CAPITOLUL **12**

Relee Logice, Logică programabilă Controlere și controlere de mișcare

OBIECTIVELE

După studierea acestui capitol, ar trebui să puteți:

- Explicați funcționarea releelor electromecanice, a releelor de întârziere în timp, a cronometrelor (numărătoarelor) și a secvențiatoarelor.
- Explicați scopul și funcționarea unei diagrame de tip scară.
- Explicați funcționarea unui controler bazat pe releu.
- Înțelegeți conceptul și scopul unui controler logic programabil (PLC).
- Înțelegeți hardware-ul și cablarea necesare într-un sistem bazat pe PLC.
- Enumerați pașii care trebuie urmați pentru ca un sistem de control PLC să fie operațional.
- Înțelegeți instrucțiunile de bază utilizate într-un program PLC.
- Diferențiați modurile în care un PLC poate fi programat.
- Înțelegeți modul în care PLC-urile sunt utilizate în rețea.
- Explicați scopul și funcționarea unui controler de mișcare.

INTRODUCERE

Controlul automat al proceselor mecanice sau fizice repetitive este o problemă comună privind controlul, care abundă în aparate majore, mașini de birou, industria prelucrătoare și industrie în general. În mod tradițional, acest tip de control al procesului a fost realizat cu dispozitive electromecanice, cum ar fi relee, cronometre și secvențiatoare. Cu această abordare, circuitul trebuie să fie reconectat în cazul în care logica de control se schimbă, care a fost o problemă specială pentru industria auto din cauza schimbărilor anuale de model. Ca răspuns la această problemă, la sfârșitul anilor 1960, General Motors a dezvoltat specificațiile pentru un controler electronic programabil care ar putea înlocui circuitele releu cu fir. Pe baza acestor specificații, Gould Modicon Company a dezvoltat primul controler logic programabil (PLC). PLC-ul este un computer mic, bazat pe microprocesor, de control al proceselor

care pot fi conectate direct la dispozitive precum întrerupătoare, motoare mici, relee și bobine și este construit pentru a rezista mediului industrial. Acest capitol va introduce principiile de control logic prin relee și apoi va descrie funcționarea generală și programarea PLC-urilor. Capitolul se încheie cu discuții privind rețelele PLC și controlerele de mișcare.

12.1 CONTROL LOGIC AL RELEELOR

Logica releelor

Releele ca dispozitiv sunt cuprinse în capitolul 4; cu toate acestea, o revizuire rapidă a elementelor de bază este prezentată aici. Un **releu electromecanic** (EMR) este un dispozitiv care utilizează forța electromagnetică pentru a închide (sau deschide) contactele de comutare - cu alte cuvinte, un comutator alimentat electric. Figura 12.1 prezintă o diagramă a unui releu. Contactele releului vin în două configurații de bază: **contacte deschise** în mod normal (NO), care sunt deschise în stare neactivată (și se închid atunci când bobina releului este activată) și contacte închise în mod **normal** (NC), care sunt închise în stare neactivată (și deschise atunci când releul este activat). Prin convenție, simbolul *releului arată întotdeauna contactele în stare neactivată.* Releele sunt disponibile cu o varietate de configurații cu mai multe contacte, dintre care două sunt prezentate în Figura 12.2.

Fara indoiala, releele au fost dezvoltate pentru prima dată pentru a satisface doua nevoi de control electric: telecomanda (capacitatea de a porni si opri dispozitivele dintr-o locatie la distanta) si amplificarea de putere. Un exemplu de amplificare a puterii este releul de pornire într-o mașină (un comutator de aprindere cu curent scăzut activează un releu, care, la rândul său, trece un curent ridicat la motorul de pornire). Proiectanții electrici au recunoscut curând că releele ar putea fi folosite pentru a implementa logica de control. ȘI, SAU, și NU porțile, precum și flip-flop-urile, pot fi cablate prin relee (Figura 12.3). Pentru poarta AND din figura 12.3 litera (a), releele A *ȘI* B trebuie să fie activate pentru ca o tensiune să fie la X. Figura 12.3 (b) arată o poartă OR. Aici, dacă fie releul A, *fie* B este activat, ieșirea X primește o tensiune. Figura 12.3 (c) arată o poartă releu NU. Dacă releul A este activat, ieșirea este de 0 V; dacă *NU* este activat, ieșirea primește o tensiune.

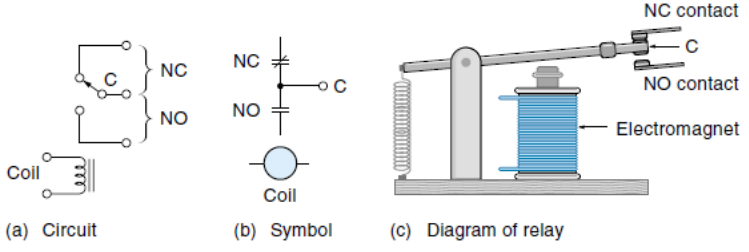
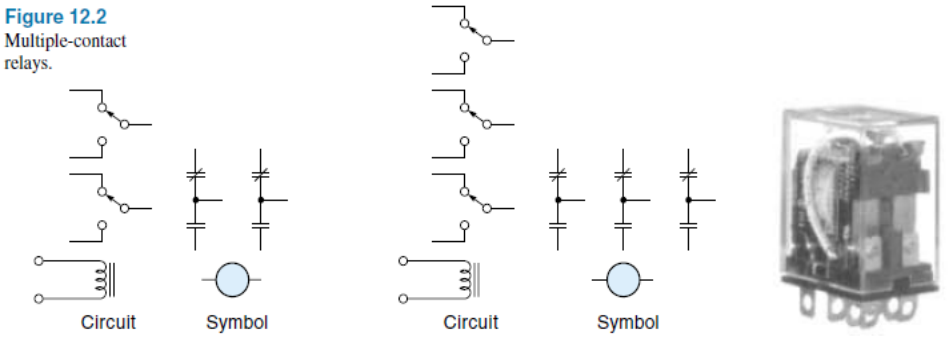


Figura 12.1
Releu: indicarea contactelor normal închise (NC) și normal deschise (NO)

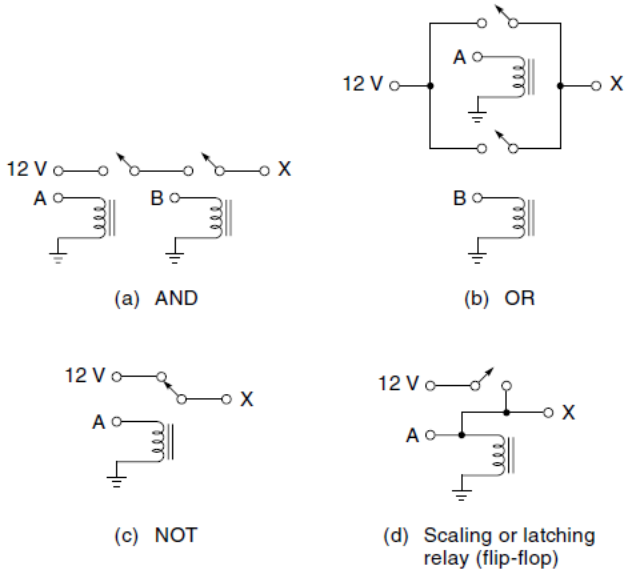
Figura 12.2
Relee cu contact multiplu.

Figure 12.2
Multiple-contact relays.



(a) Double-pole/double-throw (DPDT) (b) Triple-pole/double-throw (3PDT) (c) General-purpose relay (Courtesy of Rockwell Automation)

Figura 12.3
Funcții logice cu relee.



(a) AND (b) OR

(c) NOT (d) Scaling or latching relay (flip-flop)

Un flip-flop poate fi realizat dintr-un releu folosind un principiu numit **sigilare.** După cum se vede în figura 12.3 litera (d), odată ce releul este activat, o cale electrică prin contacte preia sarcina de a furniza energie bobinei, iar tensiunea originală A poate fi îndepărtată. (Acest proces este frecvent numit **zăvorât (blocare).** Cu toate acestea, un catalog de releu utilizează termenul *de releu de blocare* pentru a reprezenta un releu care are un mecanism de blocare mecanică încorporată.) Pentru a desface releul, puterea care vine prin contacte trebuie eliminată (temporar).

Diagrame scară

Comutatoarele și releele au devenit utilizate pe scară largă în industrie pentru controlul motoarelor, mașinilor și proceselor. Un comutator poate porni și opri o singură mașină, dar o rețea logică releu poate controla un întreg proces - pornirea unei mașini, așteptarea până la terminarea operației respective, apoi activarea următoarei operațiuni. Deciziile pot fi luate de logică - de exemplu, dacă o parte este prea înaltă, se duce într-un singur coș; în caz contrar, se duce în celălalt coș de gunoi.

În cele din urmă, **diagrama scării,** un tip special de diagramă de cablare, a fost dezvoltată pentru circuitele de control releu și comutare. Figura 12.4 prezintă o diagramă a scării (observați că seamănă cu o scară). Diagrama scării constă din două bare de **putere,** care sunt plasate vertical pe fiecare parte a diagramei, și *trepte*, care sunt plasate orizontal între barele de putere. Barele de alimentare sunt sursa de energie în circuit (AC sau DC), unde șina stângă este partea "fierbinte" (tensiune), iar șina dreaptă este neutră (AC) sau masă (DC). Prin urmare, fiecare ”treaptă” este conectată la sursa de tensiune și este un circuit independent. O treaptă conține de obicei cel puțin un set de contacte de comutare sau releu și o singură sarcină, cum ar fi o bobină de releu sau un motor. Când dispozitivele de contact dintr-o anumită treaptă

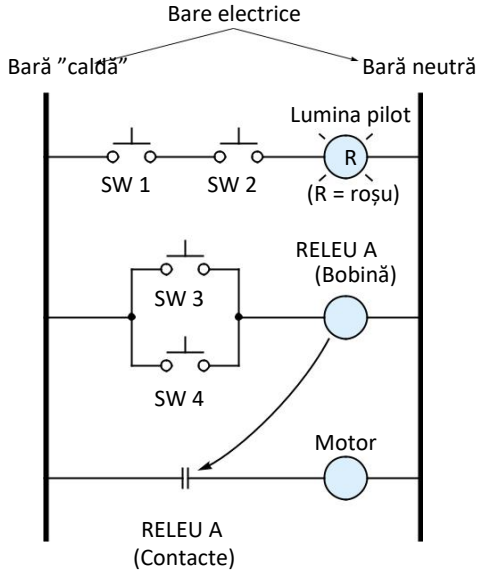


Figura 12.4
O diagramă de scară logică releu.

aproape de a face o cale continuă, apoi acea treaptă devine activă, iar sarcina sa este activată. De exemplu, treapta superioară din figura 12.4 conține două comutatoare și o lumină pilot în serie. Pentru ca treapta să devină activă, ambele comutatoare trebuie să se închidă, ceea ce va aplica apoi tensiunea luminii. Treapta din mijloc are două comutatoare în paralel, astfel încât doar unul dintre aceste comutatoare trebuie închis pentru a face treapta activă. Sarcina în această treaptă este o bobină releu (RELEU A). Treapta de jos conține un set de contacte FĂRĂ releu A, iar sarcina este un motor. În consecință, atunci când se închide SW 3 sau SW 4 al treptei medii, RELEU O bobină este activată, iar motorul (în treapta inferioară) pornește.

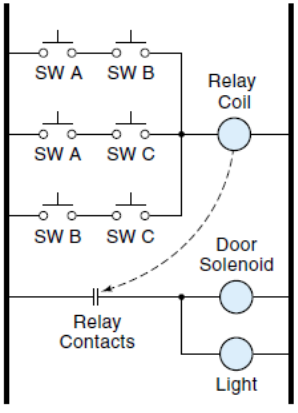
EXEMPLUL 12.1

Într-o anumită bancă, fiecare dintre cei trei ofițeri bancari are o cheie unică pentru seif. Regulile băncii impun ca doi din cei trei ofițeri să fie prezenți la deschiderea seifului. Desenați diagrama scării pentru un circuit logic releu care va desface ușa și va aprinde lumina atunci când sunt introduse două dintre cele trei taste.

SOLUȚIE

Trei combinații de taste vor deschide seiful: A și B, A și C și B și C. Fiecare dintre cele trei chei - A, B și C - se încadrează în propriul comutator de cheie care are două seturi de contacte FĂRĂ. Figura 12.5 prezintă diagrama scării completate. Treapta superioară are trei ramuri de contacte de comutare, câte una pentru fiecare posibilitate acceptabilă. Cel puțin o ramură trebuie să aibă continuitate, astfel încât bobina releului să fie activată. Treapta de jos, activată atunci când releul se închide, oferă putere solenoidului de blocare a ușii și luminii seifului.

Figura 12.5
O diagramă a scării (exemplul 12.1).



EXEMPLUL 12.2

Un robot simplu de preluare și plasare preia piese de pe o bandă transportoare și le plasează pe o altă bandă, așa cum se arată în figura 12.6 litera (a). Atunci când o parte care se deplasează de-a lungul benzii transportoare inferioare activează switch-ul 1, un dispozitiv de prindere alimentat cu solenoid se fixează pe piesă și o transportă spre banda transportoare superioară. Când dispozitivul de prindere ajunge la switch-ul 2, acesta eliberează piesa și se mută înapoi (gol) pentru a primi următoarea piesă. Când dispozitivul de prindere ajunge la Switch 3, se oprește și așteaptă ca următoarea piesă să înceapă ciclul din nou. Desenați diagrama scării logice releu pentru a controla această operație.

SOLUȚIE

Figura 12.6 litera (b) prezintă diagrama scării completate. Ciclul începe atunci când o piesă de pe banda transportoare inferioară activează Switch 1 - închizând momentan contactele *SW 1-1* (prezentate în treapta superioară). Această acțiune activează releul *MOTOR,* iar motorul începe să primească curent prin contactele *M-1* (treapta inferioară). Releul *MOTOR* este sigilat (blocat) prin contactele releu M-2 (acum chiar și atunci când se deschide SW *1-1,* releul *MOTOR* va rămâne activat prin contactele *M-2).*

În timp ce motorul este pornit, al doilea set de contacte Switch 1 (*SW 1-2*) în treapta 2 activează releul DIRECTION, determinând motorul să conducă *aderența* spre banda transportoare superioară (releul *DIRECTION* determină direcția motorului cu DPDT contacte D-3). Folosind aceeași tehnică ca și în treapta 1, releul *DIRECTION* este "pornit" printr-unul dintre contactele proprii (*D-1*).

Când mânerul ajunge la banda transportoare superioară, activează switch-ul 2, care deschide contactele *SW 2-1* închise în mod normal. Această acțiune întrerupe curentul releului DIRECTION, permițându-i să devină ne-activat, iar motorul comută direcția - acum întorcând spre partea de jos.

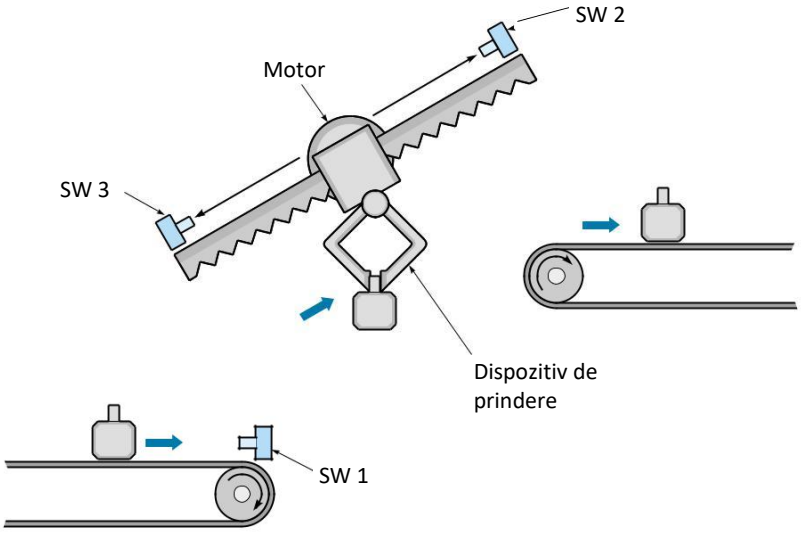
Treaptă 3 controlează solenoid GRIPPER-ului. Controlul GRIPPER-ului este simplu, deoarece *GRIPPER-ul* trebuie să fie activat pentru aceeași perioadă de timp în care releul *DIRECTION* este activat - adică din momentul în care motorul își începe călătoria în sus, până chiar înainte de a coborî.

Acțiunea finală a ciclului este atunci când mânerul coboară (gol) și activează comutatorul 3. Această acțiune deschide contactele închise în mod normal (*SW 3-1*), dezactivează motorul în treapta (care a ținut releul *MOTOR* activat), iar motorul se oprește.

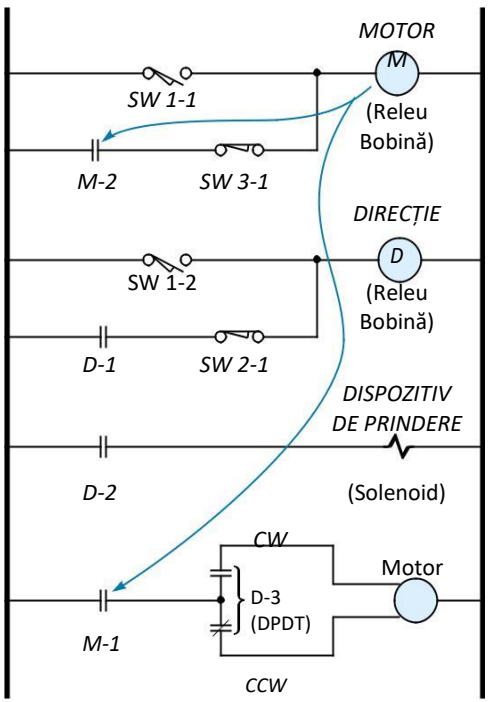
Cronometre, contoare și secvențiatoare

Multe situații de control necesită introducerea unei întârzieri de timp la un moment dat în proces. De exemplu, o operațiune de amestecare poate dura 90 s sau o bandă transportoare ar putea să ruleze 10 s pentru a atinge viteza înainte ca piesele să fie plasate pe ea. Un sistem de control releu ar folosi un **releu de întârziere pentru** a crea întârzierea de timp. Odată activat, releul de întârziere va aștepta o perioadă predeterminată de timp înainte ca persoanele de contact să se deschidă (sau să se închidă) - întârzierea

Figura 12.6
Controlul robotului pick-and-place (exemplul 12.2).



(a) Hardware



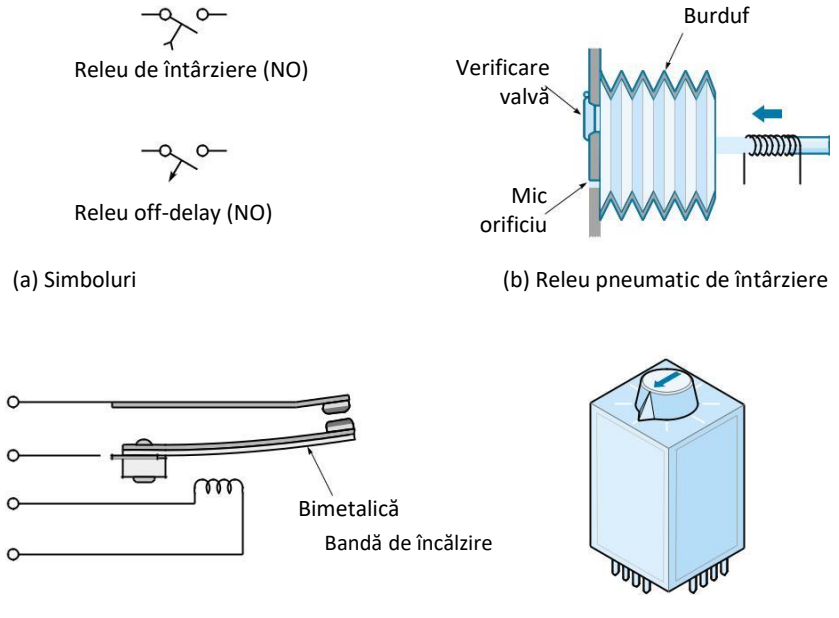
(b) Diagrama scării

poate varia de la mai puțin de o secundă la minut. Releele de întârziere a timpului sunt clasificate ca fiind fie *on-delay,* fie *off-delay* și sunt cel mai bine explicate prin exemplu. Un **releu** de 5 s **pe întârziere** va aștepta 5 s (după ce a fost activat) înainte de a închide contactele sale. Când releul este de-activat, contactele se vor deschide imediat. O aplicație a funcției on-delay este un sistem de securitate care întârzie activarea pentru o perioadă de timp după ce a fost activată, pentru a permite ocupanților să plece. Releul **off-delay** furnizează o întârziere de timp atunci când bobina este ne-activată. De exemplu, un releu off-delay de 5 s cu contacte FĂRĂ contacte și-ar închide contactele imediat când este activat; atunci când este ne-activat, cu toate acestea, contactele ar rămâne închise pentru încă 5 secunde înainte de deschidere. O aplicație a funcției off-delay sunt luminile auto care rămân aprinse pentru o perioadă de timp după ce au fost oprite (pentru a oferi lumină ocupanților în timp ce părăsesc vehiculul). Figura 12.7 litera (a) prezintă simbolurile utilizate pentru releele de întârziere.

Mai multe modele diferite de cronometre de întârziere, care utilizează principii de funcționare diferite, au evoluat de-a lungul anilor. **Releul pneumatic de întârziere** [Figura 12.7 (b)] funcționează după cum urmează: Când releul este activat, un burduf mic încărcat cu arcuri este închis, determinând aerul să scape printr-o supapă de verificare. Burdufurile sunt apoi lăsate să se extindă încet (aerul fiind evacuat printr-o gaură mică). Când burdufurile ajung din nou la dimensiunea normală, contactele se închid. Aceste relee sunt disponibile cu o întârziere fixă sau reglabilă.

Un **releu de întârziere termică** [Figura 12.7 (c)] utilizează o bandă bimetalică sensibilă la temperatură. Când releul este activat, un mic încălzitor de rezistență încălzește banda bimetalică

Figura 12.7
Relee de întârziere a timpului.



(a) Simboluri

(b) Releu pneumatic de întârziere

(c) Releu de întârziere termică solidă

(d) Releu de întârziere în stare

. Pe măsură ce banda se încălzește, se îndoaie și, în cele din urmă, închide contactele. Unitatea flasher dintr-o mașină, care controlează semnalele direcționale intermitente, funcționează pe acest principiu.

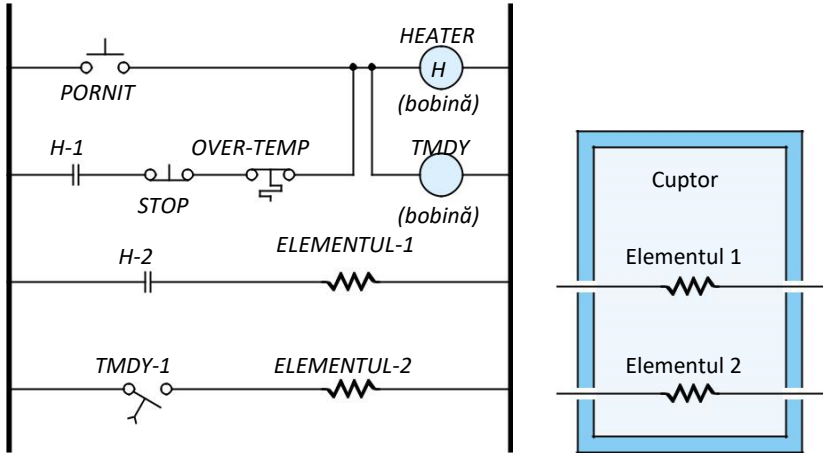
Releele de întârziere în solid-state sunt utilizate în majoritatea sistemelor mai noi care necesită relee de întârziere. Un exemplu este prezentat în figura 12.7 litera (d); butonul este pentru setarea întârzierii. Aceste unități se bazează pe întârzierea atunci când (1) se încărcă un condensator sau (2) numărarea impulsurilor de ceas de mare viteză cu un contor digital: cu cât numărul este mai mare, cu atât întârzierea este mai mare.

EXEMPLUL 12.3

Un cuptor electric mic are două elemente de încălzire care sunt activate în etape la 3 minute distanță. Adică, atunci când cuptorul este pornit, primul element de încălzire se aprinde imediat, iar al doilea element vine 3 min mai târziu. Un senzor de temperatură va închide cuptorul dacă se încălzește prea tare. Desenați diagrama scării pentru circuitul de control.

SOLUȚIE

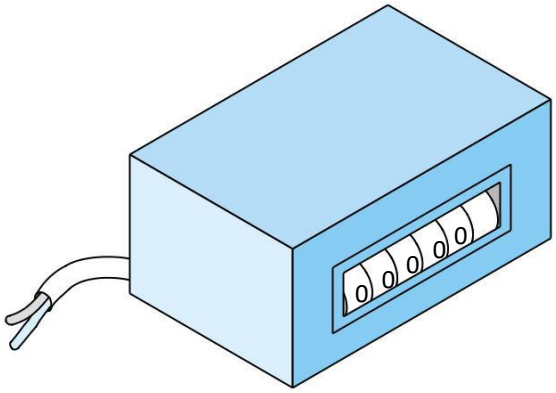
Pe treapta superioară a diagramei scării prezentată în Figura 12.8, un comutator cu buton activează releul HEATER, iar releul este sigilat de contactul *H-1*. Această acțiune furnizează energie primului element de încălzire (*ELEMENT-1*), prin contactele releului *H-2* și activează releul de întârziere *TMDY* (tip întârziere). După 3 minute, releul de întârziere activează al doilea element de încălzire prin contactele *TMDY-1* (treapta inferioară). În orice moment, dacă butonul de oprire este apăsat sau senzorul *OVER-TEMP* se deschide, sigiliul este rupt, iar releul DEACTIVEAZĂ, determinând închiderea ambelor elemente de încălzire.



(a) Diagrama scării (b) Cuptor

Figura 12.8
Controlul încălzitorului cuptorului (exemplul 12.3).

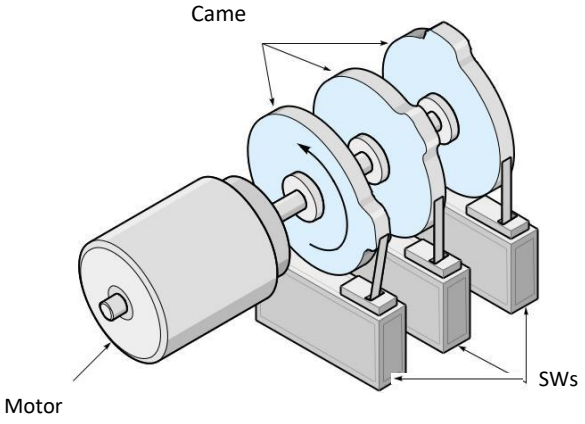
Figura 12.9
Un contor electromecanic.



Contoarele electromecanice sunt dispozitive care numără evenimentele. Contorul, care are o putere lizibilă similară cu odometrul unei mașini (Figura 12.9), crește o dată pentru fiecare puls electric primit. Contoarele, de obicei, pur și simplu țin evidența numărului de efectuate de o operație. De exemplu, ar putea număra numărul de produse care coboară pe linia de asamblare. Unele modele vor închide un set de contacte de comutare atunci când numărul ajunge la o valoare prestabilită.

Secvențiatoarele electromecanice controlează procesul care are o secvență de operații temporizate. Un exemplu în acest sens este un controler de mașină de spălat vase. Apa este încărcată mai multe secunde, apoi ciclul de spălare este activat timp de atâtea secunde, iar apoi apa este pompată timp de atâtea secunde și așa mai departe. După cum se arată în Figura 12.10, secvențiatorul (cunoscut sub numele de controler de *tambur*) constă dintr-un mic motor de sincronizare care rotește încet un cluster de came, iar camele activează comutatoarele. Fiecare comutator controlează una dintre operațiile temporizate, iar întreaga secvență se repetă cu fiecare revoluție a tamburului. Unele secvențiatoare folosesc un mecanism de cu solenoid pentru a roti camele în pași discreți, un pas pentru fiecare puls de intrare.

Figura 12.10
Un secvențiator electromecanic.



12.2 CONTROLERE LOGICE PROGRAMABILE

Introducere

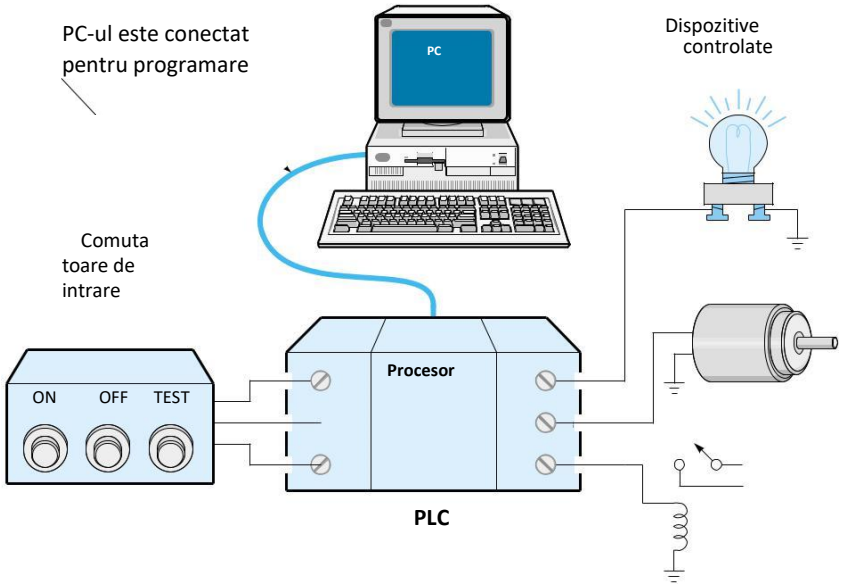
Un **controler logic programabil** (PLC) este un mic computer, autonom, robust, conceput pentru a controla procesele și evenimentele într-un mediu industrial - adică pentru a prelua job-ul făcut anterior cu controlere logice releu. Figura 12.11 prezintă un număr de tipuri diferite de PLC-uri. Din punct de vedere fizic, acestea variază în dimensiune de la cea a unei camere video la cea a unei cutii de pantofi (și uneori mai mari). Firele de la întrerupătoare, senzori și alte dispozitive de intrare sunt atașate direct la PLC; firele de alimentare ale lămpilor, motoarele mici, demaroarele motorului și alte dispozitive de ieșire sunt, de asemenea, conectate direct la PLC (Figura 12.12). Fiecare PLC conține un microprocesor care a fost programat pentru a conduce terminalele de ieșire într-un mod specificat, pe baza semnalelor de la terminalele de intrare. Programul PLC este de obicei dezvoltat pe un computer separat, cum ar fi un computer personal (PC), folosind software-ul special furnizat de producătorul PLC. Odată ce programul a fost scris, acesta este transferat sau descărcat în PLC. Din acest moment, PLC-ul poate funcționa pe cont propriu, ca un controler complet independent.

PLC Hardware

Figura 12.13 prezintă diagrama bloc pentru un PLC și include părțile fundamentale găsite în orice sistem de microprocesor (astfel cum se discută în capitolul 2). Aceste blocuri funcționale majore sunt explicate în continuare.

Figura 12.11
O selecție de PLC-uri. (Produse Allen-Bradley curtoazie de Rockwell Automation)

Figura 12.12
Un PLC și componentele sale conexe.



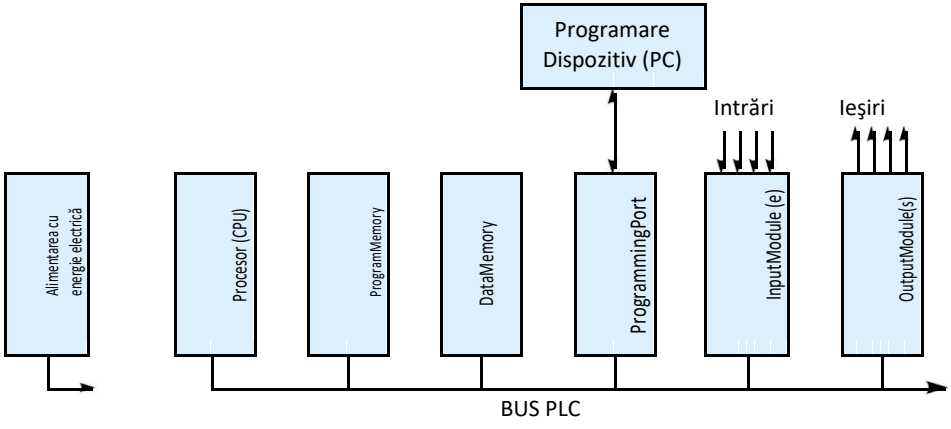
Alimentare

PLC-urile sunt de obicei alimentate direct de la 120 sau 240 Vac. Sursa de alimentare convertește curentul alternativ în tensiuni de curent continuu pentru componentele interne ale microprocesorului. De asemenea, poate furniza utilizatorului o sursă de tensiune redusă pentru a conduce switch-uri, relee mici, lămpi indicator, și alte asemenea dispozitive.

Procesor

Procesorul este bazat pe microprocesor și este partea PLC-ului care este capabil a citi și executa instrucțiunile programului, unul câte unul (cum ar fi treptele unui program logic scara).

Figura 12.13
Diagrama bloc a unui PLC.



Programul de memorie

Memoria programului primește și deține instrucțiunile programului descărcate de pe dispozitivul de programare. Dacă această memorie este ram standard, programul va fi pierdut de fiecare dată când puterea este oprită, necesitând ca aceasta să fie reîncărcată. Pentru a evita acest deranj, memoria programului poate utiliza un EEPROM sau o baterie de rezervă RAM, ambele sunt capabile să păstreze datele chiar și atunci când puterea este oprită. Un EPROM (UV erasable PROM) poate stoca, de asemenea, un program, dar acest dispozitiv necesită o unitate specială de programare care nu face parte din sistemul PLC.

Memorie de date

Memoria de date este memoria RAM folosită ca "scratch pad" de către procesor pentru a stoca temporar datele generate de programul intern și extern. De exemplu, ar stoca starea actuală a tuturor comutatoarelor conectate la terminalele de intrare și valoarea contoarelor interne și a cronometrelor.

Port de programare

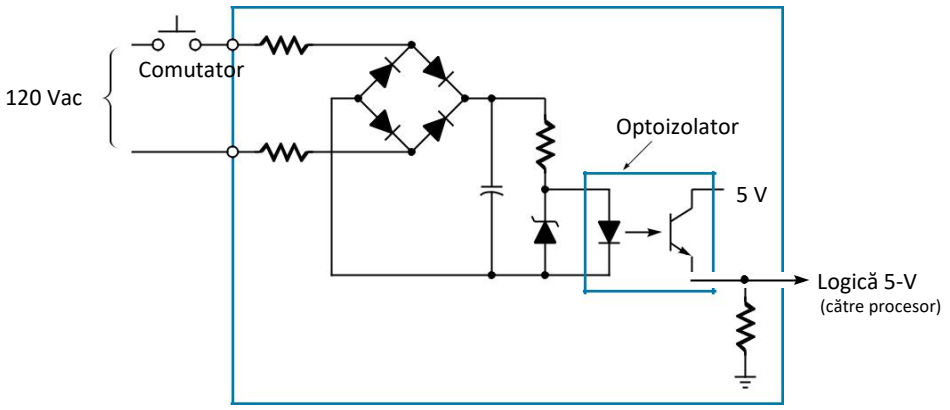
Portul de programare, un port de intrare/ieșire (I/O), primește programul descărcat de pe dispozitivul de programare (de obicei un PC). Amintiți-vă că PLC-ul nu are un panou frontal elaborat sau un monitor încorporat; astfel, pentru a "vedea" ce face PLC-ul în interior (pentru depanare sau depanare), trebuie să îl conectați la un computer, așa cum este ilustrat în Figura 12.12.

Module de intrare și ieșire

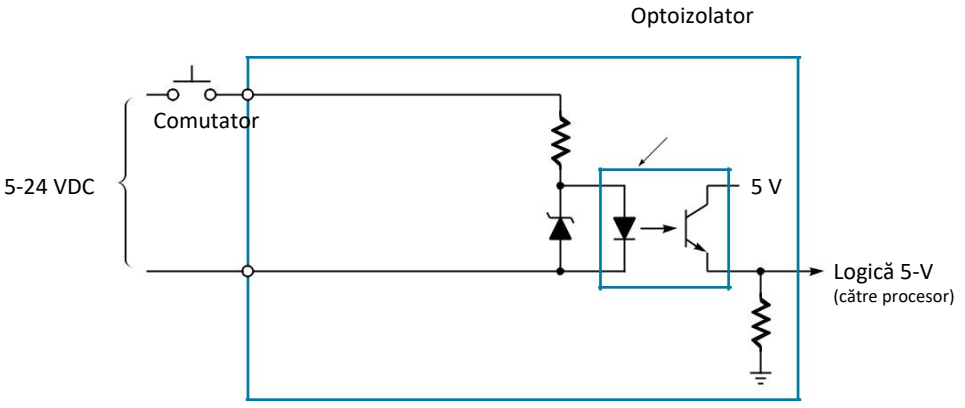
Modulele **I/O** sunt interfețe cu lumea exterioară. Aceste porturi de control pot fi încorporate în unitatea PLC sau, de obicei, sunt ambalate ca **module plug-in separate, unde** fiecare modul conține un set de porturi (a se vedea figurile 12.11 și 12.12). Cel mai comun tip de I/O se numește **I/O discret** și se ocupă de dispozitivele on-off. *Modurile discrete de intrare conectează* comutatoarele din lumea reală la PLC și sunt disponibile fie pentru AC, fie pentru DC (de obicei, 240 Vac, 120 Vac, 24 Vdc și 5 Vdc). După cum se arată în figura 12.14, circuitele din cadrul modulului convertesc tensiunea comutată într-o tensiune logică pentru procesor. Observați că o tensiune de comutare în curent alternativ trebuie mai întâi convertită într-o tensiune DC cu un redresor. De asemenea, un modul de intrare va include, de obicei, un optoizolator pentru a preveni posibile vârfuri de înaltă tensiune să intre în PLC. Un modul de intrare tipic discret ar avea 4, 8 sau 16 intrări.

Modulele de ieșire discrete oferă semnale on-off pentru a conduce lămpi, relee, motoare mici, demaroare de motor și alte dispozitive. După cum se arată în figura 12.15, sunt disponibile mai multe tipuri de porturi de ieșire: ieșirile Triac controlează dispozitivele AC, comutatoarele tranzistorului controlează dispozitivele DC și releele controlează dispozitivele AC sau DC (și asigură și izolarea). Un modul de ieșire discret tipic ar avea 4, 8 sau 16 ieșiri.

Figura 12.14
Circuitele modulului de intrare PLC.



(a) Circuitul modulului de intrare în curent alternativ



(b) Circuitul modulului de intrare dc

Modulele I/O analogice permit PLC-ului să gestioneze semnale analogice. Un *modul de intrare analogic* [Figura 12.16(a)] are unul sau mai multe ADC-uri (convertoare analogice-digitale), permițând senzorilor analogici, cum ar fi temperatura, să fie conectați direct la PLC. În funcție de modul, tensiunea sau curentul analogic este convertit într-un cuvânt digital pe 8, 12 sau 16 biți. Un *modul de ieșire analogic* [Figura 12.16(b)] conține unul sau mai multe DAC-uri (convertoare digital-analogice), permițând PLC-ului să furnizeze o ieșire analogică — de exemplu, să conducă un motor de curent continuu la diferite niveluri de tensiune.

Module specializate care îndeplinesc anumite funcții sunt disponibile pentru multe PLC-uri. Exemplele includ:

- *Modul termocuplu*- Interfațează un termocuplu la PLC.
- *Modul de control al mișcării*-Rulează independent pentru a controla mișcarea axei într-un dispozitiv, cum ar fi un robot (acoperit mai târziu în acest capitol).
- *Modul de comunicare*—Conectează PLC-ul la o rețea.

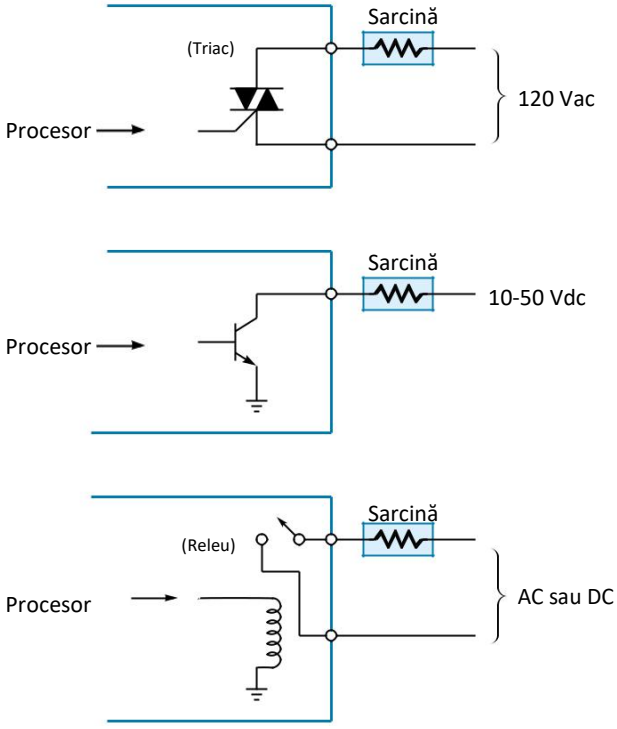


Figura 12.15
Ieșire PLC
circuite modul.

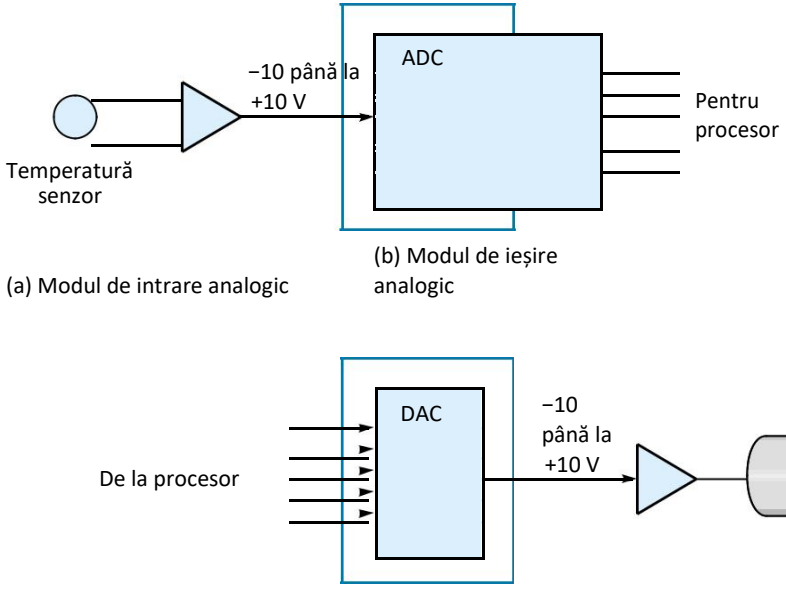


Figura 12.16
Module I/O analogice.

- *Modul contor de mare viteză*- Contorizează numărul de impulsuri de intrare pentru o perioadă fixă de timp.
- *Modul PID*- Un controler autonom PID care rulează independent (controlul PID poate fi implementat și cu software, așa cum este descris mai târziu în acest capitol).

Magistrala PLC

Magistrala PLC (Figura 12.13) sunt firele din *backplanul* unui sistem modular PLC, care conține magistrala de date, magistrala de adrese și semnalele de control. Procesorul utilizează magistrala pentru a comunica cu modulele.

Procedura de instalare PLC

PLC-ul a evoluat ca un înlocuitor pentru logica releului cu fire și a fost proiectat pentru a fi instalat, programat, utilizat și întreținut de tehnicieni care erau familiarizați cu releele industriale și circuitele de comutare. Din acest motiv, producătorii de PLC-uri au dezvoltat un limbaj de programare care permite utilizatorului să programeze logica de control dorit sub forma unei diagrame scara. Diagrama scării poate fi introdusă în sistem fie prin desenarea acesteia pe ecranul *PC-ului,* fie prin utilizarea unei casete de comutare special adaptate (numită programator de mână (HHP)). Dacă PC-ul este utilizat, trebuie instalat un software special înainte ca diagrama scării să poată fi introdusă.

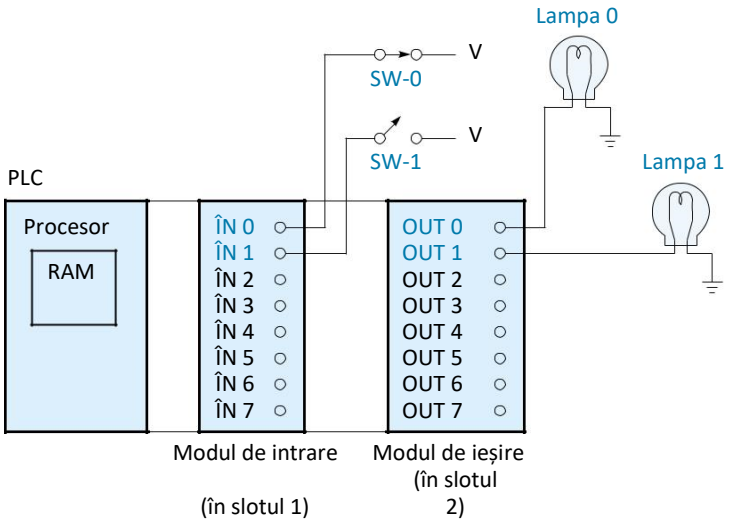
Presupunând că diagrama scării a fost introdusă într-un PC (sau un dispozitiv terminal asemănător PC-ului), următorul pas este să conectați PC-ul la PLC. În cele mai multe cazuri, pur și simplu conectați cele două unități cu un cablu sau un circuit de interfață. Un sistem mai sofisticat poate avea un număr de PLC-uri conectate la un singur PC pe o rețea LAN (rețea locală). Cu acest sistem, PC-ul poate comunica cu orice PLC din rețea. Odată ce conexiunea a fost stabilită, programul este descărcat de pe PC în PLC.

Cu programul încărcat, PLC-ul este gata să funcționeze și, în orice aplicație reală, următorul pas este testarea și depanarea. Testarea se poate face în două moduri: abordarea software totală, în care intrările sunt simulate de la terminalul de programare (PC) sau o abordare mai reală, în care intrările și ieșirile sunt conectate la destinațiile lor reale. În ambele cazuri, ecranul PC-ului devine o "fereastră" în funcționarea programului; dacă o problemă devine evidentă, programul poate fi corectat imediat și testarea continuă până când toate bug-urile sunt depanate. Când testarea este completă, PC-ul poate fi deconectat, iar PLC-ul va continua să execute programul său stocat ca o unitate independentă.

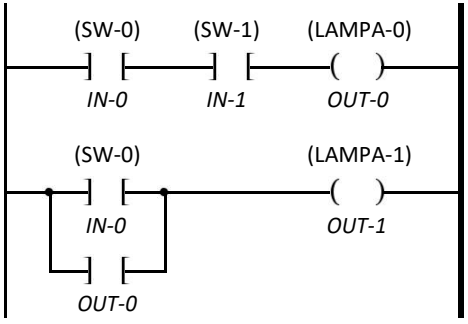
Operare PLC

PLC-ul își îndeplinește job-ul de a implementa programul de control logic al scării în stilul computerului tipic - pas cu pas. Pentru a vedea cum face acest lucru, să examinăm o aplicație simplă a PLC-ului, de pornire și oprire a lămpilor [Figura 12.17 (a)]. Observați că PLC-ul are un modul de intrare și un modul de ieșire. Două comutatoare externe (SW-0 și SW-1) sunt conectate la PLC prin terminalele *IN-0* și *IN-1* ale modulului de intrare. Terminalele

Figura 12.17
O aplicație PLC simplă și diagrama scării.



(a) Hardware



(b) Diagrama scării

modulului de ieșire (*OUT-0* și *OUT-1*) conduc două lămpi indicatoare (LAMP-0 și LAMP-1). Diagrama scării pentru această aplicație are doar două trepte [Figura 12.17 (b)]. Treapta superioară va aprinde LAMP-0 dacă *atât* SW-0, cât și SW-1 sunt închise. Treapta de jos va aprinde LAMP-1 dacă *fie* SW-0 sau *OUT-0* sunt închise. (De asemenea, *OUT-0* contacte pot fi gândite ca un releu contacte cu bobina *OUT-0* în treapta 1.)

Să presupunem că programul a fost încărcat în PLC și se află în secțiunea pro-gram de memorie. Când PLC-ul este setat să **ruleze modulul,** începe executarea programului. Execuția se realizează ca o serie de pași, care se repetă la infinit. Un ciclu complet al tuturor acestor pași este o **scanare.** Pașii sunt după cum urmează:

1. PLC-ul citește datele din modulul de intrare și stochează valorile tuturor celor opt intrări, ca cuvânt, în secțiunea Date de intrare a memoriei (Figura 12.18). Fiecare bit din cuvânt reprezintă starea prezentă a unui comutator extern.

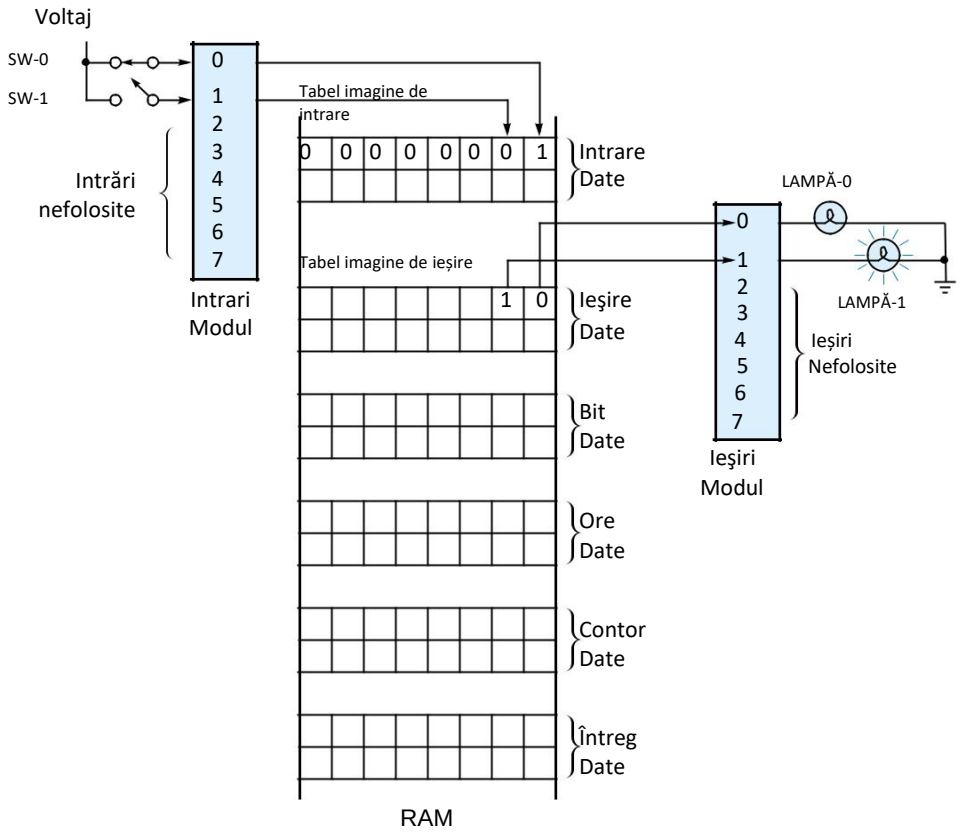


Figura 12.18
Date în memorie, afișând relația dintre poziția memoriei și terminalele I/O.

2. PLC-ul își îndreaptă atenția către diagrama scării, începând cu prima treaptă. Primele două simboluri din treaptă sunt comutatoarele de intrare, astfel încât procesorul le citește starea *din RAM*. Apoi evaluează logica indicată de aceasta. În acest caz, SW-0 este pornit, și SW-1 este oprit, astfel încât *OUT-0 nu* va fi activat. Această stare a *OUT-0* este stocată ca un bit 0 în secțiunea Date de ieșire din RAM (Figura 12.18).
3. După ce se ocupă complet de prima treaptă, PLC-ul își îndreaptă atenția către treapta a doua. Starea *IN-0* și *OUT-0* sunt citite din RAM. Folosind aceste date, PLC-ul efectuează logica programată, care este o operațiune OR între *IN-0* și *OUT-0* (amintiți-vă că *OUT-0* este ieșirea treptei superioare și este în prezent dezactivată). În acest caz, deoarece SW-0 este activat, *OUT-1 va* fi activat, iar acest rezultat este stocat ca un pic 1 în secțiunea Date de ieșire din RAM. Aceasta finalizează acțiunea pe a doua treaptă. Dacă ar exista mai multe trepte, fiecare ar fi fost evaluată în ordine până când toate ar fi fost executate.
4. Ca acțiune finală a scanării, datele de ieșire actualizate în RAM sunt trimise la modulul de ieșire, determinând actualizarea simultană a tuturor celor opt terminale de ieșire. În acest caz, LAMP-0 ar fi oprit, iar LAMP-1 ar fi pornit. Acum PLC-ul va intra într-o stare de repaus timp de câteva secunde sau mai puțin până când timpul de scanare desemnat a trecut și apoi întregul proces începe din nou cu pasul 1.

Privind peste procesul descris mai sus, unele observații pot fi făcute. În primul rând, PLC-ul își îndeplinește misiunea de control în același mod în care funcționează toate controlerele digitale, executând o buclă (scanare) din nou și din nou. (Pentru PLC, cu toate acestea, acțiunea de buclă este automată și nu trebuie programată.) Pentru fiecare scanare, intrările sunt citite, ieșirile sunt calculate pe baza intrărilor și apoi ieșirile sunt trimise în lumea reală. Aceasta înseamnă că o întârziere de până la o oră de scanare poate fi între o intrare și o ieșire. O astfel de întârziere nu este prezentă într-un adevărat circuit de releu cu fir, unde singura întârziere ar fi timpul de propagare al releelor. Acest lucru duce la a doua observație: Ordinea treptelor din diagrama scării poate fi importantă deoarece treptele sunt executate secvențial de sus în jos.

O observație finală a funcționării hardware PLC este că, în toate scopurile practice, poate face multe operațiuni de control independente în același timp. Am putea fi condiționat să se gândească la un computer ca fiind capabil de a face doar un singur lucru la un moment dat, deși foarte rapid. Și, după cum am văzut, PLC-ul își execută programul pe rând. Cu toate acestea, fiecare treaptă este un circuit separat, iar fiecare aplicație de control (în măsura în care PLC-ul este programat) este doar un set de trepte. Prin urmare, ai putea avea un PLC care rulează trei aplicații de control diferite în același timp. Fiecare aplicație ar avea propriul terminal I/O atribuit și propriul set de trepte din program. În cadrul unui ciclu de scanare a programului, PLC-ul ar actualiza cele trei seturi de ieșiri. Acest lucru este similar cu un maestru de șah, care joacă cu trei adversari simultan.

12.3 PROGRAMAREA PLC-ULUI

Diagrama scară de programare

Cel mai comun mod de a programa PLC-ul este de a proiecta circuitul de control dorit sub forma unei diagrame de scară logică cu relee și apoi se introduce această diagramă a scării într-un terminal de programare (care ar putea fi fie un PC, sau un terminal de programare PLC dedicat, fie o cutie specială de comutare portabilă). Terminalul de programare este capabil să convertească diagrama scării în coduri digitale și apoi să trimită acest program la PLC unde este stocat în memorie. Există și alte modalități de a programa un PLC în afară de **diagramele ladder**, iar acestea vor fi introduse mai târziu în acest capitol.

În prezent, există mulți producători de PLC-uri și, deși toate au asemănări de bază, detaliile hardware-ului și software-ului vor fi, desigur, diferite. Următoarea discuție privind programarea va fi păstrată cât mai generală posibil, dar este patentată după linia Allen-Bradley de PLC-uri. Aceasta pare o alegere logică, deoarece Allen-Bradley este unul dintre pionierii în domeniul PLC și produsele sale sunt respectate și bine utilizate de industrie.

PLC-ul se ocupă cu două tipuri de date care sunt menținute în memorie: un **fișier de program** în RAM sau EEPROM, care conține programul logic al scării și fișiere de **date** în

RAM, care deține datele de modificare din aplicația de control, cum ar fi setările de comutare. Tabelul 12.1 enumeră unele dintre cele mai frecvente tipuri de fișiere de date. Fiecare fișier de date ocupă o parte din memoria RAM și constă din orice număr de cuvinte pe 8 sau 16 biți. Acest aranjament de date este ilustrat în figura 12.18.

Fișierele date de intrare și ieșire sunt gestionate într-un mod special. Automat PLC transferă starea on-off a tuturor comutatoarelor de intrare direct la fișierul date de intrare în RAM, ca parte a scanării. Aceasta se numește realizarea unui *tabel de imagini de intrare* (așa cum se indică în figura 12.18). Observați că un comutator închis devine un 1 în RAM și un comutator deschis devine un 0. În mod similar, datele din fișierul Output Data (denumit tabelul *de imagini de ieșire*) sunt transferate automat către terminalele modulului de ieșire.

Astfel, programul PLC se ocupă de fapt numai cu datele de imagine din diferitele lor fișiere de date. Acesta este motivul pentru care fiecare dispozitiv de pe diagrama scării - fie că este vorba de comutator, releu, cronometru sau orice altceva - este identificat prin adresa sa în RAM. Chiar dacă programatorul se poate gândi la un anumit comutator ca fiind, să zicem, un comutator de limită de preaplin, software-ul se gândește la acesta ca fiind o anumită informație într-un anumit fișier de date.

Există o diferență fundamentală importantă între o diagramă a scării logice releu și o diagramă a scării PLC. *Diagrama scării releului este în esență o diagramă de cablare*, iar o treaptă devine activă atunci când curentul poate curge de la o șină de alimentare la alta; adică, dacă există o cale pentru curent prin contacte, atunci sarcina va fi activată. Pe de altă parte, *diagrama scării PLC este în esență o diagramă logică* formată din simboluri (numite instrucțiuni de intrare și ieșire), iar o treaptă devine continuă atunci când există o cale TRUE logică de la șină la șină. Referindu-se la Figura 12.17 (b), dacă există o cale continuă a instrucțiunilor de intrare TRUE în partea stângă a unei trepte, atunci instrucțiunea de ieșire (în partea dreaptă) va deveni TRUE. Prin urmare, scrierea unui program PLC implică plasarea instrucțiunilor în trepte, astfel încât să se obțină rezultatul dorit. Desigur, puteți utiliza numai instrucțiuni care sunt acceptate de limba utilizată, dar majoritatea PLC-urilor acceptă componentele de bază, cum ar fi comutatoare, relee, cronometre, contoare și secvențiatoare. Vom discuta despre fiecare tip de instruire în continuare.

TABELUL 12.1

Tipuri de fișiere de date

	Fișier de date	Descriere
I	Date de intrare	Starea curentă a comutatoarelor conectate extern
O	Date de ieșire	Starea dispozitivelor specificate ca ieșiri
B	Date de biți	"Tamponul" pentru a stoca biți individuali
T	Date cronometru	Datele asociate cu cronometrele
C	Date contor	Datele asociate contoarelor
R	Date de control	Datele asociate secvențiatoarelor și a altor dispozitive
N	Date întregi	Numere, cum ar fi temperaturile (în formă binară)

Instrucțiuni de biți

Switch-urile și releele sunt denumite **instrucțiuni de biți,** deoarece necesită doar 1 bit pentru a descrie dacă un comutator este deschis sau închis. Există două tipuri de instrucțiuni de comutare: unul reprezintă un comutator NO, iar celălalt un comutator NC. Următoarele sunt simbolul și instrucțiunile pentru comutatorul NO:

NO instrucțiune: -] [— (EXAMINAȚI DACĂ ON, Allen-Bradley)

Când acest simbol este plasat în diagrama scara, se comportă ca un comutator NO, împinge un buton, sau un set de contacte releu. Acesta poate fi activat fie printr-un comutator din lumea reală, fie printr-un alt comutator logic de pe diagrama scării. În ambele cazuri, atunci când este activată, instrucțiunea NO din diagrama scării merge TRUE. Starea instrucțiunii NO este menținută de un anumit bit în memorie, unde un 0 înseamnă oprit (FALSE) și un 1 înseamnă pornit (TRUE). Adresa acestui bit este plasată pe diagrama scării lângă simbolul instrucțiunii.

Următoarele sunt simbolul și instrucțiunile pentru comutatorul NC:

NC Instrucțiune: —]/[— (EXAMINE IF OFF, Allen-Bradley)

Când acest simbol este plasat în diagrama scării, se comportă logic în același mod în care o face orice comutator NC - adică este în mod normal TRUE și când este activat merge FALSE (se comportă ca un invertor logic). Trebuie avut grijă atunci când se utilizează această instrucțiune în conjuncție cu un comutator NC din lumea reală, deoarece este posibil să nu se comporte așa cum credeți. Problema este că oamenii pierd din vedere faptul că un comutator nc din lumea reală și instruirea NC *nu* sunt aceeași entitate, ci, mai degrabă, *unul conduce de altă parte;* adică comutatorul fizic controlează starea instrucțiunii NC. Să presupunem că de fapt, cu fir un comutator NC real la PLC și a folosit-o pentru a controla o instrucțiune NC. Apoi, dacă comutatorul real a fost activat, acesta ar trimite un 0 la PLC, iar instrucțiunea NC logică ar inversa acest FALSE la un TRUE, care poate fi în contradicție la ceea ce vă așteptați. Pentru a evita această cădere , amintiți-vă că comutatorul din lumea reală poate conduce comutatorul logic, dar *nu* sunt același lucru.

Următorul simbol reprezintă o bobină releu sau pur și simplu un semnal de ieșire:

OUTPUT —( )— (IEȘIRE ENERGIZE, Allen-Bradley)

Dacă există o cale TRUE continuă prin treaptă, atunci ieșirea va merge TRUE; astfel, dacă acest simbol reprezintă o bobină releu, releul se va activa. Ceea ce se întâmplă cu adevărat în interiorul PLC-ului este că bitul din memorie corespunzător acestei instrucțiuni merge la un 1. Acest bit poate fi folosit pentru a conduce un releu hardware real printr-un modul de ieșire sau pentru a controla o instrucțiune într-o altă treaptă, sau ambele. De fapt, nu există nici o limită privind numărul de ori în care acest bit poate fi folosit de alte instrucțiuni în program (care este un alt avantaj PLC-urilor).

Funcționarea instrucțiunilor pe trei biți este rezumată în tabelul 12.2.

TABELUL 12.2

Rezumatul instrucțiunilor bit

Dacă datele bit din fișier este	NO instrucțiune -] [—	Instrucțiune NC —]/[—	OUTPUT –( )—
Logica 0	FALS	ADEVĂRAT	FALS
Logica 1	ADEVĂRAT	FALS	ADEVĂRAT

Într-o diagramă de scară PLC, fiecare simbol este însoțit de adresa sa RAM (Figura 12.19), deoarece această adresă spune PLC-ului unde să găsească starea logică actuală a simbolului. Amintiți-vă că starea logică a fiecărui simbol este întotdeauna menținută în MEMORIA RAM. Titlurile componentelor din lumea reală pe care le reprezintă simbolurile, cum ar fi *Overload Sw*,sunt adăugate frecvent pentru a îmbunătăți claritatea, dar PLC-ul în sine se ocupă strict de adrese. Adresa unui bit individual poate fi specificată cu trei cantități: tipul de fișier, numărul cuvântului și poziția bitului în cadrul cuvântului. * După cum sa menționat anterior, semnalele I / O semnalele au o relație directă între adresa lor și punctul de conexiune fizică pe un modul I / O. De exemplu, în treapta de sus al figurii 12.19, adresa *Flow sw* este dată ca I : 1/0. I standuri pentru fișierul de date de intrare, 1 înseamnă că modulul de intrare este conectat la slotul 1 al șasiului PLC, iar 0 înseamnă că comutatorul este cu fir la terminalul 0 al modulului. Aceasta este o interpretare mai generală a adresei de instrucțiuni *Flow sw:*

Tipul fișierului de date (intrare în acest caz)
Adresa cuvântului în fișierul de date (sau slotul modulului I/O #)
Poziția bitului în interiorul cuvântului, dacă este cazul (sau terminalul # al modulului I/O)
I:1/0
———] [———

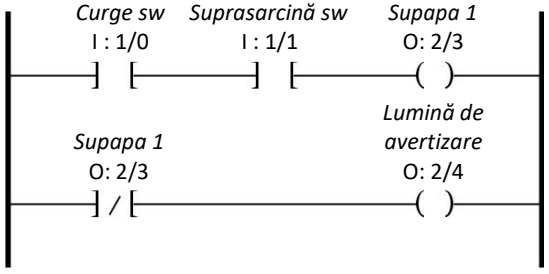


Figura 12.19
Un program PLC simplu.

A doua instrucțiune NO de pe prima treaptă (Figura 12.19) reprezintă un comutator cu fir la terminalul 1 al aceluiași modul de intrare utilizat de *Flow sw*. Instrucția OUTPUT activează terminalul 3 al modulului de ieșire conectat la slotul 2.

Funcționarea treptei superioare din Figura 12.19 este după cum urmează: Ieșirea *Valvei 1* va merge TRUE atunci când există continuitate prin treaptă. Continuitatea va fi stabilită atunci când ambele instrucțiuni NU sunt ADEVĂRATE - cu alte cuvinte, atunci când contactele atât Flow *sw,* cât și *Supraîncărcarea sw* sunt închise.

Funcționarea celei de-a doua trepte ilustrează două concepte importante. În primul rând, observați că adresa instrucțiunii NC este aceeași cu instrucțiunea de ieșire din treapta superioară, ceea ce înseamnă că instrucțiunea NC va fi controlată de *supapa 1* bit (situată la adresa 0:2/3). În al doilea rând, pentru că este o instrucțiune NC, va merge TRUE numai atunci când *Supapa 1* este FALSĂ. Deci, *lumina de avertizare* va activa atunci când supapa se stinge.

Cronometre

Instrucțiunea Timer oferă o întârziere de timp, îndeplinind funcția unui releu de întârziere. Exemple de utilizare a întârzierilor de timp includ controlul timpului pentru o operație de amestecare sau durata unui semnal sonor de avertizare. Durata întârzierii este determinată prin specificarea unei valori presetate. Cronometrul este activat atunci când condițiile de treapta devine TRUE. Odată activat, acesta cronometrează automat până când ajunge la valoarea Presetare și apoi merge TRUE (și rămâne TRUE). Simbolul instrucțiunii cronometru este o casetă sau un cerc cu informațiile pertinente plasate în sau despre aceasta, așa cum se arată și se explică în Figura 12.20. Observați că instrucțiunea Cronometru este plasată în treptă. Când *Switch* I : 1/3 merge TRUE, cronometrul începe să numere. Cinci secunde mai târziu, când valoarea acumulatorului atinge valoarea Presetare, bitul DN (terminat) merge TRUE. Observați că instrucțiunea NO în treapta secundă are bitul DN de la Timer T : 1 ca adresă. Prin urmare, va merge adevărat după 5 s, determinând instrucțiunea de ieșire la 0 : 2/4 pentru a porni o lumină.

În Figura 12.20, cronometrul acționează ca un releu de întârziere, deoarece pornește întârzierea atunci când treapta merge TRUE. De asemenea, sunt disponibile instrucțiuni de temporizare în afara întârzierii.

EXEMPLUL 12.4

Un proces de lot - care implică umplerea unei cuve cu un lichid, amestecarea lichidului și scurgerea cuvei - este automatizat cu un PLC. Figura 12.21 litera (a) prezintă hardware-ul. Secvența specifică de evenimente este după cum urmează: Când butonul de start este apăsat, procesului este pornit:

1. O supapă de umplere se deschide și permite unui lichid într-o cuvă până când este plin.
2. Lichidul din cuvă este amestecat timp de 3 minute.
3. O supapă de scurgere se deschide și drenează rezervorul.

Desenați diagrama scării pentru programul PLC.

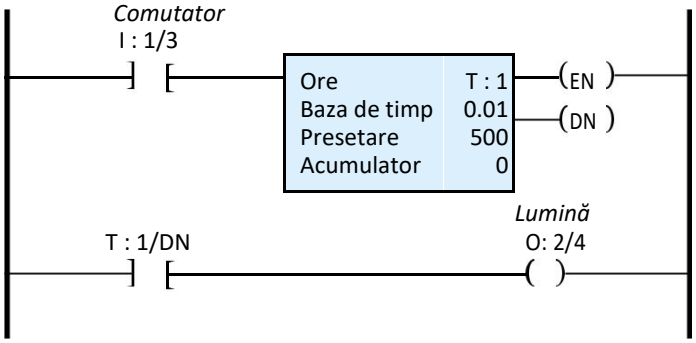


Figura 12.20
Instrucțiunea cronometrului.

T : 1	Adresa cronometrului (de fapt, prima dintre cele trei adrese în RAM) unde prima adresă deține biții de stare EN, TT și DN; cel a doua adresă deține valoarea presetată; și a treia adresă deține valoarea acumulatorului.
Baza de timp	Valoarea 0,01 înseamnă că fiecare număr corespunde lui 0,01 Secunde.
Presetare	Valoarea de 500 înseamnă că întârzierea va dura 500 contează, care în acest caz este de 5 s (0.01s × 500 = 5 s).
Acumulator	Deține valoarea numărului curent.
EN	Un pic care este adevărat, atâta timp cât rulează cronometrul este ADEVĂRAT.
TT	Un bit care este adevărat, atâta timp cât cronometrul este pe numărare, care este, adevărat pentru întârzierea de timp.
DN	Un bit care are valoarea adevărat atunci când cronometrul este făcut-cu alte cuvinte, când numărul ajunge la valoarea presetată.

SOLUȚIE

Figura 12.21 litera (b) prezintă diagrama scării completate, care este explicată treaptă cu treaptă:

1. Treaptă 1 activează supapa de umplere și va merge TRUE (și sigiliu) atunci când comutatorul de pornire de control local este activat momentan *și* rezervorul este mai mic decât plin. Comutatorul fizic de NO activat de flotor (comutator complet) conduce instrucțiunea *completă SW* NC în treapta 1; prin urmare, instrucțiunea NC va fi TRUE atâta timp cât cuva nu este plină (se schimbă în FALSE atunci când cuva este plină). Atâta timp cât întreaga treapta este TRUE, instrucțiunea de ieșire (*Supapa de umplere*) va fi TRUE, care pornește supapa de umplere reală. (În timpul perioadei de umplere, niciuna dintre celelalte trepte nu este TRUE.)

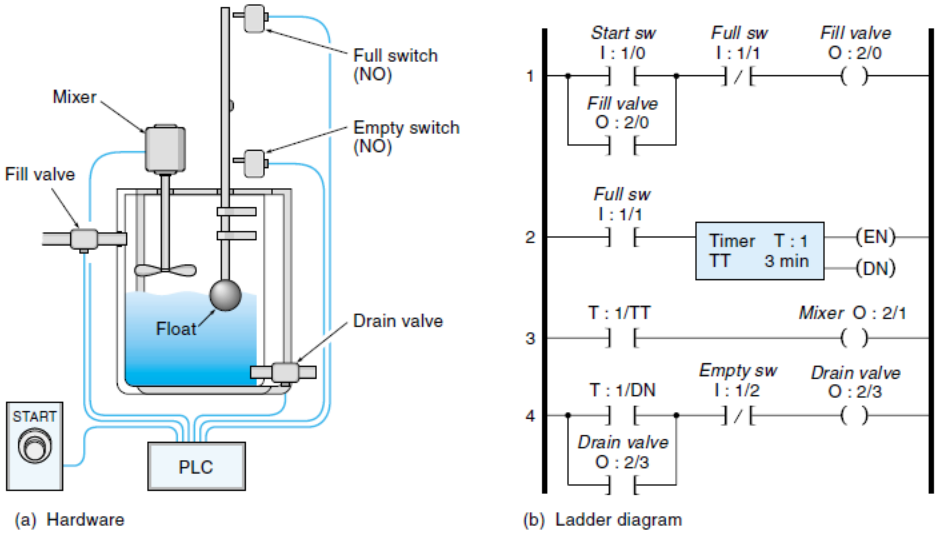


Figura 12.21

Folosind o instrucțiune de cronometru PLC pentru a controla o operațiune de amestecare în serie.

i

2. Treaptă 2 activează cronometrul de amestecare de 3 minute. Când cuva este plină, comutatorul fizic complet se închide (merge la on), astfel încât instrucțiunea NO (*Full sw*) merge TRUE, care pornește Timer T : 1 (care a fost programat să ofere timpul de amestecare de 3 minute).

3. Treaptă 3 controlează motorul mixerului. Atâta timp cât bitul TT al cronometrului este TRUE, instrucțiunea de ieșire *Mixer* va fi TRUE, care activează motorul mixerului.

4. Treaptă 4 activează supapa de scurgere. După 3 min, cronometrul "dispare", iar bitul DN (T : 1/DN) merge TRUE, făcând prima instrucțiune din treapta 4 să meargă TRUE. Instrucțiunea NC (*Empty sw*) va fi, de asemenea, TRUE, deoarece vasul este încă plin (adică comutatorul *Empty* rămâne neactivat). Prin urmare, treapta 4 va fi continuă, determinând instrucțiunea de ieșire *Supapa de scurgere* să meargă TRUE, care deschide supapa de scurgere, permițând lichidului să plece. Iată unde devine interesant: De îndată ce lichidul începe să se scurgă, comutatorul complet se va dezactiva, făcând treapta 2 să meargă FALSE, ceea ce face ca bitul DN al cronometrului să meargă FALSE. Această din urmă acțiune va face ca prima instrucțiune din treapta 4 să meargă FALSE. Deci, pentru a menține treaptă 4 TRUE până când cuva s-a scurs complet, instrucțiunea T : 1 / DN are o instrucțiune paralelă de ramură, care este activată de *supapa* de scurgere în sine. De fapt, treapta 4 este blocată până când *Empty sw* activează și rupe sigiliul.

Contoare

O instrucțiune Counter ține evidența numărului de evenimente. Numărul ar putea reprezenta numărul de piese care urmează să fie încărcate într-o cutie sau de câte ori unele operațiuni se vor face într-o zi. Contoarele pot fi de tip de numărare în sus sau de numărare în jos. Instrucțiunea Counter este plasată într-o treaptă și va incrementa (sau decrementa) de fiecare dată când treaptă face o tranziție FALSE-la-TRUE. Numărul este reținut până când este activată o instrucțiune RESET (cu aceeași adresă ca contorul). Contorul are o valoare presetată asociată cu aceasta. Când numărul ajunge până la valoarea Presetare, rezultatul devine TRUE. Acest lucru permite programului să inițieze o acțiune bazată pe un anumit număr; de exemplu, după ce 50 de elemente sunt încărcate într-o casetă, o casetă nouă este mutată în poziție. Simbolul contorului poate fi o casetă sau un cerc, cu toate datele pertinente plasate în sau despre acesta, așa cum se arată și se explică în figura 12.22.

Diagrama scării din Figura 12.22 este un circuit simplu care contorizează de câte ori *switch-ul* (adresa I : 1/2) este deschis și închis. Contorul poate fi resetat la orice

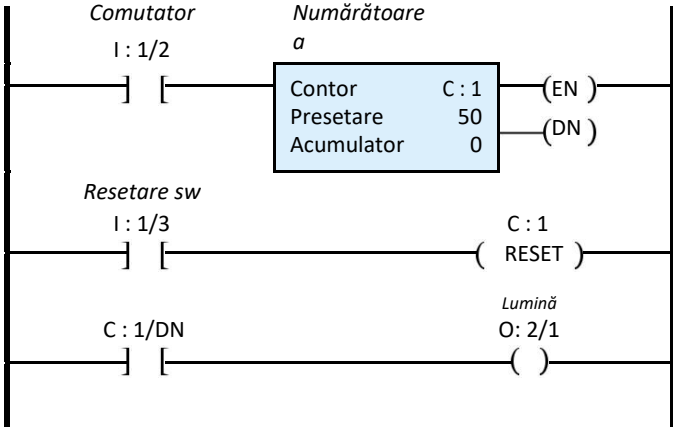


Figura 12.22
Instrucțiune a Counter.

C : 1	Adresa contorului (de fapt prima dintre cele trei adrese) unde prima adresă deține biți precum CU și DN; al doilea adresa păstrează valoarea presetată; iar a treia adresă deține valoarea acumulată.
Presetare	Când contorul ajunge la valoarea Presetare, face ca bitul DN să meargă ADEVĂRAT și continuă să numere.
Acumulator	Deține valoarea numărului curent.
EN	Merge adevărat atunci când treapta contra este adevărată.
DN	Rămâne adevărată atunci când numărul îndeplinește sau depășește valoarea PRESET.
RESET	O instrucțiune separată cu aceeași adresă de contor; atunci când acest bit merge TRUE, Acumulatorul se resetează la zero (resetează contorul).

timp prin închiderea *Reset sw,* deoarece această acțiune activează instrucțiunea RESET. A treia treaptă utilizează bitul DN de la Contor pentru a porni o lumină atunci când numărul ajunge la 50 sau mai mare.

Secvenţe

Instrucțiunea Sequencer este utilizată atunci când este necesară o secvență repetată de ieșiri (amintiți-vă exemplul unei mașini de spălat vase care trece prin pași predeterminați). În mod tradițional, secvențiatoarele electromecanice (Figura 12.10) au fost utilizate în acest tip de aplicație (unde un tambur se rotește încet, iar camele de pe tambur activează comutatoarele). Instrucțiunea Sequencer permite PLC-ului să implementeze această strategie comună de control.

Figura 12.23 prezintă instrucțiunea PLC Sequencer. Operarea este după cum urmează: Modelele de biți de ieșire dorite pentru fiecare pas sunt stocate (secvențial) ca cuvinte în memorie, ca de obicei, fiecare bit din cuvânt corespunde unui terminal specific al modulului de ieșire. De fiecare dată când pașii de instrucțiuni Sequencer, se conectează la următorul model de ieșire în memorie la modulul de ieșire desemnat. Adresa primului model (cuvânt) din secvență este dată în instrucțiunea Sequencer ca Sequencer File Adr (care este adresa B : 1 în Figura 12.23). Sequencer-ul afișat în figura 12.23 face pași atunci când treapta de execuție face o tranziție FALSE-la-TRUE. Alte versiuni ale instrucțiunii Sequencer permit programatorului să introducă o valoare de timp pentru fiecare pas.

EXEMPLUL 12.5

Un proces de spălare a pieselor necesită următoarea secvență:

1. Pulverizați apă și detergent timp de 2 minute (*Ciclu de spălare*).
2. Clătiți cu spray cu apă numai timp de 1 min (*Ciclu de clătire*).
3. Apa oprită, suflați aerul uscat timp de 3 min (*Ciclu de uscare*).

Secvența este începută cu un comutator de comutare. Desenați diagrama scării pentru acest proces (utilizând instrucțiunea Sequencer).

SOLUȚIE

Diagrama scării completate (Figura 12.24) constă din șase trepte. Treptele 1 și 2 generează întârzierea de 2 minute pentru *ciclul de spălare*. Treptele 3 și 4 generează întârzierea de 1 minut pentru *ciclul de clătire*, iar treapta 5 generează întârzierea de 3 minute pentru ciclul de *uscare*. Treaptă 6 are instrucțiunea Sequencer. Instrucțiunea Sequencer specifică faptul că fișierul secvență începe la adresa B : 1, că fișierul are trei cuvinte și că modulul de ieșire este în slotul 2. Uitându-ne la fișierul Sequence, vedem că bitul 0 controlează supapa de apă, bitul 1 controlează supapa de detergent, iar bitul 2 controlează ventilatorul uscat prin suflare.

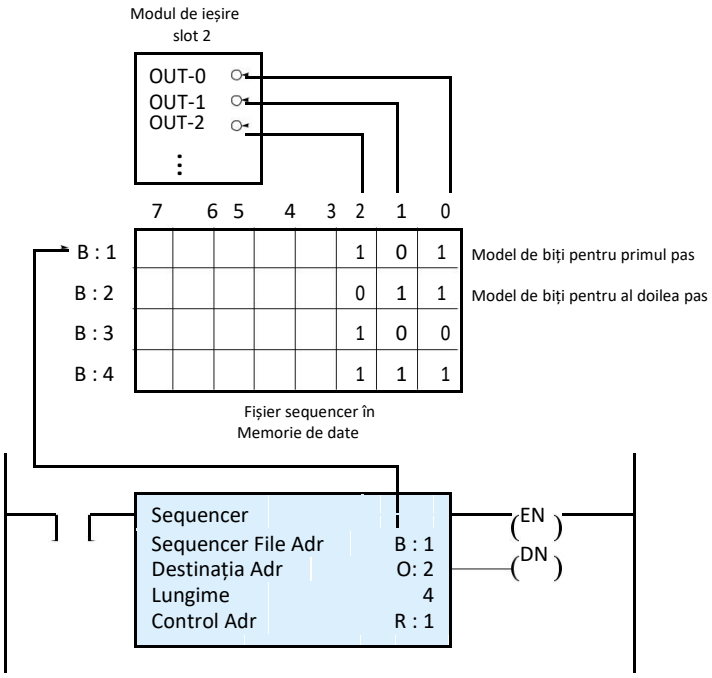


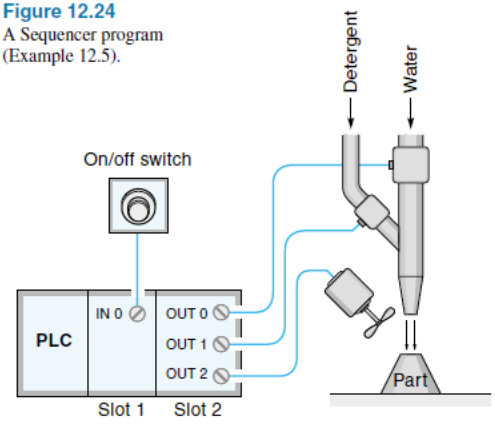
Figura 12.23
Instrucțiunea Sequencer.

Sequencer File Adr	Prima adresă a fișierului secvență, care stochează seturi de ieșiri.
Destinația Adr	Adresa de ieșire (sau slotul) la care fișierul Secvență cuvintele sunt transferate cu fiecare pas.
Lungime	Lungimea fișierului secvență, adică numărul de pașii din secvență.
ÎN	Merge TRUE atunci când treapta Sequencer este TRUE.
DN	Standuri pentru *făcut*-merge adevărat atunci când le-a operat pe ultimul cuvânt din fișierul Secvență.
Control Adr	Adresa care stochează biții de control (EN, DN) și cuvintele (lungimea) secvențiatorului.

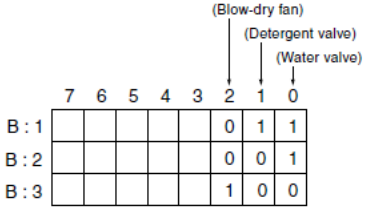
Prezentare generală a operațiunii

Programul este format din cinci trepte de cronometru și o un secvențiator. Cronometrele sunt activate, unul după altul, în josul programului - adică fiecare cronometru, când este termină numărarea, pornește cronometrul următor în linie. Ieșirile cronometrelor T : 1, T : 3 și T : 5 sunt utilizate pentru a pasa instrucțiunea Sequencer. Instrucțiunea Sequencer prezintă trei modele de biți de ieșire în modulul 2.

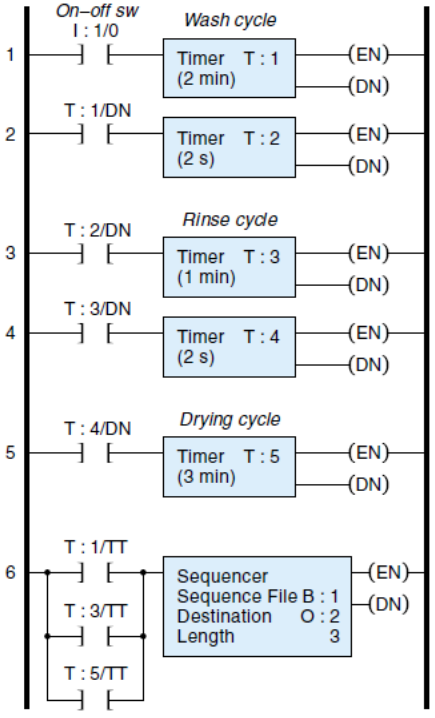
Figure 12.24
A Sequencer program (Example 12.5).



(a) Hardware



(b) Sequencer File



(c) Ladder diagram

Operare detaliată

Funcționarea programului scării este după cum urmează: Acțiunea începe pe treaptă 1 atunci când operatorul pornește comutatorul de pornire la pornire. Acest lucru oferă o tranziție FALSE-la-TRUE care pornește Timer T : 1*(Ciclu de spălare*), determinând bitul său de sincronizare (TT) să meargă TRUE timp de 2 min. Același bit de sincronizare (T : 1 /TT), face ca treapta 6 să meargă TRUE, pornind sequencerul la prima poziție (conectând cuvântul la adresa B: 1 la modulul de ieșire 0 : 2). Observați că bitul 0 și bitul 1 din cuvântul B : 1 sunt 1s, ceea ce face ca terminalele OUT-0 și OUT-1 să meargă mai departe, pornind supapele de apă și detergent.

După 2 min, bitul DN al cronometrului T : 1 (T : 1/DN) merge TRUE, care începe următorul cronometru (T : 2 în treapta 2). Acest cronometru are doar 2 s lungime, iar rolul său este de a crea un decalaj scurt între pași (pentru a permite resetarea treptei sequencerului).

Când cronometrul T : 2 se face, bitul său DN (T: 2/DN) pornește cronometrul de 1 min T: 3*(Ciclu de clătire).* Cronometrul T : 3 biți de sincronizare (T : 3/TT) determină instrucțiunea Sequencer să avanseze la a doua etapă (B : 2). Observați că în cuvântul B : 2, doar bitul 0 este un 1, determinând OUT-0 să rămână pornit (lăsând apa pornită, dar opriți detergentul).

Urmând în același mod, Cronometrul T : 4 oferă un decalaj scurt, iar apoi Cronometrul T : 5 avansează sequencerul în a treia și ultima poziție (B: 3) pentru ciclul de *uscare* de 3 minute. Observați că bit 2 din cuvântul B: 3 devine OUT-2, și se aprinde indicatorul blow-dry.

Există multe modalități posibile de a programa această secvență. Metoda aleasă (Figura 12.24) nu este neapărat cea mai scurtă, dar este o soluție clară, necompusă, care în viața reală este mai valoroasă decât un program scurt inteligent (pe care nimeni nu îl poate înțelege în afară de programator!). De asemenea, amintiți-vă că treptele suplimentare pe un program de scară PLC nu înseamnă mai mult hardware (înseamnă un timp de scanare puțin mai lung și o utilizare a memoriei, ceea ce, în majoritatea cazurilor, nu este o problemă).

Utilizarea unui PLC ca un controler în două puncte

Exemplul 12.6 arată modul în care un PLC poate fi utilizat pentru a implementa o strategie de control care necesită decizii bazate pe valori analogice. Deși este un exemplu simplu, demonstrează utilizarea de intrări analogice și instrucțiuni care se ocupă cu valori analogice.

EXEMPLUL 12.6

Temperatura într-un cuptor electric trebuie menținută de un PLC pe 16 biți la aproximativ 100 °C, folosind controlul în două puncte (intervalul real: 98-102°). Figura 12.25 litera (a) prezintă hardware-ul sistemului: un cuptor cu un element electric de încălzire acționat de un *contactor* (releu de curent înalt), un senzor de temperatură LM35 (produce 10 mV/°C — a se vedea capitolul 6), un comutator de pornire al cuptorului și PLC-ul. PLC-ul are un procesor și trei module I/O: un modul de intrare discret (slotul 1), un modul de intrare analogic pe 16 biți (slotul 2) și un modul de ieșire discret (slotul 3).

Desenați diagrama scării pentru acest sistem.

SOLUȚIE

Când cuptorul se va opri sub 98°, elementul de încălzire se va porni și va rămâne pornit până când

temperatura ajunge la 102°. Astfel, PLC-ul va acționa o ieșire discretă (încălzitorului), pe baza unei intrări analogice (temperatură).

Diagrama scării [Figura 12.25(b)] constă din trei trepte care implică logica de control; cu toate acestea, amintiți-vă că *înainte* ca PLC-ul să execute treptele, se citesc toate datele modulului de intrare. În acest caz, se va citi starea comutatorului de pornire de la modulul de intrare din slotul 1 (stocarea acestuia în memorie la adresa I : 1/0) și va citi în valoarea temperaturii prin modulul de intrare analogic în slotul 2. Modulul analogic conține un ADC care convertește tensiunea analogică într-un cuvânt pe 16 biți. Rezoluția modulului este dată ca

$$\text{Rezoluție ADC} = 0{,}305176 \text{ mV} = 1 \text{ LSB}^*$$

Folosind această rezoluție, putem găsi echivalentul digital (în binar) al temperaturilor limită inferioare și superioare, 98° și 102°:

Tensiune de intrare analogică pentru

$$98° = 98° \times \frac{10 \text{ mV}}{1°\text{C}} = \underbrace{980 \text{ mV}}_{\text{Output of LM35}}$$

Ieșire ADC pentru

$$98° = 980 \text{ mV} \times \frac{\text{LSB}}{0.305176 \text{ mV}} = 3211 \text{ LSB} = 110010001011$$

Tensiune de intrare analogică pentru

$$102° = 102° \times \frac{10 \text{ mV}}{1°\text{C}} = \underbrace{1020 \text{ mV}}_{\text{Output of LM35}}$$

Ieșire ADC pentru

$$102° = 1020 \text{ mV} \times \frac{\text{LSB}}{0.305176 \text{ mV}} = 3342 \text{ LSB}$$

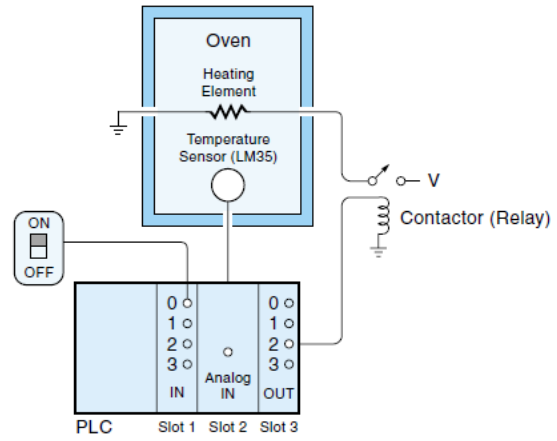
$$= 110100001110$$

Acum cunoaștem valorile digitale ale temperaturilor cu limită superioară și inferioară. Vom avea nevoie de aceste numere în program. Funcționarea programului ladder urmează [a se vedea figura 12.25 litera (b)]:

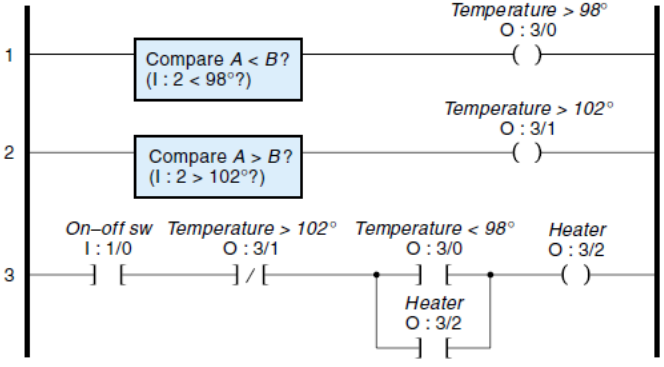
1. Treapta 1 determină dacă temperatura este sub 98 ° C și va merge TRUE atunci când sunt îndeplinite condițiile instrucțiunii de comparare. Acest instrucțiune specială Compara

*Acest raport se regăsește pe fișa de specificații și provine din următoarea analiză: Intervalul de tensiune de intrare = ±10 V = 20 V; ieșirea este de 16 biți = 65.535 stări (2s complement); LSB = 20 V/65.535 = 0,305176 mV.

Figura 12.25
Un program de controler în două puncte (Exemplul 12.6).



(a) Hardware



(b) Ladder diagram

compară două cuvinte pe 16 biți și atribuie valoarea TRUE dacă valoarea cuvântului A este mai mică decât cuvântul B. În acest caz, cuvântul A este temperatura reală (așa cum este găsită de ADC și stocată la adresa I: 2), iar cuvântul B este echivalentul binar al temperaturii limită inferioare de 98°. Prin urmare, tresa va merge TRUE dacă temperatura cuptorului este mai mică de 98°. Valoarea acestei trepte este stocată în bitul 0 : 3/0.

2. Treapta 2 determină dacă temperatura este mai mare de 102 °C. Aceasta treaptă are o altă instrucțiune compara, dar aceasta atribuie TRUE dacă cuvântul A este mai mare decât cuvântul B. Observați că cuvântul A este din nou temperatura reală (de la ADC), iar cuvântul B este temperatura limită superioară de 102 °. Prin urmare, treapta va merge TRUE dacă temperatura cuptorului este mai mare de 102 °. Valoarea acestei trepte este stocată în bitul 0 : 3/1.
3. Treaptă 3 activează elementul de încălzire prin logica de control care pornește și oprește elementul de încălzire - adică *încălzitorul cu* instrucțiuni de ieșire (0: 3/2) controlează direct elementul de încălzire. Treapta va deveni TRUE d acă comutatorul de pornire este pornit, *iar* temperatura nu este mai mare de 102° (observați că aceasta este o instrucțiune NC), *iar* temperatura este mai mică de 98 °. Odată ce *încălzitorul* este pornit, acesta este sigilat cu ramura paralelă 0: 3/2, astfel încât, chiar și atunci când cuptorul se ridică peste 98°, treapta va rămâne TRUE. Odată ce elementul de încălzire este ON, temperatura cuptorului va crește în cele din urmă la peste 102 °, și astfel instrucțiunea NC (Temperatura >*102 °)* va trece în FALSE, ruperea sigiliului și oprirea *încălzitorului.* Treapta va rămâne FALSA până când temperatura scade sub 98°, iar apoi ciclul începe.

Instrucțiuni avansate

Până în prezent, tocmai am examinat instrucțiunile de bază susținute de aproape orice PLC. PLC-urile devin din ce în ce mai sofisticate și, pe măsură ce o fac, seturile lor de instrucțiuni se extind. De fapt, multe PLC-uri mai noi seamănă mai mult cu un computer de control al procesului în timp real decât pur și simplu un substitut pentru logica releului. Multe dintre instrucțiunile avansate funcționează pe un cuvânt digital în loc de un singur bit. Unele dintre aceste cuvinte pot fi date analogice digitalizate (cum ar fi temperatura) utilizate cu un modul I/O analogic, care este un ADC sau un DAC. Valorile de acumulare a contorului și cronometrul sunt stocate sub formă de cuvinte. În multe cazuri, scopul procesării acestor cantități numerice este de a ajunge la o decizie ***da-nu***, în cazul în care ieșirea reală a programului este încă în termeni de biți individuali (adică pornirea sau oprirea unui motor). Apoi, de asemenea, este posibil să se furnizeze un cuvânt digital întreg pentru a fi convertit la o tensiune analogică de către un modul I /O. O astfel de tensiune ar putea fi utilizată pentru a controla viteza unui motor.

Instrucțiunile de nivel superior vin în diferite categorii:

- **Instrucțiunile de comparație** compară două cuvinte și dau un singur bit ca rezultat. De exemplu, un cuvânt poate fi testat pentru a vedea dacă este mai mare decât un alt cuvânt sau pentru a vedea dacă este egal cu un alt cuvânt. Aceste instrucțiuni sunt demonstrate în exemplul 12.6.

- **Instrucțiunile matematice** efectuează operații matematice asupra cuvintelor din memorie. Aceste operațiuni pot include adunare, scădere, înmulțire, împărțire și altele. O instrucțiune matematică ar putea fi utilizată pentru a multiplica un cuvânt de intrare a senzorului printr-o constantă, în scopuri de scalare.
- **Instrucțiunile logice și shift** efectuează operații logice, cum ar fi AND, OR și NOT și SHIFT-uri la dreapta și la stânga pe cuvintele stocate în memorie. De exemplu, instrucțiunea AND ar putea fi folosită pentru a masca anumiți biți într-un cuvânt.
- **Instrucțiunile de control** permit lucruri precum salturi și subrutine. De exemplu, o instrucțiune Salt, atunci când TRUE, va determina execuția să sară înainte (sau înapoi) la orice treaptă desemnată.
- **Controlul PID** este disponibil pe unele PLC-uri. În unele cazuri, este construit într-un modul I/O; în alte cazuri, este programat ca o instrucțiune în diagrama scării. Instrucțiunea PID este acoperită în secțiunea următoare.
- **Instrucțiunile de mesaje și comunicații I/O** permit PLC-urilor să trimită și să primească mesaje și date către alte PLC-uri sau un terminal (PC), pentru acele aplicații în care PLC-urile sunt în rețea împreună.

Instrucțiunea PID

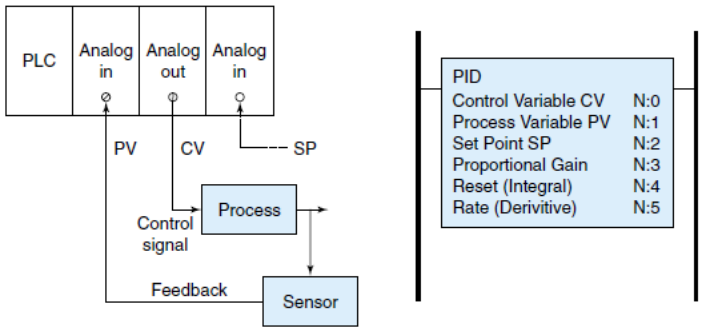
Controlul PID devine din ce în ce mai frecvent disponibilă pe PLC-uri. În majoritatea cazurilor, funcția PID este implementată printr-o instrucțiune și utilizează canale I/O (analogice) obișnuite. O altă posibilitate este un modul hardware PID separat, caz în care este de fapt un controler care operează independent și care rulează sub controlul de supraveghere al PLC-ului.

Instrucțiunea PID pune în aplicare strategia de control PID discutată în capitolul 11. Acesta ar fi utilizat într-un sistem cu buclă închisă pentru a controla un anumit parametru fizic, cum ar fi temperatura, nivelul lichidului sau chiar mișcarea (deși o problemă de mișcare poate necesita răspunsuri mai rapide decât poate oferi PLC-ul).

Un eșantion de instrucțiuni PID este prezentat în figura 12.26 (b). Se compune dintr-o cutie sau cerc cu informațiile de setare plasate în sau în jurul ei. Este, de obicei, pe o treaptă de la sine (nu există nici o logică condiționată pe partea stângă a treptei), ceea ce înseamnă că instrucțiunea PID este întotdeauna ADEVĂRATĂ. Astfel, atâta timp cât PLC-ul rulează, funcția PID va încerca să mențină un anumit parametru fizic la punctul stabilit. Hardware-ul pentru a sprijini instrucțiunea PID este prezentat în Figura 12.26 (a) și constă din

- Un canal analogic I/O pentru *CV-ul* semnalului de ieșire (variabila de control)
- Un canal I/O de intrare (analog) pentru semnalul de feedback *PV* (variabila de proces)
- Eventual un canal I/O de intrare (analog) pentru punctul setat *SP* (Cu toate acestea, SP nu are nevoie neapărat de o intrare fizică, deoarece poate fi setat de o altă instrucțiune sau este pur și simplu o constantă.)

Simbolul instrucțiunii specifică adresele din MEMORIA RAM ale parametrilor PID. După cum se arată în figura 12.26 (b), aceste locații de memorie stochează (cel puțin) valorile actuale ale *CV, PV* și *SP,* precum și constantele PID proporționale, integrale și derivative (K_P, K_I^*, K_D^*).



(a) Hardware (b) PID Instruction in ladder diagram

Reglarea controlerului PID înseamnă găsirea valorilor K_P, K_I și K_D care determină sistemul să rămână stabil, să răspundă rapid la perturbări fără depășiri excesive și să aibă un minim de eroare în stare stabilă. Metoda de reglare sugerată de Allen-Bradley este similară cu metoda ciclului continuu discutată în capitolul 11 (a se vedea exemplul 11.6). Această metodă este după cum urmează:

1. Inițial setul $K_P = 1$, $K_I = 0$, $K_D = 0$
2. Creșteți încet câștigul (K_P) până când sistemul intră într-o oscilație constantă și înregistrați perioada de oscilație (perioadă = T_C)
3. Setări inițiale: setați K_P la jumătate din ceea ce era necesar pentru a intra în oscilație, setați $K_I = 1/T_C$ și setați $K_D = T_C / 8$.

Alte limbaje de programare PLC

Programele de diagramă a scărilor sunt extrem de simbolice și sunt rezultatul anilor de evoluție a diagramelor circuitelor de control industrial. PLC-ul a venit relativ târziu și a fost adaptat pentru a utiliza logica tradițională a scării. Cu toate acestea, după cum am observat, un program PLC nu este de fapt o diagramă de cablare, ci o modalitate de a descrie relația logică dintre intrări și ieșiri. Există acum modalități mai concise de a transmite logica de control decât diagramele ladder. Aceste programe arată mai mult ca programe de calculator convenționale, folosind cuvinte și simboluri matematice în loc de imagini.

Un tip de limbaj PLC folosit de Allen-Bradley se numește pur și simplu *programare ASCII* sau "crearea unui fișier arhivă ASCII". Practic, aceasta este o metodă de a transmite o diagramă ladder folosind cuvinte în loc de imagini. Fiecare instrucțiune primește o denumire de trei litere numită **mnemonic.** Tabelul 12.3 enumeră instrucțiunile mai frecvente. În tabelul 12.3, primele trei mnemonice reprezintă dispozitive precum comutatoare și relee, iar ultimele trei mnemonice specifică modul în care dispozitivele sunt interconectate logic. Într-un ASCII

TABELUL 12.3

Instrucțiuni de programare ASCII

Simbol	Mnemonic	Explicație
—] [—	XIC (examinați dacă este închis)	Devine TRUE atunci când contactele se închid.
—] / [—	XIO (examinați dacă este deschis)	Devine TRUE atunci când contactele sunt deschise.
—( )—	OTE (energie de ieșire)	Activează releul atunci când este ADEVĂRAT.
——¡ ——	BST (ramură start)	Începe o ramură paralelă.
¦_ -¦	NXB (următoarea ramură)	Se termină ramura anterioară și începe o nouă ramură.
———— ¡	BND (capete de ramură)	Se termină ramurile paralele.

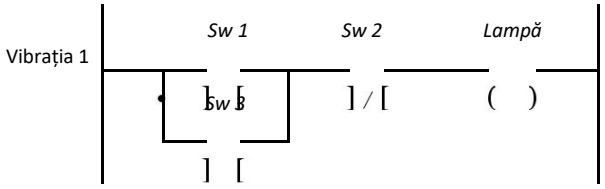


Figura 12.27
O diagramă de scară și programul ASCII corespunzător.

Instrucțiuni de program (ASCII)	Explicație
! TREAPTĂ 1	Identifică treapta.
SOR BST XIC Sw 1 NXB	Pornirea treptei; ramură începe și XIC Sw 1 este pe ramura de sus, urmează următoarea ramură.
XIC Sw 3 BND	Pe ramura următoare este XIC Sw 3, și apoi acest lucru capetele ramificate.
XIO Sw 2 OTE lampă	Pe trepta principală este XIO Sw 2, apoi un releu conducerea unei lămpi.
EOR	Această treaptă este terminată.

Instrucțiuni de program (boolean)	Explicație
START SW 1	Comutatorul 1 este prima variabilă logică.
SAU Sw 3	Comutatorul 3 este în paralel cu switch-ul 1 (care este la fel ca o operațiune OR).
ȘI	Rezultatul anterior va fi AND cu
NU Sw 2	inversa stării switch-ului 2.
AFAR	
Ă Lampă	Rezultatul ecuației logice este utilizat pentru a activa

program, diagrama scara este transmisă prin plasarea mnemonics și adrese într-o anumită ordine. Aceste concepte sunt cel mai bine prezentate de Figura 12.27, o diagramă simplă a scării urmată de programul ASCII corespunzător.

Un alt tip de limbaj de programare PLC utilizează logica booleană direct. Acest lucru implică listarea comenzilor, cum ar fi AND, OR, și NU în așa fel încât să descrie logica programului. Un exemplu al acestei abordări este prezentat în figura 11.27. Încă altele PLCs pot fi programate cu limbaje de nivel înalt, cum ar fi de bază sau C.

12.4 PLC-URI ȘI REȚELE

Rețelele sunt utilizate din ce în ce mai mult pentru a interconecta componentele sistemelor de control între ele și restul fabricii. Din punct de vedere fizic, o rețea este un fir care acționează ca o "autostradă electronică" care poate transmite mesaje între **noduri** (PC-uri și alte dispozitive electronic). Fiecare nod din rețea are o adresă unică, iar fiecare mesaj, numit pachet de **date,** include adresa de unde merge și de unde a venit. Toate datele din rețea sunt trimise serial (câte un pic) pe un singur fir. Cel mai comun tip de rețea utilizează **topologia magistralei,** ceea ce înseamnă că toate nodurile se conectează la un singur cablu (a se vedea figura 12.31). Rețeaua de tip magistrală este similară cu o sală lungă dintr-un hotel, cu uși de cameră pe fiecare parte, așa cum este ilustrat în Figura 12.28. Fiecare cameră are un număr de cameră (adresă), iar un mesaj ar putea fi transmis dintr-o cameră în alta prin hol. De exemplu, dacă persoana din camera 3 dorea să trimită un mesaj persoanei din camera 6, aceasta deschidea ușa și striga: "Am un mesaj pentru camera 6 din camera 3. Mesajul este bla, bla, bla"." Toți ceilalți ocupanți ascultau la ușile lor, dar numai ocupantul camerei 6 lua nota și primea mesajul. Pe lângă transmiterea conceptului general, această analogie ilustrează alte două puncte despre un tip de autobuz de

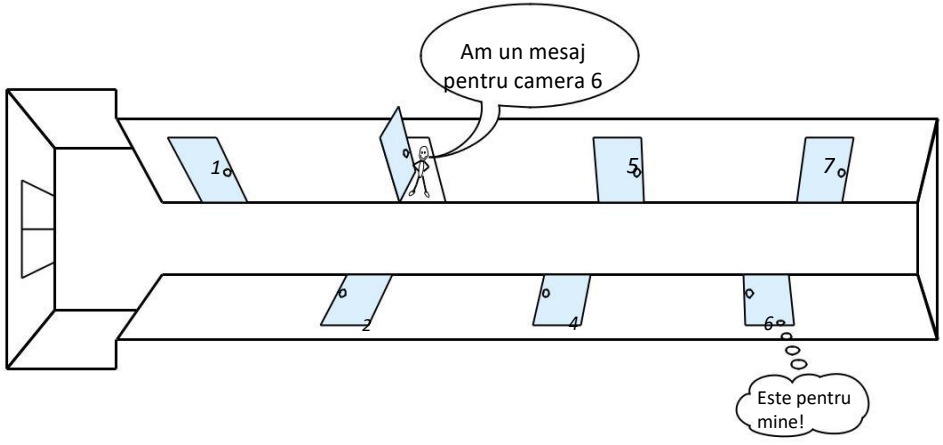


Figura 12.28
Analogie pentru un bus de rețea: a hol lung într-un hotel, în cazul în care fiecare cameră este un nod.

rețea: Fiecare mesaj este transmis pe fiecare ușă și un singur mesaj poate fi comunicat la un moment dat.*

Există trei niveluri de rețele care ar putea fi utilizate de organizațiile industriale (a se vedea figura 12.29) și toate cele trei niveluri pot fi interconectate, astfel încât orice nod să poată comunica cu orice alt nod din clădire. Cel mai înalt nivel este Ethernet. **Ethernet** este rețeaua tradițională utilizată pentru interconectarea PC-urilor și computerelor mainframe în scopul schimbului de date și fișiere de date; acesta este cunoscut sub numele de nivel de *informații.* Ethernet este bun pentru comunicarea datelor financiare, de inventariere și de producție, dar poate avea dificultăți când se ocupă de un proces în timp real, cum ar fi un PLC care răspunde rapid la o schimbare a vitezei motorului. Problema este că Ethernet nu este o rețea deterministă. O **rețea deterministică** este capabilă să atribuie prioritate unui mesaj și să garanteze că va ajunge într-un anumit timp. Ethernet poate încetini semnificativ în perioade de trafic intens.

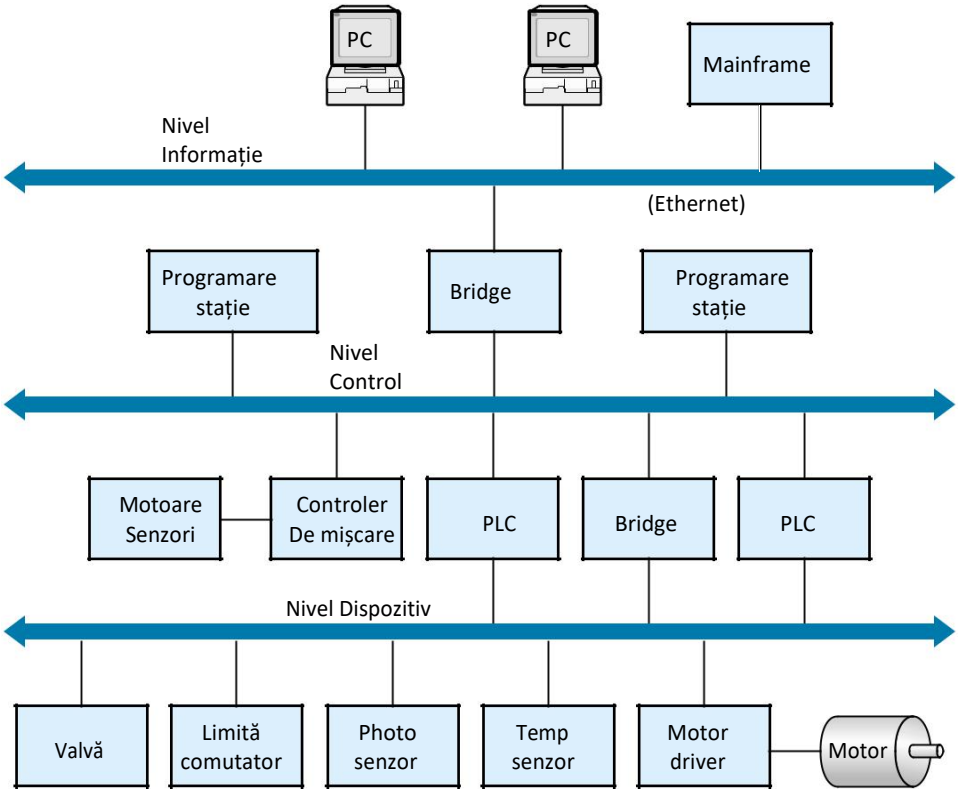


Figura 12.29
Trei niveluri de rețele.

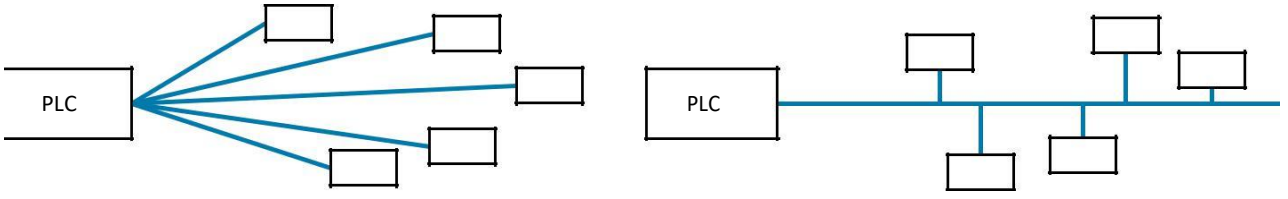
*Constanta integrală K_I este de obicei numită Reset =1 /T_C , iar constanta derivată K_D este de obicei numită *Rata* = T_D

*Acest lucru este valabil pentru majoritatea rețelelor care utilizează un semnal digital în bandă de *bază* (fără multiplexare de frecvență).

Al doilea nivel de rețea este cunoscut sub numele de nivel de *control.* O rețea la nivel de control oferă o comunicare deterministă și ar fi utilizată pentru a interconecta PLC-urile cu alte PLC-uri și cu computerele lor de supraveghere și stațiile de programare (pc-uri tipice). O rețea la nivel de control poate utiliza un sistem *token* * pentru a garanta că fiecare nod care trebuie să transmită va avea posibilitatea de a face acest lucru. Un exemplu al acestui tip de rețea este Control Net (Allen-Bradley).

Al treilea nivel de rețea este cunoscut sub numele de nivel de *dispozitiv.* O **rețea la nivel de dispozitiv** este utilizată pentru a conecta dispozitive de nivel scăzut, cum ar fi senzori și actuatori, la un PLC. În mod tradițional, senzorii individuali sunt conectați direct la PLC și ar putea părea "nejustificată" să utilizați o rețea pentru acest loc de muncă, dar există trei motive bune pentru utilizarea unei rețele:

1. O rețea de dispozitive simplifică cablarea. Figura 12.30 compară cablarea tradițională a senzorilor punct-la-punct cu rețeaua dispozitivului. În mod clar rețeaua este un sistem mai simplu care utilizează mai puțin fir. De fapt, multe mașini noi folosesc o rețea de dispozitive pentru a conecta dispozitive electrice (încuietori, comenzi și altele asemenea); acest lucru reduce cantitatea de cabluri necesare.
2. Cu o rețea, datele senzorului ajung într-o formă mai bună. În sistemul tradițional punct-la-punct, o tensiune analogică de nivel scăzut poate fi nevoită să călătorească prin mai multe noduri. Semnalul este supus atenuării și zgomotului. Chiar și sistemul de buclă curentă este supus zgomotului și altor pierderi. Cu o rețea de dispozitive, tensiunea analogică este convertită în digital *la sursă* și transmisă ca mesaj digital cu detectarea erorilor. Nu există nicio șansă ca semnalul să fie atenuat (atâta timp cât datele digitale trec), iar orice date corupte pot fi identificate ca atare și re-transmise.
3. Dispozitivele de rețea tind să fie mai "inteligente". Odată ce un dispozitiv are electronice încorporate pentru a comunica în rețea, este un pas mic pentru a construi mai multe funcționalități cu costuri suplimentare mici. De exemplu, o celulă foto ar putea trimite un mesaj spunând că nivelul de lumină s-a diminuat (indicând faptul că obiectivul se poate murdări sau că cineva l-a lovit din poziție).



a) PLC conectat la senzori și actuatori folosind cabluri punct-la-punct

(b) PLC conectat la senzori și actuatori utilizând o rețea la nivel de dispozitiv

Figura 12.30
Conectarea unui PLC la dispozitive.

* Un simbol este un pachet de date mic, care este transmis de la nod la nod. Există un singur token, iar un nod poate transmite o bucată de date numai dacă are simbolul. Acest sistem garantează că fiecare nod primește un viraj pentru a trimite date.

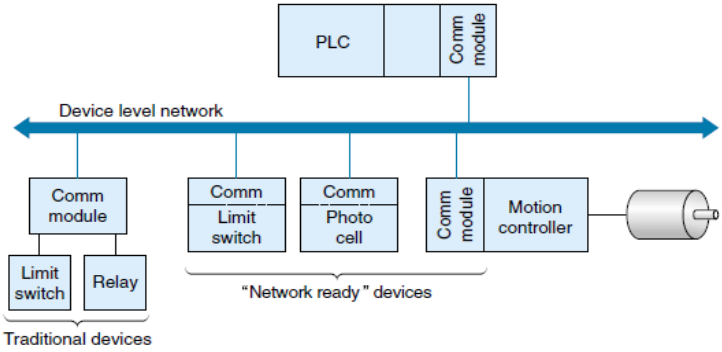


Figura 12.31
Fiecare dispozitiv conectat la rețea trebuie să utilizeze un circuit de interfață de rețea (modul de comunicare).

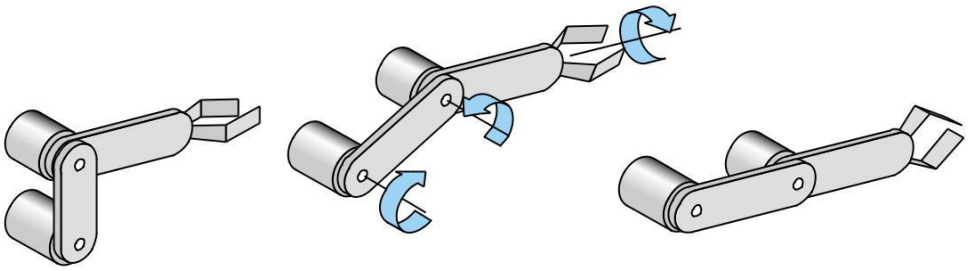
O rețea la nivel de dispozitiv poate fi capabilă să comunice în moduri diferite, în funcție de nevoile aplicației. Probabil cel mai comun sistem este sondaj I / O. Un exemplu de **I/O chestionat ciclic** apare atunci când un PLC solicită (sondaje) un senzor pentru date numai atunci când are nevoie de el. Într-un sistem **de schimbare a stării,** un senzor ar iniția un transfer de date numai atunci când a existat o modificare de raportat. De exemplu, un comutator de pe o ușă a cuptorului ar raporta numai atunci când ușa a fost de fapt deschisă. Într-un sistem *ciclic,* senzorul ar trimite datele sale la o rată predefinită, cum ar fi o dată la fiecare 100 ms. În cele din urmă, într-un sistem *de stroboscop,* un mesaj este livrat la mai multe dispozitive simultan.

Sunt necesare mai multe echipamente și etape procedurale pentru configurarea unei rețele de dispozitive și pentru punerea în funcțiune a acesteia, în comparație cu sistemul tradițional de cablare din figura 12.30 litera (a). În primul rând, fiecare dispozitiv trebuie să se conecteze la rețea printr-o unitate de interfață (a se vedea figura 12.31). Dispozitivele tradiționale simple, cum ar fi comutatoarele limită, fotocelulele și senzorii de temperatură, trebuie să se conecteze la rețea printr-un modul de comunicare. Modulul de comunicare permite operatorului să seteze adresa dispozitivului, probabil cu un comutator de degetul mare. Dispozitivele mai sofisticate, cum ar fi controlerele de acționare a motorului și alte dispozitive "gata de rețea", au circuitele de comunicare în rețea încorporate. De asemenea, pentru a face sistemul să funcționeze, programul PLC poate fi necesar să includă instrucțiuni pentru a trimite comenzi de inițializare la pornire acelor dispozitive care au nevoie de el și trebuie să fie programat să comunice cu dispozitivele sale prin rețea. Exemple de rețele la nivel de dispozitiv includ **DeviceNet** (Allen-Bradley) și **Fieldbus**.

12.5 CONTROLERE DE MIȘCARE

Controlerele de mișcare sunt circuite de control special realizate pentru a coordona mișcarea. O aplicație tipică ar fi un robot care ridică un obiect dintr-un loc, se deplasează într-un alt loc și pune obiectul în jos. Astfel cum se arată în figura 12.32, această acțiune ar putea implica

Figura 12.32
În acest exemplu de control al mișcării, trei axe se mișcă în același timp.

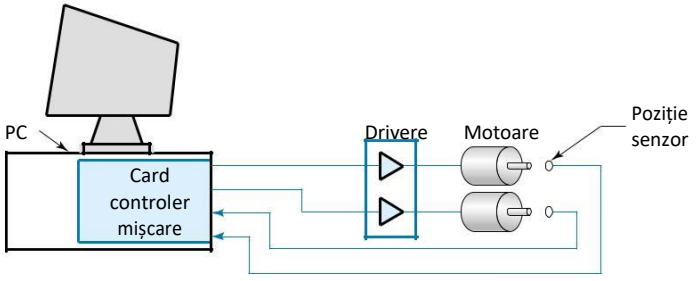


(a) Poziția de pornire (b) În mișcare (c) Poziția de sosire (trei axe în mișcare)

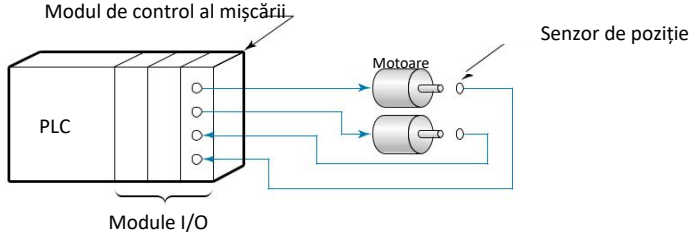
deplasarea a două sau mai multe axe în același timp. Motorul pentru fiecare axă ar accelera până la viteză, s-ar deplasa cu o viteză fixă până în vecinătatea destinației și apoi ar decelera până la o oprire. Vitezele ambelor motoare ar fi controlate astfel încât ambele să ajungă la destinație în același timp , chiar dacă asta a însemnat că s-au mișcat la viteze diferite.

Un controler de mișcare este practic un controler digital programabil similar cu cele deja discutate. Un controler tipic este construit pe o placă de circuit și poate fi capabil să realizeze un control de la 4 la 8 axe. Controlerul poate fi în buclă deschisă sau în buclă închisă. Sistemele în buclă deschisă trebuie să utilizeze motoare pas cu pas. Sistemele cu buclă închisă utilizează servomotoare DC sau AC și necesită senzori de poziție de feedback, care sunt de obicei codificatoare optice. Ieșirile de pe cardul controlerului de mișcare furnizează semnalul de turație sau cuplu motoarelor (sau unităților motorului); intrările la controler provin de la senzori. Două posibile configurații hardware sunt prezentate în Figura 12.33. Sistemul prezentat în figura 12.33 litera (a) este un PC cu

Figura 12.33
Hardware-ul controlerului de mișcare.



(a) Controler de mișcare pe un card de expansiune PC



(b) Controlerul de mișcare este un modul PLC

o cartelă de expansiune a controlerului de mișcare. Cardul poate fi programat cu software special care rulează sub Windows. Într-o altă configurație, prezentată în Figura 12.33 (b), controlerul de mișcare este un modul care rulează sub un PLC. În acest caz, PLC-ul ar încărca parametrii de mișcare în controlerul de mișcare, iar controlerul de mișcare ar funcționa apoi independent.

Ca aproape toate controlerele digitale, un controler de mișcare este practic un computer care execută un program. Ceea ce îl face un "controler de mișcare" este că hardware-ul și soft-ware-ul au fost adaptate special pentru a facilita controlul *mișcării.* Un procesor dintr-un controler de mișcare ar putea fi fie un microcontroler de uz general, fie un procesor de semnal digital (DSP). Un **DSP** este un procesor care este proiectat pentru a suporta sarcini de înaltă performanță, repetitive, intensive numeric. De exemplu, DSP-urile pot fi capabile să facă două operații multiple în același timp, pot efectua bucle strânse și rapide și pot avea seturi de instrucțiuni speciale care permit codificarea mai multor instrucțiuni în aceeași instrucțiune. Cu alte cuvinte, un DSP este bun pentru operațiunile de control în care sunt necesare operațiuni pe mai multe axe și răspuns rapid. Cu toate acestea, indiferent de tipul de procesor utilizat de controlerul de mișcare, configurarea și funcționarea de bază sunt aceleași:

1. Prin intermediul unui PC și a unui software proprietar furnizat sau specificat de furnizorul controlerului de mișcare, se introduc o serie de comenzi care descriu mișcarea dorită.
2. Comenzile de mișcare sunt procesate de software în "limbaj de mașină" și descărcate în memorie pe cardul procesorului de mișcare.
3. Când este activat, procesorul de pe cardul controlerului de mișcare execută bucla programului în limbaj de mașină . Este acest program care preia sarcina de a pro-cedarea semnalelor de feedback și trimiterea semnalelor de control al motorului către motor.

Controlerul de mișcare poate aplica mai multe strategii de control într-o secvență atunci când se deplasează la țintă. O mișcare tipică ar începe prin accelerarea la o viteză desemnată, apoi se deplasează spre țintă la acea viteză și, în cele din urmă, decelerarea și, eventual, utilizarea PID pentru oprire în poziția finală. Această secvență particulară este uneori numită mișcare *trapezoidă,* deoarece un grafic al mișcării seamănă cu un trapez (vezi figura 12.34). Alte modalități de a ajunge la țintă (în funcție de sistem) ar putea fi numite *rapide* , *circulare* sau *S-curve*. Programatorul poate specifica dacă mai mult de o axă se va muta muta în același timp, și dacă da, controlerul le va muta într-un fash coordonatele

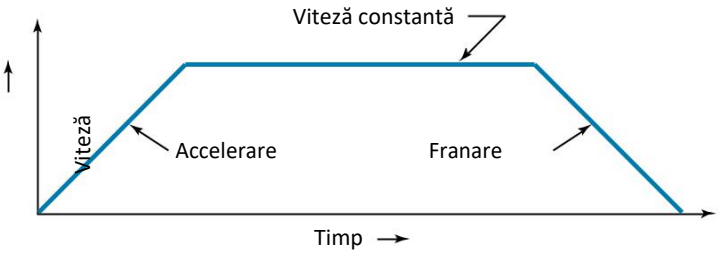


Figura 12.34
Graficul mișcării "trapezoidale"

, toate terminând în același timp. Dacă sunt specificate mai multe mișcări succesive, fără pauză între ele, prima mișcare se va fi urmată în a doua și așa mai departe, până când ținta este atinsă.

Nu există încă un limbaj de programare standard acceptat pentru controlerele de mișcare, ceea ce înseamnă că majoritatea sistemelor au propriul lor limbaj unic. Urmează un exemplu de program simplificat pentru sistemul DELTA-Tau PMAC.

Comanda	Explicație
#1->1000X	Atribuiți #1 motorului pe axa *X.*
#2->1000Y	Atribuiți #2 cu motorpe axa *Y.*
CLAR	Ștergeți conținutul existent.
LINIAR	Specificați modul de mișcare (trapez).
TA500	Setați valoarea accelerației.
TS200	Setați viteza.
X0 Y0	Deplasați-vă la poziția de pornire.
X10 Y0	Treceți la $x = 10$ și $y = 0$.
LOCUI	Așteptați 500 ms.
X0 Y10	Treceți la $x = 0$ și $y = 10$.
X0 Y0	Deplasați-vă înapoi în poziția de pornire.

REZUMAT

Un tip comun de sistem de control guvernează succesiunea repetitivă a evenimentelor. În mod tradițional, acest tip de control a fost implementat cu relee electromecanice, cronometre și secvențiatoare. Releele pot fi interconectate pentru a oferi o logică de control, de exemplu, două relee în serie devin o funcție AND. Releele de întârziere a timpului vor întârzia activarea contactelor lor pentru o anumită perioadă de timp (după ce au fost activate). Secvențiatoarele folosesc o camă rotativă pentru a deschide și închide contactele într-o anumită secvență.

O diagramă a scării, un tip special de diagramă de cablare, a fost dezvoltată pentru a documenta circuitele de control electromecanice. Acest tip de diagramă (care seamănă cu o scară) are două fire verticale (bare) pe fiecare parte a desenului; aceste fire furnizează puterea. Fiecare treaptă a diagramei scării se conectează de la o șină la alta și este un circuit separat, care constă de obicei dintr-o combinație de comutatoare, contacte releu, bobine releu și motoare. Este obișnuit ca bobina unui releu să fie într-o treaptă și contactele să fie în alta.

La sfârșitul anilor 1960, controlerul logic programabil (PLC) a fost dezvoltat pentru a înlocui controlerele electromecanice. PLC-ul este un computer mic, bazat pe microprocesor, de control al proceselor, care poate fi conectat direct la comutatoare, relee, motoare mici și așa mai departe și este construit pentru a rezista mediului industrial. PLC-ul execută un program care trebuie să fie scris pentru fiecare aplicație specifică. Programul este întotdeauna sub forma unei bucle, care are următorul model: PLC-ul citește și stochează

datele comutatorului de intrare și ale senzorului; determină care ar trebui să fie producția (ieșirile); trimite semnalele de control al ieșirii către dispozitivele care urmează să fie controlate.

Pentru a utiliza un PLC este necesară configurarea hardware-ului și software-ului. Instalarea hardware-ului constă în cablare PLC-ul la toate comutatoarele și senzorii sistemului și la dispozitive de ieșire, cum ar fi bobine releu, lămpi indicatoare sau motoare mici. Programul de control este de obicei dezvoltat pe un PC, folosind software-ul furnizat de producătorul PLC. Acest software permite utilizatorului să dezvolte programul de control sub forma unei diagrame scară pe ecranul monitorului. Odată ce programul este complet, acesta este convertit automat în instrucțiuni pentru procesorul PLC. Programul finalizat este apoi descărcat în PLC. Odată ce programul este în memoria PLC-ului, terminalul de programare poate fi deconectat, iar PLC-ul va continua să funcționeze pe cont propriu.

Din ce în ce mai multe PLC-uri sunt interconectate cu rețelele. Există trei niveluri de rețele: *nivelul* de informații pentru comunicațiile de birou, nivelul de *control* pentru PLC-urile interconectate și nivelul *dispozitivului* pentru conectarea senzorilor și actuatoarelor la PLC.

Controlerele de mișcare sunt circuite de control special realizate pentru a coordona mișcarea. Un controler de mișcare tipic funcționează cu un sistem în buclă închisă pentru a coordona deplasarea a două sau mai multe axe simultan.

GLOSAR

modul analogic I/O Un modul care se conectează la PLC și convertește semnalele ana-log din lumea reală în semnale digitale pentru PLC și invers.

instrucțiuni de biți Instrucțiunea de bază PLC care reprezintă în mod logic un comutator simplu sau o bobină releu; starea fiecărui comutator sau bobină ocupă 1 bit în memorie.

topologia magistralei Un tip de rețea în care toate nodurile sunt conectate la un singur cablu în locuri diferite, iar cablul are un început și un sfârșit (adică cablul nu este într-un inel).

sistem de schimbare a stării O metodă de comunicare într-o rețea în care un dispozitiv trimite un mesaj numai atunci când se schimbă ceva.

instrucțiuni de comparație Compară două cuvinte și produce un singur bit ca rezultat.

instrucțiuni de control Permite operații precum salturi și subrutine.

fișier de date Colectarea datelor într-un PLC care reprezintă starea actuală a comutatoarelor, contoarelor, releelor și a produselor asemenea.

memorie de date Secțiunea de memorie RAM într-un PLC unde este stocat fișierul de date.

pachet de date Un mesaj transmis într-o rețea; de obicei, include adresa de unde merge, adresa de unde a venit și mesajul în sine.

rețea determinată O rețea capabilă să atribuie prioritate unui mesaj și să garanteze că va ajunge într-un anumit timp.

rețea la nivel de dispozitiv O rețea utilizată pentru a conecta dispozitive de nivel scăzut, cum ar fi senzori și actuatori la un PLC.

DeviceNet Un tip de rețea la nivel de dispozitiv dezvoltată de Allen-Bradley. Consultați **rețeaua la nivel de dispozitiv.**

Procesor de semnal digital (DSP) Un tip de procesor digital care este proiectat pentru a suport sarcini de înaltă performanță, repetitive, intensive numeric.

I/O I/O discret care se ocupă de dispozitivele on-off. Modulele de intrare discrete sunt utilizate pentru a conecta comutatoare din lumea reală la PLC, iar modulele de ieșire discrete sunt utilizate pentru a porni lămpile, releele, motoarele și așa mai departe.

contor electromecanic Un dispozitiv care contorizează evenimentele. De fiecare dată când primește un puls, acesta incrementează un contor mecanic, care poate fi apoi citit de un operator sau utilizat pentru a activa un alt proces.

releu electromecanic (EMR) Un dispozitiv care utilizează un electromagnet pentru a închide (sau deschide) contactele comutatoare - cu alte cuvinte, un comutator alimentat electric.

sequencer electromecanic Un dispozitiv care oferă semnale de activare secvențiale; funcționează prin rotirea unui tambur cu came, care activează comutatoarele.

EMR *Vezi* **releu electromecanic.**

Ethernet Tipul de rețea utilizat pentru interconectarea PC-urilor și a computerelor mainframe în scopul schimbului de date și fișiere de date.

Fieldbus Un tip de rețea la nivel de dispozitiv. Vedeți **rețeaua la nivel de dispozitiv**

Instrucțiuni de mesaje și comunicații I/O Permiteți PLC-urilor să trimită și să primească mesaje și date către alte PLC-uri sau un terminal (PC), pentru acele aplicații în care PLC-urile sunt în rețea.

Module de intrare/ieșire Punctele de conectare în care comutatoarele, releele și alte tipuri de lucru din lumea reală sunt conectate la PLC.

diagrama scării Un tip de diagramă de cablare (care seamănă cu o scară) utilizată pentru circuitele con-trol; diagramele scării includ de obicei comutatoare, relee și dispozitivele pe care le controlează.

blocare *A se vedea* **sigilarea.**

instrucțiuni logice și shift Efectuați operații logice, cum ar fi AND, OR și NOT și SHIFT-uri la dreapta și la stânga pe cuvintele stocate în memorie.

instrucțiuni matematice PLC operații matematice pe cuvinte în memorie; aceste opera-tions pot include adunare, scădere, multiplicare, divizare, și altele.

mnemonic O abreviere de sondare în limba engleză pentru instrucțiuni logice PLC.

Controler de mișcare Un circuit de control special realizat pentru a coordona mișcarea. Un controler de mișcare tipic funcționează cu un sistem cu buclă închisă pentru a coordona mișcarea a două sau mai multe axe simultan.

nod Orice unitate conectată la o rețea care poate transmite și primi date.

în mod normal închise (NC) contacte releu contacte care sunt închise în stare de-activat și deschise atunci când releul este activat.

în mod normal, deschise (NU) contacte releu contacte care sunt deschise în stare de-activat și închis atunci când releul este activat.

releu off-delay Un releu de întârziere care, după ce a fost de-activat, așteaptă o anumită perioadă de timp înainte de activare.

releu de întârziere un releu de întârziere care, după ce a fost activat, așteaptă o anumită perioadă de timp înainte de activare.

Controlul PID O strategie comună de control. Instrucțiunile PID sunt disponibile pe unele PLC-uri.

PLC *A se vedea* controler logic **programabil.**

Magistrală PLC Firele din backplanul unui sistem modular PLC care permite micro-procesorului (în cadrul PLC-ului) să comunice cu modulele conectate.

modul plug-in Un circuit de interfață cu PLC-ul, ambalat ca un modul separat, care se conectează la PLC. Diferite module acceptă diferite dispozitive I/O, de la module simple de ieșire a comutatoarelor și releelor la modulele ADC și DAC.

releu pneumatic de întârziere a timpului Un releu de întârziere de timp care generează întârzierea prin permiterea aerului să iasă dintr-un burduf încărcat cu arcuri.

I / O intervievat Probabil cea mai comună metodă de comunicare într-o rețea, în care un PLC solicită (sondaje) un senzor pentru date numai atunci când are nevoie de ea.

bare electrice Parte a unei diagrame a scării, cele două linii verticale de pe fiecare parte a diagramului, care reprezintă liniile electrice.

procesor Parte a PLC-ului care execută instrucțiunile; este un computer mic, bazat pe microprocesor.

fișier program Conține programul pe care PLC-ul îl execută. Fișierul programului face parte din memoria PLC (cealaltă parte fiind fișierele de date).

controler logic programabil (PLC) Un computer mic, autonom, robust, conceput pentru a controla procesele și evenimentele dintr-un mediu industrial - pentru a prelua treaba făcută anterior cu controlere logice releu.

memorie program Atunci când se aplică la PLC-uri, secțiunea de memorie (RAM sau EEP-ROM) în cazul în care programul PLC este stocat.

port de programare Un conector pe un PLC prin care este descărcat programul.

modul de rulare Modul operațional al PLC-ului atunci când execută efectiv programul de control și, prin urmare, efectuează o anumită operațiune de control.

scanați Evenimentul în care procesorul face o trecere prin toate instrucțiunile din bucla programului control.

de etanșare Un releu cu fir, astfel încât o dată activat contactele sale proprii preia sarcina de a furniza energie la bobina; acesta va rămâne plin de energie, chiar dacă originalul energizant dispare.

releu de întârziere în stare solidă Un releu de întârziere care generează întârzierea cu un circuit electronic.

releu de întârziere termică Un releu de întârziere care generează întârzierea, permițând unei benzi bimetalice să se încălzească și să se îndoaie, care activează un comutator.

releu de întârziere în timp Un releu cu o acțiune intenționată de întârziere; adică contactele se închid (sau se deschid) într-o perioadă specificată de timp după *ce* releul este activat (sau de-activat).

EXERCIŢII

Secţiunea 12.1

1. Desenați o diagramă de cablare de releu pentru un circuit care implementează diagrama logică simplă prezentat în Figura 12.35. Circuitul trebuie să fie în stilul figurii 12.1 litera (a).
2. Repetați exercițiul 1, cu excepția realizați circuitul ca o diagramă a scării.
3. O casă mică are trei ferestre și două uși. Fiecare fereastră și ușă are un comutator atașat astfel încât contactele să se închidă atunci când se deschide o ușă sau o fereastră. Desenați o diagramă logică a scării care va aprinde o lumină dacă una sau mai multe ferestre sunt deschise sau dacă ambele uși sunt deschise.
4. Într-o instalație de prelucrare, borcanele de pe o bandă transportoare sunt curățate cu un jet de aer de înaltă presiune chiar înainte de a fi umplute (Figura 12.36). Când un borcan se apropie de stația de curățare, acesta activează un comutator (cu contacte atât NO, cât și NC). Banda transportoare se oprește timp de 10 s în timp ce jetul de aer este pornit; apoi începe banda transportoare, iar borcanul se mișcă de-a lungul. Desenați o diagramă logică a scării pentru a controla acest proces.

Figura 12.35

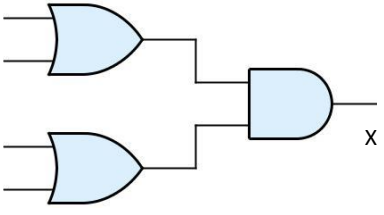
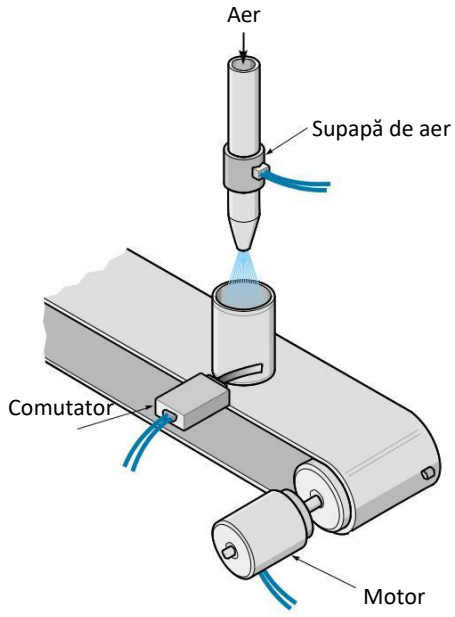


Figura 12.36



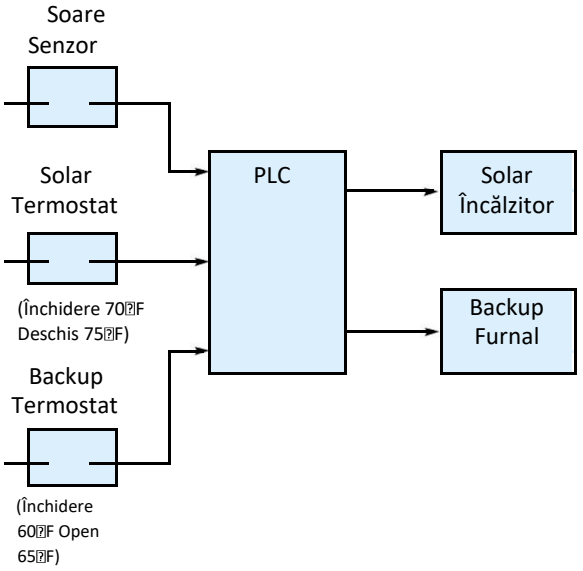
Secțiunea 12.2

5. Tocmai ați cumpărat un PLC, și este încă în cutie. Enumerați pașii generali pe care va trebui să îi urmați pentru a obține PLC-ul operațional într-o anumită sarcină.
6. Un PLC are opt intrări și opt ieșiri, așa cum se arată în figura 12.17(a). Desenați o diagramă electrică pentru acest PLC care va fi utilizat ca controler pentru situația exercițiului 3.
7. Un PLC are opt intrări și opt ieșiri, așa cum se arată în figura 12.17(a). Desenați o diagramă electrică pentru acest PLC care va fi utilizat ca controler pentru situația exercițiului 4.
8. Listați pașii pe care PLC-ul îi face pentru a executa programul diagramă a scării.

Secțiunea 12.3

9. Desenați o diagramă a scării PLC pentru situația descrisă în exercițiul 3.
10. Un motor este controlat de un buton de pornire NO, un buton de oprire NC și un dispozitiv de suprasarcină. Dispozitivul de suprasarcină este în mod normal închis și se se deschide dacă motorul se supraîncălzește. Desenați diagrama scării PLC pentru circuit. Includeți o lumină care se aprinde atunci când motorul se supraîncălzește.
11. Un PLC este de a controla sistemul de încălzire solară prezentat în figura 12.37. Sistemul are două părți interdependente: (a) Un termostat solar pornește și oprește încălzitorul solar *dacă* senzorul solar spune că soarele strălucește și (b) un termostat de rezervă pornește și oprește un cuptor convențional *dacă* energia solară este insuficentă. Ambele sisteme de încălzire au aceeași conductă, deci dacă termostatul de rezervă se închide, PLC-ul *trebuie să* pornească

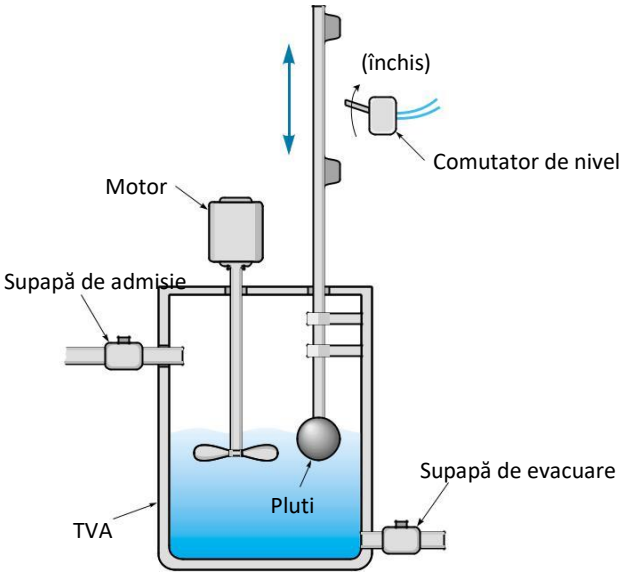
Figura 12.37



cuptorul de rezervă (și opriți încălzitorul solar dacă este pornit). Desenați diagrama scării PLC pentru acest sistem.

12. O cuvă de amestecare are o supapă de admisie, o supapă de evacuare, un motor de amestecare și un comutator de detectare a unui singur nivel (Figura 12.38). Ambele supape sunt deschise de solenoide. Comutatorul de nivel se închide atunci când cuva este plină și rămâne închis până când cuva este goală. Desenați o diagramă logică a scării PLC pentru a face următoarele:

Figura 12.38



a. Când butonul de pornire este apăsat, supapa de admisie se deschide până când cuva este plină.
b. Motorul mixerului devine activ timp de 5 minute.
c. Supapa de evacuare se deschide până când cuva este goală.

13. Modificați programul de control în două puncte din exemplul 12.6, astfel încât limitele de temperatură ale cuptorului să fie de 72-76 °C. Sistemul trebuie pornit și oprit fără comutatoare cu buton (în loc de comutatoare de comutare).

Secțiunea 12.4

14. Care sunt cele trei niveluri de rețele care ar putea fi găsite într-o industrie de control al proceselor.

15. Ce este o rețea "la nivel de dispozitiv" și ce avantaje ar putea oferi acest sistem față de cablurile tradiționale punct-la-punct?

Secțiunea 12.5

16. Prin ce este un "controler de mișcare" diferit de un PLC?

17. Utilizând programul eșantion din secțiunea 12.5 ca ghid, scrieți un program de control al mișcării pentru un sistem cu trei axe pentru a face următoarele:
Începeți de la $X = 0$, $Y = 0$ și $Z = 0$.
Treceți la $X = 5$, $Y = 5$ și $Z = 2$.
Treceți la $X = 4$, $Y = 4$ și $Z = 3$.
Pauză 500 ms.
Deplasați-vă la poziția de pornire.