SAPIENZA
UNIVERSITÀ DI ROMA

Sequential Functional Chart (SFC)
- Parte 1

Automazione

Vincenzo Suraci

STRUTTURA DEL NUCLEO TEMATICO

- INTRODUZIONE
- ELEMENTI DI BASE
- REGOLE DI EVOLUZIONE
- ESECUZIONE CICLICA
- RISOLUZIONE AMBIGUITÀ
- SINTASSI STANDARD DEL LINGUAGGIO SFC

INTRODUZIONE

Cenni storici

Prima degli anni '60 il CONTROLLO SEQUENZIALE era visto come ESTENSIONE DEL CONTROLLO DI TIPO CONTINUO o al più DIGITALE.

A partire dagli anni '60 si sviluppa la teoria sugli AUTOMI A STATI FINITI, i cui modelli formali permettono un'ANALISI MATEMATICA approfondita, ma scarsamente utile ai fini della PROGETTAZIONE degli algoritmi.

Negli anni '70 la progettazione dei sistemi di automazione è abbastanza elementare, si basa su RAPPRESENTAZIONI CIRCUITALI o su DESCRIZIONI TESTUALI.

Nel 1975 in Francia viene istituita una commissione per FORMALIZZARE uno strumento di PROGETTAZIONE di tipo DESCRITTIVO orientato al CONTROLLO SEQUENZIALE. Nasce GRAFCET.

Negli anni '80 GRAFCET viene recepito con il nome di SEQUENTIAL FUNCTIONAL CHART nello standard IEC (Comitato Elettrotecnico Internazionale) 848.

Cenni storici

Agli inizi degli anni '90 il SEQUENTIAL FUNCTIONAL CHART viene inserito nello standard IEC 61131-3 come LINGUAGGIO DI PROGRAMMAZIONE dei PLC.

Verso la fine degli anni '90 il W3C definisce lo UNIFIED MODELLING LANGUAGE. L'UML formalizza il DIAGRAMMA DEGLI STATI che è una generalizzazione del SEQUENIAL FUNCTIONAL CHART (SFC).

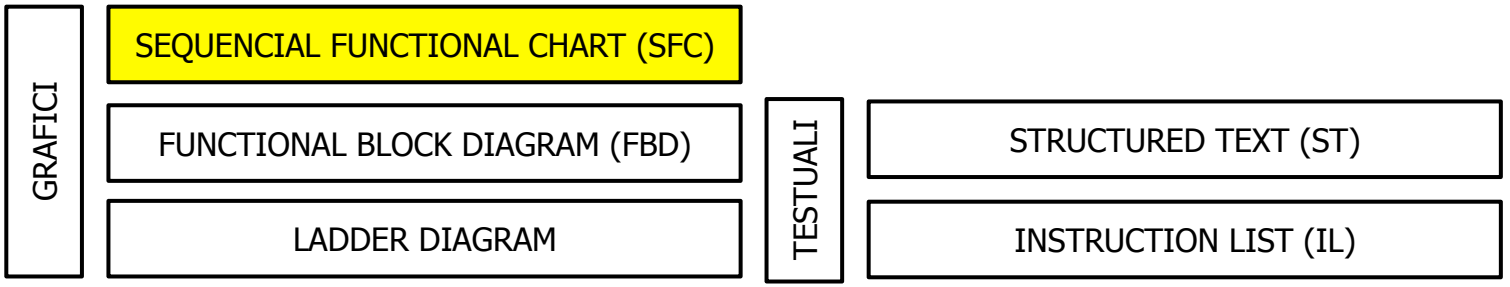
Ad oggi il SEQUENTIAL FUNCTIONAL CHART, ovvero il DIAGRAMMA DEGLI STATI UML, rappresenta lo STANDARD DE FACTO per la PROGETTAZIONE e la DOCUMENTAZIONE dei PLC.

Ad oggi i linguaggi per la PROGRAMMAZIONE dei PLC sono quelli derivati dal mondo informatico: C, Assembler, Java, Basic, etc.

Sequential Functional Chart

Abbiamo visto come il CONTROLLO LOGICO SEQUENZIALE necessiti di metodologie e strumenti per la PROGETTAZIONE e la successiva PROGRAMMAZIONE del SOFTWARE che realizza gli algoritmi del sistema di controllo.

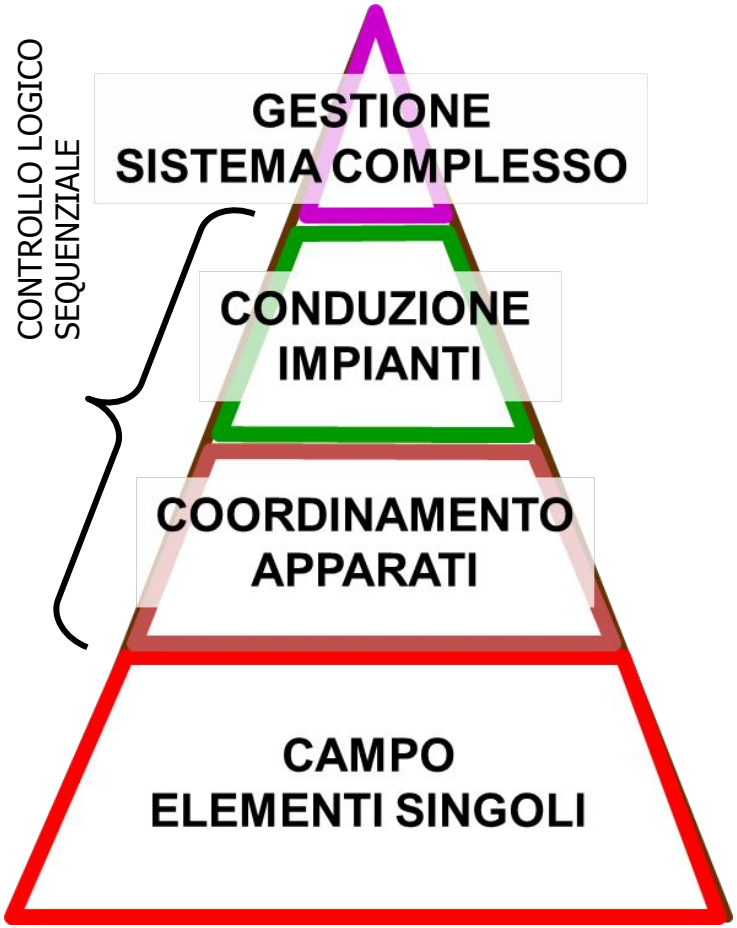
La Norma IEC 61131-3 definisce 5 linguaggi per i PLC:



A seguire studieremo il linguaggio SFC (ovvero il DIAGRAMMA DEGLI STATI di UML), inteso come strumento di PROGETTAZIONE e PROGRAMMAZIONE dei PLC.

ELEMENTI DI BASE

Generalità



OSSERVAZIONE

I controllori logico sequenziali operano principalmente a livello di **COORDINAMENTO** e di **CONDUZIONE**.

A tale livello **non ha senso** tenere conto della **DINAMICA** (evoluzione nel dominio del tempo) degli elementi singoli.

L'evoluzione dinamica dei singoli elementi può essere ***«nascosta»*** all'interno di una **CONDIZIONE OPERATIVA** del sistema.

L'evoluzione dell'intero sistema è data dalla **successione di tali condizioni operative**. Si possono definire le condizioni operative degli elementi singoli, degli apparati o degli impianti.

STATO

DEFINIZIONE

Lo **STATO** è una precisa **CONDIZIONE OPERATIVA** di una parte del sistema complesso.

OSSERVAZIONI

Lo STATO è una condizione **INVARIANTE** del sistema in esame, che può **modificarsi** o essere modificata soltanto in seguito ad un **EVENTO**.

Durante l'evoluzione dinamica del sistema complesso, un qualsiasi **STATO** di una qualsiasi parte del sistema può trovarsi solo in **due possibili condizioni**: **ATTIVO** o **INATTIVO**.

Dal punto di vista del SISTEMA DI CONTROLLO che si occupa di eseguire il PROGRAMMA di controllo LOGICO-SEQUENZIALE, uno **STATO corrisponde** ad **una o più PROCEDURE** che vengono **eseguite in maniera seriale** fintanto che lo **STATO è ATTIVO**, ovvero **fino a quando un EVENTO** non **cambia lo STATO** del sistema in esame.

AZIONE

DEFINIZIONE

Tutte le **OPERAZIONI** eseguite dal sistema in esame quando si trova in una precisa **CONDIZIONE OPERATIVA (STATO)** vengono chiamate **AZIONI**.

OSSERVAZIONI

Ogni **STATO** del sistema in esame è composto da un **insieme** determinato di **AZIONI**.

Dal punto di vista del SISTEMA DI CONTROLLO che si occupa di eseguire il PROGRAMMA di controllo LOGICO-SEQUENZIALE, una **AZIONE** corrisponde ad una o più PROCEDURE, ovvero da un **INSIEME DI ISTRUZIONI** che vengono eseguite in maniera seriale fintanto che lo **STATO è ATTIVO**, ovvero fino a quando un **EVENTO non cambia lo STATO** del sistema in esame.

TRANSIZIONI

DEFINIZIONE

Il **PASSAGGIO** da uno STATO PRECEDENTE ad uno STATO SUCCESSIVO a seguito di un **evento**, è detto **TRANSIZIONE**.

DEFINIZIONE

La **verifica di tipo LOGICO** che determina il verificarsi o meno di un EVENTO è detta **CONDIZIONE**.

OSSERVAZIONE

Ogni CONDIZIONE è associata ad una TRANSIZIONE.

Ogni CONDIZIONE è espressa sotto forma di una **FUNZIONE LOGICA**, ovvero tramite una **ESPRESSIONE BOOLEANA** che può essere **VERA o FALSA**.

Rappresentazione Grafica

Definita la parte del sistema complesso di cui si vuole PROGETTARE l'evoluzione LOGICO-SEQUENZIALE, si possono rappresentare gli STATI, le AZIONI e le TRANSIZIONI per mezzo di opportuni DIAGRAMMI (SFC o UML)

DIAGRAMMA SFC

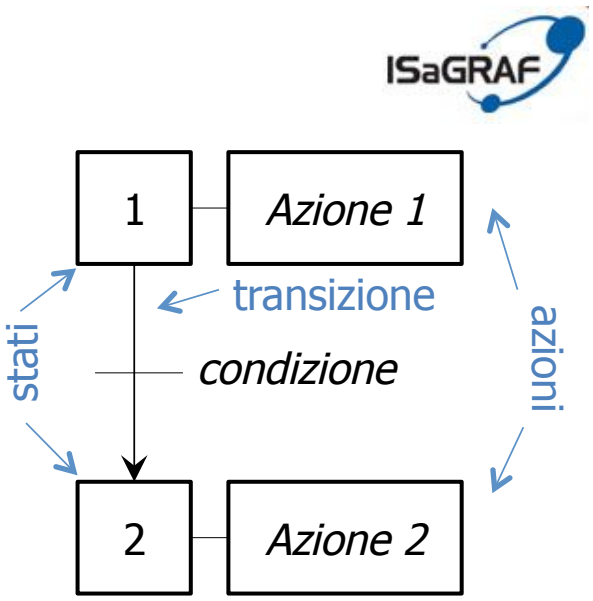


DIAGRAMMA UML DEGLI STATI (VISIO)

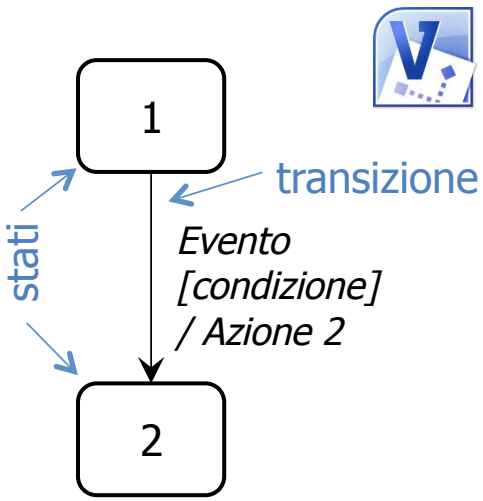
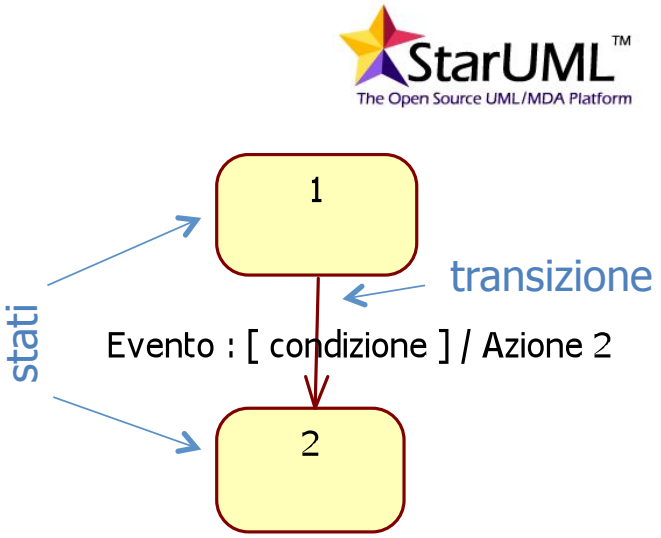


DIAGRAMMA UML DEGLI STATI (StarUML)



Rappresentazione Grafica

Da ora in avanti, SENZA PERDITA DI GENERALITÀ, faremo sempre riferimento alla notazione grafica dei DIAGRAMMI SFC.

Quando necessario, faremo vedere il parallelo con i DIAGRAMMI UML.

DIAGRAMMA SFC

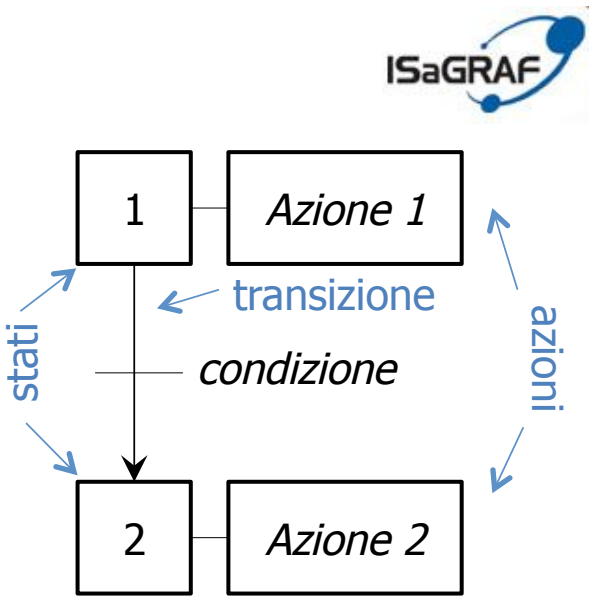


DIAGRAMMA UML DEGLI STATI (VISIO)

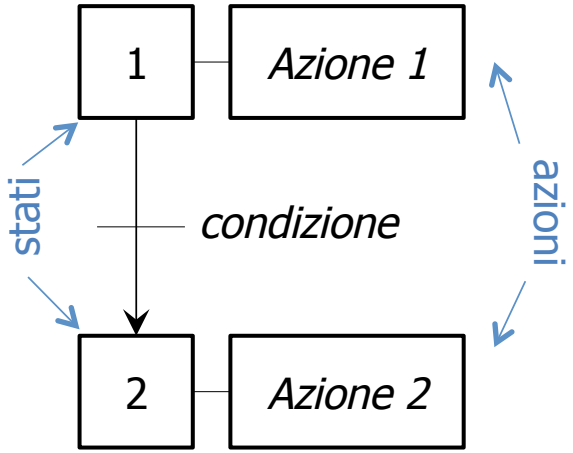
DIAGRAMMA UML DEGLI STATI (StarUML)

Rappresentazione Grafica

Rappresentazione degli STATI

Ogni STATO è rappresentato da un **RETTANGOLO** e deve essere **IDENTIFICATO** con un **nome UNIVOCO** e possibilmente **SIGNIFICATIVO**.

Nell'esempio in basso e nel seguito si utilizzeranno i **NUMERI per identificare gli STATI**, ma solo per motivi di spazio.



Rappresentazione delle AZIONI

Le **AZIONI associate ad uno STATO** sono rappresentate da un **RETTANGOLO** connesso a quello dello stato.

In esso vengono descritte (per esteso, in forma verbosa) le azioni che vengono eseguite in quello specifico stato.

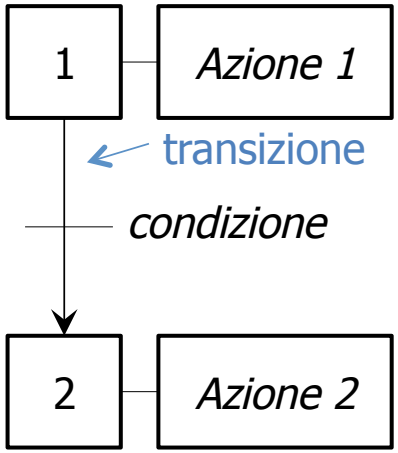
Rappresentazione Grafica

Rappresentazione delle TRANSIZIONI

Ogni **TRANSIZIONE** da uno stato precedente ad uno successivo è rappresentata da un **ARCO ORIENTATO**.

Rappresentazione delle CONDIZIONI

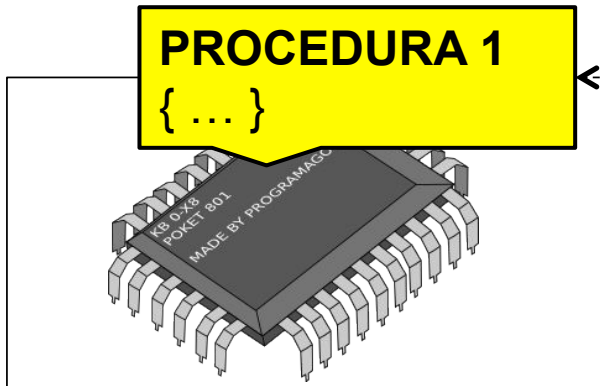
Ogni transizione deve essere **IDENTIFICATA dalla CONDIZIONE** che la determina ed è rappresentata da una **linea che taglia l'arco orientato**.

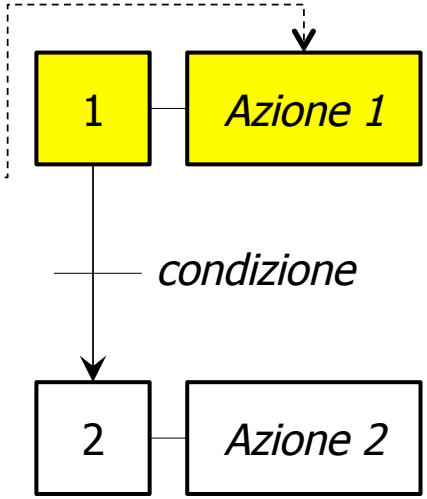


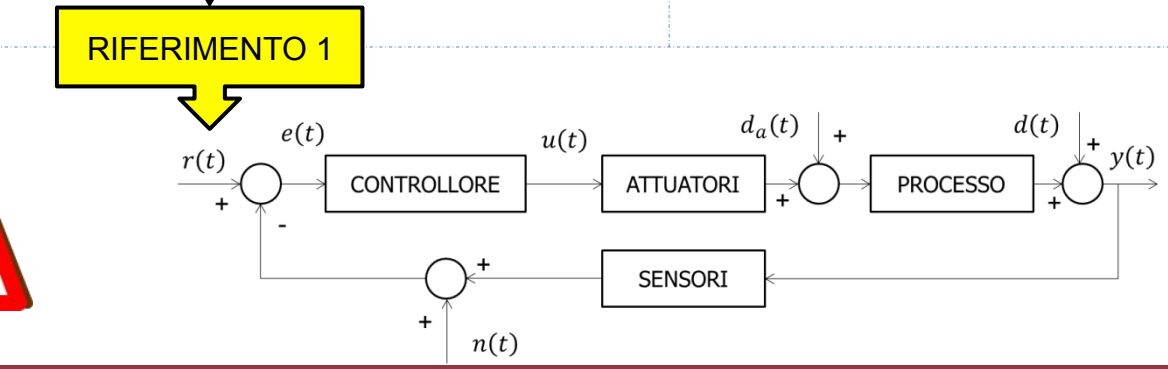
OSSERVAZIONE

Due stati NON POSSONO essere CONNESSI DIRETTAMENTE tra di loro, ma sempre attraverso una TRANSIZIONE che può avvenire a seguito di una CONDIZIONE.

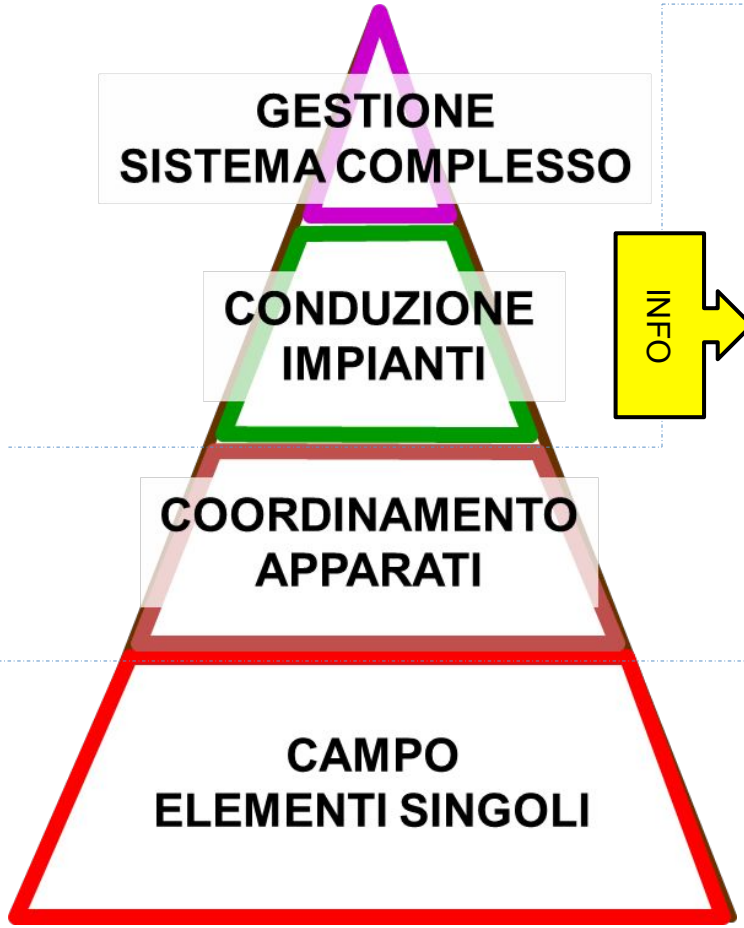
Rapporto tra Simbologia e Realtà

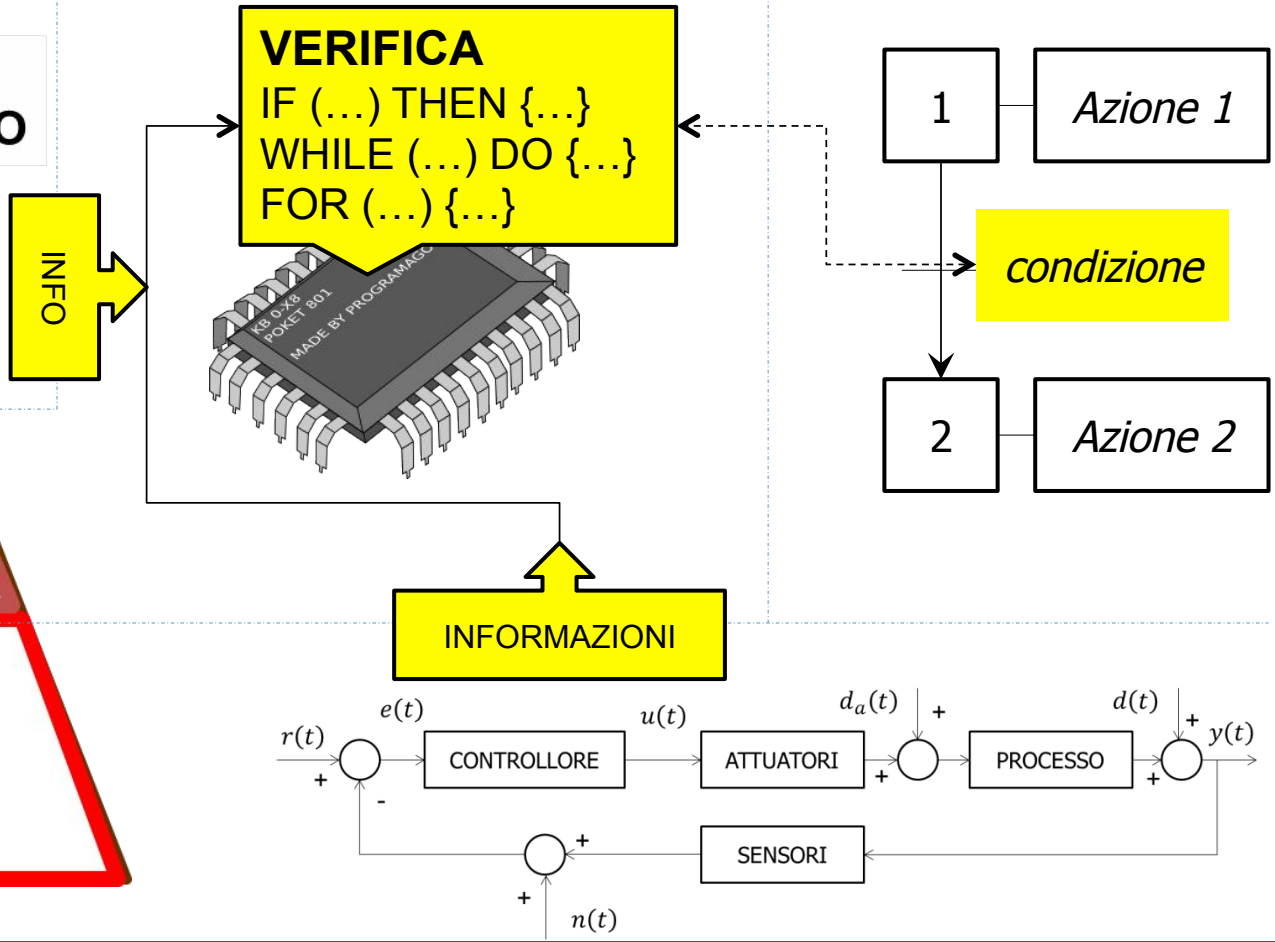




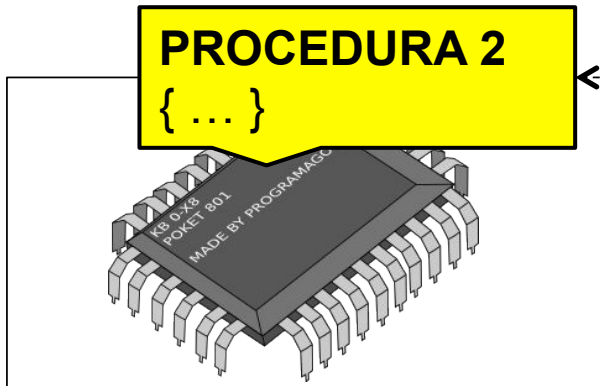


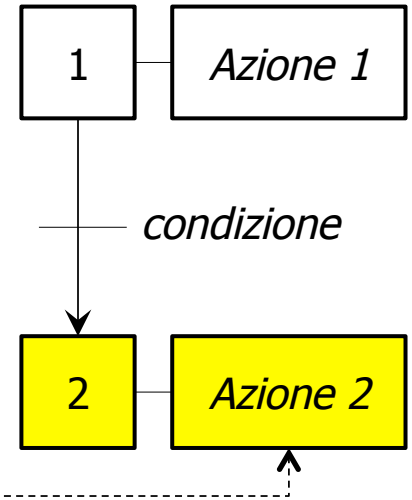
Rapporto tra Simbologia e Realtà

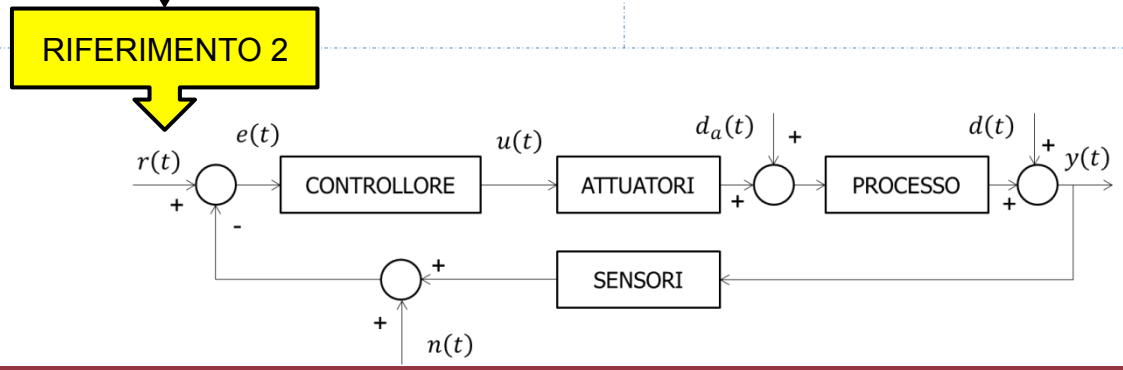




Rapporto tra Simbologia e Realtà







REGOLE DI EVOLUZIONE

STATO INIZIALE

Il primo passo per descrivere l'EVOLUZIONE del PROGRAMMA LOGICO SEQUENZIALE è individuare gli **STATI INIZIALI** che vengono attivati contemporaneamente all'AVVIO DEL SISTEMA.

DIAGRAMMA SFC

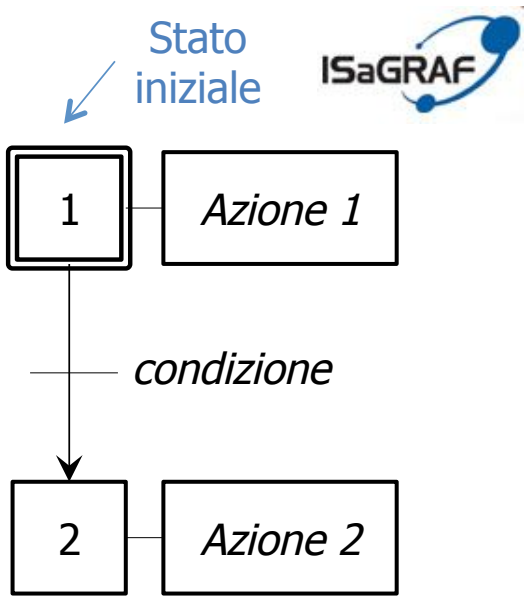


DIAGRAMMA UML DEGLI STATI (VISIO)

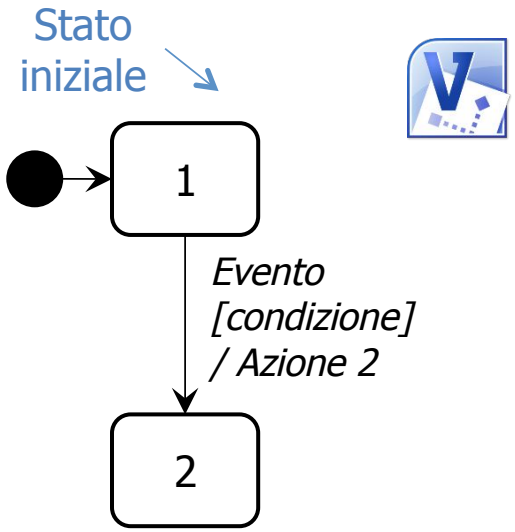
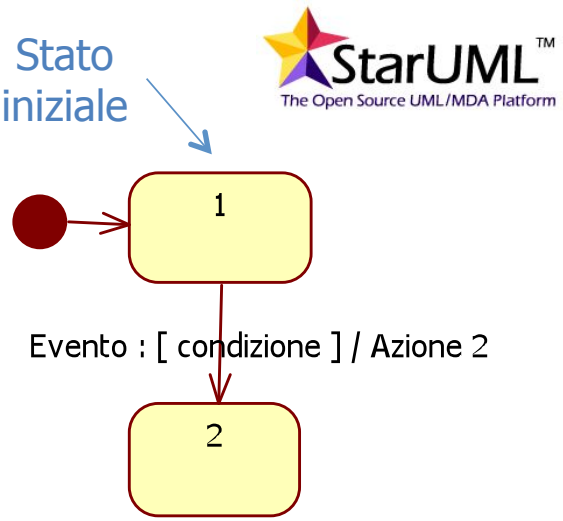


DIAGRAMMA UML DEGLI STATI (StarUML)



EVOLUZIONE

L'**EVOLUZIONE** di un DIAGRAMMA SFC (o degli STATI in UML) è rappresentata dal **PASSAGGIO DA UNO STATO AL SUCCESSIVO**.

Ogni **PASSAGGIO** avviene tramite una **TRANSIZIONE** dallo stato precedente al successivo.

Ogni **TRANSIZIONE** può avvenire (**TRANSIZIONE ATTIVATA**) se e solo se:

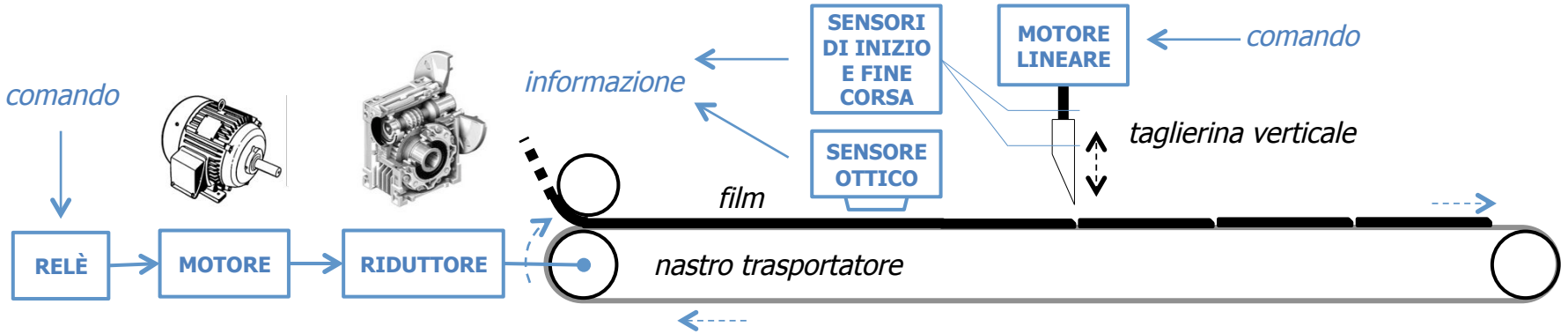
1. Lo **stato precedente è attivo** (**TRANSIZIONE ABILITATA**);
2. La **CONDIZIONE** ad essa associata è **VERIFICATA**.

Quando una TRANSIZIONE viene ATTIVATA, lo **stato precedente viene INTERROTTO** e viene **AVVIATO lo stato successivo**.

ESEMPIO

PROBLEMA

Un dispositivo di automazione industriale è composto da un **nastro trasportatore** su cui è disposto un film omogeneo che deve essere tagliato da una **taglierina** verticale.



Il nastro trasportatore è attuato da un **motore/riduttore** attivato da un **relè**. La taglierina è attuata da un **motore lineare**. Un **sensore ottico** rileva un segno presente sul film indicante quando è necessario tagliare, mentre due **sensori di inizio e fine corsa** indicano la posizione della taglierina.

ESEMPIO cont'd

I **requisiti** funzionali prevedono che:

1) Quando il sensore ottico rileva il segno sul film, deve essere effettuato il taglio;
2) Quando la taglierina esegue il taglio, il nastro trasportatore deve essere fermo;
3) Quando la taglierina viene riportata in condizioni di riposo, il nastro trasportatore può essere riattivato.

Progettare con i digrammi SFC il programma di controllo logico-sequenziale, assumendo che inizialmente la taglierina sia in posizione di riposo e il nastro trasportatore attivo.

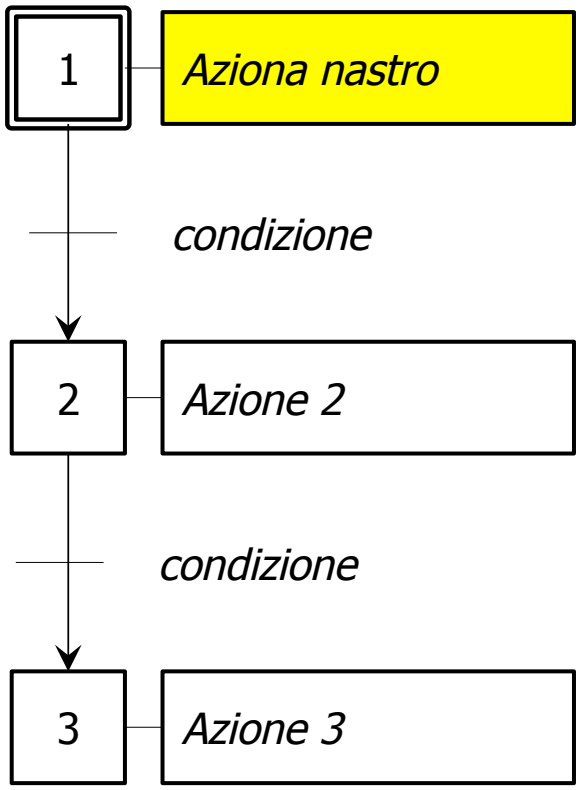
SVOLGIMENTO

Analizzando le condizioni di funzionamento del sistema che rispettano i requisiti funzionali, si evidenziano soltanto 3 stati ammissibili:

1. Nastro trasportatore acceso, taglierina a riposo;
2. Nastro trasportatore fermo, taglierina scende per eseguire il taglio;
3. Nastro trasportatore fermo, taglierina risale.

Lo STATO INIZIALE è chiaramente il primo.

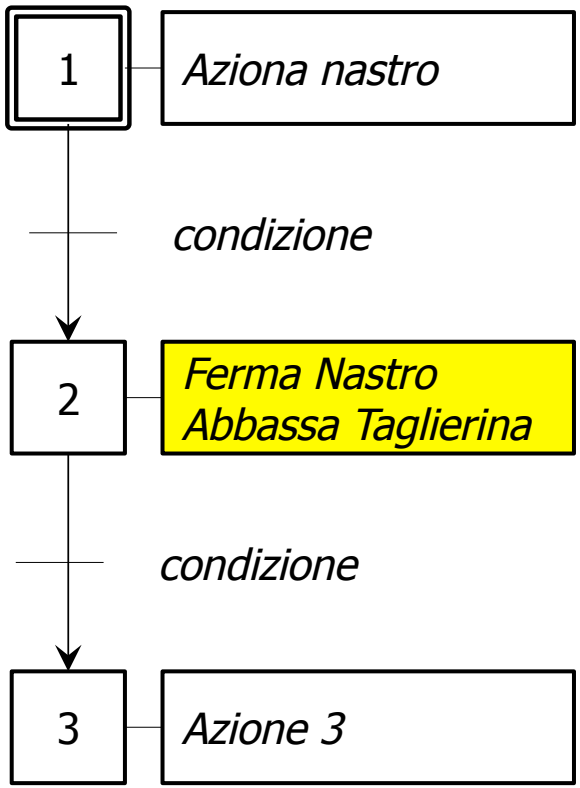
ESEMPIO cont'd



AZIONI

- L'azione svolta nello stato iniziale è semplicemente quella di movimentare il nastro trasportatore.

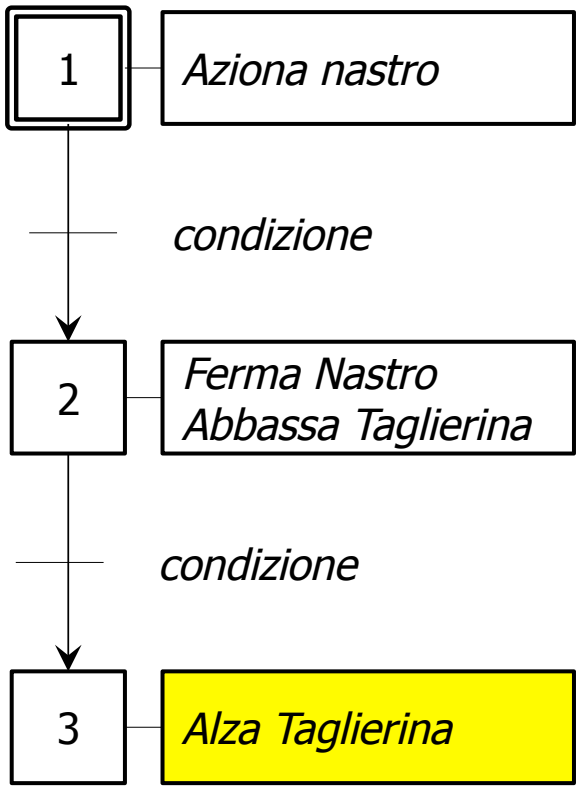
ESEMPIO cont'd



AZIONI

- L'azione svolta nello stato iniziale è semplicemente quella di movimentare il nastro trasportatore.
- Le azioni svolte nello stato 2 sono:
  - Fermare il nastro trasportatore;
  - Abbassare la Taglierina.

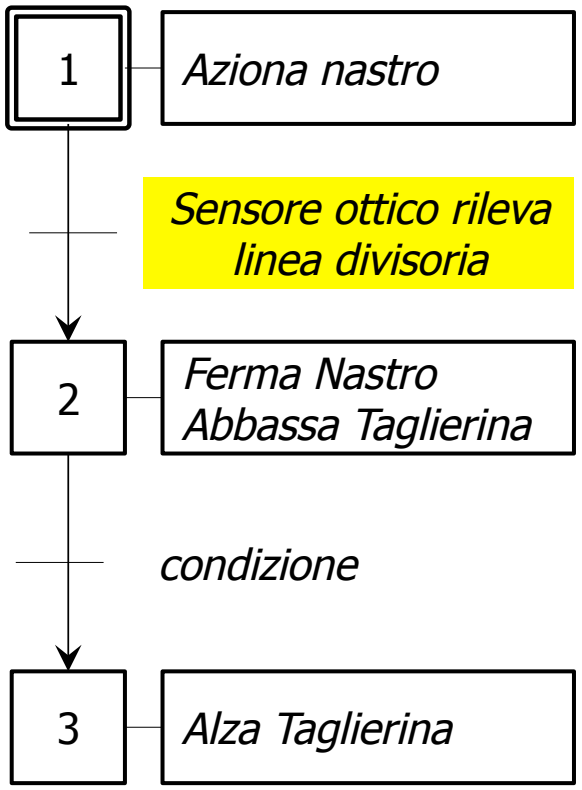
ESEMPIO cont'd



AZIONI

- L'azione svolta nello stato iniziale è semplicemente quella di movimentare il nastro trasportatore.
- Le azioni svolte nello stato 2 sono:
  - Fermare il nastro trasportatore;
  - Abbassare la Taglierina.
- L'azione svolta nello stato 3 è quella di riportare la taglierina in posizione di riposo.

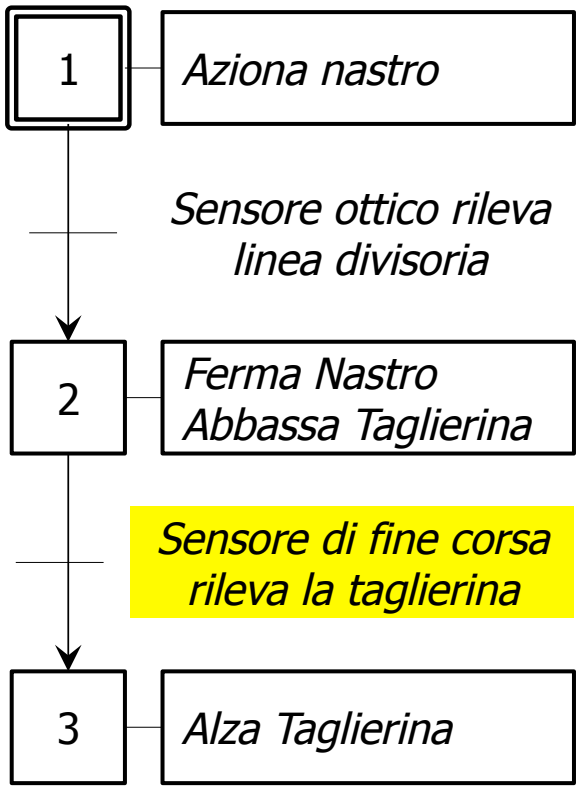
ESEMPIO cont'd



CONDIZIONI

- Quando il SENSORE OTTICO determina la linea divisoria presente sul film, viene VERIFICATA la CONDIZIONE di TRANSIZIONE dallo stato 1 allo stato 2.

ESEMPIO cont'd

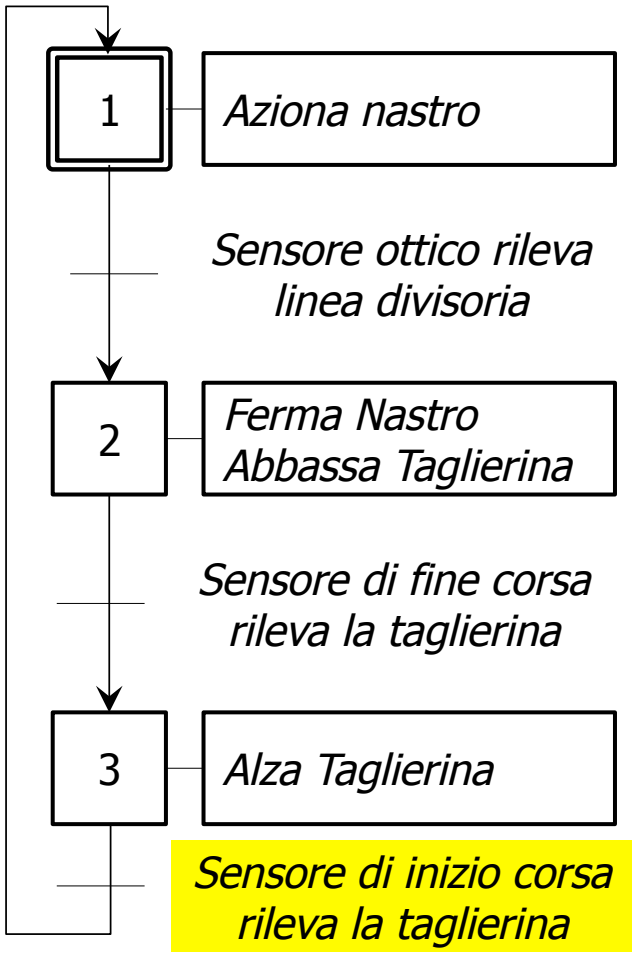


CONDIZIONI

- Quando il **SENSORE OTTICO** determina la linea divisoria presente sul film, viene VERIFICATA la CONDIZIONE di TRANSIZIONE dallo stato 1 allo stato 2.

- Quando il **SENSORE DI FINE CORSA** rileva la taglierina, il taglio del film si può considerare terminato e quindi viene VERIFICATA la CONDIZIONE di TRANSIZIONE dallo stato 2 allo stato 3.

ESEMPIO cont'd



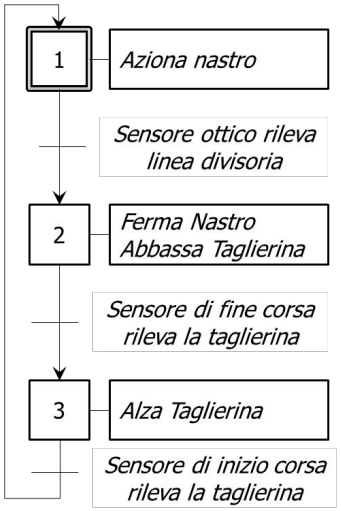
CONDIZIONI

- Quando il **SENSORE OTTICO** determina la linea divisoria presente sul film, viene VERIFICATA la CONDIZIONE di TRANSIZIONE dallo stato 1 allo stato 2.
- Quando il **SENSORE DI FINE CORSA** rileva la taglierina, il taglio del film si può considerare terminato e quindi viene VERIFICATA la CONDIZIONE di TRANSIZIONE dallo stato 2 allo stato 3.
- Quando il **SENSORE DI INIZIO CORSA** rileva la taglierina, si può considerare terminato e quindi viene VERIFICATA la CONDIZIONE di TRANSIZIONE dallo stato 2 allo stato 3.

ESECUZIONE CICLICA

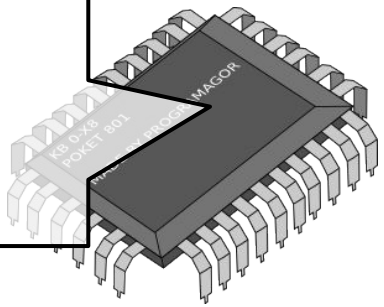
Esecuzione ciclica su PLC

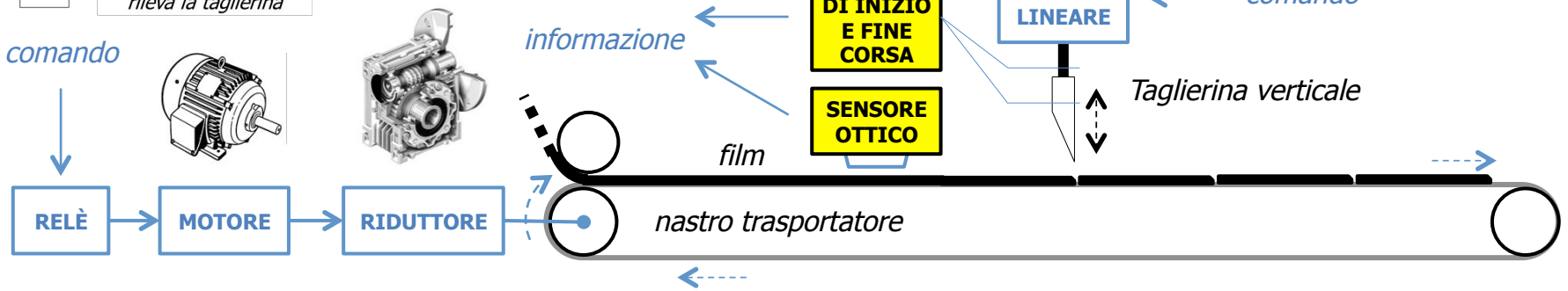
Facendo riferimento all'esempio visto in precedenza, vediamo cosa accade all'interno del PLC durante l'evoluzione del sistema. Possono essere distinte le seguenti fasi:



MACROFASE #1

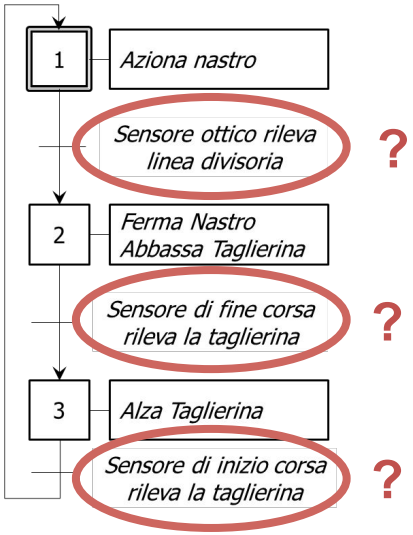
1. Lettura delle informazioni provenienti dai sensori
2. Filtraggio e scalatura dei dati





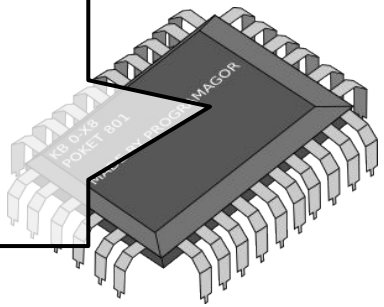
Esecuzione ciclica su PLC

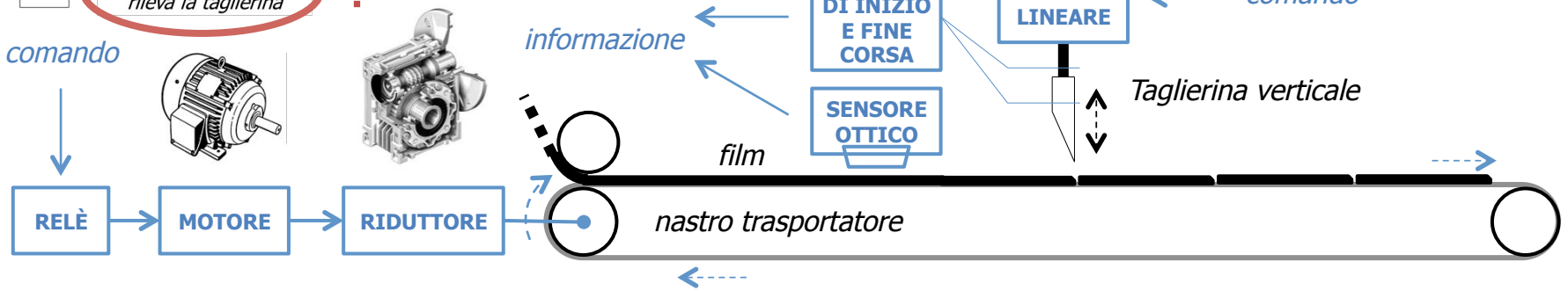
Facendo riferimento all'esempio visto in precedenza, vediamo cosa accade all'interno del PLC durante l'evoluzione del sistema. Possono essere distinte le seguenti fasi:



MACROFASE #1

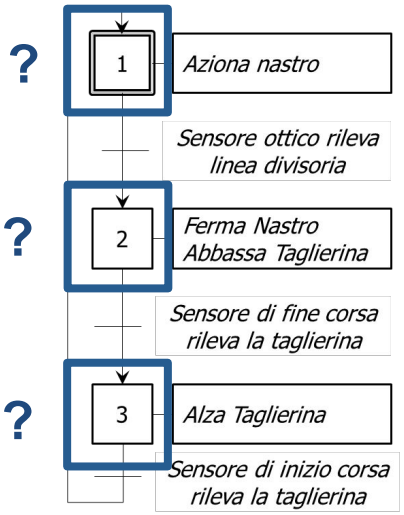
1. Lettura delle informazioni provenienti dai sensori
2. Filtraggio e scalatura dei dati
3. Verifica delle **TRANSIZIONI ATTIVATE**





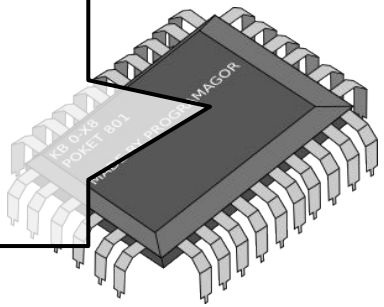
Esecuzione ciclica su PLC

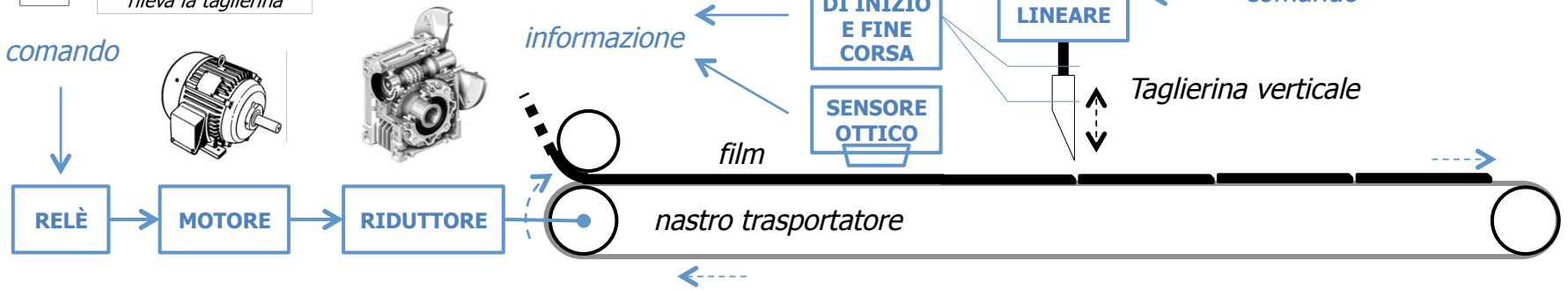
Facendo riferimento all'esempio visto in precedenza, vediamo cosa accade all'interno del PLC durante l'evoluzione del sistema. Possono essere distinte le seguenti fasi:



MACROFASE #1

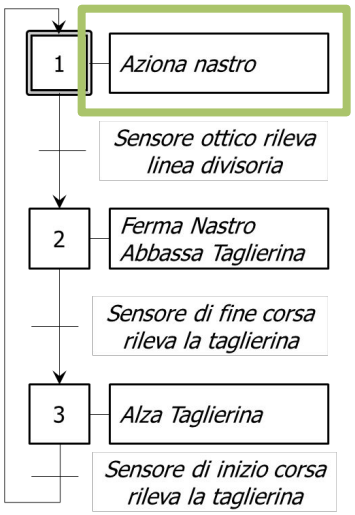
1. Lettura delle informazioni provenienti dai sensori
2. Filtraggio e scalatura dei dati
3. Verifica delle TRANSIZIONI ATTIVATE
4. Identificazione dei **NUOVI STATI ATTIVI**





Esecuzione ciclica su PLC

Facendo riferimento all'esempio visto in precedenza, vediamo cosa accade all'interno del PLC durante l'evoluzione del sistema. Possono essere distinte le seguenti fasi:

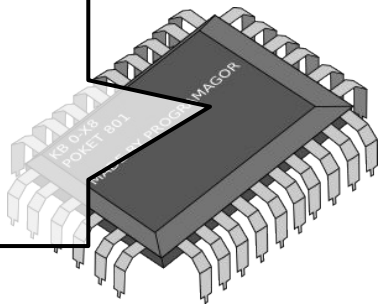


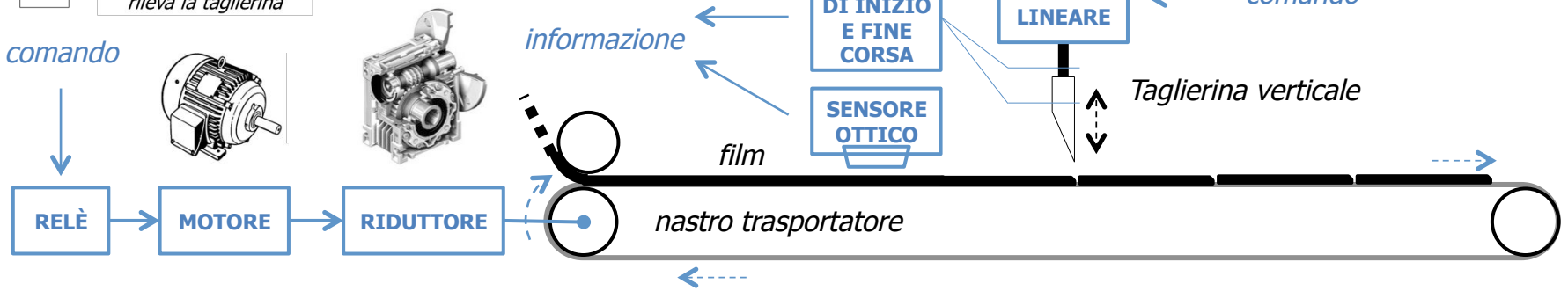
!!!

MACROFASE #1

MACROFASE #2

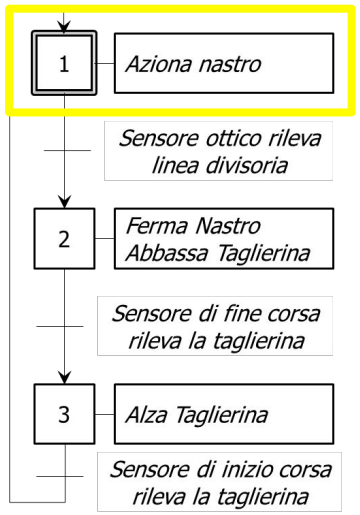
1. **ESECUZIONE DELLE AZIONI** associate agli STATI ATTIVI





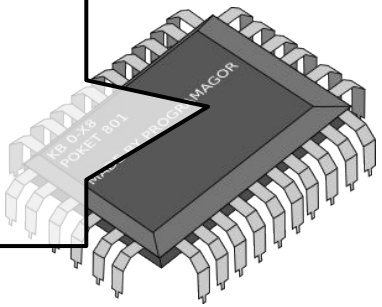
Esecuzione ciclica su PLC

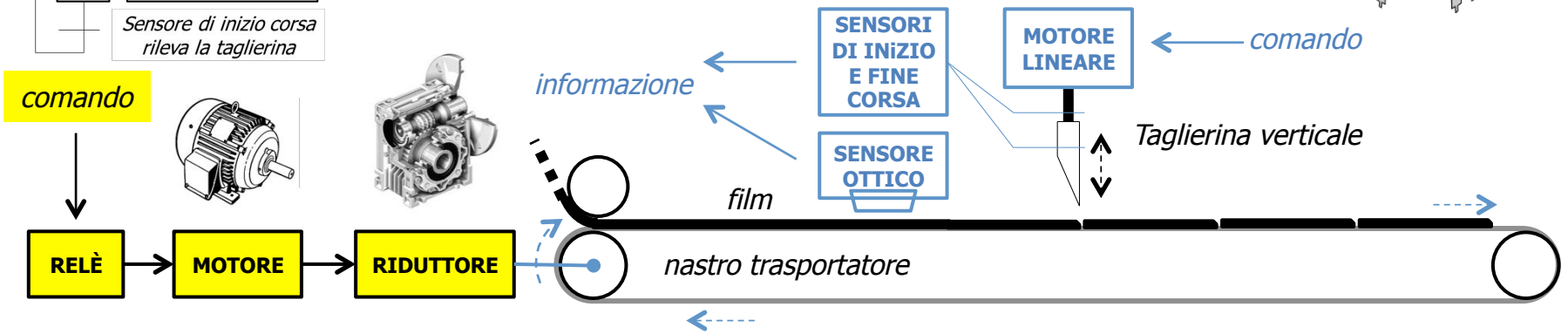
Facendo riferimento all'esempio visto in precedenza, vediamo cosa accade all'interno del PLC durante l'evoluzione del sistema. Possono essere distinte le seguenti fasi:



MACROFASE #1
MACROFASE #2
MACROFASE #3

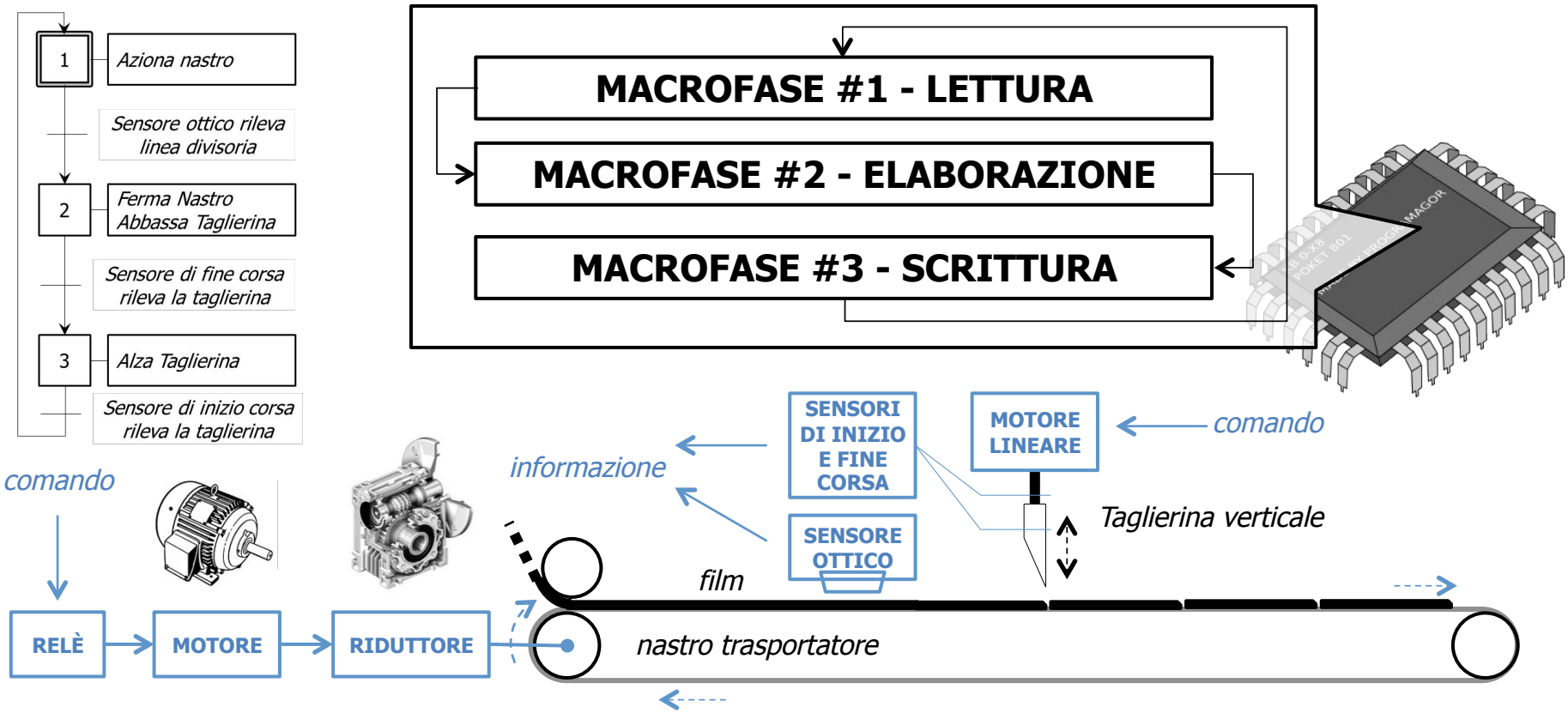
1. Verifica coerenza uscite
2. Scalatura dei dati
3. ATTUAZIONE dei comandi





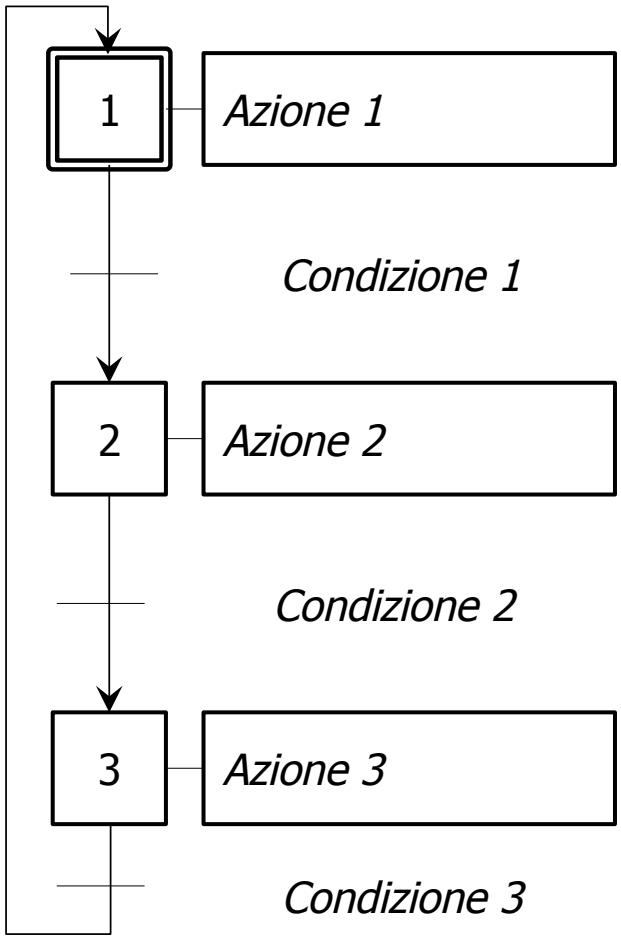
Esecuzione ciclica su PLC

Le tre MACROFASI vengono eseguite CICLICAMENTE dal PLC.



RISOLUZIONE AMBIGUITÀ

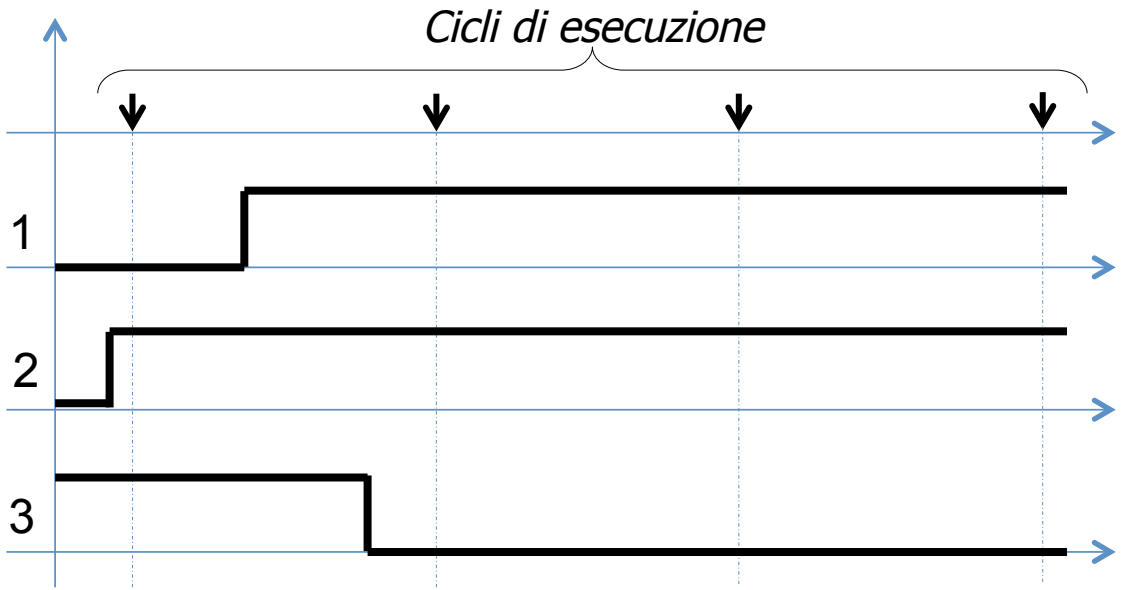
Ambiguità



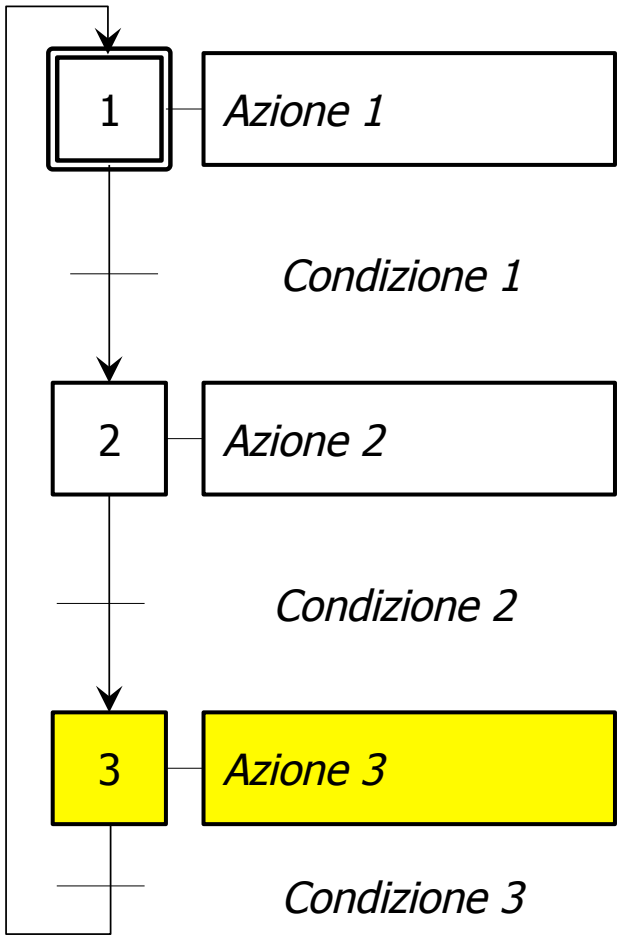
OSSERVAZIONE

Cosa succede se la CONDIZIONE associata alla TRANSIZIONE in USCITA di uno STATO appena ATTIVATO è già VERA?

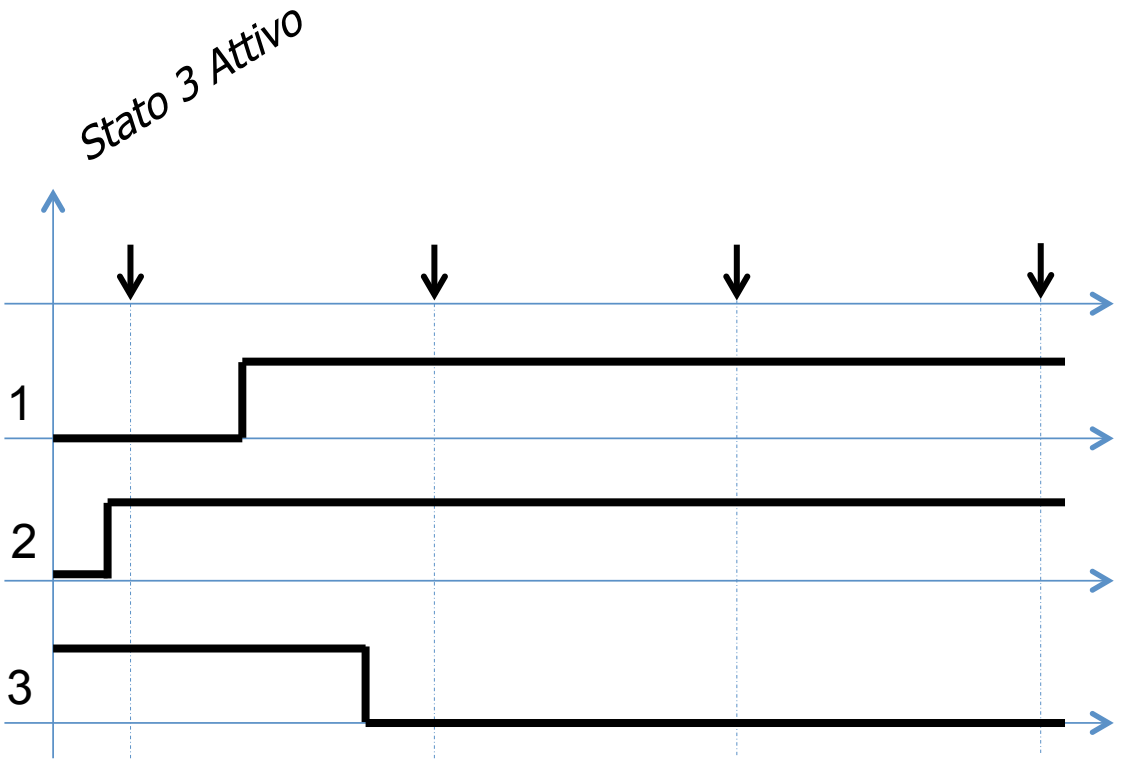
Dati i seguenti andamenti delle Condizioni 1-3:



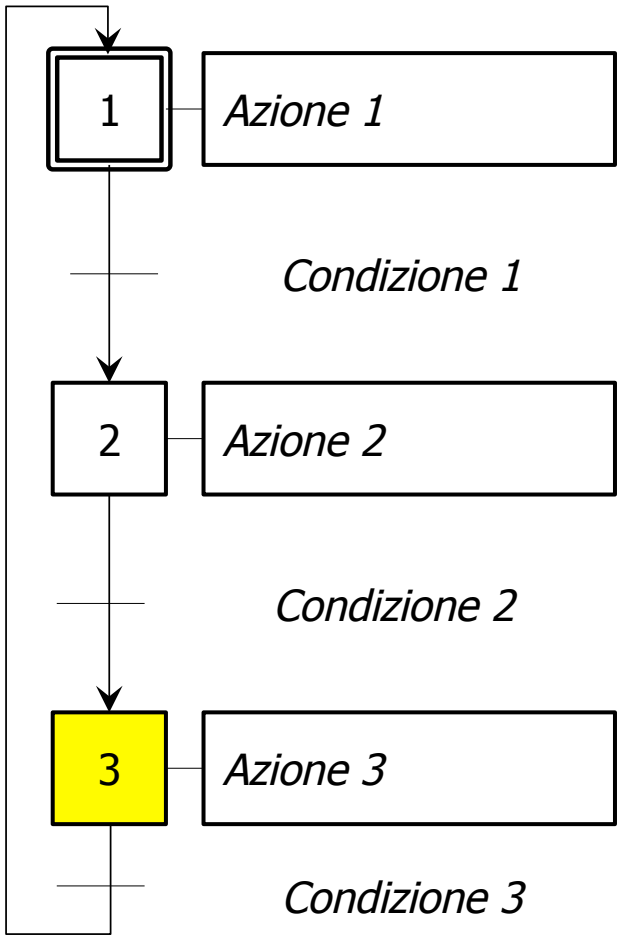
Ambiguità



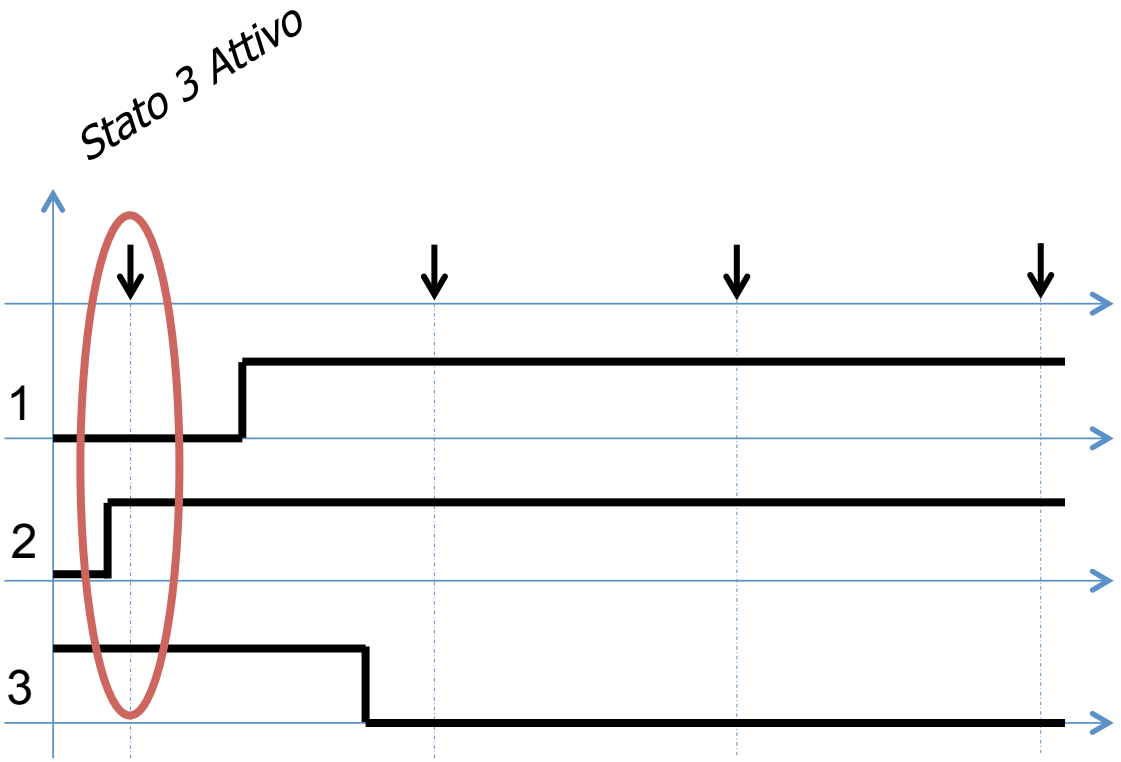
Supponiamo che al tempo t = 0, lo stato 3 sia quello ATTIVO e le relative azioni siano in esecuzione.



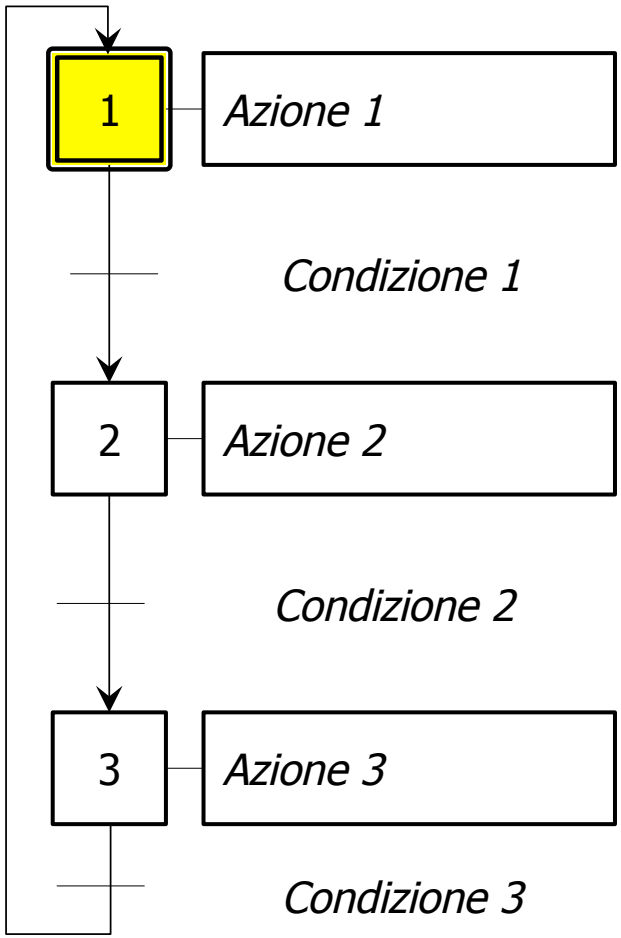
Ambiguità



Al successivo ciclo, il PLC acquisisce i dati dai sensori (MACROFASE #1)

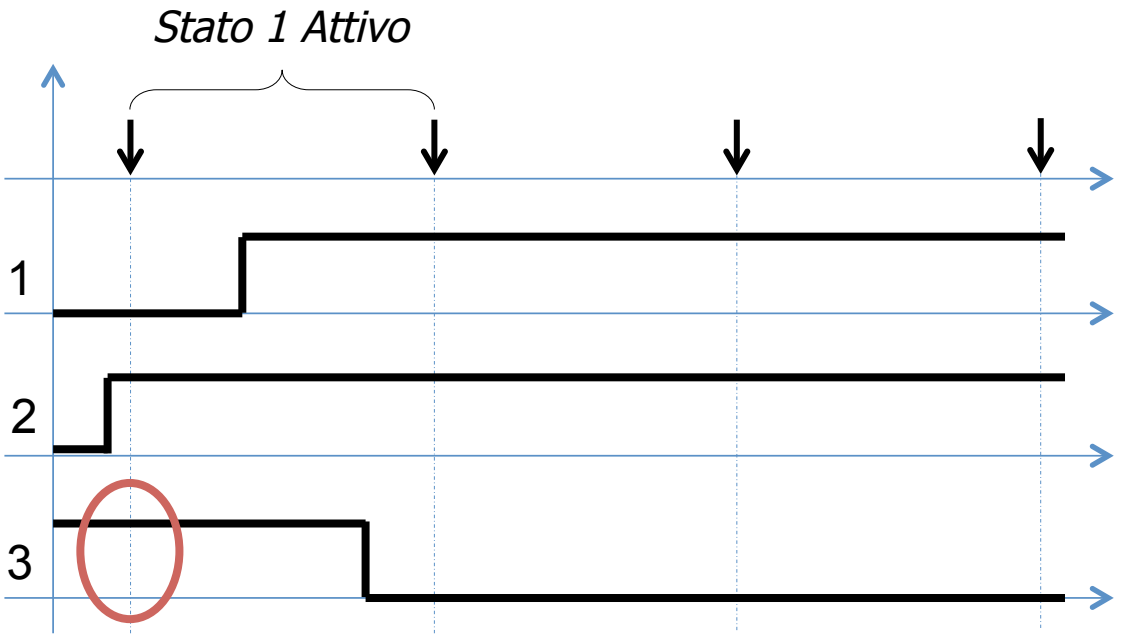


Ambiguità

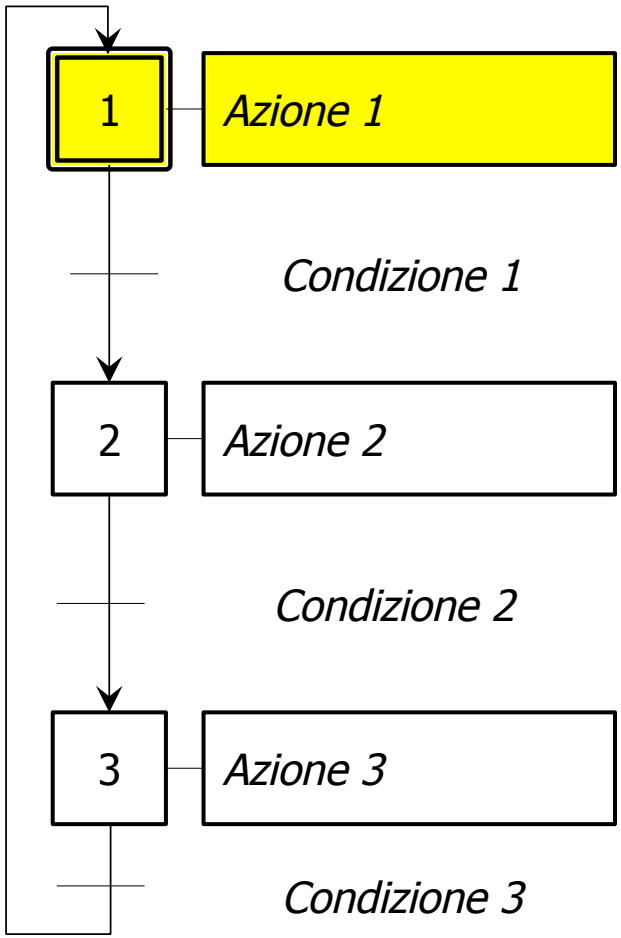


Durante la MACROFASE #1 il PLC verifica che l'unica TRANSIZIONE ABILITATA è la numero 3.

Essendo la condizione associata VERA, la TRANSIZIONE è ATTIVA, e quindi lo STATO 1 diviene ATTIVO.

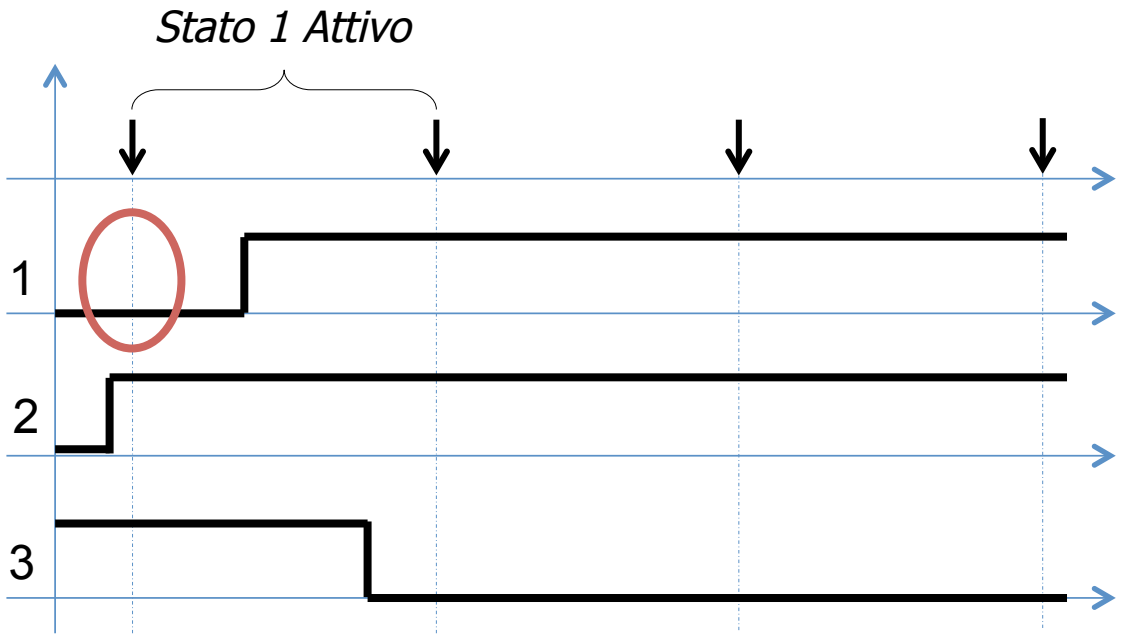


Ambiguità

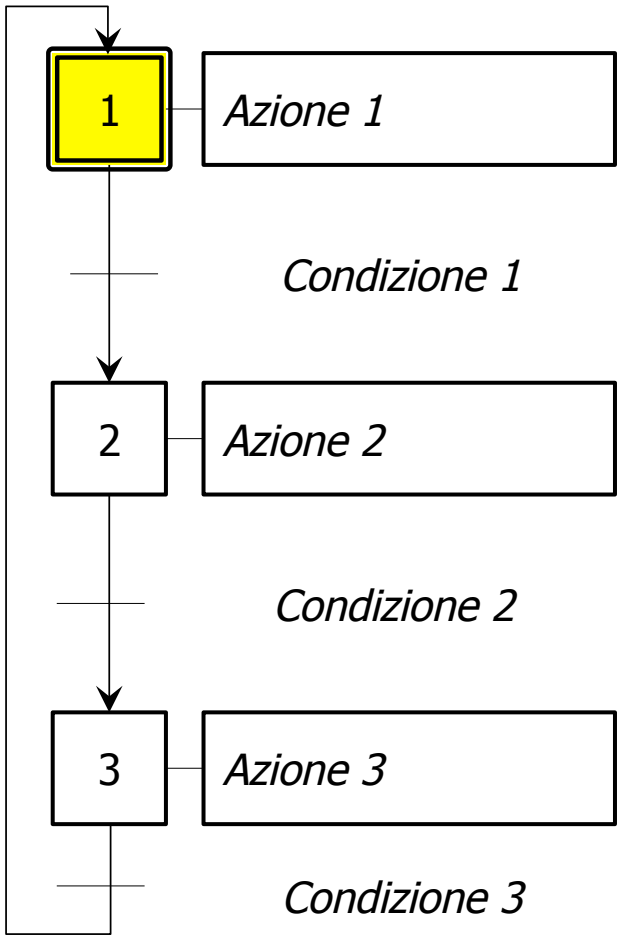


L'Azione 1 è pertanto eseguita (MACROFASE #2 e #3)

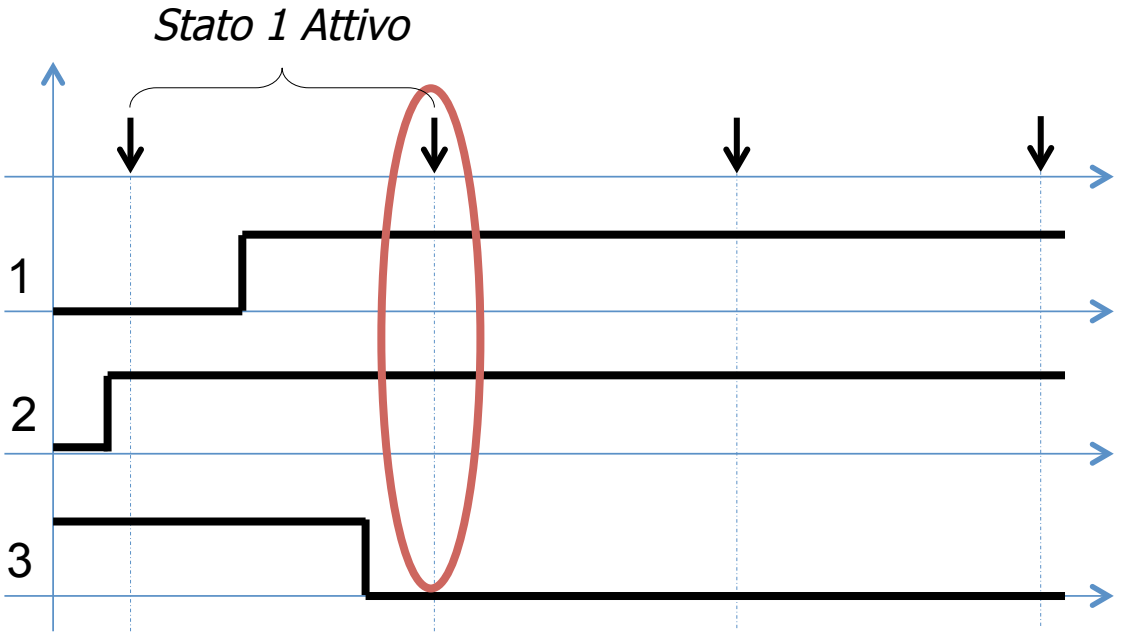
Notiamo che quando lo STATO 1 è stato ATTIVATO, la Condizione 1 NON È VERIFICATA.



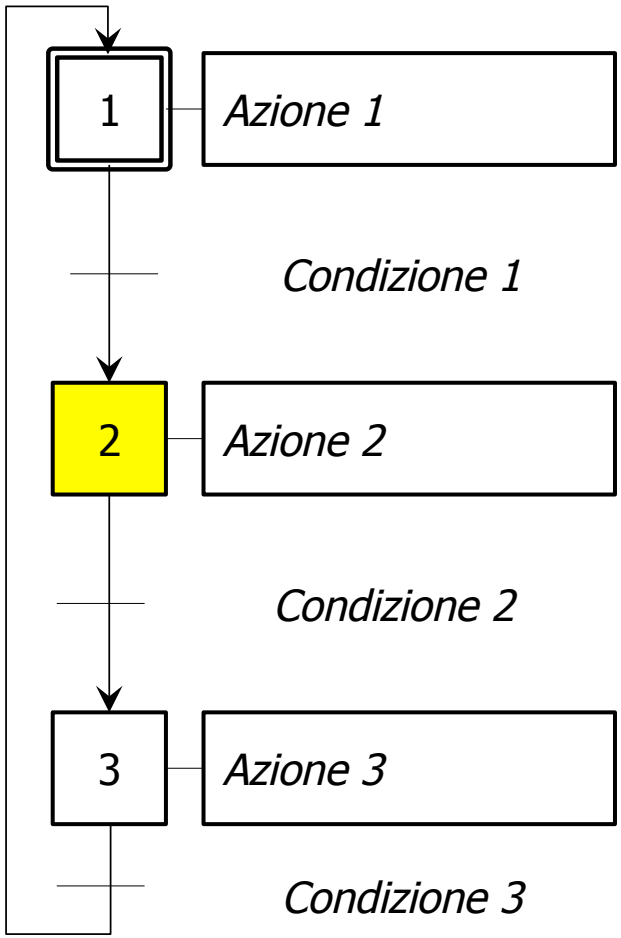
Ambiguità



Al successivo ciclo, il PLC acquisisce i dati dai sensori (MACROFASE #1)

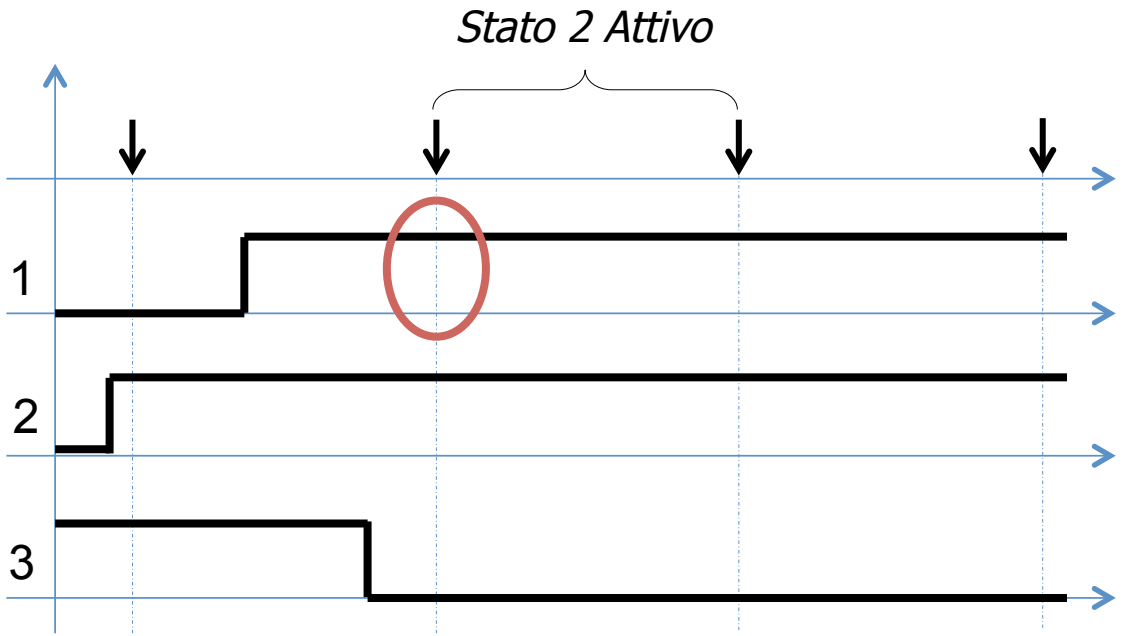


Ambiguità

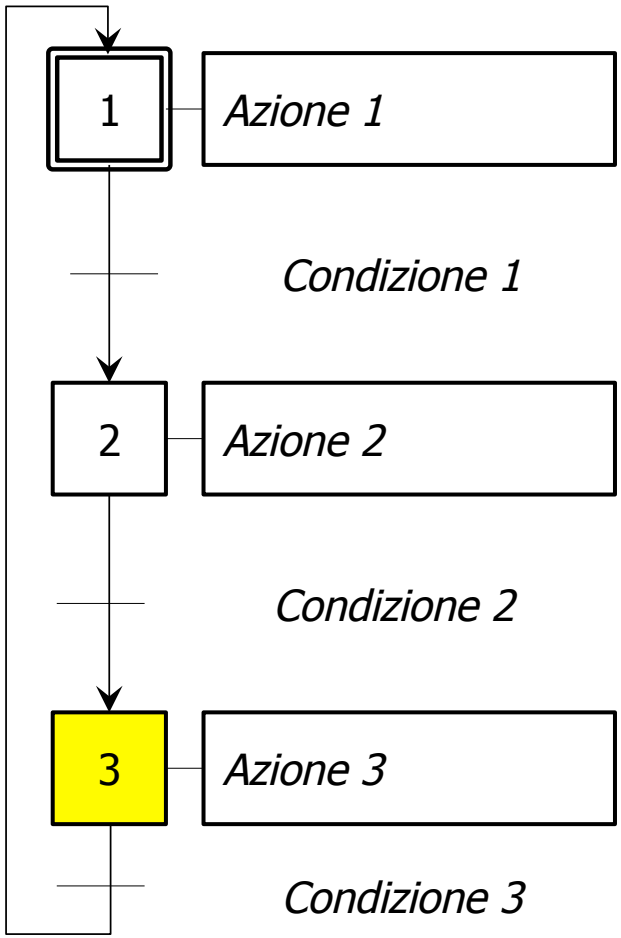


Durante la MACROFASE #1 il PLC verifica che l'unica TRANSIZIONE ABILITATA è la numero 1.

Essendo la condizione associata VERA, la TRANSIZIONE è ATTIVA, e quindi lo STATO 2 diviene ATTIVO.

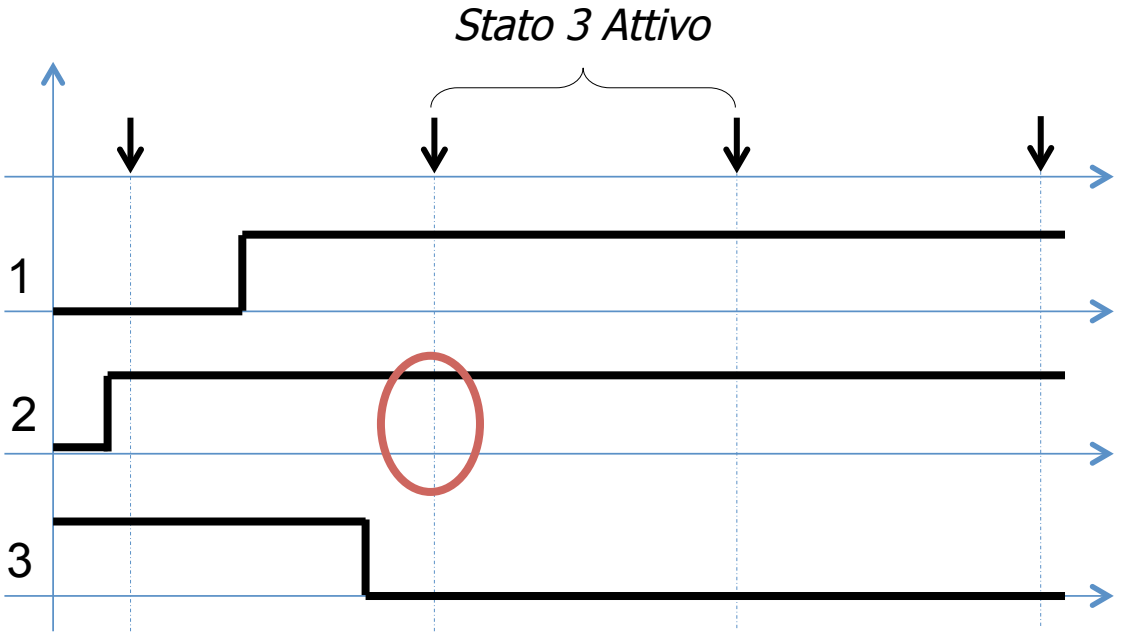


Ambiguità

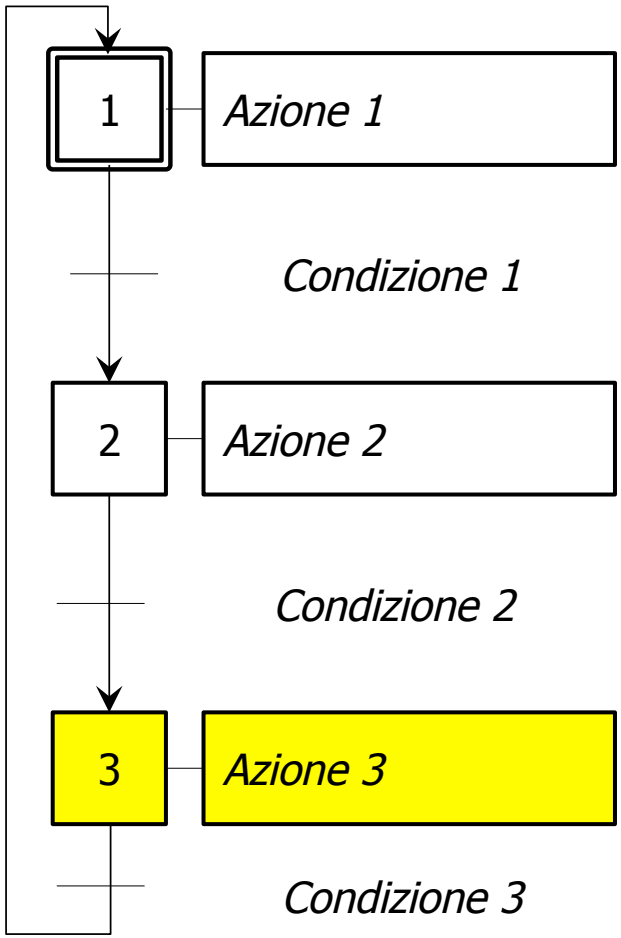


AMBIGUITÀ

Essendo ora lo STATO 2 attivo ed il PLC ancora nella MACROFASE #1, il PLC verifica che la TRANSIZIONE ABILITATA è la numero 2. Essendo la Condizione 2 VERA, la TRANSIZIONE è ATTIVA, e quindi lo STATO 3 diviene ATTIVO!!!

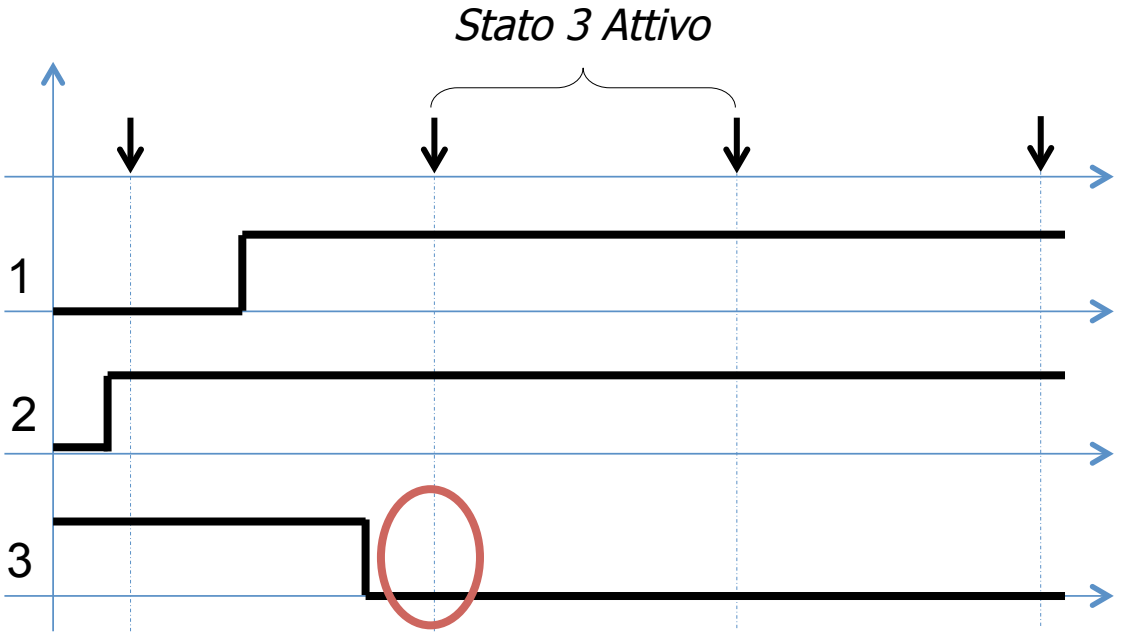


Ambiguità

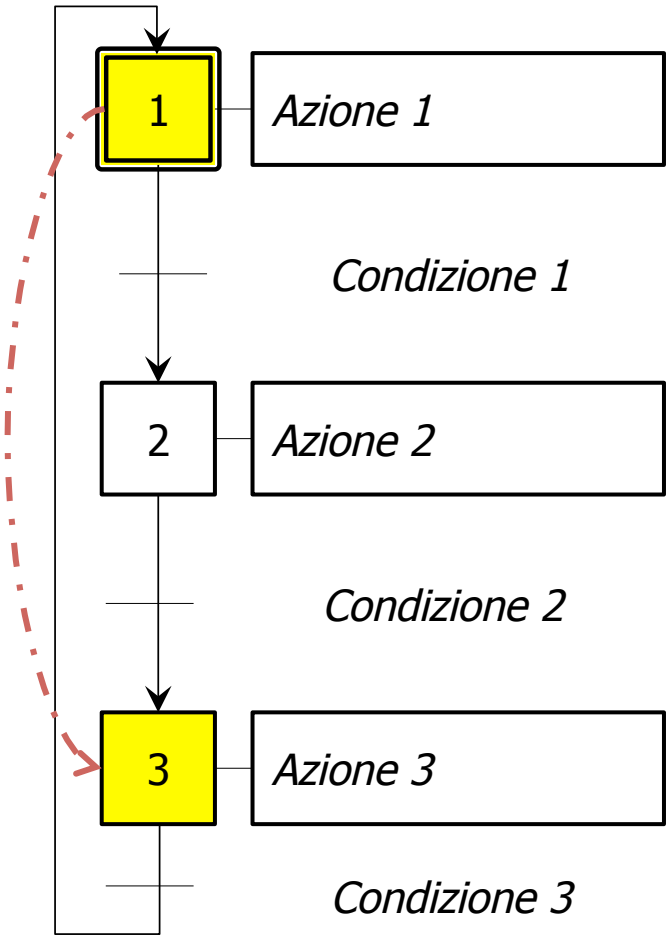


AMBIGUITÀ

Essendo ora lo STATO 3 attivo ed il PLC ancora nella MACROFASE #1, il PLC verifica che la TRANSIZIONE ABILITATA è la numero 3. Essendo la Condizione 3 FALSA, viene ESEGUITA l'azione associata allo stato 3…



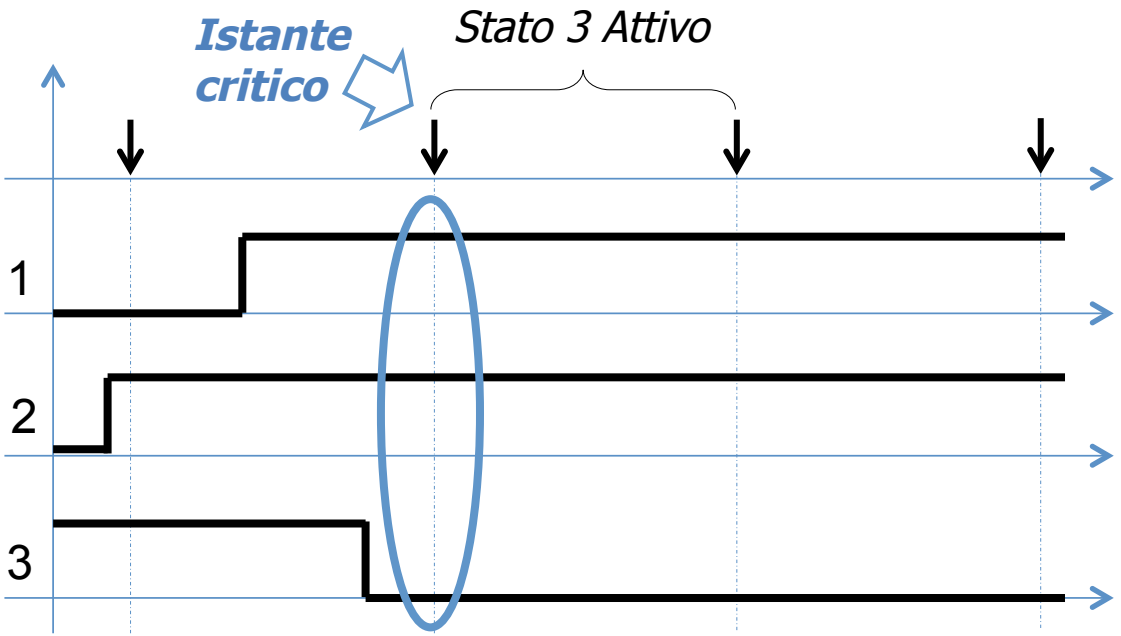
Ambiguità



AMBIGUITÀ

Il risultato è che lo STATO 2 è stato completamente SALTATO durante la MACROFASE #1…

L'istante in cui si generano le ambiguità è detto **istante critico**.



Risoluzione delle ambiguità

La risoluzione di questo tipo di ambiguità è data dalla definizione di una ulteriore REGOLA DI EVOLUZIONE:

Le azioni associate ad un NUOVO STATO ATTIVO devono sempre essere eseguite almeno per un ciclo di funzionamento.

Pertanto le azioni associate ad un qualsiasi STATO la cui transizione in uscita è caratterizzata da una condizione GIÀ VERIFICATA nel momento della sua ATTIVAZIONE, vengono eseguire almeno per un ciclo di funzionamento.

SINTASSI STANDARD DEL LINGUAGGIO SFC

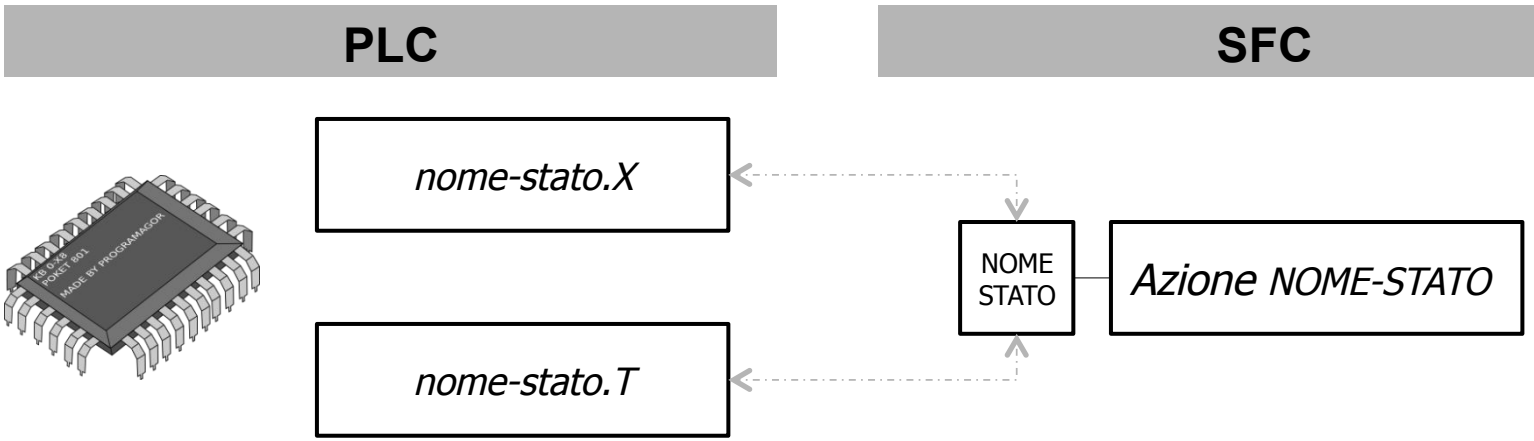
Sintassi degli STATI

Il μprocessore di un PLC definisce 2 VARIABILI per ogni STATO di un diagramma SFC:

1. **MARKER** – Variabile booleana che indica se lo stato è **ATTIVO**
2. **TIMER** – Variabile intera che indica la **DURATA** dell'intervallo **di ATTIVAZIONE**

Dato uno STATO identificato con il nome univoco ***nome-stato***, allora:

- La variabile MARKER è identificata da ***nome-stato.X***
- La variabile TIMER è identificata da ***nome-stato.T***



Variabili di STATO

OSSERVAZIONI SUI MARKER

- All'**avvio** del PLC una variabile MARKER *nome-stato.X* assume il valore VERO (TRUE) se e solo se il corrispondente STATO è uno **STATO INIZIALE**
- Durante un cilco **n-esimo** del PLC, una variabile MARKER *nome-stato.X* assume il valore VERO (TRUE) se e solo se il corrispondente STATO è uno **STATO ATTIVO**

OSSERVAZIONI SUI TIMER

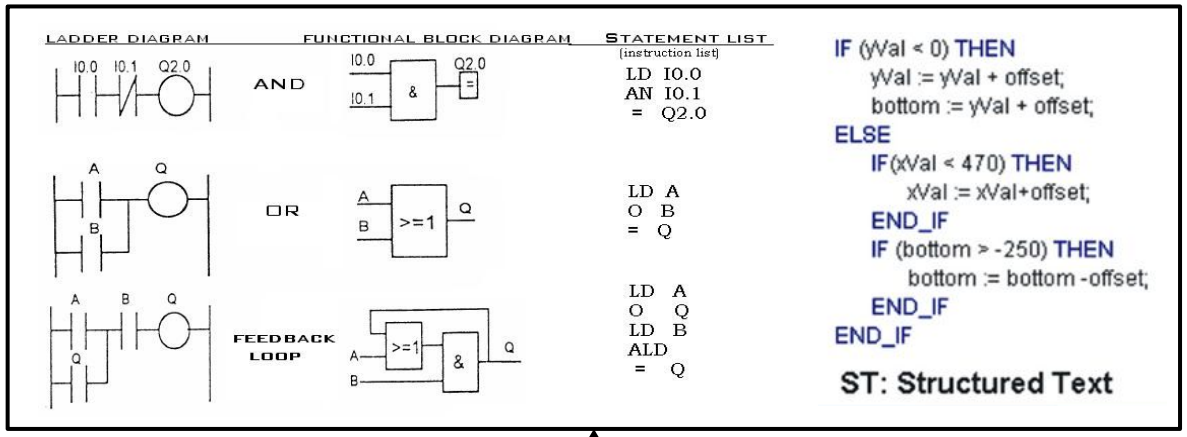
- All'**avvio** del PLC, **tutte** le variabili **TIMER** *nome-stato.T* vengono **azzerat**;
- Durante un cilco **n-esimo** del PLC vengono **incrementate tutte e sole** le variabili TIMER ***nome-stato.T*** tali che ***nome-stato.X = TRUE***
- Durante un cilco **n-esimo** del PLC vengono **azzerate tutte e sole** le variabili TIMER ***nome-stato.T*** degli stati *nome-stato* che vengono **attivati in tale ciclo**, ovvero tali che *nome-stato.X = FALSE* al ciclo (n-1)-esimo *nome-stato.X = TRUE* al ciclo n-esimo
- Le **variabili TIMER** *nome-stato.T* mantengono sempre l'**informazione** sulla **durata dell'ultimo intervallo di attivazione** dello stato *nome-stato*

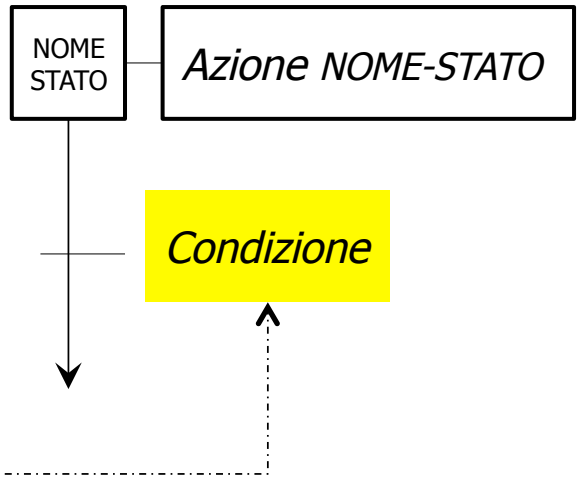
Sintassi delle CONDIZIONI

La **SINTASSI** delle **TRANSIZIONI** e delle **CONDIZIONI** ad esse ASSOCIATE può essere espressa tramite **uno qualsiasi dei seguenti linguaggi di programmazione** definiti dallo standard IEC 61131-3 di tipo GRAFICO o TESTUALE:

- LADDER DIAGRAM (LD) o FUNCTIONAL BLOCK DIAGRAM (FBD)
- STRUCTURED TEXT (ST) o INSTRUCTION LIST (IL)

LD **FBD** **IL** **ST** | **SFC**



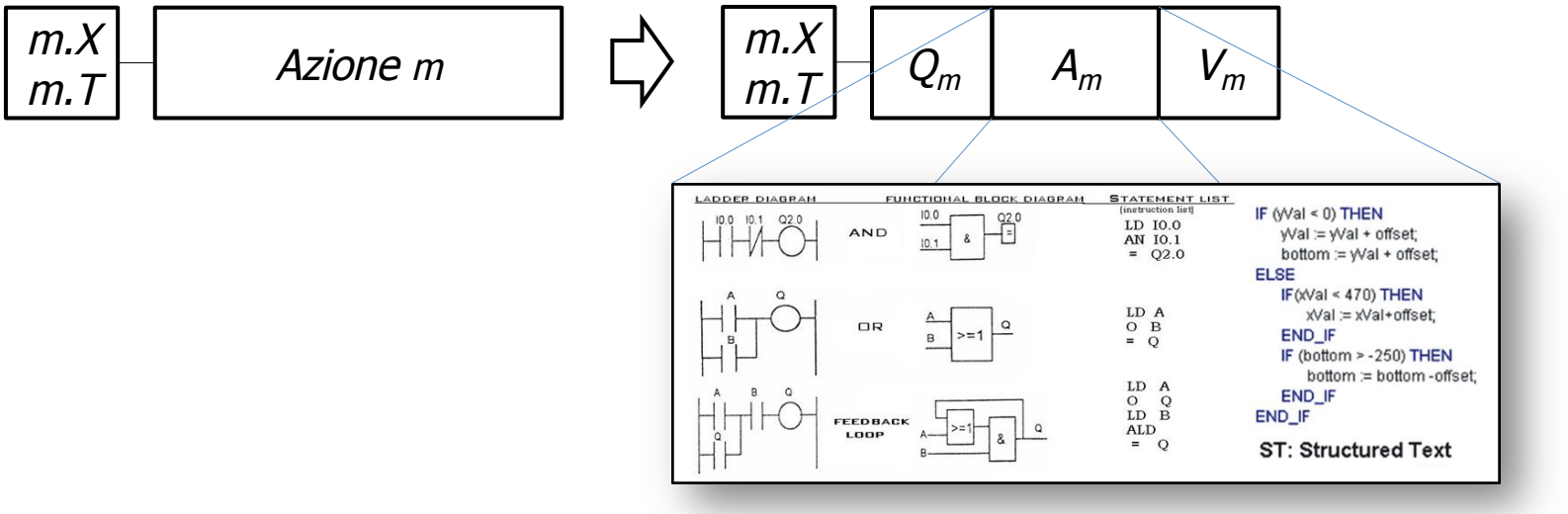


Sintassi delle AZIONI

La SINTASSI di una *AZIONE m* associata ad un generico *STATO m*, è data dalla terna:

- A_m – **IDENTIFICATORE** (UNIVOCO) della azione
- Q_m – **QUALIFICATORE** che definisce la tipologia di azione
- V_m – **VARIABILE** booleana che indica se l'azione è stata terminata

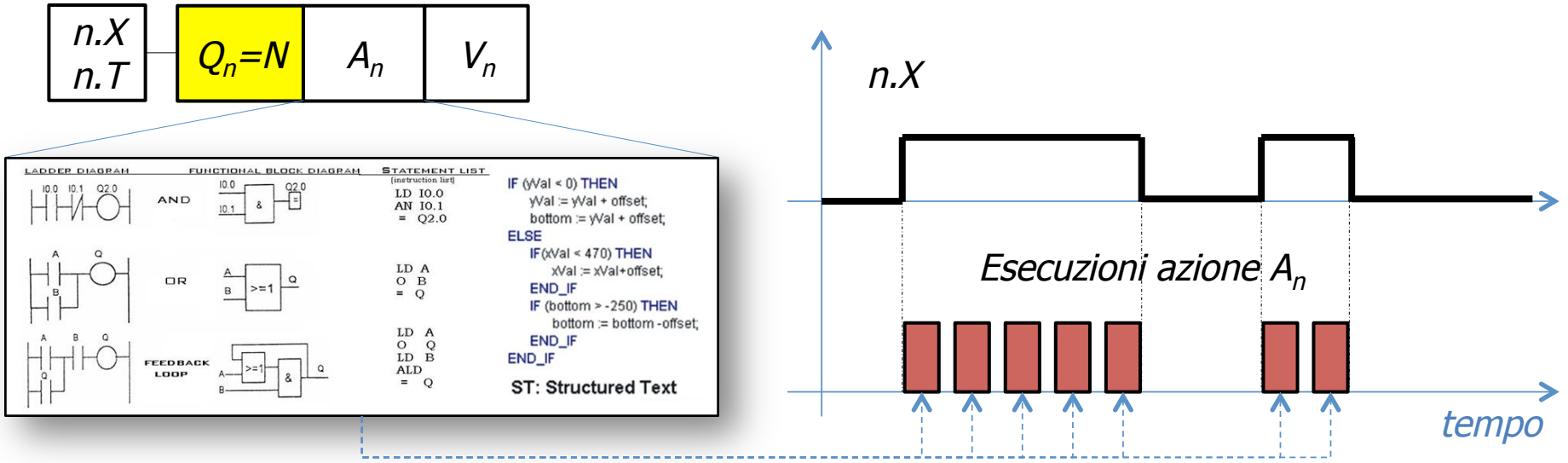
Le azioni A_m devono essere definite usando i linguaggi grafici (LD, FBD) o testuali (ST, IL) direttamente dentro il rettangolo della azione, oppure a parte.



Qualificatore Q_m

AZIONE N (Normal non stored)

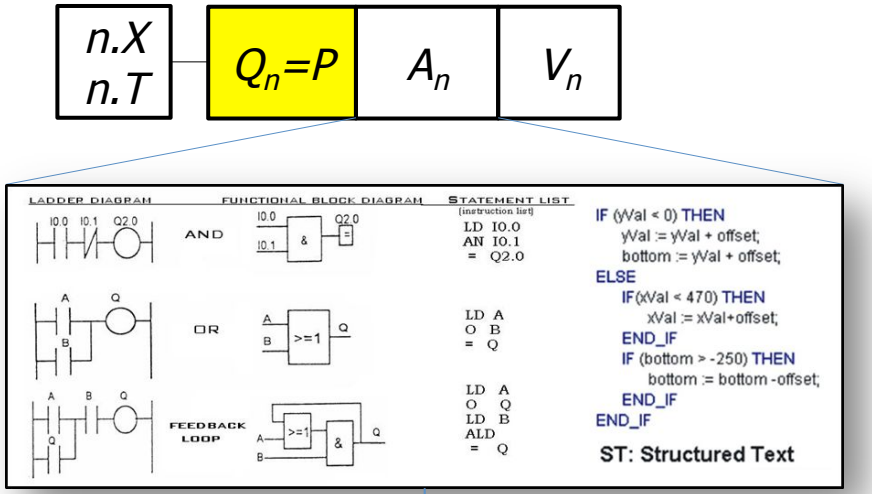
Dato uno stato *n*, Se $Q_n=N$, l'azione A_n viene ripetuta ciclicamente fintanto che lo stato rimane attivo, ovvero fintanto che *n.X* = TRUE.

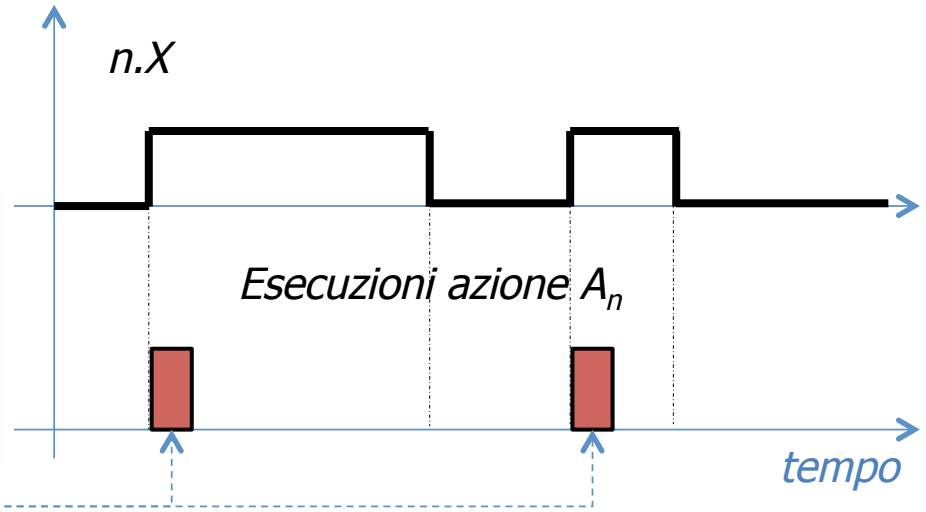


Qualificatore Q_m

AZIONE P (Pulse)

Dato uno stato *n*, Se $Q_n=P$, l'azione A_n viene eseguita UNA SOLA VOLTA fintanto che lo stato rimane attivo, ovvero fintanto che *n.X* = TRUE.

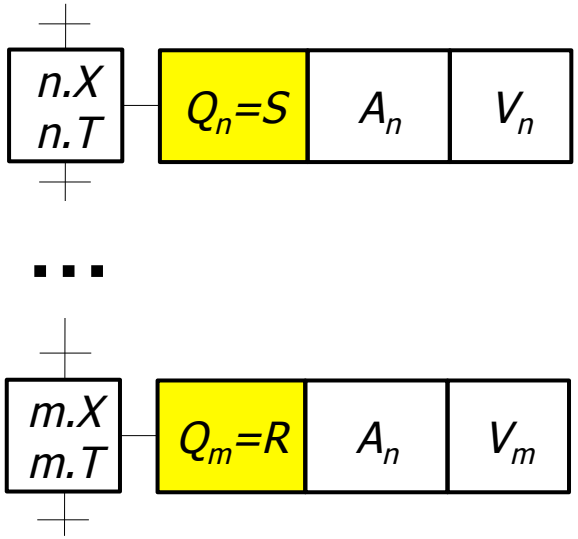


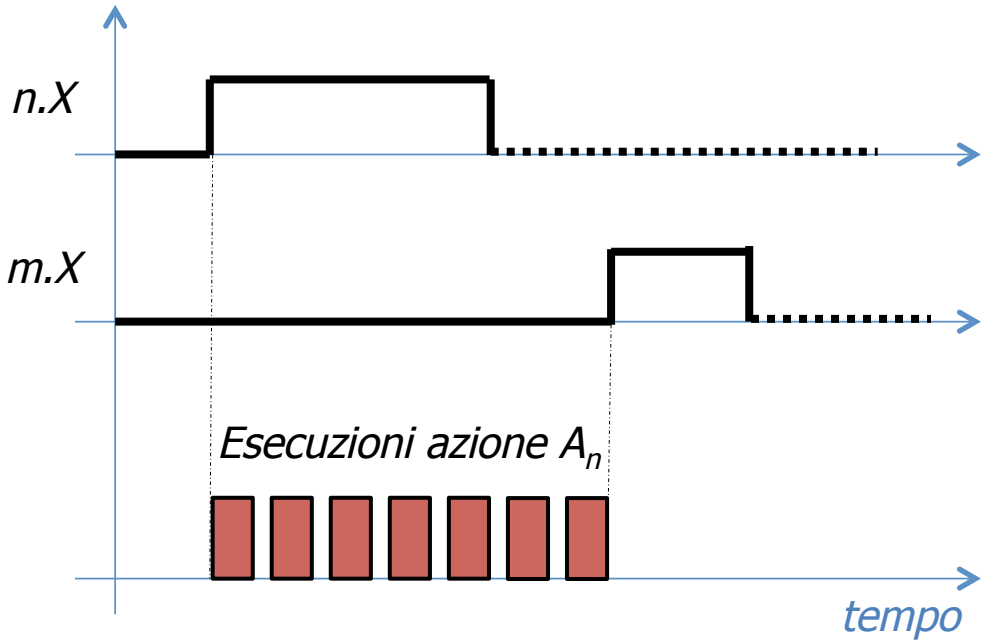


Qualificatore Q_m

AZIONE S (Set)

Dato uno stato *n*, Se $Q_n=S$, l'azione A_n viene ripetuta ciclicamente fino a quando non viene eseguita la stessa azione $A_m = A_n$ ma in uno stato successivo *m* con $Q_m=R$.

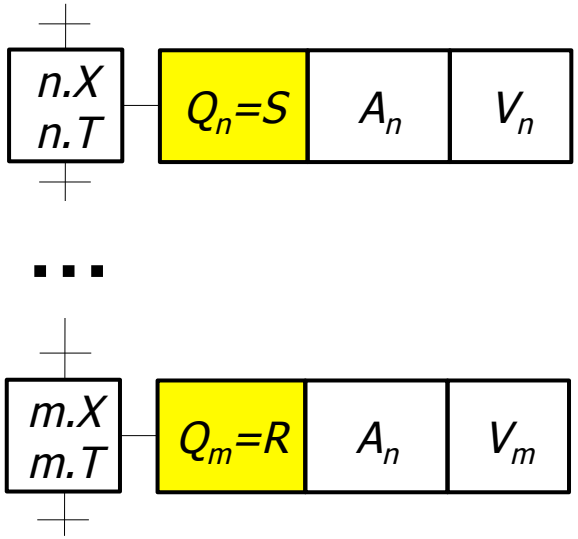


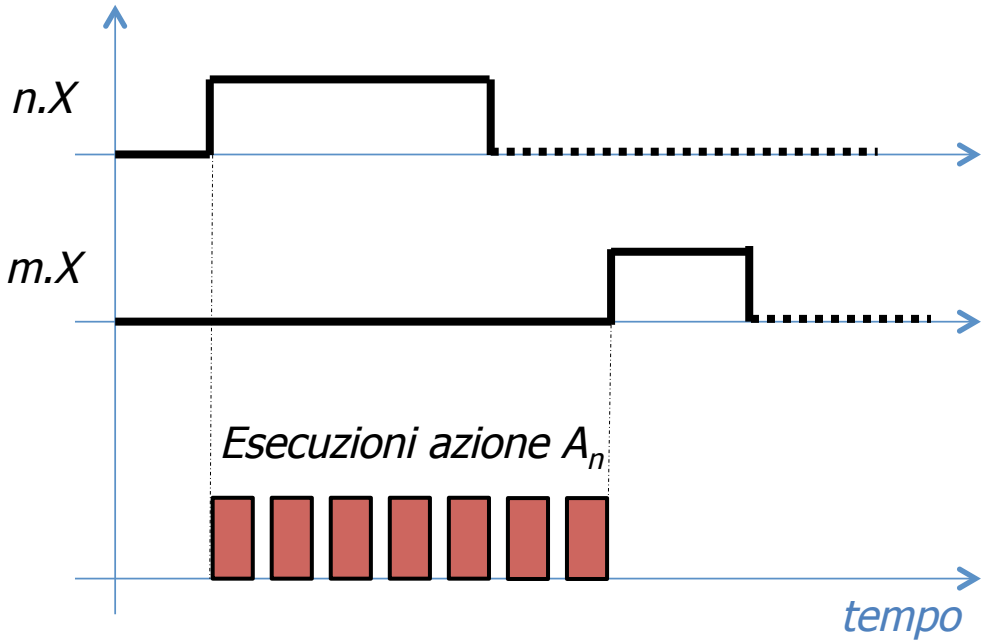


Qualificatore Q_m

AZIONE R (Reset)

Dato uno stato *m*, Se $Q_m=R$, l'azione $A_m = A_n$ precedentemente attivata allo stato *n* da un qualificatore S viene terminata.

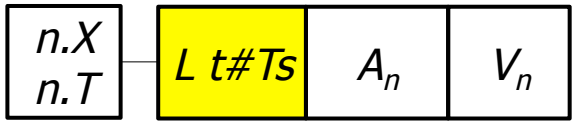


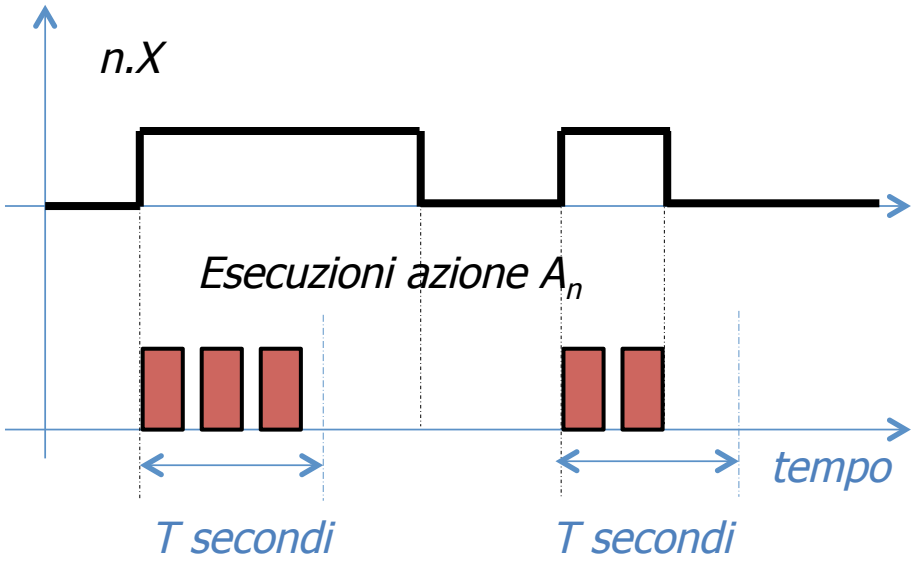


Qualificatore Q_m

AZIONE L (time Limited)

Dato uno stato *n*, Se Q_n=*L t#Ts* , l'azione A_n viene ripetuta ciclicamente fintanto che lo stato rimane attivo (*n.X* = TRUE) e che NON siano trascorsi T secondi.



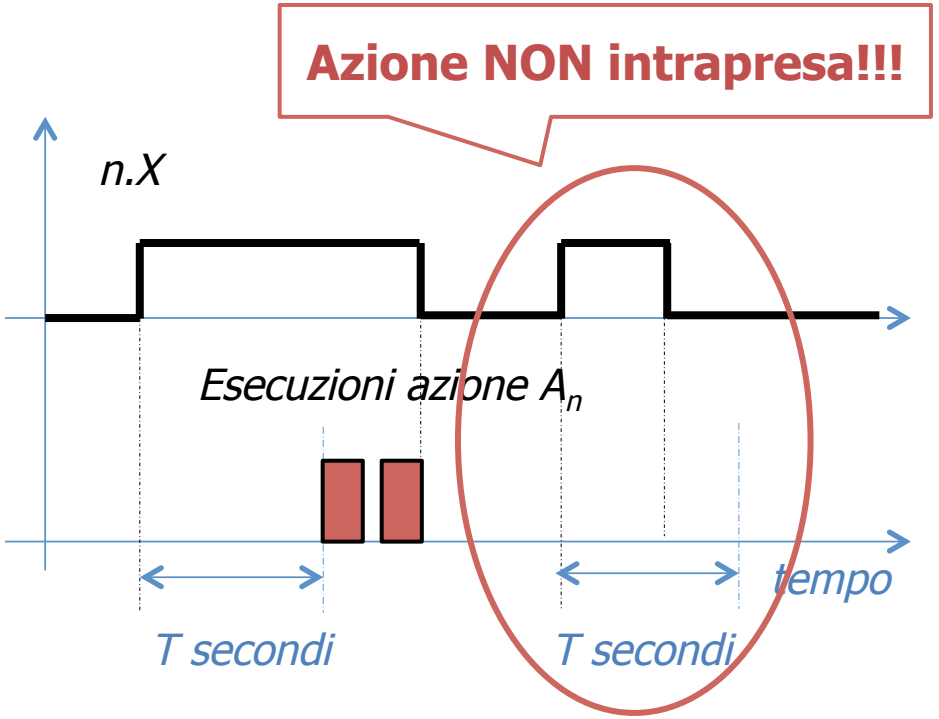


Qualificatore Q_m

AZIONE D (time Delayed)

Dato uno stato *n*, Se Q_n=*D t#Ts* , l'azione A_n viene ripetuta ciclicamente fintanto che lo stato rimane attivo (*n.X* = TRUE) e NON PRIMA che siano trascorsi T secondi.

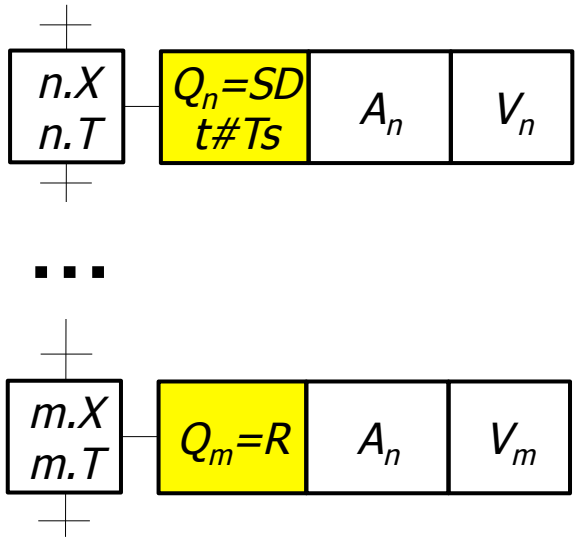
n.X *n.T*	*D t#Ts*	A_n	V_n

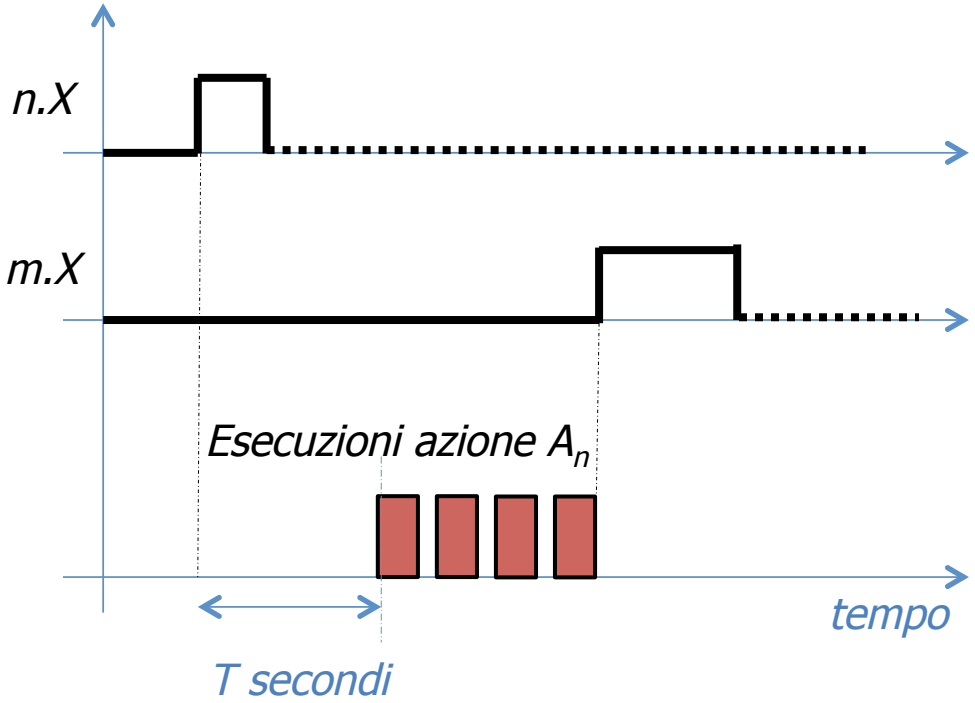


Qualificatore Q_m

AZIONE SD (time Stored/Delayed)

Dato uno stato *n*, Se Q_n=*SD t#Ts* , l'azione coincide ad una azione SET ritardata di T secondi.

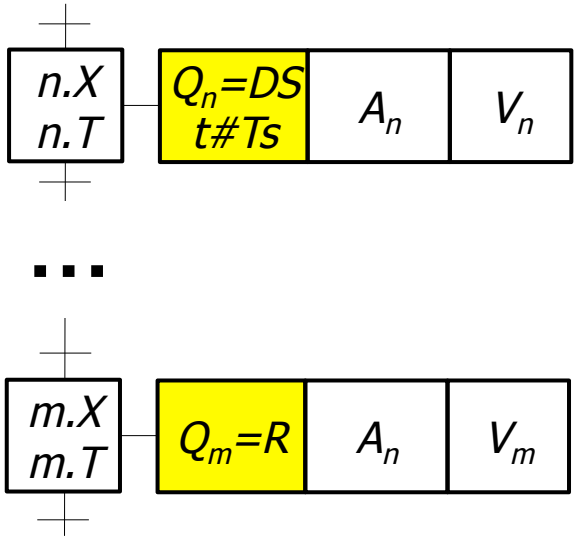


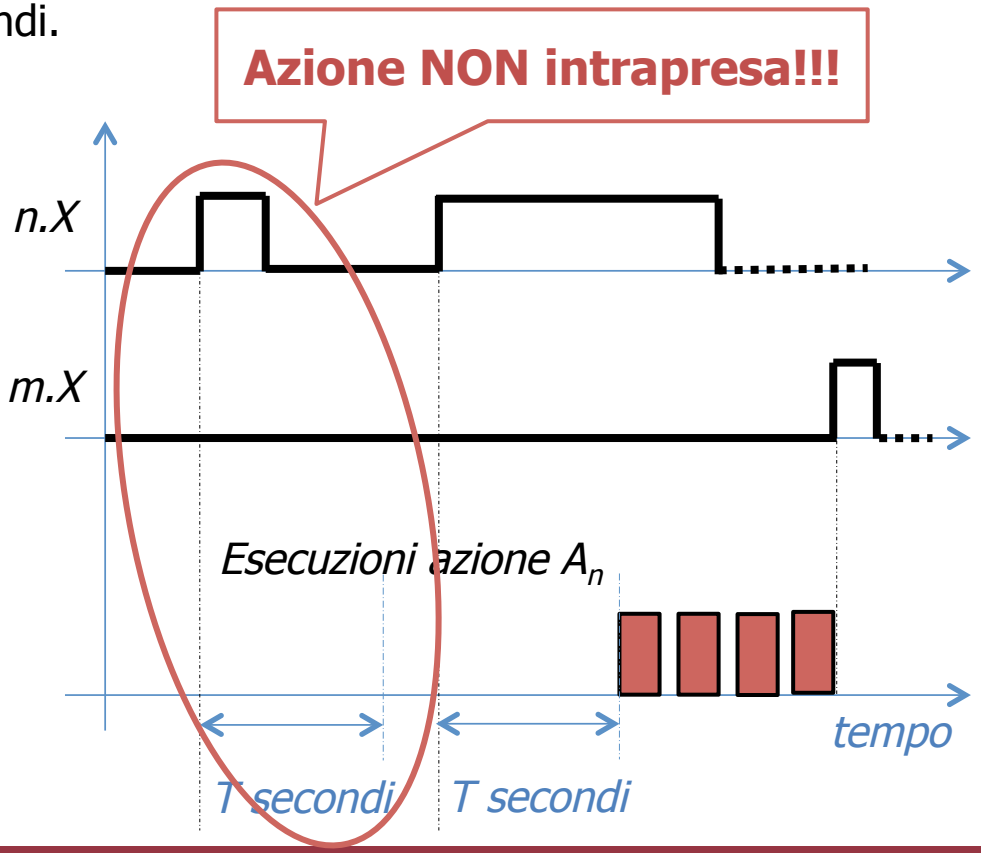


Qualificatore Q_m

AZIONE DS (time Delayed/Stored)

Dato uno stato *n*, Se Q_n=*DS t#Ts* , l'azione coincide ad una azione SET se è verificata la condizione *n.X* = TRUE per più di T secondi.

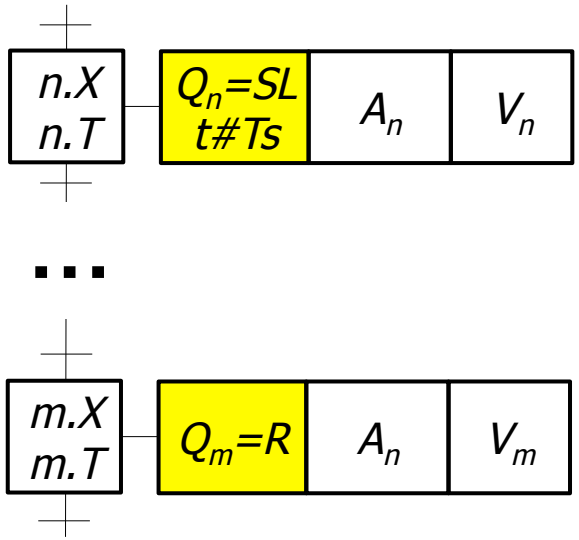


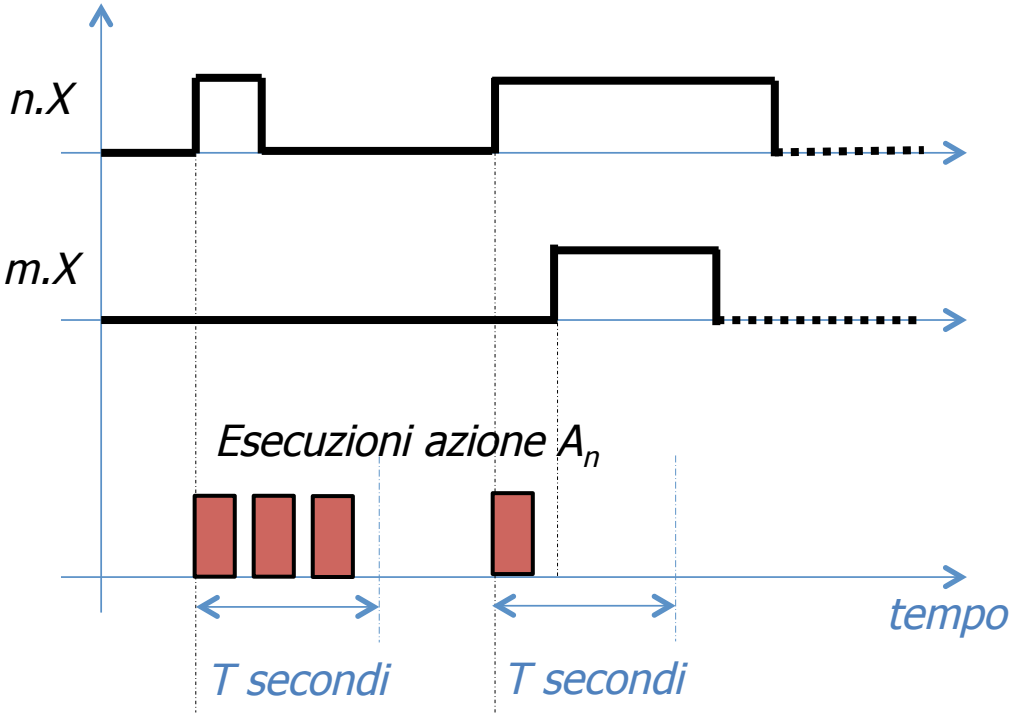


Qualificatore Q_m

AZIONE SL (Stored/time Limited)

Dato uno stato *n*, Se Q_n=*SD t#Ts* , l'azione coincide ad una azione SET e viene terminata dopo T secondi OPPURE all'occorrere di un RESET.





BIBLIOGRAFIA

Sezioni 7.1-7.2

TITOLO

Sistemi di automazione industriale

Architetture e controllo

AUTORI

Claudio Bonivento

Luca Gentili

Andrea Paoli

EDITORE

McGraw-Hill