

**CODINGO** x **posco**

**K-Digital Training** 스마트 팩토리 3기

PLC CPU

# PLC의 CPU



- 우리가 사용할 LS Electric의 CPU에는 여러가지 종류의 CPU가 있습니다.
- LS Electric의 CPU Series
  - XGT / XGB / XGS / XGR..
  - **XGT** : 중대형 컨트롤러
  - XGB : 소형 컨트롤러
  - XGS : 세이프티 컨트롤러
  - XGR : 이중화 PLC
- XGT 시리즈에 대해서 배워 보겠습니다!

# PLC의 CPU 시리즈 : XGT

- XGT-XGI

- 국제 표준 규격(IEC)을 따르는 PLC
- 국제 규격의 통신 프로토콜
- 윈도우 환경의 프로그래밍 툴 지원
- 프로그램 작성 용이

(입출력 식별자명: 실제 접속되는 기기명-한글, 한자, 영어 가능-으로 프로그래밍 가능)

XGK	XGI
P0000 	리밋_스위치 

XGI 는 한글 이름으로 사용할 수 있지만  
XGK 는 정해진 규칙을 따라서  
식별자명을 작성해야 해요!

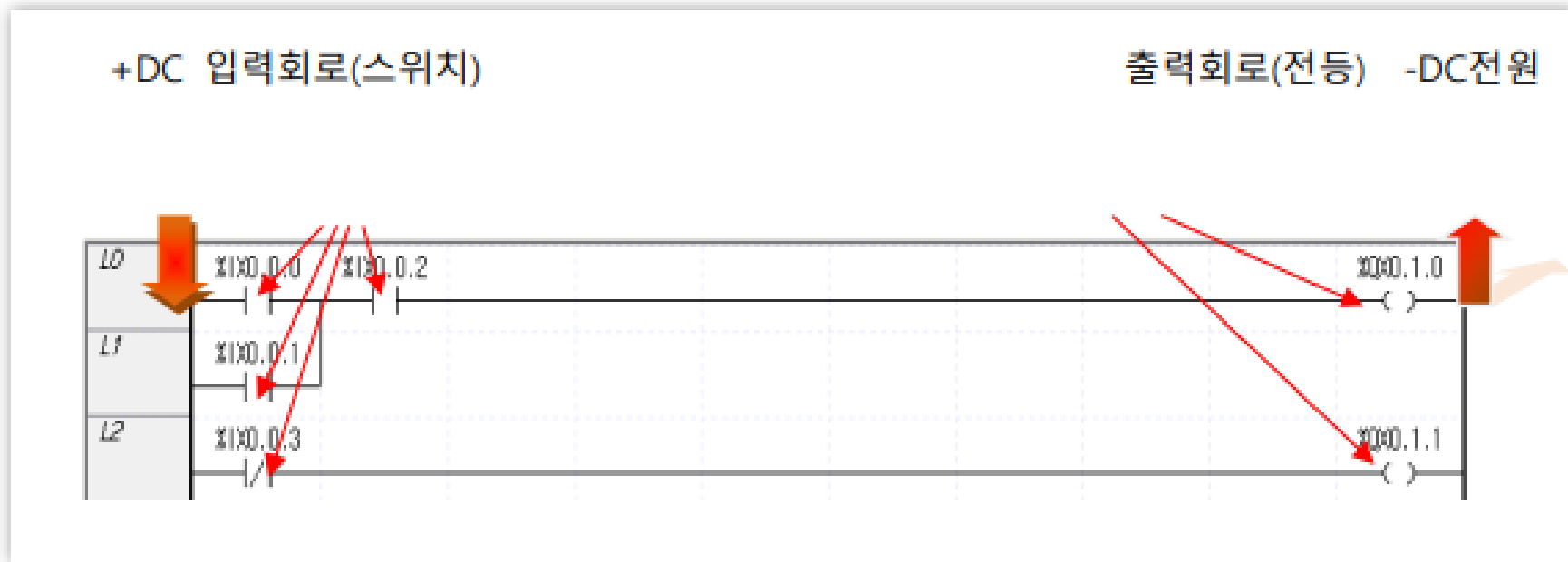
# 국제 표준 규격에 대해서 알아보까요?

- 다양한 데이터 타입 지원
- 평선, 평선블록, 프로그램과 같은 구성 요소를 이용해 프로그램을 구조적으로 작성할 수 있음
- 사용자가 작성한 프로그램을 라이브러리화하여 다른 프로젝트에서 재사용 가능
- 다양한 언어 지원, 사용자는 최적의 언어를 선택해서 사용할 수 있음
  - 도형식 언어 : LD(래더), FBD
  - 문자식 언어 : IL, ST
  - SFC

# 래더 프로그래밍

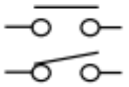

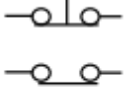

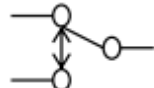
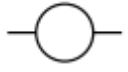
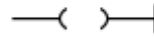

- 사다리 형태로, 릴레이 로직과 유사한 도형 기반의 언어
- 현재 가장 널리 사용되고 있음

• 예시:



# 래더 프로그래밍 사용 기호 (명령어)

- 스위치 형태의 입력과, 출력 코일
- 릴레이 로직의 기호와 흡사합니다.

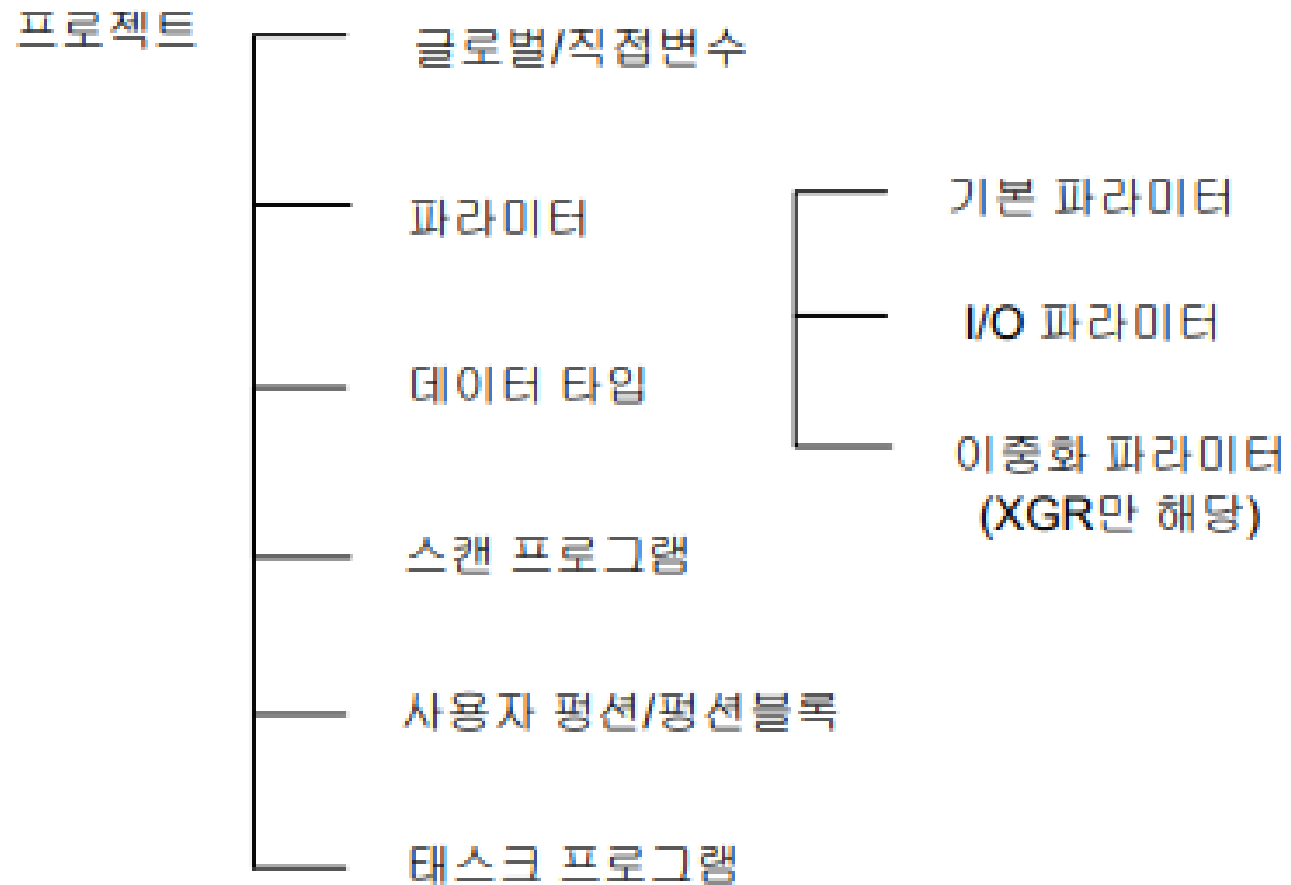
구분	릴레이 로직	PLC 로직	내용
A접점			평상시 개방(Open)되어 있는 접점 N.O. ( Normally Open ) PLC: 외부입력, 내부출력 ON/OFF상태를 입력
B접점			평상시 폐쇄(Closed)되어 있는 접점 N.C. ( Normally Closed ) PLC: 외부입력, 내부출력 ON/OFF상태의 반전된 상태를 입력
C접점		없음	a, b접점 혼합형으로 PLC에서는 로직의 조합으로 표현
출력 코일			이전까지의 연산 결과 접점 출력
응용 명령	없음		PLC응용 명령을 수행

# 래더 프로그래밍 기초 용어

- 점(Point) : 입력 8점, 출력 6점 등 PLC는 스위치나 센서 등 입출력 용량을 표시할 때 사용
- 스텝(step) : PLC 명령어의 최소 단위
- 스캔 타임(Scan Time) : 사용자가 작성한 프로그램의 1회 수행에 걸리는 시간
- WDT (Watch Dog Timer) : 이상으로 인해 출력을 하지 못할 경우 설정한 시간 대기 후 에러를 발생시키는 시스템 감지 타이머
- 파라미터(Parameter) : 프로그램과 함께 PLC 에 저장되는 운전 데이터.



# 소프트웨어 구조



# 소프트웨어 구조

- 하나의 프로젝트 안에는 PLC 시스템에 필요한 것들이 계층적으로 정의
- 프로젝트를 작성한다는 것은 PLC 시스템에 필요한 모든 구성요소를 작성한다는 의미

# 소프트웨어 구조, 프로젝트의 모든 구성 요소란?

- 글로벌/직접 변수
- 파라미터
- 데이터 타입
- 스캔 프로그램
- 사용자 평션/평션 블록
- 태스크 프로그램

데이터 타입(자료형)

기본 데이터 형 (Type)

구 분	예 약 어	데이터 형	크기 (비트)	범 위
수치 (ANY_NUM)	SINT	Short Integer	8	-128 ~ 127
	INT	Integer	16	-32768 ~ 32767
	DINT	Double Integer	32	-2147483648 ~ 2147483647
	LINT	Long Integer	64	$-2^{63} \sim 2^{63}-1$
	USINT	Unsigned Short Integer	8	0 ~ 255
	UINT	Unsigned Integer	16	0 ~ 65535
	UDINT	Unsigned Double Integer	32	0 ~ 4294967295
	ULINT	Unsigned Long Integer	64	$0 \sim 2^{64}-1$
	REAL	Real Numbers	32	-3.402823466e+038 ~ 1.175494351e-038 or 0 or 1.175494351e-038 ~ 3.402823466e+038
	LREAL	Long Reals	64	-1.7976931348623157e+308 ~ -2.2250738585072014e-308 or 0 or 2.2250738585072014e-308 ~ 1.7976931348623157e+308
시간	TIME	Duration	32	T#0S ~ T#49D17H2M47S295MS
날짜	DATE	Date	16	D#1984-01-01 ~ D#2163-6-6
	TIME_OF_ DAY	Time Of Day	32	TOD#00:00:00 ~ TOD#23:59:59.999
	DATE_AND_ TIME	Date And Time Of Day	64	DT#1984-01-01-00:00:00 ~ DT#2163-12-31-23:59:59.999
문자열	STRING	Character String	30*8	-
비트 상태 (ANY_BIT)	BOOL	Boolean	1	0, 1
	BYTE	Bit String Of Length 8	8	16#0 ~ 16#FF
	WORD	Bit String Of Length 16	16	16#0 ~ 16#FFFF
	DWORD	Bit String Of Length 32	32	16#0 ~ 16#FFFFFFFF
	LWORD	Bit String Of Length 64	64	16#0 ~ 16#FFFFFFFFFFFFFFFF

# 수치(ANY\_NUM)

- SINT/INT/DINT/LINT(8 / 16 / 32 / 64 비트)
  - (SHORT/INT/DOUBLE/LONG)
  - 각 키워드에 U가 앞에 붙으면 > Unsigned 의 의미가 됨  
(부호가 없는 양의 데이터)
    - >> SINT(Short Integer),8비트가 -128 ~ 127 까지 나타낼 수 있었다면,  
USINT(unsigned short integer)는 똑같은 크기(비트)를 가지고 있지만  
0 ~ 255까지 양으로 더 많은 범위를 나타낼 수 있다.
- REAL
  - 마찬가지로 UREAL 는 부호가 없는 실수 (32비트)

# 시간/날짜/문자열

- TIME
- DATE
- TIME\_OF\_DATE
- DATE\_AND\_TIME
- STRING

# 비트 상태 \*

- BOOL
- BYTE
- WORD
- DWORD
- LWORD



PLC 변수

# 변수 종류

- 글로벌 변수
  - 모든 프로그램에서만 사용 가능
  - 일반적인 프로그래밍에서의 전역 변수와 비슷합니다.
- 로컬 변수
  - 해당 프로그램에서만 사용 가능
  - 일반적인 프로그래밍에서의 지역 변수와 비슷합니다.

# 변수 표현 방식

- 직접 변수
  - PLC의 입출력 또는 기억 장소에 대해서 직접적으로 표현하는 것 (기본 제공)
  - 이미 지정된 식별자와 주소가 있기 때문에 변수 선언이 필요 없음
- 심볼릭 변수
  - 변수 선언 필요

# 변수 표현 방식:직접 변수

- 퍼센트 기호(%) 로 시작하고 위치 접두어, 크기 접두어와 숫자들로 구성됨

※사용 예시

종류	사용 예
입력 변수	%IX0.0.0, %IB0.0.1, %IW0.0.1, %ID0.0.0
출력 변수	%QX0.1.0, %QB0.1.1, %QW0.1.1, %QD0.1.0
내부 메모리	%MX100, %MB50, %MW100, %MD100
	%MB50.3, %MW100.10, %MD100.31

# 변수 표현 방식:직접 변수(입출력 메모리 할당)

① 위치 접두어

% I X 0 . 0 . 0  
① ② ③ ④ ⑤

접두어	의미
I	입력 위치 (Input Location)
Q	출력 위치 (Output Location)

- 위의 예시에 있는 %IX0.0.0 는 입력 위치에 대한 변수의 표현
- 만약 출력 위치에 대한 변수를 표현하고 싶다면 %QX0.0.0 이라고 표현할 수 있겠지요?

# 변수 표현 방식:직접 변수(입출력 메모리 할당)

② 크기 접두어

% l X 0 . 0 . 0  
① ② ③ ④ ⑤

접두어	의미
X(or None)	1 비트의 크기
B	1 바이트(8 비트)의 크기
W	1 워드(16 비트)의 크기
D	1 더블 워드(32 비트)의 크기
L	1 롱 워드(64 비트)

# 변수 표현 방식:직접 변수(입출력 메모리 할당)

③ & ④ 베이스 번호와 슬롯 번호

% I X N<sup>1</sup> . N<sup>2</sup> . 0  
① ② ③ ④ ⑤

- 베이스와 슬롯 번호는 0부터 시작
- 베이스 번호: [크기접두어]에 따른 N1 번째 데이터
- 슬롯 번호: N1번째 데이터 상의 N2 번째 비트

## \* 베이스와 슬롯

- PLC의 CPU의 확장을 위해서 사용하고
- 하나의 베이스에 슬롯을 추가하는 형태로 사용
- XGI 시리즈**는 단일 베이스에서 최대 12개의 슬롯을 지원하고, 확장 베이스를 이용하면 베이스도 확장 가능

# 변수 표현 방식:직접 변수(입출력 메모리 할당)

⑤ N2 슬롯에 대한 N3번째 데이터  
(크기 접두어 번호)

% I X 0 . 0 . N<sup>3</sup>  
① ② ③ ④ ⑤

%IX0.0.0 ?!

- 입력 접두어의 비트 단위의 데이터 &
- 0번베이스, 0번 슬롯의 0번째 데이터라는 의미
- 비트 단위의 데이터의 크기 접두어 X는 생략하는 것과 의미가 같으므로,  
%I0.0.0 과 동일



# 변수 표현 방식: 직접 변수 (내부 메모리 할당)

① 위치 접두어 번호

% M B N<sub>1</sub> . N<sub>2</sub>  
① ② ③ ④

접두어	의미
M	내부 메모리의 M 영역
R	내부 메모리의 R 영역
W	내부 메모리의 W 영역

- 내부 메모리에 M영역, R영역, W영역
- 내부 메모리를 할당한다면 위치 접두어로 M, R, W 가능

# 변수 표현 방식:직접 변수 (내부 메모리 할당)

② 크기 접두어 (입출력과 동일)

% M B N<sub>1</sub> . N<sub>2</sub>  
① ② ③ ④

접두어	의미
X(or None)	1 비트의 크기
B	1 바이트(8 비트)의 크기
W	1 워드(16 비트)의 크기
D	1 더블 워드(32 비트)의 크기
L	1 롱 워드(64 비트)

# 변수 표현 방식:직접 변수 (내부 메모리 할당)

$$\% \underbrace{M}_{\textcircled{1}} \underbrace{B}_{\textcircled{2}} \underbrace{N_1}_{\textcircled{3}} . \underbrace{N_2}_{\textcircled{4}}$$

③ & ④ 내부 메모리 할당은

입·출력 메모리의 할당과 기본적인 방법은 동일

- but, 베이스 번호와 슬롯 번호를 지정하지 않습니다!!

③ : N1은 크기 접두어에 대한 번호,

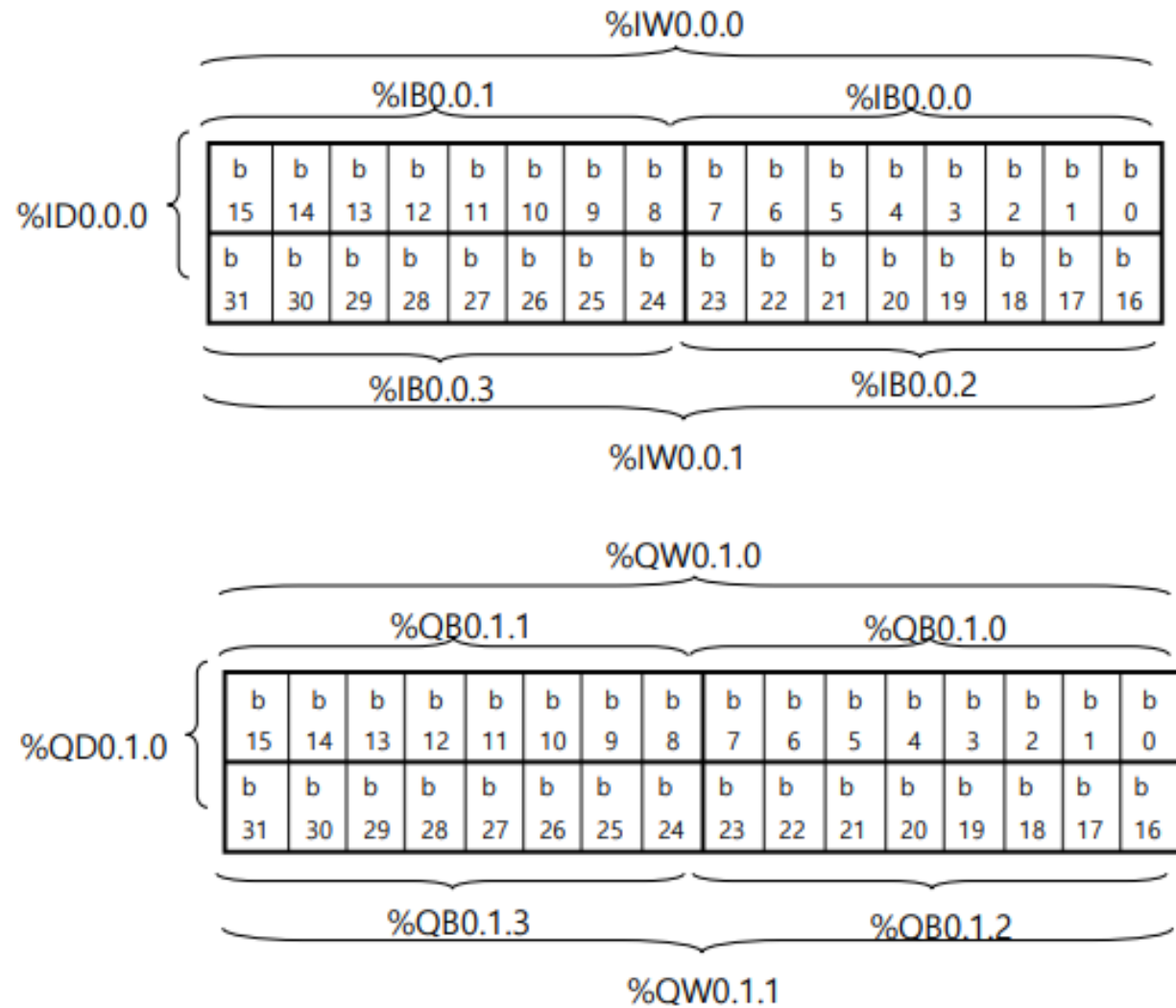
④ : 크기 접두어가 X(비트 단위)가 아닐때, 비트번호

# 변수 표현 방식:직접 변수

만약 0번 베이스 0번 슬롯에 32점 입력 모듈이 부착되어 있다면,  
⇒ 32점 입력모듈 (32 point module), 입력 접점 제한이 32 비트라는 것

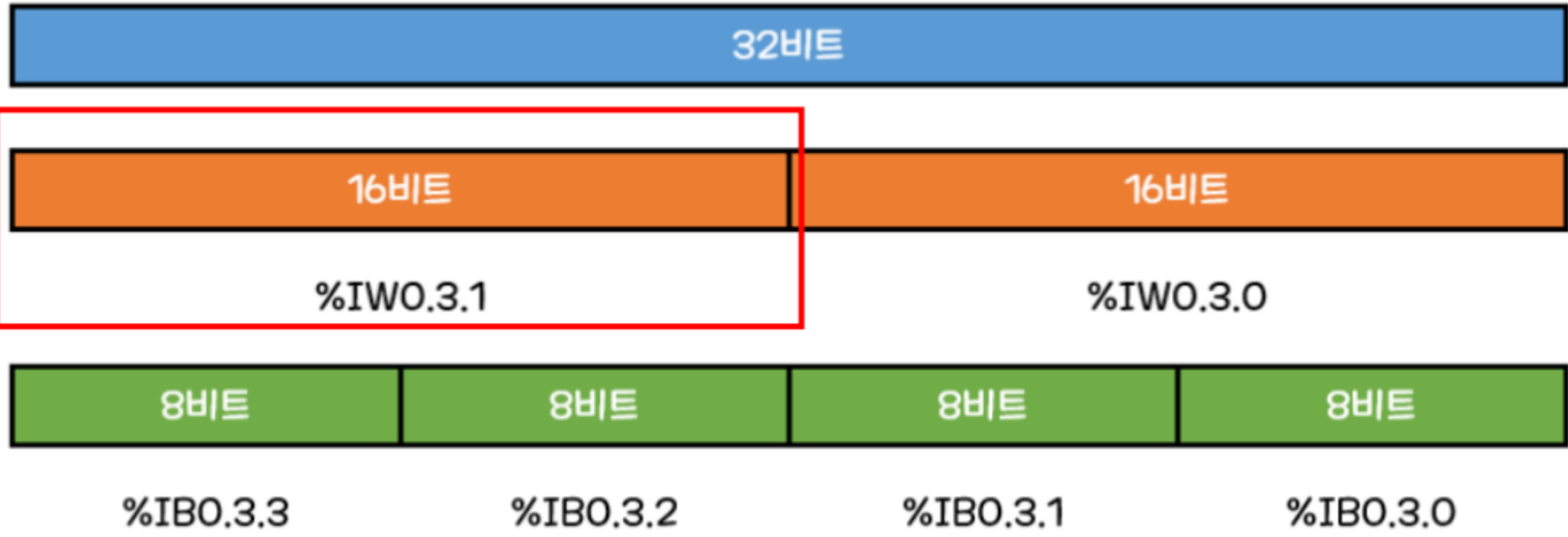
- Bit(X)로 나누어서 사용한다면 : %IX0.0.0 ~ %IX0.0.31
- Byte(B)로 나누어서 사용한다면 : %IB0.0.0 ~ %IB0.0.3
- Word(W)로 나누어서 사용한다면 : %IW0.0.0, %IW0.0.1
- Double(D)로 나누어서 사용한다면 : %ID0.0.0

# 변수 표현 방식: 직접 변수



# 퀴즈

- 0번 베이스 3번 슬롯에 32점 입력 접점이 장착되어 있을 때, %IW0.3.1 에 해당하는 메모리 주소는?
  - %IX0.3.7
  - %IX0.2.8
  - %IB0.3.5
  - %IB0.3.2



# 퀴즈

- 1번 베이스의 1번 슬롯에 32점 출력 접점이 장착되어 있을 때, %QB1.1.4 에 해당하는 메모리 주소는?
  - %QX1.1.15
  - %QB1.2.1
  - **%QW1.2.0**
  - %QD1.1.0





# 변수 표현 방식:심볼릭(Symbolic) 변수

- 사용자가 선언해서 사용하는 변수
  - 변수 이름
  - 형(data type)
  - 메모리 주소를 할당해서 사용

# 변수 표현 규칙

1. 문자나 밑줄 문자(\_)로 시작
2. 시작 문자 이후로는 문자, 숫자, 밑줄 문자('\_') 조합으로
3. 빈 칸(Space) 포함 X
4. 문자는 한자, 영문, 한글 모두 제한이 없습니다.
5. 영어일 경우 대·소문자 구별하지 않고 같은 문자면 같은 변수로 인식

# 심볼릭 변수 종류

- VAR : 읽고 쓸 수 있는 일반적인 변수
- VAR\_CONSTANT : 항상 고정된 값을 가지고 있는 읽기만 할 수 있는 변수 (상수)
- VAR\_EXTERNAL : VAR\_GLOBAL 로 선언된 변수 사용하기 위한 선언
- VAR\_EXTERNAL\_CONSTANT : VAR\_GLOBAL로 선언된 상수를 사용하기 위한 선언

# Ladder Diagram

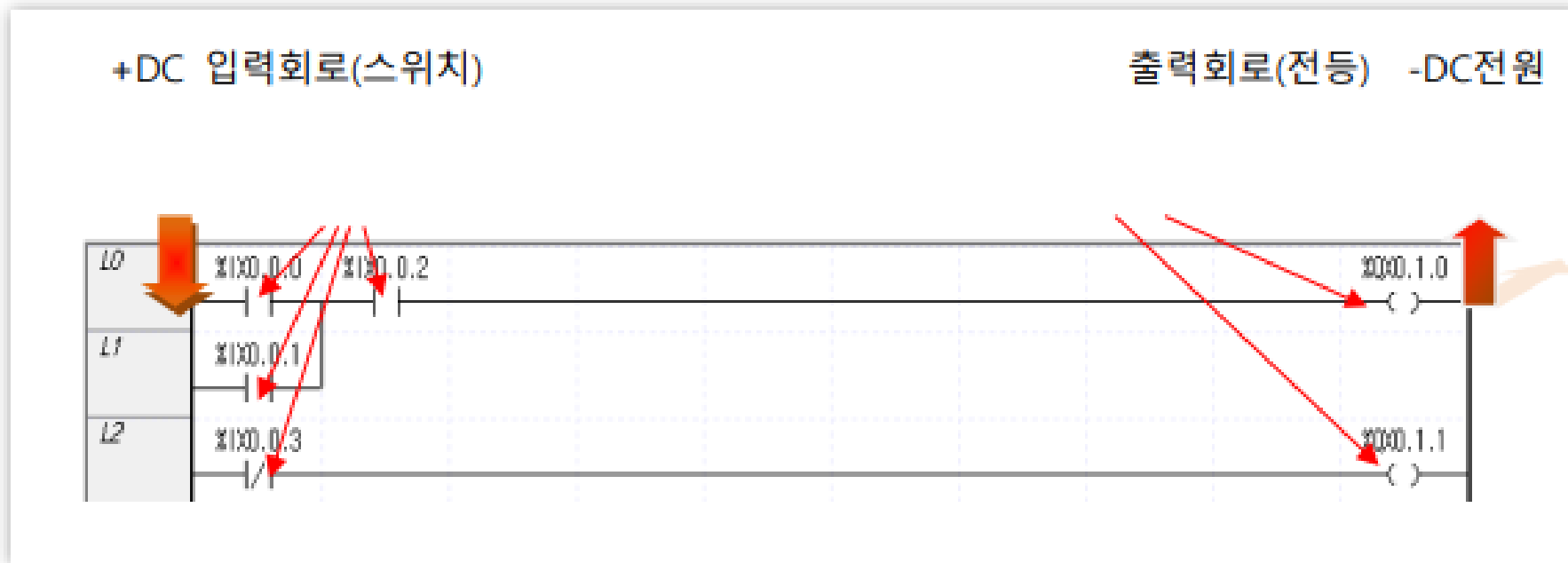
# 모선과 연결선

기호	이름	설명
	왼쪽 모선	언제나 BOOL 1의 값
	오른쪽 모선	값은 정해져 있지 않아요
	가로 연결선	왼쪽의 값 > 오른쪽으로 전달
	세로 연결선	왼쪽에 있는 가로 연결선들의 논리합




# 래더 프로그래밍

- 사다리 형태로, 릴레이 로직과 유사한 도형 기반의 언어
- 현재 가장 널리 사용되고 있음

• 예시:



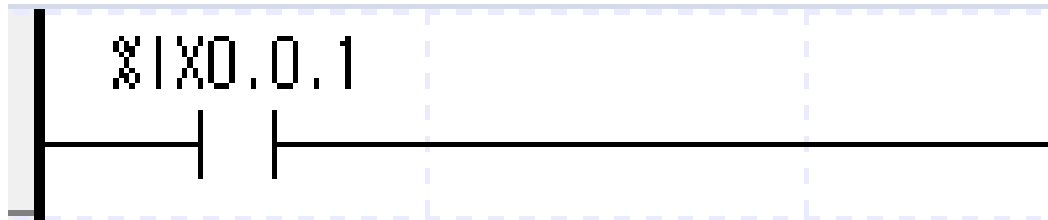
# 접점 (입력)

기호	이름	설명
	평상시 열린 접점	Toggle, on상태일 때 왼쪽의 상태가 오른쪽으로 전달 전기가 on일 때 흐른다!
	평상시 닫힌 접점	Toggle, off상태일 때 왼쪽의 상태가 오른쪽으로 전달 전기가 off일 때 흐른다!
	양 변환 <b>검출</b> 접점	평상시 열린 접점과 같이 on인 상태에서 전기가 흐르 지만, 한 스캔에 대해서만 on이 된다.
	음 변환 <b>검출</b> 접점	한 스캔에 대해서만 off가 된다.

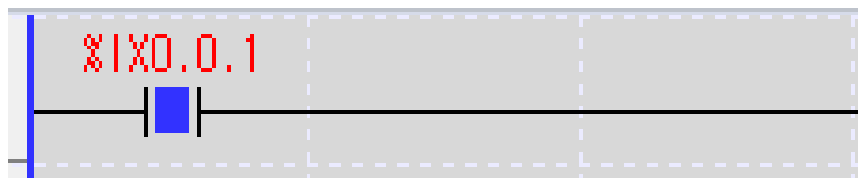
네 개의 접점 모두 스위치같은 BOOL 변수의 값이 off인지 on인지에 따라 결정됨



# 모선과 연결선, 접점의 모양



- 왼쪽 모선 : 1의 값 유지(전기가 흐른다.)
- 평상시 열린 접점
- 입력 접점에 대해서 **직접 변수** 사용! 메모리 주소 (%IX0.0.1)과 같은 변수명 사용
- on일때 오른쪽의 상태(모선의 상태 BOOL 1)가 오른쪽(연결선)으로 전달된다.

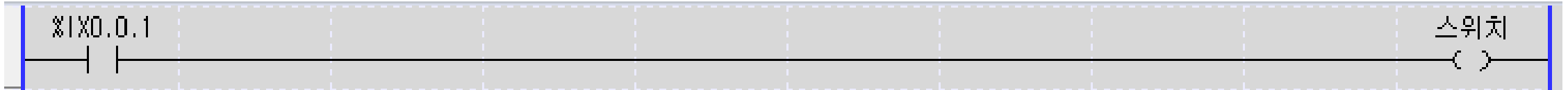


>> on일 때 전기가 흐르는 모습

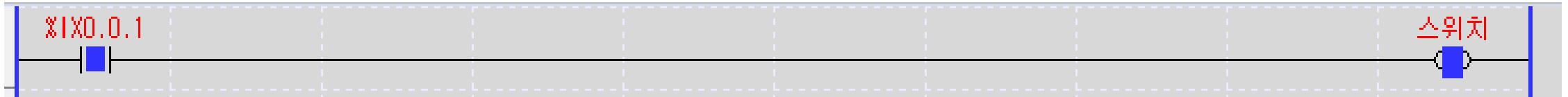
# 코일 (출력)

기호	이름	설명
—( )—	코일	왼쪽에 있는 연결선의 상태를 관련된 BOOL 변수에! 전기가 흐르는 상태라면 꺼지겠죠?
—(/)—	역 코일	왼쪽의 연결선의 역(negated)값을 관련된 BOOL 변수에.
—(S)—	Set 코일	왼쪽의 연결선 상태가 On이 되었을 때에 On. Reset 코일에 의해 Off되기 전까지는 On 되어 있는 상태로 유지
—(R)—	Reset 코일	왼쪽의 연결선 상태가 On이 되었을 때에 Off Set 코일에 의해 On되기 전까지는 Off되어 있는 상태로 유지
—(P)—	양 변환 검출 코일	코일과 동일하지만, 한 스캔 동안만 on되고 이후로는 꺼집니다.
—(N)—	음 변환 검출 코일	역 코일과 동일하지만 한 스캔 동안만 on이 되고 이후로는 off

# 코일



- 기본적으로, 평상시 열린 접점 (`%IX0.0.1`) BOOL값 0
- 왼쪽 모선의 전기는 흐르고 있지만(BOOL 1) 입력접점 이후의 오른쪽 연결선으로 전달이 되지 않는다.
- 스위치: 심볼릭 변수



- 입력접점이 켜지면 왼쪽 출력 코일까지 1 값이 전달!

프로그램 설치

# 프로그램 설치: XG5000

- LS electric의 PLC 프로그래밍 툴
- 무료!
- 미쯔비시의 GX works / 지멘스의 Simatic.. 등도 현업에서 많이 쓰임.
- 사용법은 조금씩 다르지만 모두 비슷하기 때문에 하나만 잘 익혀두면 다른 프로그램 사용도 어렵지 않게 익힐 수 있을 거예요!

# XG5000 설치

<https://sol.ls-electric.com/kr/ko/product/category/0>

# XG5000 설치

LS ELECTRIC Solution Square

커뮤니티 제품 산업 서비스

검색..

대한민국

PLC XGT XGB XGS SmartIO PLC 소프트웨어 MASTER-K GLOFA-GM

상세 정보 보기

다운로드 기술문서 소프트웨어 교육 커뮤니티 샘플 라이브러리

전체 3 사용 설명서 2 소프트웨어 1

XG5000

ENT

No.	자료명	북마크	공유	언어	업데이트	조회수
1	XG5000 소프트웨어			한/영	23-04-13	8300

버전 4.73

ReleaseNote\_En\_V4.73\_20230412.pdf 8.8 MB

ReleaseNote\_Kr\_V4.73\_20230412.pdf 9.6 MB

XG5000\_V4.73\_20230412\_En.exe 438.9 MB

XG5000\_V4.73\_20230412\_Kr.exe 384.7 MB

연관 카테고리

PLC / PLC 소프트웨어 / XG5000

이 자료가 도움이 되었나요?

😊 도움돼요 😞 부족해요

피드백을 남겨주세요.

\* 링크에 들어가서 XG5000 검색 후 한국어 버전 .exe 파일 다운로드