CHƯƠNG 2

TẬP LỆNH S7-200

1. LỆNH LOGIC

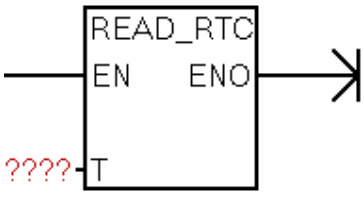
Bao gồm các lệnh:

- Tiếp điểm thường hở (NO).
- Tiếp điểm thường đóng (NC).
- Hàm đảo (NOT).
- Xung cạnh lên (P).
- Xung cạnh xuống (N).
- Ngõ ra out.
- Lệnh SET và RESET.

2. Real Time Clock (RTC)

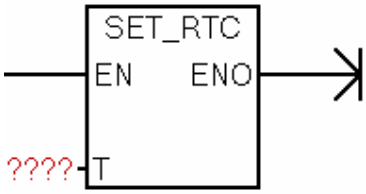
2.1. ĐỌC-RTC *(READ_RTC)*

Lệnh đọc đồng hồ thời gian thực là lệnh đọc thời gian và ngày tháng hiện hành từ đồng hồ và đưa chúng vào bộ đệm 8 byte bắt đầu tại địa chỉ T.

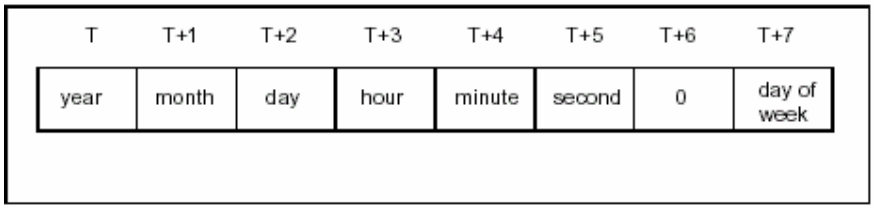


2.2. SET-RTC (*SET_RTC*)

Lệnh set đồng hồ thời gian thực là lệnh ghi thời gian và ngày tháng hiện hành đến đồng hồ bắt đầu tại bộ đệm 8 byte ở địa chỉ T.



Cấu trúc của bộ đệm 8 byte có dạng như sau:



Đồng hồ thời gian thực bao gồm: ngày, tháng, năm, giờ, phút, giây, ngày trong tuần.

Khi cài đặt đồng hồ thời thực cho PLC có 2 cách: trực tiếp từ PC và gián tiếp từ người lập trình. Với phương pháp gián tiếp từ người lập trình, các thông số nhập cho đồng hồ thời gian thực phải ở dạng số BCD.

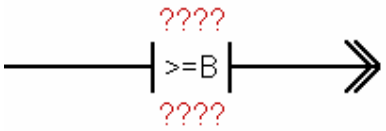
Tổ chức đồng hồ thời gian thực:

- Năm (year): yy = 00 đến 99.
- Tháng (month): mm = 01 đến 12.
- Giờ (Hour): hh = 00 đến 23.
- Phút (minute): mm = 00 đến 59.
- Giây (second): ss = 00 đến 59.
- Ngày trong tuần (Day of week): d = 01 đến 07 (với 01 là mã ngày chủ nhật).

3. CÁC LỆNH SO SÁNH

3.1. SO SÁNH BYTE

Lệnh so sánh byte dùng để so sánh hai giá trị IN1 và IN2 bao gồm IN1 = IN2, IN1 >= IN2, IN1 <= IN2, IN1 < IN2, IN1 > IN2 hoặc IN1 <> IN2.



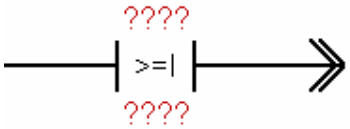
Chú ý, so sánh byte là loại so sánh không dấu.

Khi so sánh hai giá trị IN1 và IN2, kết quả so sánh đúng thì ngõ ra tác động mức cao và ngược lại.

Inputs/Outputs	Operands	Data Types
Inputs	IB, QB, MB, SMB, VB, SB, LB, AC, Constant, *VD, *AC,*LD	BYTE
Outputs (FBD)	I, Q, M, SM, T, C, V, S, L, Power Flow	BOOL

3.2. SO SÁNH INTEGER

Lệnh so sánh integer dùng để so sánh hai giá trị IN1 và IN2 bao gồm IN1 = IN2, IN1 >= IN2, IN1 <= IN2, IN1 < IN2, IN1 > IN2 hoặc IN1 <> IN2.



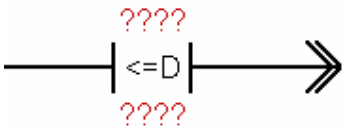
Chú ý, so sánh integer là loại so sánh có dấu (16#7FFFFFFF > 16#80000000).

Khi so sánh hai giá trị IN1 và IN2, kết quả so sánh sẽ đúng thì ngõ ra tác động mức cao và ngược lại.

Inputs/Outputs	Operands	Data Types
Inputs	IW, QW, MW, SW, SMW, T, C, VW, LW, AIW, AC, Constant, *VD, *AC,*LD	INT
Outputs (FBD)	I, Q, M, SM, T, C, V, S, L, Power Flow	BOOL

3.3. SO SÁNH DOUBLE WORD

Lệnh so sánh double word dùng để so sánh hai giá trị IN1 và IN2 bao gồm IN1 = IN2, IN1 >= IN2, IN1 <= IN2, IN1 < IN2, IN1 > IN2 hoặc IN1 <> IN2.



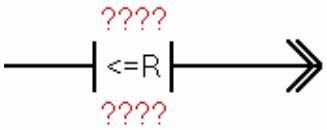
Chú ý, so sánh double integer là loại so sánh có dấu (16#FFFFFFFF > 16#80000000).

Khi so sánh hai giá trị IN1 và IN2, kết quả so sánh sẽ đúng thì ngõ ra tác động mức cao và ngược lại.

Inputs/Outputs	Operands	Data Types
Inputs	ID, QD, MD, SD, SMD, VD, LD, HC, AC, Constant, *VD, *AC, *LD	DINT
Outputs (FBD)	I, Q, M, SM, T, C, V, S, L, Power Flow	BOOL

3.4. So sánh Real

Lệnh so sánh real dùng để so sánh hai giá trị IN1 và IN2 bao gồm IN1 = IN2, IN1 >= IN2, IN1 <= IN2, IN1 < IN2, IN1 > IN2 hoặc IN1 <> IN2.



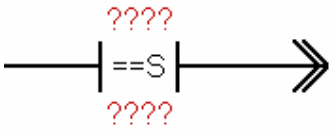
Chú ý, so sánh số thực là loại so sánh có dấu.

Khi so sánh hai giá trị IN1 và IN2, kết quả so sánh sẽ đúng thì ngõ ra tác động mức cao và ngược lại.

Inputs/Outputs	Operands	Data Types
Inputs	ID, QD, MD,SD, SMD, VD, LD, AC, Constant, *VD, *AC, *LD	REAL
Outputs (FBD)	I, Q, M, SM, T, C, V, S, L, Power Flow	BOOL

3.5. SO SÁNH CHUỖI

So sánh 2 chuỗi ký tự ASCII IN1 và IN2 xảy ra các trường hợp: IN1 = IN2, IN1 <> IN2.



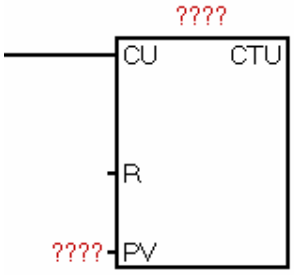
Chiều dài lớn nhất của một chuỗi là 126 byte.

4. BỘ ĐẾM - COUNTER

Có 4 loại Counter: *count up, count up/down, count down và Counter tốc độ cao.*

4.1. COUNT UP

Là bộ đếm lên, giá trị đếm thuộc trong khoảng từ 0 đến 32.767.



Khi ngõ vào chân CU chuyển từ mức thấp thành mức cao thì bộ đếm cộng thêm 1 cho đến khi giá trị đếm hiện hành của C_{xxx} lớn hơn hoặc bằng với PV (preset value) thì C_{xxx} bật lên mức 1 “ON”.

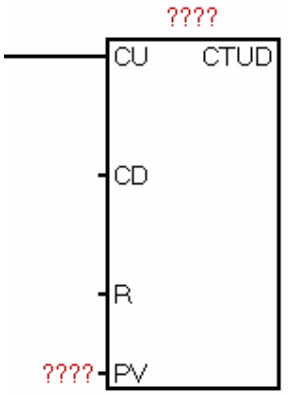
Bộ counter được reset khi ngõ vào R tác động mức 1 (tác động mức cao).

- CU: Chân đếm lên.
- R: Chân Reset.
- PV: Chân đặt giá trị đếm.

Inputs/Outputs	Operands	Data Types
Cxxx	Constant	WORD
CU, CD, LD, R (LAD)	Power Flow	BOOL
CU, CD, R, LD (FBD)	I, Q, M, SM, T, C, V, S, L, Power Flow	BOOL
PV	VW, IW, QW, MW, SMW, LW, SW, AIW, AC, T, C, Constant, *VD, *AC, *LD	INT

4.2. COUNT UP/DOWN

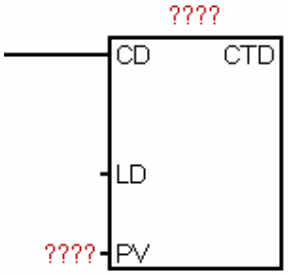
Là bộ đếm lên hoặc xuống (CU/CD).



- CU: Chân đếm lên.
- CD: Chân đếm xuống.
- R: Chân Reset.
- PV: Đặt giá trị đếm.

4.3. COUNT DOWN

Là bộ đếm xuống (CD), khi ngõ vào CD chuyển từ OFF sang ON thì giá tri PV giảm đi 1, nhưng trước khi đếm phải tác động vào chân LD để Counter gán giá trị PV.



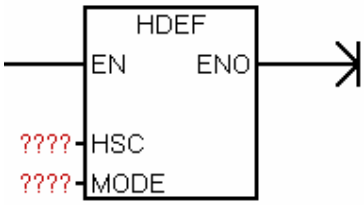
Khi PV = 0 thì C_{xxx} bật lên 1.

- CD: Chân đếm xuống.
- LD: Chân LD Load giá trị PV cho Cxxx.
- PV: Đặt giá trị đếm.

4.4. HIGH-SPEED COUNTER

4.4.1. Lệnh định nghĩa HSC - HDEF

Lệnh HDEF chọn mode hoạt động và xác định HSC. Chọn mode xác định xung đếm (clock), hướng đếm, chức năng start, và reset của HSC.

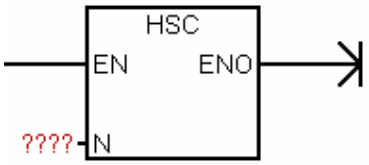


PLC 221 và PLC 222 không hỗ trợ HSC1 và HSC2. Chỉ sử dụng 1 lệnh HDEF cho mỗi HSC.

Có tất cả 6 HSC có giá trị từ 0 đến 5 và mỗi HSC có tối đa 12 Mode có giá trị từ 0 đến 11.

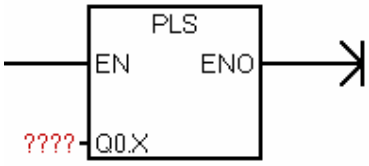
4.4.2. LỆNH HSC

Lệnh HSC định cấu hình và điều khiển HSC. Thông số N xác định HSC. Mỗi HSC xác định xung đếm, hướng đếm, start và reset.



4.5. NGÕ RA XUNG

Lệnh PLS sử dụng để điều khiển xung PTO (Pulse Train Output) và PWM (Pulse Width Modulation) ở ngõ ra tốc độ cao Q0.0 và Q0.1.

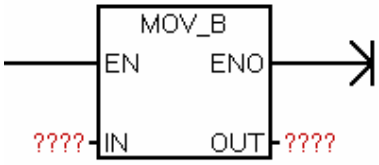


PTO cung cấp xung vuông (50% chu kỳ), còn PWM cung cấp xung với độ rộng xung tùy ý do người lập trình.

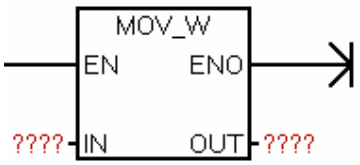
5. CÁC LỆNH DI CHUYỂN (MOVE)

5.1. MOVE BYTE, WORD, DOUBLE WORD VÀ MOVE REAL

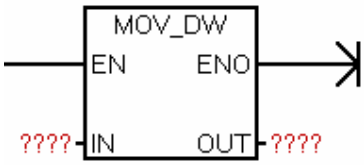
Lệnh Move Byte (***MOV_B***) di chuyển byte ngõ vào IN đến byte ngõ ra OUT mà không làm thay đổi giá trị ngõ IN.



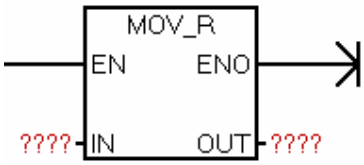
Lệnh Move Word (***MOV_W***) di chuyển word ngõ vào IN đến word ngõ ra OUT mà không làm thay đổi giá trị ngõ IN.



Lệnh Move Double Word (***MOV_DW***) di chuyển Double word ngõ vào IN đến Double word ngõ ra OUT mà không làm thay đổi giá trị ngõ IN.



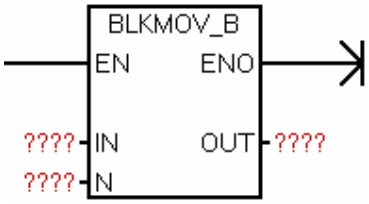
Lệnh Move Real (***MOV_R***) di chuyển số thực 32 bit ngõ vào IN đến Double word ngõ ra OUT mà không làm thay đổi giá trị ngõ IN.



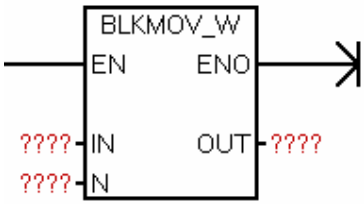
Move...	Inputs/Outputs	Operands	Data Types
Byte	IN	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
	OUT	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD	BYTE
Word	IN	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, Constant, AC *VD, *AC, *LD	WORD, INT
	OUT	VW, T, C, IW, QW, SW, MW, SMW, LW, AC, AQW, *VD, *AC, *LD	WORD, INT
Double Word	IN	VD, ID, QD, MD, SD, SMD, LD, HC, &VB, &IB, &QB, &MB, &SB, &T, &C, AC, Constant, *VD, *AC, *LD	DWORD, DINT
	OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD	DWORD, DINT
Real	IN	VD, ID, QD, MD, SD, SMD, LD, AC, Constant, *VD, *AC, *LD	REAL
	OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD	REAL

5.2. MOVE KHỐI BYTE, WORD, DOUBLE WORD

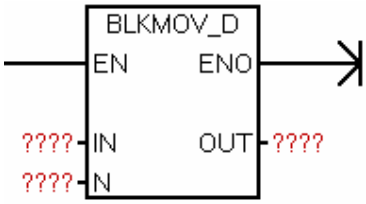
Lệnh Move khối Byte (**BLK*MOV_B***) di chuyển số N byte từ địa chỉ ngõ vào IN đến địa chỉ ngõ ra OUT mà không làm thay đổi giá trị ngõ IN. N thuộc khoảng từ 1 đến 255.



Lệnh Move khối Word (**BLK*MOV_W***) di chuyển số N word từ địa chỉ ngõ vào IN đến địa chỉ ngõ ra OUT mà không làm thay đổi giá trị ngõ IN. N thuộc khoảng từ 1 đến 255.

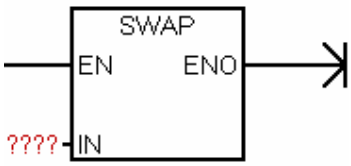


Lệnh Move khối Double Word (**BLK*MOV_DW***) di chuyển số N Double word từ địa chỉ ngõ vào IN đến địa chỉ ngõ ra OUT mà không làm thay đổi giá trị ngõ IN. N thuộc khoảng từ 1 đến 255.



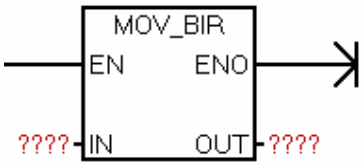
5.3. SWAP BYTES

Lệnh Swap bytes làm thay đổi byte có trọng số thấp thành byte có trọng số cao và ngược lại.



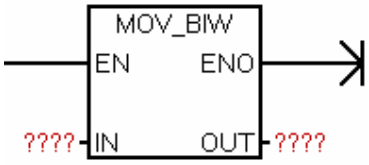
5.4. MOVE BYTE ĐỌC TỨC THỜI

Lệnh này dùng để đọc ngõ vào vật lý IN và ghi kết quả ra chân OUT.



5.5. MOVE BYTE GHI TỨC THỜI

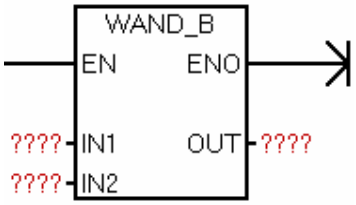
Lệnh này dùng để ghi vị trí chân IN và ghi kết quả ra chân OUT vật lý.



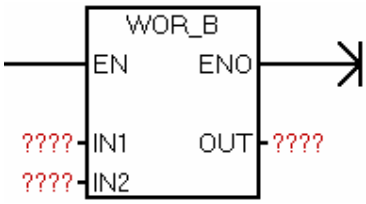
6.CÁC LỆNH LOGIC NHỊ PHÂN

6. 1. AND BYTE, OR BYTE, EXCLUSIVE OR BYTE

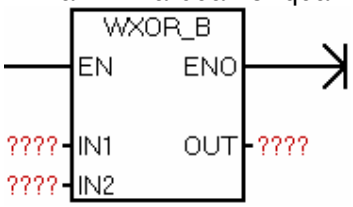
Lệnh And Byte (**WAND_B**) là lệnh AND các bit tương ứng của 2 byte ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng byte.



Lệnh Or Byte (**WOR_B**) là lệnh OR các bit tương ứng của 2 byte ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng byte.



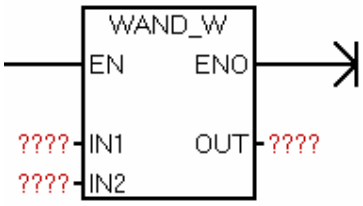
Lệnh Exclusive Or Byte (**WXOR_B**) là lệnh XOR các bit tương ứng của 2 byte ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng byte.



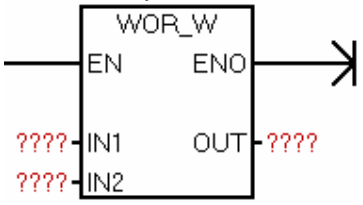
Inputs/Outputs	Operands	Data Types
IN1, IN2	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VB, IB, QB, MB,SB, SMB, LB, AC, *VD, *AC, *LD	BYTE

6. 2. AND WORD, OR WORD, EXCLUSIVE OR WORD

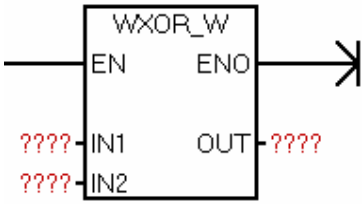
Lệnh And Word (**WAND_W**) là lệnh AND các bit tương ứng của 2 word ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng word.



Lệnh Or Word (**WOR_W**) là lệnh OR các bit tương ứng của 2 word ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng word.



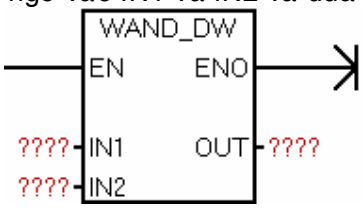
Lệnh Exclusive Or Word (**WXOR_W**) là lệnh XOR các bit tương ứng của 2 word ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng word.



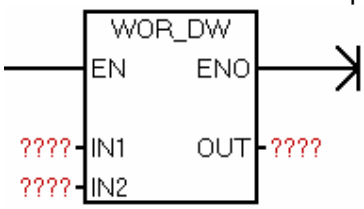
Inputs/Outputs	Operands	Data Types
IN1, IN2	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, AC, Constant, *VD, *AC, *LD	WORD
OUT	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, *VD, *AC, *LD	WORD

6.3. AND DOUBLE WORD, OR DOUBLE WORD, EXCLUSIVE OR DOUBLE WORD

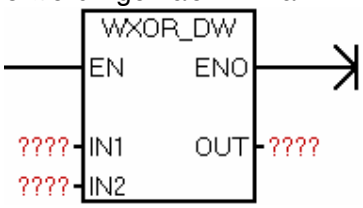
Lệnh And Double Word (**WAND_DW**) là lệnh AND các bit tương ứng của 2 Double word ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng Double word.



Lệnh Or Double Word (**WOR_DW**) là lệnh OR các bit tương ứng của 2 Double word ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng Double word.



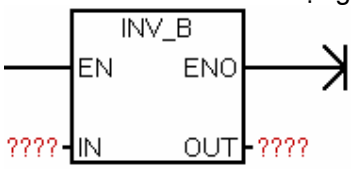
Lệnh Exclusive Or Double word (**WXOR_W)** là lệnh XOR các bit tương ứng của 2 Double word ngõ vào IN1 và IN2 và đưa kết quả ra chân OUT ở dạng Double word.



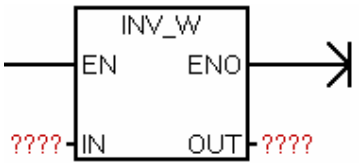
Inputs/Outputs	Operands	Data Types
IN1, IN2	VD, ID, QD, MD, SD, SMD, AC, LD, HC, Constant, *VD, *AC, SD, *LD	DWORD
OUT	VD, ID, QD, MD, SMD, LD, AC, *VD, *AC, SD, *LD	DWORD

6.4. INVERT BYTE, INVERT WORD, INVERT DOUBLE WORD

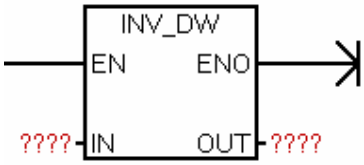
Lệnh Invert Byte (**INV_B**) thực hiện lấy bù bít với bit tương ứng và đưa kết quả chuyển đổi ra chân OUT ở dạng byte.



Lệnh Invert word (INV_W) thực hiện lấy bù bít với bit tương ứng và đưa kết quả chuyển đổi ra chân OUT ở dạng word.



Lệnh Invert double word (INV_DW) thực hiện lấy bù bít với bit tương ứng và đưa kết quả chuyển đổi ra chân OUT ở dạng double word.

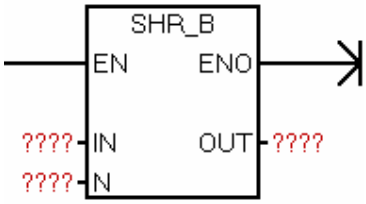


Invert...	Inputs/Outputs	Operands	Data Types
Byte	IN	VB, IB, QB, MB,SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
	OUT	VB, IB, QB, MB, SB, SMB, LB, AC,*VD, *AC, *LD	BYTE
Word	IN	VW, IW, QW, MW, SW, SMW, T, C, AIW, LW, AC, Constant, *VD, *AC, *LD	WORD
	OUT	VW, IW, QW, MW, SW, SMW, T, C, LW, AC, *VD, *AC, *LD	WORD
Double Word	IN	VD, ID, QD, MD, SD, SMD, LD, HC, AC, Constant, *VD, *AC, *LD	DWORD
	OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD	DWORD

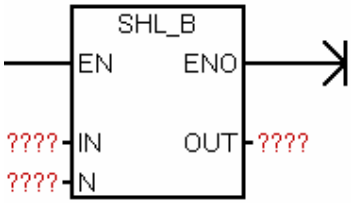
7. CÁC LỆNH DỊCH VÀ QUAY (SHIFT/ROTATE)

7.1. DỊCH VÀ QUAY DẠNG BIT

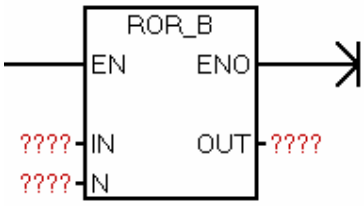
SHR_B dịch chuyển Byte ngõ vào ở chân IN qua phải (N) bit và đưa kết quả ra OUT.



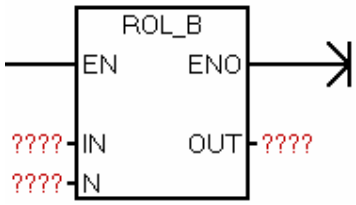
SHL_B dịch chuyển Byte ngõ vào ở chân IN qua trái (N) bit và đưa kết quả ra OUT.



ROR_B quay giá trị Byte ngõ vào ở chân IN qua phải N bit và đưa kết quả vào OUT.



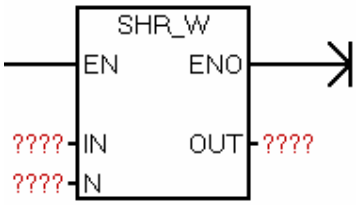
ROL_B quay giá trị Byte ngõ vào ở chân IN qua trái N bit và đưa kết quả vào OUT.



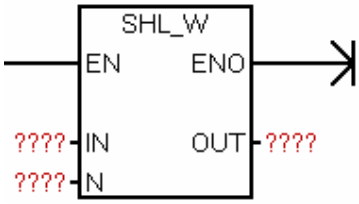
Inputs/Outputs	Operands	Data Types
IN	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD	BYTE
N	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE

7.2. DỊCH VÀ QUAY DẠNG WORD

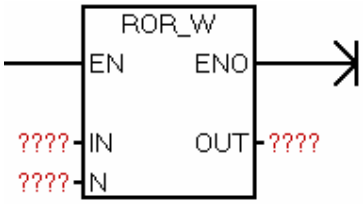
SHR_W dịch chuyển Word ngõ vào ở chân IN qua phải N bit và đưa kết qua ra OUT.



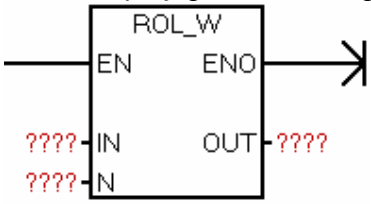
SHL_W dịch chuyển Word ngõ vào ở chân IN qua trái N bit và đưa kết qua ra OUT.



ROR_W quay giá trị Word ngõ vào ở chân IN qua phải N bit và đưa kết quả vào OUT.



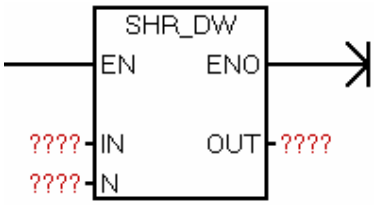
ROL_W quay giá trị Word ngõ vào ở chân IN qua trái N bit và đưa kết quả vào OUT



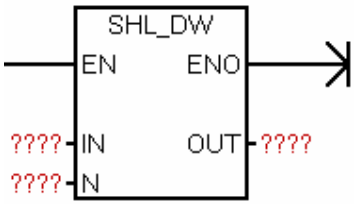
Inputs/Outputs	Operands	Data Types
IN	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, AC, Constant, *VD, *AC, *LD	WORD
N	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, *VD, *AC, *LD	WORD

7.3. DỊCH VÀ QUAY DẠNG DOUBLE WORD

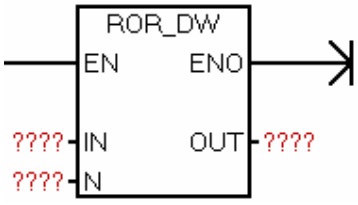
SHR_DW dịch chuyển Double Word ngõ vào ở chân IN qua phải (N) bit và đưa kết qua ra chân OUT.



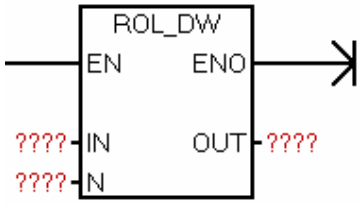
SHL_DW dịch chuyển Double Word ngõ vào ở chân IN qua trái (N) bit và đưa kết qua ra chân OUT.



ROR_DW quay giá trị Double Word ngõ vào ở chân IN qua phải N bit và đưa kết quả vào chân OUT.



ROL_DW quay giá trị Double Word ngõ vào ở chân IN qua trái N bit và đưa kết quả vào chân OUT.

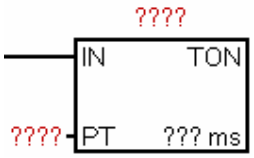


Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SD, SMD, LD, AC, HC, Constant, *VD, *AC, *LD	DWORD
N	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD	DWORD

8. CÁC LỆNH TIMER

8.1. TIMER ON-DELAY (TON)

TON đếm thời gian khi ngõ vào cho phép ON. Khi giá trị hiện hành (Txxx) lớn hơn hoặc bằng giá trị đặt (PT), thì bit của Timer ON. Giá trị hiện hành Timer sẽ bị xóa khi chân ngõ vào OFF.



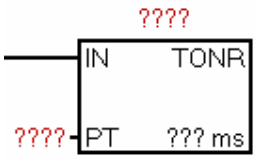
Giá trị thời gian đặt lớn nhất là 32767.

Các Timer TON, TONR, và TOF có ba độ phân giải là 1ms, 10ms, 100ms. Giá trị thời gian thực khi Timer hoạt động bằng độ phân giải nhân cho giá trị đặt PV.

Timer Type	Resolution	Maximum Value	Timer Number
TONR	1 ms	32.767 s	T0, T64
	10 ms	327.67 s	T1-T4, T65-T68
	100 ms	3276.7 s	T5-T31, T69-T95
TON, TOF	1 ms	32.767 s	T32, T96
	10 ms	327.67 s	T33-T36, T97-T100
	100 ms	3276.7 s	T37-T63, T101-T255

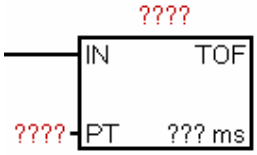
8.2. TIMER ON-DELAY CÓ NHỚ (TONR)

TONR đếm thời gian khi ngõ vào cho phép ON. Khi giá trị hiện hành (Txxx) lớn hơn hoặc bằng giá trị đặt (PT), thì bit của Timer ON. Giá trị hiện hành Timer sẽ vẫn ON khi chân ngõ vào OFF.



8.3. TIMER OFF-DELAY (TOF)

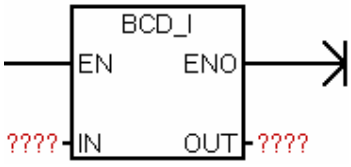
TOF dùng để trì hoãn ngõ ra OFF sau một khoảng thời gian khi ngõ vào OFF. Khi ngõ vào ngõ vào ON, bít Timer ON tức thời và giá trị hiện hành set đến 0. Khi ngõ vào OFF, timer đếm thời gian đến PV. Khi đếm xong ngõ ra OFF.



9. CÁC LỆNH CHUYỂN ĐỔI

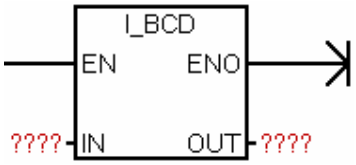
9.1. BCD thành INTEGER (BCD_I)

Lệnh BCD_I là lệnh chuyển đổi giá trị BCD ở chân IN thành giá trị integer và đưa kết quả ra chân OUT. Giá trị ở chân IN nằm trong khoảng 0 đến 9999 BCD.



9.2. INTEGER thành BCD (I_BCD)

Lệnh **I_BCD** là lệnh chuyển đổi giá trị Integer ở chân IN thành giá trị BCD và đưa kết quả ra chân OUT. Giá trị ở chân IN nằm trong khoảng 0 đến 9999 Integer.

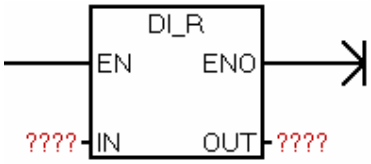


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM1.6 (lỗi BCD), SM4.3 (run-time), 0006 (địa chĩ gián tiếp).

Inputs/Outputs	Operands	Data Types
IN	VW, T, C, IW, QW, MW, SMW, LW, AC, AIW, Constant, *VD, *AC, SW, *LD	WORD
OUT	VW, T, C, IW, QW, MW, SMW, LW, AC, *VD, *AC, SW, *LD	WORD

9.3. DOUBLE INTEGER THÀNH INTEGER

Lệnh **DI_R** là lệnh chuyển đổi giá trị số Integer có dấu 32 bit ở chân IN thành giá trị số thực 32 bit và đưa kết quả ra chân OUT.

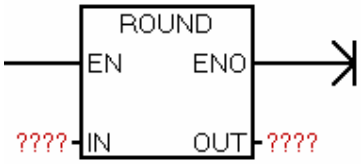


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chĩ gián tiếp).

Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SMD, AC, LD, HC, Constant, *VD, *AC, SD, *LD	DINT
OUT	VD, ID, QD, MD, SMD, LD, AC, *VD, *AC, SD, *LD	REAL

9.4. ROUND

Lệnh **ROUND** là lệnh chuyển đổi giá trị số thực ở chân IN thành giá trị double integer 32 bit và đưa kết quả ra chân OUT. Nếu kết quả là số thập phân 0.5 hoặc lớn hơn thì giá trị kết quả sẽ làm tròn.

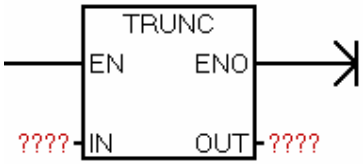


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chĩ gián tiếp), SM1.1 (overflow).

Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SMD, AC, LD, Constant, *VD, *AC, SD, *LD	REAL
OUT	VD, ID, QD, MD, SMD, LD, AC, *VD, *AC, SD, *LD	DINT

9.5. TRUNCATE

Lệnh TRUNCATE là lệnh chuyển đổi giá trị số thực 32 bit ở chân IN thành giá trị integer 32 bit có dấu và đưa kết quả ra chân OUT. Chỉ có phần nguyên được chuyển đổi còn phần sau dấu phẩy bị loại bỏ.

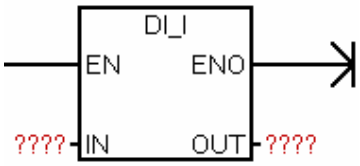


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chỉ gián tiếp), SM1.1 (overflow).

Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SMD, LD, AC, Constant, *VD, *AC, SD, *LD	REAL
OUT	VD, ID, QD, MD, SMD, LD, AC, *VD, *AC, SD, *LD	DINT

9.6.DOUBLE INTEGER THÀNH INTEGER

Lệnh DI_I là lệnh chuyển đổi giá trị số double integer 32 bit ở chân IN thành giá trị integer 16 bit và đưa kết quả ra chân OUT.



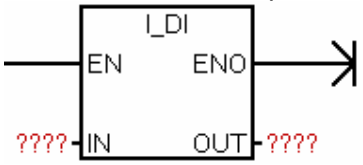
Nếu giá trị chuyển đổi lớn hơn giá trị 16 bit thì cờ tràn sẽ được set và ngõ ra sẽ không đúng.

Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chỉ gián tiếp), SM1.1 (overflow).

Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SMD, AC, LD, HC, Constant, *VD, *AC, SD, *LD	DINT
OUT	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, *VD, *LD, *AC	INT

9.7. INTEGER to DOUBLE INTEGER

Lệnh I_DI là lệnh chuyển đổi giá trị số integer 16 bit ở chân IN thành giá trị double integer 32 bit và đưa kết quả ra chân OUT.



Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chỉ gián tiếp).

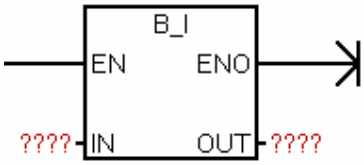
Inputs/Outputs	Operands	Data Types
IN	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, AC, Constant, *AC, *VD, *LD	INT
OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *LD, *AC	DINT

9.8. INTEGER to REAL

Để chuyển đổi số integer thành số thực, sử dụng lệnh I_DI rồi sau đó dùng lệnh DI_R.

9.9. BYTE to INTEGER

Lệnh B_I là lệnh chuyển đổi giá trị số byte ở chân IN thành giá trị integer và đưa kết quả ra chân OUT. Chú ý byte này là giá trị không có dấu.

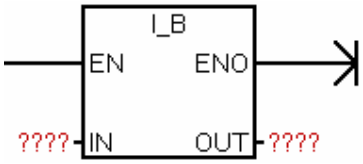


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chĩ gián tiếp).

Inputs/Outputs	Operands	Data Types
IN	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *AC, *VD, *LD	BYTE
OUT	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, *VD, *LD, *AC	INT

9.10. INTEGER to BYTE

Lệnh I_B là lệnh chuyển đổi giá trị số integer (word) ở chân IN thành giá trị byte và đưa kết quả ra chân OUT.



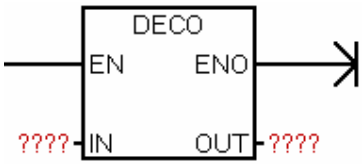
Các giá trị từ 0 đến 255 được chuyển đổi còn các giá trị khác không được chuyển đổi vì chúng sẽ bị tràn và ngõ ra sẽ xuất tín hiệu lỗi.

Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chĩ gián tiếp), SM1.1 (tràn).

Inputs/Outputs	Operands	Data Types
IN	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, AC, Constant, *VD, *LD, *AC	INT
OUT	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD	BYTE

9.11. Decode

Lệnh **DECO** chuyển đổi giá trị byte ngõ vào ở chân IN thành giá trị bit ở ngõ ra OUT tương ứng 16 bit.

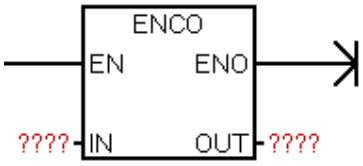


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chĩ gián tiếp), SM1.1 (tràn).

Inputs/Outputs	Operands	Data Types
IN	VB, IB, QB, MB, SMB, LB, SB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VW, IW, QW, MW, SMW, LW, SW, AQW, T, C, AC, *VD, *AC, *LD	WORD

9.12. Encode

Lệnh **ENCO** chuyển đổi giá trị bit ngõ vào ở chân IN thành giá trị số byte ở ngõ ra OUT.

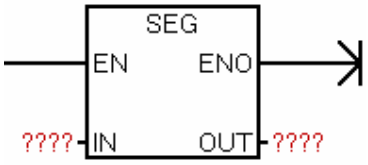


Chú ý: các điều kiện lỗi xảy ra khi: ENO = 0, SM4.3 (run-time), 0006 (địa chĩ gián tiếp), SM1.1 (tràn).

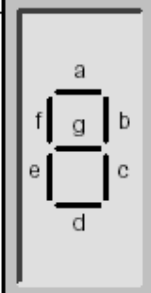
Inputs/Outputs	Operands	Data Types
IN	VW, T, C, IW, QW, MW, SMW, AC, LW, AIW, Constant, *VD, *AC, SW, *LD	WORD
OUT	VB, IB, QB, MB, SMB, LB, AC, *VD, *AC, SB, *LD	BYTE

9.13. Segment

Là lệnh xuất LED 7 đoạn với nội dung và cấu trúc như hình sau.

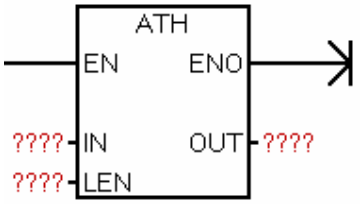


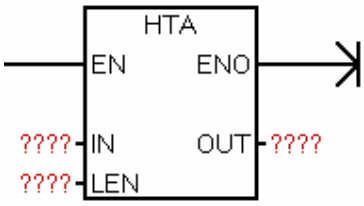
(IN) LSD	Segment Display	(OUT) -gfe dcba
0	0	0011 1111
1	1	0000 0110
2	2	0101 1011
3	3	0100 1111
4	4	0110 0110
5	5	0110 1101
6	6	0111 1101
7	7	0000 0111



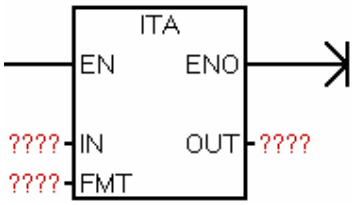
(IN) LSD	Segment Display	(OUT) -gfe dcba
8	8	0111 1111
9	9	0110 0111
A	A	0111 0111
B	b	0111 1100
C	C	0011 1001
D	d	0101 1110
E	E	0111 1001
F	F	0111 0001

9.14. ASCII to HEX, HEX to ASCII

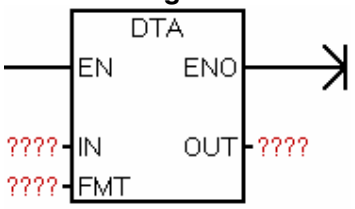




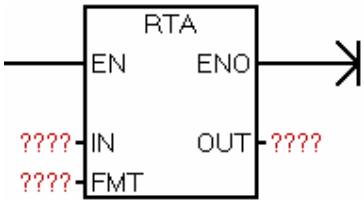
9.15. Integer to ASCII



9.16. Double Integer to ASCII

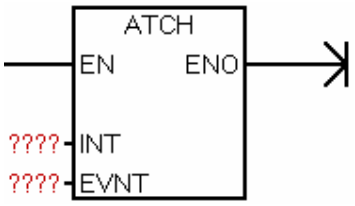


9.17. Real to ASCII

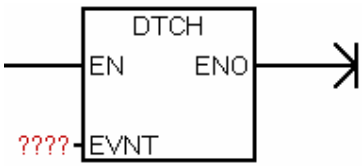


10. CÁC LỆNH NGẮT

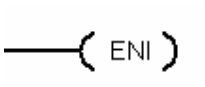
10.1.



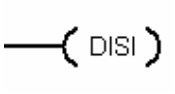
10.2.



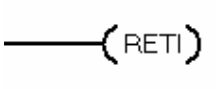
10.3.



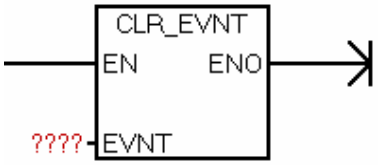
10.4.



10.5.



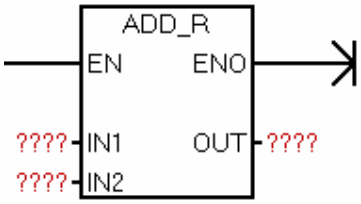
10.6.



11. CÁC LỆNH TOÁN HỌC DẤU PHẨY ĐỘNG

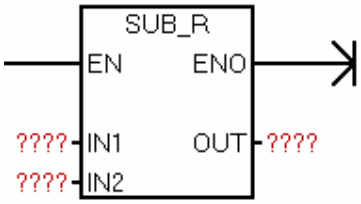
11.1. Cộng

Lệnh ADD_R là lệnh cộng 2 số thực (IN1 và IN2) 32 bit và kết quả (OUT) là số thực 32 bit.



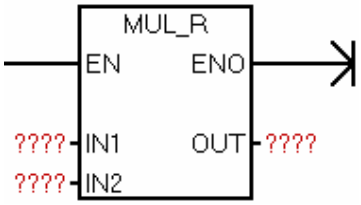
11.2. Trừ

Lệnh SUB_R là lệnh trừ 2 số thực (IN1 và IN2) 32 bit và kết quả (OUT) là số thực 32 bit.



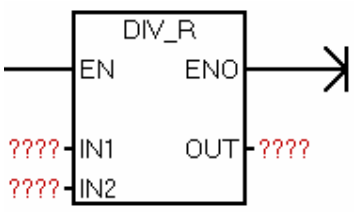
11.3. Nhân

Lệnh MUL_R là lệnh nhân 2 số thực (IN1 và IN2) 32 bit và kết quả (OUT) là số thực 32 bit.



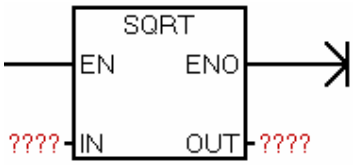
11.4. Chia

Lệnh DIV_R là lệnh chia 2 số thực (IN1 và IN2) 32 bit và kết quả (OUT) là số thực 32 bit.



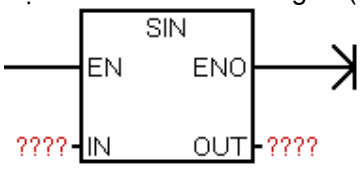
11.5. Căn Bậc hai

Lệnh SQRT lấy căn bậc hai của số thực 32 bit (IN) và kết quả (OUT) là số thực 32 bit.

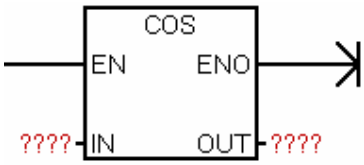


11.6. Sin

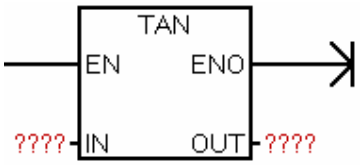
Lệnh SIN tính sin của góc (IN) số thực 32 bit và kết quả (OUT) 32 bit số thực.



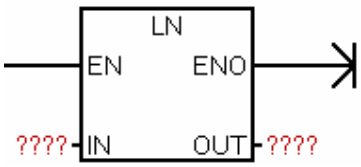
11.7. Cos



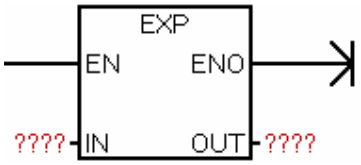
11.8. Tan



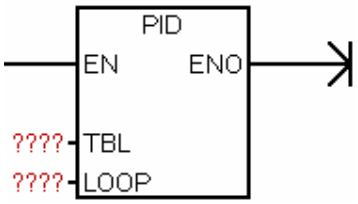
11.9. LN



11.10. EXP



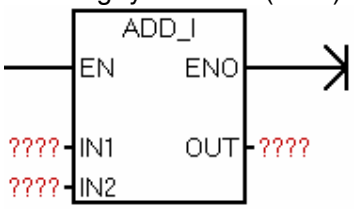
11.11. PID



12. CÁC LỆNH TOÁN HỌC SỐ NGUYÊN

12.1. CỘNG 16 BIT

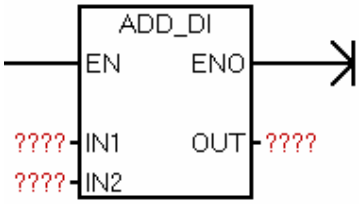
Lệnh ADD_I (Add Integer) là lệnh cộng 2 số nguyên 16 bit (IN1 và IN2) và kết quả cũng là số nguyên 16 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VW, IW, QW, MW, SW, SMW, LW, AIW, T, C, AC, Constant, *VD, *AC, *LD	INT
OUT	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, *VD, *AC, *LD	INT

12.2. CỘNG 32 BIT

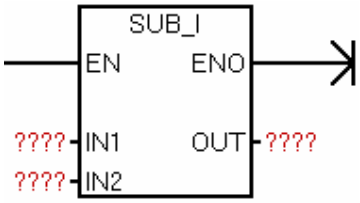
Lệnh ADD_DI (Add Double Integer) là lệnh cộng 2 số nguyên 32 bit (IN1 và IN2) và kết quả cũng là số nguyên 32 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VD, ID, QD, MD, SMD, SD, LD, AC, HC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *AC, *LD	DINT

12.3. TRỪ 16 BIT

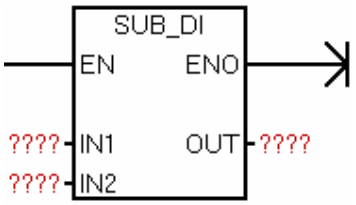
Lệnh SUB_I (Subtract Integer) là lệnh trừ 2 số nguyên 16 bit (IN1 và IN2) và kết quả cũng là số nguyên 16 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VD, ID, QD, MD, SMD, SD, LD, AC, HC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *AC, *LD	DINT

12.4. TRỪ 32 BIT

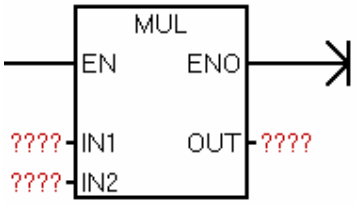
Lệnh SUB_DI (Add Double Integer) là lệnh trừ 2 số nguyên 32 bit (IN1 và IN2) và kết quả cũng là số nguyên 32 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VD, ID, QD, MD, SMD, SD, LD, AC, HC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *AC, *LD	DINT

12.5. NHÂN

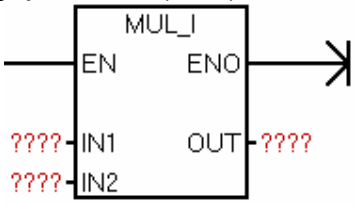
Lệnh MUL (Multiply Integer to Double Integer) nhân 2 số nguyên 16 bit (IN1 và IN2) và kết quả là số nguyên 32 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VW, IW, QW, MW, SW, SMW, LW, AC, AIW, T, C, Constant, *VD, *AC, *LD	INT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *LD, *AC	DINT

12.6. NHÂN 16 BIT

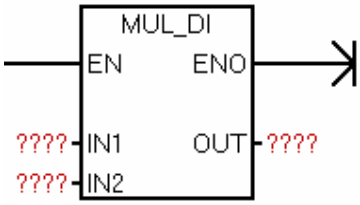
Lệnh MUL_I (Multiply Integer) nhân 2 số nguyên 16 bit (IN1 và IN2) và kết quả là số nguyên 16 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VW, IW, QW, MW, SW, SMW, LW, AIW, T, C, AC, Constant, *VD, *AC, *LD	INT
OUT	VW, QW, IW, MW, SW, SMW, LW, T, C, AC, *VD, *LD, *AC	INT

12.7. NHÂN 32 BIT

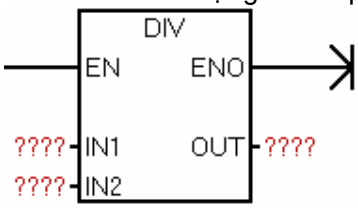
Lệnh MUL_DI (Multiply Integer) nhân 2 số nguyên 32 bit (IN1 và IN2) và kết quả là số nguyên 32 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VD, ID, QD, MD, SMD, SD, LD, HC, AC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *LD, *AC	DINT

12.8. CHIA

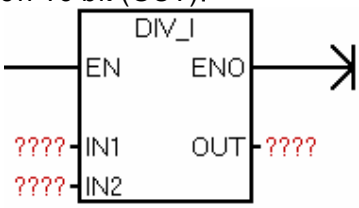
Lệnh DIV (Divide Integer to Double Integer) chia 2 số nguyên 16 bit (IN1 và IN2) và kết quả là số nguyên 32 bit (OUT) trong đó phần dư là 16 bit có trọng số cao và phần nguyên là 16 bit có trọng số thấp .



Inputs/Outputs	Operands	Data Types
IN1, IN2	VW, IW, QW, MW, SW, SMW, LW, AC, AIW, T, C, Constant, *VD, *AC, *LD	INT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *LD, *AC	DINT

12.9. CHIA 16 BIT

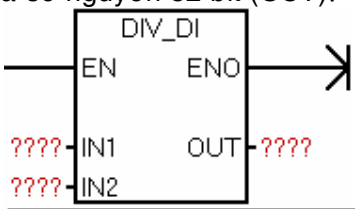
Lệnh DIV_I (Divide Integer) chia 2 số nguyên 16 bit (IN1 và IN2) và kết quả là số nguyên 16 bit (OUT).



Inputs/Outputs	Operands	Data Types
IN1, IN2	VW, IW, QW, MW, SW, SMW, LW, AIW, T, C, AC, Constant, *VD, *AC, *LD	INT
OUT	VW, QW, IW, MW, SW, SMW, LW, T, C, AC, *VD, *LD, *AC	INT

12.10. CHIA 32 BIT

Lệnh DIV_D (Divide Double Integer) chia 2 số nguyên 32-bit (IN1 và IN2) và kết quả là số nguyên 32 bit (OUT).

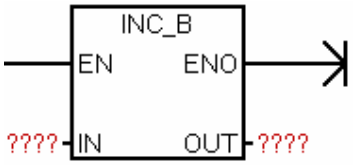


Inputs/Outputs	Operands	Data Types
IN1, IN2	VD, ID, QD, MD, SMD, SD, LD, HC, AC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SMD, SD, LD, AC, *VD, *LD, *AC	DINT

12.11.

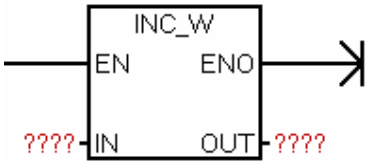
The Increment Byte and Decrement Byte instructions add or subtract 1 to or from the input byte (IN) and place the result into the variable specified by OUT.

Increment and decrement byte operations are unsigned.



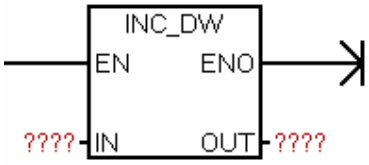
Inputs/Outputs	Operands	Data Types
IN	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VB, IB, QB, MB, SB, SMB, LB, AC,*VD, *AC, *LD	BYTE

12.12.



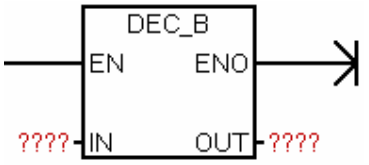
12.13.

The Increment Double Word and Decrement Double Word instructions add or subtract 1 to or from the input double word (IN) and place the result in OUT.



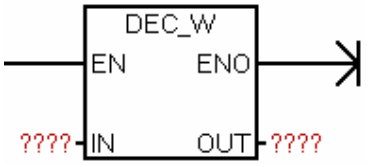
Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SD, SMD, LD, AC, HC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD	DINT

12.14.

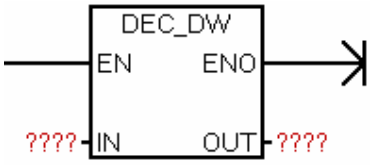


Inputs/Outputs	Operands	Data Types
IN	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
OUT	VB, IB, QB, MB, SB, SMB, LB, AC,*VD, *AC, *LD	BYTE

12.15.



12.16.

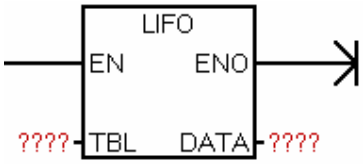


Inputs/Outputs	Operands	Data Types
IN	VD, ID, QD, MD, SD, SMD, LD, AC, HC, Constant, *VD, *AC, *LD	DINT
OUT	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD	DINT

13. Table

13.1. LIFO

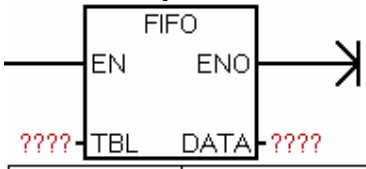
The Last-In-First-Out (LIFO) instruction moves the newest (or last) entry in the table to the output memory address by removing the last entry in the table (TBL) and moving the value to the location specified by DATA. The entry count in the table is decremented for each instruction execution.



Inputs/Outputs	Operands	Data Types
TABLE	VW, IW, QW, MW, SW, SMW, LW, T, C, *VD, *AC, *LD	INT
DATA	VW, IW, QW, MW, SW, SMW, LW, AQW, T, C, AC, *VD, *AC, *LD	WORD

13.2. FIFO

The First-In-First-Out (FIFO) instruction moves the oldest (or first) entry in a table to the output memory address by removing the first entry in the table (TBL) and moving the value to the location specified by DATA. All other entries of the table are shifted up one location. The entry count in the table is decremented for each instruction execution.

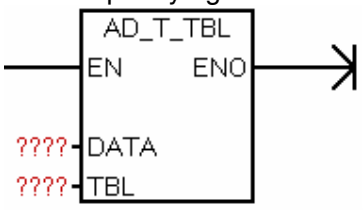


Inputs/Outputs	Operands	Data Types
TABLE	VW, IW, QW, MW, SW, SMW, LW, T, C, *VD, *AC, *LD	INT
DATA	VW, IW, QW, MW, SW, SMW, LW, AC, AQW, T, C, *VD, *AC, *LD	WORD

13.3. ADD TO TABLE

Lệnh Add To Table (ATT) instruction adds word values (DATA) to the table (TBL).

The first value of the table is the maximum table length (TL). The second value is the entry count (EC), which specifies the number of entries in the table. New data are added to the table after the last entry. Each time new data are added to the table, the entry count is incremented.A table may have up to 100 entries, excluding both parameters specifying the maximum number of entries and the actual number of entries.

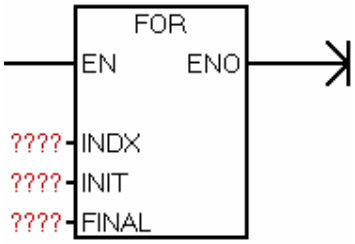


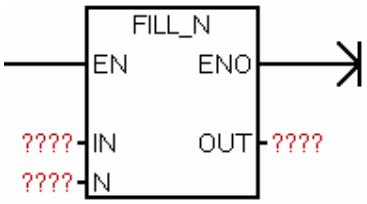
Inputs/Outputs	Operands	Data Types
DATA	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, AC, Constant, *VD, *AC, *LD	INT
TBL	VW, IW, QW, MW, SW, SMW, LW, T, C, *VD, *AC, *LD	WORD

13.4. Memory Fill

The Memory Fill (FILL) instruction writes N consecutive words, beginning at address OUT, with the word value contained in address IN.

N has a range of 1 to 255.





Inputs/Outputs	Operands
IN	VW, IW, QW, MW, SW, SMW, LW, AIW, T, C, AC, Constant, *V *AC, *LD
N	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD
OUT	VW, IW, QW, MW, SW, SMW, LW, T, C, AQW, *VD, *AC, *LD

13.5. Table Find

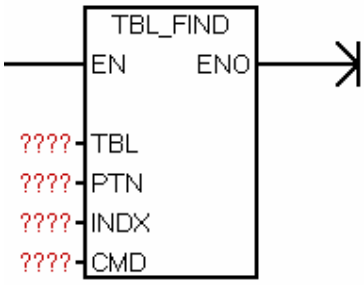
The Table Find (TBL) instruction searches a table (TBL) for data that matches certain criteria.

The Table Find instruction searches the table, starting with the table entry specified by INDX, for the data value (PTN) that matches the search criteria defined by CMD.

The command parameter (CMD) is given a numeric value of 1 to 4 that corresponds to =, <>, <, and >, respectively.If a match is found, the INDX points to the matching entry in the table.

To find the next matching entry, the INDX must be incremented before invoking the Table Find instruction again. If a match is not found, the INDX has a value equal to the entry count.A table may have up to 100 entries.

The data entries (area to be searched) are numbered from 0 to a maximum value of 99.



Inputs/Outputs	Operands	Data Types
SRC	VW, IW, QW, MW, SMW, LW, T, C, *VD, *AC, *LD	WORD
PTN	VW, IW, QW, MW, SW, SMW, AIW, LW, T, C, AC, Constant, *VD, *AC, *LD	INT
INDX	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, *VD, *AC, *LD	WORD
CMD	Constant	BYTE

14. CÁC LỆNH ĐIỀU KHIỂN CHƯƠNG TRÌNH

14.1. Lệnh FOR

Lệnh FOR thực thi các lệnh giữa FOR và NEXT. Cần xác định giá trị index hoạc số đếm lặp vòng (INDX), giá trị bắt đầu (INIT), và giá trị kết thúc (FINAL).

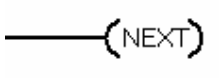
14.2. Lệnh NEXT

Lệnh NEXT đánh dấu sự kết thúc của vòng lặp FOR. Sử dụng FOR/NEXT để mô tả vòng lặp được lặp lại với số lần xác định.

Mỗi lệnh FOR cần phải có lệnh NEXT. You can nest FOR/NEXT loops (place a FOR/NEXT loop within a FOR/NEXT loop) to a depth of eight.

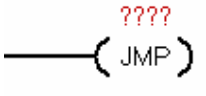
For example, given an INIT value of 1 and a FINAL value of 10, the instructions between the FOR and the NEXT are executed 10 times with the INDX value being incremented: 1, 2, 3, ...10.If the starting value is greater than the final value, the loop is not executed.

After each execution of the instructions between the FOR and the NEXT instruction, the INDX value is incremented and the result is compared to the final value. If the INDX is greater than the final value, the loop is terminated.



14.3.

The Jump to Label (JMP) instruction performs a branch to the specified label (n) within the program. When a jump is taken, the top of stack value is always a logical 1.

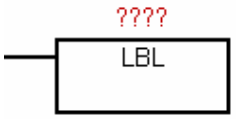


14.4.

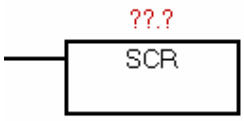
The Label (LBL) instruction marks the location of the jump destination (n).

You can use the Jump instruction in the main program, in subroutines, or in interrupt routines. The Jump and its corresponding Label instruction must always be located within the same segment of code (either the main program, a subroutine, or an interrupt routine).You cannot jump from the main program to a label in either a subroutine or an interrupt routine.

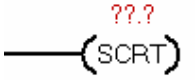
Likewise, you cannot jump from a subroutine or interrupt routine to a label outside that subroutine or interrupt routine.You can use a Jump instruction within an SCR segment, but the corresponding Label instruction must be located within the same SCR segment.



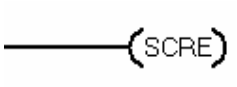
14.5.



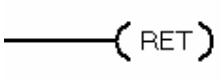
14.6.



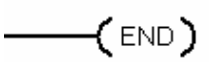
14.7.



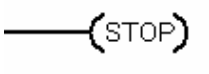
14.8.



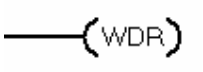
14.9.



14.10.



14.11.



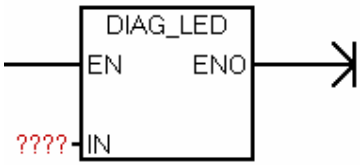
14.12.

Nếu thông số ngõ vào IN có giá trị bằng zero, thì then set the diagnostic LED OFF. Còn nếu thông số ngõ vào IN có giá trị lớn hơn zero thì then set the diagnostic LED ON (yellow).

The CPU light emitting diode (LED) labeled SF/ DIAG can be configured to indicate yellow when either the conditions specified in the System Block are true or when the DIAG_LED instruction is executed with a non-zero IN parameter.System Block (Configure LED) check box options:

1) SF/ DIAG LED ON (yellow) when an item is forced in the CPU
2) SF/ DIAG LED ON (yellow) khi module có lỗi I/O.

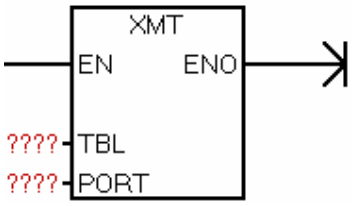
Uncheck both Configure LED options to give the DIAG_LED instruction sole control over SF/ DIAG yellow illumination. A CPU System Fault (SF) is indicated with red illumination.



15. CÁC LỆNH TRUYỀN THÔNG

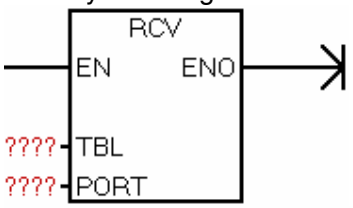
15.1. LỆNH TRUYỀN XMT

Lệnh Transmit (XMT) sử dụng trong chế độ truyền thông Freeport để truyền dữ liệu qua port truyền thông.



15.2. LỆNH NHẬN RCV

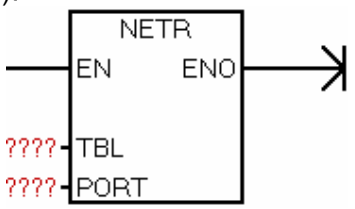
Lệnh Receive (RCV) sử dụng trong chế độ truyền thông Freeport để nhận dữ liệu qua port truyền thông.



Inputs/Outputs	Operands	Data Types
TABLE	VB, IB, QB, MB, SB, SMB, *VD, *AC, *LD	BYTE
PORT	Constant (0 for CPU 221, CPU 222, CPU 224; 0 or 1 for CPU 226)	BYTE

15.3. NETWORK READ

The Network Read (NETR) instruction initiates a communication operation to gather data from a remote device through the specified port (PORT), as defined by the table (TBL).



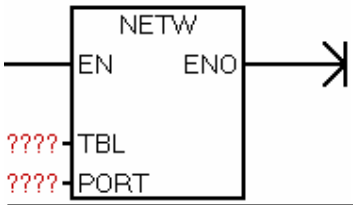
15.4. NETWORK WRITE

The Network Write (NETW) instruction initiates a communication operation to write data to a remote device through the specified port (PORT), as defined by the table (TBL).

The NETR instruction can read up to 16 bytes of information from a remote station, and the NETW instruction can write up to 16 bytes of information to a remote station.

You can have any number of NETR/NETW instructions in the program, but only a maximum of eight NETR and NETW instructions may be activated at any one time.

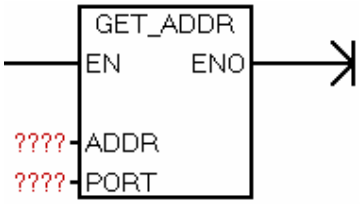
For example, you can have 4 NETRs and 4 NETWs, or 2 NETRs and 6 NETWs active at the same time in a given S7-200.



Inputs/Outputs	Operands	Data Types
TBL	VB, MB, *VD, *AC, *LD	BYTE
PORT	Constant	BYTE

15.5. GET PORT ADDRESS

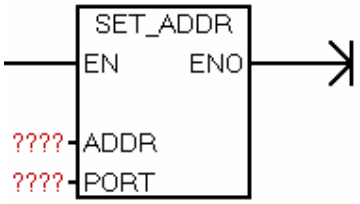
The Get Port Address (GPA) instruction reads the station address of the S7-200 CPU port specified in PORT and places the value in the address specified in ADDR.



Inputs/Outputs	Operands	Data Types
ADDR	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD	BYTE
PORT	Constant	BYTE

15.6. SET PORT ADDRESS

Lệnh Set Port Address (**SPA**) cài đặt địa chỉ port (PORT) và giá trị định nghĩa trong ADDR. The new address is not saved permanently. After a power cycle, the affected port will return to the last address (the one that was downloaded with the system block).



Inputs/Outputs	Operands	Data Types
ADDR	VB, IB, QB, MB, SB, SMB, LB, AC, Constant, *VD, *AC, *LD	BYTE
PORT	Constant	BYTE