NOŢIUNI DE AUTOMATIZĂRI INDUSTRIALE - extras

1. INTRODUCERE ÎN PROBLEMATICA AUTOMATELOR PROGRAMABILE

Un Automat Programabil = AP (sau PLC = Programable Logic Controller) este un controler secvenţial care asigură producerea unor evenimente într-o succesiune dorită şi programată, prin unitatea sa de ieşire, pornind de la o reacţie din sistemul controlat, prin unitatea sa de intrare.

În esenţă, un automat programabil este un calculator digital utilizat pentru automatizarea proceselor electromecanice şi electropneumatice, cum ar fi controlul maşinilor pe liniile de prelucrare, asamblare, al instalaţiilor de îmbuteliere, sau a instalaţiilor din parcurile de distracţii. Spre deosebire de calculatoarele de uz general, AP sunt proiectate pentru intrări şi ieşiri multiple, game extinse de temperatură, imunitate la zgomot electric şi rezistenţă la vibraţii şi impact.

Automatele programabile se prezintă în practică fie în construcţie monobloc (o carcasă ce conţine toate elementele hardware ale AP) fie în construcţie modulară (module cu funcţii diferite legate între ele printr-o magistrală de date). Diferenţele de aspect între cele două tipuri constructive sunt prezentate în Figura 1.1.

Fig. 1.1: Automat programabil monobloc (Festo FC640) şi modular (Siemens S7-300)

2. STRUCTURA HARDWARE A UNUI AUTOMAT PROGRAMABIL

Partea principală a arhitecturii unui AP (Figura 1.2) este **procesorul**, de regulă cu frecvenţa de tact mai mică decât cele folosite la PC-uri. Are rolul de a interpreta programele utilizator, de a rezolva problemele de comunicare între diferitele module, şi de a monitoriza apariţia eventualelor defecte interne.

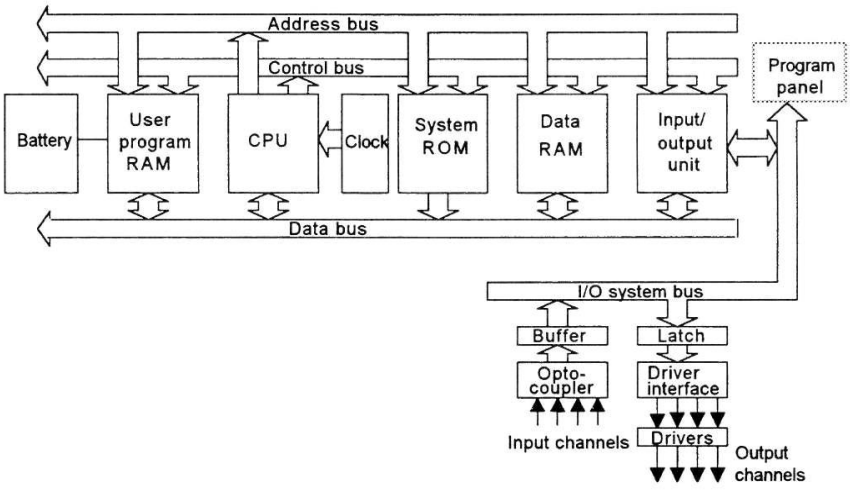

Fig. 1.2: Arhitectura hardware a aunui automat programabil

Modulele de porturi I/O (input/output, intrări/ieşiri) reprezintă interfaţa cu sistemul controlat (aici se conectează la AP senzorii şi traductoarele, respectiv actuatoarele şi elementele de avertizare). Toate semnalele de intrare şi ieşire sunt izolate galvanic de partea de procesare prin optocuploare.

Fiecare port I/O are o adresa de memorie rezervată, permiţând în acest fel monitorizarea tuturor porturilor I/O ***în mod circular continuu***, aceasta fiind o particularitate importantă a funcţionării programelor AP (Figura 1.3).

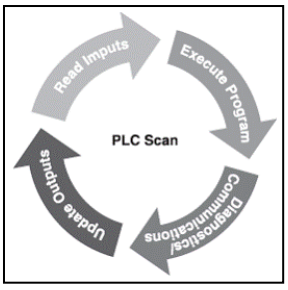

Fig. 1.3: Modul de funcţionare al unui ***ciclu de scanare***

Unităţile de memorie sunt utilizate la stocarea datelor şi a programelor care se folosesc în timpul lucrului:

- memorie ROM (Read-Only Momory) pentru stocarea permanentă a unor date de producător şi a sistemului de operare al PLC-ului;
- memorie RAM (Random-Access Memory) pentru programele utilizatorilor şi datele colectate pe porturi;
- memorii EPROM pentru programe de utilizator sau pentru date de folosinţă îndelungată, constante de programare, etc;

Programele pentru PLC şi datele de sistem pot fi stocate şi pe un PC obişnuit şi descărcate în PLC cu ajutorul reţelei sau a porturilor de comunicare ale acestuia (USB, Ethernet etc).

3. METODE STANDARDIZATE DE PROGRAMARE A AUTOMATELOR PROGRAMABILE

Metodele de programare a automatelor programabile sunt cuprinse în standardul IEC 61131. Acest standard defineşte o multitudine de metode cu scopul de a veni în ajutorul celor care programează AP şi care sunt de formaţie profesională diferită (automatişti, electricieni/electronişti, informaticieni). Producătorii de AP implementează în mediul de programare pentru un automat de regulă cel puţin două dintre metodele prezente în standard şi suportă în general conversia programului dintr-o metodă în alta. Conform standardului amintit mai sus metodele de programare ale AP sunt următoarele:

LD (Ladder Diagram – Diagrama Scară) este un limbaj semigrafic, asemănător schemelor cu circuite cu relee şi contacte şi operează în principal cu variabile logice (boole);

FBD (Function Block Diagram – Diagrama cu Blocuri de Funcţii) operează cu blocuri grafice interconectate, ce reprezintă atât operaţiile logice simple cât şi funcţii complexe.

ST (Structured Text – Text Structurat) care foloseşte instrucţiunile de atribuire, selecţie şi control a subprogramelor cu o structură apropiată de limbajele de programare de nivel înalt;

IL (Instruction List – Lista de Instrucţiuni) cu structură asemănătoare cu limbajele de asamblare ale microprocesoarelor;

SFC (Sequential Function Chart – Grafic Secvenţial de Funcţii), este un limbaj grafic compus din etape şi tranziţii cu ajutorul cărora se pot organiza structuri de programe secvenţiale şi paralele.

Metoda cu cea mai largă răspândire este **LD,** urmată de ST şi IL. În Figura 3.1 este prezentată o paralelă între diferitele moduri de realizare a programului (inclusiv în logică cablată) pentru funcţiile logice ”ŞI” respectiv ”SAU”.

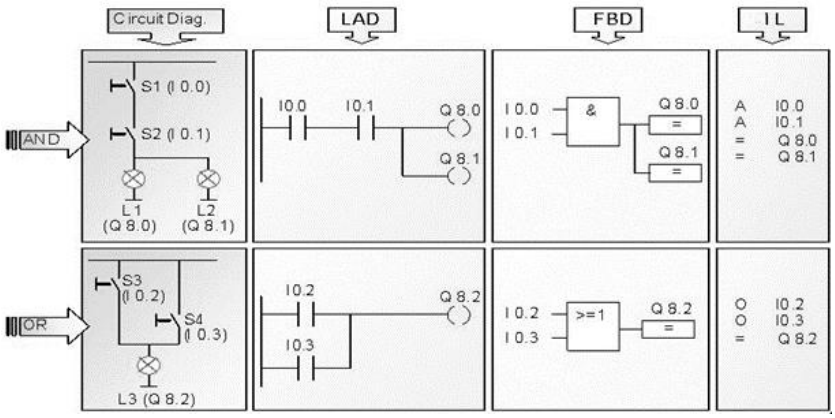

Fig.3.1: Funcţiile logice ”ŞI”, ”SAU” programate în diferite variante

Deasemenea, pe lângă metodele de programare, standardul mai defineşte şi tipurile de date elementare:

- Booleene, notate cu BOOL;
- Întregi, notate cu INT;
- Cuvinte (16 biţi), notate cu WORD;
- Cuvinte duble (32 biţi), notate cu DWORD;
- Reale (32 biţi), notate cu REAL;
- Şiruri de caractere, notate cu STRING;
- Timp şi dată, notate cu TIME respectiv cu DATE.

Este permisă şi utilizarea de date de tip tablou (ARRAY) şi structură (STRUCT), precum şi derivate ale acestora.

Identificarea datelor se face utilizând atât adrese absolute (adresare directă) cât şi simbolice (adresare indirectă).

Adresarea directă utilizează denumirea zonei de memorie asociată pentru identificarea adresei. Denumirile zonelor de memorie pot cuprinde două prefixe. Primul prefix poate fi:

- I, pentru intrări;
- Q, pentru ieşiri;
- M, pentru variabilele interne.

Al doilea prefix poate fi:

- x, y, pentru variabilele de tip boolean. Valoarea x reprezintă octetul, iar valoarea y reprezintă bitul;
- B, pentru octet (Byte);
- W, pentru cuvânt (Word);
- D, pentru dublu cuvânt (Double word).

De exemplu:

- **Ix.y**, reprezintă o variabilă de intrare booleană reprezentând bitul y din octetul x (I0.2 – adresa asociată celui de-al treilea pin din primul modul de intrare);
- **QBx**, reprezintă o variabilă de ieşire booleană reprezentată de octetul x;
- **MWx**, reprezintă o variabilă internă booleană reprezentată de cuvântul x;

Adresarea indirectă utilizează identificatorii, care sunt şiruri de caractere alfanumerice, începând cu o literă, pentru identificarea adresei. În aceste cazuri este nevoie de editarea unei tabele de simboluri pentru a face legătura dintre adresa absolută şi cea indirectă.

4. NOŢIUNI DE BAZĂ ÎN ALEGEREA ŞI PROIECTAREA SOLUŢIEI DE AUTOMATIZARE

Există mai mulţi factori care concură la *alegerea tipului de automat programabil*.

- dacă aplicaţia este mai simplă, criteriul de alegere cel mai important este numărul de intrări şi ieşiri precum şi dimensiunea programului utilizator. La aplicaţiile mai complexe, sunt luaţi în considerare şi timpii de răspuns precum şi dimensiunea memoriei care trebuie să înmagazineze un număr mare de date.
- în cazul proceselor răspândite în mai multe locaţii este mult mai indicată alegerea unor module de intrare/ieşire distribuite utilizând module de comunicaţie industrială. Această soluţie duce la reducerea numărului de cabluri de legătură cu procesul, comunicaţia între modulele de intrare/ieşire şi unitatea centrală a automatului programabil făcându-se prin intermediul magistralei de comunicaţie pe un număr redus de fire.

Cele mai comune *protocoale de comunicaţie industrială* sunt: **Profinet, IO-Link, Profibus** şi **Modbus.**

Alegerea limbajului de programare depinde de utilizator şi de complexitatea algoritmului de conducere. În cazul prelucrării datelor binare este recomandabil să se utilizeze limbajul LAD sau FBD, limbaje care sunt mult mai intuitive. În cazul manipulării de variabile complexe şi adresări indirecte este indicat limbajul STL care permite printre altele și procesarea unui volum mare de date.

Programul care cuprinde instrucţiunile necesare realizării sarcinii impuse prin tema de proiectare este recomandat a fi modular. Modulele de program pot fi: *orientate către proces*, caz în care, fiecare modul corespunde unei părţi din proces sau maşini sau *orientate funcţional*, caz în care modulele corespund funcţiilor din proces, ca de exemplu: comunicare, mod de operare, etc.

După scriere, programul este testat. Testarea poate fi făcută pe un automat programabil virtual implementat chiar în mediul de progamare, sau în automatul programabil real, după încărcarea programului în memoria de programe a acestuia.

După testarea cu succes a programului acesta este încărcat în memoria EPROM şi apoi este generată documentaţia aferentă.

Bibliografie:

1. Gheorghe Prisăcaru, Mihai Bercea, Bogdan Grămescu, Valentin Ciupe; Mecatronică Aplicată; Ed. Oamenilor de Știință din România, 2011; ISBN: 978-606-837-143-6
2. Maniu Inocentiu, Dolga Valer, Ciupe Valentin, Bogdanov Ivan, Radulescu Corneliu, Varga Stefan, Robotica. Sisteme de actionare, vol.2, Ed. Politehnica, Timisoara, ISBN 978-973-625-996-8, 2009
3. Hannse D.: Programmable Logic Controllers : A Practical Approach to IEC 61131-3 Using CoDeSys, John Wiley & Sons 2015, ISBN: 9781118949221
4. Bolton, W.: Programmable Logic Controllers, 6th Edition; Elsevier 2015; ISBN: 9780081003534;
5. Parr, E.A.: Programmable Controllers: An Engineer's Guide; Newnes 2003; ISBN: 0-7506-5757-X.