



DIPARTIMENTO DI INGEGNERIA ELETTRICA
E DELLE TECNOLOGIE DELL'INFORMAZIONE

Elaborato d'esame

in

TECNOLOGIE INFORMATICHE PER L'AUTOMAZIONE
INDUSTRIALE

AUTOMAZIONE DELLA FASE DI GERMINAZIONE DI UNA
COLTURA IDROPONICA

Professore:

Adriano Mele

Studenti:

Maria Coscione

Matr. N46002939

Stefano Licinio

Matr. N46002779

ANNO ACCADEMICO 2020/2021

Indice

1	Introduzione	4
1.1	Coltivazione idroponica: quali sono i suoi vantaggi?	4
1.2	DFT: Deep Flow Technique	5
1.3	L'idea	6
2	Automazione del processo industriale	8
2.1	Premessa	8
2.2	Obiettivi	8
2.3	Fasi del processo	9
2.3.1	Preparazione alla fase di germinazione	9
2.3.2	Fase di germinazione	9
2.4	Strumenti e tecniche utilizzate	10
2.4.1	Impianto di preparazione alla fase di germinazione	10
2.4.2	Controllo della temperatura all'interno della sala di germinazione	10
3	Progettazione della fase di preparazione	11
3.1	Soluzione proposta	11
3.2	Descrizione dei macchinari	13
3.2.1	Dispenser delle vaschette	13
3.2.2	Dispenser dei vasetti	16
3.2.3	Macchina torbatrice	22
3.2.4	Blocco 3	24
3.2.5	Blocco per l'irrigazione	34
3.3	Logica di simulazione	37

4 Regolazione della temperatura nella germination room	39
4.1 Modello	40
4.2 Taratura con formule di inversione	42
4.2.1 Requisiti di controllo	44
4.3 Simulink	45

1 Introduzione

In questo progetto si vuole automatizzare, ove possibile, una serra idroponica in un contesto urbano. Quest' idea trae origine dalla volontà di voler offrire l'opportunità di poter avere una produzione di ortaggi sempre fresca e a kilometro 0 anche in contesti proibitivi. Prima di iniziare con l'analisi del problema e della soluzione che vorremmo proporre, introdurremo brevemente cos'è l'agricoltura idroponica e i vantaggi del suo utilizzo.

1.1 Coltivazione idroponica: quali sono i suoi vantaggi?

La coltivazione idroponica è una tecnica di coltivazione agricola attraverso cui le piante crescono al di fuori del suolo: la terra è sostituita da un substrato inerte e la pianta viene irrigata con una soluzione nutritiva, la quale possiede ogni sostanza necessaria al suo normale sostentamento. I sistemi di coltivazione idroponici vengono solitamente installati in strutture *indoor* di modo che il sistema possa essere completamente controllato dal coltivatore, il quale potrà manipolare i vari fattori a cui esso sarà soggetto.

I vantaggi della coltivazione idroponica sono diversi e possono essere riassunti come segue:

1. minore consumo di acqua;
2. utilizzo mirato di fertilizzanti;
3. possibilità di produrre fuori stagione;
4. migliore sfruttamento degli spazi a disposizione.

A causa della continua crescita della popolazione mondiale e della conseguente disponibilità limitata di acqua potabile sulla Terra, la conservazione e l'utilizzo ottimale di questa risorsa è diventato ormai essenziale: colture che crescono utilizzando la tecnica idroponica o aeroponica richiedono fino al 90% di acqua in meno rispetto all'agricoltura intensiva[5], grazie al meccanismo di ricircolo e riutilizzo di essa. L'idroponica offre anche il pieno controllo del

sistema al coltivatore: proprio per questo motivo, essi non sottostanno a condizioni ambientali imprevedibili come siccità, erosione del suolo, ecc. Coltivare indoor difatti protegge le piante dalle condizioni esterne e consente agli agricoltori di produrre raccolti fuori stagione, pur essendo questo un punto che non riesce ancora a mettere d'accordo tutti: è meglio aumentare l'offerta e conseguentemente il margine di profitto oppure è meglio rispettare il ciclo stagionale delle coltivazioni, garantendo una continuità con quello che la natura ci offre?

Oltre ai vantaggi ambientali, i sistemi idroponici forniscono anche molti vantaggi pratici ai coltivatori: consentono di controllare i nutrienti e il pH nella zona delle radici portando a rendimenti costanti e più elevati della produzione, diminuendo l'uso di pesticidi, fertilizzanti ed erbicidi.

1.2 DFT: Deep Flow Technique

In questo progetto ci soffermeremo sull'utilizzo di una sola tecnica di coltivazione idroponica: la *Deep Flow Technique*. La scelta esclusiva di questa tecnica è dovuta soprattutto a due fattori:

- la robustezza di una soluzione idroponica rispetto ad una soluzione aeroponica, la quale pur consumando più acqua risulta meno incline a guastarsi col tempo[2];
- l'esistenza di impianti di produzione basati su DFT a cui potersi riferire in fase di ideazione e progettazione[3];

Le piante hanno una varietà di esigenze e requisiti diversi, per cui questo tipo di sistema deve essere monitorato per mantenere ottimali nella soluzione in cui cresceranno le piante i seguenti parametri:

- conduttività elettrica;
- pH;

- livello di ossigeno;
- concentrazione delle sostanze nutritive.

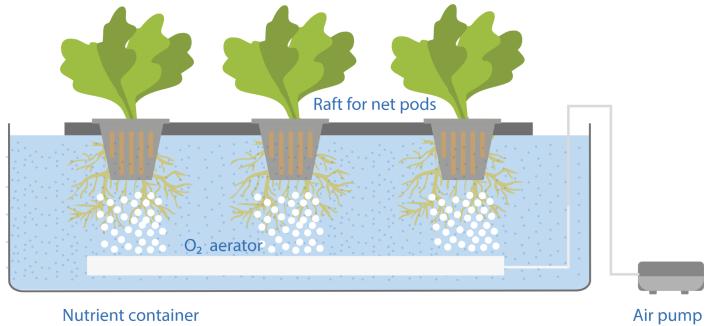


Figura 1: Schema Deep Flow Technique[5]

Il sistema mostrato nella Figura 1 è uno schema di base che ben rappresenta il funzionamento della DFT. Esso include inoltre un compressore d'aria e pietre porose di modo che il serbatoio venga aerato attivamente e i nutrienti miscelati adeguatamente. Le piante galleggiano sull'acqua utilizzando una "zattera", tipicamente realizzata in PVC a cui ci riferiremo con il nome di rack, e le radici sono immerse nella soluzione sottostante.

1.3 L'idea

L'idea di voler studiare un processo di agricoltura idroponica ci è venuta dopo aver letto il progetto *Vertical Farm 2.0*, soluzione vincitrice di un workshop europeo sull'agricoltura urbana sostenibile.[7]

Durante le varie fasi della lettura ci siamo accorti che, nonostante la qualità e la quantità dei dati forniti, la parte inherente al ciclo di vita degli ortaggi in esame (insalata e pomodori) non era stata automatizzata in maniera proporzionata al suo potenziale.

Secondo le nostre considerazioni, elaborate anche in virtù della pandemia tutt'ora in atto, ci sarebbero i seguenti punti critici da poter migliorare con soluzioni di automazione:

- ogni modulo di coltivazione (corrispondente ad un piano intero della Vertical Farm 2.0) ha un numero di lavoratori molto maggiore di quelli che si avrebbero adottando una soluzione ibrida (operatori + macchine), il che porta a costi maggiori senza un effettivo incremento della produzione. Inoltre, adottare una soluzione ibrida, permetterebbe una maggiore efficienza dello spazio utilizzato;
- ogni modulo di coltivazione è assimilabile ad un sistema chiuso: senza norme di decontaminazione adeguate e commisurate al numero di lavoratori, si potrebbe non tutelare la salute di questi ultimi, permettendo una facile diffusione di malattie;
- i moduli ad alto potenziale di automazione hanno un numero maggiore di lavoratori rispetto ai moduli non facilmente automatizzabili; adottare soluzioni ibride permetterebbe una redirezione di lavoratori dove ce ne sarebbe più bisogno, in modo da poter effettivamente aumentare la produzione senza causare disoccupazione. Un esempio potrebbe essere quello di formare come autotrasportatori (mansione non ancora automatizzabile) i lavoratori inoccupati dei moduli di coltivazione.

Dunque, essendo la sostenibilità ambientale un tema che la nostra generazione sente molto vicino, pensare ad una soluzione che potesse catalizzare il processo di installazione ed utilizzo delle serre urbane ci ha subito stuzzicato ed interessato.

2 Automazione del processo industriale

2.1 Premessa

L'obiettivo iniziale era quello di voler automatizzare ogni fase del processo, dalla semina alla raccolta e confezionamento, di una serra idroponica urbana: il problema è stato tuttavia rilassato, per una questione di tempistiche e risorse, e si farà riferimento esclusivamente all'impianto di preparazione alla fase di germinazione (il quale verrà illustrato in dettaglio nel capitolo 3) e fase di germinazione stessa per la coltivazione e produzione di insalata. La preparazione e produzione di quest'ultima è stata preferita alla coltivazione dei pomodori per i seguenti motivi:

- documentazione dettagliata: i dati raccolti ci hanno permesso di avere maggiore chiarezza e flessibilità nel modellare una soluzione che avesse più distintamente una nostra impronta;[4]
- maggiore facilità di preparazione e coltivazione: l'insalata, soprattutto in fase di preparazione, subisce un processo di lavorazione composto da task molto meccanici e, dunque, facilmente automatizzabili.

Ulteriori dettagli inerenti le fasi di coltura non trattate in questo progetto (nello specifico tutta la fase di coltivazione e produzione di pomodori) sono reperibili nella bibliografia.[7]

2.2 Obiettivi

Seppur ridimensionati, gli obiettivi di questo progetto sono:

1. **automatizzare la fase di preparazione alla germinazione**, così da avere un numero essenziale di lavoratori designati;

- tarare un regolatore PI affinché regoli automaticamente il valore della temperatura della sala di germinazione, in cui le piante trascorreranno la prima fase del loro ciclo di crescita.

2.3 Fasi del processo

2.3.1 Preparazione alla fase di germinazione

L'impianto di preparazione alla fase di germinazione dovrà preparare i vasetti con il substrato, piantare al loro interno i semi della lattuga, ricoprirli ed irrorarli con apposita soluzione nutritiva. Successivamente all'irrigazione, la quale segnerà il termine della lavorazione nell'impianto, un operatore preleverà le vaschette e le trasferirà all'interno della sala di germinazione, nella quale resteranno per quindici giorni (tempo necessario alla pianta per germogliare, come indicato dalla tabella sottostante).

Stage	Days	Temperature [°C]	Relative Humidity [%]	Light Intensity [$\mu\text{mol}/\text{m}^2/\text{s}$]	CO_2 [ppm]	Wind speed [m/s]
Germination Phase 1 [Germination room]	1,5 - 2	22	95	150	1.000	0,3-0,5
Germination Phase 2 [Nurseries]	14	22	80	200	1.000	0,3-0,5
Growth Phase 1	10	23	80	200	1.000	0,3-0,5
Growth Phase 2	9	23	80	225	1.000	0,3-0,5
Growth Phase 3	9	23	80	250	1.000	0,3-0,5

Figura 2: Condizioni di crescita della lattuga[7]

2.3.2 Fase di germinazione

La vera e propria fase di crescita dei semi di insalata inizia all'interno della **sala di germinazione**. Per permettere una crescita controllata in un caso reale, ci sarebbe bisogno di

monitorare i valori di *temperatura*, *umidità*, *luce* e *ventilazione* all'interno della sala; tuttavia, per quanto già discusso nella sezione 2.1, il nostro sistema si ridurrà ad un semplice modello ad un'equazione differenziale per lo scambio termico da una stanza ad un'altra. A capo del nostro impianto, verrà collegato un regolatore PI per poter regolare efficacemente il solo valore della temperatura in caso di uscita dal range di valori ottimali.

2.4 Strumenti e tecniche utilizzate

2.4.1 Impianto di preparazione alla fase di germinazione

Per l'implementazione della logica di controllo dei singoli macchinari è stato usato il linguaggio SFC, con qualche aggiunta di elementi in Ladder (per alcune azioni/transizioni particolari) e Structured Text (nello specifico per l'utilizzo dei flag SFCInit ed SFCPause).

Per l'implementazione della logica di simulazione, è stato utilizzato il solo linguaggio Ladder. Come editor e ambiente per le simulazioni è stato scelto CODESYS (V3.5 SP16 Patch 4). I disegni dei macchinari sono stati realizzati da noi su carta e, successivamente, su Adobe Illustrator.

2.4.2 Controllo della temperatura all'interno della sala di germinazione

Per l'implementazione di una soluzione al problema di controllo del modello di scambio termico, è stato utilizzato MATLAB dal quale abbiamo potuto ricavare facilmente tutti i diagrammi del caso. La soluzione è stata poi successivamente schematizzata su Simulink, al fine di poter essere simulata.

3 Progettazione della fase di preparazione

3.1 Soluzione proposta

In questo progetto andremo a proporre una soluzione di automazione per il processo di preparazione delle piantine alla fase di germinazione.

La soluzione adottata è stata quella di pensare a dei macchinari modulari ed indipendenti che lavorassero bene in un'ottica di catena di montaggio, scelta robusta a guasti e facile da controllare esternamente.

Nelle prossime sezioni verrà descritto il funzionamento atteso e la programmazione di ciascun macchinario: per ognuno di essi verranno illustrate sia le variabili che ne fanno parte e sia i diagrammi di funzionamento.

Successivamente, verrà illustrato anche il sistema di controllo dell'intero impianto, il quale si occuperà di svolgere compiti di supervisione e controllo in caso di emergenze.

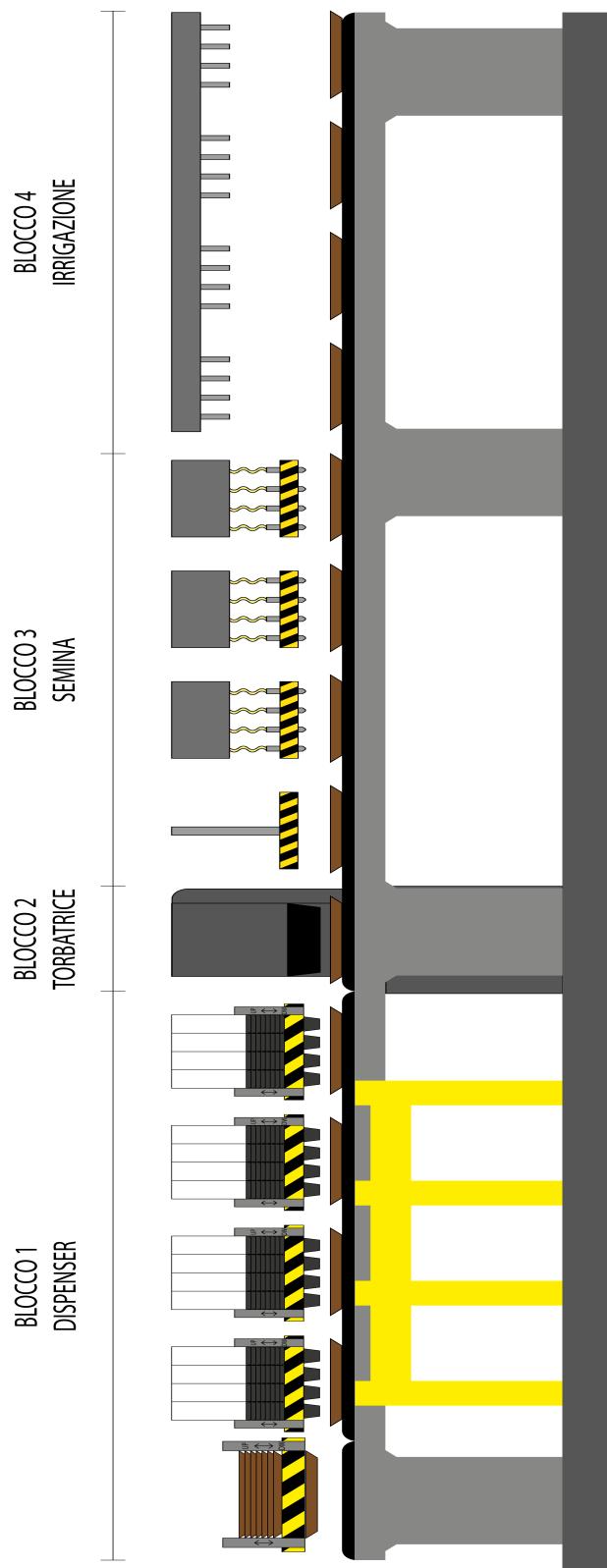


Figura 3: Impianto di preparazione

3.2 Descrizione dei macchinari

3.2.1 Dispenser delle vaschette

Generalità Il dispenser delle vaschette è un macchinario composto dal dispenser vero e proprio e da un suo nastro trasportatore. Lo scopo di questo macchinario è quello di depositare sul nastro una vaschetta, al cui interno verranno successivamente riposti i vasetti per l'insalata, e farla scorrere fino al macchinario successivo.



Logica di controllo Le immagini 5 e 6 riportano il codice inerente le variabili appartenenti ai due moduli del blocco dispenser vaschette; nelle immagini 7 e 8 sono riportate le implementazioni dei due moduli che compongono il blocco dispenser vaschette.

Figura 4: Particolare del dispenser vaschette

```

1 PROGRAM dispenser_vaschette
2 VAR RETAIN
3     // Il numero di vaschette presenti all'interno del dispenser.
4     num_vaschette: INT := 100;
5 END_VAR
6 VAR
7     // Se vero, viene allentata la presa del gancio sulla vaschetta, la quale viene rilasciata sul nastro trasportatore.
8     sgancia: BOOL := false;
9     // Attuatore che muove il dispenser verso l'alto.
10    motore_su: BOOL := false;
11    // Attuatore che muove il dispenser verso il basso.
12    motore_giu: BOOL := false;
13    // Sensore che indica il punto più alto raggiungibile dal dispenser.
14    finecorsa_alto: BOOL := true;
15    // Sensore che indica il punto più basso raggiungibile dal dispenser.
16    finecorsa_basso: BOOL := false;
17    // Azione che attiva il flag SFCPause. Porta il macchinario in uno stato di pausa dopo aver esaurito le vaschette al suo interno.
18    sospendi_macchina: BOOL;
19    // Spia che diventa vera se il macchinario va in sospensione a causa del termine delle vaschette.
20    // Si resetta automaticamente dopo il riavvio automatico che segue l'avvenuto caricamento delle vaschette nel dispenser.
21    spia: BOOL := false;
22 END_VAR
23 VAR_INPUT
24     // Variabile di input che, se vera, porterà l'SFC ad evolvere verso gli stati della gestione delle emergenze.
25     // Il suo valore verrà modificato esternamente a questo macchinario.
26     controllo_reset: BOOL;
27 END_VAR
28 VAR
29     // Questa azione agisce sul flag SFCInit, il quale porta il macchinario in uno stato di arresto.
30     ferma_macchina: BOOL;
31 END_VAR

```

Figura 5: variabili del solo dispenser vaschette

```

1 PROGRAM nastro_dispenser_vaschette
2 VAR
3     // Attuatore che mette in moto il nastro.
4     motore: BOOL;
5     // Questa azione agisce sul flag SFCPause, il quale se è vero mette in pausa l'esecuzione dei task del macchinario.
6     // Questa eccezione è gestita dal nastro dopo che è diventata vera la stessa variabile all'interno del macchinario collegato.
7     sospendi_macchina: BOOL;
8 END_VAR

```

Figura 6: variabili del nastro trasportatore associato

Il dispenser delle vaschette si trova inizialmente al finecorsa alto in uno stato di quiete. All'atto dell'avvio del macchinario, controlla che ci siano vaschette al suo interno per poter effettuare le operazioni e che non sia vera la variabile controllo_reset, la quale lo porterebbe verso uno stato di emergenza. Se non ci sono vaschette all'interno, il macchinario accende una spia per segnalarlo; solo ad avvenuto caricamento potrà ripartire. Dopo aver effettuato il controllo, attende che il suo nastro sia fermo e, se la variabile di controllo non è vera, allora inizia le seguenti operazioni:

1. scende fino al suo finecorsa basso;

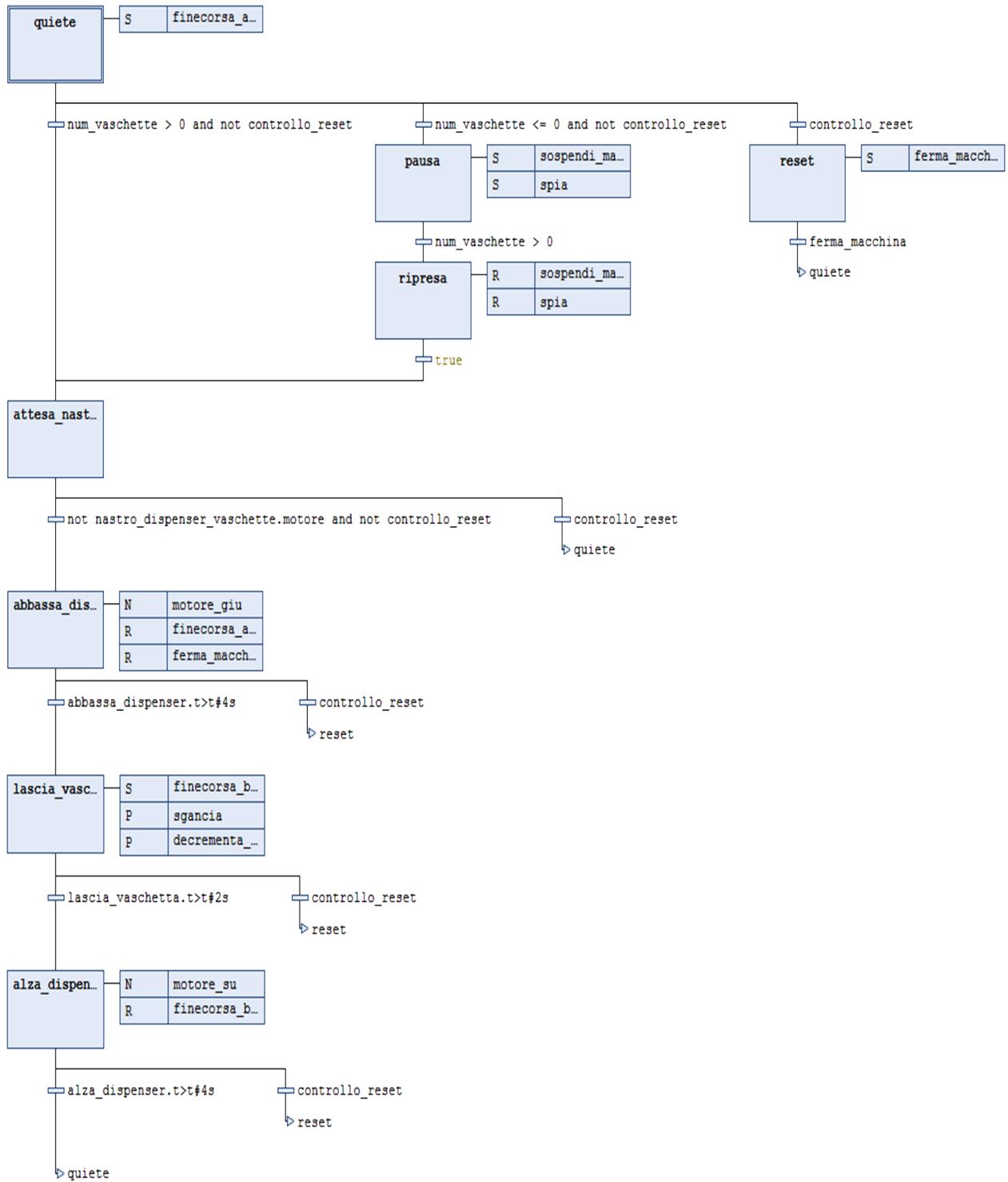


Figura 7: logica di controllo del dispenser vaschette

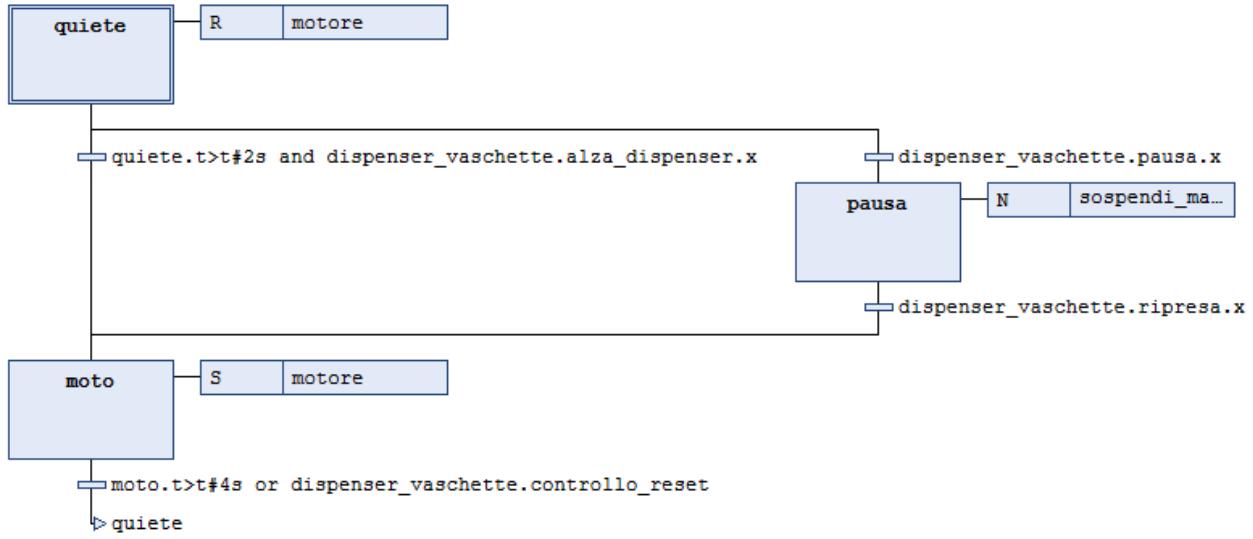


Figura 8: logica di controllo del nastro trasportatore associato

2. dopo aver raggiunto il finecorsa basso, sgancia una sola vaschetta sul nastro;
3. sale fino al suo finecorsa alto;
4. dopo aver raggiunto il finecorsa alto, si riporta nello stato di quiete iniziale.

Il nastro trasportatore è inizialmente fermo in uno stato di quiete. Quando il dispenser avrà sganciato la vaschetta, verrà attivato il motore del nastro il quale resterà acceso per 4 secondi prima di essere spento e far tornare il nostro sistema in quiete.

Inoltre, come si può notare nelle immagini 7 e 8, tra un'operazione e l'altra vi è **sempre** la possibilità che si attivi la transizione che porterebbe il macchinario nello stato di emergenza (variabile controllo_reset vera).

3.2.2 Dispenser dei vasetti

Generalità Il dispenser dei vasetti è, similmente al blocco per le vaschette, composto da un modulo dispenser e da un nastro associato: la differenza sostanziale sta nel fatto che in

questo macchinario non è il dispenser a muoversi ma il nastro stesso, tramite un sistema di pistoni idraulici (che in *figura 9* sono di colore giallo).

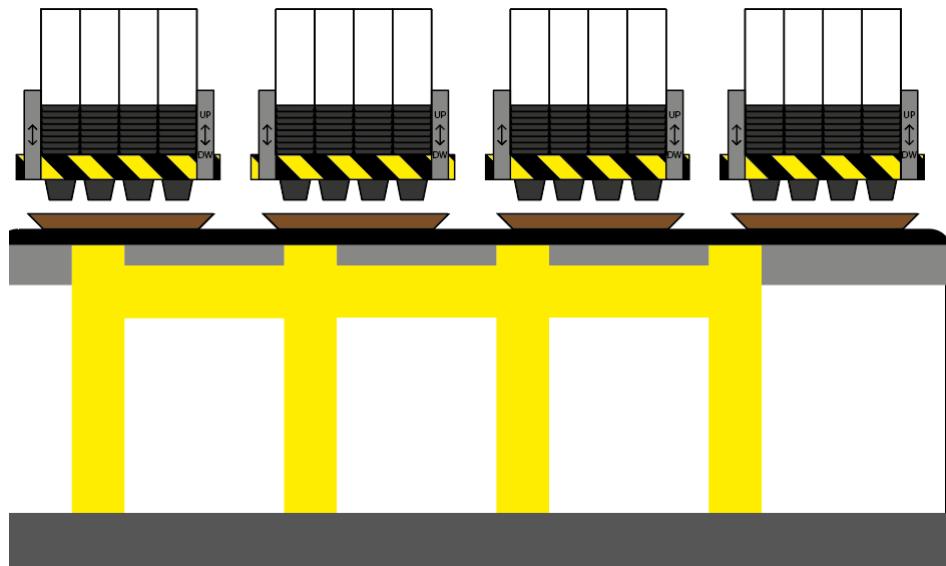


Figura 9: particolare del blocco dispenser vasetti

Logica di controllo Le immagini 10 e 11 riportano il codice inerente le variabili appartenenti ai due moduli del blocco dispenser vasetti; nelle immagini 12 e 13 sono riportate le implementazioni dei due moduli che compongono il blocco dispenser vasetti.

```

1 PROGRAM dispenser_vaschette
2 VAR RETAIN
3     // Il numero di vaschette presenti all'interno del dispenser.
4     num_vaschette: INT := 100;
5 END_VAR
6 VAR
7     // Se vero, viene allentata la presa del gancio sulla vaschetta, la quale viene rilasciata sul nastro trasportatore.
8     sgancia: BOOL := false;
9     // Attuatore che muove il dispenser verso l'alto.
10    motore_su: BOOL := false;
11    // Attuatore che muove il dispenser verso il basso.
12    motore_giu: BOOL := false;
13    // Sensore che indica il punto più alto raggiungibile dal dispenser.
14    finecorsa_alto: BOOL := true;
15    // Sensore che indica il punto più basso raggiungibile dal dispenser.
16    finecorsa_basso: BOOL := false;
17    // Azione che attiva il flag SFCPause. Porta il macchinario in uno stato di pausa dopo aver esaurito le vaschette al suo interno.
18    sospendi_macchina: BOOL;
19    // Spia che diventa vera se il macchinario va in sospensione a causa del termine delle vaschette.
20    // Si resetta automaticamente dopo il riavvio automatico che segue l'avvenuto caricamento delle vaschette nel dispenser.
21    spia: BOOL := false;
22 END_VAR
23 VAR_INPUT
24     // Variabile di input che, se vera, porterà l'SFC ad evolvere verso gli stati della gestione delle emergenze.
25     // Il suo valore verrà modificato esternamente a questo macchinario.
26     controllo_reset: BOOL;
27 END_VAR
28 VAR
29     // Questa azione agisce sul flag SFCInit, il quale porta il macchinario in uno stato di arresto.
30     ferma_macchina: BOOL;
31 END_VAR

```

Figura 10: variabili del solo dispenser vaschette

```

1 PROGRAM nastro_dispenser_vaschette
2 VAR
3     // Attuatore che mette in moto il nastro.
4     motore: BOOL;
5     // Questa azione agisce sul flag SFCPause, il quale se è vero mette in pausa l'esecuzione dei task del macchinario.
6     // Questa eccezione è gestita dal nastro dopo che è diventata vera la stessa variabile all'interno del macchinario collegato.
7     sospendi_macchina: BOOL;
8 END_VAR

```

Figura 11: variabili del nastro trasportatore associato

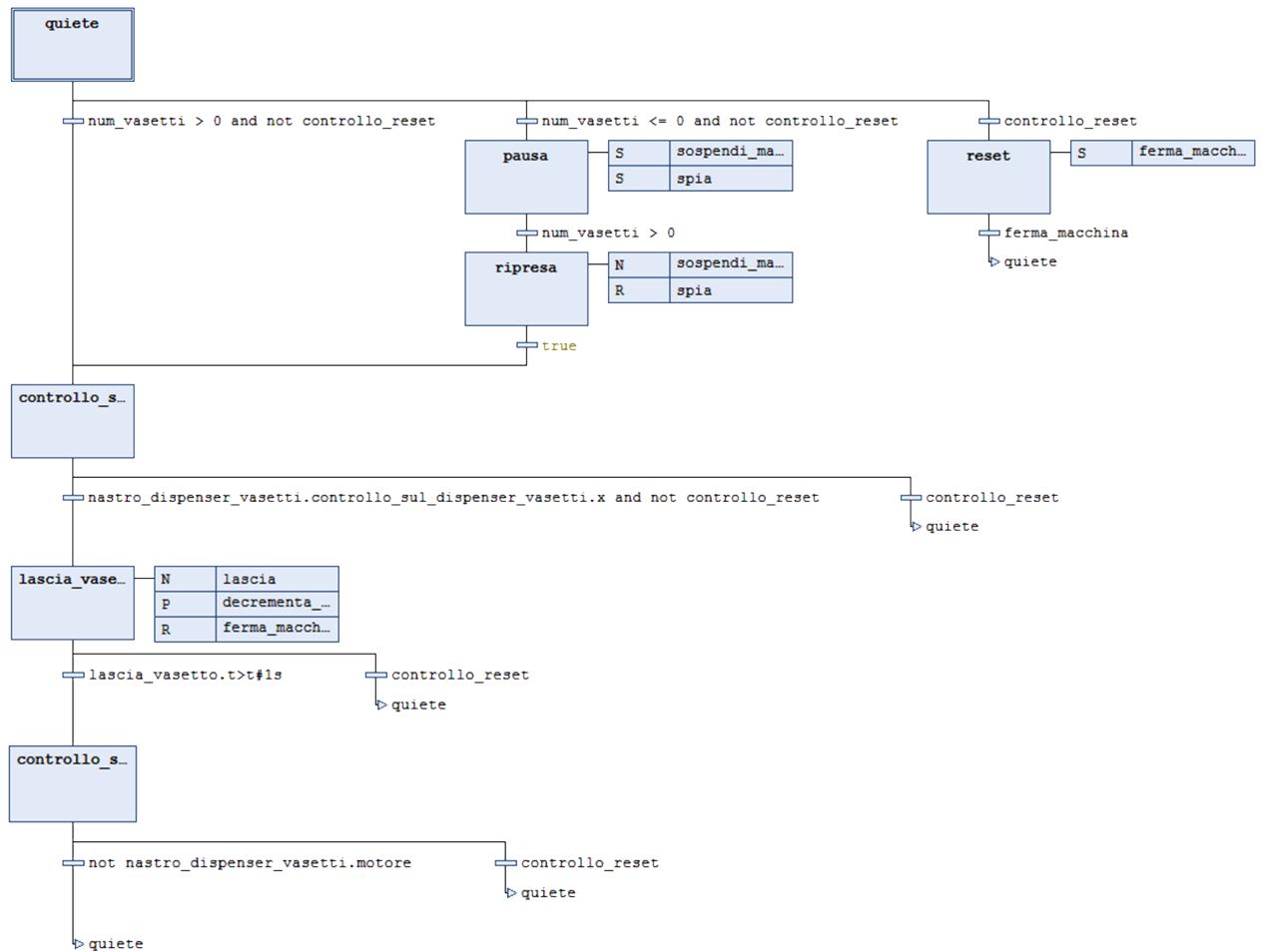


Figura 12: logica di controllo del dispenser vasetti

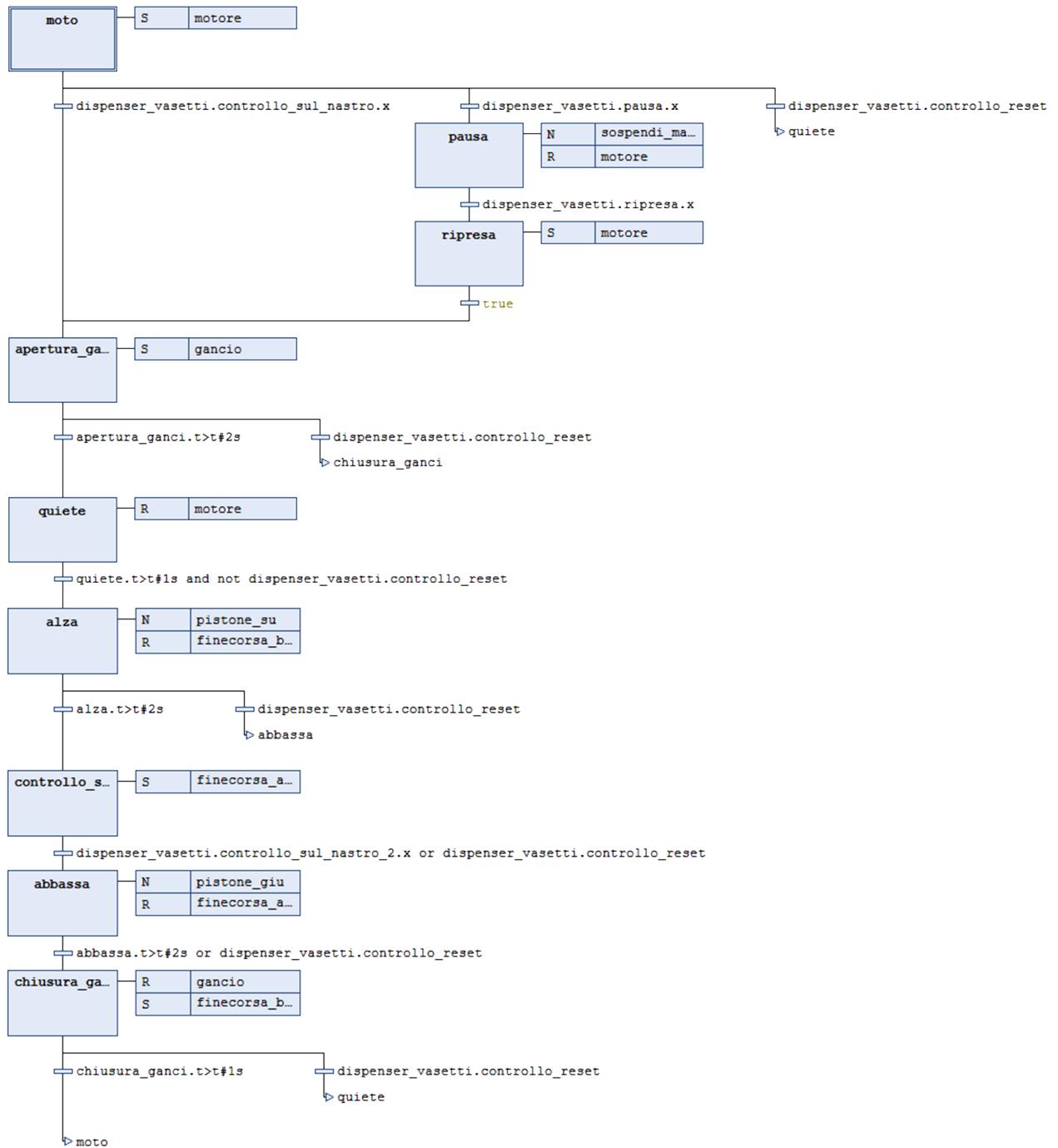


Figura 13: logica di controllo del nastro trasportatore associato

Il dispenser dei vasetti si trova inizialmente in uno stato di quiete. All'atto dell'avvio del macchinario, controlla che ci siano vasetti al suo interno per poter effettuare le operazioni e

che non sia vera la variabile controllo_reset, la quale lo porterebbe verso uno stato di emergenza. Se non ci sono vaschette all'interno, il macchinario accende una spia per segnalarlo; solo ad avvenuto caricamento potrà ripartire. Dopo aver effettuato il controllo, attende che il suo nastro abbia raggiunto il finecorsa alto e, se la variabile di controllo delle emergenze non è vera, allora rilascia i vasetti all'interno della vaschetta. Dopo aver depositato i vasetti, ritorna in uno stato di quiete.

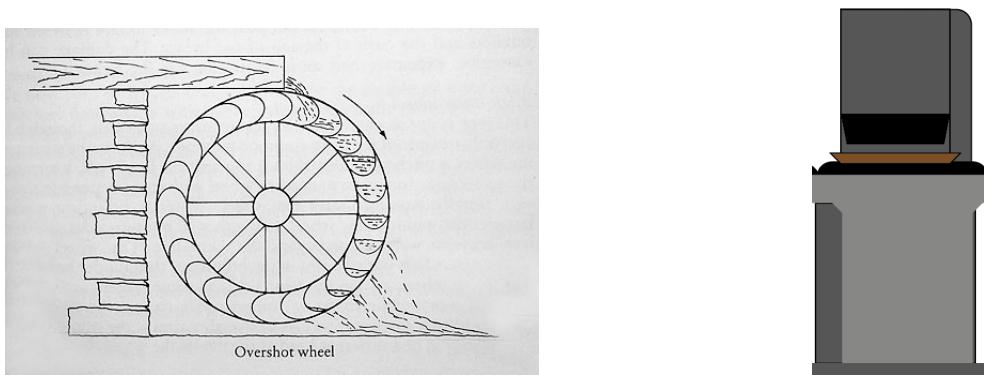
Il nastro trasportatore è inizialmente in moto, così da trarre a sé la vaschetta precedentemente depositata dal dispenser vaschette. Dopo aver controllato che il dispenser vasetti non sia in uno stato di pausa (che è diverso dallo stato di quiete) o emergenza, effettua le seguenti operazioni:

1. apre i ganci per ostacolare la corsa delle vaschette e posizionarle esattamente sotto al dispenser vasetti;
2. arresta il motore;
3. viene alzato dai pistoni fino al raggiungimento del finecorsa alto;
4. attende le operazioni di carico da parte del dispenser;
5. dopo che la vaschetta è stata caricata, l'intera struttura viene abbassata fino al raggiungimento del finecorsa basso;
6. vengono chiusi i ganci;
7. il nastro viene rimesso in moto.

Come si può notare nelle immagini 12 e 13, tra un'operazione e l'altra vi è **sempre** la possibilità che si attivi la transizione che porterebbe il macchinario nello stato di emergenza (variabile controllo_reset vera).

3.2.3 Macchina torbatrice

Generalità La macchina torbatrice è un macchinario che sfrutta un meccanismo simile a quello dei mulini ad acqua per riempire di torba i vasetti. Il meccanismo di funzionamento è molto semplice: quando il motore viene azionato, tramite un albero di trasmissione verrà messa in moto anche una catena. Essa azionerà il sistema di pale tramite cui la torba verrà prelevata dal serbatoio e fatta ricadere all'interno dei vasetti della vaschetta.



(a) schema semplificato di un mulino ad acqua

(b) particolare della macchina torbatrice

Figura 14: riferimenti per la macchina torbatrice

A differenza degli altri moduli dell'impianto, il nastro trasportatore associato alla torbatrice non ha uno schema di controllo proprio: esso sfrutta la stessa coppia motrice con la quale vengono messe in moto le pale, garantendo sempre un funzionamento perfettamente sincronizzato con quelli che sono i tempi di lavoro della macchina stessa.

Logica di controllo L'immagine 15 riporta il codice inerente le variabili della macchina torbatrice.

```

1 PROGRAM torbatrice
2 VAR
3     // Variabile che deve essere vera per poter avviare il macchinario
4     controllo_sicurezza_avvio: BOOL;
5     // Sensore che segnala se il serbatoio è carico.
6     carico: BOOL := TRUE;
7     // Sensore che segnala se il serbatoio è scarico.
8     scarico: BOOL;
9     // Motore che aziona sia il meccanismo a pale per il movimento della torba sia il nastro trasportatore collegato al macchinario.
10    motore: BOOL := TRUE;
11    // Questa azione agisce sul flag SFCPause, il quale se è vero mette in pausa l'esecuzione dei task del macchinario.
12    // Questa eccezione è gestita dal macchinario stesso in seguito alla mancanza di vaschette da sganciare.
13    sospendi_macchina: BOOL := FALSE;
14    // Spia che diventa vera se il macchinario va in sospensione a causa dello svuotamento del serbatoio.
15    // Si resetta automaticamente dopo il riovvio automatico che segue l'avvenuta ricarica del serbatoio.
16    spia: BOOL := FALSE;
17 END_VAR
18 VAR_INPUT
19     // Controllo che comunica al sistema un'emergenza. Se è vero, porta l'SFC ad evolvere verso i passi di gestione delle emergenze.
20     controllo_reset: BOOL;
21 END_VAR
22
23 VAR
24     // Se controllo_reset è vero, porta la macchina in uno stato di arresto settando su vero il flag SFCInit.
25     ferma_macchina: BOOL;
26 END_VAR

```

Figura 15: variabili della macchina torbatrice

Nell'immagine 16 è riportata l'implementazione della logica della macchina torbatrice.

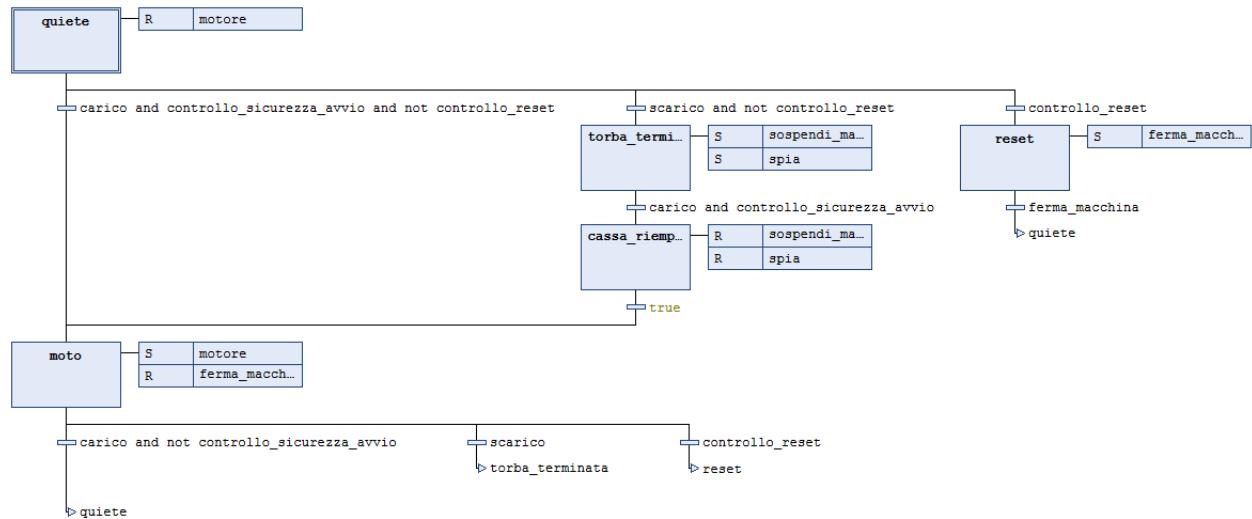


Figura 16: logica di controllo della macchina torbatrice

La macchina torbatrice è inizialmente in quiete per poter effettuare le operazioni di controllo:

1. se è vera la variabile controllo_reset, allora si porta in stato di emergenza;

2. se la variabile di reset è falsa ma il serbatoio è scarico, il macchinario accende la spia di segnalazione e resta in pausa finché non viene ricaricato;
3. se il serbatoio è carico e la variabile di reset è falsa, viene acceso il motore e la macchina inizia sia il prelievo della torba che il movimento del nastro;
4. il motore si ferma se:
 - la variabile di controllo di sicurezza all'avvio diventa falsa;
 - non c'è più torba nel serbatoio;
 - diventa vera la variabile di reset.

3.2.4 Blocco 3

Generalità I macchinari del blocco tre sono a due a due identici nel funzionamento, a meno di qualche differenza di notazione: per tale motivo, raggrupperemo il modulo pressa insieme al modulo foratrice ed il modulo seminatrice insieme al modulo riempitrice.

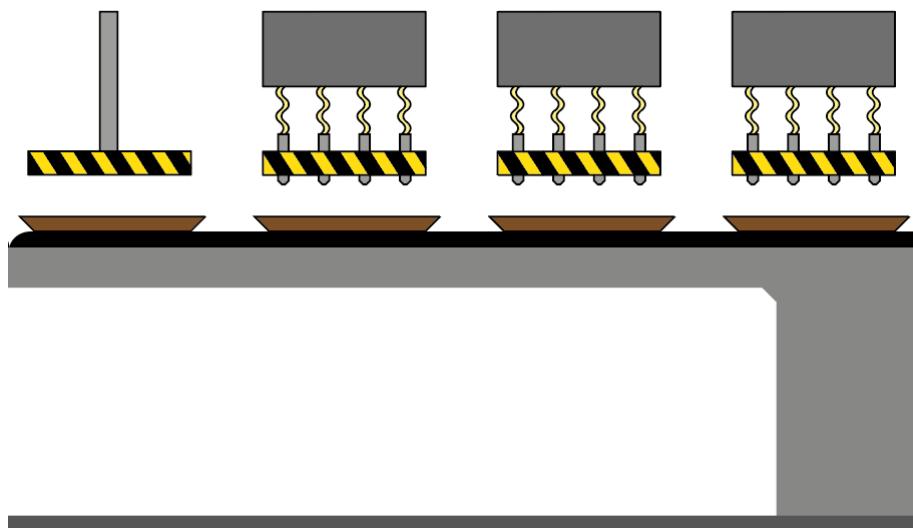


Figura 17: particolare del blocco 3, da sinistra a destra: pressa, foratrice, seminatrice, riempitrice

Quest'intero blocco effettua le seguenti operazioni:

1. pressa la terra inserita nei vasetti, rendendola uniforme e compatta;
2. pratica un foro al centro dei vasetti;
3. inserisce un seme all'interno del foro;
4. copre il seme con la torba necessaria.

Logica di controllo Dalla figura 18 alla figura 22 sono riportate le dichiarazioni delle variabili dei singoli moduli che compongono il blocco 3; dalla figura 23 alla figura 27 sono riportati gli schemi di controllo dei moduli del blocco 3.

```

1 PROGRAM pressa
2 VAR
3   // Motore che alza il macchinario.
4   motore_su: BOOL;
5   // Motore che abbassa il macchinario.
6   motore_giu: BOOL;
7   // Sensore che segnala il punto più alto raggiungibile dal macchinario.
8   finecorsa_alto: BOOL;
9   // Sensore che segnala il punto più basso raggiungibile dal macchinario.
10  finecorsa_basso: BOOL;
11 END_VAR
12 VAR_INPUT
13   // Controllo che comunica al sistema un'emergenza. Se è vero, porta l'SFC ad evolvere verso i passi di gestione delle emergenze.
14   controllo_reset: BOOL;
15 END_VAR
16 VAR
17   // Se controllo_reset è vero, porta la macchina in uno stato di arresto settando su vero il flag SFCInit.
18   ferma_macchina: BOOL;
19 END_VAR

```

Figura 18: variabili del modulo pressa

```

1 PROGRAM foratrice
2 VAR
3   // Motore che alza il macchinario.
4   motore_su: BOOL;
5   // Motore che abbassa il macchinario.
6   motore_giu: BOOL;
7   // Sensore che segnala il punto più alto raggiungibile dal macchinario.
8   finecorsa_alto: BOOL;
9   // Sensore che segnala il punto più basso raggiungibile dal macchinario.
10  finecorsa_basso: BOOL;
11 END_VAR
12 VAR_INPUT
13   // Controllo che comunica al sistema un'emergenza. Se è vero, porta l'SFC ad evolvere verso i passi di gestione delle emergenze.
14   controllo_reset: BOOL;
15 END_VAR
16 VAR
17   // Se controllo_reset è vero, porta la macchina in uno stato di arresto settando su vero il flag SFCInit.
18   ferma_macchina: BOOL;
19 END_VAR

```

Figura 19: variabili del modulo foratrice

```

1 PROGRAM seminatrice
2 VAR
3     // Sensore che segnala se il serbatoio è carico.
4     carico: BOOL := TRUE;
5     // Sensore che segnala se il serbatoio è scarico.
6     scarico: BOOL;
7     // Motore che alza il macchinario.
8     motore_su: BOOL;
9     // Motore che abbassa il macchinario.
10    motore_giu: BOOL;
11    // Sensore che indica il punto più alto raggiungibile dal macchinario.
12    finecorsa_alto: BOOL;
13    // Sensore che indica il punto più basso raggiungibile dal macchinario.
14    finecorsa_basso: BOOL;
15    // Vano semi che si apre e si chiude facendo cadere solo su richiesta i semi nei vasetti.
16    vano_semi: BOOL;
17    // Questa azione agisce sul flag SFCPause, il quale se è vero mette in pausa l'esecuzione dei task del macchinario.
18    // Questa eccezione è gestita dal macchinario stesso in seguito alla mancanza di vaschette da sganciare.
19    sospendi_macchina: BOOL;
20    // Spia che diventa vera se il macchinario va in sospensione a causa dello svuotamento del serbatoio.
21    // Si resetta automaticamente dopo il riavvio automatico che segue l'avvenuta ricarica del serbatoio.
22    spia: BOOL;
23 END_VAR
24 VAR_INPUT
25     // Controllo che comunica al sistema un'emergenza. Se è vero, porta l'SFC ad evolvere verso i passi di gestione delle emergenze.
26     controllo_reset: BOOL;
27 END_VAR
28 VAR
29     // Se controllo_reset è vero, porta la macchina in uno stato di arresto settando su vero il flag SFCInit.
30     ferma_macchina: BOOL;
31 END_VAR

```

Figura 20: variabili del modulo seminatrice

```

1 PROGRAM riempitrice
2 VAR
3     // Sensore che segnala se il serbatoio è carico.
4     carico: BOOL := TRUE;
5     // Sensore che segnala se il serbatoio è scarico.
6     scarico: BOOL;
7     // Motore che alza il macchinario.
8     motore_su: BOOL;
9     // Motore che abbassa il macchinario.
10    motore_giu: BOOL;
11    // Sensore che indica il punto più alto raggiungibile dal macchinario.
12    finecorsa_alto: BOOL;
13    // Sensore che indica il punto più basso raggiungibile dal macchinario.
14    finecorsa_basso: BOOL;
15    // Vano torba che si apre e si chiude facendo cadere solo su richiesta la torba nei vasetti.
16    vano_torba: BOOL;
17    // Questa azione agisce sul flag SFCPause, il quale se è vero mette in pausa l'esecuzione dei task del macchinario.
18    // Questa eccezione è gestita dal macchinario stesso in seguito alla mancanza di vaschette da sganciare.
19    sospendi_macchina: BOOL;
20    // Spia che diventa vera se il macchinario va in sospensione a causa del termine delle vaschette.
21    // Si resetta automaticamente dopo il riavvio automatico che segue l'avvenuto caricamento delle vaschette nel dispenser
22    spia: BOOL;
23 END_VAR
24 VAR_INPUT
25     // Controllo che comunica al sistema un'emergenza. Se è vero, porta l'SFC ad evolvere verso i passi di gestione delle emergenze.
26     controllo_reset: BOOL;
27 END_VAR
28 VAR
29     // Se controllo_reset è vero, porta la macchina in uno stato di arresto settando su vero il flag SFCInit.
30     ferma_macchina: BOOL;
31 END_VAR

```

Figura 21: variabili del modulo riempitrice

```

1 PROGRAM nastro_blocco3
2 VAR
3     // Ganci che fungono da barriere per le vaschette posti sotto al macchinario associato.
4     gancio_riempitrice: BOOL;
5     // Ganci che fungono da barriere per le vaschette posti sotto al macchinario associato.
6     gancio_seminatrice: BOOL;
7     // Ganci che fungono da barriere per le vaschette posti sotto al macchinario associato.
8     gancio_foratrice: BOOL;
9     // Ganci che fungono da barriere per le vaschette posti sotto al macchinario associato.
10    gancio_pressa: BOOL;
11    // Attuatore che mette in moto il nastro.
12    motore_nastro: BOOL := TRUE;
13 END VAR

```

Figura 22: variabili del nastro trasportatore associato all'intero blocco

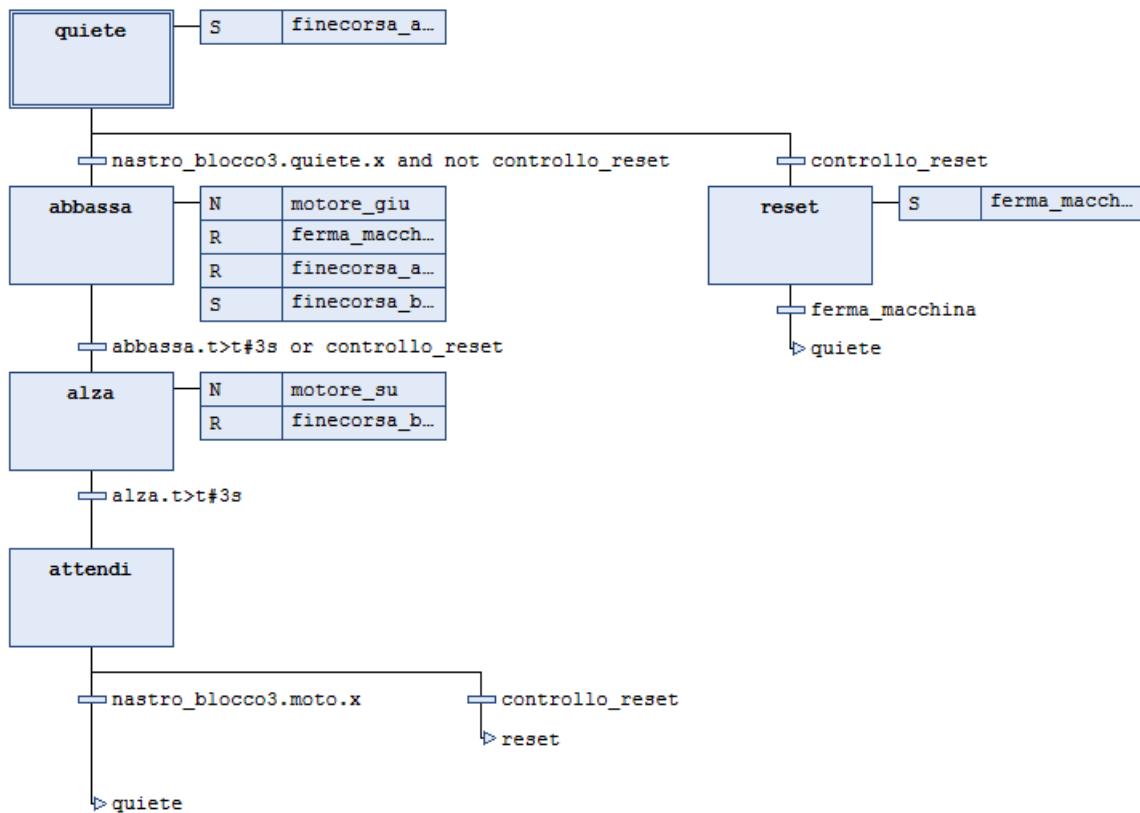


Figura 23: logica di controllo della foratrice

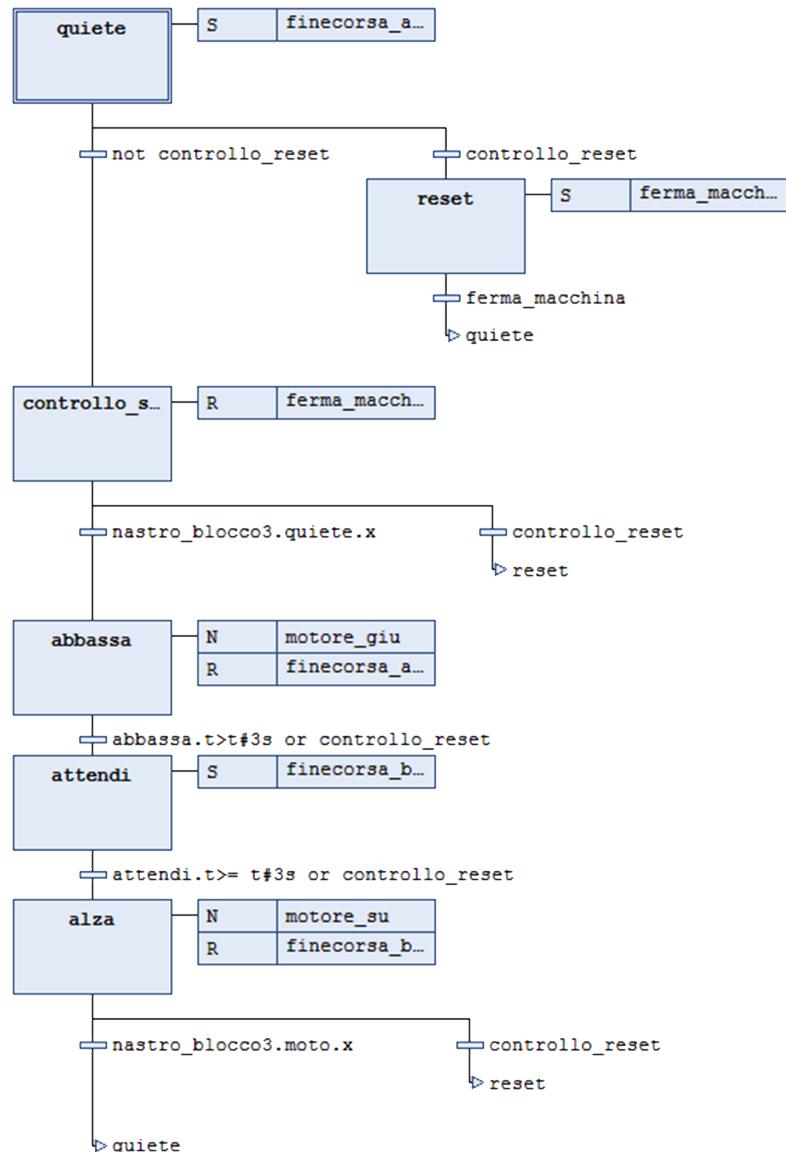


Figura 24: logica di controllo della pressa

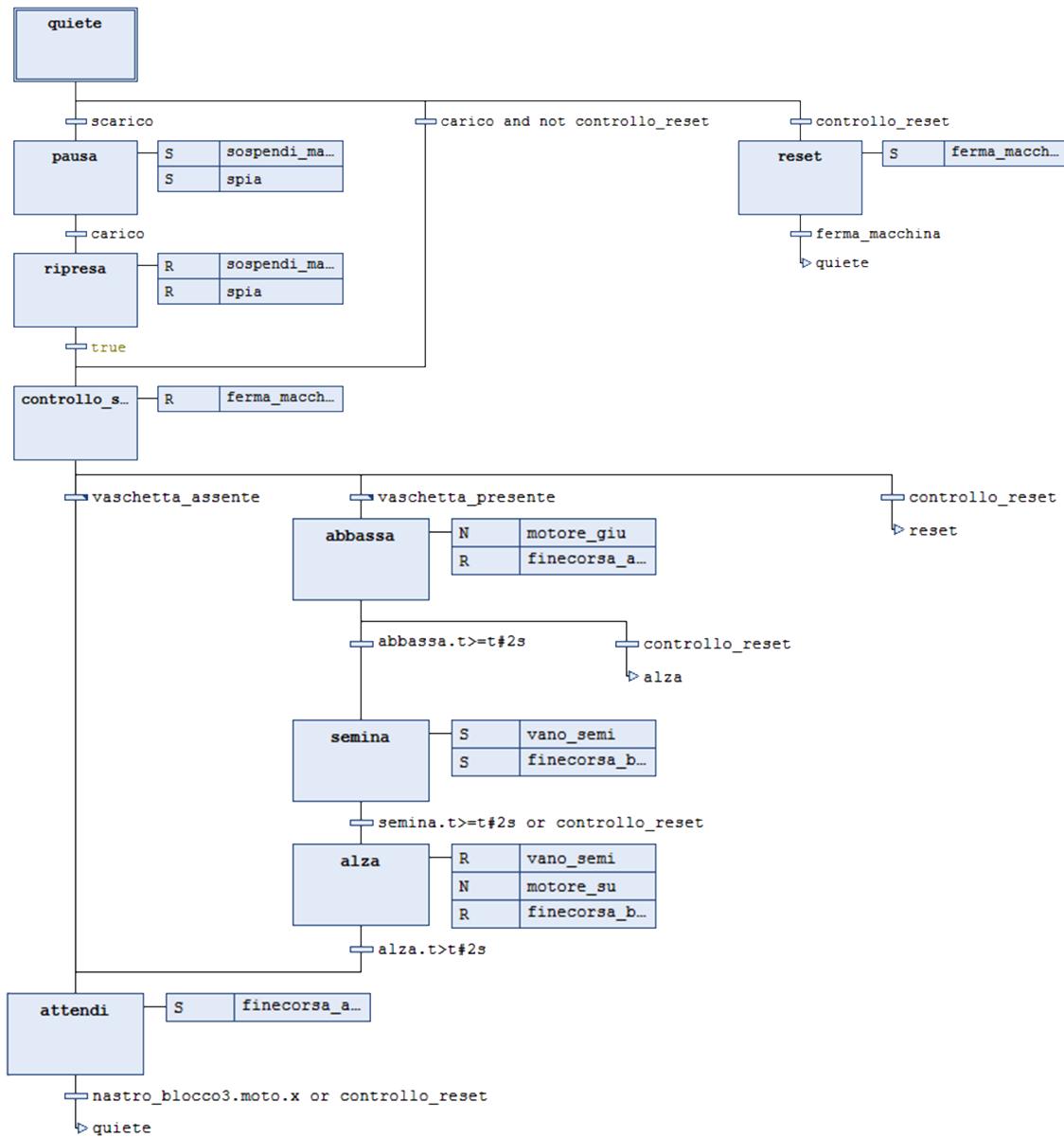


Figura 25: logica di controllo della seminatrice

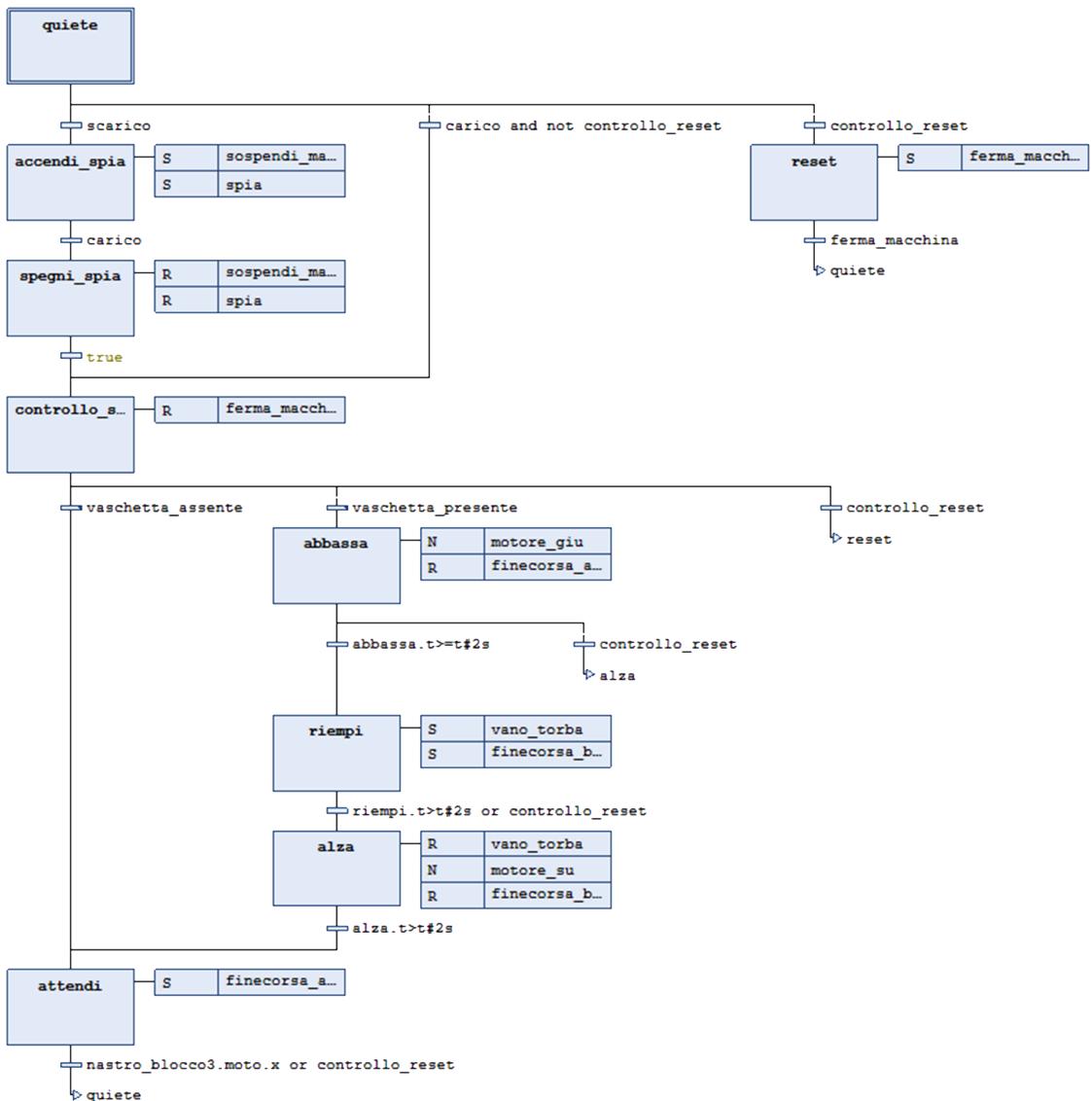


Figura 26: logica di controllo della riempitrice

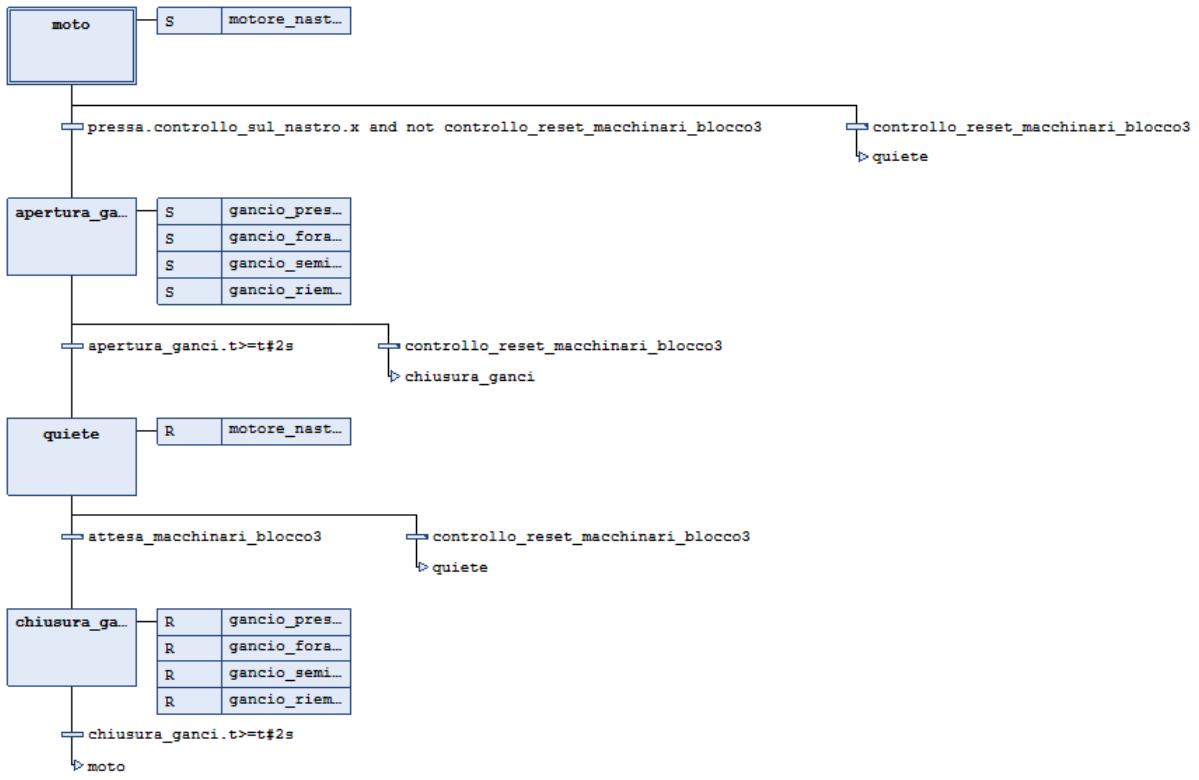


Figura 27: logica di controllo del nastro trasportatore associato al blocco 3

La pressa effettua le seguenti operazioni:

1. inizialmente è in quiete al finecorsa alto;
2. controlla se la variabile di reset è vera: in caso positivo, il controllo evolve verso uno stato di emergenza; in caso negativo, prosegue normalmente nel diagramma;
3. attende che il nastro blocchi sotto di essa la vaschetta proveniente dalla torbatrice;
4. con la vaschetta esattamente al di sotto, viene azionato il motore finché non raggiunge il finecorsa basso e vi resta per tre secondi: il finecorsa basso permette alla pressa di venire a contatto con la vaschetta;
5. terminato il livellamento, viene attivato il motore finché non viene raggiunto il finecorsa alto; nel frattempo il nastro riprende a muoversi.

La foratrice effettua le stesse operazioni della pressa, con la differenza che inizia la risalita subito dopo aver raggiunto il finecorsa basso.

La seminatrice effettua le seguenti operazioni:

1. inizialmente è in uno stato di quiete al finecorsa alto;
2. effettua le solite operazioni di controllo sulla pienezza del serbatoio e sulla variabile di reset: se è tutto in regola, procede nel diagramma;
3. a differenza degli altri macchinari (nello specifico ci si riferisce ai dispenser), i macchinari del blocco 3 lavorano con un unico nastro, perciò i tempi di lavorazione sono gli stessi e i controlli dei macchinari sul nastro sono gli stessi per tutti. Per questo motivo, è sorta l'esigenza di prevedere un ramo di funzionamento "*a vuoto*":
 - se sotto al macchinario è presente la vaschetta, abbasso la macchina fino alla vaschetta (cioè il suo finecorsa basso), rilascio un seme nel vasetto e la rialzo fino al suo finecorsa alto;
 - se sotto al macchinario non è presente alcuna vaschetta, attendo semplicemente il controllo sul nastro senza alcuna operazione;
4. dopo l'eventuale operazione di semina, vi è una fase di attesa per permettere al gancio di trasportare ai macchinari successivi le vaschette lavorate e di far arrivare le vaschette ancora da lavorare.

La riempitrice effettua le stesse operazioni della seminatrice, con l'unica differenza che non rilascerà semi all'interno dei fori nei vasetti ma ricoprirà questi ultimi con della torba.

Il nastro trasportatore del blocco 3 effettua le seguenti operazioni:

1. inizialmente è in uno stato di moto per far arrivare la vaschetta lavorata dalla torbatrice;

2. essendo la pressa il primo macchinario del blocco, è stata designata come "sentinella" preposta all'individuazione della vaschetta uscita dalla torbatrice;
3. il nastro viene notificato dalla pressa dell'arrivo della vaschetta ed apre i ganci per bloccare la vaschetta sotto ogni macchinario del blocco 3, proprio come i ganci del nastro trasportatore sottostante il dispenser vasetti;
4. una volta fermate le vaschette, viene spento il motore e il nastro si porta in uno stato di quiete;
5. il nastro attende le operazioni di lavorazione di ciascun macchinario;
6. al termine di tutte le operazioni, il nastro chiude i ganci;
7. viene rimesso in moto il nastro.

Alcuni chiarimenti Per rendere il codice più leggibile, alcune transizioni sono state accorpate e rinominate come mostrato nelle figure che seguono.

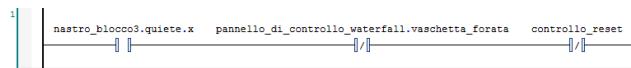


Figura 28: transizione in caso di assenza della vaschetta



Figura 29: transizione in caso di presenza della vaschetta

Le transizioni vaschetta_assente e vaschetta_presente per la seminatrice e per la riempitrice sono uguali, a meno della variabile vaschetta seminata e vaschetta forata. Dunque, verrà riportata solo un'immagine per ciascuna transizione.

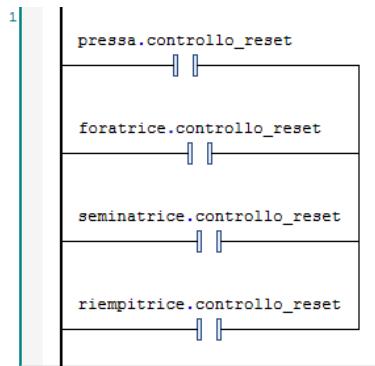


Figura 30: controllo del nastro sulle variabili di reset di ogni singolo macchinario; è necessaria che sia vera una sola variabile per arrestare l'intero ciclo

3.2.5 Blocco per l'irrigazione

Generalità Questo blocco è preposto alla corretta irrigazione delle vaschette, così da prepararle alla permanenza all'interno della sala di germinazione. Il blocco è formato da un modulo di irrigazione e dal suo nastro sottostante.

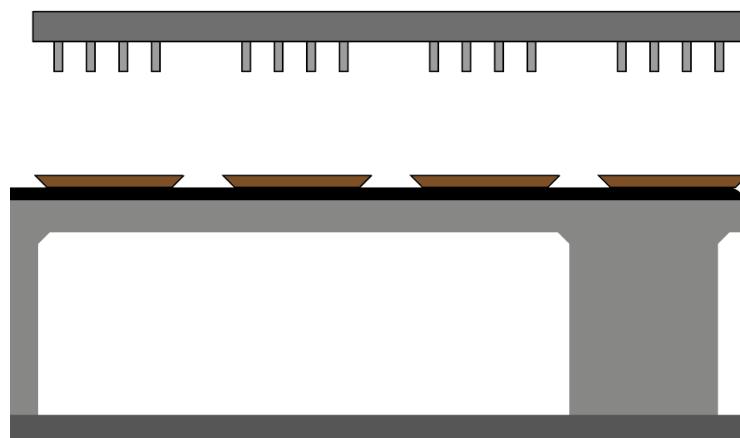


Figura 31: particolare del blocco di irrigazione

Il blocco effettua un'operazione molto semplice: controlla che ci siano vaschette e in caso affermativo apre gli irrigatori mentre il nastro è in moto e fa avanzare le vaschette.

Logica di controllo Nelle figure 32 e 33 sono riportate le variabili del modulo irrigatore e del nastro trasportatore associato; nelle figure 34 e 35 sono riportati i rispettivi schemi della logica di controllo.

```

1 PROGRAM irrigatore
2 VAR
3     // Variabile che deve essere vera per poter avviare il macchinario
4     controllo_sicurezza_avvio: BOOL;
5     // Sensore che segnala se il serbatoio è carico.
6     carico: BOOL := true;
7     // Sensore che segnala se il serbatoio è scarico.
8     scarico: BOOL;
9     // Variabile che aziona gli irrigatori su richiesta.
10    irrigatori: BOOL;
11    // Questa azione agisce sul flag SFCPause, il quale se è vero mette in pausa l'esecuzione dei task del macchinario.
12    // Questa eccezione è gestita dal macchinario stesso in seguito alla mancanza di vaschette da sganciare.
13    sospendi_macchina: BOOL;
14    // Spia che diventa vera se il macchinario va in sospensione a causa dello svuotamento del serbatoio.
15    // Si resetta automaticamente dopo il riavvio automatico che segue l'avvenuta ricarica del serbatoio.
16    spia: BOOL;
17 END_VAR
18 VAR_INPUT
19     // Controllo che comunica al sistema un'emergenza. Se è vero, porta l'SFC ad evolvere verso i passi di gestione delle emergenze.
20     controllo_reset: BOOL;
21 END_VAR
22 VAR
23     // Se controllo_reset è vero, porta la macchina in uno stato di arresto settando su vero il flag SFCInit.
24     ferma_macchina: BOOL;
25 END_VAR

```

Figura 32: variabili del modulo di irrigazione

```

1 PROGRAM nastro_irrigatore
2 VAR
3     // Attuatore che mette in moto il nastro.
4     motore: BOOL;
5     // Azione che attiva il flag SFCPause. Diventa vera se è vera la stessa variabile anche nel macchinario associato.
6     sospendi_macchina: BOOL;
7 END_VAR

```

Figura 33: variabili del nastro associato

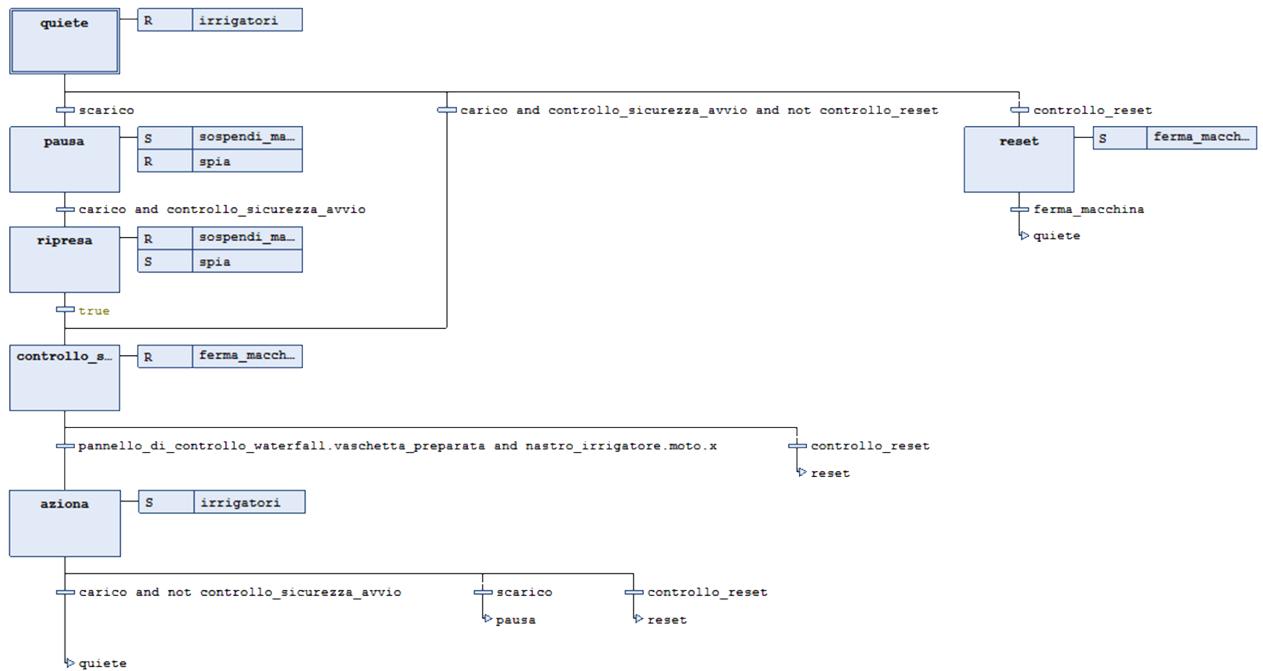


Figura 34: logica di controllo del modulo di irrigazione

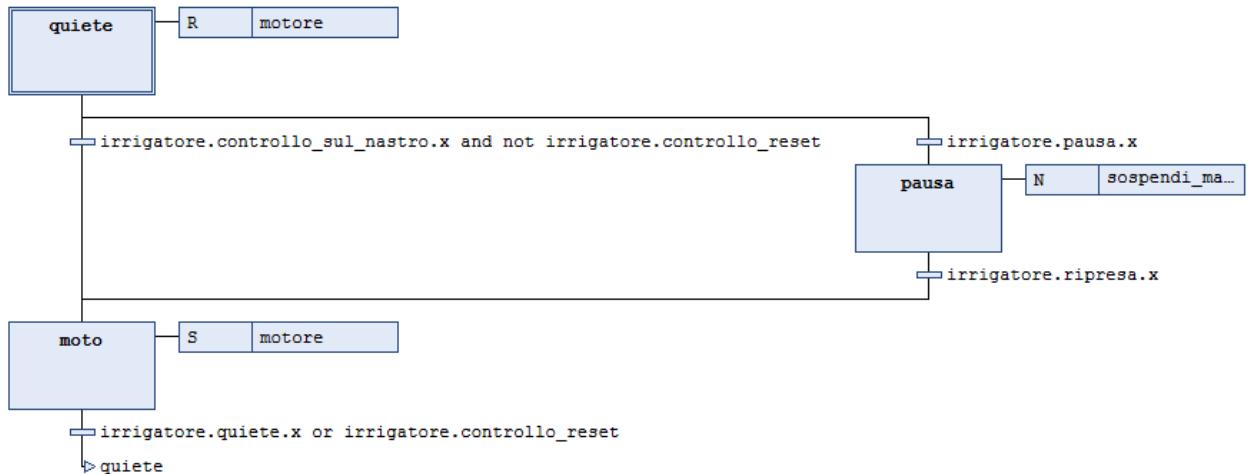


Figura 35: logica di controllo del nastro trasportatore

Il modulo di irrigazione effettua le seguenti operazioni:

1. deve essere vera la variabile di controllo per l'avvio, misura di sicurezza identica a quella della macchina torbatrice;

2. controlla che il serbatoio sia pieno e che non sia vera la variabile di reset;
3. attende che il nastro sia in moto e che ci sia una vaschetta in transito dal blocco 3;
4. aziona gli irrigatori;
5. se diventa falsa la variabile del controllo di sicurezza all'avvio, si scarica il serbatoio o diventa vera la variabile di reset, il macchinario si porta rispettivamente in stato di quiete, pausa o emergenza.

Il nastro trasportatore associato effettua le seguenti operazioni:

1. inizialmente si trova in uno stato di quiete in cui controlla che gli irrigatori non siano in attesa di essere ricaricati o in stato di emergenza;
2. dopo questi controlli, il nastro si mette in moto e vi resta finché gli irrigatori non entrano in uno stato di quiete, sospensione o emergenza.

3.3 Logica di simulazione

La logica di simulazione non verrà allegata a questa documentazione perché vi è l'impossibilità di apportare screen di qualità sufficientemente adeguata. Perciò si rimanda tutto al codice presente nella cartella del progetto.

Concettualmente invece, la logica di simulazione è stata pensata così:

- **pannello di controllo:** un programma in ladder che fornisce un input di avvio al primo macchinario (il dispenser vaschette). Successivamente, l'output di ciascun macchinario sarà l'output del macchinario che lo segue;
- **soft stop:** un programma in ladder che individua eventuale soft stop di un macchinario (lo stato *pausa* in seguito al passaggio a vero della variabile *scarico*) e mette in pausa a cascata tutti i macchinari che lo precedono;

- **hard reset:** un programma in ladder che serve per l'arresto di emergenza di un macchinario e, a cascata, di tutti i macchinari che lo precedono.

La differenza tra il programma di soft stop e quello di hard reset è che:

- nel primo caso, le variabili che portano alla pausa in cascata dei macchinari, sono interne agli stessi e vengono settate automaticamente dall'evoluzione del processo;
- nel secondo caso, sarà un supervisore ad attivare le variabili per l'arresto di emergenza.

4 Regolazione della temperatura nella germination room

In questo capitolo verrà affrontata l'implementazione del regolatore che controllerà la temperatura all'interno della sala di germinazione. Prima di scendere nei dettagli della realizzazione andranno fatte alcune premesse sulle condizioni della sala stessa, poichè saranno utili per lo studio e l'analisi delle equazioni del sistema che caratterizzeranno il nostro regolatore.

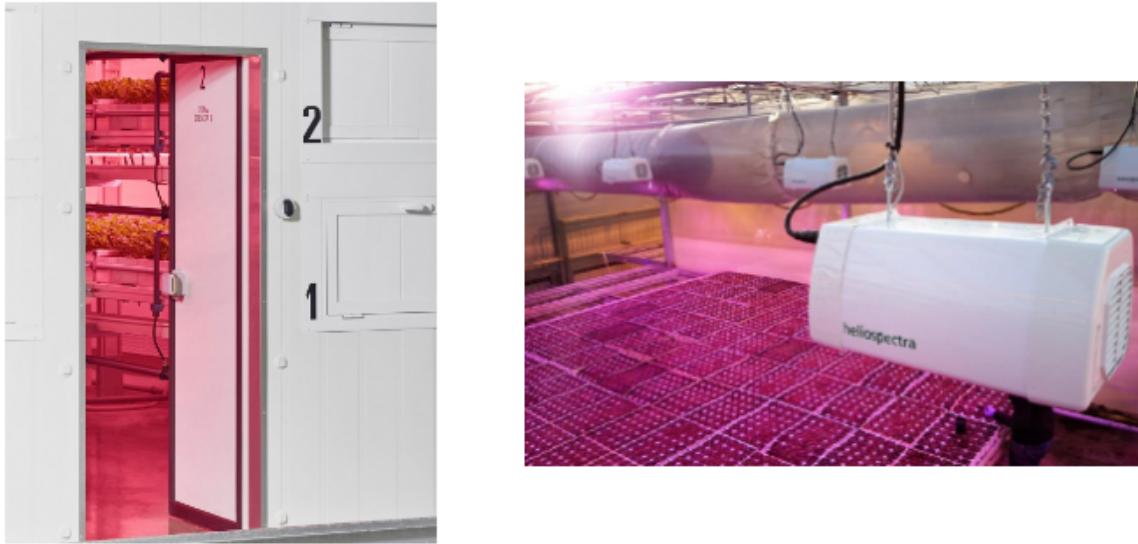


Figura 36: Entrata germination room e vista dall'alto [1]

La **germination room** sarà una stanza con pianta rettangolare di dimensioni $280m^2$ e avrà tali proprietà:

1. le mura divisorie saranno fatte in cemento e avranno una conducibilità termica k ;
2. la sua temperatura interna sarà T e la sua capacità termica C ;
3. all'esterno della stanza consideriamo una temperatura pari a quella dell'edificio T_b per semplicità;

4.1 Modello

Nel caso trattato, l'impianto a cui farà riferimento il nostro controllore sarà proprio la germination room. Quest'ultima è una semplice stanza chiusa di cui si vuole regolare la temperatura. L'equazione differenziale lineare a coefficienti costanti per ricavare il modello ISU è quella standard per un sistema che scambia calore con l'ambiente esterno.[6]

Partiamo quindi da:

$$CT = q - k(T - T_a)$$

A questa equazione andranno fatte alcune correzioni, in quanto la germination room non è posta direttamente all'esterno ma sarà un modulo presente all'interno dell'edificio, nello specifico posto nel piano che contiene anche l'impianto di germinazione. Quindi T_a sarà sostituito con T_b , temperatura all'interno della sala contenente l'impianto alla preparazione alla fase di germinazione (*building temperature*).



Figura 37: Considerazioni sulle condizioni della germination room

L'equazione del sistema sarà data da:

$$C_{gr}\dot{T} = q - k_{con}(T - T_b)$$

Tenendo conto che:

1. entrambe le mura che separano l'impianto di germinazione dalla sala di germinazione sono fatte dello stesso materiale ed hanno lo stesso spessore;
2. La temperatura della stanza contenente l'impianto di germinazione T_b sarà considerata costante e a un valore di 18° C (291.15 K);
3. k_{con} e C_{gr} rispettivamente conducibilità termica e capacità termica della germination room e q quantità di calore necessaria per portare la temperatura a regime;

La temperatura T all'interno della sala di germinazione sarà portata al suo **valore di regime** (22° C - come riportato nella tabella in Figura 2) da dei condizionatori, per semplicità trascuriamo le variazioni di umidità dovute al funzionamento degli stessi. Per quanto scritto, lo stato, gli ingressi e l'uscita del nostro sistema (ISU) saranno dati da:

$$x_1 = T \quad , \quad u_1 = q, \quad u_2 = T_b$$

$$y = T$$

In particolare descriveremo attraverso le seguenti equazioni un sistema **LTI-MISO** (lineare tempo-invariante multiple input-single output) in quanto esso avrà due ingressi e un'unica uscita.

$$\begin{cases} \dot{x}_1 = \frac{1}{C}u_1 - \frac{K_C}{C}x_1 + \frac{K_C}{C}u_2 \\ y = x_1 \end{cases}$$

In forma matriciale:

$$A = \left[-\frac{K_C}{C} \right] \quad B = \left[\frac{1}{C} \quad \frac{K_C}{C} \right] \quad C = \left[1 \right] \quad D = \left[0 \quad 0 \right]$$

Tali passaggi sono stati implementati in MATLAB e attraverso essi siamo riusciti a ricavare la nostra funzione di trasferimento del sistema $G(s)$:

```

3 %
4 % Il nostro sistema può essere modellato come:
5 % C_gr * (dT/dt) = q - k_con(T - T_b)
6 %
7 %
8 % T è la nostra variabile da controllare e rappresenta la temperatura
9 % all'interno della germination room
10 %
11 %
12 % con questo comando definisco la variabile di Laplace
13 s = tf('s');
14 %
15 A = -k_con/C_gr;
16 B = [1/C_gr k_con/C_gr];
17 C = 1;
18 D = [0 0];
19 %
20 %
21 % autovalori matrice dinamica
22 disp("Autovalori della matrice dinamica A: ");
23 eig(A)
24 %
25 % funzione di trasferimento del processo
26 disp("F.d.T del processo: ");
27 G = tf(ss(A,B,C,D));
28 G = G(1) % Scelgo la prima funzione con primo input
29

```

4.2 Taratura con formule di inversione

Essendo noto il modello matematico del sistema, tra i vari metodi di taratura trattati durante il corso, abbiamo adottato la taratura con le formule d'inversione. È possibile utilizzare le formule di inversione per assegnare pulsazione di attraversamento e margine di fase desiderate alla funzione di anello aperto $L(\omega_i)$.

```

37 % Sintesi controllore con formule di inversione
38 - wg = 1; % pulsazione di attraversamento
39 - PM = 70; % margine di fase
40
41 % Calcolo modulo e fase del sistema a ciclo aperto a w=wg
42 - [M, P] = bode(G, wg)
43
44 - Mg = wg/M;
45 - phig = (PM-90-P)*pi/180;
46

```

Sappiamo che i parametri di margine di guadagno M_g e margine di fase φ_g sono due indicatori della robustezza del sistema, poichè quantificano quanto errore nella stima del guadagno e dei ritardi di fase di $L(\omega i)$, posso tollerare senza andare incontro a instabilità. È possibile tenere conto di questi due indicatori nella taratura del nostro regolatore PI.

Dalla teoria sappiamo che, una volta trovati i valori di φ_g e M_g possiamo ottenere i valori di:

$$\begin{cases} K_P = \frac{M_g \sin(\varphi_g)}{\omega_g} \\ T_I = \frac{\tan(\varphi_g)}{\omega_g} \\ K_I = \frac{K_P}{T_I} \end{cases}$$

E di conseguenza, l'espressione del nostro regolatore (con $s = \omega i$):

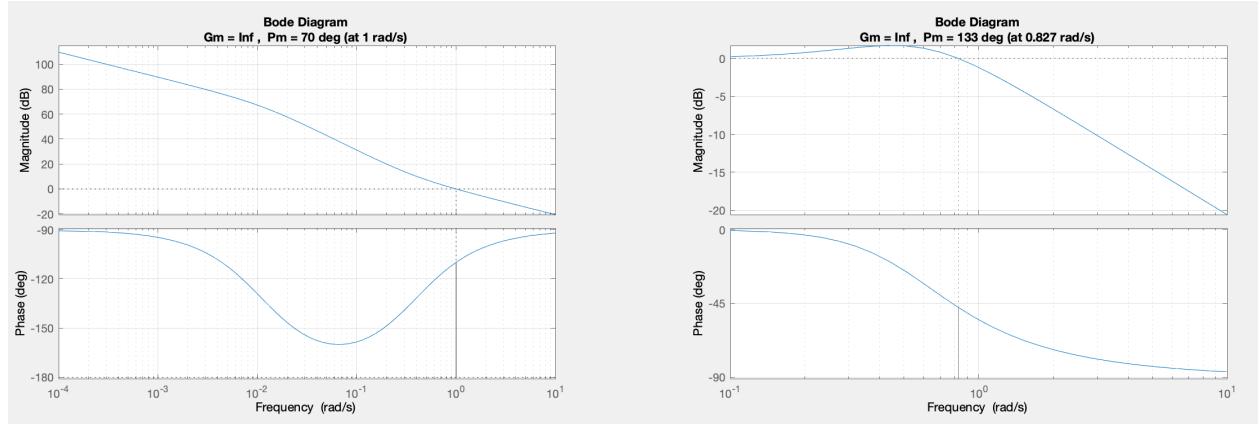
$$PI(s) = K_p s + \frac{K_I}{T_I s}$$

```

47 % Formule di inversione
48 - disp("Regolatore sintetizzato mediante formule di inversione: ");
49 - Kp_inv = Mg*sin(phig)/wg;
50 - Ti_inv = tan(phig)/wg;
51 - Ki_inv = Kp_inv/Ti_inv;
52
53 % otteniamo esattamente un PI
54 - C_inv = Kp_inv+Ki_inv/s

```

Trovata l'espressione del regolatore siamo capaci di calcolarci la nostra $L(\omega_i)$ e $F(\omega_i)$, rispettivamente funzione di anello aperto e chiuso del sistema. Con il comando *margin(sys)* plottiamo i diagrammi di Bode per i margini imposti a $L(\omega_i)$ e quelli relativi a $F(\omega_i)$.



4.2.1 Requisiti di controllo

Per la progettazione del regolatore è necessario definire dei requisiti che saranno valutati sulla risposta a gradino della funzione di anello chiuso del sistema $F(\omega_i)$: Per evincere tali caratteristiche utilizziamo il comando *stepinfo(sys)*:

```

71
72      % Display info risposta a gradino
73 -      Z = stepinfo(F)
74
75

```

Command Window

```

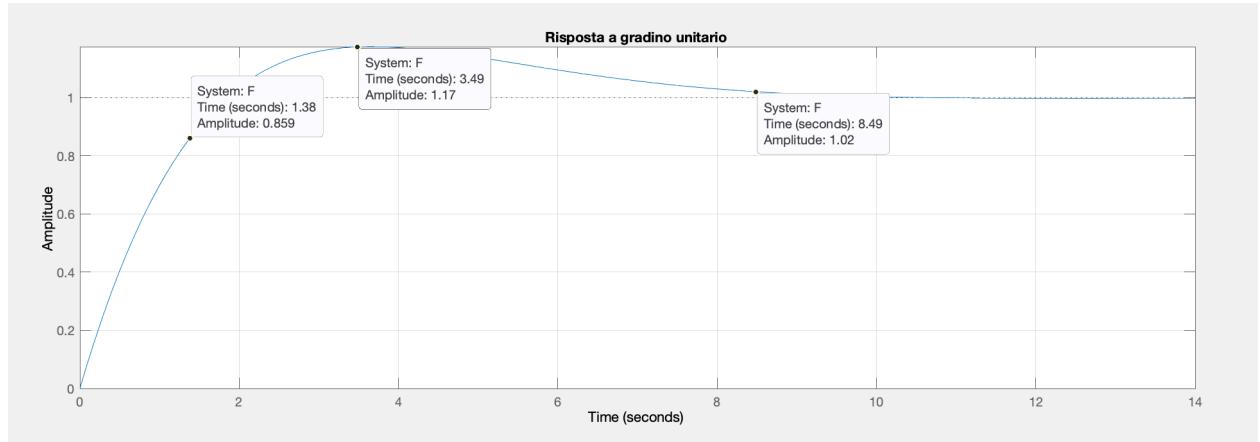
Z =

struct with fields:

    RiseTime: 1.3832
    SettlingTime: 8.4951
    SettlingMin: 0.9203
    SettlingMax: 1.1748
    Overshoot: 17.4843
    Undershoot: 0
    Peak: 1.1748
    PeakTime: 3.6944

```

- Tempo di salita** T_s : per tale parametro abbiamo rispettato il range di 2s, infatti notiamo che la risposta del sistema ad un impulso a gradino avrà T_s pari a 1.38s;
- Massima sovraelongazione** S : per quanto riguarda questo valore, abbiamo rispettato il range che ci siamo autoimposti del 25%, infatti notiamo che la nostra sovraelongazione è pari al 17%. Il motivo per il quale ci siamo imposti di rispettare una sovraelongazione minima del 25% è perchè non abbiamo previsto un'azione derivativa nel nostro regolatore che possa smorzare elevate sovraelongazioni;
- Tempo di assestamento** T_a : il range temporale per l'assestamento che ci siamo imposti di rispettare era compreso tra i 10-15s. Nel nostro caso il tempo di assestamento è pari a 8s, ciò vuol dire che il mio sistema raggiungerà il valore di regime intorno a quell'istante. Il risultato è soddisfacente, visto che non si ha necessità di raggiungere il valore di regime in modo molto rapido;



4.3 Simulink

Una volta completato lo script su MATLAB proseguiamo con la progettazione del sistema su Simulink, in un file separato "simulazione.slx". In figura 38 è mostrato lo schema del

sistema, in cui sono presenti:

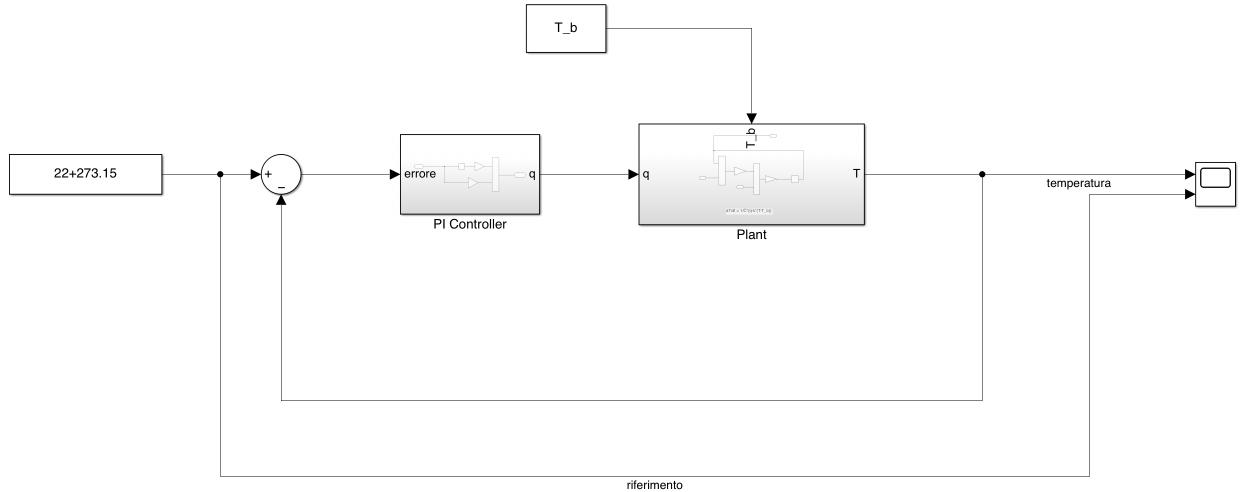


Figura 38: schema del sistema

1. il riferimento da inseguire $r(t)$ (22°C che in Kelvin sono equivalenti a $22+273.15\text{K}$) a cui verrà sottratta la retroazione dell'uscita; tale operazione ci fornirà l'errore $e(t)$, il quale rappresenta l'ingresso del nostro controllore;
2. regolatore PI, mostrato nello specifico in figura 39: l'uscita di quest'ultimo sarà data da q (quantità di calore) che deve essere fornita al sistema dall'attuatore (che non abbiamo riportato nello schema poiché considerato ideale con funzione di trasferimento pari a 1) per condizionare la temperatura se necessario;
3. impianto che raffigurerà il comportamento del nostro sistema dinamico (stanza di germinazione);
4. disturbo dell'impianto dato dalla temperatura della stanza contenente l'impianto di preparazione alla fase di germinazione (T_b);

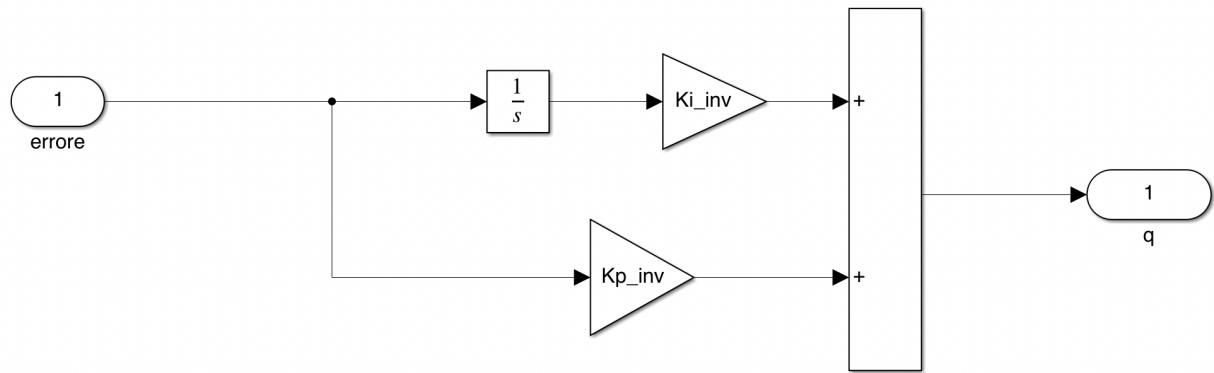
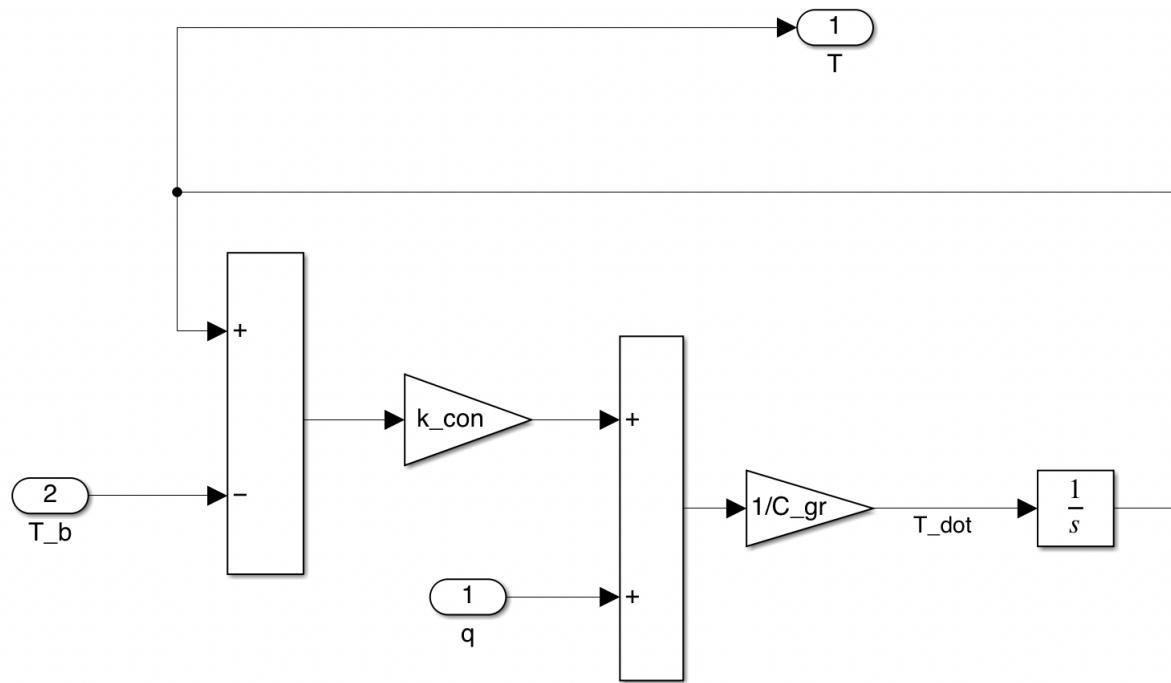


Figura 39: schema del controllore PI

In figura 39 abbiamo lo schema del PI con valori ricavati attraverso formule di inversione, avente:

1. In entrata l'errore $e(t)$;
2. Azione integrale e guadagno integrale K_I ;
3. Guadagno proporzionale K_P ;

K_P , T_I e K_I sommate daranno come risultato la nostra azione di controllo $u(t)$, che sarà fornita al nostro attuatore ideale. Infatti se supponiamo che il nostro attuatore sia il motore che alimenta le valvole di un condizionatore, q sarà proprio la quantità di energia termica fornita da esso verso il sistema per ottenere il valore di regime desiderato.



$$dT/dt = 1/C^*(q - k^*(T - T_b))$$

Figura 40: schema dell'impianto

In figura 40 abbiamo lo schema dell'impianto modellato attraverso equazioni differenziali: l'uscita sarà data proprio dalla temperatura T , variabile controllata del nostro sistema. Per finire, cliccando sullo scope riusciamo a visualizzare se la risposta del nostro sistema riesce ad inseguire il riferimento.

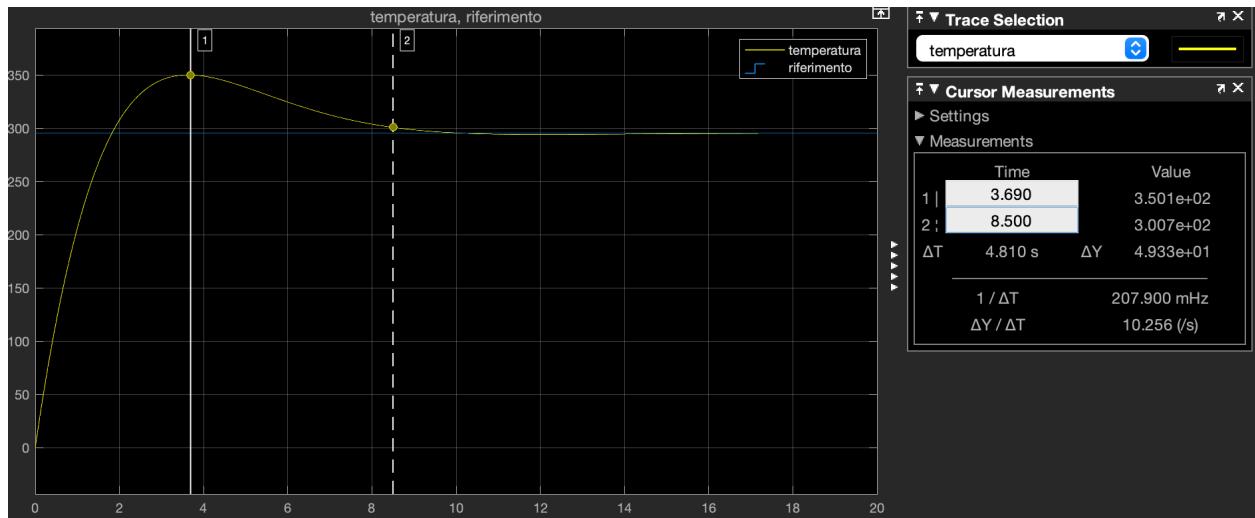


Figura 41: risposta del sistema

Stress Test Vogliamo verificare se il nostro sistema riesce a non perdere informazione durante casi di stress. Per riprodurre una situazione tale, non abbiamo fornito un riferimento costante $r(t)$ da far inseguire alla nostra variabile controllata, bensì abbiamo previsto un segnale di tipo onda quadra.

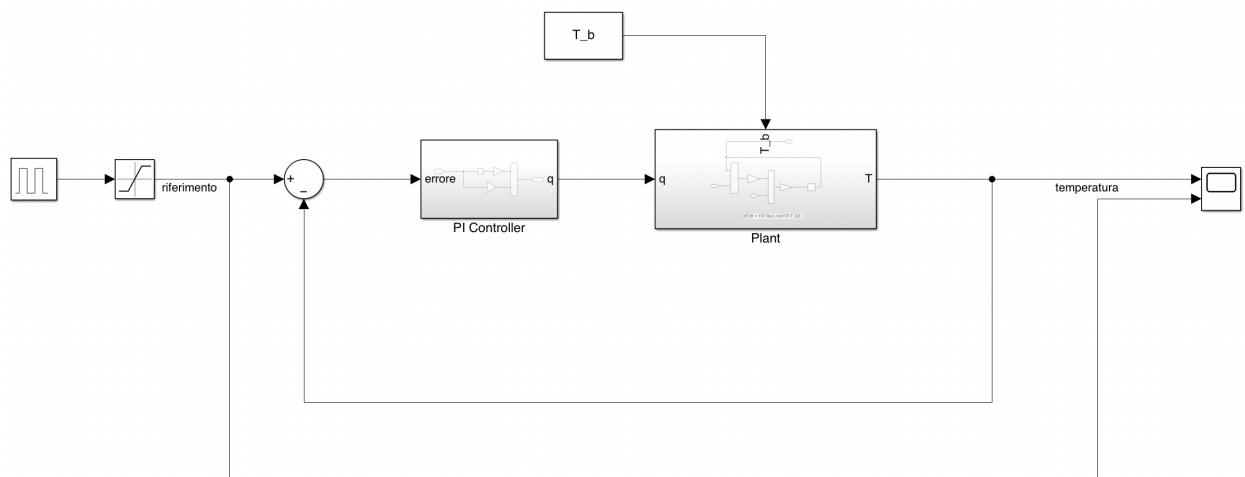


Figura 42: schema dell'impianto con un riferimento impulsivo

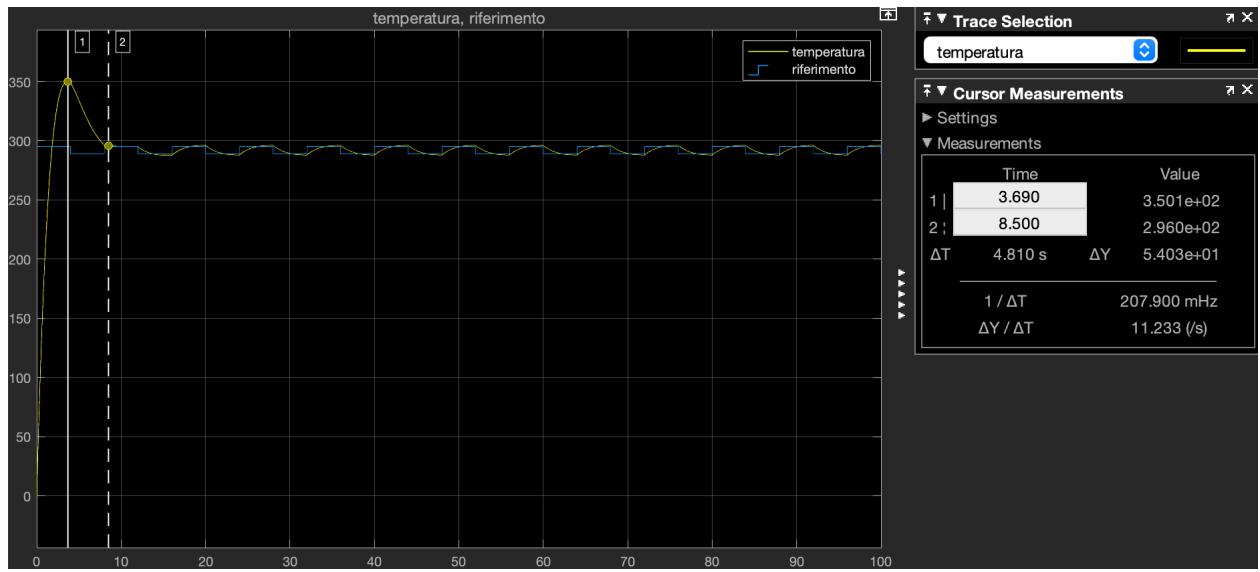


Figura 43: risposta relativa al caso di stress

Notiamo che la perdita di informazione non è significativa e che il nostro controllo in questa situazione si comporta in modo adeguato. Possiamo ammettere che questo caso è stato mostrato puramente per scopo illustrativo: ad un sistema termico del genere non sarà mai fornito un riferimento (temperatura) che cambia in modo così repentino.

Riferimenti bibliografici

- [1] AgricolaModerna. Germination room, 2021. [photo taken from website; accessed June 24, 2021].
- [2] D. Despommier. Vertical farms, building a viable indoor farming model for cities. 2019.
- [3] DiscoverAgriculture. Fully automated hydroponic farm — modern hydroponic farming — amazing agriculture technology, 2020. [video taken from YouTube as a reference for designing machinery].
- [4] C. S. Dr. M Brechner, Dr. A.J. Both. Cornell controlled environment agriculture hydroponic lettuce handbook. 2013.
- [5] J. Halveland. Design of a shallow-aero ebb and flow hydroponics system and associated educational module for tri cycle farms. 2020.
- [6] S. A. A. P. N. Petrova, A. Vershinin. Automation of monitoring the thermal conditions in a room. 2015.
- [7] C. Zeidler. Vertical farm 2.0 - designing an economically feasible vertical farm – a combined european endeavor for sustainable urban agriculture. 2015.