PLC
Programmazione basata su stati e programmazione basata su segnali

Introduzione

L'elaborato si colloca nel settore della programmazione dei PLC. Vengono brevemente introdotti i vantaggi apportati dall'introduzione dei PLC nell'ambiente industriale e ci si concentra poi sui modi in cui essi possono essere programmati.

Il documento, in particolare, affronta due soli linguaggi di programmazione, tra tutti quelli esistenti. Questi linguaggi sono Ladder Diagram e Sequential Functional Chart e rappresentano:

- il primo, lo stile di programmazione basato sul controllo dei segnali di input e outpt
- il secondo, lo stile di programmazione basato sugli stati

La scelta di tali linguaggi si è basata sulle caratteristiche che essi offrono. In particolare:

- Ladder Diagram è il linguaggio di programmazione per PLC più diffuso ed utilizzato. Ogni produttore di PLC lo supporta, ed esso può essere implementato su una qualunque macchina.
- Sequential Functional Chart è un linguaggio molto intuitivo e di facile utilizzo. Può essere considerato di alto livello poiché è più vicino al modo di pensare di una persona piuttosto che al modo di agire di una macchina.

Ladder Diagram quindi ha una elevatissima portabilità, mentre Sequential Functional Chart gode di una notevole facilità d'uso, entrambe caratteristiche sono molto vantaggiose.

Obiettivi

Nell'elaborato ci si propone di trovare un metodo che permetta di approfittare di entrambi i vantaggi dei due linguaggi di programmazione affrontati. Sarebbe

infatti molto utile avere a disposizione un linguaggio di programmazione in cui sia facile implementare sistemi estremamente complessi ottenendo codice chiaro e sintetico e nello stesso tempo in grado di funzionare su ogni genere di macchina.

La soluzione al problema, affrontata nel documento, è quella di implementare il sistema di controllo di cui si ha bisogno, in Sequential Functional Chart ed utilizzare poi un algoritmo di traduzione che trasformi il codice scritto in codice Ladder Diagram. In questo modo si puòò facilmente implementare il sistema e renderlo, tramite traduzione, portabile e utilizzabile su ogni macchina.

Tale algoritmo di traduzione del codice deve essere preciso e permettere di ottenere codice LD funzionale e concettualmente coerente con il codice SFC di partenza.

Algoritmi

Vengono trattati, all'interno del documento tre diversi algoritmi che permettono di raggiungere gli obiettivi richiesti. Per ciascuno di essi viene proposto il metodo di utilizzo, i rispettivi vantaggi e svantaggi e una serie di esercizi illustrativi.

Il primo algoritmo presentato si basa sull'*algoritmo di evoluzione*. Le sue caratteristiche vengono presentate sommariamente poiché esistono già testi che le trattano in maniera approffondita.

L'elaborato dedica invece molto spazio ai due successivi algoritmi. Entrambi si basano sulla traduzione del codice in se stesso. Partendo cioè da codice SFC, si ottiene codice LD trasformando opportunamente passi e transizione SFC in contatti e bobine LD.

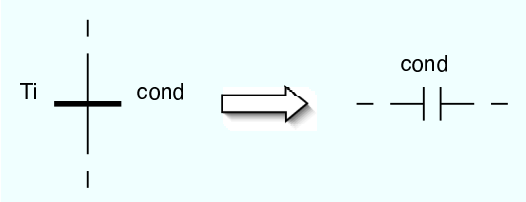

Figura 1: Traduzione di una semplice transizione

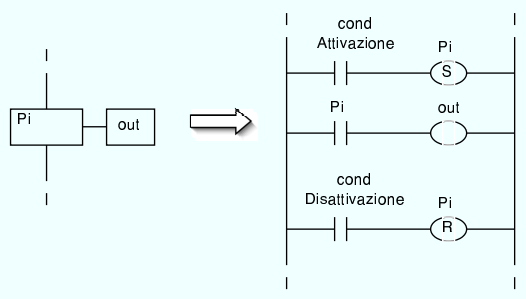

Figura 2: Traduzione di un semplice passo

Conclusioni

Gli algoritmi presentati hanno soddisfatto gli obiettivi che ci si era proposto di raggiungere. Gli esempi affrontati nell'elaborato hanno dimostrato come il loro utilizzo permetta effettivamente di unire la semplicità di programmazione alla portabilità.

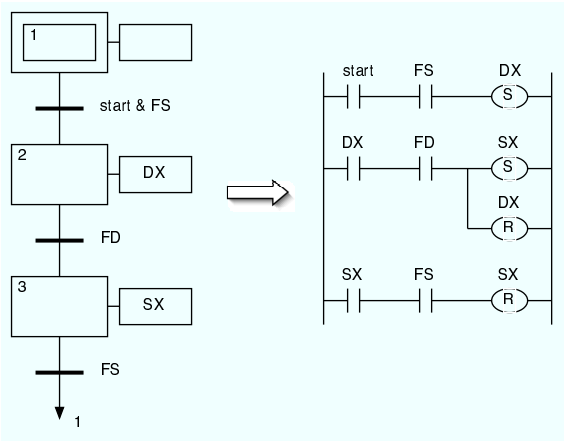

Figura 3: Semplice esempio di traduzione di codice SFC

L'elaborato propone infine un esempio applicativo complesso al quale applica gli algoritmi di traduzione descritti allo scopo di dimostrare che tali algoritmi risultano efficaci anche in sistemi che non siano puramente illustrativi