

BỘ ĐIỀU KHIỂN LOGIC LẬP TRÌNH ĐƯỢC

9.1 KHÁI NIỆM CHUNG

Trong hệ thống tự động thường gặp những thiết bị làm việc theo kiểu tuần tự, theo qui luật if ... then ... else với tín hiệu vào và ra có hai mức, ví dụ như contact hành trình, role. Các sơ đồ này có thể thực hiện bằng role và mạch định thời nhưng với sơ đồ phức tạp số lượng role khá lớn, độ tin cậy kém và nhiều khi không đạt yêu cầu. Từ những năm 70 để đáp ứng yêu cầu có những thiết bị điều khiển thay thế sơ đồ role, đã xuất hiện bộ điều khiển logic lập trình được (*Programmable Logic Controller-PLC*) và ngày càng hoàn thiện, được áp dụng rộng rãi trong công nghiệp (PLC của hãng Allen Bradley Corporation sản xuất năm 1977 sử dụng vi xử lý 8080).

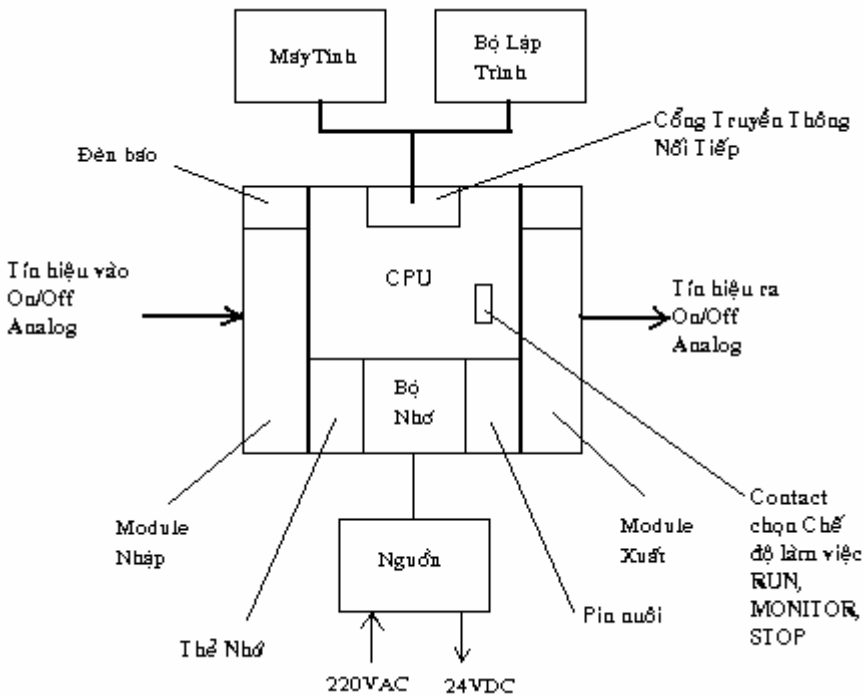
Các PLC đầu tiên chỉ thực hiện được các phép tính logic, tín hiệu vào và ra là tín hiệu rời rạc, còn hiện nay PLC có thể thực hiện được các phép tính số học, logic và làm việc được với cả tín hiệu liên tục, trong một số trường hợp PLC được sử dụng thay cho máy tính (một số hãng dùng từ PC- Programmable Controller để chỉ PLC).

Một hệ thống phức tạp thường gồm máy tính (vi xử lý) thực hiện những công việc phức tạp và PLC thực hiện các công việc mang tính chất tuần tự. Máy tính và PLC kết nối với nhau qua đường truyền nối tiếp và trao đổi thông tin cho nhau. Nhiều máy tính và PLC kết nối với nhau theo mạng điều khiển.

PLC gồm các thành phần chính sau:

- Khối CPU (Vi xử lý)

- Khối nhớ RAM, ROM, EPROM, EEPROM
- Khối nhập
- Khối xuất
- Bộ lập trình cầm tay
- Nguồn
- Pin nuôi
- Thẻ nhớ
- Module mở rộng



Hình 9.1: Cấu trúc PLC

Chương trình điều hành của nhà sản xuất, chứa trong bộ nhớ ROM (EPROM), thực hiện các công việc sau:

- Kiểm tra hoạt động bản thân PLC,
- Đọc tín hiệu vào ở khối nhập,
- Chuyển đổi chương trình người dùng chứa ở RAM hay thẻ nhớ sang mã máy của vi xử lý để vi xử lý thực hiện,
- Xuất tín hiệu ra khối xuất,

- Giao tiếp vi xử lý với bộ lập trình cầm tay (hand held programming console) hay với máy tính,
- Giao tiếp nối tiếp RS-232 hoặc RS 485.

Chương trình người dùng đưa vào PLC, tùy trường hợp, từ bộ lập trình cầm tay, bàn phím trên PLC hay từ máy tính và chứa vào RAM, một nguồn pin nuôi RAM khi cắt điện nguồn, có một tụ điện trị số khá lớn mắc song song với chân cấp nguồn của RAM để bảo đảm chương trình và dữ liệu cần thiết vẫn còn lưu lại một thời gian sau khi cắt nguồn PLC hay pin. Trong trường hợp cần thiết PLC hỗ trợ nạp chương trình vào thẻ nhớ EPROM hay EEPROM.

Bộ nguồn cho PLC có thể lấy từ nguồn xoay chiều hay nguồn một chiều 24V.

Bộ lập trình cầm tay và máy tính lập trình ghép nối với PLC qua ngõ truyền nối tiếp.

PLC có thể chế tạo dưới dạng khối gắn kết gồm các khối nguồn xử lý, bộ nhớ, khối nhập và xuất cùng chung trong một vỏ nhựa, hoặc theo dạng module (đơn thể) gồm module nguồn, module CPU và các module nhập xuất, module chức năng ...

PLC nhận tín hiệu vào và xuất tín hiệu ra dạng ON/OFF song song, nối tiếp hay dạng tương tự. Với các module phù hợp có thể cho PLC phát ra các tiếng nói cảnh báo hay hướng dẫn.

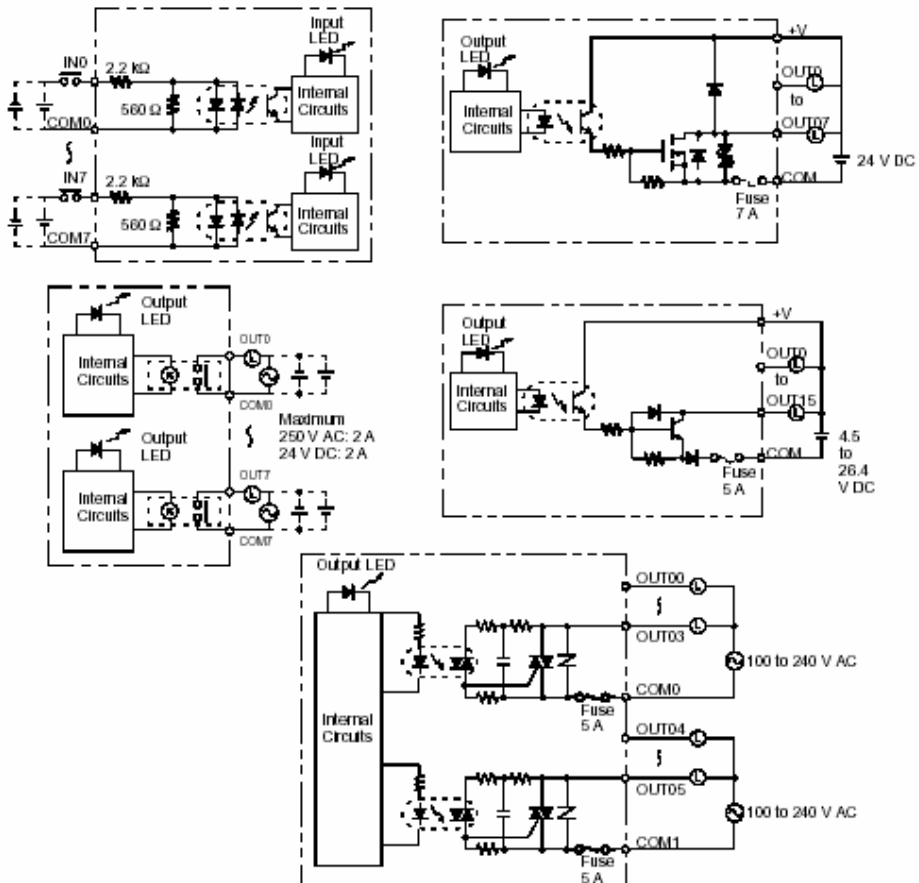
Các module chức năng giúp mở rộng khả năng của PLC như khuếch đại đo nhiệt độ, điều khiển quá trình vòng kín, điều khiển vị trí, ghép nối modem, mạng công nghiệp.

Quá trình điều khiển có thể hiển thị lên màn hình kèm với các thông số trạng thái nhờ phần mềm giao diện người-máy (HMI Human Machine Interface). Màn hình thường kết hợp với các phím bấm (OP Operator Panel) để điều khiển và quan sát thông số quá trình.

PLC được thiết kế để làm việc trong môi trường công nghiệp do đó mức tín hiệu logic vào là 24V; đối với tín hiệu tương tự nhỏ từ cặp nhiệt hay nhiệt điện trở, có sẵn khối khuếch đại chống nhiễu và không trôi đi kèm.. Do PLC làm việc theo chu kỳ quét nên nó không đáp ứng với tín hiệu thay đổi quá nhanh, điều này hạn chế áp dụng PLC cho việc điều khiển vòng kín các đối tượng

có quán tính nhỏ nhưng lại gia tăng độ tin cậy chống nhiễu của thiết bị.

Các tín hiệu xuất/nhập số và tương tự của PLC thường được ghép nối thông qua optocoupler để bảo đảm an toàn. H.9.2 trình bày sơ đồ khối nhập và xuất số.



Hình 9.2: Sơ đồ khối nhập và xuất số

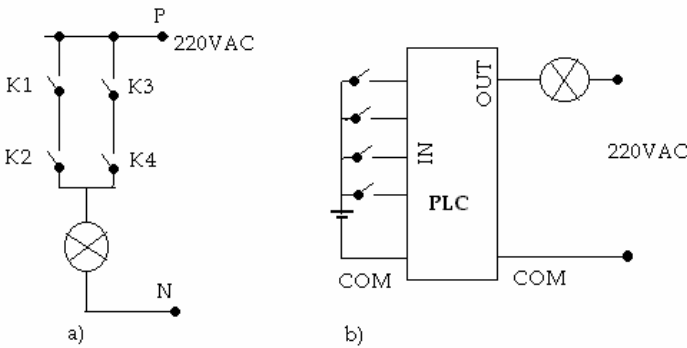
PLC thực hiện chương trình chứa trong bộ nhớ người dùng (UM- User Memory) theo chu kỳ quét. Một chu kỳ quét bắt đầu từ lệnh đầu tiên và kết thúc ở lệnh cuối cùng. Ở mỗi chu kỳ quét PLC đọc trạng thái ngõ vào, thực hiện chương trình, cập nhật ngõ ra. Thời gian thực hiện chu kỳ quét từ 0,1ms đến hàng chục ms tùy theo vận tốc xử lý của CPU và độ dài của chương trình. Thời gian thực hiện một lệnh cơ bản nhất khoảng dưới 1μs.

Chương trình PLC được viết dưới ba dạng:

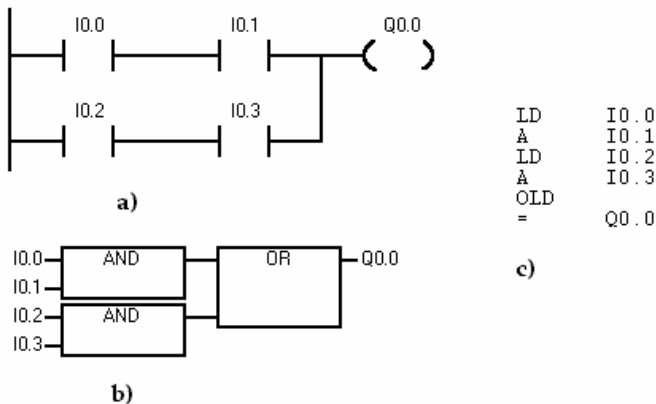
- Giản đồ thang (*Ladder diagram - LAD*)
- Khối hàm (*Control System Flowchart – CSF, FBD Function Block Diagram*)
- Bảng phát biểu (*Statement list - STL*)

Phương pháp giản đồ thang tương tự sơ đồ rơle, dạng FBD giống như các sơ đồ trong kỹ thuật số còn dạng STL tương tự các dòng lệnh của vi xử lý. Tùy theo hãng chế tạo có thể lập trình cho PLC bằng một hay nhiều dạng biểu diễn trên.

Vi dụ: xét sơ đồ tắt mở đèn dùng 4 tiếp điểm như H.9.3.

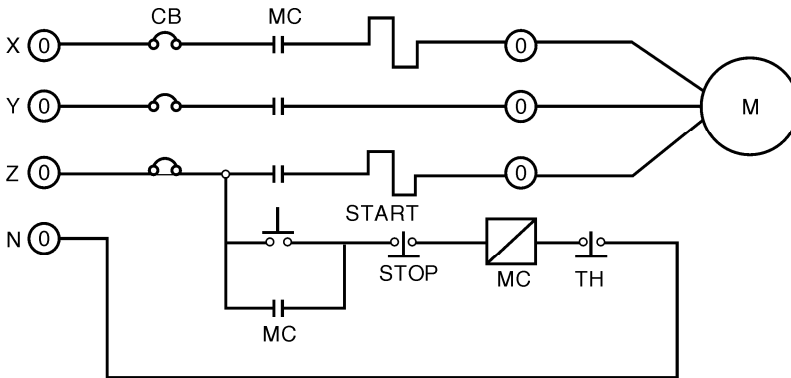


Hình 9.3: a) Sơ đồ mạch tiếp điểm; b) Sơ đồ kết nối PLC
Ta có thể biểu thị chương trình bằng ba dạng như H.9.4



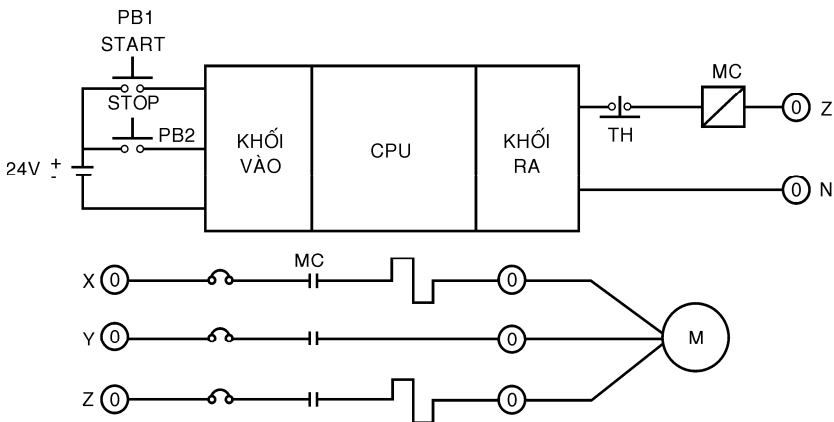
Hình 9.4: a) Dạng LAD; b) Dạng FBD; c) Dạng STL

Ví dụ: điều khiển động cơ xoay chiều theo sơ đồ H.9.5a.



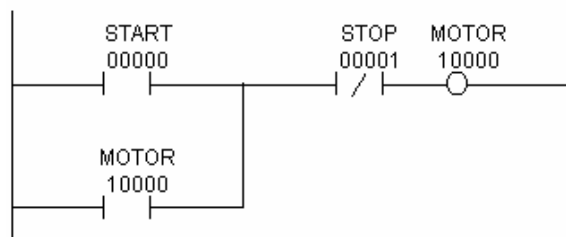
Hình 9.5a

Sơ đồ H.9.5a biến đổi thành sơ đồ điều khiển dùng PLC H.9.5b.



Hình 9.5b

Chương trình điều khiển dạng LAD (H.9.5c):



Hình 9.5c: Chương trình LAD OMRON

Các nút nhấn PB1 và PB2 nối với hai ngõ vào có địa chỉ lần lượt 00000 và 00001. Cuộn dây contactor MC nối với ngõ ra địa chỉ 10000. Chương trình dạng STL như sau:

Rung No.	Prg. Addr.	Instruction	Address	Name
0001	0000	LD	00000	START
	0001	OR	10000	MOTOR
	0002	AND NOT	00001	STOP
	0003	OUT	10000	MOTOR

Việc lập trình cho PLC được thực hiện theo các bước sau:

- Xác định thứ tự làm việc của máy
- Vẽ lưu đồ hệ thống
- Gán các địa chỉ xuất/ nhập
- Viết chương trình dạng LAD hay STL và nạp vào PLC
- Kiểm tra chương trình và sửa lỗi
- Gán các ngõ nhập và xuất cho PLC
- Chạy chương trình và sửa lỗi
- Lưu lại chương trình trên hai đĩa hay/và giấy

Có rất nhiều hãng sản xuất PLC với nhiều kiểu khác nhau và khó mà liệt kê hết được:

OMRON: ZEN, CPM1A, CPM2, C200H, CQM1H, CS1
SIEMENS: LOGO, S5-90U, S5-95U, S5-115U, S5-135U;
S5-155U, S7-200, S7-300, S7-400,
ALLEN-BRADLEY: Micrologic1000, SLC500, PLC5, LOGIX
MITSUBISHI Alpha, FX, Melsec- Q
SCHNEIDER: TSX

Trong phần sau ta sẽ đi sâu phân tích hoạt động của PLC hãng OMRON và SIEMENS.



a) LOGO

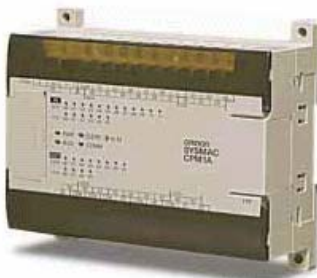


b) S7-200



c) S7-300 và màn hình OP

Hình 9.6: PLC SIEMENS



a) CPM



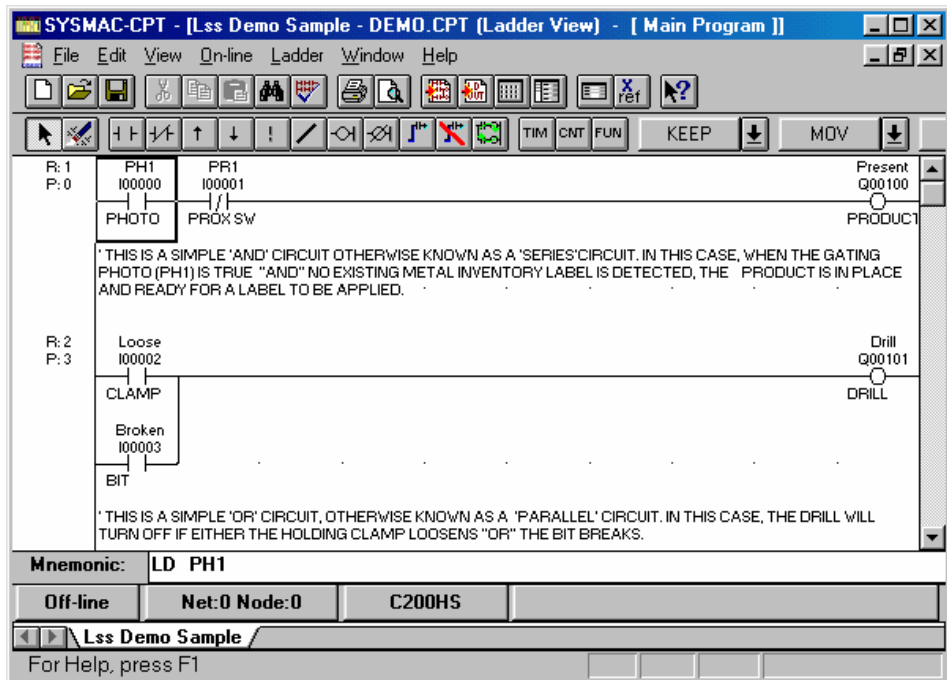
b) C200

Hình 9.7: PLC OMRON

9.2 PLC OMRON

9.2.1 Phần mềm lập trình

Phần mềm lập trình cho PLC OMRON rất đa dạng. Dạng LAD và STL được đưa vào PLC thông qua máy tính với các phần mềm lập trình như Sysmac Support Software SSS, Syswin, Sysmac-CPT, CX-Programmer. Ngoài ra còn có thể lập trình dạng STL nhờ bộ lập trình cầm tay (*programming console*).



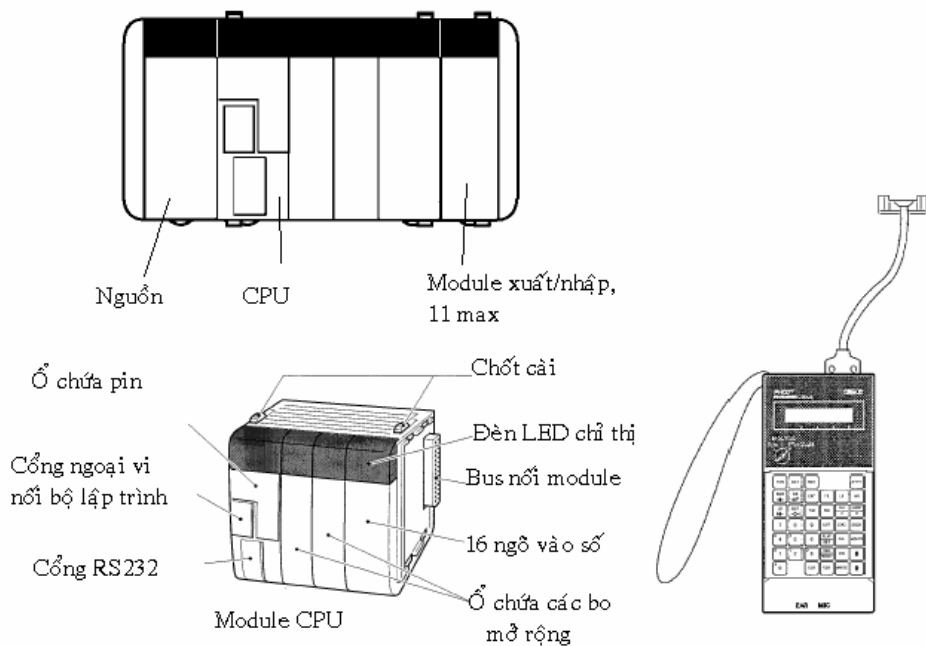
Hình 9.8: Giao diện phần mềm CPT

Các phần mềm lập trình giúp soạn thảo sửa chữa chương trình, kết nối với PLC, điều khiển PLC ở ba chế độ RUN, STOP và MONITOR, chế độ STOP (PROGRAM) dùng để nạp chương trình từ máy tính xuống PLC (download) hay chép chương trình trong bộ nhớ PLC lên máy tính (upload), ở chế độ RUN và MONITOR giá trị các ngõ vào ra, các ô nhớ, timer, counter được hiển thị trên chương trình, riêng ở chế độ MONITOR có thể thay đổi nội dung các ô nhớ. Chương trình chứa trong PLC có thể cài

mật mã để tránh chép trộm.

9.2.2 Sơ lược về cấu hình PLC Omron

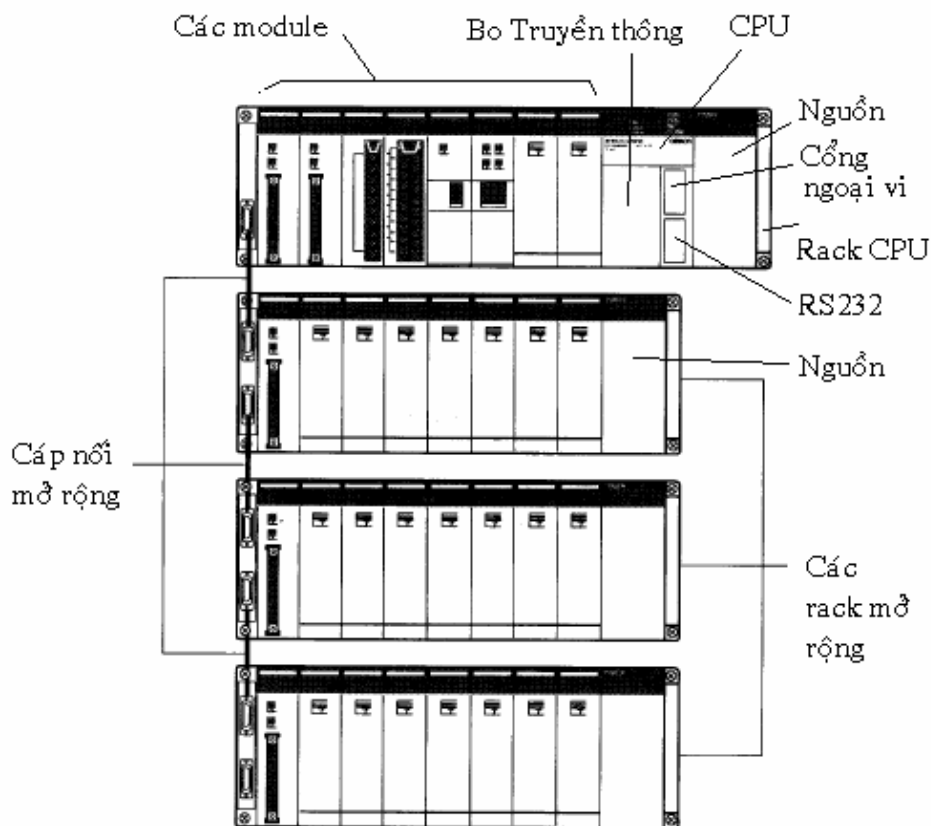
Trong phần này chúng ta chỉ khảo sát ba loại là CQM1, CPM1 và C200H. CQM1 có cấu trúc dạng module, gồm module nguồn , CPU và các module xuất/nhập. Có thể ghép tối đa đến 11 module xuất/nhập. Nếu dùng module mở rộng thì ghép thêm đến 5 module xuất/nhập. Các module ghép với nhau thông qua bus nối bên hông, toàn bộ đặt trên đường rầy (rail)



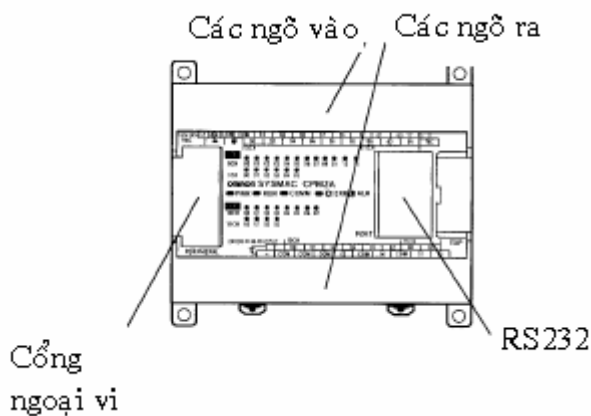
Hình 9.9: PLC CQM1H

Loại C200H có cấu trúc giá (rack) gồm các module gắn trên mặt đế (back plane), giá CPU gồm module nguồn, CPU, các module xuất/nhập, số module gắn vào tùy loại mặt đế, tối đa là 10, muốn thêm module thì dùng các giá mở rộng, tối đa 3 giá mở rộng.

Loại CPM1 cấu trúc đơn khối gọn nhẹ, có thể thêm ba khối mở rộng để tăng khả năng PLC.



:Hình 9.10: C200H



Hình 9.11: CPM2

9.2.3 Cấu trúc địa chỉ bộ nhớ PLC Omron

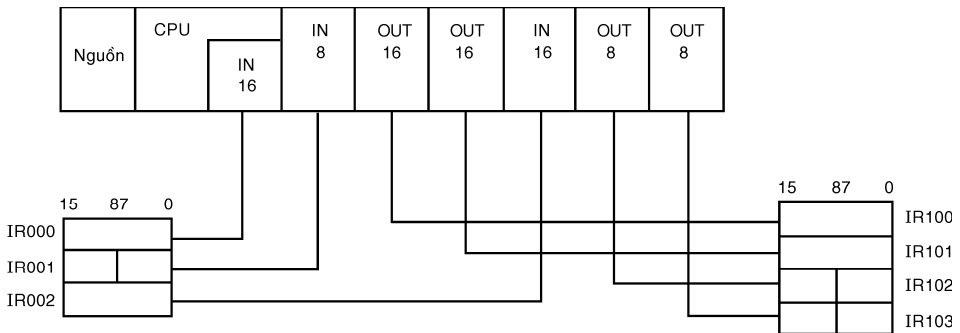
Bộ nhớ PLC Omron chia làm nhiều vùng IR, SR, TR, HR, AR, TC, DM, UM. Tùy theo kiểu PLC mà các vùng nhớ này có các độ dài khác nhau. Một số vùng nhớ có thể truy xuất theo từng bit hay từ (word = 16bit), một số chỉ có thể truy xuất theo từ. Một số vùng nhớ được lưu trữ số liệu nhờ tụ điện, thời gian lưu trữ 20 ngày, nếu có pin nuôi thời gian lưu trữ là 5 năm..

a) Vùng nhớ IR (Internal Relay) chia làm hai vùng nhỏ:

- Vùng nhập: tương ứng với các ngõ vào của thiết bị nhập, có thể xử lý theo từ hay bit
- Vùng xuất: tương ứng với các ngõ ra của thiết bị.

Địa chỉ của các ngõ xuất nhập phụ thuộc kiểu PLC và vị trí các module xuất nhập.

PLC COM1



Hình 9.12: Qui định địa chỉ module xuất nhập

Địa chỉ từ xuất/nhập tính từ trái sang, bắt đầu từ IR000 cho khối nhập và IR100 cho khối xuất, nhưng đã có khối nhập gắn sẵn trong khối CPU nên địa chỉ khối nhập gắn thêm vào sẽ bắt đầu từ IR001. Với CPU 11/21-E có tối đa 128 bit xuất/nhập còn với CPU 4X-E tối đa 192 bit. Địa chỉ tính theo từ gồm ba số, còn địa chỉ theo bit thêm hai số từ 00 đến 15 sau địa chỉ từ

CPM1A

Nhập	Xuất	Số hiệu
6 điểm 00000..00005	4 điểm 01000..01003	CPM1A-10CDR
12 điểm	8 điểm	CPM1A-20CDR
18 điểm	12 điểm	CPM1A-30CDR
24 điểm	16 điểm	CPM1A-40CDR

Có thể gắn thêm đơn vị xuất/ nhập mở rộng để thêm ngõ xuất/nhập số tương tự cho CPM1A-30, CPM1A-40 và CPM2.

C200H: PLC C200H được sắp xếp ngược với PLC CQM1, địa chỉ tính từ rãnh bên trái nhất của giá, khối I/O được gắn địa chỉ bắt đầu từ IR000.

IR 000	IR 001	IR 002	CPU	Nguồn
--------	--------	--------	-------	-----	-------

b) *Vùng làm việc:* dùng làm vùng nhớ dữ liệu, các vùng nhập và xuất ở trên nếu không liên kết với các module nhập xuất cũng có thể dùng làm vùng nhớ dữ liệu.

c) *Vùng nhớ SR (Special Relay)* dùng cho các chức năng đặc biệt như cờ hiệu, tạo xung, và làm vùng nhớ dữ liệu.

d) *Vùng nhớ HR: (Holding Relay)* dùng chứa dữ liệu được lưu khi mất điện.

e) *Vùng nhớ LR (Link Relay)* dùng để trao đổi dữ liệu giữa hai PLC.

f) *Vùng nhớ AR (Auxiliary Relay)* dùng làm cờ hiệu và bit điều khiển, được lưu khi mất điện.

g) *Vùng nhớ TC (Bộ đếm/Định thời)* dùng cho các lệnh Timer/ Counter.

h) *Vùng nhớ TR (Temporary Relay)* dùng để chứa tạm trạng thái ON/OFF của các nhánh rẽ.

i) *Vùng nhớ DM (Data Memory)* chứa thông số cấu hình của PLC và dùng làm vùng nhớ dữ liệu, chỉ truy xuất theo từ, có một số ô nhớ chỉ đọc. Nội dung được giữ lại khi mất điện. Một số ô

nhớ dùng để ghi cấu hình (DM6600..DM6655)

j) *Vùng nhớ UM (User Program Area)* chứa chương trình người sử dụng, dung lượng tùy CPU, được lưu khi mất điện.

Ngoài ra còn một số vùng nhớ khác được mô tả chi tiết trong tài liệu của nhà sản xuất. Các lệnh tham chiếu bộ nhớ phải ghi rõ tên vùng nhớ, trừ IR và SR.

Sau đây là các địa chỉ vùng nhớ theo từ (bảng 9.1).

Bảng 9.1

Vùng nhớ		CQM 1	CPM1
IR	Vùng nhập	IR 000 ÷ IR 011	IR 000 ÷ IR 009
	Vùng xuất	IR 100 ÷ IR 111	IR 010 ÷ IR 019
	Vùng làm việc	IR 012 ÷ IR 095	IR 200 ÷ IR 231
		IR 112 ÷ IR 195	
		IR 216 ÷ IR 219	
		IR 224 ÷ IR 229	
SR		SR 244 ÷ SR 255	SR 232 ÷ SR 255
Vùng mở rộng		IR 200 ÷ IR 215 IR 240 ÷ IR 243	
TR		TR 0 ÷ TR 7 (bit)	TR 0 ÷ TR 7
HR		HR 00 ÷ HR 99	HR 00 ÷ HR 19
AR		AR 00 ÷ AR 27	AR 00 ÷ AR 15
LR		LR 00 ÷ LR 63	LR 00 ÷ LR 15
TC		TC000 ÷ TC511	TC 000 ÷ TC 127
DM		DM 0000 ÷ DM 6655	DM 0000 ÷ DM 6655

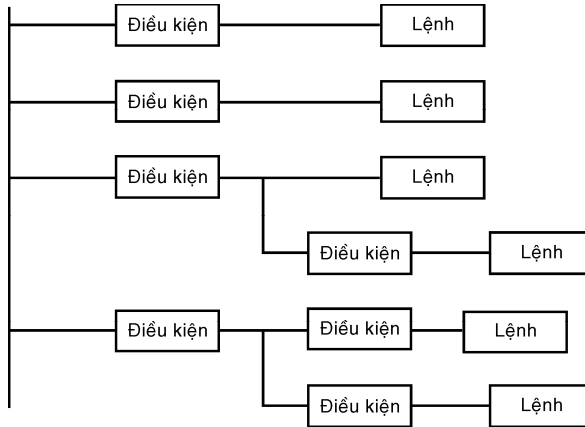
Các vùng nhớ có thể truy cập bit thì thêm hai số từ 00 đến 15 sau địa chỉ từ khi truy cập bit, Với Timer/Counter thì địa chỉ từ kèm thêm nhiệm vụ là bit trạng thái.

Địa chỉ vùng nhớ IR, SR của C200H khác với CQM1 và CPM1A:

IR1	IR 000 ÷ IR 235
SR1	SR 236 ÷ SR 255
SR2	SR 256 ÷ SR 299
IR2	IR 300 ÷ IR 511

9.3 CÁC LỆNH CƠ BẢN CỦA PLC OMRON

Chương trình LAD có cấu trúc như H9.13 gồm các network, mỗi network gồm các điều kiện, khối điều kiện và lệnh kết nối nhau, có thể có một lệnh hay nhiều lệnh. Các lệnh được thực hiện theo thứ tự từ trái sang phải và từ trên xuống dưới. Network có dòng chú thích để chương trình dễ hiểu.



Hình 9.13: Sơ đồ chương trình tuyến tính

Khối điều kiện là điều kiện đơn hay tổ hợp logic các điều kiện đơn. Điều kiện đơn biểu thị bằng một tiếp điểm thường mở hay thường đóng. Tổ hợp các điều kiện đơn là kết hợp các tiếp điểm nối tiếp hay/và song song. Mỗi tiếp điểm tương ứng với một bit nhập/xuất hay một bit nhớ. Một tiếp điểm thường mở sẽ đóng nếu bit tương ứng on, một tiếp điểm thường đóng sẽ đóng nếu bit tương ứng off.

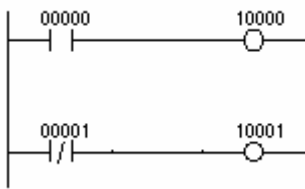
9.3.1 LOAD và LOAD NOT

Điều kiện đầu của một khối logic trong giản đồ thang ứng với lệnh LOAD (**LD**) đọc một tiếp điểm thường mở hay LOAD NOT (**LD NOT**) đọc tiếp điểm thường đóng.

9.3.2 OUTPUT và OUTPUT NOT (OUT và OUT NOT)

Hai lệnh này điều khiển một bit xuất hay một bit nhớ.

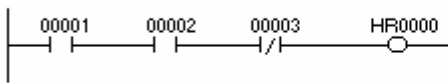
- Lệnh **OUT** b: toán hạng sẽ on nếu điều kiện on.
- Lệnh **OUTNOT** b: toán hạng sẽ on nếu điều kiện off.



LD	00000
OUT	10000
LD NOT	00001
OUT	10001

9.3.3 AND, AND NOT

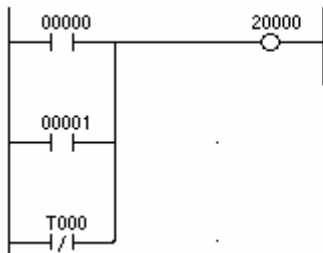
Hai lệnh này dùng để ghép hai điều kiện nối tiếp nhau:



LD	00001
AND	00002
AND NOT	00003
OUT	HR0000

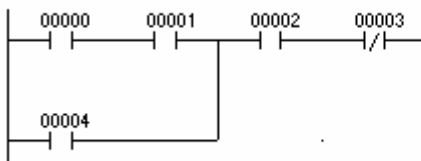
9.3.4 OR, OR NOT

Ghép hai tiếp điểm song song nhau.



LD	00000
OR	00001
OR NOT	T000
OUT	20000

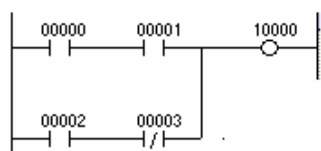
Các lệnh AND, AND NOT, OR, OR NOT có thể kết hợp với nhau:



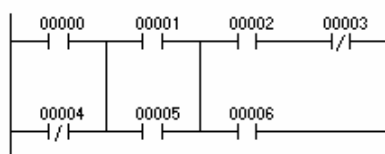
LD	00000
AND	00001
OR	00004
AND	00002
AND NOT	00003

9.3.5 AND LOAD và OR LOAD

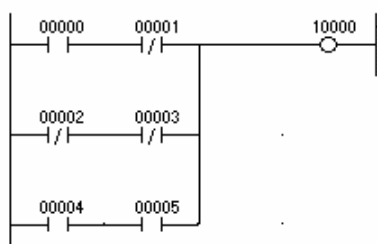
Lệnh AND LOAD (AND LD) kết hợp các khối logic nối tiếp nhau. Lệnh OR LOAD (OR LD) kết hợp các khối logic song song. Hình 9.14 cho các ví dụ sử dụng lệnh AND LD và OR LD.



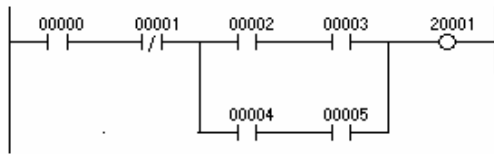
LD	00000
AND	00001
LD	00002
AND NOT	00003
OR LD	
OUT	10000



LD	00000
OR NOT	00004
LD	00001
OR	00005
AND LD	
LD	00002
AND NOT	00003
OR	00006
AND LD	



LD	00000
AND NOT	00001
LD NOT	00002
AND NOT	00003
OR LD	
LD	00004
AND	00005
OR LD	
OUT	10000

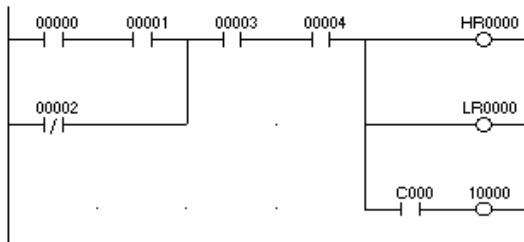


LD	00000
AND NOT	00001
LD	00002
AND	00003
LD	00004
AND	00005
OR LD	
AND LD	
OUT	20001

Hình 9.8: Ví dụ các lệnh cơ bản

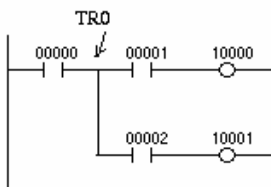
9.3.6 Mã hóa nhiều lệnh bên phải

Trong trường hợp có nhiều lệnh được thực hiện với cùng điều kiện, ta sẽ viết chương trình STL theo thứ tự từ trên xuống dưới. Trường hợp các lệnh có điều kiện khác nhau ta dùng các biến nhớ trung gian TR hay dùng lệnh INTERLOCK.

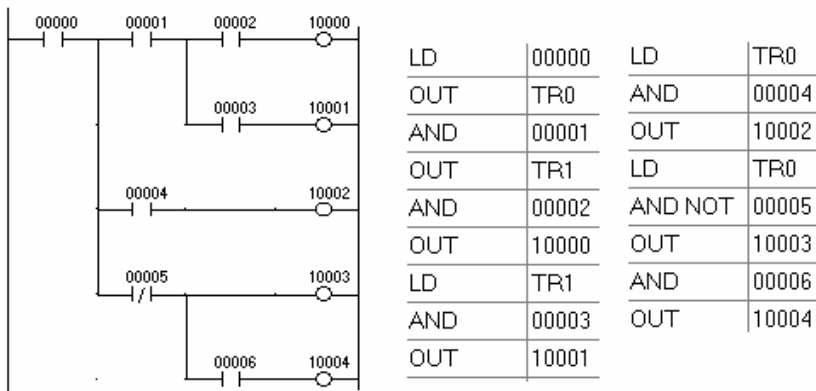


LD	00000
AND	00001
OR NOT	00002
AND	00003
AND	00004
OUT	HR0000
OUT	LR0000
AND	C000
OUT	10000

Dùng bit TR: Có 8 bit nhớ TR 0 ÷ TR 7. Kết quả điều kiện ở điểm rẽ nhánh chứa trong một bit TR



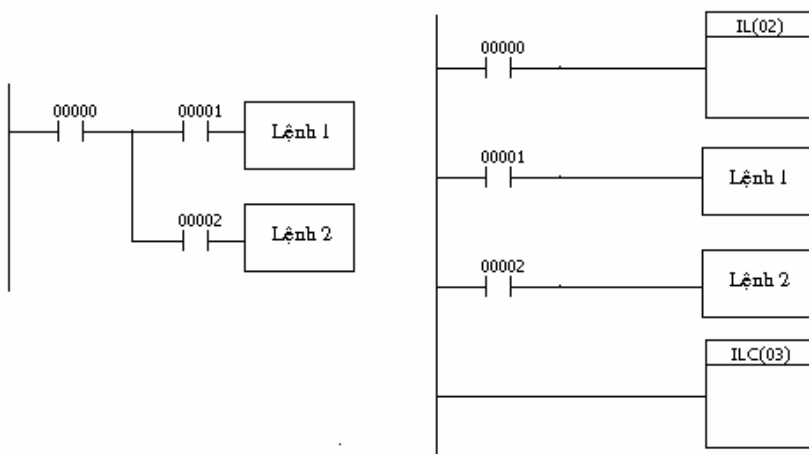
LD	00000
OUT	TR0
AND	00001
OUT	10000
LD	TR0
AND	00002
OUT	10001



Đôi khi có thể sắp xếp lại sơ đồ để loại bỏ bit TR.

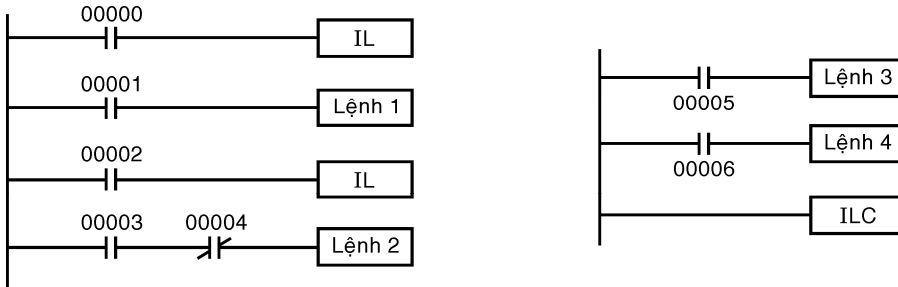
Dùng Interlock IL (02) và Interlock clear ILC (03)

Lệnh IL lưu trữ điều kiện và sử dụng cho các lệnh nằm giữa IL và ILC. Nếu điều kiện cho IL on thì thực hiện các lệnh nằm giữa IL và ILC, nếu điều kiện cho IL off thì không thực hiện các lệnh này. Có thể dùng nhiều lệnh IL và một ILC.



Trong chương trình sau, nếu IR00000 off thì các lệnh 1 đến 4 từ IL đến ILC không được thực hiện vì điều kiện off. Nếu IR00000 on thì lệnh 1 được thực hiện tùy trạng thái IR00001,

trạng thái của IR00002 được xét để làm điều kiện cho IL kế ...

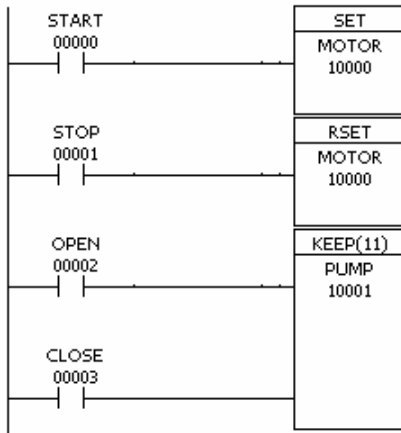


9.3.7 Lệnh SET và RESET

- Lệnh **SET b**: bit b on khi điều kiện on và giữ nguyên b on khi điều kiện trở thành off.
- Lệnh **RSET b**: bit b off khi điều kiện on và giữ nguyên off khi điều kiện trở thành off.

9.3.8 Lệnh KEEP b

Làm bit b on khi S on và bit b off khi R on. Các lệnh KEEP SET, RESET, DIFU, DIFD dùng với các bit IR, SR, AR, HR, LR, riêng lệnh OUT dùng với IR, SR, AR, HR, LR, TR.



LD	00000	START
SET	10000	MOTOR
LD	00001	STOP
RSET	10000	MOTOR
LD	00002	OPEN
LD	00003	CLOSE
KEEP(11)	10001	PUMP

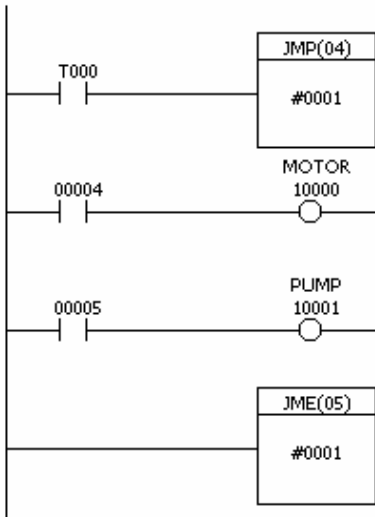
9.3.9 Lệnh vi phân lên và vi phân xuống

- **DIFU(13) b**: bit b on trong một chu kỳ khi điều kiện từ off sang on.
- **DIFD(14) b**: bit b on trong một chu kỳ khi điều kiện từ on sang off.

9.3.10 Lệnh JUMP JMP (04) VÀ JME (05)

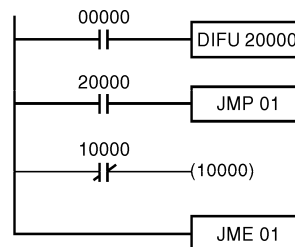
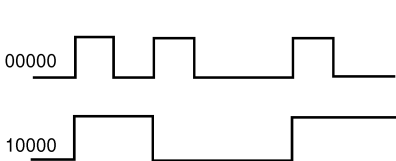
Lệnh JMP nn và JME nn đóng khung một đoạn chương trình. Nếu điều kiện cho lệnh JMP là on thì coi như không có lệnh JMP và chương trình thực hiện bình thường. Nếu điều kiện cho JMP off thì bỏ qua các lệnh trong khoảng JMP và JME nhưng vẫn giữ nguyên trạng thái các bit nhớ cũng như timer và counter, nn là số từ 00 đến 99., các số chỉ được dùng một lần trong chương trình, riêng lệnh JMP 00 có thể dùng nhiều lần với chỉ một lệnh JME 00.

Ví dụ: Khi T000 on đoạn chương trình giữa JMP 01 và JME 01 được thực hiện, bit 10000 và 10001 phụ thuộc điều kiện 00000 và 00001, khi T000 off trạng thái của bit 10000 và 10001 được giữ nguyên

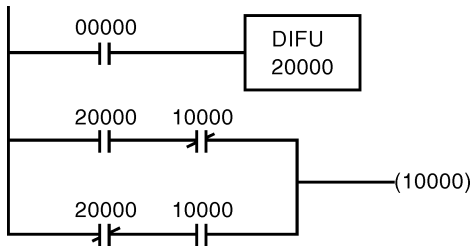


LD	T000
JMP(04)	#0001
LD	00004
OUT	10000
LD	00005
OUT	10001
JME(05)	#0001

Ví dụ:



Khi nhấn contact 00000 lệnh DIFU 20000 làm 20000 ON trong một chu kỳ quét do đó 10000 sẽ ON, đến chu kỳ quét sau 20000 OFF nên không thực hiện lệnh OUT 10000 mà giữ nguyên trạng thái của 10000. Khi nhấn 00000 lần nữa thì thực hiện lệnh OUT 10000 với điều kiện OFF do đó 10000 sẽ OFF.

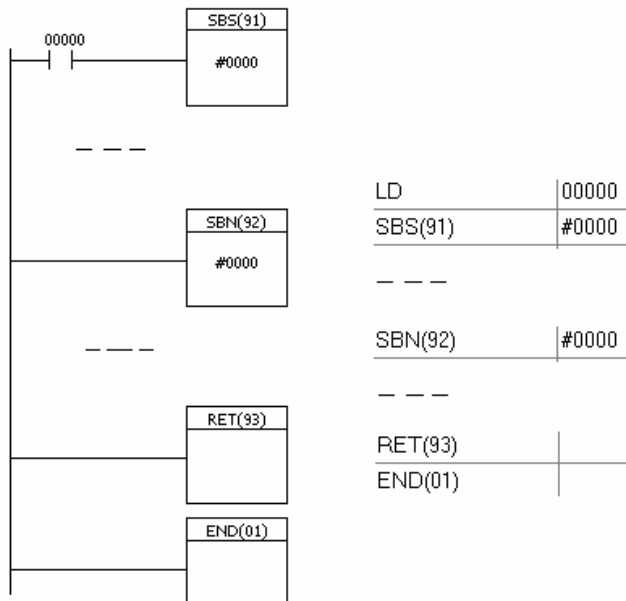


Nếu không dùng lệnh JMP JME thì có thể dùng chương trình sau:

9.3.11 Lệnh chương trình con SBS (91), SBN (92), RET (93)

Lệnh SBN nnn và RET đóng khung chương trình con còn lệnh SBS nnn dùng để gọi chương trình con. Mỗi chương trình con có một số hiệu từ 000 đến 255. Chương trình con được đặt ở đoạn cuối của chương trình chính, trước lệnh END. Các lệnh DIFU, DIFD không nên đặt trong chương trình con. Dùng chương trình con có ưu điểm là chương trình dễ đọc và thời gian thực hiện chương trình ngắn hơn.

Lệnh END (01) là lệnh cuối cùng của chương trình.



9.3.12 Địa chỉ gián tiếp

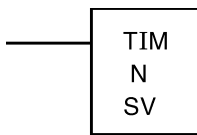
Địa chỉ gián tiếp được thực hiện qua vùng nhớ DM và ký hiệu là *DM. Từ nhớ *DM sẽ chứa địa chỉ của từ nhớ DM muốn sử dụng.

9.3.13 Lệnh vi phân

Lệnh vi phân ký hiệu bởi dấu @ đứng trước lệnh. Lệnh này chỉ thực hiện một lần khi điều kiện đi từ OFF sang ON.

9.4 CAUC LEANH NONGH THÌ VAØ ÑEÁM

9.4.1 Lệnh TIMER

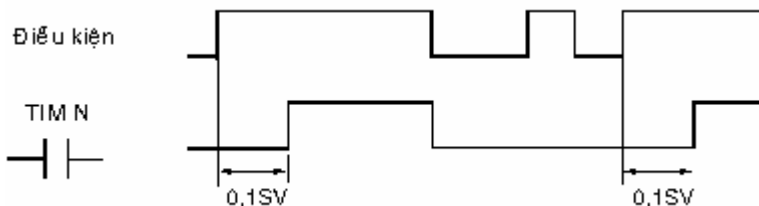


N- số từ 0 ÷ 511 tùy loại CPU

SV- giá trị đặt BCD, 0000 đến 9999 là nội dung ô nhớ :

IR, SR, AR, DM, HR, LR hay hằng số #

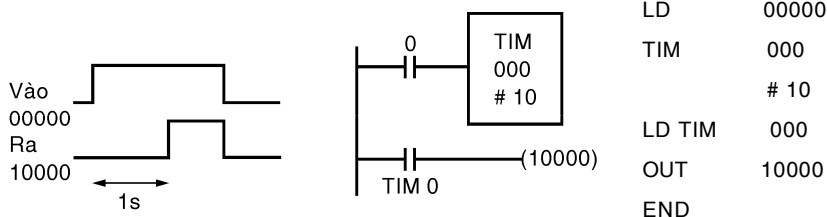
Khi điều kiện off mạch định thì reset về SV, cờ TIM N off, khi điều kiện on nội dung mạch định thì giảm cứ mỗi 0.1 giây; sau thời gian 0,1SV giây, cờ TIM N sẽ ON cho đến khi điều kiện OFF hay ngắt điện nguồn. Lệnh TIM dùng với IL sẽ reset khi điều kiện cho IL là OFF.



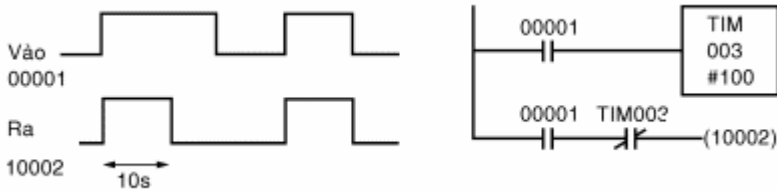
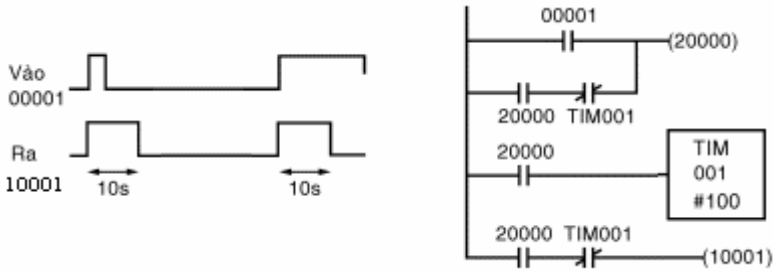
TIMH (15) cũng giống TIM nhưng thời gian trễ là 0,01SV và N trong khoảng 000 ÷ 015.

Sau đây là các áp dụng thường gặp của timer:

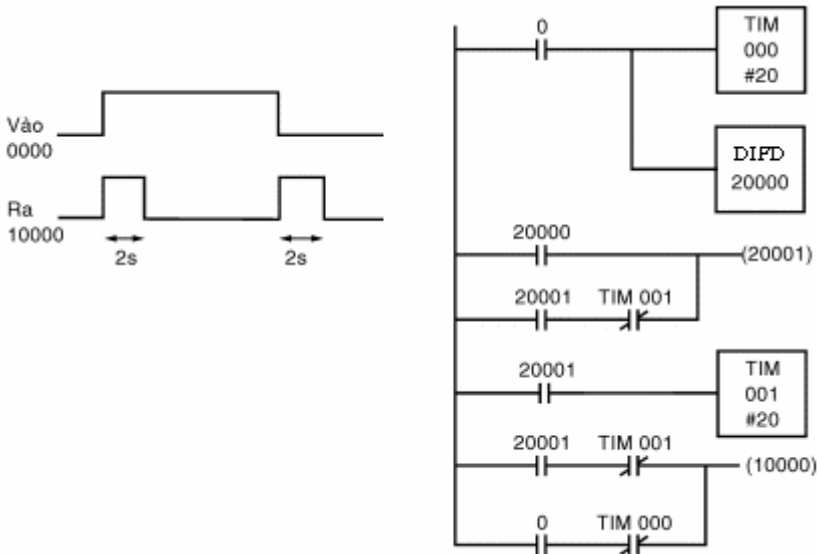
Mạch ON Delay

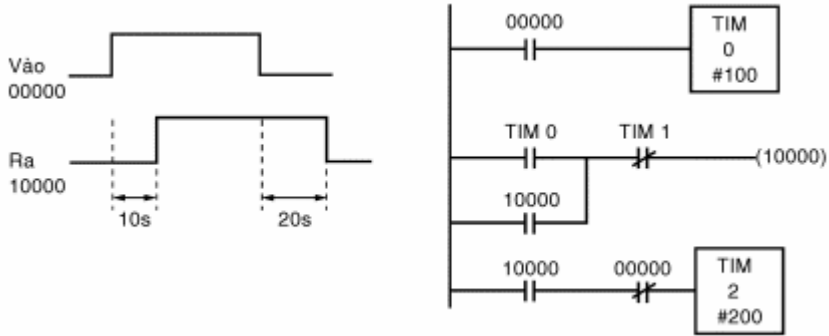


Mạch đơn ổn

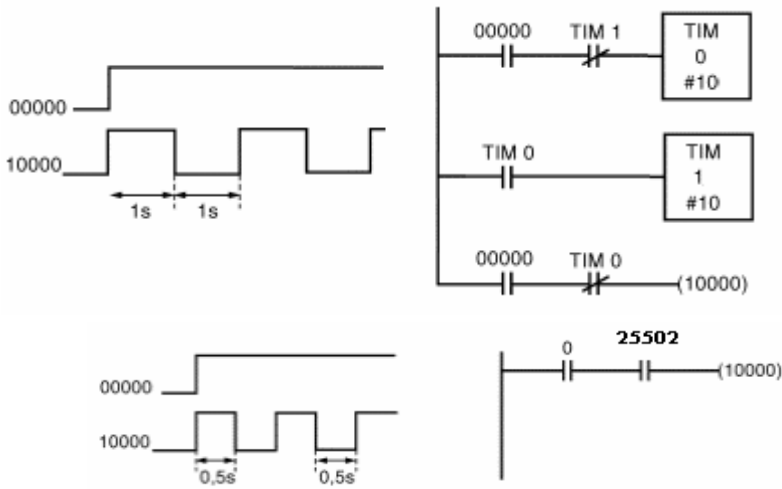


Mạch ON/OFF delay

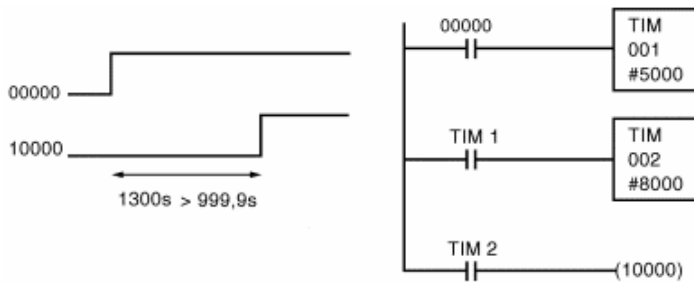




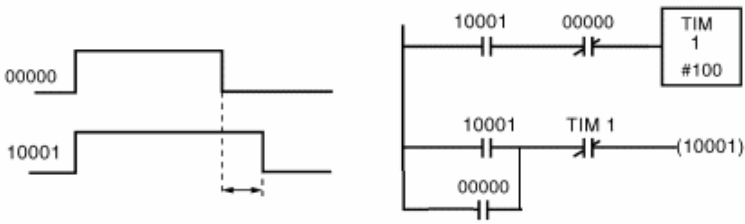
Mạch nhấp nháy



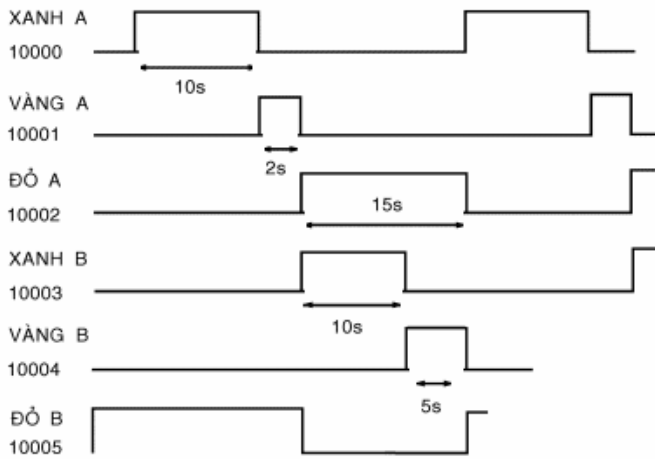
Mạch định thì dài



Mạch OFF delay

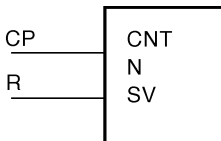


Đèn giao thông





9.4.2 Lệnh đếm CNT



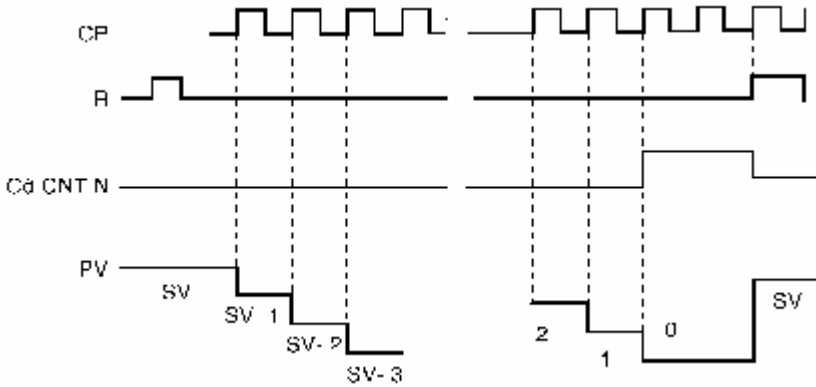
N: số từ 0 đến 511 tùy loại CPU

SV: trị đặt cho bộ đếm (BCD):0000..9999

IR, SR, AR, DM, HR, LR, #

R là ngõ vào xóa, khi R từ OFF sang ON nội dung PV của bộ đếm được đặt ở SV. Khi R trở về OFF, bộ đếm bắt đầu hoạt động, khi có xung CP từ OFF sang ON, PV sẽ giảm đi 1. PV không thay đổi khi CP từ ON sang OFF. Khi PV = 0 thì nội dung của CNT giữ nguyên ở 0, cờ CNT N sẽ ON và giữ nguyên ở ON cho đến khi

được reset bởi R.



Nội dung CNT không bị xóa khi ngắt nguồn hay trong đoạn chương trình IL.

Chú ý là trong PLC có sẵn một số bit đặc biệt sau:

25400: xung nhịp chu kỳ 1 phút; 25401: 0.02 sec

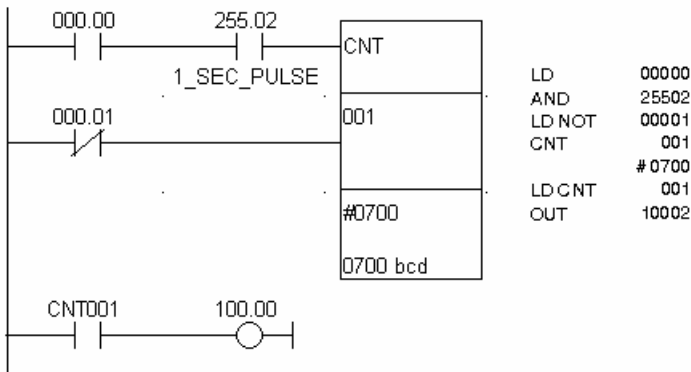
25500: 0.1 sec; 25501: 0.2 sec; 25502: 1.0 sec

25313: cờ luôn luôn on; 25314: cờ luôn luôn off

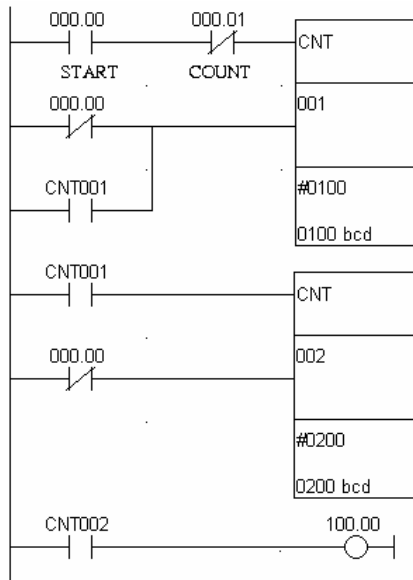
25315: cờ on ở chu kỳ đầu.

Ví dụ:

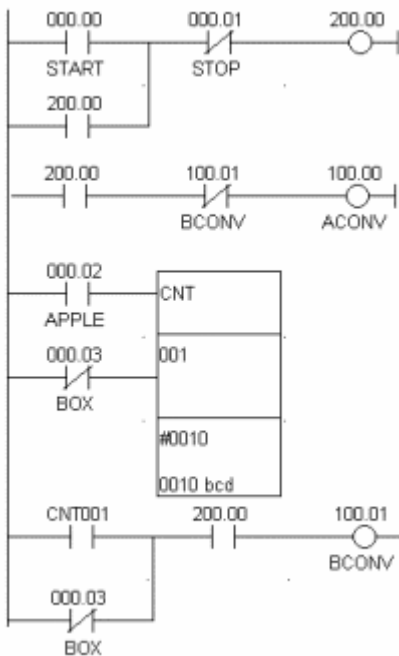
Mạch định thời 11'40"



Mạch đếm số lượng lớn 20000 xung

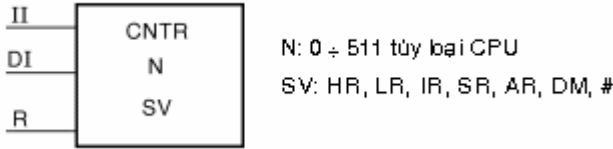


Mạch đóng gói: đóng gói 10 quả táo cho vào hộp



Cảm biến táo: 00002
 Cảm biến hộp: 00003
 Bảng chuyển táo: 10000
 Bảng chuyển hộp: 10001
 Khi bấm START, nếu không có hộp chỗ cảm biến thì bảng chuyển hộp chạy, đưa hộp vào vị trí nhận táo, lúc đó bảng chuyển hộp ngừng, bảng chuyển táo chạy, đưa táo vào hộp. Đếm đủ 10 quả táo bảng chuyển hộp chạy, bộ đếm reset khi hộp rời khỏi vị trí, bảng chuyển táo ngừng chờ hộp mới vào vị trí.

9.4.3 Đếm thuận nghịch CNTR (12)

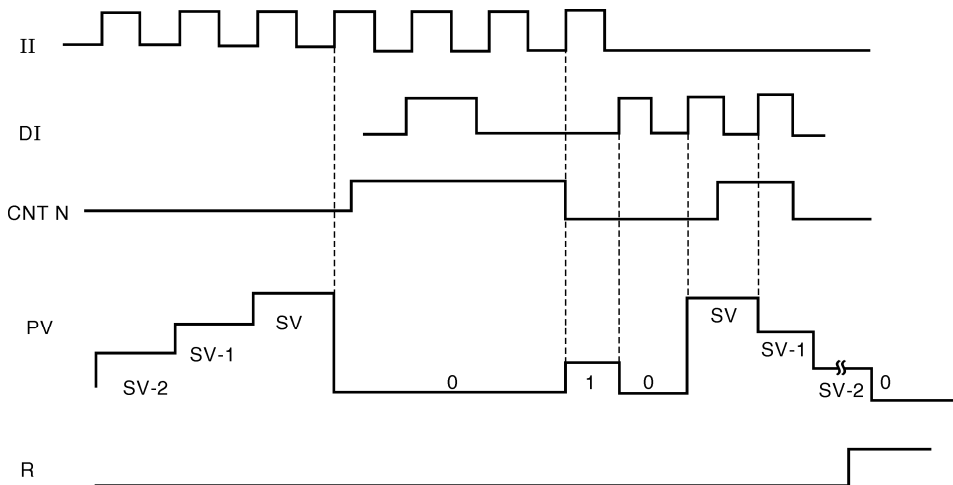


Khi R từ OFF sang ON bộ đếm được xoá về 0, PV= 0. Khi R OFF bộ đếm chuẩn bị đếm.

Khi II từ OFF sang ON thì PV tăng lên 1.

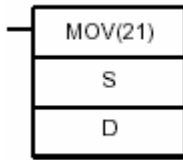
Khi DI từ OFF sang ON thì PV giảm 1.

Nếu II và DI cùng lúc từ OFF sang ON thì PV không thay đổi. Khi giảm từ 0, PV sẽ bằng SV và contact CNT N sẽ ON cho đến khi PV giảm. Khi tăng quá SV, PV sẽ bằng 0 và CNT N sẽ ON cho đến khi PV tăng. PV không bị ảnh hưởng bởi ngắt nguồn hay IL.



9.5 LỆNH DI CHUYỂN

9.5.1 Di chuyển ô nhớ

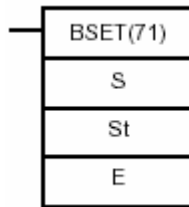


MOV (21) chuyển một từ từ S đến D

S: IR, SR, AR, DM, HR, TC, LR, #

D: IR, SR, AR, DM, HR, LR.

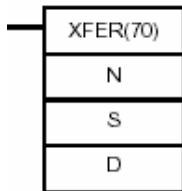
MVN (22) chép đảo của từ S sang D.



BSET chép một từ S sang một khối nhớ từ St đến E

S: IR, SR, AR, DM, HR, TC, LR, #

St, E: IR, SR, AR, DM, HR, TC, LR.

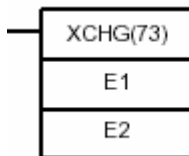


XFER chép một khối N ở nhớ bắt đầu từ S

sang khối N ở nhớ bắt đầu từ D

N: IR, SR, AR, DM, TC, LR, #

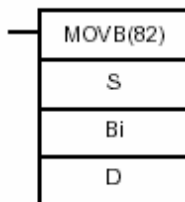
S, D: IR, SR, AR, DM, HR, TC, LR.



XCHG trao đổi 2 ô nhớ E1, E2

E1, E2: IR, SR, AR, DM, HR, TC, LR.

9.5.2 Di chuyển một số bit



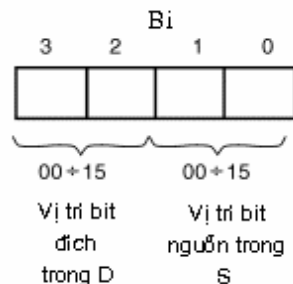
MOVB chép một bit của từ S sang một bit của từ D.

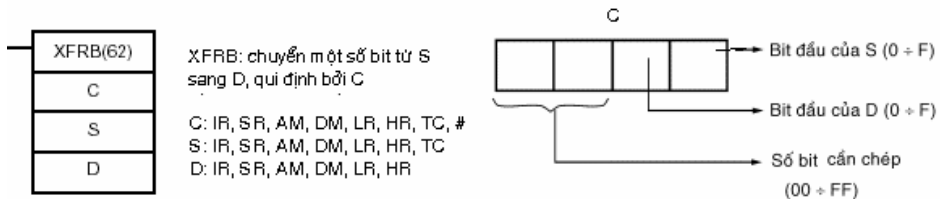
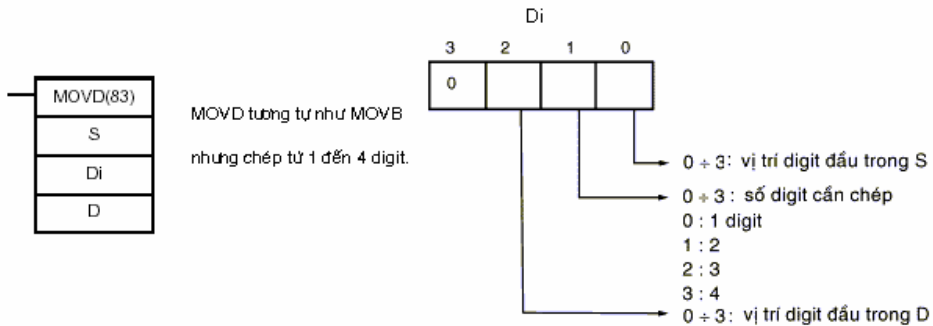
Vị trí bit chỉ bởi Bi

S: IR, SR, AR, DM, HR, LR, #

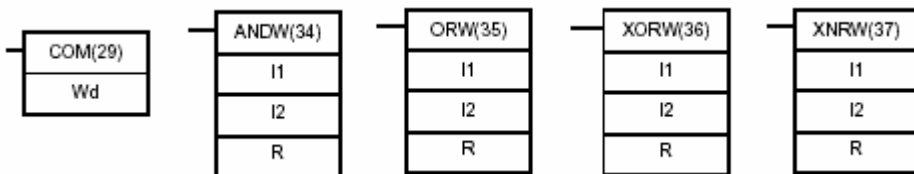
D: IR, SR, AR, DM, HR, LR.

Bi: IR, SR, AR, DM, HR, TC, LR, #





9.6 LỆNH LOGIC

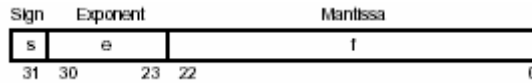


COM: đảo các bit của Wd; **ANDW:** AND hai từ; **ORW:** OR hai từ; **XORW:** XOR hai từ; **XNRW:** Exclusive Nor hai từ

9.7 LỆNH SOÁ HOÏC

9.7.1 Các loại số: PLC OMRON tính toán chủ yếu trên số thập phân BCD 4 hay 8 digit không dấu, số nhị phân có dấu và không dấu 16 bit, 32 bit. Số nhị phân không dấu 16 bit từ 0000 (0) đến FFFF (65,535), 32 bit từ 00000000 đến FFFFFFFF (4,294,967,295). Số nhị phân có dấu 16 bit dùng mã bù hai, bit 15 là bit dấu, từ 8000 (-32,768) đến FFFF (-1) và 0000 (0) đến 7FFF (32767). Số nhị phân có dấu 32 bit có giá trị từ 80000000 (-2,147,483,648) đến FFFFFFFF (-1) và 00000000 (0) đến 7FFFFFFF (2,147,483,647).

Trong một số trường hợp sử dụng số chấm nổi (số thực), chiếm 32 bit biểu thị bằng dấu s, số mũ e và định trị f: $(-1)^s 2^{e-127} (1 + f * 2^{-23})$

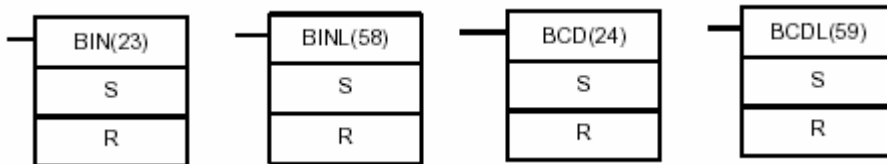


Các cờ hiệu liên quan lệnh số học là:

- N: cờ âm 25402
- OF: cờ tràn dương 25404
- UF: cờ tràn âm 25405
- ER: lệnh sai 25503
- CY: cờ nhớ 25504
- GR: cờ nhỏ hơn 25505
- EQ: cờ bằng 25506
- LE: cờ lớn hơn 25507

Cờ CY được set/reset bởi lệnh **STC/CTC**

9.7.2 Lệnh đổi dữ liệu BCD- Nhị phân



BIN: đổi số BCD 4 digit trong S ra số nhị phân trong R

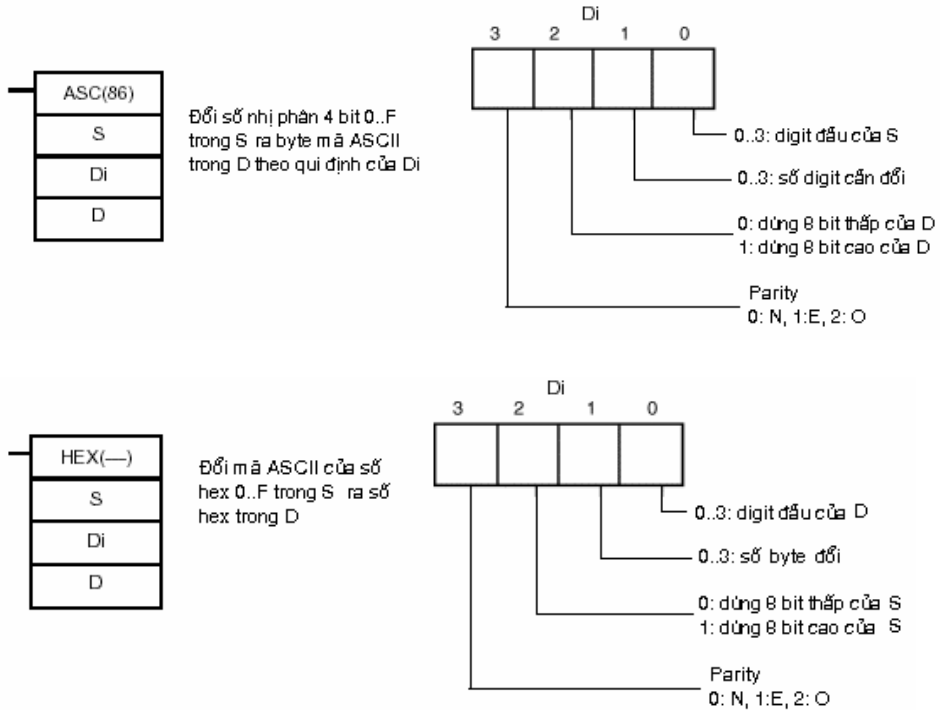
S: IR, SR, AR, DM, HR, TC, LR;

R: IR, SR, AR, DM, HR, LR

BCD: đổi số nhị phân 16 bit trong S ra số BCD trong R, nếu kết quả lớn hơn 9999 thì cờ ER báo và R không đổi

Các lệnh **BINL** và **BCDL** dùng cho từ kép

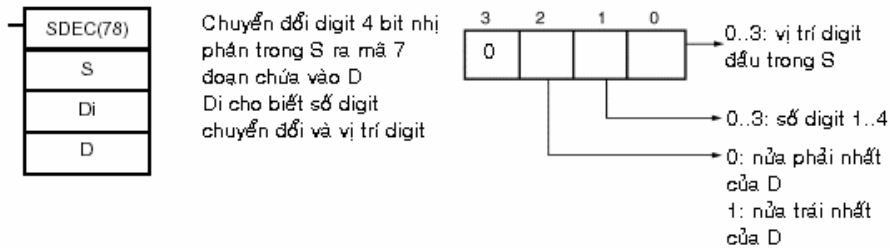
9.7.3 Lệnh đổi HEX- ASCII



9.7.4 Lệnh đổi số thực- số nguyên

- Số thực – số nguyên 16 bit FIX
- Số thực – số nguyên 16 bit FIXL
- Số nguyên 16 bit – số thực FLT
- Số nguyên 16 bit – số thực FLTL

9.7.5 Giải mã 7 đoạn



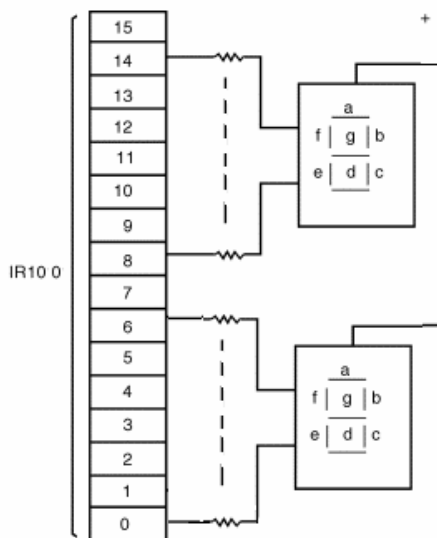
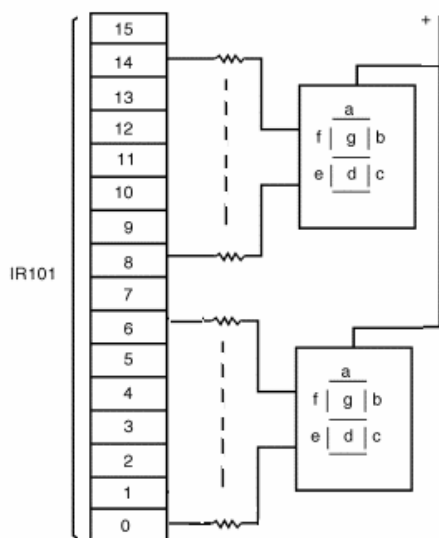
Ví dụ, muốn hiển thị nội dung ô nhớ DM0000 ra 4 đèn 7 đoạn gắn ở module xuất IR100 và IR101, ta dùng đèn 7 đoạn anode chung, nguồn cấp từ 5V đến 24V, điện trở nối tiếp tùy theo áp nguồn. Lệnh SDEC DM0000 #0030 100

9.7.6 Lệnh BCD không dấu 4 số thập phân



Cộng hai số dạng BCD Au và Ad với bit nhớ CY, kết quả chứa vào R.

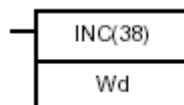
Nếu có tràn (quá 9999) thì bit nhớ CY là ON.



Trừ hai số BCD Mi, Su với bit nhớ CY, kết quả chứa vào R, nếu kết quả âm thì CY là ON và R chứa kết quả phụ 10,

muyến có kết quả đúng lấy 0 trừ số này.

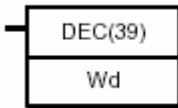
Khi tính toán phải xét CY để chỉnh lại kết quả.



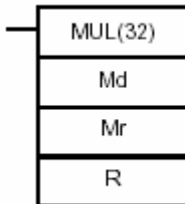
Tăng nội dung BCD của Wd lên 1

IR, SR, HR, LR, AR, DM

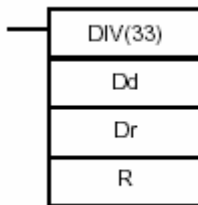
Cờ EQ on khi kết quả là 0



Giảm Wd đi 1
cờ EQ tác động khi kết quả là 0



Nhân hai số Md và Mr, kết quả chứa trong R+1 và R.
Nếu có nhớ thì CY = 1.
Md, Mr: IR, HR, LR, AR, SR, DM, TC, #
R: IR, HR, LR, AR, SR, DM

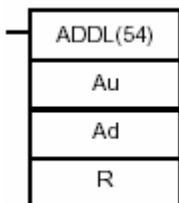


Chia Dd cho Dr, kết quả
chứa trong R, dư số trong
R+1

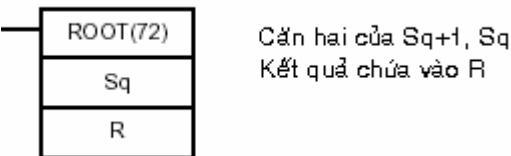
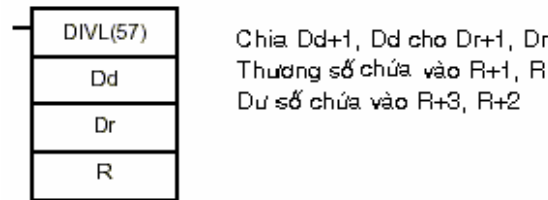
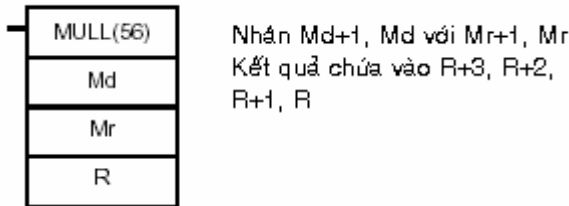
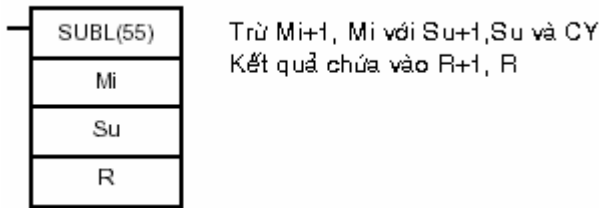
Ví dụ: Trừ ô nhớ HR20 với ô nhớ 120, kết quả chứa vào DM0100

```
LD      00003
CLC(41)
@SUBL(55)      HR 20      120      DM 0100
AND      25504
@BSET(71)      # 0000      DM 0000DM 0001
CLC(41)
@SUBL(55)      DM 0000      DM 0100DM 0100
```

9.7.7 Lệnh BCD không dấu 8 digit



Cộng Au+1, Au với Ad+1, Ad và CY
Kết quả chứa vào R+1, R
Au, Ad: IR, SR, AR, LR, DM, HR, TC
R: IR, SR, AR, LR, DM, HR



:

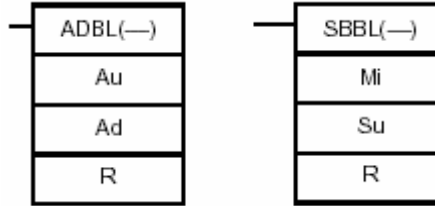
9.7.8 Lệnh số học nhị phân 16 bit không dấu



ADB: CY ON: kết quả quá FFFF, OF ON: kết quả quá 7FFF, UF ON: kết quả nhỏ hơn 8000, EQ ON: kết quả 0, N ON: bit 15 kết quả là 1. Sử dụng các cờ OF, UF khi muốn cộng trừ số có dấu.

SBB: CY ON khi $Mi < Su + CY$ (số không dấu), các cờ khác giống lệnh ADB

9.7.9 Lệnh số học nhị phân 32 bit không dấu



ADBL: CY ON: kết quả quá FFFFFFFF, OF ON: kết quả quá 7FFFFFFF, UF ON: kết quả nhỏ hơn 80000000, EQ ON: kết quả 0, N ON: bit 15 R+1 là 1. Sử dụng các cờ OF, UF khi muốn cộng trừ số có dấu.

SBBL: CY ON khi $Mi < Su + CY$ (số không dấu), các cờ khác giống lệnh ADB.

Lệnh nhân chia nhị phân có dấu

MBS: nhân 16 bit có dấu, N ON khi bit 15 của R+1 ON

MBSL: nhân 32 bit có dấu, N ON khi bit 15 của R+3 ON

DBS: chia 16 bit có dấu, N ON khi bit 15 của R ON

DBSL: chia 32 bit có dấu, N ON khi bit 15 của R+1 ON

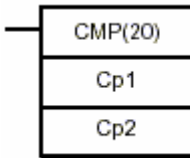


9.7.10 Lệnh số thực

Cộng	+F	Tang	TAN
Trừ	-F	Cung sin	ASIN
Nhân	*F	Cung cos	ACOS
Chia	/F	Cung tang	ATAN
Đổi độ ra rad	RAD	Căn hai	SQRT
Đổi rad ra độ	DEG	Mũ	EXP
Sin	SIN	Log	LOG
Cosin	COS		

9.8 LỆNH SO SÁNH

9.8.1 So sánh hai ô nhớ



So sánh hai từ nhị phân không dấu Cp1 và Cp2
 Cp1, Cp2: IR, SR, AR, HR, TC, LR, #
 Khi so sánh với PV của Timer/Counter giá trị là số BCD

Bit	Cp1<Cp2	Cp1=Cp2	Cp1>Cp2
GR 25505	OFF	OFF	ON
EQ 25506	OFF	ON	OFF
LE 25507	ON	OFF	OFF

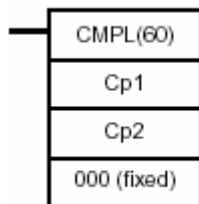
Kết quả so sánh phải đặt liền sau lệnh CMP để bảo đảm giá trị vì trong chương trình có thể có nhiều lệnh so sánh.

Vi dụ: so sánh nếu #1000 < (DM0000) < #2000 thì 10000 ON

LD	00000	AND	20000
CMP	DM0000 #1000	CMP	DM0000 #2000
AND	25505	AND	25507
OUT	20000	OUT	10000
LD	00000		

Vi dụ: tạo các ngõ ra lần lượt ở các thời điểm 200, 300, 500s

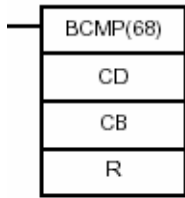
LD	00000	CMP(20)	TIM 010 # 2000
TIM	010 # 5000	AND	25507
CMP(20)	TIM 010 # 3000	OUT	10001
AND	25507	LD	TIM 010
OUT	10000	OUT	10002
LD	10000		



So sánh hai từ kép Cp1+1, Cp1 và Cp2+1, Cp2

Muốn so sánh số nhị phân có dấu, dùng lệnh CPS và CPSL

9.8.2 So sánh khoảng



So sánh số CD với khối 32 ô nhớ bắt đầu từ CB, kết quả chứa trong R

$CB \leq CD \leq CB + 1$: bit 0 của R ON

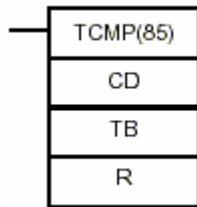
$CB + 1 \leq CD \leq CB + i + 1$: bit i của R ON

$CB + 30 \leq CD \leq CB + 31$: bit 15 của R ON

CD: IR, SR, DM, HR, TC, LR, AR, #; CB: IR, SR, DM, HR, TC, LR

R: IR, SR, DM, HR, TC, LR, AR

9.8.3 So sánh bằng

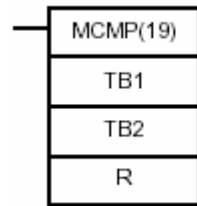


So sánh CD với 16 ô nhớ bắt đầu từ TB, kết quả so sánh chứa vào 16 bit của R. Ví dụ nếu CD bằng $TB + i$ thì bit i của R là ON.

TB, R: IR, SR, DM, HR, TC, LR

CD: IR, SR, DM, HR, TC, LR, #

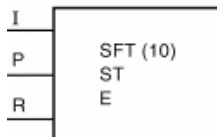
9.8.4 So sánh hai bằng



So sánh hai bằng TB1 và TB2 dài 16 từ. Nếu hai từ thứ i bằng nhau thì bit thứ i của R OFF

9.9 LỆNH GHI DỜI

9.9.1 Dời trái



Thanh ghi dời gồm một loạt từ nhớ 16 bit bắt đầu ở St và chấm dứt ở E.

Khi R ON mọi bit của thanh ghi ở trạng thái OFF.

Khi R OFF thanh ghi chuẩn bị hoạt động.

Khi P từ OFF sang ON trạng thái ở ngõ vào I chứa vào bit 0 của St, mọi bit trước của thanh ghi được dời về bên trái, bit cao nhất của E mất đi.
St, E: IR, SR, AR, HR, LR.

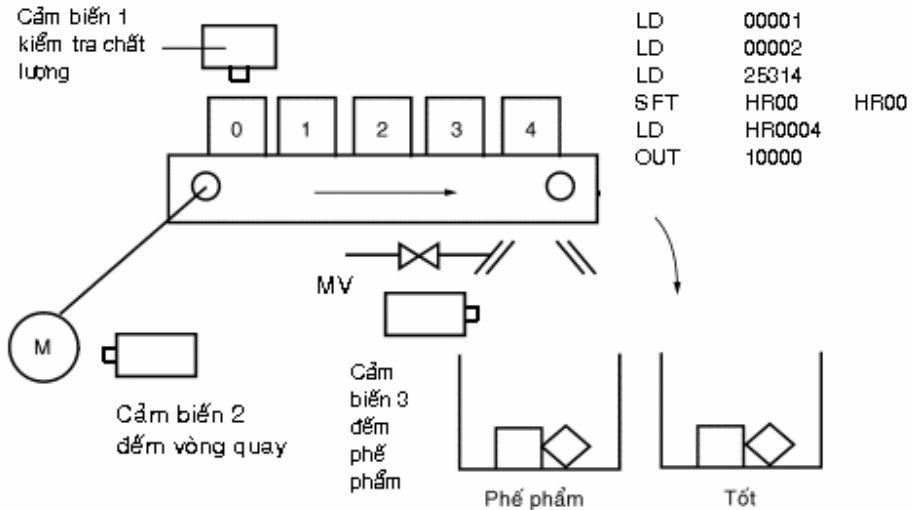
Ví dụ: chương trình phát hiện và loại bỏ phế phẩm.

Cảm biến 1, sẽ phát tín hiệu báo khi phát hiện phế phẩm và

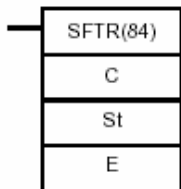
đưa vào thanh ghi (tín hiệu I)

Cảm biến 2, phát một xung mỗi khi có một sản phẩm mới vào băng chuyền dùng làm xung nhịp cho thanh ghi (tín hiệu P)

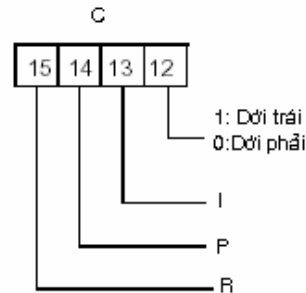
Khi phế phẩm đến vị trí số 3 (4 xung kể từ khi cảm biến 1 báo) sẽ được đẩy vào thùng chứa phế phẩm bởi van từ MV.



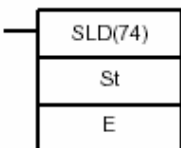
9.9.2 Dời thuận ngược



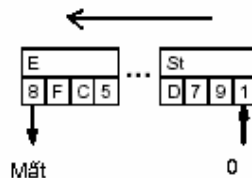
Ghi dời thuận nghịch, chiều dời qui định bởi C
 C: IR, AR, LR, HR, DM
 St, E: IR, SR, AR, LR, HR, DM



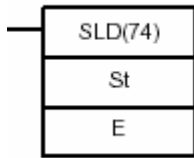
9.9.3 Dời digit sang trái



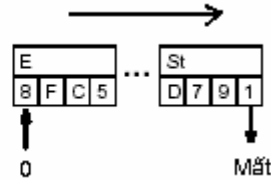
Khi thực hiện lệnh, digit được dời sang trái, 0 đưa vào digit phải nhất của St



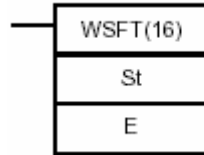
9.9.4 Dời digit sang phải



Khi thực hiện lệnh, digit được dời sang phải 0 đưa vào digit trái nhất của E

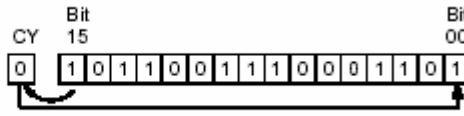
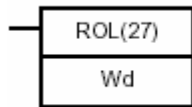


9.9.5 Dời từ

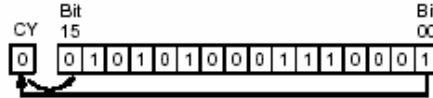
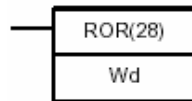


Dời một từ sang trái, 0000 đưa vào St

9.9.6 Quay trái



9.9.7 Quay phải



9.10 CHỨC NĂNG NGẮT (CQM1)

Chức năng ngắt cho phép PLC nhảy đến chương trình phục vụ ngắt khi có yêu cầu mà không cần phải thường xuyên đọc trạng thái ngõ vào.

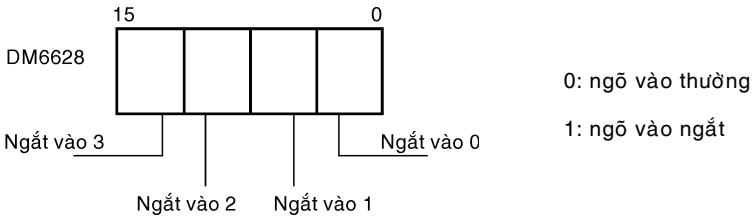
Có ba loại ngắt:

- Ngắt ngõ vào
- Ngắt thời gian
- Ngắt bộ đếm vận tốc cao

Ngắt ngõ vào: có bốn ngõ vào ngắt theo ưu tiên sau:

Ngắt vào 0 (IR00000) > ngắt vào 1 (IR00001) > ngắt vào 2 (IR00002) > ngắt vào 3 (IR00003)

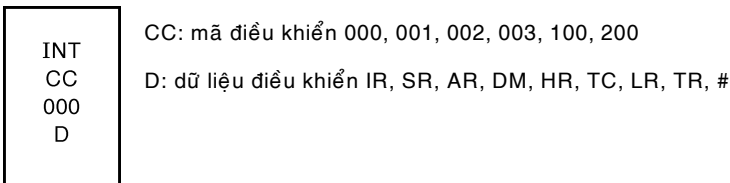
Các ngắt này gọi các hàm con theo thứ tự SBN 000 ÷ SBN 003. Muốn sử dụng ngắt ngõ vào phải đặt nội dung DM 6628



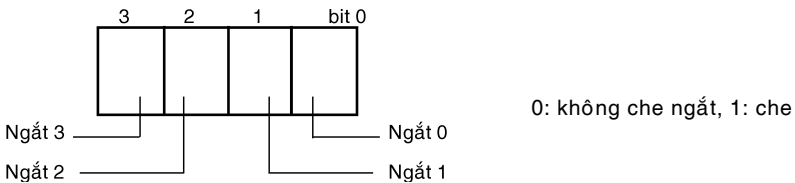
Vi dụ, muốn dùng ngõ vào IR00000 IR00001 làm ngõ vào ngắt thì đặt (DM 6628) = 0011

Ngắt có thể che hay không che với lệnh INT (89)

Lệnh này có dạng tổng quát

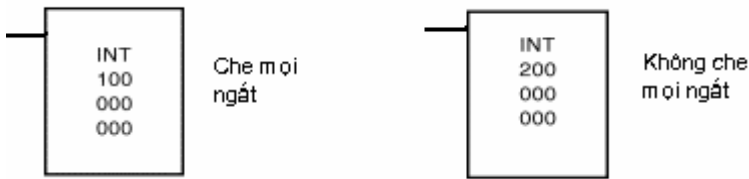


Muốn che ngắt hay không che dùng CC= 000 và D có dạng sau:



Khi bị che, tác động của ngõ vào ngắt được ghi lại nhưng không thực hiện, khi xóa che sẽ lập tức nhảy đến chương trình con phục vụ ngắt, trừ khi nó được xóa ngắt bởi CC = 001 và bit tương ứng của D là 1.

Đọc trạng thái che hay không che với CC = 002, bit tương ứng trong D sẽ ON nếu bị che.

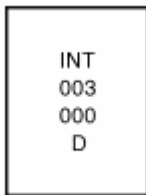


Có thể đặt ngõ vào ngắt ở chế độ đếm, nghĩa là ngắt chỉ xảy ra sau khi có một số lượng xung ở ngõ vào ngắt. Số lượng xung đặt trước ở các địa chỉ sau:

Ngắt 0	SR 244	Ngắt 2	SR 246
Ngắt 1	SR 245	Ngắt 3	SR 247

Nếu nội dung các ô nhớ trên là 0 thì ở chế độ ngắt thường, nội dung ô nhớ phải từ 0001 đến FFFF để ở chế độ ngắt đếm. Tần số xung đếm tối đa 1 KHz.

Sau khi đặt giá trị cho các ô nhớ trên dùng lệnh INT để cho phép ngắt đếm hoạt động



Nếu bit tương ứng trong D là 0 thì hoạt động ở chế độ đếm và cho phép ngắt, nếu là 1 thì không tác động. Khi có một tín hiệu ngắt vào bộ đếm sẽ tăng lên 1 và khi bằng trị đặt thì gây ra ngắt.

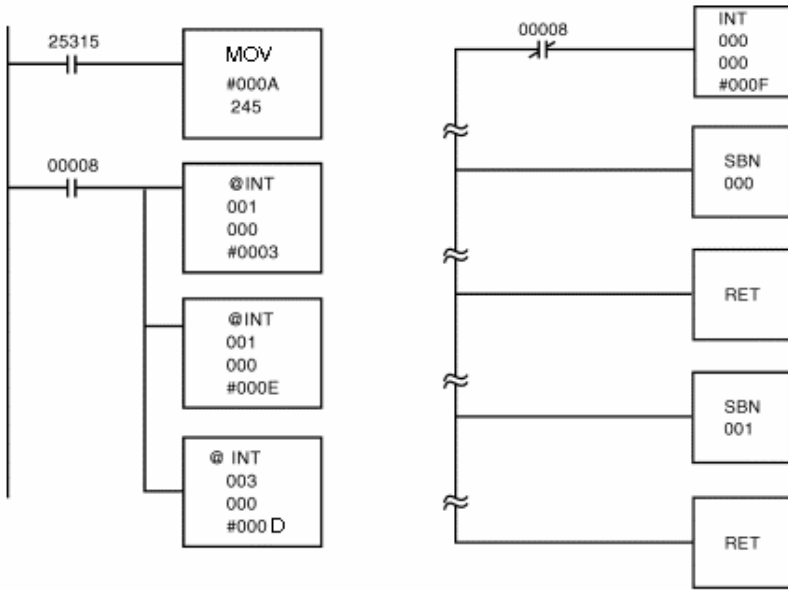
Bộ đếm sử dụng các ô nhớ sau:

Ngắt 0	SR 248	Ngắt 2	SR 250
Ngắt 1	SR 249	Ngắt 3	SR 251

Nội dung ô nhớ là nội dung bộ đếm trừ đi 1

Ví dụ: dùng ngắt 0 chế độ ngắt ngõ vào và ngắt 1 chế độ ngắt đếm. Đặt DM 6628 : 0011

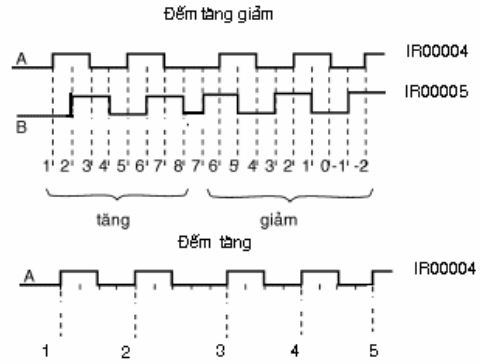
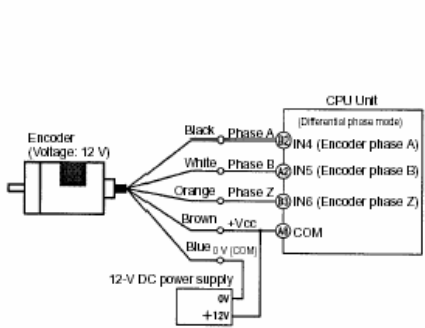
Lập trình cho PLC như sau:



Ngắt đếm v

Bộ đếm vận tốc cao (HSC High speed counter) được dùng để đếm số xung từ encoder gia số. theo chế độ ngắt. PLC có thể có nhiều HSC, HSC0 được lắp sẵn trên PLC. Dùng bo mở rộng để thêm HSC.

Sử dụng HSC0 ở chế độ đếm tăng hay đếm tăng giảm, khi dùng đếm tăng giảm tần số xung đếm tăng 4 lần so với xung thực tế, điều này tăng độ phân giải của encoder, xung A, B và Z của encoder đưa vào các ngõ IR 00004 ÷ IR 00006. Tần số xung vào khi đếm tăng giảm là 2.5KHz và 5KHz khi đếm tăng.



Nội dung HSC0 chứa trong hai từ SR 231 và SR 230:

SR 231	SR 230
--------	--------

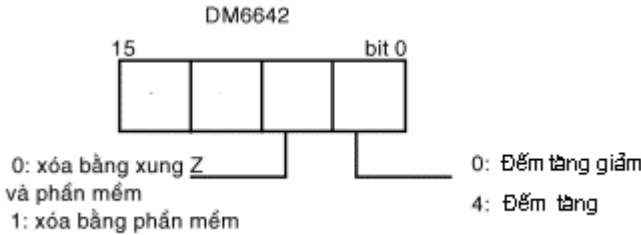
Đếm tăng giảm

Đếm tăng

F 0032768 ÷ 00032767
(- 32767)

00000000 ÷ 00065535

HSC0 được khởi động bằng cách đặt cấu hình



Muốn xóa bộ đếm dùng một trong hai phương pháp:

- Xóa phần mềm: cho SR 25200 ON

- Xóa bằng phần mềm và xung Z: khi xung Z ON và SR 25200 ON. HSC0 được xóa khi cấp nguồn hay khi bắt đầu hoạt động.

Muốn đọc nội dung PV của bộ đếm, ta đọc nội dung hai ô nhớ SR 231, SR 230 hoặc dùng lệnh PRV 000 000 P1, nội dung SR231 và SR230 sẽ chứa vào P1+1 và P1.

Gọi chương trình phục vụ ngắt bằng lệnh so sánh bảng CTBL 000 C TB,

C= 000: so sánh HSC0 với các giá trị BCD 8 digit ghi trong bảng TB, nếu bằng thì gọi một trong các chương trình con (tối đa 16 giá trị) số 0 đến 255

C= 001 so sánh HSC0 với tối đa 8 khoảng, mỗi khoảng có giới hạn dưới và trên

TB: địa chỉ đầu của bảng

TB	Số giá trị
TB+1	Giá trị so sánh số 1, 4 digit thấp
TB+2	Giá trị so sánh số 1, 4 digit cao
TB+3	Số chương trình con
TB+4	Tương tự cho giá trị số 2

TB	Giới hạn dưới số 1, 4 digit thấp
TB+2	Giới hạn dưới số 1, 4 digit cao
TB+3	Giới hạn trên số 1, 4 digit thấp
TB+4	Giới hạn trên số 1, 4 digit cao
TB+5	Số chương trình con
TB+6	Tương tự cho tầm số 2

Ví dụ: so sánh giá trị HSC0 với 1000 và 2000, gọi các chương

trình con 101 và 102

Ta đặt nội dung các ô nhớ

DM6642: 0114	DM0003: 0101
DM0000: 0002	DM0004: 2000
DM0001: 1000	DM0005: 0000
DM0002: 0000	DM0006: 0102

và viết chương trình

```
LD      25315
CTBL   000      000      DM0000
.....
SBN    101
....
RET
SBN    102
...
RET
```

Ngắt thời gian

Có ba ngắt thời gian 0, 1 và 2 được điều khiển nhờ lệnh STIM với hai chế độ hoạt động:

- Ngắt đơn ổn: gọi chương trình con một thời gian sau khi STIM hoạt động.
- Ngắt chu kỳ: gọi chương trình con theo chu kỳ cách nhau một khoảng thời gian.

Lệnh STIM (69) có dạng sau STIM C1 C2 C3. Từ điều khiển C1 dùng để chọn chế độ khởi động, ngừng và đọc giá trị của timer:

Chức năng	Timer	C ₁
Khởi động chế độ đơn ổn	0	000
	1	001
	2	002
Khởi động chế độ chu kỳ	0	003
	1	004
	2	005
Ngừng timer	0	010
	1	011
	2	012
Đọc trị PV của timer	0	006
	1	007
	2	008

Chế độ đơn ổn: C1 đặt như trên, C2+1 chứa đơn vị thời gian từ 0005 đến 0320 (0,5ms đến 32ms) và C2 chứa số lần đếm của đơn vị thời gian.

Như vậy khoảng thời gian từ lúc STIM thực hiện đến khi gọi chương trình con ngắt là:

$$(C2) * (C2+1) * 0.1ms = 0,5ms \text{ đến } 319.968 \text{ ms}$$

Nếu C2 là hằng số thì thời gian timer là số đó \times 1ms

C3: từ 0000 đến 0255 là số chương trình con phục vụ

Chế độ chu kỳ: C2; C2+1; C3: như chế độ đơn ổn

Chế độ ngừng: C2 = 000; C3 = 000

Khi chương trình con được gọi, timer tự reset và hoạt động trở lại.

Chế độ đọc thời gian hiện tại của timer:

C2: chỉ số lần bộ đếm đơn vị thời gian đã giảm

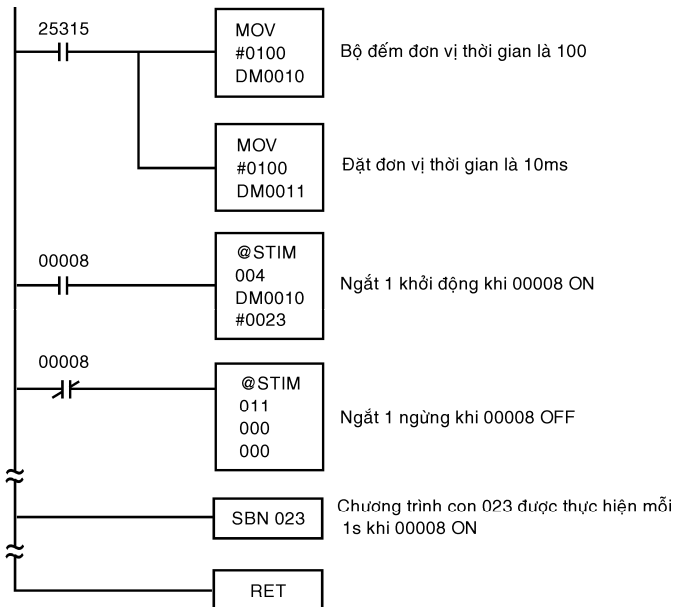
C2+1: chứa khoảng thời gian trong đơn vị thời gian

C3: chứa địa chỉ ô nhớ nhận thông số thời gian đã trôi qua từ lần giảm trước.

Thời gian tổng cộng là: $[(C2) * (C2+1) + C3] * 0.1ms$

Ngắt timer 2 không dùng khi HSC0 đã dùng. Ngắt timer 0 không dùng khi sử dụng SPED phát xung

Vi dụ: dùng ngắt thời gian 1 theo kiểu chu kỳ cứ 1s gọi chương trình con số 23 một lần.



9.11 XỬ LÝ ANALOG

Có các module AD DA và các lệnh dành riêng cho việc điều khiển.

Khối nhập analog 12 bit CQM1 - AD041

Module này có 4 ngõ vào điện áp hay dòng điện chiếm 4 ô nhớ, có địa chỉ n đến n + 3, n tùy thuộc vị trí gắn module và loại PLC.

Điện áp vào tối đa +- 15V,

Dòng vào tối đa +- 30mA,

Chọn chế độ nhờ Dip Switch

Thời gian chuyển đổi 2.5msec/kênh

Độ chính xác 1%

Khi chọn tầm +-10V trong ô nhớ sẽ chứa số F830÷ 07D0H (-2000 ÷ +2000)

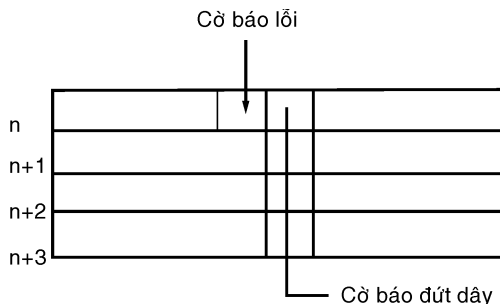
0V ÷ +10V tầm đổi 0030 ÷ 0FD0H (0048 ÷ 4048)

1V ÷ 5V hay 4mA ÷ 20mA số đổi là 0030 ÷ 0FD0.

Tổng trở nhập áp 1MΩ max, dòng 250Ω

Trường hợp đặt ở chế độ lấy trung bình sẽ lấy 8 trị số đổi rồi lấy trung bình, chu kỳ lấy trung bình ~ 72ms.

Trường hợp đặt tầm đổi 1V ÷ 5V (4mA ÷ 20mA) khi tín hiệu vào < 0,95V (hay dòng nhỏ hơn 4mA) sẽ báo đứt dây ở bit 12 của mỗi ô nhớ . Khi có lỗi thì báo bằng bit 13 của từ nhớ đầu.



Khối xuất analog CQM1 - DA021

Module có hai ngõ ra áp và hai ngõ ra dòng vi sai, thời gian

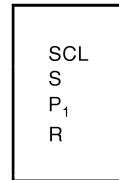
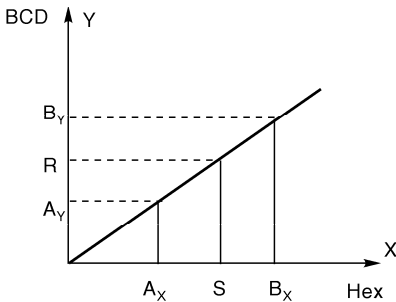
đổi 0,5ms/2 điểm. Đổi 11 bit ra điện áp hay dòng, mỗi ngõ ra chiếm một ô nhớ.

$$\begin{aligned} 0000 \div 07FF &\rightarrow 0V \div 10V & 0mA \div 20mA \\ F800 \div 07FF & & - 10V \div 10V \end{aligned}$$

Các lệnh liên quan đến tín hiệu analog và điều khiển quá trình

Tỷ lệ:	SCL (66)	Tạo hàm:	APR (-)
Tìm tối đa:	MAX (-)	Điều khiển PID:	PID
Tìm tối thiểu:	MIN (-)	Tạo xung:	PULS
Trị trung bình	AVG (-)	Tạo xung	SPED
Lấy tổng	SUM (-)	Tạo xung tần số thay đổi	PLS2
Điều khiển gia tốc	ACL	Tạo xung bề rộng thay đổi	PWM

Tỷ lệ SCL: đổi số nhị phân 4 digit sang số BCD 4 digit với tỷ lệ khác.



Đổi giá trị hex trong S sang BCD trong R theo quan hệ xác định bởi:
(P₁, P₁+1)
và: (P₁+2, P₁+3)

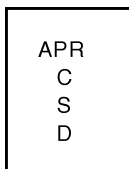
- P₁: Tọa độ Y thứ nhất A_Y
- P₁+1: Tọa độ X thứ nhất A_X
- P₁+2: Tọa độ Y thứ hai B_Y
- P₁+3: Tọa độ X thứ hai B_X

$$R = B_Y - \left[\frac{(B_Y - A_Y)}{(B_X - A_X)} (B_X - S) \right]$$

Vi dụ: đổi số hex đọc từ khối analog in địa chỉ 002 tầm 0030 ÷ 0FD0H sang 0000 ÷ 0100BCD.

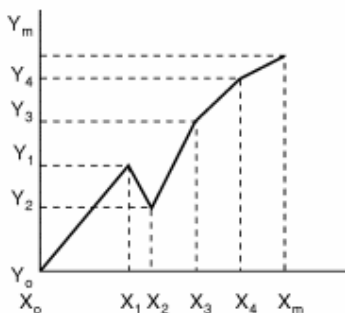
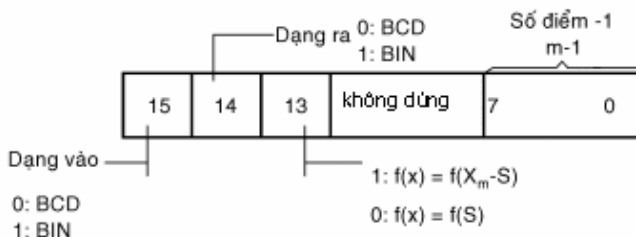
	SCL	DM 0100	0000	BCD
	002	DM 0101	0030	Hex
	DM0100	DM 0102	0100	BCD
	DM0000	DM 0103	0FD0	Hex

Tạo hàm APR (Arithmetic Process)



C = # 0000: tính SIN(θ), θ chứa trong S dạng BCD đơn vị 0,1 độ, kết quả được chứa vào D
 C = # 0001: tính COS(θ).
 (S) : 0 đến 900

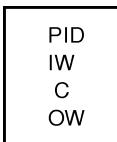
Nếu C là địa chỉ, ARP tính hàm nội suy $f(x)$. f chứa trong địa chỉ bắt đầu từ C, x chứa trong S. Hàm $f(x)$ là hàm tuyến tính từng đoạn và biểu thị bằng đồ thị, ghi trong bảng từ C+1 đến C+2m+2, C xác định số đoạn, dạng dữ liệu vào và ra. BCD hay BIN



C+1	X_m
C+2	Y_0
C+3	X_1
C+4	Y_1
C+5	X_2
C+6	Y_2
.....	
C+(2m+1)	X_m
C+(2m+2)	Y_m

Điều khiển PID

Hàm PID dùng thông số đặt trong C đến C+6 để tính OW dựa theo IW và SV:



IW: trị đo, nhị phân
 OW: tín hiệu ra khối điều khiển PID, nhị phân
 C: địa chỉ đầu bảng thông số

C: trị đặt SV, nhị phân

C+1: dải tỷ lệ P 1÷9999 ứng với dải tỷ lệ 0.1%..999.9%

C+2: hệ số tích phân BCD $T_{IK} = T_{I/\gamma}$

C+3: hệ số vi phân BCD $T_{DK} = T_{D/\gamma}$

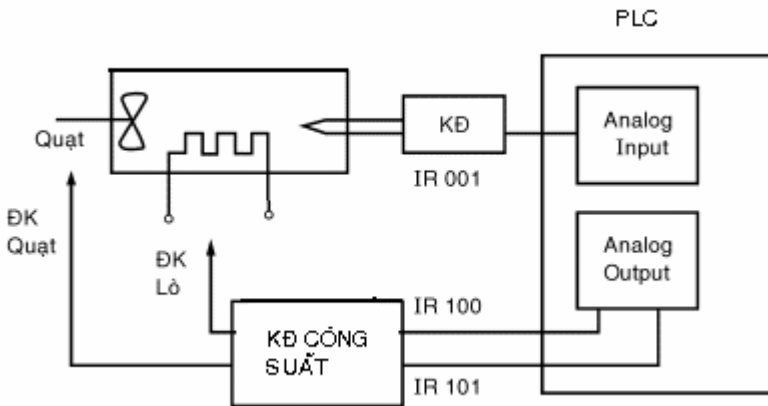
C+4: chu kỳ lấy mẫu γ từ 00.01 sec đến 99.99 sec

C+5: Bit4 ÷ bit15: thông số lọc thường chọn là 0.65 (000 BCD), bit 0÷3: 0- PID ngược, 1- PID thuận

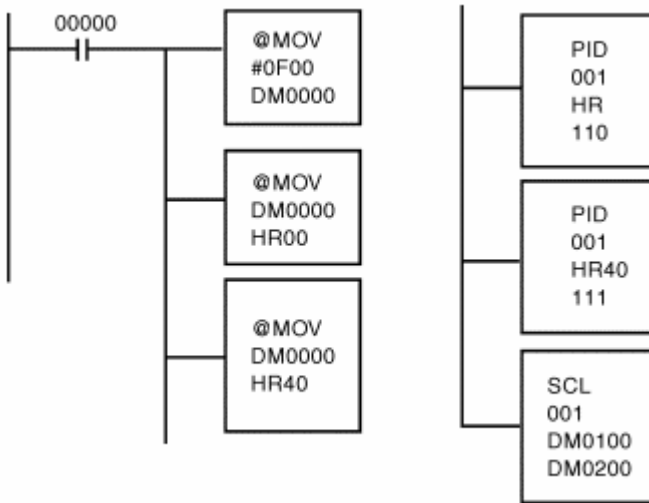
C+6: Bit 0÷3: số bit của biến ra, giá trị 0..8 ứng với số bit 8..16 bit; bit 4÷ 7: đơn vị thời gian của thời gian lấy mẫu, 0: đơn vị 100ms, 1: đơn vị 10ms; Bit 8÷11: tầm IW (như OW)

Các ô nhớ từ C+7 đến C+32 phải để trống.

Ví dụ: điều khiển nhiệt độ dùng điện trở đốt và quạt thổi



Thông số điều khiển lò		Thông số điều khiển quạt		Thông số đổi BIN-BCD	
HR00	DM0000 (trị đặt)	HR40	DM0000	DM0000	0000
HR01	0080 (P)	HR41	0060	DM0101	0000
HR02	0200 (T)	HR42	0150	DM0102	0200
HR03	0100 (T_D)	HR43	0000	DM0103	0FFF
HR04	0001 (T_S)	HR44	0001		
HR05	0000	HR45	0001		
HR6	0404 (Tắm)	HR46	0404		



Phát xung

Đôi khi cần phát xung tần số cao hay xung điều rộng ra ngoài vì để điều khiển động cơ bước hay điều khiển kiểu điều rộng xung. Dùng khối xuất transistor ta có thể phát xung ở một trong các địa chỉ IR 100 đến IR 115. Ghi vào ô nhớ DM6615 từ 00xx, xx từ 00 đến 15 tùy theo muốn dùng địa chỉ IR nào.

Dùng lệnh PULS 000 000 P1 để ấn định số xung sẽ phát, là nội dung ô nhớ P1+1, P1 từ 00000001 đến 16777215.

Lệnh SPED D M F qui định cách phát xung, D= 000..150 chọn ngõ ra 00 ..15 của từ đã qui định trong DM6615, M= 000 là mode phát số lượng xung do lệnh PULS qui định, M= 001 mode liên tục phát xung liên tục, F là tần số xung từ 0002..0100 nhân với 10Hz. Khi đang phát xung muốn đổi tần số ta thực hiện lệnh SPED với F thay đổi, nếu D= 000 thì ngừng phát xung.

9.12 TRUYỀN THÔNG

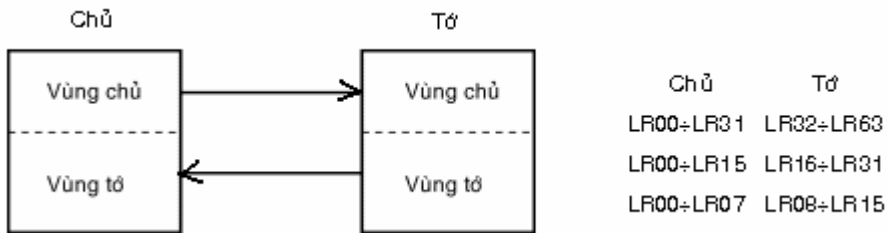
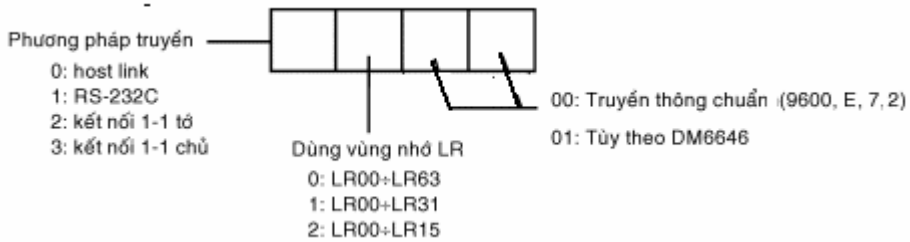
Có thể ghép nối PLC với nhau và máy tính với nhiều PLC thông qua các kết nối sau:

- Kết nối 1 ÷ 1 cho phép nối hai PLC qua cáp nối RS-232
- Host link nối một máy tính với một PLC qua cáp RS-232 hay một máy tính và nhiều PLC qua cáp 485.

- Controller link: nối nhiều PLC với nhau qua hai dây (với module mạng)
- Ethernet: nối nhiều máy tính và nhiều PLC (với module mạng)..

Ngoài ra các module xuất nhập có thể nối đến PLC từ xa bằng cách dùng hai dây theo mạng Combo Bus S, Combo Bus D.

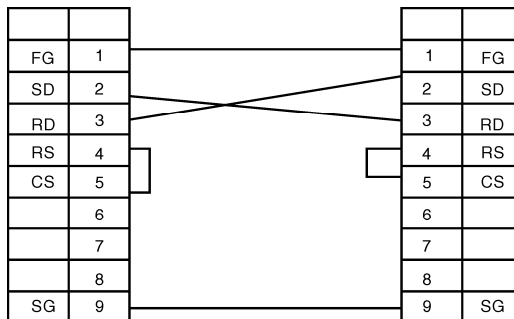
9.12.1 Kết nối 1-1: hai PLC kết nối với nhau theo chế độ chủ - tớ, đặt cấu hình qua ô nhớ DM6645



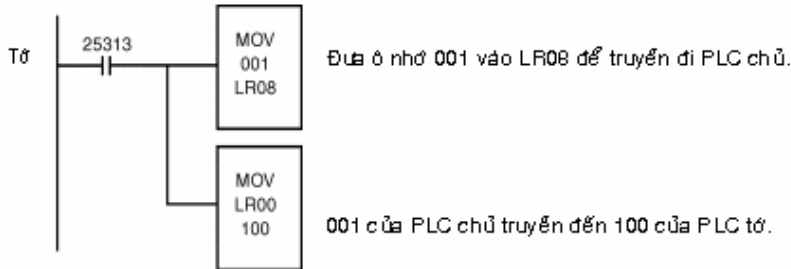
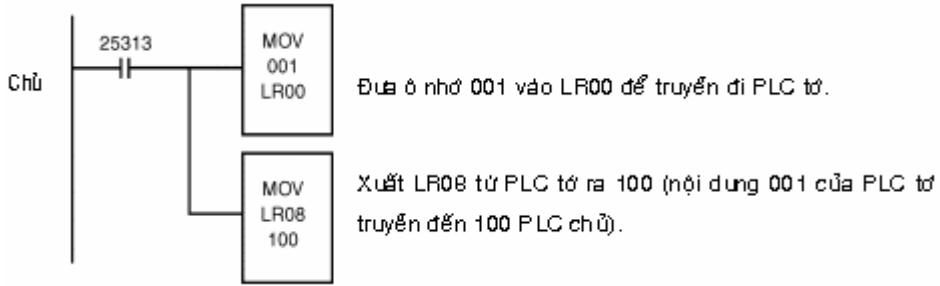
Vi dụ: kết nối 2 PLC dùng vùng nhớ LR00 đến LR15

Đặt DM6645: Chủ: 3200

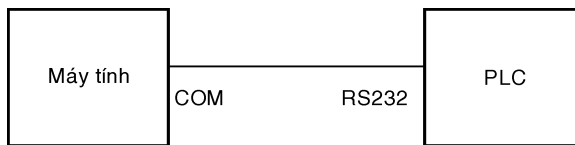
Tớ: 2200



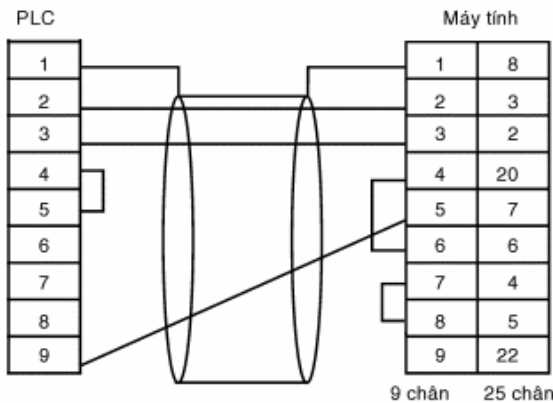
Sơ đồ đấu nối dây cáp 1 - 1

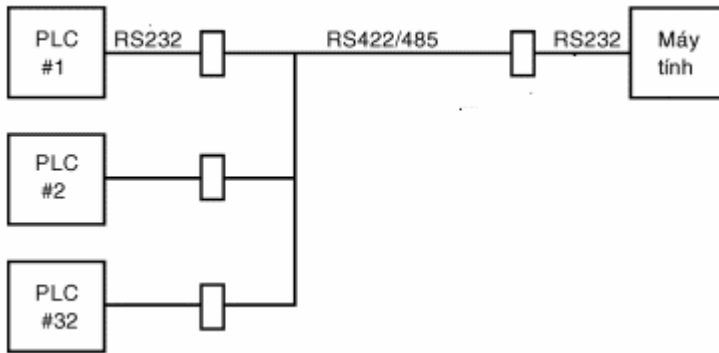


9.12.2 Host link: dùng để ghép nối máy vi tính với PLC qua cáp nối RS-232C. Nếu muốn ghép một máy vi tính với nhiều PLC ta phải dùng bộ chuyển đổi RS-232 ↔ RS-485 cho phép ghép với tối đa 32 PLC.



Sơ đồ nối dây như sau:



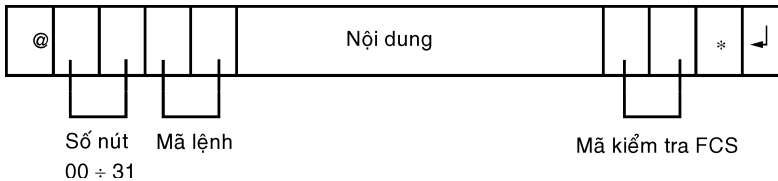


Thông qua host link có thể dùng máy tính để lập trình cho PLC hay đọc ghi bộ nhớ của PLC, từ PLC có thể truyền thông tin cho máy tính dùng lệnh TXD.

Đặt cấu hình dùng DM6645 với cấu hình chuẩn là 0000.

Nếu dùng RS-422/485 thì mỗi PLC được đánh số nút từ 0000 đến 0031 trong DM6648.

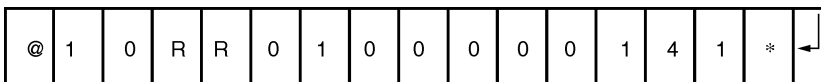
Máy tính truyền tin đến PLC theo dạng sau:



Một khối truyền dài tối đa 131 ký tự, nếu dài hơn 131 thì tách ra nhiều khối, mỗi khối kết thúc bằng ↵ (CHR\$(13)). Khối cuối kết thúc bằng *↵.

FCS (*frame check sequence*) là kết quả phép EXCLUSIVE OR các byte truyền từ đầu đến trước FCS và đổi thành hai ký tự ASCII. Khi nhận thông tin, máy tính hay PLC tính FCS rồi so sánh với FCS đã nhận.

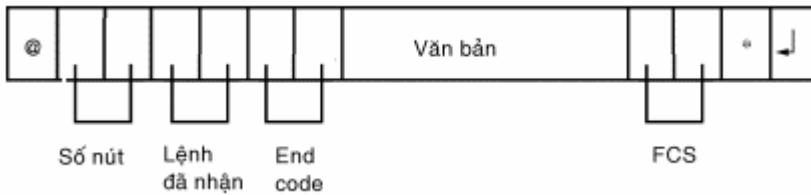
Ví dụ: truyền lệnh đọc ô nhớ 0100 ở PLC số nút 10.



Tính: EXOR @ 01000000
 1 00110001
 0 00110000

 FCS 01000001
 4 1

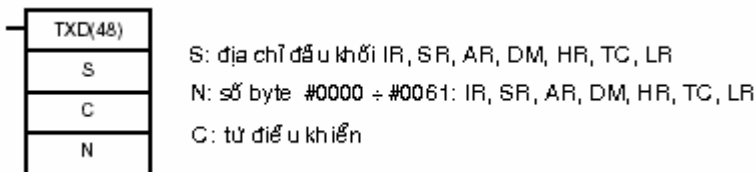
Khi nhận được thông tin từ máy tính, PLC tương ứng sẽ trả lời theo khổ sau:



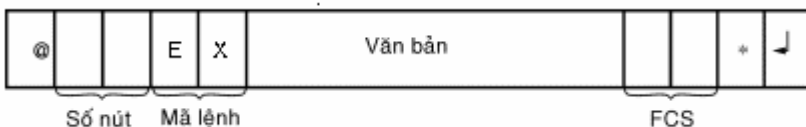
End code cho biết kết quả giao tiếp. Nếu giao tiếp tốt đẹp thì End code là 00.

Bảng đầy đủ các lệnh truyền từ máy tính đến PLC mời độc giả đọc tài liệu tham khảo của PLC OMRON.

PLC CQM1 cũng có thể chủ động truyền thông tin cho máy tính dùng lệnh TXD.



TXD đổi các byte nhị phân từ S đến S+N/2-1 ra mã ASCII, mỗi byte nhị phân đổi thành hai byte ASCII và truyền theo chuẩn qui định bởi C. Nội dung của C thay đổi tùy theo cách thức truyền và cổng nối tiếp, trường hợp đơn giản nhất là C= #0000, byte cao nhất của S được truyền đầu tiên. Khi muốn truyền phải kiểm tra bit AR0805 (cờ báo truyền xong) là ON mới được truyền. Khi lệnh TXD được thực hiện sẽ truyền theo dạng sau:

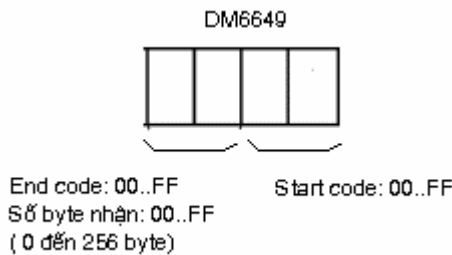
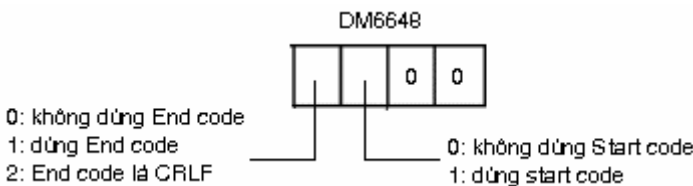


Ví dụ: truyền 10 byte chứa trong DM0000 đến DM0004 theo chuẩn Host link dùng lệnh TXD DM0000 #0000 #0010, dữ liệu truyền đi (ASCII) là @00EX1234123412341234123459*cr, giả sử các ô nhớ chứa số 1234. Máy tính phải có chương trình nhận dữ liệu.

Dùng TXD cho phép máy tính không cần thường xuyên đọc thông tin từ PLC mà PLC sẽ tự động truyền khi có nhu cầu. Nếu máy tính muốn trả lời thì truyền theo giao thức Host link đã trình bày ở trên.

9.12.3 Truyền thông tự do

Đặt ô nhớ DM6645 là 1000, dùng lệnh TXD để truyền và RXD để thu. Giao thức truyền do người dùng qui định bởi hai ô nhớ DM6648 và DM6649



Lệnh TXD giống như trong phần Host link, nhưng N có thể đến 0256, dữ liệu truyền đi được kèm thêm Start code, End code hay không tùy theo DM6848.

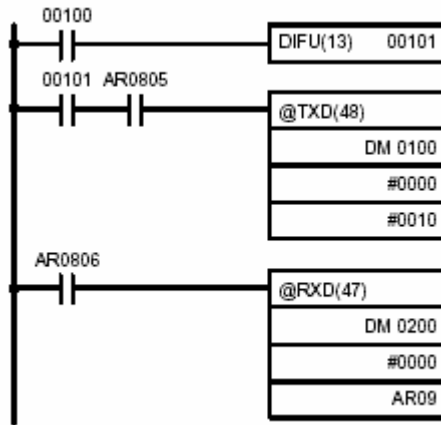
Máy tính truyền dữ liệu xuống PLC phải theo giao thức định bởi PLC. Khi PLC nhận dữ liệu xong, cờ thu AR0806 ON, tác động đến lệnh RXD, các byte ASCII được chuyển thành số nhị

phân 0..F, thông tin về nhận dữ liệu chứa trong các ô nhớ sau:

AR 0800 ..AR 0803 Mã sự cố cổng RS-232C BCD) 0: Thu bình thường, 1: Sai parity, 2: Sai Frame, 3: Tràn

AR 0804 Sai truyền thông
 AR0805 Truyền xong
 AR0905 Thu xong
 AR 0807 Cờ tràn, dữ liệu mất vì không đọc kịp
 AR 09 Số byte đã nhận (BCD)

Ví dụ: truyền 10 byte trong bảng kể từ ô nhớ DM0100 và nhận dữ liệu cất vào bảng kể từ DM0200. Đặt DM6645= 1000, DM6648= 2000, không start code, End code là CRLF



Cho bit SR25209 ON để reset cổng RS232.

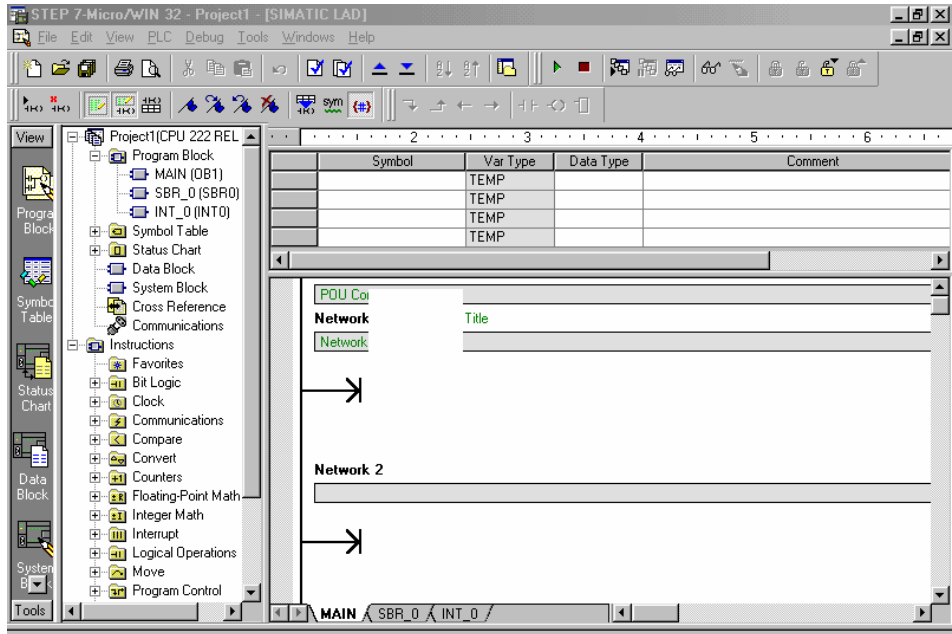
Đến đây chấm dứt phần PLC OMRON

9.13 PLC SIEMENS S7-200

PLC S7-200 là một họ gồm nhiều loại CPU như CPU-212, 214, 215, 216, 224, 226. Các họ này khác nhau ở dung lượng nhớ, module I/O, tập lệnh, số cổng giao tiếp..., tuy nhiên về đại thể là giống nhau. PLC được lập trình thông qua máy tính dùng chuẩn 485 với phần mềm lập trình Step 7 Microwin ver 2.0 hay 3.x theo kiểu kết nối PPI (point to point interface), nếu có card giao tiếp MPI (multi point interface) có thể ghép nối một PC với nhiều PLC.

Chương trình PLC S7-200 được thiết kế dưới dạng chương

trình chính (Main, OB), chương trình con (SBR), chương trình ngắt (INT), vùng nhớ dữ liệu (Data block)



S7-200 kết nối theo khối gồm khối CPU và các khối mở rộng khối CPU có các ngõ vào ra số và cổng truyền thông RS485, cổng kết nối mạng. Số lượng ngõ vào ra số tùy loại CPU. Bộ nhớ gồm ba loại ROM, EEPROM và RAM và chia làm nhiều vùng: I, Q, AI, AQ, M, SM, T, C, V, HC, AC. Các ô nhớ có thể truy cập theo bit, byte (B), từ (W), từ kép (DW).

Sau đây là bảng tóm tắt về các vùng nhớ:

Bảng 9.3

Miêu tả	CPU221	CPU222	CPU224/226
C/trình ng/dùng	2KW	2KW	4KW
Dữ liệu ng/dùng	1KW	1KW	2560W
Số module mở rộng	0	2	7
Ngõ vào số I (tối đa)	10.0..10.5	10.0..115.7	10.0..115.7
Ngõ vào I (trên module)	6	8	14

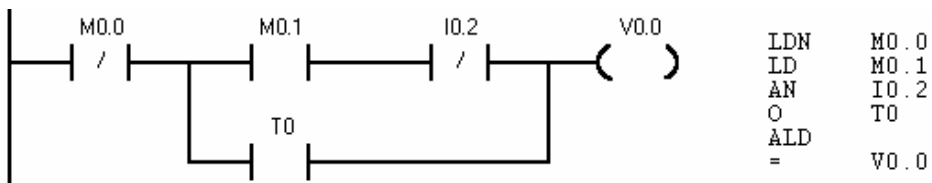
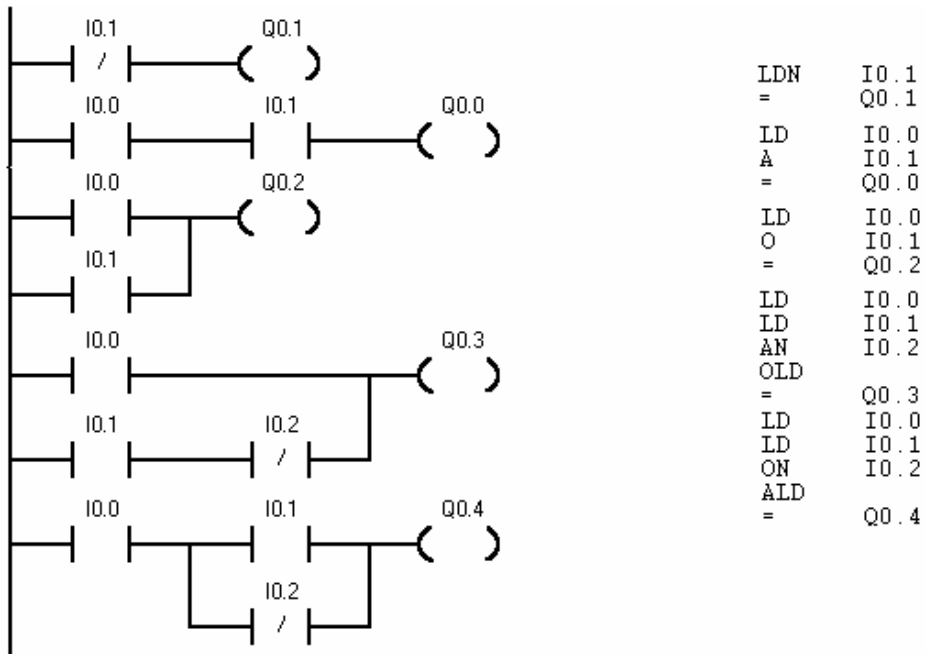
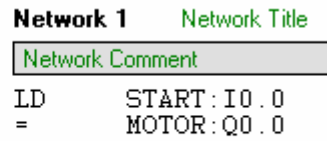
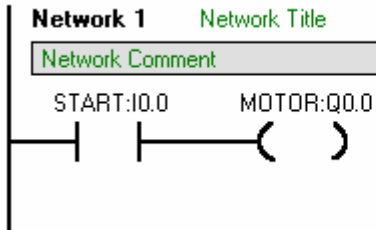
CPU)	I0.0..I0.5	I0.0..I0.7	I0.0..I0.7, I1.0..I1.5
Ngõ ra Q (tối đa)	Q0.0..Q0.3	Q0.0..Q15.7	Q0.0..Q15.7
Ngõ ra Q (trên module CPU)	4 Q0.0..Q0.3	6 Q0.0..Q0.5	10 Q0.0..Q0.7, Q1.0..1.1
Ngõ vào analog	AIW0..AIW30	AIW0..AIW30	AIW0..AIW30
Ngõ ra analog	AQW0..AQW30	AQW0..AQW30	AQW0..AQW30
Bộ nhớ thay đổi V	VB0..VB2047	VB0..VB2047	VB0..VB5519
Bộ nhớ trong M	MB0..MB31	MB0..MB31	MB0..MB31
Bộ nhớ đặc biệt SM	SMB0..SMB179	SMB0..SMB299	SMB0..SMB549
Timer	T0..T255	T0..T255	T0..T255
Counter	C0..C255	C0..C255	C0..C255
Đếm vận tốc cao	HC0..HC3	HC0..HC3	HC0..HC5
Thanh ghi ACC	AC0..AC3	AC0..AC3	AC0..AC3
Vòng PID	8 vòng	8 vòng	8 vòng

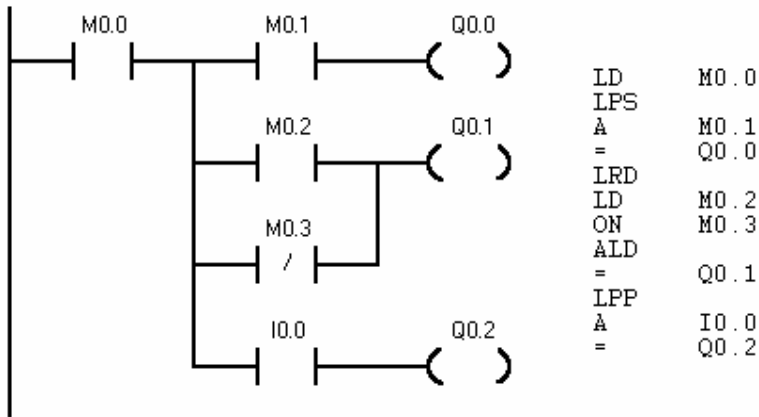
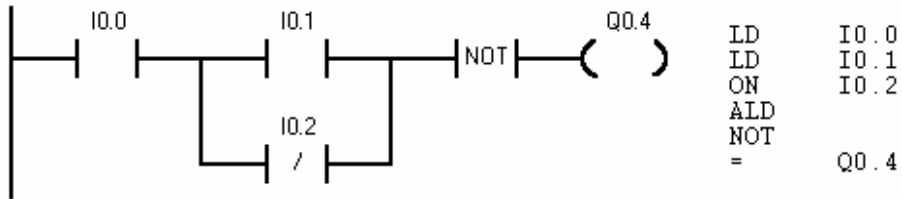
- Vùng nhớ AI, QI: chỉ truy cập theo từ (16 bit): AIW0, AQW 10.
- Vùng nhớ V, I, Q, M, SM: có thể truy cập theo bit, byte, từ hay từ kép: I0.1, QB2, VW150...
- T, C: truy cập theo bit: T1, C15 là trạng thái hay từ là nội dung 16 bit.
- AC: truy cập theo byte, từ hay từ kép .
- HC: truy cập theo từ kép.
- Vùng nhớ M, V, T, C có thể lưu lại khi mất điện.
- Một từ ví dụ VW0 gồm hai byte VB100 (Byte cao) và VB101(Byte thấp)
- Một từ kép VD101 gồm hai từ VW101 (Từ cao)và VW103, bốn byte VB100 (Byte cao), VB101, VB102, VB103 .
- Địa chỉ gián tiếp dùng ô nhớ 32 bit (V, AC) làm con trỏ, giả sử AC1 là con trỏ, lệnh MOVD &VB0, AC1 đưa địa chỉ ô nhớ VB0 vào AC1, lệnh MOVW *AC1, MW 10 đưa nội dung ô nhớ VW0 sang ô nhớ MW10
- Số : số nguyên không dấu 8 bit (BYTE), 16 bit (WORD), 32 bit (DWORD) (00..FF. 0000..FFFF. 00000000..FFFFFFFF), số có dấu mã phụ hai INT, DINT (80..7F. 6000..7FFF. 80000000..FFFFFFFF), số thực 32 bit REAL.
- Hằng số thập phân 192, số thực +1.52E-2, nhị phân 2#11000000, Hex 16#C0, chuỗi 'AT'

Phần sau trình bày các lệnh chính của PLC S7-200, bạn đọc cần biết chi tiết hơn đề nghị đọc các tài liệu chuyên sâu.

9.14. LỆNH CƠ BẢN

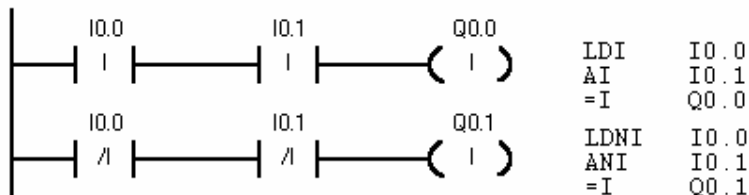
9.14.1 Lệnh bit



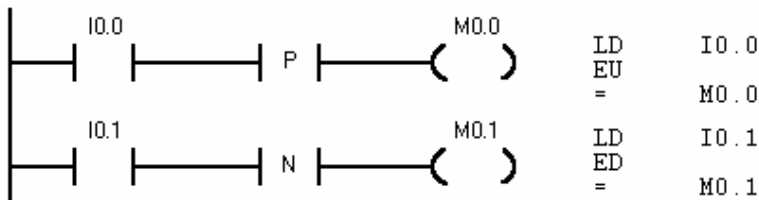


9.14.2 Lệnh lập tức

Lệnh lập tức ký hiệu bằng chữ I, bit nhớ trong sẽ phản ánh ngay ngõ vào I không chờ đến chu kỳ quét, tương tự bit nhớ trong xuất ngay ngõ xuất Q

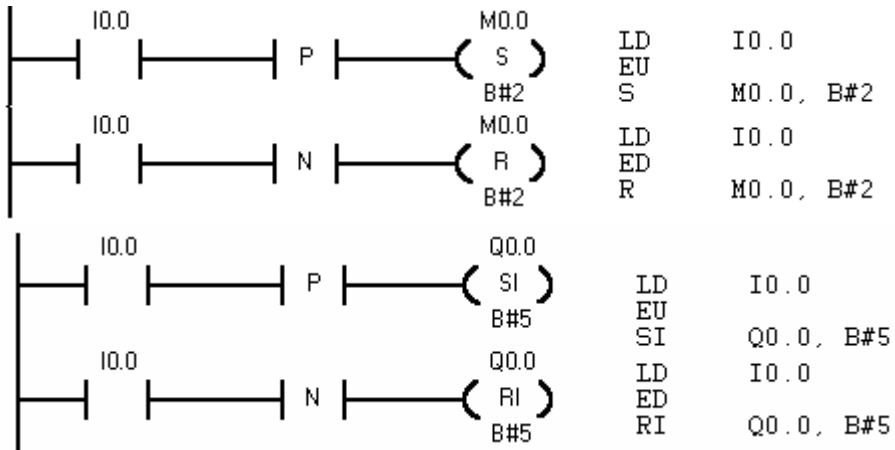


9.14.3 Lệnh vi phân



9.14.4 Lệnh đặt/ xoá

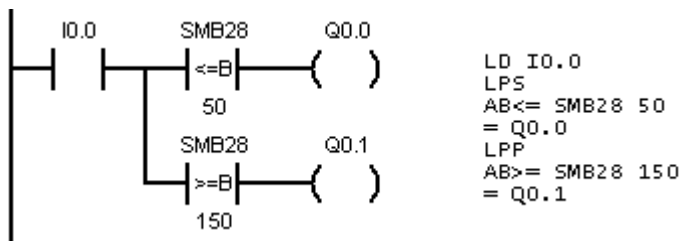
cho một loạt n bit liên tiếp nhau ON hay OFF



9.15 LỆNH SO SÁNH

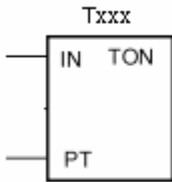
So sánh hai số n1 và n2 theo byte không dấu B, số nguyên có dấu 16 bit I, số nguyên có dấu 32 bit D, số thực R theo các phép <, ==, >, <>, <=, >=, kết quả so sánh biểu thị bằng trạng thái tiếp điểm.

Ví dụ:



9.16 LỆNH TIMER/COUNTER

Có ba loại timer là ON delay, Retentive ON delay và OFF delay. Thời gian timer bằng giá trị PT nhân hệ số tùy thuộc xxx.



Khi IN là ON sau thời gian trễ tiếp điểm TXX sẽ ON, nội dung bộ đếm tiếp tục tăng đến tối đa 32767.

Khi IN là OFF thì tiếp điểm là OFF và nội dung bộ đếm bằng 0.

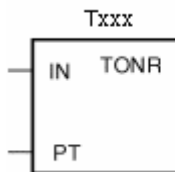
Hệ số 1ms : T32, T96

10ms : T33..T36, T97..T100

100ms: T37..T63, T101..T255

PT: VW, T, C, IW, QW, MW, SMW, SW, AC, AIW, Hằng số, *VD, *VC

PT tối đa 32767

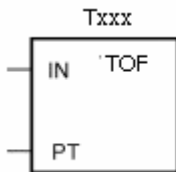


TONR khác TON ở chỗ khi IN là OFF thì nội dung timer không bị xóa, bộ đếm tiếp tục tăng khi IN từ OFF sang ON cho đến khi nội dung timer bằng PT thì tiếp điểm Txxx đóng. Bộ đếm tiếp tục đếm đến 32767. Muốn xóa Txxx ta phải dùng lệnh reset R

Hệ số 1ms : T0, T64

10ms : T1..T4, T65..T68

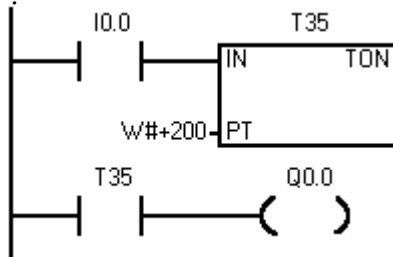
100ms: T5..T31, T69..T95



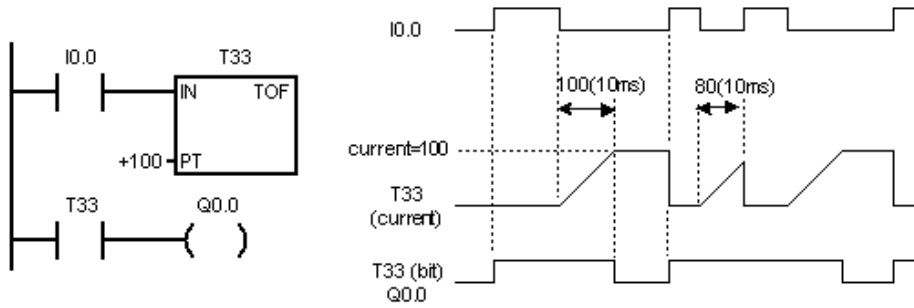
Khi IN ON thì Txx ON, bộ đếm xóa, khi IN OFF bộ đếm bắt đầu đếm lên, khi bằng PT thì Txxx OFF và ngừng đếm. Nếu thời gian IN OFF ngắn hơn delay thì Txxx vẫn ON

Hệ số TOF giống TON, không được dùng TON và TOF cùng số.

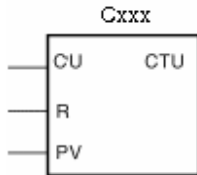
Ví dụ:



```
LD I0.0
TON T35, W#+200
LD T35
= Q0.0
```



Có ba loại đếm: đếm tăng CTU, đếm giảm CTD và đếm tăng giảm CTUD

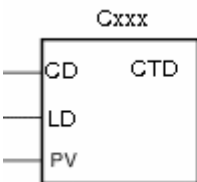


Khi R là ON thì bộ đếm bị xóa

Khi có xung từ OFF sang ON vào ngõ CU thì bộ đếm tăng 1

Khi nội dung bộ đếm bằng PV thì tiếp điểm Cxxx là ON, nội dung bộ đếm tiếp tục tăng theo IN đến tối đa 32767

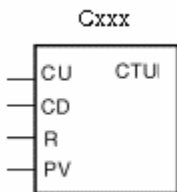
xxx: 0..255



LD ON nạp PV vào bộ đếm và xóa Cxxx

LD OFF, bộ đếm giảm khi CD chuyển từ OFF sang ON

Khi giảm đến 0 Cxxx ON, bộ đếm ngừng đếm



Bộ đếm sẽ tăng hoặc giảm tùy xung vào CU

hay CD và khi R OFF

Khi R ON nội dung bộ đếm là 0, Cxxx OFF

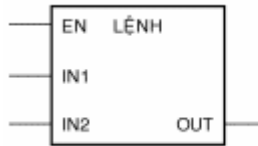
khi nội dung bộ đếm \geq PV, Cxxx ON

xxx: 0 đến 255

9.17 LỆNH SỐ HỌC

- ADD_I/ SUB_I Cộng/ trừ số nguyên 16 bit
- MUL Nhân hai số 16 bit, kết quả 32 bit
- DIV chia hai số 16 bit cho thương số và dư số 16 bit.
- MUL_I/DIV_I Nhân/ chia số 16 bit với kết quả 16 bit.

- ADD_DI/ SUB_DI, MUL_DI, DIV_DI Cộng/ trừ/ nhân/ chia số nguyên 32 bit, kết quả 32 bit.
- ADD_R/ SUB_R/ MUL_R/ DIV_R Cộng/ trừ/ nhân/ chia số thực 32 bit

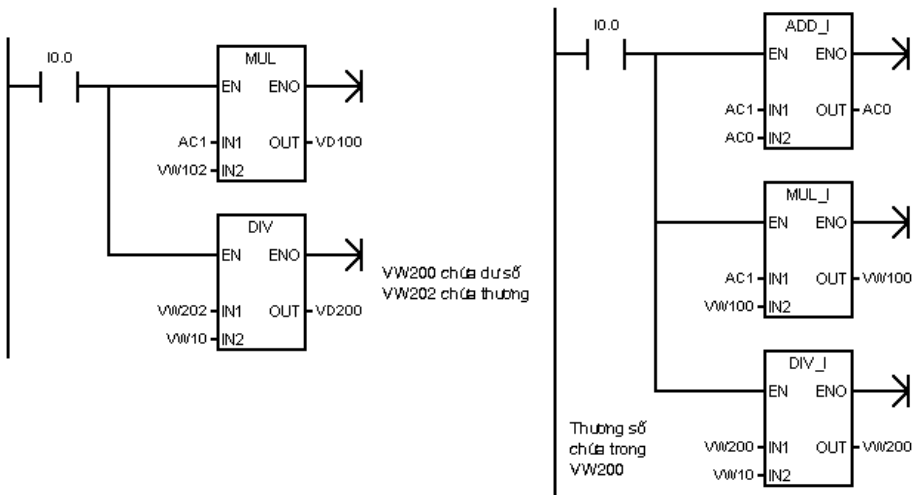


LỆNH: mã lệnh

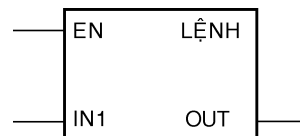
IN1, IN2: toán hạng

OUT: nơi chứa kết quả

Ví dụ:

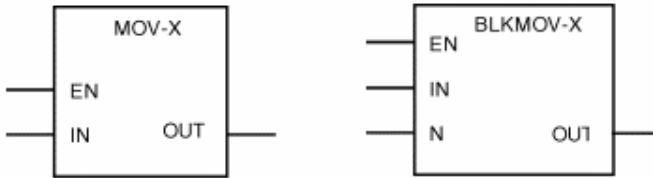


- Căn bậc hai số thực 32 bit SQRT
- Tăng/ giảm byte INC_B/ DEC_B
- Tăng/ giảm từ INC_W/ DEC_W
- Tăng/ giảm từ kép INC_DW/ DEC_DW
- SIN COS TN EXP LN



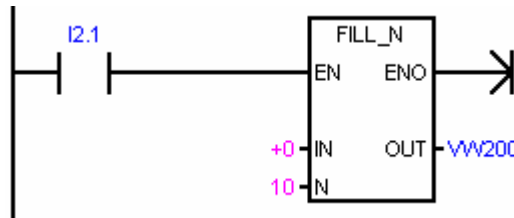
9.18 LỆNH DI CHUYỂN

Di chuyển nội dung ô nhớ IN ra ô nhớ OUT bằng lệnh MOV_X hay di chuyển một khối N ô nhớ địa chỉ bắt đầu IN sang khối ô nhớ địa chỉ bắt đầu OUT dùng lệnh BLKMOV_X, ô nhớ có thể là byte, word, double word hay real, X = B, W, DW, R

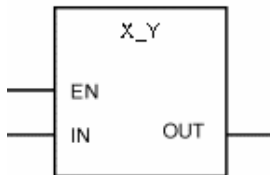


SWAP trao đổi hai byte của một từ

FILL_N chép một từ vào một loạt ô nhớ

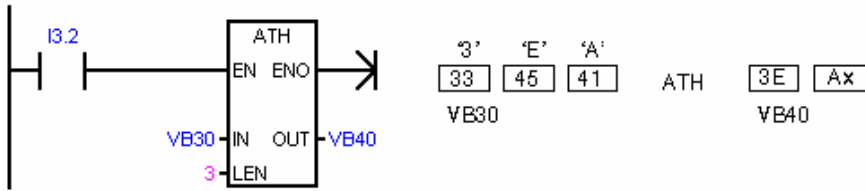


9.19 LỆNH ĐỔI



Chuyển đổi dữ liệu IN loại X sang dữ liệu OUT loại Y
 B-I, I_DI, DI_R
 DI_I, I_B
 BCD_I, I_BCD
 Các ô nhớ IN và OUT có kích thước phù hợp lệnh

- ROUND đổi số thực sang số nguyên kép làm tròn
- TRUNC đổi số thực sang số nguyên kép bỏ phần lẻ
- ATH đổi một số byte ASCII (30..39, 41..46) chiều dài LEN ra các số HEX 0..9, A..F
- HTA đổi một số digit 0..F chiều dài LEN ra mã ASCII



9.20 LỆNH GHI DỜI

- SHRB dời một bit DATA vào thanh ghi chiều dài N có địa chỉ bit LSB là S_BIT. Khi N dương DATA vào LSB còn MSB dời ra SM1.1. Khi N âm DATA vào MSB còn LSB dời ra SM1.1
- SHR_X/ SHL_X dời phải/ trái một byte, từ hay từ kép IN tùy theo X = B, W, DW vào OUT. Số bit dời qui định bởi N, bit được dời ra chứa vào SM1.1 và OUT.

9.21 LỆNH QUAY

ROR_X/ ROL_X quay phải/trái một byte, từ hay từ kép IN, bit được dời ra đưa trở lại vào IN, vào SM1.1 và vào OUT. Số lần quay định bởi N. X= B, W, DW

9.22 LỆNH LOGIC

Lệnh AND : ANDB, ANDW, ANDD

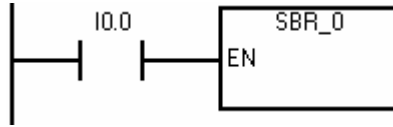
Lệnh OR: ORB, ORW, ORD

Lệnh XOR: XORB, XORW, XORD

Lệnh INV: INB, INW, IND

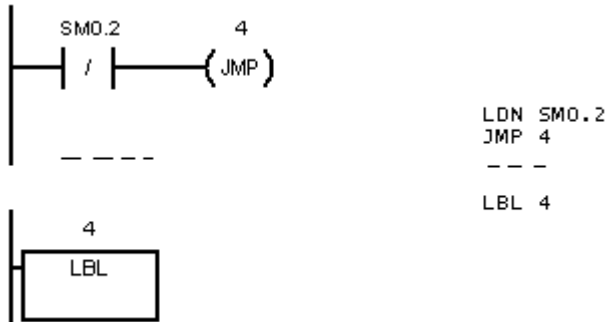
9.23 GỌI CHƯƠNG TRÌNH CON

Có thể có tối đa 64 chương trình con, chương trình con viết sau chương trình chính và được đóng khung bằng SBR_N RET. Chương trình con được gọi bằng lệnh CALL SBR_N. Chương trình con có thể kèm tham số vào ra.

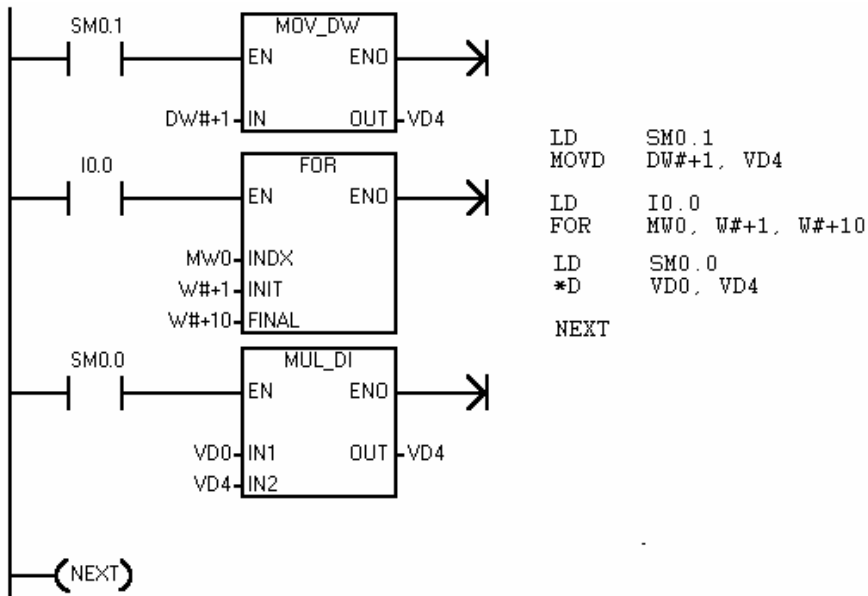


9.24 LỆNH ĐIỀU KHIỂN CHƯƠNG TRÌNH

Lệnh **JMP n** nhảy đến đoạn chương trình nhãn n

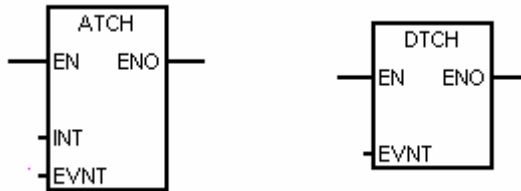


Lệnh **FOR NEXT** thực hiện lặp vòng các dòng lệnh giữa FOR và NEXT, số vòng được đếm bởi biến INDX có giá trị đầu INIT và giá trị cuối FINAL, số vòng lặp là FINAL -INIT +1. Có thể đặt các vòng lặp lồng vào nhau.



9.25 LỆNH NGẮT

Có thể có tối đa 128 chương trình phục vụ ngắt, viết sau chương trình con, đóng khung bằng INT_n RETI. Cấm/cho phép ngắt bằng lệnh ENI/DISI. INT_n được gọi đến khi xảy ra sự kiện EVNT. Có tất cả 27 sự kiện có thể gây ra ngắt. Sự kiện được liên kết với INT_n thông qua lệnh gán ATCH và tháo DTCH.



Bảng 9.5: Các sự kiện ngắt và ưu tiên

Số sự kiện	Miêu tả	Ưu tiên	Ưu tiên trong nhóm
8	Nhận ký tự ở Port 0	Cao nhất	1
9	Truyền xong Port 0		1
23	Nhận xong bản tin ở Port 0		1
24	Nhận xong bản tin ở Port 1		2
25	Nhận ký tự ở Port 1		2
26	Truyền xong Port 1		2
0	Ngắt ở cạnh lên của I0.0	Giữa	1
2	Ngắt ở cạnh lên của I0.1		2
4	Ngắt ở cạnh lên của I0.2		3
6	Ngắt ở cạnh lên của I0.3		4
1	Ngắt ở cạnh xuống của I0.0		5
3	Ngắt ở cạnh xuống của I0.1		6
5	Ngắt ở cạnh xuống của I0.2		7
7	Ngắt ở cạnh xuống của I0.3		8
12	Đếm vận tốc cao HSC0: trị đo bằng trị đặt		1
13	Đếm vận tốc cao HSC1: trị đo bằng trị đặt		9
14	HSC1 đổi hướng đếm		10

15	Xóa ngoài HSC1		11	
16	Đếm vận tốc cao HSC2: trị đo bằng trị đặt		12	
17	HSC2 đổi hướng đếm		13	
18	Xóa ngoài HSC2		14	
32	HSC3: trị đo bằng trị đặt		19	
29	HSC4: trị đo bằng trị đặt		20	
30	HSC4 đổi hướng đếm		21	
31	Xóa ngoài HSC4		22	
33	HSC5: trị đo bằng trị đặt		23	
19	Đếm xung PLS0 xong		15	
20	Đếm xung PLS1 xong		16	
10	Ngắt thời gian 0		Thấp nhất	1
11	Ngắt thời gian 1			2
21	Ngắt timer T32			3
22	Ngắt timer T96	4		

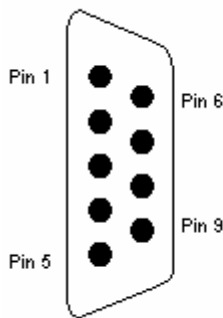
Tùy theo loại CPU có một số sự kiện không được hỗ trợ. Sau đây ta sẽ phân tích một số ngắt chính.

9.25.1 Ngắt thời gian

Ngắt thời gian 0/1 xảy ra theo chu kỳ ấn định (tối đa 255ms) bởi nội dung của SMB34/SMB35 (đơn vị ms) thường dùng để đọc hay xuất tín hiệu analog. Ngắt timer T32/T96 xảy ra khi timer T32 hoặc T96 hoàn tất thời gian trễ đã đặt.

9.25.2 Ngắt truyền thông

PLC có một hoặc hai cổng (Port 0, Port 1) dùng để truyền thông nối tiếp RS485.



Pin Number	Signal
1	Shield
2	24 V Return
2	RS-485 Signal B
4	Request-to-Send
5	5 V Return
6	+5 V
7	+24 V
8	RS-485 Signal A
9	Not applicable

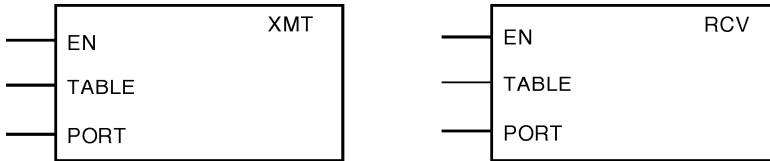
Thông qua cổng nối tiếp có thể ghép S7-200 với các thiết bị khác như S7-200, S7-300, HMI, PC. Muốn sử dụng cổng nối tiếp theo giao thức người dùng ta phải đặt cổng ở chế độ freeport bằng cách đặt nội dung cho SMB30 (port 0) và SMB130 (port 1) theo bảng sau:

Bảng 9.6

SMB 30, SMB 130	Miêu tả
Bit 7, 6	00, 10: không parity
	01: parity chẵn
	11: parity lẻ
Bit 5	0: data 8 bit
	1: data 7 bit
Bit 4, 3, 2	000: 38400 (CP212 19200)
	001: 19200
	010: 9600
	011: 4800
	100: 2400
	101: 1200
	110: 115200 (600 CPU212)
	111: 57600 (300 CPU212)
Bit 1, 0	00, 11: PPI slave
	01: Freeport
	10: PPI master

Tương tự PLC Omron, ở chế độ freeport có thể truyền và nhận data bằng hai lệnh XMT và RCV. Dữ liệu truyền lấy từ bảng có địa chỉ đầu ở TABLE, còn dữ liệu nhận chứa vào TABLE,

chiều dài bảng tối đa 255. Byte đầu của bảng cho biết chiều dài dữ liệu. Khi ký tự cuối của bảng được gửi sẽ báo sự kiện ngắt 9 hay 26 hay tác động SM4.5. Khi thu xong bằng lệnh RCV sẽ tác động sự kiện 23, 24, SMB86..94 (port 0) SMB186..194 (port 1).
Lệnh RCV



Vi dụ: Truyền và nhận ký tự

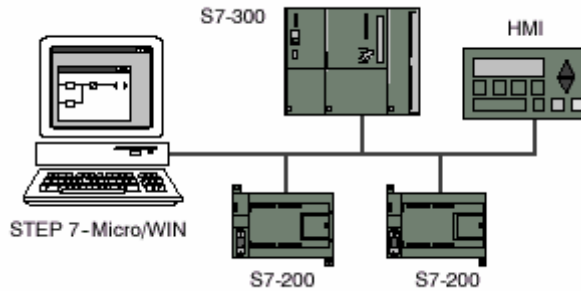
```
LD SM0.1
MOVB 9, SMB30 // Chọn Protocol 9600 baud, 8 bits/1 ký tự, không parity
MOVB 1, VB100 // chiều dài thông điệp là 1 ký tự ASCII
MOVB 16#41, VB101 // ký tự truyền là "A", có mã ASCII là 41 H
LD SM0.1 // bit SM0.1 on trong chu kỳ quét đầu tiên
ATCH INT_0, 8 // liên kết ngắt 0 (INT0) với biến cố 8.
ENI // cho phép tất cả ngắt sự kiện.
LD I0.1 // Load ngõ vào I0.1.
EU // Xét sườn lên I0.1
XMT VB100, 0 // truyền bảng có địa chỉ đầu VB100 ra port 0
```

Chương trình ngắt INT0

```
// nhận đúng ký tự truyền là "A" thì báo hiệu ở ngõ ra Q0.1
LDB= Receive_Char, 16#41
S Char_A, 1 // bật ngõ ra Q0.1.
```

Có thể dùng các ô nhớ SMB86 ÷ SMB94 và SMB186 ÷ SMB194 để đặt cấu hình cho việc thu bản tin, ngoài ra sự kiện ngắt 8, 25 sẽ xảy ra khi thu một ký tự, lúc này SMB2 chứa ký tự vừa nhận còn SM3.0 chứa kết quả kiểm tra parity.

Khi nối nhiều PLC với nhau qua mạng 485 ta dùng lệnh NETR/ NETW để đọc/ghi dữ liệu, các PLC phải có địa chỉ cụ thể.



9.25.3 Đếm vận tốc cao

Có ba bộ đếm vận tốc cao HSC0, HSC1 và HSC2 dùng để đếm xung tần số cao từ encoder, HSC0 đếm lên xuống với xung nhíp vào ở I0.0 tần số tối đa 2KHz, hướng đếm định như sau SM37.3 = 0 đếm xuống, SM37.3 = 1 đếm lên. Giá trị đếm của HSC0 chứa trong 32 bit của SMD38 gồm SMB38, SMB39, SMB40 và SMB41 (MSB trong SMB38, LSB trong SMB41). Giá trị đặt trước cho HSC0 chứa trong SMD42. Khi giá trị hiện tại bằng giá trị đặt sẽ gây ra sự kiện ngắt 12. HSC1 và HSC2 là bộ đếm vận

Chế độ	Miêu tả	I0.6	I0.7	I1.0	I1.1
0	Đếm lên xuống	(I1.2) Nhịp	(I1.3)	(I1.4)	(I1.5) Start
1	SM47.3=0 đếm xuống (SM57.3)			Xóa	
2	SM47.3=1 đếm lên (SM57.3)				
3	Đếm lên xuống với điều khiển hướng bên ngoài	Nhịp	Hướng		Start
4				Xóa	
5					
	I0.7 = 0 đếm xuống (I 1.3)	Nhịp lên	Nhịp xuống		Start
	I0.7 = 1 đếm lên			Xóa	
6	Đếm lên xuống với hai xung nhịp				
7					
8					
9	Đếm lên xuống với 2 xung AB vuông pha từ encoder	Nhịp A	Nhịp B		Start
10				Reset	
11					

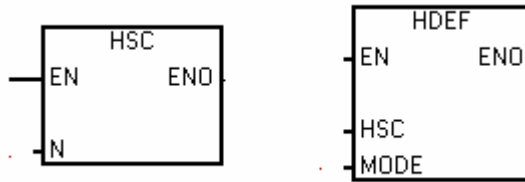
tốc cao có 12 chế độ hoạt động, giá trị hiện tại của HSC1 là SMD48 và của HSC2 là SMD58. Trị đặt của HSC1/HSC2 là SMD52/SMD62. Khi trị hiện tại bằng trị đặt sẽ phát sinh ngắt 13 với HSC1 và 16 với HSC2.

Khi chân xóa tích cực giá trị hiện tại của HSC được xóa, khi chân start tích cực cho phép HSC đếm xung. Mức tích cực của các chân này được định bởi SMB47 và SMB57.

Bảng 9.8

HSC1	HSC2	Tác dụng khi lệnh HDEF được thực hiện
SM 47.0	SM 57.0	0: chân xóa tích cực cao
SM 47.1	SM 57.1	0: chân start tích cực cao
SM 47.2	SM 57.2	0: số xung đếm bằng 4 lần xung vào A

Muốn dùng HSC, đầu tiên phải xác định số HSC, mode hoạt động, từ điều khiển, sau đó dùng lệnh HDEF để khởi động.



Bảng 9.9

HSC0	HSC1	HSC2	Miêu tả
SM 37.0	SM 47.0	SM 57.0	Không dùng sau khi HDEF đã thực hiện
SM 37.1	SM 47.1	SM 57.1	nt
SM 37.2	SM 47.2	SM 57.2	nt
SM 37.3	SM 47.3	SM 57.3	Bit hướng 0: đếm xuống; 1: đếm lên
SM 37.4	SM 47.4	SM 57.4	Ghi bit hướng 0: không ghi; 1: ghi
SM 37.5	SM 47.5	SM 57.5	Ghi trị đặt vào HSC 0: không ghi; 1: ghi
SM 37.6	SM 47.6	SM 57.6	Viết trị hiện tại mới vào HSC 0: không ghi; 1: ghi
SM 37.7	SM 47.7	SM 57.7	0: cấm HSC; 1: cho phép HSC

Các bit điều khiển chỉ tác động sau khi lệnh HSC được thực hiện. Để hiểu rõ HSC ta xét trường hợp dùng HSC1 đếm xung AB từ encoder kiểu đếm 4X, hai tín hiệu A, B nối vào I0.6 và I0.7, tín hiệu xóa vào I1.0 và tín hiệu khởi động vào I1.1

Network 1

```
LD          SM0.1      // khi quét lần đầu, gọi chương trình con 0
CALL       0          // khởi động bộ đếm HSC
```

Network 2

```
MEND
```

Network 3

```
SBR          0
```

Network 4

```
LD          SM0.0
MOVB       16#F8, SMB47 // Dừng HSC1 mode 11
```

```

HDEF          1, 11      // Ban đầu đếm lên, chân xóa và start
                    tích cực ở mức cao, 4X

MOVD          0,5MD48   // Trị hiện tại là 0
MOVD          50, SMD52 // Trị đặt là 50
ATCH         0,13      // sự kiện ngắt 13 sẽ gọi ngắt số 0
ENI           // cho phép ngắt
HSC          1         // Nạp thông số vào HSC1

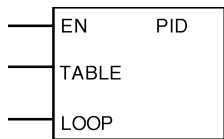
Network 5
RET

Network 6
INT          0

Network 7
LD           SM 0.0     // khi có ngắt xóa trị hiện tại
MOVD        0, SMD 48   // của HSC0
MOVB        16#C0, SMB47
HSC         1

Network 8
RETI
    
```

9.26 LỆNH PID



Lệnh này không có ở CPU 212, 214, trong chương trình có thể dùng tối đa 8 lệnh PID. Thông số lệnh chứa trong một bảng 36 byte gồm 9 thông số thực 4 byte. TABLE là địa chỉ bảng (vùng VB), LOOP từ 0 đến 7. Sau đây là bảng các thông số

Bảng 9.4

Địa chỉ lệnh	Tên	Miêu tả
0	Đại lượng hiện tại PV_n	Đại lượng được điều khiển, chuẩn hóa từ 0.0 đến 1.1
4	Đại lượng đặt SV_n	Trị đặt, chuẩn hóa
8	Đại lượng điều khiển M_n	Ngõ ra của PID, chuẩn hóa
12	Độ lợi vòng K_C	Hệ số tỷ lệ
16	Thời gian lấy mẫu T_S	Thời gian lấy mẫu, đơn vị là giây
20	Thời gian tích phân T_I	
24	Thời gian vi phân T_D	

28	Đại lượng offset MI_{n-1}	Giá trị tích phân ở thời điểm trước, chuẩn hóa
32	Đại lượng trước PV_{n-1}	Trị số đo của đại lượng được điều khiển ở thời gian lấy mẫu trước

Thuật toán PID được thực hiện theo phương trình:

$$M_n = MP_n + MI_n + MD_n$$

$$MP_n = K_c^*(SV_n - PV_n)$$

$$MI_n = MI_{n-1} + K_c^*T_S/T_I^*(SV_n - PV_n)$$

$$MD_n = K_c^*T_D/T_S^*(PV_{n-1} - PV_n)$$

Trị đặt SV và trị đo PV đều phải là số thực và được chuẩn hóa. Đầu tiên phải đổi số nguyên 16 bit ra số thực, sau đó chia cho tầm (Span) của đại lượng đó và cộng với Offset (là 0.0 nếu đơn cực và 0.5 nếu lưỡng cực)

Ví dụ, ta muốn đổi trị số đo của AIW0 ra số thực và chuẩn hóa cho 64000 cất vào VD100:

```

XORD    AC0, AC0           //Xóa ACC
MOVW    AIW0, AC0         //Cất trị đo vào ACC
LDW>=   AC0, 0            //Nếu dương
JMP     0                  //đổi sang số thực
NOT     0                  //Nếu âm
ORD     16#FFFF0000, AC0  //Khai triển dấu giá trị trong AC0
LBL     0
DTR     AC0, AC0          //Đổi số nguyên 32 bit ra số thực
/R      64000.0, AC0      //Chuẩn hóa
+R      0.5, AC0
MOVR    AC0, VD100
    
```

Tín hiệu ra M là đại lượng đã chuẩn hóa, muốn dùng để điều khiển ta phải đổi ra số nguyên 16 bit.

Ví dụ: đổi trị thực lưỡng cực ở VD108 ra số nguyên 16 bit ở AQW0

```

MOVR    VD108, ACV0
-R      0.5, AC0
*R      64000.0, AC0
TRUNC   AC0, AC0
MOVW    AC0, AQW0
    
```

Vi dụ: lập trình PID điều khiển mức nước trong bồn chứa sao cho áp suất nước ở đường xả không đổi, mức nước được duy trì ở mức 75% tối đa bằng cách điều khiển vận tốc bơm.

Đo mức: AIW0 Điều khiển bơm: AQW0
 Bảng PID: VD100 Gọi PID: gọi ngắt thời gian 0.1s
 K_C : 0.25, T_S : 0.1s, T_I : 30 phút

Chương trình dạng STL:

Network 1

LD SM0.1
 CALL 0 // Gọi chương trình con 0 đặt cấu hình PID ở lần quét đầu

Network 2

MEND

Network 3

SBR 0

Network 4

LD SM0.0
 MOVR 0.75, VD104 // Điểm đặt 75%
 MOVR 0.25, VD112 // Độ lợi vòng
 MOVR 0.10, VD116 // T_S
 MOVR 30.0, VD120 // T_I
 MOVR 0.0, VD124 // Không dùng đạo hàm
 MOVB 100, SMB34 // Ngắt thời gian 0.1s gọi PID
 ATCH 0,10
 ENI

Network 5

RET

Network 6

INT 0

Network 7

LD SM0.0
 XORD AC0, AC0
 MOVW AIW0, AC0 // Đọc mức nước và đổi sang số thực rồi
 đưa vào bảng
 DTR AC0, AC0
 / R 32000.0, AC0

MOVR AC0, VD100

Network 8

LD I0.0 // Khi I0.0 ON thì điều khiển PID

PID VB100,0

Network 9

LD SM0.0

MOVR VD108, AV0 // Xuất ra điều khiển

*R 32000.0, AC0

TRUNC AC0, AC0

MOVW AC0, AQW0

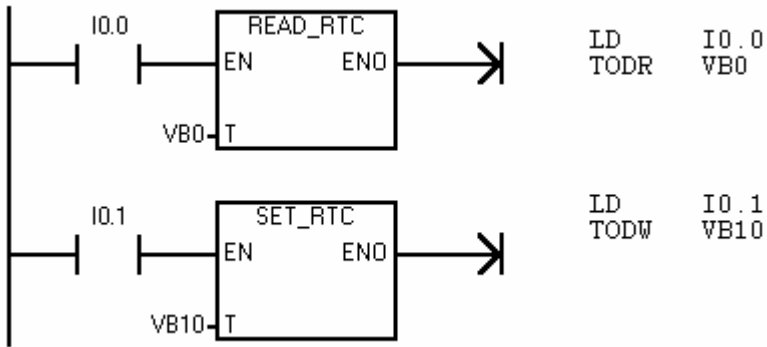
Network 10

RETI

9.27 LỆNH ĐỒNG HỒ

Đồng hồ trong PLC đếm thời gian ngày giờ phút giây, đọc thời gian bằng lệnh `READ_RTC`, chỉnh sửa thời gian bằng lệnh `SET_RTC`. Thời gian chứa trong bảng T 8 byte BCD có dạng sau

T	Năm 00..99
T+1	Tháng 01..12
T+2	Ngày 01..31
T+3	Giờ 00..23
T+4	Phút 00..23
T+5	Giây 00..59
T+6	0
T+7	Ngày trong tuần 0..7, 1: Chủ nhật, 2: thứ hai 0: Không dùng



Các bit nhớ SM cần lưu ý

SMB0, B1

SM 0.0	Luôn luôn ON
SM 0.1	ON ở chu kỳ quét đầu
SM 0.2	ON khi dữ liệu cần lưu trữ bị mất (1 chu kỳ)
SM 0.3	ON khi RUN
SM 0.4	Xung nhịp chu kỳ 1 phút
SM 0.5	Xung nhịp chu kỳ 1 sec
SM 0.6	Xung nhịp có chu kỳ bằng hai lần thời gian quét
SM 0.7	Phản ảnh chế độ hoạt động của PLC
SM 1.0	ON khi kết quả tích là Zero
SM 1.1	ON khi bị tràn
SM 1.2	ON khi kết quả âm
SM 1.3	ON khi chia cho zero
SM 1.4	ON khi bảng bị tràn (xem lệnh bảng)
SM 1.5	ON khi bảng bị trống (xem lệnh bảng)
SM 1.6	ON khi lệnh BCDI không thực hiện được
SM 1.7	ON khi lệnh ATH không thực hiện được

-----O-----

Bài tập gợi ý

- 1/ Nghiên cứu các thiết bị HMI của OMRON và SIEMENS*
- 2/ Nghiên cứu giao thức truyền Modbus*
- 3/ Lập trình giao tiếp với máy tính*