K-Digital Training 스마트 팩토리 단기 8기

codingOn

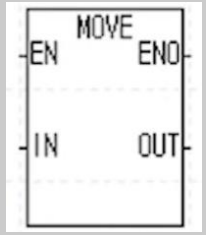
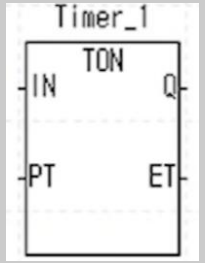
4. 펑션/펑션블록

펑션 vs 펑션블록

- 펑션(Function)
  - 특정 작업을 수행하는 독립적인 프로그래밍 단위.
  - 입력값을 받아 연산을 수행하고 단일 결과값을 반환.

- 펑션 블록(Function Block)
  - 복잡한 제어 로직을 수행하기 위한 프로그래밍 단위.
  - 내부 상태를 유지할 수 있고, 다양한 입출력 포트를 가질 수 있음.

펑션 vs 펑션블록

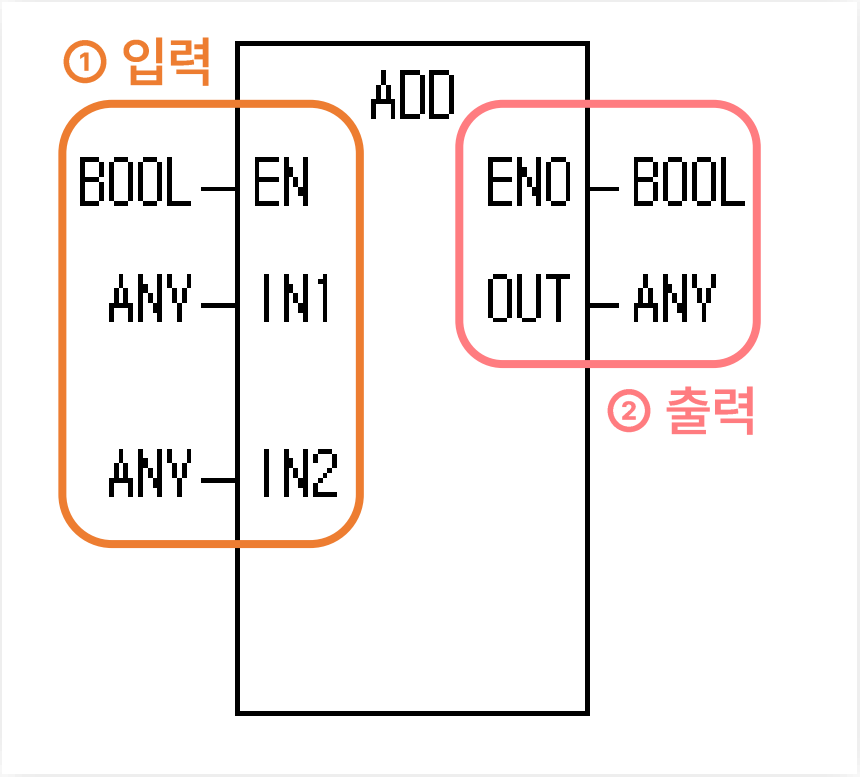
구분	펑션	펑션 블록
입력의 수	1개 이상	2개 이상
출력의 수	오직 1개	1개 이상
연산 시간	1스캔에 결과 출력	여러 스캔 누계 결과 출력
데이터	입·출력 데이터를 모두 반드시 지정	입력 데이터는 반드시 지정, 출력 데이터는 생략 가능
데이터 타입	입력 변수와 출력 변수의 모든 데이터 타입이 동일	변수의 기능에 따라 다양한 데이터 타입
예	전송, 형 변환, 비교, 산술 연산 펑션	타이머, 카운터, .. 등
		

펑션

1) 입력

- EN
  - 연결선(/모선)과 연결되는 곳
  - EN의 Bool값이 1때 함수가 동작함

- IN1/IN2
  - 펑션에 입력할 입력 변수
  - 펑션마다 입력할 변수의 개수와 종류가 다름
  - 입력값이 하나 이상이라면 두 입력 변수의 data type을 동일하게 맞춰줘야 함

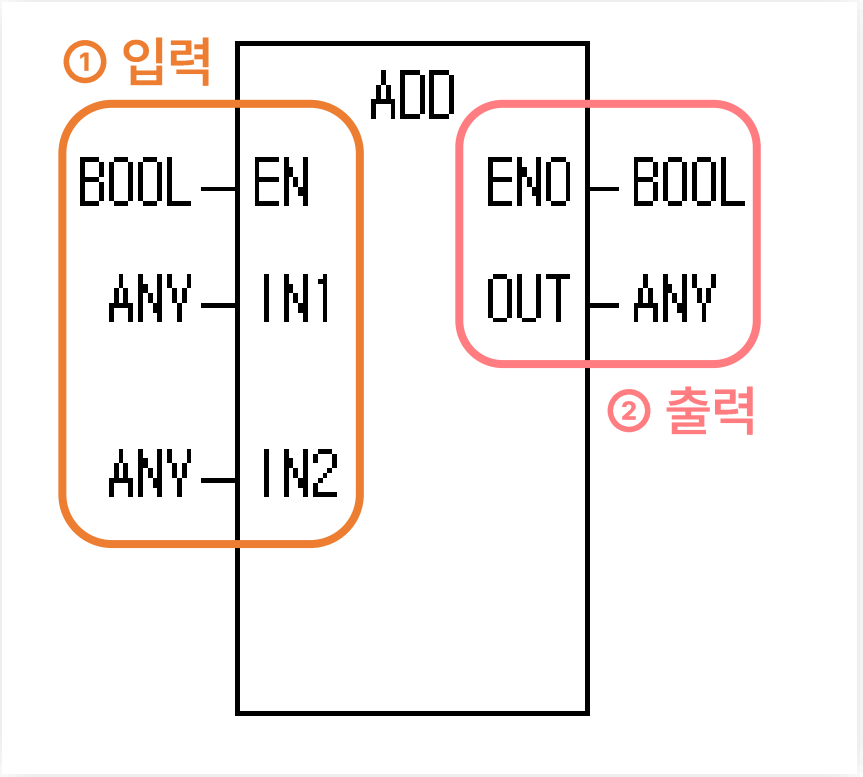


펑션

2) 출력

- ENO
  - EN에 따른 출력, 함수가 정상작동하면 1이 출력됨

- OUT
  - 입력 변수 IN1과 IN2가 함수의 연산에 의해 출력된 값
    (입력: 1, 2 → 출력: 3)
  - 입력 변수의 데이터 타입과 일치시켜야 함.
    (예외적인 펑션도 있음)
  - 출력부에는 항상 변수를 할당해줘야 함.

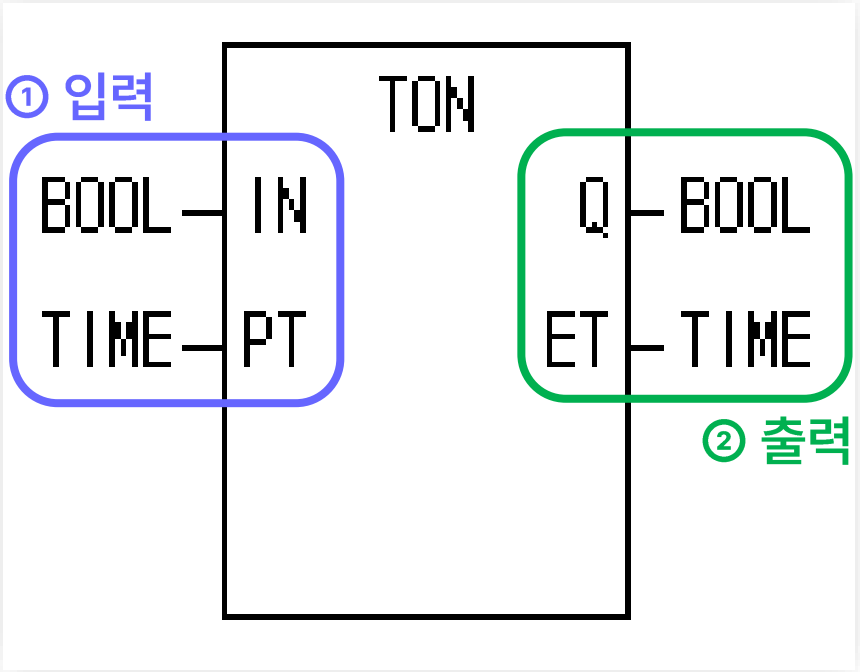


펑션 블록

1) 입력

- IN
  - 연결선(/모선)과 연결되는 곳
  - EN의 Bool값이 1때 함수가 동작함

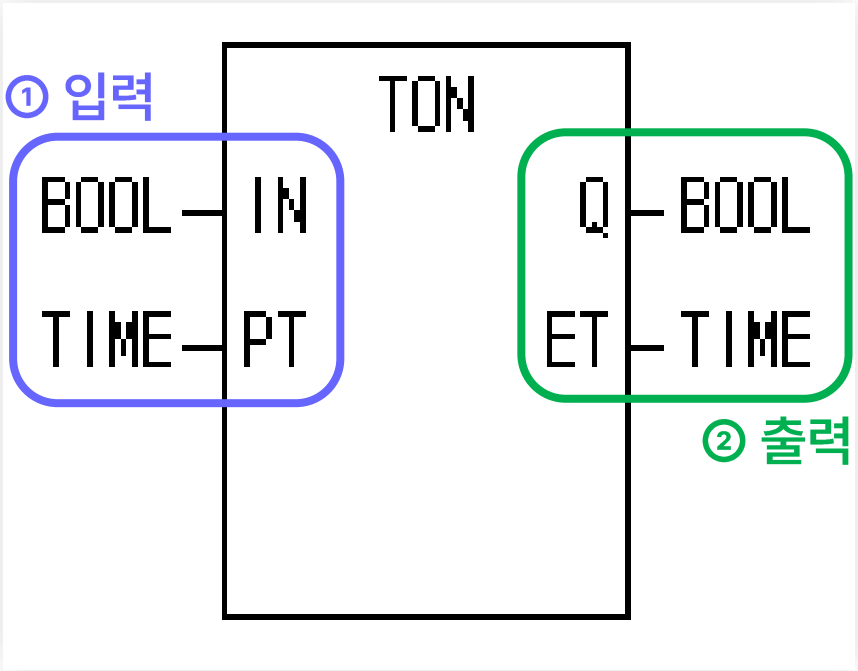
- PT
  - 펑션에 입력할 입력 변수
  - 펑션마다 입력할 변수의 개수와 종류가 다릅니다.
  - 입력값이 하나 이상이라면 두 입력 변수의 data type을 동일하게 맞춰줘야 합니다.



펑션 블록

2) 출력

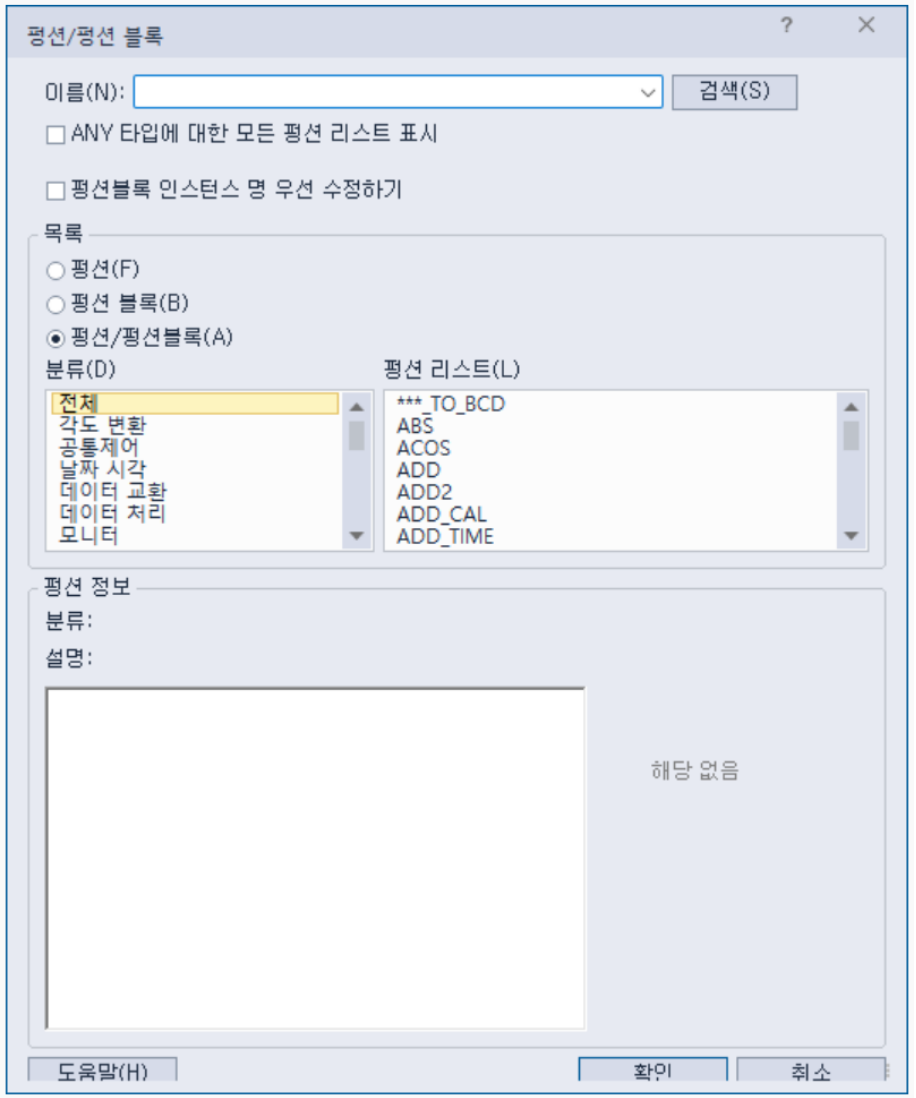
- Q
  - 펑션의 ENO 과 동일
- ET
  - 출력 데이터의 변수는 지정해주지 않아도 됨



펑션블록 내의 입 · 출력 이름으로
항상 IN, PT, Q, ET 라는 이름을
가지고 있지는 않아요!

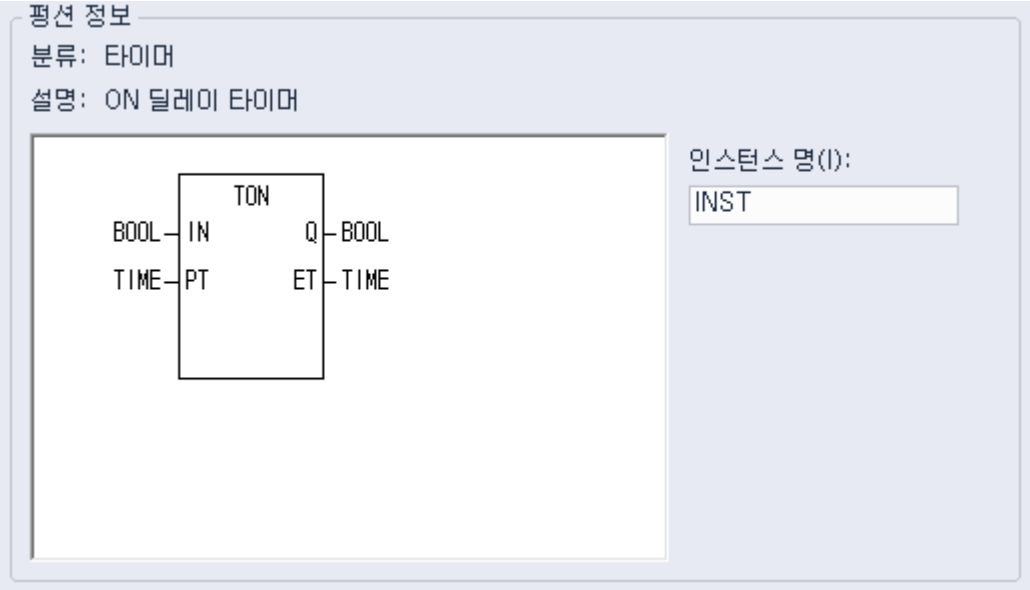
펑션/펑션 블록 추가


sF5
sF6
F10
sF7
c3
c4
c5
c6

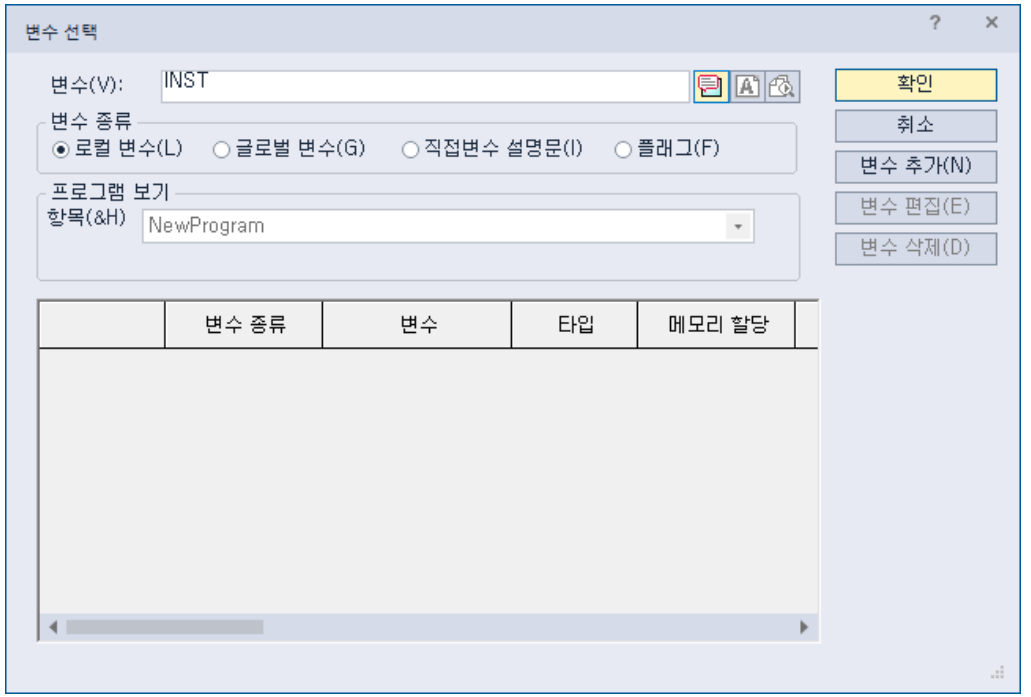

펑션/펑션 블록
이름(N):
검색(S)
ANY 타입에 대한 모든 펑션 리스트 표시
펑션블록 인스턴스 명 우선 수정하기
목록
펑션(F)
펑션 블록(B)
펑션/펑션블록(A)
분류(D)
펑션 리스트(L)
전체
각도 변환
공통제어
날짜 시각
데이터 교환
데이터 처리
모니터
***_TO_BCD
ABS
ACOS
ADD
ADD2
ADD_CAL
ADD_TIME
펑션 정보
분류:
설명:
해당 없음
도움말(H)
확인
취소

펑션/펑션 블록 추가

- 인스턴스 지정



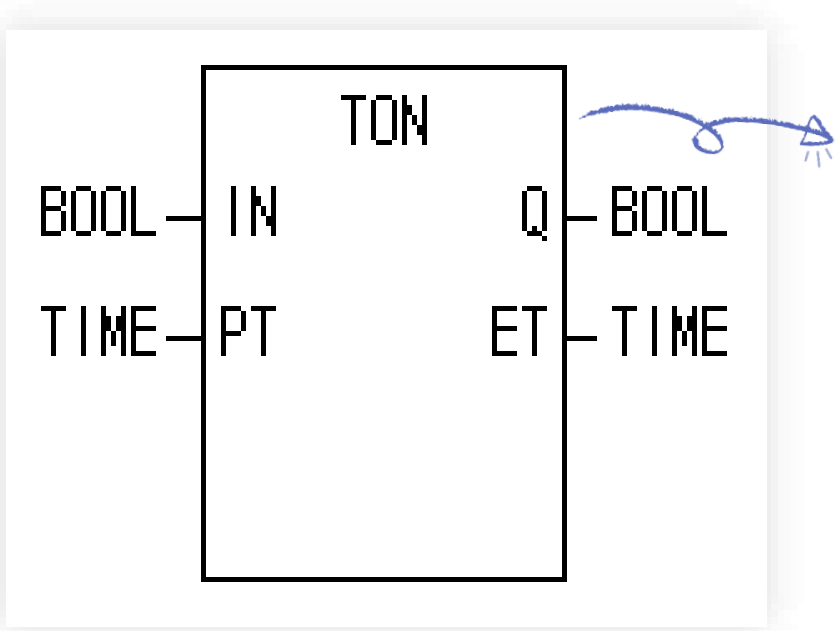
※ 펑션 블록/펑션 설정 화면



※ 펑션 블록 인스턴스 설정 화면

펑션/펑션 블록 추가

- 인스턴스 지정
  - 인스턴스 : 펑션 블록에 붙일 이름
  - 인스턴스를 지정해서 펑션 블록 내부의 데이터를 사용할 수 있음



인스턴스: T1 ➡

T1.Q
T1.IN
T1.ET
T1.PT 에 값이 할당됨

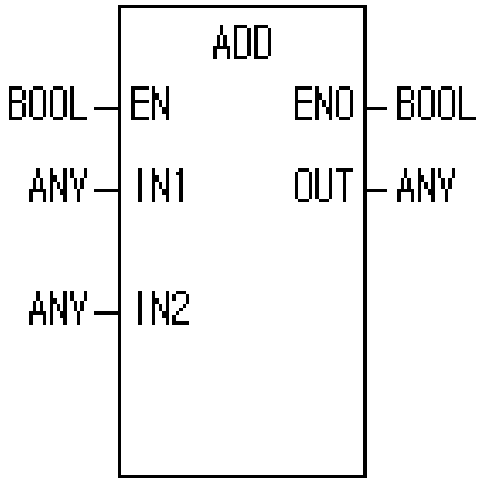
펑션블록의 경우 펑션과 다르게 출력 변수를 할당하지 않아도 TON 펑션 블록의 결과가 T1.ET에 반환됨

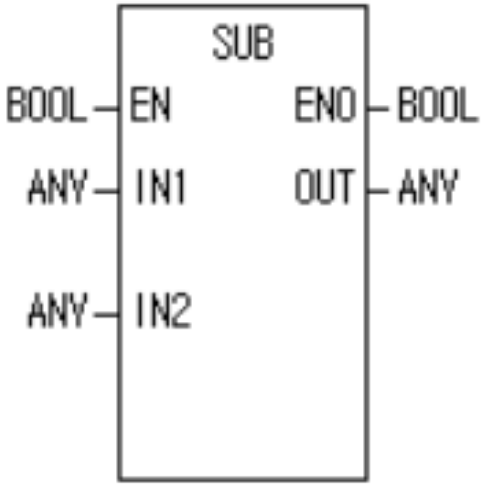
codingOn

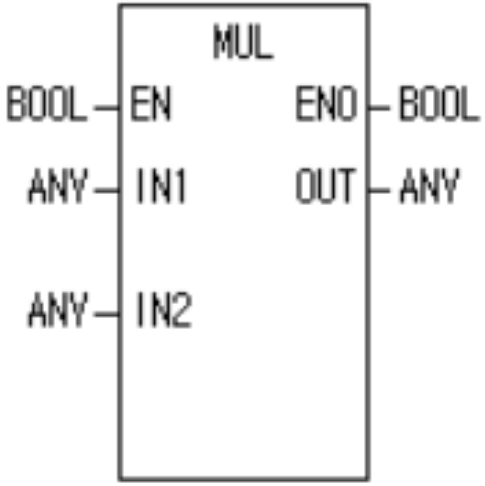
4-1. 펑션 (Function)

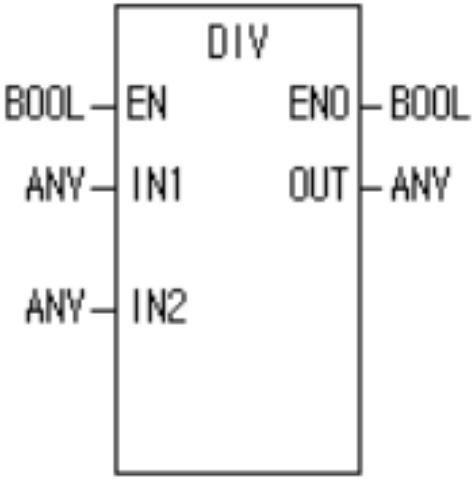
산술연산

- **산술연산**
  - 각각 ADD, SUB, MUL, DIV
  - 나머지의 경우 MOD로 구함
  - 입력의 데이터 타입과 출력의 데이터 타입이 같아야 함





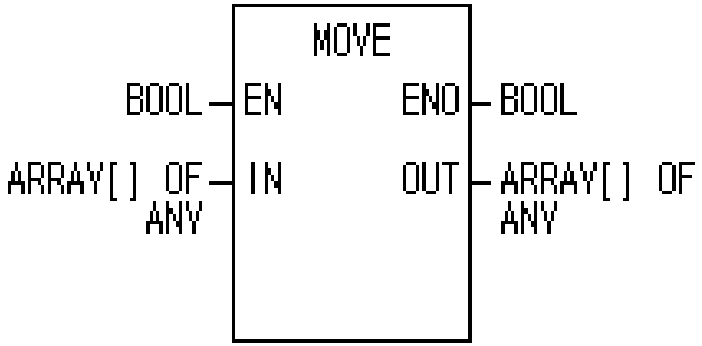




MOVE 펑션

- 전송 펑션

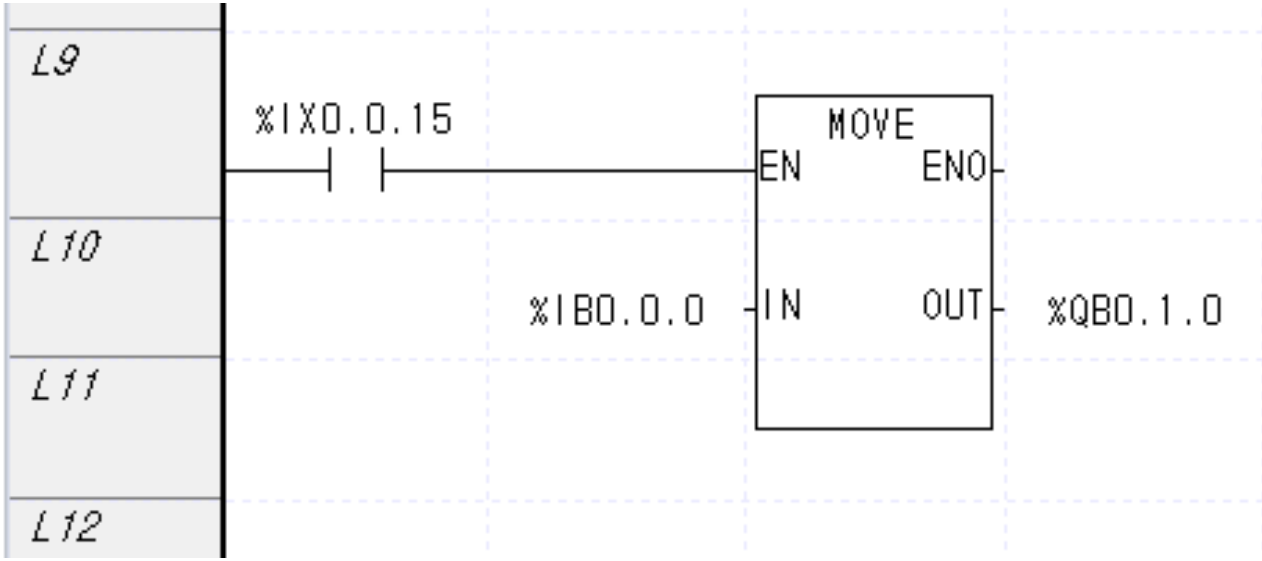
펑션 이름	기능
MOVE	데이터 전송 (IN -> OUT)
ARY_MOVE	배열 변수 부분 전송

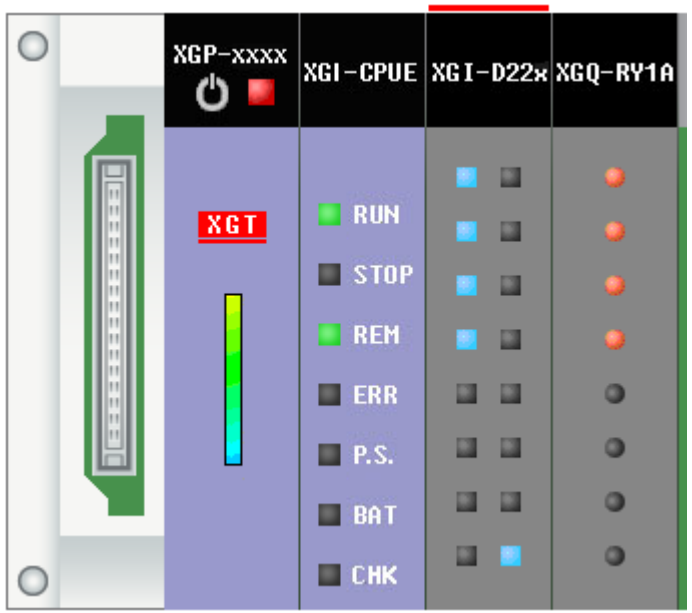


- EN이 ON 되면
- 함수가 정상 작동하면 ENO = 1
- IN에 있는 데이터를 OUT으로 복사

MOVE 펑션

- %IX0.0.15가 ON이 되면 %IB0.0.0(%IX0.0.0 ~ %IX0.0.7)의 값이 %QB0.1.0(%QX0.1.0 ~ %QX0.1.7)로 복사되도록 프로그램 작성





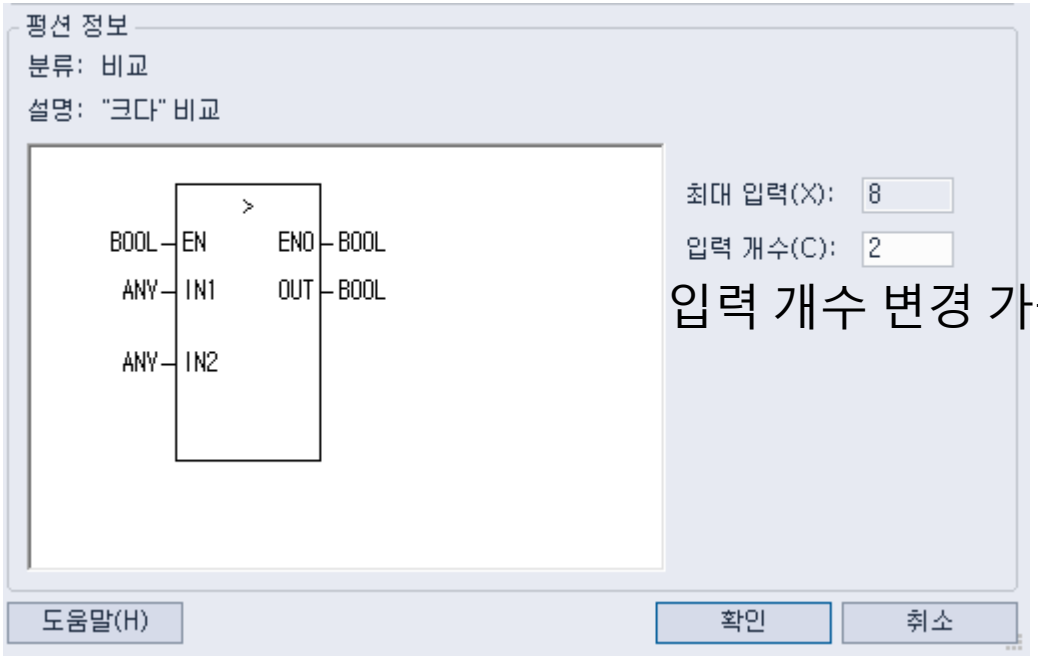
%IX0.0.15가 켜지면 첫 8비트가 복사됨

비교 펑션

펑션 이름	부호	기능
GT	>	(IN1 > IN2) and (IN2 > IN3) and ... (IN7 > IN8) -> OUT
GE	≥	(IN1 ≥ IN2) and (IN2 ≥ IN3) and ... (IN7 ≥ IN8) -> OUT
EQ	=	(IN1 = IN2) and (IN2 = IN3) and ... (IN7 = IN8) -> OUT
LE	≤	(IN1 ≤ IN2) and (IN2 ≤ IN3) and ... (IN7 ≤ IN8) -> OUT
LT	<	(IN1 < IN2) and (IN2 < IN3) and ... (IN7 < IN8) -> OUT
NE	≠	(IN1 ≠ IN2) and (IN2 ≠ IN3) and ... (IN7 ≠ IN8) -> OUT

GT 펑션

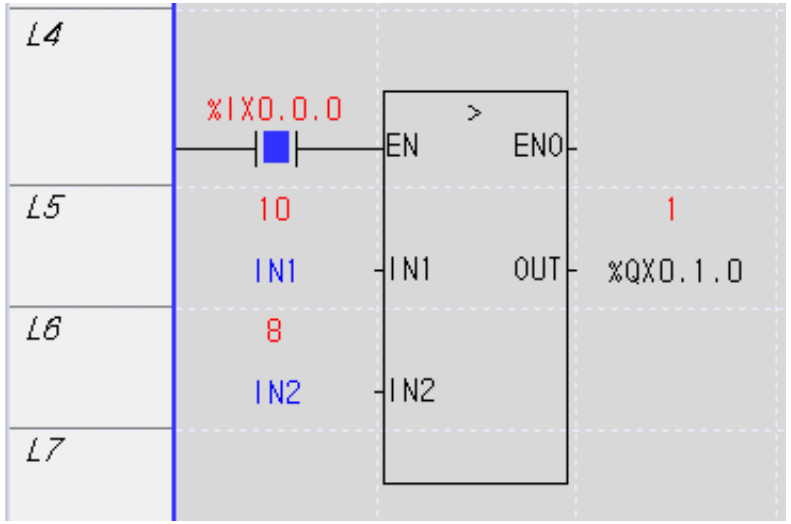
- 펑션 및 변수 추가하기

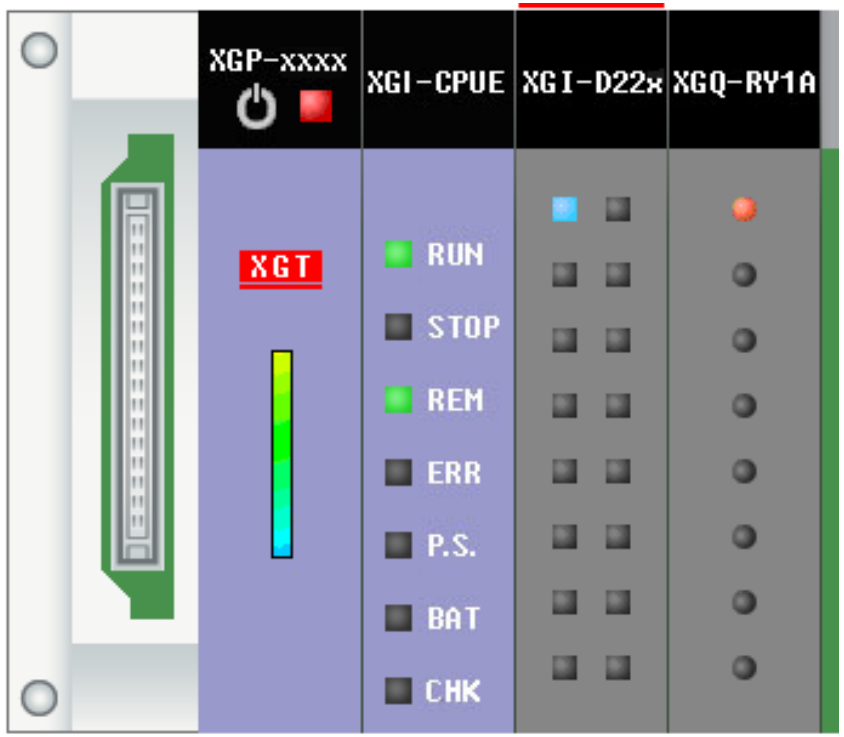


입력 개수 변경 가능

GT 펑션

- %IX0.0.0을 ON 하면 10 > 8 = True 이므로 %QX0.1.0이 ON
- 입력의 데이터 타입이 같아야 함



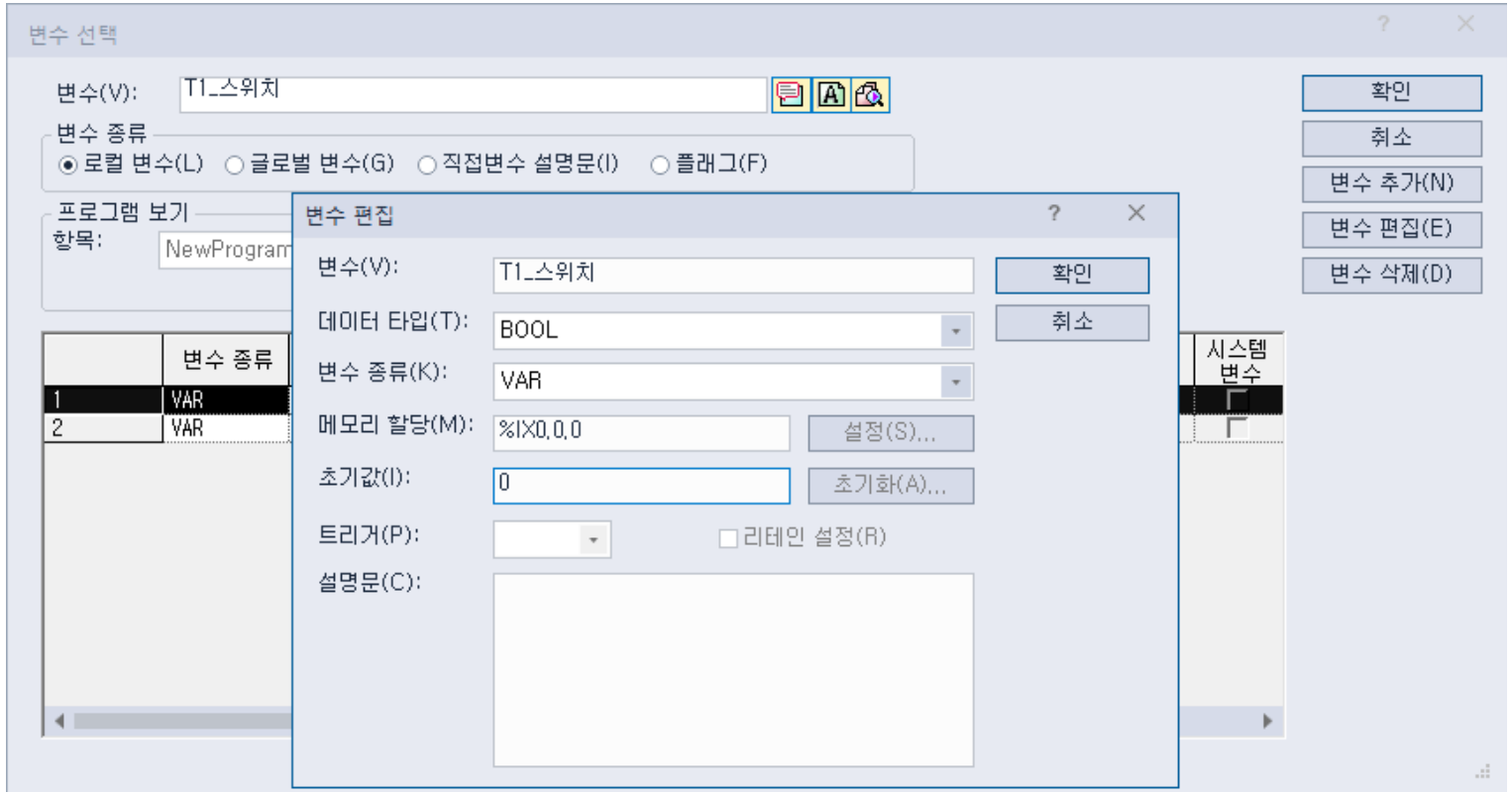


codingOn

4-2. 펑션 블록 (Function Block)

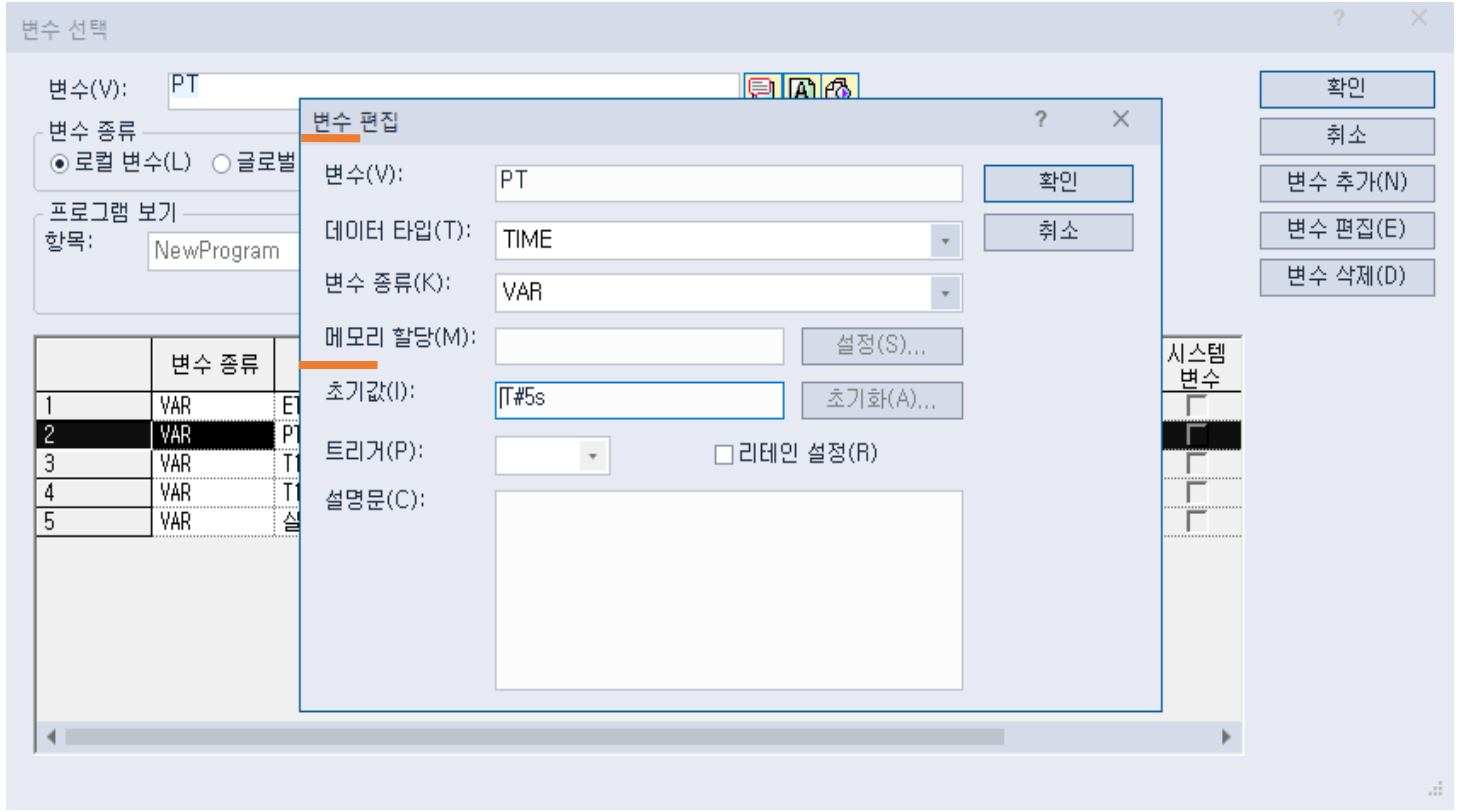
심볼릭 변수 응용

- 심볼릭 변수는 이름을 지정 가능
- 심볼릭 변수가 접근하게 될 메모리를 지정 가능



로컬 변수 응용 – Time

- TON의 PT 입력과 같은 값은 Time 변수를 사용해야 함
- Time의 값은 T#5s121ms 같이 작성(5.121초)

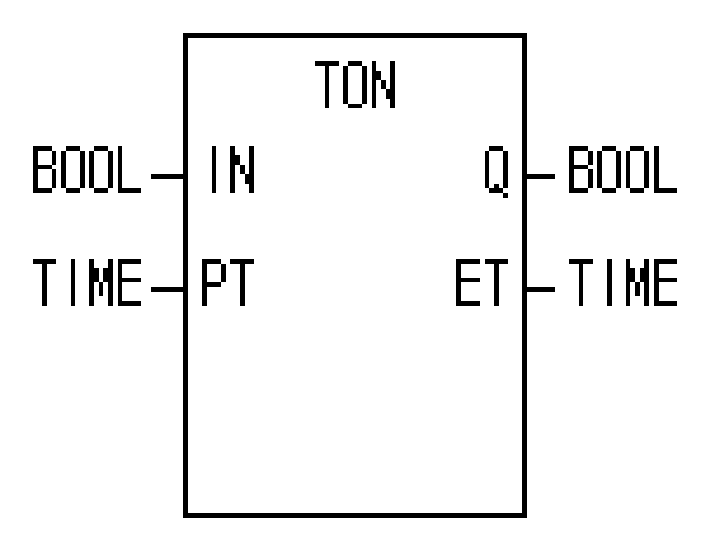


codingOn

TON, TOF

TON (On delay Timer)

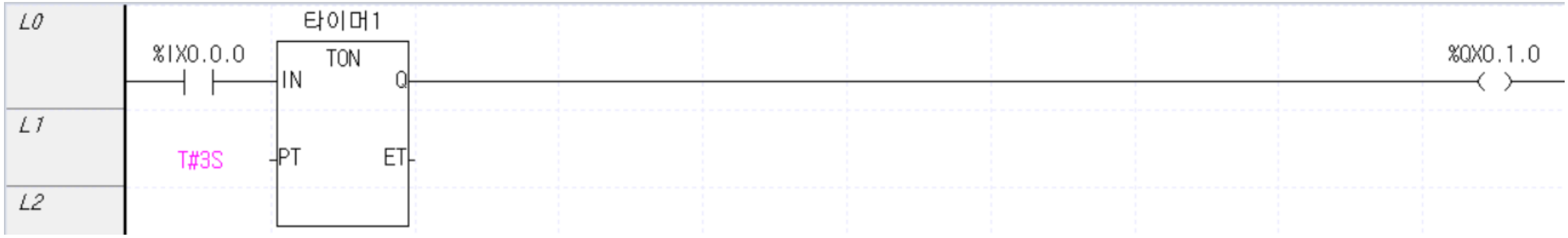
- 1 값을 전달할 때만 타이머 기능을 함

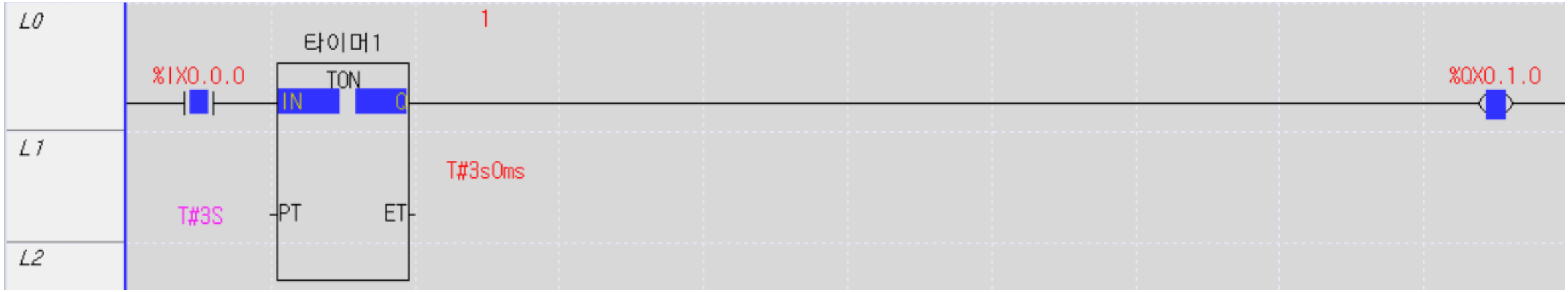


- **입력**
  - IN: 타이머 기동 조건
  - PT: 설정 시간
- **출력**
  - Q : 타이머 접점 출력
  - ET: 경과 시간(Elapsed Time)

- IN이 1이 된 후, 경과 시간이 ET로 출력
- (IN 0 → 1)PT에 설정한 시간이 모두 흐르면 IN으로 전달된 값이 Q로 전달
- (IN 1 → 0) 설정한 시간과 관계없이 IN이 0이 되면 Q도 0이 됨

TON 기본 구현


L0
L1
L2
타이머1
%IX0.0.0
TON
IN
Q
T#3S
PT
ET
%QX0.1.0


L0
L1
L2
1
타이머1
%IX0.0.0
TON
IN
Q
T#3s0ms
T#3S
PT
ET
%QX0.1.0

TON과 EQ, Set코일 사용해보기

codingOn

- EQ 펑션

펑션 이름	부호	기능
EQ	=	(IN1 = IN2) and (IN2 = IN3) and ... (IN7 = IN8) -> OUT

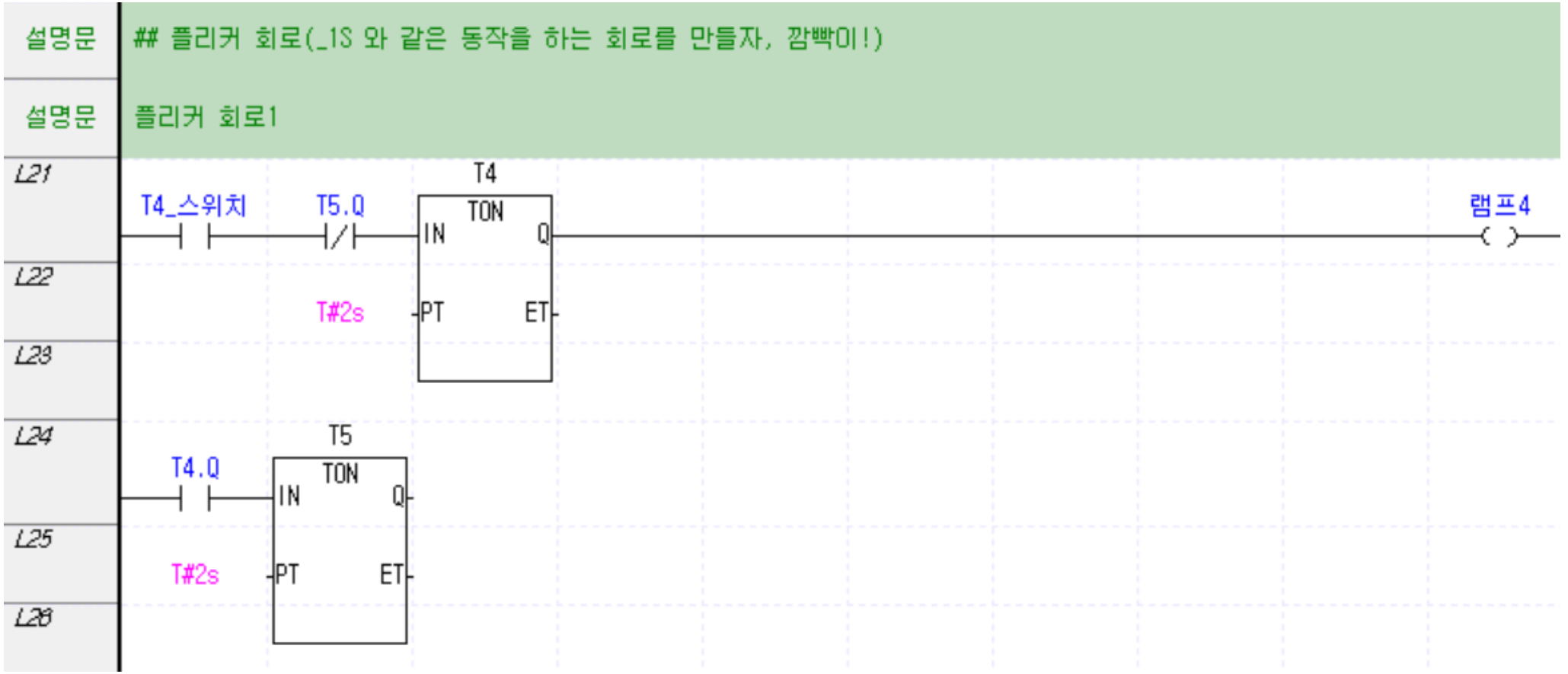
- Set 코일

기호	이름	설명
—(S)—	Set 코일	입력으로 ON 상태가 들어오면 ON되고, 상태가 유지 됨. Reset 코일을 통해 OFF 상태로 변경 가능

TON 플리커 회로

- 램프를 2초 간격으로 ON/OFF 반복시키는 회로 구현
- 회로 동작
  - 스위치를 ON하면 타이머1이 동작한다.
  - 2초 경과 후 램프가 켜짐
  - 동시에 타이머2가 동작
  - 타이머2 동작 2초 경과 후 T4가 리셋 되며 램프가 꺼짐
  - 다음 스캔에서 반복 동작(깜빡임)

TON 플리커 회로


설명문
플리커 회로(_1S 와 같은 동작을 하는 회로를 만들자, 깜빡이!)
설명문
플리커 회로1
L21
L22
L23
L24
L25
L26
T4
TON
T4_스위치
T5.Q
IN
Q
T#2s
PT
ET
램프4
T5
TON
T4.Q
IN
Q
T#2s
PT
ET

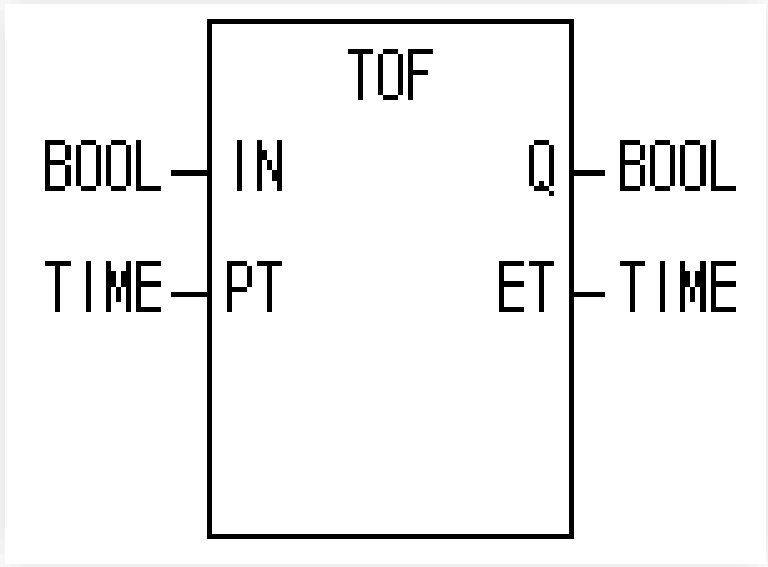
실습1 램프 순서대로 켜기

codingOn

- 앞에서 학습한 TON, 플리커를 활용해서
- 타이머 4개가 3초마다 작동하고, 순차적으로 램프 4개가 켜지도록 만들어보세요.
  - 시작 → 타이머1(3초) → 램프1, 타이머2(3초) → 램프2, 타이머3(3초) → 램프3, 타이머4(3초) → 램프4
- 램프가 모두 켜지고 나면 처음부터 다시 타이머가 작동합니다.

TOF(OFF delay Timer)

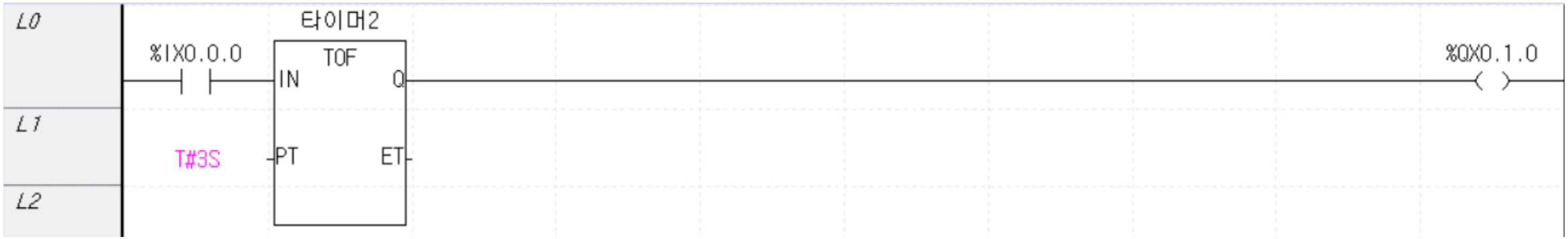
- 0 값을 전달할 때만 타이머 기능을 함

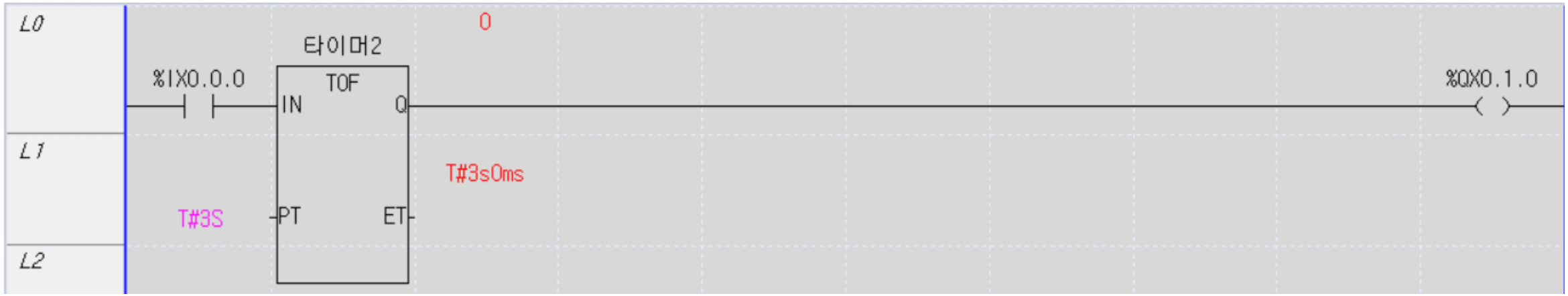


- **입력 ·출력**
  - 모두 TON과 같음

- IN이 0이 된 후, 경과 시간이 ET로 출력
- (IN 1 → 0)PT에 설정한 시간이 모두 흐르면 IN으로 전달된 값이 Q로 전달
- (IN 0 → 1) 설정한 시간과 관계없이 IN이 1이 되면 Q도 1이 됨

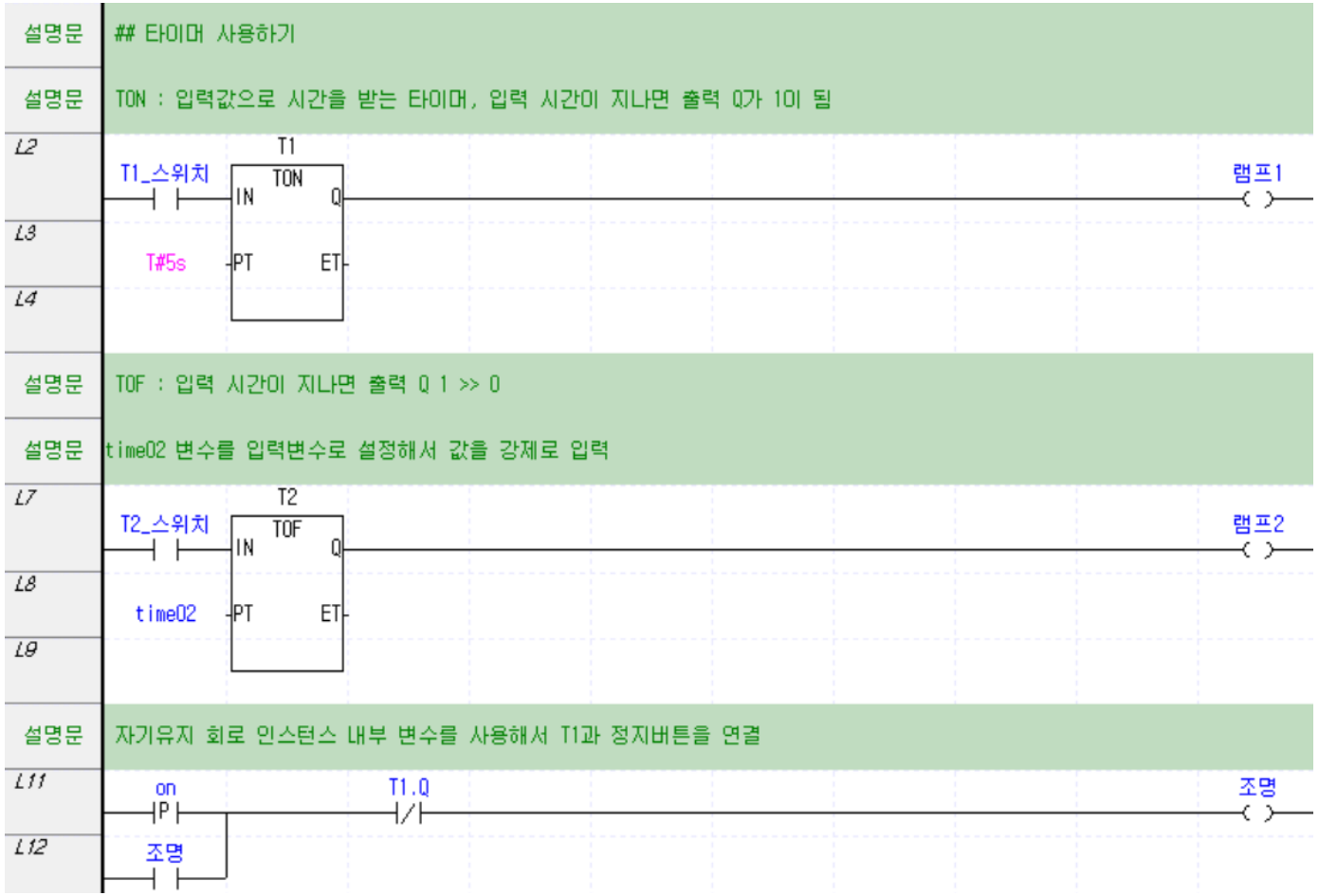
TOF 기본 구현


L0
타이머2
%IX0.0.0
TOF
IN
Q
%QX0.1.0
L1
T#3S
PT
ET
L2


L0
0
타이머2
%IX0.0.0
TOF
IN
Q
%QX0.1.0
L1
T#3s0ms
T#3S
PT
ET
L2

연습3. TON, TOF 사용

- TON, TOF를 사용하여 아래와 같이 작성해보기



실습2 TON, TOF

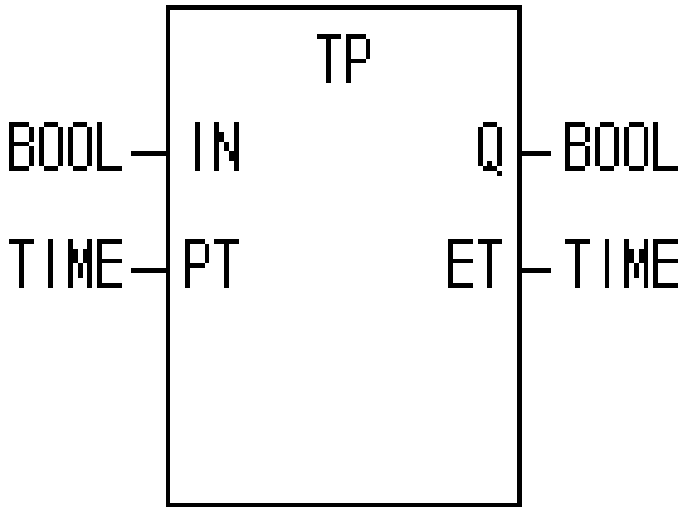
codingOn

- 하나의 스위치와 램프가 있으며, 스위치의 입력에 따라 설정한 시간 후 램프가 켜지고, 다시 설정한 시간 후 램프가 꺼지는 회로를 설계하시오.
- 동작 조건
  - 스위치를 ON하면 설정된 시간(예: 5초) 후 램프가 켜진다.
  - 스위치를 OFF하면 설정된 시간(예: 5초) 후 램프가 꺼진다.
  - 스위치와 램프는 각각 하나씩만 사용한다.

TP(Pulse Timer)

1 값을 전달할 때,
설정한 시간만큼만 1전달

- 입출력 요소는 TON과 같음
- PT로 설정한 시간 만큼만 IN의 값을 Q로 전달
- PT에서 설정한 시간이 지나면 IN이 ON 이더라도 Q는 OFF가 됨



TP 기본 구현

```
## TP : 입력시간 동안만 Q가 1인 타이머

T3 인스턴스의 변수 Q를 사용해서 램프3 ON
```

실습3. 자동 물내림

- TON과 TP를 활용하여
- 사용자가 변기에 앉으면 1초 후, 2초간 물이 나오고
- 이탈 후 즉시 3초간 물이 공급
- 입력 : 사용자를 감지하는 센서
- 출력 : 세척 밸브

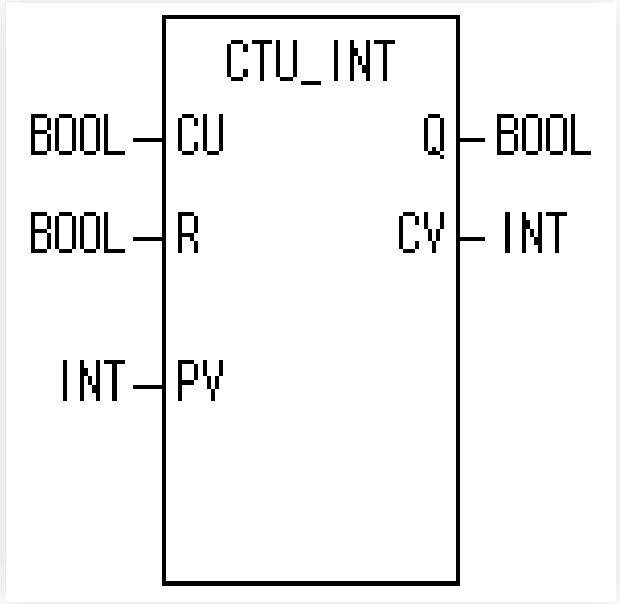
codingOn

카운터

카운터

- CTU_***/CTD_***/CTUD_***/CTR/..
  - *** : INT, DINT,LINT, UINT ,UDINT, ULINT
- 횟수을 측정하는데 사용됨
- * 카운터 최댓값: 자료형 별로 상이

CTU_*** (count up)

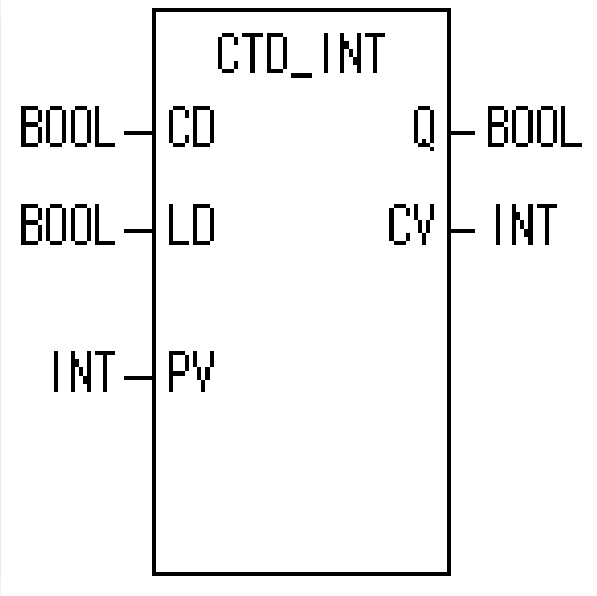


- **입력**
  - CU: 카운터 기동 조건
  - R: 리셋 조건
  - PV(Preset Value) : 설정한 횟수 이상(이하)일 때, Q에 1 전달
- **출력**
  - Q : 카운터 접점 출력
  - CV(Current Value): 카운트 횟수

입력값이 1일 때,
입력조건의 정수값 이상으로 카운트 되면 출력으로 1전달

- CU 0→1이 되는 횟수를 CV로 출력
- PV에 설정한 수 이상일 때, Q로 1값 전달
- R : CV값 리셋, 따라서 Q값도 1 → 0으로 변경됨

CTD_*** (count down)

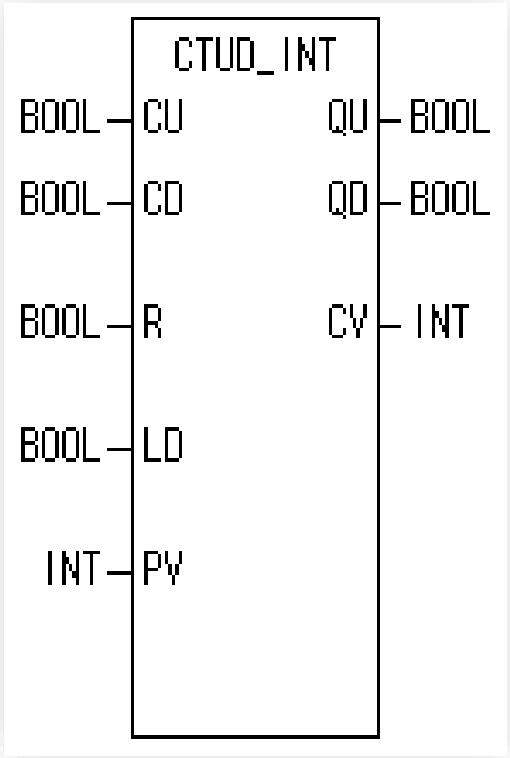


- **입력**
  - CD: 카운터 기동 조건
  - LD: CV에 PV값 로드
  - PV : 시작 숫자
- **출력**
  - Q : CV가 0이하일 때 1
  - CV: 카운트 횟수

감산 카운터.
입력값이 1일 때, CV가 0이라면
카운트 되면 출력으로 1전달

- CD 0>>1이 되는 횟수에 따라서 CV 1 감소
- CV가 0이하일 때, Q로 1값 전달
- LD : 1일 때 CV값에 PV값을 넣는다.(CV=PV)

CTUD_*** (count up/down)

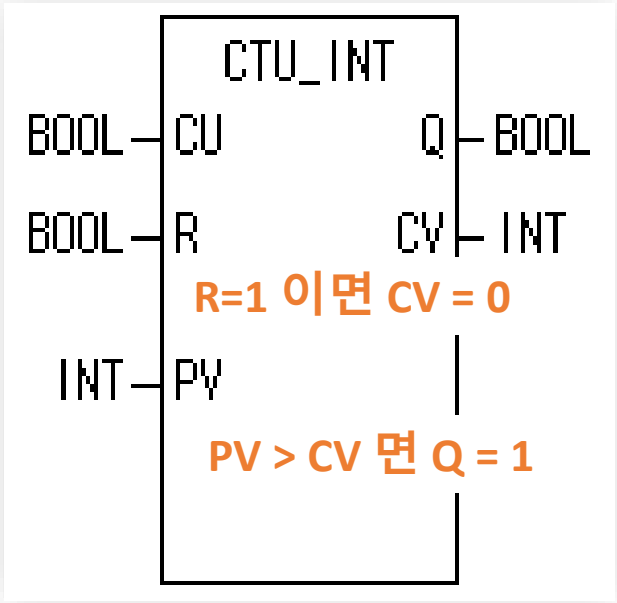


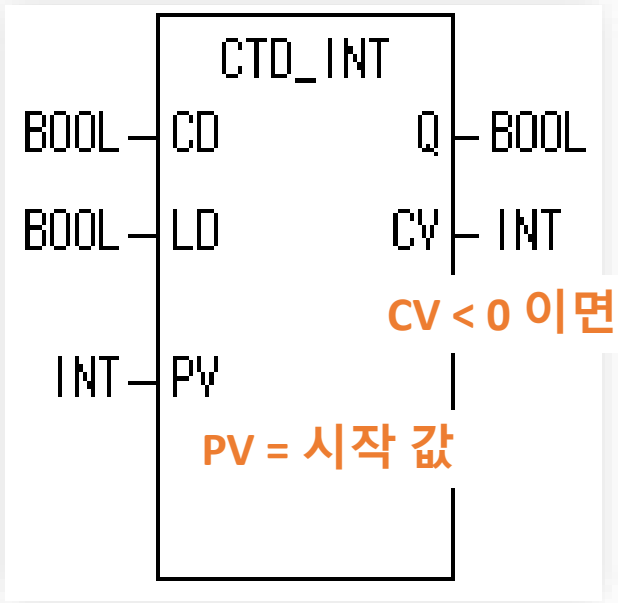
- 입력
  - CU: UP 카운터 기동 조건
  - CD: DOWN 카운터 기동 조건
  - R : up 리셋
  - LD : PV = CV 로
  - PV :기준 숫자
- 출력
  - QU: up카운트 에 대해서 출력
  - QD: down카운트 조건에 대한 출력
  - CV: up/down 되는 숫자 출력

CTU + CTD

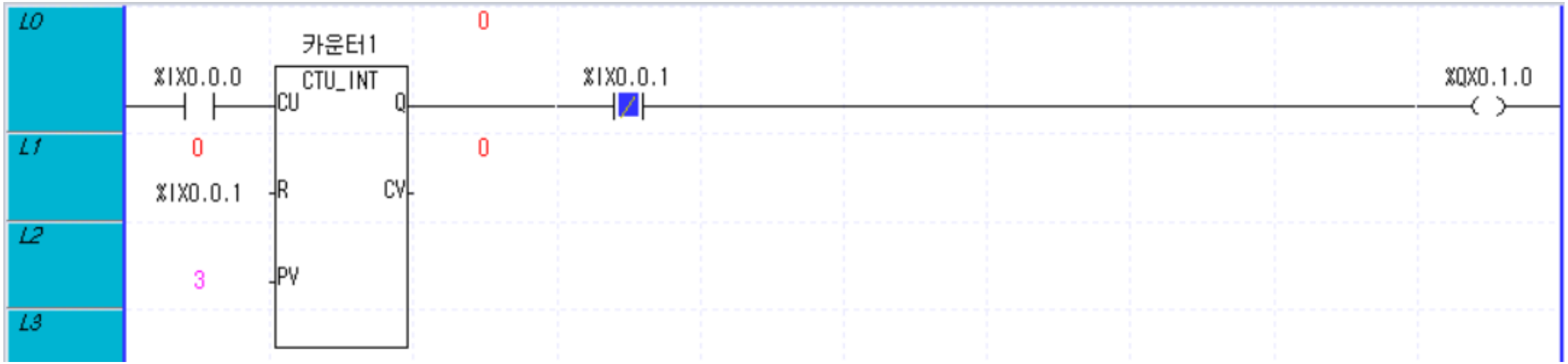
펑션 블록 - CTU, CTD

- CTU: 입력값이 0에서 1로 변할 경우 CV값 1 증가
- CTD: 입력값이 0에서 1로 변할 경우 CV값 1 감소





CTU 기본 구현


L0
L1
L2
L3
0
카운터1
%IX0.0.0
CTU_INT
CU
Q
%IX0.0.1
%QX0.1.0
0
0
%IX0.0.1
R
CV
3
PV

실습4. 플리커와 업카운터

codingOn

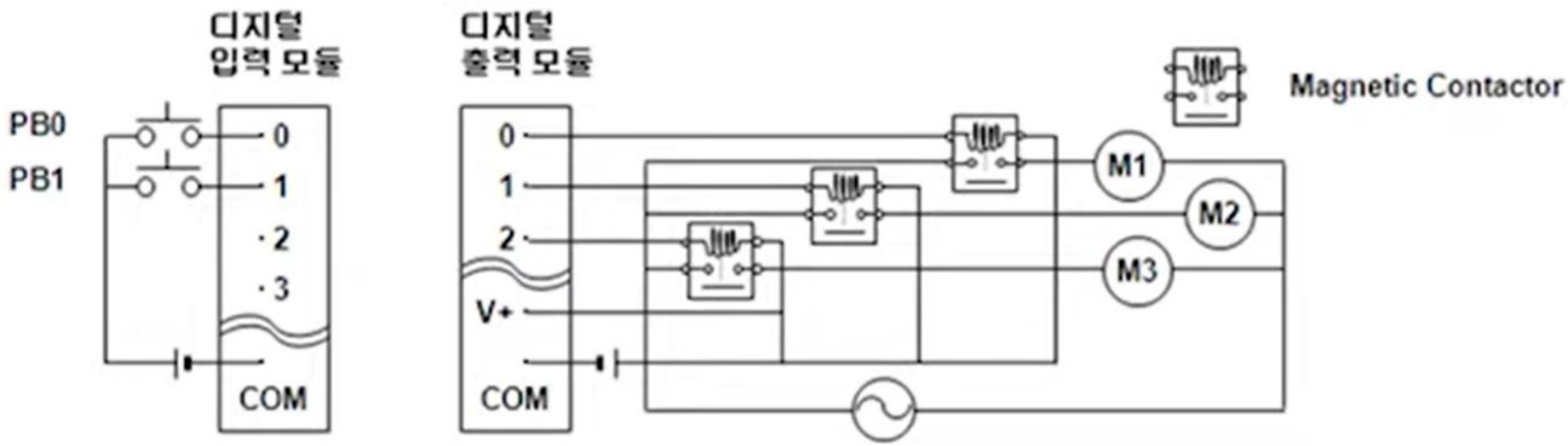
- 앞에서 배운 플리커와 함께 업카운터를 사용하여
- 램프가 3번 깜빡이면 회로가 종료되도록 구현해보기
- 램프가 꺼지는 순간 카운트를 1씩 올림
- 카운트가 3이 될 때 회로가 종료됨

실습5. 자동 물내림 심화

- TON과 TP와 CTU를 활용하여
- 사용자가 변기에 앉으면 1초 후, 2초간 물이 나오고
- 이탈 후 즉시 3초간 물이 공급
- 입력 : 사용자를 감지하는 센서
- 출력 : 세척 밸브
- 사용자가 3번 이용 할 때마다 5초간 물 공급

실습6. 모터 작동(2)

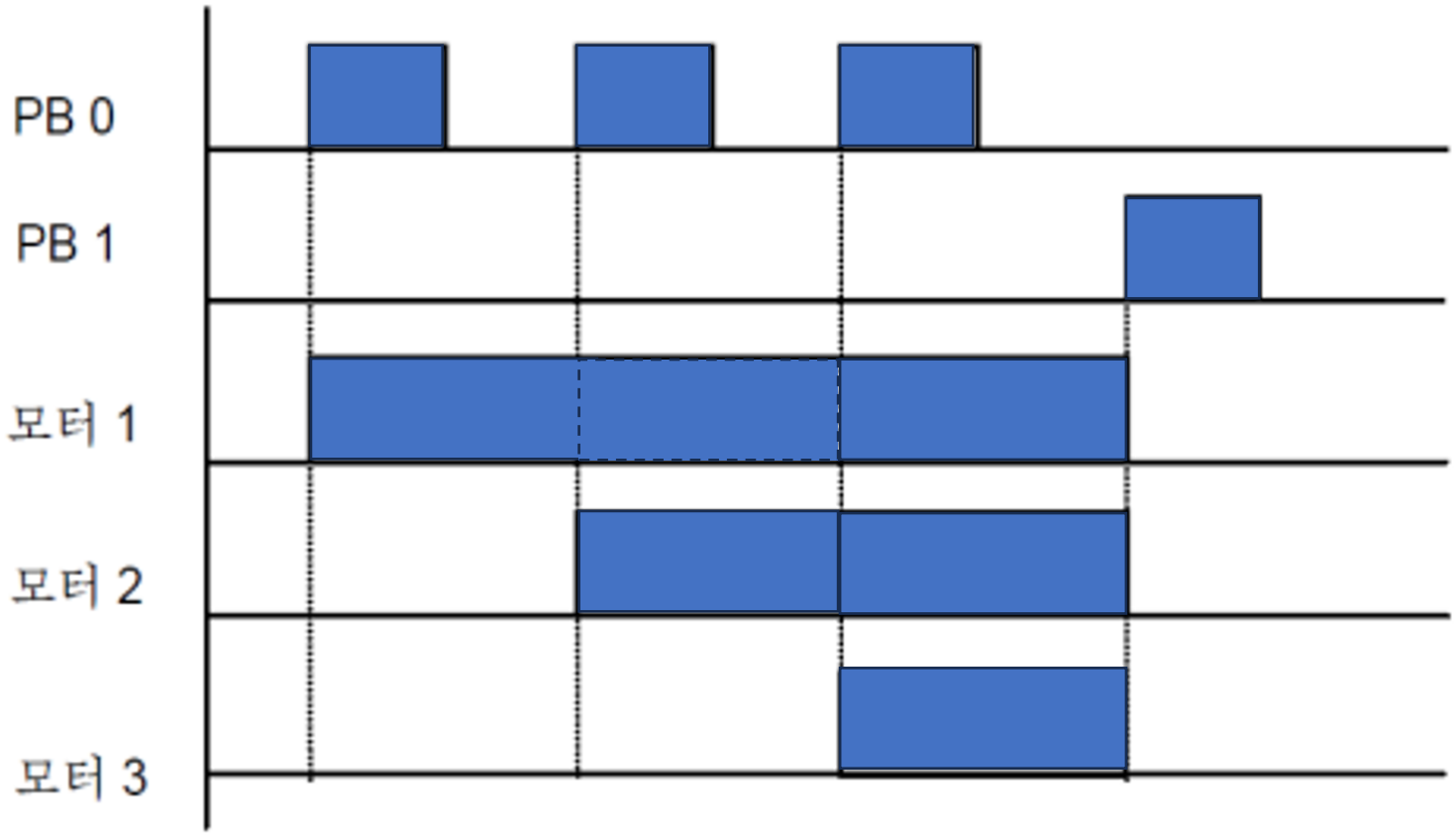
- 업카운터를 활용하여,
- PB0를 한 번 누르면 M1이 ON
- PB0를 두 번 누르면 M1, M2가 ON
- PB0를 세 번 누르면 M1, M2, M3가 ON
- PB1을 누르면 ON되어 있는 모든 모터가 정지



실습6. 모터 작동(2)

codingOn

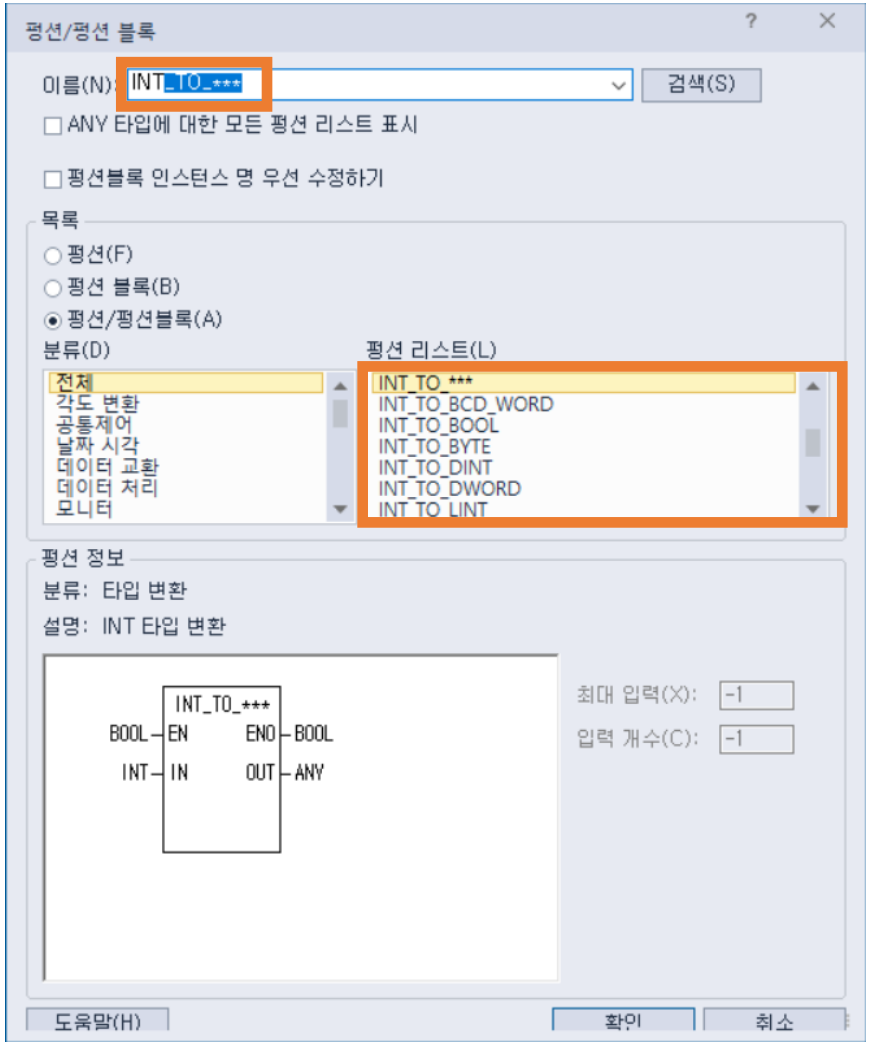
타임차트



codingOn

형 변환

형변환



- 변환전_TO_변환후로 함수 검색하여 사용
- IN에 변환전 값을 입력하면
- OUT으로 형변환된 데이터가 출력

실습7. 숫자 변환

1. 정수형으로 입력된 '숫자1'을 '숫자1_R'이라는 실수형 변수로 변환한다.
2. '숫자1_R'을 3으로 나눈 값을 '결과1_R'이라는 실수형 변수로 받는다.
3. '결과1_R'을 정수형의 '결과1'이라는 변수로 변환한다.

codingOn

감사합니다