Principi di base del linguaggio DMstate

DMstate è un linguaggio a stati ovvero, formato da un insieme di blocchi ognuno dei quali svolge una particolare funzione del programma totale.

Ogni stato (quindi ogni blocco) ha una particolarità: le uscite devono rimanere costanti.

Gli stati sono collegati tra loro attraverso dei salti tra un blocco e l'altro che dipendono dai valori degli ingressi o dai valori di alcune variabili.
Per capire meglio questo concetto, facciamo un esempio: se dovessimo creare un programma attraverso il quale dobbiamo accendere una lampadina alla pressione di un pulsante e spegnerla alla pressione di un altro pulsante, possiamo individuare:

2 ingressi (i 2 pulsanti connessi all'ingresso numero 1 e numero 2)

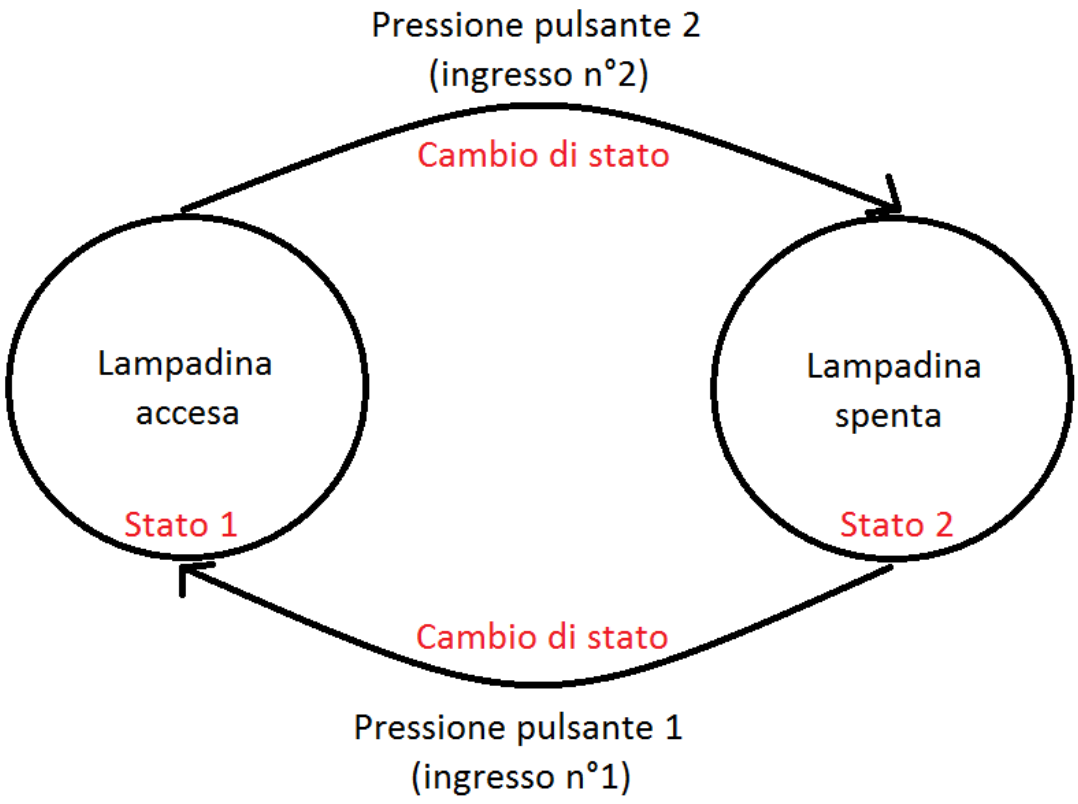
1 uscita (l'uscita numero 1 utilizzata per connetterci la lampadina).

Precedentemente abbiamo detto che uno stato dovrà avere le uscite costanti, quindi in questo esempio dovremo creare 2 stati che corrispondono ai 2 valori della lampadina (che rappresenta l'uscita):

- stato 1: Lampadina accesa
- stato2: Lampadina spenta

Il legame tra i due stati viene fatto attraverso il valore degli ingressi: se il pulsante 1 è premuto salto nello stato 1 (Lampadina accesa) mentre se il pulsante 2 è premuto salto nello stato 2 (Lampadina spenta).

Riportiamo di seguito uno schema che riassume questo concetto:



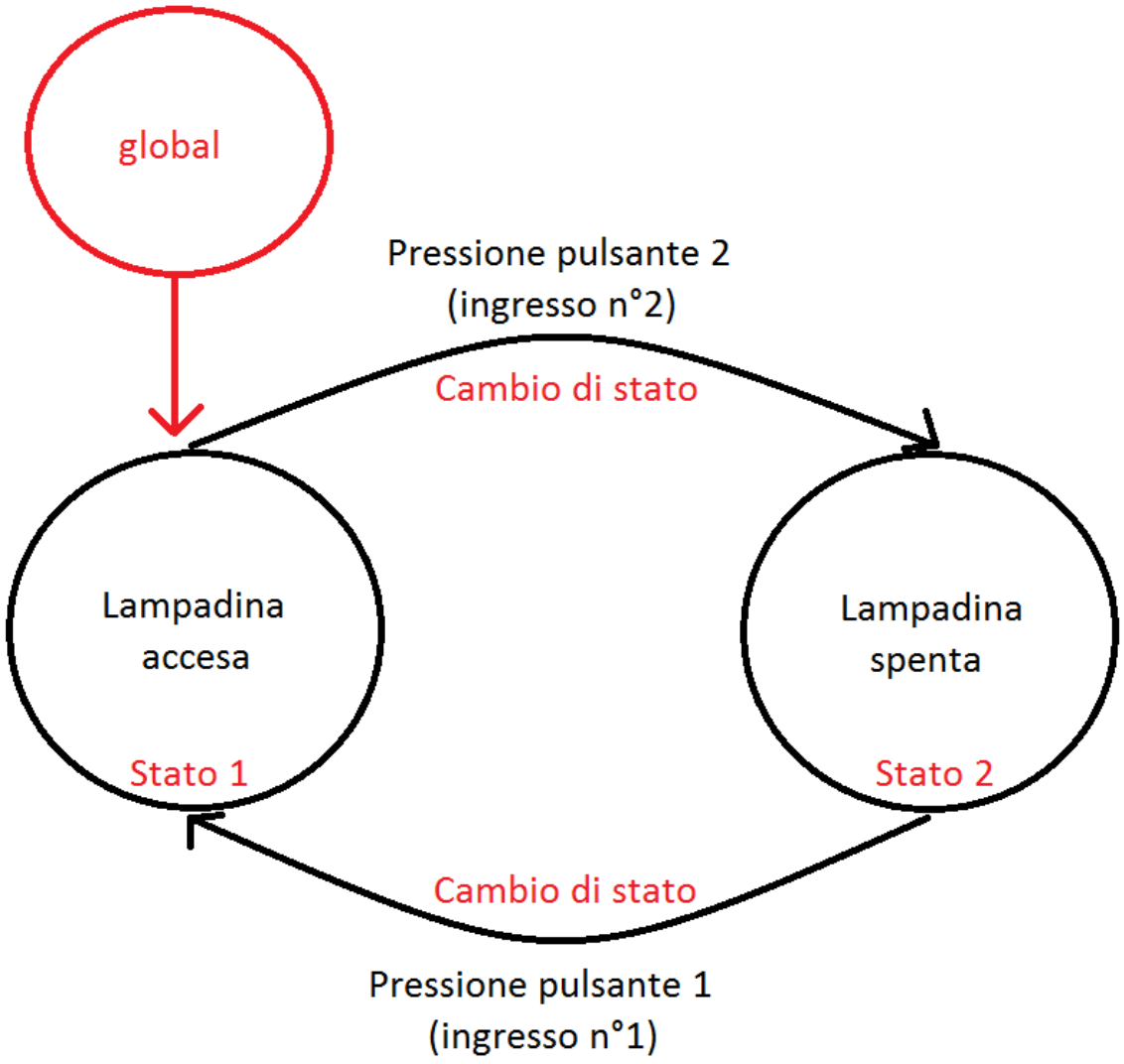
All'interno di ogni stato possono essere definite delle variabili numeriche (chiamate registri) ovvero un valore numerico identificato da un nome che può variare all'interno dello stato. Queste variabili definite all'interno dello stato, valgono solo per quello stato. Ad esempio, tornando al caso precedente, se avessimo voluto definire una variabile chiamata "ValoreLampadina" all'interno dello stato "Lampadina accesa" e avessimo assegnato a tale variabile il valore 1, una volta usciti dallo stato "Lampadina accesa", la variabile sarebbe stata eliminata e quindi nello stato "Lampadina spenta" non troveremmo la variabile "ValoreLampadina".

Per poter definire una variabile che venga vista all'interno di tutti gli stati del programma, dovremo servirci di uno stato particolare chiamato global che viene creato di default su tutti i programmi, e definire all'interno di questo stato la variabile che ci interessa.

Questo particolare stato chiamato global ha anche altre funzioni che elenchiamo di seguito:

1. definire le variabili che dovranno essere condivise (e quindi viste) in tutti gli stati
2. inizializzare la scheda ovvero definire come dovranno essere le uscite.
3. definire quale sia lo stato principale.

Quindi lo schema visto in precedenza diventa:



Oltre alle variabili numeriche, esistono anche le variabili alfanumeriche.

Mentre le variabili numeriche, a seconda di dove vengono definite, assumono un valore locale o globale, le variabili alfanumeriche possono essere definite solo nello stato global e quindi sono globali.

Tra le variabili numeriche e alfanumeriche, esiste anche un'altra distinzione; le prime, una volta definite vengono salvate in RAM (una memoria volatile che se viene tolta alimentazione alla scheda perde il suo valore), mentre le seconde (quelle alfanumeriche) vengono salvate direttamente in FLASH (una memoria non volatile che se viene a mancare l'alimentazione mantiene il valore contenuto al suo interno).

Organizzazione degli stati

Ogni stato è diviso in 3 blocchi:

- **Variables**: in questo blocco vengono definite le variabili numeriche (o numeriche e alfanumeriche per lo stato global)
- **Outputs**: in questo blocco viene definito il funzionamento dello stato assegnando i valori delle uscite o delle variabili, inserendo eventuali ritardi, inviando messaggi, inviando chiamate, ecc.
- **Jumps**: questo blocco collega tra di loro gli stati attraverso il valore degli ingressi o delle variabili definite precedentemente