PLC per Esempi in linguaggio KOP

17/11/2020

Tiziano Salvagno

SOMMARIO

Si impara molto di più dagli esempi che dalle prediche..

per questo ho scritto questa raccolta di esempi di programmazione di PLC, partendo dalle cose elementari fino alle più complesse.

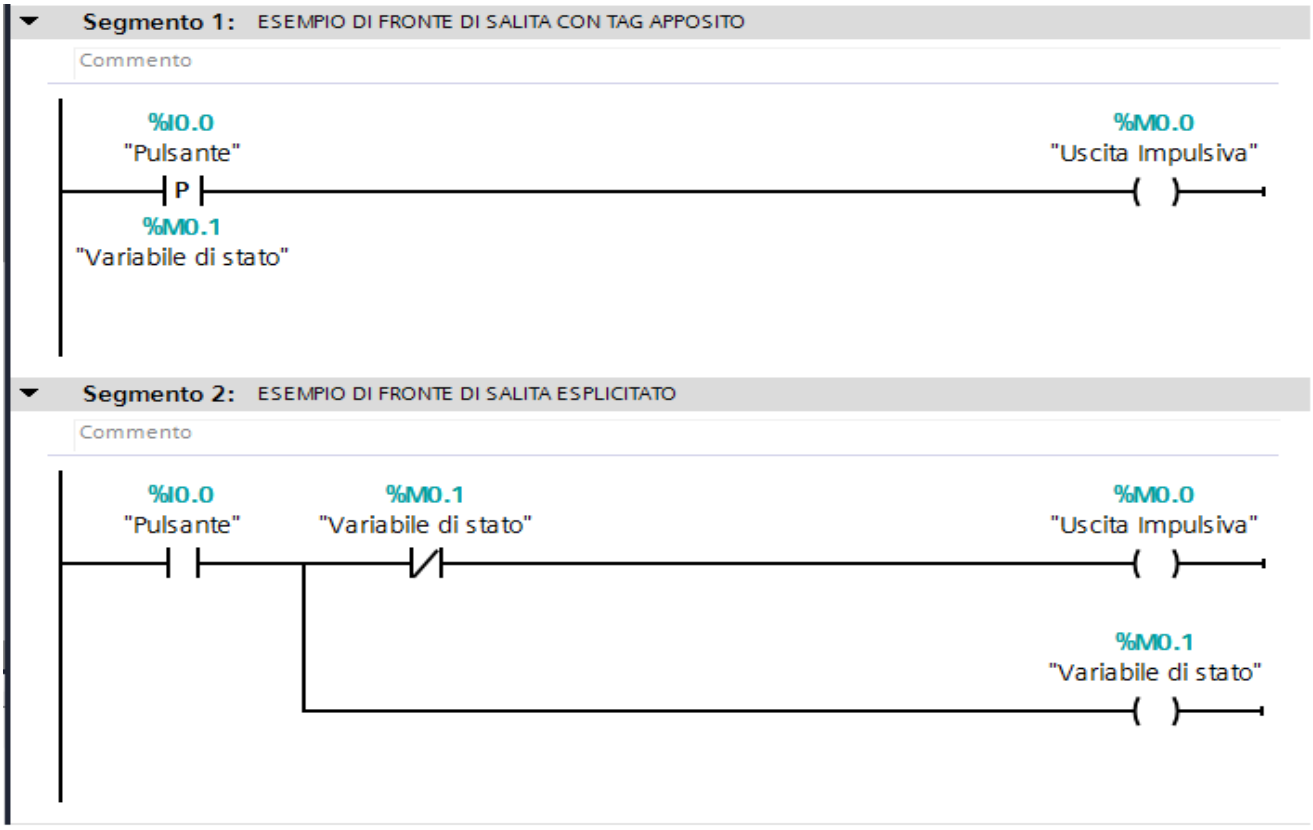
In generale la programmazione è un' unione di blocchi di codice elementari che devono essere semplici e funzionanti. Se i singoli blocchi funzionano perfettamente, anche l'unione di blocchi elementari, dovrà per forza di cose funzionare correttamente.

Per diventare un buon programmatore di PLC bisogna scrivere molti programmi ed avere padronanza del linguaggio di programmazione, che in questo caso è il più comune KOP o linguaggio a contatti. E' necessario, inoltre, conoscere bene il funzionamento del PLC e la strutturazione della memoria di ogni PLC, per evitare, per esempio, l'errore comune di usare i Merker bit di una Word già utilizzata, come ho fatto io alla prima esperienza di programmazione.

In questa guida farò riferimento al PLC Siemens S7 200 con il software microwin 4.0 ed al PLC Siemens S7 1200 con Tia Portal V14.1, che ovviamente sono in mio possesso, ma può valere anche per qualsiasi altro PLC che utilizzi il linguaggio KOP. Per questo i corsi sono anche utili, ma è molto più utile e proficuo avere a disposizione un PLC, il software di programmazione e interagire con esso programmandolo direttamente. Una lingua si impara di più parlandola con un madrelingua, che non studiandola sui libri, anche se entrambe le cose sono necessarie per una buona padronanza della lingua. La stessa cosa vale per qualsiasi linguaggio di programmazione.

Tralascerò l' ABC della programmazione del PLC, che è ampiamente trattata nei libri sui PLC e sui manuali. Questa deve essere una guida per iniziare a scrivere i primi programmi, partendo dai blocchi elementari, come per esempio il blink di un bit con un temporizzatore, il fronte di salita e di discesa, creare un clock con duty cycle 50%, o una sequenza di impulsi positivi o negativi ogni t secondi. Tutti esempi semplici che vengono poi utilizzati combinandoli assieme in blocchi più complessi. Le soluzioni nell'ambito della programmazione non sono uniche, alcune sono eleganti altre più arzigogolate. Con l'esperienza si riuscirà a trovare soluzioni ed algoritmi più efficaci nella soluzione ai problemi di programmazione.

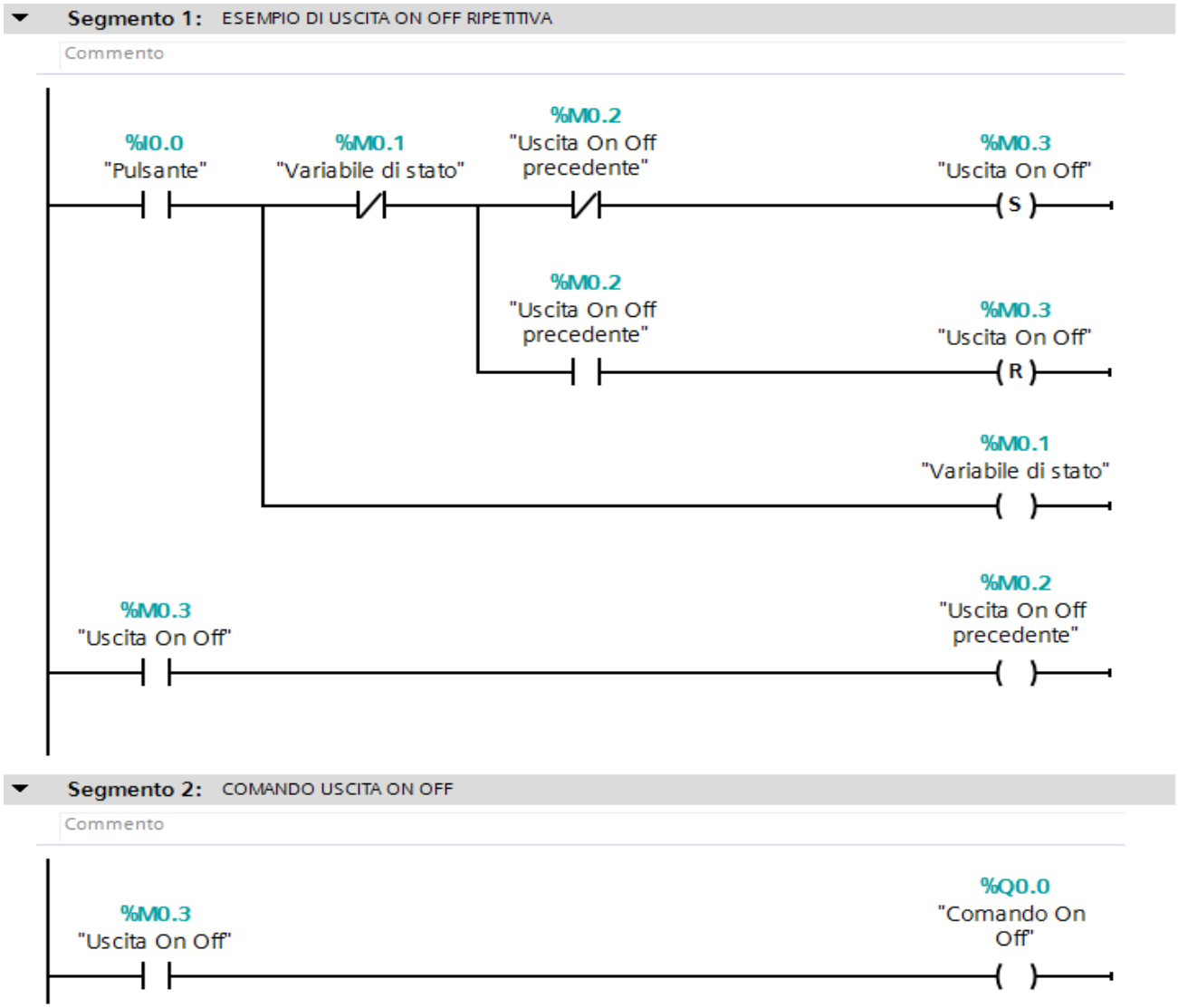
ESEMPIO 1: IMPULSO SUL **FRONTE DI SALITA**.



Una delle necessità basilari nella programmazione è creare da un ingresso un impulso unico. Necessario per far in modo che venga eseguito questo comando una sola volta, per esempio un incremento di una variabile. Il PLC ha infatti una ripetizione ciclica del programma e sono quindi da evitare i loop apri chiudi ad ogni ciclo e sia le continue ripetizioni dei comandi. Non è infatti corretto applicare un Pulsante direttamente all'incremento di una variabile, perché il conteggio sarà incrementato ad ogni ciclo del PLC.

In questo caso si possono usare sia il tag predisposto del fronte di salita o di discesa, che ha bisogno comunque di una variabile di appoggio (Esempio Segmento 1), oppure crearlo esplicitamente usando la variabile %M0.1 di appoggio. Premendo il pulsante con la variabile di stato %M0.1 bassa, la variabile %M0.0 andrà alta, e la variabile di stato %M0.1 andrà alta. Al ciclo successivo, il pulsante sarà sempre alto, mentre la variabile di stato divenuta alta porterà l'uscita %M0.0 bassa, e ci rimarrà nei cicli successivi perché la variabile di stato continua ad essere alta finché il Pulsante rimarrà alto. Quando il pulsante torna basso si ritorna alla condizione iniziale, cioè variabile di stato %M0.1 bassa, ed uscita impulsiva bassa.

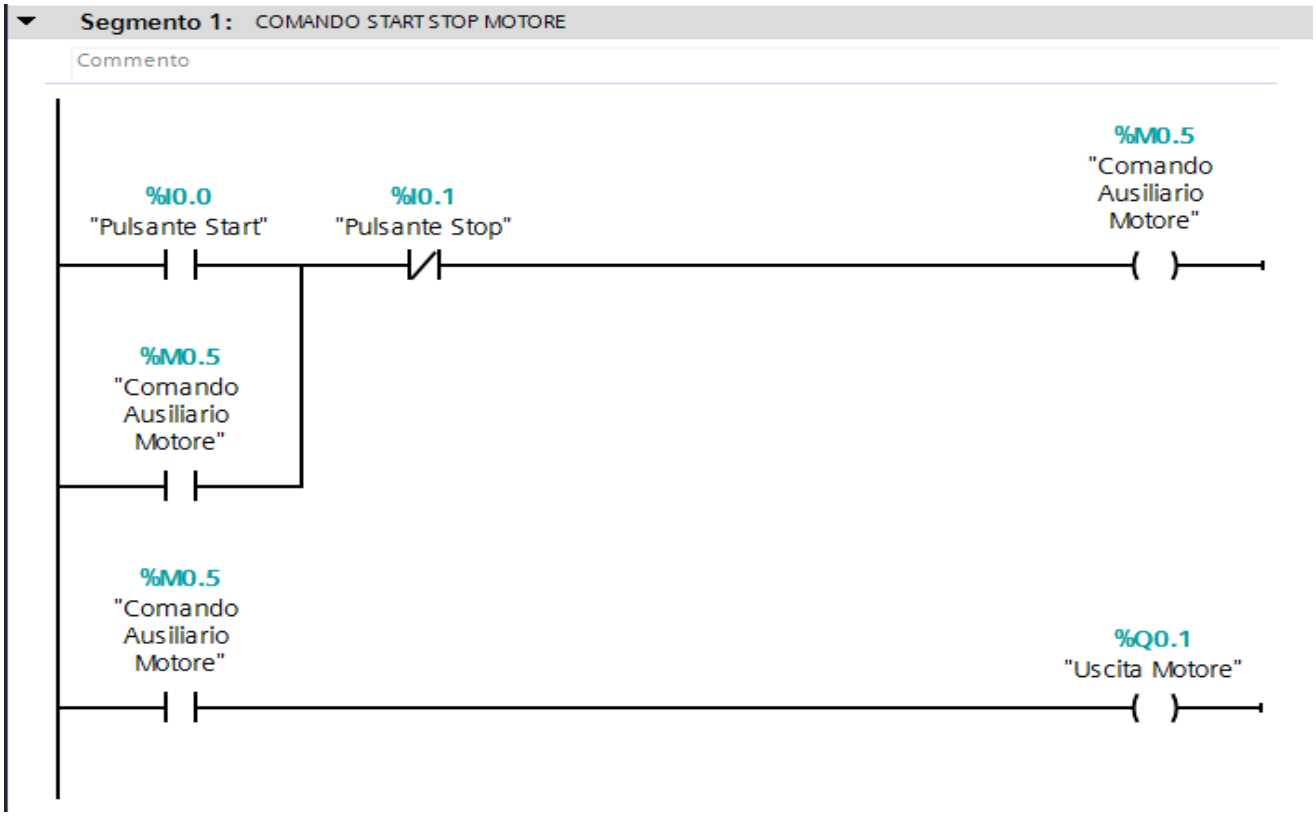
ESEMPIO 2: USCITA ON OFF RIPETITIVA.



In questo esempio generiamo l'uscita ON OFF premendo il pulsante una volta per ON e una volta per OFF. Esistono diversi modi di farlo. In questo caso generiamo un fronte di salita sul Pulsante ed usiamo la bobina set e reset. Se sul fronte di salita lo stato precedente dell'uscita on off era basso, si effettuerà un set, viceversa se era alto si effettuerà un reset. L'istruzione successiva sarà quella di aggiornare lo stato precedente sulla base dell'uscita ON OFF.

Le bobine Set Reset sono un po' pericolose da usare perché può capitare, se non gestite tutte le combinazioni, che rimangano nello stato sempre alto o sempre basso. Per questo motivo occorre essere sicuri di gestire tutte le combinazioni oppure utilizzare le bobine normali. In questo caso è sicuro che tutte le combinazioni sono gestite.

ESEMPIO 3: COMANDO START STOP MOTORE.

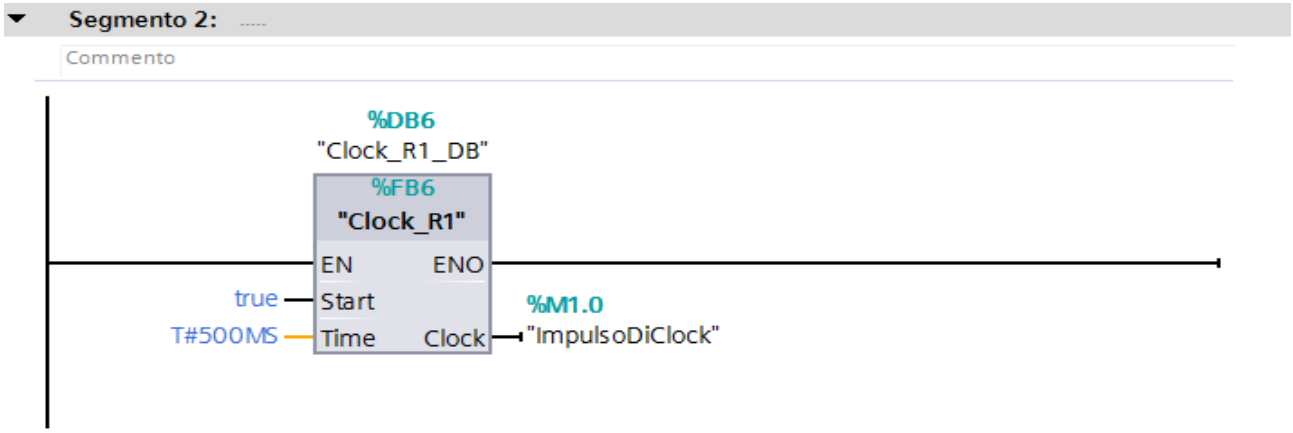


Questo esempio è il classico comando di un motore con i pulsanti di START e di STOP. In questo caso i due pulsanti sono entrambi normalmente aperti NO. Ciò significa che premendo il pulsante l'ingresso del PLC riceverà una tensione di 24 VDC e l'ingresso sarà quindi alto.

Premendo solo il pulsate di Start (ALTO) e con il pulsante Stop rilasciato (BASSO) si attiverà la bobina del Comando ausiliario Motore %M0.5, al primo ciclo del PLC. Al successivo ciclo il Comando ausiliario Motore bypasserà il pulsante di Start anche quando questo sarà rilasciato, e manterrà sempre alto il Comando ausiliario Motore %M0.5. Premendo invece il pulsante di Stop, essendo il contatto invertente porterà basso il Comando ausiliario Motore, e rimarrà basso anche dopo aver rilasciato il Pulsante di Stop, fino a quando si premerà di nuovo il pulsante di Start.

Normalmente il pulsante di Stop per motivi di sicurezza è normalmente chiuso NC, in questo caso sarà necessario usare un contatto non invertente come pulsante di Stop, in quanto l'ingresso del PLC è, a pulsante rilasciato, sempre alto a 24 VDC e quindi deve sempre essere alto per consentire al Comando ausiliario Motore di essere alto premendo il Pulsante di Start. Quando il pulsante NC verrà premuto si interromperà in circuito ed in Comando ausiliario Motore andrà basso.

ESEMPIO 4: CLOCK CON DUTY CYCLE 50%.



In questo semplice esempio generiamo un clock con duty-cycle del 50%, cioè la durata della fase ON è uguale alla durata della fase OFF. Può essere utilizzato un Function Block FB per poterlo utilizzare per clock di diversa durata.

Come si può notare si possono definire le variabili di ingresso Start e Time, la variabile di uscita Clock e alcune variabili temporanee utilizzate nel FB.

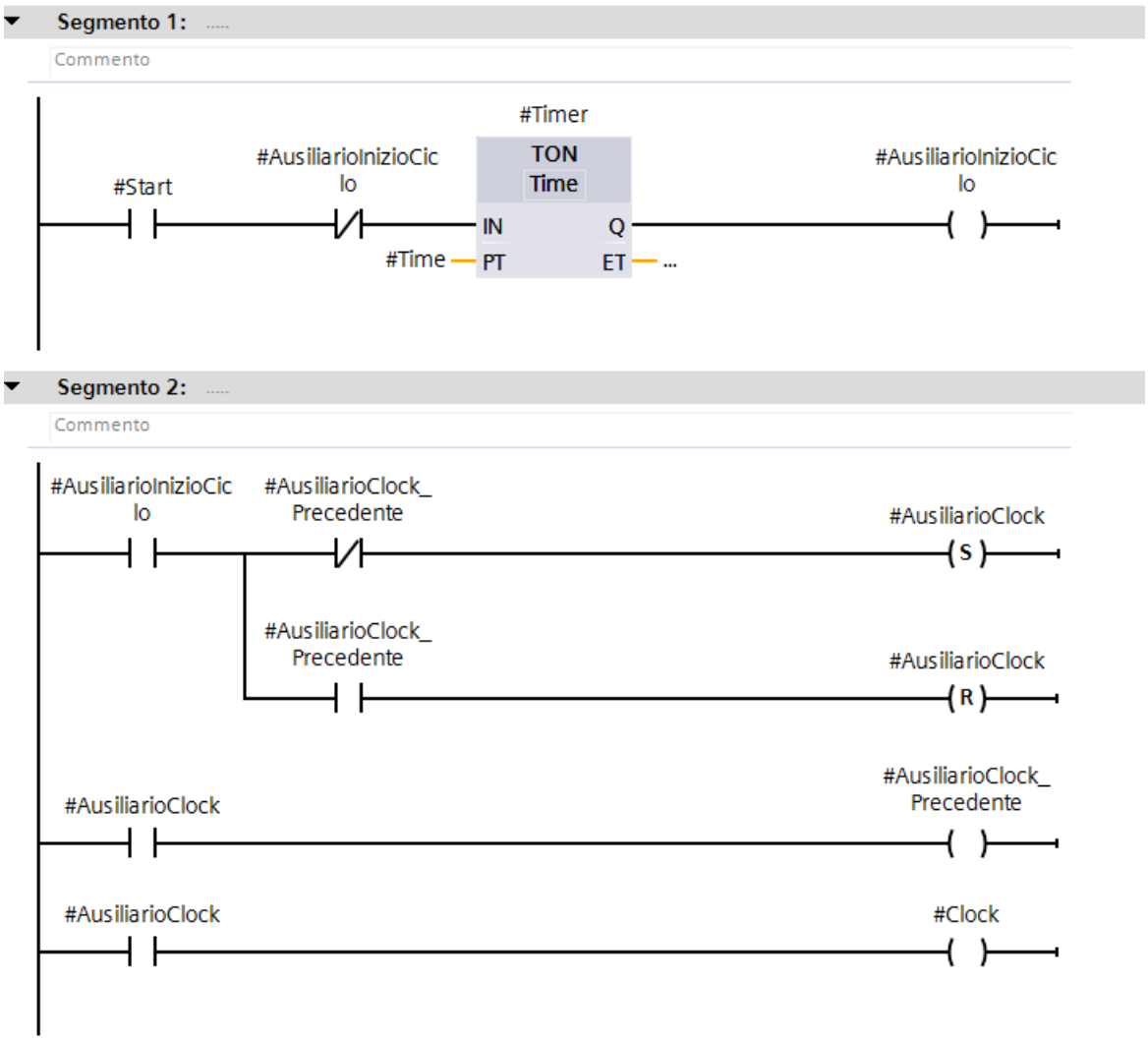
	Nome	Tipo di dati	Valore di default	Ritenzione	Accessibile ...	Scrivi...	Visibile in ..	Valore di i..	Commento
1	Input				☐	☐	☐	☐	
2	Start	Bool	false	A ritenzione	☑	☑	☑	☐	
3	Time	Time	T#0ms	A ritenzione	☑	☑	☑	☐	
4	Output				☐	☐	☐	☐	
5	Clock	Bool	false	A ritenzione	☑	☑	☑	☐	
6	InOut				☐	☐	☐	☐	
7	<Inserisci>				☐	☐	☐	☐	
8	Static				☐	☐	☐	☐	
9	AusiliarioInizioCiclo	Bool	false	A ritenzione	☑	☑	☑	☐	
10	AusiliarioClock	Bool	false	A ritenzione	☑	☑	☑	☐	
11	AusiliarioClock_Prece...	Bool	false	A ritenzione	☑	☑	☑	☐	
12	Timer	IEC_TIMER		A ritenzione	☑	☑	☑	☐	
13	Temp				☐	☐	☐	☐	
14	<Inserisci>				☐	☐	☐	☐	
15	Constant				☐	☐	☐	☐	
16	<Inserisci>				☐	☐	☐	☐	

Come si può notare nel segmento 1 si generano impulsi della durata di Time millisecondi con la variabile AusiliarioInizioCiclo. Infatti se Start è TRUE inizia il conteggio del temporizzatore fino a Time ms. Una volta raggiunto la fine del tempo Time la variabile AusiliarioInizioCiclo diventa TRUE. Al ciclo successivo del PLC viene azzerato l'ingresso del temporizzatore TON, in quanto AusiliarioInizioCiclo è TRUE e quindi il suo contatto negato è FALSE , e quindi anche la sua uscita sarà FALSE. Al ciclo successivo riprende il conteggio del temporizzatore tornando a TRUE il suo ingresso.

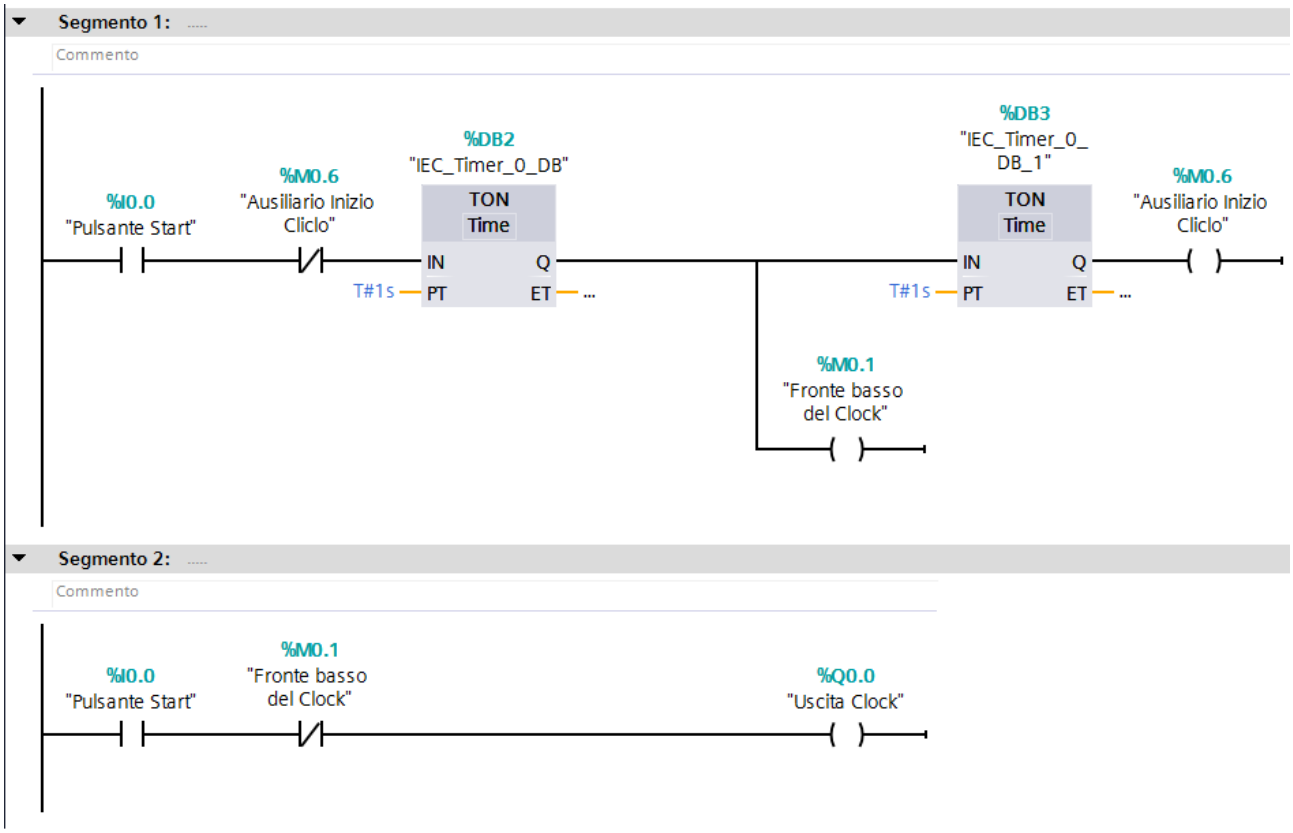
Il segmento 2 invece realizza un cambio di stato della variabile AusiliarioClock, usando lo stato precedente di questa variabile, come variabile di appoggio. Se lo stato precedente era FALSE la variabile Clock passerà a TRUE, viceversa se era TRUE passerà a FALSE. Sarà poi necessario

dopo aver impostato la variabile di clock, aggiornare la variabile AusiliarioClock_Precedente con il valore della variabile AusiliarioClock.

Infine si imposta la variabile di uscita del Clock con il valore della variabile AusiliarioClock.



ESEMPIO 5: CLOCK CON DUTY CYCLE VARIABILE.

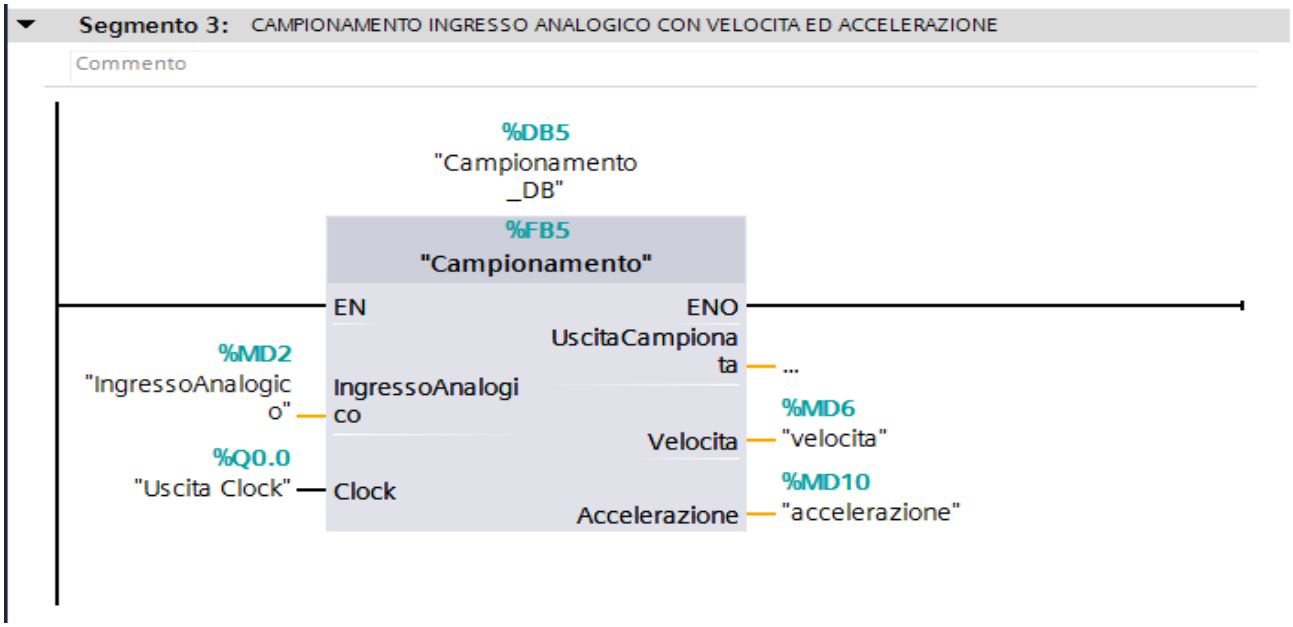


Questo è un esempio semplice di clock con due temporizzatori per gestire il duty cycle del clock. In questo caso al premere del pulsante di Start inizia il conteggio del temporizzatore per il fronte alto del clock. Al termine del conteggio inizia il conteggio del secondo temporizzatore per il fronte basso del clock. Al termine del conteggio del secondo temporizzatore andrà alto il Me4rker %M0.6 ausiliario inizio ciclo, che al successivo ciclo del PLC porterà basso l'ingresso del primo temporizzatore iniziando nuovamente il ciclo di clock. Il clock vero e proprio cioè l'uscita %Q0.0 sarà formato dalla combinazione in AND del pulsante di Start di inizio ciclo che porta alto il fronte del clock, finché il fronte basso del clock rimane basso. All'inizio del fronte basso del clock l'uscita sarà bassa quando inizia a contare in secondo temporizzatore.

Da notare che il Merker di inizio ciclo è necessario in quanto se usassimo l'uscita del temporizzatore .Q per azzerare il ciclo potrebbe capitare di perdere l'impulso di fine ciclo di clock e non resettare il ciclo. Con la presenza del Merker è invece sicuro che in un ciclo o nei successivi il Merker andrà alto e quindi l'ingresso del temporizzatore dovrà andare basso e resettare il ciclo di clock.

E' importante la questione degli impulsi singoli, che potrebbe capitare di andare persi se sono unici, causando un malfunzionamento o non funzionamento dell'impianto come si vorrebbe. Piuttosto è preferibile aggiungere un secondo temporizzatore per allargare la durata dell'impulso, per essere sicuri che non vada perso.

ESEMPIO 5: CAMPIONAMENTO E CALCOLO VELOCITA' ED ACCELERAZIONE.



In questo esempio realizziamo un campionamento di un segnale analogico sulla base di un segnale di clock. In Tia Portal è possibile creare un blocco Funzionale in modo che può essere utilizzato per diversi segnali analogici senza ripetere il programma, ma semplicemente richiamando il blocco funzionale FB.

Il programma è semplice. Ad ogni impulso di clock calcoliamo la differenza tra il valore campionato nell'istante del fronte di salita del clock ed il valore precedente ed aggiorniamo successivamente il valore del campione precedente. Sappiamo infatti che la velocità è data dal rapporto dello spazio / tempo. Quindi calcoliamo la differenza dell'analogica tra due istanti di campionamento della durata di 1 secondo e troveremo la velocità al secondo, e troveremo un valore proporzionale alla velocità.

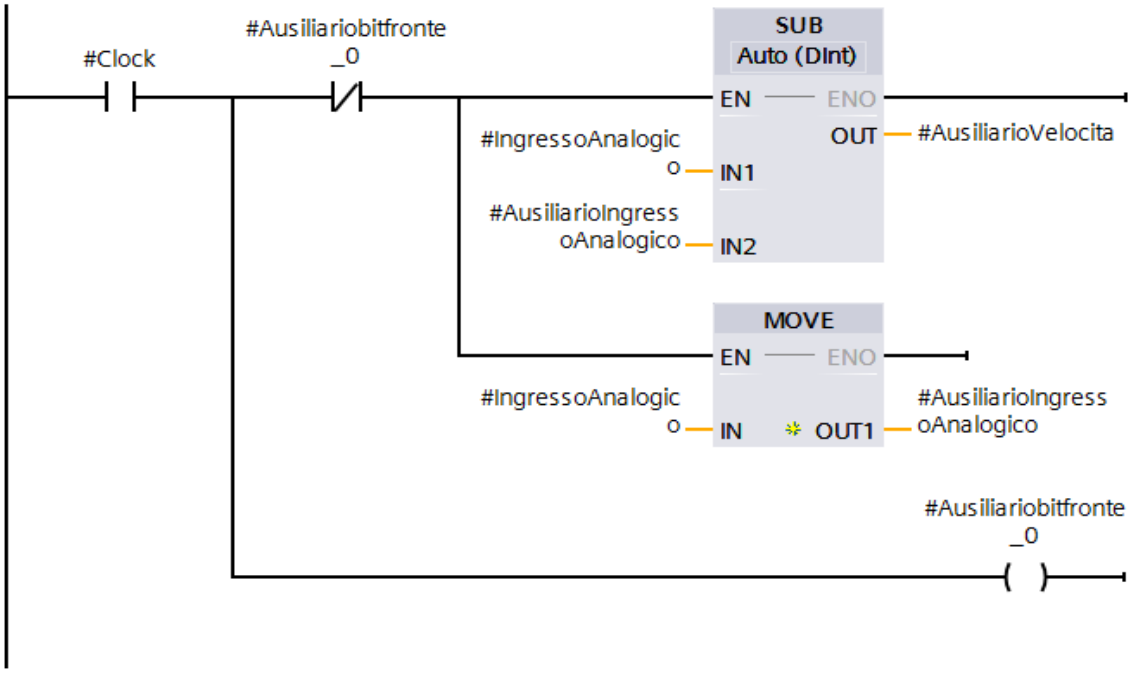
La stessa cosa si applica per l'accelerazione come differenza tra due campioni di velocità in due istanti di campionamento differenti. Troveremo un valore proporzionale all'accelerazione e basterà moltiplicarla per un fattore di scala per ottenere la velocità nelle corrette unità di misura in metri al secondo, oppure gradi centigradi al secondo, e l'accelerazione in metri al secondo quadrato, o gradi centigradi al secondo quadrato, sulla base del valore dell'analogica.

Questo esempio può essere utile per calcolare in caso di un drive la velocità e l'accelerazione di uno spostamento di un nastro trasportatore alimentato da un motore elettrico asincrono gestito tramite un drive (es: G120C della Siemens), oppure nel caso di una temperatura calcolare la velocità di incremento della temperatura nel tempo nel caso ad esempio di un bruciatore con valvola di apertura e chiusura.

Vedremo in seguito come utilizzare il programma per gestire il controllo di posizione o il controllo di temperatura allo stesso modo in cui viene usato il controllo tramite la funzione PID del PLC, che viene molto usata per il controllo di posizione o di velocità di un drive o di temperatura.

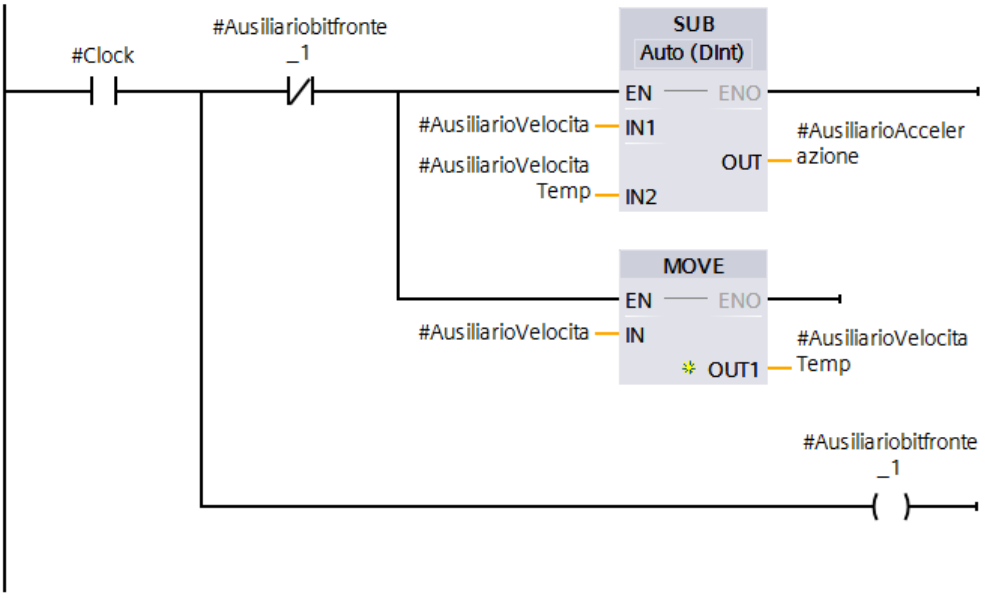
Segmento 1: CAMPIONAMENTO E CALCOLO VELOCITA'

Commento



Segmento 2: CAMPIONAMENTO E CALCOLO ACCELERAZIONE

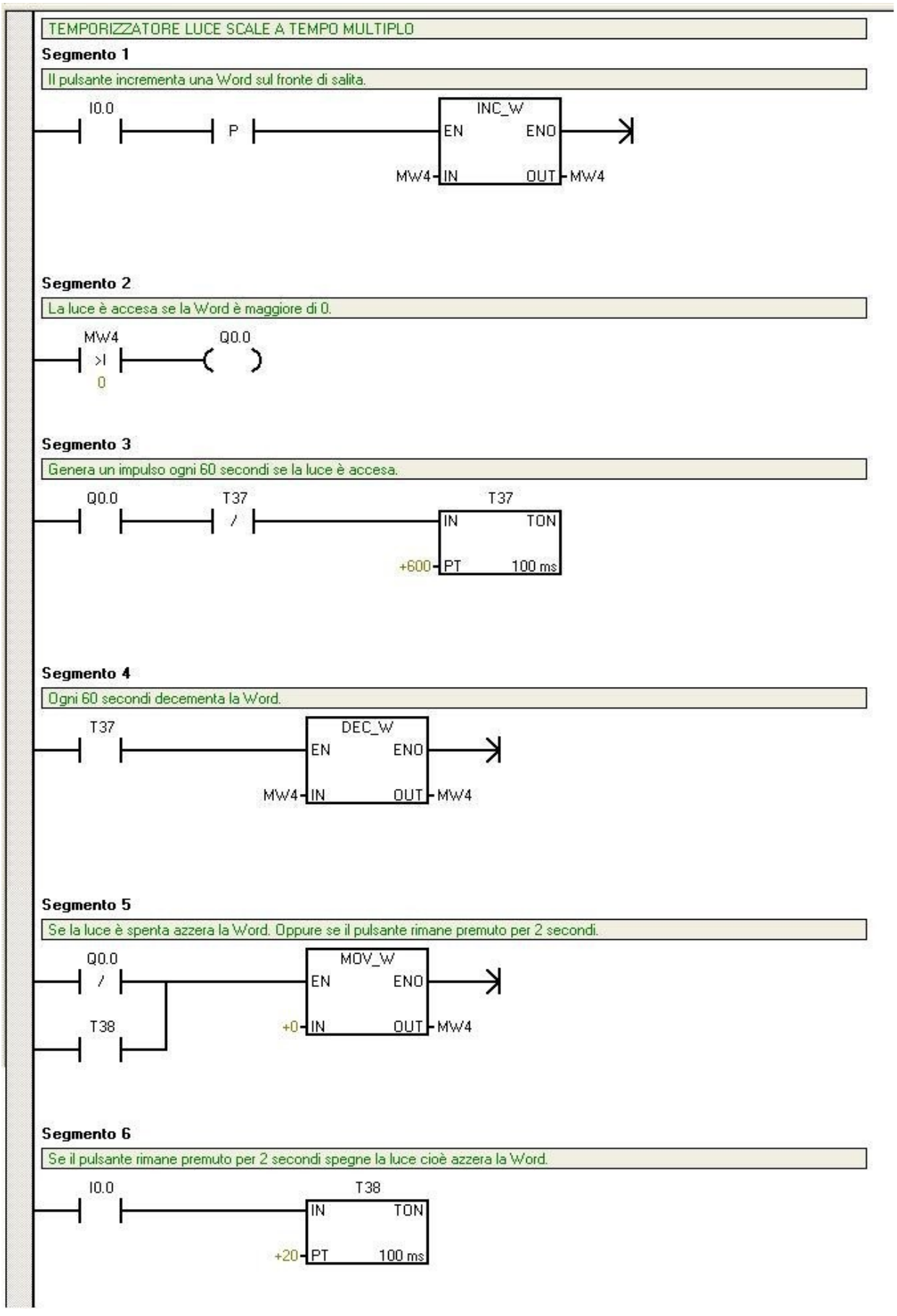
Commento



Segmento 3: .....

Commento

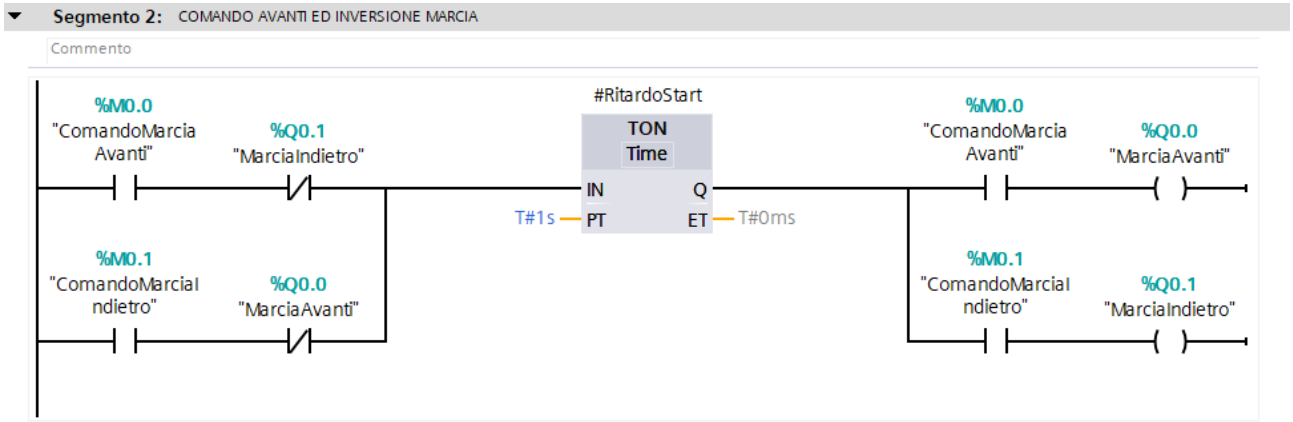
ESEMPIO 1: **TEMPORIZZATORE LUCE SCALE A TEMPO MULTIPLO DI 60 SECONDI.**



In questo esempio di temporizzatore luce scale, è comodo pensare alla rovescia. Anziché contare in avanti e quando raggiunto il conteggio voluto fermarsi, è preferibile impostare un conteggio e decrementarlo fino a raggiungere lo zero. In questo caso la lampada %Q0.0 sarà accesa quando il conteggio è maggiore di zero, e sarà spenta quando raggiunge lo zero per decrementi successivi. Occorre quindi impostare il valore del contatore realizzato tramite una semplice %MW4, incrementandola premendo più volte il pulsante luce scale %I0.0, e gestire il decremento della Word generando un impulso che la decrementa ogni 60 secondi, tramite il temporizzatore T37. Se si vuole spegnere la luce senza attendere il tempo, si può premere il pulsante per 2 secondi, tramite il temporizzatore T38, che azzera la Word e quindi spegne la luce.

Nel PLC S7 200 i temporizzatori partono da T37 quelli che hanno risoluzione minima 100 ms, quindi il conteggio 600 è pari a 600 x 100 ms = 60 secondi.

ESEMPIO 2: **COMANDO AVANTI STOP INDIETRO DI UN MOTORE**.



Questo è un semplice esempio del cuore di un programma per invertire la marcia di un motore. E' necessario infatti nell'inversione fermare il motore ed attendere un tempo di fermata, prima di ripartire in direzione opposta. In questo caso si usano due comandi di marcia avanti e marcia indietro che andranno opportunamente gestiti a monte del programma, e con due uscite, motore avanti e motore indietro, usando un solo temporizzatore.

In questo caso, invertendo la marcia l'ingresso del temporizzatore andrà a zero per un istante azzerando il conteggio, per poi ripartire in direzione opposta dopo il tempo di ritardo. Sarà cura del programma gestire i comandi Marcia Avanti e Marcia Indietro in modo che uno solo sia attivo nello stesso istante con un interblocco.

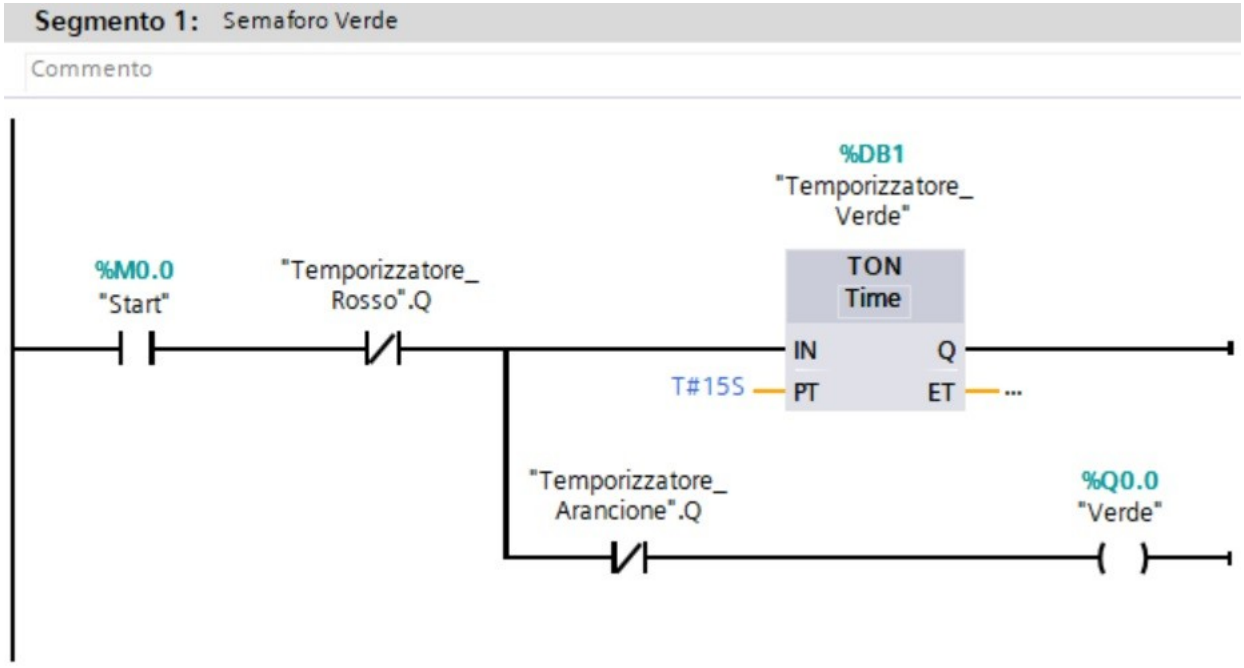
ESEMPIO 7: **SEMAFORO**.

Un altro classico esempio è il semaforo unico.

E' necessario un selettore di Start e di Stop per indicare quando il semaforo è in funzione, e quando invece deve lampeggiare il solo GIALLO.

In sequenza devono accendersi il VERDE, IL VERDE E ARANCIONE, ED IL ROSSO. Quindi ci saranno 3 temporizzatori che conteranno il tempo del VERDE, ARANCIONE e ROSSO.

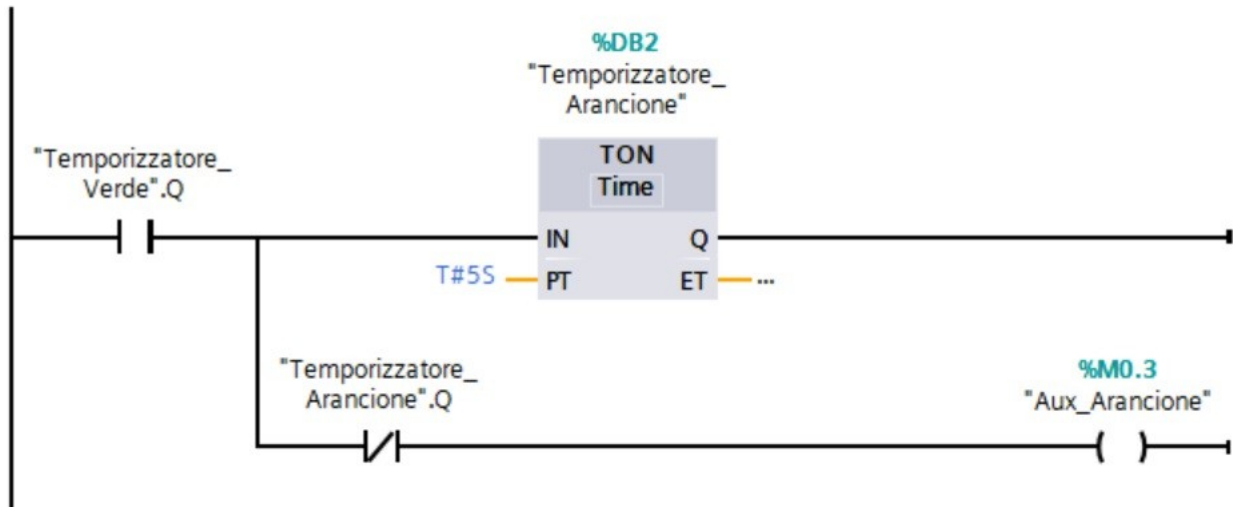
Quindi con lo START inizia il conteggio del VERDE, dopo il verde inizia a contare L'ARANCIONE e poi il ROSSO. La lampada verde si spegnerà quando finisce il tempo dell'ARANCIONE. La lampada ARANCIONE si spegne dopo il tempo di ARANCIONE. La lampada ROSSA si spegne dopo il tempo di ROSSO, cioè quando inizia nuovamente il ciclo con la lampada VERDE accesa. Dopo il conteggio del ROSSO quindi inizierà nuovamente il ciclo azzerando il temporizzatore verde per un istante, per poi riprendere il conteggio dal verde.

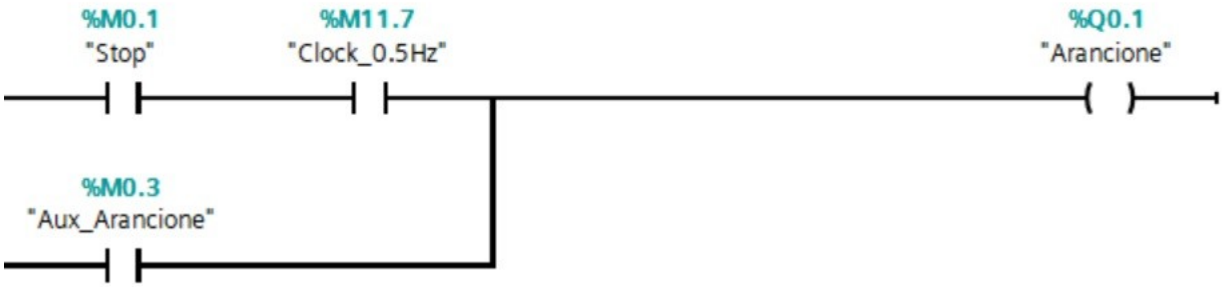


La lampada ARANCIONE avrà un ausiliario per il funzionamento normale ed un lampeggio quando è attivo il selettore di Stop.

Segmento 2: Semaforo Arancione

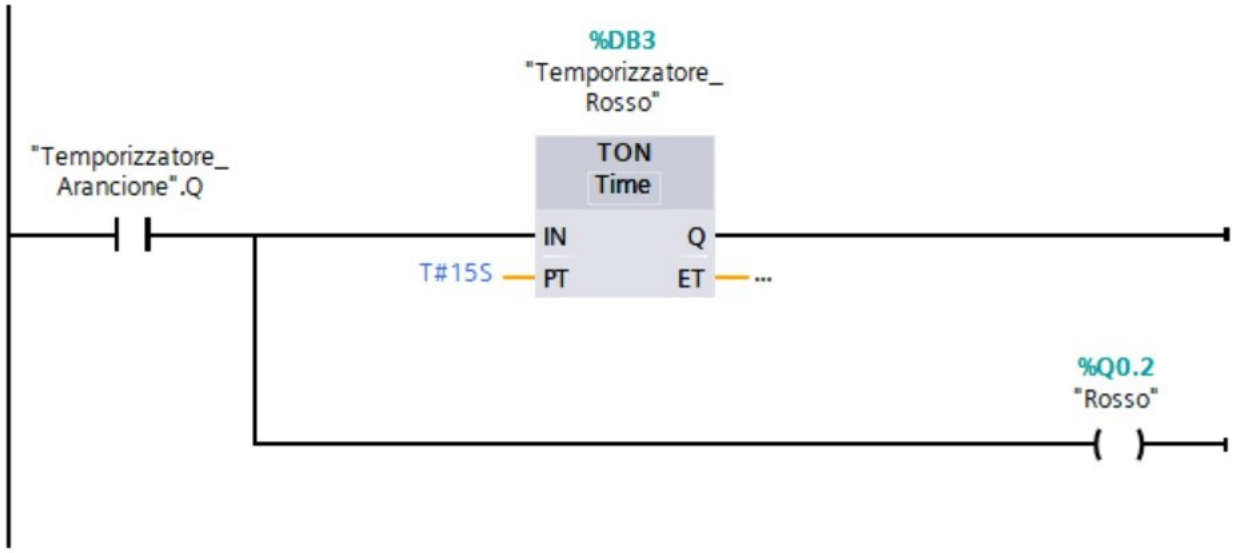
Commento





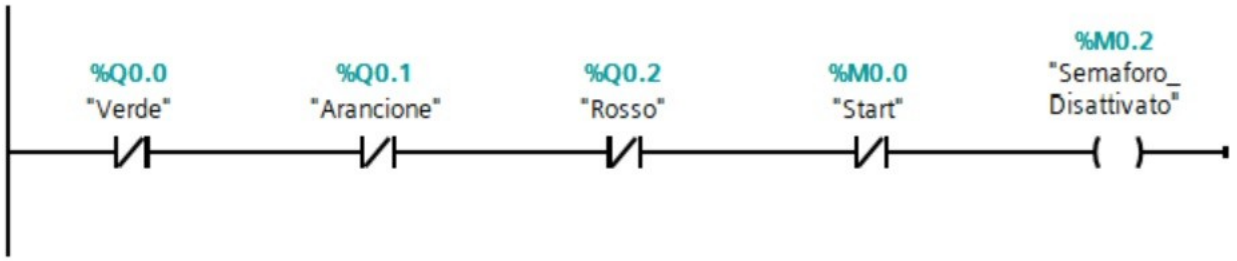
Segmento 3: Semaforo Rosso

Commento



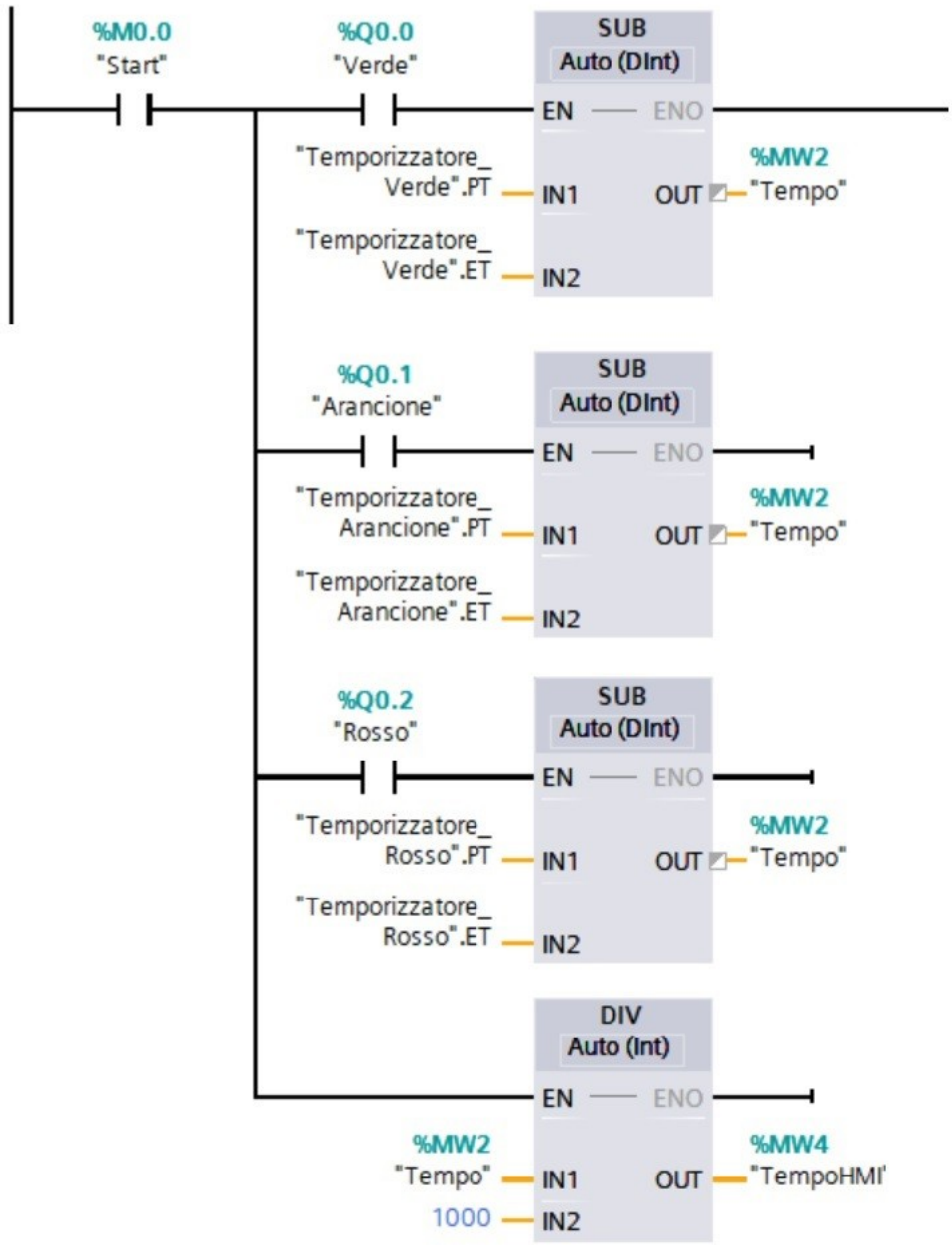
Segmento 4: Semaforo Spento

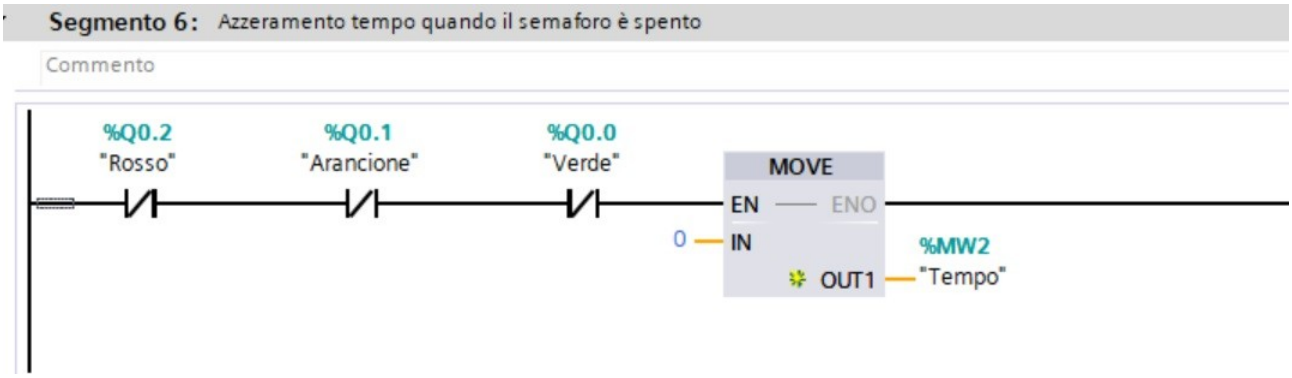
Commento



Segmento 5: temporizzatore secondi rimanenti

Commento





ESEMPIO 8: **PID o ALGORITMO DEL PILOTA?**

Se vogliamo realizzare un termostato che deve regolare la temperatura di un bruciatore ON OFF, abbiamo bisogno di un ingresso sul PLC analogico configurato come misuratore di temperatura tramite la sonda di temperatura PT100 o PT1000, e due uscite digitali che comandano il bruciatore ON ed OFF, che rispettivamente aprono e chiudono la valvola del bruciatore, che aumentano o diminuiscono la potenza del bruciatore.

Un algoritmo del PLC che realizza questa regolazione è il PID. In sostanza il segnale proveniente dalla sonda di temperatura viene confrontato con il SetPoint ed alla differenza dei due (errore) viene applicata una funzione di regolazione con tre termini. P proporzionale, cioè direttamente proporzionale all'errore, un termine I integrativo che è legato all'integrale dell'errore, ed il termine D derivativo legato alla derivata dell'errore.

I termini P I D devono essere calcolati opportunamente per garantire il raggiungimento del Setpoint e il mantenimento secondo una regolazione che eviti una eccessiva sovraelongazione, cioè il superamento del setpoint e successiva alternanza tra valori superiori ed inferiori del Setpoint, prima di raggiungere il valore desiderato di regolazione.

Questo procedimento di regolazione non è sempre semplice, per questo ho ideato un algoritmo diverso di regolazione. Non so se esiste in letteratura un algoritmo di questo tipo, ma si basa semplicemente su quello che fa il nostro cervello quando è alla guida, e può per questo essere chiamato algoritmo del Pilota.

Infatti, quando siamo alla guida e dobbiamo affrontare una curva, il nostro cervello non calcola la funzione PID per decidere quando iniziare a frenare, per non raggiungere la curva troppo velocemente ed uscire di strada (sovraelongazione). Inoltre quando siamo alla guida non guardiamo il cofano dell'auto e nemmeno l'orizzonte, ma guardiamo ad una distanza dall'auto che dipende dalla velocità dell'auto stessa. In questo modo cogliamo eventuali ostacoli ed inizieremo a frenare in tempo.

Il nostro cervello calcola in base alla velocità dell'auto la distanza giusta in cui dirigere gli occhi e vedere quando inizierà la curva. Nel momento in cui, in base allo spazio di frenata siamo troppo veloci e corriamo il rischio di uscire per la tangente della curva, il cervello darà il comando al piede di iniziare a frenare. Questa regolazione tra velocità, distanza della curva e piede del freno permette

di affrontare la curva. Un pilota esperto sarà in grado di affrontare la curva nel migliore dei modi, senza frenare troppo in anticipo oppure troppo in ritardo ed uscire di strada.

Applicando questo algoritmo alla regolazione di temperatura, dovremo calcolare la velocità di salita della temperatura, calcolare dove sarà la temperatura dopo un certo tempo, e poi verificare se stiamo andando troppo veloci e supereremo il Setpoint ed allora dovremo iniziare a frenare la temperatura cioè chiudere la valvola del bruciatore.

Una volta raggiunta la temperatura dovremo passare alla fase di regolazione dove verificheremo in base al valore della temperatura in istanti differenti, se stiamo tendendo a superare o essere inferiori al Setpoint e quindi dovremo comandare con piccole regolazioni la valvola del bruciatore.

In sostanza questo è quello che si dovrebbe fare e che fa il nostro cervello. Inoltre se modifichiamo il Setpoint, il PID impiega un certo tempo per rispondere al cambiamento, a causa del termine Integrativo. Con questo algoritmo invece è immediato stabilire se siamo sopra o sotto il Setpoint ed il comando del Bruciatore sarà pressoché immediato.

L’implementazione è abbastanza semplice. Dovremmo calcolare la velocità e l’accelerazione di temperatura con le funzioni che abbiamo visto in precedenza e confrontare il valore di temperatura previsto dopo n istanti di tempo, con il setpoint e quindi sulla base del valore della differenza comandare il bruciatore su ON oppure OFF.