ossidi metallici nelle polveri. La potenza superiore a quella del laser permette alla macchina di fondere polveri altofondenti, realizzando parti con alta densità e molto forti. Tra i metodi che sfruttano polveri di materiali si può classificare anche la stampa a getto d'inchiostro. Questo metodo permette di ottenere oggetti tridimensionali in tempi inferiori, con costi minori e con facilità d'uso elevata. Il materiale viene disteso su una piattaforma e pressato da un rullo e, a differenza del SLS, i granuli non vengono sinterizzati ma legati con liquido adesivo erogato da un'apposita testina. Successivamente, come per gli altri metodi, la base si abbassa di uno strato e il procedimento si ripete fino al completamento del prodotto finale.

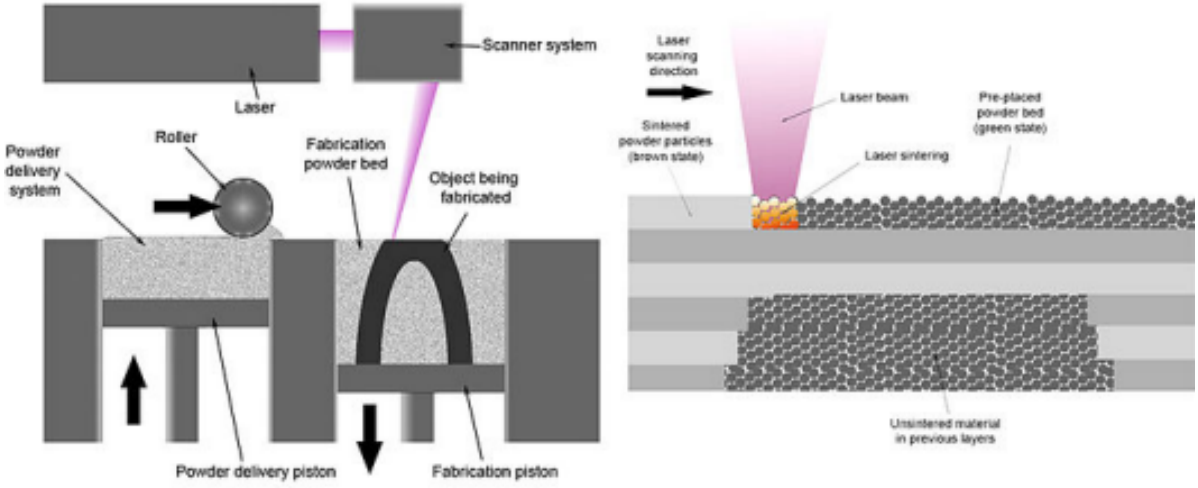


Figura 40: tecnica di produzione Selecting Laser Selting (SLS)

7.6. Tecnologia SLS Vantaggi e Svantaggi

Uno dei vantaggi più evidenti della Sinterizzazione Laser Selettiva, è la mancata necessità di fare uso di strutture di supporto, in quanto essa è completamente autoportante. Consente quindi di costruire parti all'interno di altre parti in un processo chiamato nestino. Ciò comporta due cose: una riduzione dei costi del materiale che sarebbe invece impiegato nella stampa delle strutture di supporto, come nel caso della tecnologia FDM; la possibilità di gestire con più facilità geometrie ad alta complessità. Alcuni prodotti sono così complessi che

senza questa tecnologia sarebbe molto difficile produrli. In generale, la SLS è considerata la tecnologia di stampa 3D più veloce, sia per la stampa di prototipi funzionali durevoli che di parti destinate al consumatore finale. La durabilità è, inoltre, supportata dall'uso di materiali resistenti come il nylon, che consente anche una certa libertà nella funzionalità della stampa 3D finale. Inoltre, grazie alle sue eccellenti proprietà meccaniche, il materiale utilizzato nella SLS è spesso un valido sostituto della tipica plastica per stampaggio a iniezione. Allo stesso tempo, la tecnologia SLS produce parti considerate tra le più forti e rigide, con una buona resistenza chimica. Parti complesse con componenti interne o canali, possono essere costruite senza correre il rischio che del materiale rimanga incastrato, o che la superfice del prototipo sia danneggiata a seguito della rimozione del supporto. Figura 10: Sagoma di scarpa realizzata con SLS [23] 16 La precisione è un altro importante vantaggio della tecnologia SLS. Il risultato finale è sempre ad alta precisione. Il processo di stampa è più rapido paragonato alle tecnologie già analizzate. Grazie alla scalabilità del software è possibile realizzare la stampa di un singolo oggetto o di dozzine di pezzi con facilità. Di solito i prototipi vengono spediti in un lasso di tempo che va da 1 a 4 giorni, garantendo alle aziende velocità sul mercato. Tuttavia, i prodotti stampati tramite SLS presentano una certa porosità sulla superficie, richiedendo una fase di post lavorazione, come nel caso della Modellazione a Deposizione Fusa.

7.7. Struttura Laminare

Il Laminated Object Manufacturing (LOM) è un metodo che consiste nel laminare e depositare insieme fogli del materiale da utilizzare, impregnati con una colla adesiva al diossido di carbonio. Un meccanismo trascina i fogli sulla piattaforma di lavoro e un apposito rullo riscaldato incolla lo strato sul supporto. In seguito, una testina laser taglia i contorni della sezione. La piattaforma quindi si abbassa di un layer e il processo continua fino a terminare l'oggetto.
Questo metodo si è però evoluto nel Selective Deposition Lamination (SDL). La variazione principale è il modo di applicare la colla: mentre nel LOM viene incollato tutto nello stesso modo, nel SDL la colla è più densa al centro e meno densa sulla parte di supporto. Questo porta a una resistenza maggiore e alla possibilità di semplificare le operazioni di formazione e altre modifiche.

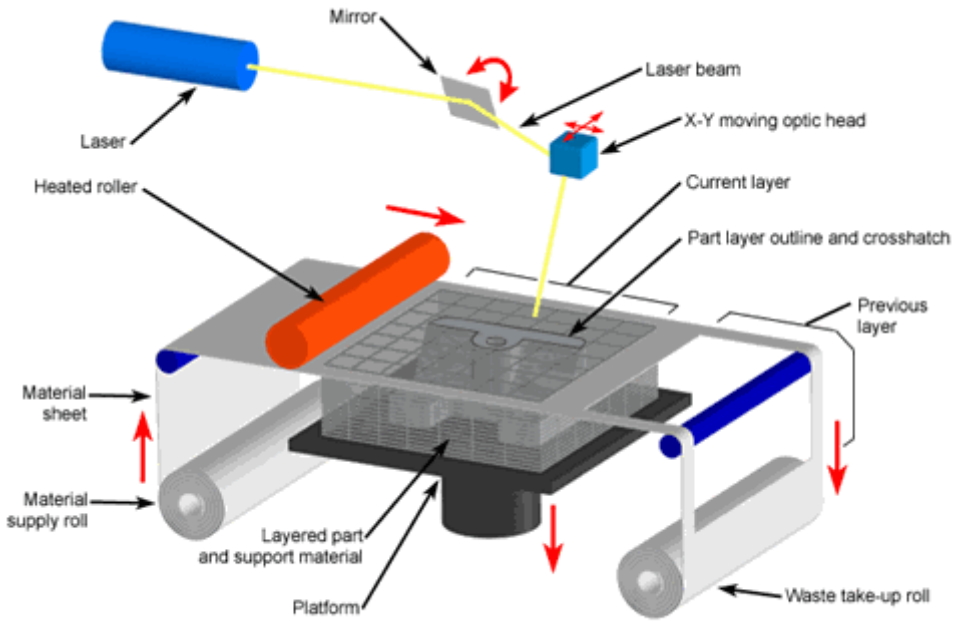


Figura 41: Tecnicha di produzione Laminated Object Manufacturing (LOM)

7.8. Caratteristiche tecniche delle stampanti

Le stampanti 3D non differenziano tra loro solo per la tecnica che utilizzano nella produzione. Ogni tipologia ha, infatti, le proprie caratteristiche tecniche. Le caratteristiche più significative sono: la dimensione della camera di lavoro, il consumo energetico, la velocità di stampa, lo spessore degli strati e la temperatura di fusione. Prima di riportare lo schema che confronta le tecnologie, è utile dare alcune direttive.
Per quanto riguarda la velocità, questa dipende molto dal tipo di materiale utilizzato. È importante associare che una velocità troppo elevata non corrisponde a un lavoro di qualità superiore.
Per motivi tecnici il primo strato ha sempre uno spessore superiore a quello degli altri. Deve essere, infatti, in grado di sostenere il peso degli strati successivi durante la lavorazione.
La temperatura di lavoro è di difficile non deve essere troppo bassa perché non si verifichino problemi alle componenti presenti, ma allo stesso tempo deve essere più bassa possibile in modo da consentire di produrre alla velocità indicata. Di seguito il confronto tra le quattro tecnologie:

Tabella 3: Confronto tra le caratteristiche principali delle tecniche di produzione

	ESTRUSIONE	DIGITAL LIGHT PROCESSING	FUSIONE DI GRANULI	STRUTTURA LAMINARE
DIMENSIONI W x D x H [mm]	100x100x100 1147x1000x1188	35x21.8x75 340x440x380	160x200x140 250x250x180	256x169x150
CONSUMO ENERGETICO	100 W – 300 W	100 W – 300 W	100 W – 300 W	100 W – 300 W
VELOCITÁ DI STAMPA	40 mm/s – 200 mm/s	3000 mm/s	100 mm/s – 200 mm/s	100 mm/s – 200 mm/s
SPESSORE DEGLI STRATI	1 strato: 0,33 mm Altri: 0,1 – 0,2 mm	1 strato: 0,33mm Altri: 0,003 – 0,1 mm	1 strato: 0,33mm Altri: 0,08 - 0,15mm	1 strato: 0,33mm Altri: 0,1 – 0,2 mm
TEMPERATURA (dipende dai materiali)	PLA: 160°C–220°C ABS: 220°C–260°C	PLA: 160°C–220°C ABS: 220°C–260°C	PLA: 160°C–220°C ABS: 220°C–260°C	PLA: 160°C–220°C ABS: 220°C–260°C

Si può notare come molte delle caratteristiche siano comuni alle varie tecnologie come il consumo energetico, il range di temperatura di lavoro, e lo spessore del primo strato. La velocità di produzione e la dimensione della camera di lavoro sono quelle che differenziano particolarmente le varie stampanti e sono i dati principali cui un'azienda deve fare riferimento, oltre il prezzo e i materiali utilizzabili ovviamente, per effettuare l'investimento dell'acquisto della macchina.

La tabella in basso mostra un confronto qualitativo tra le differenti tecniche di stampa 3D. La tabella riporta anche la presenza di ulteriori tecnologie di stampa 3D che sono state volutamente omesse dalla precedente trattazione.

Tabella 4: Confronto tecnologie 3D Printing

TECHNOLOGIES	Process	Materials used	Complexity	Speed	Max Part Size (cm)	Accuracy	Surface Finish	Strengths	Weaknesses	Pricing	Application Area	Application Examples
Fused Deposition Modeling (FDM)	Layers of melted plastic	ABS Filaments, Polycarbonate, Resin, Nylon	●●●●	Fair	30x30x50	Fair	Fair	Durable; ideal for conceptual models	Low resolution	€€	Aerospace, automotive, industrial, medical	Wind turbines, aircraft components
Selective Laser Sintering (SLS)	Plastic powder melted by laser	Paper, plastic, metal, glass, ceramic, composites	●●●	Fast	34x34x60	Good	Fair	Resistant, durable, flexible	Needs post-processing	€€	Automotive, consumer products, aerospace	Small production batches and prototypes
Stereolithography (SLA)	Polymerization scanned by UV laser	Liquid photopolymer, composites	●●●	Fast	30x30x50	Very good	Very good	High res; complex geometries	Only photopolymer materials	€€€	Aerospace, automotive, consumer goods	Medical models of anatomic human parts
Photopolymer Jetting (POLYJET)	Inkjet method with liquid photopolymers	Metals, plastic, wax	●●●	Fast	39x31x19	Very good	Good	More materials at the same time	Only photopolymer materials; not durable	€€€	Medical devices, multimaterial prototypes	Medical stethoscopes
Selective Laser Melting (SLM)	Metal powder melted by laser	Metals: copper, aluminium, tungsten etc.	●●	Fair	28x28x36	Fair	Fair	Manufactures high density parts	Price; needs post-processing	€€	Dental products, mechanical components	Lightweight components for aircraft
Electron Beam Melting (EBM)	Melted powder selected by electron beam	Metals: cobalt, chrome, nickel	●●●	Fast	20x20x20	Fair	Poor	Less thermal stress	Limited set of metals	€€€	Dental, medical implants, automotive	Bone tissue medical models
Electron Binder Jetting (BJ)	Powder distributed by jetting machine	Ceramic, metals, plastic, sand, composite	●	Fast	40x20x10	Fair	Fair	No support structure; multicolour prints	Fragile with limited mechanical properties	€	Architecture, mechanical structures	Pots and general home furniture
Continuous Fibre Fabrication (CFF)	Double nozzle laying/melting method	Plastic, carbon composites, nylon	●●●●	Fair	32x43x16	Fair	Fair	Robust parts, no post-process needed	Limited fibre placement	€€€	Aerospace	Lightweight components
Material Jetting (MJ)	Inkjet method with wax materials	Wax	●●	Slow	30x18x20	Very good	Good	High resolution	Limited wax-like materials; requires support structure	€€	Prototypes for form, fit testing; Casting patterns	Lost Wax Casting in Jewellery and Medical fields

LEGEND:

● = Simple; ●● = Fair; ●●● = Complex; ●●●● = Very complex.

€ = Cheap; €€ = Fair; €€€ = Expensive; €€€€ = Very expensive.

7.9. Materiali

Uno dei principali fattori che si devono prendere in considerazione per l'acquisto di una stampante, oltre al costo del macchinario, al costo per la stampa del prototipo e alla capacità di usare diversi colori, è la scelta dei materiali da utilizzare.

Quando si parla di Stampa 3D, la scelta dei materiali è di cruciale importanza. In passato, i materiali impiegati risultavo essere poco resistenti e si deterioravano facilmente. Oggi, invece, con il diffondersi delle tecniche di produzione additiva in tutto il mondo, c'è un interesse sempre crescente nei riguardi della stampa 3D. Molte sono le ricerche dedicate a questo nuovo campo della tecnologia, grazie anche all'idea generalmente diffusa che quest'ultima abbia il potenziale di suscitare modalità innovative di produzione. Dalle ricerche condotte e

dalle ulteriori analisi effettuate è stato possibile dare vita a nuovi materiali. Oggi il mercato offre una grande varietà nella scelta dei materiali. Dai polimeri e i metalli, passando per le ceramiche e i materiali compositi, molti sono quelli di nuova generazione, ognuno di essi con vantaggi e svantaggi.
Alcuni esempi sono:
• Prototyping Plastic, adatto ad una prototipazione rapida ed economica;
• Resina ad alto dettaglio, adatta alla stampa di disegni complessi e sculture;
• SLS Nylon, per prototipi funzionali e parti di uso finale;
• Nylon rinforzato con fibre, per la progettazione di parti forti;
• Rigid Opaque Plastic, per prototipi ad alta precisione;
• Plastica in simil-gomma, che produce un effetto simile a quello della gomma;
• Plastica trasparente, per creare parti trasparenti e prototipi;
• ABS simulato, con stampi ad alta precisione e funzionali;
• Arenaria di colore pieno, per modelli fotorealistici;
• Metalli industriali, per prototipi e parti di uso finale. L'offerta è ovviamente molto più ampia di quanto riportato qui. Per quanto riguarda invece la domanda, l'uso di un determinato materiale piuttosto che un altro è fortemente influenzato non solo dal tipo di tecnologia adottata, ma anche dalla popolarità della stampante3D.
Il materiale più comune è la plastica, in particolare il tipo ABS (Acrilonitrile – Butadiene - Stirene) che può essere utilizzata sia a iniezione sia a estrusione. È un materiale non biodegradabile e richiede temperature di estrusione di circa 240°C. È un materiale molto deformabile, può essere, infatti, flesso più volte su se stesso senza che si spezzi, caratteristica che lo rende adatto per parti mobili o flessibili. Non è un materiale semplice da stampare perché durante la fase di raffreddamento subisce notevoli deformazioni e necessita rifiniture attraverso l'utilizzo di solventi anche pericolosi come l'acetone. Ha bisogno di una camera di produzione chiusa e con piano di lavoro riscaldato. Le sue caratteristiche la rendono adeguata per la realizzazione di parti piccole o per creare prodotti resistenti agli urti e all'usura, come per esempio i mattoncini dei Lego.

Un altro materiale diffuso è il PLA (acido polilattico), un polimero completamente biodegradabile composto da amido di mais o da altri prodotti di origine vegetale come scarti, alghe o materiali poco nobili. Si estrude a temperature di circa 200°C e non necessita di un piano di lavoro riscaldato. Avendo origini vegetali va evitata l'esposizione prolungata in ambienti con forte umidità. Gli oggetti in PLA sono più rigidi, quasi cristallini, e per questo si possono spezzare più facilmente. Inoltre a temperature di circa 60°C si ammorbidiscono. Ha una temperatura più bassa rispetto all'ABS, una più ampia gamma di colori e una maggiore elasticità grazie alla quale si producono oggetti come molle.
Alcune varianti popolari del PLA sono il Laywoo-D3 e il LayBrick. Il primo è una miscela di PLA e polvere di legno riciclato (circa il 40%) con il quale è possibile stampare oggetti che assomigliano al legno e hanno addirittura un odore simile. Secondo la temperatura di lavoro questo assumerà una diversa tonalità di marrone. Dopo la stampa,

l'oggetto può essere rifinito quasi come fosse di vero legno (taglio, levigazione, pittura, etc.). Il secondo è una miscela di PLA e polvere di gesso, che conferisce agli oggetti stampati un aspetto simile alla pietra.
Anche il nylon viene utilizzato oggi nella stampa 3D. Questo, il cui nome tecnico della famiglia di materiali è poliammidi sintetiche, è resistente all'usura, elastico, colorabile attraverso tinture e resistente ai solventi. Quando viene stampato, se è utilizzato per strutture sottili è flessibile, se aumentano gli spessori aumenta di solidità e robustezza. Viene lavorato a temperature di circa 260°C e necessità di un piano di lavoro riscaldato per il primo strato è critico e mostra difficoltà nell'adesione al piano di stampa.

Figura 42: Bobine di ABS

Altro materiale utilizzato è il PET (polietilene tereftalato), venduto con nomi spesso più esotici come T-Glase in modo da nascondere la vera natura del materiale e poter applicare prezzi più elevati. Molto utilizzato nell'industria perché adatto al contatto con gli alimenti, ma se trattato con la tecnica FDM non ha ancora le certificazioni necessarie per essere utilizzato in questo modo. Viene trattato a temperature di circa 230°C con un piano di lavoro riscaldato.
Non poteva mancare tra i materiali uno con caratteristiche simili alla gomma. Trovando nell'utilizzo della gomma vera e propria difficoltà di trattamento, è stato individuato il TPU (Termo Plastic Urethane) ottimo surrogato che ne possiede le caratteristiche. Lavorato a temperature di circa 250°C, non necessita di un piano di lavoro riscaldato.
Importanti nella produzione con le stampanti 3D sono anche i materiali utilizzati per realizzare i supporti. Uno di questi è il PVA (Alcool Polivinilico), un composto chimico idrosolubile, un altro è l'HIPS (High Impact PolyStyrene) rimovibile attraverso un liquido chiamato limonene. Entrambi vengono utilizzati all'interno di stampanti con doppio estrusore, in modo che uno produca l'oggetto vero e proprio, mentre l'altro utilizzi questi materiali per realizzarne contemporaneamente i supporti.

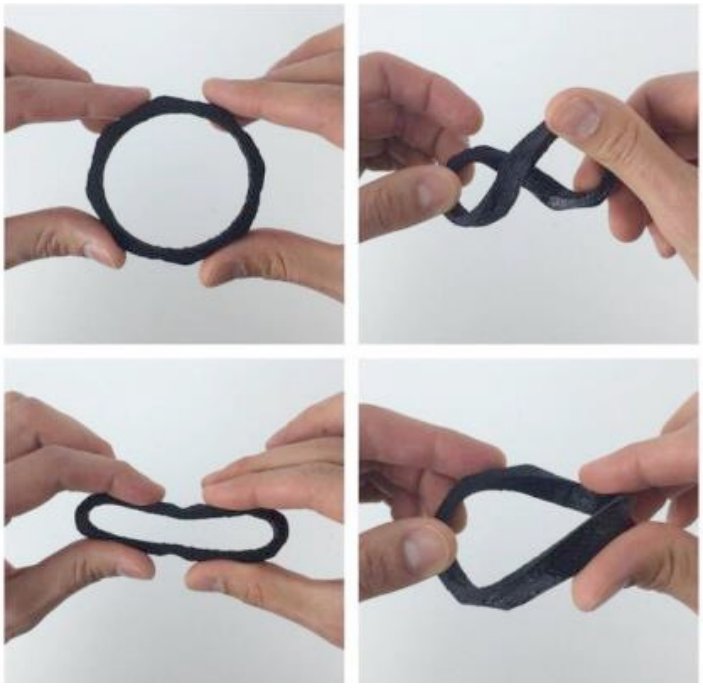


Figura 43: Oggetto stampato in TPU

7.10. Limiti relativi ad ogni tipo di tecnologia di stampa 3D

Ora verranno specificate alcune limitazioni, raggruppate in base a ciascun tipo di tecnologia:

• Limitazioni della tecnologia FDM

Scarsa finitura della superficie e bassa velocità di stampa rispetto ad altre tecnologie. Le dimensioni di stampa per le stampanti da tavolo corrispondono a 20x20x20 cm. Sono necessarie le strutture di supporto per la realizzazione di angoli inferiori ai 45 gradi. Le pareti devono avere uno spessore minimo di 0,8 mm. I dettagli incisi o in rilievo possono raggiungere una precisione di non oltre 0,6 mm in larghezza e 2 mm in altezza. I "ponti" orizzontali superiori ai 10 mm necessitano di supporto. In generale, non è possibile stampare fori di diametro inferiore a 2 mm. Per garantire il successo del tentativo di stampa è necessario che il modello non abbia dimensioni inferiori ai 2 mm. Per quanto riguarda i connettori, il diametro minimo consigliato è di 3 mm. La tolleranza (precisione dimensionale) prevista è ± 0,5% (± 0,5 mm circa).

• Limitazioni della Sinterizzazione Laser Selettiva (SLS)

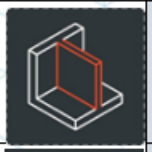
Il volume medio di costruzione è di circa 30x30x30 cm. La finitura è leggermente ruvida al tatto e opaca, se si desidera una finitura lucida e liscia, si consiglia di ultimare il prodotto in fase di post lavorazione. Le pareti devono avere uno spessore minimo di 0,7 mm. I dettagli in rilievo o incisi devono avere una larghezza e un'altezza non inferiore a 1 mm. In generale, non è possibile stampare fori di diametro inferiore a 1,5 mm. Per rimuovere il materiale di supporto (polvere non sinterizzata), è necessaria la presenza di fori nelle parti. Queste

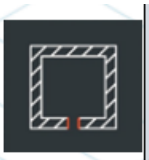
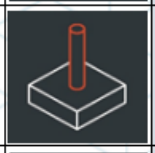
devono avere un diametro minimo di 5 mm. La dimensione minima stampabile di una sezione è di circa 0,8 mm, lo stesso vale per i connettori. La tolleranza prevista è di ± 0,3% (± 0,3 mm).

• Limitazioni della Stereolitografia (SLA)

Le stampanti SLA hanno generalmente un volume di costruzione molto più piccolo rispetto alla maggior parte delle stampanti, ad eccezione di quelle industriali, che possono avere una dimensione di stampa di 14,5 x 14,5 x14,5 cm. Quando il pezzo è più grande, è meglio suddividerlo in sezioni più piccole e assemblarle in seguito. Le resine hanno un costo molto elevato (da € 150 / litro). La maggior parte delle parti stampate attraverso questo tipo di tecnologia richiede un processo di post-polimerizzazione all'interno di un forno UV. La larghezza minima consigliata delle pareti è di 0,5 mm (se la parete in questione è collegata ad un altro elemento) o di 1 mm (se non lo è). Le strutture di supporto sono sempre richieste per gli elementi a sbalzo (con conseguente aumento dei costi). Le incisioni e i rilievi devono 51 avere una larghezza e un'altezza non inferiore ai 0,4 mm. I fori possono avere un diametro minimo di circa 0,5 mm. Per consentire al materiale in eccesso di fuoriuscire in, è necessario prevedere dei fori di fuga, che dovrebbero avere un diametro di circa 4 mm. Al fine di garantire una buona riuscita nella stampa, la dimensione minima di ogni sezione dovrà essere di 0,2 mm. Per i connettori, si consiglia un diametro di 0,5 mm. La tolleranza prevista o la precisione dimensionale è di circa ± 0,5% (± 0,15 mm).

Tabella 5: Tabella comparativa

LIMITAZIONI			TECNOLOGIE		
NOME	**DESCRIZIONE**	**MODELLO**	**FDM**	**SLA**	**SLS**
Pareti con Supporto	Pareti collegate al resto della stampa su almeno due lati.		0.8 mm	0.5 mm	0.7 mm
Parenti senza Supporto	Pareti non supportate collegate al resto della stampa su meno di due lati.		0.8 mm	1 mm	
Ssupporti e Sporgenze	Ampiezza massima di stampa di un angolo senza bisogno di supporto.		45°	Supporto sempre necessario	
Incisioni e Rilievi	Elementi al di sopra o al di sotto della superficie del prototipo.		Larghezza 0.6 mm, spessore 2 00	Larghezza e spessore 0.4 mm	Larghezza e spessore 1 mm
Ponti Orizzontali	Ampiezza massima stampabile senza l'ausilio di un supporto.		10 mm		
Fori	Diametro minimo di un foro		2 mm	5 mm	1.5 mm
Parti di Giunzione e Movibili	Spazio minimo consigliato tra due parti movibili o di giunzione.		0.5 mm	0.5 mm	0.3 mm per le parti mobili e 0.1 mm per le giunzioni

Fori di rilascio	Diametro minimo del foro di rilascio per permettere la rimozione del materiale di supporto.			4 mm	5 mm
Funzioni Minime	Dimensioni minime consigliate per favorire la buona riuscita della stampa.		2 mm	0.2 mm	0.8 mm
Diametro del Perno	Diametro minimo di un perno		3 mm	0.5 mm	0.8 mm
Ttolleranza	Livello di tolleranza previsto (accuratezza del diametro) di una specifica tecnologia.		± 0.5% (limite minimo ± 0.5 mm)	± 0.5% (limite minimo ± 0.15 mm)	± 0.3% (limite minimo ± 0.3 mm)

8. Progettazione e realizzazione del case di protezione del dispositivo IoT SWP

Questo paragrafo descrive la progettazione e la realizzazione del package di protezione per la board 1 e le board 2 e 3 che costituiscono il dispositivo IoT “Smart Work Platform”. La componentistica elettronica come specificato in modo approfondito nei paragrafi in precedenza sono pianificati sulle board in tal modo:

- **Board 1 – Unità di Controllo e Sensori Primari (Sensori di accelerazione, fiamma e corrente)**
- **Board 2 – Unità Ambientale Multi sensore collegata alla**
- **Board 3 – Estensione Gas e Composti Volatili**

È importante sottolineare che la progettazione di due case di protezione per dispositivi elettronici richiede un approccio metodico che parte dall'analisi funzionale delle schede da proteggere e arriva fino alla realizzazione fisica degli involucri tramite stampa 3D. Nel caso specifico, sono richiesti due case distinti: il primo per ospitare una singola board (Board 1), il secondo per accogliere due board collegate tra loro (Board 2 e Board 3).

8.1. Analisi preliminare e requisiti di progetto

La prima fase del processo consiste nell'analisi dettagliata delle esigenze funzionali e dimensionali delle board elettroniche da proteggere. Si raccolgono le dimensioni esatte di ciascuna scheda, la posizione e la forma dei componenti più critici (come i sensori di gas, connettori di alimentazione, porte di comunicazione, LED di stato), l'ubicazione dei fori di fissaggio e l'eventuale necessità di dissipazione del calore. In particolare, è indispensabile prevedere che i sensori di gas risultino esposti direttamente all'ambiente esterno, senza ostruzioni, per garantire misure precise. Se disponibile, si acquisisce un modello 3D delle schede in formato neutro (STEP), altrimenti si procede a una misurazione accurata manuale, eventualmente supportata da una scansione tridimensionale.

In sintesi, le informazioni da tenere in forte considerazione sono i seguenti:

- Dimensioni esatte delle board 1, 2 e 3 (inclusi ingombri di componenti sopra e sotto la PCB).
- Punti di fissaggio già presenti sulle board (fori di montaggio, socket, connettori).
- Connessioni tra board 2 e 3 (cavi, flat cable, header pin) e loro posizionamento.
- Zone critiche: per esempio, i sensori gas devono essere esposti direttamente verso l’esterno senza ostruzioni.
- Vincoli termici: attenzione al surriscaldamento dei componenti interni; prevedere aperture o griglie di ventilazione.

- Vincoli di accessibilità: porte USB, pulsanti, LED devono essere facilmente accessibili/visibili dall'esterno del case.

8.2. Creazione del modello tridimensionale della board in SolidWorks

Una volta raccolte le informazioni dimensionali, si procede alla modellazione tridimensionale delle schede in SolidWorks. Se i modelli CAD delle board non sono forniti, si ricostruiscono fedelmente i profili delle schede partendo da uno schizzo bidimensionale, riportando fedelmente tutti i riferimenti critici come fori di montaggio, ingombri massimi dei componenti e zone sensibili che devono restare libere. Si realizza poi un'estrusione dello sketch per ottenere il volume tridimensionale rappresentativo. Eventuali componenti sporgenti (connettori, sensori, pulsanti) vengono modellati come feature aggiuntive per verificare eventuali interferenze future con il case.

8.3. Progettazione del case di protezione per Board 1

La progettazione del primo involucro, destinato alla protezione della sola Board 1, comincia dalla definizione dell'ingombro esterno. Viene creata una struttura avvolgente attorno al modello della board, lasciando una tolleranza interna di circa 1,5 mm su ciascun lato per consentire il facile inserimento della scheda e compensare eventuali variazioni dovute alla stampa 3D. La struttura viene suddivisa in due parti principali: una base, che ospita la board, e un coperchio, che sigilla il dispositivo. All'interno della base si disegnano dei supporti verticali detti stand-off, con fori per il fissaggio della scheda tramite viti M2 o M3, in corrispondenza dei fori già presenti sulla PCB.
In prossimità dei sensori di gas viene modellata un'apertura precisa, calibrata per consentire il corretto scambio di gas tra interno ed esterno senza compromettere la protezione fisica. Tali aperture possono essere dotate di griglie di protezione o predisposte per l'inserimento di membrane filtranti idrofobiche, che permettono il passaggio dei gas ma proteggono contro polveri e liquidi. Vengono infine progettate opportune feritoie per favorire la ventilazione interna, importanti sia per prevenire il surriscaldamento dei componenti elettronici sia per migliorare la risposta dei sensori in tempo reale. In sintesi, gli step da tenere in considerazione sono i seguenti:

1. **Importazione o creazione del modello della board**:
    - Se esiste, importare la board come file CAD (formato STEP, IGES).
    - In alternativa, ricreare rapidamente la forma della board usando gli strumenti di Sketch e Boss-Extrude.
2. **Sketch di base**:

- Creare uno **schizzo 2D** che segua il perimetro esterno della board più un margine di tolleranza (tipicamente 1-2 mm).
- Definire l'altezza del case (considerando componenti sopra e sotto la PCB).

3. **Generazione della "scatola"**:
   - Estrudere il profilo per formare il **corpo principale** del case.
   - Creare **due parti**: il **bottom housing** (scocca inferiore) e il **top cover** (coperchio).
4. **Forature e aperture**:
   - **Creare aperture** allineate ai sensori gas per permettere la corretta esposizione.
   - Inserire **aperture di ventilazione** o **feritoie** dove necessario.
   - Predisporre **fori filettati** o incastri a scatto per la chiusura del case.
5. **Fissaggio della Board**:
   - Modellare **colonnine di supporto** (stand-off) con fori o guide per viti in corrispondenza dei fori di montaggio PCB.
6. **Interfacce esterne**:
   - Disegnare apposite aperture per connettori USB, Ethernet, alimentazione, ecc.
   - Realizzare eventuali **bottoni** o **finestrature trasparenti** per la visualizzazione di LED.

Di seguito una immagine della board 1 progettata.

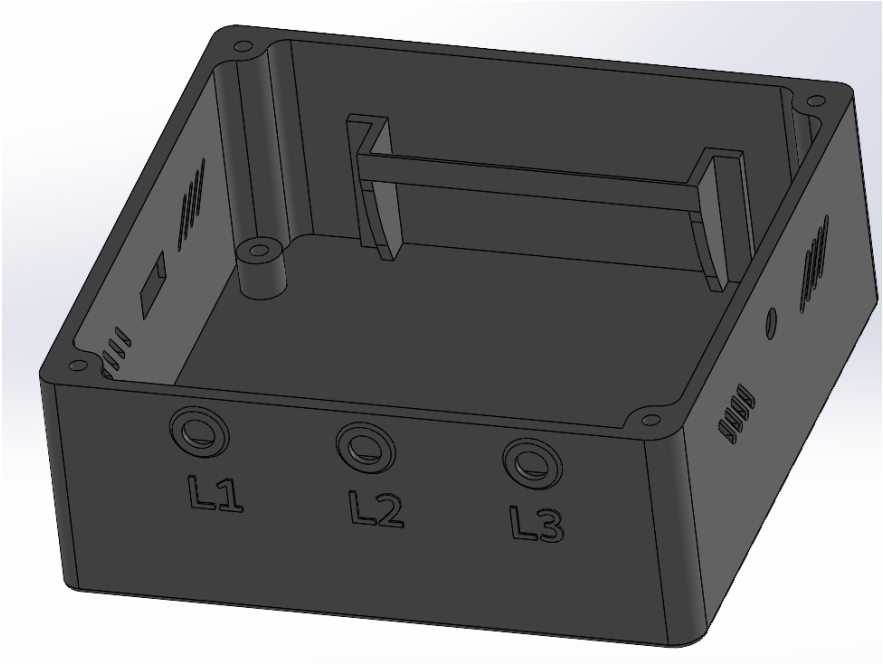


Figura 44: case per la board 1

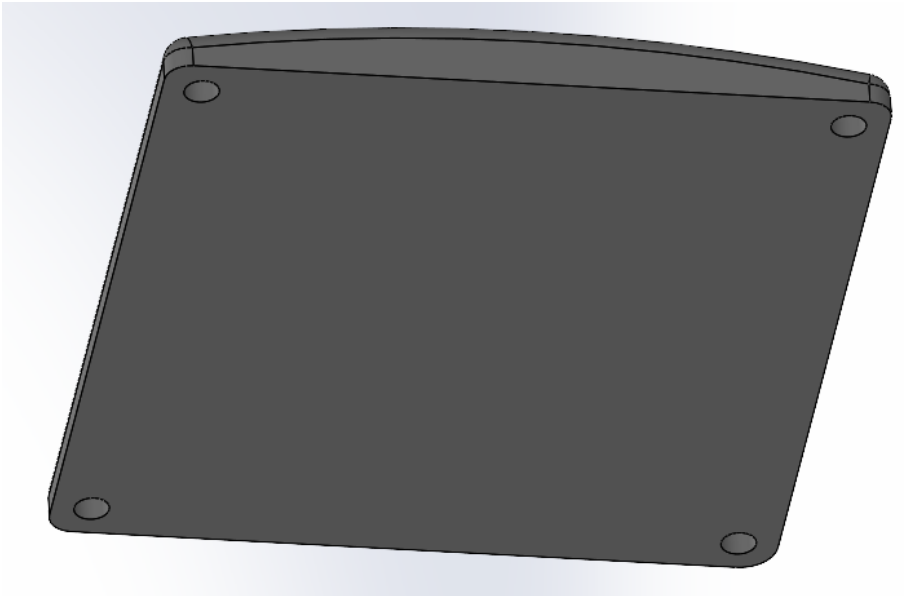


Figura 45: top del case per la board 1

8.4. Progettazione del case combinato per Board 2 e Board 3

La seconda progettazione riguarda il case che deve contenere due board collegate tra loro. Si parte definendo l'esatta disposizione spaziale: le schede possono essere montate una sopra l'altra, oppure affiancate orizzontalmente a seconda della praticità d'uso e dei vincoli dimensionali. Se si opta per una configurazione sovrapposta, si modellano due livelli di supporto: uno inferiore per Board 2 e uno superiore per Board 3, mantenendo un'altezza sufficiente per consentire il passaggio dei cablaggi senza schiacciamenti.
Si predispongono delle guide per l'alloggiamento dei cavi, progettando canaline interne o clip di fissaggio integrate, utili per evitare che i cavi interferiscano con la chiusura del case o con i flussi di ventilazione. Anche in questo caso, si creano aperture esposte per i sensori di gas, sagomate per ottimizzare l'areazione locale, e si inseriscono zone di ventilazione supplementari, considerando il carico termico complessivo delle due board operative.
Le due parti del case (base e coperchio) possono essere unite mediante viti autofilettanti su bossoli integrati, oppure con incastri a pressione, a seconda delle esigenze di manutenzione e del numero di cicli di apertura/chiusura previsti. Particolare cura viene posta nel disegnare le giunzioni tra base e coperchio per garantire un'adeguata protezione contro polvere e spruzzi d'acqua: si possono aggiungere battenti o piccoli profili per l'inserimento di guarnizioni morbide. Sulla stessa metodologia prima elencata, la progettazione segue una logica simile ma con accorgimenti ulteriori:

1. **Studio del posizionamento delle due board**:
    - Le board 2 e 3 devono essere ospitate **nello stesso case** o in **due compartimenti collegati**.
    - Predisporre uno spazio interno che garantisca il corretto alloggiamento dei cavi di connessione.
2. **Progettazione modulare**:
    - Possibile realizzare una **struttura doppia**, con una paratia interna che separi i due moduli oppure un'unica camera con guide interne.
    - Definire bene il routing dei cavi.
3. **Esposizione dei sensori gas**:
    - Inserire **aperture a griglia** o **membrane protettive** sopra i sensori gas montati su board 2 o 3.
    - Studiare flussi d'aria ottimali tramite posizionamento strategico di finestre e feritoie.
4. **Sistema di apertura**:
    - Case con apertura a **coperchio incernierato** o **smontabile** per ispezione/manutenzione.
5. **Assemblaggio e supporti**:
    - Inserire stand-off per fissaggio delle due PCB.
    - Predisporre sistemi di guida o vassoi estraibili se necessario.

Prima della stampa è opportuno per entrambi i case progettati in Solidworks eseguire i seguenti passi:

- **Simulazione di montaggio**: testare su SolidWorks il montaggio virtuale di board e componenti nel case.
- **Verifica collisioni**: usare il tool di **Interference Detection** per evitare sovrapposizioni tra PCB, cavi, e struttura.
- **Analisi tolleranze**: inserire giochi (gap) adeguati per compensare le tolleranze di stampa FDM.

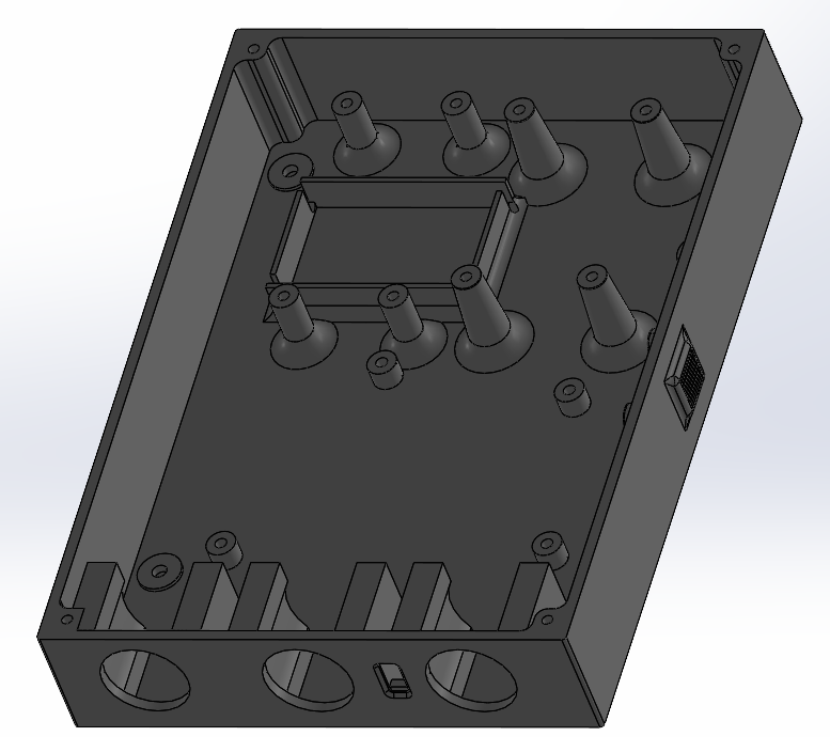


Figura 46: Case 2

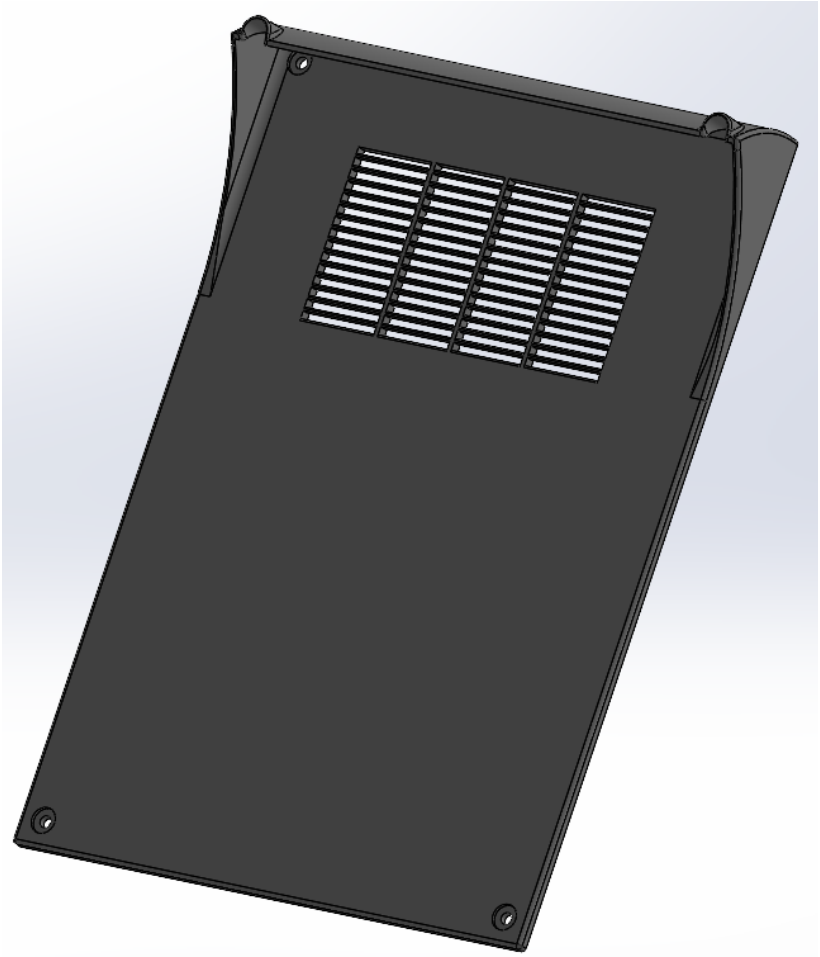


Figura 47: top del case 2

In basso una immagine della totalità del case 2.

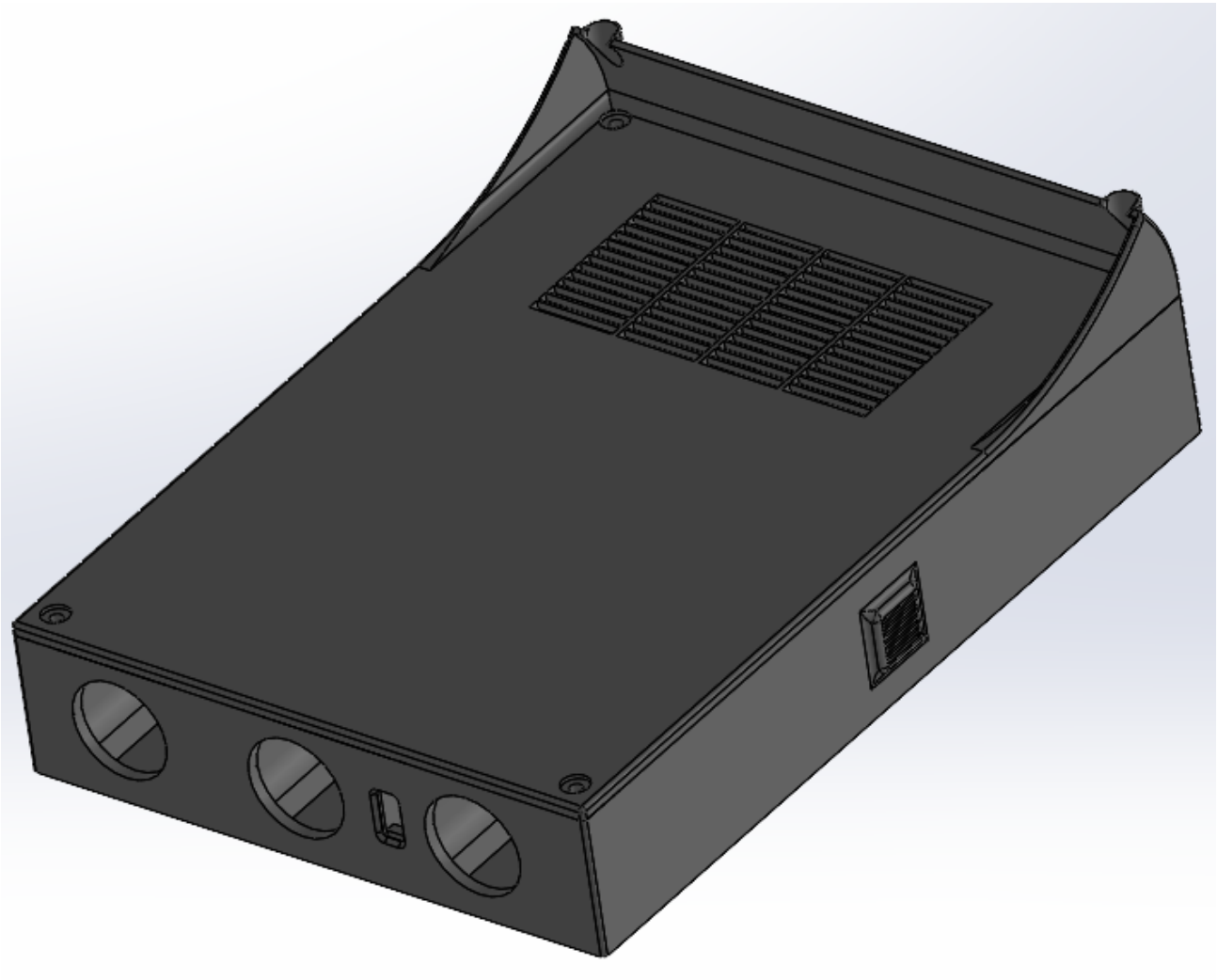


Figura 48: case 2

8.5. Preparazione del modello per la stampa 3D

Completata la modellazione in SolidWorks, si procede all'esportazione dei modelli in formato STL, scegliendo una risoluzione elevata per mantenere tutti i dettagli geometrici precisi (errori di triangolazione inferiori a 0,01 mm). Prima di passare allo slicing, si verifica la "watertightness" del file STL, ovvero si controlla che non siano presenti buchi o superfici aperte che potrebbero compromettere la stampa.

Quindi per la preparazione della stampa 3D è necessario provvedere ad:

- **Esportare** il modello CAD da SolidWorks in formato STL con risoluzione adeguata.
- **Controllare il modello STL** tramite software per verificare eventuali errori (mesh non chiuse, superfici invertite).

8.6. Impostazione della stampa 3D con tecnologia SLS

Procediamo nella descrizione del processo di realizzazione inserendo immagini e foto al fine di rendere più agevole la comprensione al lettore. Si precisa che l'impiego delle stampanti 3D prevede che l'operatore lavori in condizioni di assoluta sicurezza. Ciascun operatore/lavoratore che opera con le stampanti 3D deve osservare specifici regolamenti e normative legate alla sicurezza e ai rischi del luogo di lavoro. In tale documento, si omette la descrizione relative ai comportamenti che gli operatori devono assumere al fine di lavorare in sicurezza. Si può procedere nella descrizione delle fasi operative che portano alla sua realizzazione fisica: la scultura è stata riprodotta in scala con polvere di nylon.

Posizionamento nell'area di lavoro per la stampante EOS120

Il 'job di lavoro' viene realizzato attraverso il programma MATERIALIZE dove vengono caricati gli.stl precedentemente prodotti:

- Gli.stl vengono importati preferibilmente singolarmente e dovranno essere posizionati e scalati nell'area di lavoro della stampante riportato col programma (cubo di lavoro; trasla/ruota/scala/piano inferiore e superiore/rilevamento collisioni/analisi incatenamenti/ecc.).
- Per salvare il job si dovrà creare preliminarmente una cartella ad hoc - File – Salva tutto in directory (oppure selezionare i pezzi e salvarli come parti)

Slicing e Caricamento del progetto per la stampante EOS120

Successivamente aprire il programma RPTOOLS:

- File – Prepare STL (prendere gli.stl dalla cartella creata in precedenza) – P.slice 0.12m EOS (in questo modo creiamo file .sli) – prendiamo solo i file .sli in formato 120 EOS

Apriamo ora il tool PSW:

- File – Load parts (prendiamo e carichiamo gli .sli) – Save job as (in formato .eosjz)

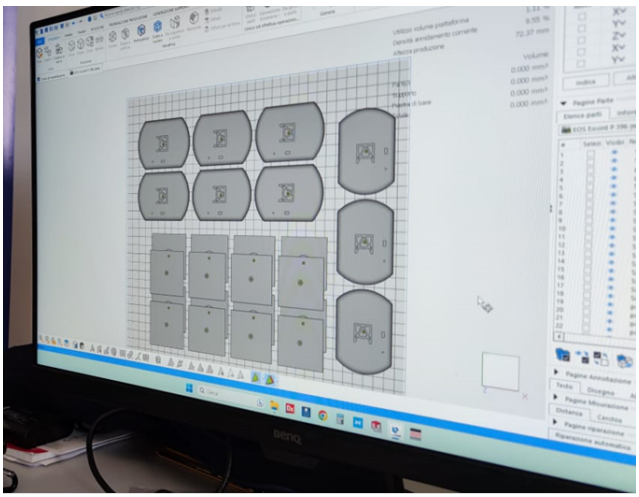


Figura 49: Slicing e caricamento del progetto

Portare la pennetta e copiare il file in formato.eosjz sul PC della stampante (PC EOS) - aprire nuovamente il PSW – File – Carica job – prendere il file .eosjz (verrà importato in automatico il file già pronto per la stampa).

Per far partire il job:

- Caricare la polvere nel re-coater col dosatore (ad esempio a destra) e portarlo a sinistra così da creare un primo strato di polvere (circa 10/20 o fin quando copre il piano)
- Play – non saltare la fase di riscaldamento – spessore 6 mm – periodo minimo di riscaldamento 120 min

ricordarsi di lavorare sempre con macchina chiusa altrimenti dà errori.

Figura 50: Stampa 3D in lavorazione

Terminata la stampa, attendere il raffreddamento della macchina (dalle 24-36 ore circa, dipende dalla grandezza e riempimento della stampa):

- portare il job nella posizione di estrazione con il tasto “posizione di estrazione del telaio intercambiabile”;
- aprire il portellone sottostante ed estrarre la box per portarla nell’unità di spacchettamento.

Figura 51: estrazione job

Nell’unità di spacchettamento, dopo aver portato lentamente la box al piano, accendere preliminarmente il tubo di estrazione dell’aria (manopola esterna blu) e stando attenti a tenere la schermata dell’unità il meno aperta possibile.

Figura 52: unità di spacchettamento job

Utilizzare gli strumenti dedicati per la pulizia preliminare dei pezzi (spazzole, ecc.) accendendo la griglia di presa della polvere collegata al bidone sottostante (da cambiare ogni qualvolta si riempie).

Figura 53: pulizia e rimozione polvere nylon non sinterizzata

Passare i pezzi nella micro-pallinatrice/sabbiatrice dove verranno perfezionati con maggiore cura:

- accendere la macchina lateralmente
- impostare una estrazione di aria tale da riuscire a lavorare con i manicotti
- inserire i pezzi lateralmente e lavorare con la pistola sabbiatrice fissa (pedale) e con la pistola aria compressa (tubo flessibile)

È stato necessario un passaggio nella burattatrice per la rifinitura/lisciatura finale dei dispositivi realizzati:

- accendere la macchina con manopola rossa PRESA B (impianto elettrico) – controllare che la pompa verde (impianto idraulico) sia inserita alla tubazione a muro e alla pompa sottostante il quadro elettrico/di lavoro – aprire l'acqua (manopola rossa in verticale)
- scaricare l'acqua dalla macchina preventivamente (leva nera con tubazione nella tanica)

- fare un giro di lavaggio della macchina in modo da bagnare anche i pellet abrasivi aprendo l'acqua con il tasto EV INGRESSO ACQUA da utilizzare in manuale (la quantità di acqua che uscirà dipende dalla leva in basso in arancione della pompa)
- operiamo la movimentazione della burattatrice con il tasto MARCIA VIBRATORE
- dopo qualche minuto spegnere l'acqua e scaricare la macchina (controllare che i pellet siano ben bagnati)
- inserire i pezzi da rifinire – inserire tempo di lavoro (bottoni di inserimento tempo) – immettere acqua come visto prima e tenere lo scarico della macchina chiuso – accendere la MARCIA VIBRATORE e regolare la sua velocità con bottone in alto REGOLAZIONE VELOCITA' - spegnere acqua se si è in manuale, altrimenti portarlo in automatico se i tempi di lavoro sono lunghi
- al termine del lavoro, scaricare l'acqua della macchina e togliere i pezzi adoperando la vibrazione

8.7. Realizzazione fisica dei case per il dispositivo SWP

La realizzazione fisica dei **case protettivi** per le diverse board che costituiscono il dispositivo **SWP** ha rappresentato una fase fondamentale per garantire **robustezza meccanica, funzionalità ergonomica e protezione ambientale** dei componenti elettronici, in particolare in contesti industriali caratterizzati da vibrazioni, polveri e variazioni termiche.

Il processo ha avuto inizio con la **modellazione tridimensionale dettagliata** delle singole schede (r, moduli sensore, interfacce, convertitori di segnale, moduli di acquisizione dati, etc.) utilizzando **software CAD parametrici** (es. SolidWorks, Fusion 360). Ogni contenitore è stato progettato in modo **customizzato**, tenendo conto di:

- **Ingombri reali e tolleranze di assemblaggio**

Ogni case è stato progettato rispettando le dimensioni fisiche effettive dei dispositivi e considerando le tolleranze di fabbricazione per garantire un assemblaggio preciso e sicuro, evitando interferenze tra le parti e assicurando un corretto accoppiamento meccanico.

- **Accessibilità ai connettori (USB, GPIO, Ethernet, HDMI, CSI, ecc.)**

Le aperture laterali dei case sono state sagomate per garantire l'accesso facile e diretto ai principali connettori delle board, consentendo un cablaggio efficiente e l'interfacciamento con dispositivi esterni senza dover smontare il contenitore.

- **Gestione del flusso d'aria e dissipazione termica passiva e/o attiva**

La conformazione interna dei case include canalizzazioni e aperture per favorire la circolazione dell'aria naturale o forzata, supportando l'integrazione di ventole o dissipatori in corrispondenza dei componenti a maggiore generazione termica (es. processori e moduli di potenza).

- **Sistema di fissaggio su pannelli o strutture portanti mediante viti, clip o rail DIN**

I case sono stati dotati di fori passanti e asole compatibili con viti standard M3/M4, clip a sgancio rapido o guide DIN industriali, per consentire un'installazione flessibile e sicura su strutture meccaniche o armadi elettrici.

- **Modularità per consentire la manutenzione, la sostituzione dei componenti o l'espansione futura**

Ogni case è stato progettato in modo modulare e indipendente, permettendo di sostituire una singola board o di aggiungere nuovi moduli senza dover smontare l'intero sistema, agevolando gli interventi di manutenzione o aggiornamento.

- **Canalizzazione ordinata dei cablaggi interni** mediante feritoie, passacavi e guide integrate

Sono state integrate nel design del case guide per il passaggio ordinato dei cavi e slot per fascette o clip ferma-cavo, per mantenere una disposizione interna pulita e minimizzare rischi di interferenza elettromagnetica e surriscaldamento.

- **Alloggiamenti per ventole o dissipatori passivi** nei punti critici di generazione del calore

I case includono predisposizioni geometriche per montare dissipatori passivi o ventole da 40 mm o 60 mm nei punti di maggiore carico termico. In alcuni prototipi è stato testato anche l'utilizzo di pad termici e griglie protettive.
Le pareti dei case includono aperture o supporti interni per alloggiare e sostituire rapidamente schede SD o SIM, fondamentali per le funzionalità di memoria espandibile o connettività mobile del sistema.
I case sono stati quindi realizzati con tecnologia **Sinterizzazione Laser Selettiva SLS** utilizzando materiali plastici **ad alte prestazioni.** Il processo ha previsto diverse iterazioni tra progettazione e stampa per ottimizzare la **precisione degli incastri**, il **raffreddamento interno** e la **stabilità meccanica**, con successivi test di **fit-check** sulle schede elettroniche reali.

Le immagini in basso documentano il **montaggio completo con le board installate e cablate**. Tali enclosure rappresentano un compromesso ideale tra **funzionalità ingegneristica, durabilità meccanica ed estetica tecnica**, fondamentali per l'adozione del sistema in ambienti industriali e dimostrativi.

In basso sono mostrati i case per la board 1 ed il case che ingloba la board 2 e 3.

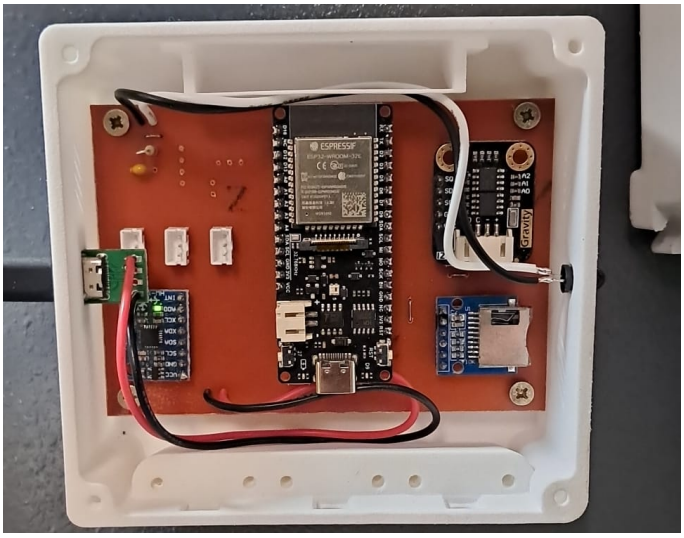


Figura 54: case per la board 1

Figura 55: Case per la board 2 e 3

9. Implementazione e realizzazione del firmware ed integrazione con il sistema hardware

Parallelamente alla progettazione elettronica, è stato sviluppato il firmware necessario per il corretto funzionamento del dispositivo, garantendo l'integrazione e la gestione efficiente dei sensori e dei moduli di comunicazione. Il firmware è stato scritto e ottimizzato per assicurare prestazioni elevate, stabilità operativa e compatibilità con le esigenze del progetto. L'attività di implementazione e integrazione dell'hardware e del firmware è stata completata con successo, senza riscontrare anomalie, garantendo il pieno rispetto delle specifiche progettuali. Sulla base dei risultati ottenuti nelle fasi precedenti di progettazione e sviluppo, il team ha eseguito l'integrazione tra i componenti hardware e il firmware, assicurando la corretta interazione tra i sensori, i moduli di comunicazione e il sistema di elaborazione del dispositivo IoT.
Particolare attenzione è stata dedicata alla verifica della compatibilità tra i vari elementi del sistema, al fine di ottimizzare le prestazioni e garantire un funzionamento stabile ed efficiente. Sono stati condotti test per validare l'interfacciamento tra le componenti elettroniche e il software embedded, assicurando la corretta acquisizione, elaborazione e trasmissione dei dati in tempo reale.
In parallelo, è stata valutata e successivamente implementata la realizzazione di un supporto meccanico dedicato, progettato per vincolare in modo sicuro il dispositivo IoT alla macchina da monitorare. Questa soluzione ha permesso di garantire un'installazione stabile e affidabile, minimizzando eventuali vibrazioni o interferenze che avrebbero potuto compromettere l'accuratezza delle rilevazioni.
Di seguito si riporta la documentazione relativa all'implementazione del firmware con tutte le relative librerie.

Libreria S4NTO_SWP

La libreria **S4NTO_SWP** è stata sviluppata per gestire in modo integrato le operazioni di:

- **Configurazione e gestione dei sensori**
- **Comunicazione tramite BLE e WiFi/MQTT**
- **Gestione della memoria** (incluso un database simulato basato su un Ring Buffer su SPIFFS)
- **Gestione dei certificati SSL** e della configurazione del dispositivo

Questa libreria è progettata per dispositivi basati su ESP32 e utilizza il framework Arduino.

9.1. Flusso di Lavoro

9.1.1. Inizializzazione (begin())

La funzione begin() del controller deve essere invocata all'avvio e svolge le seguenti operazioni:

- Inizializza il **LED RGB**.
- Inizializza l'**interrupt** per abilitare/disabilitare il BLE.
- Inizializza i **sensori**.
- Inizializza il **clock RTC**.
- Inizializza il **controller di memoria**. In caso di fallimento, viene invocata una funzione di errore bloccante.
- Disabilita i sensori all'avvio, se specificato nella configurazione (tramite un apposito file di disabilitazione).
- Carica i **certificati SSL** e ne verifica lo stato.
- Carica la **configurazione del dispositivo** dalla memoria.
- Inizializza il **controller BLE**.
- Inizializza il controller che gestisce i **Record** (database simulato tramite un Ring Buffer su SPIFFS).
- Inizializza gli **handler dei protocolli di comunicazione** (attualmente solo MQTT).
- Inizializza il **controller WiFi**.
- Verifica la configurazione corrente: se la configurazione prevede l'uso di WiFi o LTE (non implementato), testa la connessione di rete e, se ha successo, avvia il protocollo scelto (solo MQTT implementato).

9.1.2. Ciclo Operativo (update())

La funzione update() deve essere invocata nel loop() di Arduino. Essa gestisce:

- Il toggling del BLE, abilitando o disabilitando il modulo BLE in base a un flag di interrupt.
- Il **polling** del BLE quando attivo.
- La gestione della **comunicazione di rete** (es. connessione e aggiornamento MQTT) se il BLE non è attivo.
- L'invocazione delle funzioni specifiche per la gestione dei sensori, il controllo delle configurazioni e l'invio dei dati.

9.1.3. API e Funzionalità

9.1.4. Inizializzazione e Aggiornamento

- **begin()**
  Inizializza l'intero sistema, includendo:
  - LED RGB, interrupt per BLE e sensori.
  - RTC e controller di memoria.
  - Caricamento dei certificati SSL e della configurazione del dispositivo.
  - Inizializzazione dei controller BLE, del database (Ring Buffer) e dei protocolli di comunicazione (MQTT).
  - Inizializzazione del controller WiFi e verifica della connessione di rete (se previsto).
- **update()**
  Gestisce dinamicamente il BLE e la comunicazione di rete:
  - Alterna l'attivazione/disattivazione del BLE in base a un interrupt.
  - Esegue il polling BLE e aggiorna la connessione di rete (MQTT).
  - Invoca le funzioni specifiche per la gestione dei sensori e per l'invio dei dati.
- **checkHeapMemory()**
  Funzione di debug per ottenere dettagli sulla memoria disponibile.
- **error()**
  Funzione bloccante che gestisce gli errori critici, impostando pattern di blink e colori differenti sul LED RGB.

9.1.5. Gestione della Connettività

- **handleWiFi()**
  Avvia una connessione di rete via WiFi:
  - Utilizza le informazioni in memoria per connettersi.
  - Aggiorna la data corrente e imposta il RTC.
  - Restituisce lo stato della connessione.
- **handleLTE()**
  Simile a handleWiFi(), ma attualmente non implementata.
- **handleConnection()**
  Gestisce i protocolli di comunicazione:
  - Termina eventuali connessioni precedenti.
  - Configura il callback per MQTT e, se disponibili, i certificati (CA, client e chiave).
  - Avvia la comunicazione con il broker MQTT e sottoscrive i sensori abilitati a topic specifici.

- **handleMQTTsubscription()**
  Gestisce l'iscrizione o disiscrizione dal topic di un sensore a runtime, a seconda dello stato abilitato/disabilitato.
- **`mqttCallback()`**
  Gestisce i topic MQTT e invoca il `dataHandler` per validare e aggiornare la configurazione ricevuta.

9.1.6. Gestione BLE e Sensori

- **initBLE()**
  Imposta i callback necessari per la gestione delle operazioni BLE.
- **enableSensor() / disableSensor()**
  Abilitano o disabilitano un sensore a runtime, gestendo la sottoscrizione al topic del sensore se richiesto dalla configurazione. Possono essere invocate anche senza passare per begin().
- **handleSensorBehavior()**
  Gestisce il comportamento dei sensori all'avvio, invocando il controller di memoria per abilitare/disabilitare i sensori designati.
- **createSensor()**
  Crea un'istanza del sensore in base alla configurazione ricevuta (standard o specializzato).
- **initializeSensor()**
  Recupera la configurazione di un sensore, crea l'istanza, abilita il debug se necessario ed esegue il begin() del sensore. Può essere invocata indipendentemente dal begin() globale.
- **initializeAllSensors()**
  Inizializza tutti i sensori noti, indipendentemente dal begin() globale.
- **popValue() / popValues()**
  - **popValue()**: Recupera il valore (o i valori) di un sensore specifico. Non è invocabile se il BLE è attivo.
  - **popValues()**: Recupera i valori di tutti i sensori noti.

9.1.7. Gestione della Memoria e dei Certificati

- **uploadCertificate()**
  Può essere invocata standalone o via BLE per scrivere un certificato in memoria. Dopo il caricamento,

eventuali connessioni esistenti vengono terminate e ne viene avviata una nuova. Il formato dei certificati può essere ridefinito tramite macro.

- **checkCertificate()**
  Verifica la correttezza formale del certificato.
- **uploadConfiguration()**
  Può essere invocata standalone o via BLE per aggiornare la configurazione del dispositivo e della rete. Se il protocollo di comunicazione viene specificato, vengono verificate le configurazioni e, se corrette, vengono stabilite le connessioni con la rete e con il broker/backend.
- **getConfiguration()**
  Restituisce la configurazione corrente del dispositivo.
- **getSensorConfiguration()**
  Restituisce la configurazione di un sensore specifico.
- **flushMemory()**
  Cancella l'intero contenuto della memoria o un singolo file (ad es. configurazione, protocollo, certificato, ecc.). Richiede il riavvio automatico del dispositivo alla fine delle operazioni.

9.1.8. Gestione dei Dati dei Sensori

- **sendData()**
  Funzione specifica per MQTT che:
  - Recupera i record memorizzati (tramite un Ring Buffer su SPIFFS).
  - Incapsula i record in un oggetto JSON.
  - Invia il pacchetto al broker MQTT.
  - In caso di successo, cancella i dati inviati dalla memoria.
- **processMQTTconfiguration()**
  Gestisce la configurazione ricevuta via MQTT, validandola e aggiornando il dataHandler.

9.1.9. Reset della Configurazione e dei Dati Precedenti

Quando vengono inviate nuove configurazioni via BLE, la libreria offre la possibilità di **resettare i dati precedenti**. Questo processo:

- Cancella i record memorizzati per ogni sensore (tramite il metodo flush()).

- Rimuove le configurazioni correnti, inclusi eventuali timestamp di poll e trasmissione (es. utilizzando LAST_POLL.clear() e LAST_TX.clear()).

Esempio:

```
// Quando si ricevono nuove configurazioni via BLE:
dataHandler.reset();  // Resetta configurazioni e dati memorizzati

// Inoltre, per gestire poll e TX:
LAST_POLL.clear();
LAST_TX.clear();
```

9.2. Configurazione Personalizzata delle Partizioni su ESP32

1. Creazione della Partizione Personalizzata
Per assegnare 3MB all'applicazione e 8MB a SPIFFS, occorre creare un file CSV personalizzato per la mappa delle partizioni.

Percorso della cartella delle partizioni su sistemi `Windows`:
-
`C:\Users\`**username**`\AppData\Local\Arduino15\packages\esp32\hardware\esp32\`**3.0.7**`\tools\partitions`

File CSV `large_spiffs_8MB_s4nto`
```
# Name,   Type, SubType, Offset,  Size, Flags
nvs,      data, nvs,     36K,     20K,
otadata,  data, ota,     56K,     8K,
app0,     app,  ota_0,   64K,     3M,
spiffs,   data, spiffs,  ,        8M,
coredump, data, coredump,,        64K,
```

per utilizzare il tool `ESP32 Sketch Data Upload` occorre definire gli indirizzi in formato esadecimale:
```
# Name,   Type, SubType, Offset,   Size,   Flags
nvs,      data, nvs,     0x9000,   0x5000,
otadata,  data, ota,     0xe000,   0x2000,
app0,     app,  ota_0,   0x10000,  0x300000,
spiffs,   data, spiffs,  0x310000, 0x800000,
coredump, data, coredump,0xB10000, 0x10000,
```

``nvs``

Offset: 0x9000

In decimale: 0x9000 = 36 × 1024 = 36K (36 × 1024 = 36.864 byte)
Size: 0x5000
In decimale: 0x5000 = 20K (20 × 1024 = 20.480 byte)
La partizione nvs occupa i byte da 0x9000 fino a 0x9000 + 0x5000, cioè fino a 0xe000.

``otadata``

Offset: 0xe000 (che corrisponde a 56K, poiché 0xe000 = 57.344 byte, approssimativamente 56K se si considerano gli standard usati nelle partizioni ESP32)
Size: 0x2000
In decimale: 0x2000 = 8K (8 × 1024 = 8.192 byte)
La partizione otadata va da 0xe000 fino a 0xe000 + 0x2000 = 0x10000.

``app0``

Offset: 0x10000
In decimale: 0x10000 = 64K (64 × 1024 = 65.536 byte)
Size: 0x300000
In decimale: 0x300000 = 3M (3.145.728 byte)
Quindi, la partizione app0 va da 0x10000 fino a 0x10000 + 0x300000 = 0x310000.

``spiffs``

Offset: 0x310000
Questo viene calcolato perché spiffs deve iniziare subito dopo la fine di app0.
Poiché app0 termina a 0x310000, questo diventa l'offset di partenza per spiffs.
Size: 0x800000
In decimale: 0x800000 = 8M (8.388.608 byte)

``coredump``

Offset: 0xB10000

Questo offset è calcolato in base alla fine della partizione spiffs. Infatti, spiffs inizia a 0x310000 e occupa 0x800000 byte; quindi termina a 0x310000 + 0x800000 = 0xB10000. Questo diventa l'offset di partenza per la partizione coredump.

Size: 0x10000

In decimale: 0x10000 = 64K (64 × 1024 = 65.536 byte)

La partizione coredump occupa i byte da 0xB10000 fino a 0xB10000 + 0x10000, cioè un totale di 64KB.

[CSV](./docs/large_spiffs_8MB_s4nto.csv)
[ESPRESSIF](./docs/espressif.md)

2. Aggiunta della Configurazione Personalizzata nel File `boards.txt`
Per rendere disponibile la nuova partizione nell'IDE di Arduino, occorre modificare il file `boards.txt`.

Percorso del file boards.txt su sistemi `Windows`:
- `C:\Users\`**username**`\AppData\Local\Arduino15\packages\esp32\hardware\esp32\`**3.0.7**

Modifica del file (aggiunta delle seguenti righe)
- `dfrobot_firebeetle2_esp32e.menu.PartitionScheme.huge_app_ext=LARGE SPIFFS S4NTO (3MB No OTA/8MB SPIFFS)`
- `dfrobot_firebeetle2_esp32e.menu.PartitionScheme.huge_app_ext.build.partitions=large_spiffs_8MB_s4nto`
- `dfrobot_firebeetle2_esp32e.menu.PartitionScheme.huge_app_ext.upload.maximum_size=3145728`

3. Risoluzione del Bug dell'IDE Arduino
Dopo aver modificato il file `boards.txt`, Arduino IDE non rileva le nuove partizioni a causa di un problema con la cache della piattaforma.

Occorre eliminare (effettuare prima un backup, per sicurezza) la cartella di cache dell'IDE (**arduino-ide**) presente al percorso:
- `C:\Users\`**username**`\AppData\Roaming\arduino-ide`

Dopo aver eliminato questa cartella, è sufficiente riavviare Arduino IDE per ricaricare correttamente le partizioni.

Parametri di configurazione della board ESP32-WROOM-32E (D0WDR2-V3)
```
CPU Frequency: "240MHz (WiFi/BT)"
Flash Frequency: "40MHz"
Flash Mode: DIO"
Flash Size: "16MB (128Mb)"
Partition Scheme: "LARGE SPIFFS S4NTO (3MB No OTA/8MB SPIFFS)"
PSRAM: "Enabled"
Upload Speed: "921600"
```

9.3. Configurazione ed Esempi di Dati Trasmessi

Il presente paragrafodescrive in modo sintetico i seguenti step:

- Esempio di configurazione e trasmissione dati
- Esempio per un sensore che restituisce in output
    - un valore
    - Due valori
    - Nove valori

Esempio di configurazione e trasmissione dati dei sensori

Di seguito si descrive come configurare e ricevere i dati da sensori collegati a una scheda identificata come **SWP001** tramite il protocollo **MQTT**, un protocollo di messaggistica leggero molto usato in ambito IoT.

Config per topic sulla board con identificativo SWP001 per sensore con id N

mosquitto_pub -h "your_host" -p "your_port" -t " SWP001/N" -m "your_config"

per quanto riguarda “your_config” ecco un esempio:

{
"txTimeGlobal": 60000,
"txTime": 0,
"pollingTime": 5000,
"maxSamples": 30,
"outOfRange": false,
"thresholdMin": 0.0,
"thresholdMax": 0.0

}

- txTime txTimeGlobal: se entrambi pari a 0, la trasmissione dei dati viene interrotta per il sensore di interesse.
- txTime: valutato se e solo se diverso da 0. Utilizzato come parametro di configurazione temporale specifica, per il sensore di interesse.
- maxSamples: se pari a 0 o superiore alle capacità del buffer circolare, assume un valore di default (definito nella relativa classe di implementazione).
- outOfRange: se true, abilita la valutazione delle soglie thresholdMin e thresholdMax per i sensori che restituiscono un singolo valore ad ogni poll.

Sottoscrivere al topic della board con identificativo SPW001 per il sensore con ID N:
mosquitto_sub -h "your_host" -p "your_port" -t " SPW001 /N"
Fatto questo si ottengono i dati dal sensore con id N.
Esempio di dati ottenuti con una configurazione

- txTime:10
- maxSamples:3

Sensori che forniscono in output un valore

Di seguito si descrive il formato dei dati che vengono restituiti da un sensore che fornisce un valore misurato. In questo caso, il sensore è un "Amperometrico" (sensore di corrente) e l'output è strutturato in un array di oggetti.

Ecco una spiegazione dettagliata:

1. **ID del sensore**: Il sensore descritto è identificato dal numero **7**. In generale, ogni sensore nel sistema è associato a un ID univoco.
2. **Formato dei dati (JSON)**: L'output del sensore è in formato JSON, che è un formato di scambio dati molto comune. In questo caso, l'oggetto JSON contiene una chiave "data" che è un array (cioè una lista ordinata di elementi).

Per il sensore con ID 7 che corrisponde al sensore Amperometrico (sensore di corrente) l'output è il seguente:
{

"data": [{ "Timestamp": "t1", "Sample": “A1”}, {"Timestamp": "t2", "Sample": “A2”}, {"Timestamp": "t3", "Sample":”A3”}]
}

In sintesi, il sensore con ID 7 restituisce un array di dati, ognuno dei quali contiene un timestamp e il valore (campione) della corrente misurata in quel momento. Ogni coppia "timestamp - valore" fornisce un'istantanea della lettura del sensore in un determinato punto temporale.

Sensori che forniscono in output due valori

Questo paragrafo descrive come vengono gestiti i sensori che restituiscono più parametri di misura, come nel caso di un sensore di CO2 (ID 11) che fornisce anche la temperatura e l'umidità relativa.

La configurazione di esempio indica che la trasmissione dei dati è impostata con:

- **txTime**: 10, il che implica che i dati verranno trasmessi ogni 10 unità di tempo.
- **maxSamples**: 3, il che significa che verranno acquisiti e trasmessi solo 3 campioni per parametro.

L'output restituito dal sensore per ogni campione contiene tre valori separati:

1. **CO2 in ppm** (parti per milione, la concentrazione di CO2).
2. **Temp in °C** (la temperatura in gradi Celsius).
3. **Umidità in %** (la percentuale di umidità relativa).

Per ogni timestamp (t1, t2, t3), i dati relativi a CO2, temperatura e umidità sono registrati e associati con il relativo valore di ciascun parametro. Questo significa che ogni campione trasmesso dal sensore fornirà una tripla di valori (CO2, Temp, Umidità) con il relativo timestamp, che indica quando è stato registrato il dato.

In sintesi, il sensore restituisce tre parametri (CO2, temperatura, umidità) in ogni campione, e per ogni timestamp vengono raccolti questi tre valori.

Nel caso di un sensore che restituisce più parametri ad esempio CO2 (ID 11), temperatura e umidità relativa. Considerando la seguente configurazione:

- txTime:10
- maxSamples:3

Otteniamo

{

"data": [{"Timestamp": "t1","Sample": “CO2 in ppm”}, { "Timestamp": "t1","Sample": “Temp in °C”},

{ "Timestamp": "t1", "Sample": “Umidità in %” }

{"Timestamp": "t2","Sample": “CO2 in ppm”}, { "Timestamp": "t2","Sample": “Temp in °C”},

{ "Timestamp": "t2", "Sample": “Umidità in %” }

{"Timestamp": "t3","Sample": “CO2 in ppm”}, { "Timestamp": "t3","Sample": “Temp in °C”},

{ "Timestamp": "t3", "Sample": “Umidità in %” }]

}

Sensori che forniscono in output nove valori

Di seguito si descrive il comportamento dei sensori che producono in uscita un insieme di nove valori per ciascun campionamento, come accade nel caso di un sensore IMU (Inertial Measurement Unit). Questo tipo di sensore restituisce un pacchetto complesso di informazioni che comprende dati relativi all'orientamento (pitch e roll), temperatura, accelerazioni lineari lungo i tre assi (X, Y, Z) e accelerazioni angolari lungo gli stessi assi. Quando il sensore viene configurato per inviare i dati, ad esempio tramite protocollo MQTT su un topic specifico, la trasmissione produce una sequenza di tuple, ciascuna delle quali contiene tutti e nove i parametri sopra elencati. Il numero di tuple inviate è determinato dal parametro `maxSamples` definito nella configurazione, rappresentando il numero di campionamenti consecutivi acquisiti e trasmessi. Ogni tupla è associata a un timestamp che permette di identificare il momento esatto della rilevazione, abilitando così una tracciabilità temporale dei dati utile in fase di analisi o correlazione con altri eventi o misurazioni. In pratica, il sensore fornisce una fotografia completa e sincronizzata delle sue nove misure per ogni istante in cui effettua una trasmissione.

Quindi per sensori con più parametri si ottengono n tuple di valori.

Se prendiamo il sensore IMU che legge

- Pitch (numero di rotazioni lungo l'asse Y)
- Roll (numero di rotazioni lungo l'asse X)
- Temperatura (in °C)
- Accelerazione X (in metri al secondo^2)
- Accelerazione Y
- Accelerazione Z
- Accelerazione angolare X (in gradi al secondo^2)
- Accelerazione angolare Y
- Accelerazione angolare Z

Per ogni dato inviato al topic del sensore otteniamo N tuble(maxSamples) contenenti tutti i valori precedentemente elencanti.

Configurazione di Base

```
{
  "txTimeGlobal": 60000,
  "txTime": 30000,
  "pollingTime": 5000,
  "maxSamples": 0,
  "outOfRange": false,
  "thresholdMin": 0.0,
  "thresholdMax": 0.0
}
```

```
{
    "txTimeGlobal": 60000,
    "txTime": 0,
    "pollingTime": 5000,
    "maxSamples": 30,
    "outOfRange": false,
    "thresholdMin": 0.0,
    "thresholdMax": 0.0
}
```

Regole di Valutazione:

- **txTime** **txTimeGlobal**: se entrambi pari a 0, la trasmissione dei dati viene interrotta.
- **txTime**: valutato se e solo se diverso da 0.
- **maxSamples**: se pari a 0 o superiore alle capacità del buffer circolare, assume un valore di default.
- **outOfRange**: se `true`, abilita la valutazione delle soglie **thresholdMin** e **thresholdMax** per i sensori che restituiscono un singolo valore ad ogni poll.

Dati Sensori Standard

```
{
  "data": [
    {
      "Timestamp": "2025-03-04 12:17:31",
      "Sample": 0.78323
    },
    {
      "Timestamp": "2025-03-04 12:17:36",
      "Sample": 0.782525
    },
    {
      "Timestamp": "2025-03-04 12:17:41",
      "Sample": 0.783029
    }
  ]
}
```


Dati Sensori (2 Valori restituiti)


```
{
 "data": [
   {
     "Timestamp": "2025-03-04 12:23:06",
     "Sample": 0
   },
   {
     "Timestamp": "2025-03-04 12:23:06",
     "Sample": 23.87407
   },
   {
     "Timestamp": "2025-03-04 12:23:11",
     "Sample": 0
   },
```

```
    {
     "Timestamp": "2025-03-04 12:23:11",
     "Sample": 23.9761
    },
    {
     "Timestamp": "2025-03-04 12:23:17",
     "Sample": 0
    },
    {
     "Timestamp": "2025-03-04 12:23:17",
     "Sample": 23.87407
    }
  ]
}
```


Dati Sensori (9 Valori restituiti)

Pitch	Roll	Temperature	Accelerometer	Gyroscope
---	---	---	x,y,z	x,y,z


```
{
  "data": [
    {
     "Timestamp": "2025-03-04 12:37:28",
     "Sample": -2
    },
    {
     "Timestamp": "2025-03-04 12:37:28",
     "Sample": 0
    },
    {
     "Timestamp": "2025-03-04 12:37:28",
```

```
 "Sample": 24.05941
},
{
 "Timestamp": "2025-03-04 12:37:28",
 "Sample": 0.459422
},
{
 "Timestamp": "2025-03-04 12:37:28",
 "Sample": -0.143569
},
{
 "Timestamp": "2025-03-04 12:37:28",
 "Sample": 9.188438
},
{
 "Timestamp": "2025-03-04 12:37:28",
 "Sample": 4.320561
},
{
 "Timestamp": "2025-03-04 12:37:28",
 "Sample": 5.282382
},
{
 "Timestamp": "2025-03-04 12:37:28",
 "Sample": -0.809151
},
{
 "Timestamp": "2025-03-04 12:37:33",
 "Sample": -2
},
{
 "Timestamp": "2025-03-04 12:37:33",
 "Sample": -1
},
```

```
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": 23.96529
    },
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": 0.394816
    },
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": -0.196211
    },
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": 9.207582
    },
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": 4.457964
    },
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": 5.251848
    },
    {
     "Timestamp": "2025-03-04 12:37:33",
     "Sample": -0.778617
    }
  ]
}
```


Il **topic MQTT** al quale inviare un messaggio publish è definito dal nome assegnato al dispositivo in fase di configurazione, e dall'ID del sensore (presente nella configurazione fisica della board). La configurazione del

dispositivo viene mantenuta anche in seguito a riavvii o interruzioni di alimentazione, ma è necessario eseguire nuovamente un publish al topic di interesse per avviare nuovamente la ricezione dei dati (**publish del sensore su broker**).
Ad esempio, per attivare la trasmissione dei dati dal sensore con ID **1** o **10** della board **firebeetle**, i topic MQTT da utilizzare saranno:
firebeetle/1
firebeetle/10
L'elenco completo degli ID dei sensori è disponibile nella sezione successiva.

Lista Sensori e Valori restituiti

```cpp
enum SensorID {
   SENSOR_ID_AMBIENT_LIGHT        = 1,
   SENSOR_ID_FLAME_ALARM          = 2,
   SENSOR_ID_VOC                  = 3,
   SENSOR_ID_NITROGEN_DIOXIDE_NO2  = 4,
   SENSOR_ID_AMMONIA_NH3          = 5,
   SENSOR_ID_METHANE_CH4          = 6,
   SENSOR_ID_AC_CURRENT_1         = 7,
   SENSOR_ID_AC_CURRENT_2         = 8,
   SENSOR_ID_AC_CURRENT_3         = 9,
   SENSOR_ID_PM                   = 10,
   SENSOR_ID_CO2                  = 11,
   SENSOR_ID_MEMS                 = 12,
   SENSOR_ID_GAS_CO               = 13,
   SENSOR_ID_GAS_O2               = 14,
   SENSOR_ID_GAS_O3               = 15
};
```

ID	Sensore	Valori
1	SENSOR_ID_AMBIENT_LIGHT	Volt
2	SENSOR_ID_FLAME_ALARM	Volt
3	SENSOR_ID_VOC	Volt
4	SENSOR_ID_NITROGEN_DIOXIDE_NO2	Volt
5	SENSOR_ID_AMMONIA_NH3	Volt
6	SENSOR_ID_METHANE_CH4	Volt
7	SENSOR_ID_AC_CURRENT_1	Ampere
8	SENSOR_ID_AC_CURRENT_2	Ampere
9	SENSOR_ID_AC_CURRENT_3	Ampere
10	SENSOR_ID_PM	PM 1.0, 2.5, 10
11	SENSOR_ID_CO2	Media CO2, Media Temperatura, Media Umidità Relativa
12	SENSOR_ID_MEMS	Vedi esempio
13	SENSOR_ID_GAS_CO	Concentrazione Gas, Temperatura
14	SENSOR_ID_GAS_O2	Concentrazione Gas, Temperatura
15	SENSOR_ID_GAS_O3	Concentrazione Gas, Temperatura

Esempi di utilizzo con Mosquitto
Modalità NON crittografata
Pubblicazione di una richiesta al sensore 1 della board firebeetle

mosquitto_pub -h "your_host" -p "your_port" -t "firebeetle/1" -m "your_config"

Sottoscrizione di un topic per ricevere i dati dal sensore 1 della board firebeetle

mosquitto_sub -h "your_host" -p "your_port" -t "firebeetle/1"

Modalità crittografata (TLS/SSL)

Pubblicazione di una richiesta al sensore 1 della board firebeetle

mosquitto_pub -h "your_host" -p "your_port" --cafile "your_ca.crt" --cert "your_client.crt" --key "your_client.key" -t "firebeetle/1" -m "your_config"

Sottoscrizione di un topic per ricevere i dati dal sensore 1 della board firebeetle

mosquitto_sub -h "your_host" -p "your_port" --cafile "your_ca.crt" --cert "your_client.crt" --key "your_client.key" -t "firebeetle/1"

Nota: se si desidera utilizzare un broker di test per questo progetto, è possibile raggiungerlo al seguente indirizzo:

smartworkplatform.duckdns.org

porta 23191 per connessioni non autenticate e non crittografate;

porta 23190 per connessioni crittografate.

Ulteriori chiarimenti:

Il presente servizio non è parte integrante del progetto (è utilizzato internamente per scopi di R&D), per cui potrebbe non essere più disponibile (temporaneamente o definitivamente).

Per le connessioni crittografate occorre contattare l'admin di progetto al fine di ottenere i certificati necessari. Mosquitto offre broker di test che possono essere utilizzati secondo esigenze specifiche (Hello MQTT World). L'utilizzo di **Mosquitto** quale broker mqtt non è vincolante; è possibile utilizzare il broker mqtt che meglio si adatta alle proprie esigenze.

9.4. Sezione Utilities

Gestione dei Sensori

Il file di intestazione (Sensors.h) definisce e gestisce un insieme di sensori utilizzabili su un microcontrollore FireBeetle 2. Il codice include identificatori, configurazioni hardware e informazioni descrittive sui sensori.

Struttura del codice

1. Definizione dei Sensori (SensorID)

I sensori sono identificati da un enum SensorID, che assegna un ID univoco a ciascun sensore.
Esempi di sensori supportati:

- **SENSOR_ID_AMBIENT_LIGHT** → Sensore di luce ambientale
- **SENSOR_ID_FLAME_ALARM** → Sensore di fiamma
- **SENSOR_ID_VOC** → Sensore di composti organici volatili
- **SENSOR_ID_CO2** → Sensore a infrarossi per CO2
- **SENSOR_ID_MEMS** → Sensore MEMS a 6 assi (giroscopio + accelerometro)

2. Mappatura dei Pin (SensorPIN)

L'enum SensorPIN assegna un pin specifico a ciascun sensore. Se un sensore non richiede un pin specifico (es. I2C), viene assegnato **SENSOR_PIN_NONE** (-1).

3. Tipologie di Sensori (SensorType)

L'enum SensorType classifica i sensori in base alla loro funzione.

4. Struttura SensorInfo

Questa struttura contiene informazioni testuali sui sensori:

```
struct SensorInfo {
   std::string name;
   std::string description;
};
```

Esempio di mappatura SensorID → SensorInfo:

```
{
   SENSOR_ID_AMBIENT_LIGHT,
   {"Ambient Light", "Detects ambient light levels."}
}
```

5. Struttura Sensor

La struttura Sensor rappresenta un sensore e si compone di id, pin, tipo e descrizione:

```
struct Sensor {
   SensorID id;
   SensorPIN pin;
   SensorType type;
   SensorInfo info;
};
```

6. sensorInfoMap e sensorConfigMap

Due mappe permettono di accedere rapidamente alle informazioni sui sensori:

- sensorInfoMap → Associa un SensorID con il nome e la descrizione del sensore.
- sensorConfigMap → Associa un SensorID con il SensorPIN e il SensorType.

Possibili Estensioni

È possibile aggiungere o rimuovere sensori manipolando l'header file, a patto di mantenerne la struttura. Questo codice fornisce una base modulare e scalabile per la gestione dei sensori in un sistema embedded.

Struttura della libreria

La libreria è strutturata attorno a una gerarchia di classi che implementano un'astrazione per i sensori:

- **SensorBase** (classe base astratta)
- **SensorStandard** (sensori collegati direttamente alla scheda)
- **SensorSpecialized** (sensori che richiedono una gestione speciale tramite I2C)
- **SensorStandardExternal** (sensori gestiti da un ADC esterno)

SensorBase

La classe SensorBase rappresenta l'astrazione comune per tutti i sensori. Ogni sensore è configurato tramite la struttura SensorConfig, che include:

- **Struttura Sensor**
- **Stato di inizializzazione**
- **Stato** (abilitato/disabilitato)
- **Latenza minima tra le letture** (se richiesto dall'implementazione)

Metodi principali (elenco non esaustivo):

- getConfiguration(): restituisce la configurazione del sensore.
- getId(), getPin(), getType(), getName(), getDescription(): restituiscono informazioni sul sensore.
- debug(Stream& stream): abilita il debug su una porta seriale.

Ogni sensore derivato deve implementare:

- begin(rate, latency): metodo di inizializzazione del sensore.
- configure(rate, latency): metodo di modifica delle impostazioni del sensore.
- toString(): rappresentazione testuale dello stato del sensore.
- getValue(): metodo per ottenere il valore del sensore.

SensorStandard

La classe SensorStandard eredita da SensorBase e gestisce i sensori connessi direttamente ai pin della scheda.

Metodi aggiuntivi:

- getDigital(): legge il valore digitale dal pin del sensore.

- getVoltage(): converte il valore analogico letto in un valore di tensione (usando **VREF = 3.3V** e un ADC a 12 bit).

Il metodo begin() imposta il pin in modalità di lettura e verifica che il valore letto sia valido.

SensorSpecialized

SensorSpecialized è una classe astratta che estende SensorBase e rappresenta i sensori che richiedono una gestione più complessa, tipicamente tramite **I2C**.

Le implementazioni concrete devono definire:

- begin()
- configure()
- getValue()
- toString()

Questa classe funge da base per l'integrazione di sensori avanzati come IMU, sensori ambientali o di gas.

SensorStandardExternal

SensorStandardExternal è simile a SensorStandard, ma utilizza un **ADC esterno** (**Adafruit ADS1115**) per la lettura dei valori.

Differenze rispetto a SensorStandard:

- Utilizza il bus **I2C** per comunicare con l'ADC.
- Supporta una risoluzione a **16 bit** (*EXT_ADC_MAX_VALUE = 32767*).
- Se l'ADC non è inizializzato, begin() tenta di configurarlo sugli indirizzi **0x48** o **0x49**.
- I metodi getDigital() e getVoltage() leggono i dati dall'ADC esterno invece che dai pin della scheda.

Questa libreria è pensata per essere estesa facilmente aggiungendo nuove classi di sensori specializzati.

9.5. Sezione Time

Gestione RTC

Questa libreria permette di gestire un modulo RTC (Real-Time Clock) basato su DS1307 per mantenere l'ora e la data anche in caso di perdita di alimentazione. È progettata per funzionare con il microcontrollore FireBeetle 2 e integra funzionalità per aggiornare l'orologio tramite una connessione di rete.

Funzionalità Principali

Lettura della data e dell'ora dal modulo RTC.

Impostazione della data e dell'ora utilizzando una connessione di rete.

Conversione della data in formato *timestamp* leggibile.

Debug via porta seriale.

1. Inizializzazione del modulo RTC

Prima di usare l'RTC, è necessario inizializzarlo invocando il metodo begin().

2. Ottenere data e ora dal modulo RTC

È possibile ottenere la data e l'ora attuale dal modulo RTC:

```
uint16_t dateTime[TIME_LEN];

char timestamp[TIMESTAMP_LEN];

bool success = rtcController.getDateAndTime(dateTime, TIME_LEN);

if(success)

   rtcController.getTimestamp(timestamp, TIMESTAMP_LEN, dateTime, TIME_LEN);
```

Nota -> Il metodo getDateAndTime() verifica se il valore memorizzato nell'RTC è valido. Se non lo è, tenta di recuperare un valore salvato in EEPROM.

3. Impostare la data e l'ora tramite rete

L'RTC può essere sincronizzato con un server NTP utilizzando un modulo Wi-Fi o LTE. Esempio di utilizzo con la classe WiFiManager:

```
struct tm timeinfo;

if (wifiManager.getDateAndTime(&timeinfo))

    if (rtc.setDateAndTime(timeinfo))

        Serial.println("RTC aggiornato con successo dalla rete.");
```

Nota -> Il metodo setDateAndTime() verifica se il valore ricevuto è valido. Se lo è, salva il valore corrente in EEPROM.

4. Debug

La libreria supporta il debug tramite seriale.

Il modulo RTC deve essere alimentato costantemente con una batteria tampone per mantenere l'ora anche in assenza di alimentazione principale.

9.6. Sezione Rom

Controller di Memoria

Questa documento descrive la classe MemoryController, responsabile della gestione della memoria interna (SPIFFS) all'interno della libreria. La classe si occupa di:

- Inizializzare il filesystem SPIFFS, con la possibilità di cancellare l'intero contenuto della memoria all’avvio.
- Fornire metodi per elencare, leggere, scrivere ed eliminare file (inclusi quelli nascosti).
- Gestire il caricamento, il salvataggio e la verifica di certificati SSL.

- Gestire il caricamento e il salvataggio della configurazione (formato CSV) del dispositivo, che può contenere informazioni relative ai protocolli di comunicazione (MQTT, socket) e alle configurazioni di rete (WiFi, LTE).
- Gestire i sensori disabilitati all'avvio, con metodi per aggiungere o rimuovere sensori dalla lista di avvio.

Funzionalità Principali

Inizializzazione e Gestione del Filesystem

- **begin()**
  Inizializza SPIFFS. Se richiesto (in base alla configurazione interna), cancella l'intero contenuto della memoria all'avvio. Durante l'inizializzazione, il metodo restituisce il contenuto del filesystem, escludendo i file nascosti.
- **listDir()**
  Restituisce il contenuto del filesystem, elencando i file e le directory presenti.

Gestione dei Certificati

- **loadCertificates()**
  Carica tutti i certificati dalla memoria.
- **loadCertificate(CType type)**
  Carica un singolo certificato specificato dal parametro type (ad es. certificato dell'autorità, certificato del client o chiave del client).
- **storeCertificate(CType type, const char* content)**
  Salva il certificato specificato in memoria.
- **certStatus(CType type = CType::ALL)**
  Verifica che il certificato di interesse sia valido.

Gestione della Configurazione

- **loadConfiguration()**
  Carica la configurazione dalla memoria. La configurazione è memorizzata in formato CSV e può includere informazioni relative al protocollo di comunicazione (es. MQTT) e alle configurazioni di

rete (WiFi, LTE).
Il parsing della configurazione avviene tramite metodi specifici:
    - **parseConnectionConfig()**: elabora le impostazioni di connessione.
    - **parseWiFiConfig()**: elabora la configurazione della rete WiFi.
    - **parseLTEConfig()**: elabora la configurazione della rete LTE.
- **storeConfiguration(Config config)**
  Salva la configurazione corrente in memoria.
- **configStatus()**
  Verifica che la configurazione caricata sia valida.
- **getConfiguration()**
  Restituisce la configurazione corrente.

Gestione dei Sensori

- **loadSensors(size_t& count, const int SENSOR_ID_NONE)**
  Recupera l'elenco dei sensori che sono disabilitati all'avvio, restituendo un array di ID.
- **addSensor(uint8_t sensorID, int* sensors, size_t sensorCount)**
  Aggiunge un sensore all'elenco dei sensori da disabilitare all'avvio.
- **removeSensor(uint8_t sensorID, int* sensors, size_t sensorCount)**
  Rimuove un sensore dall'elenco dei sensori da disabilitare all'avvio.

Operazioni di Base sul Filesystem (Utility)

- **readFile(const char* path)**
  Legge il contenuto di un file e lo restituisce.
- **writeFile(const char* path, const char* content)**
  Scrive il contenuto specificato in un file.
- **deleteFile(const char* path)**
  Elimina un singolo file dal filesystem.
- **deleteAll(fs::FS &fs, const char *dirname, uint8_t levels)**
  Elimina tutti i file presenti in una directory, inclusi quelli nascosti.
- **restoreBackupFile(const char* path)**
  Ripristina un file al suo stato precedente. Ogni volta che un file viene aggiornato, viene creata una copia di backup della configurazione precedente, che può essere ripristinata se necessario.

- **Flush e Hardflush**
  I metodi flush() e hardFlush() vengono invocati in combinazione con deleteFile() e deleteAll(), analogamente alle operazioni di base di lettura e scrittura, per assicurare la completa sincronizzazione e cancellazione dei file.

Struttura dei File e Percorsi Utilizzati

I percorsi principali utilizzati dalla classe sono:

- **Certificati**:
  - AUTH_CERT: /certs/ca.crt
  - CLIENT_CERT: /certs/client.crt
  - CLIENT_KEY: /certs/client.key
- **Configurazioni**:
  - CONFIG: /config/device.cfg
  - MQTT: /config/mqtt.cfg
  - SOCK: /config/socket.cfg
  - WIFI_CONFIG: /config/wifi.cfg
  - LTE_CONFIG: /config/lte.cfg
- **Sensori**:
  - SENSORS: /sensors/disabled

Strutture Dati Utilizzate

```
struct Certs {
   char* authorityCert;
   char* clientCert;
   char* clientKey;
};
struct Connection {
   ConnType type;
   char* address;
   uint16_t port;
   char* username;
   char* psw;
```

```
};
struct WIFI {
   char* ssid;
   char* psw;
};
struct LTE {
   char* apn;
   char* username;
   char* psw;
};
struct Config {
   char* dname;
   float latitude;
   float longitude;
   Connection connection;
   WIFI wifi;
   LTE lte;
};
```

Configurazioni ammissibili

```
enum class ConnType {
   NONE          = 0,
   MQTT_OVER_WIFI  = 10,
   MQTT_OVER_LTE   = 20,
   SOCK_OVER_WIFI  = 30,
   SOCK_OVER_LTE   = 40
};
```

9.7. Controller WiFi e MQTT

Questa documento descrive le classi che implementano il controller WiFi ed il controller del protocollo MQTT all'interno della libreria.

- **MQTTClientManager**: Gestisce la connessione a un broker MQTT, incluse le operazioni di connessione, pubblicazione, sottoscrizione e disconnessione. Supporta sia connessioni non criptate che criptate (TLS) tramite l'utilizzo di certificati e chiavi.
- **WiFiManager**: Gestisce la connessione WiFi del dispositivo, la configurazione della rete e la sincronizzazione dell'orario tramite un server NTP.

MQTTClientManager

Panoramica

La classe MQTTClientManager utilizza la libreria **PubSubClient** per la comunicazione MQTT; a seconda della disponibilità dei certificati SSL, può utilizzare:

- WiFiClient per connessioni non criptate.
- NetworkClientSecure (o WiFiClientSecure su ESP32) per connessioni criptate.

Funzionalità Principali

- **Inizializzazione e configurazione**
  - begin(const char* server, uint16_t port, const char* user, const char* psw): Inizializza il client MQTT con il server, la porta e le credenziali fornite. Imposta il tipo di connessione (criptata o meno) in base alla presenza di certificati.
- **Gestione dei certificati**
  - setCACert(const char* rootCA): Imposta il certificato dell'autorità (Root CA) per la connessione TLS.
  - setCertificate(const char* clientCert): Imposta il certificato del client.
  - setPrivateKey(const char* clientKey): Imposta la chiave privata del client.
- **Gestione della connessione**
  - connect(const char* ID): Tenta di connettersi al broker MQTT utilizzando l'ID fornito e le credenziali. In caso di fallimento, effettua un nuovo tentativo senza credenziali.
  - disconnect(): Disconnette il client dal broker.
  - isConnected(): Restituisce lo stato della connessione.
- **Operazioni MQTT**
  - publish(const char* topic, const char* payload): Pubblica un messaggio su un topic specifico.

    - subscribe(const char* topic): Sottoscrive un topic per ricevere messaggi.
    - unsubscribe(const char* topic): Cancella la sottoscrizione ad un topic.
- **Altri metodi**
    - update(): Esegue il loop del client MQTT per la gestione dei messaggi in arrivo.
    - setCallback(MQTT_CALLBACK_SIGNATURE): Imposta una funzione di callback per gestire i messaggi ricevuti.
    - getState(): Restituisce lo stato corrente del client MQTT, fornendo anche una descrizione testuale.
    - getStatus(): Restituisce lo stato interno dell'oggetto.
    - flush(): Resetta il client MQTT, liberando le risorse e le credenziali in uso.

WiFiManager

Panoramica

La classe WiFiManager è implementata si occupa della gestione della connessione WiFi del dispositivo. Essa permette di configurare le credenziali della rete, connettersi, riconnettersi, disconnettersi e sincronizzare la data e l'ora tramite un server NTP.

Funzionalità Principali

- **Configurazione della rete WiFi**
    - configure(const char* ssid, const char* psw): Configura le credenziali della rete WiFi. Se il dispositivo è già connesso, chiude la connessione attuale prima di configurare la nuova rete.
- **Gestione della connessione WiFi**
    - connect(): Avvia la connessione alla rete WiFi configurata, gestendo il timeout e fornendo feedback sullo stato della connessione.
    - disconnect(): Disconnette il dispositivo dalla rete WiFi.
    - reconnect(): Tenta di riconnettersi alla rete WiFi se la connessione è persa.
    - isConnected(): Verifica se il dispositivo è attualmente connesso alla rete WiFi.
- **Sincronizzazione della data e dell'ora tramite NTP**
  La classe fornisce tre metodi per ottenere la data e l'ora:
    - getDateAndTime(): Recupera e stampa in debug la data e l'ora ottenute da un server NTP.
    - getDateAndTime(char* date, size_t dateSize, char* time, size_t timeSize): Recupera la data e l'ora, formattandole in due stringhe separate (per data e ora).

- getDateAndTime(struct tm *timeinfo): Popola una struttura tm con la data e l'ora correnti; restituisce true in caso di successo.

Debug

Entrambe le classi supportano il debug tramite una porta seriale. È possibile abilitare il debug invocando il metodo di debug() delle rispettive classi. Il debug consente di monitorare le operazioni e rilevare eventuali errori durante l'esecuzione.

9.8. Gestione Storico Dati Sensori e Database Simulato su SPIFFS

Questo documento descrive le classi **RingBuffer** e **DataHandler**, che insieme implementano la gestione dello storico dei dati dei sensori ed un database simulato su SPIFFS.

1. RingBuffer

La classe RingBuffer gestisce un buffer circolare per la memorizzazione dei record dei dati dei sensori su SPIFFS. Utilizza un file per conservare i dati e mantiene un header contenente le informazioni sul buffer (indice di lettura, scrittura, numero di record e capacità).

Strutture Dati

- **DataHeader**
  Una struttura di 16 byte composta da 4 campi uint32_t:
  - head: indice del primo record valido.
  - tail: indice del prossimo record da scrivere.
  - count: numero di record attualmente presenti.
  - capacity: capacità massima del buffer (definita da RING_BUFFER_CAPACITY).
- **Record**
  Rappresenta un singolo campione dei dati:
  - ID: identificatore del sensore.
  - timestamp: stringa contenente il timestamp (formato definito da TIMESTAMP_LEN).
  - sample: valore campionato (ad es. lettura analogica).

Metodi Principali

- **begin()**
  Inizializza il buffer verificando che il controller di memoria (MemoryController) sia attivo.
- **appendRecord(const Record &record, char* fname)**
  Aggiunge un record al buffer. Se il buffer è pieno, sovrascrive il record più vecchio.
  - Se il file specificato da fname non esiste, viene creato insieme a un header iniziale.
  - Aggiorna il campo tail e il conteggio dei record.
- **readRecords(uint32_t n, std::array<Record, RING_BUFFER_CAPACITY> &records, uint32_t &numRead, char* fname)**
  Legge fino a n record, partendo dal record più vecchio, e li inserisce in un array fisso.
  - numRead indica il numero effettivo di record letti.
- **readRecords(uint32_t n, std::vector<Record> &records, char* fname)**
  Variante dinamica che legge fino a n record e li inserisce in un vettore.
- **removeRecords(uint32_t n, char* fname)**
  Consuma (rimuove) n record dal buffer, aggiornando l'header (spostando il puntatore head e decrementando il conteggio).

Metodi Privati

- **readHeader(DataHeader &header, File &file)**
  Legge l'header dal file, posizionato all'inizio.
- **writeHeader(const DataHeader &header, File &file)**
  Scrive l'header aggiornato nel file.

2. DataHandler

La classe DataHandler gestisce le configurazioni dei sensori e l'archiviazione dei campioni (record) utilizzando il RingBuffer. Essa svolge il ruolo di "database simulato" su SPIFFS e si occupa di:

- Effettuare il parsing della configurazione (in formato JSON) per ogni sensore.
- Memorizzare e aggiornare le configurazioni dei sensori.
- Aggiungere campioni nel ring buffer associato a ciascun sensore.
- Recuperare i campioni registrati (tramite versioni "statiche" e "dinamiche").
- Eseguire operazioni di "flush" per cancellare i dati salvati.
- Gestire il reset delle configurazioni e dei dati.

Strutture Dati e Definizioni

- **SConfig**
  Struttura che definisce la configurazione di un sensore:
  - ID: identificatore del sensore.
  - topic: topic MQTT associato al sensore.
  - txTime, pollingTime: tempi di trasmissione e polling.
  - maxSamples: numero massimo di campioni da memorizzare.
  - outOfRange: flag per segnalare campioni fuori range.
  - thresholdMin e thresholdMax: soglie minime e massime.
  - status: stato della configurazione (ad es. reset o aggiornamento).

Metodi Principali

- **parseConfiguration(uint8_t sID, char* topic, uint8_t* configuration, unsigned int length)**
  Esegue il parsing di una configurazione JSON ricevuta in input, verificandone la validità e memorizzando i parametri nel mapping sensorConfigs per uno specifico sensore.
- **getConfiguration(uint8_t sID, SConfig& config)**
  Recupera la configurazione associata al sensore identificato da sID.
- **resetConfigurationStatus(uint8_t sID)**
  Resetta lo stato della configurazione di un sensore. Lo stato aiuta a tenere traccia degli aggiornamenti nella configurazione di un sensore.
- **addSample(uint8_t sID, float sample, const char* timestamp)**
  Aggiunge un campione (con il relativo timestamp) al ring buffer associato ad uno specifico sensore.
  - Utilizza un filename generato dinamicamente (/.sensor_<ID>_data.bin) per archiviare i dati.
- **Metodi di Lettura dei Campioni**
  - **Statici**:
    - getSamplesStatic(uint8_t sID, uint8_t& samples): legge i campioni in un array statico.
    - getSamplesStatic(uint8_t sID, uint32_t n, uint32_t &numRead): legge fino a n campioni, aggiornando numRead.
  - **Dinamic**:
    - getSamples(uint8_t sID, uint8_t& samples): legge i campioni e restituisce il numero di campioni letti.
    - getSamples(uint8_t sID, uint32_t n, std::vector<Record>& records): legge fino a n campioni e li inserisce in un vettore.

- **flush(uint8_t sID)**
  Cancella i campioni salvati per uno specifico sensore (viene rimosso il contenuto del ring buffer associato).
- **reset()**
  Esegue il reset globale:
  - Cancella i dati per ciascun sensore.
  - Libera la memoria allocata per i topic e le configurazioni.
  - Resetta il buffer di lettura.
- **getTopic(uint8_t sID)**
  Restituisce il topic MQTT associato ad uno specifico sensore, se presente.

Supporto Statico e Dinamico

- La classe utilizza un array statico _readBuffer per le operazioni di lettura "statiche", mentre per quelle "dinamiche" viene usato un std::vector<Record>.

Debug

Entrambe le classi supportano la modalità debug. È possibile abilitare il debug invocando il metodo debug() e passando un oggetto Stream (tipicamente Serial):

9.9. BLE Handler

Questo documento descrive la classe BLEHandler, responsabile della gestione della comunicazione BLE all'interno della libreria. Il controller BLE viene utilizzato per abilitare l'accesso remoto al dispositivo tramite servizi e caratteristiche BLE, che includono:

- Servizio di Controllo delle Autorizzazioni
- Servizio di Informazioni sul Dispositivo
- Servizio di Supporto ai Protocolli Internet
- Servizio di Gestione dei Sensori
- Servizio di Gestione dello Storage
- Servizio di Gestione della Batteria (non implementato)

Panoramica

Il metodo begin() inizializza il modulo BLE settando i servizi e le caratteristiche associate. Queste caratteristiche consentono di:

- **Controllare l'accesso autorizzato al dispositivo**
  Il servizio di autorizzazione utilizza UUID personalizzati (AUTH_SERVICE_UUID e AUTH_CHARACTERISTIC_UUID) e un flusso di autenticazione basato su chiave segreta (SECRET_KEY).
- **Configurare il dispositivo**
  Il servizio di configurazione (CONFIG_SERVICE_UUID, CONFIG_CHARACTERISTIC_UUID) permette di leggere e scrivere la configurazione del dispositivo. Vengono gestiti callback specifici per la lettura (configReadCallback) e la scrittura (configWriteCallback) della configurazione.
- **Gestire i certificati SSL**
  Tramite il servizio dedicato (CERT_SERVICE_UUID e relative caratteristiche per i certificati del client, dell'autorità e la chiave privata), è possibile inviare e ricevere i certificati necessari per la connessione sicura. Viene offerto un callback per la scrittura dei certificati tramite certWriteCallback.
- **Gestire i sensori**
  Il servizio per i sensori (SENSOR_SERVICE_UUID) permette di leggere e scrivere le configurazioni dei sensori. Vengono gestiti due callback per il sensore: uno per la lettura (sensorConfigReadCallback) e uno per la scrittura (sensorConfigWriteCallback).
- **Gestire lo storage**
  Il servizio di gestione dello storage (FLASH_SERVICE_UUID) consente di eseguire operazioni sullo storage, come la cancellazione della memoria flash. Viene impostato un callback tramite setStorageCallback.

Funzionalità e Metodi Principali

- **Inizializzazione BLE**
  - Il metodo begin() (privato) inizializza il BLE, configurando tutti i servizi e le relative caratteristiche, nonché i relativi descriptor (ad es. per autenticazione, configurazione, certificati, sensori e storage).
  - Le caratteristiche sono associate a callback statici (es. onAuthCharacteristicWriteStatic, onConfigCharacteristicWriteStatic, etc.) che, tramite il singleton, rimandano alle funzioni non statiche per la gestione degli eventi.

- **Gestione dello Stato**
  - enableBLE() e disableBLE() permettono di abilitare o disabilitare il modulo BLE.
  - poll() viene utilizzato per eseguire il loop di gestione degli eventi BLE (ad esempio, per ricevere dati dai dispositivi centrali).
- **Callback Personalizzabili**
  - **Configurazione**
    - setConfigReadCallback(ConfigReadCallback callback): imposta la callback per leggere la configurazione.
    - setConfigWriteCallback(ConfigWriteCallback callback): imposta la callback per scrivere la configurazione.
  - **Certificati**
    - setCertWriteCallback(CertWriteCallback callback): imposta la callback per la scrittura dei certificati.
  - **Sensori**
    - setSensorConfigReadCallback(SensorConfigReadCallback callback): imposta la callback per leggere la configurazione di un sensore.
    - setSensorConfigWriteCallback(SensorConfigWriteCallback callback): imposta la callback per scrivere la configurazione di un sensore.
  - **Storage**
    - setStorageCallback(StorageCallback callback): imposta la callback per gestire operazioni di storage (ad es. cancellazione flash).
- **Gestione degli Eventi di Connessione**
  - onConnect(BLEDevice central) e onDisconnect(BLEDevice central) gestiscono gli eventi di connessione e disconnessione dei dispositivi centrali.
- **Gestione delle Scritture sulle Caratteristiche**
  - onAuthCharacteristicWrite(): gestisce le scritture per il flusso di autenticazione.
  - onConfigCharacteristicWrite(): gestisce le scritture per la configurazione del dispositivo.
  - onCertCharacteristicWrite(): gestisce le scritture per la gestione dei certificati SSL.
  - onSensorConfigCharacteristicRead() e onSensorConfigCharacteristicWrite(): gestiscono rispettivamente la lettura e la scrittura delle configurazioni dei sensori.
  - onStorageCharacteristicWrite(): gestisce le scritture per le operazioni sullo storage.
- **Utilità e Dettagli**

- La classe definisce costanti, UUID e descrizioni per ogni servizio e caratteristica (ad es. CONFIG_DESC, CERT_CLIENT_DESC, etc.), e supporta il debug tramite un oggetto Stream.

1. Flusso di Autenticazione

- Il flusso di autenticazione utilizza una chiave segreta definita (SECRET_KEY).
- Viene gestito un tempo massimo per l'autenticazione (TTA), dopo il quale il dispositivo non sarà considerato autenticato se non riceve il flusso corretto.
- La caratteristica di autorizzazione invia/controlla pacchetti di autenticazione (strutturati tramite AuthPacket definito in Packet.h).
- Se la chiave ricevuta corrisponde a quella attesa, il dispositivo viene considerato autenticato e vengono abilitati gli altri flussi (configurazione, certificati, sensori e storage).

2. Flusso di Configurazione

- **Metodi Coinvolti:**
  - **setConfigCharacteristic()**
    - Ottiene la configurazione attuale tramite la callback impostata (configReadCallback).
    - Se non è presente una configurazione, viene impostato un valore di default (definito in EMPTY_CONFIG).
    - Il valore viene inserito in un ConfigPacket e scritto nella caratteristica di configurazione.
  - **onConfigCharacteristicWrite(BLEDevice central, BLECharacteristic characteristic)**
    - Questo metodo legge il pacchetto di configurazione (ConfigPacket) inviato e lo elabora.
    - I vari campi ottenuti rappresentano:
      - La configurazione del device
      - Le impostazioni di connessione
      - La configurazione WiFi
      - La configurazione LTE
    - Viene quindi invocato il callback configWriteCallback per aggiornare la configurazione. Se la scrittura ha successo, la caratteristica viene aggiornata con la nuova configurazione.

3. Flusso di Gestione dei Certificati

- **Servizio e Caratteristica:**
  Il servizio dedicato ai certificati è definito da CERT_SERVICE_UUID e le caratteristiche per:
  - Il certificato del client (CERT_CLIENT_CHARACTERISTIC_UUID)
  - Il certificato dell'autorità (CERT_AUTHORITY_CHARACTERISTIC_UUID)
  - La chiave privata del client (CERT_CLIENT_KEY_CHARACTERISTIC_UUID)
- **Metodi Coinvolti:**
  - **onCertCharacteristicWrite(BLEDevice central, BLECharacteristic characteristic)**
    - Identifica quale caratteristica sta ricevendo i dati in base al suo UUID.
    - Per ciascun tipo di certificato, controlla il tempo trascorso (usando CERT_TIMEOUT) per garantire che l'intero pacchetto venga ricevuto in tempo.
    - Chiama le funzioni handleClientCert(), handleAuthorityCert() o handleClientKey() per gestire il salvataggio dei dati ricevuti.
    - Se il tempo di aggiornamento supera il timeout, il pacchetto viene scartato e il processo viene resettato.

4. Flusso di Configurazione dei Sensori

- **Servizio e Caratteristica:**
  Il servizio per i sensori è definito da SENSOR_SERVICE_UUID. Le caratteristiche coinvolte sono:
  - Lettura della configurazione dei sensori (SENSOR_CONFIG_CHARACTERISTIC_READ_UUID)
  - Scrittura della configurazione dei sensori (SENSOR_CONFIG_CHARACTERISTIC_WRITE_UUID)
- **Metodi Coinvolti:**
  - **onSensorConfigCharacteristicRead(BLEDevice central, BLECharacteristic characteristic)**
    - Quando il client richiede la configurazione di un sensore, il metodo legge un pacchetto (SensorConfigReadPacket) dalla caratteristica.
    - Utilizza il callback sensorConfigReadCallback per ottenere la configurazione attuale per il sensore richiesto.
    - Se la configurazione non è disponibile o è malformata, viene impostato un valore di errore (ad es. "-1" o "0" come stringa).

        - Se la configurazione è corretta, la stringa viene analizzata e viene creato un pacchetto di risposta che viene scritto nella caratteristica.
    - **onSensorConfigCharacteristicWrite(BLEDevice central, BLECharacteristic characteristic)**
        - Questo metodo gestisce l'aggiornamento della configurazione dei sensori inviato dal client.
        - Legge il pacchetto (SensorConfigWritePacket) che contiene:
            - Il comando (abilitazione/disabilitazione temporanea o permanente)
            - L'ID del sensore
            - Il valore della latenza
        - In base al comando ricevuto, viene invocato il callback sensorConfigWriteCallback per aggiornare la configurazione.
        - Viene poi restituito un feedback mediante debug, per indicare se l'aggiornamento è avvenuto con successo o meno.

5. Flusso di Gestione dello Storage

- **Servizio e Caratteristica:**
  Il servizio di gestione dello storage è definito da FLASH_SERVICE_UUID e la caratteristica associata è ERASE_FLASH_CHARACTERISTIC_UUID.
- **Metodi Coinvolti:**
    - **onStorageCharacteristicWrite(BLEDevice central, BLECharacteristic characteristic)**
        - Legge un pacchetto di configurazione flash (FlashConfigPacket) che contiene una chiave e, opzionalmente, un percorso.
        - Viene estratta la chiave e verificato che corrisponda a quella attesa (definita da SECURE_KEY).
        - Se la chiave non corrisponde, viene restituito un messaggio di errore (il valore della chiave può essere nascosto in debug).
        - Se la chiave è valida, viene chiamato il callback storageCallback passando il percorso. Se il percorso è vuoto, viene indicato di eseguire il flush completo della memoria.

Debug e Callback

La classe BLEHandler supporta il debug tramite un oggetto Stream (tipicamente Serial) che permette di monitorare l’attività e diagnosticare eventuali errori.

10. Smart Work Platform - Client di Configurazione

Questa sezione descrive in dettaglio il funzionamento del client sviluppato in **React Native** utilizzando il framework **Expo**. Il client è stato realizzato per configurare una board **ESP32**, guidando l'utente attraverso una serie di passaggi che gli consentono di connettersi ed impostare il dispositivo in maniera semplice e sicura. La documentazione fornisce una panoramica delle tecnologie utilizzate, delle funzionalità implementate e delle specifiche tecniche per ogni fase della configurazione.

Il client SWP è stato progettato per semplificare e standardizzare la configurazione delle board **ESP32** all'interno dell'ecosistema IoT. L'applicazione offre una guida passo-passo (wizard) che consente all'utente di:

- Rilevare e connettersi al dispositivo ESP32 tramite **BLE**.
- Configurare parametri essenziali quali nome, posizione e tipo di configurazione.
- Personalizzare le impostazioni di rete scegliendo tra **WiFi** e **LTE**.
- Configurare il protocollo di comunicazione con il backend, con supporto per **MQTT** (v3.1.1) o **Socket** (TCP/HTTP/HTTPS).
- Gestire i certificati per la sicurezza delle comunicazioni TLS.

La struttura modulare e dinamica del client permette di adattarsi a differenti ambienti di rete e requisiti di comunicazione, garantendo una configurazione robusta e sicura del dispositivo.

In altri termini, il client di configurazione della **Smart Work Platform (SWP)** è un'applicazione mobile sviluppata in **React Native**, utilizzando il **framework Expo**, che permette una gestione cross-platform (Android e iOS) efficiente e performante. Il suo scopo è guidare in modo semplice, sicuro e standardizzato la configurazione delle board **ESP32**, particolarmente diffuse in ambito IoT per la loro versatilità, basso consumo energetico e capacità di connessione wireless. L'architettura e tecnologie utilizzate si riassumono in:

- **React Native + Expo**: Permette uno sviluppo rapido, supporto integrato a Bluetooth Low Energy (BLE), semplicità di deployment e debugging tramite Expo Go.
- **Modularità del codice**: La struttura modulare consente la facile integrazione di nuove funzionalità o il supporto a diversi modelli di ESP32.
- **Componenti dinamici**: Ogni fase della configurazione è rappresentata da un modulo autonomo, facilitando il debugging e l'eventuale aggiornamento del singolo step.

Le funzioni implementate sono:

1. **Rilevamento del dispositivo tramite BLE**:
    - Il client scansiona i dispositivi nelle vicinanze e rileva automaticamente la board ESP32 attiva tramite **Bluetooth Low Energy**.
    - Il pairing avviene con verifica dell'identità per evitare interferenze o accessi non autorizzati.
2. **Configurazione dei parametri iniziali**:
    - Una volta connesso, l'utente può assegnare un **nome al dispositivo**, specificare la **posizione geografica** (tramite inserimento manuale o geolocalizzazione automatica) e indicare il **tipo di configurazione** desiderata (es. monitoraggio, automazione, ecc.).
3. **Impostazioni di rete**:
    - Il sistema permette di configurare la connessione a una rete WiFi oppure a una rete LTE.
    - Il wizard guida l'utente nell'inserimento delle credenziali (SSID e password) o nell'associazione del dispositivo a un modulo LTE esterno.
4. **Configurazione del protocollo di comunicazione**:
    - L'utente può scegliere tra i seguenti protocolli:
        - **MQTT (v3.1.1)**, noto per la leggerezza e l'efficienza nei contesti a bassa larghezza di banda.
        - **Socket TCP/HTTP/HTTPS**, per applicazioni che richiedono comunicazioni più complesse o compatibilità con server esistenti.
    - È possibile definire l'endpoint del backend, le porte di comunicazione, i topic MQTT o i percorsi REST.
5. **Gestione della sicurezza e dei certificati TLS**:
    - Il client consente l'upload di certificati digitali (root, client, e CA) per abilitare la **comunicazione criptata TLS** tra il dispositivo e il backend.
    - È supportata anche la validazione della firma dei certificati e la gestione dei certificati scaduti o non validi.

I vantaggi di questa configurazione sono:

- **Adattabilità**: L'architettura dinamica consente al sistema di funzionare in ambienti di rete eterogenei, sia pubblici che privati.
- **Robustezza**: La separazione logica delle fasi evita blocchi totali in caso di errore in uno specifico step.
- **Sicurezza**: Oltre alla gestione TLS, l'interfaccia prevede anche autenticazione a due fattori per l'accesso alla configurazione avanzata.

- **User Experience (UX)**: Il wizard è progettato con una UI semplice, interattiva e intuitiva, anche per operatori non tecnici.

10.1. Architettura e Tecnologie Utilizzate

L'architettura del client Smart Work Platform è costruita su una solida integrazione di tecnologie moderne, progettate per garantire efficienza, portabilità e sicurezza nelle operazioni di configurazione dei dispositivi ESP32. Al cuore del sistema troviamo **React Native**, una delle soluzioni più avanzate per lo sviluppo di applicazioni mobile cross-platform. React Native permette di scrivere codice in JavaScript, traducendolo in componenti nativi per dispositivi Android e iOS, ottenendo prestazioni elevate e una user experience fluida. La gestione dello stato dell'applicazione è affidata a strumenti come **Redux** o la **Context API**, che garantiscono un controllo centralizzato delle informazioni condivise tra i vari componenti dell'interfaccia, migliorando al contempo la reattività dell'app.
L'ambiente di sviluppo è ulteriormente potenziato dall'uso di **Expo**, un framework e piattaforma open source che semplifica tutte le fasi di sviluppo, dal debug al deployment. Grazie a Expo è possibile testare in tempo reale il comportamento dell'app su dispositivi fisici o emulatori, senza la necessità di configurazioni complesse o compilazioni intermedie, riducendo significativamente i tempi di sviluppo.
La componente hardware principale coinvolta nel sistema è la **board ESP32**, un microcontrollore estremamente versatile e diffuso nell'ambito IoT. Il dispositivo viene inizialmente rilevato e configurato tramite connessione **BLE (Bluetooth Low Energy)**, una tecnologia ideale per comunicazioni a corto raggio e a basso consumo energetico. Questa modalità di connessione consente all'applicazione mobile di stabilire un canale sicuro con il dispositivo ESP32, anche in assenza di una rete dati. Una volta completata la configurazione iniziale, la board può essere collegata alla rete tramite connessione **WiFi** oppure **LTE**, a seconda dell'ambiente operativo.
Dal punto di vista della comunicazione tra client e backend, il sistema supporta diversi protocolli per garantire flessibilità e scalabilità. In particolare, viene implementato **MQTT (versione 3.1.1)**, un protocollo di messaggistica leggero basato su pubblicazione/sottoscrizione, largamente utilizzato nei contesti IoT per la sua efficienza nella trasmissione di piccoli pacchetti di dati in tempo reale. Alternativamente, è possibile utilizzare protocolli basati su **socket TCP**, sia in chiaro (**HTTP**) che cifrati (**HTTPS**), per abilitare una comunicazione più diretta e, se necessario, crittografata.
L'applicazione integra inoltre funzionalità avanzate di **geolocalizzazione**, sfruttando le API native offerte dai sistemi operativi mobili per raccogliere dati relativi alla posizione (latitudine e longitudine) del dispositivo durante la fase di configurazione. Questi dati possono risultare fondamentali per il corretto tracciamento e posizionamento del dispositivo nell'ambiente industriale in cui sarà installato.

Infine, un aspetto centrale dell'architettura è la **gestione dei certificati TLS**, fondamentali per garantire la sicurezza nelle comunicazioni. Il client consente di caricare, aggiornare o rimuovere certificati digitali in modo semplice, assicurando così che tutti i dati trasmessi tra il dispositivo e il server siano protetti tramite crittografia end-to-end. Questa funzione è progettata per essere facilmente gestibile anche da utenti non esperti, contribuendo a mantenere elevati standard di sicurezza senza compromettere l'usabilità del sistema.

Di seguito si riepilogano gli aspetti principali:

- **React Native** Consente lo sviluppo cross-platform sfruttando JavaScript e componenti nativi. La gestione dello stato (ad es. tramite Redux o Context API) e le transizioni fluide sono implementate per migliorare l'esperienza utente.

- **Expo**
  Fornisce strumenti di sviluppo, debug e distribuzione semplificati per applicazioni React Native, consentendo testing su dispositivi reali in modo rapido ed efficace.

- **ESP32**
  Microcontrollore versatile per applicazioni IoT, la board ESP32 viene configurata per comunicare con il client via BLE e, successivamente, tramite rete WiFi o LTE.

- **BLE (Bluetooth Low Energy)**

Tecnologie di comunicazione a basso consumo energetico per la ricerca e connessione al dispositivo ESP32.

Protocolli di Comunicazione

- **MQTT (v3.1.1)** Protocollo di messaggistica leggero, ideale per comunicazioni in tempo reale nell'IoT, configurabile sia in ambienti WiFi che LTE.

- **Socket (TCP/HTTP/HTTPS)** Permettono una comunicazione diretta e sicura con il backend, con supporto per i protocolli HTTP e HTTPS per le trasmissioni cifrate.

Altre Tecnologie

- **Geolocalizzazione** Utilizzo delle API native di geolocalizzazione per ottenere latitudine e longitudine del client durante il processo di configurazione.

- **Gestione dei Certificati TLS** Il client supporta il caricamento, l'aggiornamento e la rimozione dei certificati necessari per garantire una comunicazione sicura e criptata tra dispositivo e backend.

10.2. Funzionalità Principali

La fase iniziale del processo di configurazione prevede la rilevazione e la connessione del dispositivo ESP32 tramite tecnologia BLE (Bluetooth Low Energy). Il client, sviluppato in React Native, esegue una scansione dei dispositivi BLE presenti nell'ambiente circostante utilizzando un modulo specifico che interagisce con le API native del sistema operativo. Questa fase è cruciale per identificare correttamente il dispositivo da configurare, e viene eseguita attraverso una procedura robusta e ottimizzata per ridurre il consumo energetico e massimizzare l'affidabilità della rilevazione.
Una volta individuato il dispositivo ESP32, viene avviata una procedura di autenticazione tramite "handshake", durante la quale il client e il dispositivo stabiliscono un canale sicuro per la comunicazione. Questo passaggio garantisce non solo la corretta associazione tra client e dispositivo, ma anche la sicurezza nella trasmissione dei dati durante l'intero processo di configurazione. A connessione stabilita, il dispositivo transita in uno stato attivo, pronto a ricevere e interpretare i comandi inviati dal client.
La configurazione vera e propria è realizzata attraverso un'interfaccia utente di tipo wizard, che guida l'utente in modo sequenziale attraverso i vari passaggi necessari all'impostazione del dispositivo. Ogni step del wizard è progettato per raccogliere, validare e, ove necessario, testare parametri specifici. L'utente viene innanzitutto guidato nell'inserimento delle informazioni di base, come il nome del dispositivo, la sua posizione fisica e la tipologia di configurazione desiderata.

In seguito, viene richiesto di selezionare la modalità di rete preferita, scegliendo tra connessione WiFi o LTE, e il tipo di protocollo di comunicazione con il backend, tra MQTT (versione 3.1.1) o Socket (TCP/HTTP/HTTPS). In questa fase, il sistema si adatta dinamicamente alle scelte dell'utente, mostrando solo le opzioni compatibili con la configurazione corrente. L'interfaccia consente inoltre di verificare in tempo reale le impostazioni inserite, ad esempio attraverso un test di connessione alla rete WiFi selezionata, fornendo così un riscontro immediato sull'esito della configurazione.
L'approccio modulare del wizard, unito alla possibilità di personalizzare ogni fase del processo, rende il sistema estremamente flessibile e adatto a una vasta gamma di scenari applicativi nel contesto dell'Internet of Things (IoT).
Di seguito si riepilogano gli aspetti principali:

- **Scansione BLE**:

Il client effettua una scansione dei dispositivi BLE nelle vicinanze per individuare la board ESP32.

  - *Approccio Tecnico*: Viene utilizzato il modulo BLE di React Native che si interfaccia con le API native per la gestione del Bluetooth.
  - *Autenticazione*: Una volta identificato il dispositivo, si procede con una fase di "handshake" per garantire la connessione sicura.

- **Collegamento**:
  Stabilita la connessione, il dispositivo passa allo stato attivo per ricevere i comandi di configurazione.

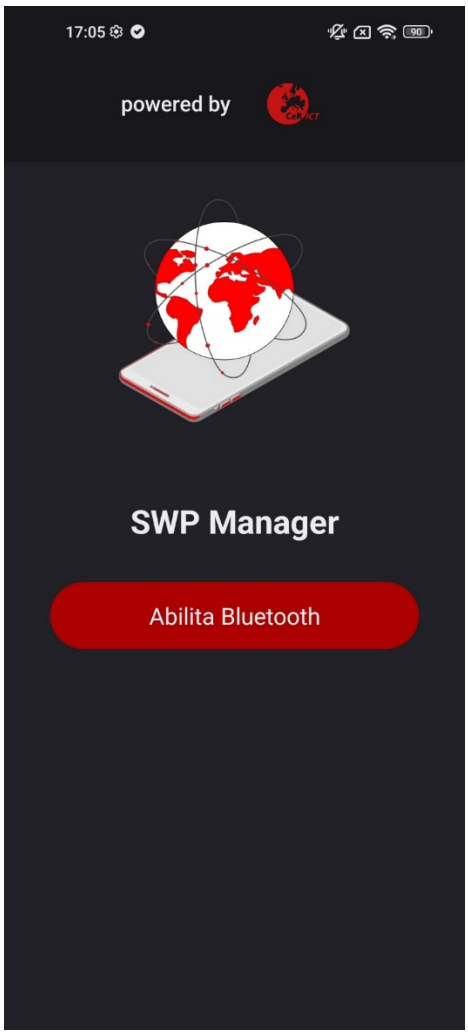


Figura 56: Abilitazione Bluetooth

Configurazione Guidata (Wizard)

La procedura di configurazione guidata è suddivisa in step, ciascuno progettato per acquisire e validare parametri specifici.

Il wizard:

- Guida l'utente nella configurazione di base (nome, posizione e tipologia).

- Definisce la modalità di rete e la tipologia di protocollo (MQTT o Socket) in modo dinamico.
- Permette la verifica interattiva delle impostazioni (ad esempio, test di connessione WiFi).

10.3. Dettagli della Configurazione Guidata

Il processo di configurazione guidata all'interno dell'applicazione Smart Work Platform è strutturato in una sequenza di fasi successive, ciascuna delle quali è pensata per rendere semplice, efficace e sicura la parametrizzazione del dispositivo ESP32. La logica adottata si basa su un'interfaccia interattiva, dinamica e reattiva, che si adatta in tempo reale alle scelte dell'utente, garantendo al contempo la coerenza e la validità delle informazioni inserite.


17:06
powered by
SWP Manager
Disconnetti
Multiparameter Station
Configurazione Guidata
Configurazione Manuale
(disabilitato)

Figura 57: disconnessione e configurazioni possibili

Nella **prima fase**, denominata Configurazione del Device, l'obiettivo principale è inizializzare il dispositivo assegnandogli parametri identificativi fondamentali. L'utente è chiamato a fornire un nome univoco al dispositivo, il quale viene sottoposto a un controllo automatico sulla validità sintattica e semantica — ad esempio, viene verificata la lunghezza del nome e l'uso esclusivo di caratteri ammessi, evitando simboli non standard. In parallelo, viene richiesto il consenso all'utilizzo della geolocalizzazione, grazie alla quale è possibile ottenere la posizione esatta (latitudine e longitudine) del dispositivo tramite API native. Questo passaggio avviene in modalità asincrona e prevede una gestione attenta dei permessi di sistema, per rispettare la privacy dell'utente. Successivamente, si procede alla selezione della modalità operativa: l'utente può scegliere tra diversi profili di configurazione, ognuno dei quali attiva automaticamente specifici rami del processo guidato. La flessibilità del client si manifesta proprio in questo punto: la procedura viene rimodulata dinamicamente in base alla tipologia selezionata, predisponendo schermate e opzioni dedicate alla configurazione WiFi o LTE, nonché alla scelta del protocollo di comunicazione da utilizzare.
La **seconda fase**, denominata Configurazione di Rete, ha come scopo l'impostazione della connessione che permetterà al dispositivo di comunicare con il backend. Per le configurazioni basate su WiFi, il sistema effettua una scansione delle reti wireless presenti nell'area, restituendo all'utente un elenco di SSID disponibili tra cui scegliere. In caso si opti per la modalità LTE — al momento ancora in fase di sviluppo — viene simulata una verifica di rete, predisponendo però l'infrastruttura per future integrazioni. Una volta selezionata la rete, è possibile inserire le credenziali di accesso (SSID e password), con relativa validazione del formato. Un test interattivo consente di verificare immediatamente la stabilità e l'efficienza della connessione prima di procedere agli step successivi.
La **terza fase**, denominata Configurazione del Protocollo di Comunicazione, è dedicata all'impostazione del canale attraverso cui il dispositivo scambierà dati con il backend. L'utente può configurare le credenziali e i parametri per la comunicazione, scegliendo tra i protocolli MQTT e Socket. In particolare, viene richiesto di indicare l'endpoint del server (indirizzo e porta), unitamente a username e password per autenticare il dispositivo. Il sistema integra una funzionalità di validazione interattiva che esegue controlli in tempo reale sui dati immessi: eventuali errori di formattazione, valori mancanti o incoerenze vengono immediatamente notificati tramite alert visivi e suggerimenti contestuali, permettendo così una correzione rapida e guidata.
Infine, nella **quarta fase**, denominata Configurazione dei Certificati di Rete per TLS, viene affrontato il tema della sicurezza della connessione. Il client fornisce un'interfaccia intuitiva per il caricamento dei certificati necessari alla cifratura delle comunicazioni. È possibile caricare il certificato dell'Autorità di Certificazione (CA), il certificato del client e la relativa chiave privata. L'utente ha inoltre la possibilità di aggiornare certificati esistenti o, se necessario, procedere alla loro rimozione completa. Ogni operazione viene monitorata tramite meccanismi di validazione dei formati (ad esempio controllo sull'estensione e sull'intestazione dei file)

e verifica dell'integrità degli stessi, al fine di evitare problemi nella fase operativa e garantire la sicurezza del dispositivo una volta attivo.
In sintesi, l'intero flusso di configurazione è pensato per fornire un'esperienza solida e controllata, minimizzando gli errori e garantendo un'elevata personalizzazione secondo il contesto applicativo specifico. Di seguito riepilogo gli step principali:

Step 1: Configurazione del Device

- **Obiettivo**:
  Inizializzare il dispositivo impostando i parametri di base.

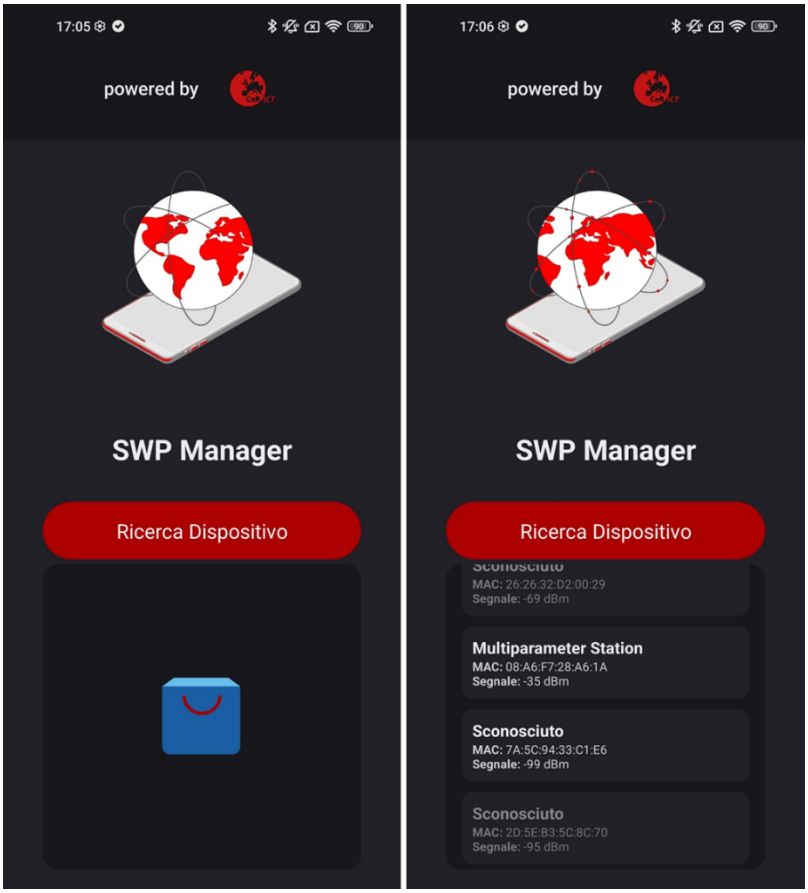


Figura 58: ricerca del dispositivo

- **Funzionalità**:

- **Impostazione del Nome**:

  L’utente inserisce un nome per il dispositivo. *Validazione*: Controllo su lunghezza e caratteri ammessi per garantire un formato valido.

- **Recupero della Geolocalizzazione**:

  Utilizzo dell’API di geolocalizzazione per ottenere latitudine e longitudine. *Tecnologia*: Richieste asincrone e gestione dei permessi utente.

- **Selezione del Tipo di Configurazione**:

  L’utente può scegliere tra diverse modalità operative che influenzano i passaggi successivi.

- **Configurazione Dinamica**:

  Il client adatta la procedura in base alla scelta effettuata, abilitando opportune schermate per WiFi o LTE e per il tipo di protocollo (MQTT o Socket).

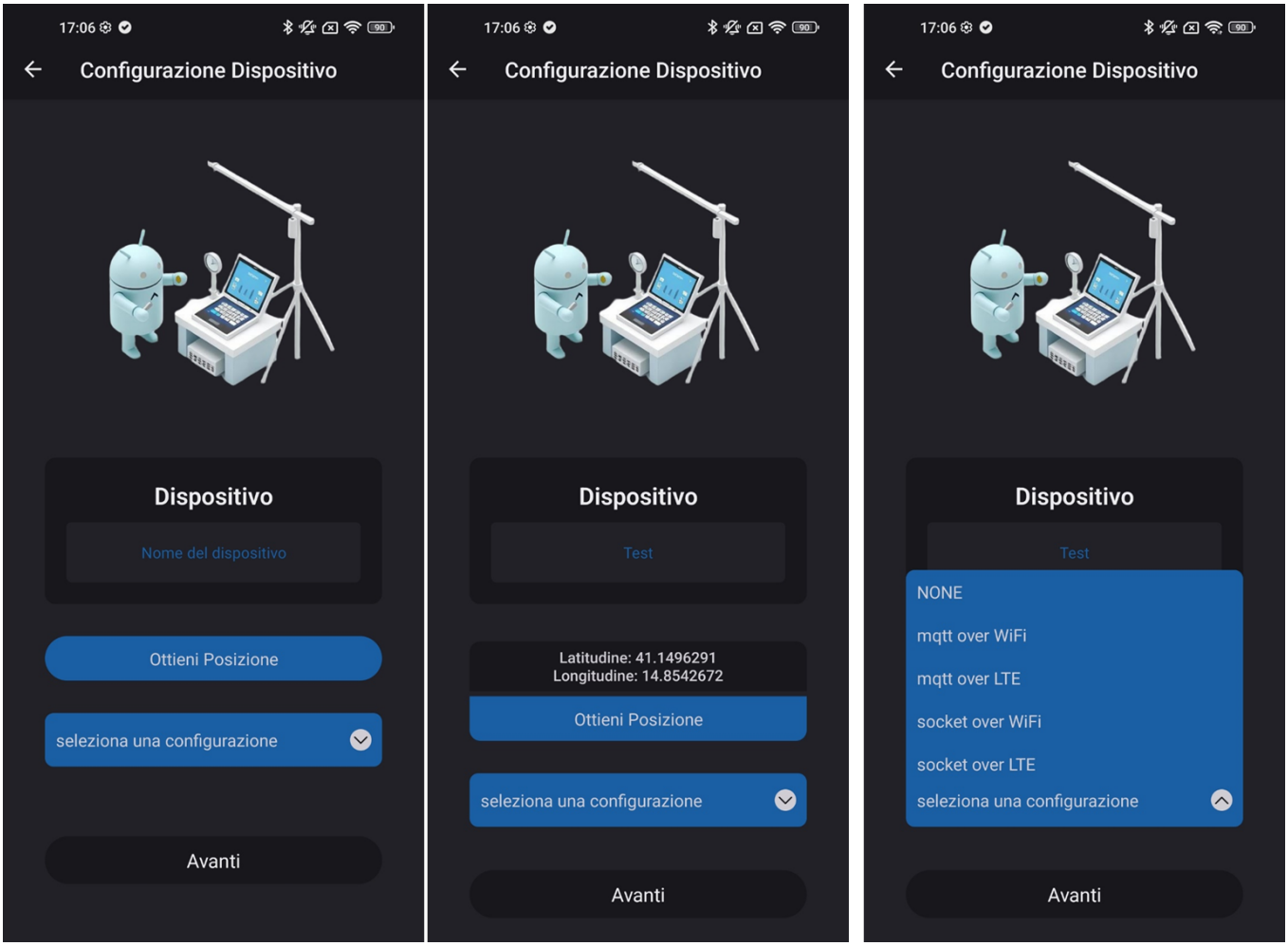


Figura 59: Configurazione del dispositivo

Step 2: Configurazione di Rete

- **Obiettivo**:
  Configurare la connessione di rete per la trasmissione dei dati al backend.
- **Funzionalità**:
  - **Ricerca delle Reti Disponibili**:
    - Per WiFi: Scansione delle reti wireless disponibili ed elenco di SSID rilevati.

        - Per LTE: Verifica della presenza di una rete LTE e impostazione dei parametri di connessione (non implementato).

- **Configurazione dei Parametri di Connessione**: Impostazione di SSID e password per stabilire la connessione.
- **Test della Connessione**:

  L'utente può effettuare una verifica per assicurarsi che la connessione funzioni correttamente prima di procedere.

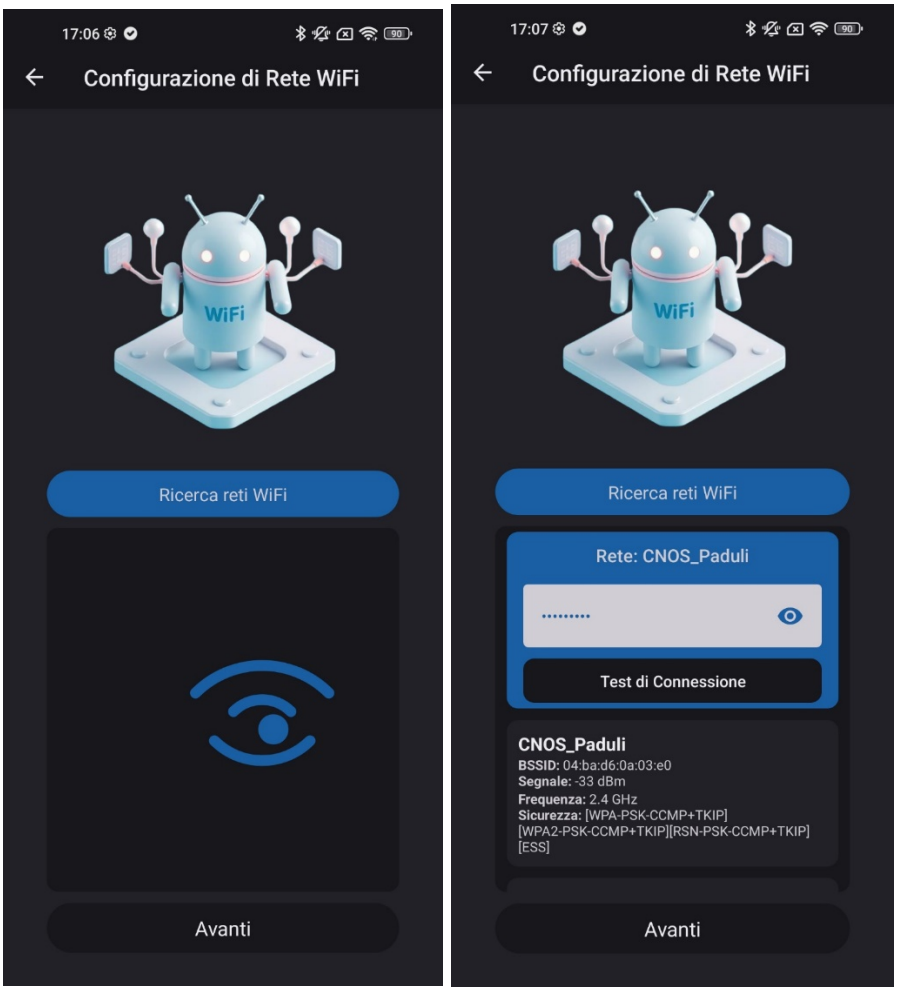


Figura 60: Configurazione della rete

Step 3: Configurazione del Protocollo di Comunicazione

- **Obiettivo**:
  Impostare i parametri per la comunicazione con il backend che gestirà le interazioni di tipo MQTT o Socket.

- **Funzionalità**:

  - **Definizione dei Parametri di Connessione**:

- **Endpoint**: Inserimento dell'indirizzo del backend e del numero di porta.
- **Credenziali**: Inserimento di username e password per l'autenticazione, garantendo un accesso sicuro.

- **Validazione Interattiva**:

Verifica in tempo reale dei parametri inseriti e feedback immediato in caso di errore, mediante messaggi di alert e validatori personalizzati.

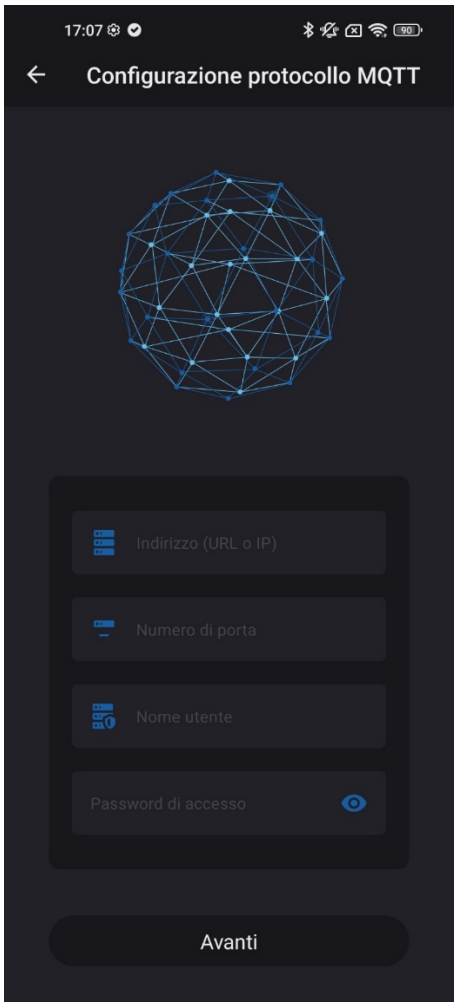


Figura 61: Configurazione protocolli di comunicazione

Step 4: Configurazione dei Certificati di Rete per TLS

- **Obiettivo**:
  Fornire un'interfaccia per gestire la sicurezza della connessione mediante certificati TLS.

- **Funzionalità**:

  - **Caricamento dei Certificati**:

    - **Certificato CA**: Importazione del certificato dell'Autorità di Certificazione.
    - **Certificato del Client e Chiave**: Caricamento del certificato e della chiave privata del dispositivo.

  - **Aggiornamento dei Certificati**: Possibilità di aggiornare uno o più certificati esistenti, garantendo la continuità della sicurezza.

  - **Rimozione dei Certificati**: Se non selezionati, il client consente la rimozione totale dei certificati.
    Nota: Ogni operazione è accompagnata da una validazione dei formati e un controllo dell'integrità del file.

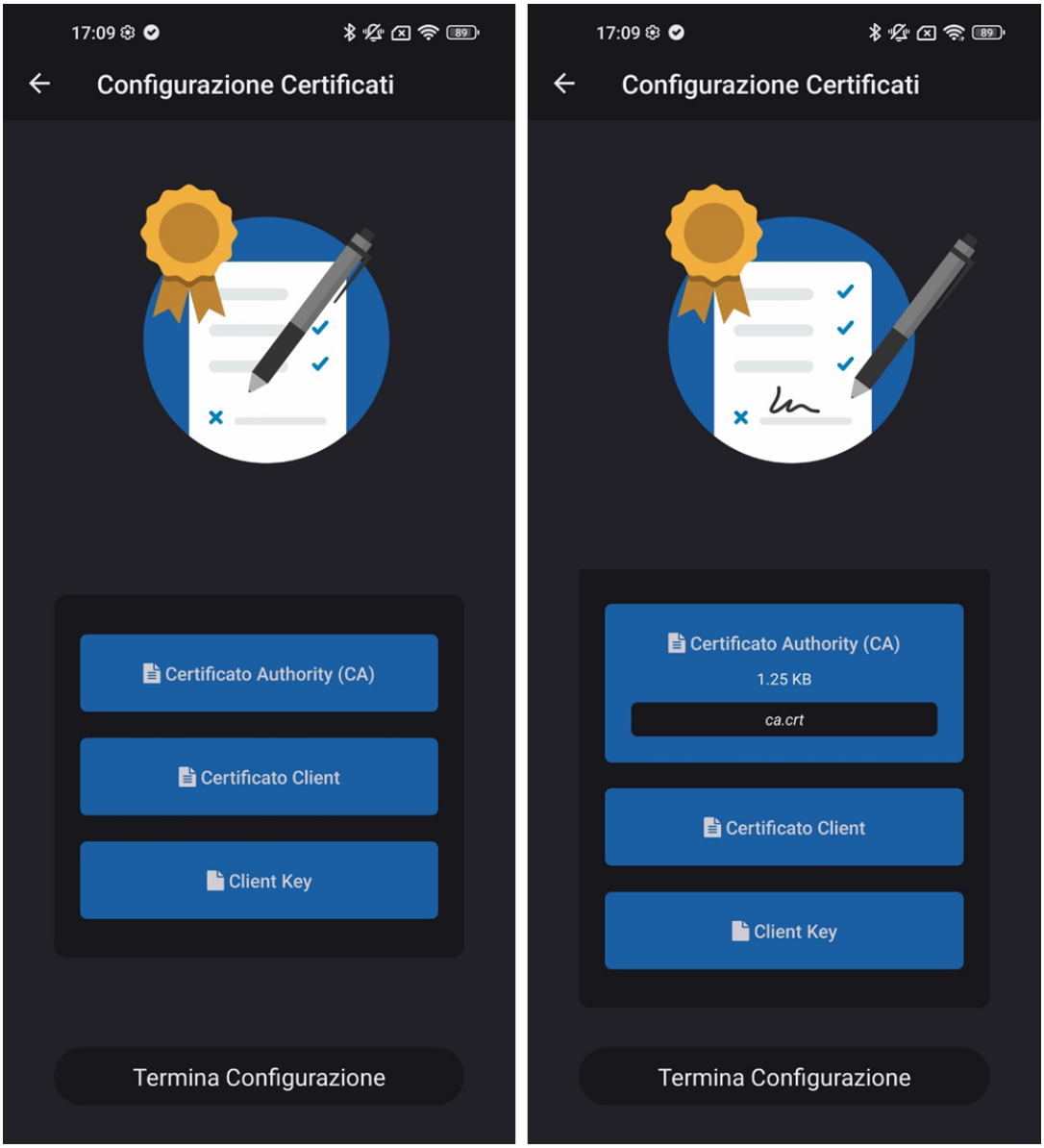


Figura 62: Configurazione dei certificati

10.4. Sintesi e Invio della Configurazione

Al termine della configurazione guidata, il client fornisce un riepilogo completo delle informazioni inserite, permettendo all'utente di rivedere ogni singolo parametro prima di proseguire. Questo passaggio finale è

fondamentale per garantire che tutte le configurazioni siano corrette e pronte per essere applicate al dispositivo. Il processo si articola in diverse fasi, che comprendono la generazione di un pacchetto di configurazione, l'invio di tale pacchetto alla board ESP32, e la gestione dei certificati di sicurezza.

Il primo passo consiste nella generazione del pacchetto di configurazione, dove i dati raccolti durante la procedura guidata vengono raggruppati e organizzati in un formato strutturato, pronto per essere inviato al dispositivo. Questo pacchetto include tutte le impostazioni configurate dall'utente, come il nome del dispositivo, la modalità di rete, le credenziali di connessione, e i parametri di comunicazione. Una volta creato, il pacchetto viene trasmesso alla board ESP32 tramite Bluetooth Low Energy (BLE), una tecnologia scelta per la sua efficienza energetica e la sua capacità di garantire una comunicazione stabile su brevi distanze.

Per assicurare una comunicazione senza interruzioni, il sistema include un meccanismo di retry, che interviene nel caso in cui un'operazione di invio fallisca. Se il dispositivo non riceve correttamente il pacchetto, il client tenta di inviarlo nuovamente, con un controllo del tempo (timeout) per evitare blocchi prolungati del processo. Questo meccanismo assicura che la configurazione venga trasmessa correttamente, riducendo il rischio di errori.

Un altro elemento cruciale riguarda la gestione dei certificati TLS, utilizzati per garantire la sicurezza della connessione tra il dispositivo e il backend. Durante questa fase, il client si occupa del caricamento, dell'aggiornamento e, se necessario, della rimozione dei certificati TLS, essenziali per stabilire un canale crittografato sicuro. Poiché la gestione dei certificati implica operazioni delicate, è previsto un ulteriore processo di autenticazione per garantire che solo l'utente autorizzato possa apportare modifiche a queste configurazioni critiche.

In altri termini, una volta completata la configurazione e trasmesse tutte le informazioni al dispositivo, il client esegue una disconnessione dalla board ESP32. Questo passo ripristina il client allo stato iniziale, pronto per una nuova sessione di configurazione, garantendo che ogni processo venga completato in modo ordinato e sicuro. Una volta completati i passaggi della configurazione guidata, il client presenta un riepilogo dettagliato dei dati inseriti, consentendo all'utente di verificare ogni parametro. Il processo prevede:

1. **Generazione del Pacchetto di Configurazione**: I dati raccolti vengono strutturati in un pacchetto dati che contiene tutte le impostazioni necessarie per la configurazione della board ESP32.

2. **Invio del Pacchetto alla Board ESP32**: Il pacchetto viene trasmesso via BLE. *Meccanismo di Retry*: Sono implementate logiche di timeout durante la comunicazione.

3. **Gestione dei Certificati per la Connessione Crittografata**: In maniera analoga a quanto descritto per il pacchetto di configurazione, vengono gestite operazioni di caricamento, aggiornamento o rimozione

dei certificati TLS (per cui è necessaria una ulteriore autenticazione), garantendo la sicurezza del canale di comunicazione verso il backend.

4. **Disconnessione e Reset del Client**: Una volta completata la configurazione, il client si disconnette automaticamente dalla board e ripristina lo stato iniziale, pronto per una nuova sessione di configurazione.

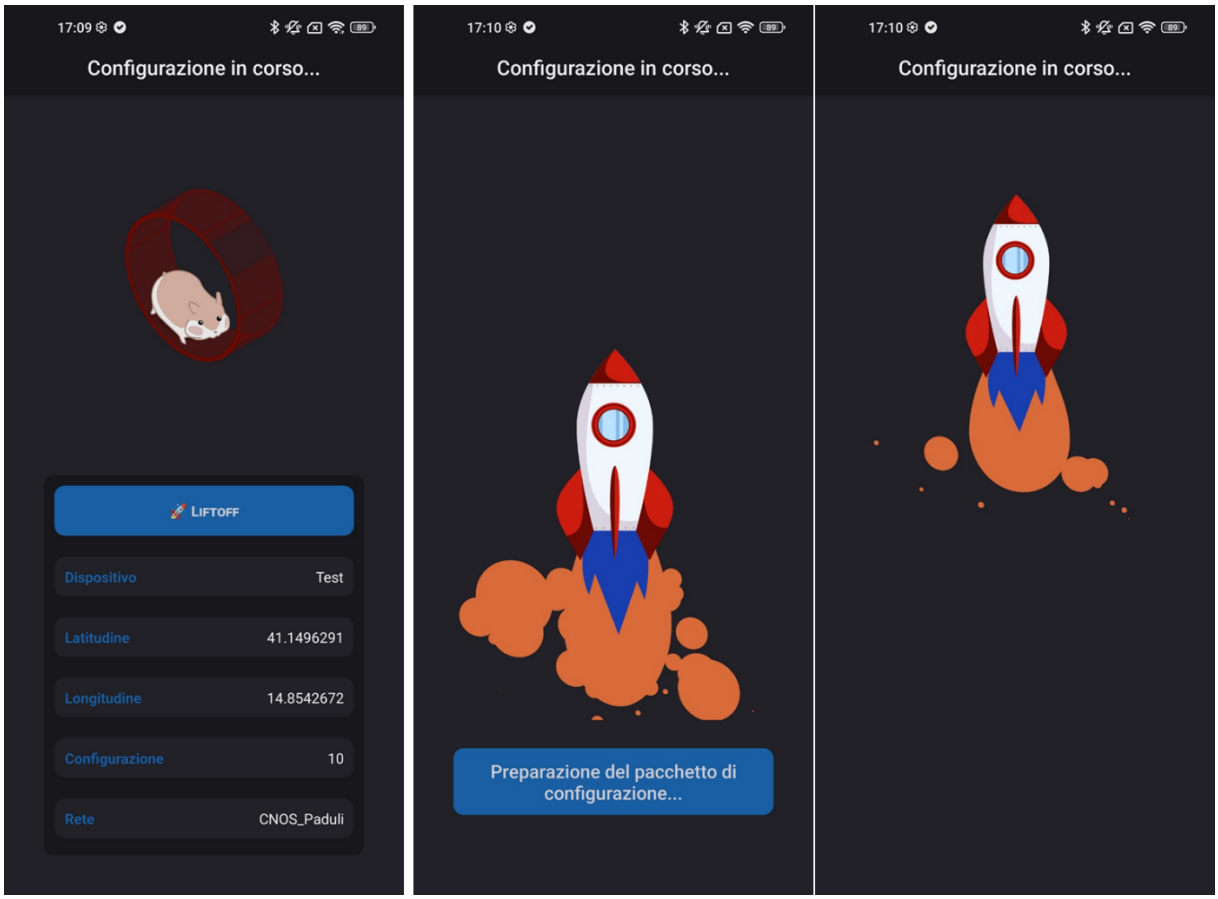


Figura 63: configurazione in corso

10.5. Modalità di Configurazione Manuale

La modalità di configurazione manuale, sebbene attualmente non implementata nel client, rappresenta una funzionalità avanzata che offre agli utenti una gestione più dettagliata e specifica della board. Questa modalità

si avvale delle capacità della libreria di gestione della board, permettendo una serie di operazioni avanzate e personalizzate. In particolare, una delle principali funzioni è il recupero della configurazione corrente, che consente di leggere direttamente i parametri attuali della board, fornendo una visione chiara dello stato del dispositivo.
Inoltre, l'utente ha la possibilità di rimuovere la configurazione corrente, un'operazione che consente di cancellare le impostazioni precedenti, con l'opzione di resettare completamente il dispositivo. Questo permette una gestione più fluida e flessibile, utile soprattutto in contesti di troubleshooting o quando si desidera iniziare una nuova configurazione da zero. Per garantire maggiore sicurezza e affidabilità, la modalità manuale offre anche la funzione di ripristino dello stato precedente, basata sui backup automatici generati ad ogni aggiornamento del dispositivo. Questa funzione è fondamentale per il recupero dei dati in caso di errori, riducendo il rischio di perdita di configurazioni importanti.
La gestione avanzata dei sensori è un altro aspetto cruciale della modalità manuale. Gli utenti possono recuperare e aggiornare la configurazione dei sensori, nonché impostare le preferenze operative per ciascun sensore, come l'abilitazione o disabilitazione a runtime o all'avvio. Questo consente un controllo dettagliato sulla funzionalità dei sensori, adattandoli alle esigenze specifiche dell'applicazione. Inoltre, la modalità manuale consente di acquisire i dati dai sensori, offrendo così la possibilità di monitorare in tempo reale i parametri misurati dal dispositivo.
Infine, la modalità manuale include anche operazioni di manutenzione e monitoraggio, che permettono di cancellare completamente la memoria del dispositivo e recuperare lo stato della batteria, ove disponibile. Questi strumenti garantiscono che il dispositivo possa essere gestito e mantenuto in maniera ottimale, anche nel lungo termine, fornendo all'utente tutte le risorse necessarie per un controllo preciso e continuo del sistema.
Ecco un riepilogo della modalità di configurazione manuale (attualmente non implementata nel client) che sfrutta le funzionalità esposte dalla libreria di gestione della board. Tale modalità consente operazioni avanzate quali:

- **Recupero della Configurazione Corrente**:

  Permette di leggere i parametri di configurazione attuali dalla board.

- **Rimozione della Configurazione Corrente**:

  Consente di eliminare le impostazioni già applicate, con possibilità di resettare il dispositivo.

- **Ripristino dello Stato Precedente**:

Il dispositivo genera backup automatici ad ogni aggiornamento. La modalità manuale offre la possibilità di ripristinare lo stato precedente in caso di errore.

- **Gestione Avanzata dei Sensori**:
    - Recupero e aggiornamento della configurazione dei sensori.
    - Impostazioni operative (abilitazione/disabilitazione a runtime o all'avvio).
    - Acquisizione dei dati dai sensori.
- **Manutenzione e Monitoraggio**:
    - Cancellazione completa della memoria.
    - Recupero dello stato della batteria (ove disponibile).

10.6. Considerazioni Tecniche Avanzate

La validazione dei dati inseriti è un passo fondamentale per garantire la corretta configurazione del dispositivo. Ogni input fornito dall'utente è sottoposto a una rigorosa validazione lato client. A tal fine, sono stati implementati validatori personalizzati per ciascun campo di input, assicurando che le informazioni siano nel formato corretto. Per esempio, il nome del dispositivo, l'SSID della rete, l'indirizzo IP e i numeri di porta vengono controllati per verificare che rispettino gli standard previsti, evitando errori che potrebbero compromettere la funzionalità del sistema. Inoltre, i certificati TLS e le chiavi private sono validati per garantire che siano nel formato richiesto, come PEM, CRT o KEY, fondamentali per la sicurezza della connessione.
Per quanto riguarda i permessi, la validazione si estende anche alla verifica della presenza dei permessi necessari, come quelli per la geolocalizzazione e l'accesso al Bluetooth, senza i quali alcune funzionalità cruciali non sarebbero eseguibili. Questo approccio garantisce che l'utente sia sempre a conoscenza di eventuali mancanze o incongruenze, riducendo il rischio di malfunzionamenti.
In caso di errori, il sistema offre una gestione intelligente tramite un'interfaccia utente chiara e dettagliata. Se si verifica un errore durante il processo di configurazione, vengono visualizzati messaggi espliciti che descrivono il problema, accompagnati da suggerimenti su come risolverlo. Questo approccio aiuta l'utente a correggere facilmente eventuali malintesi senza compromettere la user experience. Inoltre, tutte le attività e gli errori vengono registrati in un sistema di logging e monitoraggio, il quale consente di tenere traccia degli eventi. Questo non solo facilita il debug durante lo sviluppo, ma aiuta anche nella diagnosi e risoluzione dei problemi in fase di produzione.

Sicurezza e Crittografia

La sicurezza è un aspetto cruciale nella gestione della configurazione dei dispositivi, soprattutto quando si tratta di connessioni a reti sensibili. Il sistema adotta il protocollo TLS per garantire che la comunicazione tra il client e il backend avvenga in modo sicuro e criptato. La gestione dei certificati TLS è realizzata con il massimo rigore. Gli utenti possono caricare certificati in modo sicuro, con un controllo approfondito dell'integrità e della validità dei file, assicurando che non vi siano compromissioni della sicurezza. Inoltre, il sistema consente di aggiornare o revocare i certificati in qualsiasi momento, mantenendo alta la conformità alle best practice di sicurezza e permettendo al sistema di adattarsi a eventuali nuovi requisiti di sicurezza.
Accanto alla gestione dei certificati, sono previsti meccanismi di autenticazione e autorizzazione che limitano l'accesso alle funzionalità sensibili. Il dispositivo e l'utente vengono autenticati prima di consentire l'accesso alla configurazione, garantendo che solo gli utenti legittimati possano intervenire sulle impostazioni e sulla gestione del dispositivo.

Estendibilità e Manutenzione

L'architettura del client è progettata per essere altamente modulare, con componenti indipendenti che gestiscono funzionalità specifiche come la ricerca di dispositivi, la configurazione della rete, e la gestione dei certificati. Questo design facilita gli aggiornamenti e la manutenzione, consentendo agli sviluppatori di intervenire su singole sezioni del sistema senza compromettere l'intero flusso. Inoltre, ogni modulo è pensato per integrarsi facilmente con futuri aggiornamenti, sia della board ESP32 che dei protocolli di comunicazione utilizzati. La possibilità di aggiornare e personalizzare singoli moduli permette una gestione ottimale delle risorse e un'evoluzione del sistema in risposta alle nuove necessità.
Un altro punto di forza è la possibilità di aggiungere una modalità manuale, che, sebbene non ancora implementata, è prevista per permettere operazioni avanzate di configurazione. Questa modalità offrirà agli sviluppatori strumenti aggiuntivi per la gestione e il monitoraggio dei dispositivi, migliorando la flessibilità della piattaforma.
Validazione e Gestione degli Errori

- **Validazione dei Campi**:

  Ogni input inserito dall'utente è sottoposto a validazione lato client. Sono stati implementati form validator personalizzati per:

  - Verificare il formato del nome, SSID, indirizzo IP e numeri di porta.

- Assicurarsi che i certificati e le chiavi rispettino i formati attesi (PEM, CRT, KEY).
- Controllare la presenza dei permessi necessari (ad es. geolocalizzazione, accesso Bluetooth).

- **Gestione degli Errori**:
  - **Interfaccia Utente (UI)**: In caso di errore, vengono mostrati messaggi dettagliati per facilitare la correzione del problema.
  - **Logging e Monitoraggio**: Registrazione degli eventi e degli errori per facilitare la diagnosi e il debugging in fase di sviluppo e in produzione.

Sicurezza e Crittografia

- **TLS e Certificati**:

  La gestione dei certificati TLS garantisce la sicurezza della comunicazione con il backend.

  - **Importazione Sicura**: I certificati vengono caricati in modo sicuro, con controllo dell'integrità e validità.
  - **Aggiornamenti e Revoche**: Consente operazioni di aggiornamento e revoca, garantendo la conformità alle best practice di sicurezza.

- **Autenticazione e Autorizzazione**:
  - Il sistema prevede meccanismi per autenticare il dispositivo e l'utente, limitando l'accesso alle funzioni di configurazione.

Estendibilità e Manutenzione

- **Modularità del Codice**:
- Il client è strutturato in moduli indipendenti per ciascuna funzionalità (ricerca, configurazione di rete, gestione certificati, ecc.).
  - Questa architettura facilita l'aggiornamento di singoli componenti senza compromettere l'intero sistema.

- **Aggiornamenti e Compatibilità**:

- La soluzione è pensata per integrarsi con futuri aggiornamenti della board ESP32 e dei protocolli di comunicazione.
- La modalità di configurazione manuale, anche se non ancora implementata, offre un percorso per future estensioni e per l'aggiunta di funzionalità personalizzate.

Il client di configurazione per la Smart Work Platform rappresenta una soluzione completa e versatile per la gestione dei dispositivi basati su ESP32. Grazie all'utilizzo di tecnologie moderne come **React Native** ed **Expo**, il sistema permette:

- Una facile configurazione tramite procedura guidata.
- Supporto per differenti modalità di rete (WiFi, LTE) e protocolli di comunicazione (MQTT, Socket).
- Una robusta gestione della sicurezza grazie alla configurazione TLS e alla validazione dei dati.
- La possibilità di estendere le funzionalità tramite una futura modalità manuale, offrendo agli sviluppatori strumenti avanzati per la manutenzione e il monitoraggio dei dispositivi.

Il client di configurazione per la Smart Work Platform è una soluzione versatile e completa che facilita la gestione dei dispositivi IoT basati su ESP32. Utilizzando tecnologie moderne come React Native e Expo, il sistema consente una configurazione intuitiva e sicura, supportando diverse modalità di rete e protocolli di comunicazione. La gestione dei certificati TLS e l'implementazione di meccanismi di validazione e monitoraggio degli errori garantiscono che la configurazione avvenga senza intoppi e in totale sicurezza.
Inoltre, la progettazione modulare e la possibilità di estendere il sistema con funzionalità avanzate, come la modalità manuale, pongono le basi per una piattaforma altamente scalabile e facilmente integrabile con futuri sviluppi. La documentazione fornita non solo è una guida per gli sviluppatori e i manutentori, ma rappresenta anche un riferimento per le migliori pratiche di configurazione e comunicazione con il dispositivo ESP32.

11. Conclusioni

Il documento ha riportato i risultati delle attività A3.2 Studio ed identificazione dei componenti hardware (sensori ed elettronica di condizionamento a supporto) e dell'attività A3.3 Implementazione/integrazione dell'hardware e firmware presenti nel WP3 dal titolo Studio, progettazione ed implementazione dei componenti hardware e firmware del prototipo IoT

L'attività è stata completata con successo, senza anomalie o criticità.