Implementarea unei parcari cu cod de acces folosind TwidoSoft

Gostinar Nicolae – Grupa 332 AC

Petricu Ramona – Grupa 333 AA

Somoiag Mihaela – Grupa 333 AA

Chirita Alexandru – Grupa 334 AA

Introducere

Subiectul lucrarii a fost ales pentru a demonstra usurinta implementarii unei aplicatii practice folosind mediul de programare TwidoSoft si un automat programabil TWDLCAA24DRF.

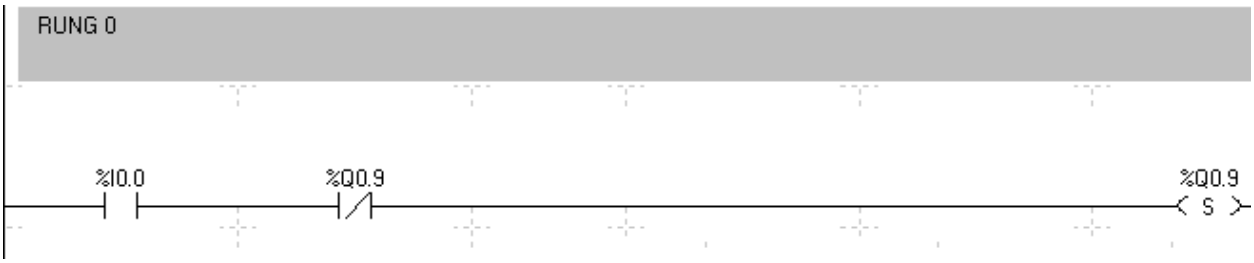
Modul in care programul merge, prezentat pe scurt:

- Se citeste un cod de catre cel care gestioneaza parcarea (in cazul de fata codul este 0-2-1, dar el poate fi modificat);
- In unele din cazurile in care codul 0-2-1 a fost scris gresit, automatul revine la starea initiala (celelalte cazuri erau mai dificile si trebuiau tratate special);
- Dupa ce codul a fost corect, parcarea este rezervata pentru o perioada definita dinainte (in cazul nostru 10 secunde, dar poate fi modificat);
- Sunt gestionate 5 parcari. Parcarile se rezerva in ordine, de la 1 la 5;
- Daca numarul de parcari este maxim, nu se mai intampla nimic (desi se poate scrie parola, dar se rezerva niciun loc).

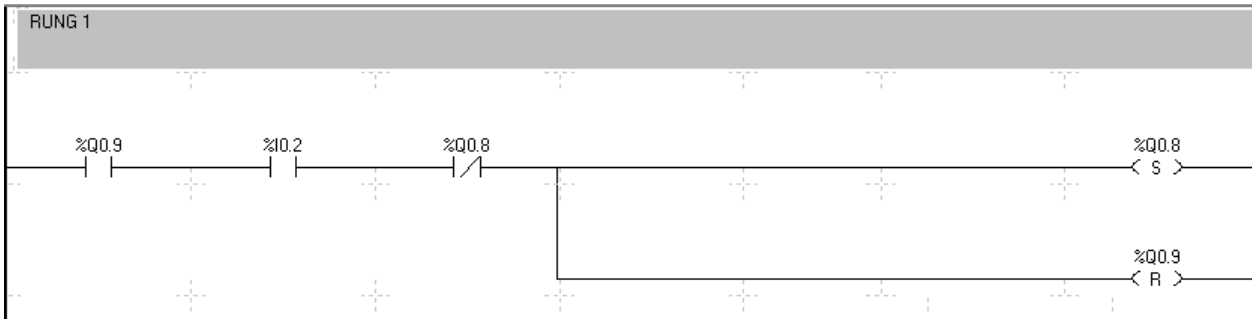
Explicare program

1. Introducerea codului de acces 0-2-1:

In rungul zero este verificat codul de acces 0 (primul buton de pe automat, de la adresa de intrare %I0.0). Daca este apasat codul de acces 0 si starea 9 este inactiva (cu adresa %Q0.9), atunci se activeaza starea 9 (respectiv ultimul led de pe automat).

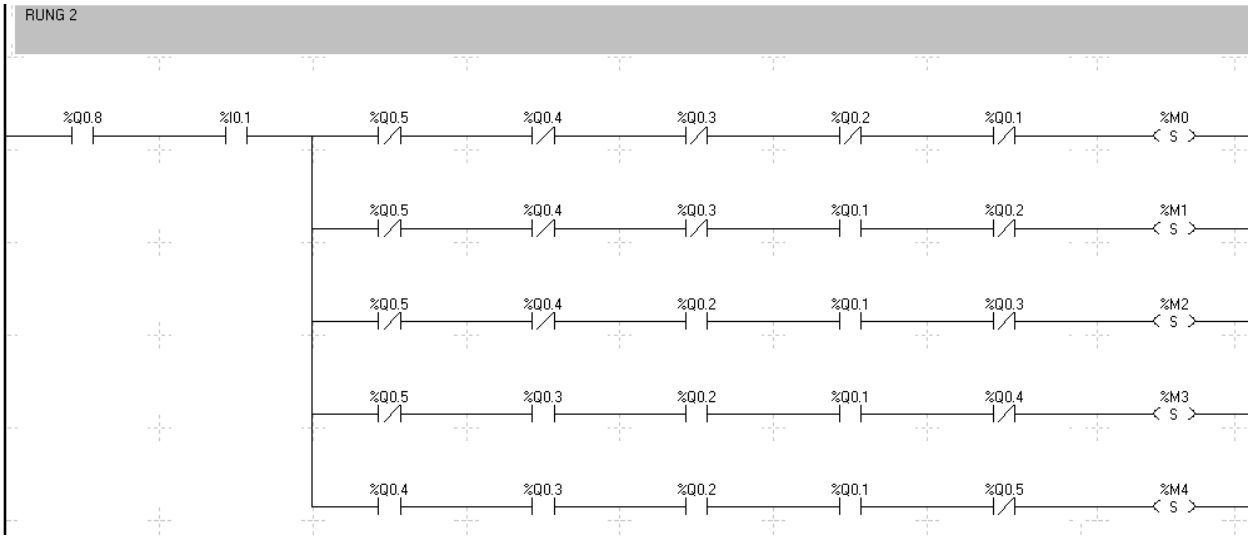


Rungul 1 verifica este activa starea 9 (adica daca a fost validat primul cod), daca se apasa codul de acces 2 si starea 8 este inactiva atunci se activeaza starea 8 (a fost validat si cel de-al doilea cod de acces) si devine inactiva starea 9.



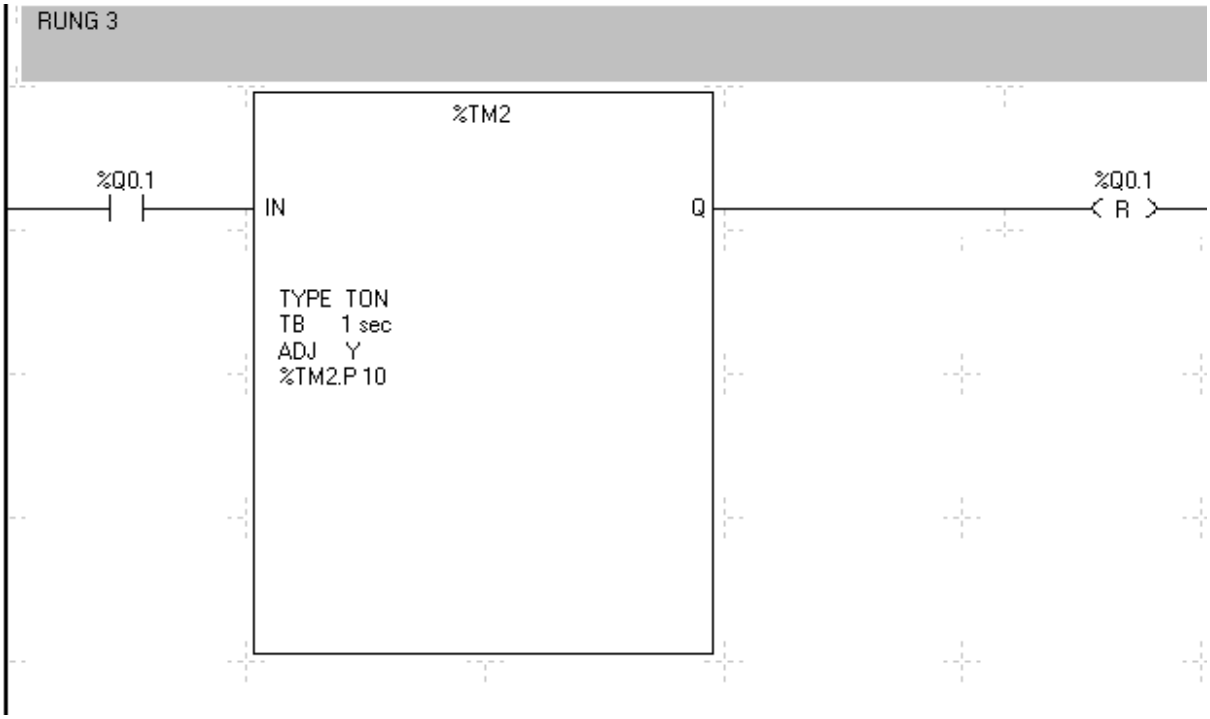
Rungul 2 verifica daca starea 8 e activa si se primeste codul de acces 1 (al doilea buton, cu adresa %I0.1), atunci se activeaza casutele din memorie corespunzatoare fiecarei parcari, pe rand. Casuta de memorie zero (cu adresa %M0) este corespunzatoare activarii parcarii 1, casuta de memorie 1 (cu adresa %M1) este corespunzatoare activarii parcarii 2, casuta de memorie 2 (cu adresa %M2) este corespunzatoare activarii parcarii 3, si tot asa pentru celelalte parcari.

Aceste casute de memorie lucreaza precum niste jump-uri, deoarece aveam de efectuat mai multe instructiuni si nu aveam loc intr-un singur rung.

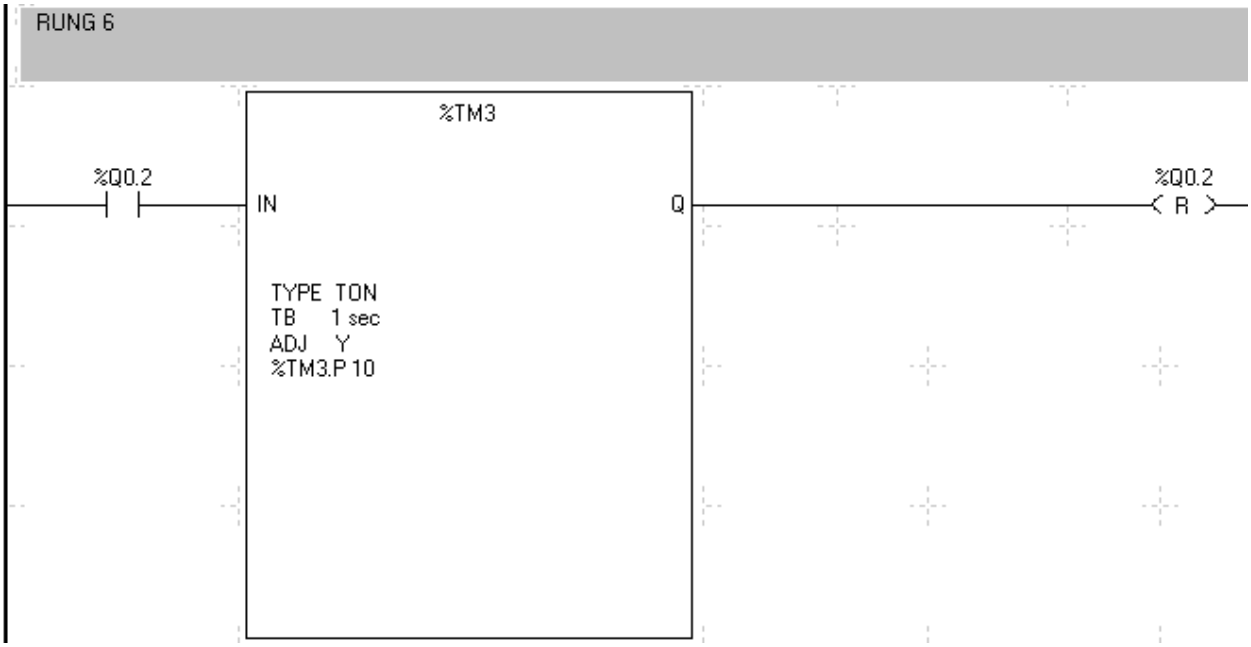


2. Rezervarea pentru o perioada de timp a parcarilor:

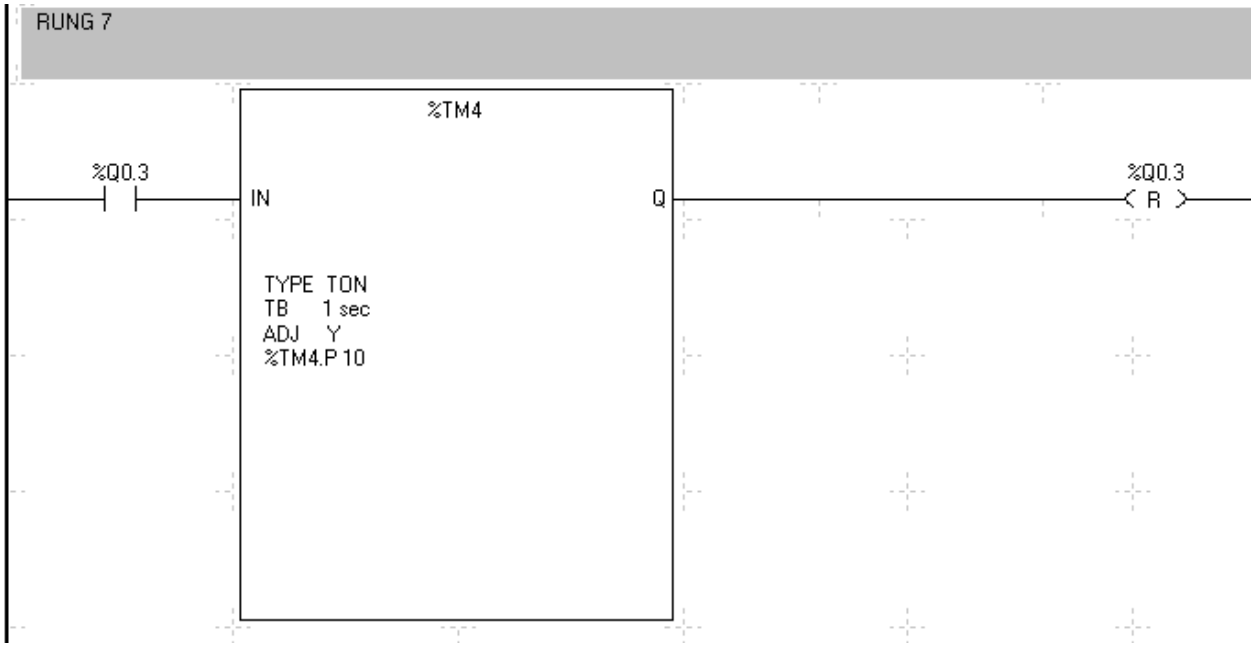
In rungul 3 se reseteaza starea 1 (in care parcarea a fost rezervata), se pastreaza parcarea pentru o perioada de timp (in cazul de fata 10 secunde). Dupa 10 secunde, starea 1 (cu adresa %Q0.1) se dezactiveaza (parcarea poate fi rezervata din nou.



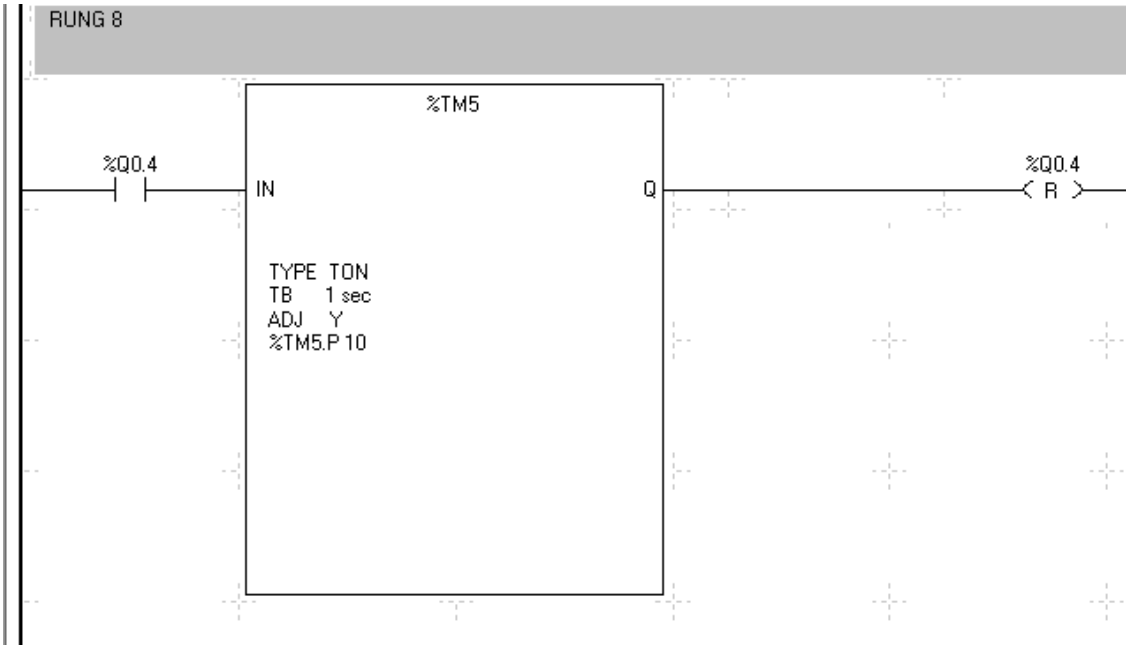
In rungul 6 se dezactiveaza starea 2 (cu adresa %Q0.2) in cazul in care a fost activata. Starea 2 este corespunzatoare parcarii cu numarul 2.



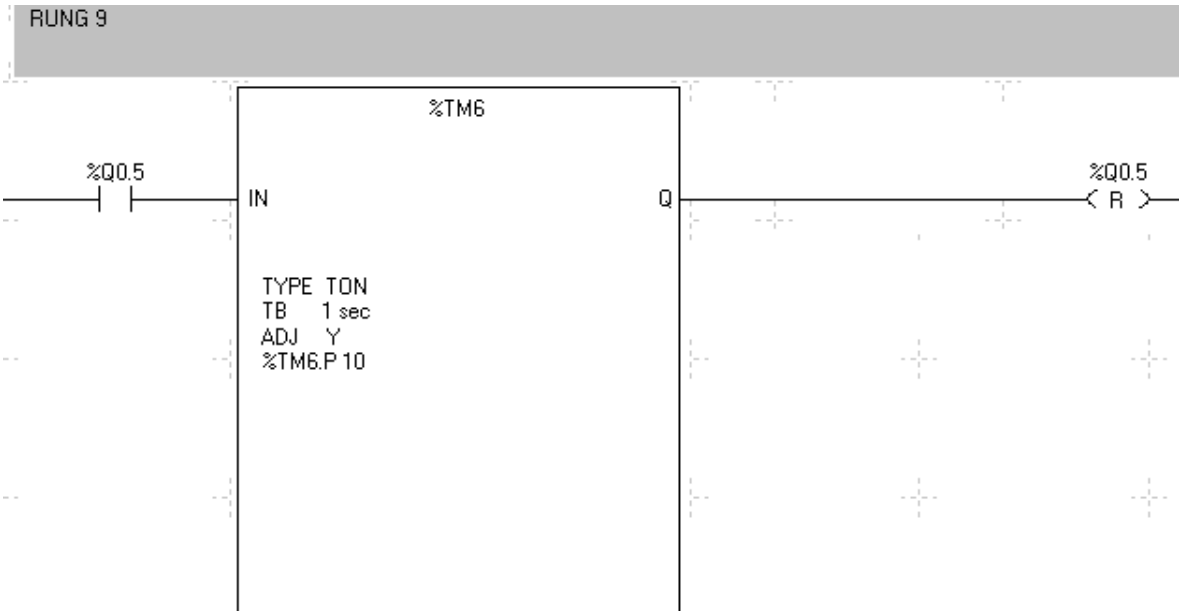
In rungul numarul 7 se dezactiveaza starea 3 (cu adresa %Q0.3) ce corespunde parcarii cu numarul 3.



In rungul numarul 8 se termina rezervarea parcarii numarul 4 (se dezactiveaza starea 4, cu adresa %Q0.4).

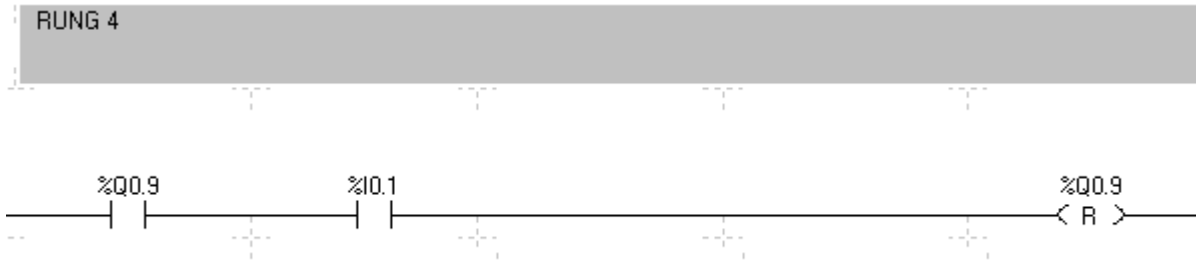


In rungul numarul 9 este utilizat un timer care opreste rezervarea din parcarea cu numarul 5 (se dezactiveaza starea corespunzatoare 5, cu adresa %Q0.5).

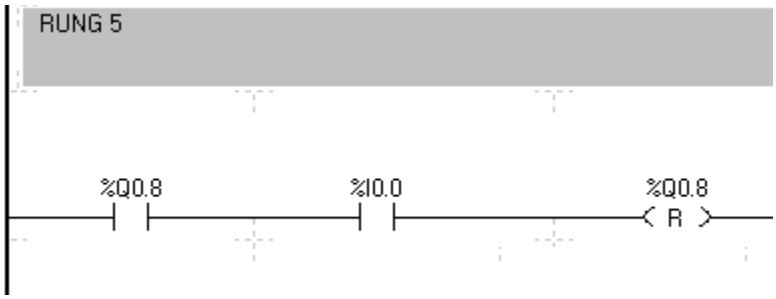


3. Verificarile suplimentare legate de codurile de acces introduse:

In rungul 4 este dezactivata starea 9 in cazul in care apesi codul de acces 1 (cu adresa %I0.1) deoarece nu este un cod corect.

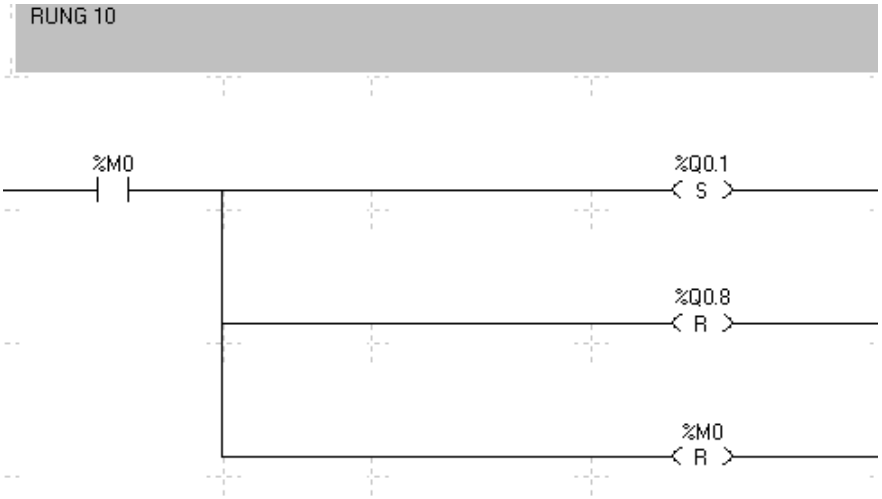


In rungul 5 este dezactivata starea 8 (cu adresa %Q0.8) in cazul in care este apasat codul de acces 0 (cu adresa %I0.0) deoarece este un cod gresit.

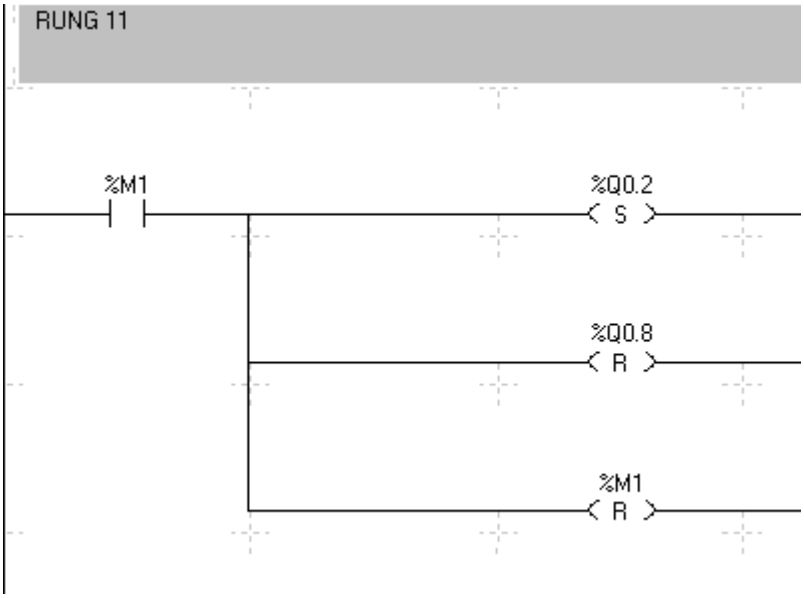


4. Rezervarea propriu-zisa a parcarilor:

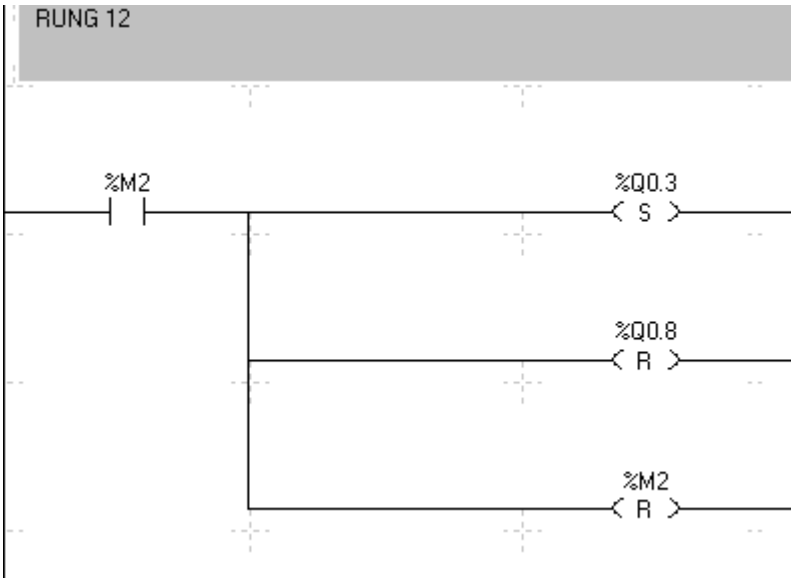
In rungul numarul 10 se verifica memoria 0 (cu adresa %M0). In cazul in care aceasta memorie are valoarea 1, atunci se reseteaza starea 8 (cu adresa %0.8), se reseteaza memoria 0 (cu adresa %M0) si se seteaza starea 1 (corespunzatoare rezervarii primului loc de parcare). In acest mod se rezerva primul loc de parcare.



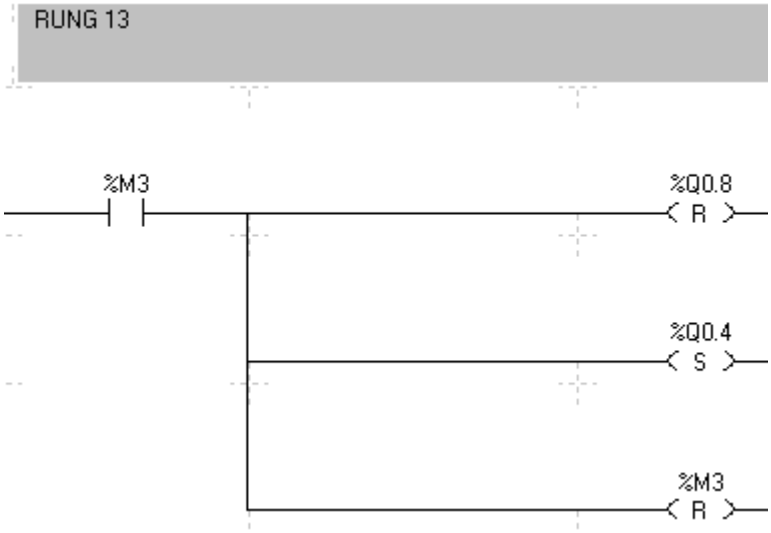
In rungul 11 se rezerva locul de parcare cu numarul 2. Este utilizata in acest scop memoria 1 (cu adresa %M1) care rezerva locul cu numarul 2 si reseteaza starea 8.



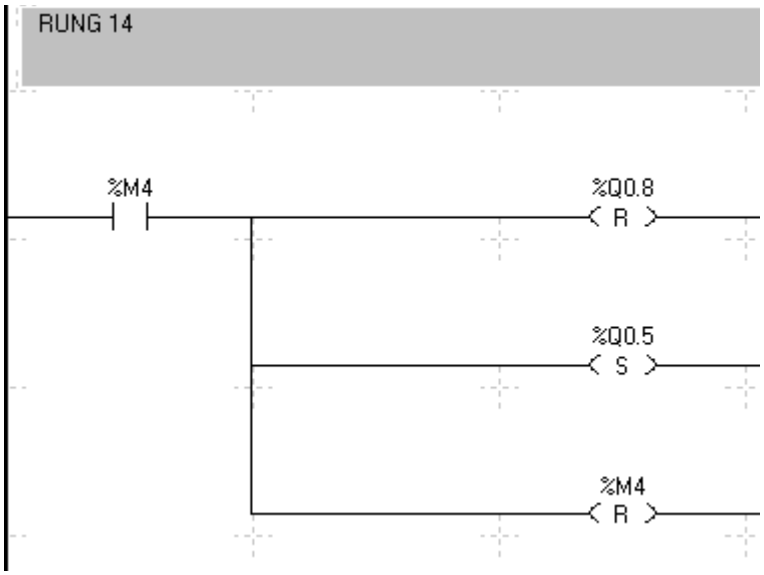
In rungul 12 se rezerva locul de parcare cu numarul 3. Memoria 2 este utilizata in locul unui jump, deoarece jump-urile creeau unele probleme.



In rungul 13 se rezerva locul pentru locul de parcare numarul 4. Memoria utilizata are adresa %M3 si este resetata imediat dupa rezervarea locului de parcare.



In rungul cu numarul 14 se rezerva ultimul loc de parcare. Memoria folosita are adresa %M4 si reseteaza adresele %Q0.8 respectiv %M4. Rezervarea ultimului loc de parcare se face prin setarea adresei %Q0.5.



Concluzii

- Aceasta gestiune de parcare, desi simpla, poate fi utilizata si in viata reala.
- Programul este simplu si usor de inteles.
- Am intalnit unele probleme la utilizarea jump-urilor dar am folosit memorii (%M0, %M1, …, %M4) si un mic artificiu in locul jump-ului propriu-zis.
- Am utilizat diverse rutine ale Twido, precum: Contact, Negated Contact, Set Coil, Reset Coil, Timer.
- Twido permite o manipulare destul de mare a automatului programabil, dar nu este indeajuns de flexibil si necesita o experienta destul de mare a programatorului pentru a modela aplicatii complexe.
- Din pacate mediul de programare Twido nu permite simulare, motiv pentru care procesul de dezvoltare este ingreunat.