# Capitolo 1

# Introduzione, concetti generali

## 1.1 Definizioni

L'Automazione Manifatturiera studia i sistemi di produzione e il modo in cui è possibile effettuare ed eventualmente automatizzare il controllo dei processi produttivi.

Un sistema di produzione è un sistema che trasforma un bene in un altro bene. La trasformazione è puntata a fornire valore aggiunto al bene in ingresso onde ottenere un prodotto di valore (economico) maggiore in uscita.

Tutte le industrie possono essere suddivise in 3 settori: settore primario, settore secondario e settore terziario. Il primo si occupa dell'estrazione del bene (p. es. agricoltura, estrazione mineraria, estrazione petrolifera, …), il secondo si occupa della lavorazione del bene (p. es. industria aerospaziale, industria per il raffinamento del petrolio, industria elettronica,…) ed infine il terzo si occupa dell'erogazione di servizi ovvero facilitare l'uso dei beni (p. es. banche, industria dell'informazione, settore educativo, …).

È possibile fornire una classificazione per i tipi di beni e per i tipi di produzione.

I beni prodotti da un sistema di produzione possono essere continui o discreti.

I *beni continui* sono quelli per i quali non è possibile individuare entità separate e distinte. Tipici esempi sono beni liquidi come il petrolio ma anche altri materiali, come la carta o le polveri.

I *beni discreti* sono invece quelli per i quali è possibile individuare entità separate e distinte. Tipici esempi sono tutti gli oggetti che utilizziamo quotidianamente, dal computer, al telefono, all'automobile.

Anche per ciò che concerne la produzione è possibile individuare una tipologia continua ed una discreta, indipendentemente dal tipo di bene prodotto.

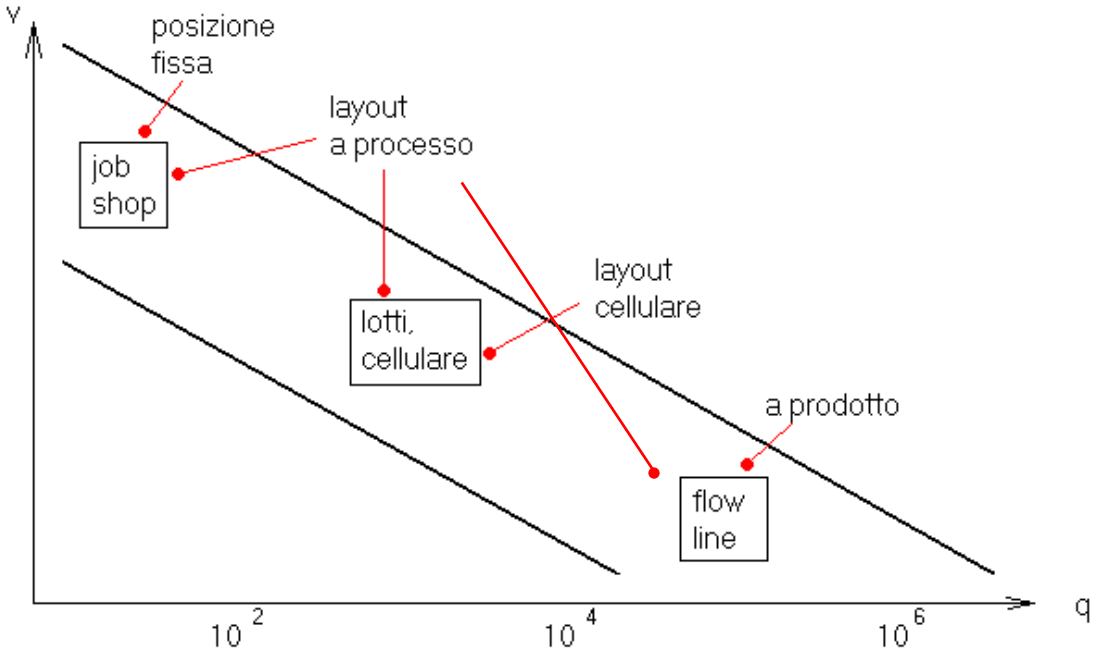
Una *produzione* si dice *continua* quando l'impianto lavora i beni in transito (continui o discreti) senza interruzioni. Un tipico esempio relativo a un bene continuo è l'industria della carta.

Una *produzione* si dice *discreta* quando l'impianto lavora i beni in transito (continui o discreti) organizzandoli in lotti. Nel caso di beni discreti questo può servire, come vedremo, a ridurre i tempi e i costi di riattrezzaggio (*setup*) delle macchine. Nel caso di beni continui questo tipo di produzione può essere necessario se il bene viene prodotto mediante reazioni chimiche che avvengono in determinati recipienti dotati di una certa capacità.

## 1.2 Rapporto tra quantità prodotte e varietà di beni producibili da un impianto

Per una generica industria è possibile stabilire il legame tra la varietà dei beni prodotti (i beni in uscita dal sistema di produzione) e la loro quantità. Una industria diversificata difficilmente potrà disporre di una produzione di grande quantità, viceversa un'industria che produce una grande quantità di beni probabilmente fabbricherà prodotti molto simili o al limite tutti uguali tra loro.

Tale legame è rappresentabile graficamente su un piano cartesiano tramite una fascia con pendenza negativa delimitata da due rette limite. La maggior parte delle industrie si colloca in questa fascia. Suddetto piano cartesiano ha in ascissa la quantità prodotta ed in ordinata la varietà dei beni prodotti. In generale si ha che per produzioni basse ($q < 100$ pezzi/anno) si ha un'alta varietà di beni prodotti, per produzioni medie ($100 < q < 10.000$ pezzi/anno) si ha una varietà media di beni prodotti ed infine, per produzioni alte ($10.000 < q < 1.000.000$ pezzi/anno), si ha una bassa varietà di beni prodotti.




## 1.3 Tipi di layout e di organizzazione della produzione

Un impianto manifatturiero prevede diversi tipi di organizzazione del sistema di produzione inteso come cammino che deve compiere il bene in ingresso per essere lavorato. Evidenziamo i seguenti tipi di organizzazione: *job shop, flow line,* a *lotti e cellulare.*

- Organizzazione *job shop*: ogni tipo di pezzo effettua un proprio percorso all'interno del sistema di produzione.

- Organizzazione *flow line*: tutti i pezzi effettuano lo stesso percorso.

- L'organizzazione a *lotti* prevede che un insieme di pezzi simili vengano posti in lotti e quindi ogni lotto effettui un proprio percorso all'interno del sistema di produzione.

- L'organizzazione *cellulare* prevede la suddivisione dell'intero sistema in celle produttive indipendenti ciascuna delle quali produce un insieme di parti molto simili fra loro.

La *job shop* e la *flow line* si trovano agli estremi per tipo di organizzazione: la prima usa un percorso diverso per ogni tipo di pezzo da lavorare, mentre la seconda fa esattamente l'opposto. L'organizzazione *a lotti* e l'organizzazione *cellulare* rappresentano, invece, un tipo intermedio di organizzazione tra job shop e flow line.

Con il termine *layout* indichiamo il modo con cui le macchine sono disposte all'interno di un'azienda manifatturiera. Distinguiamo 4 tipi di layout: *Layout a posizione fissa, Layout a processo, Layout cellulare e Layout a prodotto*.

- Il *Layout a posizione fissa* è caratterizzato da un sistema di costruzione e manipolazione di oggetti posti come satelliti attorno al bene in produzione. Tipici esempi in cui le macchine lavorano intorno all'oggetto e quest'ultimo è fisso sono la produzione di navi o la produzione di aerei.

- Il *Layout a processo* è caratterizzato da una suddivisione dell'impianto in una moltitudine di dipartimenti. All'interno di ogni cella si realizza un tipo di processo, quale può essere una verniciatura, una laminatura, un taglio e così via. Al termine del processo il bene prodotto da una certa cella viene passato ad un'altra cella la quale modifica ulteriormente il bene. Quindi ogni cella compie un determinato processo ed il pezzo in lavorazione viaggia nel sistema di cella in cella fino a diventare il prodotto finale. In genere è possibile produrre beni diversi, per ogni tipo di prodotto esiste un percorso diverso tra i dipartimenti. Generalmente i beni viaggiano nel sistema raggruppati in lotti (batch) per minimizzare i costi ed i tempi di trasporto e di setup delle macchine.

- Il *Layout cellulare* è caratterizzato da una suddivisione dell'impianto in una moltitudine di celle (Group Technology), macroscopicamente simile al caso precedentemente analizzato. La differenza sostanziale sta nel fatto che ogni cella produce un bene e non si occupa del solo incremento di valore aggiunto; quindi ogni cella si comporta come se fosse un mini impianto operando un processo completo.

- Nel *Layout a prodotto* le macchine sono organizzate lungo una linea: ogni pezzo effettua quindi lo stesso percorso e quindi le differenze tra le parti prodotte, se ci sono, sono minime.

È possibile riconoscere una stretta corrispondenza tra layout e organizzazione del sistema di produzione. Il job shop è il tipo di produzione in cui ogni parte segue un proprio percorso e può essere realizzato su un layout a posizione fissa oppure a processo. Quando il numero di parti supera una certa quantità (dell'ordine del centinaio) la produzione viene eseguita a lotti (batch) oppure organizzata in cellule (group technology). La produzione batch viene realizzata da layout a processo; quella a cellule in layout cellulare. Infine, nel caso che la varietà sia limitata mentre la quantità molto consistente, conviene organizzare la produzione secondo una flow line, tipicamente realizzata in un layout a prodotto.

## 1.4 L'automazione nell'impianto manifatturiero

All'interno di un generico impianto è possibile riconoscere almeno tre tipi diversi di automazione: fissa, flessibile e programmabile.

L'automazione *fissa* è un tipo di automatizzazione dell'impianto rigida e quindi non riconfigurabile per un nuovo impianto o per un diverso processo produttivo.

L'automazione *programmabile* è invece un tipo di automatizzazione dell'impianto facilmente riconfigurabile ovvero modificabile in modo semplice, utilizzata se la quantità è medio bassa e organizzata in batch. Questo tipo di automatizzazione fa uso di PLC (controllori logici programmabili) e del controllo numerico.

L'automazione *flessibile* è un'estensione dell'automazione programmabile con la caratteristica fondamentale di essere molto veloce da riconfigurare, cioè avente tempi di setup trascurabili.

Non sempre l'automazione è la risposta migliore alle esigenze dell'impianto. Esistono cioè dei casi per i quali è più indicato l'uso di manodopera umana. Il lavoro manuale è la scelta migliore se l'automatizzazione è difficoltosa per un determinato processo oppure se il prodotto ha durata limitata (cioè la domanda di quel prodotto

ha durata limitata) e quindi i costi di automatizzazione non sarebbero ammortizzati. Nel caso il prodotto sia nuovo sul mercato, e quindi sia ancora in fase di test, è possibile osservare una migrazione verso l'automatizzazione della produzione. Si passa dalla produzione *manuale* in cui le lavorazioni e i trasporti dei beni tra le macchine sono tutti assistiti da operatore umano, alla produzione *automatizzata*, in cui le macchine lavorano in modo sostanzialmente autonomo ma i trasporti dei beni tra le macchine sono manuali, alla produzione *automatizzata integrata*, in cui sia i trasporti sia le lavorazioni sono completamente automatizzate.

## 1.5 Indici di prestazione da ottimizzare

Un processo produttivo è modellabile in vari modi generalmente molto complessi, perché combinano una dinamica continua con una a eventi, includendo una componente stocastica sostanziale. Un indice di prestazione è un numero che riassume in modo molto sintetico il comportamento del sistema di produzione. Il problema del controllo di un sistema di produzione viene spesso formulato come problema di ottimizzazione di un indice di prestazione. Questo dovrebbe garantire un buon comportamento complessivo del sistema, per quanto ciò sia vero solo approssimativamente, se si tiene presente che ottimizzando un indice si può rendere inaccettabile il valore di un altro indice. È quindi necessario valutare bene le proprietà del sistema e in genere trovare una soluzione sub-ottima che ottimizzi .un po' tutti gli indici di prestazione d'interesse in un determinato contesto. I principali indici di prestazione che richiamiamo in queste note sono: il *tasso produttivo del sistema o throughput*, il *work in process o WIP*, il *manufacturing lead time o MLT*, il *coefficiente di utilizzazione delle macchine*, il *rispetto delle date di consegna (due dates)*.

- Il **throughput** è il numero di parti che vengono prodotte dal sistema nell'unità di tempo. Obiettivo del controllo è quello di minimizzare lo scostamento tra domanda e tasso produttivo. Assumendo però in molti casi un valore infinito per la domanda, ossia supponendo che il sistema di produzione sia sottodimensionato rispetto alla domanda esterna, si tende spesso a massimizzare il tasso produttivo. Il tasso produttivo di una linea è limitato dalla v*elocità della macchina più lenta*, dalla *mancanza di sincronizzazione* e

dalla *variabilità nei tempi di lavorazione*, risolvibili queste ultime utilizzando magazzini intermedi.

- Il **WIP** (acronimo di Work in Process) rappresenta il numero medio di pezzi nel sistema a regime, ed è una grandezza fortemente legata alla dimensione dei buffer intermedi. Massimizzare il WIP non implica l'ottimizzazione di tutto il processo: difatti se da un lato si verifica il disaccoppiamento delle macchine – e quindi la massimizzazione del throughput, dall'altra si ha il verificarsi di diversi fenomeni negativi come *l'immobilizzazione del capitale* (le parti contenute nei magazzini hanno un costo, e non producono profitto finché non escono dal sistema), l'aumento del pericolo di *danneggiamento*, problematiche legate al *controllo di produzione* e, come ovvio, problemi di *spazio di immagazzinamento* e di *gestione del flusso delle parti*.
  Per *disaccoppiamento* delle macchine si intende la non interruzione del lavoro di una macchina a valle nonostante il guasto della macchina immediatamente a monte (e viceversa). Tale comportamento è possibile solo introducendo buffer opportunamente dimensionati che quindi contengano pezzi da lavorare pronti per entrare nella macchina.

  Per *aumento di pericolo di danneggiamento* invece si intende il danno prodotto al titolare dell'impianto qualora si verificasse un incidente (p.e. incendio) causando la perdita di tutti i pezzi all'interno dell'impianto. L'entità del danno cresce col crescere del WIP, ovvero il pericolo di danneggiamento o deperimento delle parti è proporzionale al tempo di permanenza nel sistema.

  Per *problematiche legate al controllo di produzione* si intende la perdita che si ha per il malfunzionamento di una macchina la quale produce pezzi danneggiati, dovuta al fatto che il controllo sui danni dei pezzi sia fatto non per ogni macchina ma per esempio al termine di tutta la catena. In tal caso, il controllo rileverebbe il malfunzionamento di una delle macchine della catena solo dopo che altri pezzi difettosi sono stati prodotti e, se i magazzini intermedi sono grandi, ciò potrebbe comportare un gran numero di pezzi da scartare.

- Il tempo medio di permanenza nell'impianto (manufacturing lead time) **MLT**. Esso consiste nella differenza temporale tra l'istante temporale in cui il pezzo entra nel sistema e l'istante in cui esso esce, ovvero consiste nel tempo che

ciascuna parte impiega ad attraversare il processo. L'ottimizzazione di questo indice si ottiene in linea di principio minimizzandolo. Tuttavia anche per l'MLT valgono le considerazioni appena esposte per il WIP: si tratta infatti di due indici di prestazione fortemente correlati tra loro.

- Il coefficiente di utilizzazione delle macchine, se l'impianto è stato dimensionato correttamente, deve essere prossimo all'unità. Coefficienti troppo bassi equivalgono a un sovradimensionamento dell'impianto rispetto alla domanda effettiva.

- Le date di consegna costituiscono uno degli indici più importanti per quanto riguarda il soddisfacimento del cliente. Esso è legato alle problematiche del WIP: finire i prodotti troppo in anticipo rispetto alla data di consegna comporta tutti i problemi discussi sopra in relazione a valori del WIP troppo elevati. Finire i prodotti in ritardo ha evidenti ricadute negative sull'attività futura dell'azienda.

## 1.6 Legge di Little

Come detto sopra, WIP e MLT sono due grandezze correlate. Più precisamente, WIP e MLT sono legati in maniera proporzionale, al crescere del numero medio di entità all'interno del sistema cresce anche il tempo medio di attraversamento del sistema. Questo legame è descritto dalla legge di Little.

In base alla legge di Little, il numero medio N di pezzi all'interno del sistema è pari a regime e in condizioni di equilibrio (ossia quando il flusso medio dei pezzi in ingresso eguaglia il flusso medio dei pezzi in uscita) alla frequenza media $\lambda$ con cui i pezzi entrano (e escono, per quanto appena detto) nel sistema, per il tempo medio MLT di permanenza (in servizio o in coda) dei pezzi nel sistema:

$$WIP = \lambda \, MLT$$

Per un impianto manifatturiero la legge si traduce nel fatto che il numero medio di pezzi in lavorazione (WIP) è pari al prodotto della frequenza (tasso) media con cui essi sono richiesti per il tempo medio di lavorazione (MLT).

**Esempio 1.** Consideriamo un sistema inizialmente vuoto che produce 10 parti all'ora, ossia $\lambda$=10 p/h. supponiamo che ciascuna parte richieda un tempo medio di lavorazione di 2h, ossia MLT=2h. Quanto vale il WIP, ossia il numero di parti che in media troveremo nel sistema? Dalla legge di Little: WIP = $\lambda$ MLT = 20, cioè ci aspettiamo 20 parti presenti all'interno del sistema. Infatti, nelle prime 2 ore ci aspettiamo un tasso di ingresso di 10 p/h senza che nessuna parte sia terminata, ovvero senza che nessuna parte possa uscire dal processo di lavorazione. Le prime parti escono dopo 2h con un tasso di 10p/h (sono quelle che sono entrate nel sistema per prime). Dopo questo istante le parti presenti nel sistema saranno sempre 20, si è quindi arrivati a regime.

**Esempio 2.** Si consideri un impianto al quale arrivano richieste con frequenza $\lambda$=10 pezzi/h costituito da due macchine in cascata caratterizzate dal $MLT_1$=1h e $MLT_2$=2h. Il WIP totale può essere calcolato in due modi diversi, o come somma del WIP di ogni macchina oppure considerando le due macchine un unico sistema con tempo medio di lavoro pari alla somma dei tempi di lavoro.

Nel primo caso si ha $WIP = WIP_1 + WIP_2 = \lambda MLT_1 + \lambda MLT_2 = 30$ pezzi.

Nel secondo caso si ha $WIP = \lambda(MLT_1 + MLT_2) = 30$ pezzi.

**Esempio 3.** La legge di Little può essere utilizzata anche in ambiti diversi da quello produttivo. Si consideri per esempio un servizio di trasporto che, lungo una certa linea, assicuri il passaggio di un mezzo ogni 10 minuti. Si assuma che per percorrere l'intera linea occorra un'ora. Quanti mezzi viaggiano contemporaneamente lungo la linea? La risposta, peraltro facile da determinare intuitivamente, può essere ottenuta come applicazione della legge di Little, dove il numero di mezzi nella linea va interpretato come WIP, la frequenza di passaggio del mezzo come tasso di arrivo $\lambda$=6 mezzi/h, il tempo di percorrenza come MLT=1 h. Si ha quindi che il numero di mezzi in viaggio sarà pari a $\lambda \cdot MLT = 6 \cdot 1 = 6$ mezzi di trasporto.

# Capitolo 2

# Tecnologie nell'automazione manifatturiera

## 2.1 Introduzione

In questa sezione si descrivono le tecnologie principali utilizzate nell'automazione dei sistemi di produzione. Automatizzare un sistema di produzione significa rendere il sistema capace di funzionare limitando l'intervento di operatori umani.

Per l'automazione di un sistema di produzione sono necessari 3 elementi:

- una fonte di energia;
- un programma di istruzioni;
- un sistema di controllo che lo attui.

L'automazione incide sul processo a diversi livelli: a livello *apparecchiatura* si occupa del controllo degli attuatori e dei sensori di una macchina, a livello *macchina* si occupa dell'automatismo della macchina, a livello *cella* o sistema, si occupa dell'automazione dell'insieme di macchine appartenenti ad una cella (carico e scarico dei pezzi sulle macchine, sincronizzazione), a livello di *impianto* l'automazione consiste nel controllo di tutto il sistema di produzione, infine a livello di *impresa*, l'automazione include anche le operazioni esterne al processo, ovvero quelle di progettazione.

Le attuali tecnologie permettono l'utilizzo di PLC, controllori logici programmabili, e sfruttano il NC, controllo numerico per il controllo a livello di macchine e di cella.

Si è già evidenziata la differenza, all'interno del processo e dell'industria, tra componenti continui e componenti discreti. L'industria, per esempio, produce quantità di tipo continuo se il risultato del processo produttivo è costituito da liquidi, polveri o gas; altresì produce quantità di tipo discreto.

La classificazione “continuo - discreto” può essere estesa anche alle variabili di stato e di controllo dell’impianto e, quindi, ai sensori e agli attuatori. Si ricorda infatti, che un sistema di produzione è un sistema ibrido, caratterizzato sia da grandezze continue sia da grandezze discrete.

Le *variabili di stato o di controllo* continue sono per esempio i livelli di un liquido, la posizione di una parte da lavorare, la temperatura di un reagente, la concentrazione di sostanze chimiche in una reazione o le forze per effettuare una lavorazione mentre quelle discrete sono per esempio il numero di pezzi nel magazzino o qualsiasi tipo di condizione logica da verificare (apertura o chiusura di una valvola oppure confronti tra variabili continue).

I *sensori* continui sono per esempio le termocoppie, i sensori di portata o i sensori di forza mentre quelli discreti sono per esempio le uscite di cellule fotoelettriche o di spire magnetiche.

Le *azioni di controllo* continue sono per esempio le regolazioni della forza applicata o della velocità mentre quelle discrete sono per esempio le operazioni di apertura o chiusura di una valvola, di inizio lavorazione o di scarico pezzo.

Prima degli anni ’60 il controllo continuo era ottenuto tramite controllori basati su circuiti analogici, mentre il controllo discreto era essenzialmente costituito da banchi di Relè. Il *relè* è un dispositivo elettromeccanico che in presenza di determinate condizioni logiche sul circuito primario, ovvero in presenza di corrente $i_1$, apre o chiude un circuito detto secondario, permettendo il flusso della corrente $i_2$. La corrente $i_1$ eccita una bobina collegata ad un interruttore che chiude o apre il secondario.

Dopo gli anni ’60 crebbe l’utilizzo di tecnologia digitale, con la realizzazione dei primi controllori continui tramite calcolatore e dei primi controllori discreti basati su PLC. *La fine degli anni ’60 segna il passaggio dall’elettronica analogica all’utilizzo del*

*calcolatore, per il controllo continuo; ed il passaggio dai banchi di relè all'utilizzo di PLC per il controllo discreto.*

Gli svantaggi dalle vecchie tecniche che resero vincenti le nuove furono molteplici, primo fra tutti l'ingombro ed il consumo di corrente dei banchi di relè. Inoltre la difficoltà di progetto e modifica dei banchi di relè era accompagnata da un difficile riconoscimento dei guasti. Infine la caratteristica che rese vincente l'utilizzo dei PLC sui banchi di relè fu senza dubbio l'enorme differenza di velocità di elaborazione.

Nel seguito di questo capitolo ci si soffermerà sulla descrizione generale dei PLC e dei loro linguaggi di programmazione. Nel Cap. 3 si discuterà la programmazione di un PLC prodotto dalla Siemens, il Simatic S7.

## 2.2 I PLC

I controllori logici programmabili, PLC (1968), sono controllori discreti che sostituiscono i vecchi banchi di relè.

Il PLC riceve in ingresso il valore restituito dai sensori del processo, esegue il programma ed invia i risultati alle uscite del PLC che comandano gli attuatori.

Secondo lo Standard 1131 del Comitato Elettrotecnico Internazionale "Un PLC è un sistema elettronico a funzionamento digitale, destinato all'uso in ambito industriale, che utilizza una memoria programmabile per l'archiviazione interna di istruzioni orientate all'utilizzatore per l'implementazione di funzioni specifiche, come quelle logiche, di sequenziamento, di temporizzazione, di conteggio e di calcolo aritmetico, e per controllare, mediante ingressi ed uscite sia digitali che analogici, vari tipi di macchine e processi".

### 2.2.1 Componenti sistema PLC e modalità di funzionamento del PLC

Il PLC è composto da un processore, da moduli ingresso / uscita e dai terminali di programmazione (collegati al PLC solo in sede di programmazione).

Il Processore funziona ripetendo un ciclo composto da tre fasi: nella prima fase, detta *aggiornamento degli ingressi*, il segnale in ingresso al PLC, ovvero l'uscita del sensore, viene prelevata e mantenuta costante durante tutto il tempo di esecuzione del ciclo; la seconda fase è detta *esecuzione del programma* ed infine, nell'ultima fase, *l'aggiornamento delle uscite*, vengono spediti i risultati dell'elaborazione in ingresso agli attuatori.

Il tempo di scansione (cioè la durata del ciclo) dipende dalla lunghezza del programma: se il programma è lungo pochi KB il tempo di scansione è compreso tra l'unità e la decina di ms.

Il processore possiede una memoria, divisa in 5 zone: l'area del sistema operativo (di tipo ROM) contenente l'interprete dei comandi del PLC, l'area di lavoro del sistema operativo (di tipo RAM) contenente lo spazio di memoria utilizzato dai calcoli del sistema operativo, l'area input/output (di tipo RAM) contenente la memoria per gli ingressi e le uscite, l'area dei programmi utente (di tipo RAM/PROM perché modificabile come una RAM nella *modalità di programmazione* – vedere sotto la definizione delle modalità di funzionamento di un PLC - ma non è modificabile nella *modalità di esecuzione*) contenente il listato del programma che il PLC deve compiere ed infine l'area di lavoro dei programmi utente (di tipo RAM) ovvero l'area di memoria per i calcoli del programma utente.

Per dare un'idea delle dimensioni della memoria e per introdurre un po' di notazione, si supporrà che l'area dati della memoria RAM del PLC comprenda 32 word (=parole, ogni parola è composta da 16 bit) riservate agli ingressi. Un ingresso analogico sarà memorizzato in 16 bit (ossia occuperà un'intera parola), un ingresso binario sarà associato a un singolo bit. Analogamente, 32 word saranno dedicate alle uscite, mentre ben 512 word saranno disponibili per le variabili di lavoro (interne) dei programmi utente. Infine 16 parole saranno disponibili per le variabili di conteggio e 16 per quelle di temporizzazione. Come notazione, Ix indicherà la parola numero x dell'area degli ingressi (p. es. I5 sarà la quinta parola dell'area degli ingressi) mentre Ix:y indica il bit y della word x degli ingressi (p. es. I2:3 è il terzo bit della seconda parola associata agli ingressi). Stesse regole si applicano alle parole riservate alle uscite (parole U), alle variabili interne (W), a quelle dei contatori (C) e infine a quelle dei temporizzatori (T). Per semplicità di lettura, conviene associare dei nomi mnemonici ad alcune variabili. Per esempio, VALVOLA = U4:5 significa che

il 5° bit nella word 4 dell'area di memoria riservata alle uscite viene richiamata col simbolo VALVOLA e quindi è un bit associato per esempio all'apertura o alla chiusura di una valvola. La notazione TEMP = I2 associa il nome simbolico TEMP (per es. temperatura) alla lettura di un sensore analogico (quindi una word intera) conservato nell'indirizzo I2.

Un PLC non è dotato di tastiera o schermi per la comunicazione con il programmatore. Per tale comunicazione, allora, esistono dei terminali a tastiera appositi, che si connettono al PLC tramite una porta di comunicazione (per es. seriale). Un'alternativa più recente è quella di avere tutti i PLC collegati con una rete a un PC che provvede a caricare i programmi su tutti i PLC connessi.

Il PLC può lavorare in 3 modalità di funzionamento:

- esecuzione;
- programmazione;
- validazione.

Nella prima il PLC esegue il ciclo di scansione già specificato precedentemente ed è la fase normale di funzionamento in cui il PLC controlla il processo per cui è stato programmato, aggiornando ingressi e uscite; nella seconda il PLC viene scollegato dal processo e viene collegato a un PC o altro terminale di programmazione per la scrittura del programma di esecuzione; nell'ultima modalità il PLC viene sottoposto a test e simulazione.

I linguaggi di programmazione per programmare i PLC sono: il linguaggio a contatti, il diagramma funzionale sequenziale (SFC), il diagramma funzionale a blocchi, la lista di istruzioni ed infine il testo strutturato.

### 2.2.2 Linguaggio a contatti

È disponibile a tutt'oggi su tutti i PLC e quindi in alcuni casi è necessario avere delle procedure di traduzione per passare dagli altri linguaggi a quello a contatti. Le prime applicazioni di automazione industriale venivano realizzate utilizzando dispositivi elettromeccanici come relè, temporizzatori, contatori. Il primo linguaggio venne

quindi sviluppato il più vicino possibile a quel tipo di tecnologia per poter essere compreso e utilizzato dai tecnici dell'epoca, privi di conoscenze informatiche. Pertanto le prime istruzioni disponibili furono quelle che rappresentavano il contatto normalmente aperto o normalmente chiuso di un relè, la sua bobina di eccitazione, il temporizzatore e il contatore.

Anche la forma grafica del programma deriva dalla logica a relè: le linee verticali della scala (i montanti) rappresentano l'alimentazione e quelle orizzontali (i pioli della scala detti rung) alimentano una bobina se tutti i contatti sono chiusi. I contatti possono essere associati agli ingressi esterni oppure a condizioni interne, tutte rappresentate in bit (0 = contatto aperto - se normalmente aperto, 1 = chiuso - se normalmente aperto). La bobina può essere associata a un bit di memoria ed eventualmente comandare un'uscita o segnalare una condizione interna.

I possibili tipi di istruzioni sono:

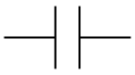
- Istruzioni di base;
- Istruzioni di temporizzazione e conteggio;
- Istruzioni per il controllo del flusso del programma;
- Istruzioni per la manipolazione dei dati.

Ricordiamo che, in accordo con quanto detto all'inizio, il funzionamento del PLC è ciclico ed ogni ciclo comprende le seguenti azioni: lettura degli ingressi ed aggiornamento delle variabili corrispondenti; esecuzione del programma, nel caso a contatti di tutta la scala ad esso associata, piolo per piolo (durante tale esecuzione le variabili di ingresso restano costanti); scrittura delle variabili di uscita sulle uscite del PLC. Illustriamo qui di seguito questi tipi di istruzioni.

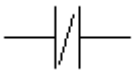
## ISTRUZIONI DI BASE

Sono istruzioni di base il contatto normalmente aperto, il contatto normalmente chiuso e la bobina.

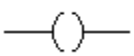
**Contatto normalmente aperto.** Il contatto normalmente aperto è associato ad un ingresso o ad una variabile interna del programma con la seguente relazione: quando la variabile è alta il contatto si chiude, quando è bassa si apre.



**Contatto normalmente chiuso.** Il contatto normalmente chiuso è associato ad un ingresso o ad una variabile interna del programma con la seguente relazione: quando la variabile è bassa il contatto si chiude, quando è alta si apre.



**Bobina.** La bobina è associata ad una variabile di uscita o interna del programma. Sta sempre all'estrema destra del rung. La variabile corrispondente è alta se esiste almeno un percorso dal montante sinistro alla bobina attivo, ovvero i contatti che uniscono il montante sinistro alla bobina sono tutti chiusi. Il suo simbolo grafico ricorda il circuito di eccitazione di un interruttore a relè.



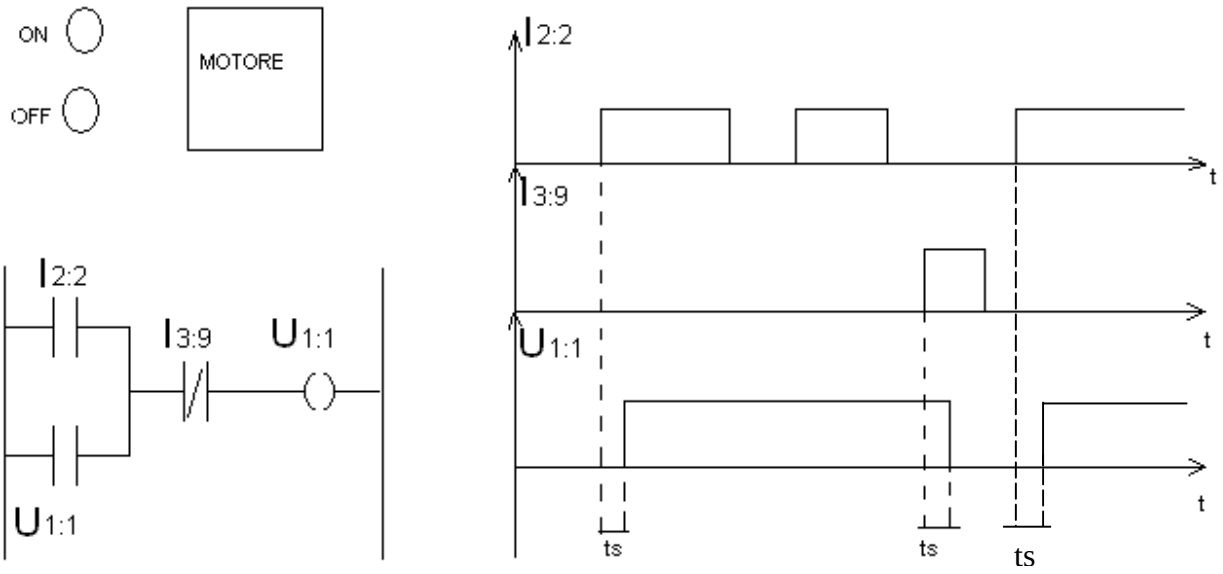
Con le istruzioni di base è già possibile scrivere un semplice programmino per un PLC che realizza un elemento di memoria (flip-flop) con pulsante di set e reset.

#### 2.2.2.1 Circuito di set reset

Se per ipotesi volessimo fare un circuito di Set-Reset, cioè un circuito con memoria, potremmo usare con successo un PLC. Esemplifichiamo questo con una possibile applicazione reale in cui si vuole accendere un motore con un pulsante di ON e mantenerlo acceso fintanto che non viene premuto un pulsante di OFF. Lo schema del PLC che attua ciò è quello presentato di seguito. I2:2 è l'ingresso di ON mentre I3:9 è l'ingresso di OFF.

Finché il pulsante ON è basso ed il tasto di OFF è basso, l'uscita, ovvero il motore, è disabilitato. Nel momento in cui il tasto di ON viene premuto l'ingresso I2:2 diventa alto (l'ingresso I3:9 continua a rimanere basso quindi connette il ramo) ed il motore comincia a girare (in realtà parte con un tempo di ritardo pari al tempo di ciclo o di

scan ts) cioè l'uscita U1:1 diviene alta. Quando I2:2 torna basso, il contatto U1:1 è ancora alto quindi il motore continua a girare. Quando I3:9 diviene alto il circuito viene disconnesso (con un tempo di ritardo pari a ts).




## ISTRUZIONI DI TEMPORIZZAZIONE E CONTEGGIO

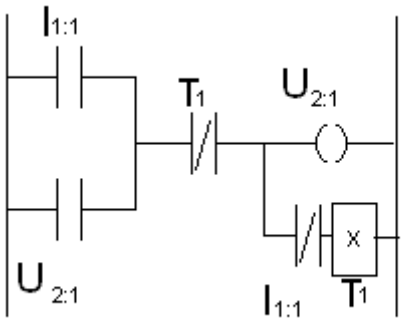
Permettono di controllare operazioni basandosi sul trascorrere del tempo e sul conteggio di eventi. Distinguiamo quindi due tipi di istruzioni: di temporizzazione e di conteggio.

**Temporizzatore.** Si rappresenta con un blocco rettangolare con dentro una durata temporale. Se tale durata è 60 secondi e T2 è la variabile associata al temporizzatore, il funzionamento è il seguente: appena il rung (il piolo) su cui si trova il temporizzatore fa passare corrente, esso comincia a contare 60 secondi con T2 che resta falsa. Quando sono trascorsi 60 secondi T2 diventa vera e resta tale finché la condizione descritta dal rung di alimentazione resta vera. Si resetta (T2=0) appena tale condizione diventa falsa, eventualmente anche durante il conteggio.

Esistono anche temporizzatori a ritenuta il cui conteggio viene conservato se le condizioni di alimentazione diventano false. Per resettare un temporizzatore a ritenuta si utilizza un'istruzione apposita (una bobina con scritto dentro *RES* associata all'indirizzo del temporizzatore).

**Contatore.** Si rappresenta con un blocco rettangolare con dentro un numero intero che indica il count-down del contatore e una variabile associata al contatore stesso. Supponiamo che tali valori siano: 7 (il numero intero di count-down) e C1 (la variabile del contatore). Il contatore incrementa di una unità il suo conteggio quando le condizioni sul rung dove si trova passano da falso a vero e assegna un livello alto a C1 quando tale conteggio diventa uguale a 7. La variabile C1 viene resettata (cioè viene posta nuovamente a 0) con un'istruzione di reset (una bobina con scritto dentro RES associata all'indirizzo del contatore). Contatori e temporizzatori possono essere usati in cascata per intervalli di conteggio maggiori delle loro capacità massime.

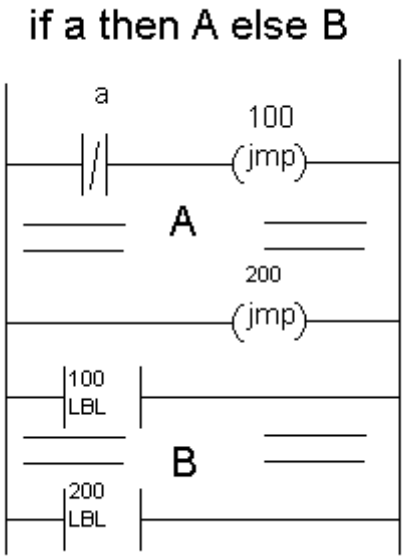
**Esempio.** Si scriva il programma per un PLC che ritardi lo spegnimento di un dispositivo di x secondi (per esempio una ventola che raffreddi un motore ancora per x secondi dopo che il motore si è fermato). Una soluzione possibile è il circuito di figura, supponendo che I1:1 sia il segnale di accensione del motore e U2:1 il segnale che comanda il movimento della ventola. Finché I1:1 e T1 sono bassi il circuito è non connesso. Appena I1:1 (il segnale di alimentazione del motore) diviene alto, il ramo I1:1 - T1 - U2:1 si chiude e la ventola comincia a girare. Quando I1:1 diviene basso, il circuito continua ad essere connesso per opera del contatto U2:1 ed il temporizzatore comincia a contare il tempo. Dopo x secondi il temporizzatore diviene alto disconnettendo il circuito e tornando immediatamente dopo basso.




## ISTRUZIONI DI CONTROLLO DEL FLUSSO DEL PROGRAMMA

**Istruzioni di salto.** Per le istruzioni di salto si utilizza la "bobina di jump" che quando è alta fa si che il programma salti direttamente alla label indicata dalla bobina. Mediante l'istruzione salto è semplice realizzare le strutture IF-THEN-ELSE, REPEAT-

UNTIL, DO-WHILE. Per esempio, un costrutto if-then-else può essere rappresentato dal seguente schema:




ISTRUZIONI PER LA MANIPOLAZIONE DEI DATI

Ne riportiamo alcune a titolo esemplificativo. Un blocco rettangolare con scritto dentro MOV, OP1, OP2, quando viene alimentato (ossia quando le condizioni sul rung dove si trova divengono vere) sposta il contenuto della locazione di memoria OP1 nella locazione OP2. Un blocco rettangolare con scritto dentro OPERAZIONE, OP1, OP2, RES, quando viene alimentato effettua l'operazione OPERAZIONE sugli operandi i cui valori sono nelle variabili OP1, OP2 e scrive il risultato in RES. L'OPERAZIONE indicata può essere una somma (ADD), un prodotto (MUL), una sottrazione (SUB), una divisione (DIV), un AND o un OR logico, ecc.

### 2.2.3 Linguaggio SFC

Si tratta di un linguaggio grafico finalizzato alla descrizione del comportamento dei sistemi a eventi discreti, cioè di quei sistemi in cui le variazioni dello stato non sono direttamente legate al trascorrere del tempo ma all'occorrenza di eventi. Un sistema di controllo di un processo industriale è un sistema a eventi discreti e quindi l'SFC (Sequential Functional Chart) può essere impiegato con profitto nella progettazione

degli algoritmi di controllo. L'SFC è il linguaggio più innovativo riportato nello Standard 61131 del Comitato Elettrotecnico Internazionale, documento che ha dato un certo ordinamento alle numerose versioni e dialetti dei linguaggi comunemente usati per programmare un PLC. Tale linguaggio presenta numerosi vantaggi rispetto agli altri: essendo il modo naturale per descrivere un sistema logico-sequenziale, rappresenta anche un modo chiaro per definire il comportamento desiderato per un sistema (non solo di produzione) e quindi del dispositivo che lo controlla, evitando ogni possibile forma di ambiguità che un documento in lingua corrente potrebbe creare. Risulta inoltre facile da esaminare e da modificare qualora necessario.

Se il PLC per cui è scritto il programma di controllo accetta il linguaggio SFC (cosa per nulla garantita in generale) non vi è bisogno di ulteriori operazioni, altrimenti occorrerà tradurre l'algoritmo di controllo in un altro linguaggio, accettato dal PLC. I dispositivi Siemens che si proverà a programmare nel seguito di fatto non accettano l'SFC: si insegnerà più avanti (si veda in particolare l'Esempio 4) un modo per tradurre un programma scritto in linguaggio SFC in un programma scritto in linguaggio a contatti: sarebbe invero molto più arduo scrivere (e interpretare e/o modificare) direttamente un programma in linguaggio ladder piuttosto che lavorare sulla sua versione SFC. La procedura di traduzione ha anche il vantaggio di permettere in modo relativamente semplice l'introduzione di modifiche al programma stesso.

Gli elementi di base dell'SFC sono:

- la fase, con le azioni ad essa associate;
- la transizione, con la condizione ad essa associata;
- l'arco orientato.

#### 2.2.3.1 La fase.

Le fasi sono rappresentate da rettangoli contenenti un numero progressivo. Ad ogni fase possono essere associate delle azioni, rappresentate anch'esse mediante rettangoli. In uno schema SFC tutte le azioni associate a una fase avvengono contemporaneamente. Le azioni associate ad ogni fase possono essere per esempio "accendi motore", "apri valvola", "spegni ventilatore", ecc. Quando il sistema di controllo si trova in una certa fase, questa viene detta **ATTIVA**, e, graficamente, ciò

viene rappresentato ponendo un pallino al suo interno. Se il pallino non c'è la fase è detta **INATTIVA.**

La fase iniziale (cioè quella da cui parte il programma di controllo) è rappresentata per convenzione con una cornice doppia. La variabile che descrive l'attivazione o meno di una certa fase è detta "**marker di fase**". Si indica con $X_n$ il marker della fase n. Tale variabile vale 0 se la fase n è inattiva e vale 1 se la fase n è attiva.

#### 2.2.3.2 Transizione

Una TRANSIZIONE viene indicata con una barretta e la sigla $T_n$, con n il numero che la identifica. Alla transizione viene associata una condizione logica che viene riportata accanto alla transizione stessa.

#### 2.2.3.3 Archi orientati

Gli archi orientati collegano tra loro le fasi, stabilendo la sequenza, e sono interrotti dalle transizioni, le quali determinano le condizioni da verificare per passare da una fase (disattivandola) a un'altra (attivandola).

Tra due fasi collegate da un arco ci deve sempre essere una transizione e tra due transizioni sempre almeno una fase.

Per convenzione, quando due o più fasi convergono su una sola transizione, o da una sola transizione partono più fasi, si utilizza rappresentare il collegamento con una doppia linea.

#### 2.2.3.4 Regole di evoluzione del linguaggio SFC

Si definisce condizione di un SCF l'insieme delle sue fasi attive. Un SFC può cambiare la sua condizione attraverso il superamento delle transizioni. Perché una transizione sia superabile, devono verificarsi entrambe le due condizioni seguenti:

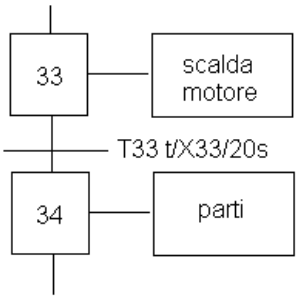
- tutte le fasi a monte della transizione sono attive, e la transizione è detta abilitata;
- la condizione associata alla transizione è vera.

Quando una transizione è superabile essa viene superata: le fasi a monte vengono disattivate (si cancella il pallino nel loro interno) e vengono attivate quelle a valle (si disegna un pallino nel loro interno).

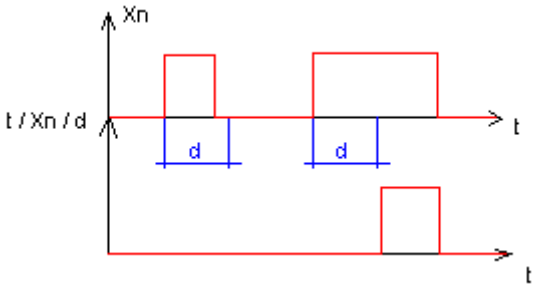
### 2.2.3.5 Variabile temporale

Si indicano con la notazione $t/X_n/d$, dove t identifica la temporizzazione, $X_n$ è il marker associato alla fase la cui attivazione fa partire la temporizzazione, d è la durata della temporizzazione. La variabile temporale assume il valore 0 all'istante iniziale, conserva tale valore finché la fase n è inattiva, diventa 1 quando è trascorso un tempo d dall'ultima attivazione della fase n che deve essere rimasta attiva. Torna a 0 quando n si disattiva. La variabile temporale corrisponde all'istruzione di temporizzazione nel caso del linguaggio a contatti.

**Esempio.** Se si volesse scaldare il motore di una macchina per 20 secondi prima di partire sarebbe sufficiente imporre la transizione con variabile temporale come segue.

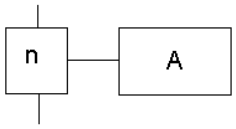



Si otterrebbe un comportamento del seguente tipo:

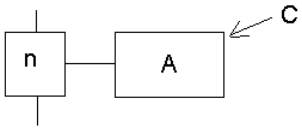


### 2.2.3.6 Tipi di azioni

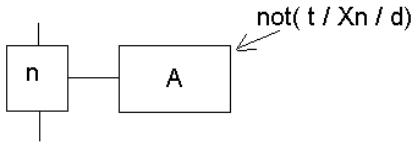
**Azione continua**: l'azione viene avviata non appena la fase n a cui è associata diventa attiva e continua finché la fase n non viene disattivata.



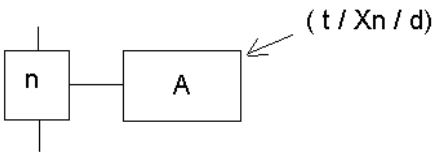

**Azione condizionata**: l'azione viene effettuata solo se la fase a cui è associata è attiva e la condizione C associata risulta vera.



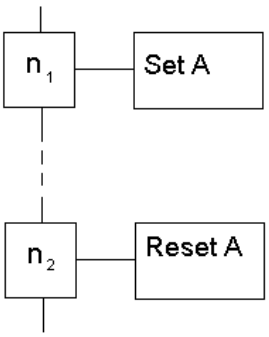

**Azione limitata nel tempo**: è un caso particolare di azione condizionata, in cui la condizione è una variabile temporale $t/X_n/d$ associata alla stessa fase a cui è associata l'azione (per esempio serve se un'elettrovalvola deve essere comandata con impulsi di una certa durata).




**Azione ritardata**: l'azione viene avviata solo dopo che è trascorso un tempo d da che la fase n è diventata attiva e dura finché la fase rimane attiva.

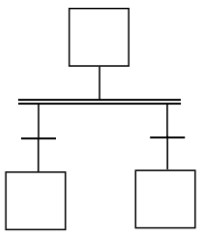



**Azione memorizzata**: l'azione viene avviata quando la fase $n_1$ diviene attiva e viene interrotta quando la fase $n_2$ diviene attiva.

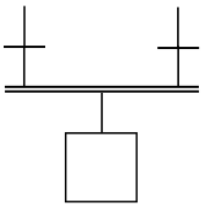



### 2.2.3.7 Possibili strutture del programma

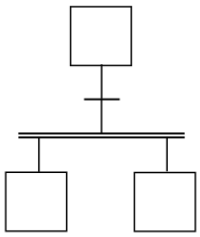
Quando a una fase seguono due o più transizioni ci troviamo di fronte a una struttura di tipo *scelta*: il ramo che viene attivato è quello associato alla transizione con la condizione logica verificata (per questo motivo le condizioni delle transizioni a valle di una stessa fase devono in questo caso essere mutuamente esclusive).



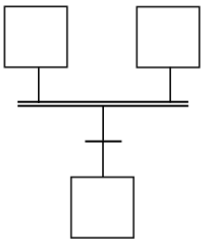
Ci troviamo di fronte a una *convergenza* quando più rami convergono in una stessa fase.



Si ha invece una *concorrenza* (o *parallelismo*) quando a una stessa transizione seguono due o più fasi: in questo caso tutti i rami a valle della transizione vengono attivati.



Infine, una *sincronizzazione* è costituita dal convergere di due o più rami in una stessa transizione.



Gli abbinamenti possibili sono:

- Scelta seguita da convergenza;
- Parallelismo seguito da sincronizzazione.

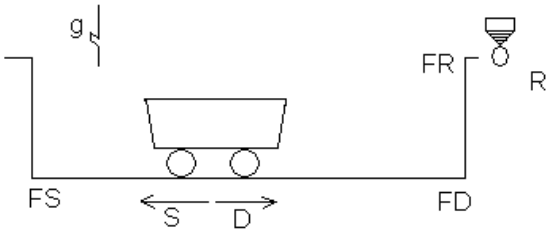
Sono invece errati o ambigui gli altri abbinamenti possibili, e cioè:

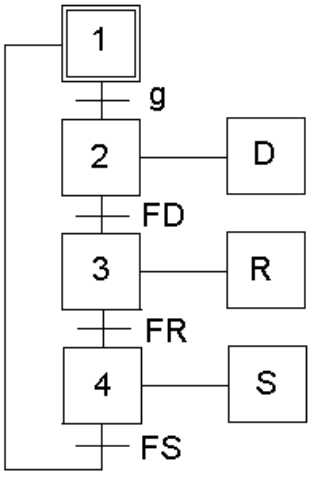
- Scelta seguita da sincronizzazione (errata: il programma si blocca);
- Parallelismo seguito da convergenza (ambigua: l'esecuzione del programma può risultare difficilmente prevedibile).

**ESEMPI.** Si riportano qui di seguito alcuni esempi molto semplici per illustrare come programmare un PLC mediante il linguaggio SFC. Tali esempi verranno ripresi nel Cap. 3 dove verranno implementati nel PLC Siemens Simatic S7-1200.

## Esempio: carrello trasportatore

Si vuole controllare un carrello per il trasporto di una merce da un luogo ad un altro. Quando il magazzino è pieno, viene inoltrata la chiamata al carrello, tramite un comando "g" (che può essere impartito da un operatore oppure essere basato sui dati provenienti da un sensore). Il carrello si sposta verso destra (attuatore D) fino ad arrivare alla fine della rotaia dove un sensore di fine corsa (FD) avverte che il carrello è arrivato a destinazione. Qui il magazzino viene svuotato (attuatore R) nel carrello e una volta terminata l'operazione un sensore FR emette un segnale. A questo punto il carrello comincia a muoversi a sinistra (attuatore S) fino ad arrivare a destinazione (fine corsa FS).




Dalla Fase 1 si passa alla fase 2 se arriva la richiesta di scarico del magazzino g.

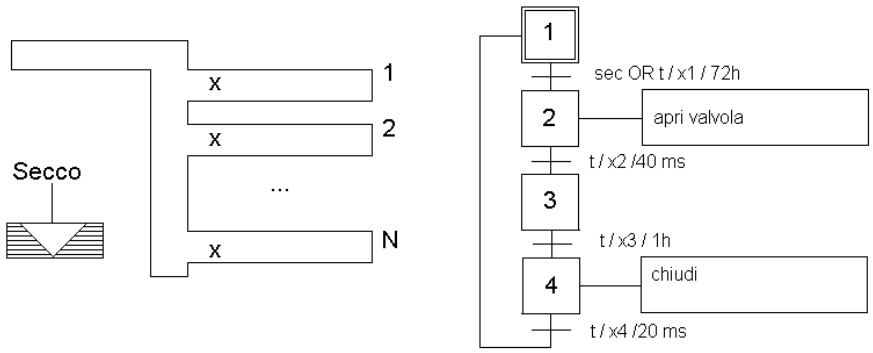
Dalla Fase 2 (spostamento D verso destra del carrello) si passa alla fase 3 se il carrello arriva al fine corsa destro FD.

Dalla Fase 3 (rovesciamento del magazzino R) si passa alla fase 4 se il fine corsa del magazzino FR rivela lo scarico completato.

Dalla Fase 4 (spostamento S verso sinistra del carrello) si torna alla fase 1 se il carrello arriva al fine corsa sinistro FS.

**Esempio: sistema di irrigazione**

Si consideri il seguente problema di irrigazione. Sul terreno da irrigare è presente un sensore che controlla se il campo è secco. Il campo è dotato di N rami, ma poiché l'acqua non è sufficiente per irrigare contemporaneamente tutto il campo, si è optato per l'irrigazione di un ramo alla volta, un'ora per ramo. Il campo viene irrigato ogni 72 ore o in caso di campo secco (sensore sec). Si tenga conto che la valvola di ogni ramo impiega 40ms per aprirsi e 20 ms per chiudersi.




N.B. La figura sopra è fatta per il caso di N = 1 (un solo ramo). Per N rami, occorrerebbe ripetere N volte il tratto compreso tra la fase 2 (inclusa) e la transizione t/x4/20ms (inclusa).

Dalla Fase 1 si passa alla fase 2 se il rivelatore di terreno secco è attivo o se sono passate 72 ore.

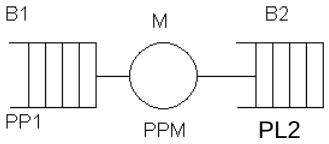
Dalla Fase 2 (apertura della valvola) si passa alla fase 3 se sono trascorsi 40 ms, il tempo necessario ad aprire la valvola.

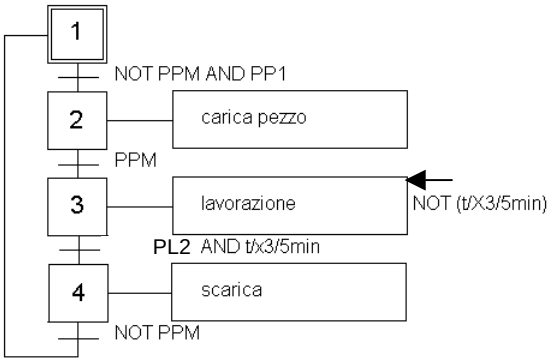
Dalla Fase 3 (aspetta che tutto sia innaffiato) si passa alla fase 4 dopo un'ora.

Dalla fase 4 (chiusura della valvola) si torna alla fase 1 se sono trascorsi 20 ms, il tempo necessario per chiudere la valvola.

**Esempio: processo di produzione**

Si controlli un processo produttivo, sapendo che il tempo di lavorazione di ogni pezzo è di 5 minuti e che le azioni da compiere sono: “carica pezzo”, “scarica pezzo” e “lavora pezzo”. Il processo è dotato di una macchina per il lavoro, e di due magazzini: uno in ingresso (B1) ed uno di uscita (B2). Il magazzino in ingresso possiede un sensore (PP1) che fornisce 1 se è presente almeno un pezzo in B1. Il magazzino di uscita possiede un sensore (PL2) che fornisce 1 se il magazzino B2 ha almeno un posto libero. La macchina (M) possiede un sensore (PPM) che fornisce 1 se la macchina è occupata, ovvero se sta lavorando già un pezzo.




Dalla Fase 1 si passa alla fase 2 se non ci sono pezzi nella macchina e contemporaneamente se ci sono pezzi in coda al buffer B1.

Dalla Fase 2 (caricamento del pezzo) si passa alla fase 3 se il pezzo è stato effettivamente caricato.

Dalla Fase 3 (lavorazione del pezzo) si passa alla fase 4 se sono contemporaneamente vere due ipotesi: sono trascorsi 5 minuti e c’è posto nel buffer B2 di uscita. In tutti i modi la lavorazione dura solo 5 minuti grazie alla condizione posta sull’azione associata alla fase 3 (azione condizionata).

Dalla Fase 4 (scarica pezzo) si torna alla fase 1 se la macchina è stata effettivamente liberata del pezzo lavorato.

# Capitolo 3

# Programmazione di un PLC Siemens Simatic S7-1200 mediante TIA Portal V13

I passi da seguire per sviluppare un progetto in TIA Portal V13 sono i seguenti:

1) Nella vista Portale inserire il dispositivo di controllo (Simatic S7-1200, CPU 1214C AC/DC/RY, 6ES7 214-1BG40-0XB0)

2) Si entra quindi nella vista Progetto e qui occorre inserire le variabili che verranno utilizzate nel programma all'interno della Tabella delle variabili standard (menu a sinistra Variabili PLC dentro PLC_1). Da osservare che le variabili di ingresso sono designate con una sigla del tipo Ix.y (con y intero da 0 a 7), le uscite con Qx.y e le variabili interne con Mx.y.

3) Scrivere ora il programma utilizzando il linguaggio Ladder (KOP nell'ambiente TIA Portal) andando dentro Blocchi programma (sempre dentro PLC_1) e cliccando due volte su Main [OB1]

4) Prima di caricare il programma sul PLC verificare che la CPU sia su STOP (led run/stop arancione). Altrimenti fermarla con l'apposito pulsante sul menu in alto a destra. Se però il PLC non è ancora stato collegato al PC (pur essendo già fisicamente connesso ad esso), questo comando apre una finestra per il collegamento:

- selezionare PN/IE e scheda di rete in uso sul PC;

- fare avvia ricerca: se tutto va bene si dovrebbe stabilire la connessione.

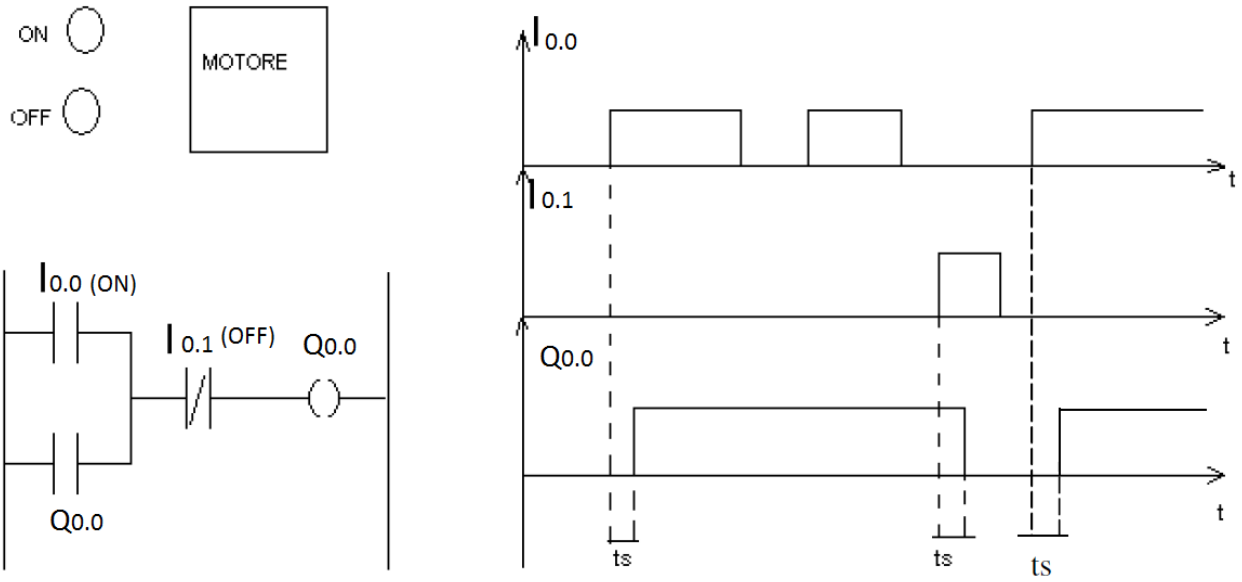
5) Caricare il programma con il pulsante in alto al centro. Se tutto va bene (in particolare la compilazione), spuntare Avvia tutto e poi pigiare su Fine. Se il PLC stava già in modalità STOP e quindi si è omesso il passo precedente, può essere necessario procedere al collegamento PLC-PC come detto al punto 4.

A questo punto, il programma funziona sul PLC anche se si interrompe (fisicamente) il collegamento col PC. Il programma in azione può anche essere visto sul PC se si fa Collega online e si pigia sugli occhialini nei pulsanti sopra il programma stesso.

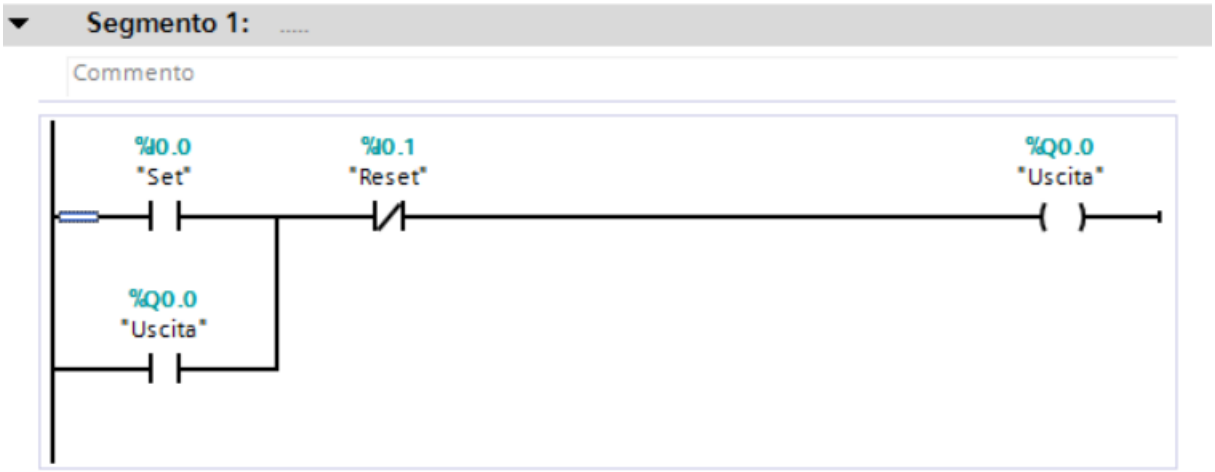
Si riportano di seguito alcuni semplici esempi.

## Esempio 1. Set-Reset (flip/flop)

**Riprendiamo il primo esempio introdotto qualche pagina fa per descrivere il linguaggio a contatti e mostriamo come sia possibile implementarlo sul PLC Siemens considerato.**

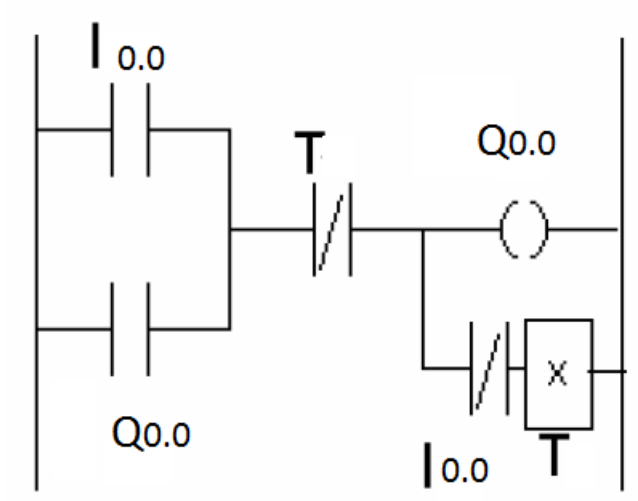



**Il nome del progetto su TIA Portal è *SetReset*. E' interessante osservare come il PLC, una volta caricato questo programma, riproduca il comportamento illustrato nel grafico riportato a destra nella figura precedente.**

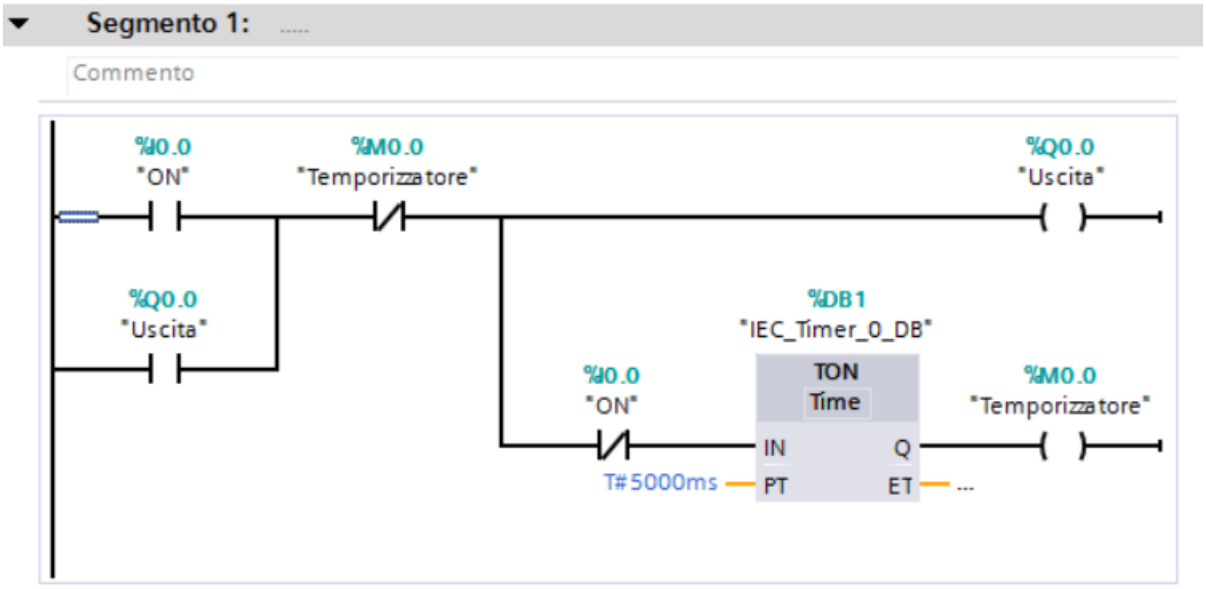


## Esempio 2. Ventola raffreddamento.

**L'altro esempio riportato nel testo per descrivere il funzionamento di un temporizzatore nell'ambito del linguaggio Ladder è quello qui riproposto. Si ottiene un'uscita (Q0.0) che vale uno quando l'ingresso (I0.0) è uno e torna a zero solo dopo che l'ingresso è tornato a zero da un certo tempo (come la ventola di raffreddamento di un dispositivo quale un proiettore).**

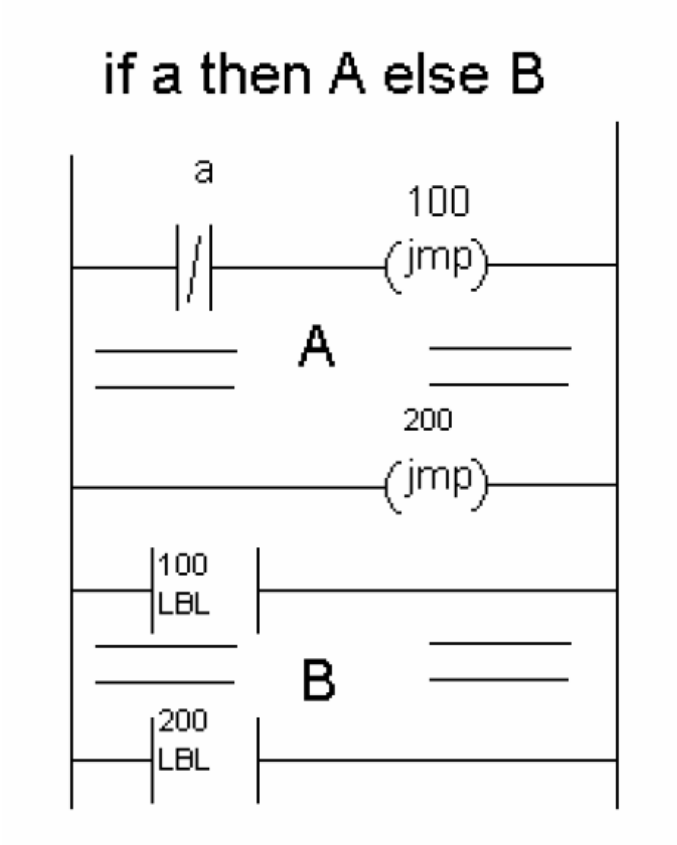



**Il nome del progetto su TIA Portal è *Ventola*. Il temporizzatore si comporta graficamente in modo diverso su TIA Portal da come descritto qualche pagina fa: non c'è in realtà una variabile T associata al temporizzatore ma un'uscita Q (vedi figura sotto) che diventa uno quando IN vale uno da almeno PT unità di tempo (anche qui il conteggio del tempo riparte ogni qualvolta IN torna a zero).**



## Esempio 3. If-then-else.

**Si consideri ancora uno degli esempi di programma scritto in linguaggio Ladder riportato qualche pagina fa.**




**Alla pagina seguente l'implementazione su TIA Portal. L'istruzione implementata è:**

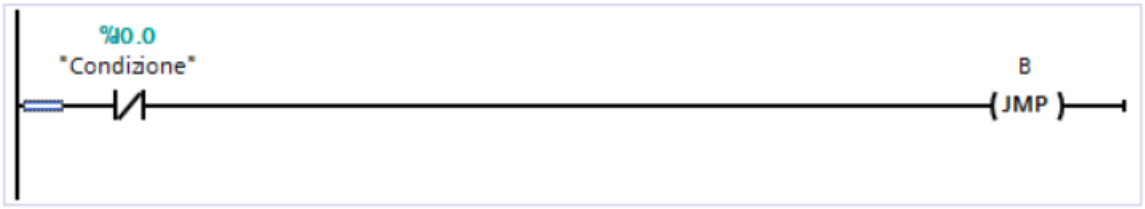
***if condizione then casoA else casoB***

**Il programma è completamente diverso da come viene riportato nella figura precedente perché nel linguaggio supportato da TIA Portal non potevano esserci due Jump nello stesso segmento e la label viene sempre messa all'inizio di un segmento.**

**E' stato anche necessario mettere una variabile superflua (*dummy*) perché non era possibile fare un segmento con sola etichetta. Tuttavia, in un programma normale, è ragionevole supporre che ci siano altre istruzioni dopo l'if-then-else, per cui ci saranno queste invece della variabile *dummy*.**

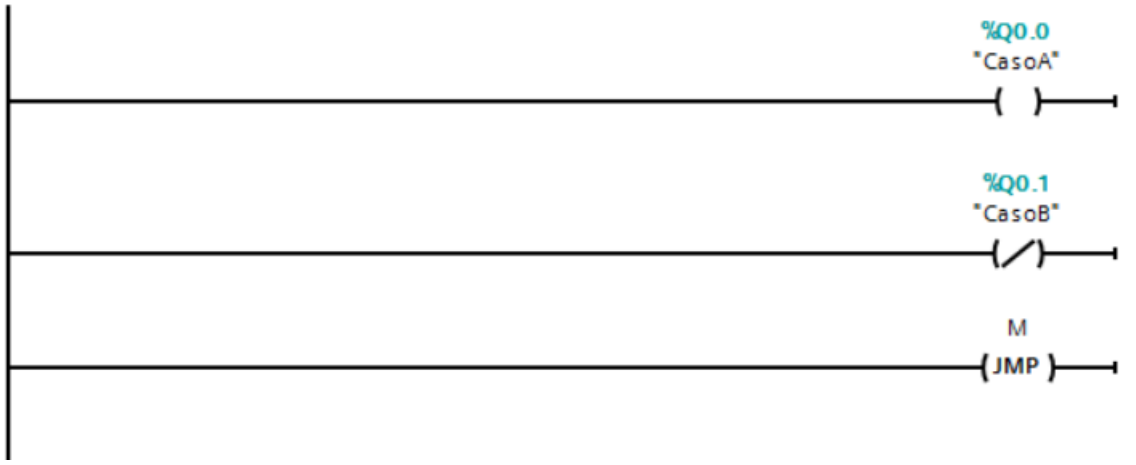
**Segmento 1:** .....

Commento




**Segmento 2:** .....

Commento

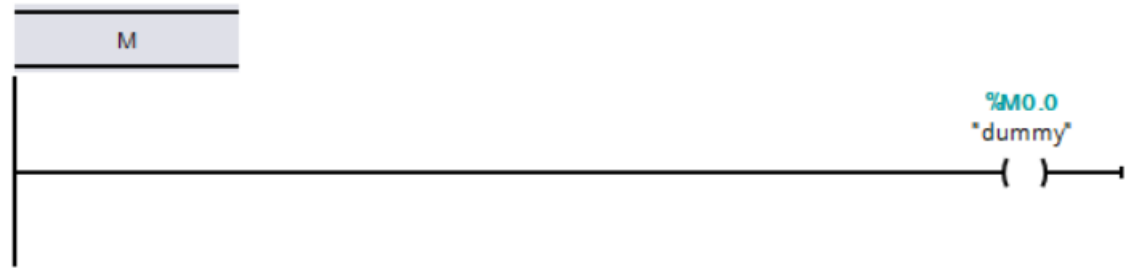



**Segmento 3:** .....

Commento

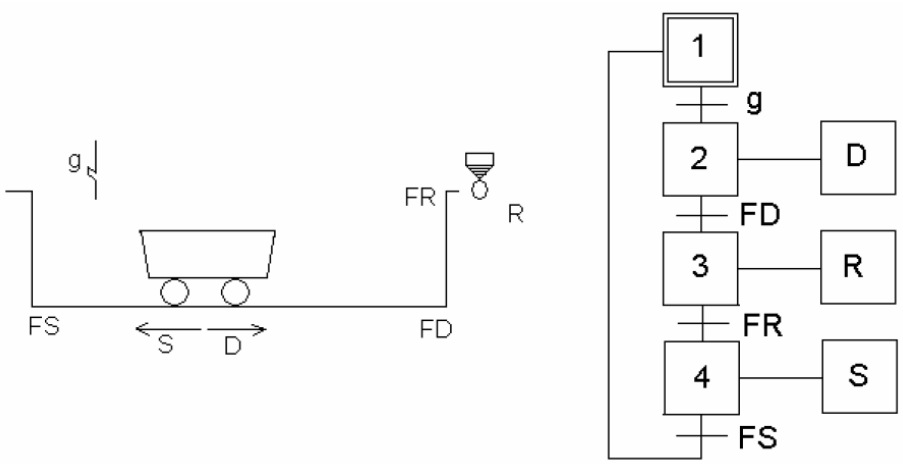

**Segmento 4:** .....

Commento



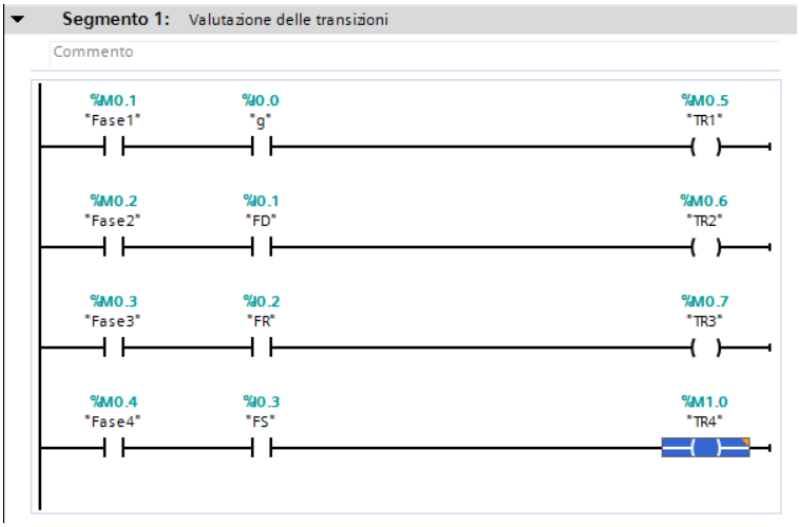
## Esempio 4. Carrello automatico.

**Si consideri il primo esempio di programma scritto in linguaggio SFC riportato qualche pagina fa.**

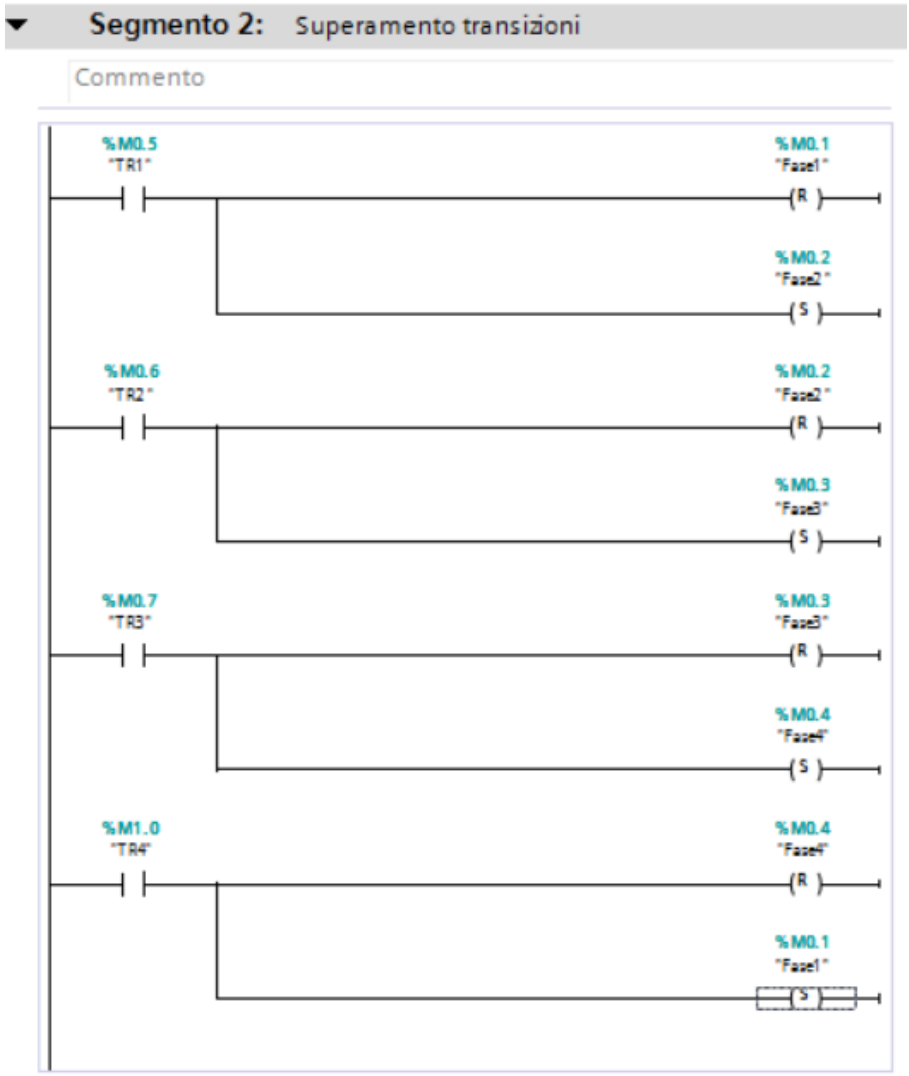



**Occorre prima tradurlo in linguaggio a contatti. Si seguiranno qui le regole descritte nel testo di Chiacchio e Basile (cfr. sezione bibliografica sul sito del corso). La procedura seguita in questo esercizio vale in generale per l'implementazione di qualsiasi programma SFC su TIA Portal. Innanzitutto va usato un blocco [OB100] che ha la proprietà di essere eseguito solo alla prima scansione del programma e che inizializza a uno la fase iniziale (Fase1).**

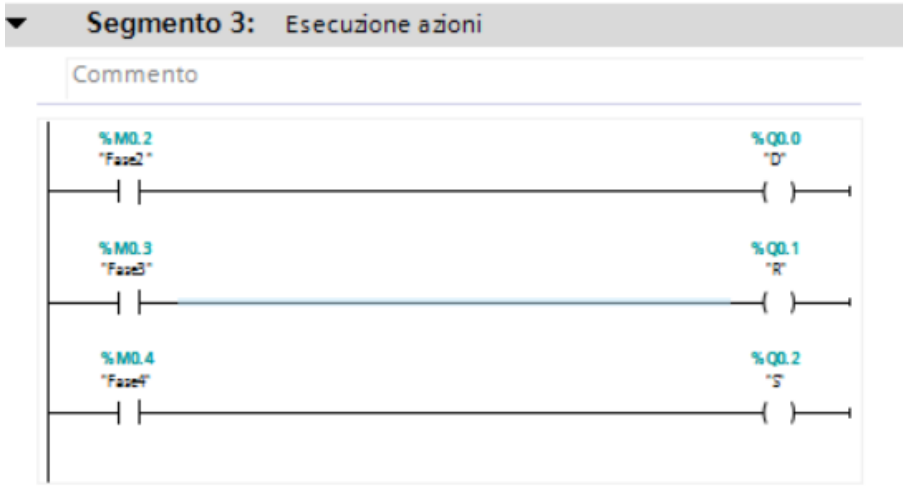

**Sul Main[OB1] (quello che viene eseguito sempre ciclicamente dal PLC) ci sono 3 segmenti. Il primo è di *Valutazione delle transizioni*. Nel caso specifico (progetto *carrelloAutomatico*):**



**Segue poi un segmento di *Superamento delle transizioni* in cui vengono disattivate le fasi a monte e attivate quelle a valle delle varie transizioni superate. Si noti che a tale scopo vengono usate bobine di Set e Reset.**

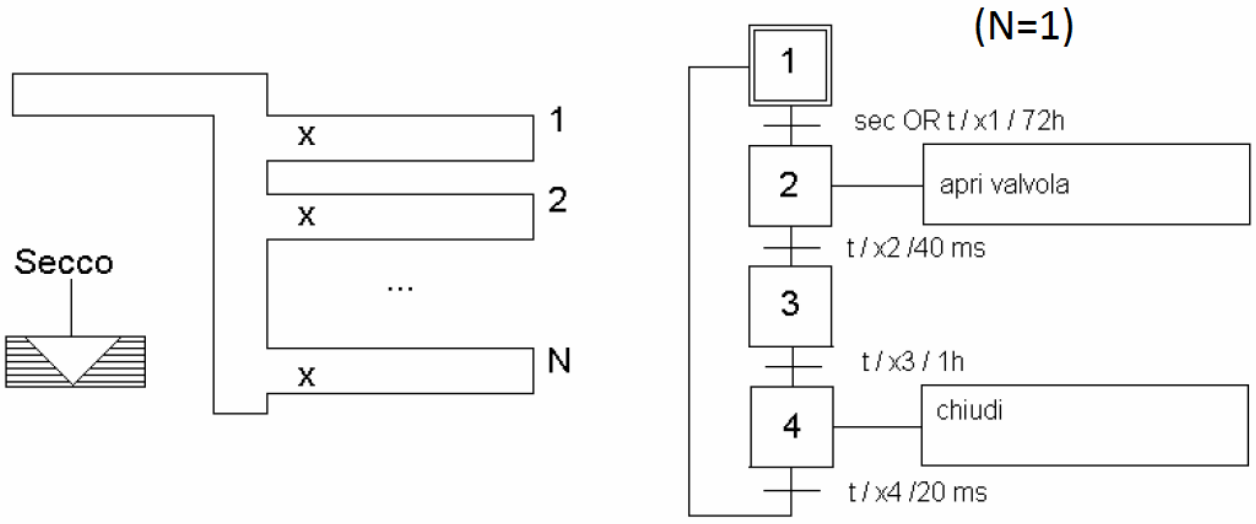



**Infine, un ultimo segmento serve ad *eseguire le azioni* associate alle fasi:**

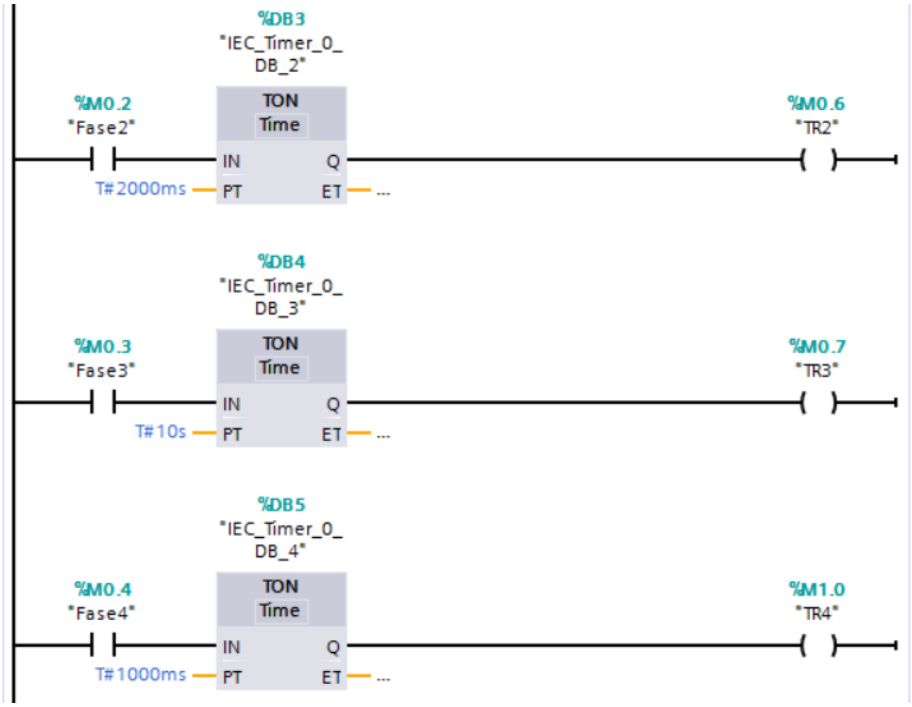


## Esempio 5. Impianto di irrigazione (un solo ramo).

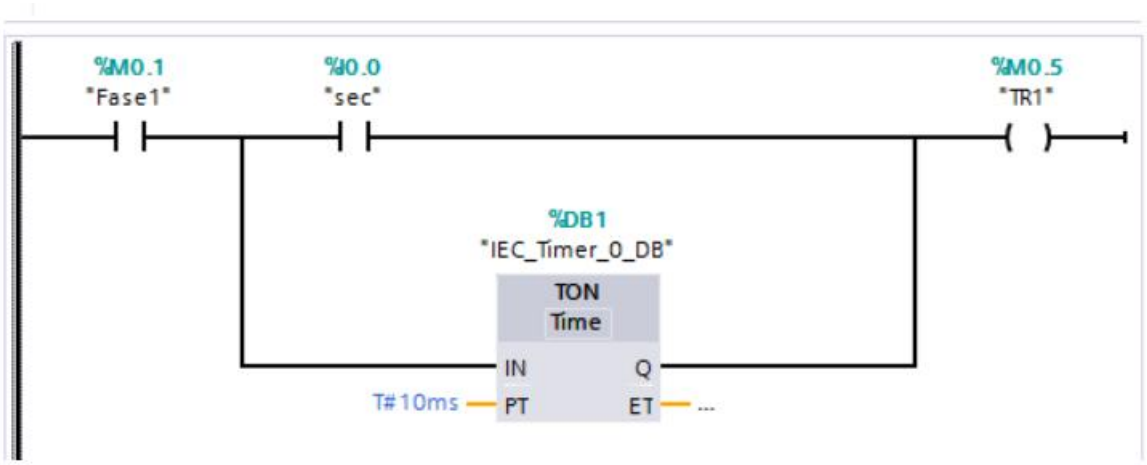
**Si consideri ora il secondo esempio di programma in linguaggio SFC del Cap. 2.**



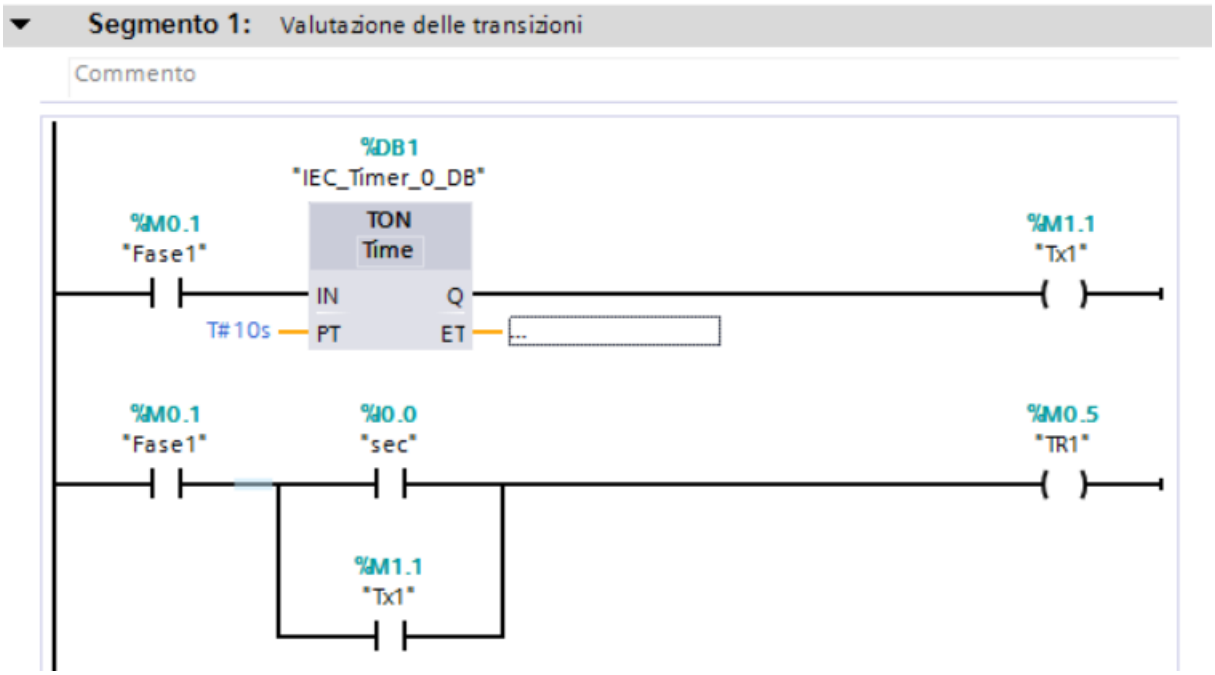

**La soluzione su TIA Portal (progetto *irrigazione*) è simile a quella dell'esercizio precedente con stesso blocco di inizializzazione [OB100] e, nel blocco principale [OB1], stesso segmento per il *Superamento delle transizioni* (sono infatti 4 fasi anche qui). Analogo è il segmento delle *azioni* che qui sono associate solo alla Fase 2 (Apri) e 4 (Chiudi). Abbastanza diverso invece è il segmento di *Valutazione delle transizioni*. Infatti qui le condizioni associate alle transizioni sono delle variabili temporali associate alle fasi. Queste possono in generale essere effettuate nel seguente modo per le transizioni da 2 a 4, utilizzando un temporizzatore TON (i tempi sono diversi da quelli dell'SFC riportato sopra per apprezzarli praticamente):**



**In sostanza, quando si è in una fase per un tempo maggiore di quello indicato in PT, Q diventa 1 e la variabile di transizione TR diviene vera, segnalando che può essere superata. Il problema nasce nella prima fase dove c'è un OR logico tra t/x1/72h e il sensore di umidità SEC. Una struttura come la seguente, che pure sarebbe corretta in linea di principio, non è ammessa nel linguaggio KOP implementato su TIA Portal perché dice che rami in parallelo possono contenere solo contatti:**

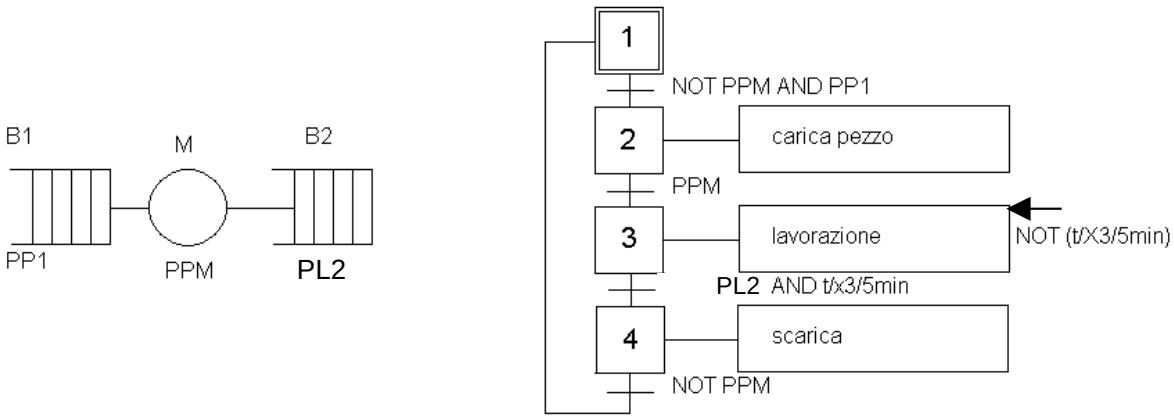



**Allora la precedente è stata trasformata nel seguente modo equivalente e ammissibile nel linguaggio KOP. In sostanza la variabile Tx1 diviene uno quando sono nella Fase 1 da almeno PT unità di tempo. La transizione è superabile quando vale 1 dunque l'OR tra Tx1 e SEC:**

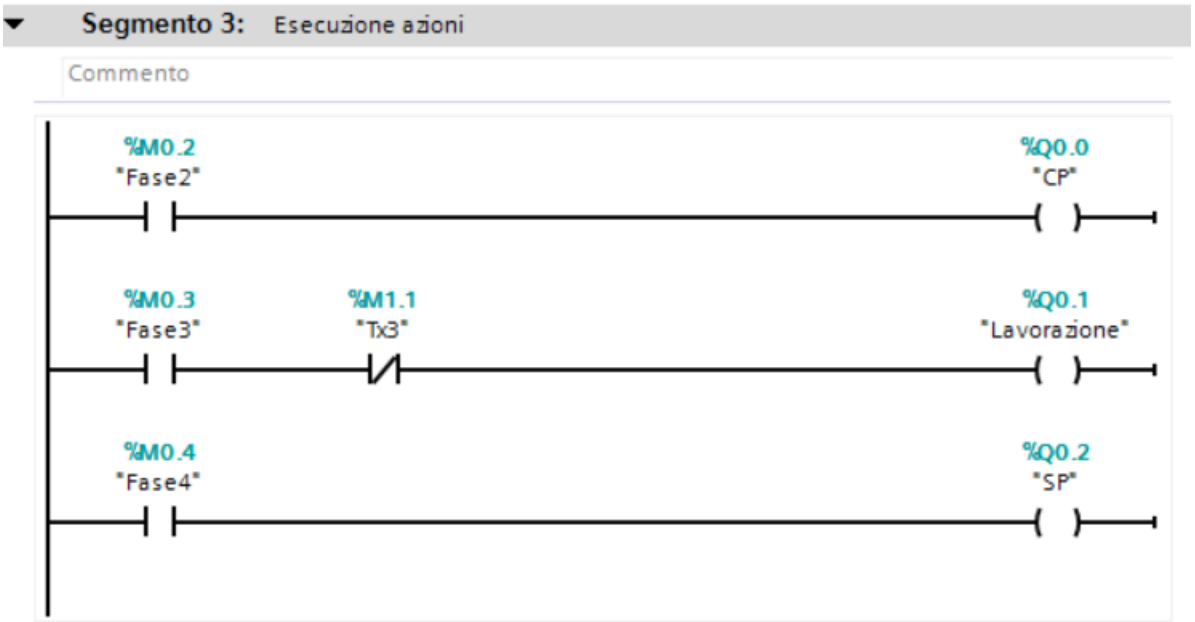


## Esempio 6. Sistema di produzione.

**Si consideri il terzo esempio di programma in linguaggio SFC proposto alla fine del Cap. 2.**

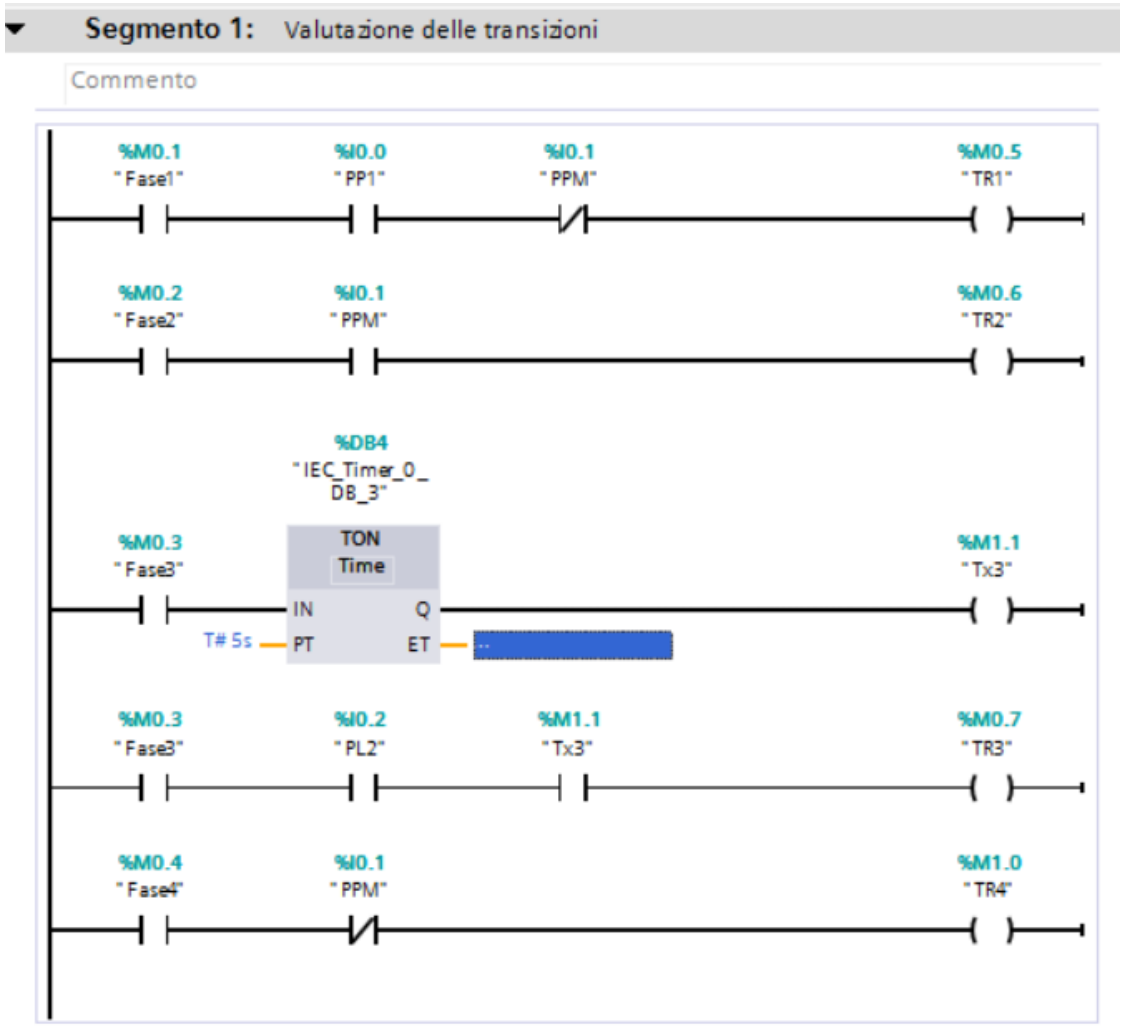



**La soluzione su TIA Portal (progetto *sistemaProduzione*) è simile a quella dell'esercizio precedente con stesso blocco di inizializzazione [OB100] e, nel blocco principale [OB1], stesso segmento per il *superamento delle transizioni* (sono infatti ancora 4 le fasi dell'SFC). Analogo e facilmente comprensibile è il segmento delle *azioni*, in cui la variante principale è quella di avere delle azioni condizionate. In particolare, quella della fase 3, va eseguita solo per 5 minuti (5 secondi nel programma) e quindi occorre anche qui una variabile Tx3 che diventa uno dopo che sono passati 5 secondi che sono nella Fase 3 (tale variabile Tx3 serve anche, come si vedrà e come si è visto nel progetto precedente, per valutare la transizione alla fase 4).**

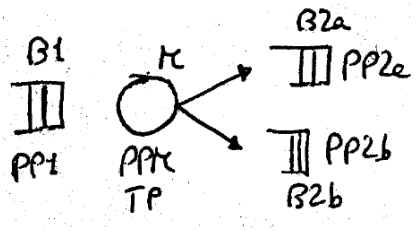



**Il segmento di *Valutazione delle transizioni* è facilmente deducibile in base ai progetti precedenti. In questo caso la condizione di transizione dalla fase 3 alla fase 4 è un AND tra la variabile temporale Tx3 associata alla fase 3 e PL2. Ciò si ottiene mettendo in serie un temporizzatore con un interruttore associato a PL2. Questo nel KOP del TIA Portal si può fare**

**senza problemi, tuttavia occorre fare attenzione a mettere nella serie suddetta il temporizzatore prima dell'interruttore perché altrimenti, se PL2=0, il conteggio del tempo che passo nella fase 3 non parte. Poiché però la variabile temporale Tx3 serve anche per condizionare l'azione della *Lavorazione*, tanto vale generarla e usarla anche, per maggiore chiarezza, in sede di valutazione delle transizioni.**



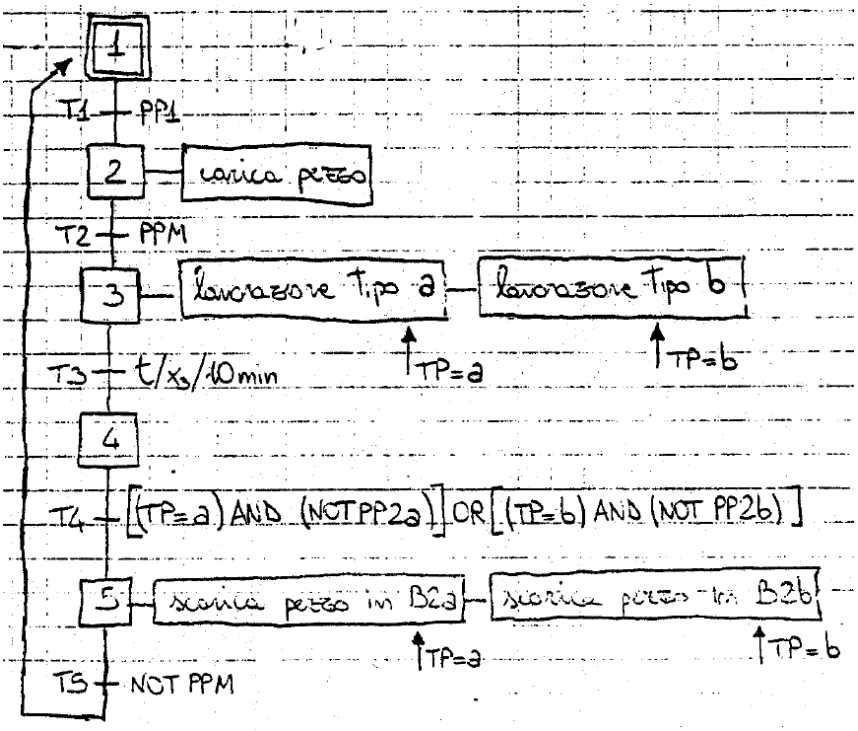
## Esempio 7. Ripreso dall'esame di AutMan del 12 giugno 2001.



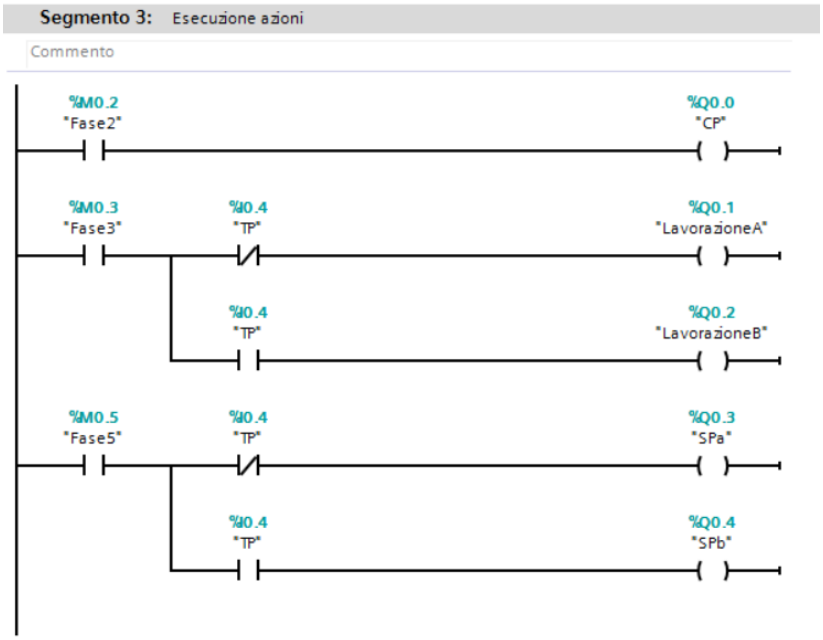

**Il testo del compito considerato è più o meno il seguente.**

**Si consideri il sistema di produzione indicato in figura. Il sistema è dotato dei seguenti sensori: PP1 = presenza pezzo nel buffer B1, PP2a = buffer B2a pieno (cioè PP2a=1 se B2a è pieno), PP2b = buffer B2b pieno, PPM = presenza pezzo nella macchina M (cioè PPM=1 se c'è un pezzo nella macchina), TP=0 se il pezzo in lavorazione è di tipo a, TP=1 se è di tipo b. Dal buffer B1 arrivano alla macchina pezzi di due tipi, a e b, in ordine casuale. La macchina carica un pezzo nuovo se non ci sono pezzi in essa e se c'è un pezzo disponibile sul buffer B1. Dopo aver caricato il nuovo pezzo, il sensore TP è in grado di capire se tale pezzo è di tipo a o b. A questo punto comincia la lavorazione che dipende dal tipo di pezzo e dura, in ogni caso, meno di 10 minuti. Ultimata la lavorazione, il pezzo va scaricato nel buffer corrispondente (pezzi a in B2a, pezzi b in B2b) se c'è posto, altrimenti la macchina attende che tale posto si liberi. Scrivere l'SFC del PLC di controllo.**

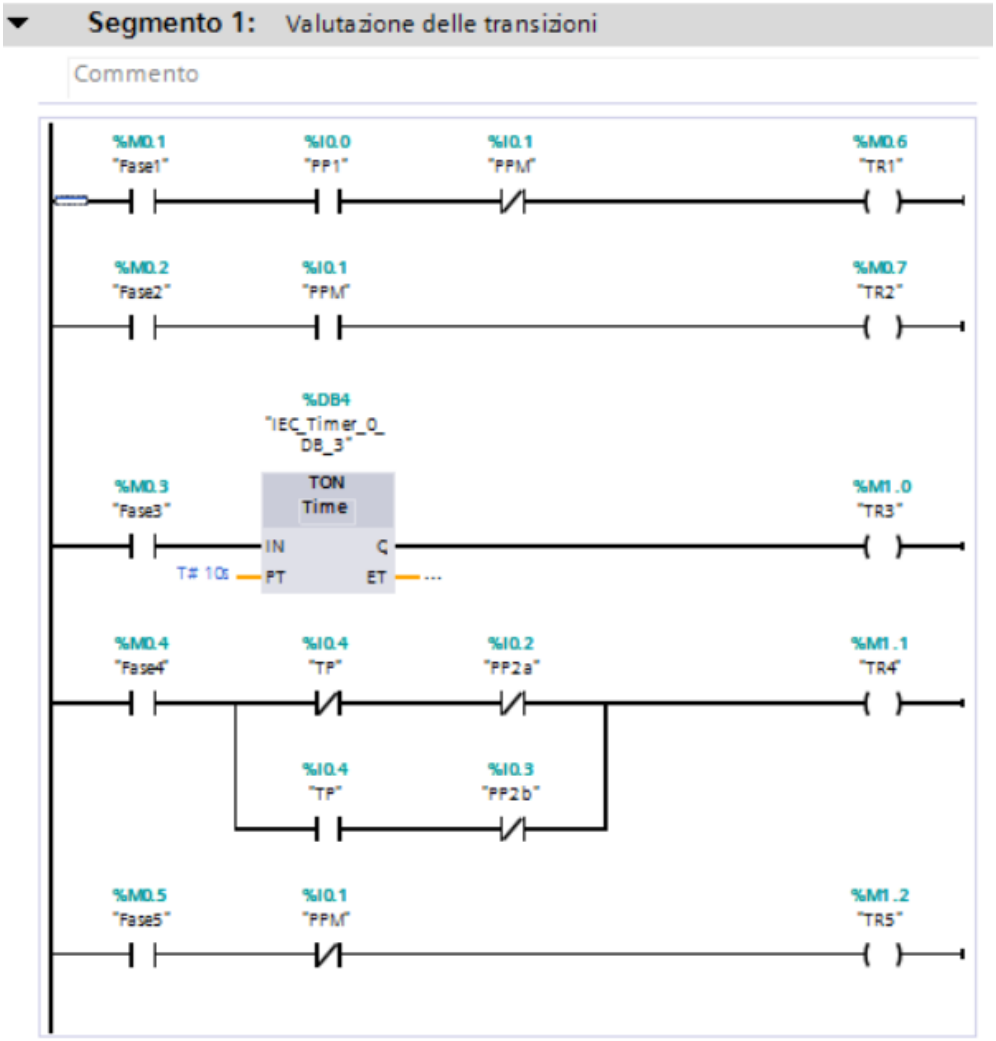
**Una soluzione molto compatta di questo esercizio è la seguente.**




**La soluzione su TIA Portal (progetto *esame12giu2001*) segue ancora l'impostazione degli esercizi precedenti, con stesso blocco di inizializzazione [OB100] e, nel blocco principale [OB1], stesso segmento per il *superamento delle transizioni* (con la sola trascurabile differenza che ora le fasi sono 5). Facilmente comprensibile è il segmento delle *azioni*, in cui la variante principale è quella che a una stessa fase possono essere associate più azioni condizionate (si vedano le Fasi 3 e 5).**

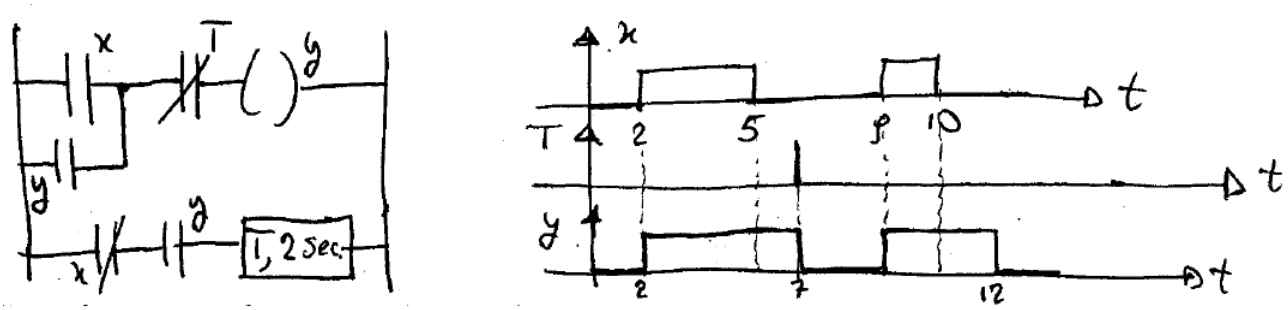



**Il segmento di *Valutazione delle transizioni* risulta anche piuttosto semplice e questo perché nell'SFC si è introdotta una fase 4 di eventuale attesa che permette di separare la condizione temporale da quella di disponibilità di spazio sul buffer a valle.**

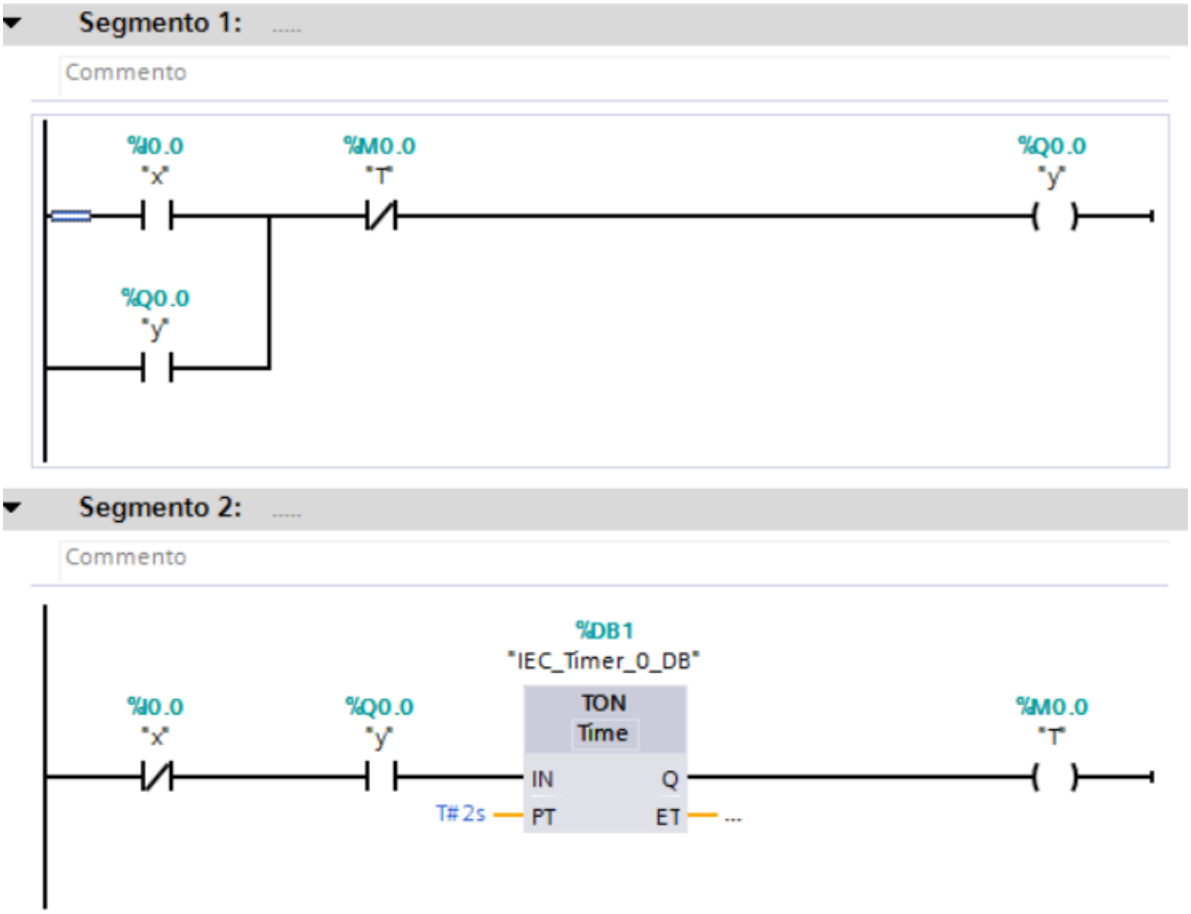


## Esempio 8. Ripreso dall'esame di AutMan dell'8 giugno 2004.

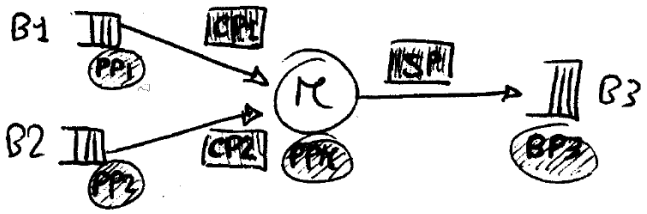
**Si tratta di un esempio di programma scritto già in linguaggio Ladder. Questo programma in effetti produce lo stesso comportamento dell'esempio 2 del Cap. 2 (Ventola di raffreddamento). Il programma e un suo possibile comportamento sono riportati nella figura seguente.**




**La soluzione su TIA Portal (progetto *esame8giu2004*) non è quindi molto diversa da quella dell'esempio 2, comprendendo solo un paio di segmenti nel blocco [OB1].**



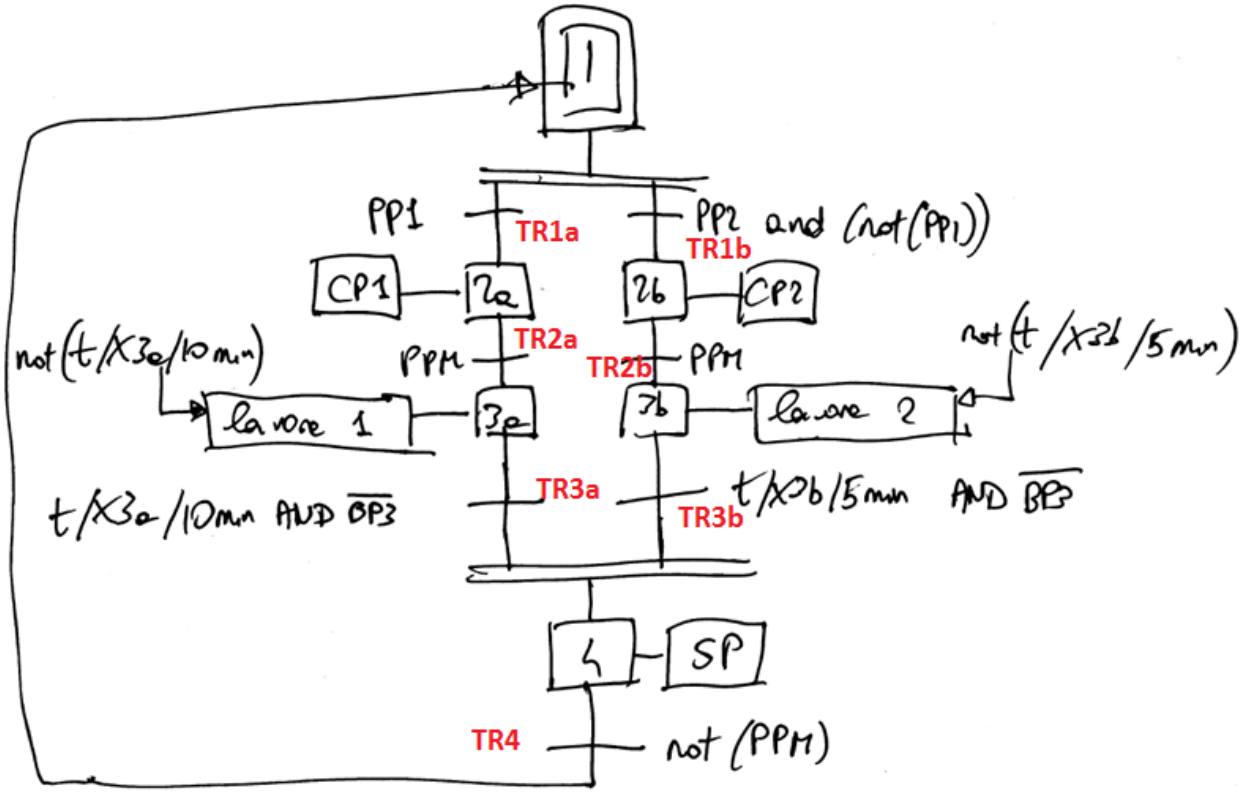
## Esempio 9. Ripreso dall'esame di AutMan del 7 settembre 2001.




**Il testo del compito considerato è più o meno il seguente.**

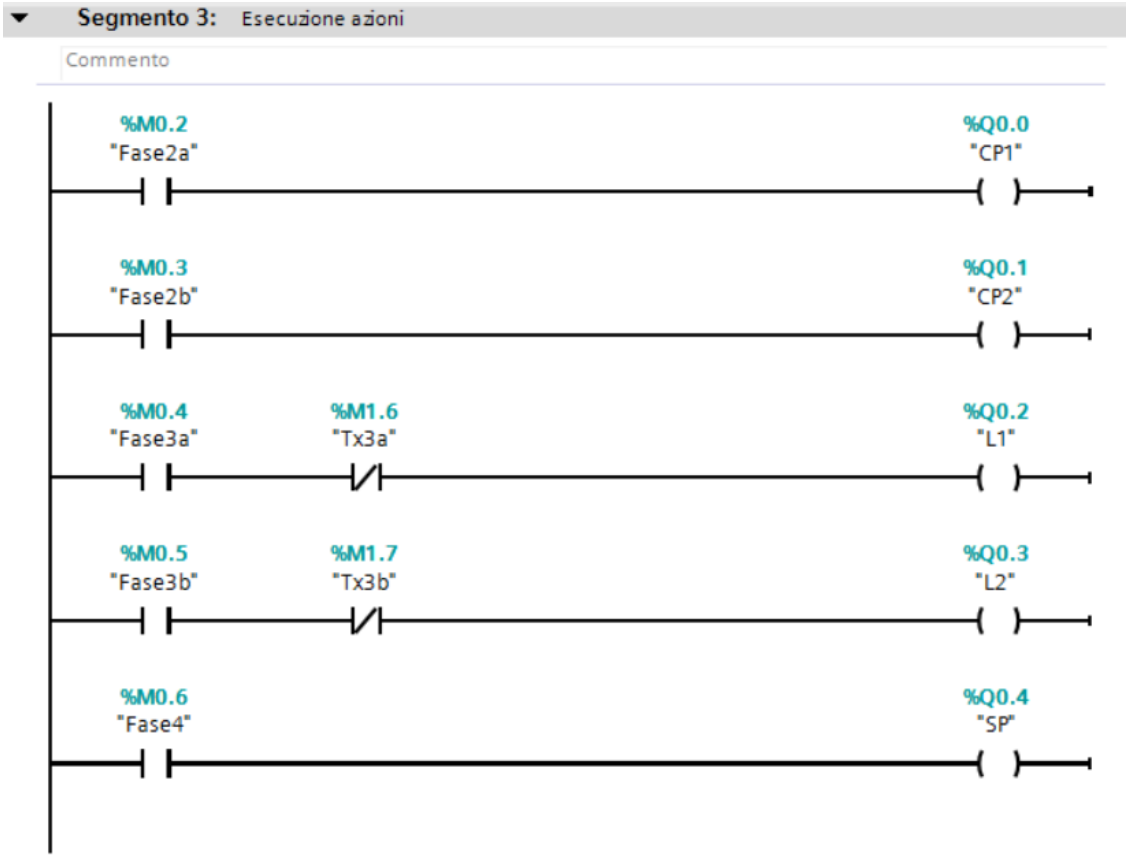
**Si consideri il sistema di produzione indicato in figura. Il sistema è dotato dei seguenti sensori: PP1 = presenza pezzo nel buffer B1, PP2 = presenza pezzo nel buffer B2, PPM = presenza pezzo nella macchina M, BP3 = buffer B3 pieno. Il sistema di controllo agisce coi seguenti comandi: CP1 = carica pezzo di tipo 1, CP2 = carica pezzo di tipo 2, SP = scarica pezzo. Sul buffer B1 arrivano pezzi di tipo 1 che richiedono una lavorazione di 10 minuti e hanno priorità maggiore rispetto ai pezzi di tipo 2 che arrivano sul buffer B2 e possono essere caricati sulla macchina per essere lavorati solo se non ci sono pezzi in attesa di tipo 1, con un tempo di lavorazione di 5 minuti. Scrivere il diagramma funzionale sequenziale (SFC) del PLC di controllo tenendo conto che la macchina non può smettere di lavorare un pezzo una volta che lo ha caricato e non può caricarne uno nuovo finché non ha scaricato il precedente.**

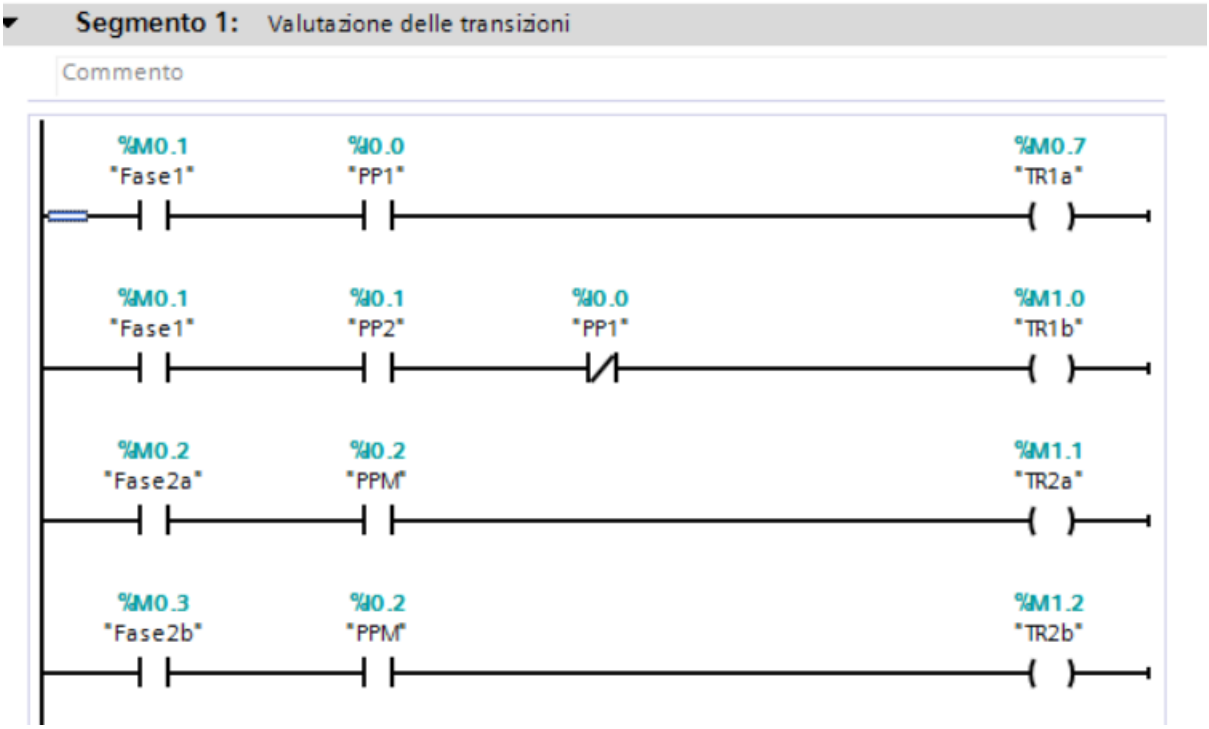
**Una possibile soluzione di questo esercizio è la seguente.**

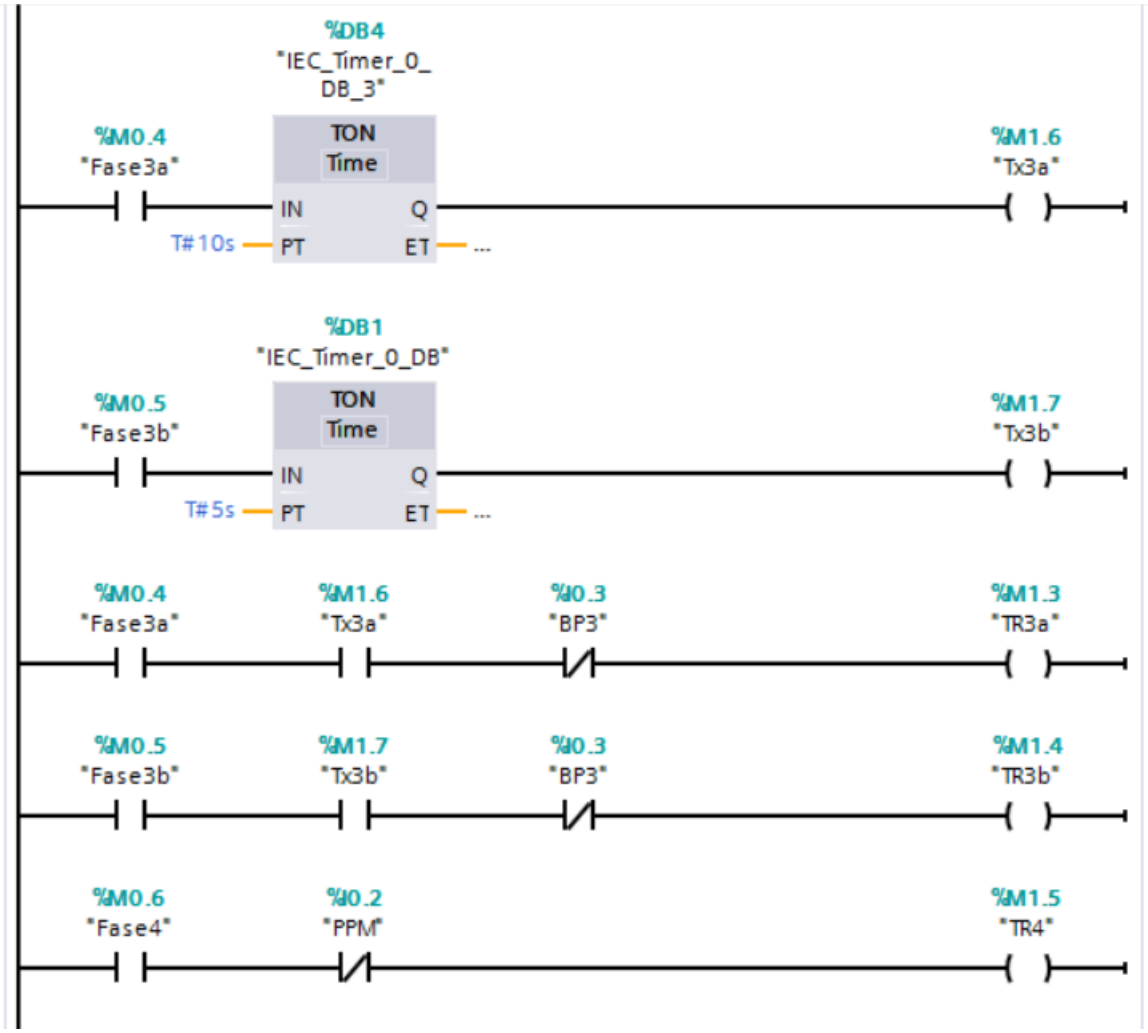



**Si riprende con gli esempi di linguaggio SFC la cui struttura va tradotta in Ladder. Qui le fasi sono 6. La soluzione su TIA Portal (progetto *esame7set2001*) segue quindi l'impostazione degli esercizi precedenti di questo tipo, con il solito blocco di inizializzazione [OB100]. Nel blocco principale [OB1] il segmento per il *Superamento delle transizioni* è grosso modo lo stesso dei casi**

**precedenti salvo ora la variante dei nomi delle fasi. Si è a tal proposito deciso di associare a ciascuna transizione la variabile TRx come indicato in rosso nella figura precedente. Si riportano quindi i blocchi di *Esecuzione delle azioni* e di *Valutazione delle transizioni*. Si noti che anche qui, come nell'Esempio 6, è stato necessario introdurre delle variabili Tx di temporizzazione.**

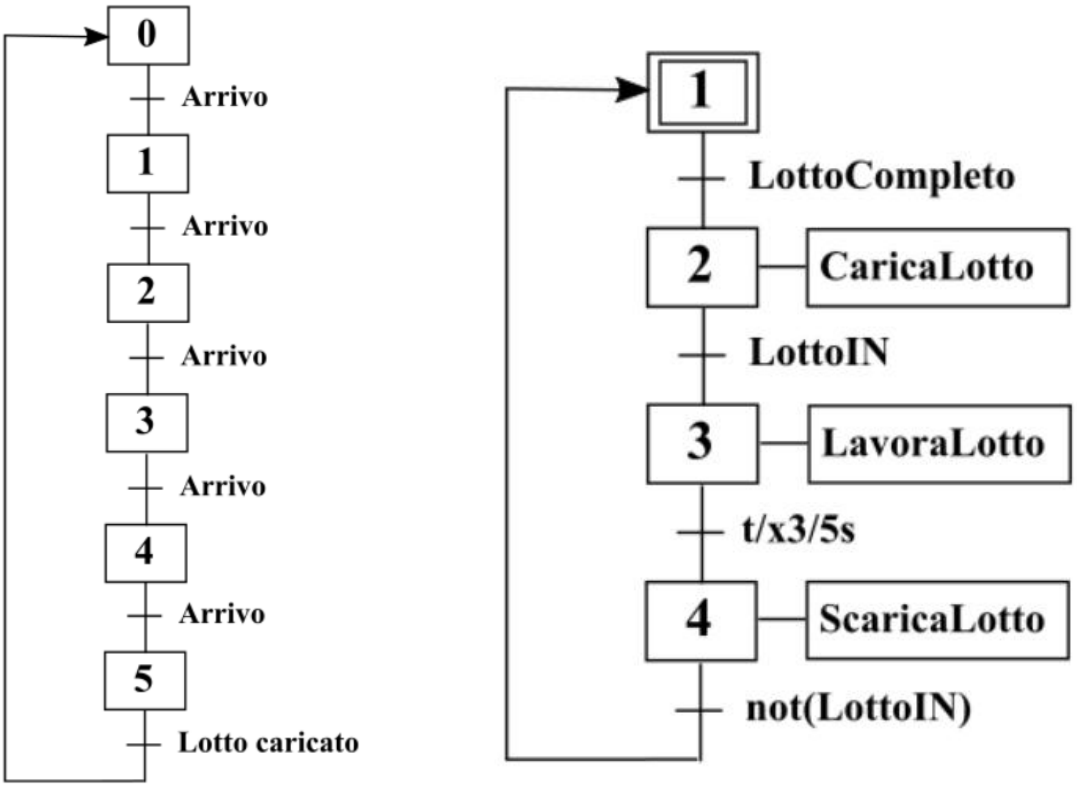


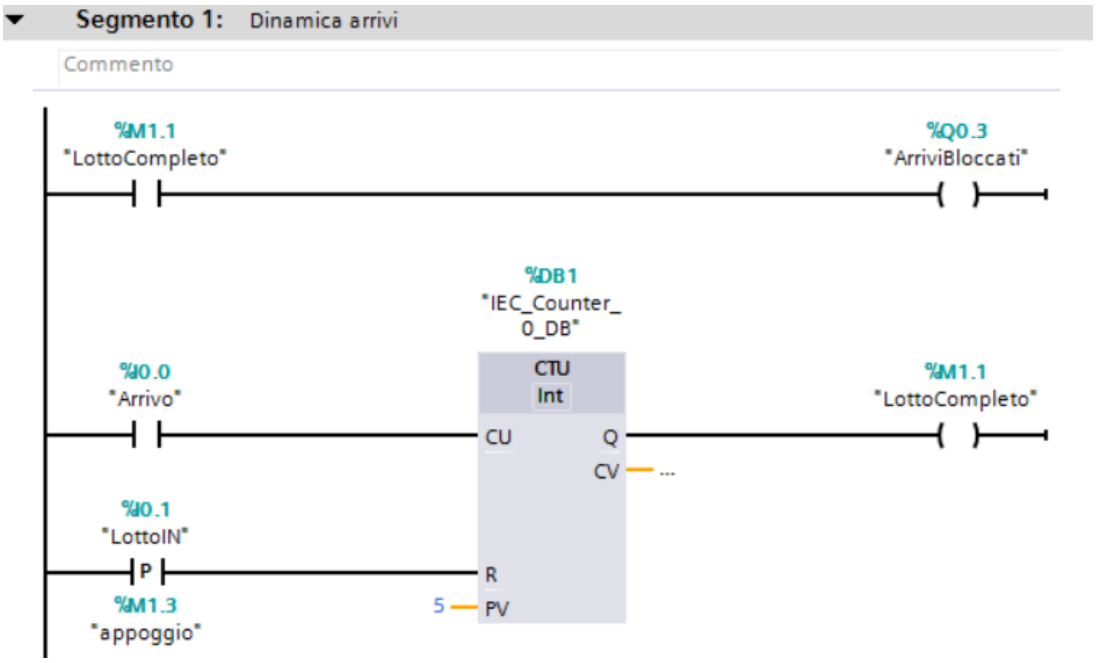

%DB4
"IEC_Timer_0_
DB_3"
%M0.4
"Fase3a"
TON
Time
IN
Q
T#10s
PT
ET
...
%M1.6
"Tx3a"
%DB1
"IEC_Timer_0_DB"
%M0.5
"Fase3b"
TON
Time
IN
Q
T#5s
PT
ET
...
%M1.7
"Tx3b"
%M0.4
"Fase3a"
%M1.6
"Tx3a"
%I0.3
"BP3"
%M1.3
"TR3a"
%M0.5
"Fase3b"
%M1.7
"Tx3b"
%I0.3
"BP3"
%M1.4
"TR3b"
%M0.6
"Fase4"
%I0.2
"PPM"
%M1.5
"TR4"

## Esempio 10. Lavorazione in lotti.

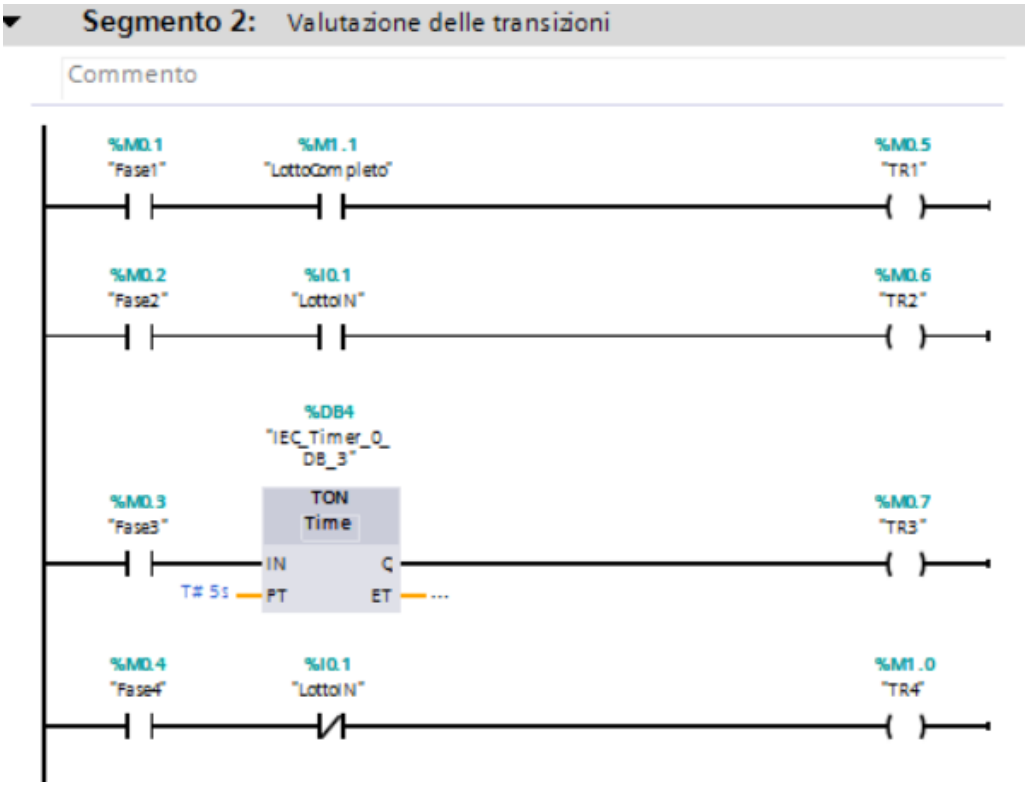
**In questo esempio si utilizza un contatore. Il problema considerato consiste in una macchina cui arrivano dei pezzi (evento *arrivo*). Quando il numero di pezzi arrivati ha raggiunto 5 (dimensione del lotto), il lotto viene caricato sulla macchina per essere lavorato. La lavorazione dura 5 secondi. Dopodiché il lotto viene scaricato. Gli arrivi vengono sempre accolti anche mentre la macchina lavora o scarica il lotto. Tuttavia quando il lotto è completo (numero pezzi = 5) gli arrivi vengono ignorati (ossia respinti). Solo quando il lotto viene caricato nella macchina il conteggio degli arrivi riprende. L'SFC considerato è il seguente (in realtà ce ne sono due, uno che descrive la dinamica degli arrivi, riportato a sinistra e solo implicitamente implementato nel programma, e un altro che descrive il sistema di controllo della macchina, riportato a destra).**




**Nell'implementazione su TIA Portal (progetto *produzioneLotti*) è stato inserito un segmento specifico per simulare la dinamica degli arrivi (e quindi l'SFC di sinistra). L'evento *Arrivo*, nella seconda riga del segmento (vedere sotto), va a incrementare un contatore di tipo CTU. Questo è un tipo di contatore che incrementa il conto a ogni fronte di salita che legge sull'ingresso CU e fa diventare uno la variabile Q quando tale conto diventa pari a PV. Un valore uno su R lo resetta a zero. Un valore uno della variabile Q segnala un *LottoCompleto* e quindi, prima riga del segmento, viene segnalato con una variabile di uscita (Q0.3) che gli arrivi sono bloccati (*ArriviBloccati*). L'interruttore -|P|-, che ha bisogno di una variabile di appoggio per lavorare, vale uno solo quando la variabile associata (*LottoIN*) passa da 0 a 1: in pratica, quando il lotto viene caricato, il conteggio degli arrivi riparte da zero. Questo segmento in definitiva simula l'SFC di sinistra (cioè la dinamica degli arrivi) con la condizione *Lotto Caricato* dell'ultima transizione resa da un interruttore -|P|- associato alla variabile *LottoIN*.**

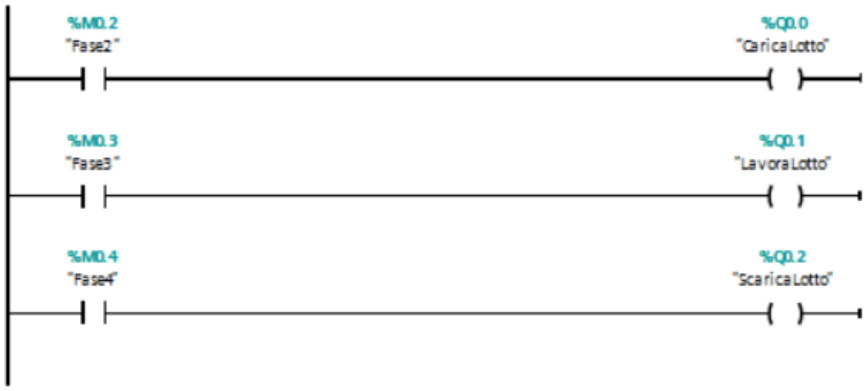



**Per il resto l'implementazione dell'SFC di controllo segue l'impostazione degli esercizi precedenti di questo tipo, con il solito blocco di inizializzazione [OB100]. Nel blocco principale [OB1] compare il segmento per il *Superamento delle transizioni* che è lo stesso dei casi precedenti e non viene riportato nel seguito. Si riportano invece i blocchi di *Esecuzione delle azioni* e di *Valutazione delle transizioni*, la cui struttura è abbastanza intuitiva.**



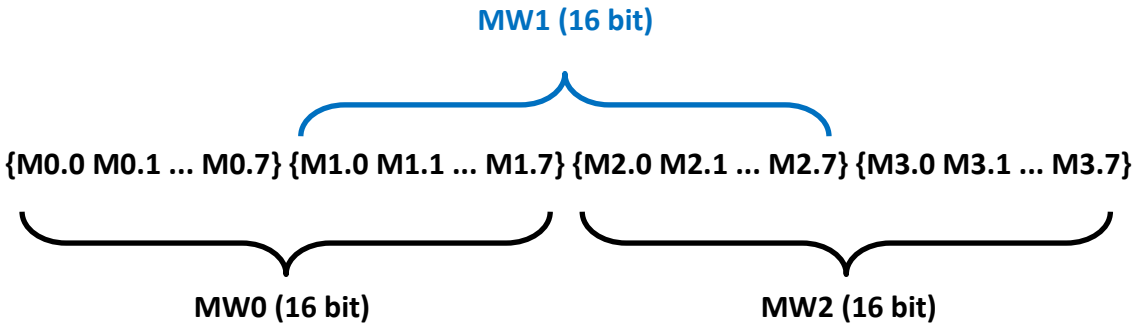
**Segmento 4:** Esecuzione azioni

Commento

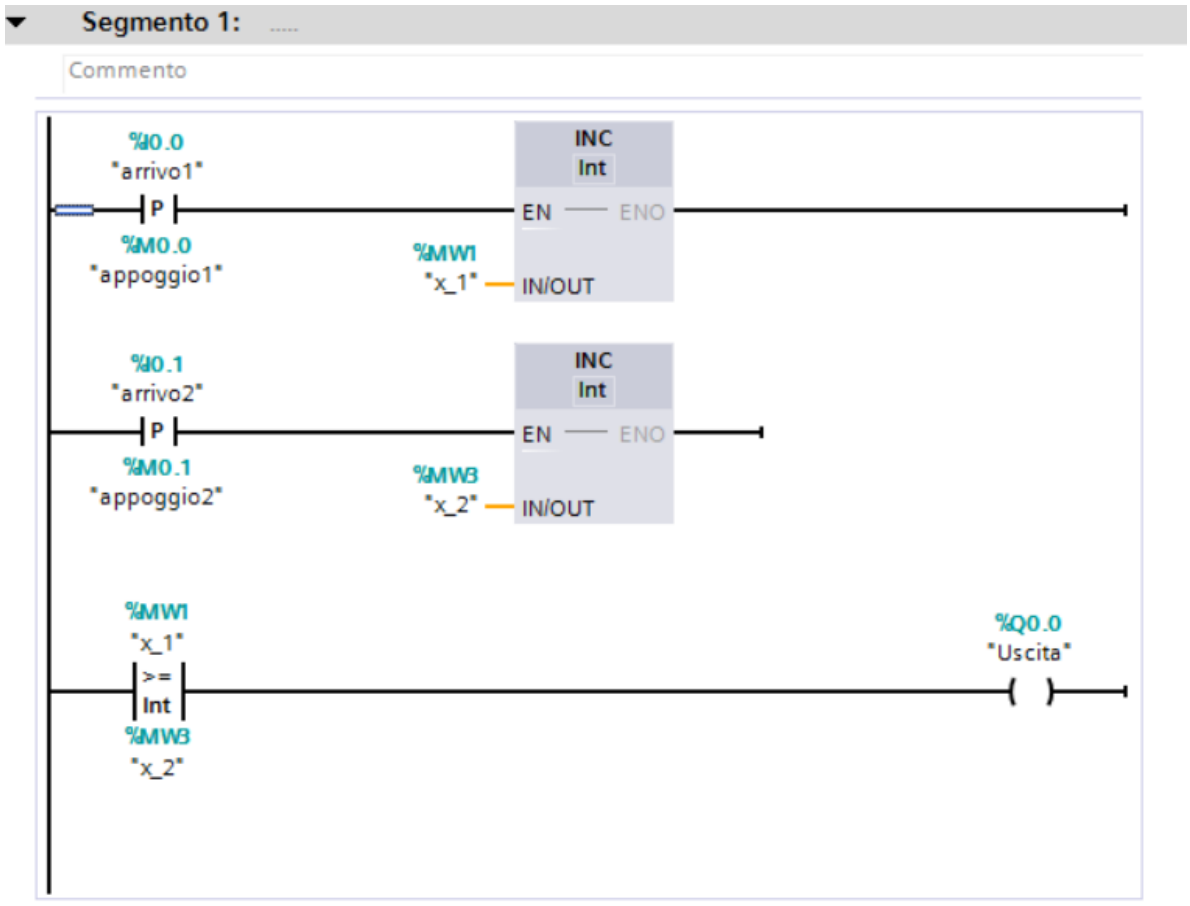


## Esempio 11. Dinamica di arrivi in due buffer.

**Questo esempio riguarda la dinamica di due buffer il cui contenuto, $x_1$ e $x_2$ rispettivamente, viene modificato dagli arrivi di nuovi pezzi. Le variabili $x_1$ e $x_2$ sono numeri interi e pertanto richiedono 16 bit, cioè una word. Per questo, nella tabella delle variabili, come indirizzo hanno rispettivamente MW1 e MW3. Attenzione: $x_2$ non poteva essere associato a MW2 perché MW2 comincia con il secondo byte della word MW1 (che occupa due byte). Anche bisogna fare attenzione a non usare M0.x, M1.x, M2.x e M3.x (con x=0,1,...,7) in quanto questi bit sono utilizzati nelle word MW1 e MW3. In sostanza la situazione è questa:**




**La dinamica degli arrivi è manuale: il fronte di salita dell'ingresso I0.0 incrementa di 1 $x_1$, il fronte di salita di I0.1 incrementa $x_2$. Il progetto su TIA Portal si chiama *arriviManuale* e il blocco OB1 (l'unico presente in questo caso) è così fatto:**



**Si nota l'interruttore di tipo P (già introdotto nel progetto precedente) che serve proprio a rilevare il fronte di salita della variabile I0.0 nel primo rung. Ha bisogno di una variabile di appoggio (M0.0 nel primo caso) per poter lavorare. Il blocco INC incrementa di uno la variabile collegata al pin IN/OUT (in questo caso $x_1$, cioè MW1) tutte le volte che vede uno sul pin EN.**

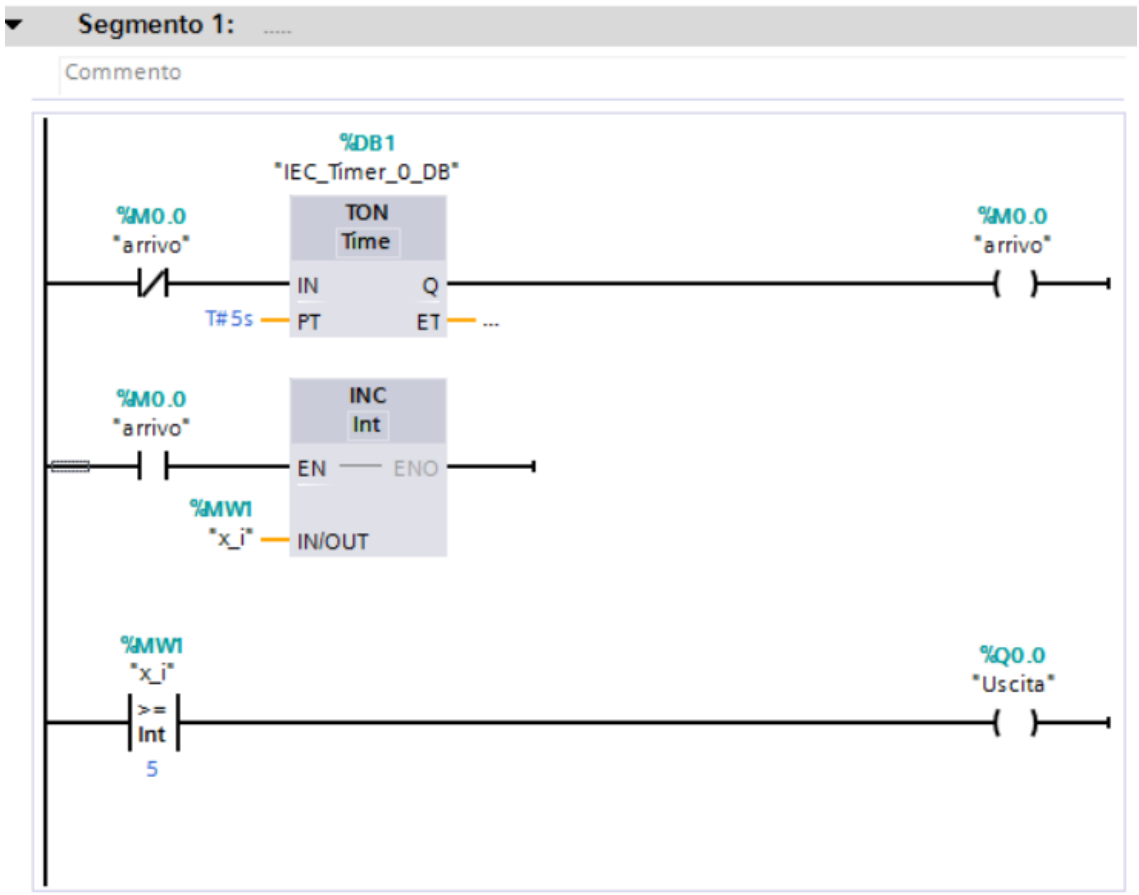
**Sul secondo rung si fa la stessa cosa per quanto concerne il secondo buffer.**

**Infine, nel terzo rung, si confrontano le due variabili $x_1$ e $x_2$: se $x_1 \geq x_2$, si imposta a uno l'uscita Q0.0 che altrimenti rimane a 0.**

## Esempio 12. Dinamica di arrivi automatica.

**Questo esempio è una variante dell'esempio precedente: gli arrivi non sono più comandati manualmente dagli interruttori collegati agli ingressi I0.0 e I0.1 ma si verificano in base a una temporizzazione. Si è scelto di considerare in questo caso la dinamica di un solo buffer (anziché i due del progetto precedente), il cui contenuto è stato genericamente indicato $x_i$.**

**Il progetto su TIA Portal si chiama *arriviAutomatico* e il blocco OB1 (l'unico presente in questo caso) è così fatto:**



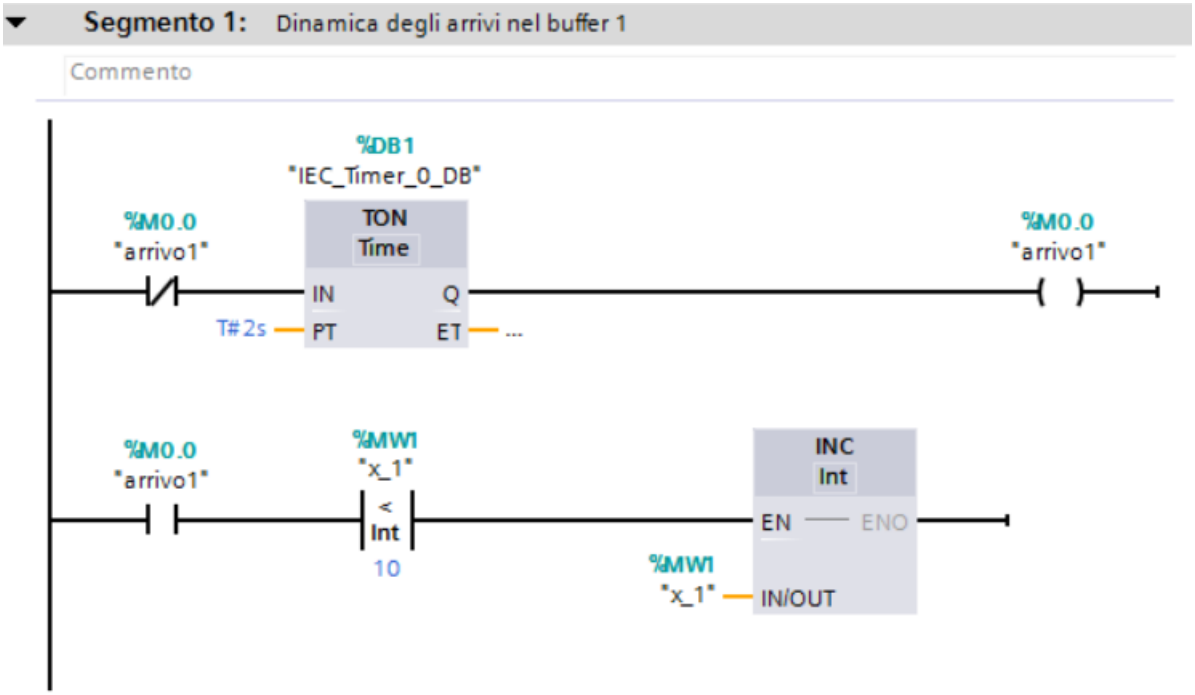

**In questo caso, il temporizzatore TON, finché l'ingresso IN vale 1 (cosa che accade finché M0.0, cioè *arrivo*, è 0) conta il tempo trascorso e assegna all'uscita Q (e quindi a M0.0, cioè ad *arrivo* stesso) il valore 1 quando tale conteggio ha raggiunto il tempo indicato in PT (5 secondi in questo caso). Con la variabile *arrivo* (cioè M0.0) pari a 1 il blocco INC incrementa di uno il buffer. Ma questo avviene (come deve essere) solo una volta perché nel ciclo successivo M0.0 pari a 1 fa aprire il contatto in ingresso al temporizzatore che viene quindi azzerato. Però come il temporizzatore viene azzerato, *arrivo* torna a 0 facendo ripartire il conteggio del tempo.**

**Nel terzo rung, analogamente all'esempio precedente, si confronta il valore di $x_i$ con 5: se $x_i \geq 5$, si imposta a uno l'uscita Q0.0 che altrimenti rimane a 0.**

## Esempio 13. Dinamica di arrivi in tre buffer automatica con capacità limitata.

**Questo esempio è una piccola variante dell'esempio precedente: la capacità dei buffer è limitata e gli arrivi vengono ignorati quando il contenuto del buffer ha raggiunto il suo valore massimo. Si è scelto di considerare in questo caso la dinamica di tre buffer, il cui contenuto è stato indicato $x_i$, i = 1,2,3.**

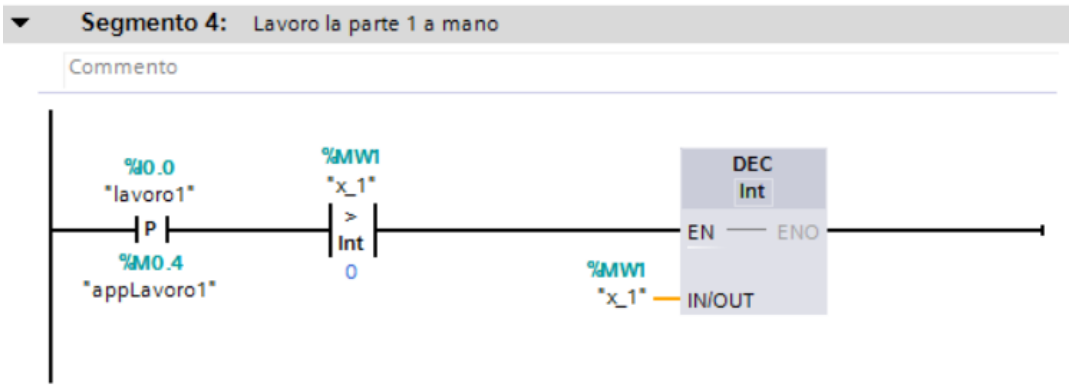
**Il progetto su TIA Portal si chiama *arriviAutomaticoCapLim* e il blocco OB1 (l'unico presente in questo caso) è sostanzialmente identico a quello del progetto precedente. L'unica differenza è il contatto sul secondo rung in cui viene fatto il confronto tra il contenuto del buffer ($x_1$ nel segmento riportato sotto) e la capacità (10 in questo caso). Solo se $x_1 < 10$, il contenuto del buffer viene incrementato.**




**Il segmento riportato qui sopra viene replicato tre volte nel blocco, una per ogni tipo di parte. Occorre fare attenzione: se si crea il secondo segmento come copia-incolla del precedente, oltre a cambiare il nome delle variabili (cioè per esempio *arrivo2* al posto di *arrivo1* e $x_2$ al posto di $x_1$), è necessario anche modificare la variabile associata al temporizzatore (DB1 in questo caso), che altrimenti sarebbe condivisa dai due temporizzatori, con conseguenze potenzialmente catastrofiche.**

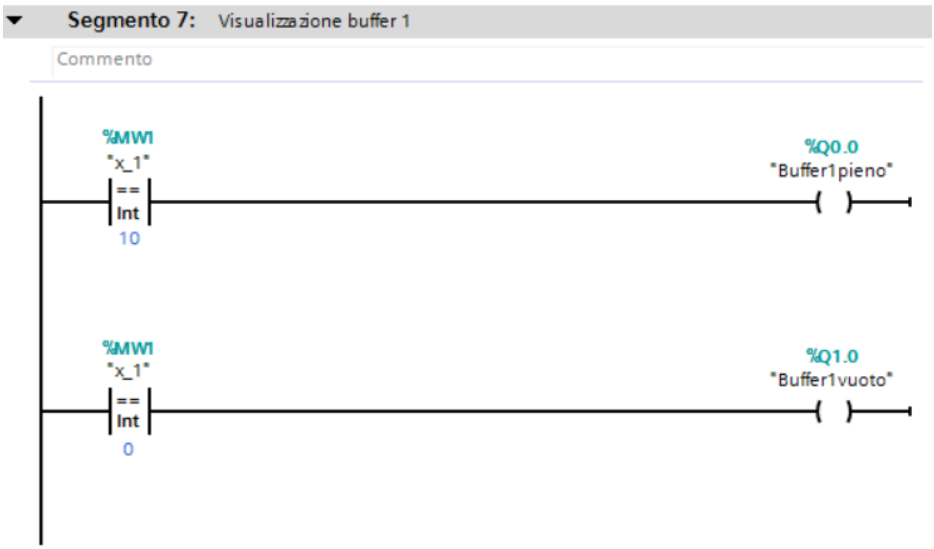
## Esempio 14. Lavorazione (manuale) di buffer con arrivi automatici.

**Questo esempio aggiunge al precedente la possibilità di lavorare (e quindi decrementare) i vari buffer (che sono tre anche qui). Su TIA Portal il progetto in esame si chiama *lavoraManualeArriviAutomatici*. La parte di dinamica degli arrivi è identica a quella del precedente progetto: nel blocco OB1 (che anche qui è l'unico presente) i primi tre segmenti sono la copia del segmento riportato nel progetto precedente. Vi sono poi tre segmenti (il 4, il 5 e il 6, cioè uno per ogni parte) riguardanti le lavorazioni: un fronte di salita sull'ingresso (I0.0 nel caso del primo buffer) decrementa di uno il contenuto del buffer ($x_1$ in questo caso), ma solo se è positivo, facendo un po' quello che si faceva con gli incrementi del progetto *arriviManuale*. I segmenti appena descritti sono del seguente tipo:**




**Il fronte di salita su I0.0 chiude l'interruttore P, la qual cosa, se $x_1$>0, porta a 1 il pin EN del blocco DEC, facendo decrementare la variabile ($x_1$) sul pin IN/OUT. Analogamente viene fatto nei segmenti 5 e 6 del blocco.**

**Infine, i segmenti 7, 8 e 9 sono di visualizzazione: se il buffer 1 è pieno, si accende il led associato alla variabile di uscita Q0.0, se il buffer 1 è vuoto, si accende il led associato a Q1.0.**




**La stessa identica cosa viene effettuata nei segmenti 8 e 9 in relazione agli altri due buffer.**

## 3.1 Uso di un pannello (virtuale) HMI.

Per la visualizzazione di quello che accade nel PLC e per comandarne alcune variabili, si può utilizzare un pannello HMI. Si tratta di inserire nel progetto un dispositivo HMI. A tal fine, andare nella *Vista Portale* e scegliere l'opzione *Aggiungi nuovo dispositivo*. Scegliendo un dispositivo di tipo HMI, comparirà una lista di Display. Scegliere per esempio il *TP1500 Basic*, che compare tra i display da 15''. A questo punto un *Assistente* permetterà di completare la definizione del dispositivo e di inserirlo effettivamente nel progetto. Fare riferimento al Manuale Introduttivo riportato sul sito del corso (il primo dei tre manuali presenti) per effettuare le scelte corrette (quasi tutte possono essere lasciate pari al loro valore di default).

Una volta inserito il pannello, è possibile crearvi oggetti grafici e associarli alle variabili del PLC per ottenere una visualizzazione on-line di quello che accade effettivamente nel PLC e per modificare in real-time il valore delle sue variabili. Sono disponibili moltissimi oggetti grafici nonché diverse possibili animazioni. Per maggiori dettagli, fare riferimento agli esempi riportati in questa sezione nonché al manuale citato pocanzi.

Una volta completata la pagina HMI, è possibile simularla scegliendo il menu (in alto) *Online -> Simulazione -> Avvia* (è anche presente un pulsante scorciatoia che permette l'avvio diretto della simulazione).

**ATTENZIONE**: affinché il pannello HMI veda/influenzi effettivamente le variabili PLC, occorre che *l'interfaccia Siemens PG/PC* presente nel pannello di controllo del PC sia settata su *S7online -> Porta Ethernet in uso sul PC*. Su Windows 8.1 per esempio, per verificare/effettuare questo settaggio, portarsi nel Pannello di Controllo e, cliccando sulla lente accanto a *Impostazioni del PC*, immettere nel campo ricerca il termine *'PG/PC'*. Dovrebbe comparire un'icona che permette di andare al menu di *Impostazione interfaccia PG/PC*. Sulla scheda *Via d'accesso*, scegliere dall'elenco la scheda di rete in uso sul PC (quella collegata al PLC se ce ne dovesse essere più d'una). Se il nome della scheda di rete appare tre volte nell'elenco, scegliere quella che non contiene né la parola *ISO* né la parola *Auto*. A questo punto la simulazione del pannello HMI dovrebbe funzionare correttamente.

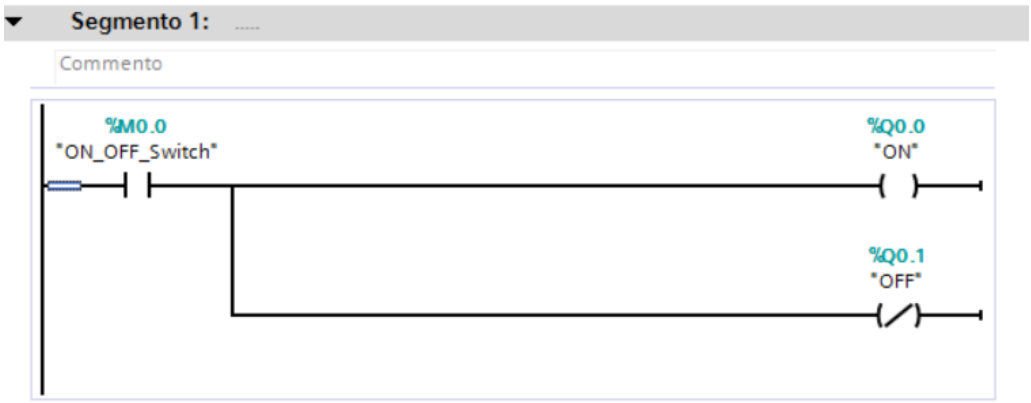
**ATTENZIONE 2**: quando nel progetto sono presenti sia un PLC sia un dispositivo HMI, occorre fare molta attenzione a effettuare le varie azioni sul dispositivo desiderato, pena il blocco di TIA Portal (blocco che potrebbe richiedere la chiusura forzata del

TIA Portal e, a volte, anche il riavvio del PC per poter riaprire il progetto su cui si stava lavorando). Per esempio, se si modifica il programma del PLC e poi lo si vuole caricare sul PLC, fare attenzione che, quando si pigia il pulsante *Carica nel dispositivo*, la vista sia settata su PLC e non su HMI.

Si riportano qui di seguito un paio di esempi di progetto in cui si è utilizzato un pannello HMI.

## Esempio 15. Interruttore con pannello HMI.

**Questo semplice esempio permette di comandare una variabile interna del PLC cliccando col mouse su un interruttore presente nel pannello HMI e di visualizzare lo stato di due variabili (di uscita in questo caso) del PLC mediante due led presenti nel pannello HMI. Il progetto si chiama *usoHMI*. Il codice a contatti è semplicemente il seguente:**

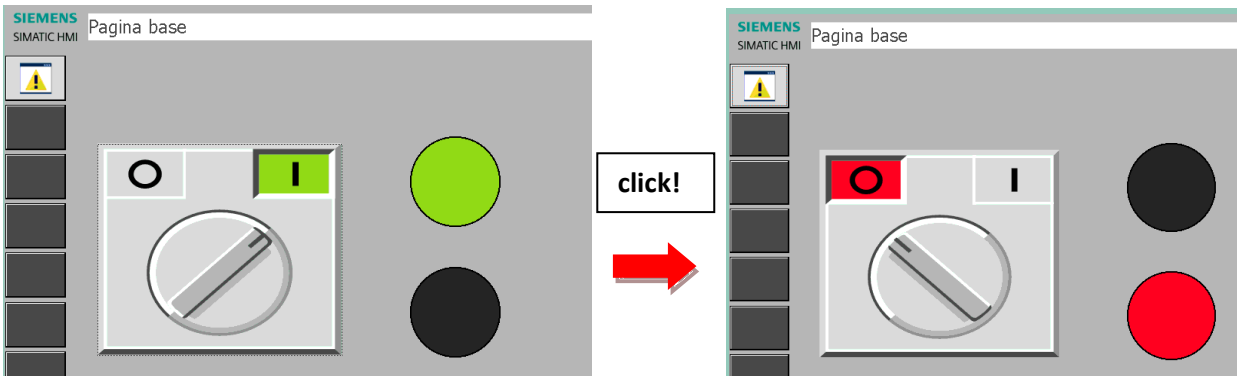



**Nel pannello HMI la variabile ON_OFF_Switch, che è di tipo M (cioè interno), viene associata a un interruttore. Le variabili ON e OFF, di tipo Q (cioè di uscita), vengono associate a due led. Per realizzare il pannello HMI, dopo aver aggiunto il dispositivo HMI, andare nel menu a sinistra dentro HMI e scegliere *Pagine -> Pagina base*. Per inserire l'interruttore, cercare sul menu a destra *Biblioteche globali* e quindi scegliere *Buttons-and-Switches -> Copie master -> RotarySwitches -> Rotary_RG*. Trascinare sulla pagina l'interruttore e dargli la posizione e la dimensione desiderate. Per associarlo alla variabile ON_OFF_Switch, cliccarci sopra col tasto destro e scegliere *Proprietà*. Il collegamento con le variabili PLC dovrebbe far apparire un collegamento tra i due dispositivi (PLC e HMI) nella vista dispositivi (per vedere tale vista, andare sulla Vista Portale e scegliere *Dispositivi & Reti -> Configura reti*).**

**I due led sono invece stati inseriti scegliendo un cerchio dal menu a destra *Oggetti semplici* e trascinandolo nella pagina HMI. Cliccando col tasto destro sul cerchio e scegliendo *Proprietà*, è possibile definirne le modalità di visualizzazione e animazione, anche qui in collegamento con le variabili del PLC (in questo caso ON e OFF).**

**Le seguenti due figure mostrano due schermate del pannello HMI creato in questo esempio. Cliccando sull'interruttore si modifica il valore della variabile ad esso associata (ON_OFF_Switch)**

**e si cambia quindi lo stato dei due led collegati alle variabili ON e OFF. Attenzione: se la variabile associata all'interruttore non fosse stata di tipo M (interna) ma di tipo I (ingresso), l'interruttore non avrebbe avuto effetto su di essa. Ciò è corretto perché la variabile di ingresso del PLC dipende dall'ingresso effettivo presente sul PLC.**




## Esempio 16. Arrivi automatici con lavorazione manuale e pannello HMI.

**Questo esempio riprende l'Esempio 14 descritto qualche pagina fa e consente di visualizzare il contenuto dei tre buffer del sistema produttivo, presenti in tale esempio, sul pannello HMI. Il progetto si chiama *lavoraManualeArriviAutomaticiConHMI*. Il codice è identico a quello di *lavoraManualeArriviAutomatici*, anche se sono stati eliminati i segmenti di segnalazione mediante variabili di uscita degli eventi di buffer vuoto e pieno: dal momento che il contenuto dei tre buffer è visualizzato on-line sul pannello HMI, queste segnalazioni non sono più così necessarie. Dopo aver aggiunto il dispositivo HMI, per realizzare il pannello, andare nel menu a sinistra dentro HMI e scegliere *Pagine -> Pagina base*. La macchina è stata realizzata scegliendo un semplice rettangolo dal menu a destra *Oggetti semplici*. Trascinando il rettangolo sulla pagina, si disegna effettivamente un rettangolo le cui dimensioni possono essere modificate a piacimento così come altre sue proprietà (pigiare a tal fine sul rettangolo col tasto destro del mouse). I buffer sono invece stati realizzati scegliendo e trascinando nella pagina HMI l'oggetto *Barra grafica* dal menu a destra *Elementi*. Anche qui col tasto destro del mouse si accede a Proprietà dove è possibile selezionare il range della barra e la variabile (intera in questo caso) che essa visualizza. La figura seguente riporta una schermata del pannello HMI in funzione.**