UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base
Corso di Laurea in Ingegneria Informatica

Elaborato finale in
Tecnologie Informatiche per l'Automazione Industriale

Simulazione e controllo di un impianto industriale per la produzione di materiali in ambiente OpenPLC e Factory I/O

Relatore
Chiar.mo Prof.
Adriano Mele

Candidato
Antonio Borzillo
matr. N46/3179

Anno Accademico 2019/2020

Indice

Introduzione

Il lavoro svolto per chiudere questo percorso triennale, racchiuso in questo documento di tesi, ha come obiettivo la descrizione delle potenzialità offerte dai controllori logico programmabili attraverso il compimento di un'attività di controllo di un impianto industriale virtuale dedito alla produzione e all'assemblaggio di materiali.

In particolare l'elaborato guiderà minuziosamente il lettore sia nella realizzazione dell'impianto, attraverso l'ambiente di sviluppo Factory I/O, che nell'implementazione dell'algoritmo di controllo per il SoftPLC che verrà realizzato in ambiente OpenPLC e con il quale si realizzerà l'effettiva attività di controllo per l'impianto progettato.

Nel primo capitolo quindi, dopo aver introdotto la figura del controllore logico programmabile, si presenteranno brevemente i due ambienti software che hanno reso possibile la realizzazione di una tipica attività di PLC Training: Factory I/O e OpenPLC.

Successivamente, nel Capitolo 2, si ripercorrerà nel dettaglio la fase di progettazione dell'impianto virtuale in ambiente Factory I/O nonché la fase di realizzazione del SoftPLC in ambiente OpenPLC, discutendo infine circa come sia stato possibile configurare la comunicazione fra i due ambienti sofware per concretizzare la vera e propria attività di controllo.

Capitolo 1

Questo primo capitolo è stato dedicato ad una breve introduzione delle tre unità fondamentali sulle quali è basato questo elaborato di tesi: il controllore logico programmabile e gli ambienti OpenPLC e Factory I/O.

In una prima fase infatti, dopo averlo inquadrato storicamente, vengono presentati gli aspetti fondamentali di un PLC: la sua struttura, sia hardware che software, le sue modalità di funzionamento e alcuni cenni sulla sua programmazione.

Successivamente invece sono presentati i due ambienti software che hanno permesso di realizzare una tipica attività di PLC Training mediante il controllo di un impianto virtuale.

1.1 Il controllore logico programmabile

Il PLC è un dispositivo elettronico digitale specializzato per il controllo logico/sequenziale di processi fisici.

Tale dispositivo fu introdotto alla fine degli anni 60 quando il progresso dell'elettronica digitale portò le industrie a richiedere lo sviluppo di controllori logico/sequenziali che riuscissero ad ovviare alle limitazioni dei primi sistemi di automazione industriale. Questi, prima dell'avvento del PLC, erano infatti realizzati perlopiù da componenti elettromeccanici troppo ingombranti, poco flessibili ma soprattutto difficilmente riprogrammabili, come relè e temporizzatori.

Tali elementi venivano interconnessi fra loro formando complessi circuiti elettrici che implementavano precise logiche di controllo, pertanto un'eventuale modifica a tale logica richiedeva un intervento sull'intero circuito elettrico, che risultava spesso un processo lungo e tedioso.

Pertanto nel giro di pochi anni, grazie alle sue caratteristiche di flessibilità, robustezza, e semplicità di programmazione il PLC andò a rimpiazzare le macchine logiche a relè, imponendosi come standard per il controllo logico sequenziale.

1.1.1 L’architettura di un PLC

Trattandosi di un elaboratore a tutti gli effetti, il controllore a logica programmabile è costituito da una parte hardware formata diversi componenti fisici e da una parte software costituita dalle istruzioni che formano il programma che deve essere eseguito e che regolamenta il comportamento del dispositivo stesso.

I moderni PLC possono eseguire funzioni complesse alla stregua di un calcolatore convenzionale con cui condividono molti aspetti hardware, come l'avere una CPU, una memoria e dei bus.
Nonostante ciò possiamo distinguere due aspetti che differenziano un controllore logico digitale da un comune PC.
Il primo di questi è che l'hardware di un PLC deve essere tanto robusto da sopravvivere allo stress di un tipico ambiente industriale.
Il secondo è che il suo software deve essere real-time.

Per quanto concerne la parte hardware, i PLC hanno tipicamente un’architettura modulare a bus, suddivisa in 4 parti chiave: rack, alimentatore, CPU, input/output.

Il rack (o armadio) è la parte più esterna del dispositivo, esso è caratterizzato da un backplane nella parte posteriore che consente la comunicazione tra ogni modulo del PLC.

Il modulo alimentatore invece, ha il compito di garantire una tensione stabile, priva di interferenze e fluttuazioni e deve inoltre segnalare la mancanza di alimentazione per far sì che il PLC possa avviare le procedure di spegnimento.

I moduli di input sono invece utilizzati per la lettura di segnali dai sensori istallati sul campo. Ci sono differenti tipi di moduli input che dipendono dai sensori con cui devono comunicare, ma possiamo comunque limitarci a differenziarli in due categorie: moduli analogici e digitali. I moduli input digitali lavorano con segnali discreti generati dai dispositivi, i classici on/off per intenderci, mentre i moduli analogici convertono quantità fisiche in valori digitali da far processare alla CPU del PLC.
Viceversa i moduli di output controllano i dispositivi installati sul campo e come i precedenti possono essere analogici o digitali.

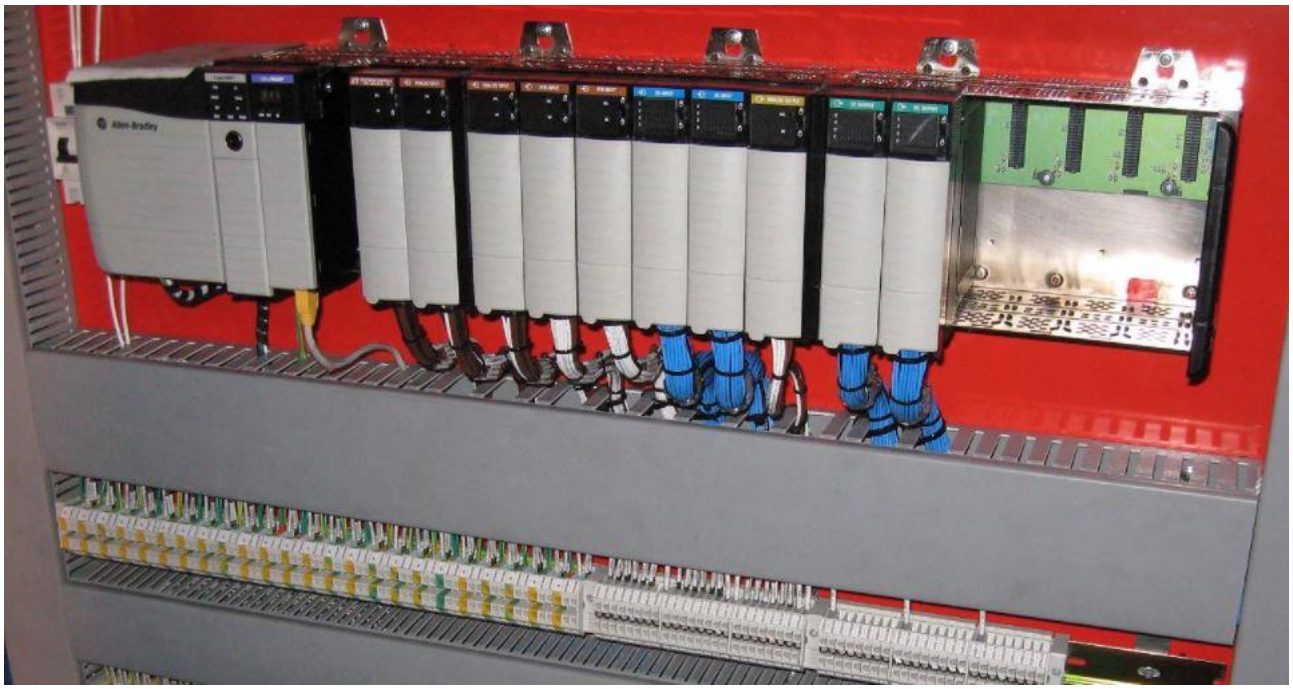
Il modulo più importante di un PLC è probabilmente il modulo processore.
Esso è responsabile del processamento delle informazioni ricevute dai moduli di input e, seguendo la particolare logica di programmazione del dispositivo, invia impulsi verso i moduli d'uscita.

La memoria di un PLC fornisce uno spazio di archiviazione permanente al sistema operativo per i dati utilizzati dalla CPU; nello specifico la memoria ROM del sistema memorizza i dati in modo permanente per il sistema operativo, la memoria RAM memorizza le informazioni di stato per i dispositivi di input e output, insieme ai valori per timer, contatori e dispositivi interni.

Nonostante la modalità d’esecuzione dei programmi possa essere periodica, ciclica o ad eventi, la maggior parte delle volte la logica memorizzata nei registri interni del PLC viene eseguita in maniera ciclica.
In tal caso la velocità di elaborazione del modulo processore viene solitamente descritta dal tempo di scansione, ovvero l'arco temporale che intercorre fra due attivazioni successive della stessa porzione di programma applicativo.

L’esecuzione ciclica avviene tipicamente con una modalità che è detta: ciclo a copia massiva degli ingressi e delle uscite.
All’inizio di ogni ciclo, il PLC provvede a fare una scansione di tutti gli ingressi e le uscite e copiare i valori letti in un’apposita zona di memoria.
Il PLC in tal modo crea un’immagine interna dell’impianto che viene aggiornata all’inizio di ogni ciclo d’esecuzione.
Una volta creata tale immagine il SO legge i dati da quest’ ultima (che in questo modo risulta essere statica), calcola i valori delle uscite di controllo e li salva in un’altra area di memoria.
Al termine del ciclo, provvede a prelevare tali valori per inviarli agli attuatori.

Uno dei vantaggi della copia massiva è che essa garantisce che i valori degli ingressi e delle uscite restino invariati durante l’esecuzione dei programmi, allo stesso tempo però il sistema risulta essere ‘‘cieco’’ tra una scansione e l’altra del ciclo.
Questo vuol dire che se si verifica un ingresso inaspettato, indicatore di un guasto ad esempio, il SO lo vede soltanto al ciclo successivo in cui questo guasto avviene, e di conseguenza potrà prendere provvedimenti solo al suo termine.



[fig1]: *L'immagine raffigura un rack di un PLC Allen-Bradley, un esempio comune di una configurazione PLC che include una CPU, ingressi analogici, uscite analogiche e uscite CC.*

Per quanto concerne la parte software una particolare esigenza dei dispositivi a logica programmabile, valida per avere un controllo effettivo sui processi, è che i PLC devono essere sistemi real-time.

Pertanto tali controllori sono progettati non tanto per diminuire i tempi di risposta ma piuttosto devono garantire delle risposte nei limiti specificati da un periodo temporale finito, la deadline.

La deadline di un PLC è il suo tempo di scansione, quindi tutte le risposte devono essere fornite prima o al momento la scansione raggiunge la fine del ciclo.

1.1.2 Il SoftPLC

E’ anche possibile emulare il comportamento di un PLC classico su architetture differenti come PC o microcontrollori.

In tal caso parliamo di Soft-PLC, nient’ altro che un package software che genera un ambiente che garantisce le funzionalità di un PLC standard unendo i vantaggi di un design ad architettura aperta alla potenza dei computer.

Lavorare con un SoftPLC, piuttosto che con un PLC standard, presenta aspetti positivi e negativi. Uno dei pro è sicuramente il fatto che questi siano facilmente interfacciabili con software non direttamente legati al controllo dell’impianto, ma uno svantaggio tipico è la ridotta quantità di moduli I/O di cui sono dotati. In questi casi nasce l'esigenza di utilizzare moduli di I/O esterni collegandoli al PC che esegue il package SoftPLC.

1.1.3 Cenni alla programmazione PLC

Come chiarito precedentemente un PLC è un dispositivo che monitora segnali di input/output e prende delle decisioni logic-based per il controllo di processi fisici. Tale dispositivo quindi dovrà essere opportunamente programmato per la realizzazione dei suoi scopi.

A tal proposito la normativa IEC 61131-3 del 1993 definisce 5 linguaggi di programmazione attraverso i quali programmare un PLC. Tale standard inoltre, ha permesso di ovviare a diversi "limiti storici" della programmazione dei PLC, come ad esempio la difficoltà nel produrre software riutilizzabile, riducendo gli errori e le incomprensioni e definendo tecniche di programmazione utilizzabili in più ambienti.

Si noti però che lo standard è da considerarsi un modello di riferimento non vincolante: non tutti i PLC infatti lo rispettano, soprattutto PLC più datati .

Tra i cinque linguaggi di programmazione normati dallo standard ritroviamo:

- due linguaggi testuali: Instruction List (IL) e Structured Text(ST)
- tre linguaggi grafici: Function Block Diagram (FBD), Ladder (LD) e il Sequential Functional Chart (SFC).

Per l'implementazione della logica di controllo alla base del SoftPLC si è utilizzato principalmente l'SFC e alcune righe di Ladder.
Riferimenti più dettagliati a tali linguaggi saranno presenti nei capitoli successivi.

1.2 Il progetto OpenPLC

OpenPLC è un controllore logico programmabile open source basato su un software user-friendly.

Grazie all'utilizzo di diversi software esso permette di progettare e testare PLC sia in software che in hardware.

Nel 2014 il suo sviluppatore Thiago Alves condivise sul forum di Arduino un post nel quale spiegò le ragioni che lo portarono allo sviluppo di un nuovo progetto, prima fra tutte la volontà dello sviluppatore di fornire una soluzione a basso costo per l'automazione industriale, la domotica e la ricerca.

A causa degli elevati costi, infatti, al giorno d'oggi ci sono ancora molti posti nel mondo in cui l'automazione risulta essere inaccessibile.

Come lo stesso Alves spiega, OpenPLC fu creato per surclassare gli evidenti impedimenti da cui i paesi risultavano afflitti, due fra tutti le conoscenze teoriche ed i costi.

Nella maggior parte dei casi infatti le aziende produttrici non forniscono informazioni dettagliate sul funzionamento di questi dispositivi che spesso risultano anche essere closed source.

Il progetto quindi, proponendo un ambiente interamente open source, incentiva il diffondersi della tecnologia e conoscenza limitandone i costi.

OpenPLC Project è composto da 3 parti: Runtime, Editor e HMI Builder.

[fig2] : *il progetto OpenPLC*.

Il Runtime deve essere installato sul dispositivo di interesse ed è la parte software responsabile dell'esecuzione dei programmi PLC.

Esso supporta diversi sistemi embedded e può anche essere installato su Windows e Linux come soft-PLC, in questo caso è possibile espandere le porte Input/Output utilizzando, ad esempio, le schede Arduino come slave devices.

L'Editor è il software che permette la scrittura dei programmi PLC per il Runtime. L'intero progetto venne creato in conformità con lo standard IEC 61131-3, per tale motivo è possibile scrivere i programmi PLC utilizzando tutti i linguaggi in esso normati.

Infine ScadaBR è l' HMI Builder associato attraverso il quale realizzare dei sistemi SCADA (Supervisory Control And Data Acquisition) impiegati per il monitoraggio e la supervisione dei sistemi fisici.

L’utilizzo diretto dell’Editor e del Runtime offerti da OpenPLC sarà oggetto della fase successiva del documento.

1.3 L’ambiente Factory I/O

FactoryIO è un ambiente software che permette la simulazione 3D di stabilimenti industriali per l'apprendimento delle tecnologie di automazione.

Esso è comunemente utilizzato per il PLC-Training, in tal caso quindi attraverso il software stesso si realizzerà una simulazione dell'ambiente operativo del PLC, dopodichè fissando una connessione fra il software e il PLC stesso si testerà l'efficienza del programma logico caricato sul controllore. Tale scenario però, non è l'unico possibile, Factory I/O infatti permette anche l'interfacciamento con microcontrollori (vedi Arduino), SoftPLC, Modbus e altre differenti tecnologie.

Differentemente dal progetto OpenPLC stiamo parlando di un software a pagamento disponibile in varie versioni ognuna delle quali include un pacchetto di driver per una tecnologia specifica (ad es. Allen-Bradley Edition,Siemens Edition, ...).

In tutte le versioni è possibile costruire delle fabbriche virtuali utilizzando alcune delle parti più comuni di ambienti industriali, fra cui: sensori, nastri trasportatori, bracci meccanici, pallettizzatori ecc.

[fig3] : *una vista di una parte dei pezzi industriali offerti da Factory I/O.*

La maggior parte di questi pezzi include I/O digitali e analogici che devono essere opportunamente governati per realizzare un particolare algoritmo di controllo.

Oltre alle parti atomiche, Factory I/O offre una serie di scene di default ispirate a tipiche applicazioni industriali, utili come punto di partenza per la costruzione del proprio impianto industriale.

All'atto dell'utilizzo di ogni pezzo il software gli assegna di default uno o più tags.

Un tag è composto da un nome e un valore e può essere di tre diversi tipi di dato a seconda del tipo e della configurazione di sensori e attuatori caratterizzanti la parte scelta:

- **bool** per valori on / off, esempio per mettere in moto o stoppare un nastro trasportatore.
- **float** per valori analogici, rappresentanti ad esempio il valore analogico associato ad una cisterna, utile per regolarne il riempimento.
- **int** per dati specifici.

Queste istanze vengono utilizzate per collegare i valori degli attuatori e dei sensori del pezzo industriale ad un controller, per l'attività di PLC-Training.

Tuttavia però FactoryIO da anche la possibilità di utilizzare i tags per un controllo manuale degli attuatori, scenario utilissimo in fase di testing.

Prima di controllare una scena con un controller esterno, infatti, è raccomandabile testarlo manualmente assicurandosi che la scena progettata funzioni come previsto.
In tal caso sarà compito dell'utente simulare valori provenienti da un ipotetico controllore forzando i tags degli attuatori.

Le potenzialità dell'ambiente di sviluppo saranno presentate nel prossimo capitolo in cui si avrà modo di creare un tipico ambiente industriale mediante la composizione di più pezzi offerti dal software.

Capitolo 2

In questo secondo capitolo si ripercorreranno nel dettaglio i vari stadi attraversati per arrivare al compimento dell'attività di simulazione e controllo dell'impianto industriale.

Inizialmente sarà quindi descritta in modo dettagliato la fase di progettazione che ha permesso la realizzazione dell'ambiente virtuale in Factory I/O.

Successivamente verrà mostrato e opportunamente motivato l'intero codice sorgente alla base dell'eseguibile con il quale si è implementata la specifica logica di controllo per l'impianto succitato.

Infine verrà discusso circa come sia stato possibile instaurare una comunicazione fra il SoftPLC, realizzato tramite il caricamento dell'eseguibile in OpenPLC Runtime, e l'ambiente Factory I/O, concretizzando la vera e propria attività di controllo.

2.1 Progettazione dell'impianto virtuale in ambiente Factory I/O

La scena che ho deciso di realizzare in Factory I/O consiste in un tipico impianto di produzione industriale.

Nel dettaglio lo stabilimento assolve alle sue finalità articolando l'intero processo fisico in 4 fasi fondamentali.

In un primo momento si tratta la lavorazione di pezzi grezzi i quali, opportunamente elaborati, andranno a generare due materiali di produzione componibili. I materiali così generati attraverseranno poi una fase secondaria di sorting nella quale vengono smistati in maniera efficace verso una zona di assemblaggio.

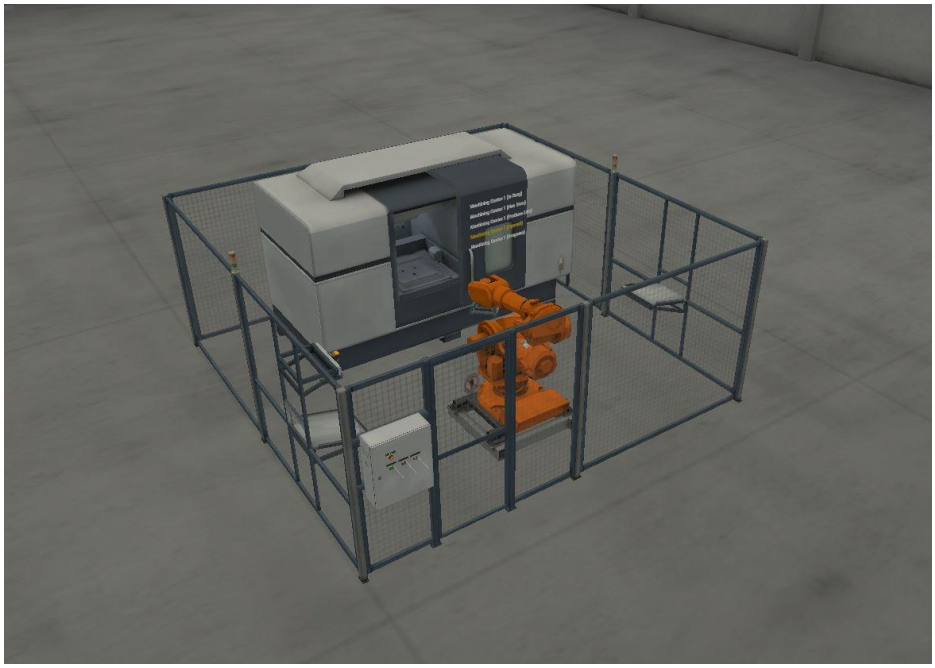
La terza fase infatti sarà relativa ad una particolare area dell'impianto in cui è presente un sistema dedito all'accorpamento dei due diversi pezzi ricevuti, arrivando alla realizzazione del prodotto finale.

L'impianto sarà infine dotato di un meccanismo di conteggio dei materiali in uscita attraverso il quale monitorare e valutare l'attività di produzione complessiva.

Tali 4 fasi sono state sfruttate come linee guida per una progettazione di tipo bottom-up dell'impianto industriale in ambiente Factory I/O.

2.1.1 Fase di lavorazione

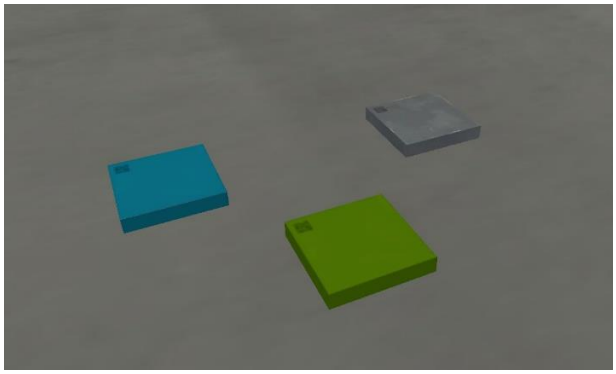
Per adempiere alle finalità di questa prima fase si è sfruttato il "Machining Center", una delle le tante stazioni offerte dal software di simulazione.



[fig4]: *Il Machining Center, la il centro di lavorazione di materie grezze offerto da FactoryIO.*

Il Machining Center è infatti un centro di produzione utilizzato per la realizzazione di coperchi e basi a partire da materie prime.

I materiali grezzi accettati dal centro di lavorazione possono essere di tre tipi: materiale blu, verde o di metallo. Per il progetto presentato in tale elaborato verranno utilizzate materie prime blu.



[fig5]: *una vista completa dei tre materiali grezzi disponibili per l'attività di lavorazione*

La scelta del materiale è comunque irrisoria ai fini della lavorazione, in tutti i casi infatti il braccio robotico attende per prima cosa che la materia prima venga collocata in ingresso al centro e, quando viene rilevato un nuovo materiale, esso lo carica nella macchina CNC che inizierà la produzione di un articolo.

Ogni tipo di articolo richiede un intervallo temporale di produzione diverso, i coperchi occupano la CNC per sei secondi, le basi per tre. Una volta completata l'operazione, il braccio meccanico posiziona l'oggetto in uscita e si mette in attesa di ricevere un altro pezzo da lavorare.

Come spiegato già nell'introduzione al software del capitolo precedente, all'atto dell'utilizzo di ogni pezzo come parte integrante della propria scena, Factory I/O gli associa uno o più tags.

Per controllare il Machining Center, si possono utilizzare i seguenti:

Tag	Controller I/O	Type	Description
Machining Center # (Produce Lids)	Output	Bool	Set True to produce lids; False to produce bases.
Machining Center # (Start)	Output	Bool	Start.
Machining Center # (Stop)	Output	Bool	Stop.
Machining Center # (Reset)	Output	Bool	Reset.
Machining Center # (Is Busy)	Input	Bool	Indicates whether an item is being processed.
Machining Center # (Has Error)	Input	Bool	Indicates whether an invalid item was detected on the entry bay.
Machining Center # (Opened)	Input	Bool	True when the door is open.
Machining Center # (Progress)	Input	Float	Machining progress (0-100).

[fig6]: *tags di input ed output associati ai sensori e attuatori per il controllo dell'attività del Machining Center.*

La configurazione standard del centro di lavorazione si articola complessivamente in 8 tags: 4 associati a sensori, 4 ad attuatori.

Attraverso le istanze booleane associate agli attuatori è possibile in qualsiasi momento avviare, riavviare o arrestare l'attività di produzione. Inoltre è possibile scegliere il tipo di lavorazione della CNC fra produzione di coperchi o basi.

I tags associati ai sensori invece sono utili per monitorare lo stato interno del Machining Center.

Compreso il modus operandi di tale stazione, per la fase di lavorazione si è deciso di utilizzare ben due centri di lavorazione in parallelo, per una produzione più rapida.

La prima stazione produce coperchi, la seconda produce basi.

Per instradare le materie prime verso il processo di lavorazione ho posizionato due nastri trasportatori separati in ingresso a tali centri produttivi.

Stesso discorso vale per l'exit bay di ciascuna delle stazioni, dove vengono disposti una serie di nastri trasportatori opportunamente posizionati in modo tale da far convergere ambo i materiali lavorati verso un nastro d'uscita comune.

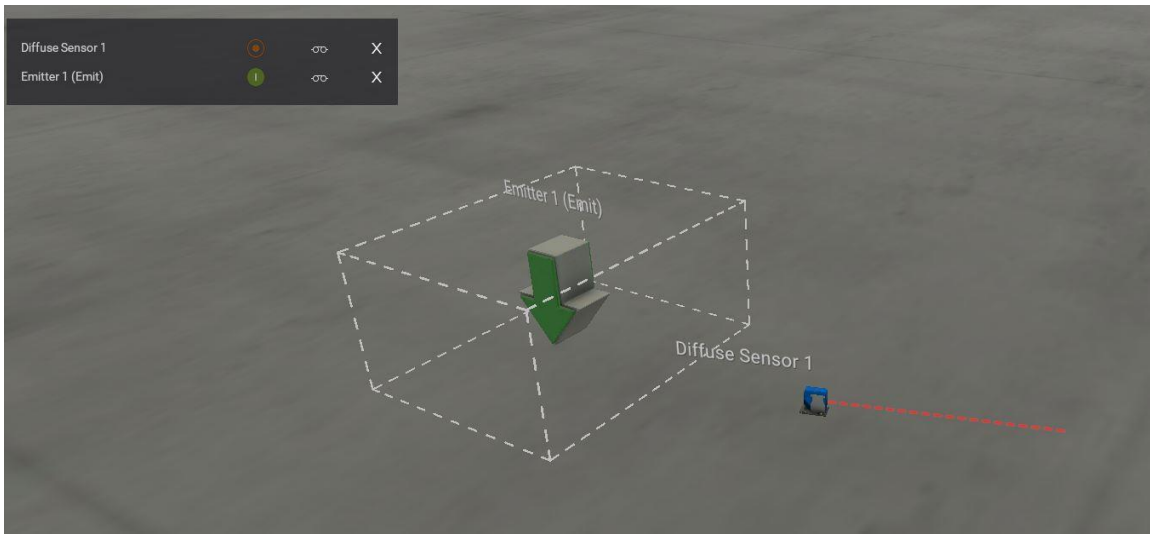
A tal proposito Factory I/O mette a disposizione una vasta scelta di nastri, in varie misure, che si adattano a particolari circostanze.

Per la realizzazione del mio impianto ho utilizzato dei trasportatori classici in 3 misure (2m, 4m e 6m).

Ogni nastro ha un singolo tag booleano per valori on/off del motore.

[fig7]: *tipologia di nastri trasportatori utilizzati nell'impianto , di metri 6,4 e 2 rispettivamente.*

La parte di impianto dedita alla fase di lavorazione è stata ultimata in primis apponendo sui due nastri di ingresso degli emitters, configurandoli come generatori di materie prime blu, e successivamente posizionando nei pressi delle uscite dei due Machining Center due sensori di prossimità il cui compito è quello di segnalare l'effettiva presenza di un prodotto in uscita dal rispettivo centro.



[fig8]: *le parti Emitter e Diffuse sensor di FactoryIO.*

Da come si evince dalla **[fig8],** sia l'emitter che il sensore di prossimità son caratterizzati da un singolo tag.

Factory IO assegna infatti all'emettitore, il quale non è altro che un generatore di items, una variabile booleana che ne regola l'attività di emissione. Nel caso del sensore invece, viene istanziato un tag di input che diviene vero non appena il sensore rileva un qualsiasi oggetto solido.

Il Risultato della prima fase di progettazione dell'impianto è il seguente:

[fig9]: *due Machining Centers e una serie di sensori e nastri trasportatori compongono la zona di un impianto industriale dedita alla lavorazione di materie prime .*

2.1.2 Fase di Sorting

Come detto in precedenza, l'impianto è stato pensato in maniera tale che i prodotti risultanti dall'attività di lavorazione vengano fatti convergere su di un nastro comune. Tale nastro è l'entry point per la seconda fase dell'impianto: la fase di sorting.

In questa fase le basi e i coperchi dovranno essere smistati su due nastri d'uscita differenti.

Per realizzare a tale attività sono risultati utili di due parti offerte da FactoryIO: un pusher ed un sensore visivo.

Quest ultimo è stato configurato in maniera tale da rilevare la presenza di una base, difatti è dotato di un singolo tag di input che diviene true all'atto del riconoscimento.

Tramite il sensore visivo sarà fatto variare lo stato del pusher, che nella configurazione scelta è dotato di 3 tags, di cui due associati ai sensori che indicano i limiti anteriore e posteriore (semplici finecorsa) ed un altro associato all'attuatore che attiva la spinta in avanti del pusher stesso.

[fig10]: *Un pusher ed un sensore visivo , i due elementi chiave per la realizzazione dell'attività di sorting.*

Con l'ausilio di due nastri trasportatori e due allineatori è stata conclusa la zona di impianto dedita al sorting, che appare come segue **[fig11]**.

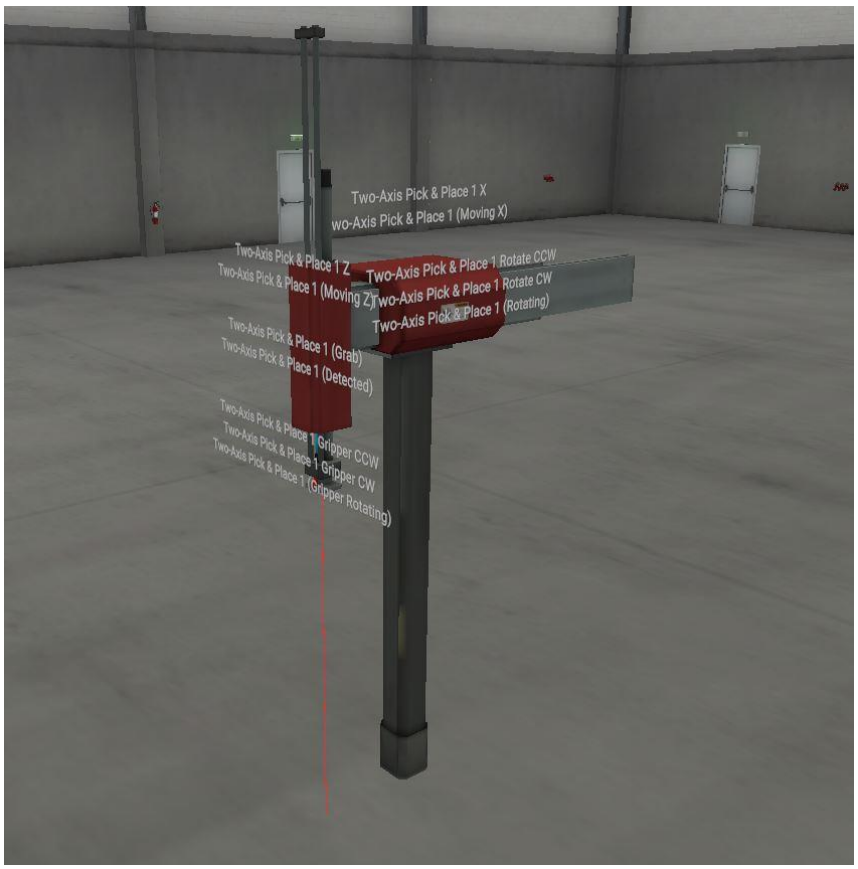
[fig11]: *una particolare disposizione di un sensore visivo ed un pusher per la realizzazione di un'attività di smistamento merci.*

2.1.3 Fase di assemblaggio

Come precedentemente descritto, l'attività di sorting si occupa di smistare basi e coperchi su due nastri separati. Tali due nastri instradano i rispettivi materiali verso la zona di impianto dedita all'attività di assemblaggio.

Quest' ultima è realizzata apponendo i coperchi sulle basi in modo tale da creare il prodotto finale che viene poi posto in uscita.

Per comporre il prodotto finale, a partire dai due materiali iniziali, si è utilizzata un'altra stazione offerta da Factory I/O: il Pick and Place a due assi.



[fig12]: *un Pick and Place a due assi in ambiente Factory I/O con i rispettivi tags che ne regolamentano il comportamento.*

Per la realizzazione del particolare tipo di impianto a cui si fa riferimento questa parte è stata utilizzata per assemblare coperchi su basi, ma è possibile anche utilizzarla per prelevare e posizionare oggetti da un luogo all'altro.

Il Pick and Place ha quindi la possibilità di muoversi, a partire dalla sua posizione di riposo, in avanti lungo il suo asse frontale, di scendere lungo l'asse verticale e di attivare una ventosa per afferrare un oggetto.

Il suo comportamento è dettato dai seguenti tags:

Tag	Controller I/O	Type	Description
Two-Axis Pick & Place # Z	Output	Bool	Move along Z-axis.
Two-Axis Pick & Place # X	Output	Bool	Move along X-axis.
Two-Axis Pick & Place # Rotate CW	Output	Bool	Rotate clockwise.
Two-Axis Pick & Place # Rotate CCW	Output	Bool	Rotate counterclockwise.
Two-Axis Pick & Place # Gripper CW	Output	Bool	Rotate gripper clockwise.
Two-Axis Pick & Place # Gripper CCW	Output	Bool	Rotate gripper counterclockwise.
Two-Axis Pick & Place # (Grab)	Output	Bool	Activate suction cup.
Two-Axis Pick & Place # (Moving X)	Input	Bool	Moving along X-axis.
Two-Axis Pick & Place # (Moving Z)	Input	Bool	Moving along Z-axis.
Two-Axis Pick & Place # (Rotating)	Input	Bool	Rotating.
Two-Axis Pick & Place # (Gripper rotating)	Input	Bool	Gripper rotating.
Two-Axis Pick & Place # (Detected)	Input	Bool	Detecting an item.

[fig13]: *tags di input ed output associati ai sensori e attuatori per il controllo dell'attività del Pick and Place.*

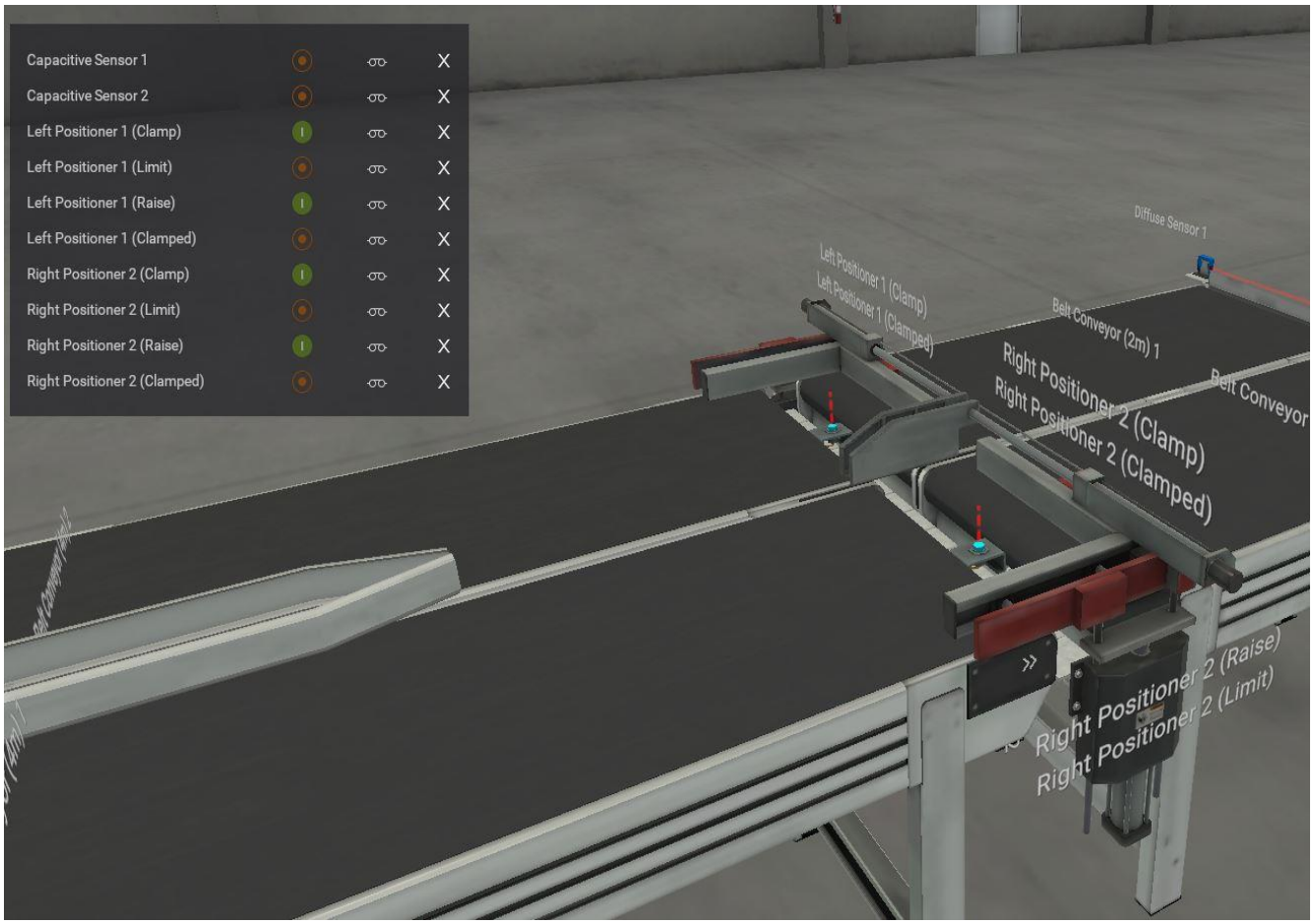
In realtà, per le finalità del Pick and Place all'interno dell'impianto, i tags di rotazione non verranno riferiti. I tags di input, associati a sensori, invece risultano relativamente importanti per la realizzazione dei vari movimenti permessi dagli attuatori.

Da notare bene che per garantire una corretta esecuzione, gli oggetti riferiti dal Pick and Place devono essere allineati perfettamente, rispettando delle precise posizioni sul piano d'appoggio.

Servono a tale scopo le due barre posizionatrici e i due sensori di presenza a supporto del Pick and Place stesso.

I posizionatori si occupano di dare la giusta posizione, per la fase di assemblaggio, ai materiali la cui presenza è indicata dai sensori sopraindicati.

Nell'illustrazione **[fig14]** sono anche indicati i relativi tags che ne regolano l'utilizzo.



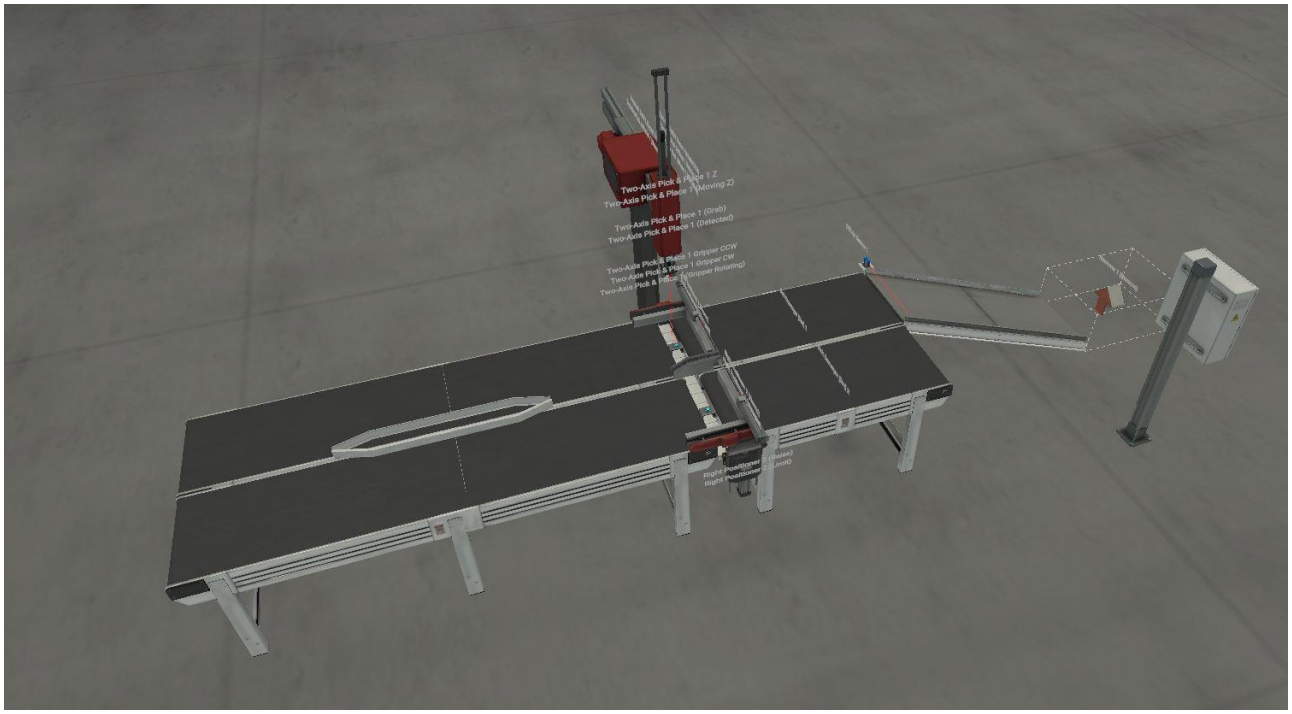
[fig14]: *una particolare disposizione di due barre posizionatrici e due sensori di presenza a supporto del processo di assemblaggio eseguito da un Pick and Place. In alto a sinistra i tags di controllo.*

Una volta terminata l'attività d'assemblaggio da parte del Pick and Place, il prodotto finale viene fatto scorrere verso un remover in uscita dove sarà dissolto dopo pochi secondi, simulando il trasferimento del pezzo verso una stazione successiva.

Proprio verso la fine del nastro d'uscita è posto un ulteriore sensore di prossimità il quale regola il conteggio dei prodotti finali, mostrandolo su un display.

[fig15]: *un display ed un remover con i propri tags in ambiente Factory I/O.*

Attraverso questi ultimi due elementi si è conclusa anche l'ultima zona dell'impianto, relativa alla produzione e al conteggio dei materiali finali **[fig16].**



[fig16]: *un Pick and Place a due assi ed un insieme di sensori, posizionatori e nastri trasportatori, compongono la zona di un impianto industriale dedita all'assemblamento e al conteggio di materiali.*

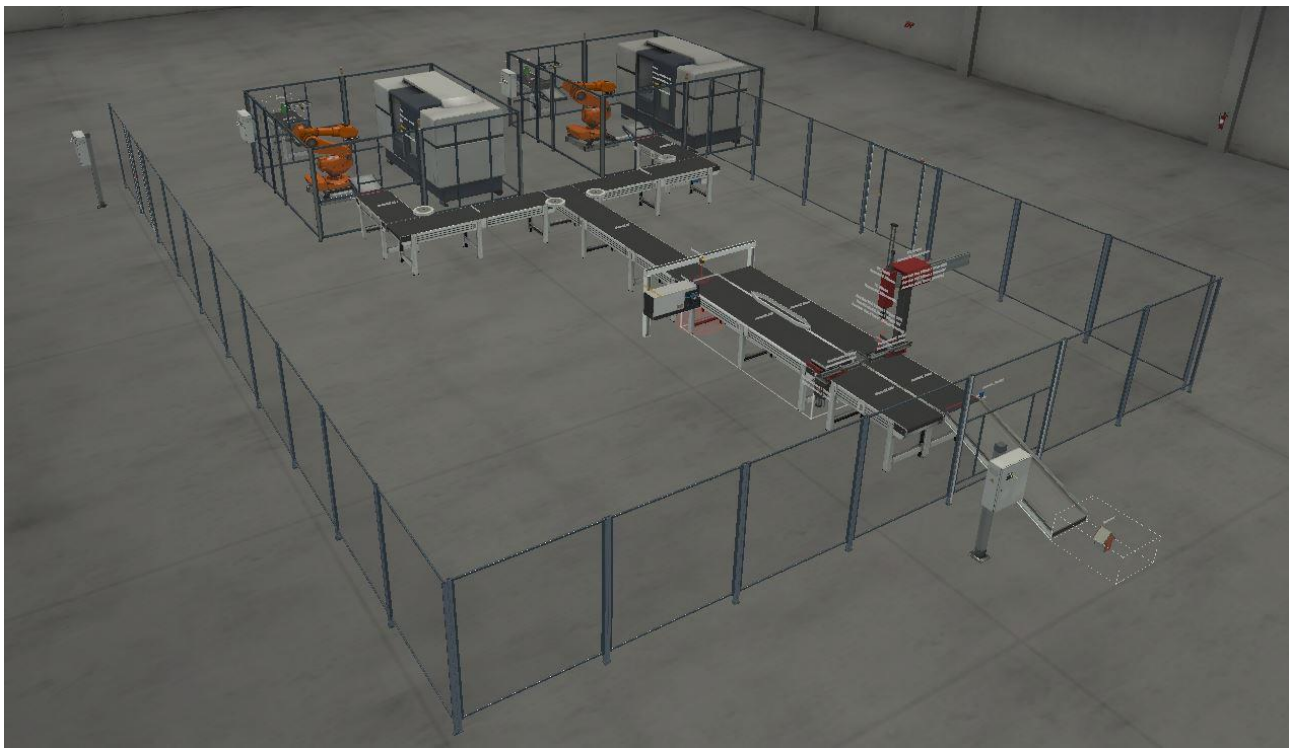
A completamento di tutte le aree precedentemente illustrate è stata aggiunta una switchboard con due pulsanti, start e stop, per permettere ad un operatore esterno di governare l'intero processo senza interferire direttamente con le parti dell'impianto **[fig17]**.

[fig17]: *switchboard di controllo dell'impianto ed i relativi tags.*

I pulsanti di start e stop sono due semplici operatori dotati di due tags ciascuno associati a variabili di tipo bool.

Il tag di output regola il segnale luminoso del pulsante, quello di input segnala quando il pulsante è premuto.

Componendo opportunamente tutte le parti e recintando la zona non direttamente accessibile, si otterrà l'impianto finale **[fig18]**.



[fig18] *Simulazione di un impianto industriale per la produzione di materiali in ambiente FactoryIO* .

2.2 Implementazione dell'algoritmo di controllo attraverso OpenPLC Editor

Attraverso l'attività svolta in ambiente Factory I/O si è ottenuto l'impianto virtuale prefissato e con esso un insieme di tags di input e output associati rispettivamente ai sensori e agli attuatori di ogni parte attiva dello stabilimento.

In questa fase verrà realizzato, attraverso OpenPLC Editor, un eseguibile che implementi una particolare logica di controllo per tale impianto andando a sfruttare le potenzialità di due linguaggi grafici supportati dall'editor stesso: l'SFC ed il Ladder.

Per realizzare l'eseguibile di controllo è stato suddiviso il progetto in 5 POUs di tipo programma, di cui 4 in lingua SFC ed una in Ladder.

I 5 programmi coinvolgono le 4 fasi già ampiamente discusse nel paragrafo precedente più una fase di gestione dei pulsanti presenti sul quadro elettrico di comando dell'intero impianto.

Abbiamo quindi un programma per la produzione, uno per il sorting, uno per l'assemblaggio, uno per il conteggio ed infine un programma main.

Per il progetto è stata creata una configurazione di risorsa apposita nella quale sviene definito un task singolo per le istanze di ogni programma sviluppato.

Nello specifico quindi si avranno 5 tasks ognuno dei quali in maniera ciclica, ad intervalli di 20 ms, va ad eseguire le istanze dei programmi a cui si riferisce.

Essendo tasks separati è stata attribuita a tutti priorità massima, 0.

Tasks:

Nome	Triggering	Singolo	Interval	Priorità
task0	Ciclico		T#20ms	0
task1	Ciclico		T#20ms	0
task2	Ciclico		T#20ms	0
task3	Ciclico		T#20ms	0
task4	Ciclico		T#20ms	0

Instances:

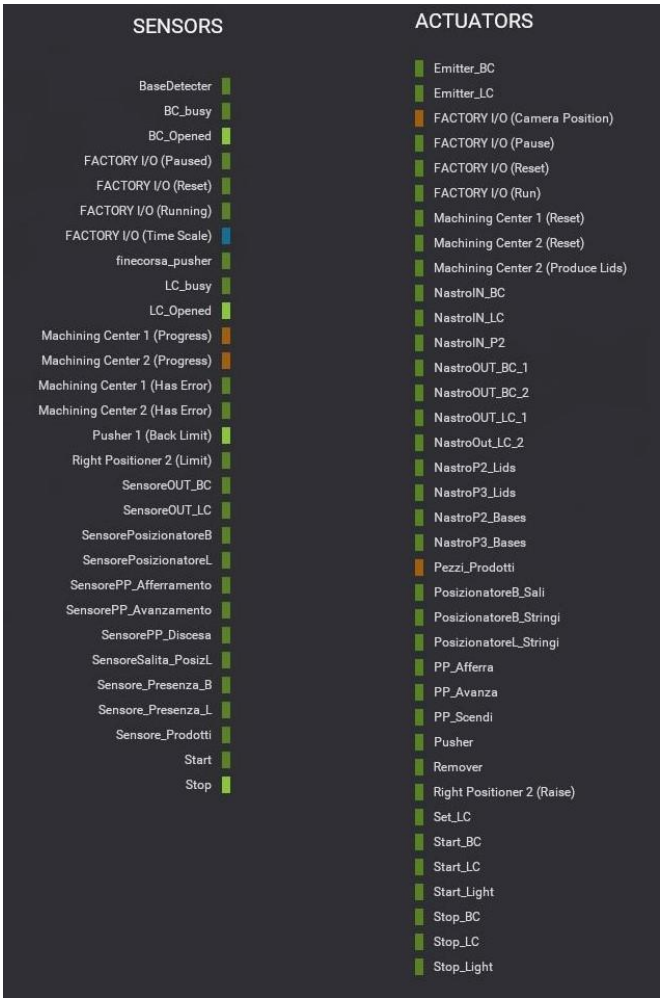
Nome	Yipo	Task
instance0	main	task0
instance1	produzione	task1
instance2	sorting	task2
instance3	assemblaggio	task3
instance4	conteggio	task4

[fig18]: *file di configurazione.*

2.2.1 Le variabili

Per arrivare a definire l'intero set di variabili è molto utile avere una vista completa di tutti i tags dell'impianto che deve essere controllato, da cui poi opportunamente ricavare quelli utili per la particolare politica di controllo da realizzare.

A tale scopo è d'aiuto la sezione Drivers di FactoryIO, da cui è appunto presa la seguente illustrazione.



[fig19]:*una vista completa dei tags dell'impianto virtuale.*

Factory I/O suddivide tutti i tags dell'impianto in tags di input e tags di output a rappresentanza dei sensori e attuatori della scena.

L'obiettivo è quello di dotare l'eseguibile di un set di variabili rappresentative di tali sensori e attuatori, utilizzandole per implementare la logica di controllo alla base del PLC.

Per il progetto sono state utilizzate principalmente variabili locali e globali, queste ultime andranno poi richiamate come extern nelle varie POUs che ne usufruiscono.

La definizione delle variabili globali è effettuata nel file di configurazione.

Filtro Classi: Tutti

#	Nome	Classe	Tipo	Location	Initial Value	Opzione	Documentazione
1	Stop	Global	BOOL	%IX100.0	1		Variabile legata al bottone di Stop, governante l'intero processo
2	processo1	Global	BOOL		0		Variabile d'appoggio utilizzata come punto di ingresso per la fase di produzione
3	processo2	Global	BOOL		0		Variabile d'appoggio utilizzata come punto di ingresso per l'attività di sorting
4	processo3	Global	BOOL		0		Variabile d'appoggio utilizzata come punto di ingresso per la fase d'assemblaggio

[fig20]:*Insieme di variabili globali del progetto in OpenPLC.*

Si noti che il campo location costituisce l'indirizzo di memoria utilizzato per il PLC addressing, espresso nella seguente forma: %ABxxx.

La specifica è riferita a Modbus, uno dei possibili standard per il trasferimento dati su bus di campo.

La scelta del particolare indirizzamento delle variabili sarà chiarità nella parte finale dell’elaborato.

Detto ciò nella seguente illustrazione si presenta l'intero set di variabili del progetto, opportunamente documentate.

Nome	Classe	Yipo	Location	Initial Value	Opzione	Documentazione
Start	Locale	BOOL	%IX100.1			Variabile legata al pulsante di Start , governante l'intero processo. Vera se il pulsante è premuto.
Start_Light	Locale	BOOL	%QX100.0			Segnale luminoso relativo al pulsante di Start (VERDE)
Stop_Light	Locale	BOOL	%QX100.1			Segnale luminoso relativo al pulsante di Stop (ROSSO)
Emitter_LC	Locale	BOOL	%QX100.2			Emettitore di materie prime per il Machining Center per la produzione di coperchi (LidCenter). Se true genere pezzi per la lavorazione.
Emitter_BC	Locale	BOOL	%QX100.3			Emettitore di materie prime per il Machining Center per la produzione di basi (BaseCenter). Se true genere pezzi per la lavorazione.
NastroIN_LC	Locale	BOOL	%QX100.4			Nastro in ingresso al LidCenter
NastroIN_BC	Locale	BOOL	%QX100.5			Nastro in ingresso al BaseCenter
Start_LC	Locale	BOOL	%QX100.6			Attiva il Machining Center per la produzione di Lid
Start_BC	Locale	BOOL	%QX100.7			Attiva il Machining Center per la produzione di Basi
Stop_LC	Locale	BOOL	%QX101.0			Pone in stato di STOP il Machining Center per la produzione di Lid
Stop_BC	Locale	BOOL	%QX101.1			Pone in stato di STOP il Machining Center per la produzione di Basi
NastroOUT_LC_1	Locale	BOOL	%QX101.2			Primo Nastro d'uscita dal LC
NastroOUT_LC_2	Locale	BOOL	%QX101.3			Secondo Nastro d'uscita dal LC
NastroOUT_BC_1	Locale	BOOL	%QX101.4			Primo Nastro d'uscita dal BC
NastroOUT_BC_2	Locale	BOOL	%QX101.5			Secondo Nastro d'uscita dal LC
NastroIN_P2	Locale	BOOL	%QX101.6			Nastro d'uscita dal Processo1 . Esso instrada i materiali uscendi dalla fase di produzione , verso la fase di sorting
SetLC	Locale	BOOL	%QX101.7			Variabile di inizializzazione del LidCenter. Se essa è true la CNC relativa produce una parte superiore (LID) , una base altrimenti
LC_busy	Locale	BOOL	%IX100.2			LidCenter Occupato
BC_busy	Locale	BOOL	%IX100.3			BaseCenter Occupato
SensoreOUT_LC	Locale	BOOL	%IX100.4			Fotocellula di segnalazione di un materiale in uscita dal LidCenter
SensoreOUT_BC	Locale	BOOL	%IX100.5			Fotocellula di segnalazione di un materiale in uscita dal BaseCenter
LC_Opened	Locale	BOOL	%IX102.1			Segnalatore chiusura CNC Lids . Quando LC_Opened è false , la CNC sta producendo il materiale
BC_Opened	Locale	BOOL	%IX102.2			Segnalatore chiusura CNC Basi . Quando BC_Opened è false , la CNC sta producendo il materiale
time_dummy_SYNC	Locale	BOOL				variabile d'appoggio
NastroP2_Bases	Locale	BOOL	%QX102.0			Nastro trasportatore Basi
NastroP2_Lids	Locale	BOOL	%QX102.1			Nastro trasportatore parti superiori (Lid)
Pusher	Locale	BOOL	%QX102.2			Se true si attiva il pusher
BaseDetecter	Locale	BOOL	%IX100.6			Sensore Visivo che diviene true non appena riconosce una Base
finecorsa_pusher	Locale	BOOL	%IX100.7			limite superiore pusher, quando è true il pusher rientra.
NastroP3_Lids	Locale	BOOL	%QX102.3			Nastro trasportatore lids
NastroP3_Bases	Locale	BOOL	%QX102.4			Nastro trasportatore Basi
Remover	Locale	BOOL	%QX102.5			dissolvenza prodotti in uscita
PosizionatoreB_String	Locale	BOOL	%QX102.6			Stringi posizionatore basi
PosizionatoreB_Sali	Locale	BOOL	%QX102.7			Sali posizionatore basi
PosizionatoreL_String	Locale	BOOL	%QX103.0			Stringi posizionatore lids
PP_Afferra	Locale	BOOL	%QX103.1			Ventosa Pick and Place
PP_Avanza	Locale	BOOL	%QX103.2			Pick and Place avanza
PP_Scendi	Locale	BOOL	%QX103.3			Pick and Place scendi
finecorsa_PosizL	Locale	BOOL	%IX101.0			true quando ho posizionato correttamente la lid
finecorsa_PosizB	Locale	BOOL	%IX101.1			true quando ho posizionato correttamente la base
PP_Detected	Locale	BOOL	%IX101.2			Sensore ventosa Pick and Place
SensoreAvanza_PP	Locale	BOOL	%IX101.3			Pick and Place sta avanzando
SensoreScendi_PP	Locale	BOOL	%IX101.4			Pick and Place sta scendendo
SensorePresenzaB	Locale	BOOL	%IX101.6			Sensore presenza base nella regione di lavoro del PP
SensorePresenzaL	Locale	BOOL	%IX101.7			Sensore presenza lid nella regione di lavoro del PP
time_dummy_PP	Locale	BOOL				
time_dummy1_PP	Locale	BOOL				
dummy_stop	Locale	BOOL				
StopPressed	Locale	BOOL				
CTU0	Locale	CTU				
Sensore_Prodotti	Locale	BOOL	%IX102.0			Fotocellula segnalatrice di pezzi prodotti
Val_Max_Prod	Locale	INT		100		Arrivati a 100 , resetto il contatore
Val_Prod	Locale	INT	%QW100.0	0		Valore conteggio

[fig21]:*Set di variabili istanziate per il progetto in OpenPLC.*

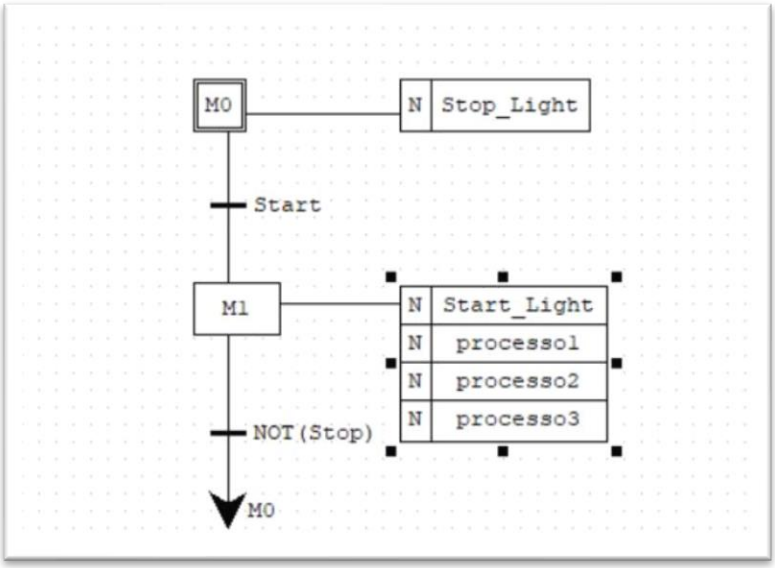
Si noti che laddove non venga espresso il valore iniziale associato alle variabili booleane è false.

2.2.2 Gli SFC

In questo paragrafo verranno presentati in maniera dettagliata l'insieme di SFC con cui è stata realizzata la logica di controllo dell'intero impianto. Saranno presi in analisi singolarmente tutti i 5 programmi da cui è composto il progetto.

Main

Come detto in precedenza il programma main regola la gestione dei pulsanti di Start e Stop presenti sulla switchboard.



[fig22]: *SFC di controllo del programma main .*

Lo stato di quiete dell'intero ambiente di produzione è rappresentato da una fase iniziale M0 in cui è ON la luce relativa al pulsante di STOP (impianto a riposo).

Nel momento in cui viene premuto il pulsante di START si passa in una fase secondaria M1 in cui le variabili associate alle tre azioni, essendo queste di tipo Normal, rimangono true finchè la fase M1 resta attiva.

Pertanto finchè non viene premuto il pulsante di STOP rimane accesa la luce del pulsante START (impianto in attività), e sono true le tre variabili globali processo1, processo2 e processo3. Come vedremo tra poco, e come in realtà si poteva già evincere dalla documentazione, tali variabili logiche sono associate alle transizioni abilitanti degli SFC di controllo dei differenti processi in cui si articola l'attività dell'impianto intero.

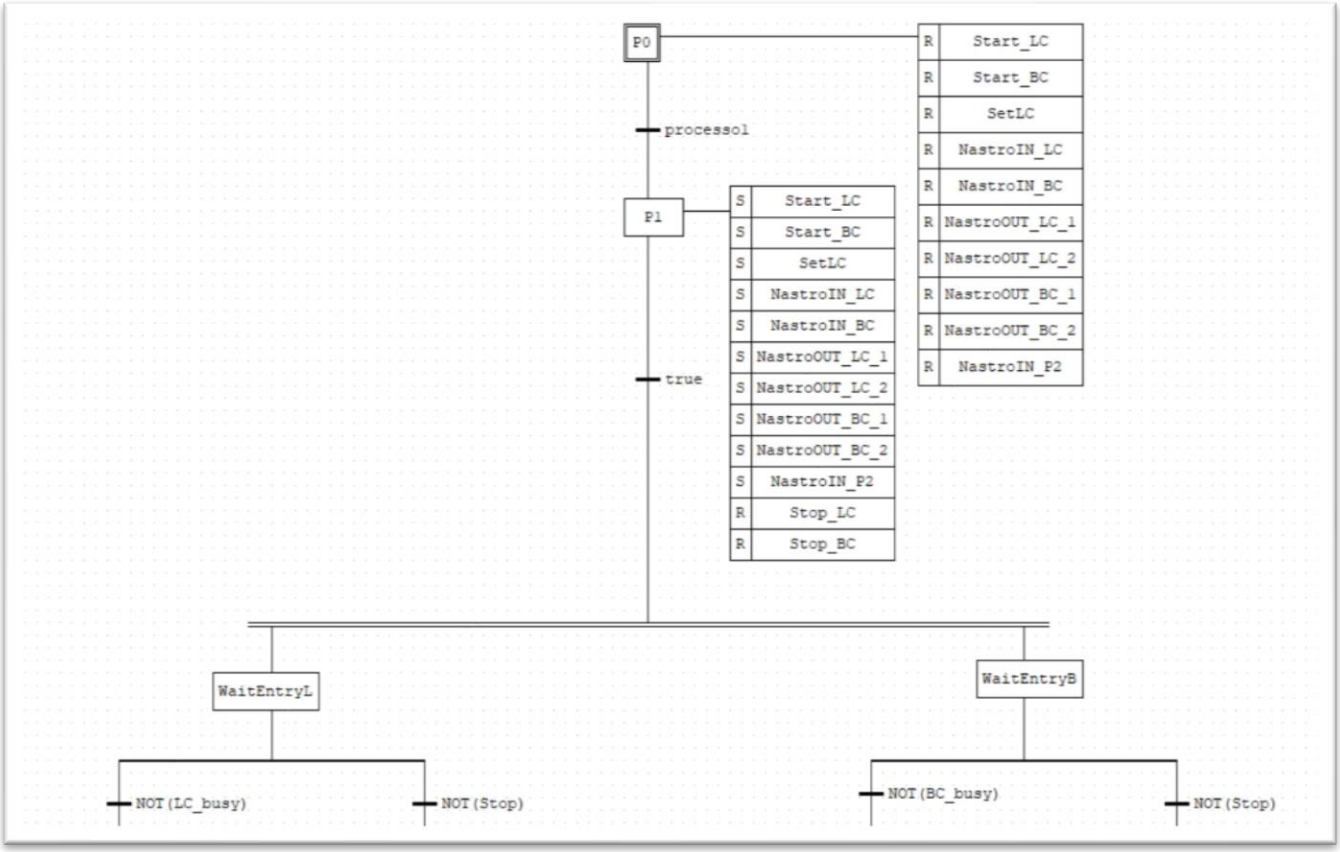
Produzione

Tale programma si riferisce alla fase di lavorazione delle materie prime mediante i Machining Centers e quindi alla relativa produzione di coperchi e basi.

Per arrivare all' implementazione delle finalità di tale fase ho realizzato due SFC separati.

Il primo controlla l'attività di produzione di basi e coperchi con riferimento alle due stazioni di lavorazione, mentre il secondo implementa una particolare logica di controllo dei nastri di uscita delle CNC in situazioni critiche.

Partiamo con l'analizzare il primo dei due.



[fig23]: *Prima parte del primo SFC di controllo del programma produzione.*

La fase iniziale P0 è caratterizzata da tutte azioni di tipo Reset, pertanto ogni volta che viene attivata P0 sono resettate le variabili booleane dei due centri di produzione e quelle relative a tutti i nastri utili alla fase di lavorazione.

Ciò vuol dire che finchè non vengono settate tali variabili i centri di produzione e tutti i nastri relativi rimangono inattivi. Il processo di produzione è in STOP.

La fase iniziale di questo primo SFC diviene superabile non appena la variabile globale 'processo 1' diventa true e, da come è stato spiegato in precedenza, ciò accade quando è premuto il pulsante di START posto sulla switchboard dell'impianto.

Al superamento della transizione 'processo1' si passa in una seconda fase, P1, in cui si settano i valori di start dei due Machining Centers nonchè tutti i nastri dediti al trasporto dei materiali per la fase di lavorazione.

Si setta inoltre anche la variabile SetLC che permette di configurare il relativo centro di lavorazione come produttore di coperchi (LidCenter).

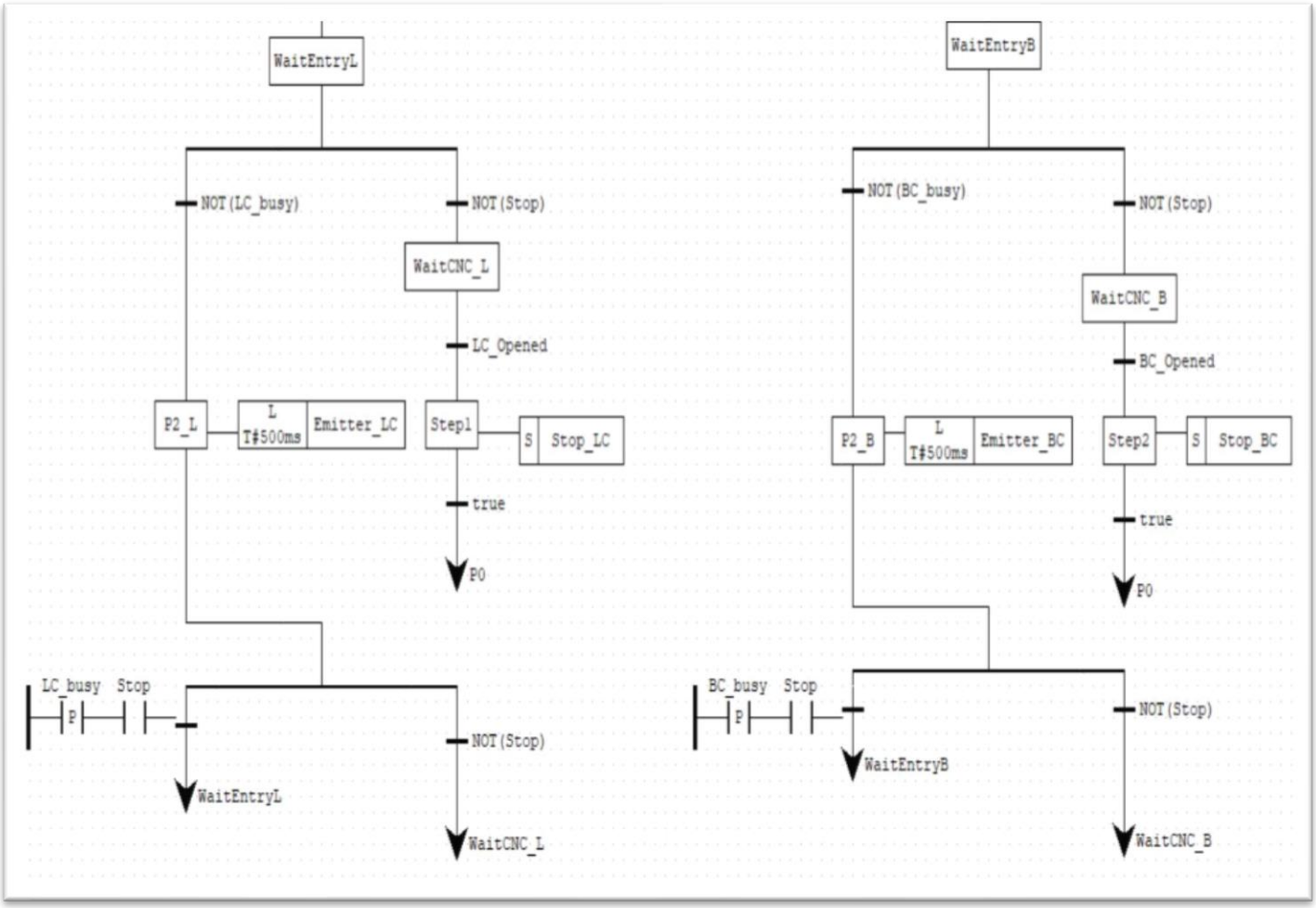
Si noti come la fase P1 rappresenti una cosiddetta 'fase instabile'. La transizione a valle infatti diviene superabile non appena la fase a monte viene attivata e come tale viene superata. Nonostante questo le azioni relative a tale fase sono comunque eseguite per una volta.

Una possibile alternativa è quella di utilizzare variabili logiche associate alle transizioni, vere dopo un determinato intervallo di tempo.

Dalla Fase P1 si passa quindi, mediante un costrutto di simultanea divergenza, in due fasi d'attesa: WaitEntryB e WaitEntryL.

Attraverso il costrutto di parallelismo infatti si è suddiviso il flusso esecutivo in due rami eseguiti in parallelo.

Le due fasi di attesa in cui si ritrova l'SFC rappresentano l'attesa in cui sono impegnati gli emettitori, rispettivamente in ingresso al centro di produzione di basi e coperchi, i quali aspettano che i Machining Centers a cui si riferiscono diventino liberi prima di generare un nuovo pezzo.



[fig24]: *Seconda parte del primo SFC di controllo del programma produzione.*

I due flussi esecutivi paralleli son basati sulla stessa logica di controllo, in relazione però ai due centri di produzione differenti.

In ambo i casi si rimane in attesa fino a che si verifica una fra le due condizioni:

- Il centro di lavorazione è pronto a lavorare un pezzo e lo stop non è premuto

- Il pulsante di stop viene premuto.

Per realizzare una condizione di scelta è stato utilizzato un costrutto divergenza in cui le condizioni devono sempre essere mutuamente esclusive fra loro.

Nel caso in cui il centro di lavorazione non è già occupato allora la prima transizione viene superata ed il flusso si porta in una fase P2 in cui si mantiene true per mezzo secondo la variabile bool associata all'emitter.

L'emettitore produrrà quindi una materia prima che, posta sul nastro di ingresso al centro di produzione lo impegnerà dopo poco.

Per mantenere vera la variabile associata all'emitter per un certo tempo è utilizzata un'azione limitated, caratterizzata da un marker di tipo L, indicandone l'intervallo temporale.

Sarebbe stato possibile anche utilizzare un’azione di tipo Pulse, marker P, ma usare una limited, creando un impulso più lungo, evita possibili errori in fase di simulazione in FactoryIO.

La fase P2 è abbandonata non appena si verifica una delle due condizioni (costrutto di divergenza):

- Il Machining Center passa da libero a impegnato e lo stop non è premuto.

- Il pulsante di STOP viene premuto.

Se la variabile indicatrice dello stato del Machining Center passa da libero ad occupato vuol dire che il materiale generato dall'emitter nella fase precedente ha effettivamente impegnato il centro di lavorazione.

In tal caso il flusso esecutivo ritorna alla fase di attesa in cui l'emitter si ritrova prima di generare un altro pezzo per il centro di lavorazione.

Le due divergenze che ho utilizzato servono a realizzare una forzatura dell'SFC non appena viene premuto il pulsante di stop.

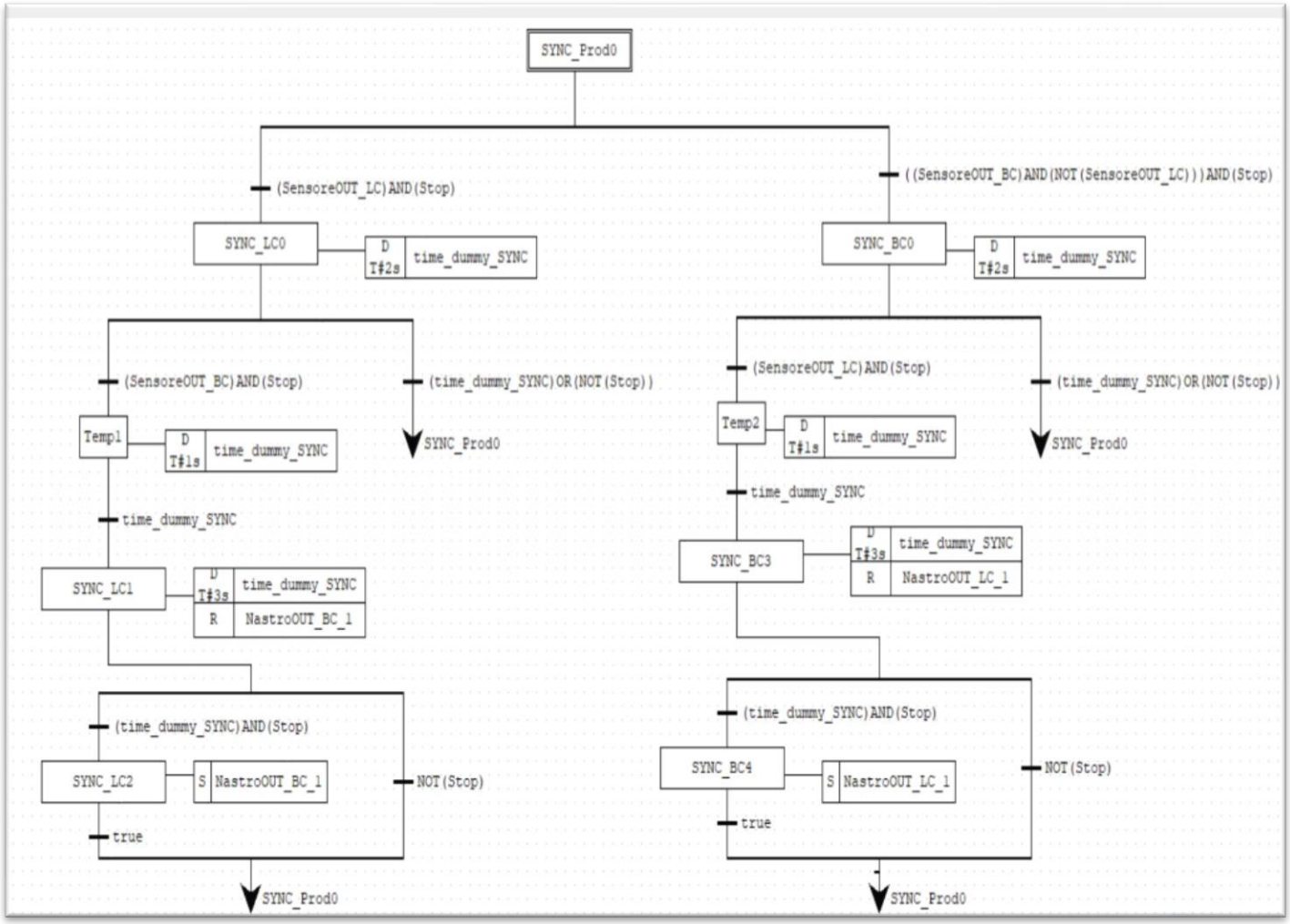
Se infatti la variabile Stop diviene true mentre il flusso esecutivo è in attesa di superare la a transizione a valle, si attende che le macchine CNC non siano occupate dopodichè si passa alla fase iniziale P0 arrestando nastri e centri di produzione.

Fermare l'attività di lavorazione delle macchine CNC causa infatti la produzione di materiali errati che non possono essere riutilizzati.

Il secondo SFC di controllo invece regola un eventuale caso critico che si verifica nel momento in cui le due produzioni, base e parte superiore (lid), sono completate allo stesso tempo o in due istanti temporali prossimi.

Tale situazione porterebbe ad avere sul nastro di ingresso al processo successivo (sorting) due materiali di produzione molto vicini o addirittura attaccati. In tal caso il sensore di riconoscimento visivo non riuscirebbe a distinguere il pezzo e se pure ci riuscisse l'azione di spinta del pusher arriverebbe a coinvolgere ambo i pezzi con errori irreparabili sull'intera attività di produzione.

Per tale motivo se le fotocellule di riconoscimento poste in uscita alle due macchine di produzione sono attivate l'una dopo l'altra, non rispettando un gap temporale fra le due attivazioni di almeno 2s, allora il nastro d'uscita su cui giace il pezzo prodotto per secondo (l'ultimo ad aver attivato la relativa fotocellula) è stoppato per 3s, in maniera tale da distaccare i due prodotti.



[fig25]: *Secondo SFC di controllo del programma produzione dedito alla gestione di casi critici.*

L'SFC che implementa tale logica di controllo è inizialmente in una fase priva di azioni.

Tale fase è superabile se si verifica una delle seguenti condizioni in maniera mutuamente esclusiva:

- Sensore d'uscita al centro produttivo di basi diventa true.

- Sensore d'uscita al centro produttivo di coperchi diventa true.

I due rami uscenti dalla fase iniziali sono frutto di un costrutto di divergenza quindi, come detto in precedenza, non potranno mai essere impegnati allo stesso tempo.

Entrambi comunque sono strutturati nel medesimo modo:

Se il sensore d'uscita di uno dei due centri di produzione risulta vero, allora si passa in una fase successiva dotata di un'azione di tipo Delayed applicata ad una variabile d'appoggio temporale time_dummy_SYNC. Tale variabile diviene vera dopo 3s.

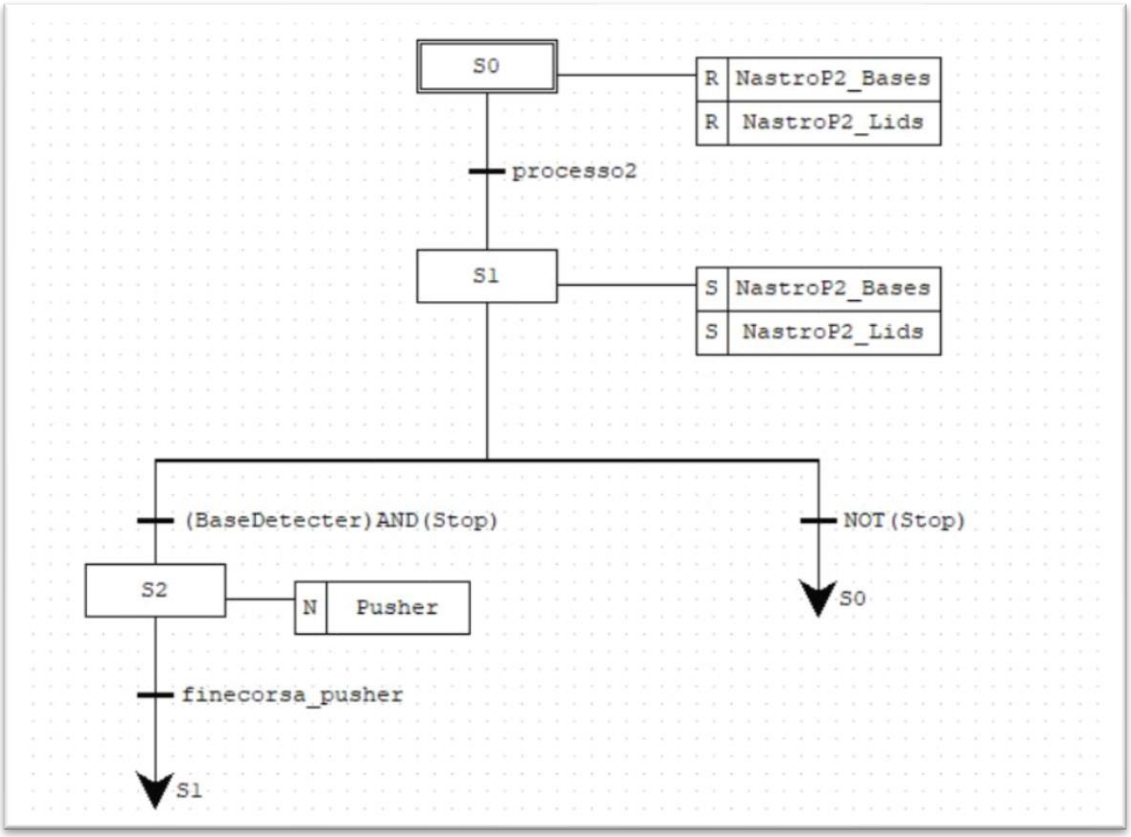
E’ stata utilizzata un’azione di questo tipo per emulare il comportamento di un variabile timer che, seppur prevista dall’editor di OpenPLC, crea problemi in fase di compilazione.

La transizione a valle è ancora una scelta: se nel giro di 2 secondi accade che il sensore d'uscita dell'altro centro produttivo diventa true, vuol dire che le due produzioni sono troppo prossime. Pertanto ci si ritrova in un caso critico e viene resetto il nastro che ospita il materiale arrivato per secondo per 3 secondi in maniera tale da distaccar i due prodotti. Passati i 3 secondi questo è riattivato.

Se invece nel giro di due secondi il sensore d'uscita dell'altro centro produttivo non rileva una produzione, oppure viene premuto il pulsante di stop, si ritorna alla fase iniziale dell'SFC aspettando che tale ciclo di valutazione riparta.

Sorting

La fase di sorting è gestita in maniera molto semplice attraverso un singolo SFC.



[fig26]: *SFC dedito al controllo dell'attività di sorting.*

Tale SFC si smuove dalla condizione iniziale, caratterizzata dai due nastri spenti, non appena la transizione rappresentata dalla condizione booleana 'processo2' risulta essere superabile.

Non appena si supera S0, si attivano i due nastri utili e l'SFC si pone in una fase d'attesa, in cui aspetta il verificarsi di una fra due condizioni (costrutto di scelta):

1) il sensore visivo ha riconosciuto una base e il pulsante di stop non è premuto

In tal caso dalla situazione d'attesa si passa ad una fase S2 in cui si rende true la variabile Pusher fino a che non si arriva al finecorsa, dopodichè si salta alla fase iniziale.

Tale sequenza di azioni porta a posizionare la base rilevata sul nastro apposito che la instraderà verso il processo di assemblaggio.

2) Si preme il pulsante di stop.

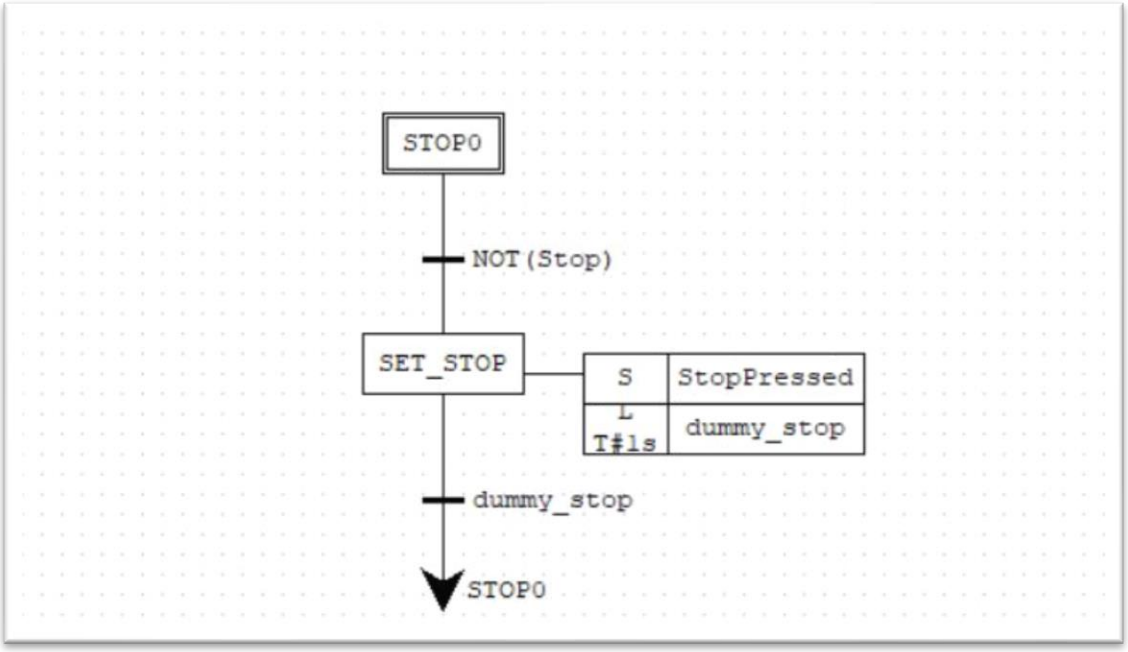
In tal caso si spengono i nastri ritornando poi in S0.

Assemblaggio

Il programma dedito alla fase di assemblaggio è strutturato in due SFC.

Il secondo SFC risulta necessario per la particolare gestione realizzata nel caso in cui viene premuto il pulsante di stop.

Esso si compone di due sole fasi da come si evince dalla seguente.

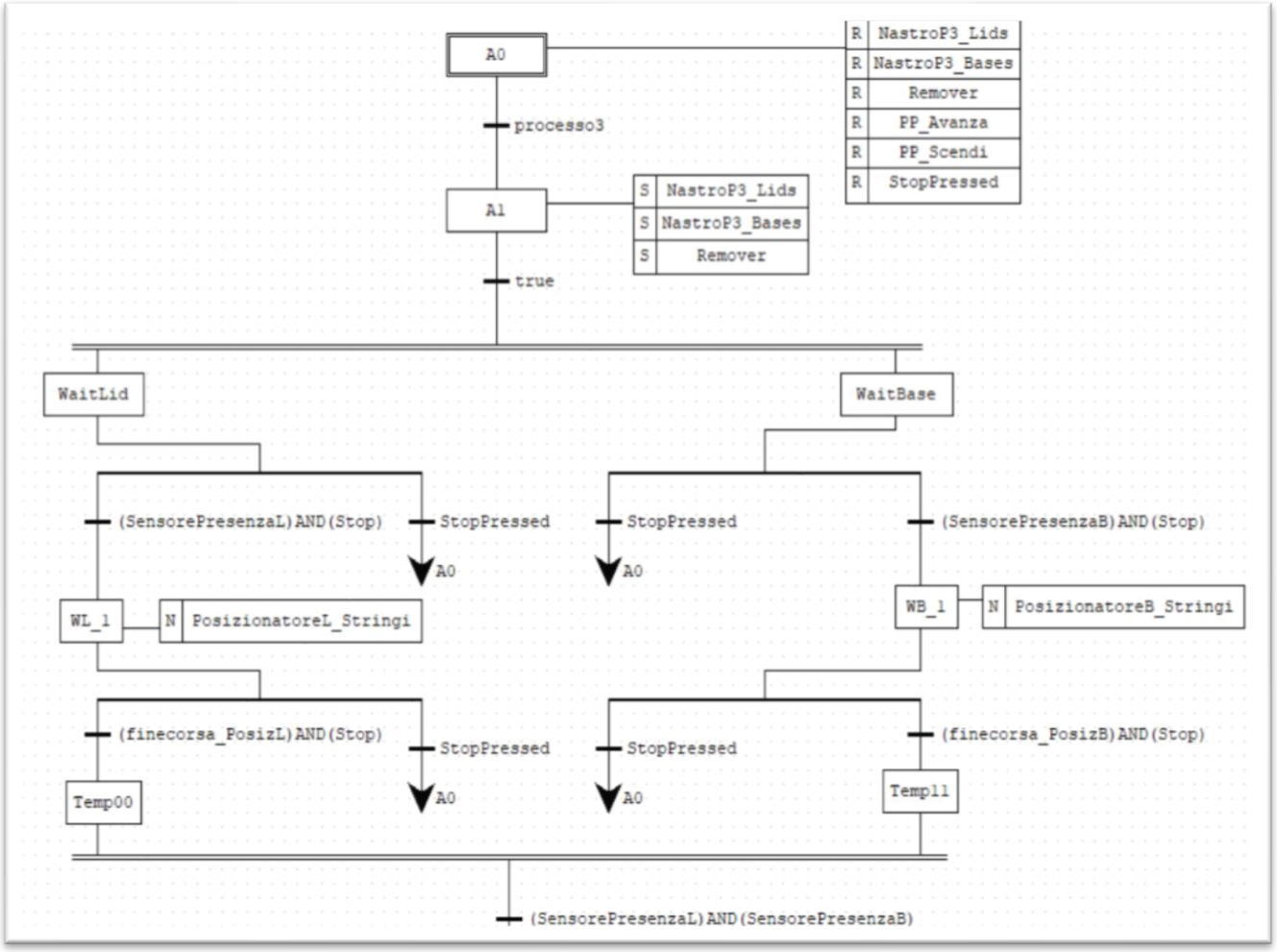


[fig27]: *Secondo SFC di controllo del programma assemblaggio dedito alla gestione del pulsante di stop .*

All'atto del superamento della fase iniziale, che avviene non appena è premuto il pulsante di stop, si passa ad una fase secondaria dove si setta una variabile d'appoggio StopPressed, indicatrice dell'avvenuta pressione del pulsante stop, si aspetta un secondo e si salta di nuovo alla fase iniziale.

Il secondo di attesa è necessario per non rendere vere entrambe le transizioni a valle e a monte di S2 all'atto della pressione di Stop.

L'SFC principale controlla l'attività di assemblaggio tenendo conto della variabile Stop_pressed regolata dal primo SFC.



[fig28]: *L'SFC principale che controlla l'attività di assemblaggio (parte1).*

Esso consiste in una fase iniziale A0 caratterizzata da tutte azioni Reset.

Ogni volta che essa viene attivata quindi si resettano le variabili relative ai nastri trasportatori, il remover dei prodotti finali, la posizione del Pick and Place e si pulisce la variabile d'appoggio StopPressed.

Non appena tale fase diviene superabile, ovvero quando processo3 è true (sistema in azione), si settano i nastri e il remover, dopodichè si passa immediatamente in due fasi di attesa, sdoppiando il flusso di esecuzione attraverso un costrutto di simultanea divergenza. Le due fasi di attesa seguono la stessa logica, applicata però a basi e coperchi.

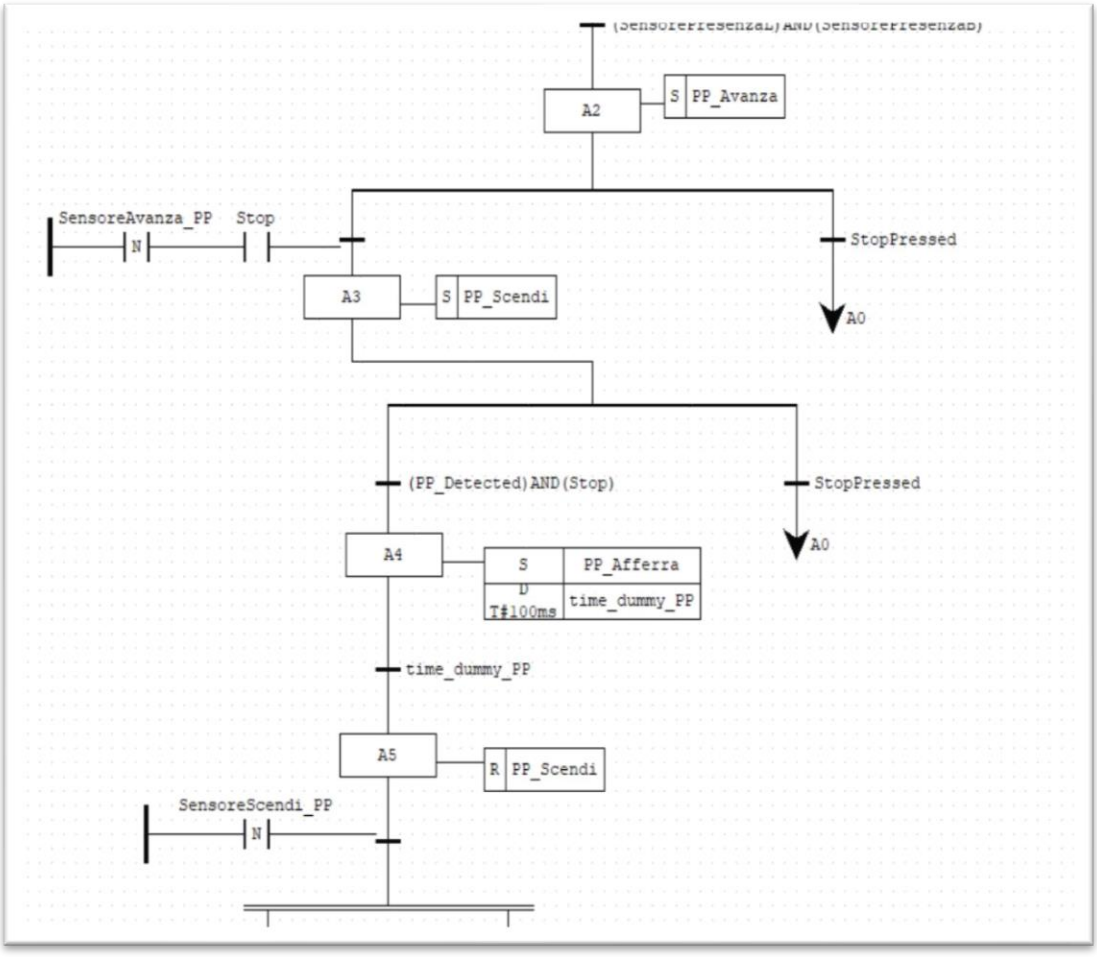
In ambo i casi la fase di Wait è superata non appena il sensore di presenza, locato poco prima delle barre posizionatrici, segnala che il materiale è pronto per la fase di assemblaggio e StopPressed è false. Se è vera la variabile StopPressed invece si salta alla fase A0 iniziale.

Nel primo caso è compito del rispettivo posizionatore regolarne la posizione. Ciò è reso possibile attraverso una semplice azione Normal eseguita fino al raggiungimento del finecorsa del posizionatore stesso.

Successivamente si arriva in un'ulteriore fase d'attesa necessaria per la sincronizzazione dei due flussi esecutivi attraverso un costrutto di convergenza. Si può passare alla fase successiva solamente dopo che sia la base che il coperchio sono stati opportunamente posizionati, risultando pronti per il processo di accorpamento.

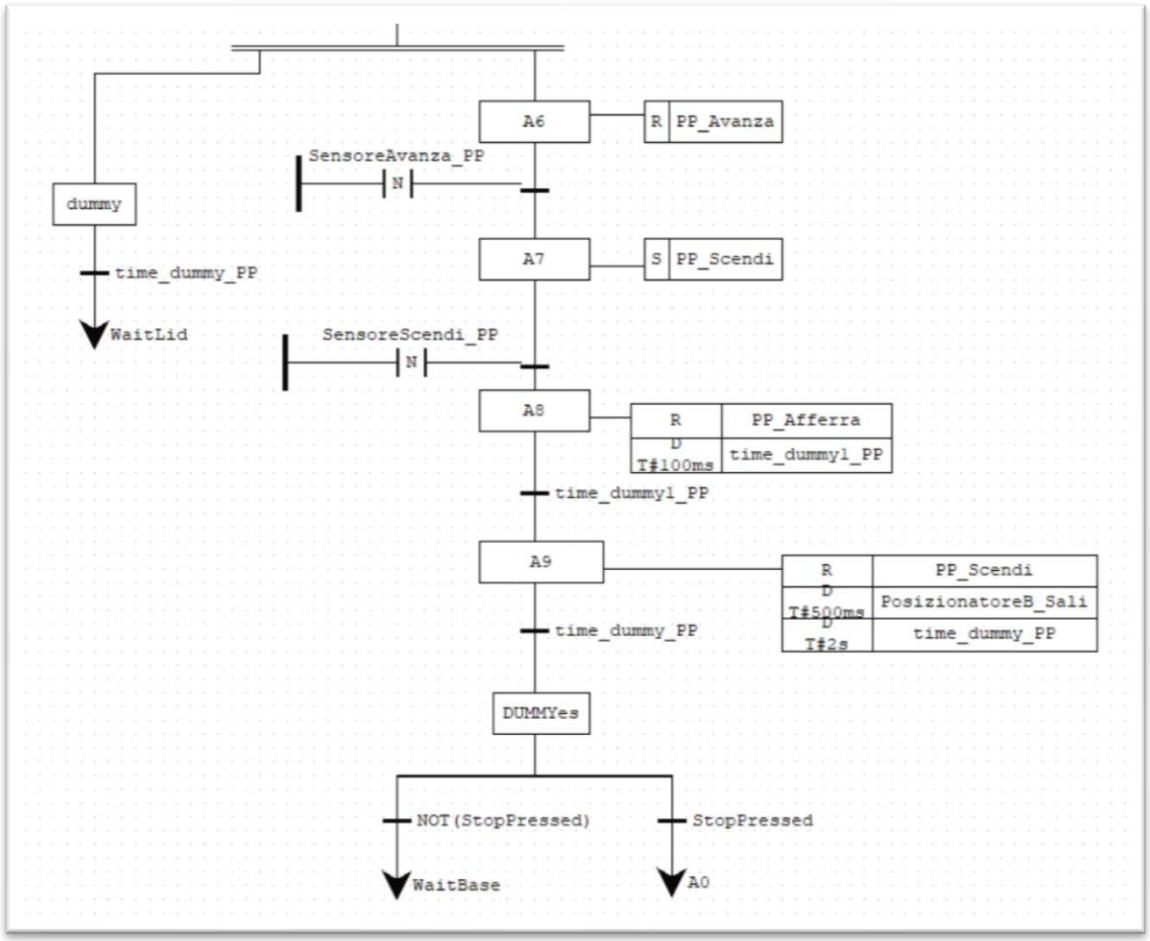
In questo caso l'SFC attraversa una serie di fasi in cui il Pick and Place si allunga posizionandosi al di sopra di un coperchio, scende toccandolo e dopo aver attivato la ventosa, lo tira su.

Nello specifico nella fase A2 si setta la variabile PP_Avanza che allunga il pick and place, dopodichè si attende che la variabile PP_StaAvanzando passi da 1 a 0, cioè che il PP abbia terminato di avanzare risultando nella posizione corretta. Si passa quindi in A3 in cui viene settata la variabile PP_scendi e si attende che la variabile che segnala il contatto fra la ventosa ed il pezzo diventi true. Verificatosi ciò si attiva la ventosa e dopo pochi millisecondi si resetta la variabile PP_Scendi che riporta il braccio del Pick and Place in alto.



[fig29]: *L'SFC principale che controlla l'attività di assemblaggio (parte2).*

Dalla fase A5 poi sdoppio ancora il flusso d'esecuzione con una simultanea divergenza.



[fig30]: *L'SFC principale che controlla l'attività di assemblaggio (parte3).*

Il primo ramo attende una variabile temporale per poi saltare nella fase s'attesa di un coperchio. Il secondo ramo invece, seguendo la stessa logica fin qui descritta, resetta la posizione del Pick and Place e appone il coperchio sulla base in attesa producendo il materiale finale.

La fase A9 riporta il braccio del Pick and Place in posizione di riposo e dopo mezzo secondo alza il posizionatore che teneva il prodotto finito, il quale, se non è vera StopPressed, verrà portato in uscita attraverso il nastro su cui giace.

Si noti che fino a quando non si passa alla fase A4 per ogni transizione viene valutata la variabile StopPressed che, nel caso sia true, forza l'arresto dell'intero processo ritornando alla fase iniziale.

Dalla fase A4 in poi tale variabile non è più valutata per ogni transizione dell'SFC ma solamente alla fine del processo di assemblaggio.

Questo perchè, onde evitare errori del Pick and Place, si è deciso che una volta afferrato il coperchio l'intero processo di assemblaggio deve esser portato a termine indipendentemente da un eventuale segnale di stop.

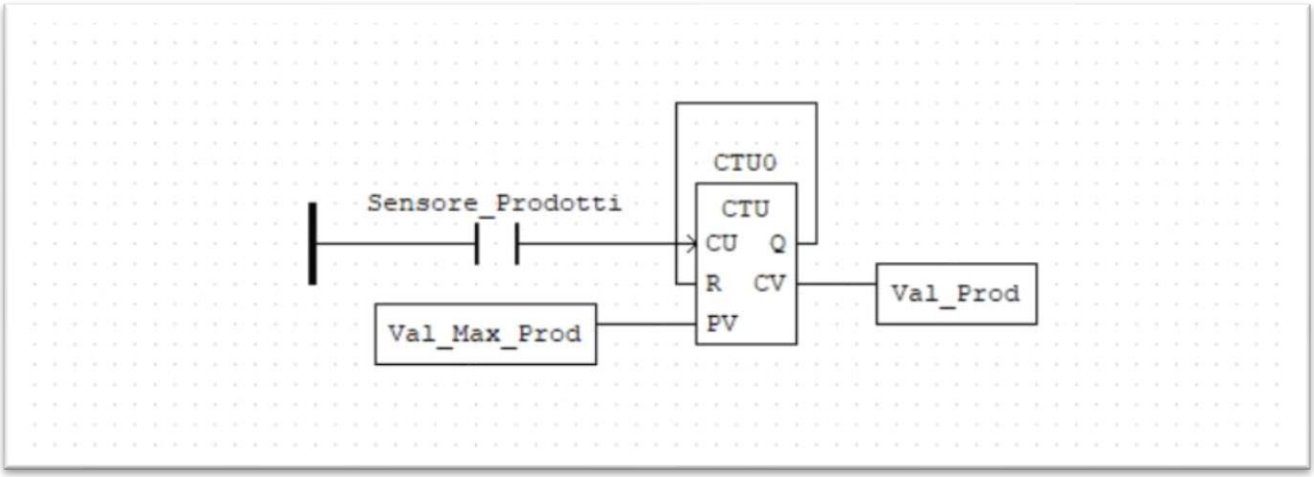
Se quindi un utente dell'impianto premesse lo stop button dopo che il braccio del Pick and Place ha afferrato il coperchio, ciò che accade è che il processo di accorpamento viene comunque terminato e solo alla fine di questo si salta in fase A0 resettando nastri e Pick and Place.

Conteggio

Per la fase di conteggio è stato realizzato un programma in linguaggio Ladder in cui si utilizza un semplice contatore ad incremento, uno fra i blocchi funzionali standard messi a disposizione dall'Editor.

Il contatore è incrementato seguendo la variabile associata alla fotocellula che rileva i prodotti finali in uscita dall'impianto.

Esso è azzerato non appena si contano 100 prodotti.



[fig31]: *Il contatore ad incremento con cui si realizza l'attività di conteggio dell'impianto.*

Compresa la logica di controllo implementata attraverso il progetto basterà compilare per generare il file eseguibile che avrà estensione '.st'.

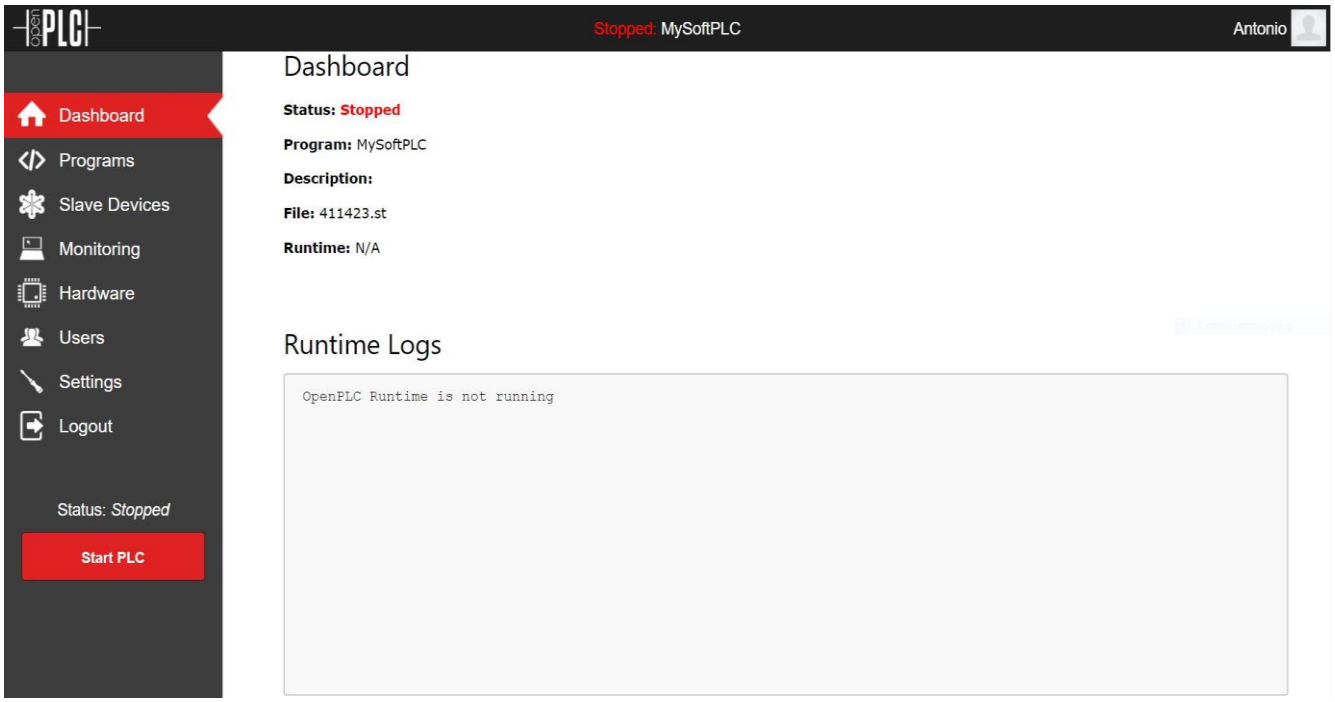
2.3 Controllo dell’impianto virtuale mediante SoftPLC

Generato correttamente il file eseguibile l'obiettivo è quello di utilizzare il Runtime di OpenPLC per mandarlo in esecuzione, simulando l'attività di un PLC standard.

A tal proposito il runtime OpenPLC ha un webserver integrato che consente, tra le tante cose, di caricare ed eseguire programmi. È possibile accedere al Webserver aprendo il browser Web sul computer e digitando l'indirizzo IP del dispositivo OpenPLC sulla porta 8080.

Nel caso in cui si utilizzasse Windows basterà digitare localhost: 8080 per accedervi ed una volta dentro potrà essere caricato il nuovo programma in formato .st.

Caricato il file esso dovrà nuovamente essere compilato e in caso di successo potrà essere eseguito.



[fig32]: *Una vista del webserver di OpenPLC runtime.*

La grande versatilità di OpenPLC è testimoniata dall’utilizzo del protocollo Modbus/TCP, uno dei protocolli più utilizzati a livello industriale che permette una comunicazione affidabile e veloce tra dispositivi di campo e d'automazione.

Tale protocollo realizza una comunicazione ETHERNET TCP/IP basata sul modello client/server e, in questo caso specifico, ha reso possibile il controllo dell'impianto virtuale in ambiente Factory I/O attraverso il SoftPLC concretizzato dall'attività del Runtime di OpenPLC.

Per poter connettere Factory I/O al SoftPLC bisognerà anzitutto impostare FactoryIO come un server TCP modbus, esplicitando nella sezione Drivers dello stesso software un dispositivo Modbus TCP/IP Server.

Tale dispositivo andrà configurato attraverso l'apposita finestra di configurazione. Il campo host solitamente è autogenerato ed indica l'IP della macchina su cui gira il sofware. In caso contrario va riempito manualmente con l'indirizzo IP della macchina host. Il passo successivo è impostare la porta che il modbus utilizzerà. La porta standard per modbus è la 502, tuttavia anche l'istanza OpenPLC la utilizzerà, quindi per evitare conflitti è necessario utilizzare una porta diversa in ambiente Factory I/O. Basterà comunque utilizzare un numero di porta più grande, 503 va bene. Infine va inserito uno Slave ID (1 va bene), per poi chiudere la schermata di configurazione e salvare.

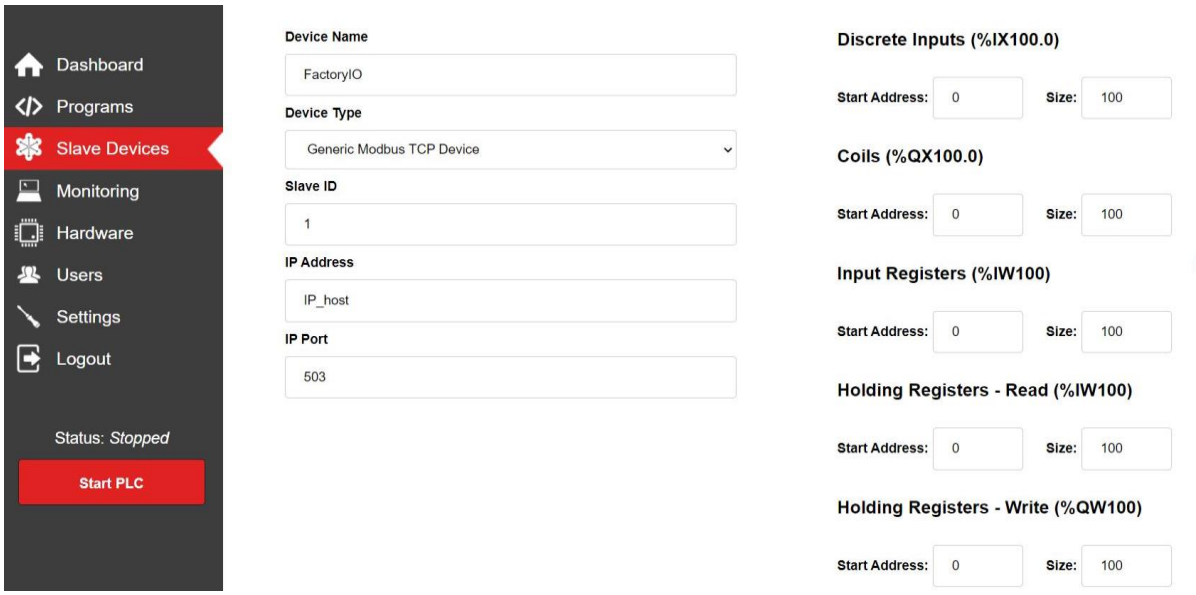
Si noti che all'atto della configurazione si possono anche settare il numero di I/O points con il quale lavorerà l'impianto. Se tutto va a buon fine FactoryIO dovrebbe ora eseguire un server modbus.

Conseguentemente andrà configurato, attraverso l'interfaccia web di OpenPLC runtime, un Modbus slave device per la comunicazione.

Lo slave device dovrà essere del tipo “Generic Modbus TCP Device” e settato con le specifiche già viste per il device server: IP della macchina host e numero di porta 503.

E' molto importante assicurarsi che lo slave id e il numero di porta indicati coincidano con quelli segnati in FactoryIO.

La sezione Slave Device del webserver di OpenPLC runtime permette inoltre di configurare i registri Modbus a cui si fa riferimento. Attraverso il campo Start Address è possibile indicare l'indirizzo del primo registro da leggere/scrivere, mentre con il campo Size si indica l'intervallo nonchè il numero di registri dopo il primo che vanno letti/scritti.



[fig33]: *Sezione Slave Devices dell’interfaccia web di OpenPLC runtime.*

La principale difficoltà nella configurazione di FactoryIO con OpenPLC è l'indirizzamento delle variabili.

Gli ingressi e le uscite di FactoryIO sono mappati su indirizzi OpenPLC a partire da 100 proprio perché i dispositivi Slave Device configurati per il Runtime di OpenPLC sono attaccati a partire da quella locazione di memoria.

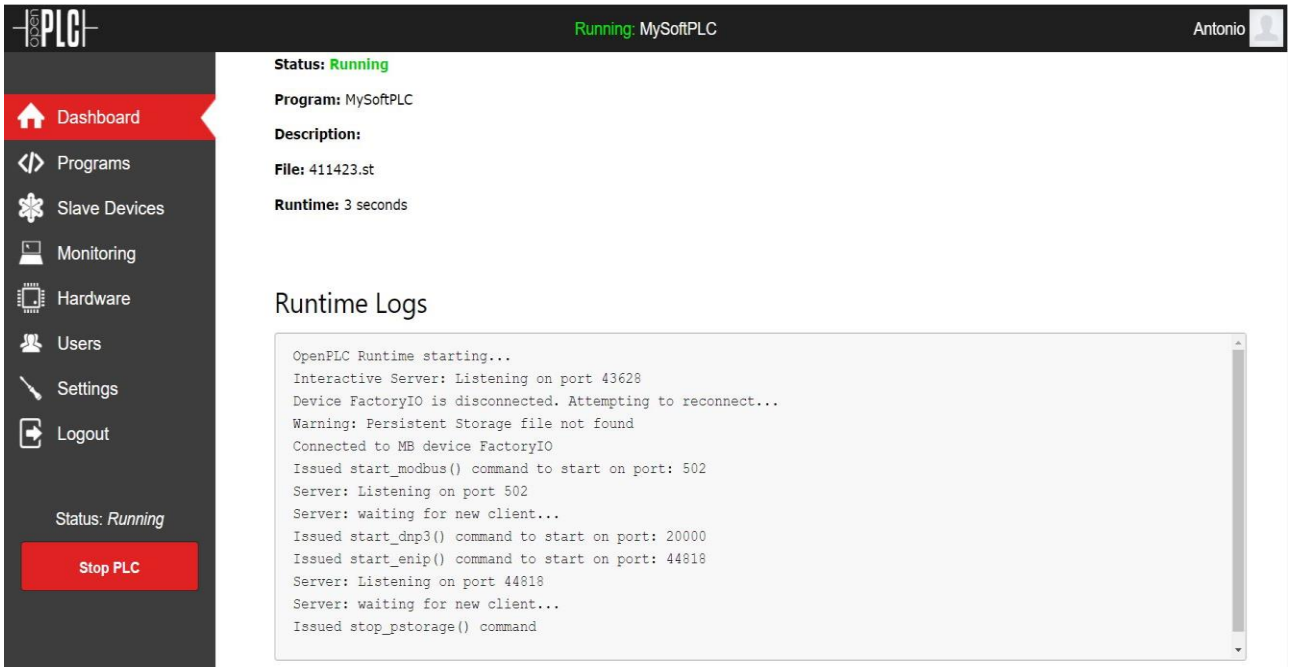
Modbus table	PLC Address	Data Size	Access†
Discrete Output Coils	%QX100.0 - %QX149.7	8-bit	W
Discrete Input Contacts	%IX100.0 - %IX149.7	8 bit	R
Analog Input Registers	%IW100 - %IW500	16-bit	R
Analog Output Holding Registers	%QW100 - %QW500	16-bit	RW

[fig34]:*Modbus Table per il PLC Addressing.*

Seguendo tale criterio in Factory IO andranno trascinati i vari tags nei rispettivi i/o points del dispositivo precedentemente settato.

[fig35]:*Configurazione degli i/o points del Driver in FactoryIO.*

Nel caso in cui tutto vada a buon fine, facendo eseguire il programma di controllo al Runtime nell'interfaccia web dovrebbe comparire un messaggio di effettiva connessione stabilita con lo Slave Device.



[fig36]: *Il runtime logs del webserver di OpenPLC Runtime mostra un messaggio di connessione riuscita del SoftPLC con lo SlaveDevice precedentemente settato.*

L'impianto virtuale in Factory I/O sarà così controllato mediante l'attività del SoftPLC realizzato attraverso OpenPLC Runtime.

Conclusioni

In questo elaborato di tesi è stata simulata una tipica attività di controllo di un processo industriale sfruttando le potenzialità offerte dai controllori a logica programmabile.

L'attività di PLC Training è stata realizzata attraverso l'utilizzo di OpenPLC come software di simulazione per PLC e di Factory I/O come ambiente di sviluppo per la creazione di un tipico scenario industriale da controllare.

L’obiettivo prefissato nella stesura di questo elaborato è stato quello di mostrare come la cooperazione di tali ambienti software sia da considerarsi come una possibile soluzione a basso costo che consente di effettuare la validazione e il test del software di controllo, prima della messa in servizio in un impianto reale.

Fonti Bibliografiche e Sitografia

Chiacchio P. e Basile F., Tecnologie Informatiche per L'Automazione, McGraw-Hill, 2004

Lucidi del Chiar.mo prof. Adriano Mele

Home Page OpenPLC, https://www.openplcproject.com/

Getting Started OpenPLC, https://www.openplcproject.com/getting-started

Storia di OpenPLC, https://forum.arduino.cc/index.php?topic=236945.0;wap

Forum OpenPLC, https://openplc.discussion.community/

Home Page Factory I/O, https://factoryio.com/

OpenPLC e Slave devices, https://www.openplcproject.com/runtime/modbus-slaves/

Protocollo Modbus, https://www.simplymodbus.ca/FAQ.htm

Setting up OpenPLC con Factory I/O, http://www.josephgardiner.com/setting-up-openplc-with-factoryio/