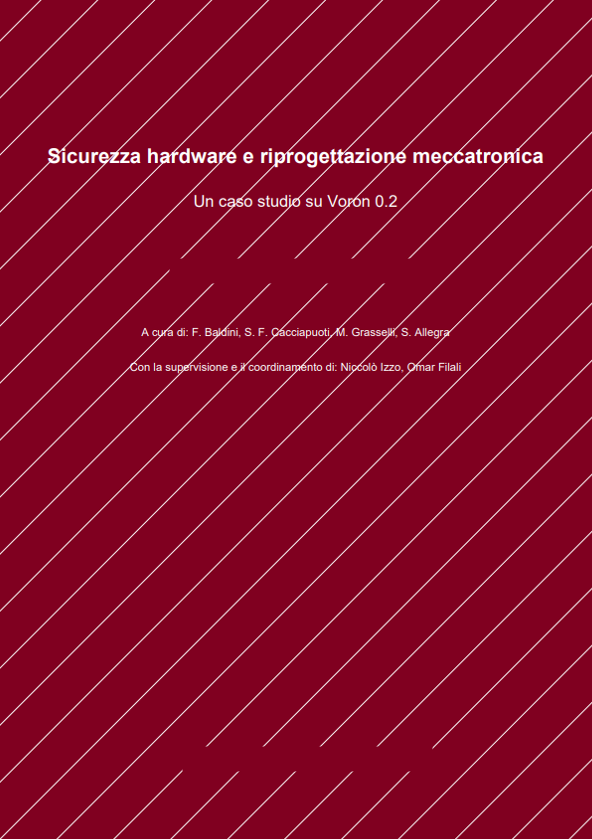
**t**

**SICUREZZA HARDWARE E Riprogettazione Meccatronica: Un Caso Studio su Voron 0.2**

A close up of words

AI-generated content may be incorrect.**Percorsi per le Competenze Trasversali e per l'Orientamento**  
**A cura di:** F. Baldini, S. F. Cacciapuoti, M. Grasselli, S. Allegra  
**Con la supervisione e il coordinamento di:** Niccolò Izzo, Omar Filali

*Il seguente documento propone lo sviluppo di un sistema meccatronico tridimensionale a cinematica parallela, progettato interamente secondo i principi dell’open source e dell’open hardware. L’obiettivo è realizzare una piattaforma versatile e accessibile per l’analisi sperimentale degli effetti degli impulsi elettromagnetici (EMP) su dispositivi elettronici, con particolare attenzione alle memorie non volatili. Il sistema, dotato di tre gradi di libertà e di un’interfaccia di controllo programmabile, è pensato per essere facilmente replicabile e adattabile da parte di scuole, università e centri di ricerca. L’impiego di componenti standard e a basso costo ne favorisce la diffusione in contesti educativi, offrendo uno strumento concreto per lo studio di fenomeni fisici complessi legati alle interferenze elettromagnetiche.*

Contenuti

[CONTESTO E OBIETTIVI DEL PROGETTO 3](#_Toc201828488)

[Analisi delle Memorie NAND: Limiti e Vulnerabilità 3](#_Toc201828489)

[Analisi dell’Attacco EMFI: Principi e Implicazioni 4](#_Toc201828490)

[Genesi del Progetto: Dall’Analisi alla Proposta di Soluzione 5](#_Toc201828491)

[REQUISITI 5](#_Toc201828492)

[Requisiti Funzionali 5](#_Toc201828493)

[Requisiti Non Funzionali 6](#_Toc201828494)

[Vincoli 6](#_Toc201828495)

[ARCHITETTURA DEL SISTEMA 7](#_Toc201828496)

[Hardware 7](#_Toc201828497)

[Modello 3D 7](#_Toc201828498)

[Software 8](#_Toc201828499)

[Marlin 8](#_Toc201828500)

[Ambiente di Sviluppo 8](#_Toc201828501)

[Librerie 10](#_Toc201828502)

[IMPLEMENTAZIONE 10](#_Toc201828503)

[RIPROGETTAZIONE HARDWARE DELLA STAMPANTE 3D: OBIETTIVI E SCELTE TECNICHE 10](#_Toc201828504)

[Verso una Nuova Architettura di Stampante 3D 10](#_Toc201828505)

[Stampante di Riferimento: Voron 0.2 11](#_Toc201828506)

[Il Generatore di Impulsi EMFI**:** Faulty Cat 12](#_Toc201828507)

[Soluzione Realizzata: Il Prodotto Hardware Finale 12](#_Toc201828508)

[RICONFIGURAZIONE SOFTWARE DELLA STAMPANTE 3D: OBIETTIVI E SCELTE TECNICHE 13](#_Toc201828509)

[Dal Concetto al Codice: Linee Guida per lo Sviluppo 13](#_Toc201828510)

[Integrazione del Firmware Marlin 13](#_Toc201828511)

[Integrazione tra Moduli 14](#_Toc201828512)

[Logiche di Controllo 14](#_Toc201828513)

[Soluzione Realizzata: Il Prodotto Software Finale 15](#_Toc201828514)

[CONFIGURAZIONE E SETUP 15](#_Toc201828515)

[Istruzioni per l’Assemblaggio della Stampante 3D 15](#_Toc201828516)

[Installazione del Firmware 19](#_Toc201828517)

[Parametri di configurazione 20](#_Toc201828518)

[RIFERIMENTI 20](#_Toc201828519)

# CONTESTO E OBIETTIVI DEL PROGETTO

## Analisi delle Memorie NAND: Limiti e Vulnerabilità

Le memorie NAND costituiscono una tipologia di memoria non volatile, ovvero in grado di conservare i dati anche in assenza di alimentazione elettrica. Esse si basano su una struttura a celle di memoria che impiega transistor MOSFET dotati di *floating gate*, un elemento elettricamente isolato all’interno del quale viene intrappolata una carica elettrica, utilizzata per rappresentare lo stato logico del bit.

A seconda della tecnologia impiegata, ogni cella è in grado di memorizzare uno o più bit: le celle SLC (*Single-Level Cell*) memorizzano un solo bit, le MLC (*Multi-Level Cell*) due, le TLC (*Triple-Level Cell*) tre e le QLC (*Quad-Level Cell*) quattro. L’aumento della densità di memorizzazione comporta, tuttavia, una riduzione dell’affidabilità e della durata operativa del dispositivo.

Le celle sono organizzate in pagine e blocchi: la scrittura avviene a livello di pagina, mentre la cancellazione è possibile solo a livello di blocco. Tale architettura rende la gestione della memoria particolarmente complessa, demandandola a un controller interno incaricato di eseguire operazioni fondamentali quali il *wear leveling* (distribuzione uniforme dell’usura), la *garbage collection* (riciclo dei blocchi inutilizzati) e la correzione degli errori mediante codici ECC (*Error Correction Code*).

Le memorie NAND trovano ampia applicazione in dispositivi quali SSD (*Solid State Drive*), smartphone, SD cards (*Secure Digital Cards*) e unità flash USB (*Universal Serial Bus*), grazie all’elevata densità di integrazione e al basso costo per bit, caratteristiche che le rendono particolarmente adatte all’archiviazione di grandi volumi di dati in spazi ridotti. Tuttavia, esse presentano vulnerabilità intrinseche che ne compromettono l’affidabilità nel tempo. Tra queste si annoverano: la perdita di carica nel *floating gate*, l’usura derivante dai cicli ripetuti di programmazione e cancellazione, e la suscettibilità a disturbi elettromagnetici, i quali possono generare errori temporanei o permanenti nei dati memorizzati.

Una criticità di rilievo è rappresentata dall’assenza di meccanismi di protezione ECC in aree sensibili della memoria, come quelle destinate alla memorizzazione delle chiavi di autenticazione. In tali contesti, anche un singolo *bit flip* (inversione del valore logico di un bit) può compromettere l’integrità del sistema, facilitare attacchi fisici, come l’EM-FI (*Electromagnetic Fault Injection*), e consentire l’accesso non autorizzato al dispositivo mediante la manipolazione selettiva di dati sensibili.

La comunicazione tra la memoria NAND e il processore avviene tramite interfacce standardizzate, quali ONFI (*Open NAND Flash Interface*) o JEDEC (*Joint Electron Device Engineering Council*). In alcuni dispositivi è inoltre implementato un meccanismo di autenticazione tra il SoC (*System on Chip*) e la memoria NAND, basato su chiavi crittografiche condivise o funzioni hash, al fine di garantire l’autenticità del componente e prevenire la sostituzione con unità contraffatte.

In tale contesto, un attacco fisico come l’EMFI, che sfrutta impulsi elettromagnetici per indurre errori nei dati memorizzati, può essere impiegato per alterare selettivamente un bit della chiave crittografica presente in una NAND contraffatta. L’obiettivo è quello di farla accettare come legittima dal sistema, compromettendo l’integrità del dispositivo e consentendo l’accesso non autorizzato.

## Analisi dell’Attacco EMFI: Principi e Implicazioni

Gli impulsi elettromagnetici (EMP) rappresentano una minaccia concreta e sofisticata per la sicurezza dei sistemi elettronici contemporanei, con particolare riferimento alle memorie NAND Flash, largamente impiegate nei dispositivi mobili in virtù della loro elevata densità di integrazione e della capacità di conservazione dei dati non volatili.

In scenari avanzati di attacco, un impulso elettromagnetico può essere impiegato come vettore per l’alterazione mirata e selettiva di dati critici all’interno di una memoria NAND contraffatta, con l’intento di compromettere i meccanismi di autenticazione incrociata implementati nei dispositivi.

Nel contesto di un attacco fisico, un soggetto non autorizzato potrebbe installare una memoria NAND contraffatta contenente una chiave crittografica quasi identica a quella originale, differente eventualmente per un solo bit. In tale scenario, mediante l’impiego di un generatore open-source di impulsi elettromagnetici ad alta energia, Faulty Cat, generati da un circuito di scarica capacitive; posizionato in prossimità del chip NAND, l’agente ostile può eseguire un attacco di tipo EMFI, sincronizzando l’impulso con la fase di lettura della chiave da parte del SoC. Se correttamente calibrato, l’impulso elettromagnetico può indurre un bit flip.

Sebbene tale alterazione possa apparire trascurabile, essa può generare conseguenze critiche: qualora il bit modificato corrisponda a quello necessario per far coincidere la chiave della NAND contraffatta con quella attesa dal SoC, il dispositivo autenticherebbe erroneamente la memoria come legittima, consentendo così l’accesso completo al sistema. Questa vulnerabilità risulta ulteriormente aggravata dall’assenza, in numerose implementazioni, di meccanismi di correzione degli errori nelle aree critiche della memoria, in particolare quelle destinate all’autenticazione, rendendo possibile la manipolazione dei dati sensibili senza che vengano rilevati errori.

Tale tipologia di attacco evidenzia come la sicurezza dei sistemi embedded non possa prescindere da un’analisi approfondita delle vulnerabilità fisiche, in particolare nei confronti di interferenze elettromagnetiche intenzionali. L’alterazione mirata di un singolo bit all’interno di una chiave di autenticazione, ottenuta mediante un impulso EMP, dimostra che anche i meccanismi crittografici più robusti possono essere aggirati in assenza di adeguate contromisure a livello hardware.

## Genesi del Progetto: Dall’Analisi alla Proposta di Soluzione

L’idea fondante alla base del presente progetto consiste nello sviluppo di un sistema meccatronico tridimensionale, concepito come un treppiede a cinematica parallela in grado di muoversi nello spazio secondo tre gradi di libertà, interamente progettato secondo i principi dell’open source e dell’open hardware.

L’obiettivo primario è quello di realizzare una piattaforma versatile e accessibile, destinata allo studio sperimentale degli effetti indotti da EMP su dispositivi elettronici, con particolare riferimento alla compromissione delle memorie non volatili come le NAND Flash. Il sistema sarà dotato di un’interfaccia di controllo programmabile e documentazione tecnica completa, in modo da garantire la massima trasparenza e replicabilità del progetto.

La scelta di adottare un’architettura open hardware consente non solo la libera distribuzione dei file di progetto, ma anche la possibilità di personalizzazione e adattamento da parte di enti terzi, come istituti scolastici, laboratori universitari e centri di ricerca. Un ulteriore punto di forza dell’iniziativa risiede nella sostenibilità economica: la costruzione del treppiede si basa su componenti standard e facilmente reperibili, con un costo complessivo contenuto e compatibile con i budget tipici delle realtà educative e accademiche.

In questo modo, il progetto si propone come strumento didattico e sperimentale ad alto valore tecnologico, capace di coniugare accessibilità, rigore scientifico e applicabilità concreta nello studio di fenomeni fisici complessi come le interferenze elettromagnetiche.

# REQUISITI

## Requisiti Funzionali

Il sistema sviluppato si configura come una piattaforma meccatronica tridimensionale a cinematica parallela, destinata all’analisi sperimentale degli effetti degli impulsi elettromagnetici su dispositivi elettronici, con particolare riferimento alle memorie non volatili di tipo NAND. In tale contesto, i requisiti funzionali definiscono l’insieme delle capacità operative che il sistema deve garantire per adempiere efficacemente agli obiettivi progettuali.

In primo luogo, esso deve consentire il posizionamento tridimensionale ad alta precisione di una sonda elettromagnetica, con risoluzione micrometrica, al fine di indirizzare in modo selettivo gli impulsi verso aree specifiche del dispositivo bersaglio. Tale posizionamento deve essere ripetibile e controllabile, per assicurare la validità statistica delle prove sperimentali.

Il sistema dovrà inoltre integrare un’interfaccia di controllo basata su firmware Marlin, selezionato per le sue caratteristiche di modularità, robustezza e ampia compatibilità con architetture open-source. Marlin, firmware embedded progettato per microcontrollori a risorse limitate, si distingue per la sua capacità di gestire in tempo reale i movimenti degli assi, implementare logiche di sicurezza avanzate (come la protezione termica e il ripristino automatico dopo interruzioni) e supportare configurazioni meccaniche non convenzionali, come quelle a cinematica parallela.

La sua architettura modulare consente inoltre l’integrazione di sensori e attuatori personalizzati, rendendolo particolarmente adatto a contesti sperimentali ad alta specializzazione.

## Requisiti Non Funzionali

I requisiti non funzionali delineano le qualità generali del sistema, ovvero le caratteristiche che, pur non essendo direttamente legate alle funzionalità operative, ne determinano l’efficacia, l’affidabilità e l’idoneità all’uso in contesti educativi e di ricerca. In primo luogo, il sistema dovrà garantire un’elevata affidabilità, intesa come la capacità di mantenere il corretto funzionamento anche in presenza di condizioni operative avverse o anomalie. A tal fine, dovranno essere implementati meccanismi di gestione delle eccezioni, come il rilevamento di errori di homing, la perdita di alimentazione o il malfunzionamento di un asse, con conseguente attivazione di procedure di arresto sicuro e ripristino controllato.

Dal punto di vista delle prestazioni, il sistema dovrà assicurare tempi di risposta rapidi e precisione micrometrica nei movimenti, in modo da garantire l’efficacia degli esperimenti di iniezione elettromagnetica. La compatibilità con componenti open-source e open-hardware rappresenta un ulteriore requisito fondamentale, in quanto consente la replicabilità del progetto, la sua estensione da parte di terzi e la riduzione dei costi.

In termini di sicurezza, il sistema dovrà rispettare le normative vigenti in materia di utilizzo di dispositivi elettronici in ambito scolastico e formativo, come le linee guida del Ministero dell’Istruzione italiano, che promuovono l’uso consapevole e controllato delle tecnologie digitali.

## Vincoli

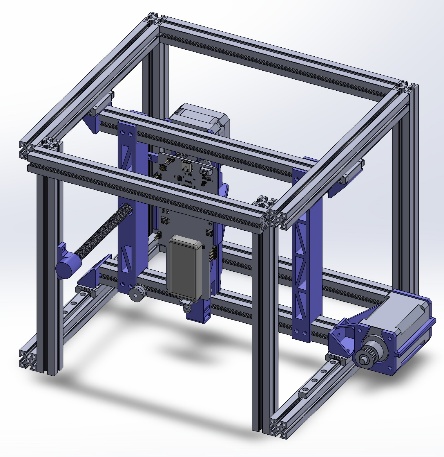
Il progetto è soggetto a una serie di vincoli che ne delimitano il perimetro operativo e ne influenzano le scelte progettuali. In primo luogo, il vincolo temporale impone il completamento dell’intero sistema entro un arco di due settimane, rendendo necessaria una pianificazione rigorosa delle attività e l’impiego di componenti, architetture o metodologie progettuali che sono state precedentemente testate e validate. Dal punto di vista economico, il progetto deve essere realizzato con un budget contenuto, compatibile con le risorse tipiche di un contesto scolastico o universitario. Ciò implica l’impiego di componenti standard, facilmente reperibili e a basso costo, senza compromettere la qualità e l’affidabilità del sistema.

A livello tecnologico, il progetto è vincolato all’utilizzo di architetture open-source, sia per quanto riguarda l’hardware sia per il software (firmware), al fine di garantire trasparenza, accessibilità e possibilità di personalizzazione. Infine, il rispetto delle normative tecniche e di sicurezza costituisce un vincolo imprescindibile: il sistema dovrà operare in conformità con le direttive europee in materia di compatibilità elettromagnetica (EMC), sicurezza elettrica e tutela della salute nei luoghi di apprendimento. Particolare attenzione dovrà essere riservata alla schermatura dei componenti sensibili, alla gestione delle emissioni elettromagnetiche e alla protezione degli operatori da eventuali esposizioni non intenzionali. Tali vincoli, sebbene stringenti, rappresentano una guida progettuale utile per garantire la qualità, la replicabilità e la sostenibilità del sistema sviluppato.

# ARCHITETTURA DEL SISTEMA

## Hardware

### Modello 3D



Il modello tridimensionale rappresenta la configurazione hardware del sistema meccatronico sviluppato, evidenziando una struttura a cinematica parallela basata su un treppiede rigido. La composizione meccanica si articola attorno a un telaio realizzato con profili in alluminio estruso, assemblati mediante giunti angolari e viteria standard, che garantiscono robustezza e modularità. Il sistema di movimentazione è affidato a tre assi indipendenti, ciascuno dotato di guide lineari ad alta precisione e viti a ricircolo di sfere, che assicurano scorrimenti fluidi e minimizzazione dell’attrito. I motori passo-passo, montati su supporti dedicati e accoppiati alle viti tramite giunti flessibili, forniscono il controllo del movimento lungo i tre gradi di libertà. Una piastra centrale, integrata nella struttura, ospita l’elettronica di controllo, inclusi i driver dei motori e l’interfaccia di comunicazione. L’intero sistema è concepito per essere replicabile e modificabile, in linea con i principi dell’open hardware, e costituisce la base fisica per l’esecuzione di test sperimentali sugli effetti degli impulsi elettromagnetici (EMP) su dispositivi elettronici, con particolare attenzione alle memorie non volatili.

## Software

### Marlin

Marlin è uno dei firmware open-source più longevi e ampiamente adottati nel panorama della stampa 3D, specificamente progettato per il controllo di macchine basate su tecnologia FDM (Fused Deposition Modeling). Si tratta di un sistema embedded, ovvero un software progettato per essere eseguito direttamente su un dispositivo hardware dedicato, dotato di risorse computazionali limitate ma ottimizzate per svolgere compiti specifici in tempo reale. In questo contesto, Marlin viene installato su schede di controllo elettronico equipaggiate con microcontrollori, che costituiscono il nucleo di elaborazione della stampante e gestiscono in modo autonomo tutte le funzioni operative: il movimento degli assi, la regolazione delle temperature, la lettura dei sensori e l’elaborazione dei comandi G-code.

La configurazione del firmware avviene principalmente attraverso due file: Configuration.h e Configuration\_adv.h. Il primo definisce i parametri fondamentali della macchina, tra cui la cinematica, le dimensioni dell’area di lavoro, i passi per millimetro dei motori, i limiti di temperatura e la presenza di sensori o finecorsa.

Il secondo estende queste impostazioni con funzionalità avanzate, come il livellamento automatico del piano, la gestione multi-estrusore, la calibrazione termica dell’hotend e del piano riscaldato, la protezione termica e la ripresa della stampa dopo interruzioni.

Marlin si distingue per la sua architettura modulare, la vasta compatibilità con componenti hardware di terze parti e una documentazione tecnica approfondita, che lo rendono adatto sia a utenti esperti sia a chi desidera approfondire il funzionamento interno delle stampanti 3D. In quanto progetto open-source, il suo codice sorgente è liberamente accessibile, modificabile e mantenuto da una comunità globale di sviluppatori, maker e produttori. Questo ha reso Marlin una piattaforma condivisa determinante per la diffusione e la democratizzazione della stampa 3D.

### Ambiente di Sviluppo

Per lo sviluppo, la personalizzazione e la gestione del firmware del sistema meccatronico, è stato predisposto un ambiente di sviluppo completo, basato su strumenti open source e compatibili con architetture embedded. Di seguito si riportano i componenti principali installati e configurati:

* Visual Studio Code  
  Editor di codice sorgente multipiattaforma, utilizzato come interfaccia principale per la scrittura, la modifica e la gestione del codice. Visual Studio Code è leggero ma estremamente potente, e offre funzionalità avanzate come l’evidenziazione della sintassi, il completamento automatico, il refactoring del codice e l’integrazione nativa con sistemi di controllo versione come Git. La sua modularità e l’ampia disponibilità di estensioni lo rendono particolarmente adatto allo sviluppo firmware in ambito open hardware.
* PlatformIO IDE  
  Estensione per Visual Studio Code progettata specificamente per lo sviluppo embedded. PlatformIO fornisce una toolchain completa per la compilazione, il caricamento e il monitoraggio del firmware su microcontrollori compatibili (es. AVR, ARM Cortex-M, ESP32, ecc.). Tra le funzionalità principali si annoverano:
* Gestione centralizzata delle librerie e delle dipendenze;
* Supporto per ambienti multipli, utili per mantenere configurazioni distinte per diverse revisioni hardware;
* Configurazione semplificata dei parametri di build e delle piattaforme target;
* Monitor seriale integrato per il debug in tempo reale;
* Compatibilità con sistemi di continuous integration e versionamento.

L’integrazione con Visual Studio Code consente un flusso di lavoro fluido e altamente produttivo, ideale per progetti complessi e collaborativi.

* Python 3.x  
  Linguaggio di scripting necessario per l’esecuzione degli script interni di PlatformIO. Python viene utilizzato per orchestrare i processi di build, la gestione delle dipendenze, la configurazione dinamica dei progetti e l’interfacciamento con dispositivi esterni. La presenza di un interprete Python correttamente installato è indispensabile per garantire il corretto funzionamento dell’ambiente PlatformIO su tutte le principali piattaforme operative (Windows, macOS, Linux).
* Git  
  Sistema di controllo versione distribuito utilizzato per il download del codice sorgente del firmware Marlin e per la gestione delle modifiche locali. Git consente di mantenere una cronologia dettagliata delle revisioni, facilitare il lavoro collaborativo e garantire la tracciabilità delle modifiche. L’integrazione con Visual Studio Code permette di eseguire operazioni Git direttamente dall’interfaccia grafica, semplificando la gestione dei rami, dei commit e delle fusioni. L’utilizzo di Git è particolarmente coerente con la filosofia open source del progetto, favorendo la trasparenza e la condivisione del codice.

### Librerie

Nel contesto dello sviluppo software del progetto, sono state utilizzate due librerie fondamentali per la gestione delle funzionalità di comunicazione e temporizzazione: serial e time. La libreria serial, fornita dal pacchetto esterno pyserial, è stata impiegata per stabilire e gestire la comunicazione seriale tra il sistema di controllo e dispositivi esterni, come microcontrollori o moduli embedded. Questa consente l’apertura di porte seriali, la trasmissione e ricezione di dati in formato byte, e l’implementazione di protocolli di comunicazione personalizzati, risultando essenziale per il monitoraggio e il controllo in tempo reale del posizionatore.

La libreria time, parte integrante della libreria standard di Python, è stata utilizzata per introdurre ritardi temporizzati, misurare intervalli di tempo e sincronizzare eventi critici all’interno del flusso di esecuzione. L’integrazione di queste due librerie ha reso possibile la realizzazione di uno script robusto, reattivo e compatibile con le esigenze di controllo del sistema meccatronico, mantenendo al contempo un’elevata leggibilità e manutenibilità del codice.

# IMPLEMENTAZIONE

## RIPROGETTAZIONE HARDWARE DELLA STAMPANTE 3D: OBIETTIVI E SCELTE TECNICHE

### Verso una Nuova Architettura di Stampante 3D

Il progetto prevede l’impiego di un kit di stampa 3D come base hardware da riconfigurare integralmente, sia dal punto di vista meccanico che software, con l’obiettivo di realizzare una piattaforma sperimentale conforme ai principi dell’open hardware e dell’open software. La scelta è ricaduta sulla stampante Voron 0.2, un sistema open source ampiamente documentato e caratterizzato da un’elevata modularità, che offre notevoli margini di personalizzazione e adattabilità progettuale.

Rispetto alla configurazione originale della Voron 0.2, è stata introdotta una modifica sostanziale: l’eliminazione della mobilità del piano di stampa lungo l’asse Z. In questa implementazione, infatti, la scheda elettronica oggetto di analisi sarà fissata in modo permanente all’interno del macchinario Hercules, al quale il sistema verrà integrato. Hercules è un computer basato su architettura FPGA (*Field-Programmable Gate Array*), progettato specificamente per il controllo di memorie Managed NAND. Grazie alla sua struttura riconfigurabile, esso garantisce elevate prestazioni e una notevole flessibilità nella gestione di protocolli complessi e interfacce di memoria avanzate. Un FPGA è un dispositivo logico programmabile che consente di definire il comportamento dell’hardware anche dopo la produzione, rendendolo particolarmente adatto a scenari sperimentali e ad applicazioni ad alta specializzazione.

Il fissaggio statico della scheda garantirà la stabilità necessaria per l’esecuzione controllata di attacchi fisici tramite impulsi elettromagnetici, assicurando al contempo la ripetibilità delle condizioni sperimentali. Di conseguenza, l’architettura meccanica del sistema sarà modificata per consentire il movimento del posizionatore di sonda lungo tre assi cartesiani (X, Y, Z), superando la limitazione a due assi tipica della configurazione standard della stampante. Questo adattamento è essenziale per garantire precisione micrometrica e ripetibilità nel posizionamento della sonda rispetto al target, in un contesto in cui le dimensioni complessive del sistema saranno comparabili a quelle del macchinario Hercules.

Parallelamente, il software di controllo dovrà essere completamente riprogettato per supportare la gestione tridimensionale del posizionatore. Sarà necessario implementare un sistema di controllo coordinato, in grado di interpretare le coordinate spaziali e tradurle in comandi di movimento accurati e sincronizzati. L’architettura software dovrà inoltre essere concepita per garantire robustezza e tolleranza ai guasti, anche in presenza di condizioni operative anomale. In particolare, dovranno essere previsti meccanismi di gestione delle eccezioni per affrontare situazioni critiche quali un posizionamento iniziale non valido o la perdita improvvisa di alimentazione elettrica. In tali circostanze, il sistema dovrà essere in grado di rilevare l’anomalia, interrompere in sicurezza le operazioni in corso e, ove possibile, ripristinare lo stato operativo senza compromettere l’integrità del processo sperimentale o dei componenti coinvolti.

### Stampante di Riferimento: Voron 0.2

La Voron 0.2 è una stampante 3D di fascia alta, estremamente compatta e progettata per la tecnologia FDM (Fused Deposition Modeling), un processo di manifattura additiva in cui un filamento termoplastico viene riscaldato fino alla fusione ed estruso attraverso un ugello per essere depositato strato su strato, dando forma a un oggetto tridimensionale. Questa tecnologia, ampiamente diffusa nel settore desktop, si distingue per la sua economicità, affidabilità e adattabilità a una vasta gamma di materiali.

Il progetto nasce all’interno dell’ecosistema Voron Design, una piattaforma interamente basata su principi open-source e open-hardware. In questo contesto, open-source indica che il codice sorgente, la documentazione tecnica e i file di configurazione sono liberamente accessibili, modificabili e redistribuibili. Il concetto di open-hardware estende questa filosofia all’ambito meccanico ed elettronico, rendendo disponibili i file CAD e le distinte base BOM. I file CAD (Computer-Aided Design) sono modelli tridimensionali digitali che descrivono con precisione geometrica ogni componente della macchina, mentre la BOM (Bill of Materials) è un elenco strutturato e dettagliato di tutti i materiali, componenti e parti necessari per l’assemblaggio, comprensivo di codici, quantità e specifiche tecniche. Questi strumenti rendono il progetto completamente replicabile e personalizzabile da chiunque.

La Voron 0.2 adotta una cinematica CoreXY, che consente movimenti rapidi e precisi sugli assi X e Y grazie a un sistema a cinghia incrociata, mentre l’asse Z è affidato al movimento verticale del piatto di stampa. Il telaio, realizzato in profilati di alluminio estruso, garantisce rigidità strutturale e stabilità termica, elementi fondamentali per ottenere stampe di alta qualità. Il design modulare e altamente personalizzabile consente l’integrazione di componenti avanzati e l’adattamento a esigenze specifiche.

Dal punto di vista del controllo, la Voron 0.2 è compatibile con firmware open-source come Klipper e Marlin. Il firmware rappresenta il livello software che interfaccia l’hardware della stampante con i comandi di stampa, gestendo in tempo reale il movimento degli assi, la regolazione delle temperature, la lettura dei sensori e le funzioni di sicurezza. Klipper sfrutta l’elaborazione esterna di un microcomputer (tipicamente un Raspberry Pi) per ottenere prestazioni cinematiche superiori, mentre Marlin, più tradizionale, è noto per la sua ampia compatibilità e la maturità del suo ecosistema.

La Voron 0.2 non è soltanto una macchina per la fabbricazione personale: è l’espressione concreta di una cultura tecnica fondata sulla condivisione del sapere, sull’ingegneria partecipativa e su un modello di sviluppo distribuito, in cui l’innovazione nasce dal dialogo continuo tra progettazione, sperimentazione e comunità. In essa convergono precisione meccanica, efficienza elettronica e intelligenza collettiva, rendendola uno strumento emblematico della stampa 3D moderna e dell’evoluzione tecnologica collaborativa.

### Il Generatore di Impulsi EMFI**:** Faulty Cat

Faulty Cat v1.1 è un dispositivo open-source progettato per l’esecuzione di attacchi di tipo Electromagnetic Fault Injection (EMFI), una tecnica avanzata utilizzata per indurre malfunzionamenti controllati in circuiti elettronici, con l’obiettivo di analizzare la robustezza e la sicurezza dei sistemi digitali. Questo strumento consente di generare impulsi elettromagnetici ad alta energia e brevissima durata, capaci di alterare temporaneamente il comportamento di microcontrollori e chip integrati, simulando condizioni di guasto o interferenza.

Grazie alla sua architettura isolata e alla presenza di una schermatura plastica obbligatoria, il dispositivo garantisce un utilizzo sicuro anche in ambienti sperimentali non professionali, pur mantenendo prestazioni comparabili a strumenti di fascia superiore. È particolarmente adatto per attività di ricerca, reverse engineering, analisi della sicurezza hardware e formazione tecnica avanzata. La comunità open-source che lo supporta ne favorisce l’evoluzione continua, rendendolo uno strumento accessibile, documentato e altamente personalizzabile.

### Soluzione Realizzata: Il Prodotto Hardware Finale

Rispetto alla configurazione originaria della stampante Voron 0.2, che prevedeva la traslazione del piano di stampa lungo l’asse Z, è stata introdotta una modifica strutturale significativa: il movimento verticale sarà gestito manualmente, mentre il sistema manterrà il controllo attivo sugli assi X e Y. Tale scelta progettuale è stata determinata da due fattori principali. In primo luogo, l’analisi funzionale ha evidenziato che, nell’ambito delle operazioni previste, la regolazione della quota verticale della sonda non richiede variazioni dinamiche durante l’esecuzione delle prove. Una volta definita la posizione lungo l’asse Z in fase di calibrazione, essa può essere mantenuta costante senza compromettere la precisione del posizionamento o la ripetibilità delle condizioni sperimentali. In secondo luogo, il progetto è stato sviluppato all’interno di un vincolo temporale stringente, che ha imposto una razionalizzazione delle risorse progettuali e una selezione mirata delle funzionalità da implementare.

La semplificazione dell’asse Z ha consentito di concentrare lo sviluppo sull’ottimizzazione del controllo planare, garantendo al contempo affidabilità meccanica, riduzione della complessità del sistema e rispetto delle scadenze progettuali. Per adattare la piattaforma alla nuova configurazione, sono state progettate ex novo specifiche componenti meccaniche, ottimizzate per le finalità sperimentali del sistema. Tali componenti saranno rese disponibili in formato digitale, al fine di garantire la piena replicabilità della piattaforma e promuovere la sua diffusione secondo i principi dell’open hardware e dell’open software. Questa evoluzione progettuale, pur semplificando l’architettura meccanica, preserva l’obiettivo di realizzare una piattaforma sperimentale ad alta precisione, documentata, accessibile e tecnicamente solida, destinata allo studio controllato di fenomeni fisici complessi su dispositivi elettronici.

## RICONFIGURAZIONE SOFTWARE DELLA STAMPANTE 3D: OBIETTIVI E SCELTE TECNICHE

### Dal Concetto al Codice: Linee Guida per lo Sviluppo

Per procedere alla riconfigurazione funzionale della stampante 3D Voron 0.2, è necessario intervenire direttamente sull’unità originale, seguendo una sequenza operativa strutturata e tecnicamente definita. Il processo si articola in più fasi, a partire dalla conversione delle unità di misura: i parametri geometrici e dimensionali vengono inizialmente espressi in micrometri (μm), ma per garantire la compatibilità con il firmware Marlin , il sistema di controllo open-source comunemente utilizzato per la gestione delle stampanti 3D, tali valori vengono convertiti in millimetri (mm), unità standard per la generazione del codice G (G-code). Il G-code è un linguaggio di programmazione a basso livello utilizzato per impartire istruzioni dirette alla macchina, come movimenti, velocità, temperature e sequenze operative.

Successivamente, viene eseguita la funzione di “return home”, ovvero il comando di homing, che consiste nel posizionamento iniziale della macchina rispetto ai propri assi di riferimento. Questo passaggio è fondamentale per garantire che tutti i movimenti successivi siano eseguiti con precisione assoluta rispetto a un’origine nota e ripetibile. Una volta completato l’homing, viene generato un file G-code contenente le istruzioni necessarie alla riconfigurazione, che sarà trasmesso al firmware Marlin attraverso una connessione seriale stabilita con la scheda di controllo.

La comunicazione con la board avviene mediante l’invio sequenziale dei comandi G-code, riga per riga, assicurando che ogni istruzione venga interpretata ed eseguita correttamente dal microcontrollore. Questo approccio consente un controllo fine e verificabile di ogni fase del processo, riducendo al minimo il rischio di errori o comportamenti imprevisti. Al termine dell’esecuzione completa delle istruzioni, la macchina viene riportata automaticamente alla posizione di homing, ristabilendo così lo stato iniziale e pronto per l’operatività successiva.

Questa procedura garantisce una riconfigurazione coerente, ripetibile e conforme agli standard tecnici richiesti per l’ottimizzazione delle prestazioni meccaniche ed elettroniche della stampante Voron 0.2, assicurando al contempo l’integrità del sistema e la tracciabilità delle operazioni eseguite.

### Integrazione del Firmware Marlin

L’integrazione del firmware Marlin all’interno della piattaforma sperimentale sviluppata nel presente progetto offre vantaggi significativi in termini di gestione delle condizioni operative critiche e adattabilità funzionale. Grazie alla sua architettura modulare e altamente configurabile, Marlin consente l’implementazione di logiche di controllo personalizzate, utili per affrontare scenari anomali o condizioni di rischio, come il rilevamento di un posizionamento iniziale non valido, la perdita improvvisa di alimentazione o la presenza di ostacoli sul piano di lavoro.

Tali funzionalità possono essere realizzate sia mediante l’estensione del codice sorgente, sia attraverso l’attivazione di moduli già presenti nel firmware, come i sistemi di ripristino automatico, le routine di sicurezza termica e le verifiche di stato all’avvio. Inoltre, la compatibilità con un’ampia gamma di sensori e attuatori consente l’integrazione di meccanismi di feedback e diagnostica, migliorando l’affidabilità, la resilienza e la capacità di risposta del sistema in ambienti sperimentali ad alta precisione.

In questo contesto, Marlin si configura come una soluzione solida, flessibile e perfettamente allineata agli obiettivi del progetto, offrendo un’infrastruttura firmware già collaudata ma aperta a estensioni e personalizzazioni, in linea con i principi dell’open hardware e della ricerca applicata.

### Integrazione tra Moduli

Il codice sviluppato per il controllo del posizionatore tridimensionale sarà integrato all’interno di un sistema software più ampio, responsabile della gestione coordinata delle funzionalità sperimentali. In particolare, riceverà in ingresso le coordinate spaziali da un modulo superiore incaricato della definizione dei punti di misura.

Il modulo di controllo interpreterà le coordinate ricevute e attiverà i movimenti necessari attraverso la cinematica del treppiede. Questa architettura modulare consente una chiara separazione delle responsabilità, favorendo la scalabilità del sistema e la possibilità di aggiornare o sostituire singoli componenti senza compromettere la stabilità complessiva.

### Logiche di Controllo

Questa funzionalità consente al sistema di salvare periodicamente lo stato operativo (inclusi la posizione corrente, i parametri di movimento e lo stato della sonda) in una memoria non volatile, solitamente una EEPROM o una scheda SD. In caso di interruzione dell’alimentazione o di arresto anomalo, il firmware è in grado di ricostruire il contesto operativo e riprendere l’attività dal punto in cui era stata interrotta, evitando la necessità di riavviare l’intero processo o ripetere la calibrazione.

La funzionalità può essere attivata tramite la direttiva POWER\_LOSS\_RECOVERY, che abilita la scrittura periodica dei dati di stato. È inoltre possibile configurare la frequenza di salvataggio, il tipo di supporto di memoria utilizzato e le condizioni di sicurezza per la ripresa (ad esempio, verifica della temperatura, stato dei finecorsa, ecc.).

Ulteriori accorgimenti implementabili includono: la verifica automatica dell’integrità dei dati salvati, per evitare riprese da stati corrotti; la gestione personalizzata degli errori, con notifiche all’utente o log dettagliati su file; l’integrazione con sensori di tensione, per anticipare l’interruzione e salvare lo stato in modo proattivo.

Queste logiche di controllo contribuiscono a rendere il sistema non solo più robusto e affidabile, ma anche conforme alle esigenze di replicabilità e sicurezza richieste in ambito accademico e di ricerca.

### Soluzione Realizzata: Il Prodotto Software Finale

Il prodotto software finale consiste in un modulo di controllo sviluppato in linguaggio Python, progettato per interfacciarsi con la scheda elettronica del sistema meccatronico tramite connessione seriale. Il nucleo del software è rappresentato dalla classe Connection, che incapsula la logica di comunicazione e controllo del posizionatore. La classe include un costruttore parametrico, che accetta come argomento il baud rate (bit al secondo) e inizializza la connessione seriale con la scheda, memorizzata come attributo interno.

Tra i metodi principali figurano send\_command, che consente l’invio di stringhe G-code alla board, homing, che attiva i motori e avvia il posizionamento assoluto degli assi tramite il comando G28, e set\_position, che riceve le coordinate in formato micrometrico, le converte in millimetri e genera il comando G1 per il movimento lineare interpolato. Il sistema implementa inoltre una logica di interrogazione dello stato tramite il comando M114, che restituisce la posizione corrente della sonda.

Le risposte vengono lette in modo asincrono e accumulate in una lista fino alla ricezione del messaggio di conferma ok. Se la risposta indica una posizione x=0, y=0, z=5, il sistema riconosce che l’homing è stato eseguito correttamente e procede con il posizionamento; se invece la risposta è x=0, y=0, z=0, viene sollevata un’eccezione e viene eseguita la procedura di homing. Il comando M17 viene utilizzato per abilitare l’alimentazione dei motori passo passo prima dell’esecuzione di qualsiasi movimento. L’intero modulo è stato progettato per essere integrato in un sistema software superiore, che fornirà le coordinate target e gestirà la sequenza delle operazioni sperimentali.

L’architettura del codice garantisce modularità, riusabilità e compatibilità con ambienti di test automatizzati, mantenendo al contempo un’interfaccia semplice e robusta per l’interazione con il firmware Marlin.

# CONFIGURAZIONE E SETUP

## Istruzioni per l’Assemblaggio della Stampante 3D

La prima fase dell’assemblaggio prevede la preparazione e l’organizzazione di tutti i componenti meccanici come mostrato nella vista esplosa del modello 3D. È consigliabile disporre i profili in alluminio, le guide lineari, le staffe e i supporti su un piano di lavoro ampio e pulito, seguendo l’ordine logico suggerito dal modello.

Several metal parts of a machine

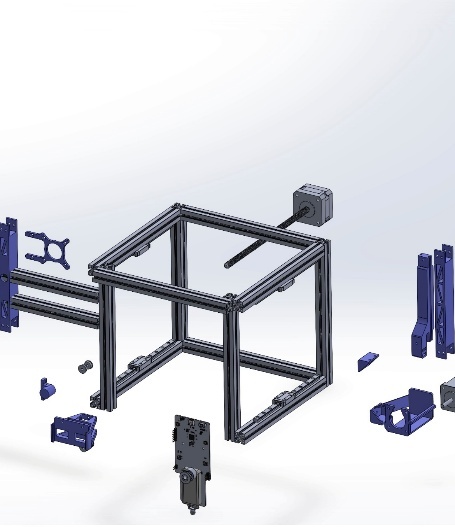
AI-generated content may be incorrect.

Si inizia con la costruzione del telaio di base, unendo i profili mediante giunti angolari e viti M3, facendo attenzione a mantenere la squadratura della struttura. È utile utilizzare una squadra da officina per verificare l’ortogonalità durante il serraggio. Una volta completata la base, si procede con il montaggio delle guide lineari sugli assi orizzontali con viti M2 e dadi M3. Le guide devono essere posizionate con precisione e fissate, assicurandosi che siano parallele tra loro per garantire un movimento fluido dei carrelli. I carrelli lineari vengono poi inseriti con cautela sulle guide, come mostrato in foto, evitando danneggiamenti delle superfici di scorrimento.

A close-up of several parts

AI-generated content may be incorrect.

A questo punto dell’assemblaggio, come mostrato in figura, si procede con la chiusura superiore della struttura e l’integrazione dei componenti meccanici rimanenti. I profili in alluminio che costituiscono la parte superiore del telaio vengono fissati ai montanti verticali, completando così la geometria cubica del sistema. Anche in questa fase è fondamentale verificare l’allineamento e la scorrevolezza dei carrelli, regolando eventualmente la posizione delle guide prima del serraggio definitivo.



In questa fase dell’assemblaggio, ci si concentra sull’integrazione del sistema di movimentazione orizzontale superiore e sul completamento della struttura meccanica. Successivamente, si installano le piastre di montaggio e i supporti intermedi, assicurandosi che tutti i componenti siano ben serrati e che non vi siano giochi meccanici. Le guide lineari orizzontali vengono fissate ai profili superiori del telaio, e i carrelli vengono montati in modo da scorrere lungo queste guide, supportando il movimento dell’organo terminale.

A blue and grey metal cube

AI-generated content may be incorrect.

In questa fase dell’assemblaggio, si procede con l’installazione di uno dei motori principali, visibile nella parte inferiore centrale della struttura. Il motore viene posizionato all’interno del telaio e fissato a una staffa di supporto mediante viti M3, assicurandosi che l’accoppiamento sia stabile e privo di giochi.

A blue and grey metal cube

AI-generated content may be incorrect.

In questa fase dell’assemblaggio viene montato il componente che funge da supporto per la vite del motore. Questo elemento ha un ruolo strutturale fondamentale: serve a mantenere la vite a ricircolo di sfere perfettamente allineata e solidale con la struttura, evitando flessioni o disallineamenti che potrebbero compromettere la precisione del sistema. Il supporto viene fissato ai profili in alluminio tramite viti e staffe, e accoglie l’estremità libera della vite, opposta al motore.

A blue and grey metal frame

AI-generated content may be incorrect.

In questa fase dell’assemblaggio viene installato il supporto destinato ad accogliere la scheda elettronica denominata "Faulty Cat". Il componente è progettato per garantire un fissaggio stabile e sicuro della scheda all’interno del telaio.

A blue and silver metal frame

AI-generated content may be incorrect.

In questa fase dell’assemblaggio, si procede con l’installazione di ulteriori componenti funzionali all’interno della struttura cubica. Questi elementi, visibili sia all’interno del telaio sia tra le parti ancora da montare, includono supporti e alloggiamenti destinati a ospitare sottosistemi meccanici o elettronici. Il loro corretto posizionamento è fondamentale per garantire la funzionalità complessiva del sistema.

A blue and silver metal cube

AI-generated content may be incorrect.

In questa fase dell’assemblaggio viene installato il supporto laterale del motore, un componente fondamentale per garantire la stabilità e l’allineamento del sistema di trasmissione. Il supporto viene fissato lateralmente alla struttura principale, in corrispondenza della posizione prevista per il motore.

A blue and grey metal frame

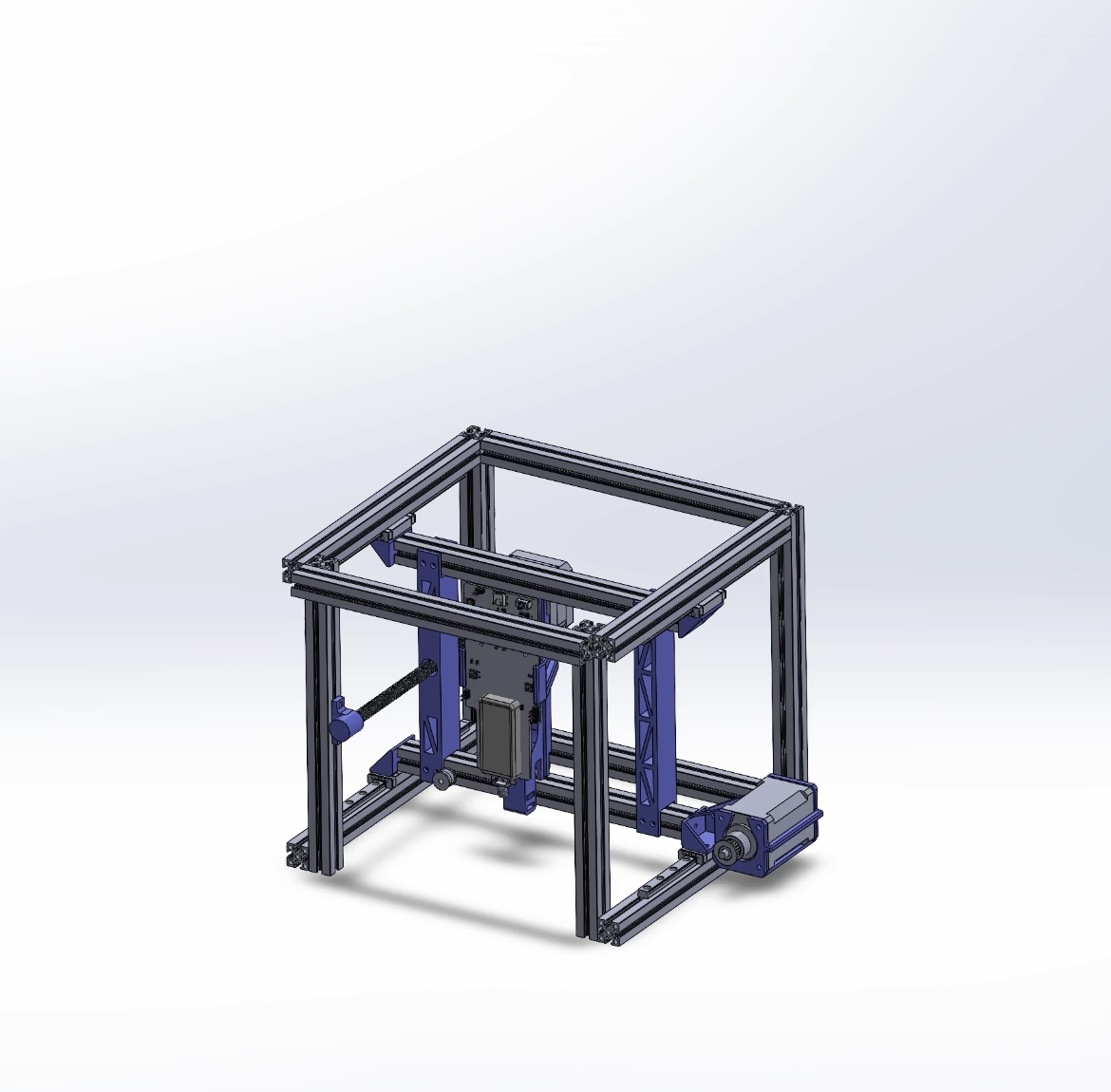
AI-generated content may be incorrect.

In questa fase dell’assemblaggio, il motore è stato inserito all’interno del supporto laterale precedentemente montato. Il motore è ora correttamente alloggiato nella sua sede, fissato tramite viti al supporto in modo da garantire stabilità e coassialità con la vite di traslazione o l’elemento meccanico che andrà ad azionare.

A blue and black machine

AI-generated content may be incorrect.

In questa fase conclusiva dell’assemblaggio hardware, la scheda Faulty Cat viene fissata all’interno del supporto precedentemente installato. La scheda viene posizionata con cura, allineando i fori di montaggio con quelli del supporto, e viene assicurata tramite viti e distanziatori, che garantiscono un corretto isolamento meccanico ed elettrico.



Per ulteriori dettagli tecnici, file di progetto, istruzioni aggiornate e documentazione completa, si rimanda al repository ufficiale disponibile su GitHub.

## Installazione del Firmware

Per la personalizzazione e la compilazione del firmware Marlin, è stato predisposto un ambiente di sviluppo basato su Visual Studio Code (VS Code) con l’estensione PlatformIO IDE, soluzione ampiamente adottata in ambito embedded per la sua flessibilità, modularità e compatibilità con microcontrollori di diversa architettura.

Il codice sorgente di Marlin viene organizzato secondo la struttura prevista da PlatformIO. Il file centrale di configurazione è platformio.ini, che definisce la piattaforma hardware target, le opzioni di compilazione e le librerie necessarie. I file principali da personalizzare sono:

* Marlin/Configuration.h: contiene i parametri generali della stampante, come dimensioni dell’area di lavoro, tipo di cinematica, sensori, driver motori, ecc.
* Marlin/Configuration\_adv.h: include opzioni avanzate, come la gestione della ripresa dopo interruzioni, il controllo termico, le strategie di livellamento del piano e le ottimizzazioni di movimento.

Una volta configurato l’ambiente, la procedura di compilazione si articola come segue:

1. Clonazione del repository ufficiale di Marlin tramite Git, oppure download manuale del pacchetto sorgente.
2. Importazione del progetto in Visual Studio Code tramite l’interfaccia di PlatformIO.
3. Selezione della piattaforma hardware nel file platformio.ini, in base alla scheda madre della stampante 3D.
4. Modifica dei file di configurazione per adattare il firmware alle specifiche esigenze meccaniche ed elettroniche del sistema.
5. Compilazione del firmware tramite il comando Build di PlatformIO, che genera il file binario (.hex o .bin) pronto per il caricamento.
6. Upload del firmware sulla scheda di controllo, tramite connessione USB diretta o tramite scheda SD, a seconda della piattaforma utilizzata.

## Parametri di configurazione

Nel contesto dello sviluppo del sistema meccatronico tridimensionale, una fase cruciale ha riguardato la personalizzazione del firmware Marlin, al fine di adattare il comportamento del posizionatore alle specifiche esigenze sperimentali del progetto. In particolare, si è intervenuti sui file di configurazione per definire due insiemi di parametri fondamentali: l’area di lavoro e la velocità di spostamento.

L’area di lavoro rappresenta il volume tridimensionale entro cui il posizionatore può muoversi. Nel nostro caso, tale area è estremamente contenuta, in quanto il sistema è destinato ad operare su superfici di dimensioni ridotte, tipicamente corrispondenti a porzioni millimetriche o micrometriche di circuiti stampati. Di conseguenza, i limiti spaziali sono stati configurati per riflettere una zona operativa compatibile con le dimensioni fisiche della scheda elettronica da analizzare, garantendo al contempo la precisione e la sicurezza del movimento.

Un altro parametro critico è la velocità di spostamento del posizionatore. Sebbene Marlin consenta l’impostazione di velocità elevate, nel nostro caso si è ritenuto opportuno adottare una velocità moderata. Questa scelta è motivata da diversi fattori: la presenza potenziale di ostacoli fisici sul piano di lavoro (quali componenti elettronici sporgenti o cablaggi), la necessità di evitare vibrazioni o urti che potrebbero compromettere l’integrità dei dispositivi in prova, e l’esigenza di garantire un’elevata accuratezza nel posizionamento della sonda. Inoltre, una velocità contenuta facilita l’osservazione e l’analisi dei fenomeni indotti, migliorando la qualità complessiva dell’esperimento.

Tutti i parametri sono stati configurati in modo da mantenere la massima flessibilità: l’utente potrà modificarli agevolmente tramite l’interfaccia di controllo, adattando il comportamento del sistema alle specifiche condizioni operative.

# RIFERIMENTI

|  |  |
| --- | --- |
| [1] | Nemgrea, "VORON ZERO ASSEMBLY MANUAL," May 2024. [Online]. Available: https://raw.githubusercontent.com/VoronDesign/Voron-0/Voron0.2r1/Manuals/VORON\_V0.2r1\_Assembly\_Manual.pdf. |