UNIVERSITA' DI SALERNO

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE ED ELETTRICA E MATEMATICA APPLICATA

Laurea Magistrale in Ingegneria Informatica

Project work

Deliverable 2

Sistemi Embedded

Gruppo: 8

Marotta Giuseppe - 0622702302 - g.marotta31@studenti.unisa.it

Rea Gaetano - 0622702190 - g.rea7@studenti.unisa.it

Squitieri Giuseppe - 0622702339 - g.squitieri8@studenti.unisa.it

Tramice Davide - 0622702194 - d.tramice@studenti.unisa.it

ANNO ACCADEMICO 2023/2024

Indice

1 Implementazione

In accordo con la progettazione descritta nel capitolo precedente, in questo capitolo verrà dettagliata l'implementazione del modello Stateflow. Si inizierà con una panoramica generale, per poi scendere progressivamente a livelli di dettaglio sempre maggiori.

1.1 Descrizione del Modello

Il modello Stateflow è stato sviluppato per gestire i vari stati e le transizioni del sistema di controllo del cancello automatico. I principali componenti del modello includono:

- **Sistema di Controllo del Cancello**: Gestisce gli stati di apertura, chiusura, controllo dei tempi e gestione degli ostacoli.
- **Pulsanti di Controllo**: Include i pulsanti B1, B2 e B3 per l'apertura/chiusura, la regolazione del tempo di chiusura e la regolazione del tempo di lavoro, rispettivamente.
- **Sensori di Ostacolo**: Sensori P1 e P2 per rilevare la presenza di ostacoli e per il controllo sulla chiusura completa del cancello.
- **Indicatori LED**: LED verde, giallo e rosso per indicare lo stato del cancello (aperto, chiuso, in movimento, errore).

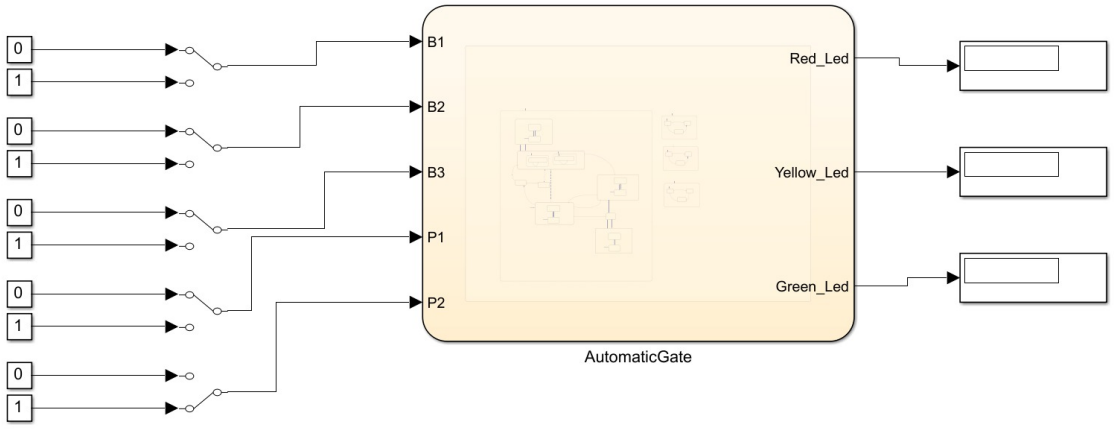

Figura 1: Vista Generale del Sistema.

Di seguito vengono presentati gli input e gli output del sistema insieme alle costanti interne che vengono utilizzate per una corretta implementazione e un giusto funzionamento:

Type	Name	Value	Port
Input_Data	B1	-	1
Input_Data	B2	-	2
Input_Data	B3	-	3
Output_Data	Red_Led	-	1
Output_Data	Green_Led	-	2
Output_Data	Yellow_Led	-	3
Input_Data	P1	-	4
Input_Data	P2	-	5
Constant_Data	BUTTON_OFF	0	-
Local_Data	WORK_DUR	10	-
Constant_Data	P_ON	1	-
Constant_Data	P_OFF	0	-
Local_Data	OPEN_DUR	10	-
Constant_Data	LED_ON	1	-
Constant_Data	LED_OFF	0	-
Constant_Data	BUTTON_ON	1	-
Constant_Data	EMER_DUR	10	-
Constant_Data	YELLOW_DUR	1	-
Constant_Data	GREEN_DUR	0.5	-
Constant_Data	EMER_P1_DUR	30	-
Local_Event	B1_PRESSED	-	-
Local_Event	B2_PRESSED	-	-
Local_Event	B3_PRESSED	-	-

Tabella 1: Variabili utilizzate nel modello Stateflow

1.2 Introduzione al Core del Cancello

Nel core del nostro sistema, troviamo vari stati che lavorano in parallelo: **Automatic Gate, B1, B2, B3**. Lo stato **Automatic Gate** gestisce tutta la logica di funzionamento del cancello, mentre gli altri stati gestiscono la corretta pressione dei rispettivi pulsanti.

1.2.1 Pulsante B1

Descriviamo nel dettaglio lo stato del pulsante B1. Questo componente viene utilizzato per richiedere l'apertura o la chiusura del cancello e sono previsti tre super-stati: **RELEASED, PRESSED, LONGPRESSED**.

1. **RELEASED**: Stato iniziale. Si passa allo stato **PRESSED** quando viene rilevata la pressione del pulsante B1.

2. **PRESSED**: Stato attivo durante la pressione del pulsante. Se viene rilevato un fronte di discesa, si passa allo stato **LONGPRESSED**.

3. **LONGPRESSED**: Stato transitorio che, quasi istantaneamente, ritorna allo stato di **RELEASED**. Prima di tornare allo stato di **RELEASED**, viene eseguita un'azione che indica l'evento "B1_pressed".

Successivamente, descriveremo come questo evento impatta sulla logica di funzionamento nello stato **AUTOMATIC GATE**.

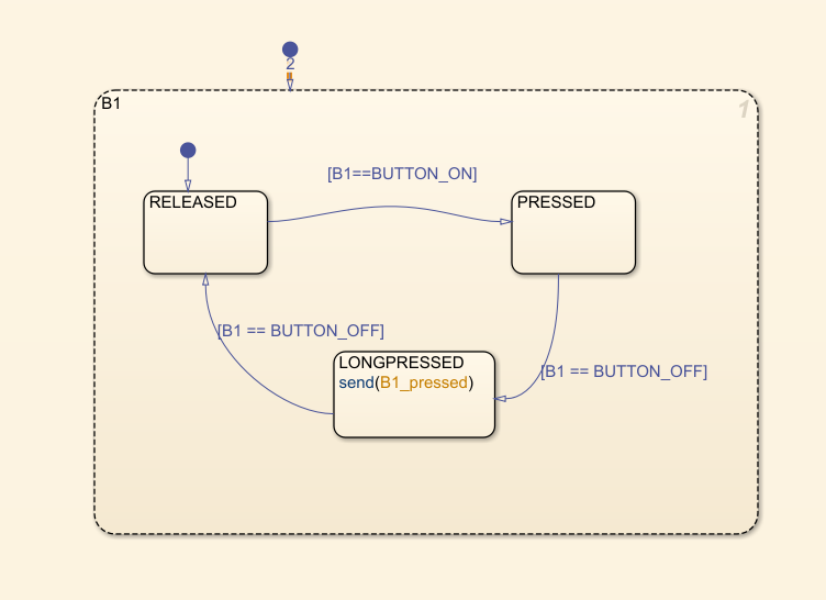

Figura 2: Pulsante B1.

1.2.2 Pulsante B2 e B3

Il Pulsante B2 è utile per regolare il tempo di chiusura automatica del cancello, mentre il pulsante B3 è usato per il settaggio dei valori del Tempo di Lavoro in fase di apertura e chiusura. Entrambi condividono lo stesso funzionamento del Pulsante B1 descritto precedentemente.

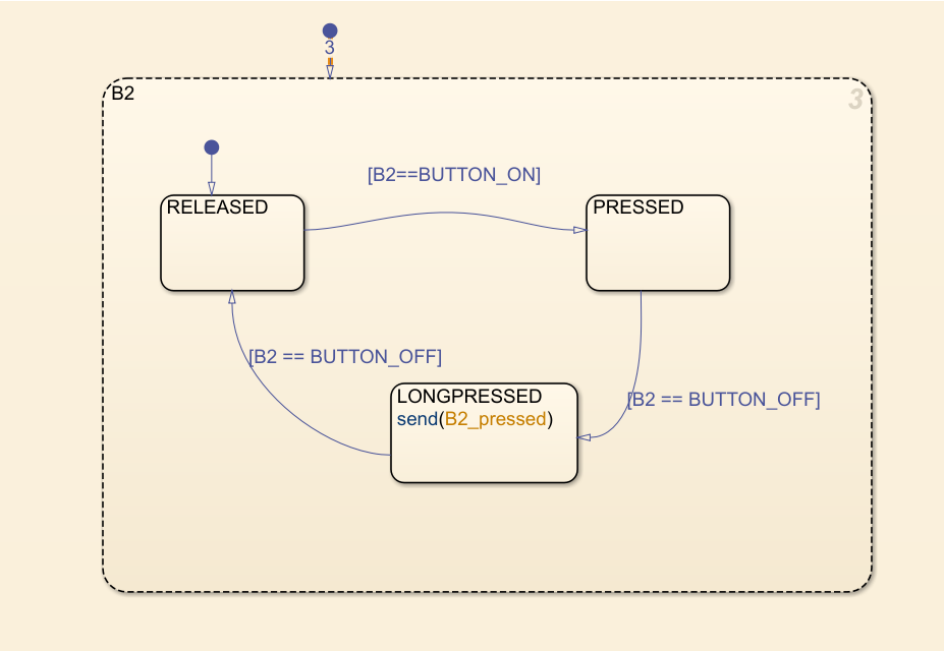

Figura 3: Pulsante B2.

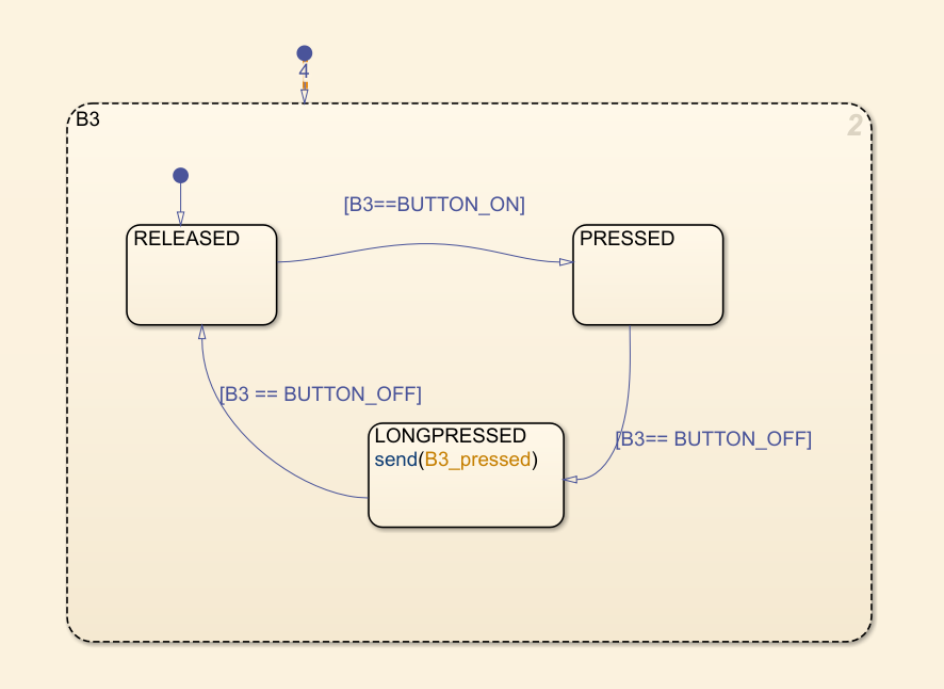

Figura 4: Pulsante B3.

1.2.3 Automatic Gate

Arriviamo ora a descrivere lo stato che implementa la logica di funzionamento. Innanzitutto, come scelta implementativa, consideriamo come Stato Iniziale lo stato **CLOSING**. Consideriamo quest'ultimo tale poiché le specifiche richiedono che al momento dell'attivazione del sistema, se il cancello è aperto, ovvero P2 è inattivo, quest'ultimo passi in uno stato di chiusura descritto appunto dal suddetto stato. All'interno troviamo due super-stati **YELLOW_ON** e **YELLOW_OFF** che descrivono il Toggle del LED Giallo che avviene con frequenza 0.5 Hz.

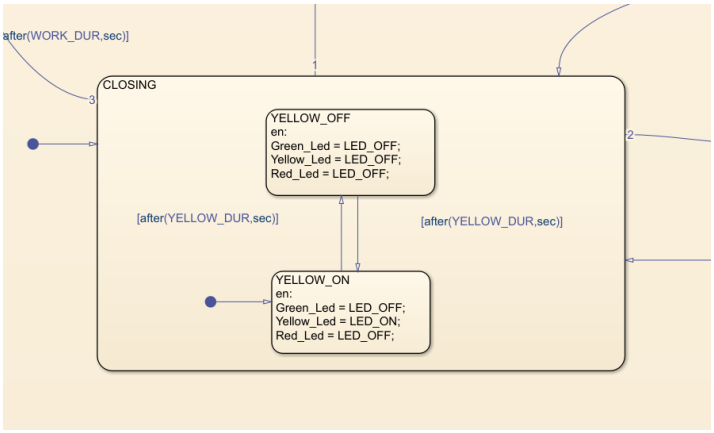

Figura 5: Closing State.

Nel momento in cui il cancello si chiude completamente o è già chiuso (P2 è attivo), si passa allo stato **CLOSED**. In questo stato, tutti i LED sono spenti e al suo interno troviamo due super-stati: **OPEN_DUR_SETTING** e **WORK_DUR_SETTING** che descrivono il settaggio dei parametri del tempo di chiusura automatica e del tempo di lavoro, che può essere attuato solo se il sistema si trova in questo stato. Per descrivere questi stati interni prenderemo in esame **OPEN_DUR_SETTING**, poiché il discorso è analogo per **WORK_DUR_SETTING**. Quando viene premuto B2, viene generato un evento che poi viene catturato da quest'ultimo stato e viene effettuata un'operazione che modifica la variabile **OPEN_DUR**:

$$\text{OPEN\_DUR} = \text{mod}((\text{OPEN\_DUR}), 120) + 10$$

Questo perché i valori impostati hanno un range (10, 120).

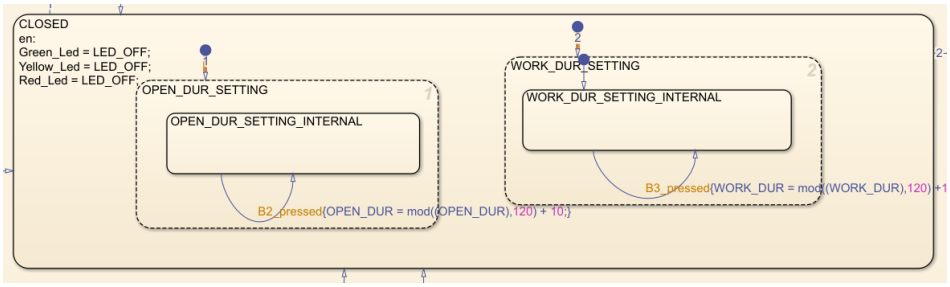

Figura 6: Closed State.

Chiaramente dallo stato di **CLOSED** si passa a **OPENING** nel momento in cui viene premuto B1. Al suo interno troviamo il Toggling del LED Giallo con lo stesso meccanismo di **CLOSING**.

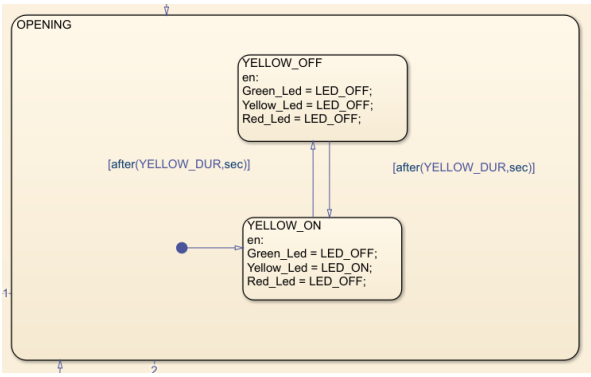

Figura 7: Opening State.

Appena finito il tempo di lavoro, il cancello va nello stato **OPEN** dove tutti i LED sono accesi.

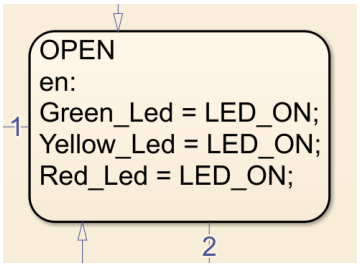

Figura 8: Open State.

1.2.4 Emergency States

Dallo stato **OPEN**, per tornare di nuovo in uno stato di chiusura, è possibile premere il pulsante B1 o attendere la chiusura automatica. Tuttavia, come specificato, il cancello non eseguirà il comando di chiusura se il sensore P1 è attivo, ma passerà nello stato **EMERGENCY_P1_OPEN**. Il sistema rimane in questo stato per 30 secondi o finché il sensore P1 non ritorna inattivo. Al suo interno troviamo due super-stati che descrivono il toggling del LED verde: **GREEN_ON** e **GREEN_OFF** con frequenza 1 Hz.

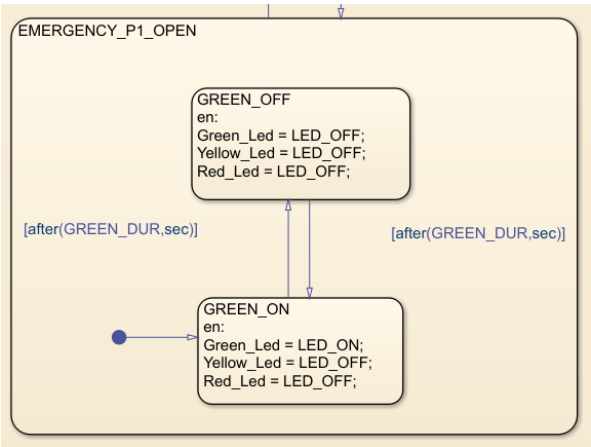

Figura 9: Emergency P1 Open State.

Se P1 è attivo quando il cancello è chiuso e viene premuto il pulsante B1, il sistema passerà in uno stato analogo: **EMERGENCY_P1_CLOSED**.

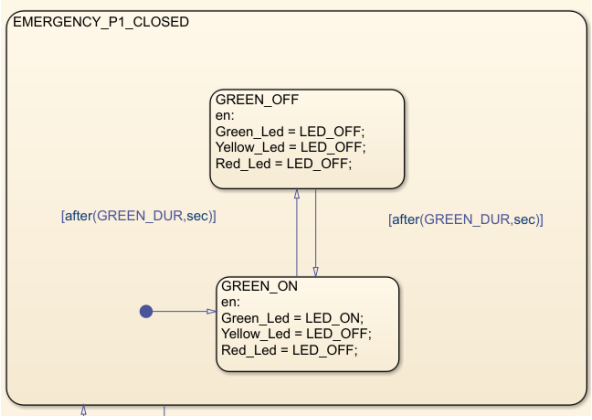

Figura 10: Emergency P1 Closed State.

Infine, descriviamo due stati che indicano una situazione di emergenza più grave (vero e proprio malfunzionamento del sistema): **EMERGENCY** e **EMERGENCY_LED**. Lo stato **CLOSING** può durare al massimo un tempo T di lavoro, oltrepassato il quale, se P2 non è attivo, il sistema va in un primo stato **EMERGENCY** dove tutti i LED sono spenti e rimane per 10 secondi. Trascorsi questi 10 secondi, se P2 non è attivo, il sistema passa nello stato **EMERGENCY_LED** dove è attivo il LED di emergenza rosso.

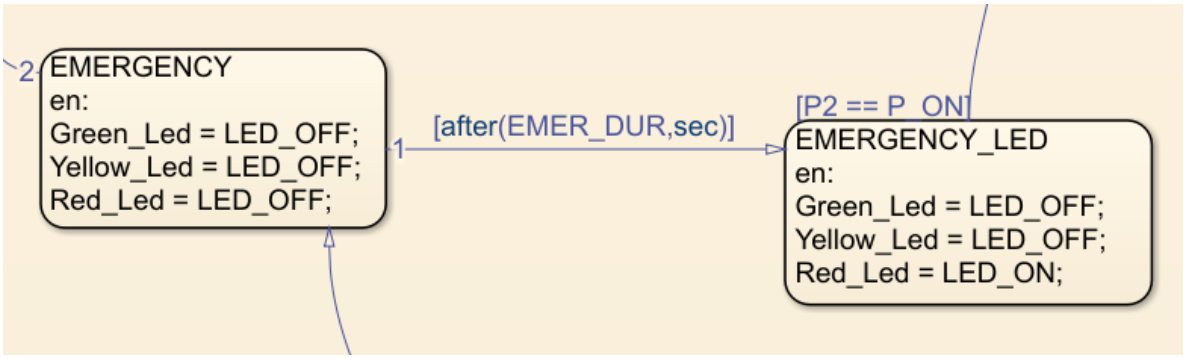

Figura 11: Emergency State.

2 Testing

In questo capitolo verranno illustrate le modalita per effettuare il testing del sistema. Questa parte è fondamentale per assicurarsi che non ci siano errori concettuali all'interno del modello e consegnare alla fase implementativa un progetto completo e senza imprecisioni.
Per testare lo State Model in Simulink si è utilizzato il tool Simulink Test che permette di descrivere procedure di testing in maniera agevole e soprattutto riproducibili.
Di seguito verranno descritti tutti i casi di test eseguiti che coprono quasi totalmente il sistema.

2.1 Note al Testing

Dopo un'attenta revisione, ci siamo resi conto che gli Activity diagrams attuali includono un numero eccessivo di casi differenti. Questo approccio, seppur inizialmente pensato per essere completo, ha reso la fase di test su Simulink Test estremamente complessa e difficile da gestire.
Per rendere i test più efficaci e praticabili, abbiamo deciso di spezzare i diagrammi di attività in parti più piccole e specifiche. Questo permetterà di isolare e testare singolarmente i vari casi, facilitando l'identificazione di eventuali problemi e migliorando la qualità complessiva del progetto.
Riteniamo che questa suddivisione ci consentirà di ottenere una validazione più accurata del sistema, garantendo al contempo una gestione più agevole dei test su Simulink Test.

2.2 Test Case1: Opening/Closing

In questo primo scenario testiamo il corretto funzionamento del sistema, dalla fase di chiusura fino a quando è chiuso e poi dalla fase di apertura fino a che non è completamente aperto. Questo test e molti dei successivi possono essere eseguiti molteplici volte ciclicamente.

- Il cancello automatico parte da uno stato di chiusura in cui la fotocellula **P2** non rileva nulla. In questa fase viene testato il blinking del led giallo (yellow_led).
- La fotocellula viene occupata entro il tempo di lavoro e quindi il cancello passa nello stato **CLOSED**. In questo stato si controlla che tutti i led siano spenti.
- Viene richiesta l'apertura del cancello generando un fronte di salita e poi uno di discesa con il segnale del pulsante **B1**
- Il cancello passa nello stato di **OPENING** durante il quale viene di nuovo testato se il blinking avviene alla frequenza giusta.
- Quando è passato il tempo di lavoro, inizialmente impostato a 10 secondi, si controlla che tutti i led siano accesi e che quindi il sistema si trova nello stato **OPEN**.

- Quando il sistema è nello stato open si avvia un timer che quando è scaduto fa passare il sistema nello stato **CLOSING**
- A questo punto si può ripartire dal primo punto e ripetere il test.

Di seguito in Figura ... è presente il test in Simulink Test.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0;	1. true	Test_closing_state	Inizializzazione degli input
Test_closing_state	1. after(10,sec)	Start_test_closed_state	Test dello stato di chiusura intercettando gli stati del Led_Yellow
Test_closing_state_1 verify(Yellow_Led == 1,'TEST FAILED')	1. after(1,sec)	Test_closing_state_2	Yellow_led == ON
Test_closing_state_2 verify(Yellow_Led == 0,'TEST FAILED')	1. after(1,sec)	Test_closing_state_1	Yellow_led == OFF
Start_test_closed_state P2 = 1;	1. after(2,sec)	Test_closed_state	Inizializzazione variabili per andare nello stato CLOSED
Test_closed_state verify(Green_Led == 0,'TEST FAILED') verify(Yellow_Led == 0,'TEST FAILED') verify(Red_Led == 0,'TEST FAILED')	1. after(5,sec)	Start_test_opening_state	Verifica che tutti i led sono spenti
Start_test_opening_state	1. after(0.2,sec)	Test_opening_state	Generare un fronte di salita e discesa del pulsante B1 per cambiare stato in OPENING
Start_test_opening_state_1 B1 = 1;	1. after(0.1,sec)	Start_test_opening_state_2	
Start_test_opening_state_2 B1=0;	1. after(0.1,sec)	Start_test_opening_state_1	
Test_opening_state	1. after(10,sec)	Test_open_state	Testing dello stato di OPENING intercettando il blinking del Yellow_Led
Test_opening_state_1 verify(Yellow_Led == 1,'TEST FAILED')	1. after(1,sec)	Test_opening_state_2	
Test_opening_state_2 verify(Yellow_Led == 0,'TEST FAILED')	1. after(1,sec)	Test_opening_state_1	
Test_open_state P2 = 0; verify(Green_Led == 1,'TEST FAILED') verify(Yellow_Led == 1,'TEST FAILED') verify(Red_Led == 1,'TEST FAILED')	1. after(10,sec)	Test_closing_state	Testing dello stato OPEN verificando che tutti i led sono accesi

Figura 12: Test 1.

2.3 Test Case 2: Opening emergency (P1 is ON)

Il secondo scenario testa l'entrata e le uscite da uno degli stati di emergenza del cancello automatico, in particolare quello nel quale entra dallo stato **CLOSED** quando viene richiesta l'apertura, ma il sensore **P1** è impegnato.

- Il sistema parte dallo stato **CLOSING**, quindi per eseguire il test viene portato nello stato **CLOSE** attivando **P2**.
- Viene controllato l'output del sistema in modo tale da assicurarsi che il cancello sia veramente chiuso.

- Si attiva la fotocellula con il valore 1 e successivamente, come nel caso precedente, si richiede l'apertura tramite **B1**. Il risultato atteso è quello che il led verde (Green_Led) lampeggi ad una frequenza di 1Hz per 30 secondi se P1 rimane attiva o che si ritorni nello stato **CLOSED** appena c'è un fronte di discesa.

- Il primo caso di uscita testato è quello che utilizza il timer impostato a 30 sec.

- Per testare la seconda uscita si forza il sistema a rientrare nello stato di emergenza e poi si disattiva **P1**.

- Entrambi i test sono stati superati dal sistema.

In Figura ... è rappresentato il test nella sua interezza. Si noti come sarebbe stato dispendioso utilizzare due casi di test separati per un solo stato di emergenza raggiungibile.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0;	1. true	Going_To_Closed_State	Inizializzazione degli input
Going_To_Closed_State P2 = 1;	1. after(1,sec)	Testing_Closed_State	P2 a 1 in modo tale da arrivare nello stato CLOSED
Testing_Closed_State verify(Green_Led == 0,' TEST FAILED') verify(Yellow_Led == 0,' TEST FAILED') verify(Red_Led == 0,' TEST FAILED')	1. after(5,sec)	Requesting_openinig_with_P1	Check dello stato CLOSED
Requesting_openinig_with_P1	1. after(0.2,sec)	Testing_Emergency_P1_Closed	Richiesta di apertura con P1 attivo
Requesting_openinig_with_P1_1 P1 = 1; B1 = 1;	1. after(0.1,sec)	Requesting_openinig_with_P1_2	
Requesting_openinig_with_P1_2 B1 = 0;	1. after(0.1,sec)	Requesting_openinig_with_P1_1	
Testing_Emergency_P1_Closed	1. after(30,sec)	Testing_Closed_State1	Test dello stato di emergenza visto che P1 è attivo, si attende per 30 sec che il led verde smette di lampeggiare
Testing_Emergency_P1_Closed_1 verify(Green_Led == 1, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Closed_2	
Testing_Emergency_P1_Closed_2 verify(Green_Led == 0, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Closed_1	
Testing_Closed_State1 verify(Green_Led == 0,' TEST FAILED') verify(Yellow_Led == 0,' TEST FAILED') verify(Red_Led == 0,' TEST FAILED') P1 = 0;	1. after(5,sec)	Requesting_openinig_with_P1_2	Ritorno nello stato CLOSED
Requesting_openinig_with_P1_2	1. after(0.2,sec)	Testing_Emergency_P1_Closed1	Richiesta apertura con P1 impegnato
Requesting_openinig_with_P1_2_1 P1 = 1; B1 = 1;	1. after(0.1,sec)	Requesting_openinig_with_P1_2_2	
Requesting_openinig_with_P1_2_2 B1 = 0;	1. after(0.1,sec)	Requesting_openinig_with_P1_2_1	
Testing_Emergency_P1_Closed1	1. after(15,sec)	Removing_the_object_from_P1	Si attende 15 secondi nello stato di emergenza
Testing_Emergency_P1_Closed_1 verify(Green_Led == 1, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Closed_2	
Testing_Emergency_P1_Closed_2 verify(Green_Led == 0, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Closed_1	
Removing_the_object_from_P1 P1 = 0;	1. after(1,sec)	Testing_Closed_State2	Si rimuove l'oggetto da P1 e si controlla se lo stato è CLOSED
Testing_Closed_State2 verify(Green_Led == 0,' TEST FAILED') verify(Yellow_Led == 0,' TEST FAILED') verify(Red_Led == 0,' TEST FAILED')			Check dello stato CLOSED

Figura 13: Test 2.

2.4 Test Case 3: Closing emergency (P1 is ON)

Questo caso di test rappresenta una condizione di emergenza simile alla precedente, ma raggiungibile solo dallo stato **OPEN**. La descrizione è del tutto simile a quella sopra quindi si è scelto di ometterla.
Di seguito è riportato il caso di test in Figura ... .
Si sottolinea inoltre che la frequenza di lampeggio e le condizioni di uscita dallo stato di emergenza sono identiche al caso precedente.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0;	1. true	Going_To_Closed_State	Inizializzazione degli input
Going_To_Closed_State P2 = 1;	1. after(1,sec)	Testing_Closed_State	P2 a 1 in modo tale da arrivare nello stato CLOSED
Testing_Closed_State verify(Green_Led == 0,' TEST FAILED') verify(Yellow_Led == 0,' TEST FAILED') verify(Red_Led == 0,' TEST FAILED')	1. after(5,sec)	Requesting_Openinig_State	Check dello stato CLOSED
Requesting_Openinig_State	1. after(0.2,sec)	Testing_Opening	Richiesta di chiusura con P1 attivo
Requesting_Openinig_State1 B1 = 1;	1. after(0.1,sec)	Requesting_Openinig_State2	
Requesting_Openinig_State2 B1 = 0;	1. after(0.1,sec)	Requesting_Openinig_State1	
Testing_Opening	1. after(10,sec)	Testing_Open_State_1	Test dello stato di emergenza visto che P1 è attivo, si attende per 30 sec che il led verde smette di lampeggiare
Testing_Opening1 verify(Yellow_Led== 1, 'TEST_FAILED')	1. after(1,sec)	Testing_Opening2	
Testing_Opening2 verify(Yellow_Led == 0, 'TEST_FAILED')	1. after(1,sec)	Testing_Opening1	
Testing_Open_State_1 verify(Green_Led == 1,' TEST FAILED') verify(Yellow_Led == 1,' TEST FAILED') verify(Red_Led == 1,' TEST FAILED')	1. after(5,sec)	Requesting_closing_with_P1	Ritorno nello stato OPEN
Requesting_closing_with_P1	1. after(0.2,sec)	Testing_Emergency_P1_Open1	
Requesting_closing_with_P1_1 P1 = 1; B1 = 1;	1. after(0.1,sec)	Requesting_closing_with_P1_2	
Requesting_closing_with_P1_2 B1 = 0;	1. after(0.1,sec)	Requesting_closing_with_P1_1	
Testing_Emergency_P1_Open1	1. after(30,sec)	Testing_Open_State_2	
Testing_Emergency_P1_Open1_1 verify(Green_Led == 1, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Open1_2	
Testing_Emergency_P1_Open1_2 verify(Green_Led == 0, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Open1_1	
Testing_Open_State_2 verify(Green_Led == 1,' TEST FAILED') verify(Yellow_Led == 1,' TEST FAILED') verify(Red_Led == 1,' TEST FAILED')	1. after(5,sec)	Requesting_closing_with_P1_2	
Requesting_closing_with_P1_2	1. after(0.2,sec)	Testing_Emergency_P1_Open_2	Richiesta chiusura con P1 impegnato
Requesting_closing_with_P1_2_1 P1 = 1; B1 = 1;	1. after(0.1,sec)	Requesting_closing_with_P1_2_2	
Requesting_closing_with_P1_2_2 B1 = 0;	1. after(0.1,sec)	Requesting_closing_with_P1_2_1	
Testing_Emergency_P1_Open_2	1. after(15,sec)	Removing_the_object_from_P1	Si attende 15 secondi nello stato di emergenza
Testing_Emergency_P1_Open_2_1 verify(Green_Led == 1, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Open_2_2	
Testing_Emergency_P1_Open_2_2 verify(Green_Led == 0, 'TEST_FAILED')	1. after(0.5,sec)	Testing_Emergency_P1_Open_2_1	
Removing_the_object_from_P1 P1 = 0;	1. true	Testing_Open_State2	Si rimuove l'oggetto da P1 e si controlla se lo stato è OPEN
Testing_Open_State2 verify(Green_Led == 1,' TEST FAILED') verify(Yellow_Led == 1,' TEST FAILED') verify(Red_Led == 1,' TEST FAILED')	1. after(10,sec)	Going_To_Closed_State	Check dello stato OPEN

Figura 144 Test 3.

2.5 Test Case 4: Obstacle during the Closing phase

Il test effettuato in questa sezione è molto importante nel caso in cui il sistema venisse implementato nel mondo reale, infatti controlla se il cancello automatico, che si suppone in fase di chiusura, si riapra nel caso venga rilevato un ostacolo davanti al sensore **P1**

- Il sistema già parte nello stato **CLOSING** quindi non c'è bisogno di forzare alcuno stato.
- Viene simulata la presenza di un ostacolo attivando **P1**.
- Si controlla che il sistema continui ad avere il blinking del led giallo (yellow_led) in uscita.
- Infine possiamo capire che il sistema è passato dalla fase di chiusura a quella di apertura intercettando lo stato finale. Se tutti i ledi sono accesi allora il test va a buon fine.
- Dopo aver controllato che il sistema si trovi nello stato **OPEN** si genera l'evento che avvia la fase di chiusura, ovvero la pressione di **B1**. In questo modo il test può essere ripetuto diverse volte.

In Figura ... sono riportate le varie fasi in Simulink Test.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0; Add step after - Add sub-step -	1. true	Test_closing_state	Inizializzazione degli input
Test_closing_state	1. after(5,sec)	Start_test_P1	Test dello sato di chiusura intercendo gli stati del Led_Yellow
Test_closing_state_1 verify(Yellow_Led == 1, 'TEST FAILED');	1. after(1,sec)	Test_closing_state_2	Yellow_Led == ON
Test_closing_state_2 verify(Yellow_Led == 0, 'TEST FAILED');	1. after(1,sec)	Test_closing_state_1	Yellow_Led == OFF
Start_test_P1 P1=1;	1. true	Test_opening_state	Inizializzazione variabile per andare nello stato di OPENING, a cusa dell'ostacolo davanti a P1
Test_opening_state	1. after(10,sec)	Test_open_state	Test dello sato di apertura intercendo gli stati del Led_Yellow
Start_test_opening_1 verify(Yellow_Led == 1, 'TEST FAILED');	1. after(1,sec)	Start_test_opening_2	Yellow_Led == ON
Start_test_opening_2 verify(Yellow_Led == 0, 'TEST FAILED');	1. after(1,sec)	Start_test_opening_1	Yellow_Led == OFF

Test_open_state
verify(Green_Led == 1, 'TEST FAILED');
verify(Yellow_Led == 1, 'TEST FAILED');
verify(Red_Led == 1, 'TEST FAILED');
	1. after(1,sec)	Test_change_P1	Testinf dello stato OPEN verificando che tutti i led sono accesi

Test_change_P1 P1=0;	1. true	Start_test_closing_state	Inizializzazione sensore P1 a 0 per togliere l'ostacolo
Start_test_closing_state	1. after(0.2,sec)	Test_closing_state	Genera fronte di salita e discesa del pulsante B1 per cambiare da stato OPEN A CLOSING
Start_test_closing_state_1 B1=1;	1. after(0.1,sec)	Start_test_closing_state_2	B1 = 1
Start_test_closing_state_2 B1=0;	1. after(0.1,sec)	Start_test_closing_state_1	B1 = 0

Figura 15: Test 4.

2.6 Test Case 5: Closing error (P2 is not ON)

In questo test viene verificato che il sistema raggiunge correttamente due stati di errore nel caso in cui la chiusura del cancello automatico non avvenga nei tempi previsti definiti dall'utente.

- Lo stato in cui si parte è sempre quello di **CLOSING**, in questo scenario **P2** viene attivata dopo che è passato il tempo di lavoro.
- Il sistema funzionante correttamente entra prima in un uno stato di errore per 10 secondi in cui tutti i led spenti.
- Una volta verificato lo stato precedente e passati i 10 secondi il sistema attica il led rosso (red_led).
- Successivamente si verifica che il sistema permane in questo stato fino a quando non viene attivato il sensore **P2**. Quando arriva il segnale il sistema passa nello stato **CLOSED**

Nella Figura ... è illustrato un caso di test. È importante notare che, sebbene questo test risulti parzialmente ridondante poiché include condizioni già verificate nei controlli precedenti, esso introduce anche nuove verifiche. Si è quindi deciso di mantenere la ripetizione per aumentare ulteriormente la sicurezza del sistema.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0;	1. true	Test_closing_state	Inizializzazione variabile. Sono nello stato di chiusura
Test_closing_state	1. after(5,sec)	active_P2	Testing dello stato CLOSING, accertando il blinking del Yellow_Led
Test_closing_state_1 verify(Yellow_Led == 1, 'TEST FAILED');	1. after(1,sec)	Test_closing_state_2	Yellow_Led == ON
Test_closing_state_2 verify(Yellow_Led == 0, 'TEST FAILED');	1. after(1,sec)	Test_closing_state_1	Yellow_Led == OFF
active_P2 P2=1;	1. after(1,sec)	Test_closed_state	Inizializzazione variabili per andare nello stato CLOSED

Test_closed_state
verify(Green_Led == 0,'TEST FAILED');
verify(Yellow_Led == 0, 'TEST FAILED');
verify(Red_Led == 0, 'TEST FAILED');
	1. true	Start_test_opening_state	Testing dello stato CLOSED, accertando che tutti i Led siano spenti

Start_test_opening_state	1. after(0.2,sec)	Test_opening_state	Generare un fronte di salita e discesa del pulsante B1 per cambiare stato in OPENING
Start_test_opening_state_1 B1=1;	1. after(0.1,sec)	Start_test_opening_state_2	
Start_test_opening_state_2 B1=0;	1. after(0.1,sec)	Start_test_opening_state_1	
Test_opening_state	1. after(10,sec)	Test_open_state	Testing dello stato di OPENING intercettando il blinking del Yellow_Led
Test_opening_state_1 verify(Yellow_Led == 1, 'TEST FAILED');	1. after(1,sec)	Test_opening_state_2	
Test_opening_state_2 verify(Yellow_Led == 0, 'TEST FAILED');	1. after(1,sec)	Test_opening_state_1	
Test_open_state P2=0; verify(Green_Led == 1, 'TEST FAILED') verify(Yellow_Led == 1, 'TEST FAILED') verify(Red_Led == 1, 'TEST FAILED')	1. after(10,sec)	Test2_closing_state	Testing dello stato di OPEN verificando che tutti i led sono accesi. Passo allo stato di CLOSING dopo il WORK_DUR(10 sec).
Test2_closing_state	1. after(10,sec)	Test_error_state	Testing dello stato CLOSING, accertando il blinking del Yellow_Led
Test2_closing_state_1 verify(Yellow_Led == 1, 'TEST FAILED');	1. after(1,sec)	Test2_closing_state_2	Yellow_Led == ON
Test2_closing_state_2 verify(Yellow_Led == 0, 'TEST FAILED');	1. after(1,sec)	Test2_closing_state_1	Yellow_Led == OFF
Test_error_state	1. after(10,sec)	Test_error_state_red_led	Testing ERROR State_dopo il timeout di WORK_DUR(10 seocndi)
Test_error_state_red_led verify(Red_Led == 1, 'TEST FAILED');	1. after(5,sec)	Test_exit_form_error_state	Testing accesino Red_Led dopo secondi dall'entrata nell' ERROR_State
Test_exit_form_error_state P2=1;	1. after(2,sec)	Test_close_state_after_error	Inizializzazione variabili per andare nello stato CLOSED

Test_close_state_after_error
verify(Green_Led == 0, 'TEST FAILED');
verify(Yellow_Led == 0, 'TEST FAILED');
verify(Red_Led == 0, 'TEST FAILED');
			Testing dello stato CLOSED, accertando che tutti i Led siano spenti

Figura 16: Test 5.

2.7 Test Case 6:

In questo test, viene verificato che l'impostazione del tempo di chiusura automatico e il tempo di lavoro funzionino in maniera ciclica. Una volta raggiunto il massimo tempo impostabile (120 secondi), ritorna al tempo iniziale (10 secondi).

- Il cancello automatico parte da uno stato di chiusura in cui la fotocellula **P2** non rileva nulla. In questa fase viene testato il lampeggio del LED giallo (yellow LED).

- La fotocellula **P2** viene attivata entro il tempo di lavoro e quindi il cancello passa allo stato **CLOSED**. In questo stato si controlla che tutti i LED siano spenti.
- Viene richiesto l'aumento per dodici volte del tempo di chiusura automatico e del tempo di lavoro generando un fronte di salita e poi uno di discesa con il segnale dei pulsanti **B2** e **B3**.
- Viene richiesta l'apertura del cancello generando un fronte di salita e poi uno di discesa con il segnale del pulsante **B1**.
- Vengono attesi 10 secondi per verificare se la fotocellula **P2** non rileva nulla; ciò significa che il cancello è passato dallo stato di **CLOSED** allo stato di **OPEN** nel tempo desiderato.
- La fotocellula **P2** non viene attivata entro il tempo di lavoro e quindi il cancello passa allo stato **OPEN**. In questo stato si controlla che tutti i LED siano accesi.
- Quando è trascorso il tempo di chiusura automatico, impostato a 10 secondi, il sistema passa dallo stato **OPEN** allo stato di **CLOSING**. In questo modo si verifica se il tempo di chiusura è ritornato allo stato iniziale.
- Si verifica che il cancello passi dallo stato di **CLOSING** allo stato di **CLOSED**. Nel caso in cui non si chiuda entro 10 secondi, il sistema passa in stato di errore e dopo ulteriori 10 secondi, si verifica l'accensione del LED rosso (**Red LED**).

In Figura 17 sono riportate le varie fasi in Simulink Test.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0;	1. *true*	Going_to_closed_state	Inizializzazione degli input
Going_to_closed_state P2=1;	1. true	Test_closed_state	Inizializzazione variabili per andare nello stato CLOSED

Test_closed_state
 	
verify(Green_Led==0,'TEST FAILED');
verify(Yellow_Led==0,'TEST FAILED');
verify(Red_Led==0,'TEST FAILED');
	1. after(1,sec)	Increase_OPEN_DUR_WORK_DUR	Verifica che tutti i led sono spenti

Increase_OPEN_DUR_WORK_DUR	1. after(2.4,sec)	Requesting_opening	Aumento OPEN_DUR(tempo di chiusura automatico) e WORK_DUR(tempo di lavoro) per 12 volte in modo da partire da 10 e tornare a 10(viene fatto per vedere se funziona la ciclicità dei due timer)
Increase_OPEN_DUR_WORK_DUR_1 B2=1; B3=1;	1. after(0.1,sec)	Increase_OPEN_DUR_WORK_DUR_2	B2=1; B3=1;
Increase_OPEN_DUR_WORK_DUR_2 B2=0; B3=0;	1. after(0.1,sec)	Increase_OPEN_DUR_WORK_DUR_1	B2=0; B3=0;
Requesting_opening	1. after(0.2,sec)	Testing_Work_Duration	Richiedo apertura tramte un fronte di salita e discesa del pulsante B1
Requesting_opening_1 B1=1;	1. after(0.1,sec)	Requesting_opening_2	B1=1;
Requesting_opening_2 B1=0;	1. after(0.1,sec)	Requesting_opening_1	B1=0;
Testing_Work_Duration	1. after(10,sec)	Testing_OPEN_state	Test della durata del tempo di lavoro

Testing_OPEN_state
 	
P2 = 0;
 	
verify(Green_Led==1,'TEST FAILED');
verify(Yellow_Led==1,'TEST FAILED');
verify(Red_Led==1,'TEST FAILED');
	1. after(10,sec)	Testing_CLOSING	Verifico che tutti i led siano accesi, in questo modo il cancello è OPEN.
			Aspetto 10 secondi in modo tale da farlo chiudere automaticamente

Testing_CLOSING	1. after(10,sec)	Test_emergency_state	Aspetto 10 secondi in modo tale da far passare il tempo di lavoro e vedere se entra nello stato di EMRGENCY
Testing_CLOSING_1 verify(Yellow_Led==1,'TEST FAILED');	1. after(1,sec)	Testing_CLOSING_2	Yellow_Led==1
Testing_CLOSING_2 verify(Yellow_Led==0,'TEST FAILED');	1. after(1,sec)	Testing_CLOSING_1	Yellow_Led==0
Test_emergency_state	1. after(10,sec)	Test_Red_Led	Aspetto 10 secondi in modo tale da verificare se il RED_LED dello stato di EMRGENCY sia acceso
Test_Red_Led verify(Red_Led==1,'TEST FAILED');			

Figura 17: Test 6.

2.8 Test Case 7:

In questo test, viene verificato che l'impostazione del tempo di chiusura automatico e il tempo di lavoro funzioni correttamente. I tempi vengono impostati da 10 a 120 secondi per entrambi i casi tramite i relativi pulsanti.

- Il cancello automatico parte da uno stato di chiusura in cui la fotocellula **P2** non rileva nulla. In questa fase viene testato il lampeggio del LED giallo (yellow LED).
- La fotocellula **P2** viene attivata entro il tempo di lavoro(iniziale di 10 secondi) e quindi il cancello passa allo stato **CLOSED**. In questo stato si controlla che tutti i LED siano spenti.
- Viene richiesto l'aumento per undici volte del tempo di chiusura automatico e del tempo di lavoro generando un fronte di salita e poi uno di discesa con il segnale

dei pulsanti **B2** e **B3**. Passando cosi da 10 secondi a 120 secondi in entrambi i casi.

- Viene richiesta l'apertura del cancello generando un fronte di salita e poi uno di discesa con il segnale del pulsante **B1**.
- Vengono attesi 120 secondi per verificare se la fotocellula **P2** non rileva nulla; ciò significa che il cancello è passato dallo stato di **CLOSED** allo stato di **OPEN** nel tempo desiderato.
- La fotocellula **P2** non viene attivata entro il tempo di lavoro e quindi il cancello passa allo stato **OPEN**. In questo stato si controlla che tutti i LED siano accesi.
- Quando è trascorso il tempo di chiusura automatico, impostato a 120 secondi, il sistema passa dallo stato **OPEN** allo stato di **CLOSING**. In questo modo si verifica se il tempo di chiusura è ritornato allo stato iniziale.
- Si verifica che il cancello passi dallo stato di **CLOSING** allo stato di **CLOSED**. Nel caso in cui non si chiuda entro 120 secondi, il sistema passa in stato di errore e dopo ulteriori 10 secondi, si verifica l'accensione del LED rosso (**Red LED**).

In Figura 18 sono riportate le varie fasi in Simulink Test.

Step	Transition	Next Step	Description
Run %% Initialize data outputs. B1 = 0; B2 = 0; B3 = 0; P1 = 0; P2 = 0;	1. *true*	Test_closing_state	Inizializzazione degli input
Test_closing_state	1. after(10,sec)	Start_test_closed_state	Test dello stato di CLOSING intercettando gli stati del Led_Yellow
Test_closing_state_1 verify(Yellow_Led== 1 , 'Test Failed');	1. after(1,sec)	Test_closing_state_2	Yellow_Led= ON
Test_closing_state_2 verify(Yellow_Led== 0 , 'Test Failed')	1. after(1,sec)	Test_closing_state_1	Yellow_Led= OFF
Start_test_closed_state P2=1;	1. true	Test_closed_state	Impostazione variabili per andare nello stato CLOSED
Test_closed_state verify(Green_Led== 0 , 'Test Failed') verify(Yellow_Led== 0 , 'Test Failed') verify(Red_Led== 0 , 'Test Failed')	1. after(2,sec)	Start_test_increse_work_time	Verifica che tutti i led sono spenti nello stato CLOSED
Start_test_increse_work_time	1. after(2.2,sec)	Start_test_opening_state	Generare un fronte di salita e discesa del pulsante B2 e B3 in modo da cambiare il tempo di OPEN_DUR e WORK_DUR
Start_test_increse_work_time_1 B2=1; B3=1;	1. after(0.1,sec)	Start_test_increse_work_time_2	B2 =1 B3 = 1
Start_test_increse_work_time_2 B2=0; B3=0;	1. after(0.1,sec)	Start_test_increse_work_time_1	B2 = 0 B3 = 0
Start_test_opening_state	1. after(0.2,sec)	Test_opening_state	Generare un fronte di salita e discesa del pulsante B1 per cambiare stato in OPENING
Start_test_opening_state_1 B1=1;	1. after(0.1,sec)	Start_test_opening_state_2	B1=1
Start_test_opening_state_2 B1=0;	1. after(0.1,sec)	Start_test_opening_state_1	B1=0
Test_opening_state P2=0;	1. after(120,sec)	Test_open_state	Testing dello stato di OPENING intercettando il blinking del Yellow_Led
Test_opening_state_1 verify(Yellow_Led== 1 , 'Test Failed');	1. after(1,sec)	Test_opening_state_2	Yellow_Led= ON
Test_opening_state_2 verify(Yellow_Led== 0 , 'Test Failed');	1. after(1,sec)	Test_opening_state_1	Yellow_Led= OFF
Test_open_state verify(Green_Led== 1, 'Test Failed') verify(Yellow_Led== 1 , 'Test Failed') verify(Red_Led== 1, 'Test Failed')	1. after(120,sec)	Test2_closing_state	Testing dello stato OPEN verificando che tutti i led sono accesi. Dopo OPEN_DUR il cancello inizia a chiudersi automaticamente.
Test2_closing_state	1. after(120,sec)	Start_emergency	Test dello stato di CLOSING intercettando gli stati del Led_Yellow Il cancello entra in emergency_state dopo 120 sec(WORK_DUR)
Test2_closing_state_1 verify(Yellow_Led== 1 , 'Test Failed');	1. after(1,sec)	Test2_closing_state_2	Yellow_Led= ON
Test2_closing_state_2 verify(Yellow_Led== 0 , 'Test Failed');	1. after(1,sec)	Test2_closing_state_1	Yellow_Led= OFF
Start_emergency	1. after(10,sec)	Test_emergency_test	Test per l'accensione del RED_LED
Test_emergency_test verify(Red_Led== 1 , 'Test Failed')			Verifico che il RED_LED sia acceso nel emergency_state

Figura 18: Test 7.